

Session III

This week we will be working with Queries and the Structured Query Language (SQL).

Working with Functions:

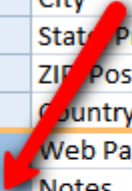
Let's begin with some simple modifications of the select query. Start Access 2016. Your last few works should be visible go to the File tab in the upper left corner and click on Open and the **Faculty database** under the CIS 28A folder should appear. Please select it. **This database was developed in Session I** and should be under your folder for the class. When you open the database **Make certain you select the Enable this Content in the Microsoft Office Security Options Dialog Box.**

Step1

In the Navigation Pane select the table **Faculty**. You can also right-click on the table in the Navigation Pane and save a step.

Deleting unnecessary fields from table Faculty:

Because we are working with a template there are some fields we wish to delete. In the Navigation Pane right click on the table Faculty and select Design View. Select the fields as shown below and press the delete key. You can use the Shift key to select multiple rows. A dialog box will pop up asking if you are permanently deleting these records select **Yes**.



Address	Text
City	Text
State/Province	Text
ZIP/Postal Code	Text
Country/Region	Text
Web Page	Hyperlink
Notes	Memo
Attachments	Attachment
Physician Name	Text
Physician Phone Number	Text
Allergies	Text
Medications	Text
Emergency Contact Name	Text
Emergency Contact Phone 1	Text
Emergency Contact Phone 2	Text

Close out table Faculty and save the changes you have made.

Make Table Query:

The Make Table Query is often used when you want to archive data. Suppose you want to archive employees who are no longer working for the company or archive data that are more than several years old.

To demonstrate we will back up the “Faculty database”.

Step 1

In the Ribbon click on the **Create** tab, click on the **Query Design** in the **Query** group. The Show Table should appear with “Faculty” as an option to select.

Step 2

Click on **Add** and then Close out the box. Double-click (or pull down to design grid). Select all fields. Fields should appear in the Design Grid.

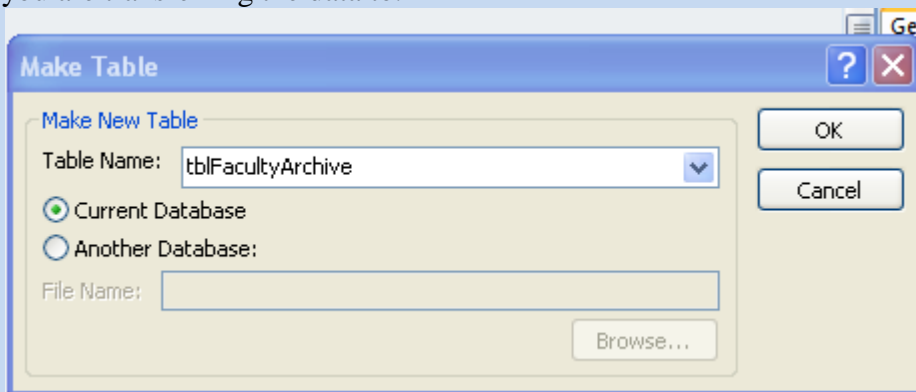
Step 3

In the Ribbon above select the **Make Table** option:



Step 4

The Make Table Dialog Box appears. Enter the name “tblFacultyArchive” for the table you are transferring the data to.



Step 5

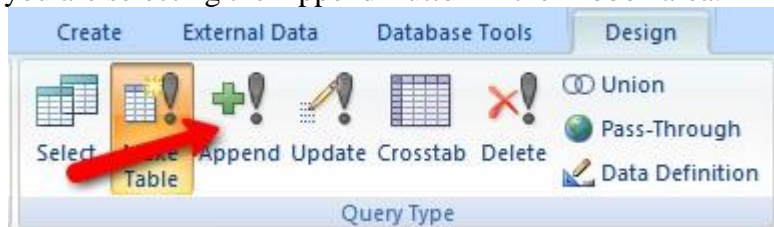
Click the **OK** button.

Step 6

Click the **Run** Button on the Ribbon and then close out the query without saving it. Remember that Run is under the Design tab on the upper left side. Look in the Unassigned Objects in the Navigation Pane for the **tblFacultyArchive** table. **Open** the table to verify that the data has been copied into the table. **Close** the table.

Append Queries:

If you already have an archive table and you want to add additional back up records you can use the Append Query command. A window pops up allow you to enter the **destination table** name to transfer the data. It is the same process as Make Table except you are selecting the Append Button in the Ribbon area.



Update Queries:

The Update Query is a bit more complicated. The Update Query is used if you want to change the data for one or more records. Let's suppose that employees received a 5% increase in pay.

Update Queries:

Step 1

Go through the steps as above when you selected the Query Design button. Again select the "Faculty" table but in this case only select the Salary Field to place in the grid below. Since this increase will apply to all records we don't have to identify which of the employees to change.

Step2

In the Ribbon click on the Design Tab then click on the button for Update Query.



Step 3

Now we need to enter the code to create the updates. In the row at the bottom of the screen Update to: **[Salary] * 1.05**. Notice the field salary is in brackets. The asterisk (*) in this case stands for multiply the salary times 1.05.

Field:	Salary
Table:	Faculty
Update To:	[Salary]*1.05
Criteria:	
or:	

Step 4

Click the **Run** Button.

Step 5

Close out the query and save the query as “**qryUpdateFacultySalary**”. By looking at the archive table and the “Faculty” table you can compare the salaries. **Close** both tables

Step 6

Let's do one more update but in this case for only one employee. **Open** in **design view** the “**qryUpdateFacultySalary**” and add the field **ID**. In the criteria for this field add the number **1**.

Field:	Salary	ID
Table:	Faculty	Faculty
Update To:	[Salary]*1.05	
Criteria:		1
or:		

Step 7

Run the query and check out the results in the **Faculty** table note that the employee with an ID of 1 is the only employee receiving an additional increase in pay.

Save and **Close** your query.

We are now going to empty out the data in the **Faculty** database and reload it with our original data from the Archive database.

Delete Queries:

Step 1

Click on the **Create** tab and select the **Query Design** option. Again, select the “Faculty” table.

Step 2

Double-click on the **asterisk** to place it in the Field cell and then select the **Delete** Button in the Ribbon.



Step 3

Run the query it should ask if you want to delete the rows select **yes** and close it with the name “**qryDeleteFaculty**”. Go to the “Faculty” table and verify that the records are no longer there.

Step 4

Since we may be using this table again go ahead and use the **append query** and copy the data from the “tblFacultyArchive” database. Save the query as “**qryAppendFromtblFaculty Archive**”. Verify that the data has been copied.

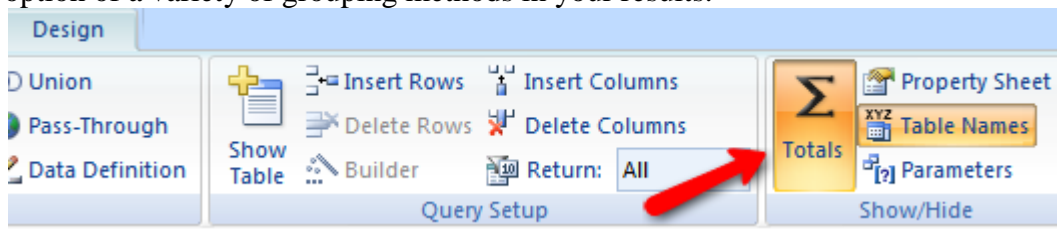
In the next two queries we won’t go through a process of actually developing the queries. But it is important that you know that the queries exist.

Union Queries:

Union Queries take two or more tables or queries of a similar structure and then adds the records from one table to the other table. If you had an archive database of all employees who have left the company and you wanted to add these records to all the active employees, you would use a union query to create the results.

Totals:

If you are developing a query and you select Totals from the Ribbon you will have the option of a variety of grouping methods in your results.



Some of the options available are: Group By, Sum, Avg, Min, Max, Count, St. Dev., Var, First, Last, Expression, and Where. For example, if you wanted to count the number of employees who have left your company you would use the **Count Aggregate function**.

Some of these functions will work with numbers such as Avg, Min, Max, and Sum. Others can be used with text such as Count, First, and Last.

Field:	ID
Table:	tblFacultyArchive
Total:	Count <input type="button" value="v"/>
Sort:	
Show:	<input checked="" type="checkbox"/>
Criteria:	
or:	

QUERY PROPERTIES:

Some of the query properties that are helpful to understand when you write action queries are:

Output All Fields: use this property to show all the fields in the queries underlying data source. When this property is set to yes, the only field you need to include in the design grid are the fields on which you want to sort by or specify criteria.

Unique Values: use this property to include only unique results when a query is run. The query will display only the records that are unique given the combination of all the included, or specifically selected, fields.

Unique Records: use this property to return from the underlying data source only unique records that are based on all the fields in the data source, not just the ones displayed in the query results. This property is ignored when the query uses only one table.

Source Database and Source Connect Str: use these properties to access external data that either is not or cannot be directly linked to your database. The Source Database Properties specifies the external source database where the data resides. The Source Connect Str property specifies the name of the program used to create the external file. For an Access database, the Source Database property is the path and database name. The Source Connect Str property is not necessary for an Access database.

Destination Table, Destination DB, Dest Connect Str: use these properties for append and make -- table queries. The destination table property represents the name of the table to which you are appending. The destination DB property is the name and path of the database to which you are pending, if it is outside the current database. The Dest connect Str property specifies the destination database. If you're appending to the current database, destination DB is the current database, and Dest Connect Str is not used.

Record locks: use this property to specify whether the records in a multiuser environment are locked so that other users cannot access them while the queries are being run. This property is typically set for action queries, so it is of particular importance when you create queries to import and archive records.

Your three choices for locking records are: **No Locks** option, the default for select queries, means that the records are **not** blocked while the query is being run. The **No Locks** option allows another user to alter the data while a report is being run. The **All Records** option allows users to read records but prevents editing, adding to, or deleting them from the underlying record source until the query has finished running. The **Edited Record** option, the default for action queries, locks the pager records as soon as the query is run. The page is a portion of the database file on which record data is stored. A page is 4KB in size and in Access 2016, depending on the size of the record, page may contain more than one record, or just a portion of a record. You can change this property, but be aware is that it affects record locking in forms and reports as well.

STRUCTURED QUERY LANGUAGE (SQL):

Because there are many features in SQL we will deal mainly with the fundamental concepts. For more information on SQL there is CIS 63 which will be taught next Spring (Fortunately there is a text book for this class!). SQL is a language that underlies Access queries. It is the most common database query language used today. The versions of SQL provided with Access 2016 have two different modes. One supports Access SQL and the other supports SQL-92 extensions. The SQL-92 extensions are only used with ActiveX Data Objects (ADO). We will be learning about ADO later in the class.

