Queries & SQL - "Beyond the Basics"

March 30, 2023

Agenda

Search Tool Query Tips
Running Queries in the Public Website
SQL JOIN statement dissected
Sources of GG information

Today's Goals

1. Learn ways to document your queries
2. Create SQL queries using JOIN (involving multiple tables)
3. Use SQL resources at hand as a starting point
4. Understand different operators and SQL reserved words

Why Learn SQL?

Why Learn SQL?

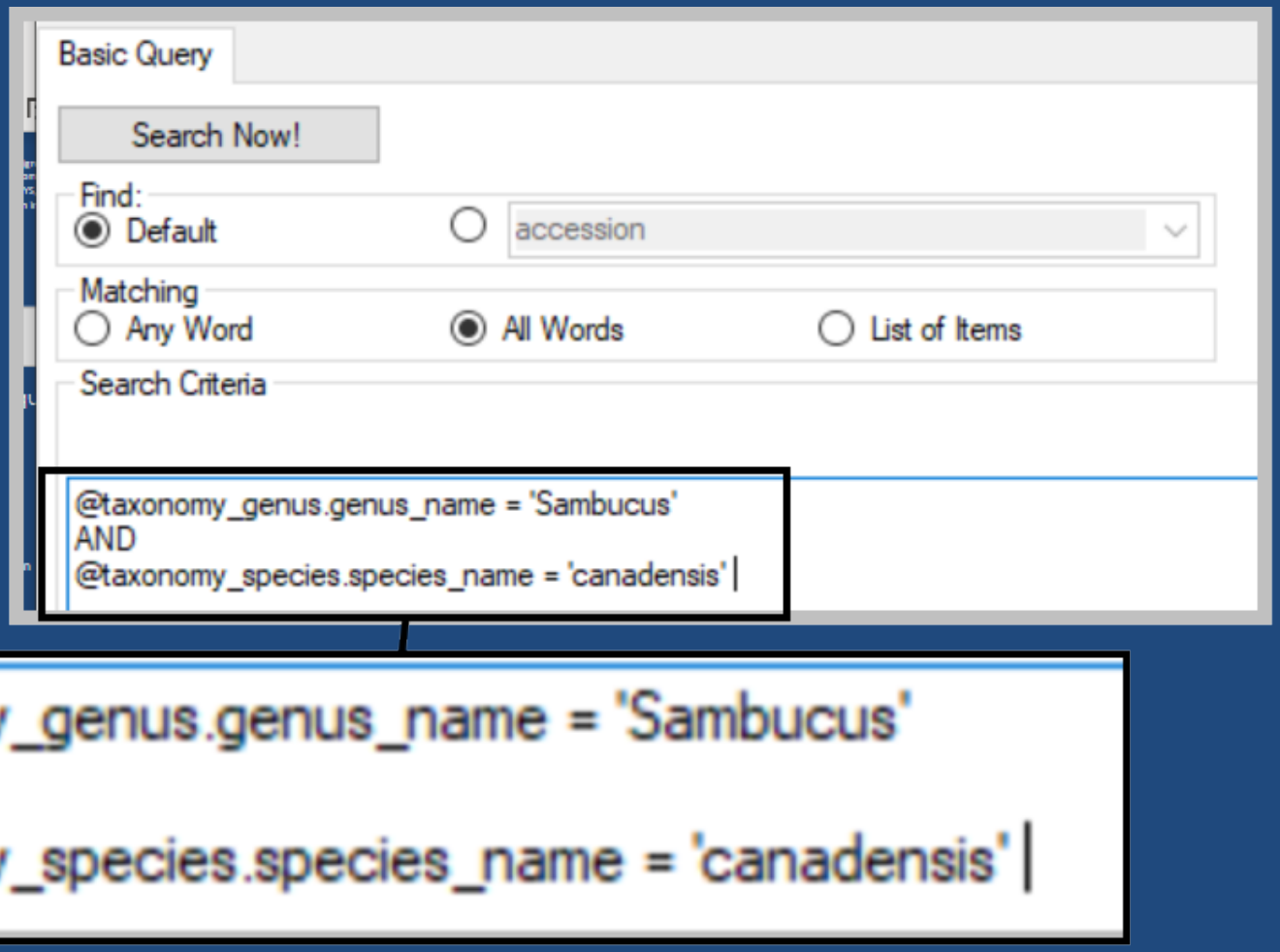
An email I received today:

"…I work with oat pedigree, and I wonder if you can do me a favour and pull data from your GG. I have pulled all oat pedigrees from GENESYS, but this query would make it easier to get all the names when inventory names >1 …"

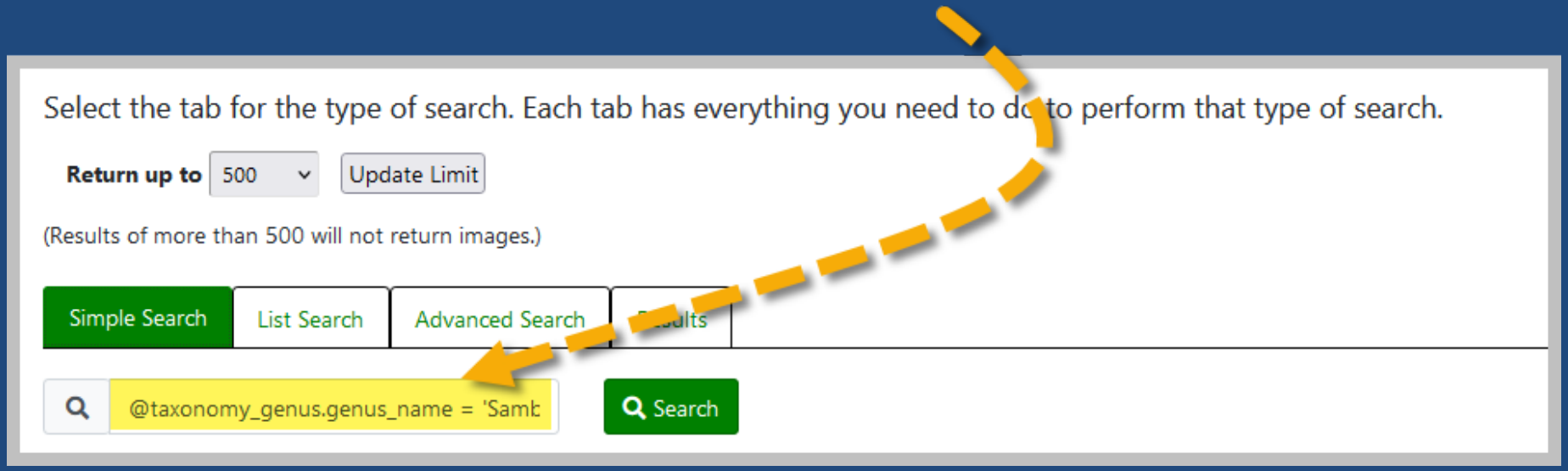
Any DQ* questions?

