**Microsoft Access**

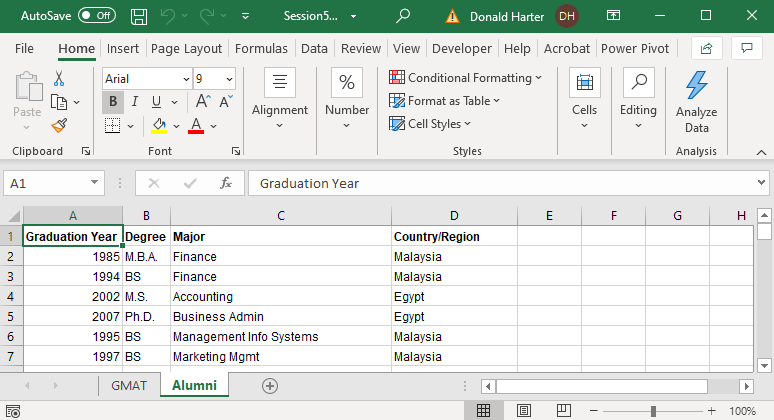
**Background**

In Excel, we could have multiple spreadsheets within a workbook. However, these spreadsheets did not necessarily relate to each other. In Microsoft Access, we will use a similar concept, but it’s called tables. A table is like a spreadsheet. The key difference is that multiple tables can relate to each other and be combined to form calculations, summaries, queries, and reports. When tables are combined, we say they are joined. Tables can be joined based on relationships. This tutorial explores tables, relationships, and their uses in retrieving and adding data to a database.

**4.3 MS Access: Importing Data**

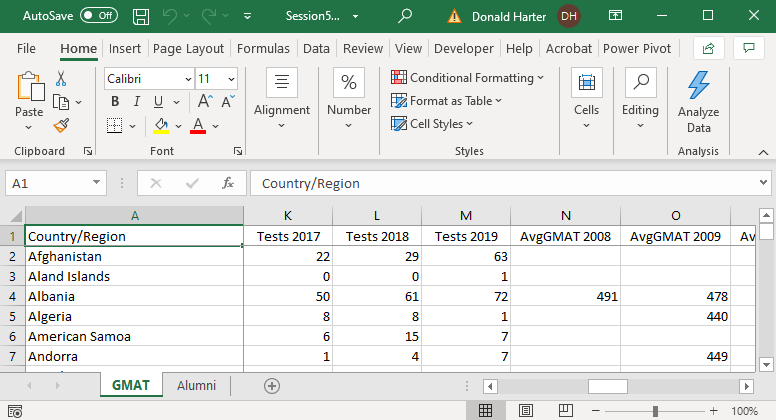
Generally, if you are starting from scratch, it’s easiest to build data into an Excel spreadsheet and import it into Access. Excel allows data to easily be imported from the web or entered directly. Editing and formulas can manipulate the data before you convert it to Access format.

Let’s first examine data in the Alumni tab of your spreadsheet.



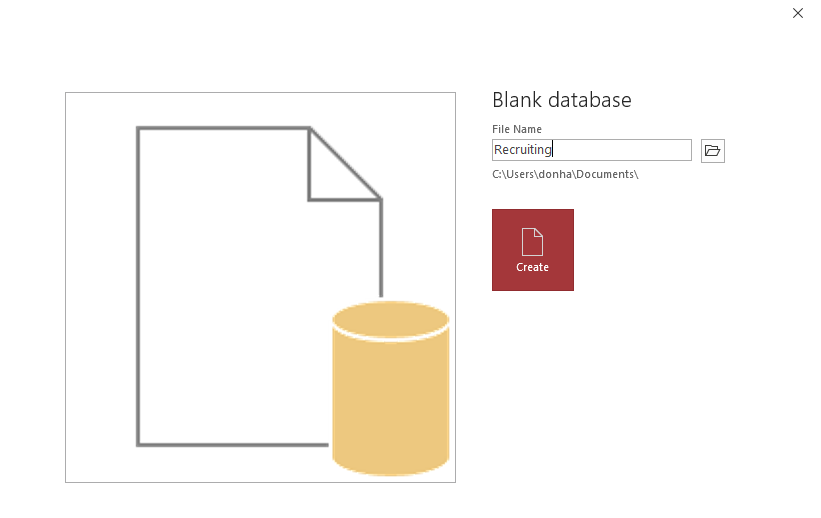
Data includes graduation year, degree, major, and country/region for each student.

Data for GMAT test takers is shown in the GMAT tab.

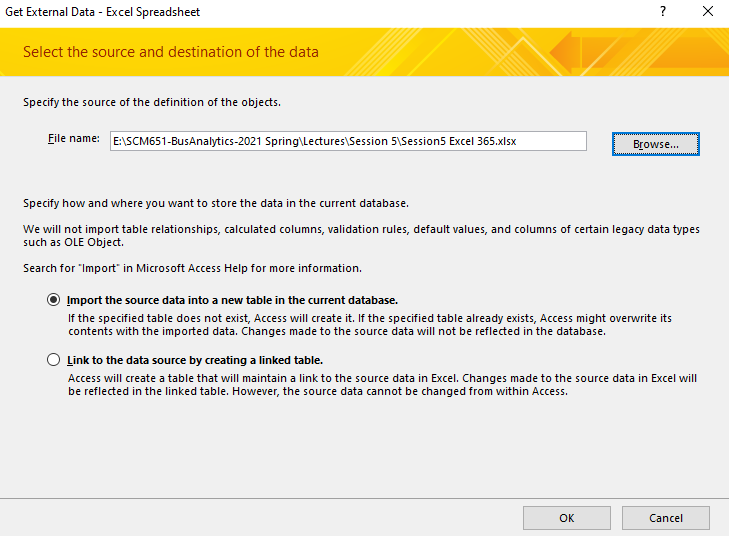


Let’s import the data from Excel to Access by using the Access Import capability.

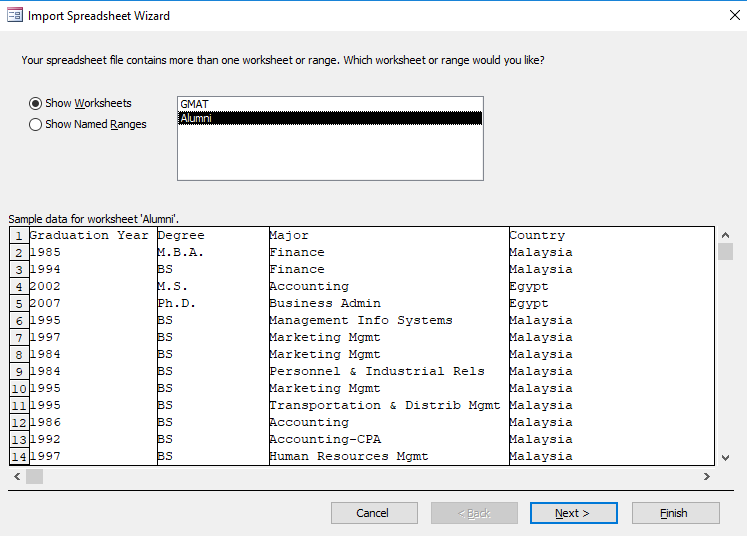
1. Open Microsoft Access by clicking on the Start button, all Programs, Microsoft Office, Access.
2. Click on Blank Database.
3. In the File Name field, enter Recruiting, then click Create.



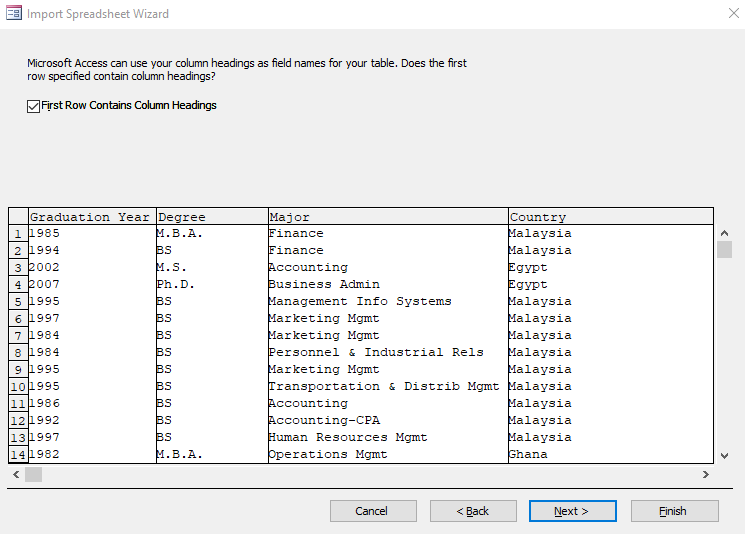
1. A default table called Table 1 was created; to delete this table, click on the X at the top of Table1.
2. Click on the tab External Data, then New Data Source, From File, Excel.
3. Search for file with Browse, Open, OK.



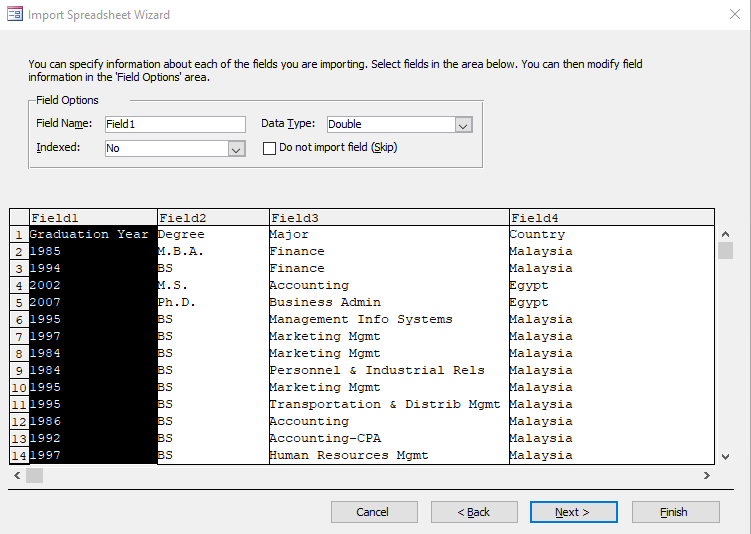
1. Select the appropriate worksheet from the workbook. In this case, use Alumni, then Next.



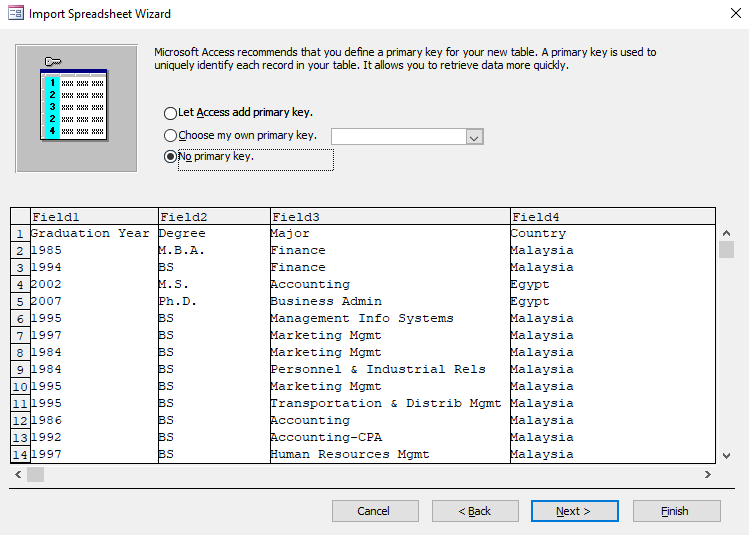
1. Access will try to detect if you have labels in the first row. Check the box for First Row Contains Column Headings and click Next.



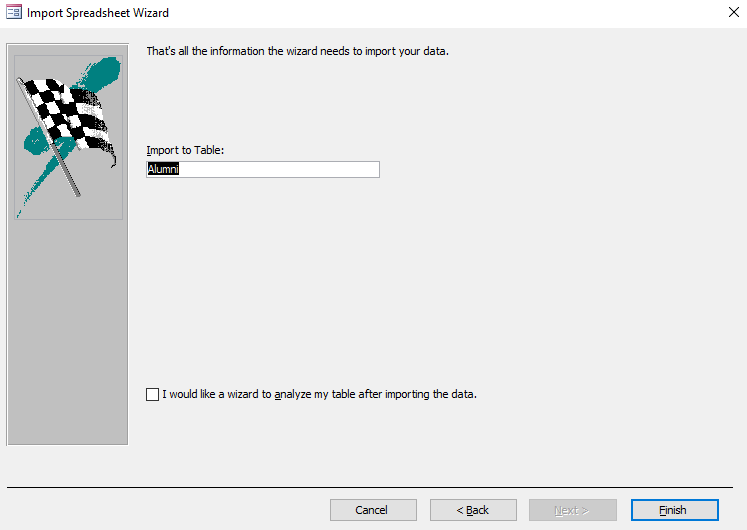
1. The next screen allows you to determine whether each column is text, numeric, etc., and set the characteristics. Access is pretty good at setting these automatically. Click Next.



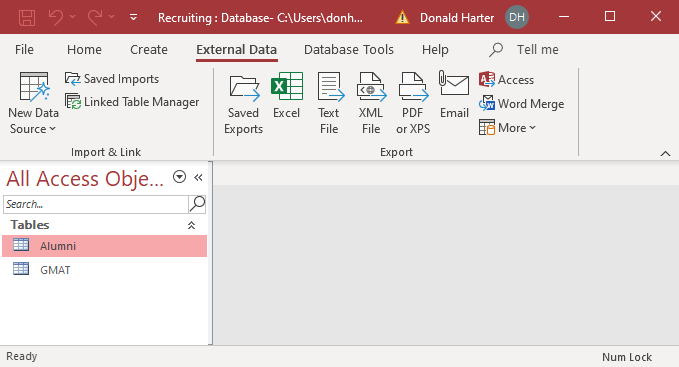
1. Next, Access allows you to add a primary key. Primary keys allow you to uniquely identify each row in a table. However, if you let Access add one, it always adds an arbitrary key that means nothing. Click on No primary key, then Next.



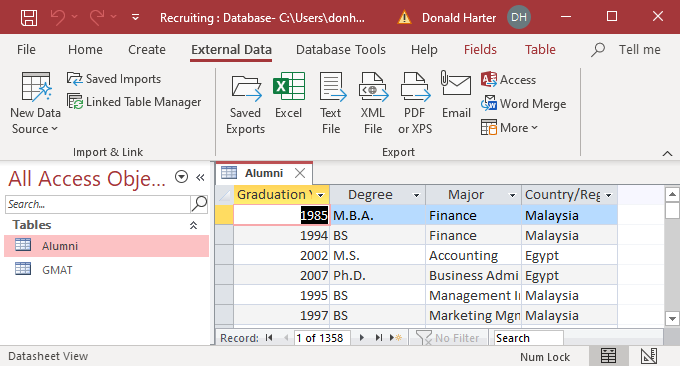
1. The last screen allows you to name the table. Let’s call it Aumni. Click Finish, then Close.



1. Now follow the same set of steps to add the GMAT data from your spreadsheet. Both tables should appear in your Access database.



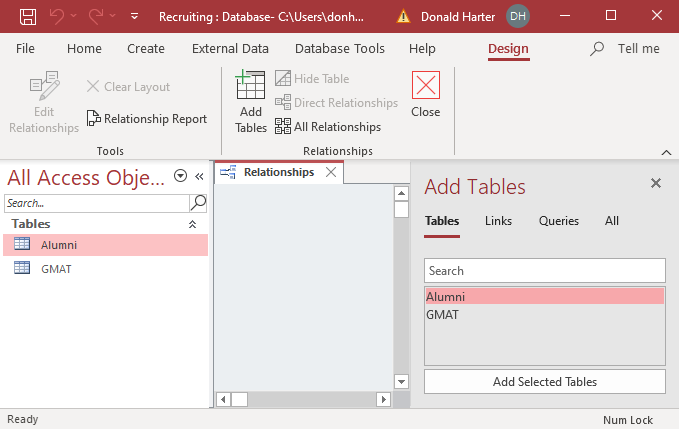
1. Double click on the table name Alumni to view the data.
2. Click on the X next to Alumni when you are done.



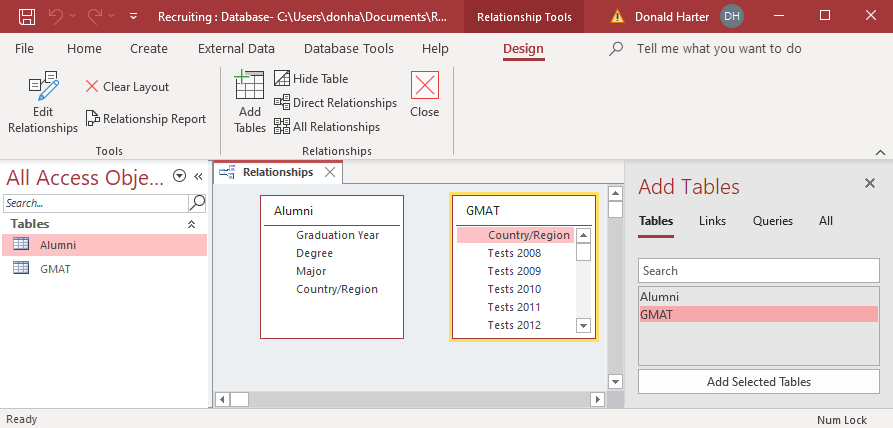
**4.4 MS Access: Creating Relationships**

Relationships describe how tables relate to each other.

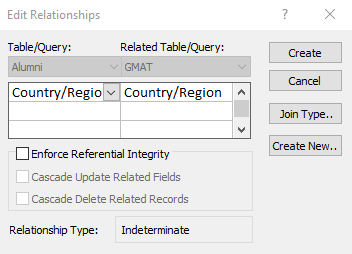
1. Click on the Database Tools tab, then Relationships. You should see the following pop-up.



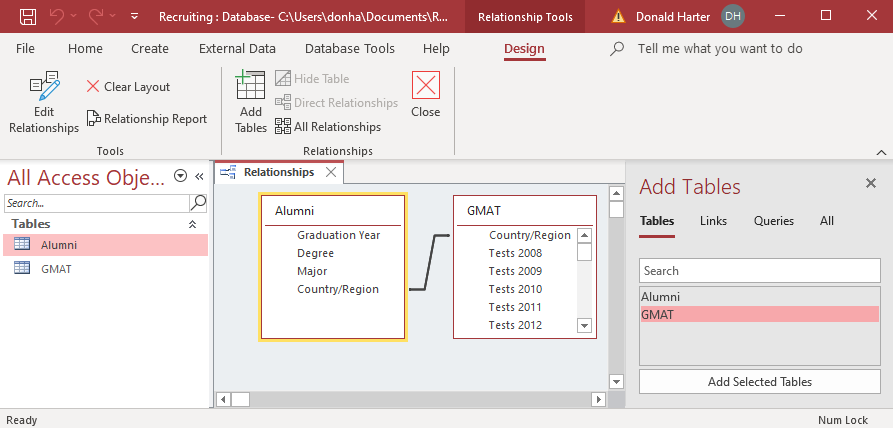
1. On the right side, double click on Alumni.
2. On the right side, double click on GMAT.



1. There is no relationship (yet) between these tables. Click on Country/Region in Alumni, then drag to Country/Region in GMAT.



1. Now click Create. This “joins” the two tables using the field Country/Region in each table.
2. Click on the X to the right of Relationships. When it asks if you want to save Relationships, click Yes.