Data - Definition Queries:

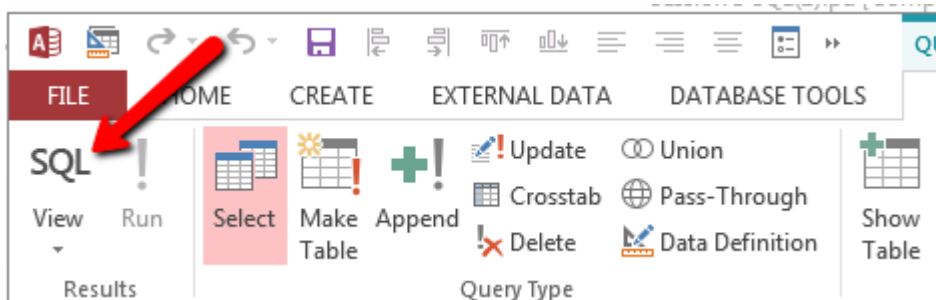
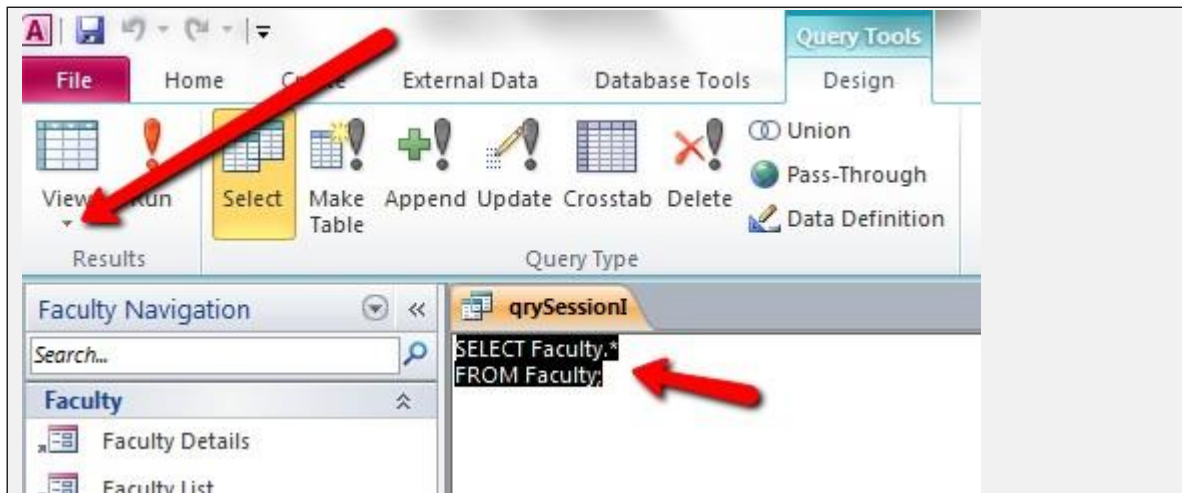
Data - definition queries use the **data - definition language** component of SQL to create objects such as tables and indexes. The data - definition commands in access SQL include create table, alter table, drop table, and create index.

Pass - through queries:

Pass - through queries pass a SQL statement through an external database server without trying to interpret the statement. This query is intended for the target database (such as SQL Server) to interpret and then to return only results. In other words, Access sends a pass through query directly to the back-end database server without understanding it. Access acts as a front-end in this case, yet the back-end database server may use a completely different dialect of SQL than Access. This type of query is also useful if the action you want to take is supported by your back-end database server, but not supported by Access SQL.

Another significant benefit of past-through queries is that you can retrieve only the fields and records you want, rather than having to import an entire table from an external database server, which could be a great deal larger than the 2 GB limit imposed by access on the entire database size. Keep in mind that query record sets are limited in size to 1 GB.

Any time you create a query that query is also created using SQL. In the “Faculty” database rt-click on the query “**qrySessionI**”. Click on the **Design** tab and then click on the arrow below View. Select the option **SQL** and you will see the following code:



Just as you used the Run Button to run queries in the Access Query by Example (QBE) Grid you can also use the Run Button to run queries that are written in SQL. Go ahead and select the **Run** Button.

SQL in Access precedes all field names with the name of the table. You can erase the name of the table before the field names and the procedure will still function.

It is important to know SQL because you can construct queries without using the (QBE) Grid, some applications aren't support by the QBE grid, if you want to build the record source for a form or a report on the fly. You will also need to know SQL for use in Data Access Objects (DAO) and ADO code.

In this course we can provide the means so that you can create basic SQL queries such as the SELECT, UPDATE, and DELETE commands.

The SELECT Statement:

The structure of the SELECT statement is:

```
SELECT * FROM Faculty;
```


Simply, you are saying give me all (*) the records from the table Faculty.

Not only can you select records from **a table** you can also select from **another query**.

The keywords in this clause are SELECT and FROM.

At the end of the SELECT clause be sure to place a (;) in some databases this is not necessary but is considered good coding and is **necessary** in this class

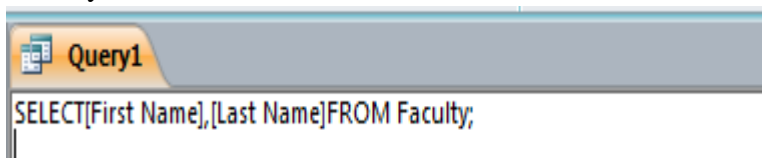
Step 1

Go to the Ribbon and select a new Create Query Design. Don't select a table instead close the show table. Select the SQL button and write the following:

Step 2

SELECT [First Name], [Last Name] FROM Faculty;

In the above clause you are asking for a record set of all First and Last Names in the Faculty table.



Step 3

Run your statement.

Distinct Row and Distinct Keywords

When you use the Distinct keyword, Access eliminates duplicate rows, based on the fields that you include in the query result. This also happens when you set the Unique Values property to yes in the query properties dialog box.

When you use the Distinct Row keyword, Access eliminates any duplicate rows based on all fields of all tables you include in the query (whether they appear in the query results or not). This also happens when you set the Unique Records property to yes in the properties dialog box.

WHERE

You can also restrict which records you wish to view by including the WHERE command.

Step1

In SQL view, change the statement to the following:

```
SELECT [First Name], [Last Name] FROM Faculty WHERE [Last Name] = 'Brown';
```

In this case only the record for the employee with the last name of Brown should appear in the record set.

Step 2

Run your statement.

ORDER BY

You also have the ability to order the set of records using the ORDER BY clause.

```
SELECT [First Name], [Last Name] FROM Faculty ORDER BY [Last Name].
```

By default, all the last names are ordered in ascending order. If you want to order in descending order you add the optional keyword **DESC**.

Step 1

In SQL view change the statement to the following:

```
SELECT [First Name], [Last Name] FROM Faculty ORDER BY [Last Name] DESC;
```

Step 2





Run your statement.

Step 3

Close without saving the query

So far we have been able to use the tables and queries in the Faculty DB but now we need to add some additional information. As we did in the previous sessions **you need to load a spreadsheet called “Departments” from your CIS28 folder**. You will be loading the data into a new table called **Departments**

I have provided some images to show you the steps.

 Classes	7/24/2015 5:12 PM	Microsoft Excel W...
 Departments	7/24/2015 5:12 PM	Microsoft Excel W...
 Faculty	7/24/2015 5:12 PM	Microsoft Excel W...
 FacultyClass	7/24/2015 5:13 PM	Microsoft Excel W...

Import Spreadsheet Wizard

Microsoft Access can use your column headings as field names for your table. Does the first row specified contain column headings?

☒ First Row Contains Column Headings

	Department	Chair	Bldg
1	World Languages	MacDougal	Quad
2	Computer Information Systems	Herzig	Bus Ed
3	Behavioral Sciences	Stearns	LoveKin Complex
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			
17			
18			
19			
20			

Cancel < Back Next > Finish

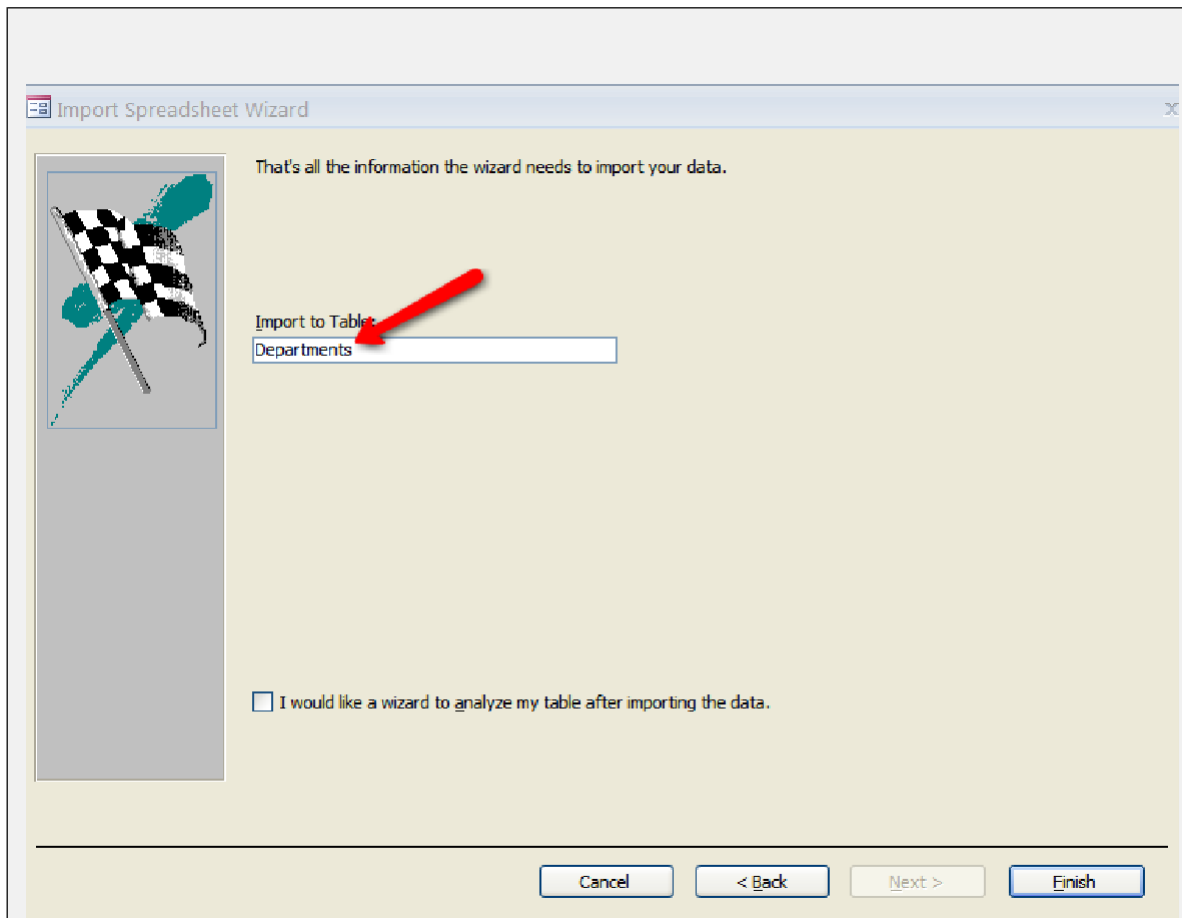
Import Spreadsheet Wizard

Microsoft Access recommends that you define a primary key for your new table. A primary key is used to uniquely identify each record in your table. It allows you to retrieve data more quickly.

