

Introduction to Access

Hunting database

Key points:

Access is one brand of “database manager”; it is the most commonly used, but also the smallest and least powerful. Other database manager brands include SQL Server, Oracle, MySQL (which is open-source software), and FoxPro. All of these other brands are much more powerful.

Access can hold about 2 gigabytes worth of data. Unlike Excel, there isn’t a hard and fast number I can give you in terms of how many records that might be. It all depends on how many tables and fields you have and how big those fields are. So for example, if you have a table with 20 fields and each field is the full 255 bytes wide, that will take up much more space than a table with 20 fields that are each 2 bytes wide. A general rule of thumb is that with a “typical” database you can probably get a million or so records. Once you start getting above a million records or you have a database that encompasses many tables – then you’ll have to go to more powerful software like SQL Server.

So it’s crucial that when you make a request for data, you ask the source about the size of the data. How many megabytes or gigabytes? How many tables? How many records? How many fields? This will give you an idea of whether or not you have the appropriate software. If you don’t have the right software, you may have to change your request to limit the number of records or only get key fields or something like that.

The types of analysis where Access is the better tool include:

- Summarizing your data – for example, taking a database where each record represents one campaign contribution and summarizing it to find out which person gave the most money. These are called “group by” queries in Access and they are very similar to how we used Pivot Tables in Excel. However, Access gives you much greater flexibility and power for this kind of summarizing.
- Linking two datasets together that were not intended to go together. For example, taking a table of convicted felons and matching it to a table of school bus drivers by linking names, dates of birth and other identifying information found in both tables.
- It’s also easier to bring information together from multiple sources for other reasons. For example, let’s say you have a table where each record is a county and you have the most recent population totals. And then you have another table where you have information from the schools about how many school-age children live in each county. You might want to merge the two together to find

out what percentage of the total population is of school-age – giving you the opportunity to find places that have a greater need for new schools, for example.

An Access database is kind of like a “shell” that can hold many types of objects. These objects can include tables, saved queries, reports, forms and other more advanced items. [Open a new Access database and look at the navigation box]. The tabs on the left side of the navigation box allow you to toggle around and see the different types of objects stored in your database.

Let’s look at the hunting accidents data. This database has a single table called “deer.” We have a record layout for this table that tells us the name of each field, what its data type is, how many bytes it takes up and a short description of what it contains. It also shows us what order the fields appear in. When you get new data, having a record layout is the key piece that you need to begin.

Let’s look at the design view of this table. You’ll see that there are many different data types and if you click on one of them, you’ll see that Access has a pull-down menu showing the possible data types (text, date, numeric).

The design view of the table is also the place where you can add new fields or rename existing fields. It’s good practice, however, to leave existing fields alone. Don’t make any changes to them.

There are two ways to ask questions in Access. One is the graphical user interface (called the design view) and the other is by using Structured Query Language (SQL). We’re going to learn SQL for two reasons:

- 1) Knowing SQL allows you to easily move into other database manager software. The design view is specific to Access
- 2) It’s easier to share your SQL with others, particularly when you need to troubleshoot a problem or convey to me how you got the answer to a question in an assignment or a test.

About SQL:

SQL is essentially English boiled down into keywords. You need a minimum of 2 lines of SQL to run a basic query and could have a maximum of 5 or 6 lines. Here are the primary SQL codes we're going to use:

SELECT
FROM
WHERE
GROUP BY
ORDER BY

The main thing to remember right now is that the computer requires you to be precise – just like with formulas in Excel. If you type something just a little bit wrong, you'll either get an error message or get the wrong answer.

HUNTING DATABASE:

When Bob Imrie, an Associated Press reporter in Wausau, WI, went searching for information about hunting accidents, he ended up right on target. The state filed hardcopy reports on every accident detailing information such as weather, topography and the number of pieces of orange the victim was wearing. Imrie had no way of digging out trends from the voluminous hard-copy reports and he had never used a computer database. But with some help over the phone from some nerdy colleagues, he created his own database to look at trends in hunting accidents in Wisconsin.