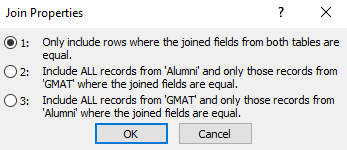
****

**Join Types**

There are four types of joins:

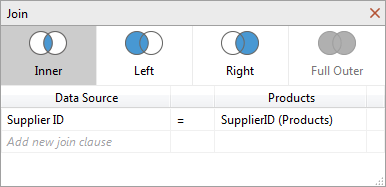
* Inner join: Only include rows where the joined fields from both tables are equal
* Left join: Include all records from the left table and only those records from the right table where the joined fields are equal
* Right join: Include all records from the right table and only those records from the left table where the joined fields are equal
* Outer join: Include all records from the left and right table where the key field appears in either table

Access only supports inner join, left join, and right join:



Outer join is available via SQL, Tableau, and Power BI.

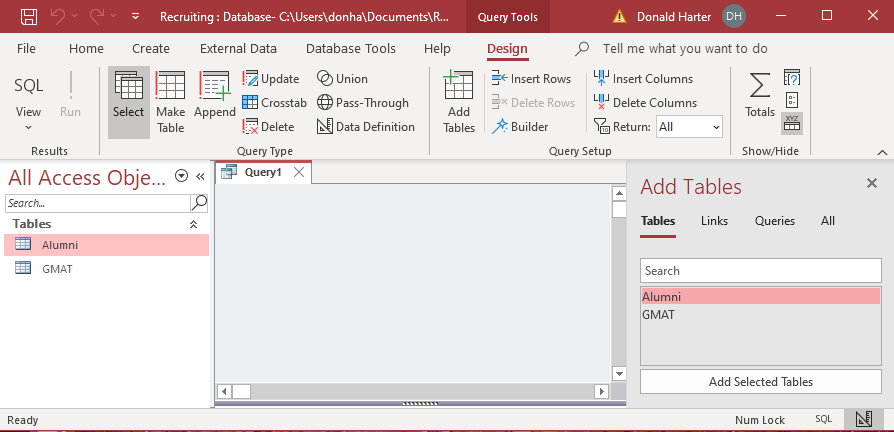
Graphical representation of each join is shown below.



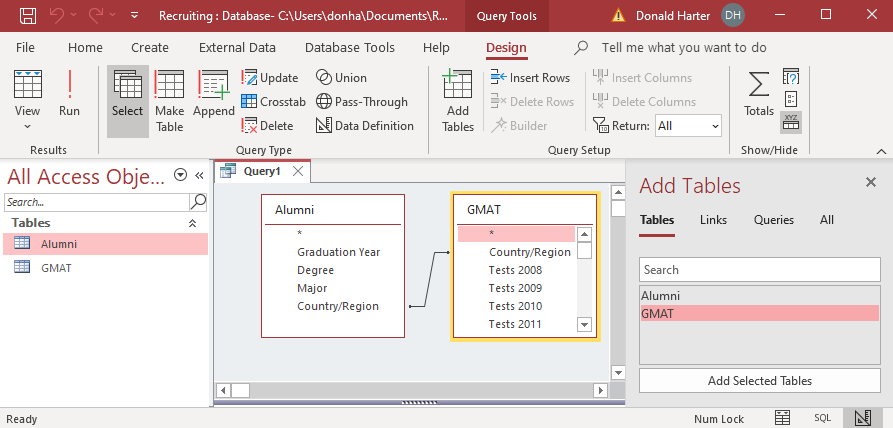
**4.5 MS Access: Simple Queries**

When you want to retrieve specific data from a database, you can use a query. A query is like a filter in Excel, but much more powerful. It can allow you to restrict what data is displayed, sort the data, and perform calculations. However, dirty data can corrupt your queries. Let’s see how.

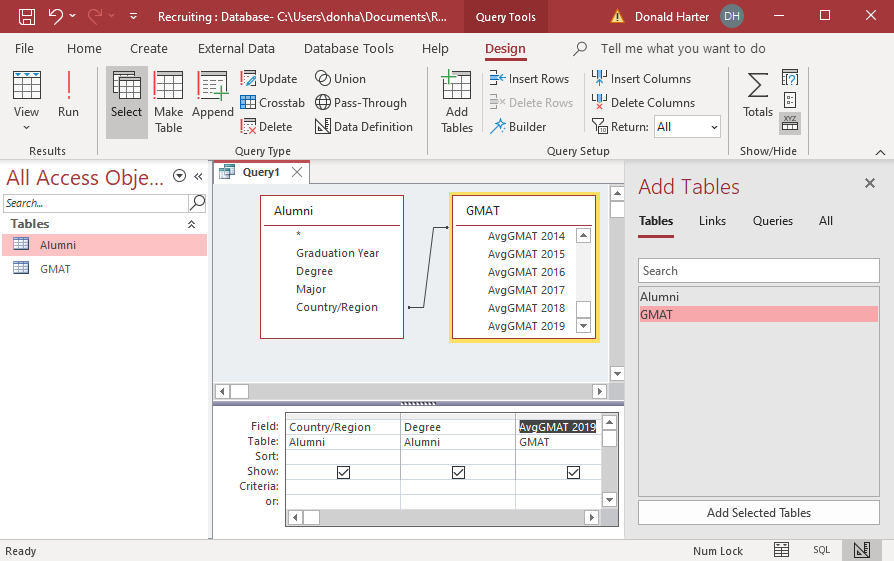
1. To create a query, click on CREATE, then Query Design.



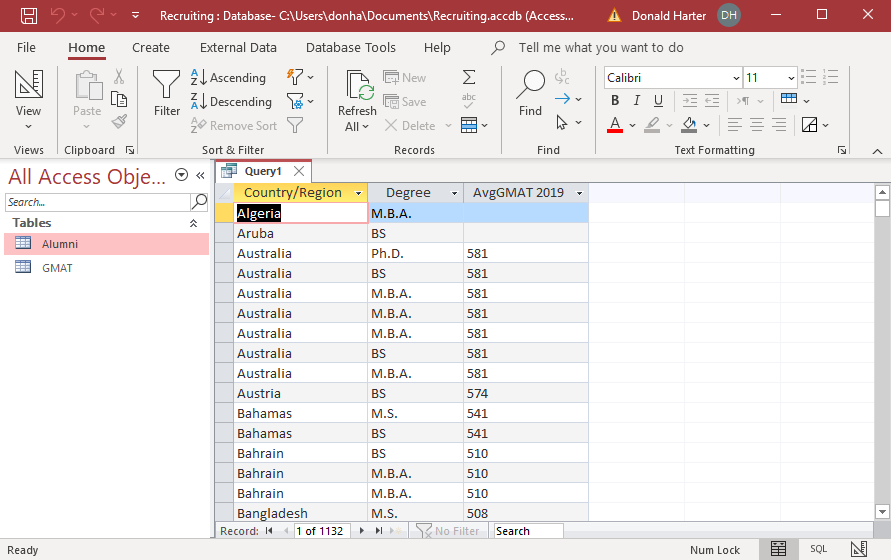
1. On the right side, double click on Alumni, then double click on GMAT.



1. In the table Alumni, double click Country/Region.
2. In the table Alumni, double click Degree.
3. In the table GMAT, double click AvgGMAT 2019.

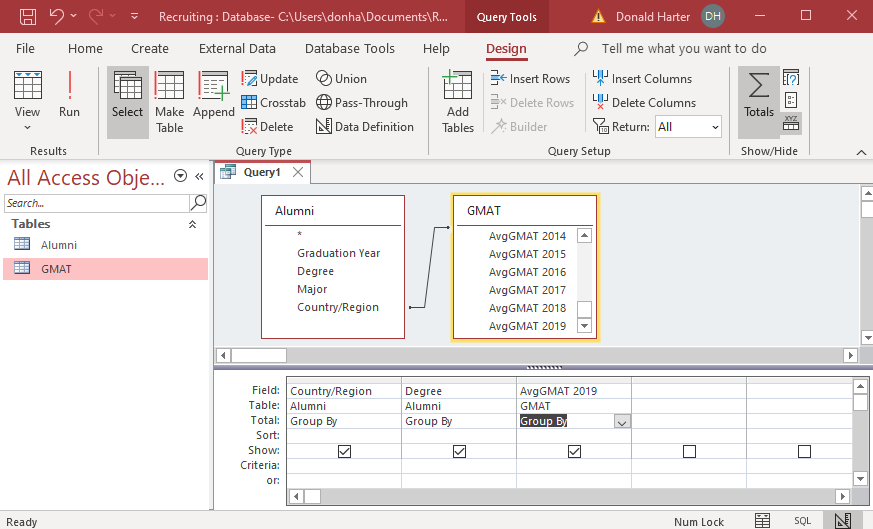


1. Now click Run ! in the upper left corner.



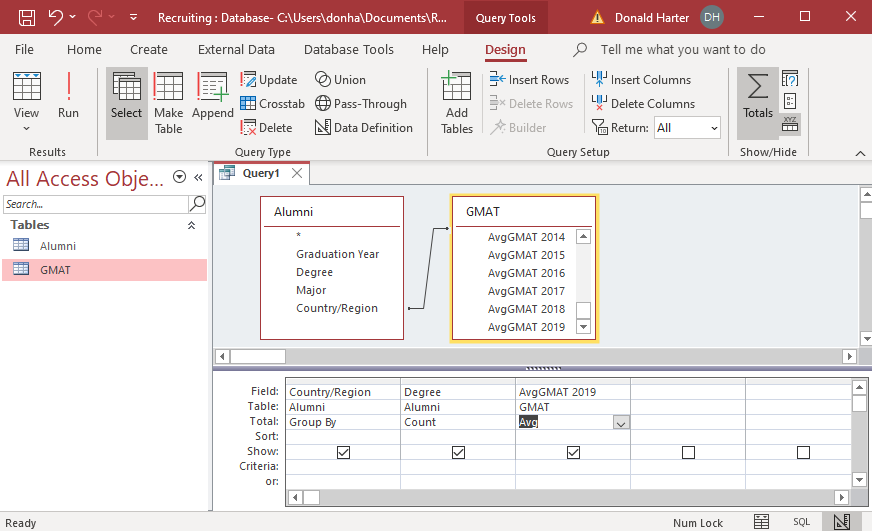
This lists each student, rather than a summary.

1. Click on View, Design View, to go back to the query.
2. Click on the Sigma (summation sign).

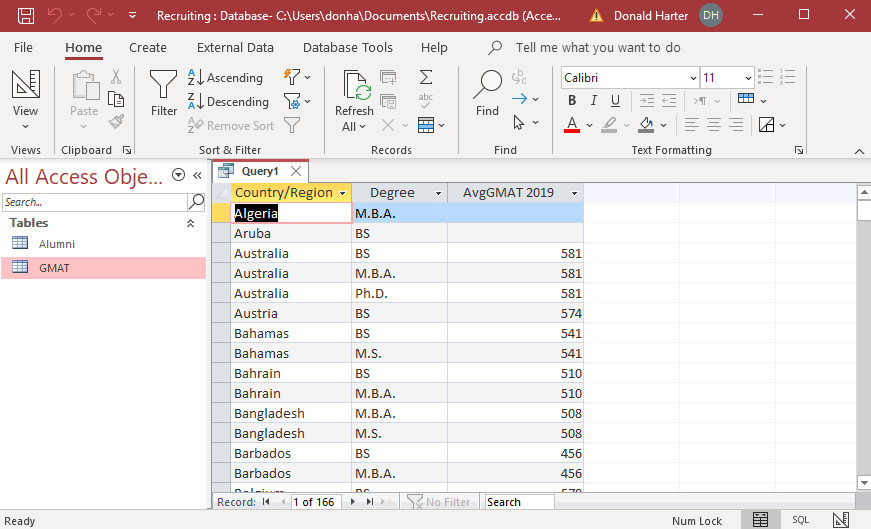


It now adds the Total line Group By. Group by collapses or aggregates for a variable. We want to Group By country/region, but we really want to count degrees and average the GMAT scores.

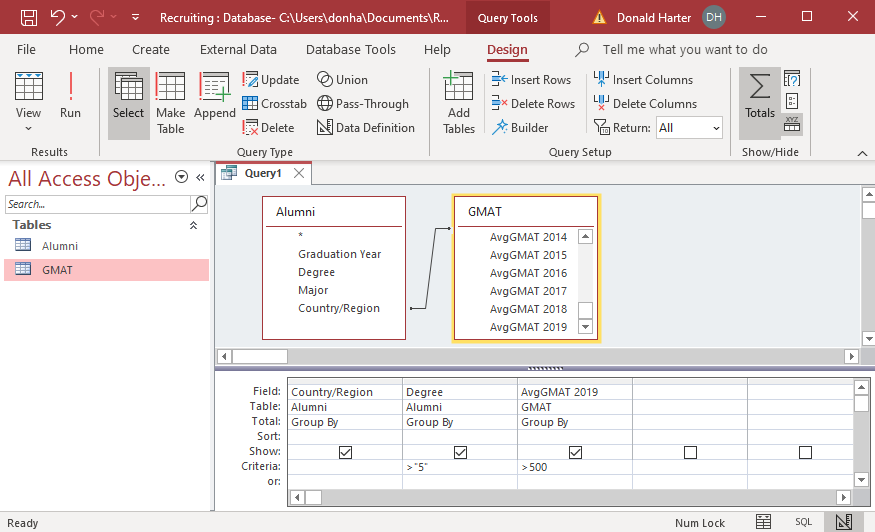
1. Where it says Group By under Degree, use the drop-down menu to change to Count.
2. Where it says Group By under AvgGMAT 2019, use the drop-down menu to change to Average (Avg).



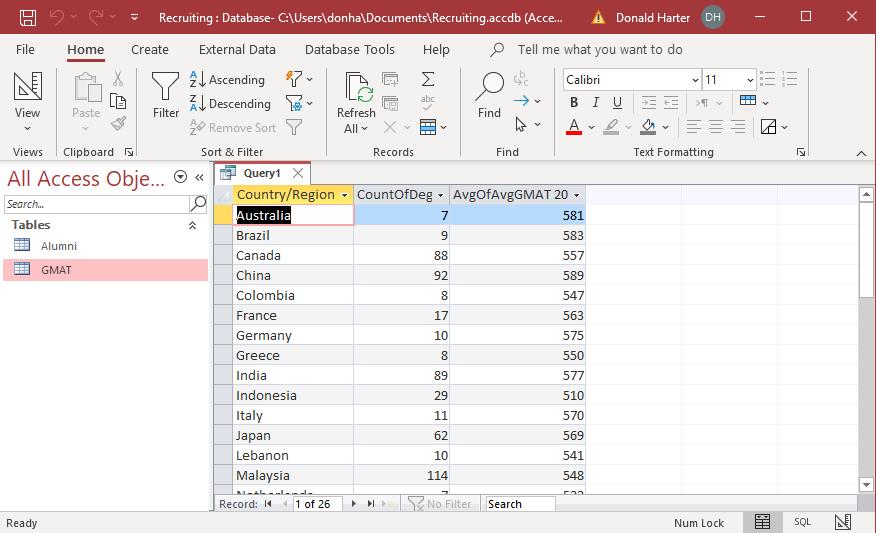
1. Now click Run !.



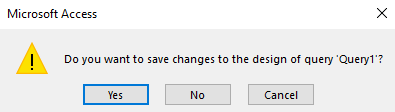
1. Go back to the Design and use the Sort field to change how data is sorted. You can select Ascending (increasing) or Descending (decreasing).
2. The criteria field allows you to filter results. Click on View, Design. Find the countries that have more than five alumni and average GMAT more than 500 by using >5 and >500.



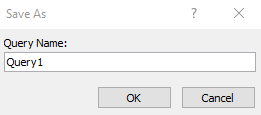
1. Click Run !.



1. To close the query, click on the X next to Query 1.
2. When asked if you want to save, click Yes.



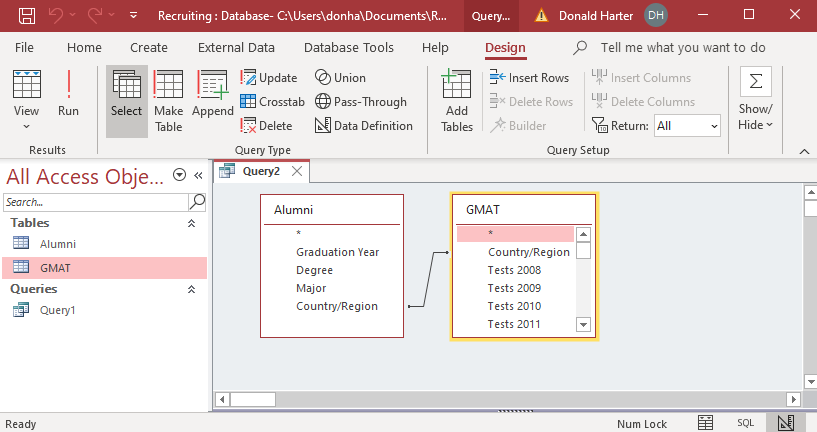
1. Type in the name of your query, then click OK.

****

**4.6 MS Access: Fixing Dirty Data**

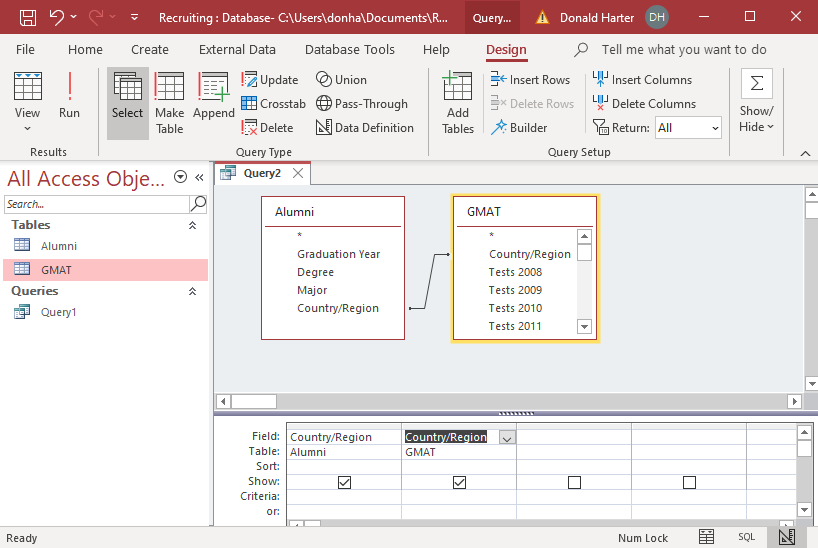
When you want to retrieve specific data from a database, you can use a query. A query is like a filter in Excel, but much more powerful. It can allow you to restrict what data is displayed, sort the data, and perform calculations. However, dirty data can corrupt your queries. Let’s see how.

1. To create a query, click on CREATE, then Query Design.
2. Double click on Alumni, then double click on GMAT.

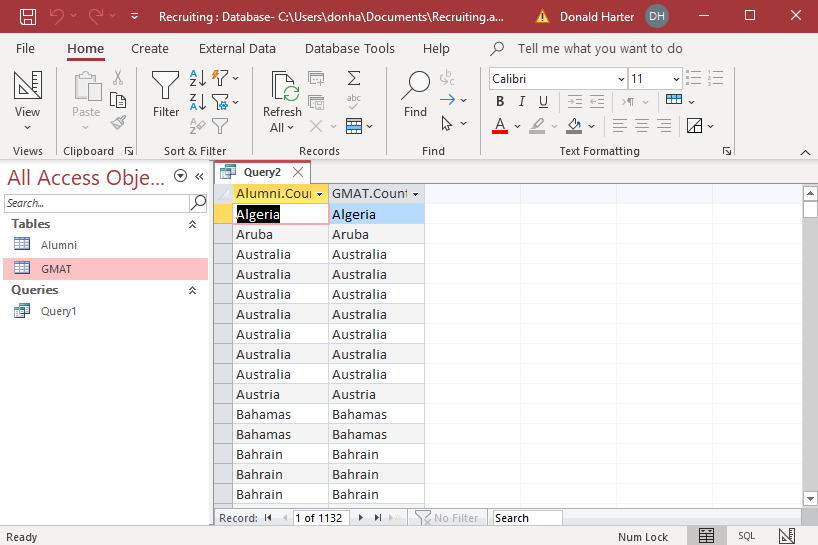


Let’s compare Country/Region in the Alumni table to Country/Region in the GMAT table.