☐ Let Access add primary key.
 ☒ Choose my own primary key. Department
 ☐ No primary key.

	Department	Chair	Bldg
1	World Languages	MacDougal	Quad
2	Computer Information Systems	Herzig	Bus Ed
3	Behavioral Sciences	Stearns	LoveKin Complex
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			
17			
18			
19			
20			

Cancel < Back Next > Finish

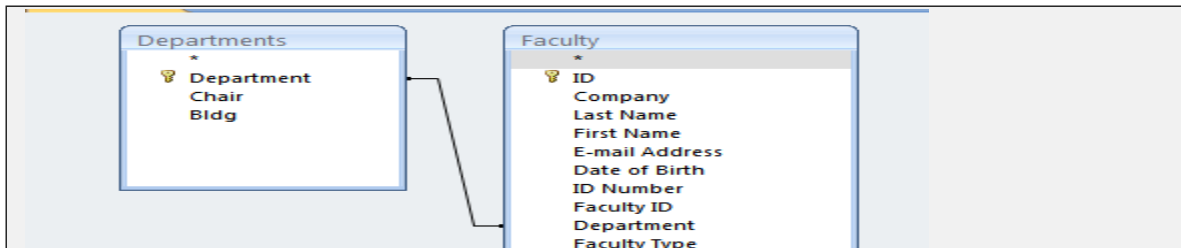


JOIN:

Now that you have added the additional table we can explore the Join clause using SQL. If you are joining tables, you will use the JOIN clause.

Step 1

Open the table Departments and verify the field Department is the primary key for the table.



Step 2

Open up a new query utilizing the Show Table window select the tables “**Faculty**” and “**Departments**”.

Go to the Ribbon and select the SQL view.

Step 3

Type in the following code:

```
SELECT Faculty.[First Name], Faculty.[Last Name], Departments.Chair
FROM Departments INNER JOIN Faculty ON Departments.Department =
Faculty.Department;
```

The first **SELECT** clause is selecting the fields you want displayed.

The next clause **FROM** is selecting tables however, it uses an **INNER JOIN** to connect the table “Departments” with the table “Faculty”.

The last clause **INNER JOIN** is using the field “Department” as the means to connect the two tables. This is referred to as a one-to-many relationship. There are (many)

Step 4

Click on the Run Button

Inner Join:

An inner join tells a query that rows from one of the joined tables corresponds to rows in the other table on the basis of the data in the join fields. Checks for matching values in the join fields and when it finds matches, combines the records and displays them as one record in the query results.

An inner join is also referred to as a **Natural Join**.

Outer Join:

An outer join tells a query that although some of the rows on both sides of the join correspond exactly, the query should include all rows from one table even if there is no match in the other table. Each matching record from two tables is combined into one

record query results. One table contributes all of its records even if the values in its join field do not match the field values in the other table. Outer joins can be left outer joins or right outer joins. In a query with a left outer join, all rows in the left table are included in the results and only rows from the other table where the joining field contains values common to both tables are included. The reverse is true with a right outer join.

Cartesian Join

When more than one table is added to the Query Design view, but without any join lines established, the result set is known as a Cartesian join. This is the case where all possible combinations of rows for the tables are represented. If you have a large number of records in the tables it results in a massive listing.

Junction Table or Bridge Table

A Junction Table is a table that is used to connect two tables in a many: to: many relationship. In this case one of the tables is “Faculty” there are many faculty who will teach many classes. So the second table is “Classes”. By using the Junction table, we connect each faculty’s ID with the Section of the class they will be teaching. Again, as we did previously you need to load in a spreadsheet call “**Classes**”. Bring in this spreadsheet into a table which you will label “**Classes**”.

See the images below if you have questions about the process.

Your spreadsheet file contains more than one worksheet or range. Which worksheet or range would you like?

☒ Show Worksheets
☐ Show Named Ranges

Sheet1
Sheet2
Sheet3

	Class	ClassDesc	Units	Prerequisite	Prerequisite Req
1	CIS-1A	Intro to Comp Info Sys	3	No	
2	CIS-2	Fund. Of Systems Analysis	3	No	
3	CIS-1B	Adv Concepts of CIS	3	Yes	CIS-1A
4	CIS-5	Program Logic Using C++	3	No	
5	CIS-62	MS Access DBMS:Comprehensive	3	no	
6	CIS-63	SQL	3	No	
7	CIS-28A	Programming in Access	3	No	
8	CIS-80	Word for Windows	3	No	
9	CIS-34A	Intro MS Word for Windows	1.5	No	
10	CIS-18A	Java Programming: Objects	3	No	
11	SPA-1	Spanish 1	5	No	
12	SPA-2	Spanish 2	5	Yes	SPA-1
13	SPA-3	Spanish 3	5	Yes	SPA-2

Cancel < Back Next > Finish

Import Spreadsheet Wizard

Microsoft Access can use your column headings as field names for your table. Does the first row specified contain column headings?


☒ First Row Contains Column Headings

	Class	ClassDesc	Units	Prerequisite	Prerequisite Req
1	CIS-1A	Intro to Comp Info Sys	3	No	
2	CIS-2	Fund. Of Systems Analysis	3	No	
3	CIS-1B	Adv Concepts of CIS	3	Yes	CIS-1A
4	CIS-5	Program Logic Using C++	3	No	
5	CIS-62	MS Access DBMS:Comprehensive	3	no	
6	CIS-63	SQL	3	No	
7	CIS-28A	Programming in Access	3	No	
8	CIS-80	Word for Windows	3	No	
9	CIS-34A	Intro MS Word for Windows	1.5	No	
10	CIS-18A	Java Programming: Objects	3	No	
11	SPA-1	Spanish 1	5	No	
12	SPA-2	Spanish 2	5	Yes	SPA-1
13	SPA-3	Spanish 3	5	Yes	SPA-2
14	SPA-4	Spanish 4	5	Yes	SPA-3

Cancel < Back Next > Finish

Import Spreadsheet Wizard

Microsoft Access recommends that you define a primary key for your new table. A primary key is used to uniquely identify each record in your table. It allows you to retrieve data more quickly.



☐ Let Access add primary key.

☒ Choose my own primary key.

☐ No primary key.

	Class	ClassDesc	Units	Prerequisite	Prerequisite Req
1	CIS-1A	Intro to Comp Info Sys	3	No	
2	CIS-2	Fund. Of Systems Analysis	3	No	
3	CIS-1B	Adv Concepts of CIS	3	Yes	CIS-1A
4	CIS-5	Program Logic Using C++	3	No	
5	CIS-62	MS Access DBMS:Comprehensive	3	no	
6	CIS-63	SQL	3	No	
7	CIS-28A	Programming in Access	3	No	
8	CIS-80	Word for Windows	3	No	
9	CIS-34A	Intro MS Word for Windows	1.5	No	
10	CIS-18A	Java Programming: Objects	3	No	
11	SPA-1	Spanish 1	5	No	
12	SPA-2	Spanish 2	5	Yes	SPA-1
13	SPA-3	Spanish 3	5	Yes	SPA-2
14	SPA-4	Spanish 4	5	Yes	SPA-3

Cancel < Back Next > Finish

Import Spreadsheet Wizard

That's all the information the wizard needs to import your data.

Import to Table:
Classes

☐ I would like a wizard to analyze my table after importing the data.

Cancel < Back Next > Finish

We are going to do the same thing with another file called “**FacultyClass**”. Please load this Excel File. **Please note that in this table we don’t have a primary key.**

Import Spreadsheet Wizard

Microsoft Access recommends that you define a primary key for your new table. A primary key is used to uniquely identify each record in your table. It allows you to retrieve data more quickly.

☐ Let Access add primary key.
☐ Choose my own primary key.
☒ No primary key.

	FacultyID	Class
1	1	CIS-1A
2	1	CIS-2
3	1	CIS-1B
4	4	CIS-1A
5	6	CIS-5
6	4	CIS-62
7	4	CIS-63
8	4	CIS-28A
9	6	CIS-80
10	6	CIS-34A
11	6	CIS-18A
12	3	SPA-1
13	3	SPA-2
14	3	SPA-3

Cancel < Back Next > Finish

Import Spreadsheet Wizard

That's all the information the wizard needs to import your data.

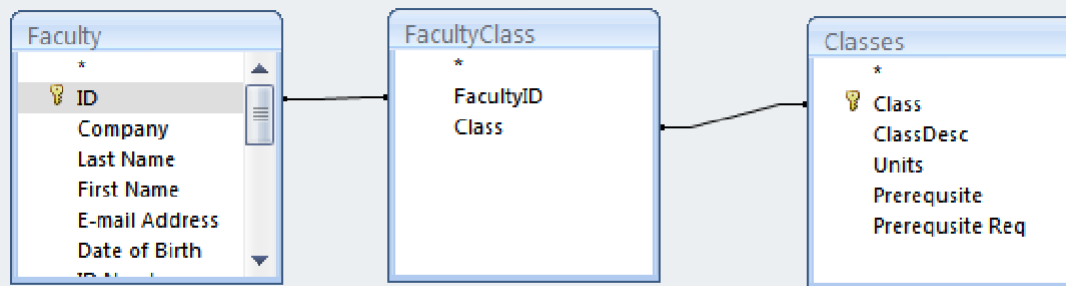
Import to Table:
FacultyClass

☐ I would like a wizard to analyze my table after importing the data.

Cancel < Back Next > Finish

Open a new query utilizing the Query Design Grid. Select the tables “**Faculty**”, “**FacultyClass**” and “**Classes**”.

You may need to create a link between Faculty.ID and FacultyClass.FacultyID.



Suppose I want to look at all my faculty and what classes they are teaching this Summer.

Step 1

Select the **Last and First Name** from the **Faculty** table and the **Class** and **ClassDesc** from the **Classes** table.

Step 2

Run the query.

Last Name	First Name	Class	ClassDesc
Brown	Jane	CIS-1A	Intro to Comp Info Sys
Brown	Jane	CIS-2	Fund. Of Systems Analysis
Brown	Jane	CIS-1B	Adv Concepts of CIS
Schwartz	Bill	CIS-1A	Intro to Comp Info Sys
Holding	Jim	CIS-5	Program Logic Using C++
Schwartz	Bill	CIS-62	MS Access DBMS:Comprehensive
Schwartz	Bill	CIS-63	SQL
Schwartz	Bill	CIS-28A	Programming in Access
Holding	Jim	CIS-80	Word for Windows
Holding	Jim	CIS-34A	Intro MS Word for Windows
Holding	Jim	CIS-18A	Java Programming: Objects
Hernandex	Fran	SPA-1	Spanish 1
Hernandex	Fran	SPA-2	Spanish 2
Hernandex	Fran	SPA-3	Spanish 3
Smith	Gail	ANT-1	Physical Anthropology
Smith	Gail	ANT-2	Cultural Anthropology
Green	Bob	BUS-10	Intro to Business
Green	Bob	BUS-18A	Business Law 1
Green	Bob	BUS-18B	Business Law 2

In this type of design, you have minimum duplication. The only duplication being in the junction tables and in that table the duplications are only the primary keys of the connecting tables.

