Searching in GRIN-Global

Revision Date
June 26 2025

The Search Engine has evolved in GRIN-Global. This documentation refers to the Search Engine used in server release 1.9.9.2 or higher.

Please send any questions related to marty.reisinger@usda.gov.

The Appendix contains change notes pertaining to this document.

Author
Martin Reisinger

Major Contributor
Kurt Endress

TOC

Search Engine and Search Tool Overview

Overview

Currently, the main two GRIN-Global (GG) user applications are the Curator Tool and the Search Tool. This document differentiates between the "Search *Tool*" and the "Search *Engine*." The Search Tool is the application which a genebank staff person uses to communicate with the search engine. The search engine is the logic/program that queries the database and returns matching records. Besides the search tool application, the GG public website also uses the search engine. The GG search engine has evolved since GG started and periodically is updated with additional enhancements.

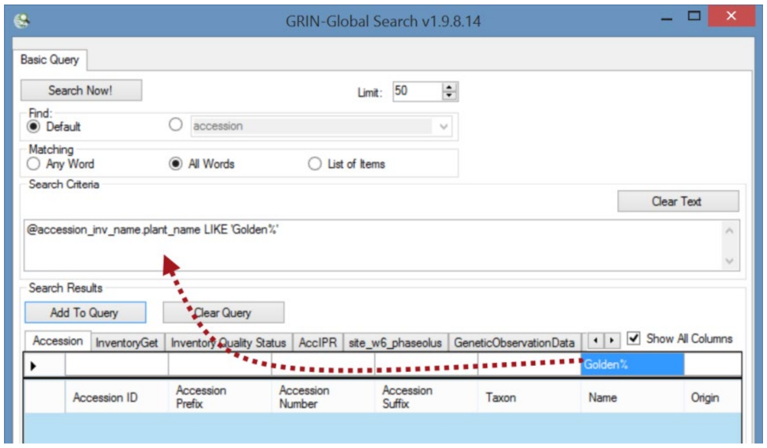
SQL Server Full-Text Indexing

At the U.S. National Plant Germplasm System (NPGS), Microsoft SQL's Full-Text Indexing feature was implemented simultaneously with server release 1.9.9.2. SQL Server Full-Text Indexing allows searching for single words in large text fields (such as notes) without specifying wildcards.

The GG DBA should consider implementing Microsoft SQL's Full-Text Indexing feature. Microsoft documentation is available on the internet.

Several notes:

- Search Tool & Public Website use the same search *engine*, but the PW search capabilities are supplemented by some PW code
- The Search Tool is a stand-alone program
- The Search Tool has two distinct modes
    - Text Box
    - Query By Example grid ("QBE") Recommended ***method***



Limit: 50

Find: [✓ Default] accession

Matching: Any Word [✓ All Words] List of Items

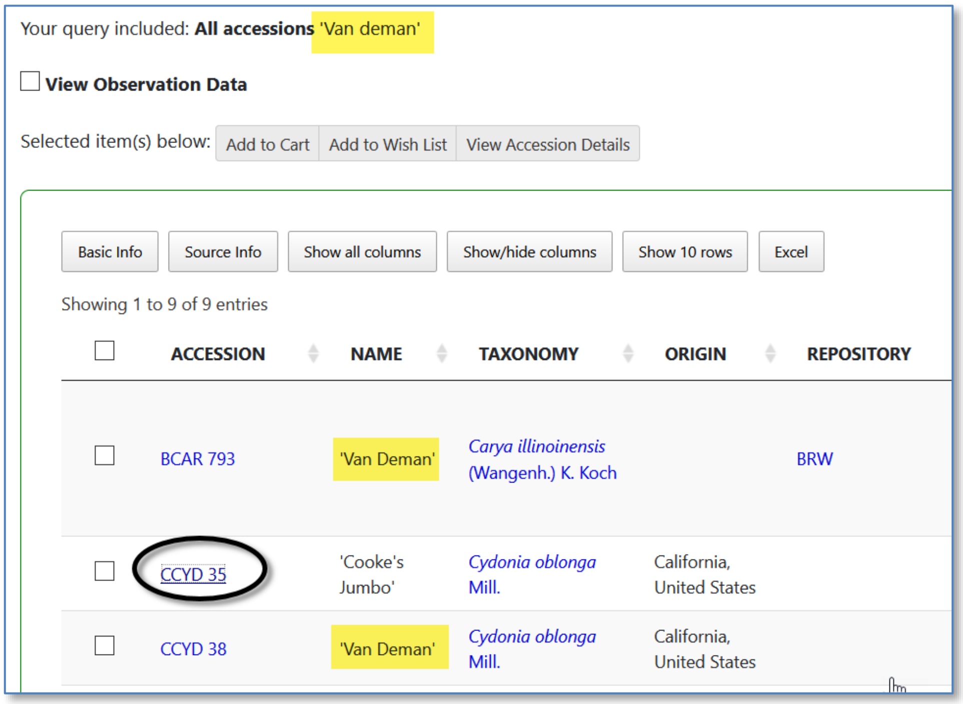
```
@accession_inv_name.plant_name LIKE 'Golden%'
```

[✓ Show All Columns]

Accession ID	Accession Prefix	Accession Number	Accession Suffix	Taxon	Name	Origin
					Golden%	

Think of the search engine as using a "wide net." At first glance, it may not be obvious why some records are returned by the search. The "odd" results are most likely due to the search finding matches in multiple fields.

In the following example, the search string was “Van deman” In the search results, it is obvious why the first and third accessions are listed, but why the second?



Your query included: **All accessions** 'Van deman'

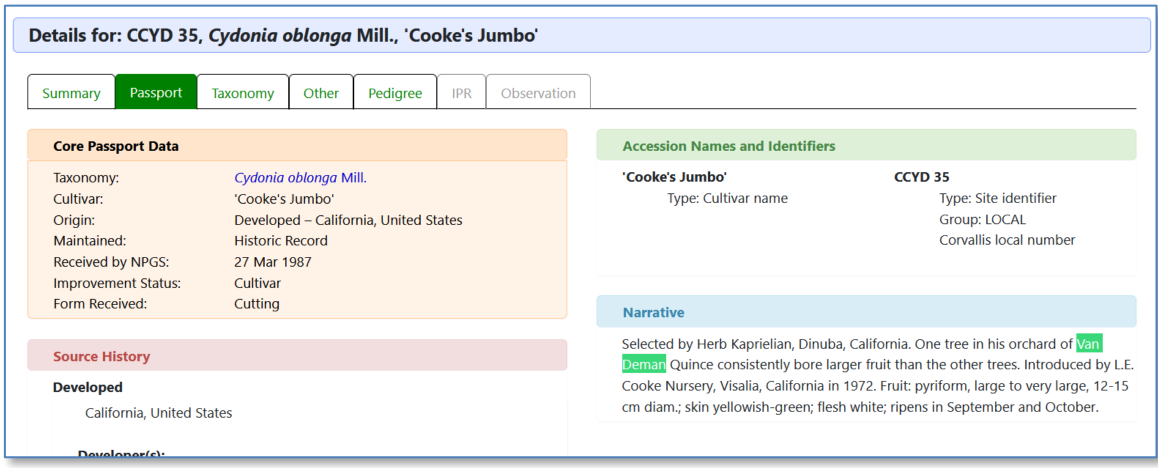
☐ **View Observation Data**

Selected item(s) below:

Showing 1 to 9 of 9 entries

☐	ACCESSION	NAME	TAXONOMY	ORIGIN	REPOSITORY
☐	BCAR 793	'Van Deman'	*Carya illinoinensis* (Wangenh.) K. Koch		BRW
☐	CCYD 35	'Cooke's Jumbo'	*Cydonia oblonga* Mill.	California, United States	
☐	CCYD 38	'Van Deman'	*Cydonia oblonga* Mill.	California, United States	

Looking at the accession’s details, the Narrative mentions “Van Deman.” On the PW, the Narrative comes from the Accession **Note** field, and in this database, that field was one that the DBA had indexed – hence it was searched.