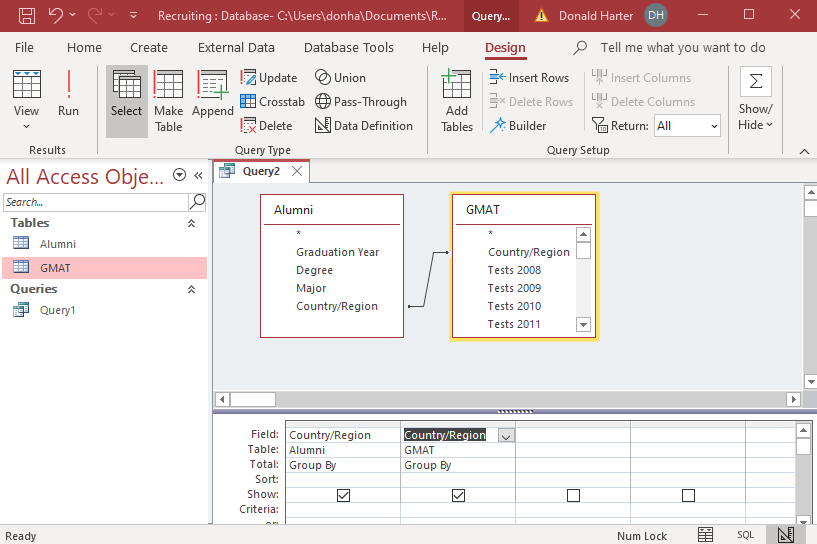
1. In the table Alumni, double click Country/Region.
2. In the table GMAT, double click Country/Region.
3. Click Run !



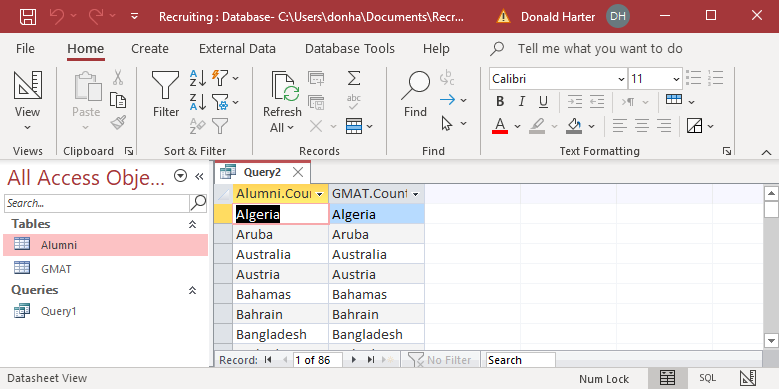
1. Now click Run ! in the upper left corner.



1. Notice that this lists duplicates of each country. Next, we will eliminate duplicates.
2. Click on View, Design View.
3. Click on the summation sign (Sigma) for Totals.
4. A Group By appears.
5. Click Run !.



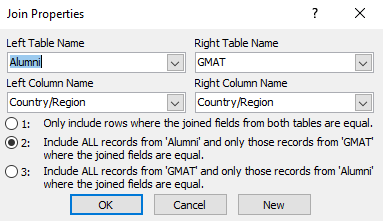
1. Now duplicates are eliminated.

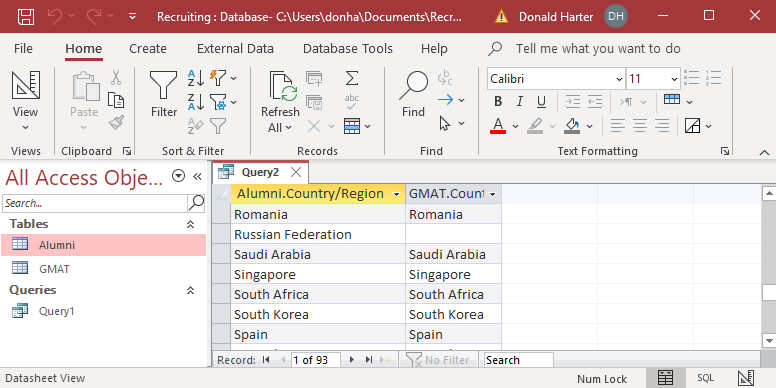


1. Look for Russia in the data. Does it appear?
2. Open the table for GMAT. Does Russia appear?
3. Open the table for GMAT. Does Russia appear?
4. What’s the difference?

Now, let’s allow Access to identify data discrepancies.

1. In your query, click on View, Design View.
2. Right click on the line connecting the tables.
3. Click on Join Properties
4. Change the selection to 2: Include All records from “Alumni” and only those records from “GMAT” where the joined fields are equal, then click OK. This is called a Left Join.
5. Click Run !.
6. Are there any entries in column 1 (Alumni.Country/Region) that have blank entries in column 2 (GMAT.Country/Region)? Why?

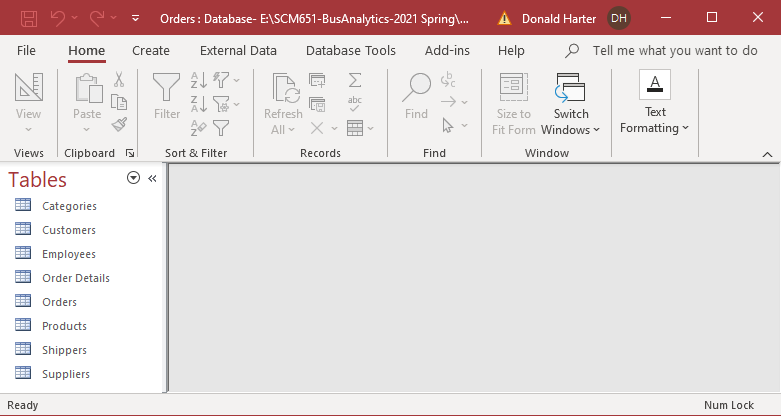




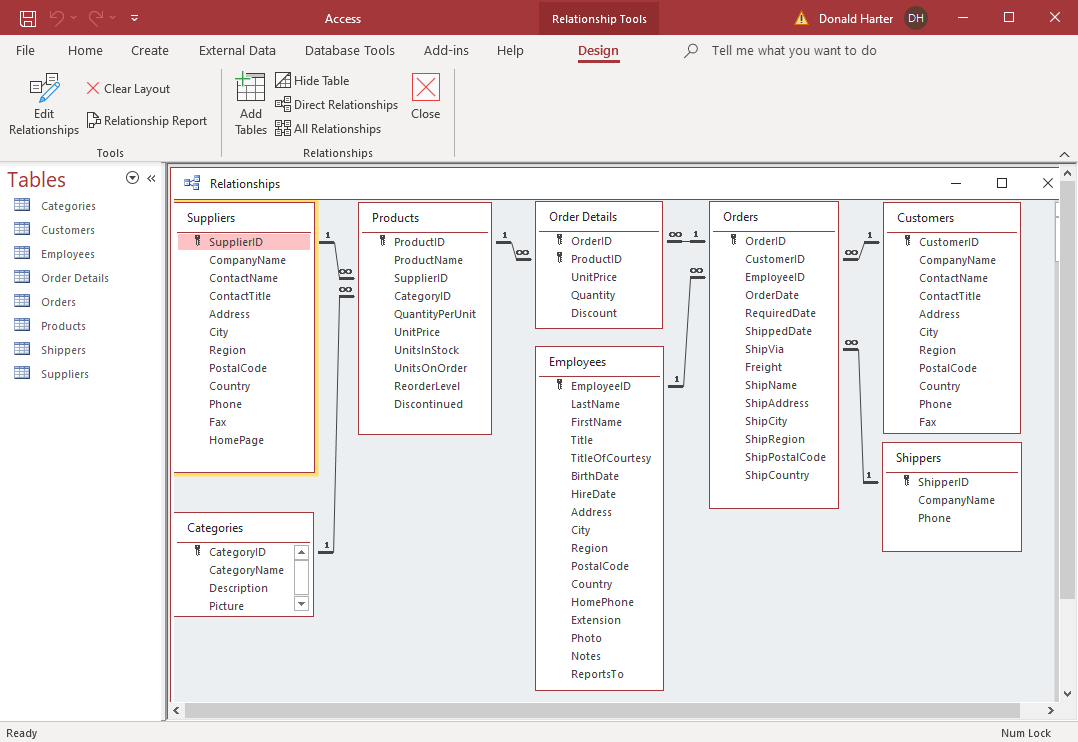
1. Change the relationship to selection 3. This is a right join. What happens?
2. Inner joins hide dirty data. Left joins and right joins identify inconsistencies in the key fields, exposing dirty data. The full outer join, available in SQL, Tableau, and Power BI, identifies dirty data in both directions simultaneously.

**4.7 MS Access: Complex Relationships**

1. Open the database titled Orders.



1. Click on Database Tools, Relationships.

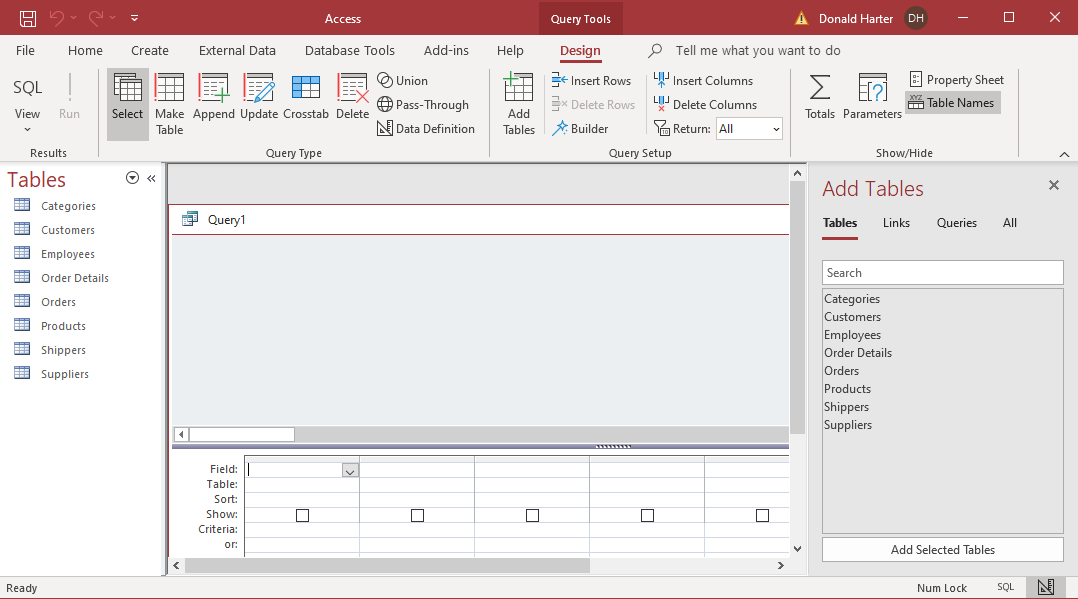


1. The relationships are already defined. If they were not defined, you could add the relationships as shown earlier in this session.
2. What does the notation on the relationship line mean? One-to-infinity means that a one-to-many relationship exists. For every one item in one table, there can be many items in the second table.
3. Key fields are identified by a picture of a key.
4. When a key field in one table is joined to a non-key field in another table, that non-key field is called a foreign key.
5. When you enforce referential integrity, all non-key fields in one table must have an instance of each value as the key in the other table. For example, if a SupplierID appears in the Products table, then the same SupplierID must also appear in the Suppliers table.
6. Click Close to close the relationship screen.

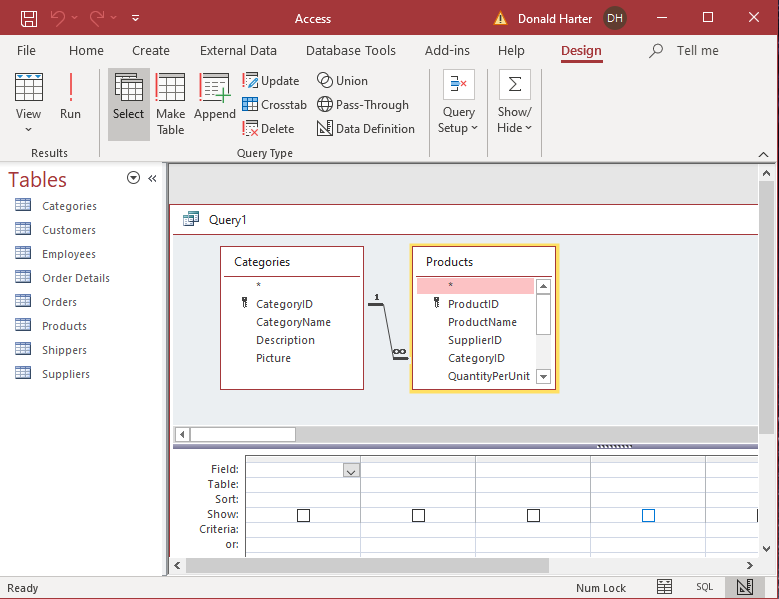
**Complex Queries**

Let’s find all ProductNames that have a CategoryName of beverages. Looking at the tables in the relationship diagram, can you find ProductNames and CategoryName? What tables are they in?

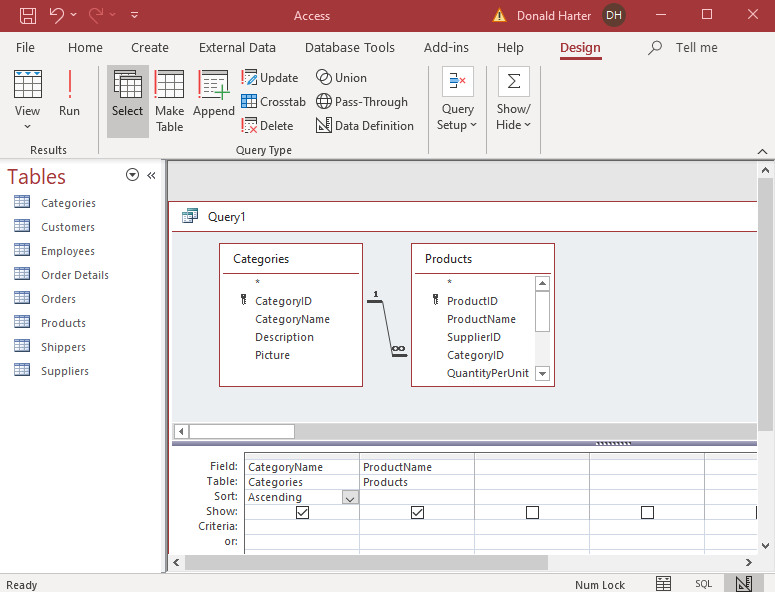
1. To create a query, click on the Create tab, then Query Design.



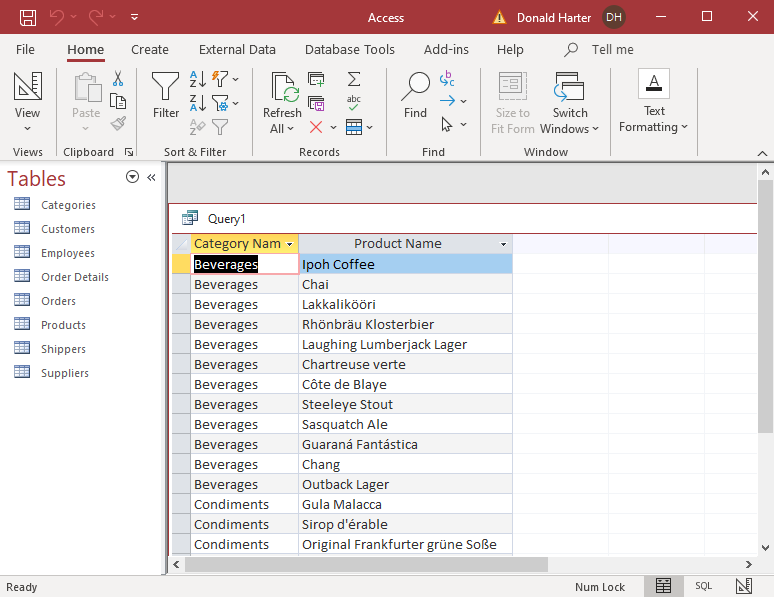
1. Since we want to use the tables Categories and Products, double click on each one.
2. Click on the X to the right of Add Tables to free up some space.



1. At the top of the screen are the two tables. At the bottom is a set of options to set up the query.
2. We want to find all the Product Names for each Category Name. First, double click on Category Name. It should appear at the bottom of the page.
3. Next, double click on Product Name. It should appear next to Category Name.
4. If you want to sort the data, click on sort under Category Name, then ascending.



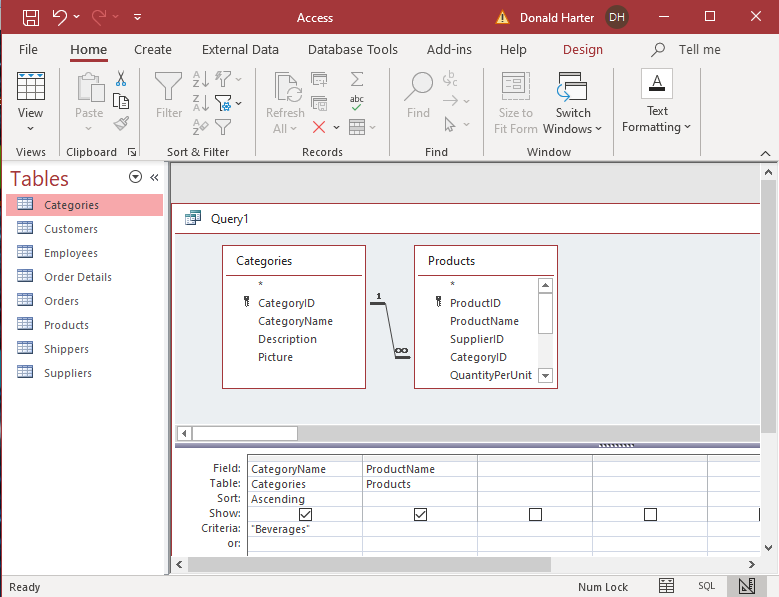
1. To run the query, click Run (see red exclamation point). You should see the following result.