We have seen how to develop the query in design mode go ahead and look at the query in SQL mode.

```
SELECT Faculty.[Last Name], Faculty.[First Name], Classes.Class, Classes.ClassDesc
FROM Faculty INNER JOIN (Classes INNER JOIN FacultyClass ON Classes.Class = FacultyClass.Class) ON Faculty.ID = FacultyClass.FacultyID;
```

Notice the keywords SELECT and FROM. You already have had experience with these keywords.

The keywords INNER JOIN and ON allow you to select the fields of the two tables you are joining. In this case we are telling Access to join the “Class” table utilizing the Class field to the “FacultyClass” table utilizing the same field.

For example: Classes INNER JOIN FacultyClass ON Classes.Class = FacultyClass.Class)

The two tables Classes and FacultyClass are joined INNER JOIN by the fields Classes.Class and FacultyClass.class

We also join Faculty and Faculty Class with the field FacultyID

ON Faculty.ID = FacultyClass.ID

Close the query and do not save.

Close the Faculty Database.

Project

MovieCam Technologies has accumulated a significant amount of employee data, production data, and other historical data that needs to be incorporated in its new database. Amanda asks you to determine the best method to import this existing data into the MovieCam database. As you have learned, the managers maintain much of the data, including time cards and employee information, in spreadsheets stored in Excel workbooks. Other data, such as personnel and payroll information, has been entered in MoveCam's accounting software program, ACC90

The company's accounting software is an older version of ACC90 and does not export to a format recognized by Access. Randy Harper, an ACC90 consultant, has already exported some of the data to a tab-delimited text file. Amanda asks you to work out the importing process using both the sample accounting file Randy has provided and the sample excel data provided by the product managers.

To Open a Tab-delimited Text File:

Step 1

Open the **Session3Movie database** and do the following:

Click **External Data** in the **Ribbon**, in the **Import** group point and click on the **Text File**.









Step 2

In your CIS 28 folder find Session 3 Movie.txt and select it then click on **Open**.

Step 3

The Get External Data window opens and you need to select Import the source data into a new table in the current database.

Click OK.

Name	Date modified	Type	Size
 Classes	8/26/2009 10:53 AM	Microsoft Excel W...	9 KB
 Departments	8/26/2009 10:55 AM	Microsoft Excel W...	9 KB
 Faculty	6/24/2011 3:42 PM	Microsoft Excel W...	11 KB
 FacultyClass	8/26/2009 10:54 AM	Microsoft Excel W...	9 KB
 Logo	9/3/2009 10:22 AM	WMF File	23 KB
 Movie3	9/5/2009 7:44 PM	Text Document	3 KB

Step 4

The Import Text Wizard appears, make certain that the Delimited option is selected, click **Next**.

Import Text Wizard

Your data seems to be in a 'Delimited' format. If it isn't, choose the format that more correctly describes your data.

☒ Delimited - Characters such as comma or tab separate each field
☐ Fixed Width - Fields are aligned in columns with spaces between each field

Sample data from file: C:\RCC CLASSES\CIS28A\MOVIE2.TXT.

	Employee No	Depart No	Employee Name	Hourly Rate	Hourly/Salary	Employee Type	Terminated
1	200200	JUAN GERARDO	25.80	NO	Yes	NO	1/21/2001
2	004/30/2003	1900 Loretta	NO	Sonoma	CA	707	
3	2000500	DAN SMITH	210	HO	Yes	NO	10/1/2002
4	012/14/2004	125 Lakewood Avenue	NO	Cotati	CA		
5	2300500	SANDRA LOW	36.50	SO	NO	NO	2/22/2003
6	03/4/2005	574 Lorraine Court	NO	Santa Rosa	CA		
7	3000200	JACK HUFT	39.99	SO	NO	NO	4/11/2001
8	012/3/2003	15 Western Street	NO	Petaluma	CA		
9	3100500	CHUCK GOLDBERG	26.03	SO	NO	NO	6/30/2004
10	06/30/2005	722 Blair Place	NO	Santa Rosa	CA		
11	4000500	JUDY ELVY	12.90	HO	NO	NO	5/1/2004
12	05/15/2004	567 Montecito	NO	Rohnert Park	CA		
13	4200500	TODD COMBS	14.21	SO	NO	NO	7/12/2004
14	010/1/2005	811 Vineyard	NO	Santa Rosa	CA		
15	100200	THOMAS ARQUETTE	59.50	SO	NO	NO	10/5/2007
16	012/65	Sycamore Avenue	NO	Santa Rosa	CA		
17	1500100	CAROLYN VALDEZ	45.25	SO	NO	NO	10/5/2007
18	002/756	Heartland Way	NO	Bodega Bay	CA		
19	2100800	MARTIN WOODWARD	39.33	SO	NO	NO	10/5/2007
20	002/752	Adobe Road	NO	Petaluma	CA		
21	5000500	ALAN COOK	19.60	HO	NO	NO	10/5/2007
22	002/762	Montgomery Drive	NO	Santa Rosa	CA		
23	6000200	GLORIA CAULDWELL	39.22	HO	NO	NO	10/5/2007
24	002/22	Madrone Way	NO	Glen Ellen	CA		
25	7000500	ERNEST GOLDO	17.00	HO	NO	NO	10/5/2007
26	011/1	First Street	NO	Santa Rosa	CA		

Advanced... Cancel < Back Next > Finish

Step 5

Access has determined that the delimited is Tab, be sure to select the First Row Contains Field names, click **Next** again.

Import Text Wizard

What delimiter separates your fields? Select the appropriate delimiter and see how your text is affected in the preview below.

Choose the delimiter that separates your fields:

☒ Tab
 ☐ Semicolon
 ☐ Comma
 ☐ Space
 ☐ Other:

☒ First Row Contains Field Names
 Text Qualifier: {none}

Employee No	Depart No	Employee Name	Hourly Rate	Hourly/Salary	Employee Type	Terminate
20	20	JUAN GERARDO	25.80	S	R	Yes
200	50	DAN SMITH	21	H	R	Yes
230	50	SANDRA LOW	36.50	S	R	No
300	20	JACK HUFT	39.99	S	R	No
310	50	CHUCK GOLDBERG	26.03	S	R	Yes
400	50	JUDY ELVY	12.90	H	R	No
420	50	TODD COMBS	14.21	S	R	No
10	20	THOMAS ARQUETTE	\$59.50	S	R	No
150	10	CAROLYN VALDEZ	\$45.25	S	R	No
210	80	MARTIN WOODWARD	\$39.33	S	R	No
500	50	ALAN COOK	\$19.60	H	R	No
600	20	GLORIA CAULDWELL	\$39.22	H	R	No
700	50	ERNEST GOLD	\$17.00	H	R	No
800	10	ANN GARCIA	\$15.30	H	H	No

Step 6

In the next screen change Zip **Code** to Short Text then click **Next** on the following screen.

Step 7

In the following screen click **Choose my own primary key** and Access will have Employee No as the Primary Key.

Import Text Wizard

Microsoft Access recommends that you define a primary key for your new table. A primary key is used to uniquely identify each record in your table. It allows you to retrieve data more quickly.

☐ Let Access add primary key.
☒ Choose my own primary key. Employee No
☐ No primary key.

Employee No	Depart No	Employee Name	Hourly Rate	Hourly/Salary	Employee Type	Terminate
20	20	JUAN GERARDO	25.80	S	R	Yes
200	50	DAN SMITH	21	H	R	Yes
230	50	SANDRA LOW	36.50	S	R	No
300	20	JACK HUFT	39.99	S	R	No
310	50	CHUCK GOLDBERG	26.03	S	R	Yes
400	50	JUDY ELVY	12.90	H	R	No
420	50	TODD COMBS	14.21	S	R	No
10	20	THOMAS ARQUETTE	\$59.50	S	R	No
150	10	CAROLYN VALDEZ	\$45.25	S	R	No
210	80	MARTIN WOODWARD	\$39.33	S	R	No
500	50	ALAN COOK	\$19.60	H	R	No
600	20	GLORIA CAULDWELL	\$39.22	H	R	No
700	50	ERNEST GOLD	\$17.00	H	R	No
800	10	ANN GARCIA	\$15.30	H	H	No

Advanced... Cancel < Back Next > Finish

In this screen you must select the table to insert the data, change “Session 3 Movie” to “tblEmployeesImport”

Import Text Wizard

That's all the information the wizard needs to import your data.

Import to Table: tblEmployeesImport

☐ I would like a wizard to analyze my table after importing the data.

Advanced... Cancel < Back Next > Finish

Step 9

Click the **Finish** button and don't save the Import steps click on **Close**

Correcting the data types:

Step 1

Open the **tblEmployeesImport** table in design view.

Note if you get an error message ignore it and continue on.

Step 2

Adjust the data type in the table to those shown in the image below.

Field Name	Data Type
Employee No	Number
Depart No	Number
Employee Name	Short Text
Hourly Rate	Currency
Hourly/Salary	Short Text
Employee Type	Short Text
Terminated?	Yes/No
Exempt Code	Short Text
Allowed To	Yes/No
Date Created	Date/Time
Date Modified	Date/Time
AddressL1	Short Text
AddressL2	Short Text
City	Short Text
State	Short Text
Telephone	Short Text
Zip Code	Short Text
Birth Date	Date/Time
Hire Date	Date/Time

Delete any empty rows of data.

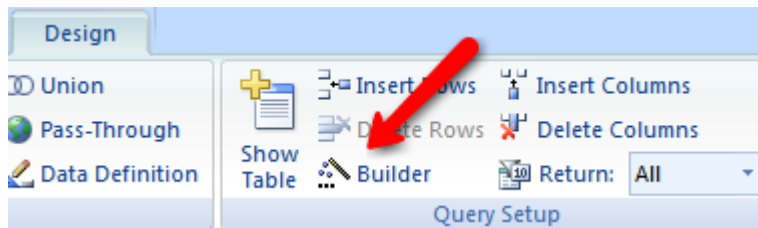
You have successfully imported data into Access and modified the data types.

However, the employees' last names are not in the format that you want. The first and last names are in one field. We want the first and last names to be in different fields.