*Not Dairy Queen

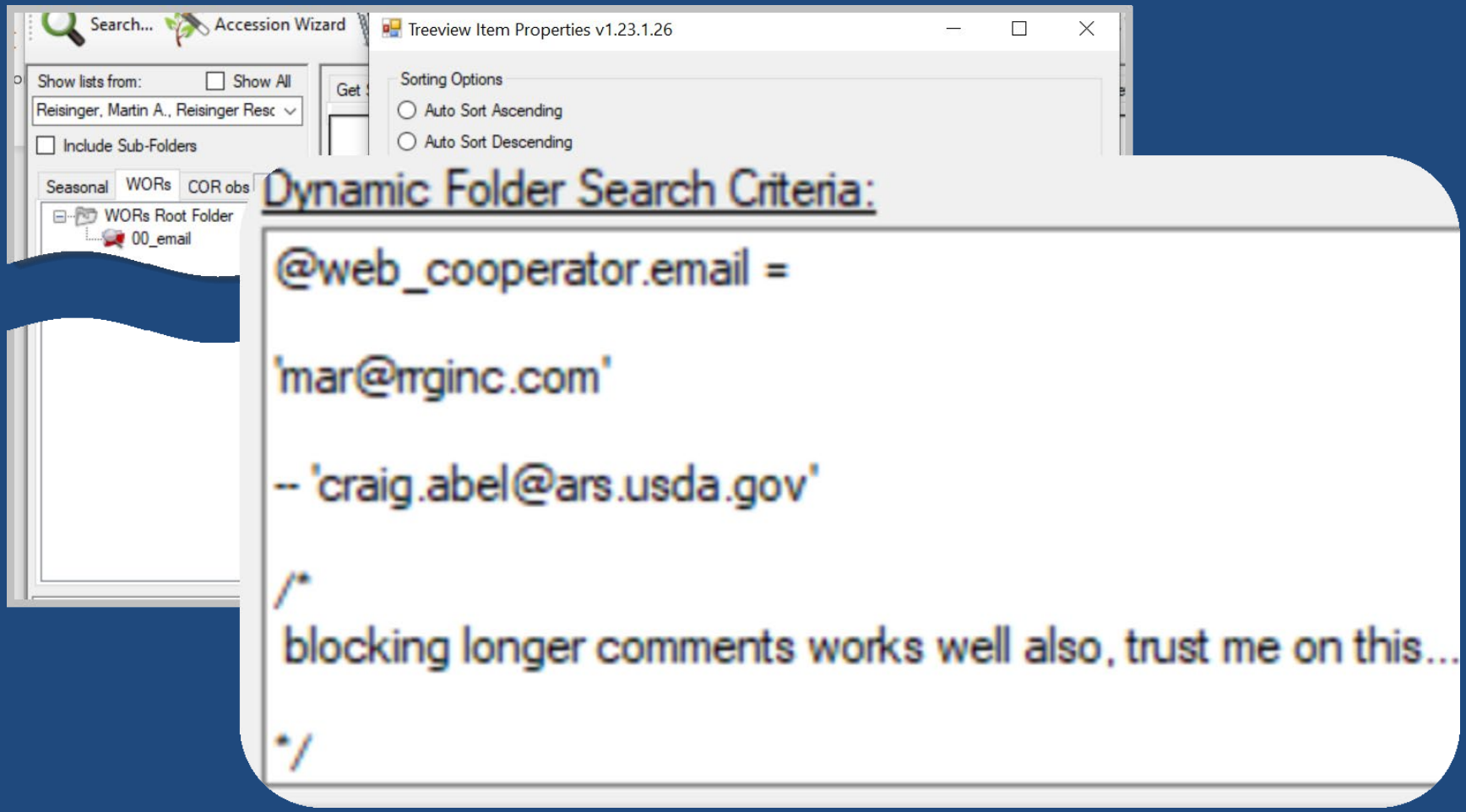
Reminder


Basic Query
Search Now!
Find:
Default
accession
Matching
Any Word
All Words
List of Items
Search Criteria
@taxonomy_genus.genus_name = 'Sambucus'
AND
@taxonomy_species.species_name = 'canadensis'
@taxonomy_genus.genus_name = 'Sambucus'
AND
@taxonomy_species.species_name = 'canadensis'