The record layout, which is the map to any database, is below. It shows the fields Imrie created in the database, whether they were characters or numbers and the width of each field:

- 1 CASE Character 10 This is an identifier he created
- 2 DATE Date 8 Date of accident
- 3 TIME Numeric 4 Time of accident
- 4 COUNTY Character 11 County of accident
- 5 AREA Character 6 Region of state (he created the areas)
- 6 WOUND Character 6 Part of body wounded
- 7 INJURY Character 5 Severity of injury
- 8 TYPE Character 2 Self-inflicted, or other (si=self-inflicted, sp=second person)
- 9 CAUSE Character 30 Description of cause
- 10 SAGE Numeric 2 Age of shooter
- 11 VAGE Numeric 2 Age of victim
- 12 FIREARM Character 7 Type of firearm
- 13 FACTION Character 6 Faction of gun
- 14 ALCOHOL Character 3 Alcohol involved

15 ALCOLEV Numeric 5 Alcohol level
 16 WEATHER Character 11 Weather
 17 TOPOGRO Character 10 Typography
 18 SEXPER Numeric 2 Years of shooter's experience
 19 VEXPER Numeric 2 Years of victim's experience
 20 SGRADUATE Character 3 Shooter graduate of safety school
 21 VGRADUATE Character 3 Victim graduate of safety school
 22 SSEX Character 1 Sex of shooter
 23 VSEX Character 1 Sex of victim
 24 GUNBRND Character 13 Brand of gun
 25 GUNGUAGE Character 10 Gauge of gun
 26 TEMP Numeric 2 Temperature
 27 MUZDIS Numeric 5 Muzzle distance
 28 LAND Character 3 Private or public
 29 VORANGE Numeric 1 Number of pieces of orange worn by victim
 30 GUNSIGHT Character 5 Open, scope or other
 31 VACTIVITY Character 10 Victim's activity
 32 LOCATION Character 6 Location of victim
 33 PRECIP Character 3 Precipitation

Do some basic queries:

- 1) How many people were injured or killed in incidents that occurred in Jackson County?
- 2) How many were injured or killed in incidents that occurred in St. Croix County and Pierce County combined?
- 3) How many people were injured or killed in incidents where the cause indicates some type of careless use of the weapon?
- 4) Do a query to show all the injuries occurring physically below, and including, the ankle. How many injuries were there?
Hint: here's a list of all possible wounds: abdomen, ankle, arm, back, butt, chest, ear, eye, face, finger, foot, groin, hand, head, hip, knee, leg, neck, scrotum, shoulder, stomach, thigh, thumb, toe.
- 5) How many accidents were self-inflicted, caused by shooters older than 50?
- 6) How old was the oldest victim in Jackson County? What wound did they have? How old was the youngest? What wound? (hint: this might require more than one query)
- 7) How many accidents occurred overall on the first day of deer hunting season (November 19, 1994)?
- 8) How many accidents involved a shooter under age 18? Of those, how many were fatal?
- 9) How many accidents involved either a female victim or a female shooter?

- 10) How many self-inflicted accidents involved a shooter who did NOT take a gun safety class ?
- 11) How many fatalities involved alcohol?

Group By Queries in Access:

So far we've merely filtered our data to get answers to our questions, but most of the time you'll want to ask a question that requires you to summarize either all of your data or part of it. This is where you need to do what's called a group by query. It's essentially the same thing as a pivot table in Excel, but is immensely more powerful and flexible. Open up the Hunting database we worked with last class. Remember we asked it to show us how many accidents occurred in Jackson County? Wouldn't it be great to see all the counties and how many there were in each one?

Let's step back a minute and talk about when we would need a group by query: Typically the records in your database will each represent one thing (1 hunting accident, 1 baseball player, 1 crime incident, 1 campaign finance donation) and you may want to count them or summarize them or "group" them and compare the groups.

So questions that might require a group by query would be like:

Hunting accidents -- Which county had the greatest number of accidents? Or How many victims were women, compared to how many were men?

Baseball salaries -- Which team paid the most in salaries last year? Or What are the average salaries for each position?

Crime incidents -- What type of crime occurred most often last year? Did they occur more often at night than during the day?

Campaign finance -- Which donor gave the most money? Or which candidate received the most money? Or which candidate received the greatest number of contribution checks?