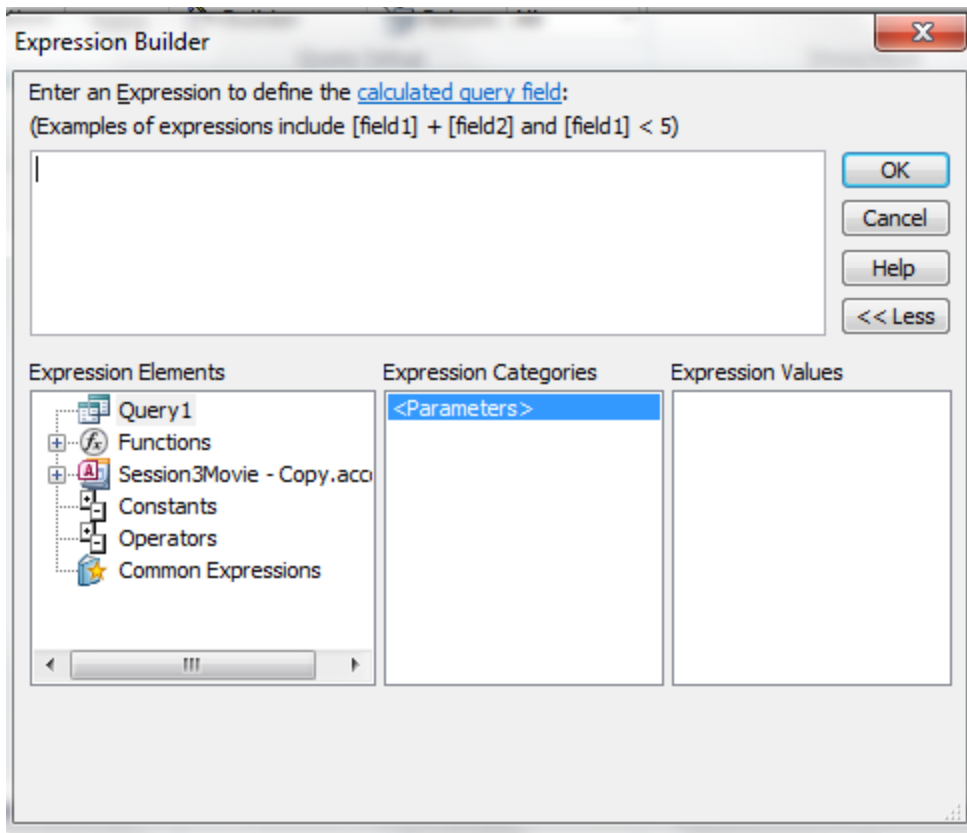
Step 1 create a new query using **tblEmployeesImport** table

Using the Expression Builder:

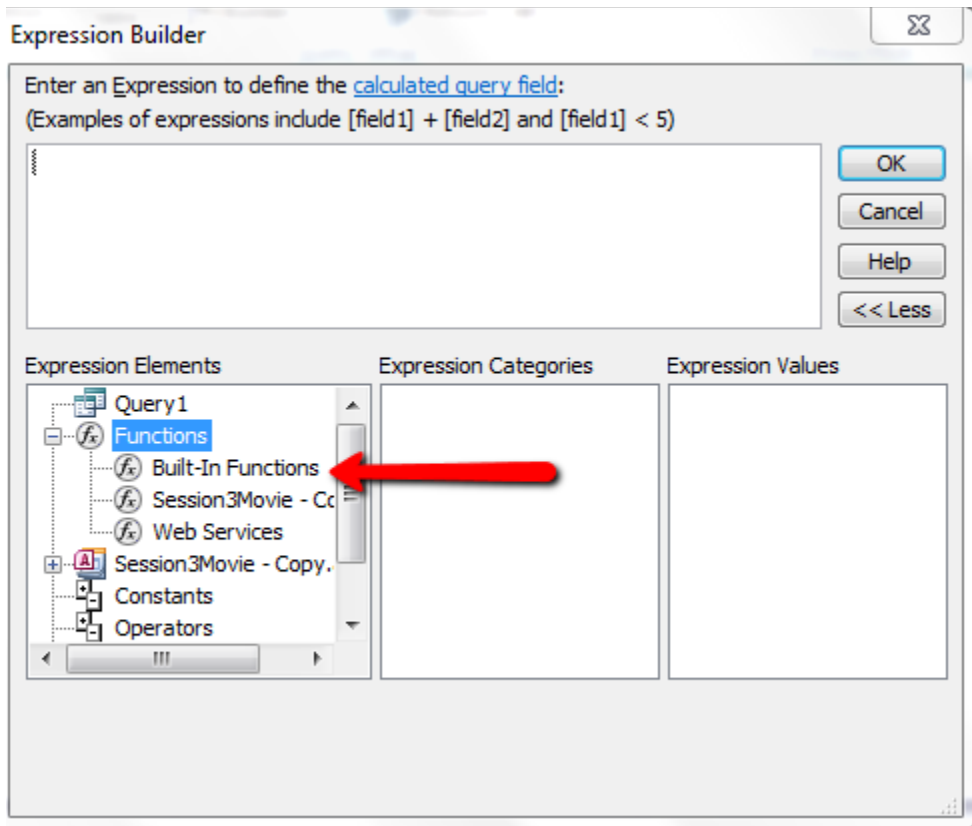
To help build expressions for your query Access has provided the Expression Builder. To actuate the expression builder, click in the field cell of your query design grid and then select the Ribbon tab Design and click Builder on the Ribbon group "Query Setup".



The following Expression Builder appears:



Note that there are three columns. The first column shows the objects in the database. After you select an object in the left column you can then select elements from that object in the second and third columns. Those items selected will appear in the window above. See if you can locate the fields and space function to create the field "Name". Go to the first column showing Objects (on the left side) and select **Build-In Functions**. This shows the categories of functions in the second column. The third column shows the specific functions available.



The Expression Builder is Made Up of Three Main Sections:

Section	Description
Expression Box	Used to build expression. To build an expression, you select elements from the other two sections and paste them in this box. You also can type parts of the expression directly in the box.
Operator Buttons	A toolbar of commonly used operators for the selected expression element. The four types of operators are mathematical, comparison, logical, and reference. Clicking one of these buttons adds it to the expression (in the expression box) at the current insertion point.
Expression Elements	This section consists of three list boxes. The left box contains folders that list the objects (tables, queries, forms, and reports) in the database, built-in and user defined functions, constants (values that do not change such as a specific number or text), operators, and common expressions. The middle box lists specific elements or element categories for the folder that is selected in the left box. The right box lists the values (if any) for the element selected in the left and middle boxes.

Here are some of the more important built- in functions used with text manipulations:

Function	Description	Example	Result
Len	Returns the number of char in a string	Len("pencil")	6
LCase	Returns the lowercase version of a string	LCase("PENCIL")	pencil

UCase	Returns the uppercase version of a string	UCase("pencil")	PENCIL
Right	Returns the far right char(s) of a string	Right("pencil",2)	il
Left	Returns the far left char(s) of a string	Left("pencil",2)	pe
Mid	Returns a portion of the string	Mid("pencil",2,3)	enc
LTrim	Removes leading spaces from a string	LTrim(" pencil")	pencil
RTrim	Removes trailing spaces from a string	RTrim("pencil ")	pencil
Trim	Removes leading and trailing spaces	Trim(" pencil ")	pencil
InStr	Returns the position of the first occurrence of one string within another.	InStr(1,"Parrot","o")	5

Selecting the First Name from tblEmployeesImport

Step 1

Select the **Create tab** in the Ribbon and click on **Query Design**.

Step 2

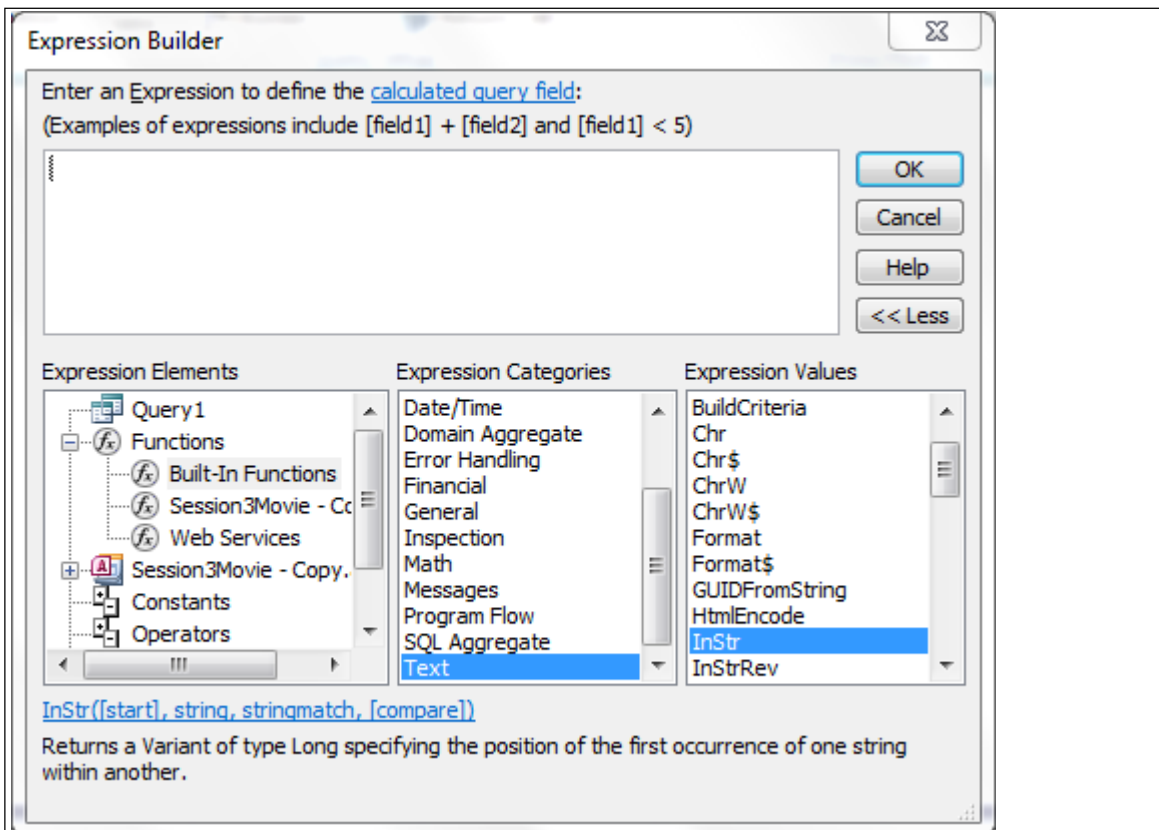
Select **tblEmployeesImport** from the Show Table window.

Step 3

Be sure that the insertion point is in the first empty field box next in the Design Grid and then **right-click** and select the **Build** option. The Expression Builder dialog box opens:

Step 4

Select **Functions** Then **Built-In Functions**, then **Text**, then double-click **InStr**.



To simplify this expression for you what the InStr built-in function is doing is coming up with the length of the employee's first name. Here is the code used to find the length:

InStr(1,[Employee Name]," ")

The <<start>> argument tells Access where you want to begin. We will begin in position **1**;

The next argument <<stringexpr1>> tells Access what field you want to search; in this case it is the **Employees name**. Since we are dealing with a field we place the field name in brackets [].