Details for: CCYD 35, *Cydonia oblonga* Mill., 'Cooke's Jumbo'

Core Passport Data

Taxonomy:	*Cydonia oblonga* Mill.
Cultivar:	'Cooke's Jumbo'
Origin:	Developed – California, United States
Maintained:	Historic Record
Received by NPGS:	27 Mar 1987
Improvement Status:	Cultivar
Form Received:	Cutting

Source History

Developed

California, United States

Developer(s):

Accession Names and Identifiers

'Cooke's Jumbo'
Type: Cultivar name

CCYD 35
Type: Site identifier
Group: LOCAL
Corvallis local number

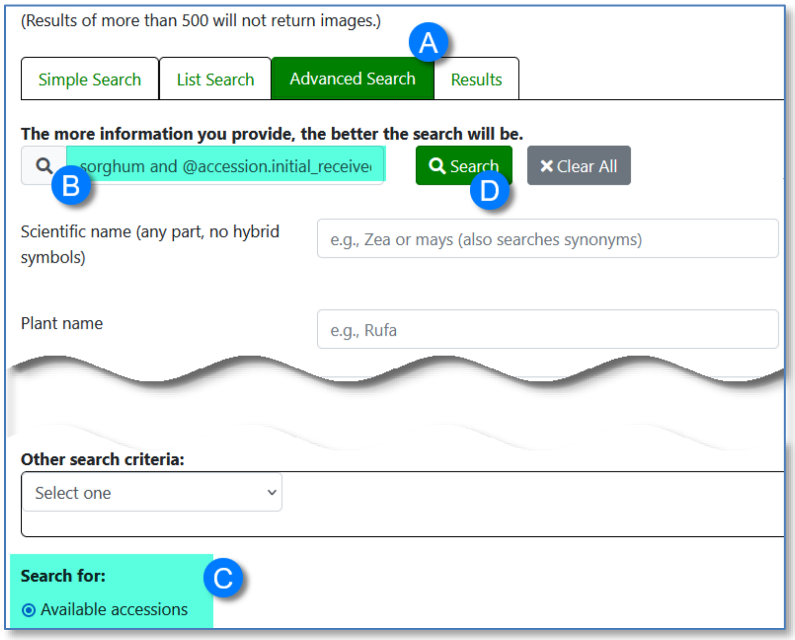
Narrative

Selected by Herb Kaprielian, Dinuba, California. One tree in his orchard of Van Deman Quince consistently bore larger fruit than the other trees. Introduced by L.E. Cooke Nursery, Visalia, California in 1972. Fruit: pyriform, large to very large, 12-15 cm diam.; skin yellowish-green; flesh white; ripens in September and October.

Using Search Text in the **Public Website**

The Public Website search can handle text in the search when constructed properly, such as the following:

sorghum and @accession.initial_received_date > '2020'


(Results of more than 500 will not return images.)

A
Simple Search
List Search
Advanced Search
Results
The more information you provide, the better the search will be.
sorghum and @accession.initial_receive
B
Search
D
Clear All
Scientific name (any part, no hybrid symbols)
e.g., Zea or mays (also searches synonyms)
Plant name
e.g., Rufa
Other search criteria:
Select one
Search for:
C
Available accessions

In this example, the user was looking for sorghum accessions that were recently added to the collection (after 2020). More on this in the section Public Website Searches Using the @

Search Engine Evolution - Enhancements in Server Release 1.9.9.2

The Search Engine (SE) has evolved, and in server release 1.9.9.2, the functionality has expanded. For example, the search engine now has extended SQL support. This SE fixes many of the issues between the PW and SE regarding visible, active, and available (status) check boxes. The latest version of the SE implemented a completely new way for the PW to filter by these status values. Other changes include:

- Speed increases on simple searches
- Full text indexing
- **List of Items** Change
- Changes to Public Website queries
- Extended SQL Support to additional key words: BETWEEN, EXCEPT, UNION, INTERSECT, NOT IN, …

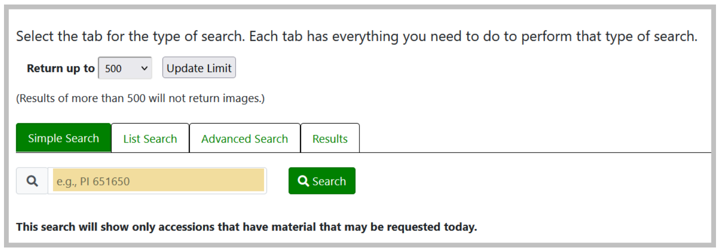
Speed Improvements

Checking for web visibility or availability was slowing down simple PW searches such as **PI 500000** because the search's formatted section might be only @accession.is_web_visible = 'Y' -- this would return 800,000 results, taking a few seconds to complete, whereas now the SE examines the freeform section first and converts the results into a criteria to combine with the formatted section.

Public Website

Basically…

the **Search For** box on the Public Website is equal to the text box in the Search Tool.

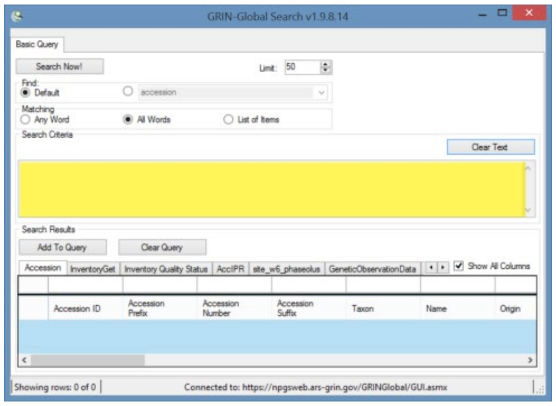

Select the tab for the type of search. Each tab has everything you need to do to perform that type of search.

Return up to 500 Update Limit

(Results of more than 500 will not return images.)

This search will show only accessions that have material that may be requested today.

=



Limit: 50

Find: [✓ Default] accession

Matching: Any Word [✓ All Words] List of Items

Search Criteria

[✓ Show All Columns]

Showing rows: 0 of 0

Connected to: https://npgsweb.ars-grin.gov/GRINGlobal/GUI.asmx

The difference is that in some cases the Public Website uses additional logic to handle the **is_web_visible** flags and other issues specific to the PW.

