



Exercise2

Querying Tables

Introduction

In ArcGIS, searching for values in your data to answer spatial questions and perform spatial analysis is done using Structured Query language (SQL). ArcMap provides two methods for searching and selecting records: Select by Attribute and Select by Location. In this exercise, we will use the Select by Attribute method to build queries.

Overview of major concepts:

Querying different data sources
Searching Strings
Combining query expressions
Wildcards

Assumptions:

- A basic knowledge of working in ArcGIS
- A basic knowledge of working in Windows
- The data for the course was downloaded and unzipped
- You know the location of the Data folder—pathnames in the exercise start with the Data folder.



Contents

Part 1: Querying different data sources	2
A. Learn to use the correct syntax for searching different kinds of data sources. First, you will take a look at a File GDB and then a Personal GDB.....	2
B. Now let's look at the same query, only this time in a personal geodatabase.....	4
C. Let's look at case sensitivity in the query builder.....	5
Part 2: Searching Strings	6
A. Learn to search for several strings or values in a field. When searching for multiple values, you use the IN operator.	7
Part 3: Combining Query Expressions.....	8
A. Learn to write expressions using multiple search criteria	8
Part 4: Wildcards.....	10
A. Learn to write expressions using wildcards.....	10

Part 1: Querying different data sources

A query is a statement that selects required data. The syntax you use to build a Structure Query Language (SQL) expression differs depending on the data source. This is because although SQL is a standard, not all database software implements the same dialect of SQL.

- **To query file-based data** (such as file geodatabases, coverages, shapefiles, INFO tables, dBASE tables, and CAD data): you use a dialect of SQL implemented within ArcGIS that supports a subset of the features and functions available in personal and ArcSDE geodatabases.
- **To query personal geodatabases:** you use the same syntax used in Microsoft Access.
- **To query an ArcSDE geodatabase:** you use the SQL syntax of the underlying database management system (DBMS)—either Oracle, SQL Server, DB2, Informix, or PostgreSQL.

A. Learn to use the correct syntax for searching different kinds of data sources. First, you will take a look at a File GDB and then a Personal GDB.

1. Launch ArcMap .

2. Open **Query_Tables.mxd** located in: ...\\Data

The project contains 2 identical data frames, however one contains data from a File GDB (**Resources.gdb**) and the other contains data from a Personal GDB (**Resources.mdb**). Let's examine the different SQL syntax used between the 2 data sources.

3. Activate the **FILE** geodatabase data frame, if needed.

Hint: Right-click on the name of the data frame and choose Activate.

4. Open the attribute table of the **Fire History Polygons** layer.

5. Click the **Select by Attribute** button. 

The SQL dialog boxes helps you create the correct syntax for the data you're querying—it lists the field names and values with the appropriate delimiters (example: ', ', [], etc.). It also selects the relevant keywords and operators for you.