The next argument <<stringexpr2>> tells Access what character(s) you are looking for, in this case we are looking for the space at the end of the employees first name so we will enter a " " to indicate a space.

If we enter this expression in the field name Access will begin the expression with Expr1: To make Expr1 more meaningful let's change Expr1 to **NameSpace** to indicate that is where the employee's first name ends.

Step 5

Enter the following in the next field name:

NameSpace: InStr(1,[Employee Name]," ")

Note to make the expression easier to work with you can right-click on the expression and select zoom.

Step 6

Click the **show check box** and **run** the query; a list of numbers should appear showing the end of the employees' first name.

Step 7

Since we have the length of the employees last name we use another function in the Expression Builder called **Left** which will extract all the letters up to the length of the employees' first name:

FirstName: Left([Employee Name],[NameSpace]-1)

The -1 subtracts one space from the second argument because NameSpace counts the position up to and including the space, and you don't want the space included at the end of the first name.

Step 8

Place the expression **FirstName: Left([Employee Name],[NameSpace]-1)** in the next field name and select the **show check box**.

Step 9

Run the query and the employees first name should appear.

To Select the Last Name from tblEmployeesImport:

Step 1

Select the design view of the query.

Since we know where the space begins between the first and last name we can again use the expression Builder to create a function that will allow us to only view the last name of the employee.

The last name requires using two additional functions. To extract the far right characters from the Employee Name value, you need to know the number of characters in the value, starting with the space and counting to the end of the string. To determine this value, you'll calculate the total number of characters in the employee Name using the Len function, and then subtract the value of NameSpace from the total. You'll use the right function to extract the far right characters from the Employee Name field.

Step 2

In the expression Builder select Functions, Built-In Functions, Text, and Len.

Step 3

Type in the following in the next field name box: **Len ([Employee Name])**

Step 4

Replace Expr1 with NameLen: So you have the following

NameLen: Len ([Employee Name])

You now know the length of the Employees name and by subtracting Name space from NameLen you are able to extract the last name of the employee.

Step 5

Enter the following in the next field text box: **LastName: Right([Employee],[NameLen]-[NameSpace])** and select the Show check box.

Step 6

From the list of available fields in tblEmployeesImport drag down the **Employee Name** from the **tblEmployeesImport** and place it in the first field of the Design Grid. The other fields should move over one column to the right.

Step 7

Run the query and the following results should show:

Query1					
Employee Name	NameSpace	FirstName	NameLen	LastName	
THOMAS ARQUETTE	7	THOMAS	15	ARQUETTE	
CAROLYN VALDEZ	8	CAROLYN	14	VALDEZ	
JUAN GERARDO	5	JUAN	12	GERARDO	
DAN SMITH	4	DAN	9	SMITH	
MARTIN WOODWARD	7	MARTIN	15	WOODWARD	
SANDRA LOW	7	SANDRA	10	LOW	
JACK HUFT	5	JACK	9	HUFT	
CHUCK GOLDBERG	6	CHUCK	14	GOLDBERG	
JUDY ELVY	5	JUDY	9	ELVY	
TODD COMBS	5	TODD	10	COMBS	
ALAN COOK	5	ALAN	9	COOK	
GLORIA CAULDWELL	7	GLORIA	16	CAULDWELL	
ERNEST GOLD	7	ERNEST	11	GOLD	
ANN GARCIA	4	ANN	10	GARCIA	
JANICE SMITTY	7	JANICE	13	SMITTY	

You are now ready to add additional fields to the query.

Append data in tblEmployeesImport for table tblEmployees:

Step 1

Bring all of the fields from **tblEmployeesImport** into the design grid, make certain the show check boxes are selected. **Deselect** the Employee Name because you have the LastName and FirstName.

Step 2

Run the query to ensure all fields show

Step 3

Close and save query as **qryRowsToImport**

Remember that tblEmployee has a primary key EmployeeNo. The Primary Key must be unique for each record. The data from the import file has duplicates which cannot be appended because it would create duplicates in the tblEmployee and thus it wouldn't be permitted. So the first thing we must do is to get rid of the duplicates.

Getting rid of duplicates in the tblEmployeeImport

Step 1

Select Create and then Query Design from the Ribbon.

Step 2

Important to choose tblEmployeesImport and then tblEmployees from the Show table.

Step 3

Connect the EmployeeNo from tblEmployeesImport and drag over to EmployeeNo in tblEmployees.

Step 4

Drag down the all symbol * from tblEmployeesImport to the Design Grid.

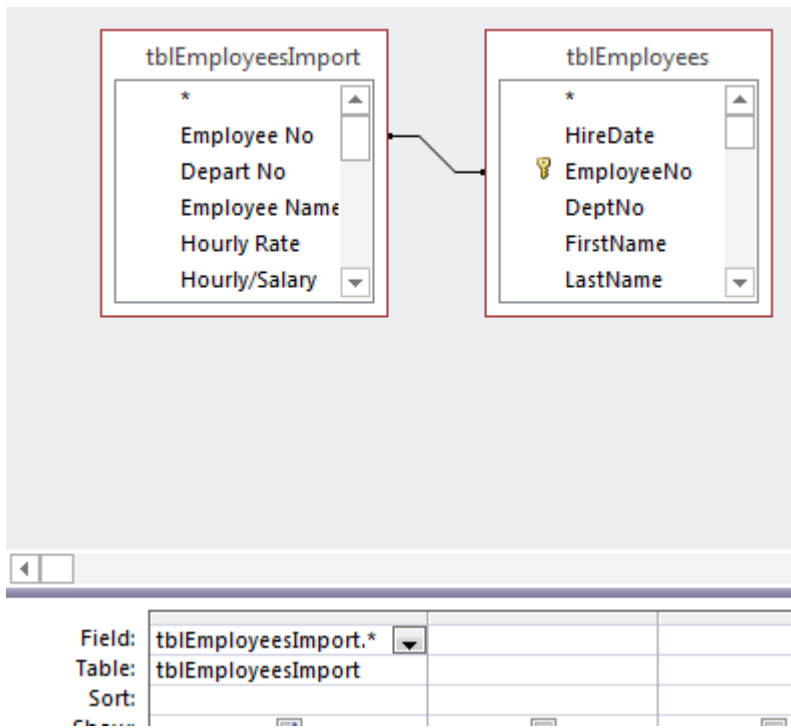
Step 5

Click the Run button to verify that you are looking only at the duplicate records. (8 records)

Employee N	Depart No	Employee Name	Ho
10	20	THOMAS ARQUETTE	
150	10	CAROLYN VALDEZ	
210	80	MARTIN WOODWARD	
500	50	ALAN COOK	
600	20	GLORIA CAULDWELL	
700	50	ERNEST GOLD	
800	10	ANN GARCIA	
99	10	JANICE SMITTY	

Step 6

Now that you know which records are being deleted Click on the Delete button on the Ribbon and then click on the Run Button. All duplicate records are now deleted from tblEmployeesImport.



Step 7

Open tblEmployeesImport and see that the duplicate records have been deleted.

Step 8

Close query and name it **qryDeleteDups**

Now that we have gotten rid of the duplicates we want to append the records in **tblEmployeesImport**

Copying data into tblEmployees

Appending data from one table into another table can often be a difficult problem. Sometimes the fields with the same name may have different data types or the data types may be of different sizes or there may be a field which is required and yet there is no value to place in that field.

In the case of **tblEmployeeImport** we have such a problem. Some of the fields in **tblEmployees** have a required value set to YES and this would prevent the data from **tblEmployeeImport** from actually being appended. So let's solve the problem right up front. **Open tblEmployees in design view and change the required value for fields DateHired and BirthDate. From**

Yes to NO.

Now we are ready for the Append Process

Step 1

Create a new query and select **qryRowsToImport** as the query.

Step 2

Bring all the fields down into the design grid **except for NameSpace, NameLen** Make certain the Show boxes are selected on all the fields.

Step 3

Select the **Append** button from Query type.

Step 4

Select **tblEmployees** from the Append window.

Step 5

Fill in the correct field for each of the fields in the Append to Box.

Append To:	EmployeeNo	DeptNo	HourlyRate	HourlySalaried	EmpType	Terminated	E
------------	------------	--------	------------	----------------	---------	------------	---

Step 6

Click the **run** button.

Step 7

Close out query and save it as **qryEmployeesAppend**

Step 8

Open tblEmployees to verify the data has been appended

Step 9

Close tblEmployees

When you viewed the records in `tblEmployees` all the first and last names were capitalized. The next steps are to update or modify the names so only the first letter of the first and last name are capitalized and the remaining letters are in small case. To complete this process, we will import a utility function from the Sample database.

Import a Standard Module from the Sample database.

Make certain you have copied the file `Samples` over to your computer and place in the same directory as your other CIS28A files.

Step 1

Select the **External Data** tab in Ribbon and then click on **Access** find the folder containing the file “`Samples`” and select that file.

Step 2

Continue with the importing until you come to the **Import Objects** window. Click on the tab **Modules** and you will see the file **`mdUtilityFunctions`**.

Step 3

Select the file and press **OK** and close the window without saving the import steps.

Step 4

Verify that **`mdUtilityFunctions`** is in your Navigation Pane.

Create an Update Query

Step 1

Create a new Query and select **`FirstName`** and **`LastName`** from **`tblEmployees`** table.

Step 2

Select the **Update** button from the Ribbon.

Step 3

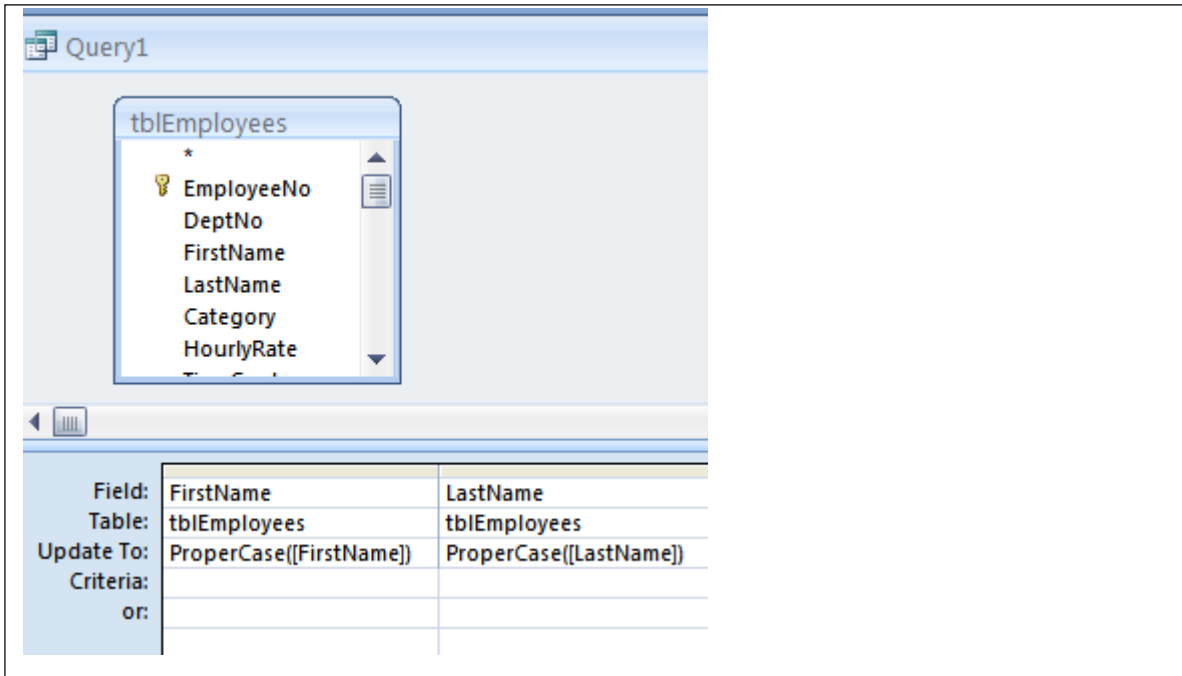
In the `FirstName` field type the following line in the **Update** text box.