There are three levels of sort on the output of the public website searches:

1. The highest weighted field is found first (genus hits before others)
2. Accessions with PI prefixes before Non-PIs*
3. Most-recently received accessions are found first

Organizations other than NPGS that are running GRIN-Global may set the preferred prefix from “PI” to their organizations preferred prefix. Notes for administrators relevant to this are in Appendix B.

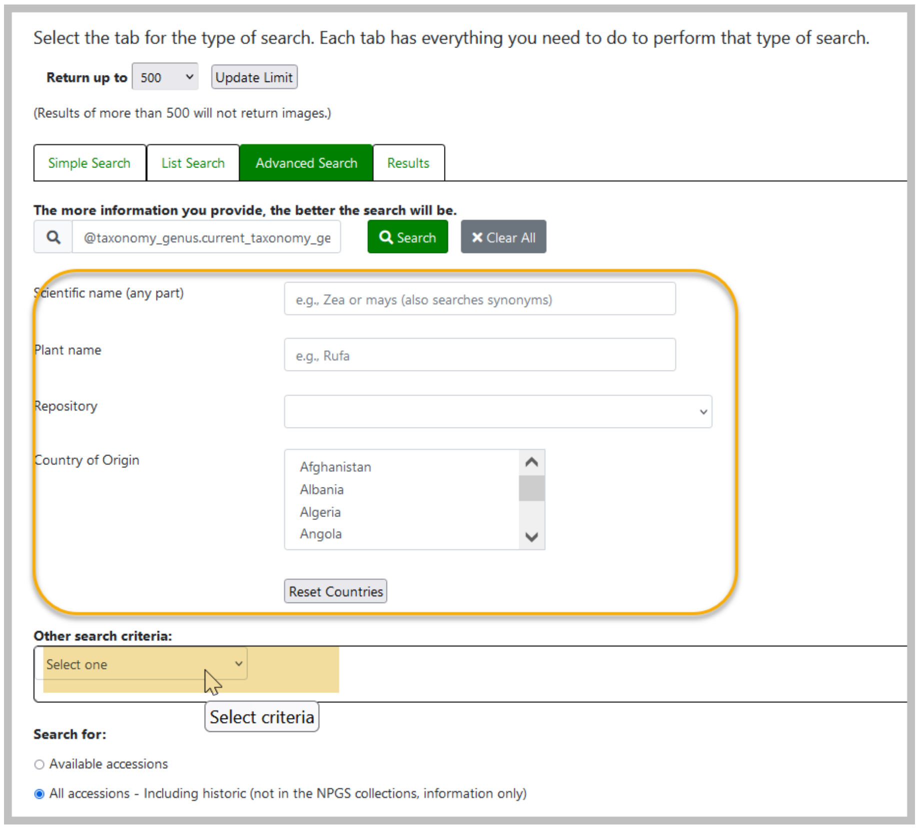
When there are more than 500 (or whatever your limit is set to) accessions that are genus hits on PI numbers, the most recent of those is first. If there are less than 500 PI records for the genus you are going to see recent non-PI genus hits further down the list and recent PI non-genus hits even further down. That is not all recent accessions will be at the top because the other sorts have a higher precedence.

The Public Website has a dual personality. External users (non genebank staff), use the PW to search for, and order accessions. Internal staff, whose Public Website logins have been associated by the GG Admin to their CT login, have additional features, including the ability to run SQL queries against the database. Refer to the appendix section *SQL Queries for Searching the Database* on using the Public Website to search the database using SQL queries.

The search text formatted in the Search Tool text box used by internal genebank staff can also be copied and used in the Search box on the Public Website. This may be handy when an external user requests assistance obtaining information from the database that is not available via any Public Website options. An internal staff user can format the query in the Search Tool, send the query text to the external user and explain how to drop the query text in the search box. (See Public Website search constructs.)

Advanced Searches

On the **Advanced Search** tab, additional criteria may be included to supplement the text inputted in the search box:


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 | Results

The more information you provide, the better the search will be.

@taxonomy_genus.current_taxonomy_ge Search Clear All

Scientific name (any part): e.g., Zea or mays (also searches synonyms)

Plant name: e.g., Rufa

Repository:

Country of Origin: Afghanistan, Albania, Algeria, Angola

Reset Countries

Other search criteria:

Select one

Select criteria

Search for:

○ Available accessions

● All accessions - Including historic (not in the NPGS collections, information only)

What Does the Search Engine Search?

The search engine (SE) has three main code sections:

1. Formatted
2. Lists
3. Unformatted

1. **Formatted: What it's told to search for**

   The user creates formatted searches from QBE with SQL-like syntax starting with the at sign (@)
   Ex: **@accession.accession_number_part1 = 'PI' AND @site.site_id IN (3)**

2. **Lists: Identifier (ID) lists**

   The Search List function looks for certain patterns in the text provided in the listed items. It first determines the number of blocks of text separated by spaces (also known as "tokens").

Number of Tokens	the Search Engine Assumes	Example
4	Inventory identifier	NA 51425 .001 PL
3	Accession identifier	GMAL 3764 .a
1 (text)	Accession identifier	CZ12345twery
1 (numeric)	Order Request identifier	345102
	Plant Names	

   When there are 4 tokens – the SE assumes the items are inventory items, since the inventory identifier may have up to four items (prefix, number, suffix, and inventory type form). When there are three tokens, it assumes these are the three parts of the accession identifier. The List Search is also programmed to use a single token and look for accessions matching the one text string (some genebanks use only the accession prefix field to contain the entire accession identifier).

3. **Unformatted - Freeform Searches**

   Enter words (and/or numbers) and the SE tries to find them as best it can

   A. It will first search IDs such as PI 500000
   (using either accession or inventory ID)

   B. Each word is checked for an exact match on 22 fields (determined by the DBA using the **sys_search_autofield** table) (see autofields)

   C. Words are also checked in any existing full-text indexes.
   The GG DBA can index any text field, usually large fields such as **Note** fields (comments) to meet an organization's requirements. The GG table **sys.fulltext.indexes** lists these fields.
   (see Full-Text Indexing)

Users can combine formatted and freeform criteria or append formatted criteria to the end of a list search. Searches work best when the formatted text is appended after the list of items (since that is where the PW tacks it on).

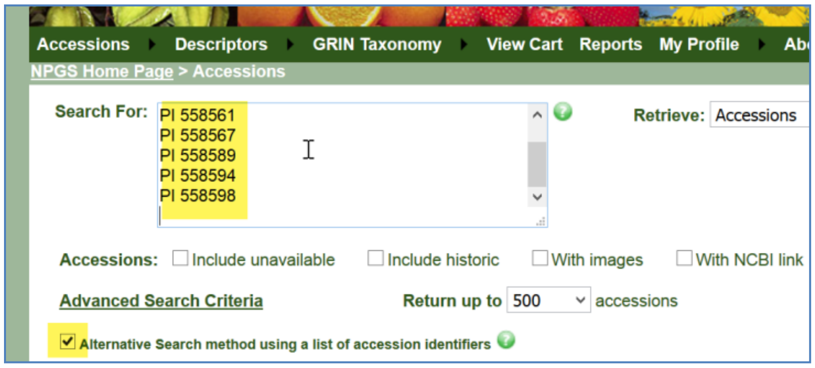
List Search on the Public Websites

The original GRIN-Global Public Website had a checkbox that need to be selected in order to use the List Search. In the current Public Website, the List search has its own tab.

