

# Exercise 2

## Querying Tables in ArcGIS Pro



### Introduction

In ArcGIS Pro, searching for values in your data to answer spatial questions and perform spatial analysis is done using Structured Query language (SQL). ArcGIS Pro 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



## Contents

Part 1: Querying geodatabases.....	3
A. Query a file geodatabase .....	3
Part 2: Searching Strings .....	5
A. Search for several values in a field. ....	5
Part 3: Combining Query Expressions.....	6
A. Write an expression with the AND operator. ....	6
B. Write an expression with the OR operator .....	6
C. Write an expression with the NOT and NULL operators .....	7
Part 4: Wildcards .....	8
A. Write an expression using a wildcard .....	8

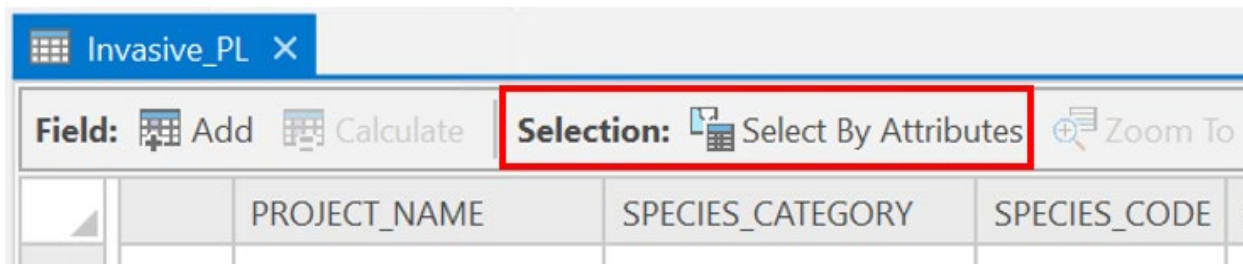
# Part 1: Querying geodatabases

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 Pro that supports a subset of the features and functions available in ArcSDE geodatabases.
- **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.

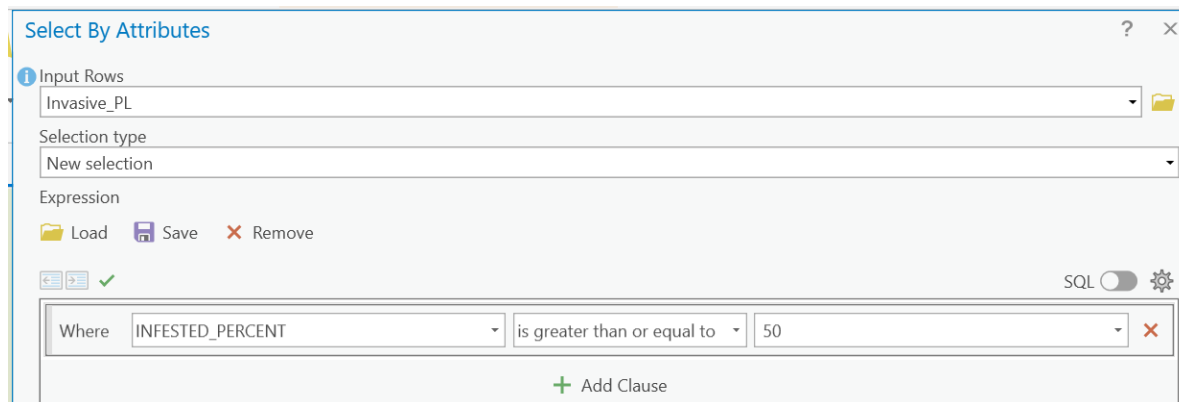
## A. Query a file geodatabase

1. Open the **Tables.aprx** project file if it isn't already.
2. Open the **Invasive\_PL** table.
3. At the top of the **Invasive\_PL** table click **Select By Attributes**.



The SQL menus in the middle of the window help you create the correct syntax for the data you're querying—it lists the field names and values, and also provides the relevant keywords and operators for you.

- i. Create the following query with the menus: Where **INFESTED\_PERCENT** is **greater than or equal to** 50.