6. Build the following query to select all the fires that burned more than 1000 acres:

ACRES > 1000

SELECT * FROM Fire_History_pl WHERE:
ACRES >1000

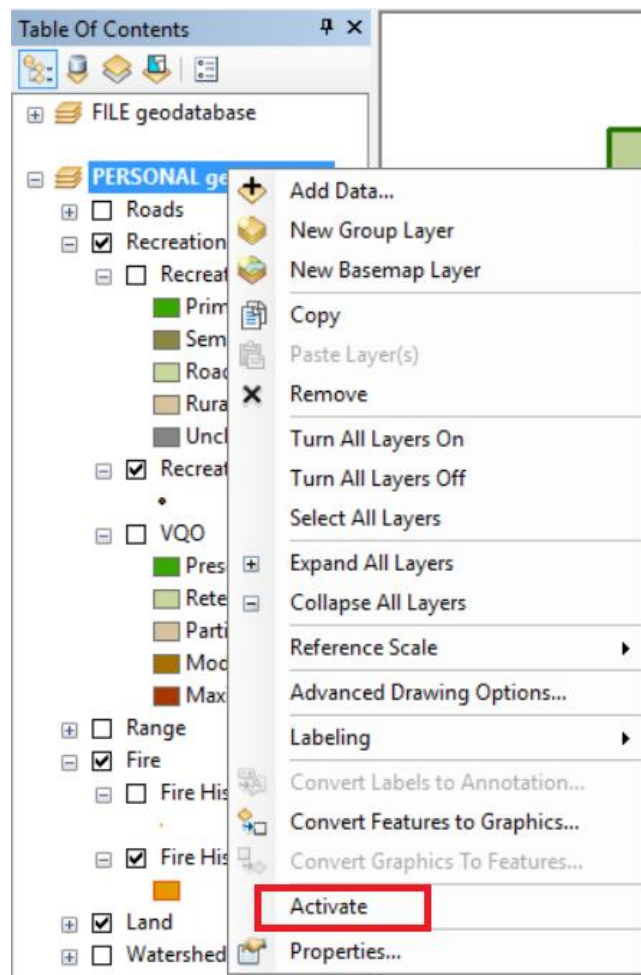
The File Geodatabase SQL syntax sometimes places *double quotes* around the field. In the example shown above, it didn't, but it will still work. If you build the same query in a Personal Geodatabase, you will see that the fields are *enclosed in brackets*.

7. Select **Apply** and **Close**. *16 of the 150 records are selected.*
8. **Close** attribute table.

B. Now let's look at the same query, only this time in a personal geodatabase.

9. Activate the Personal Geodatabase data frame and expand all the layers.

(Hint: Right-click on the name of the data frame and choose Activate.)

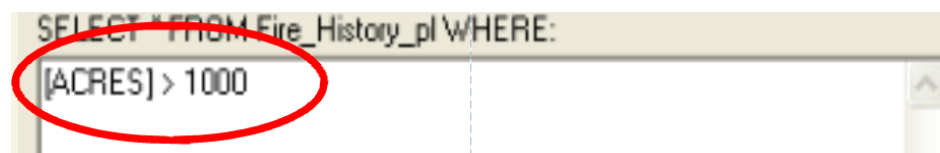


10. Open the **Fire History Polygons** attribute table, and open the **Select by Attributes**

window.

11. Build the same query, selecting all fires larger than 1000 acres:

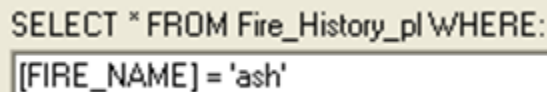
[ACRES] > 1000



12. Click **Apply** to run the query.

C. Let's look at case sensitivity in the query builder.

13. Build a new query to select a fire named Ash. Instead of selecting the Value 'Ash' from the query builder, type 'ash' with a lower case, (see query in following screen capture). **[FIRE_NAME] = 'ash'**



14. Click **Apply** and **Close** the query builder window. The Personal geodatabase is not case sensitive and should return 2 results.

If you are clicking to enter the expression, the Query builder window will create the correct syntax for the data you're querying. Why should you be concerned about different SQL dialects if the query builder applies the correct syntax? Well, when you build your own queries from scratch, combine expressions, or import expressions you may run into syntax errors. SQL is sensitive to unnecessary spaces, and file geodatabases are even case sensitive (Personal GDB are not).

15. **Close** the table.

16. Activate the **FILE** Geodatabase data frame.

17. Open the **Fire History Polygons** table, and open the **Select by Attributes** window.

18. Build a query to **select** the fire named **Ash**. Again, instead of selecting the Value 'Ash' from the query builder, type 'ash' in all lower case. **FIRE_NAME = 'ash'**

19. Click **Apply** to run the query. Select **OK**.

No records are selected because the SQL for File geodatabases is case sensitive.