On the public website, if you enter a valid order ID in the List Search box, the search will return the accessions included in the request.

Original PW

*The checkbox "**Alternative Search method…**" initiates the List Search*



Search For:
PI 558561
PI 558567
PI 558589
PI 558594
PI 558598

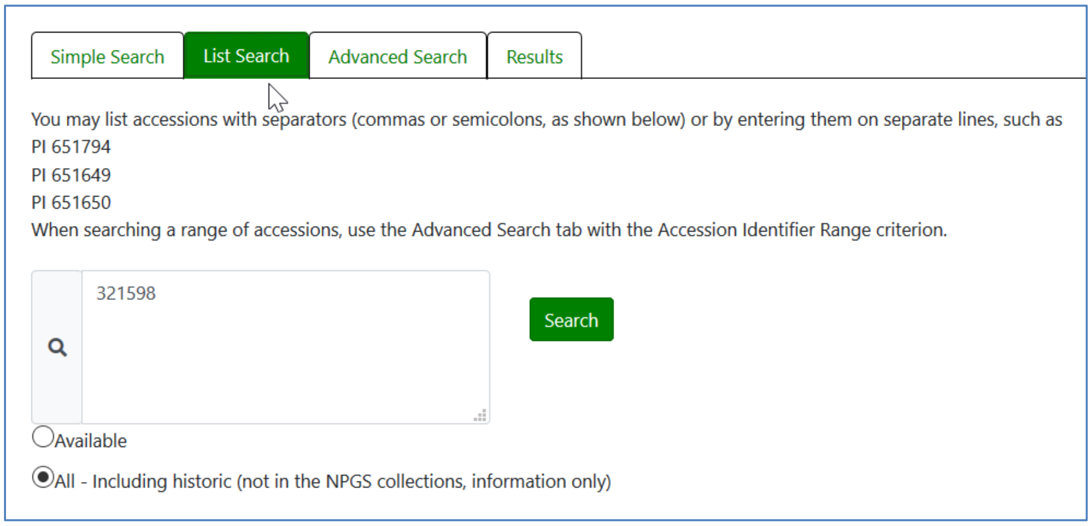
Retrieve: Accessions

Accessions: ☐ Include unavailable ☐ Include historic ☐ With images ☐ With NCBI link

Advanced Search Criteria   Return up to 500 accessions

☑ Alternative Search method using a list of accession identifiers

Current Public Website List Search



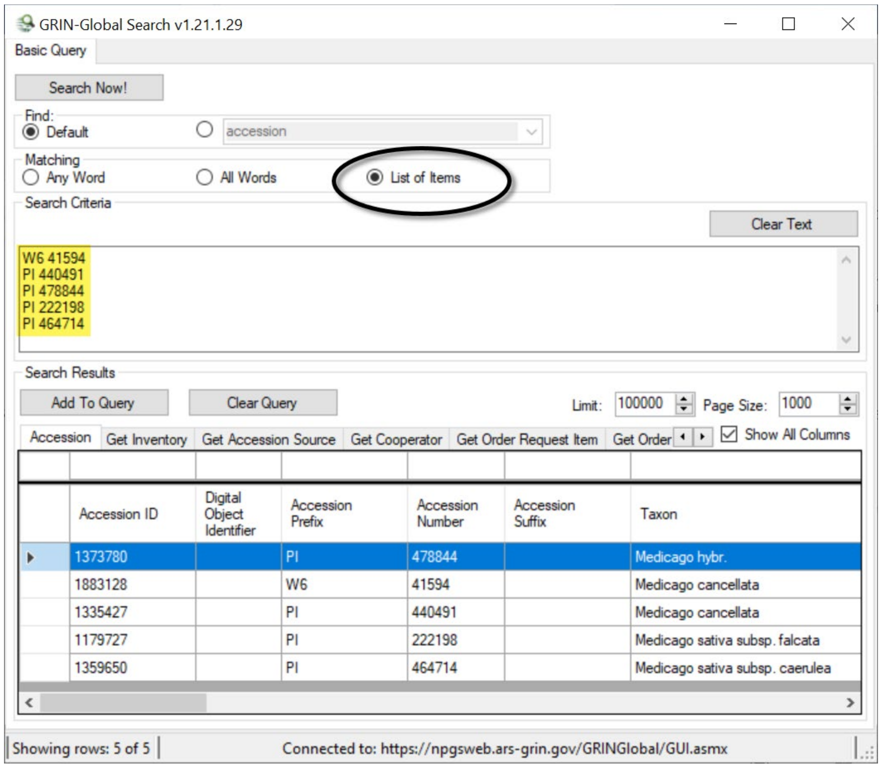
You may list accessions with separators (commas or semicolons, as shown below) or by entering them on separate lines, such as
PI 651794
PI 651649
PI 651650
When searching a range of accessions, use the Advanced Search tab with the Accession Identifier Range criterion.

321598

○ Available
● All - Including historic (not in the NPGS collections, information only)

List Search in the Search Tool

In releases prior to server 1.9.8.2, the search would work with a list in the text box even when this radio button wasn't selected. Now it must be selected for a list of IDs.


GRIN-Global Search v1.21.1.29

Basic Query

Search Now!

Find: [✓ Default] accession

Matching: Any Word | All Words | [✓ List of Items]

Search Criteria

Clear Text

```
W6 41594
PI 440491
PI 478844
PI 222198
PI 464714
```

Search Results

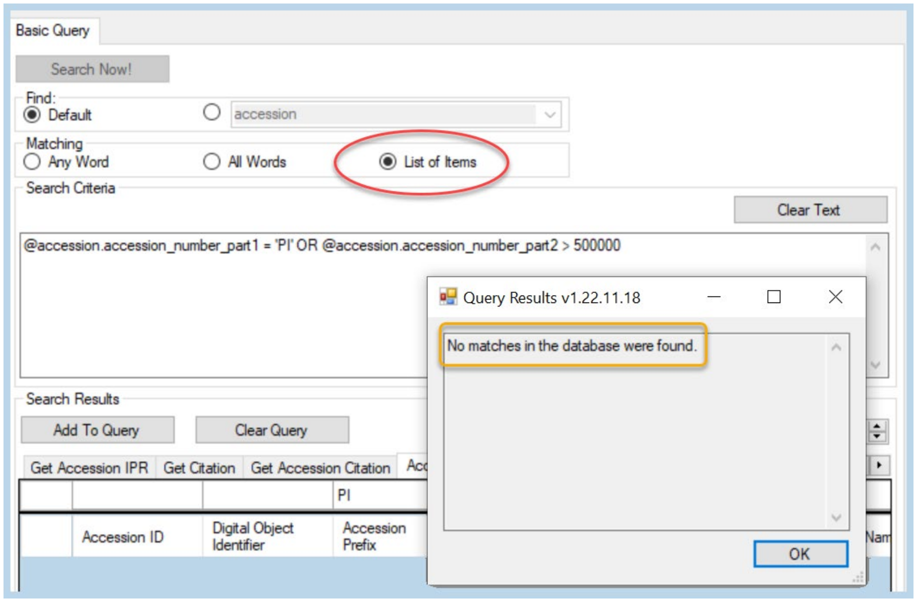
Add To Query | Clear Query | Limit: 100000 | Page Size: 1000