I've underlined the key words that should tip you off that you might need a group by query to get the answer. Notice how many of these questions ask you for one thing in your answer -- i.e. "which team paid the most". Although you're asking a question that implies you need just one thing in your answer, you really will need to see ALL the teams to put them in order based on the total money they paid out. Then whichever team lands at the top of your list, is the answer to your question.

Before we do a group by query, let's look at some of the FUNCTIONS in Access that we are going to use as part of our group by queries.

The main ones that you will find yourself using are:

COUNT
SUM
AVG

The COUNT function will count the number of records. The SUM will total all the values in whatever field you've selected. And the AVG function will average all the values in whatever field you've selected. NOTE: Access does not have a built-in median function (one of my dislikes about Access!)

Let's put these into action in the Hunting database, just to see how they work.

Remember that the Hunting database is one record for each person injured or killed in a hunting accident in Wisconsin between 1987 and 1994.

So we can use the COUNT function to count how many injuries/deaths there were, by using this query:

```
SELECT COUNT(*)  
FROM DEER
```

This query will return one number (87) because that's all the query asked for. It doesn't ask to see any of those records. Notice that I put an asterisk inside the parentheses.

The COUNT function doesn't require us to specify a particular column (unlike the SUM and AVG functions). You can do that, however I'd advise against it. If you did put a column name in there instead of the asterisk, it will only count the records where that column has a value (it will ignore the NULL or blank values).

This database doesn't have a good field for using the SUM function (a dollar field is what you'll use it for most often). But the AVERAGE function works the same way. So let's find out the average age of the victims (the field VAGE in the hunting database)

```
SELECT AVG(VAGE)  
FROM DEER
```

This query also returns one number. In this case, it's 30.8. In other words, the average age of the victims was about 31 years.

The two queries we just run generated numbers based on all the records in the database.

Let's say we want to count the number of accidents in each county. In other words, we want one record for each county, showing the number of accidents in each. For this, we'll use the COUNT function, along with a group by.

Let's try this one out first and then we'll explain how it works. Here's the SQL syntax:

```
SELECT COUNTY, COUNT(*)  
FROM DEER  
GROUP BY COUNTY
```

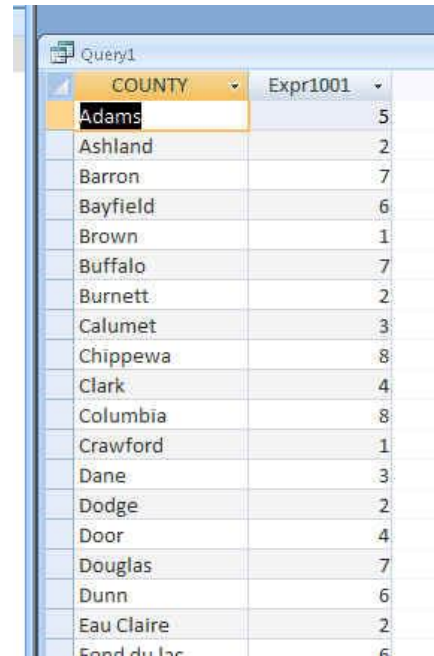
To the right is a picture of the first few records from our answer.

You'll see that we got 64 records back – that's the number of counties in Wisconsin.

You'll also see that we only have 2 columns in our answer.

Notice that our SELECT line only has two columns – the name of the county and the result of the COUNT.

Here's what Access just did: Because we did NOT have a where clause, it looked at all the records in our database and then it divided them up into virtual "piles"; all the Adams county ones in a pile; all the Ashland county ones in another, etc. Then it went through and counted the number of records in each "pile" and returned that number in our second column.



COUNTY	Expr1001
Adams	5
Ashland	2
Barron	7
Bayfield	6
Brown	1
Buffalo	7
Burnett	2
Calumet	3
Chippewa	8
Clark	4
Columbia	8
Crawford	1
Dane	3
Dodge	2
Door	4
Douglas	7
Dunn	6
Eau Claire	2
Fond du Lac	6

Notice that "county" (the name of the column) appears in BOTH the SELECT line and the GROUP BY line. Why? That's one of the hard and fast rules of SQL. Whatever criteria (or field) you are using to dictate your groups or "piles" must appear in both the Group By and the Select lines. AND ...you can't have any other fields in the select line (only a function like COUNT or SUM or AVG)

It returned the answer in alphabetical order by the county field (the first field) because that's the default when you don't include an ORDER BY statement.