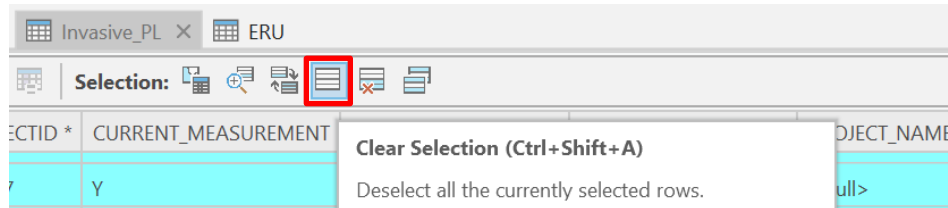
- ii. Open the SQL window by clicking the button to the right of **SQL**. Your query is now shown in SQL syntax.

✓

SQL ☒

INFESTED\_PERCENT >= 50

4. Click **OK**. 3,009 records are now selected.
5. Click the **Clear Selection** button at the top of the table.



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.

## Part 2: Searching Strings

A string is an ordered sequence of symbols/data types. When querying for attribute values, you are essentially using search strings. In the next steps you will build a query with the **IN** operator to search for multiple values within a field

### A. Search for several values in a field.

1. At the top of the **Invasive\_PL** table click **Select By Attributes**.
2. Build the following query: Where **Common\_Name** includes the value(s) **bull thistle,saltcedar,Russian olive**.

The screenshot shows the 'Select By Attributes' dialog box. The 'Where' field is set to 'COMMON\_NAME', the operator is 'includes the value(s)', and the values are 'bull thistle,saltcedar,Russian olive'. The 'SQL' button is highlighted.

3. Click the **SQL button** and examine the syntax. Notice the **IN** operator is used for this query.

The screenshot shows the SQL query editor with the query: `COMMON_NAME IN ('bull thistle', 'saltcedar', 'Russian olive')`.

4. Click **OK**. 870 records are now selected.
5. **Clear** the selection.

## 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 by using the **AND** operator. Next, we will build an expression in which one of the criteria must be true by using the **OR** operator. Finally, we will build an expression with the **NOT** and **NULL** operators.

### A. Write an expression with the AND operator.

1. At the top of the **Invasive\_PL** table click **Select By Attributes**.
2. Build this query: Where **SPECIES\_CATEGORY** is equal to **Plant**.
3. Click **Add Clause**.
4. Add this query: **And INFESTED\_AREA is greater than 1**.

5. Examine the SQL syntax.
6. Click **OK**.
7. Inspect the result.
8. **Clear** the selection.

### B. Write an expression with the OR operator

1. At the top of the **Invasive\_PL** table click **Select By Attributes**.
2. Build this query: Where **INVENTORY\_DATE** is on or after **1/1/2010** Or **OWNER\_NAME** is equal to **FS**.

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.

3. Examine the SQL syntax.
4. Click **OK**.
5. Inspect the result.
6. **Clear** the selection.

### C. Write an expression with the NOT and NULL operators

**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.

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**.

1. At the top of the **Invasive\_PL** table click **Select By Attributes**.
2. Build this query: **SPECIES\_CATEGORY** is equal to **Vertebrate** And **FS\_Unit\_Name** is not equal to **CARSON NATIONAL FOREST** and **ERU** is not null.

Where	SPECIES_CATEGORY	is equal to	Vertebrate	✖
And	FS_UNIT_NAME	is not equal to	CARSON NATIONAL FOREST	✖
And	ERU	is not null		✖
+ Add Clause				

3. Examine the SQL syntax.
4. Click **OK**.
5. Inspect the result. There should be **4** records selected.
6. **Clear** the selection.

## 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 require the **LIKE** operator.

### A. Write an expression using a wildcard

1. At the top of the **Invasive\_PL** table click **Select By Attributes**.

Suppose you are searching a scientific name that you know begins with “carduus” but you are unsure how the rest of it is spelled.

2. **Build** this query: **SCIENTIFIC\_NAME** contains the text **carduus**.

3. Examine the SQL syntax. Notice the **LIKE** operator and the % symbols on both sides of the search string.

```
SCIENTIFIC_NAME LIKE '%carduus%'
```

4. Click **Apply**. There should be no records selected. That’s because SQL queries for file geodatabases are case sensitive. To avoid this problem you can omit the first letter from the search string.
5. Run the same query again but without the first letter in the word “carduus”:  
**SCIENTIFIC\_NAME** contains the text **arduus**.

```
SCIENTIFIC_NAME LIKE '%arduus%'
```

- 177 records are selected now.

6. **Clear** the selected records.

Congratulations on finishing this exercise!