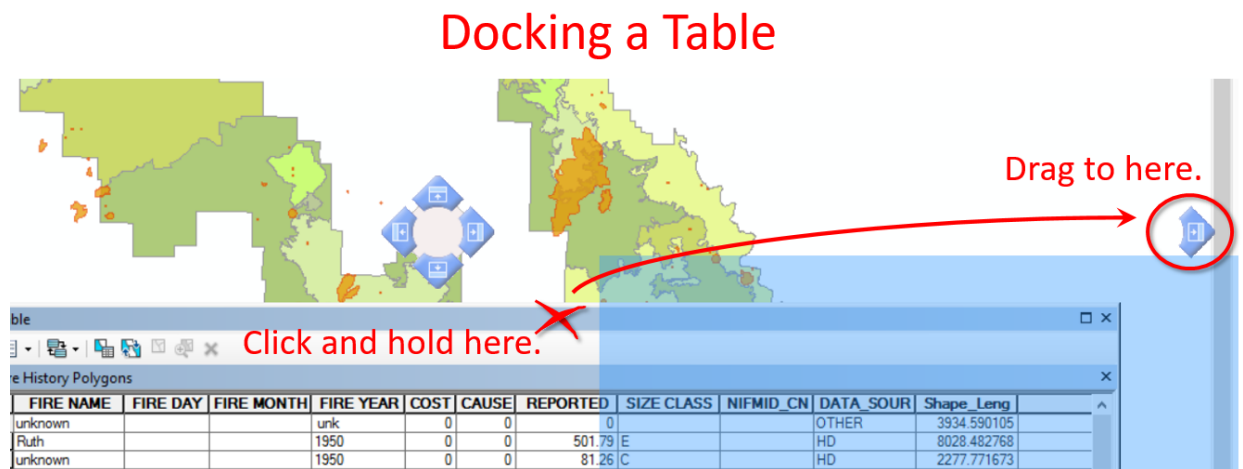
20. Correct the lower case 'ash' to Upper case 'Ash', click **Apply** and **Close**.

The File geodatabase now returns 2 matching records. When querying File geodatabases you should be aware of the case sensitivity. One way to avoid case sensitivity errors is to select from

the query builder Unique Values. Another method that may be useful to search for records with varying cases is using a wildcard in your search (refer to Step 4 to learn about wildcards).

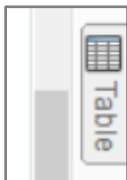
21. **Clear** all selected features.

22. Dock the **Attribute table** by dragging it to and dropping it on the blue arrow that appears on the right edge of the ArcMap window when you start dragging it.



23. Place the Attribute table in Auto Hide mode by clicking on **Auto Hide** in the top right corner.

This minimizes the table window and turns it into a table tab which will open if you place your mouse over the tab.



Part 2: Searching Strings

This part of the exercise will only use a File geodatabase (since this is the most commonly used geodatabase). You will learn how to build basic queries to search and select attributes from your datasets.

A string is an ordered sequence of symbols/data types. When querying for attribute values, you are essentially using search strings. Strings must always be enclosed within single quotes in the SQL statement.

In the next steps you will build queries to: search for multiple values within a field, search based on multiple criteria, and use wildcards to locate values that match any character with a field.

A. Learn to search for several strings or values in a field. When searching for multiple values, you use the IN operator.


1. Activate the **FILE** geodatabase data frame, if needed.
2. **Open** the attribute table for the Mgt Areas layer and **open** the Select by Attributes window.
3. Build the following query to search for the Mgt Areas: Castle Creek, Cedar Bench, Pine Mountain, Sycamore Canyon:
4. The four management areas are selected.

```
MGT_AREA_D IN ('Castle Creek', 'Cedar Bench', 'Pine Mountain', 'Sycamore Canyon')
```

```
SELECT * FROM Mgt_Areas WHERE:
```

```
MGT_AREA_D IN ('Castle Creek', 'Cedar Bench', 'Pine Mountain', 'Sycamore Canyon')
```

Hint: Clicking on a field name, the IN operator, the parentheses, and the 'Get Unique Values' button will save you from having to type in the names (Click on a name to add it to the query). You will still need to type the commas and spaces.

5. Click **Apply**.
6. Close the **Select by Attributes** window.
7. Clear all **selected records**. 
8. Click on the data view to **close** the attribute table (if needed). The Mgt Areas Table is now a tab in the Table window, which is reduced to a tab.