Accession | Get Inventory | Get Accession Source | Get Cooperator | Get Order Request Item | Get Order | ✓ Show All Columns

Accession ID	Digital Object Identifier	Accession Prefix	Accession Number	Accession Suffix	Taxon
1373780		PI	478844		Medicago hybr.
1883128		W6	41594		Medicago cancellata
1335427		PI	440491		Medicago cancellata
1179727		PI	222198		Medicago sativa subsp. falcata
1359650		PI	464714		Medicago sativa subsp. caerulea

Showing rows: 5 of 5 | Connected to: https://npgsweb.ars-grin.gov/GRINGlobal/GUI.asmx

Remember to switch radio buttons after a List Search. Otherwise, a typical search will fail.


Basic Query

Search Now!

Find: [✓ Default] accession

Matching: Any Word | All Words | [✓ List of Items]

Search Criteria

Clear Text

```
@accession.accession_number_part1 = 'PI' OR @accession.accession_number_part2 > 500000
```

Query Results v1.22.11.18

No matches in the database were found.

OK

Search Results

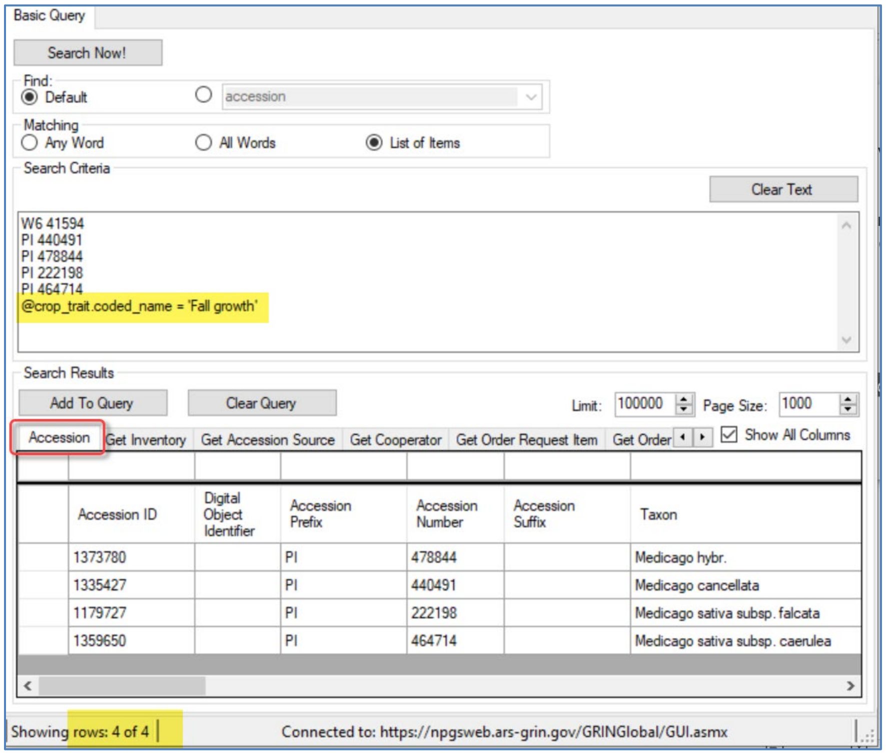
Add To Query | Clear Query

Get Accession IPR | Get Citation | Get Accession Citation

Accession ID	Digital Object Identifier	Accession Prefix

Extension of the List Search

It is possible to append a formatted search string to a list of items. For example, the following example is a valid search:



Find: [✓ Default] accession

Matching: Any Word All Words [✓ List of Items]

Search Criteria

```
W6 41594
PI 440491
PI 478844
PI 222198
PI 464714
@crop_trait.coded_name = 'Fall growth'
```

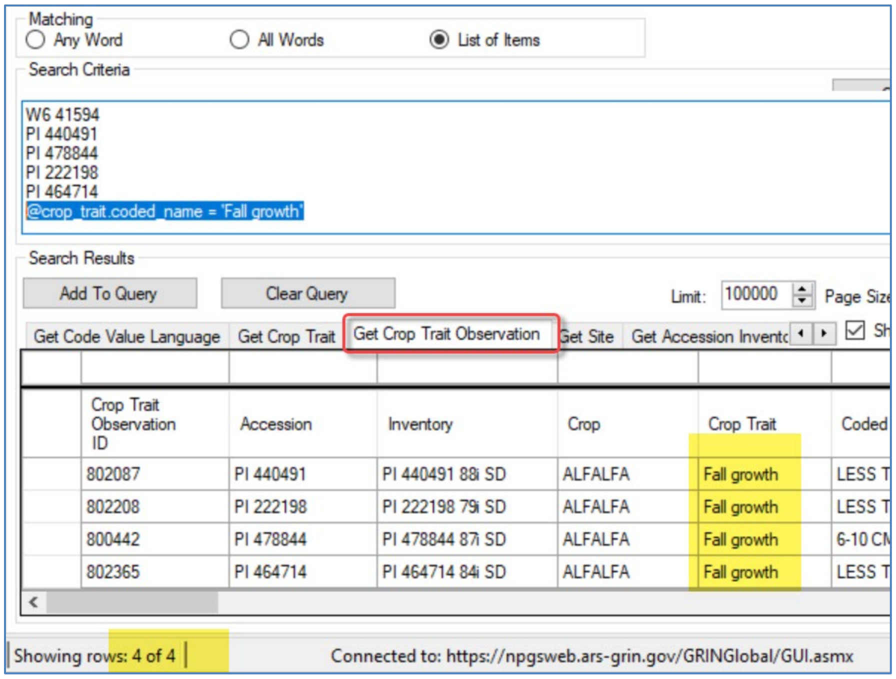
Limit: 100000 Page Size: 1000 [✓] Show All Columns

Accession

Accession ID	Digital Object Identifier	Accession Prefix	Accession Number	Accession Suffix	Taxon
1373780		PI	478844		Medicago hybr.
1335427		PI	440491		Medicago cancellata
1179727		PI	222198		Medicago sativa subsp. falcata
1359650		PI	464714		Medicago sativa subsp. caerulea

Showing rows: 4 of 4 Connected to: https://npgsweb.ars-grin.gov/GRINGlobal/GUI.asmx

In the search without the @crop_trait.coded_name = 'Fall growth' statement, five accessions were found, but with it, four. When switching to the **Crop Trait Observation** dataview and *re-running the search*, the reason is more apparent:



Matching: Any Word All Words [✓ List of Items]

Search Criteria

```
W6 41594
PI 440491
PI 478844
PI 222198
PI 464714
@crop_trait.coded_name = 'Fall growth'
```

