SQL Query & Field Calculator Tips

for Data QC & Analysis in ArcGIS Desktop

Jennifer.Harrison@TeachMeGIS.com
A note on Languages

There are several languages supported in the ArcGIS Desktop user interface

For queries:
  – SQL

For calculations:
  – Python
  – VB Script

For labels/display expressions:
  – Python
  – VBScript
  – Java Script
  – Text formatting tags
SQL
Where can a Query be Used?

Anytime you are filtering your data:

– Select By Attribute
– Definition Query
– Label Classes
– Etc.
What can you do with SQL?

SQL is a robust language used to query many databases.

You can use many SQL operators and functions, even if there is not a button that exposes that functionality in the Query Builder dialogs.

– SQL Operators
– SQL String Functions
– SQL Mathematic Functions
Let’s review

- Wildcards
- NULL values
- Operators

Demo...GOM wells
Some special operators
There are some special operators that you can use that are not exposed through buttons

- BETWEEN
- IN
Instead of this:

```
"HC_TYPE" = 'OIL' OR "HC_TYPE" = 'GAS' OR
"HC_TYPE" = 'MIXED'
```

Use this:

```
"HC_TYPE" IN ('OIL', 'GAS', 'MIXED')
```

- Reduces the query size by eliminating repetitive OR statements.

The list must be comma separated.

The word IN is not case sensitive

Also works for numeric lists (omit the quotes)

```
"VALUE" in (150000, 2300000)
```
Instead of this:

“WATER_DEPTH” > 100 AND “WATER_DEPTH” < 500

Use this:

“WATER_DEPTH” BETWEEN 100 AND 500

• Reduces the query size by eliminating repetitive AND statements.

Between is inclusive of both values.

The word BETWEEN is not case sensitive

NOTE
Works on geodatabases, but not shapefiles
Subqueries

- A query within another query
- Used within geodatabases
- Provides ability to query from other layers and tables

- **IN**
  
  "OWNER" IN (SELECT "OWNER" FROM partners)

- **Scalar**, which uses comparison operators & returns single value
  
  "TOTAL_DEPTH" > (SELECT MAX("TOTAL_DEPTH") FROM water_wells)

Table must be in same geodatabase as the data for it to work!
Suppose someone gives you an excel spreadsheet of 10 wells in the Gulf of Mexico, and you want to quickly highlight those 10 wells. How would you do it?

1. Load the table into a geodatabase table (in the same geodatabase as the feature class).
2. Run a subselect, using IN

   “WELL_NAME” IN (SELECT “WELL_NAME” FROM ten_wells)

There is also a way to do this with a table join. Do you know how?
Suppose you have a feature class with well data from Texas and another feature class with well data from Arkansas. Now suppose you would like to find all of the Arkansas gas wells that have a total production greater than the biggest gas producer in Texas. How would you do it?

1. Load the Texas wells feature class into the same geodatabase as the Arkansas wells feature class.

2. Run a query on the Arkansas wells feature class using a subquery and a SQL function:

   "GAS_PROD" > (SELECT MAX ("GAS_PROD")
   FROM texas_wells)
Using Functions in SQL
1) Thing on the left of the operator whittles down to a value
2) Thing on the right of the operator whittles down to a value
3) The 2 values are compared by the operator

```
SELECT * FROM Schools WHERE:
"GRADERANGE" LIKE "%5%"

'01 02 03 04 05'
'03 04 05 06'
'05 06'
'1-5'
```
String Functions

**UPPER (<string>)**

\[ \text{UPPER(" COUNTY\_NAME"')} = 'HARRIS' \]

- Sets string to uppercase before making the comparison

**LOWER (<string>)**

\[ \text{LOWER(" CITY\_TYPE"')} = 'town' \]

- Sets string to lowercase before making the comparison

Strings are case-sensitive in expressions
String Functions

How would you find out if there are any records in your attribute table for which the STATE_NAME value is not all uppercase?

- “NAME” <> upper(“NAME”)

Some other examples:

- Upper(“NAME”) NOT IN (‘TEXAS’, ‘LOUISIANA’, ‘ARKANSAS’)

**SUBSTRING – SUBSTRING**(string_exp, start, length)

SUBSTRING(“REFNO”, 3, 2) = 20

• Returns values from within a string.