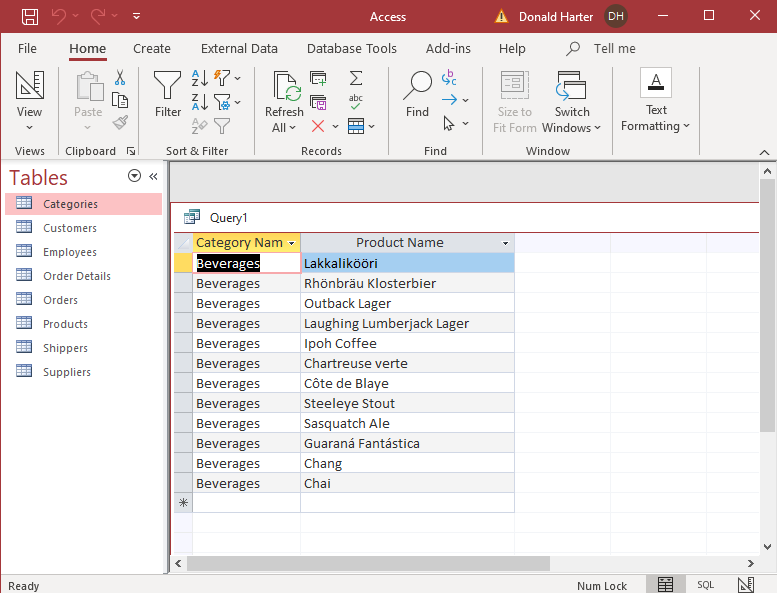
**Filtering or Using Criteria to Restrict the Output**

Just as you filtered data in Excel, you can filter data in Access. If we only want Beverages listed, use the following steps.

1. On the Home tab, click on View, Design View. This takes you back to the design of the query.
2. In the field called Criteria under Category name, enter the word Beverages. Notice that it puts quotation marks around it automatically.



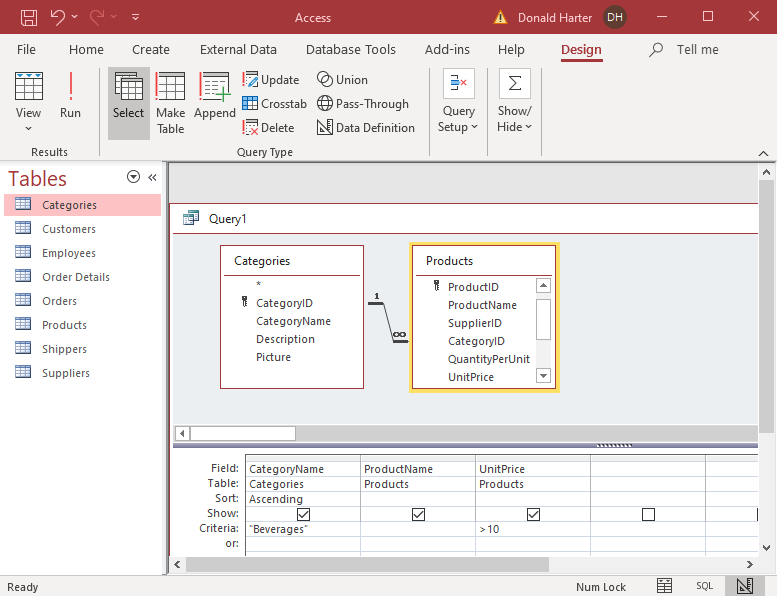
1. Run the query. Only Beverages are displayed.



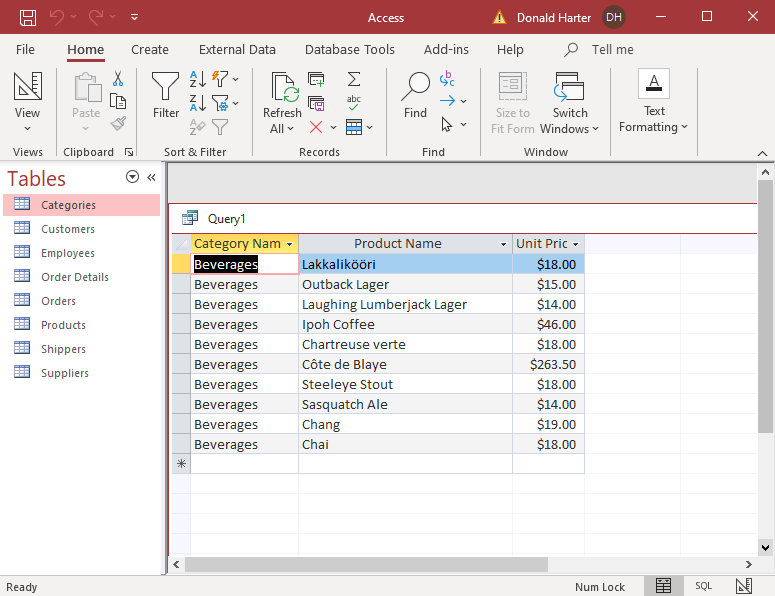
**Queries with Two or More Criteria Using “And”**

You can also run queries when multiple conditions must be true. Let’s find beverages with Unit Price greater than 10.

1. Click on the Home tab, View, Design View.
2. Since we don’t have Unit Price in the query screen, add it by double clicking on Unit Price in Products.
3. In the criteria for Unit Price, enter >10.



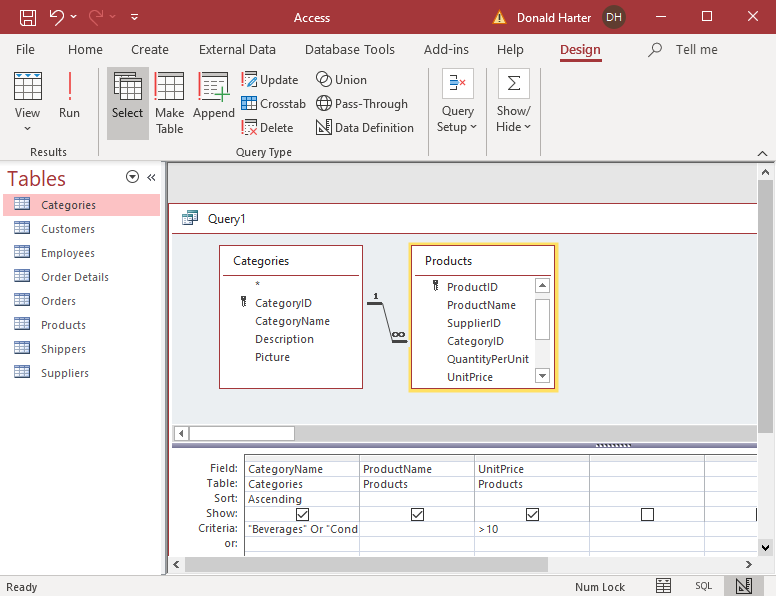
1. Run the query.



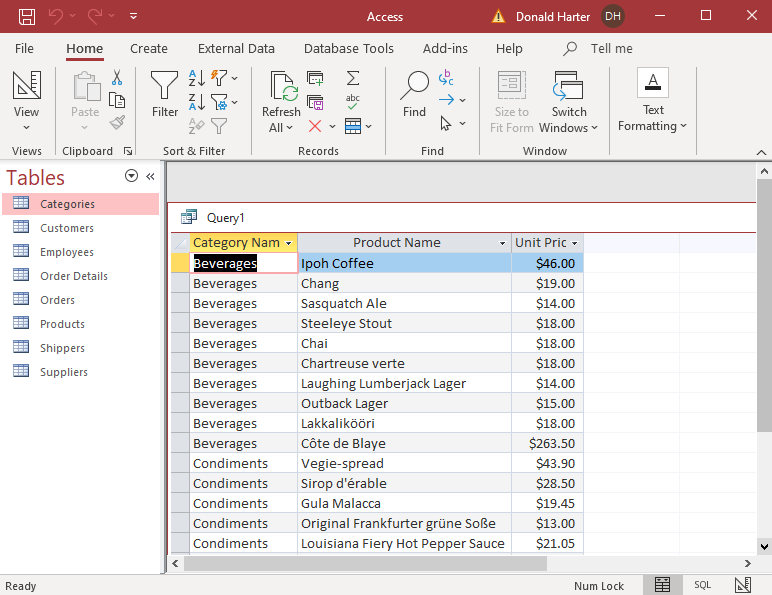
**Query with an “Or” Condition—within One Field**

You can also have an OR condition.

1. For example, if we want Beverages or Condiments, we can put “Beverages” OR “Condiments” in the criteria field for Category Name.
2. This query will return (Beverages OR Condiments) AND (Price>10).



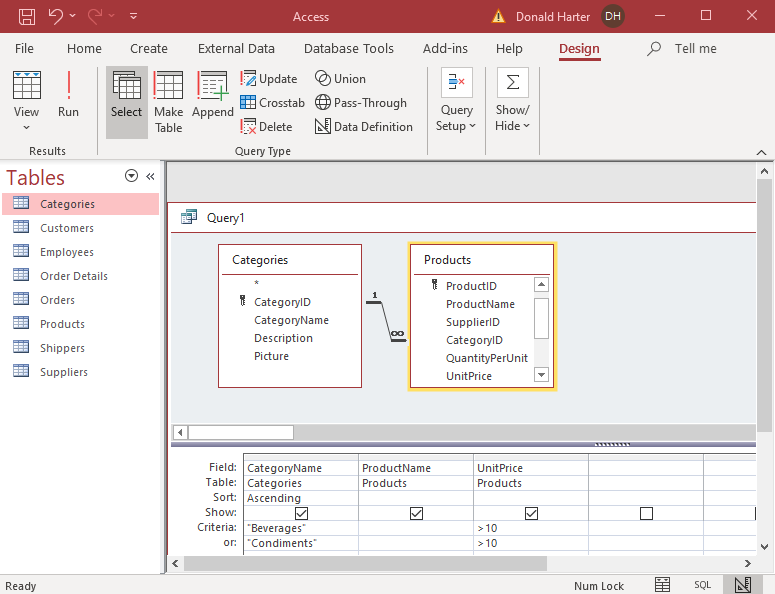
1. The result of running the query is shown below.



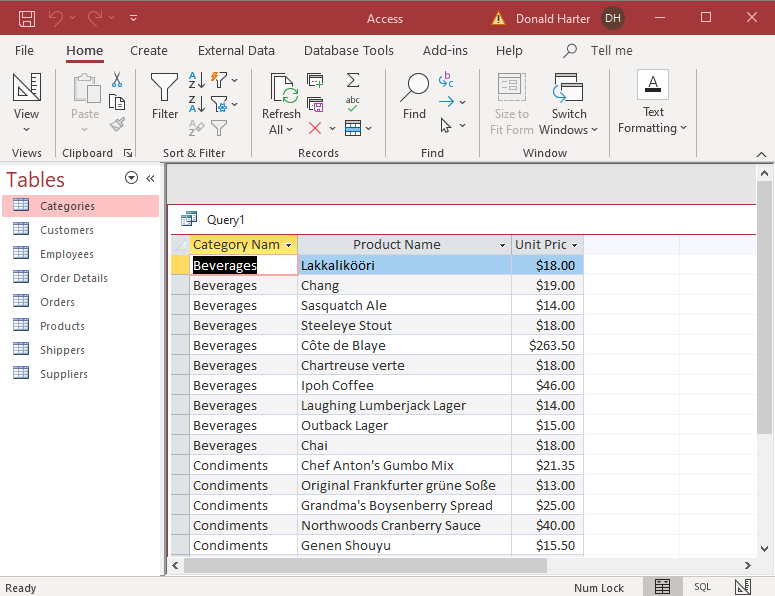
**Query with an “Or” Condition—between Fields**

You can also use multiple criteria rows for an OR condition instead of cramming everything into one cell. Let’s use multiple lines for the previous query. Note that if we also want >10, that >10 has to be on both lines.

1. Enter Beverages as a criteria for CategoryName
2. Enter Condiments as a criteria for CategoryName on the next criteria line (or:)
3. Enter >10 for both lines as a criteria for UnitPrice



1. Click Run !.

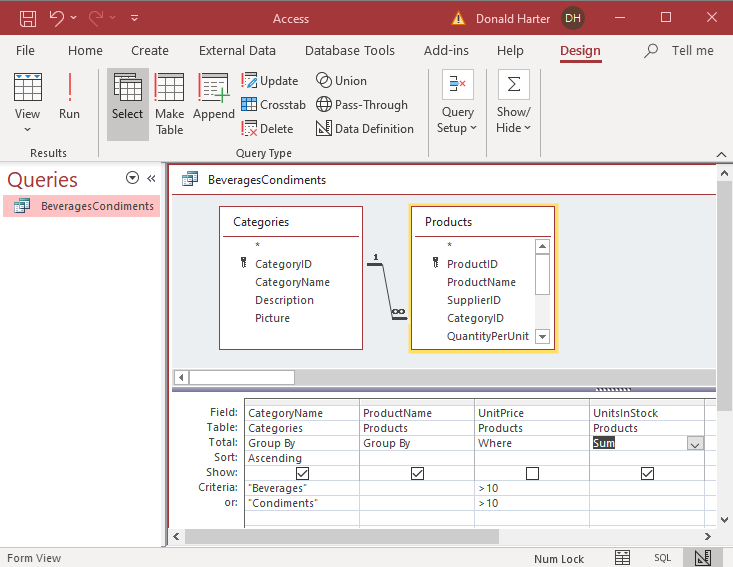


1. What would happen if UnitPrice > 10 were only added to one line?

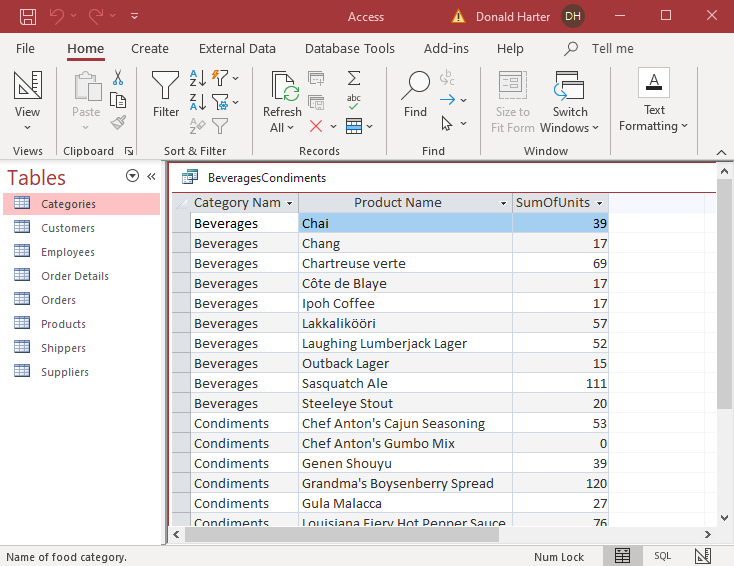
**Calculations**

The calculation of sums, averages, counts, minimum, maximum can also be performed in a query. Let’s sum the total number of units in stock using our current criteria.

1. First, add UnitsInStock from the Products table.
2. Click on the Design tab, then click on the summation symbol ∑.
3. In the Total:, Group by should appear, where it says Group by for each variable.
4. We want to have groupings of Categories and Products, but not UnitPrice. For UnitPrice, use the down arrow on GroupBy and change it to Where. A Where clause is used to filter data; you never group on a where.
5. We don’t want to group on UnitsInStock, instead we want to sum the total. Change GroupBy for UnitsInStock to Sum. You should have the following.



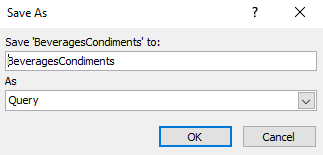
1. Now run the query.



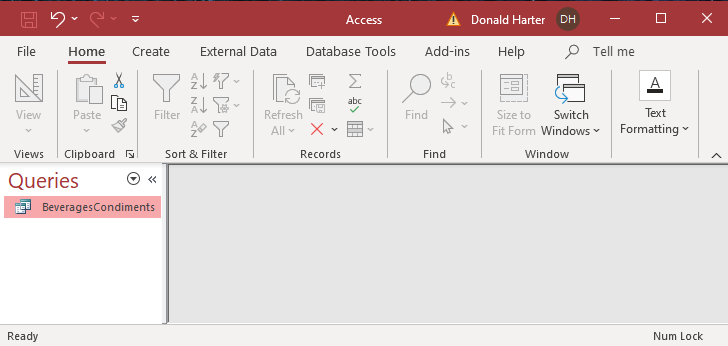
**Saving a Query**

To save a query

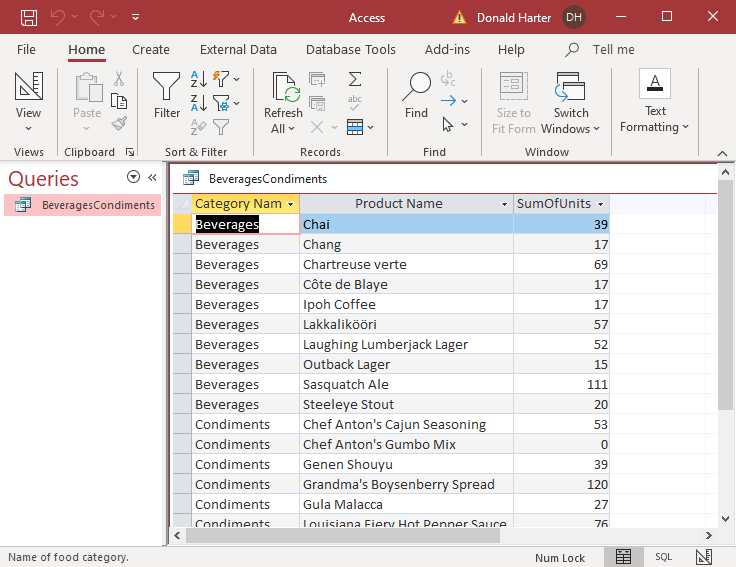
1. Click on File, Save As, Save Object As, then click on the Save As Icon.
2. In the pop-up menu, enter BeveragesCondiments in Save “Query1” to:
3. Click OK.



1. You can retrieve old queries by clicking on the down arrow next to the word Tables, then selecting Queries.



1. Double click on the query to reopen it.



**4.9.1 SQL**

**Recommended Reading**

The best resources for learning SQL are:

1. *Sams Teach Yourself SQL in 10 Minutes* by Ben Forta, 2004
2. *SQL Guide (Quickstudy: Computer)* by Inc. Barcharts, 2005
3. Lynda.com (now LinkedIn Learning)

**Files**

Download the Orders database and the Recruiting database from the course platform.

**SQL Commands for Queries**

The essential SQL commands for queries are:

SELECT identify fields to be reported

AS creates an alias for reporting labels

FROM identify tables to be used

WHERE filter criteria

GROUP BY clustering criteria

HAVING additional criteria based on a calculation

ORDER BY sorting criteria

IN part of a list

BETWEEN range of numbers

IS NULL empty values

LIKE matches string of characters

**Order of SQL commands**

When building SQL commands, the order of words is important. In English, the order is generally:

Noun => Verb => Object

In SQL the order is:

SELECT

FROM

WHERE

GROUP BY

HAVING

ORDER BY

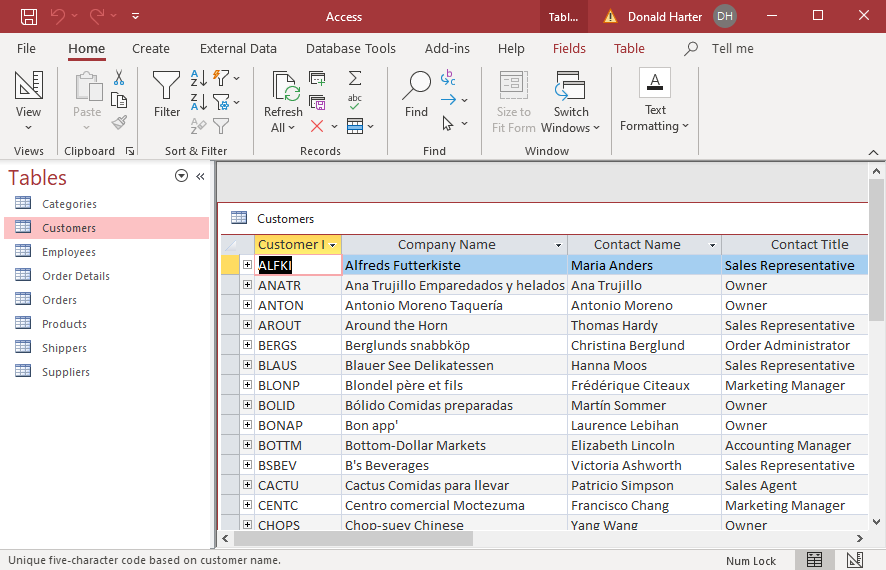
Failure to follow this order will result in a syntax error.