Limit: 100000 Page Siz

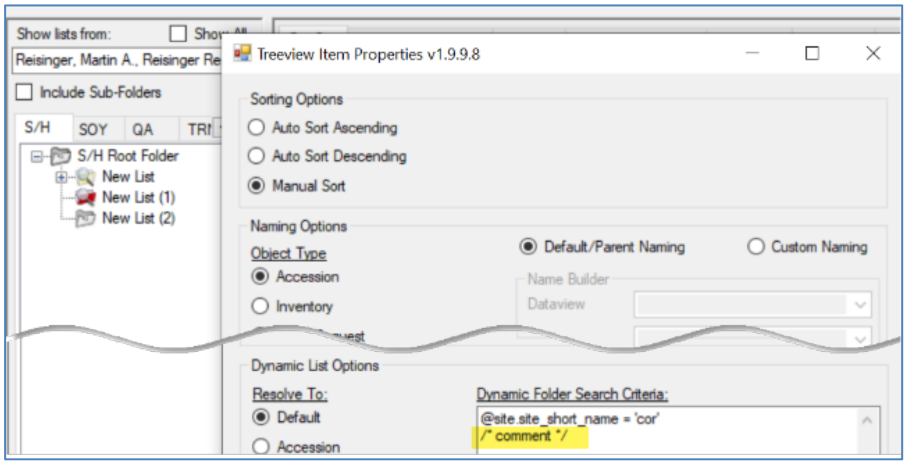
Get Crop Trait Observation

Crop Trait Observation ID	Accession	Inventory	Crop	Crop Trait	Coded
802087	PI 440491	PI 440491 88i SD	ALFALFA	Fall growth	LESS T
802208	PI 222198	PI 222198 79i SD	ALFALFA	Fall growth	LESS T
800442	PI 478844	PI 478844 87i SD	ALFALFA	Fall growth	6-10 CI
802365	PI 464714	PI 464714 84i SD	ALFALFA	Fall growth	LESS T

Showing rows: 4 of 4 Connected to: https://npgsweb.ars-grin.gov/GRINGlobal/GUI.asmx

Search Comments

When using the Search Tool, you can include comments. This is helpful when copying the search statement to the Curator Tool to build a Dynamic Folder:



```
@site.site_short_name = 'cor'
/* comment */
```

- when you use a double dash -- on a line, anything after the double dash is treated as a comment
- to comment multiple lines, start with **/*** and then end your comment with ***/**

Curator Tool – Searching via Dynamic Folders

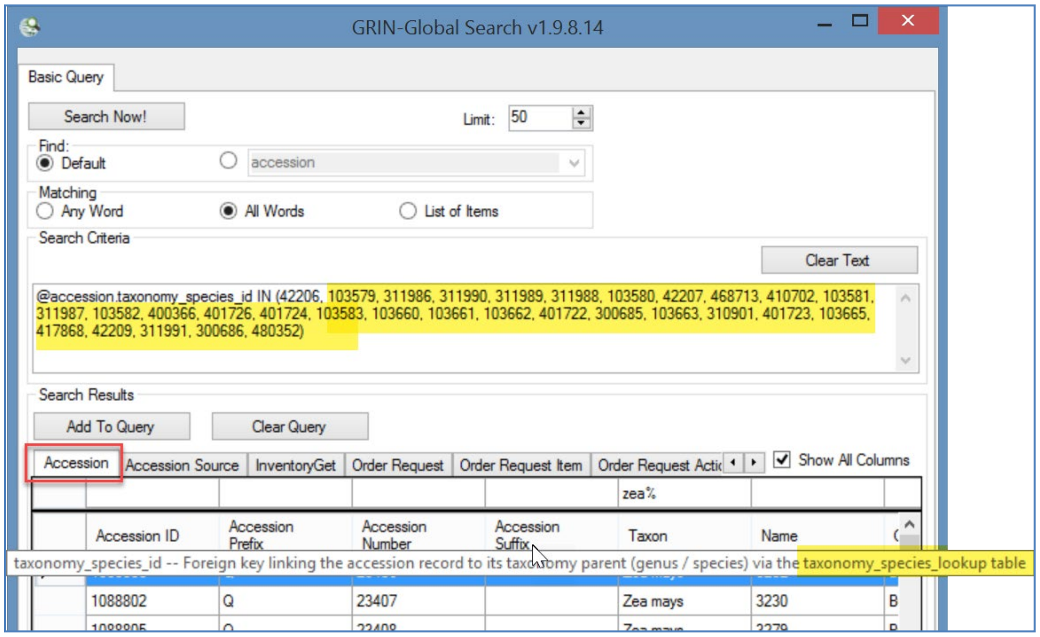
Dynamic Folders

The Curator Tool has two types of folders, static and dynamic. Dynamic folders (also referred to as dynamic queries) are basically stored queries in a CT user's List Panel. The query most likely was created by copying generated text from a Search Tool query. A big advantage of setting up a dynamic folder is that after the query folder has been created, the folder retains your search criteria and eliminates the need to redoing a query in the Search Tool.

A complete Dynamic Folders guide is online: https://www.grin-global.org/docs/gg_dynamic_folders.pdf

IDs & Lookups

When searching on a field that uses LOOKUP IDs, the ID numbers are listed in the search statement. If you are curious, open the respective dataview and look for the corresponding records.

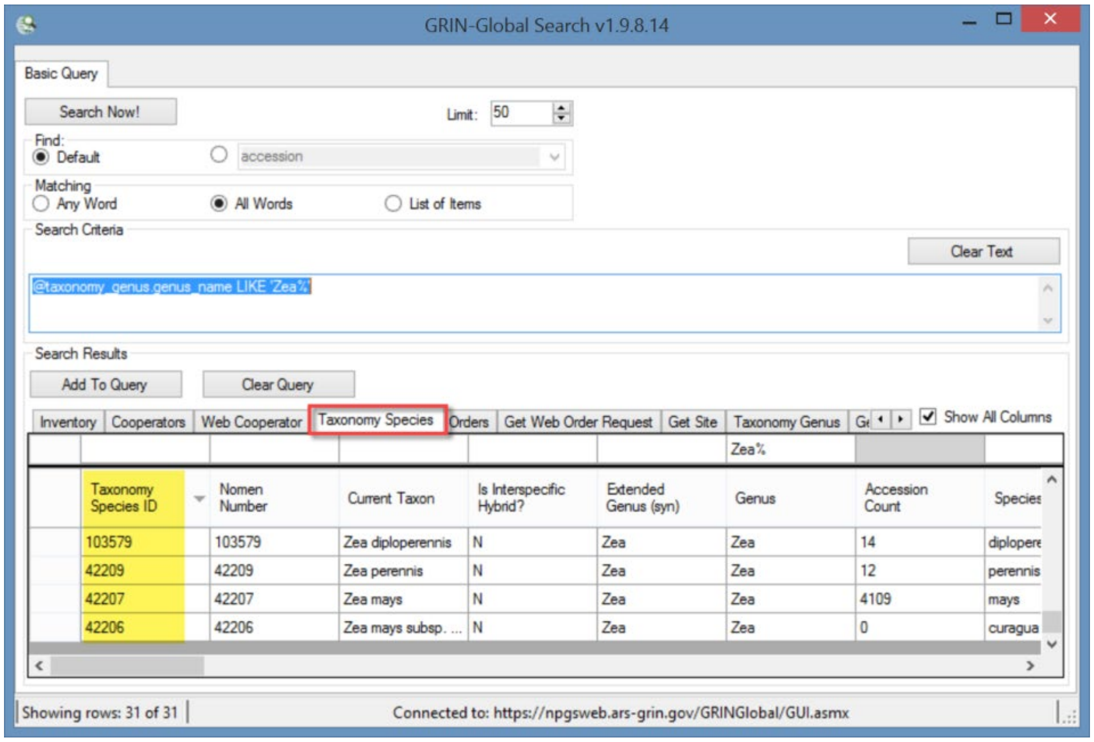


```
@accession.taxonomy_species_id IN (42206, 103579, 311986, 311990, 311989, 311988, 103580, 42207, 468713, 410702, 103581,
311987, 103582, 400366, 401726, 401724, 103583, 103660, 103661, 103662, 401722, 300685, 103663, 310901, 401723, 103665,
417868, 42209, 311991, 300686, 480352)
```