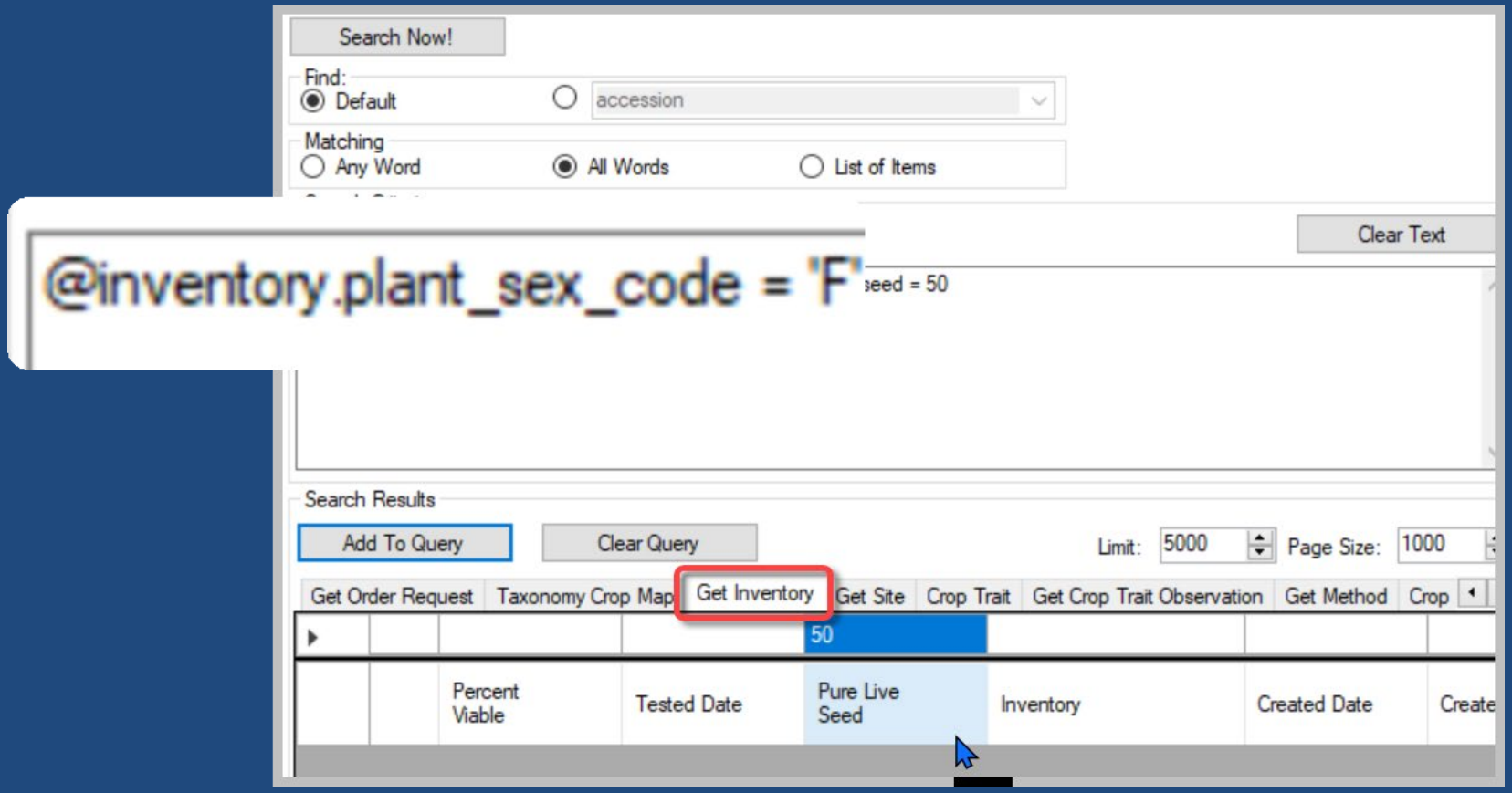
…Reminder


Select the tab for the type of search. Each tab has everything you need to do to perform that type of search.
Return up to 500
Update Limit
(Results of more than 500 will not return images.)
Simple Search
List Search
Advanced Search
@taxonomy_genus.genus_name = 'Samb
Search

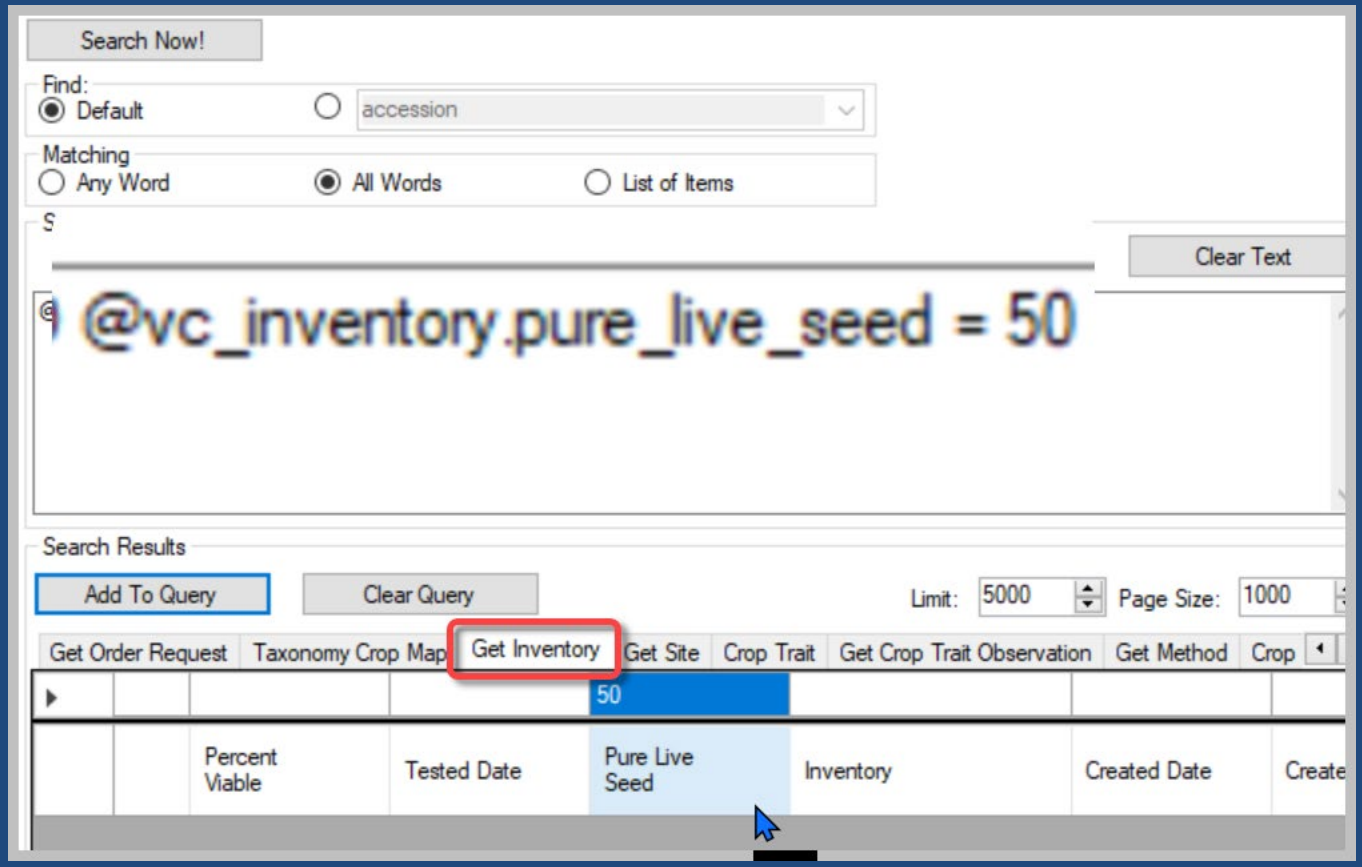
Document your stuff!