In the following examples, we will enter the commands in uppercase letters, only to make them stand out. You do not need to use uppercase, but it makes the SQL easier to read. Also, each command will be entered on a separate line, only to make it easier to read.

All SQL commands end with a semicolon (;).

**4.9.2 Viewing the Database**

1. Open the database Orders.accdb.
2. Click Enable, then Yes for trust.
3. Once the list of data base tables appears (customers, orders, order details, etc.), double click on customers to open the Customers table.
4. After viewing the contents of the Customers table, close it by clicking on the X in the upper right corner of the customers table screen.
5. Similarly, open the Orders and Order Details tables.

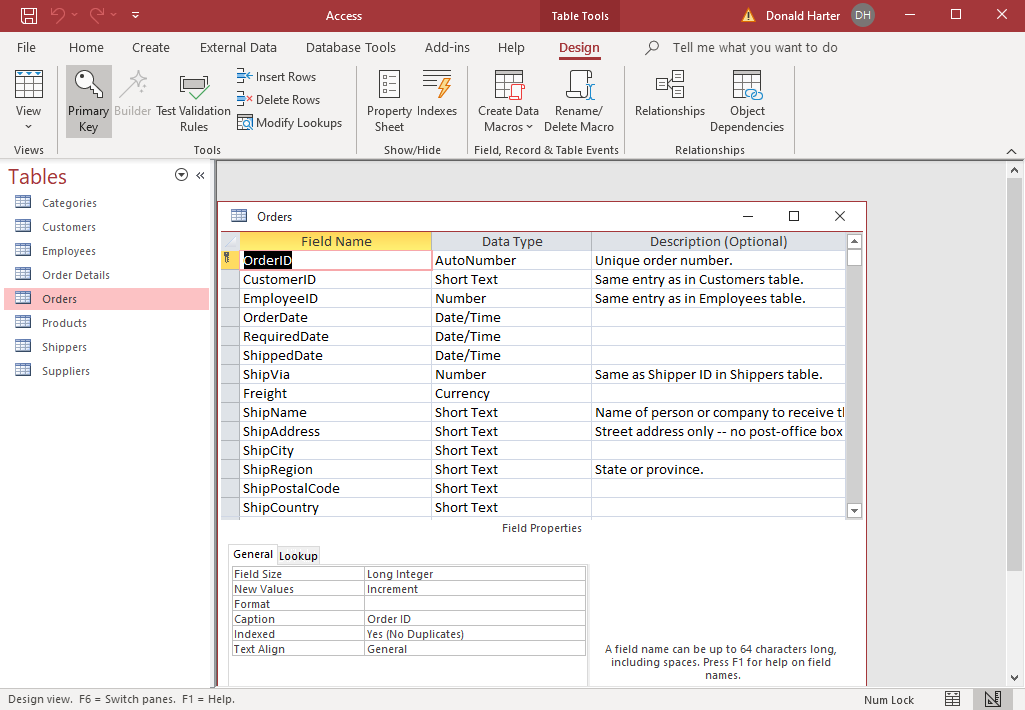


1. Terminology
   1. Table: set of data in columns and rows
   2. Field: column
   3. Record: row

**Viewing the Design of a Table**

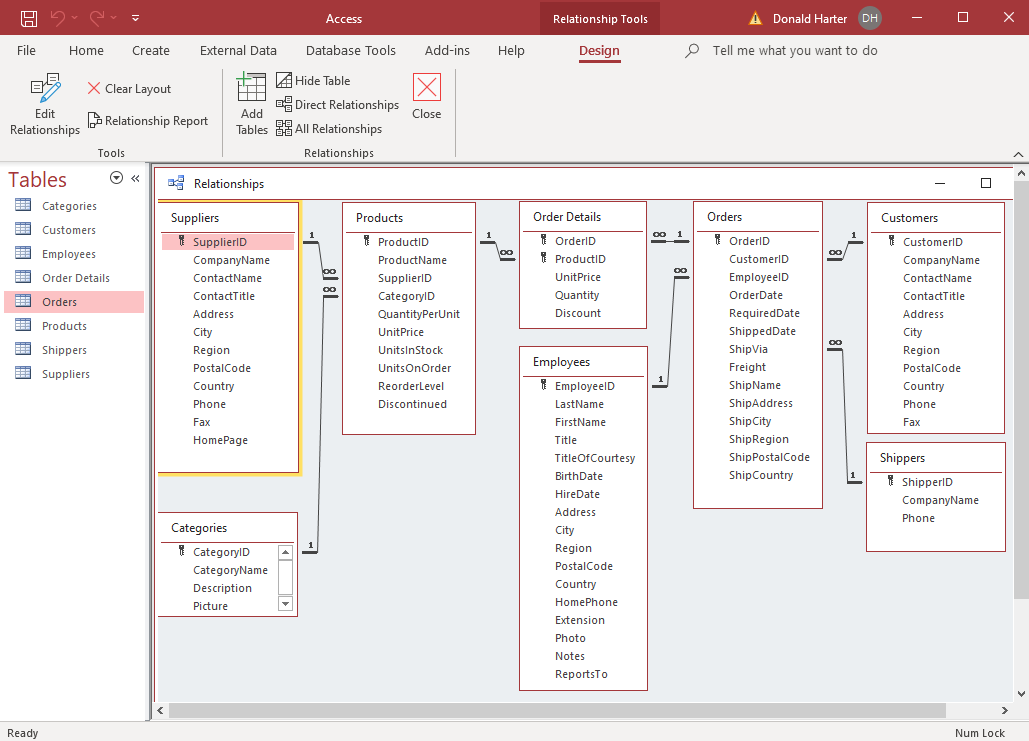
It is important to understand how data is store in the database. Data can be stored as text or numbers, in a variety of formats. To view the design of the Orders table:

1. Double click on the Orders table to open it.
2. Click on View > Design View.
3. A data field can have one of several data types:
   1. Autonumber: automatically increments when you add new data
   2. Text: alphanumeric data
   3. Number: numeric data
   4. Date/Time: Year, month, day, time
4. Note that the Field Name “OrderDate” has no space between the words, but the Caption is “Order Date.” The field name is used in SQL, but the caption will appear as the heading in a table.
5. To return to the Datasheet View, click on View > Datasheet View.
6. Close the table after viewing the Data Types.

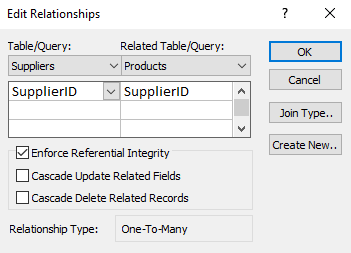
****

**Viewing the Structure of the Database**

1. Now click on Database Tools, then Relationships.
2. You should see all of the tables, data elements in the tables, key fields, and links between tables. You will need to refer to this during this lab.
3. A key field uniquely identifies each record in the table. Key fields have “keys” next to them.

****

1. Double click on the relationship line between Suppliers and Products.
   1. Click on Join Type to confirm that this is an inner join.
   2. What other types of join are there? Click cancel to return.
   3. What is the Relationship Type? Click Cancel to close the Edit Relationship screen, and cancel again.

****

1. If you want close the relationships, click on X to close the relationships screen. In this case, keep the relationship screen open for reference.

**4.9.3 SQL: Selecting Data with SELECT and FROM**

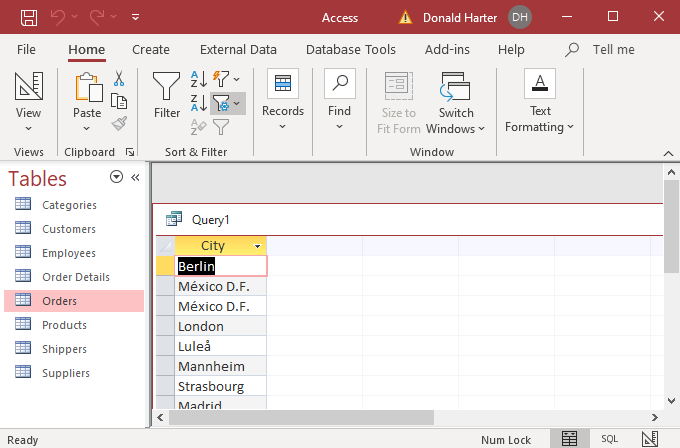
First retrieve all the cities where we have customers. Returning to the Relationships view, which table has customer information? The table is Customers. What is the name of the field (column) that holds city information? Field is City.

1. Returning to the database screen (you should see the list of tables again); click on Create, Query Design.
2. At the top left of the screen, click on SQL View, then SQL View again. Notice that you can toggle between SQL and Design views.
3. You should see “SELECT;”. Delete this so we can start from scratch.
4. Type in the following SQL command. Notice that it ends with a semicolon.

SELECT City

FROM Customers;

1. At the top of the screen, click on View, Datasheet View. (You could also click on Run !.)
2. This query will select all city names from the customer table. Notice that Mexico D.F. and London appear multiple times.

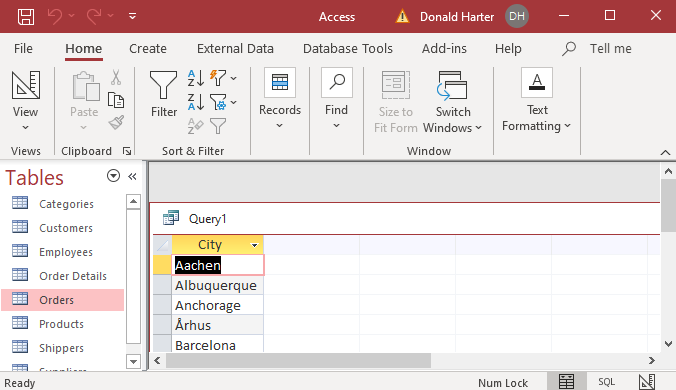


1. To eliminate duplicates, click on View, SQL, now use the following query.

SELECT DISTINCT City

FROM Customers;

1. Click Run !.
2. Duplicates have now been eliminated.



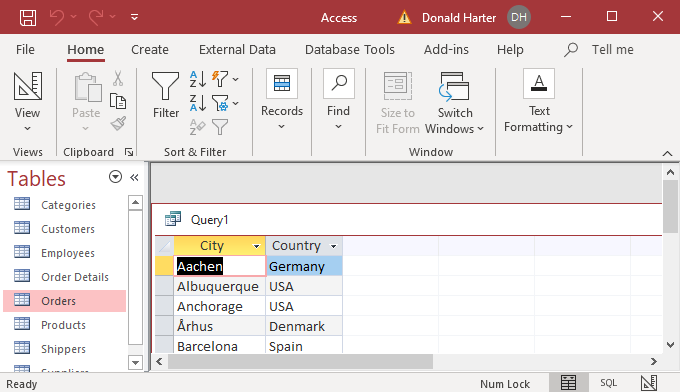
**Query Retrieving Two or More Columns**

1. To retrieve more than one field of data, put commas between the data field names.
2. Retrieve City and Country from the Customers table.

SELECT DISTINCT City, Country

FROM Customers;

1. Click Run !.



**Retrieving All Columns from a Table**

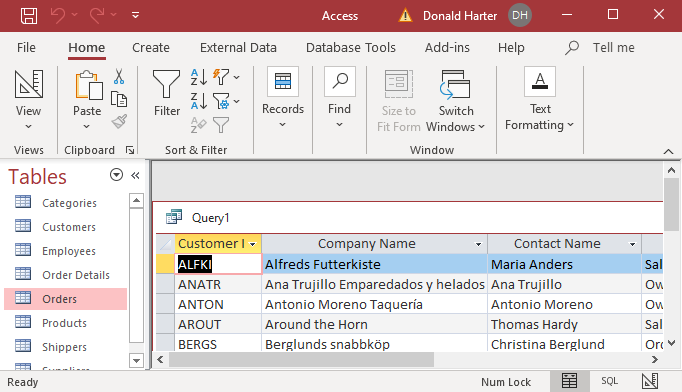
Sometimes you might want to display everything in a table. Use the wildcard character \* for this.

1. To retrieve all columns in the Customers table, enter the SQL:

SELECT \*

FROM Customers

1. Click Run !.



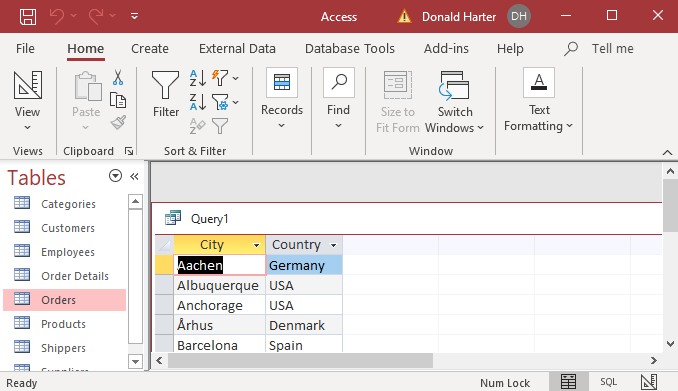
**4.9.4 SQL: Sorting Using ORDER BY**

1. To sort the output, use the ORDER BY command. In this first example, display City and Country from the Customers table, sorting by City.

SELECT City, Country

FROM Customers

ORDER BY City;

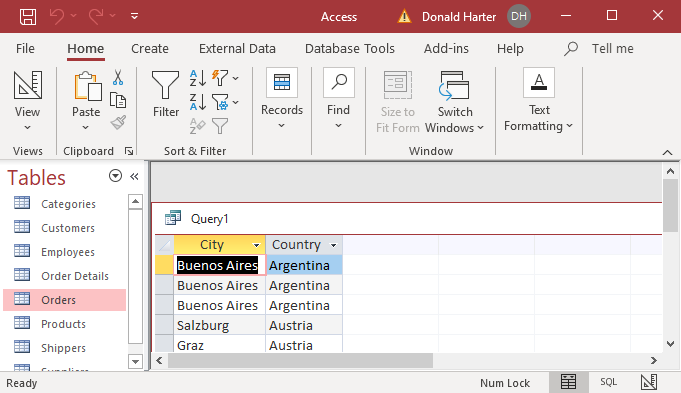


1. Now, order by Country.
2. Click on View, SQL to return to the SQL screen.

SELECT City, Country

FROM Customers

ORDER BY Country;

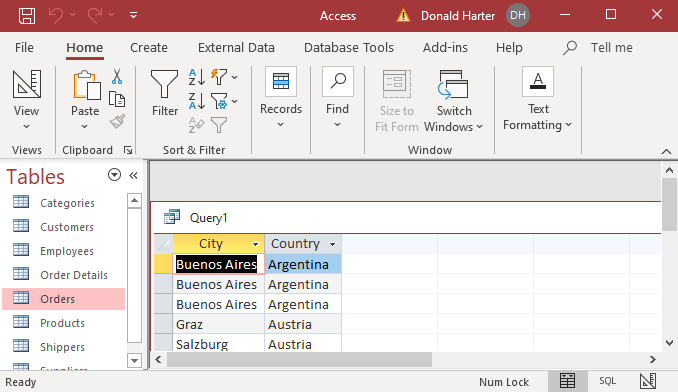


1. Click on View, SQL to return to the SQL screen.
2. You can have multiple sort fields, where the first field is sorted first, the second field is second, etc. In this example, sort by Country first, then City.

SELECT City, Country

FROM Customers

ORDER BY Country, City;



1. Alternatively, you can use the position order to specify the sort order.

SELECT City, Country

FROM Customers

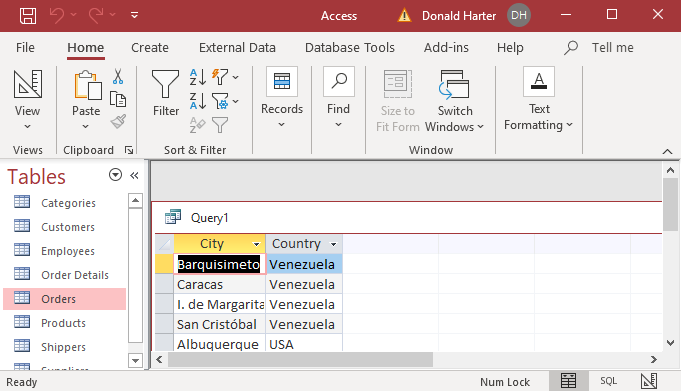
ORDER BY 2, 1;

1. You can reverse the order of the sort by using the DESC (descending) command after the appropriate sort term. In this example, use a descending sort on Country, but keep the default ascending sort for City (no command necessary).

SELECT City, Country

FROM Customers

ORDER BY Country DESC, City;

****

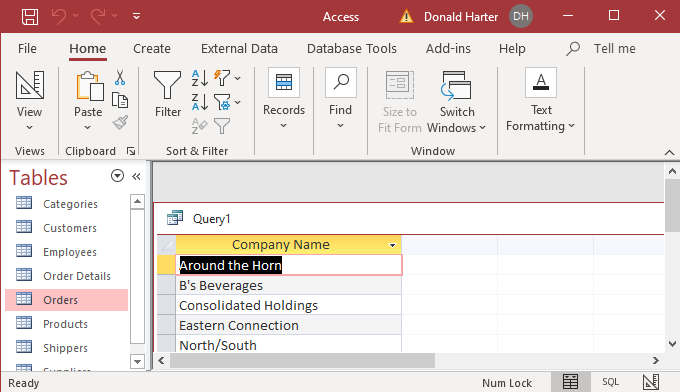
**4.9.5 SQL: Criteria Using WHERE, Boolean Operators, IN, LIKE**

1. Now identify all customer companies in London. Use the following command. Notice that the column names (e.g., CompanyName) has no space in it, and that London is in double quotes. In a Where clause, number filters do not use quotes, text fields use quotes.

SELECT CompanyName

FROM Customers

WHERE City="London";



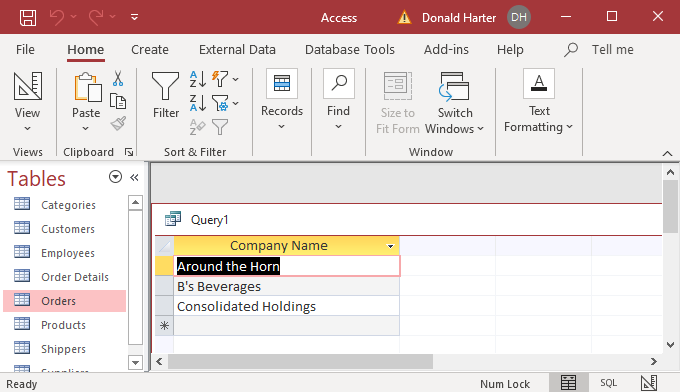
**Filters Using Where Clauses with Boolean Operators**

1. Now select CompanyName where the City is London and the ContactTitle is Sales Representative. Add quotation marks around Sales Representative. Enter the Boolean operator AND between the criteria.
2. Why is there no space in CompanyName and ContactTitle, but there is a space in Sales Representative?