taxonomy_species_id -- Foreign key linking the accession record to its taxonomy parent (genus / species) via the taxonomy_species_lookup table

	Accession ID	Accession Prefix	Accession Number	Accession Suffix	Taxon	Name
					zea%	
	1088802	Q	23407		Zea mays	3230

In the screen above, there were 31 species records matching “Zea.” The following screen shows the corresponding 31 species records in the **Taxonomy Species** dataview:



```
@taxonomy_genus.genus_name LIKE 'Zea%'
```

Taxonomy Species ID	Nomen Number	Current Taxon	Is Interspecific Hybrid?	Extended Genus (syn)	Genus	Accession Count	Species
					Zea%		
103579	103579	Zea diploperennis	N	Zea	Zea	14	diplopere
42209	42209	Zea perennis	N	Zea	Zea	12	perennis
42207	42207	Zea mays	N	Zea	Zea	4109	mays
42206	42206	Zea mays subsp. ...	N	Zea	Zea	0	curagua

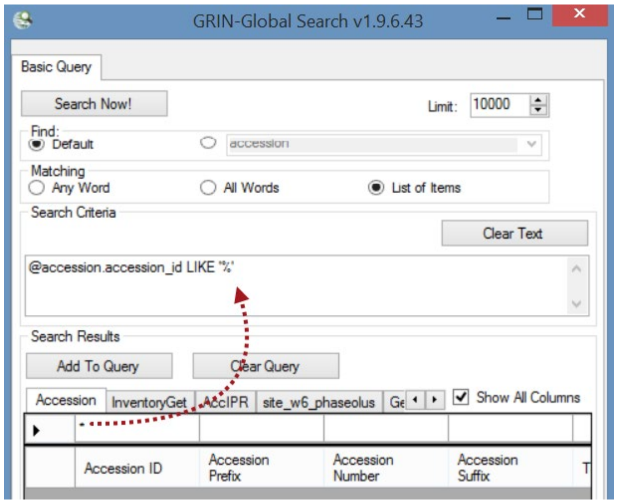
Showing rows: 31 of 31 | Connected to: https://npgsweb.ars-grin.gov/GRINGlobal/GUI.asmx

Wildcards and finding Empty / Missing Stuff / Nulls

The Search Tool uses the percent sign (%), the asterisk (*), and the underscore (_) as wildcard characters.

% and * behave differently in the Public Website. Full text indexing will handle asterisks at the end of the word (*), whereas if you use a trailing % sign, only the autofield search is used.

In the Search Too, the * **converts to LIKE '%'**



@accession.accession_id LIKE '%'

Using Quotes

Using quotes ensures that the full term is searched. Two examples below, with and without quotes - and the number of found records:

Search string	Records Found	What the Search Engine is Looking For
'yellow rain'	0	the two words **yellow rain** - exactly as entered
yellow rain	66	either word, **yellow**, or **rain**, in any of the fields that are searched
'rain'	638	any occurrence of the word **rain** in any of the fields being searched
rain	638	any occurrence of the word **rain** in any of the fields being searched

Sometimes it is desirable to find "what's missing."

NOT EQUAL TO

!= operator (same as <>)

Use the **!=** or the **<>** ("not equal to") operator as needed, as in:

> **@accession.accession_number_part1 != 'PI'**
> **-or -**
> **@accession.accession_number_part1 <> 'PI**

Nulls

NULL values represent missing unknown data. By default, a table column can hold NULL values.

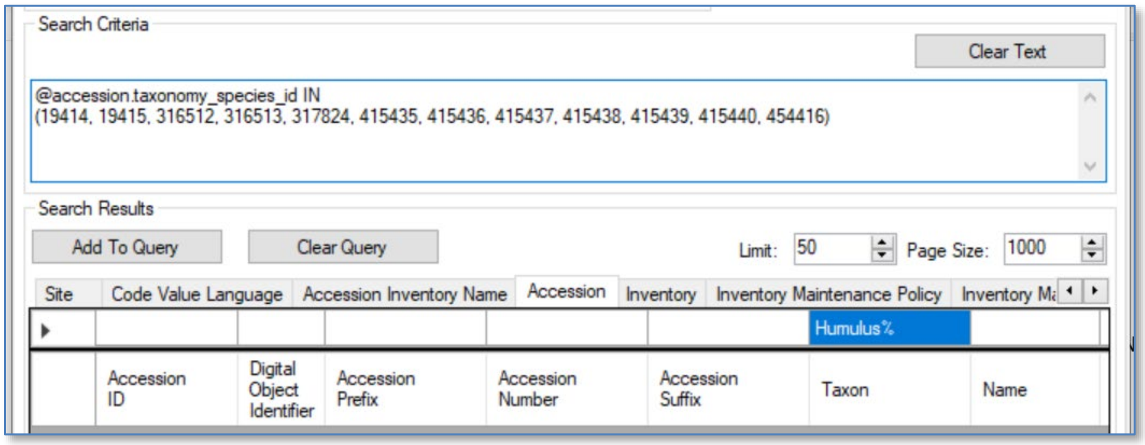
> **@accession.accession_number_part3 IS NULL**

If necessary (because of the dataviews), if the QBE will not generate the **IS NULL** or **IS NOT NULL** code, hand code the appropriate clause in the Search Criteria box:

> **AND inventory.parent_inventory_id IS NOT NULL**

NOT IN

Used when fields involve lookup values. For example, when you have a search such as:



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

You can use **NOT IN** to exclude lookup values.

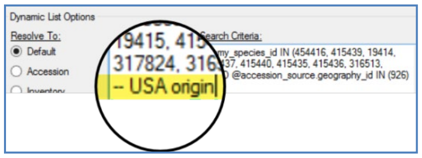
Extended SQL Support

Additional SQL terms can be used now:

- BETWEEN
- WHERE
- EXCEPT
- INTERSECT
- GETDATE()
- DATEDIFF()
- COUNT
- DISTINCT

The ST can't handle an entire SQL select statement, only a statement beginning with a SQL WHERE clause.