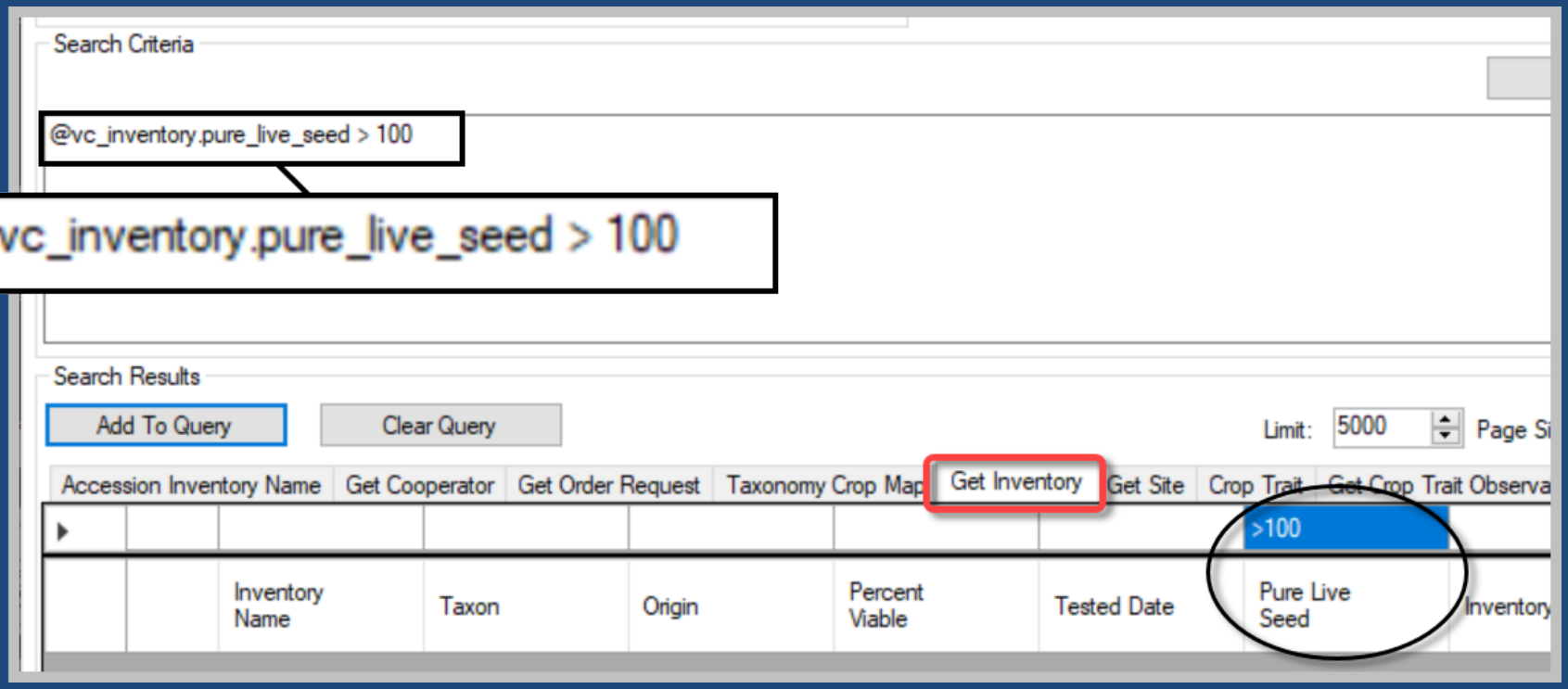
Use ST to obtain DB info



Use ST to obtain DB info



vc_inventory ?



vc_inventory

vc_inventory is a database view that appears just like a table to any SQL SELECT query

Use SQL to obtain DB info …demo

```
SELECT
sdf.field_name,
sdf.sort_order,
sd.dataview_name,
sd. database_area_code,
st.table_name,
stfl.title,
stfl.description
  FROM sys_dataview sd
  JOIN sys_dataview_field sdf
        ON sd.sys_dataview_id = sdf.sys_dataview_id
  LEFT JOIN sys_table_field stf
        ON sdf.sys_table_field_id = stf.sys_table_field_id
  LEFT JOIN sys_table st
        ON stf.sys_table_id = st.sys_table_id
  LEFT JOIN sys_table_field_lang stfl
        ON stf.sys_table_field_id = stfl.sys_table_field_id
        AND  stfl.sys_lang_id = 1
WHERE
  sd.category_code = 'Client'
/*remove and clause to see all */
AND
sd.dataview_name ='get_accession'

ORDER BY  sd.dataview_name, sdf.sort_order, sdf.field_name
```

Super Tip!

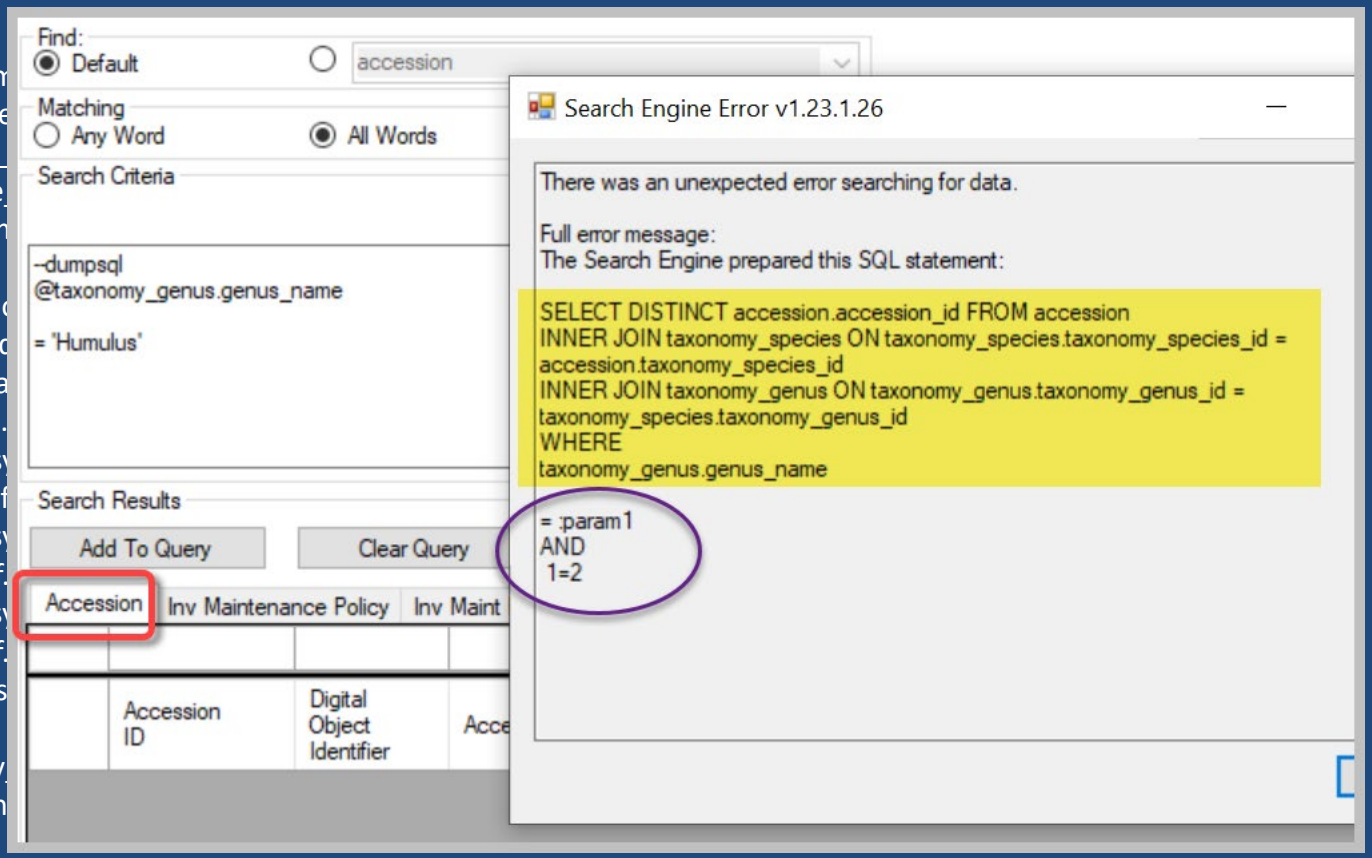
```
--dumpsql
@taxonomy_genus.genus_name = 'Humulus'
```

The “--dumpsql” comment is a debug tool that makes the ST throw an error and shows the SQL the query would use

Must be 2 hyphens on the *first* line

Use SQL to obtain DB info

```
SELECT
sdf.field_nam
sdf.sort_orde
sd.dataview_
sd. database_
st.table_nam
stfl.title,
stfl.descriptio
 FROM sys_d
 JOIN sys_da
        ON sd.
 LEFT JOIN sy
        ON sdf
 LEFT JOIN sy
        ON stf.
 LEFT JOIN sy
        ON stf.
        AND s
WHERE
 sd.category_
/*remove an
AND
sd.dataview_name ='get_accession'

ORDER BY  sd.dataview_name, sdf.sort_order, sdf.field_name
```