SELECT CompanyName

FROM Customers

WHERE City="London" AND ContactTitle="Sales Representative";

****

**Selecting All Data Using Where Clauses with Boolean Operators**

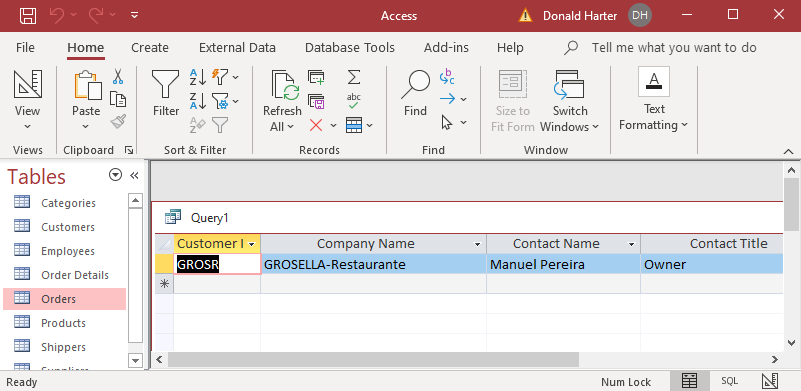
What if you wanted to find out everything about each company where the company contact was the owner and the company was in Caracas?

1. Recall that SELECT \* will select all fields from a table.
2. You can use Boolean operators to filter on several columns at once:

SELECT \*

FROM Customers

WHERE ContactTitle="Owner" AND City="Caracas";

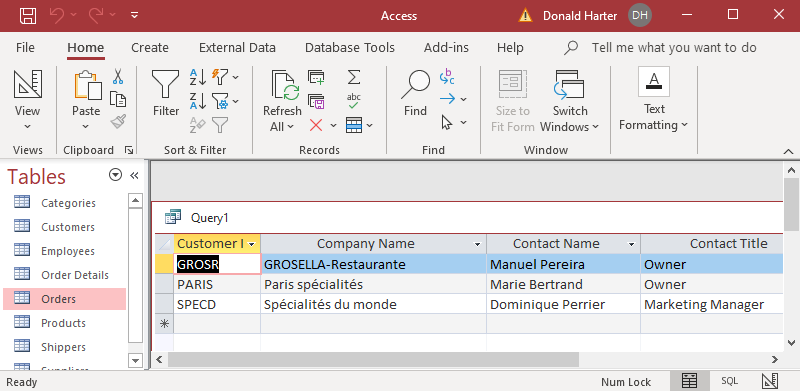


1. Now find Owners in Caracas or Paris.

SELECT \*

FROM Customers

WHERE ContactTitle="Owner" AND City="Caracas" OR City="Paris";

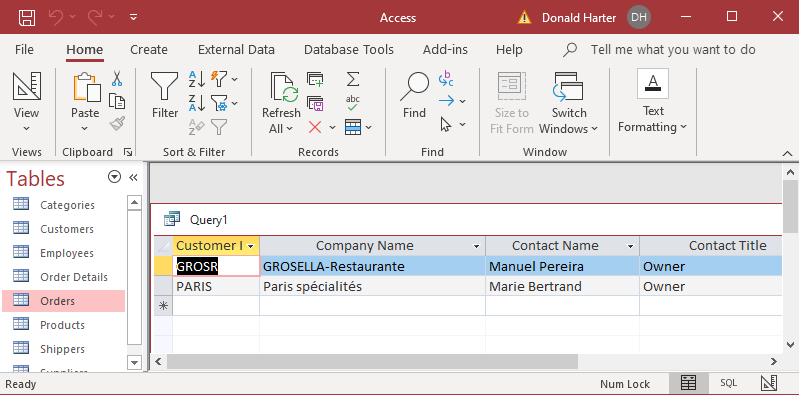


1. What went wrong? The order of operation of AND and OR conditions executes ANDs before ORs. To group the conditions properly, you must put parentheses around the conditions. SQL executes within the parentheses first, then works outward.

SELECT \*

FROM Customers

WHERE ContactTitle="Owner" AND (City="Caracas" OR City="Paris");



**Filters Using Numbers**

1. To retrieve records using a filter with conditions on numbers, use the following operators:

= equal

<> not equal

!= not equal (*not available in Access*)

< less than

!< not less than (*not available in Access*)

<= less than or equal

> greater than

!> not greater than (*not available in Access*)

>= greater than or equal

BETWEEN between two numbers (e.g., BETWEEN 10 AND 20)

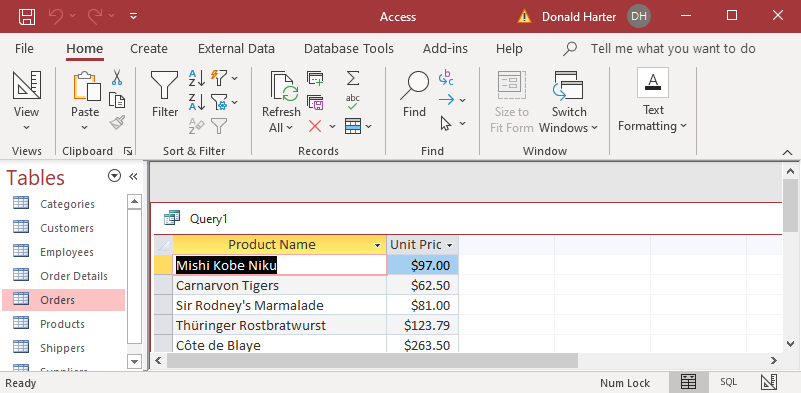
IS NULL is a null or no value

1. For this example, retrieve ProductName and UnitPrice from the Products table where a UnitPrice greater than 50.

SELECT ProductName, UnitPrice

FROM Products

WHERE UnitPrice>50;

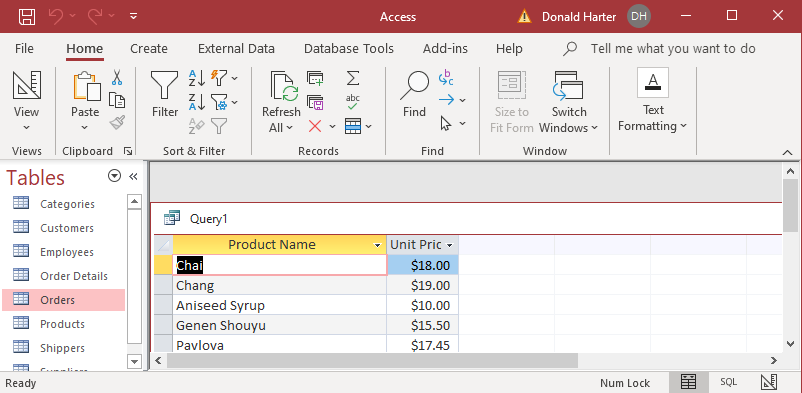


1. Now retrieve ProductName with UnitPrice BETWEEN 10 AND 20.

SELECT ProductName, UnitPrice

FROM Products

WHERE UnitPrice BETWEEN 10 AND 20;

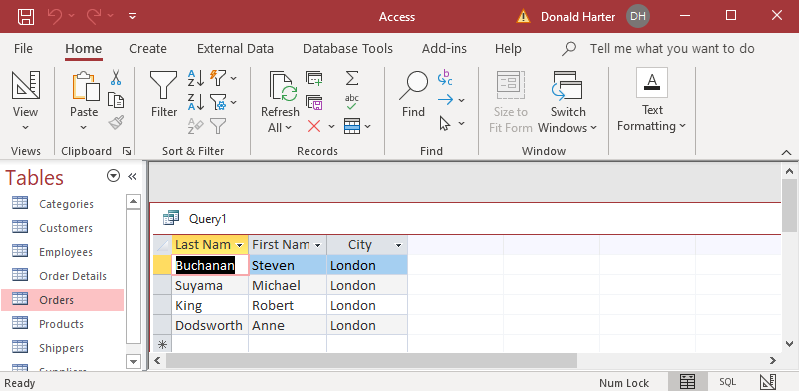
****

1. The Employees table has missing data in the Region field. Why? Retrieve the LastName and FirstName of Employees, and the City of Employees, where the Region is null.

SELECT LastName, FirstName, City

FROM Employees

WHERE Region IS NULL;



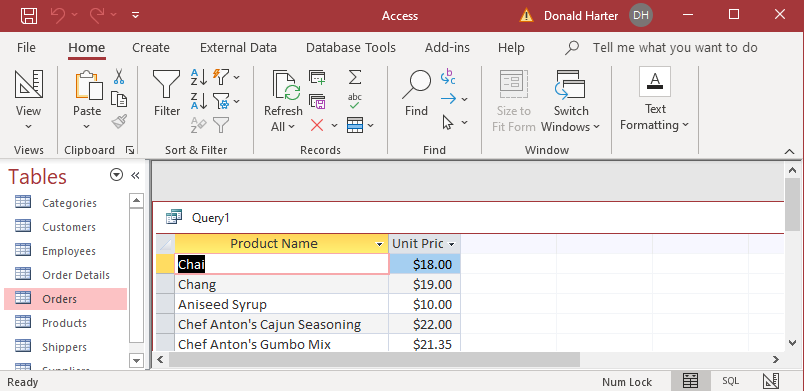
**NOT Operator**

1. The NOT operator can be placed in a where clause to negate the condition; in this case NOT > 50 means <= 50.

SELECT ProductName, UnitPrice

FROM Products

WHERE NOT UnitPrice>50;

****

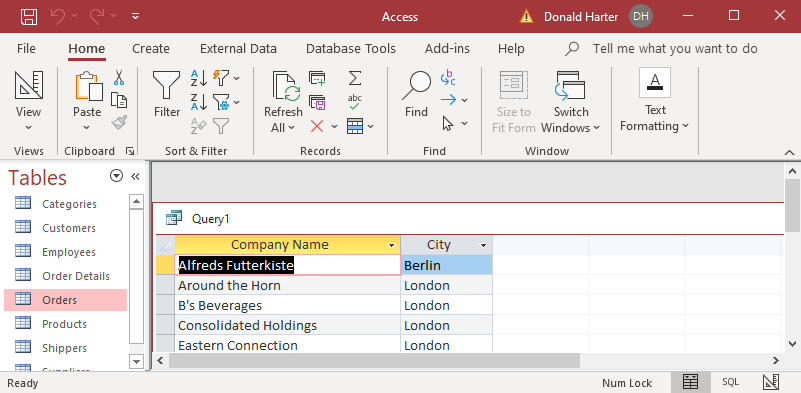
**Additional Boolean Operators and Uses of the IN List Operator**

1. Retrieve all the names of companies in London or Berlin using the OR operator.

SELECT CompanyName, City

FROM Customers

WHERE City="London" OR City="Berlin";



1. Now do the same query using the IN operator. The list of cities is enclosed in parentheses, separated by commas). You should see the same result as above.

SELECT CompanyName, City

FROM Customers

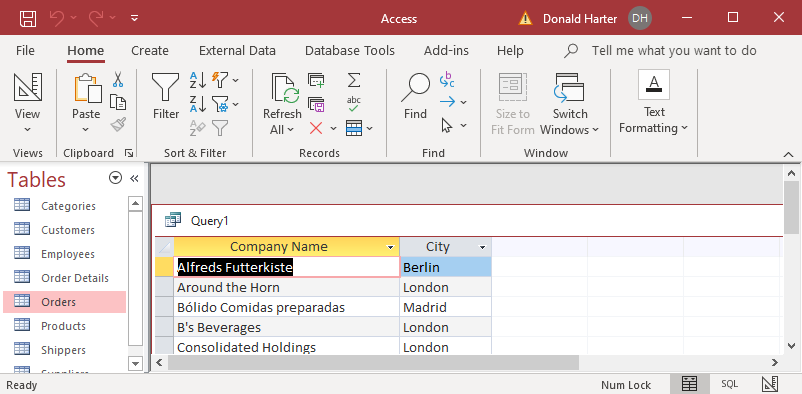
WHERE City IN ("London","Berlin");

1. Notice it’s much easier to add to a list when using the IN list option.

SELECT CompanyName, City

FROM Customers

WHERE City IN ("London","Berlin","Madrid");



**LIKE Command: Matching Part of a String of Characters**

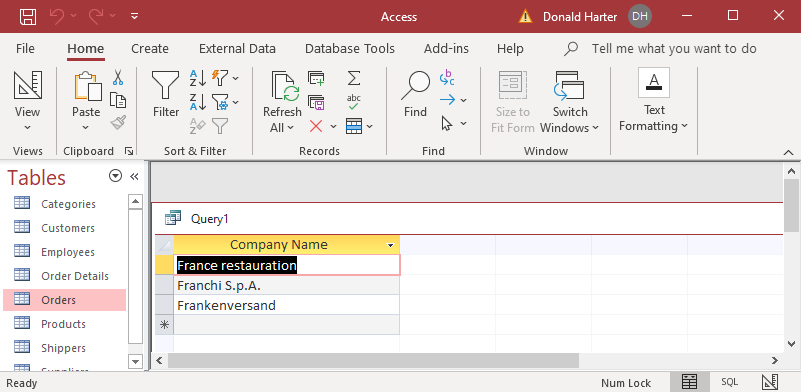
The LIKE command in a where clause matches a group of characters to the data in a field.

1. Retrieve all the names of companies that start with the letters “Fr”; the wildcard character \* means that any character can follow Fr. (Note: Other DBMS software use % as the wildcard character.)

SELECT CompanyName

FROM Customers

WHERE CompanyName LIKE "Fr\*";

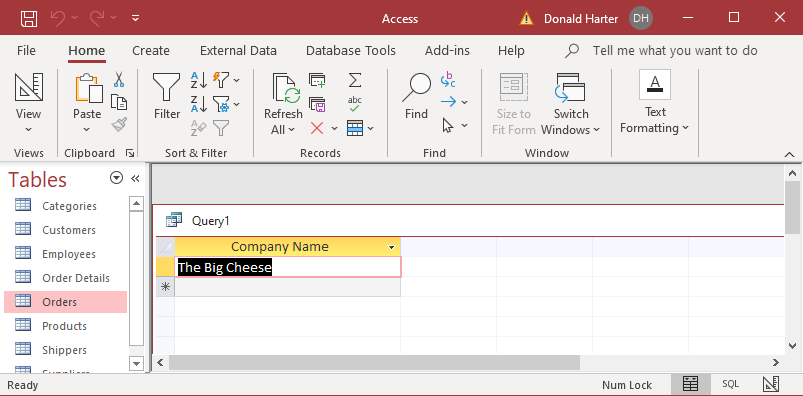


1. Now retrieve the names of companies that have Big in the name. Since we don’t know if Big is at the beginning, middle, or end, add a wildcard character \* before and after Big.

SELECT CompanyName

FROM Customers

WHERE CompanyName LIKE "\*Big\*";



**WHERE Clauses with Date Functions**

Let’s retrieve names of companies with a specific order date (company name is in the Customers table, orders in the Orders table).

1. Retrieve all the names of companies with an order on July 5, 1995 using a join (this only works with the United States date format).
2. Since we are retrieving from two tables (Customers and Orders), we need to include them in the FROM clause.
3. The WHERE clause must include the join for the two tables and the date value that we want.

SELECT CompanyName

FROM Customers, ORDERS

WHERE Customers.CustomerID=Orders.CustomerID

AND OrderDate=#7/5/95#;



**Data Ranges Using BETWEEN**

When you want to retrieve data that is in a range, such as dates, you can use the BETWEEN command.

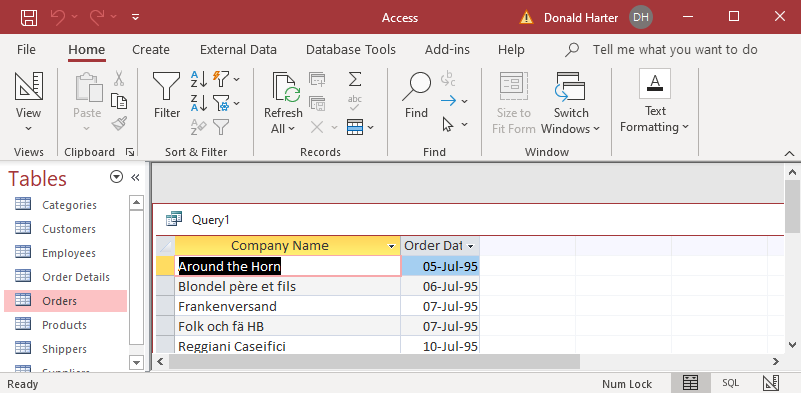
1. Retrieve all the names of companies with an order between July 5 and July 12, 1995.

SELECT CompanyName, OrderDate

FROM Customers, Orders

WHERE Customers.CustomerID=Orders.CustomerID

AND OrderDate BETWEEN #7/5/95# AND #7/12/95#;



**4.9.6 SQL: Mathematical Operations**

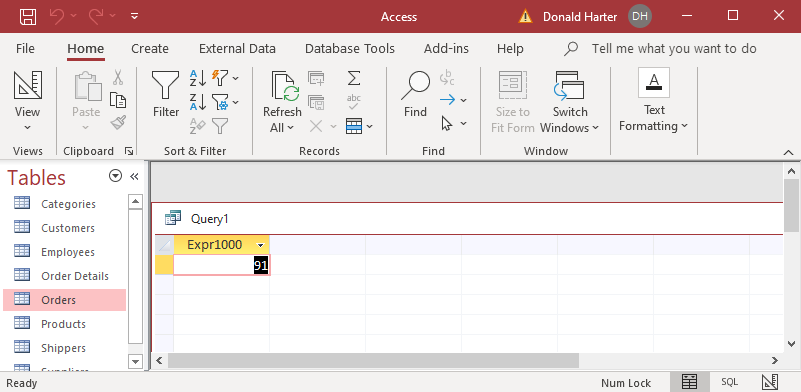
Next we will use some mathematical operations. These include count, sum, and avg (average). To use a mathematical operation, type the math expression, then (field). If you want to count the number of rows in a table, use count(\*).

1. First let’s count the number of records in the Customers table.

SELECT count(\*)

FROM Customers;

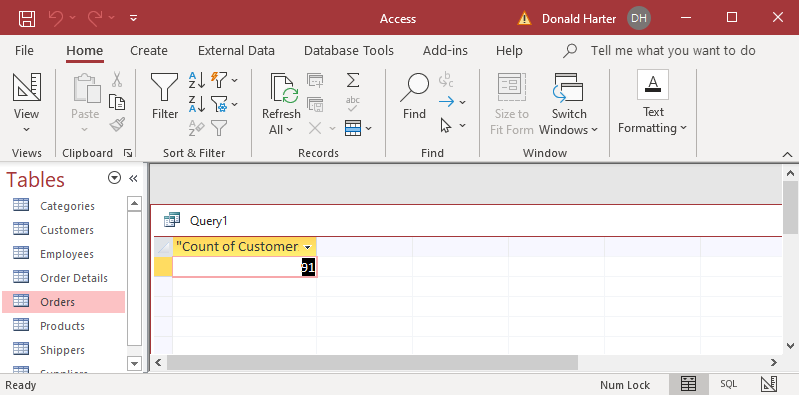
1. You should have counted 91. You can check this by looking at the contents of the table again. At the bottom of the screen, it tells you the number of records.