So if we wanted to adjust this query so that we see which county had the MOST accidents, simply add an ORDER BY, like this:

```
SELECT COUNTY, COUNT(*)  
FROM DEER  
GROUP BY COUNTY  
ORDER BY 2 DESC
```

Why does it say 'ORDER BY 2 DESC'?

We could say "ORDER BY COUNT(*) DESC", but Access allows us to use a little shortcut here and just refer to the column number – the column in our answer. In this case, the name of the county is column one (it's the first item in our SELECT line) and the COUNT

result is our second column. So we want the answer to be displayed largest to smallest, according to that COUNT. You can use this shortcut anytime you use ORDER BY. It doesn't need to be a group by query.

What if we want to create our "piles" based on more than one criteria? For example, what if I wanted to see the number of accidents in each county that were self-inflicted and the number that were second person. A cross-tab query would be a fancier way to do this, but let's just see what happens when we put these two things together in a regular group by query:

```
SELECT COUNTY, TYPE, COUNT(*)  
FROM DEER  
GROUP BY COUNTY, TYPE  
ORDER BY 1
```

You'll see now we have more than 1 record for each county. In the case of Barron County, we have 3 – because there were 2 records where the TYPE field was not filled in, 1 where the injury was self-inflicted and 4 where it was a second person.

Probably one of the most confusing things for a newbie learning Access is this idea that with a group by query you can't see everything in one swoop.

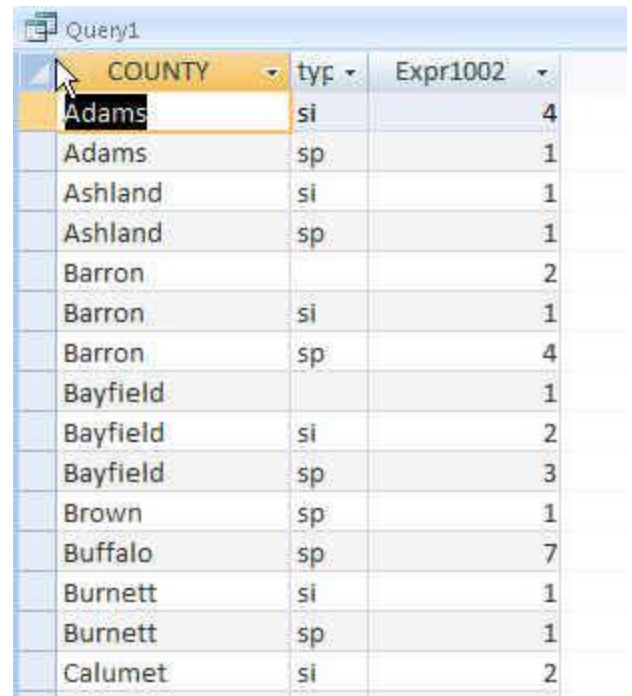
I've seen reporters flummoxed by this because they want to find out which county had the most accidents, but they also want to be able to see – at the same time – whether the majority of them were self-inflicted or not.

The problem is that the query we just ran won't give you both answers at the same time. Notice that if you sort this last query by the COUNT field, you'll end up with Buffalo County at the top – but this isn't showing ALL the accidents in Buffalo; it's only showing the 7 that were second-person.

You either need to run the two queries separately or get into more sophisticated queries (like a cross tab query). As a beginner, I'd recommend you stick with simple queries and just constantly remind yourself that each group by query is only going to tell you one thing.

The best way to approach Group By queries is to first draft your question, then think about what you expect the answer to look like.

Let's try out another question....