--get list of client dataviews

```
SELECT *
FROM sys_dataview sd
```

--get list of client dataviews

```
SELECT *
FROM sys_dataview sd
WHERE
category_code = 'Client'
```

-- get list of Web Report's

```
SELECT dataview_name, database_area_code,
title, description
FROM sys_dataview sd
JOIN sys_dataview_lang sdl ON
sdl.sys_dataview_id = sd.sys_dataview_id
WHERE category_code = 'Web Reports'
ORDER BY 2,3
```

```
/* display a dataview's SQL
*/

SELECT sd.dataview_name
, REPLACE(sql_statement, CHAR(13) +
CHAR(10), CHAR(60)+'br'+CHAR(62))  AS
html_sql
FROM sys_dataview sd
JOIN sys_dataview_sql sds ON
sds.sys_dataview_id = sd.sys_dataview_id
WHERE sd.dataview_name =
    'web_qry_crop_traits_and_codes'

AND database_engine_tag = 'sqlserver'
```

…demo

- Use the resulting code and edit

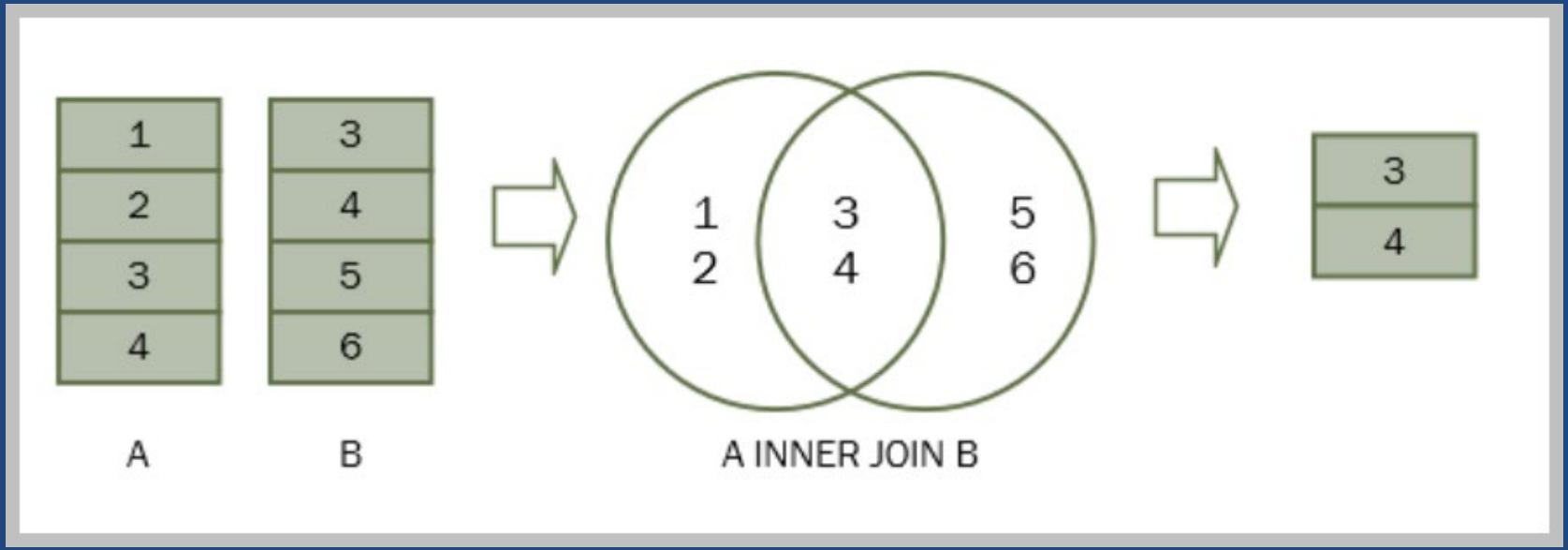
SQL Statements: Basic Components

SELECT – what columns (fields) / data to display

FROM – what table(s) to search

WHERE – what criteria to specify

(Inner) JOIN



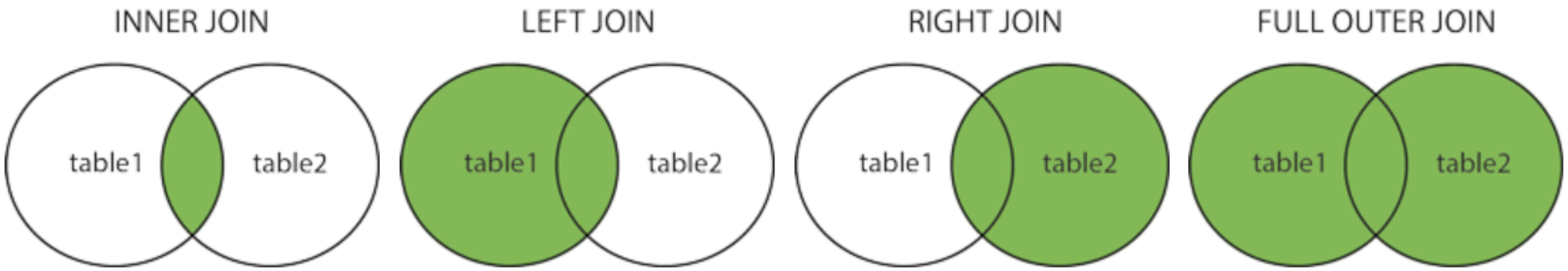
https://www.sqltutorial.org/sql-inner-join/

JOINS

Different Types of SQL JOINs

Here are the different types of the JOINs in SQL:

- **(INNER) JOIN**: Returns records that have matching values in both tables
- **LEFT (OUTER) JOIN**: Return all records from the left table, and the matched records from the right table
- **RIGHT (OUTER) JOIN**: Return all records from the right table, and the matched records from the left table
- **FULL (OUTER) JOIN**: Return all records when there is a match in either left or right table



JOIN & ALIASES

Alias is an alternative name for either a table (or a field)