Suppose you have an attribute table with census block groups, where the first 2 characters indicate the state and the next three characters indicate the county, followed by a unique code, like this:

48157670300
48 157 670300

And you want to select all of the records in county 157. What would the query be?

SUBSTRING(“STFID”, 3, 3) = ‘157’
**CHAR_LENGTH – CHAR_LENGTH(string_exp)**

CHAR_LENGTH(“Name”) > 25

- Returns the length of a string.

Suppose you are trying to check that all of your parcel_id codes are exactly 15 digits long. What would your query be?

CHAR_LENGTH(“parcel_id”) <> 14

Suppose you are trying to find any values where the “road_name_full” value is shorter than the “road_name” value?

CHAR_LENGTH(“road_name_full”) < CHAR_LENGTH(“road_name”)
Functions can be nested.

The inner-most function gets interpreted first.

\[
\text{UPPER(SUBSTRING("ROAD\_NAME", 1, 2))} = \text{`CR'}
\]

(*CR indicates County Road in Texas)
MOD – MOD(numeric_exp, divisor)

MOD(“INTERSTATE_NUMBER”, 2) = 0

- Returns the remainder after a division.
- The above expression would find all east/west running interstates. East/west interstates have even numbers. If the interstate number is evenly divisible by 2 (the remainder is zero), this equation evaluates to “true” and the interstate record is selected.
Problem: How can you make index contours stand out?

Use a definition query and the function
Here are the steps:

1. Copy the contour layer.

2. On the copied layer, build a definition query that selects only the index contours.

3. Change the symbol of the index contour layer so that the line is thicker and, if you want, darker.

```sql
mod('ELEVATION', 50) = 0
```
Field Calculator Tips for Data QC and Analysis
There are several languages supported in the ArcGIS Desktop user interface.

For queries:
- SQL

For calculations:
- Python
- VB Script

For labels/display expressions:
- Python
- VBScript
- Java Script
- Text formatting tags
Python and VB have different syntax

<table>
<thead>
<tr>
<th>VBScript Function</th>
<th>Result</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCase(“Field1”)</td>
<td>Returns string in uppercase</td>
<td>Ucase(“Type”)</td>
</tr>
<tr>
<td>Replace(“Field1”,old,new)</td>
<td>Replaces all instances of old substring with new substring</td>
<td>Replace(“Type’,RES’,’COM’)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Python Function</th>
<th>Result</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>!Field1!.upper()</td>
<td>Returns string in uppercase</td>
<td>!Type!.upper()</td>
</tr>
<tr>
<td>!Field1!.replace(old,new)</td>
<td>Replaces all instances of old substring with new substring</td>
<td>!Type!.replace(&quot;COM&quot;,&quot;RES&quot;)</td>
</tr>
</tbody>
</table>
String Concatenation

- **vbScript**
  
  ```vbScript
  "COUNTY_NAME" = "COUNTY" & ' County'
  ```

- **Python**
  
  ```python
  !COUNTY_NAME! = !COUNTY! + " County"
  ```
**String Functions**

- **LEFT (<string>,<number>)**
  
  ```
  "COMPANY_ABREVIATION" = LEFT("COMPANY_CODE",2)
  ```
  
  Returns specified number of characters starting at beginning of string

- **RIGHT (<string>,<number>)**

  ```
  "UNIQUE_ID" = RIGHT("API_NUM",5)
  ```
  
  Returns specified number of characters at end of string

- **MID (<string>,<start>[,<length>])**

  ```
  "SMALL_NAME" = MID("COMPANY",6,5)
  ```
  
  Returns specified substring

- **Python equivalent: slicing!**

  ```
  !Field1!.[start, num_characters]
  ```
  
  Zero-based

  ```
  "COMPANY_ABREVIATION" = !COMPANY_CODE!. [0,2]
  "UNIQUE_ID' = !API_NUM!. [-5]
  ```
String Functions

REPLACE (<string>,<old>,<new>)

!Field1!.replace(<old>,<new>)

“CO_NAME” = REPLACE ("CO_NAME","Exxon","ExxonMobil")

Replaces a portion of a string
Any questions?

Jennifer.Harrison@TeachMeGIS.com