COUNTY	typ	Expr1002
Adams	si	4
Adams	sp	1
Ashland	si	1
Ashland	sp	1
Barron		2
Barron	si	1
Barron	sp	4
Bayfield		1
Bayfield	si	2
Bayfield	sp	3
Brown	sp	1
Buffalo	sp	7
Burnett	si	1
Burnett	sp	1
Calumet	si	2

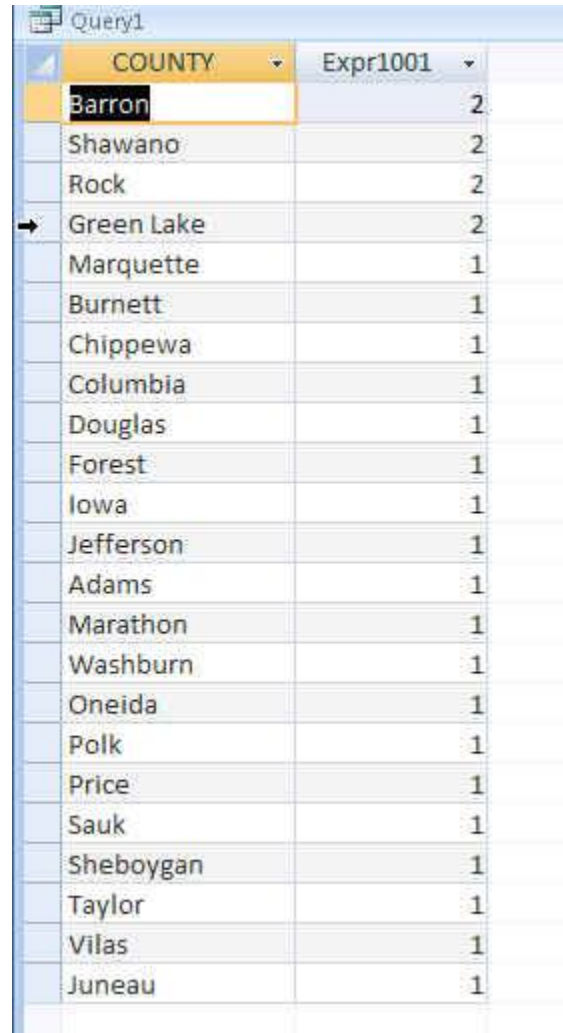
Which county had the most fatal accidents?

Let's deconstruct our question. First, we're asking it to ONLY look at fatal accidents. So that means we need to filter our data – in other words, use a WHERE statement.

Second, we're asking for the name of a county. And in order to see which county had the MOST, we need to see all the counties in a list. So we're going to need the field COUNTY in our SELECT line.

Finally, we want a number for each county – so we need to COUNT the records. And because we're counting, we need to use GROUP BY. And we'll need an ORDER BY putting the records in descending order by the COUNT field so we can quickly see which county had the most fatal accidents.

So that means our SQL would look like this:
SELECT COUNTY, COUNT(*)
FROM DEER
WHERE INJURY= "FATAL"
GROUP BY COUNTY
ORDER BY 2 DESC



COUNTY	Expr1001
Barron	2
Shawano	2
Rock	2
Green Lake	2
Marquette	1
Burnett	1
Chippewa	1
Columbia	1
Douglas	1
Forest	1
Iowa	1
Jefferson	1
Adams	1
Marathon	1
Washburn	1
Oneida	1
Polk	1
Price	1
Sauk	1
Sheboygan	1
Taylor	1
Vilas	1
Juneau	1

The best way to learn Group By queries is to practice, practice, practice. So here are some other questions to answer:

12) What was the most common wound?

13) What were the most common types of self-inflicted wounds?

- 14) What was the most common cause of hunting accidents?
- 15) Which brand of gun is involved in the most hunting accidents?
- 16) What is the average age of victims in self-inflicted accidents?
- 17) What is the average age for the shooters in butt-injury accidents?
- 18) On which date did the most accidents occur? After getting this answer, then run a query to find out where those accidents occurred. Which county(s) had the most on that date?
- 19) What percentage of victims were wearing 2 or more pieces of orange?
- 20) Write a paragraph with two or three of the most interesting things you found in the data (either through the questions or through your own analysis). Writing should meet AP style and be free of grammar and spelling errors. Also apply numeracy concepts you read about (not using too many numbers, avoiding decimals, etc)