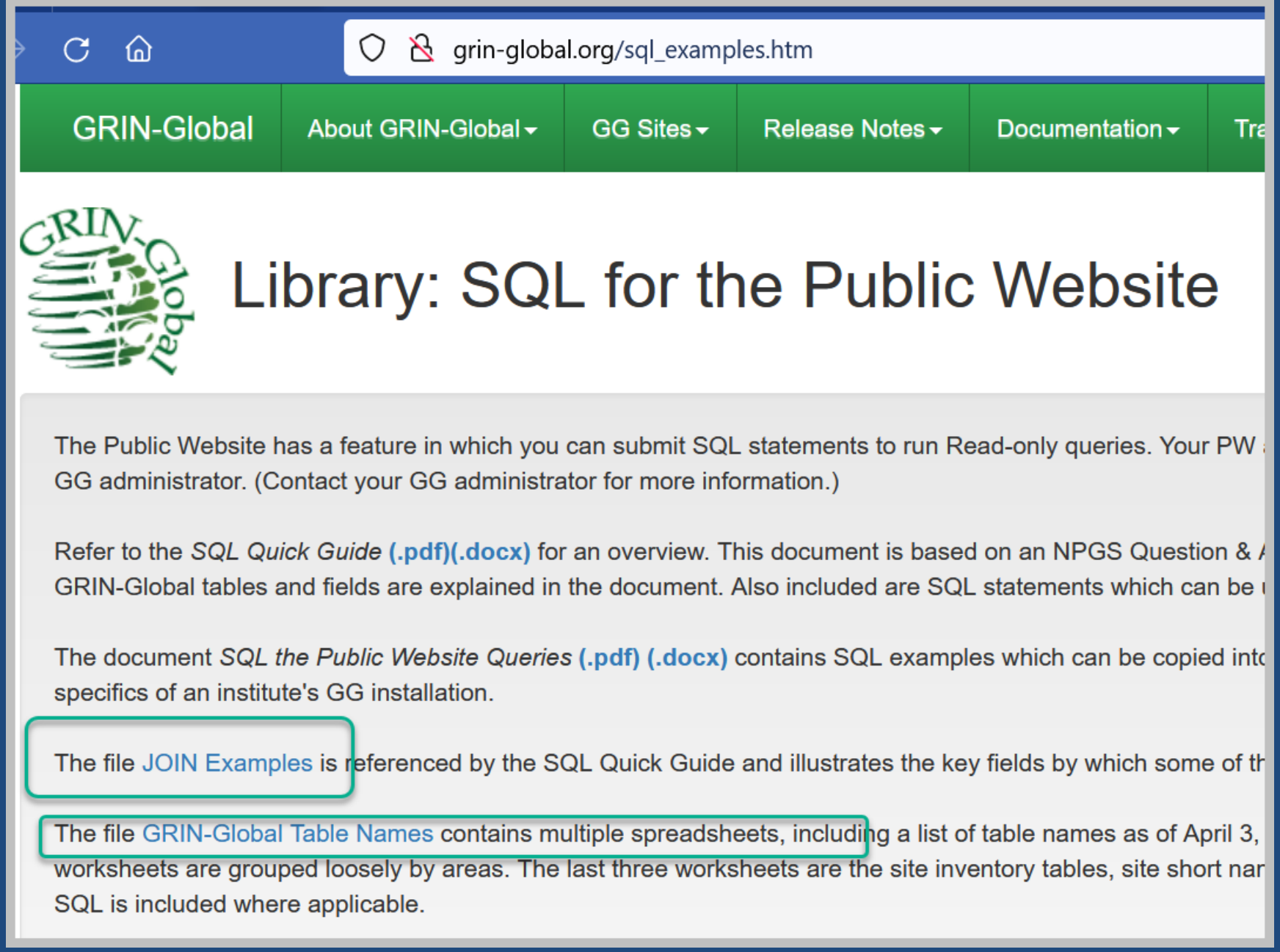
```
SELECT
a.accession_number_part1, a.accession_number_part2,
a.accession_number_part3,
ts.name

FROM taxonomy_species ts
JOIN accession  a  ON ts.taxonomy_species_id = a.taxonomy_species_id

WHERE ts.name LIKE 'Trit%'
   AND  a.status_code = 'ACTIVE'
```

JOINS

“The first table mentioned is the left side and the second table is the right. When you’re joining from parent to child (FROM parent JOIN child ON…), the parent is the left side.

… Whenever I’m joining in the reverse direction from parent to child, I’m usually focusing on the children so an INNER JOIN is fine because GG doesn’t have parentless children.”

-- a SQL guru


grin-global.org/sql_examples.htm
GRIN-Global
About GRIN-Global
GG Sites
Release Notes
Documentation
Library: SQL for the Public Website
The Public Website has a feature in which you can submit SQL statements to run Read-only queries. Your PW
GG administrator. (Contact your GG administrator for more information.)
Refer to the SQL Quick Guide (.pdf)(.docx) for an overview. This document is based on an NPGS Question &
GRIN-Global tables and fields are explained in the document. Also included are SQL statements which can be
The document SQL the Public Website Queries (.pdf) (.docx) contains SQL examples which can be copied int
specifics of an institute's GG installation.
The file JOIN Examples is referenced by the SQL Quick Guide and illustrates the key fields by which some of th
The file GRIN-Global Table Names contains multiple spreadsheets, including a list of table names as of April 3,
worksheets are grouped loosely by areas. The last three worksheets are the site inventory tables, site short nar
SQL is included where applicable.

Example

--dumpsql

@inventory.availability_status_code IS NOT NULL

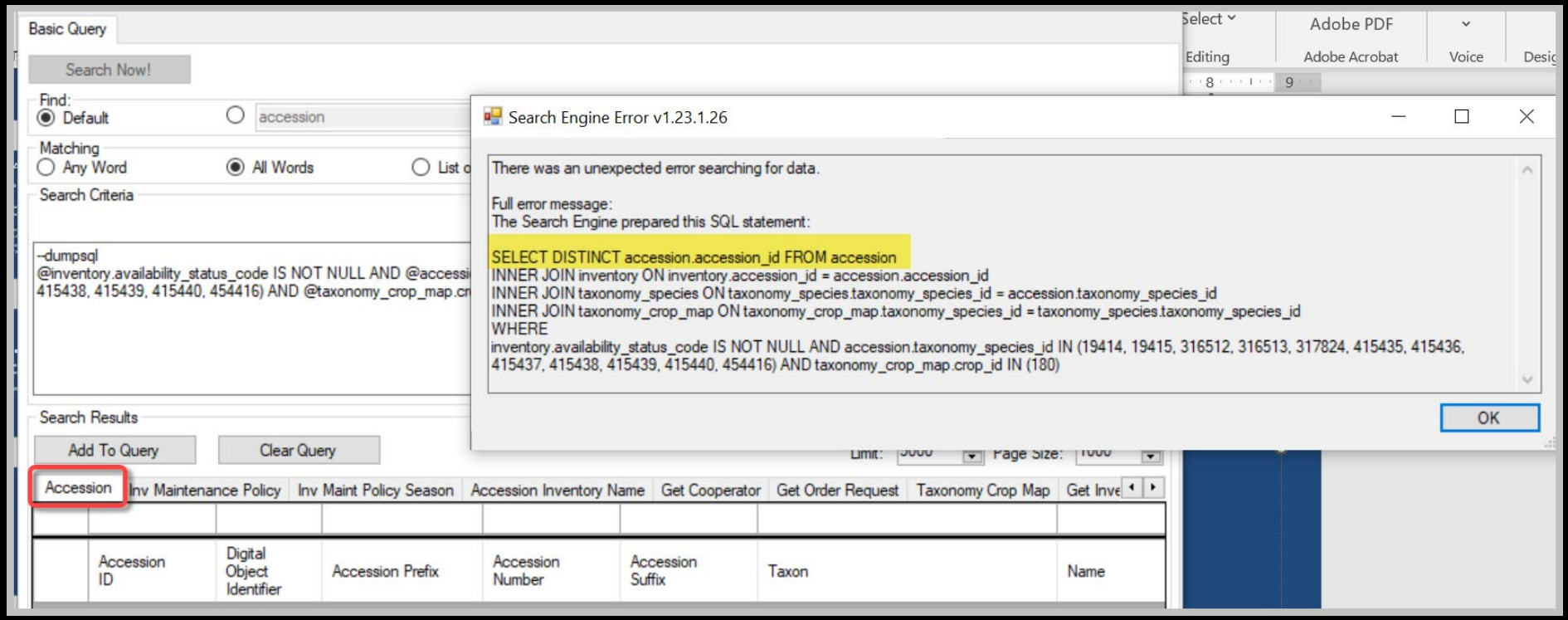
AND

@accession.taxonomy_species_id IN (19414, 19415, 316512, 316513, 317824, 415435, 415436, 415437, 415438, 415439, 415440, 454416)