1. The “Expr1000” is not very informative. Let’s create a new label using the AS clause.

SELECT count(\*) AS "Count of Customers"

FROM Customers;



**4.9.7 SQL: Grouping Using GROUP BY**

You can also perform math operations on subsets of the table and report them separately. Summarization functions include:

AVG average of column values

COUNT number of records

MIN minimum of column values

MAX maximum of column values

SUM sum of column values

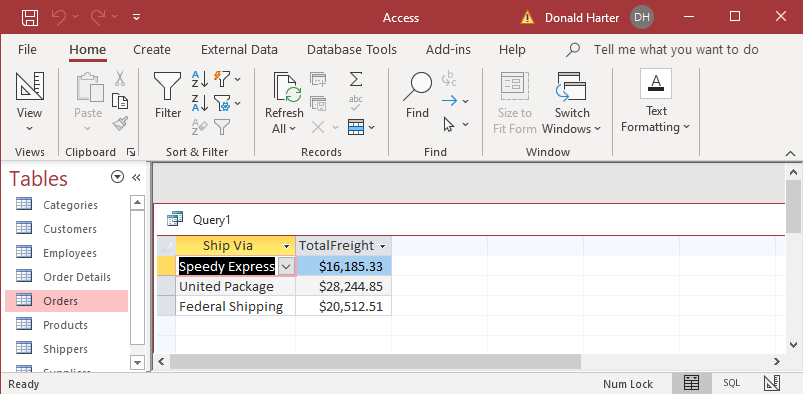
1. For example, next look at the orders table and calculate how much each shipper collected in shipping charges.

SELECT ShipVia, SUM(freight) AS TotalFreightCost

FROM Orders

GROUP by ShipVia;

1. You should have seen three shippers and the total freight charges. This one used the group by command, which basically organizes the data by “ShipVia” then calculates the sum. Make sure that you understand this one before you continue.



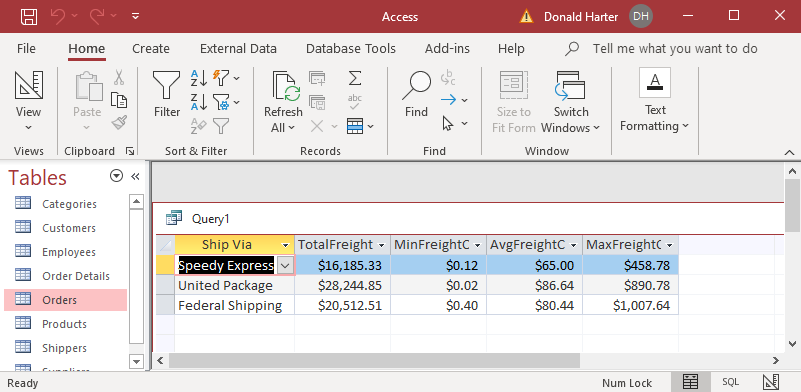
1. Now include the minimum, average, and maximum as separate columns. Note that the commands can be on multiple lines.

SELECT ShipVia, SUM(freight) AS TotalFreightCost, MIN(freight) AS MinFreightCost,

AVG(freight) AS AvgFreightCost, MAX(freight) AS MaxFreightCost

FROM Orders

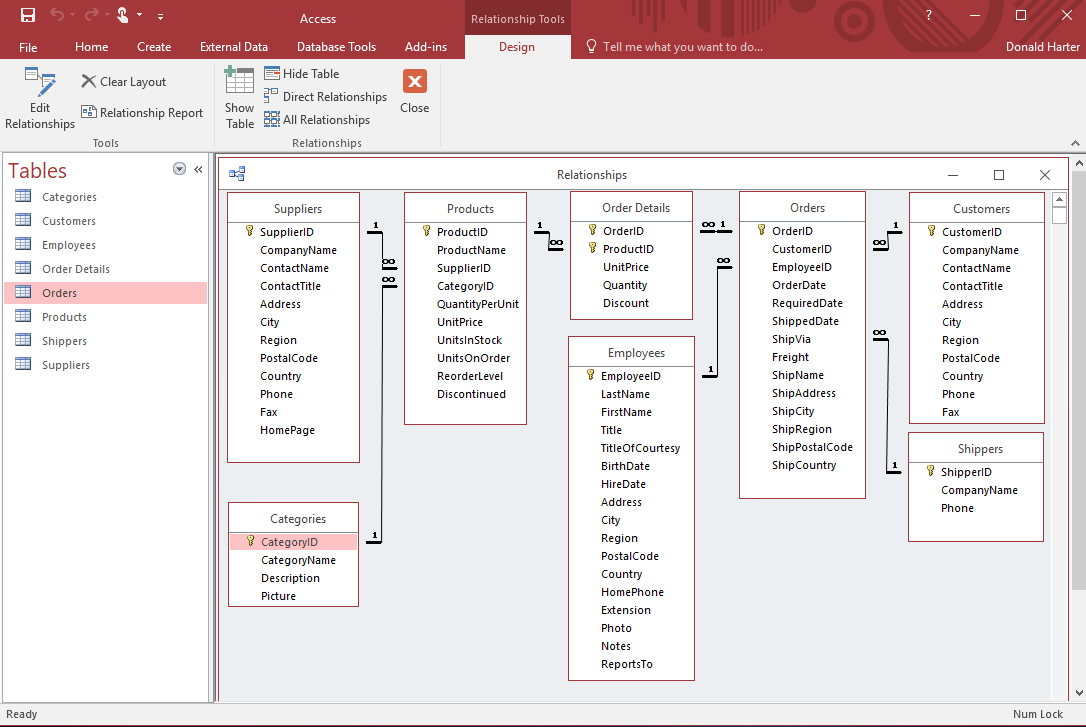
GROUP BY ShipVia;



**4.9.8 SQL: Queries on Multiple Tables**

Now the tricky one. What if you want information that is contained in more than one table? To accomplish that you need to use a join.

1. When using more than one table, each field (data column) must be identified by the table name and column name. To do this, use Table.Field. For example, to select CompanyName from the Customers table, type Customers.CompanyName
2. To retrieve data from two tables, you need to use a join command. This is just a Where clause that includes the two fields that must match to bring the tables together. Refer to the Relationships chart below.

****

1. Create a query to find out how much each customer in London paid in freight charges. Note that CompanyName and City are in the Customers table, but freight is in the Orders table.
2. Since we are using two tables, both table names must be in the From line, separated by commas.
3. The WHERE clause must identify the City as London, but also link the two tables using the common key (Customers.customerID=Orders.customerID).
4. The GROUP BY simply says that we should calculate the sum of freight charges after we have grouped the companies together.

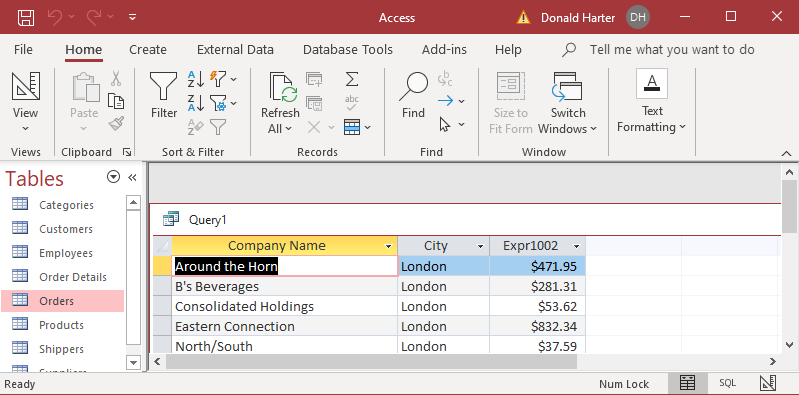
SELECT Customers.CompanyName, Customers.City, sum(Orders.freight)

FROM Customers, Orders

WHERE Customers.City="London"

AND Customers.customerID=Orders.customerID

GROUP BY Customers.City, Customers.CompanyName;



**4.9.9 HAVING Command and Table Aliases**

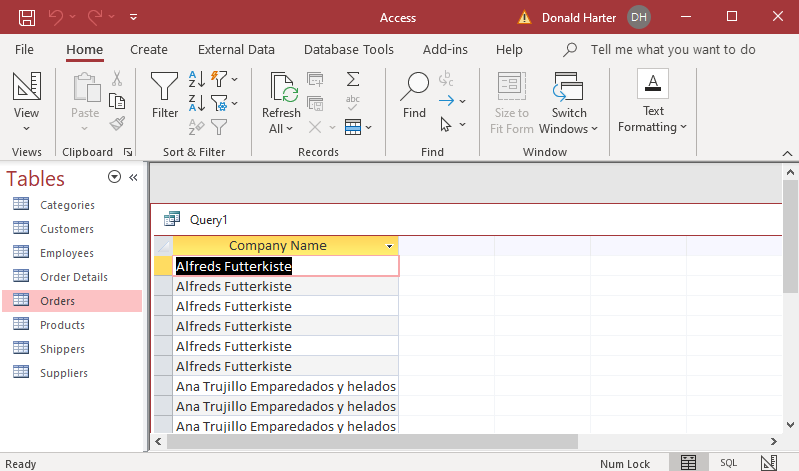
Pretty easy, isn’t it? Now try a tough one. Retrieve all company names with more than 20 orders. Use the option of a table alias to shorten the command.

1. A table alias is another name for a table that allows you to shorten your commands; in the FROM clause, entering Customers C means that to refer to table Customers, you can now abbreviate it to C.

SELECT CompanyName

FROM Customers C, Orders O

WHERE C.CustomerID=O.CustomerID;



1. The HAVING clause combines a WHERE clause and a calculation. Use the HAVING clause to restrict the data displayed to only companies with more than 20 orders. We use the same SQL as the above example, adding a GROUP BY and HAVING clause.

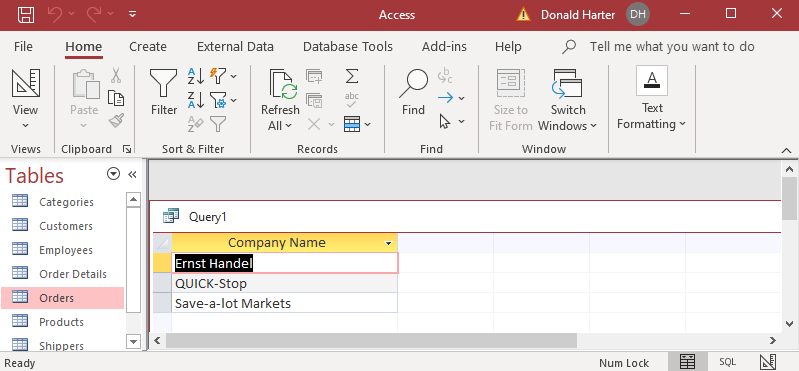
SELECT CompanyName

FROM Customers C, Orders O

WHERE C.CustomerID=O.CustomerID

GROUP BY C.CompanyName

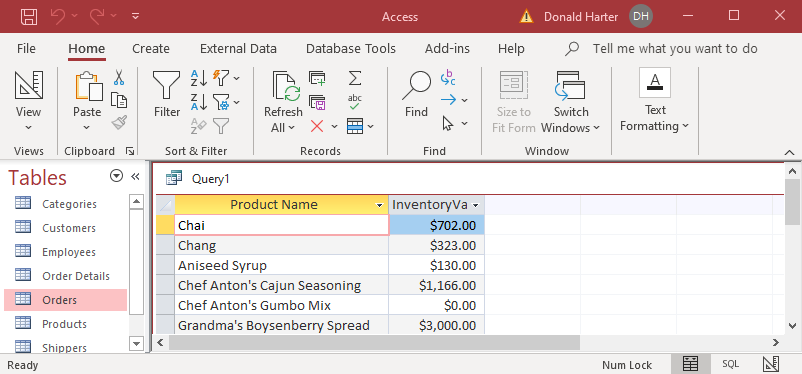
HAVING count(\*)>20;



1. The HAVING clause can also allow you to find a product inventory value over a given threshold. In this example, find ProductName with total inventory value over 3000.
2. Inventory value is not in the database, but UnitPrice and UnitsInStock is in the Products table; we can multiply these together.
3. First, list Product Name (Products table) and their InventoryValue (UnitPrice\*UnitsInStock).

SELECT ProductName, UnitPrice\*UnitsInStock AS InventoryValue

FROM Products;



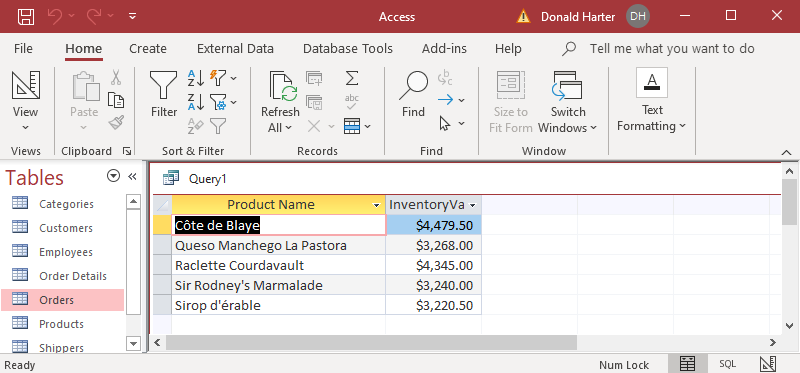
1. In order to use HAVING, we first need to GROUP BY. The GROUP BY will list a line for each GROUP BY field. The InventoryValue will need to be represented as a sum.

SELECT ProductName, SUM(UnitPrice\*UnitsInStock) AS InventoryValue

FROM Products

GROUP BY ProductName

HAVING SUM(UnitPrice\*UnitsInStock)>3000;



**4.9.10 SQL: Subqueries with Lists and the IN Command**

A subquery is a query embedded within a query. The IN command looks for items in a list. The SELECT within the parentheses is executed first, then work outward toward the outer SELECT. The IN option matches items in a list.

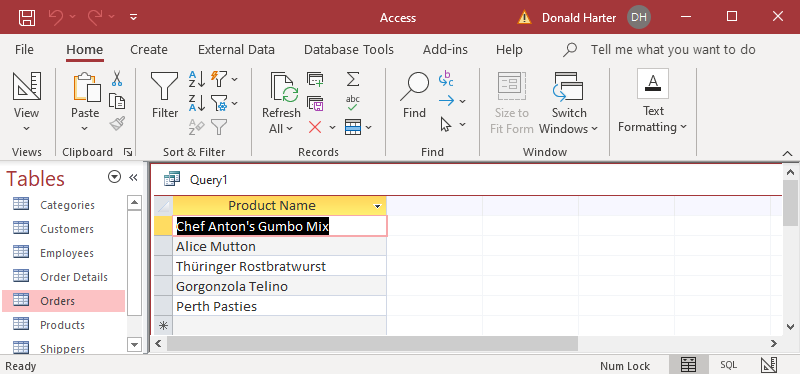
Now, retrieve companies that supply products which are out of stock, i.e., UnitsInStock=0.

1. First create a query to find all Products within the Beverages Category.

SELECT ProductName

FROM Products

WHERE UnitsInStock=0;



1. Then find all Company Names from the Suppliers table that have a Product in this list.

SELECT CompanyName, ProductName

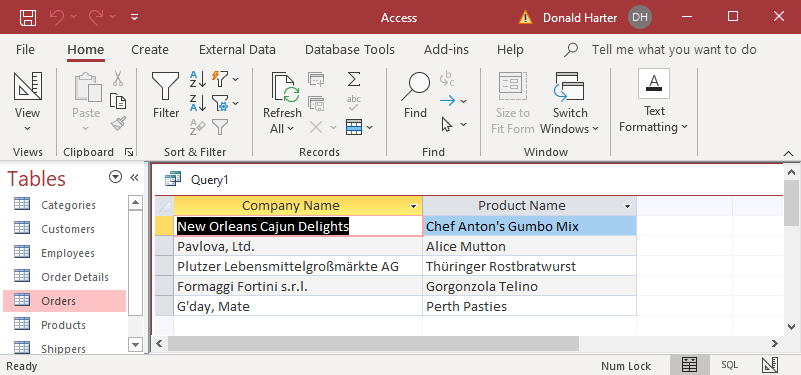
FROM Suppliers, Products

WHERE Suppliers.SupplierID=Products.SupplierID AND ProductName IN

(SELECT ProductName

FROM Products

WHERE UnitsInStock=0);

****

**4.9.11 SQL Limiting Results**

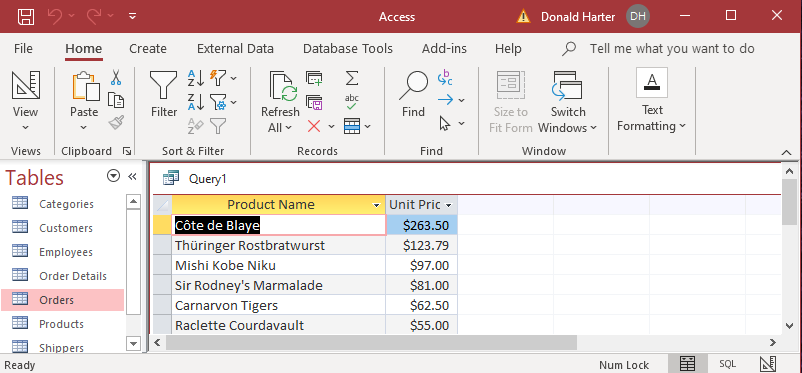
You sometimes will want to list the top items of a list. For example, you might want to know the most expensive products, without listing all products.

1. First, list all products by ProductName and UnitPrice, sorting on UnitPrice from high to low (DESC).

SELECT ProductName, UnitPrice

FROM Products

ORDER BY UnitPrice DESC;

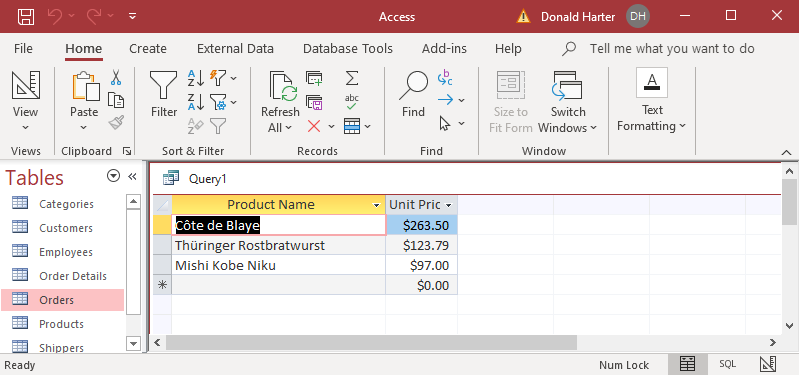


1. To select only the top three, use the TOP option.

SELECT TOP 3 ProductName, UnitPrice

FROM Products

ORDER BY UnitPrice DESC;

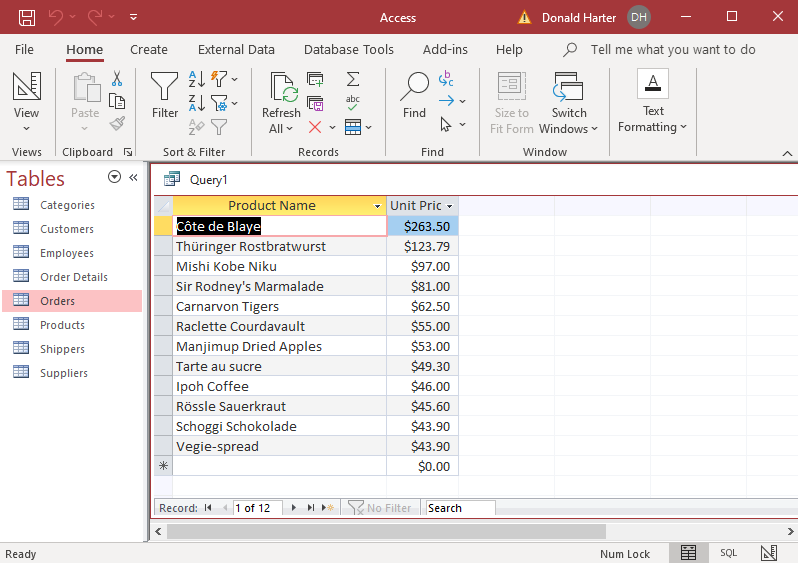


1. How would you select the bottom three? Order by Ascending, then TOP 3.
2. Change TOP 3 to TOP 11 and rerun.

SELECT TOP 11 ProductName, UnitPrice

FROM Products

ORDER BY UnitPrice DESC;

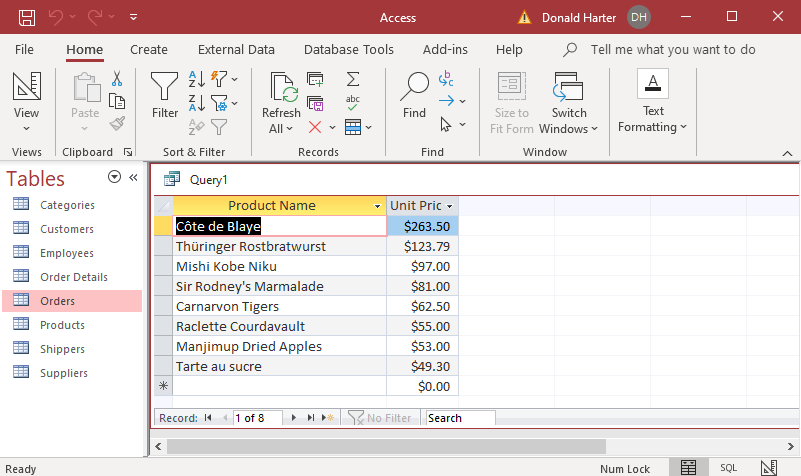


1. Note that it returned 12 records, not 11. When there is a tie in Access, it returns the tied values, even if it exceeds the TOP limit.
2. You can also retrieve by a Percent. In this example, retrieve the TOP 10 PERCENT.