`ProperCase([FirstName])`

Step 4

In the `LastName` field type the following line in the **Update** text box.

`ProperCase([LastName])`



Step 5

Run the query and verify the names have been changed.

Step 6

Save the query as **qryChangeToProperNames**

Martin Woodward, one of MovieCam's product manager has provided you with an Excel workbook file "Jobs" which contains data on the jobs that MovieCam has had in the past. Martin would like you to import the data into **tblJobs**.

Remember when we were uploading data for the Employees table we discovered that the table already had Primary Key data that would be duplicated and thus an import would not be possible unless we got rid of the duplicate data in the import file. Since we will be importing data from the Jobs spreadsheet we have to get rid of the duplicate data. You can use the method we used in the Employee table or if you wish you can go into the spreadsheet and delete the two records that are duplicated. You need to change the data types in Excel to the data types in the Access table. Once you have gotten rid of the three duplicate records import the rest of the data into **tblJobs**.

In the process of importing the data be sure to select "Append a copy of the records to

tblJobs”..If you get an error message ignore and continue through the process.

Import the Excel Jobs Spreadsheet into Access:

Step 1

Open **tblJobs** and delete all the records. We will replace them with the import.

Step 2

Close **tblJobs**

Step 3

The Jobs Excel spreadsheet was downloaded earlier into your CIS 28 folder. Using the Ribbon under the tab **External Data** select the **Excel button** and choose the **Jobs Excel spreadsheet**.

Step 4

Next we want to **Append** a copy of the records to **tblJobs** (See Below)

Step 5

The next image shows the records and the headings for the fields.

tblJobs				
JobID	Model	Date Create	Date M	
90000	1BM1000	8/12/2013		
91547	2BM1001	8/12/2013		
92777	ART2000	8/12/2013		
93451	ART2010	8/12/2013		
98378	XCAM108OSB	11/3/2014		
99562	ZTSN1022	11/3/2014		
99899	NS22282	11/3/2014		

Step 6

This completes the import process for the Excel file. Close tblJobs

Martin is also providing you another Excel workbook file “Movies3” that contains sample data used to track time cards and hours.

You plan to import this table and then use it to write the queries needed to populate the tables **tblTimeCards** and **tblHours** in the MovieCam database.

Import the Excel Spreadsheet into Access:

Step 1

The Movie 3 Excel spreadsheet can be found in your CIS 28 folder.

Step 2

Using the Ribbon select the External Data tab and choose the **Excel button** and choose the **Movies3 Excel spreadsheet**.

Step 3

Continue through the process making certain you select the first row of data as the column heading. We will convert the fields later. Select **NO Primary Key**.

Step 4

At the end of the process save the data into a table you will label **tblTimeCardsImport**. Do not save the import steps.

Step 5

Open **tblTimeCardsImport** in Design View and change **DeptNo**, **TimeCardNo**, **EmployeeNo** and **JobNo** from **text** data to Number and Long Integer.

Step 6

Close **tblTimeCardsImport**. 56 Records have been appended.

Storing data in the Excel file led to a lot of data duplication. Data in the spreadsheet will often be repeated in the table which you are importing. In this case tblTimeCards can be deleted and the data replenished with the import data time. To minimize duplicate data in the table, you apply the Unique Values property in a query. Table tblTimeCards and table tblHours have duplicate data that must be deleted before you import the data from the spreadsheet.

To Append Unique Data to the tblTimeCards Table:

Step 1

Create a **new query** in design view, add the **tblTimeCardsImport** table to the Select query window, and close the **Show table dialogue box**.

Step 2

Add the **TimeCardNo**, **TimeCardDate**, and **EmployeeNo** fields to the design grid in the lower pane of the Select Query window.

Step 3

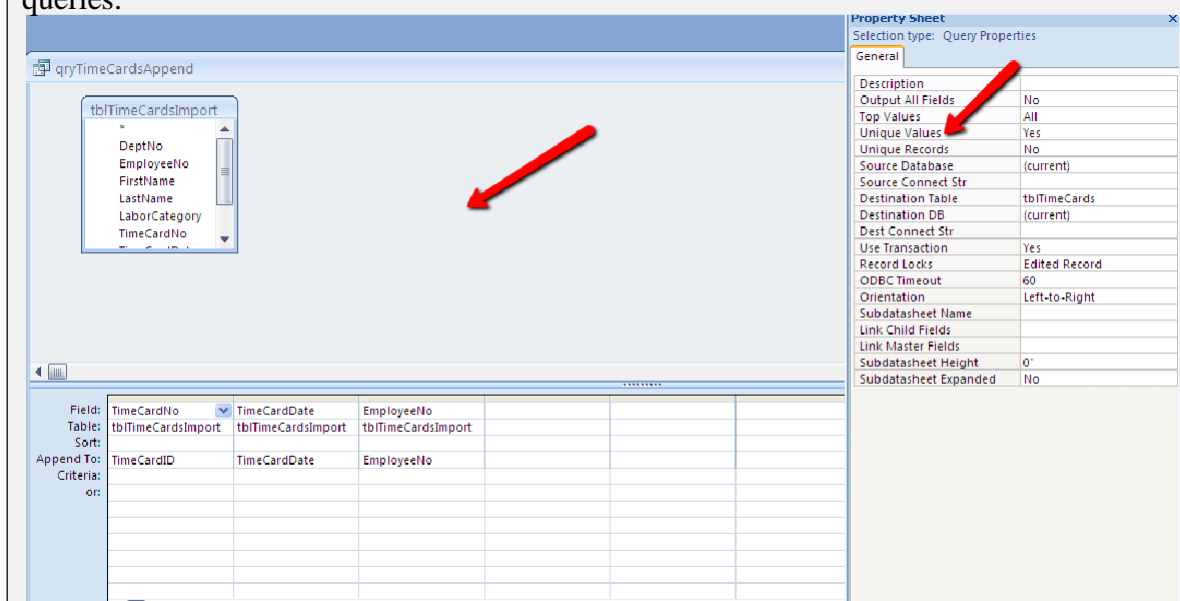
Run the query to test the results. There should be **56 records** in the results. Note the duplicate data.

Step 4

Switch to **design view** and then right-click anywhere in the window outside of the table or fields.

Step 5

Select properties from the dropdown list and change the Unique Values property to **Yes**. The Unique Values property applies to select queries and make-table and append action queries.



Step 6

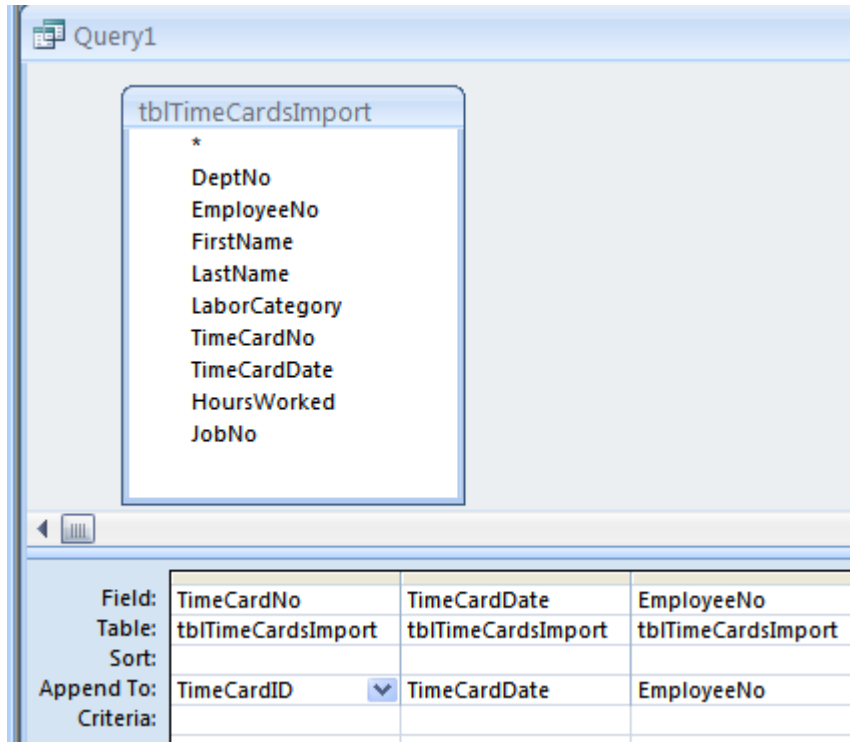
Run the query again and you should only see **10 unique records**.

Step 7

Return to **Design view**, click **Append query** to open the Append query dialogue box.

Step 8

Select **tblTimeCards**, click **OK** and **run** the query and then **verify** in the Design grid that all the fields have been appended.



Step 9

Run the query and save it as “**qryTimeCardsAppend**”.

Step 10

Open table **tblTimeCard** to verify your work. Close the table.

We are going to use the same process as above to load data into tblHours. Remember that tblHours had 2 Primary Keys. Unfortunately, this make it difficult to load data that doesn't follow the integrity rules of Databases. So we must get rid of the restraints before we load the excel file.

Next go into the design view of tblHours and remove the Primary keys from both fields. Make certain that no field has the “**Required properties**” set to **Yes**.

We have set up tblHours to accept data without any constraints. You will build back in the constraints after the data is loaded.

Create a new query using **tblTimeCardImport** and label it **qryHoursAppended**. Append the data to **tblHours** using fields **TimeCardNo**, **HoursWorked**, and **JobNo** fields.