9. **Open** the Fire History Polygons attribute table and **open** the Select by Attribute query builder.
10. **Clear** any existing query within the expression box.
11. Build a query to search for the fires named: Ash, Ash Creek, Cedar Bench.

Hint: `FIRE_NAME IN ('Ash', 'Ash Creek', 'Cedar Bench')`

```
SELECT * FROM Fire_History_pl WHERE:
FIRE_NAME IN ( 'Ash', 'Ash Creek', 'Cedar Bench' )
```

Four fires areas selected.

12. Click **Apply** and **Close** dialog box.
13. **Clear** all selected records.
14. Minimize the **Fire History** Table (if you have it set to Auto hide it will minimize itself), or undock the table and close it.

Part 3: Combining Query Expressions

Sometimes your search requires multiple criteria. In this case combining expressions is useful. Complex expressions can be built by combining expressions with the **AND** and **OR** operators. To build an expression with more than one criteria, you build a query for which **both criteria must be true** or you can build an expression **that at least one of the criteria must be true**. First we will build an expression in which both criteria must be true. In cases such as this, you need to use the **AND** operator.

A. Learn to write expressions using multiple search criteria

1. Open the Mgt Areas attribute table and open the Select by Attributes window.
2. **Build** a query to search for Wilderness areas larger than 10,000 Acres. (**Hint:** see the screen capture below for the expression.):


```
SELECT * FROM Mgt_Areas WHERE:
MGT_AREA_T = 'Wilderness' AND GIS_ACRES > 10000
```

Hint: You need to manually type the 10000.

3. Take a look at the results, **clear** selected records, and close the **Select by Attributes** window.
4. Open the **Fire History Polygons** table and open the **Select by Attributes** window.
5. **Build** a query to search for fires which cost more than \$10,000 or burnt over 100 acres.

```
SELECT * FROM Fire_History_pl WHERE:
COST > 10000 OR ACRES > 100
```

When you use the **OR** operator, at least one side of the expression of the two separated by the **OR** operator must be true for the record to be selected.

6. Click **Apply** and **Close**.
7. Look at **Fire History Polygons** attribute table. 64 records have been selected.
8. Reopen the **Select by Attributes** dialog window and clear the query expression box.

NOT expressions can be combined with **AND** and **OR**. Use the **NOT** operator at the beginning of an expression to find features or records that do not match the specified expression.

9. **Build** the expression to select all fires from the year 1980 that are not caused by cause code zero:

```
SELECT * FROM Fire_History_pl WHERE:
FIRE_YEAR = '1980' AND NOT CAUSE = 0
```

10. Click **Apply**. 10 records have been selected.

If you apply this query using the OR operator instead of AND, all the fires from 1980 will be selected and all other fires that do not have fire cause code zero. Let's give this a try.

11. Replace the **AND** operator with **OR** in the query builder expression window.

```
SELECT * FROM Fire_History_pl WHERE:
FIRE_YEAR = '1980' OR NOT CAUSE = 0
```

12. Click **Apply**.
13. Close the Select by Attributes window.

146 of the 150 records have been selected. The Fires with cause code zero have been omitted from the selected fires that occurred in 1980.

14. **Clear** all selected features.
15. **Close** the table.

Part 4: Wildcards

Wildcards are used to conduct partial string searches. This is useful if exact spelling is unknown or when searching for a group of characters. Wildcards also depend on the data source you are querying. If querying a coverage, shapefile, INFO table, dBASE table, or shared geodatabase you use the '_' (underscore) to represent any one character and '%' for any group of characters. When using wildcards, you use the LIKE operator. Let's practice a wildcard query in the File geodatabase.

A. Learn to write expressions using wildcards

1. Turn on the **Recreation Group** Layer.

You will search for Henderson Cabin, however, you are not sure if the spelling is Henderson or Hendersen. You will place an underscore instead of the last vowel.

2. **Open** the Recreation Sites table and **Select by Attributes** window.

3. **Build** the following query to search for 'Henders_n Cabin' and click **Apply**:

```
"REC_SITE_N" LIKE 'Henders_n Cabin'
```

```
SELECT * FROM Recreation_Site_pt WHERE:
REC_SITE_N LIKE 'Henders_n Cabin'
```

Henderson Cabin is now selected.