SELECT TOP 10 PERCENT ProductName, UnitPrice

FROM Products

ORDER BY UnitPrice DESC;

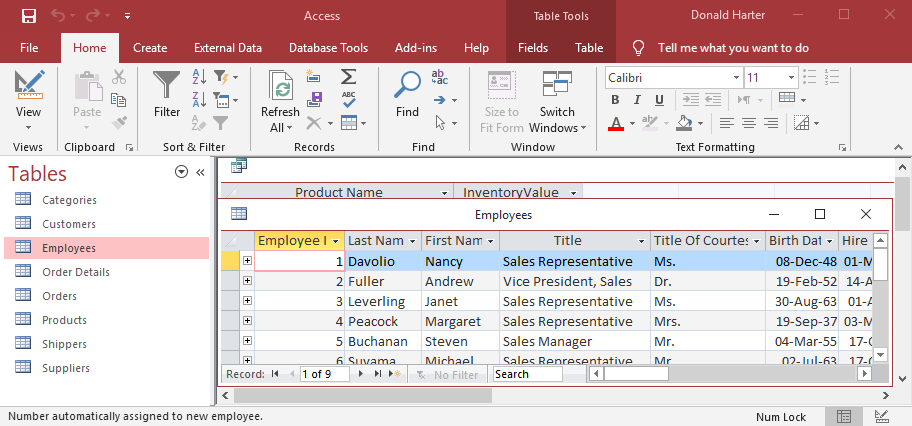


1. There are 77 records, so 10 percent of 77 is approximately 8 records.

**4.9.12 SQL: Concatenation**

Occasionally, you will want to combine fields. This is called concatenation.

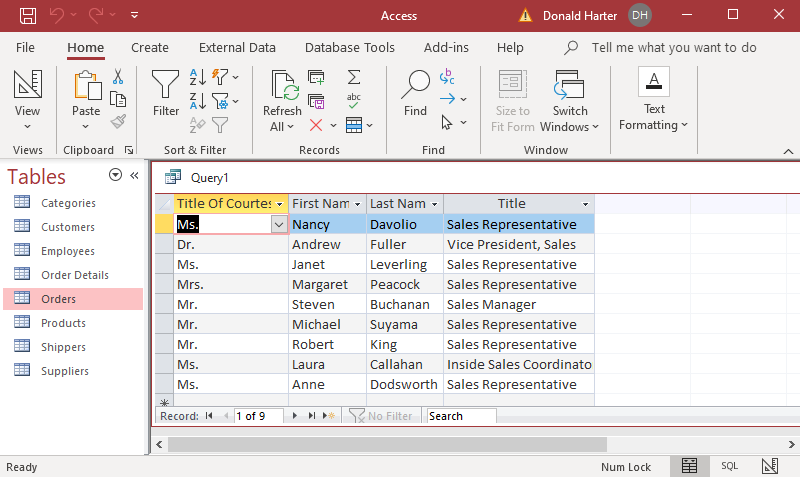
1. The Employees table has an employee’s last name, first name, title, and title of courtesy.



1. To create a query that listed title of courtesy, then first name, then last name, the title, we would write:

SELECT TitleofCourtesy, FirstName, LastName, Title

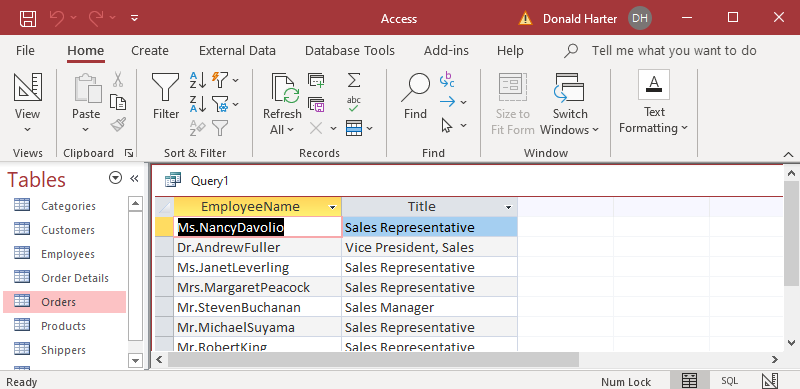
FROM Employees;



1. Next, concatenate (merge) the Title of Courtesy, First Name, and Last Name, labelling as EmployeeName, by inserting + between the fields.

SELECT TitleofCourtesy + FirstName + LastName AS EmployeeName, Title

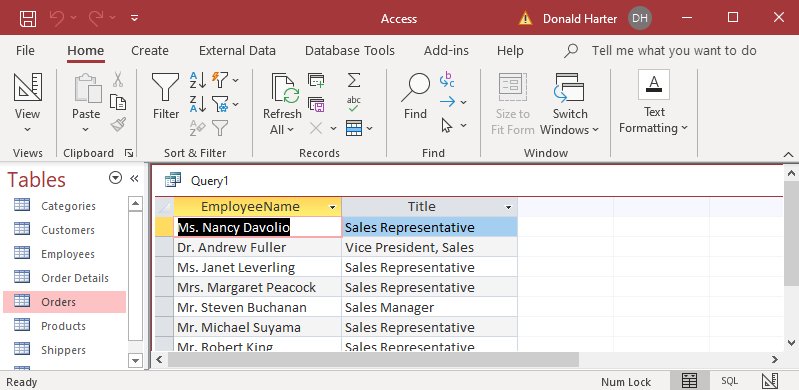
FROM Employees;



1. Note that we lose some spacing. That can be fixed by inserting spaces.

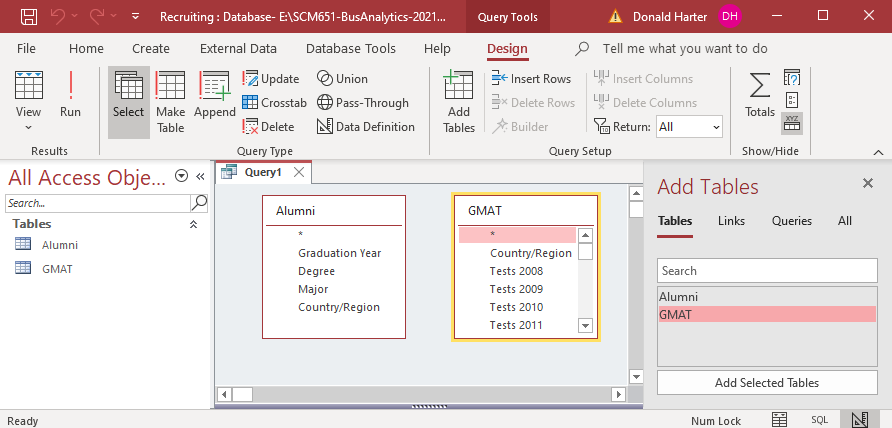
SELECT TitleofCourtesy + ' ' + FirstName + ' ' + LastName AS EmployeeName, Title

FROM Employees;

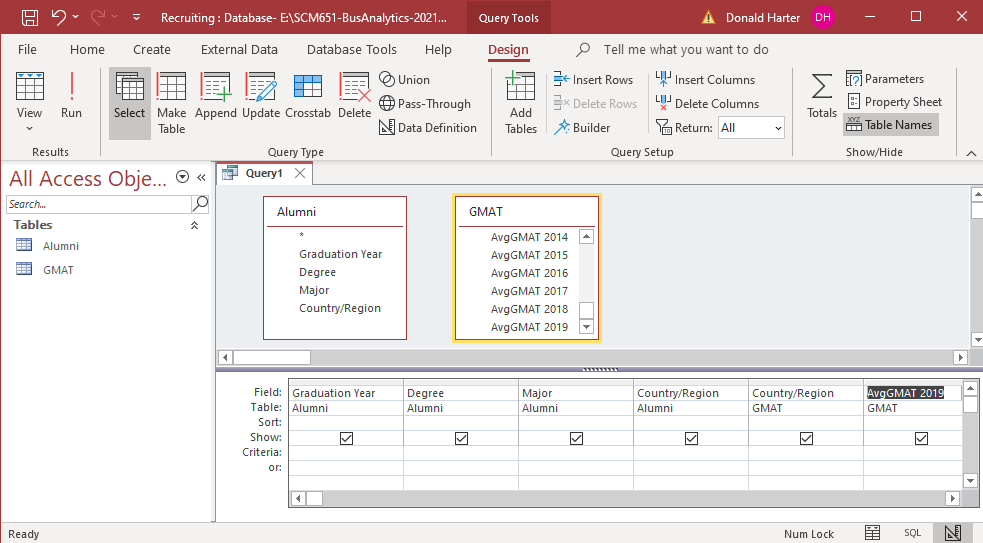


**4.9.13 SQL: Joins**

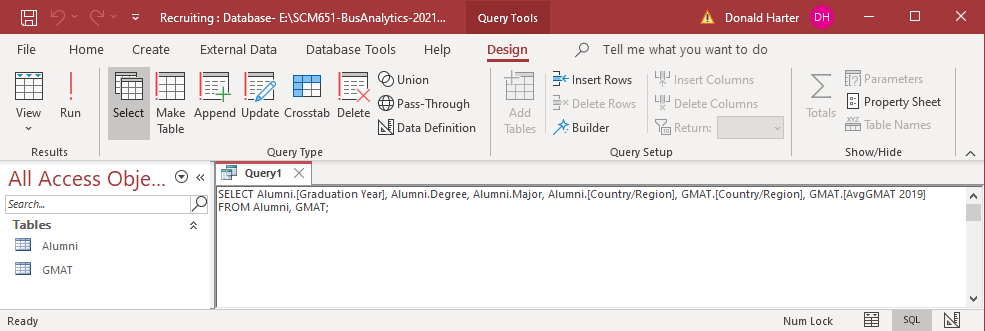
1. For this exercise, use the Recruiting database.
2. Click Enable, then Yes for trust.
3. Create a query by clicking on Create, Query Design.
4. In Add Tables, double click on Alumni.
5. In Add Tables, double click on GMAT.
6. Click on the X in Add Tables to close the right window.



1. Notice that there is no relationship between Alumni and GMAT. We will create the join in SQL.
2. Double click on Graduation Year, Degree, Major, and Country/Region in the Alumni table.
3. Double click on Country/Region and AvgGMAT 2019 in the GMAT table.



1. Click on View, SQL.

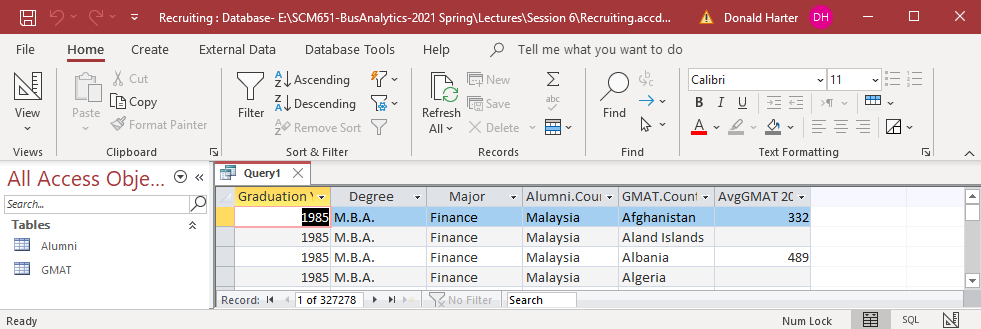


1. You should see the SQL:

SELECT Alumni.[Graduation Year], Alumni.Degree, Alumni.Major, Alumni.[Country/Region], GMAT.[Country/Region], GMAT.[AvgGMAT 2019]

FROM Alumni, GMAT;

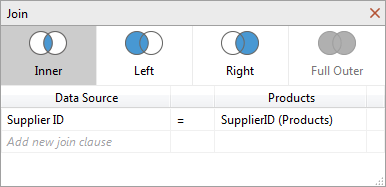
1. Note that Graduation Year has a space between Graduation and Year. To accommodate this space, Access add square brackets [ ] to the field name. Similarly Country/Region has a / between the words; we need to add brackets.
2. Also note that when Country/Region is used twice, Access adds the table name in front of Country, e.g., Alumni.Country/Region and GMAT.Country/Region.
3. Why are there square brackets around AvgGMAT 2019?
4. Click Run !.
5. How many records are there?
6. This is the Cartesian product: 1358 records in Alumni x 241 records in GMAT = 327,278 combinations.



**Join Types**

There are four types of database joins:

* Inner join: Only include rows where the joined fields from both tables are equal
* Left join: Include all records from the left table and only those records from the right table where the joined fields are equal
* Right join: Include all records from the right table and only those records from the left table where the joined fields are equal
* Outer join: Include all records from the left and right table where the key field appears in either table



**INNER JOIN**

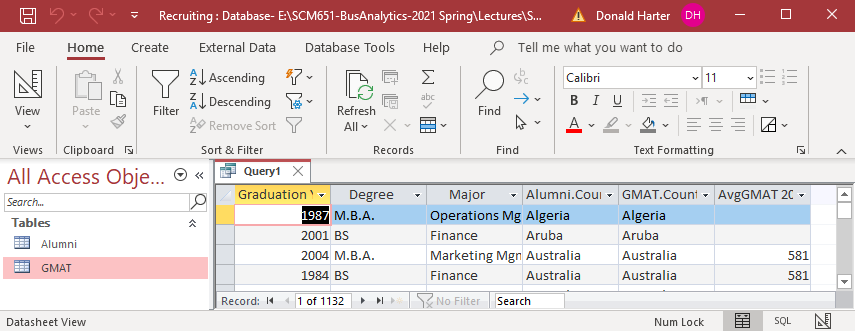
1. Click View, Design View to return to the SQL.
2. Create an INNER JOIN with the following changes:

SELECT Alumni.[Graduation Year], Alumni.Degree, Alumni.Major, Alumni.[Country/Region], GMAT.[Country/Region], GMAT.[AvgGMAT 2019]

FROM Alumni INNER JOIN GMAT

ON Alumni.[Country/Region] = GMAT.[Country/Region];

1. Click Run !.
2. How many records are there?



**LEFT JOIN**

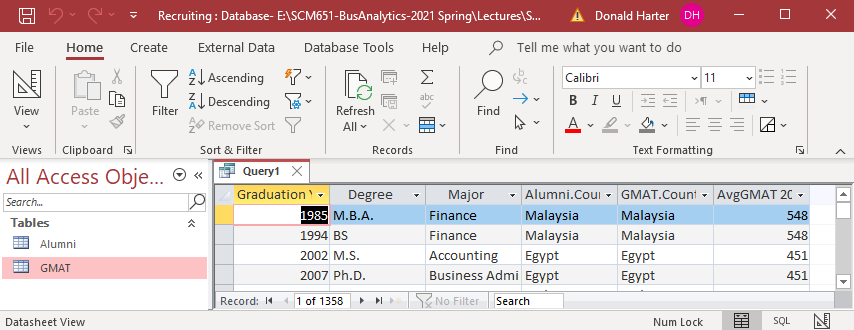
1. Click View, Design View to return to the SQL.
2. Create a LEFT JOIN with the following changes:

SELECT Alumni.[Graduation Year], Alumni.Degree, Alumni.Major, Alumni.[Country/Region], GMAT.[Country/Region], GMAT.[AvgGMAT 2019]

FROM Alumni LEFT JOIN GMAT

ON Alumni.[Country/Region] = GMAT.[Country/Region];

1. Click Run !.
2. How many records are there?



**RIGHT JOIN**

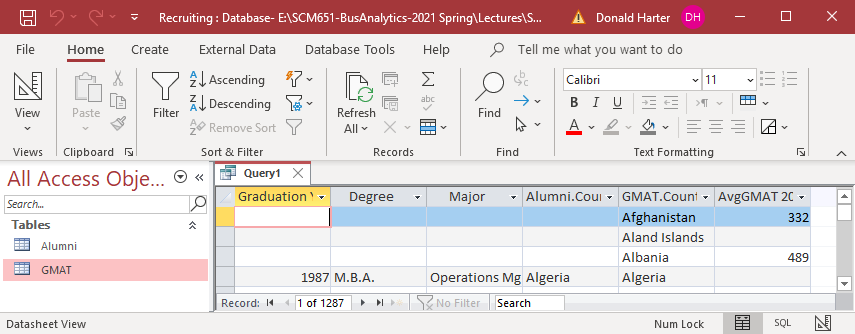
1. Click View, Design View to return to the SQL.
2. Create a LEFT JOIN with the following changes:

SELECT Alumni.[Graduation Year], Alumni.Degree, Alumni.Major, Alumni.[Country/Region], GMAT.[Country/Region], GMAT.[AvgGMAT 2019]

FROM Alumni RIGHT JOIN GMAT

ON Alumni.[Country/Region] = GMAT.[Country/Region];

1. Click Run !.
2. How many records are there?



**OUTER JOIN**

1. An OUTER JOIN can be created by performing a UNION on a LEFT JOIN and a RIGHT JOIN.
2. Click View, Design View to return to the SQL.
3. Create an OUTER JOIN with the following changes:

SELECT Alumni.[Graduation Year], Alumni.Degree, Alumni.Major, Alumni.[Country/Region], GMAT.[Country/Region], GMAT.[AvgGMAT 2019]

FROM Alumni LEFT JOIN GMAT

ON Alumni.[Country/Region] = GMAT.[Country/Region]

UNION

SELECT Alumni.[Graduation Year], Alumni.Degree, Alumni.Major, Alumni.[Country/Region], GMAT.[Country/Region], GMAT.[AvgGMAT 2019]

FROM Alumni RIGHT JOIN GMAT

ON Alumni.[Country/Region] = GMAT.[Country/Region];

1. Click Run !.
2. How many records are there?