The screenshot shows the 'qryHoursAppended' query design view in Microsoft Access. The design grid is as follows:

Field:	TimeCardNo	HoursWorked	JobNo	
Table:	tblTimeCardsImport	tblTimeCardsImport	tblTimeCardsImport	
Sort:				
Append To:	TimeCardID	HoursWorked	JobID	
Criteria:				
or:				

Once the data is appended **manually** input the line item numbers as **show below**.

tblHours			
Line Item No	Time Card N	Job ID	Hours Worked
1	106	99562	10
2	106	98378	12
3	106	99899	8
4	106	99562	7
5	106	99899	3
1	107	99899	8
2	107	99562	7
3	107	98378	12
4	107	90000	4
5	107	99562	5
6	107	98378	4
1	108	99562	4
2	108	98378	6
3	108	90000	10
4	108	98378	8
5	108	99562	4
6	108	99899	8
1	109	99562	5
2	109	99899	5
3	109	99562	10

Go back and enter the restraints: relationship window, primary Keys, and required fields. The primary keys can have duplicates but the combination of the two Primary Keys must be unique

The table should have 56 records.

Amanda wants you to optimize database performance by storing the outdated transactional data in separate tables. She wants you to develop queries for archiving data that is more than a year old. The time card and hour's data go back several years.

Archiving Data

When you are archiving data it is usually a good idea to make a backup of the data before you begin.

Step 1

Make a backup copy **tblHours** and **tblTimeCards**,
(Rt-click the table you wish to copy, select copy, then rt-click in navigation pane and select paste)

Step 2

Create a new query using **tblHours** and **tblTimeCards**.

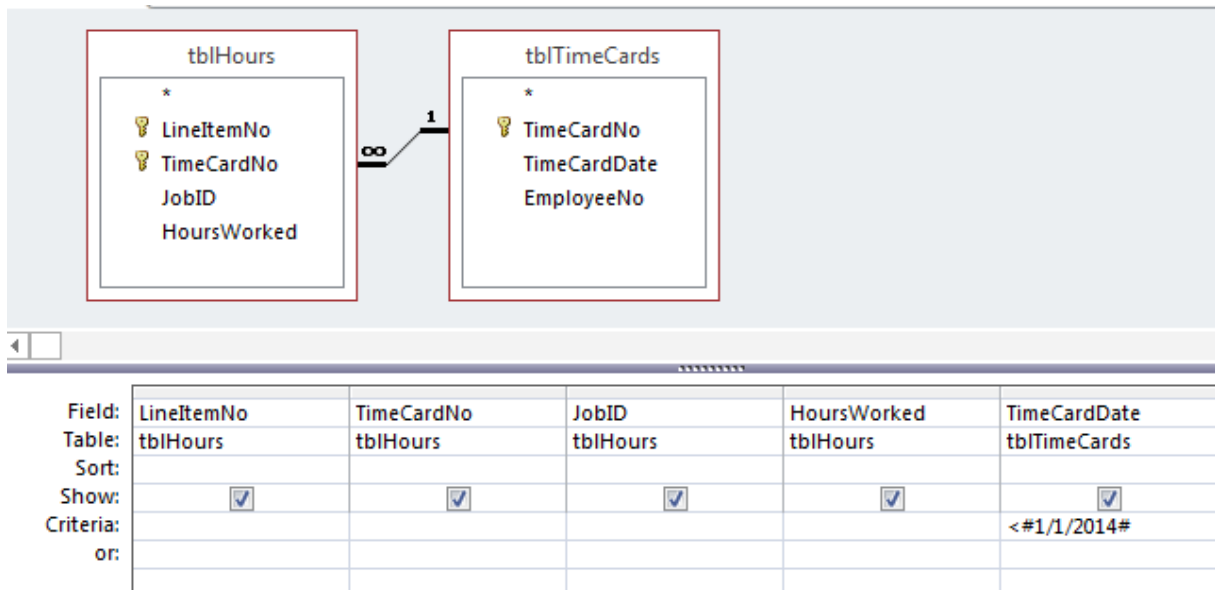
Step 3

Make certain the two tables are connected through the common field **TimeCardNo**.

Step 4

Pull down **LineItemNo**, **TimeCardNo**, **JobID**, **HoursWorked** from **tblHours**. Pull down field **TimeCardDate** from **tblTimeCards**. Set the criteria of field **TimeCardDate** to:

<#1/1/2014#.



The screenshot displays the Microsoft Access Query Design View. Two tables, **tblHours** and **tblTimeCards**, are shown. **tblHours** has fields: **LineItemNo** (primary key), **TimeCardNo** (foreign key), **JobID**, and **HoursWorked**. **tblTimeCards** has fields: **TimeCardNo** (primary key), **TimeCardDate**, and **EmployeeNo**. A one-to-many relationship is established between **TimeCardNo** in **tblHours** and **TimeCardNo** in **tblTimeCards**. Below the design view, a table shows the query criteria:

Field:	LineItemNo	TimeCardNo	JobID	HoursWorked	TimeCardDate
Table:	tblHours	tblHours	tblHours	tblHours	tblTimeCards
Sort:					
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Criteria:					<#1/1/2014#
or:					

Step 5

Select the **Make Table** button from the Ribbon. Name the table “**tblHoursArchived**”

Step 6

Run the query

Step 7

Save the query as qryMaketblHoursArchived.

Complete the same process by making a table “**tblTimeCardsArchive**” using only the table tblTimeCards and all three fields. Set the date criteria to <#1/1/2014#

Step 8

Verify that both **tblHoursArchived** and **tblTimeCardsArchived** have the archived records.

Step 9

Save the query as **qrytblTimeCardsArchive**.

Step 10

Create queries to **Delete** records **archived** in both tblHours and tblTimeCards

Step 11

Don't save the queries

Step 12

Verify that all your data is correct and then **Delete** the backup copies of **tblTimeCards** And **tblHours**.

Amanda would like you to look into the potential usefulness of sub queries as well as other commands and clauses available to you in Structured Query Language (SQL), which is the query language Access uses.

Finally, Amanda wants you to develop queries to integrate the archived records, with the active records for those situations when the product managers need to compile statistics from all the data.

To Write a Query in SQL:

Step 1

Create a new query in Design view, and then closes the Show table dialog box without selecting a table.

Step 2

Click the **SQL** button in the Ribbon. Access automatically inserts and selects **SELECT** into the otherwise empty SQL view window; if it does not, this is not a problem.

Step 3

Type **SELECT * FROM tblHours;** and then run the query to check the results. The record set contains **28 records** showing four fields from the **tblHours table**.

Step 4

Switch to SQL view



To Add a WHERE clause in SQL:

Step1

Position the insertion point at the end of the SQL statement **SELECT tblHours. * FROM tblHours;** and then press the **Enter** key, making sure the semicolon is at the end of the code.

Step 2

Type **WHERE tblHours.TimeCardNo = 106.** Quotation marks are not required around the number because the TimeCardNo field is a Number field. The line of code should end with a semicolon.

Step 3

Run the query and view the results. The **5 records** with TimeCardNo equal to 106 should appear.

Step 4

Switch back to SQL view.

To use the ORDER BY Clause in SQL:

Step 1

Click to position the insertion point before the semicolon at the end of the WHERE clause, and then press the **Enter** key.

Step 2

Type **ORDER BY tblHours.LineItemNo;**

Step 3

Run the query and check the results. The records are sorted in ascending order by the LineItemNo.

Use the JOIN clause in SQL:

Step 1

Position the insertion point at the end of SELECT tblHours. *, and type ", **tblTimeCards.***" (Make certain you **insert a comma** before tblTimeCards.*. Press the enter key to create a second line.

Step 2

Position the insertion point after the FROM tblHours, press the **Spacebar**, and then type **INNER JOIN tblTimeCards ON tblHours.TimeCardID = tblTimeCards.TimeCardID**

Step 3

Run the query to check your results

Step 4



```
SELECT tblHours.*,tblTimeCards.*  
FROM tblHours  
INNER JOIN tblTimeCards ON tblHours.TimeCardNo = tblTimeCards.TimeCardNo  
WHERE tblHours.TimeCardNo= 106  
ORDER BY tblHours.LineItemNo;
```

Line Item No	Time Card N	Job ID	Hours Worked	Time Card N	Time Card D	Employee N
1	106	98378	12	106	11/3/2014	10
2	106	99899	8	106	11/3/2014	10
3	106	99562	7	106	11/3/2014	10
4	106	99899	3	106	11/3/2014	10
5	106	99562	10	106	11/3/2014	10
*	0	0	0			

Use the GROUP BY Clause in SQL

Step 1

Create a new query in Design view, Close the Show Table dialog box without selecting any tables, and then switch to SQL view.

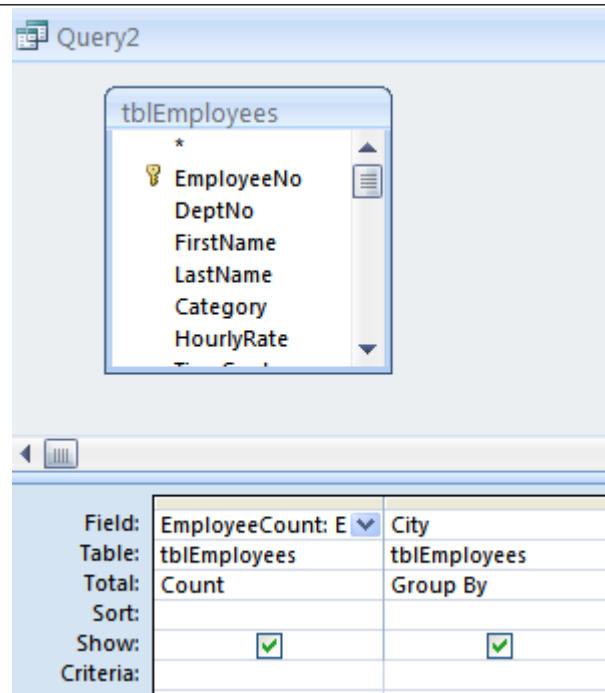
Step 2

Type the following:

```
SELECT COUNT (tblEmployees.EmployeeNo) AS  
EmployeeCount, tblEmployees.City  
FROM tblEmployees  
GROUP BY tblEmployees.City;
```

Step 3

Switch to Design view to see how the query looks in the grid: (See below) Note that the name that appears in the title bar of the query window is “Query2” unless you had other practice queries open as well. The number in the query name depends on how many queries you have created in a session and does not affect the query itself. You will save this query with a different name later.



Step 4

Run the query to test the SQL statement. The results should show the records grouped by city.

Step 5

Switch to SQL view.

Using the HAVING clause in SQL:

Step 1

Modify the SQL statement by adding the HAVING Clause:

HAVING (((tblEmployees.City) Like "S*"));

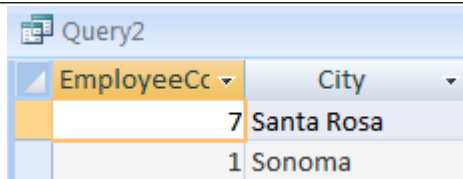
Remember to only have one semicolon (;) which is at the end of the SQL statement.

Step 2

Switch to Design view, note that Like "S*" has been added to the Criteria text box.

Step 3

Switch to datasheet view and your results should appear as below:



EmployeeCc	City
7	Santa Rosa
1	Sonoma

Step 4

Switch to SQL view.

Step 5

Close this query and save it as “**tmpqryHavingWhere**”.

Step 6

Submit your work for grading.