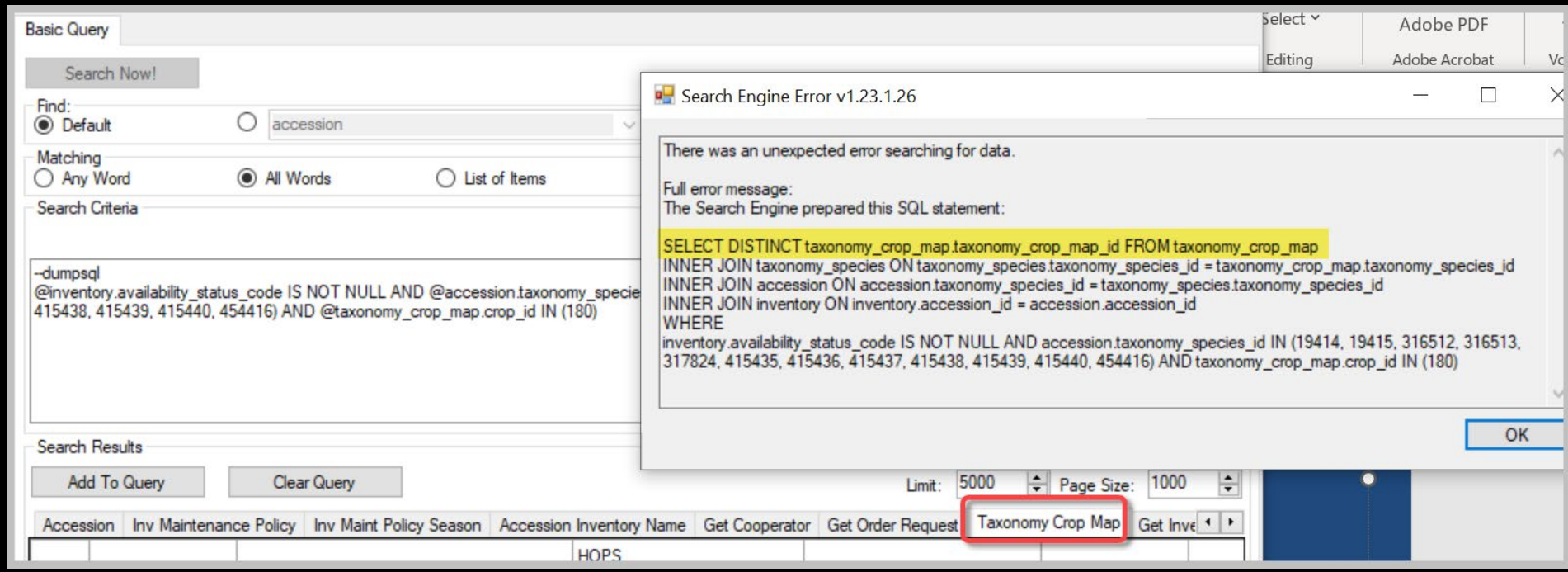
AND

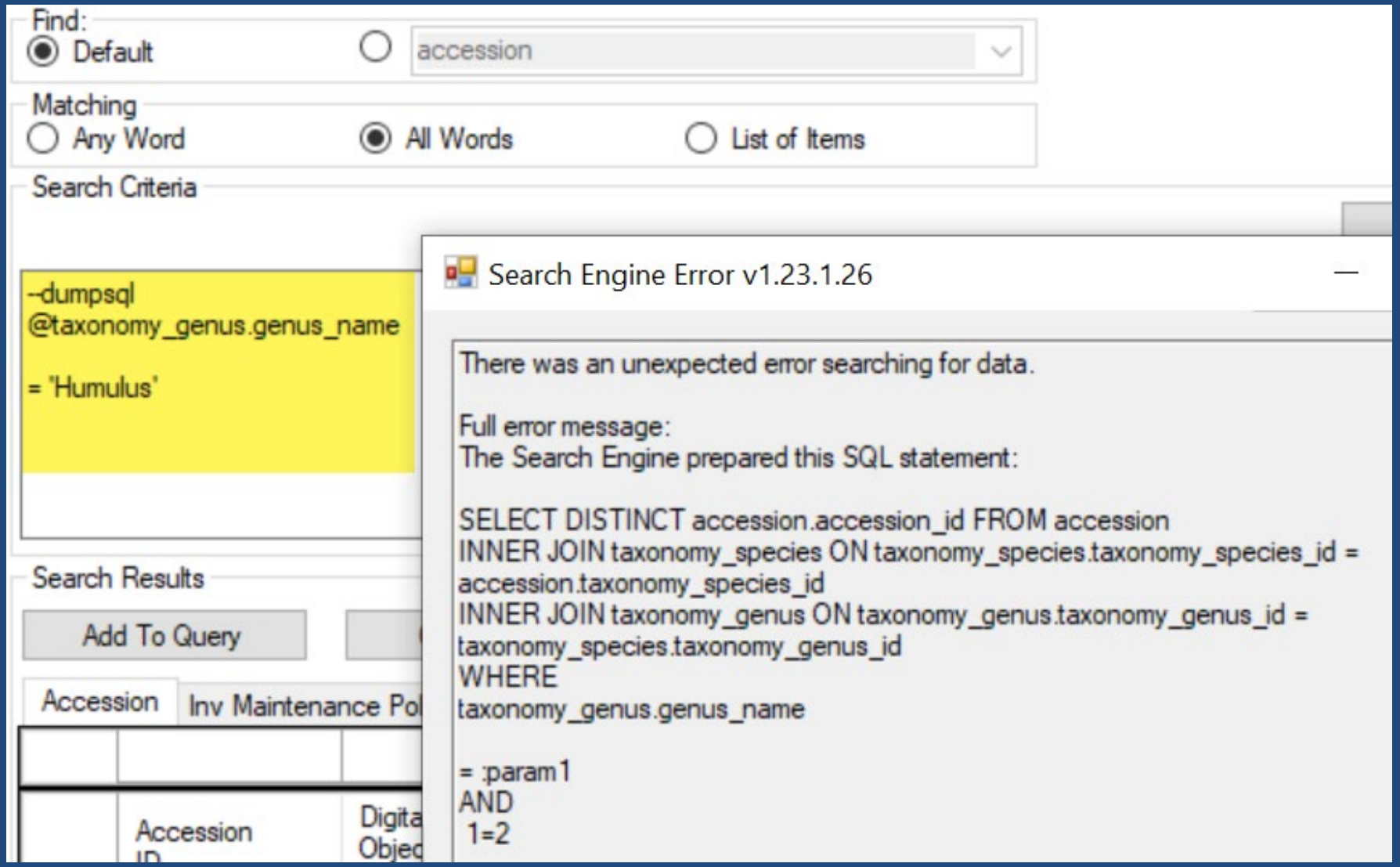
@taxonomy_crop_map.crop_id IN (180)