NOTE: Remember that SQL in File geodatabases are case sensitive. Using the underscore/wildcard instead of the first letter may be useful when performing a query. Now let's try to search for all the recreation site names that include the word Lynx.

4. **Clear** the previous expression.
5. **Build** the following query to search for all the records that include the word Lynx in the recreation site name and click **Apply**:

```
"REC_SITE_N" LIKE '%Lynx%'
```

```
SELECT * FROM Recreation_Site_pt WHERE:
REC_SITE_N LIKE '%Lynx%'
```

All 7 records containing Lynx have been selected.

Hint: Use the operator button or manually place the percentage (%) symbol before and after the word Lynx, but make sure they are inside the single quotes.

6. **Clear** all selected records.
7. Close the **Select by Attributes** window and all tables that are open.

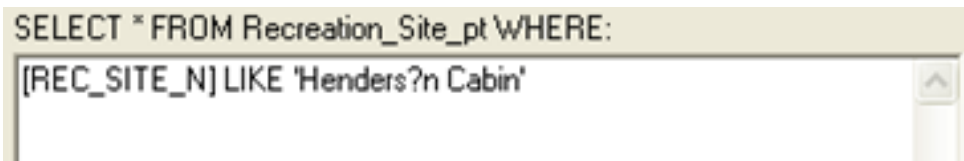
As mentioned earlier, wildcards depend on the data source you are querying. You just learned to search using wildcards in a File geodatabase. If you are still using Personal geodatabases, you need to use different wildcard symbols. In a Personal geodatabase, you use the '?' to represent any one character and '*' for any group of characters. We will perform the same queries as we just did in the File geodatabase. Remember to use the LIKE operator!

8. **Activate** the PERSONAL geodatabase Data Frame.
9. Check to make sure the **Recreation Group** layer is turned on.
10. **Open** the Recreation Sites table and **Select by Attributes** window for the Recreation Sites layer.

Again, you will search for Henderson Cabin, however, you have already forgotten if the spelling is Henderson or Hendersen. You will place a question mark instead of the last vowel.

11. **Build** a query to search for Henders?n Cabin and click **Apply**:

```
[REC_SITE_N] LIKE 'Henders?n Cabin'
```



12. **Clear** the previous expression.

Now let's try to search for all the recreation sites with names that include the word Lynx.

13. **Build** the following query to search for all the records that include the word Lynx:

```
[REC_SITE_N] LIKE '*Lynx*'
```



Hint: Use the operator button in the Select by Attributes window or manually place the asterisk (*) symbol before and after the word Lynx, but make sure they are inside the single quotes.

14. Click **Apply**.

All 7 records containing Lynx have been selected.

15. **Clear** all selected records.
16. Close the Select by Attributes window.
17. Close all open tables.

Another useful SQL operator is the **NULL** operator. The **NULL** keyword selects records that have null values. This may be useful when edits need to be applied to null values or such values may be omitted when performing analysis. The NULL keyword is always preceded by **IS** or **IS NOT**.

18. Activate the **FILE geodatabase** Data Frame.
19. Turn on the **Roads group** layer and expand it.
20. **Open** the attribute table and **Select by Attributes** window of the Trail Events layer.
21. **Build** a query to search for all the null records in the Juris Code field and click **Apply**.
(Hint: "Juris_Code" IS NULL)

```
SELECT * FROM TrailEvents WHERE:
Juris_Code IS NULL
```

13 records are selected. These records happen to be part of the USFS trail system, so let's apply a quick edit using the Field Calculator.

22. In the Attribute table, **right-click** the field header of the Juris_Code and **select** the Field Calculator.
23. Click **yes**, if prompted about making changes outside an edit session.
24. Enter "USFS" and click **OK**. (Hint: Make sure you have the double quotes around USFS since it is text.)

```
Juris_Code =
"USFS"
```



The null records have been updated.

25. **Save** your project and **close** ArcMap.

-END OF EXERCISE

Congratulations on finishing this exercise!