Where you are matters



Where you are matters




Find:
Default
accession
Matching
Any Word
All Words
List of Items
Search Criteria
--dumpsql
@taxonomy_genus.genus_name
= 'Humulus'
Search Engine Error v1.23.1.26
There was an unexpected error searching for data.
Full error message:
The Search Engine prepared this SQL statement:
SELECT DISTINCT accession.accession_id FROM accession
INNER JOIN taxonomy_species ON taxonomy_species.taxonomy_species_id = accession.taxonomy_species_id
INNER JOIN taxonomy_genus ON taxonomy_genus.taxonomy_genus_id = taxonomy_species.taxonomy_genus_id
WHERE
taxonomy_genus.genus_name
= :param1
AND
1=2
Search Results
Add To Query
Accession
Inv Maintenance Pol
Accession

= param#

Used to avoid SQL injection attacks. The SE automatically parameterizes the SQL before executing it.

The query

```
@taxonomy_genus.genus_name = 'Humulus'
```

Is translated before execution to this SQL

```
@taxonomy_genus.genus_name = :param1
```

All the literal strings are replaced by parameters the SQL is formed for execution. The strings are sent as a separate structure that is won't be executed as SQL. This prevents attacks by a user sneaking SQL commands into data such as a genus name.

1=2

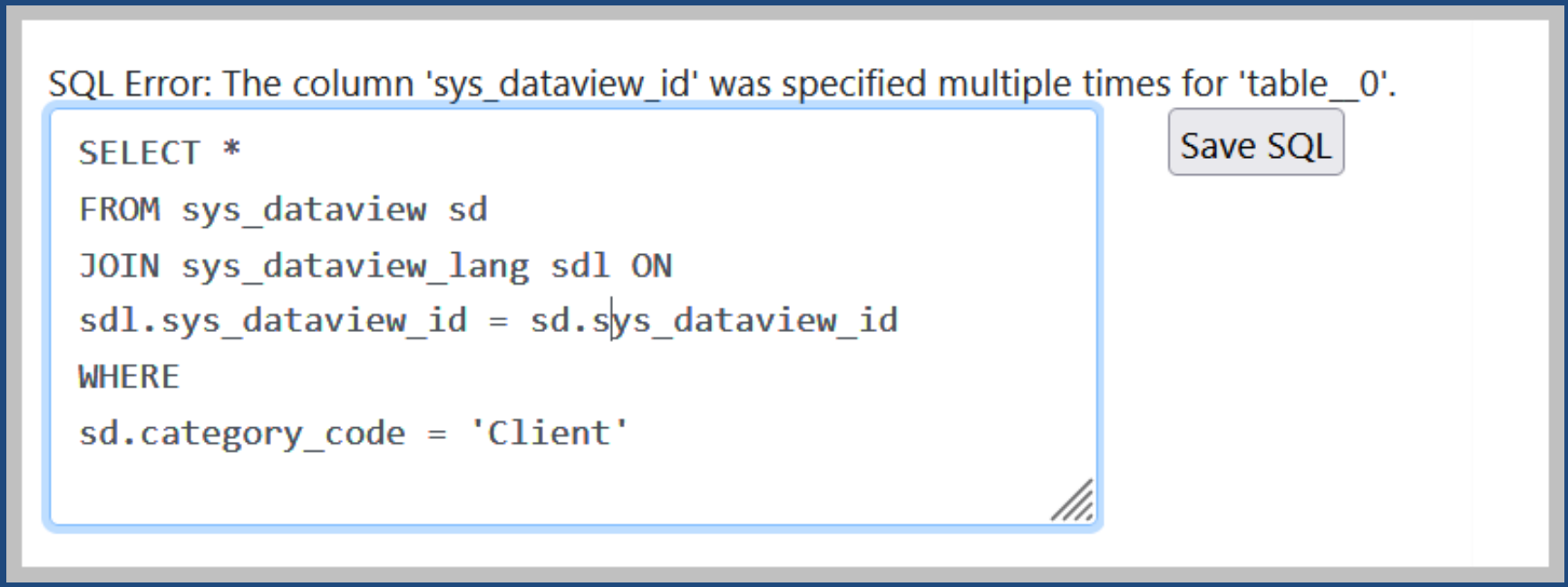
```
SELECT DISTINCT
accession.accession_id FROM accession
WHERE
accession.accession_number_part1 =
:param1
AND
 1=2
```

1=2

```
SELECT DISTINCT
accession.accession_id FROM accession
WHERE
accession.accession_number_part1 =
:param1
AND
 1=2
```

When unknown text cannot be found, the autosearch inserts the SQL expression "1=2" (simply translated, means “false”)

A common error message…



Debugging Example

```
SELECT
a.accession_number_part1,
a.accession_number_part2,
a.accession_number_part3,
plant_name,
ts.name
FROM taxonomy_species ts
JOIN accession a ON ts.taxonomy_species_id =
a.taxonomy_species_id
JOIN inventory i ON a.accession_id = i.accession_id
JOIN accession_inv_name invn ON invn.inventory_id = i.inventory_ijd
WHERE ts.name LIKE 'Sorghum%'
AND a.status code = 'ACTIVE'
```

DISTINCT

```
SELECT su.user_name, sg.group_tag, sp.permission_tag, st.table_name,
CONCAT(c.first_name, ' ', c.last_name) AS owner
FROM sys_user su
JOIN sys_group_user_map sgum ON sgum.sys_user_id = su.sys_user_id
JOIN sys_group sg ON sg.sys_group_id = sgum.sys_group_id
JOIN sys_group_permission_map sgpm ON sgpm.sys_group_id = sg.sys_group_id
JOIN sys_permission sp ON sp.sys_permission_id = sgpm.sys_permission_id
LEFT JOIN sys_table st ON st.sys_table_id = sp.sys_table_id
JOIN cooperator c ON c.cooperator_id = su.cooperator_id
WHERE
sg.group_tag  LIKE  'MANAGE_SITE%'
AND c.site_id = 3
/* omit line above to display all sites/cooperators */
/* or search for...  MANAGE_COOPERATOR% */
ORDER BY sp.permission_tag, c.last_name
```

COUNT() Syntax

SELECT COUNT(*column_name*)
FROM *table_name*
WHERE *condition*;

COUNT Example

```
SELECT
   COUNT(*) AS Active_Accessions
FROM    accession a
WHERE  status_code = 'ACTIVE'
```

Sources

Check out:

- https://www.grin-global.org/sql_examples.htm
- https://www.w3schools.com/sql/default.asp
- https://www.sqltutorial.org/sql-where/
- https://www.tutorialspoint.com/sql/sql-operators.htm