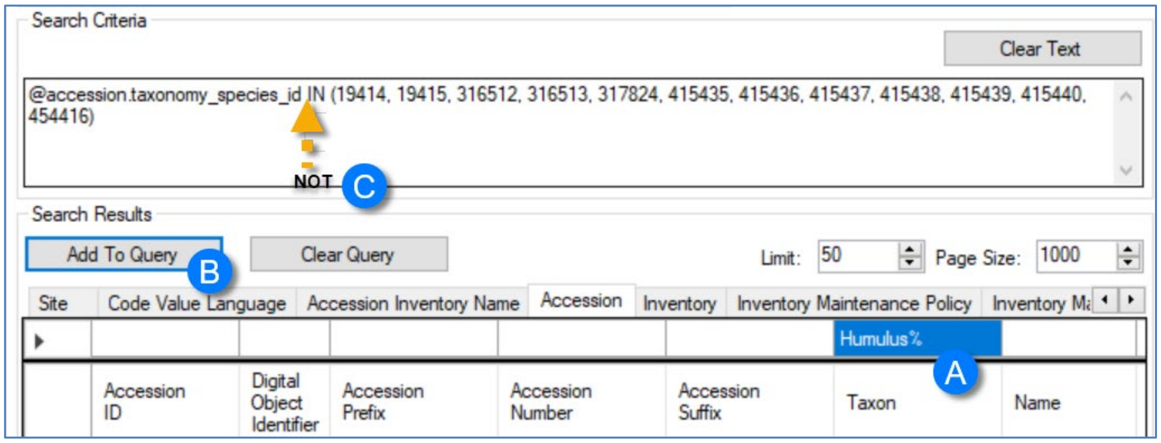
- comments are valid (double dash) --
  also valid w/ dynamic folders:



- WHERE may be used in the Search Tool:

WHERE accession.accession_number_part2 BETWEEN 500000 AND 500050

Rather than type from scratch in the Search Tool's **Search Criteria** box, first input a s ample in the QBE cells above the grid, and then edit the generated text.



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

Reserved Words & Wildcards -- Examples

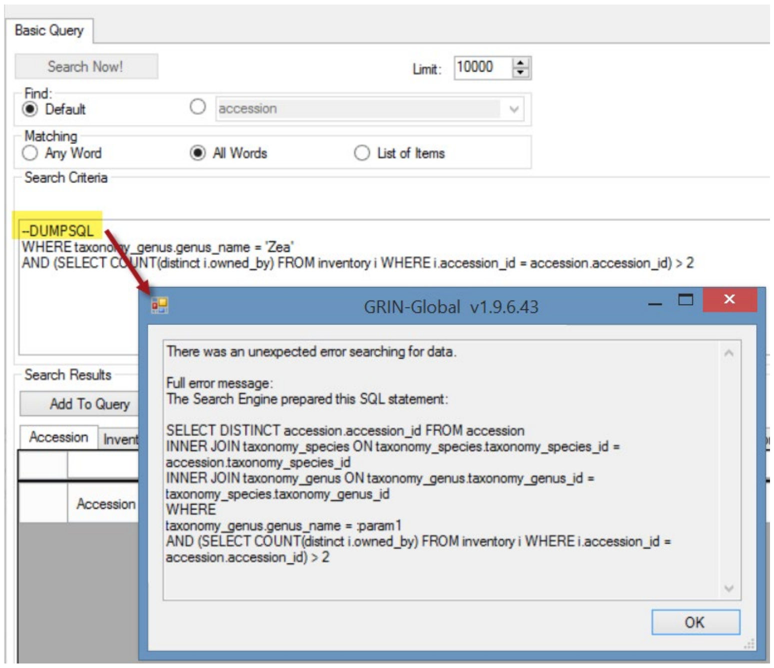
Wildcard / Operator / Reserved Words	Examples / Notes
% (percent symbol) * (asterisk) also It is recommended to use the % rather than the *. (Date searches work with %, but not with * - this is a known (reported) bug.)	Use to broaden searches, especially when the exact spelling is unknown. The field must be a text field. Either wildcard (% or *) allows a match of any string of any length (including zero length) Examples: **Rubus%** **Prunus%var** will locate any Prunus with "var" included; **%var%** will locate any accessions with the text "var" as part of its taxon **'2015%'**
_ (underscore)	The wild card underscore character **_** Represents any *single* character. Multiple underscores may be used if needed. The field must be a text field. **Solanum_x%** will find: **Solanum x** doddsii and **Solanum x** sucrense If you need to search for the underscore character rather than have it act as wildcard, enclose it in brackets, such as: @inventory_action.action_name_code LIKE 'INS[_]%' (in this example, the 4th character must be an underscore character)
<> != (not equal to)	Can be used to indicate "not equal to." The field can be either a text or numeric field. – when the field is a *text* field, the criterion must be enclosed by quotes – single quotes: 'PI' or double quotes: "PI" – when the field is a *numeric* field, the criterion *is not* enclosed in quotes

Wildcard / Operator / Reserved Words	Examples / Notes
IS NULL / IS NOT NULL	NULL values represent missing unknown data. By default, a table column can hold NULL values. Note: NULL and 0 are not equivalent.
IN / NOT IN	Used when the criterion field is using a lookup table. (Lookups generate an **IN (…)** clause.) The numbers in the parentheses are the Lookup Key values in the database.
LIKE	The LIKE operator is used to search for a specified pattern. Example: **LIKE 'CAPSICUM%'** In this case the QBE is saying find any text that begins with "Capsicum." The trailing percent symbol indicates that any records with any text after "capsicum" should be included if found.
BETWEEN	When a range of values is needed, construct your criteria using a range. For example: @order_request.ordered_date > '2015-01-31' AND order_request.ordered_date < '2015-03-01' (finds the orders for February, 2015) Same results, using BETWEEN @order_request.ordered_date BETWEEN '2015-01-31' AND '2015-03-01' Note: BETWEEN can be used with text as well, such as searching for a range between 'GBK-0100' and 'GBK-0200'
Date Fields	Searching for dates can be tricky because the date field includes the time of day as well. Refer to Date Fields for details. The following are valid searches: **@accession.created_date like '2015%'** **@accession.created_date like '2015-09-%'** **@accession.created_date like '2015-09-05%'** **@accession.created_date like '2015-%-05%'**

Wildcard / Operator / Reserved Words	**Examples / Notes**
GETDATE()	Retrieves database current date/time in SQL Server
DATEDIFF()	Calculates the difference between two dates
WHERE	The ST can't handle an entire SQL select statement, but it can handle parts of a SQL WHERE clause. The Search Engine looks at which fields you use so it knows which table to join when it builds the FROM clause. And the dataview definition specifies which fields get selected. **WHERE taxonomy_genus.genus_name like 'Triticum%'** **AND NOT EXISTS (SELECT * FROM accession_source acs** **WHERE accession.accession_id = acs.accession_id** **AND acs.source_type_code = 'COLLECTED')**
COUNT(*)	A query using COUNT to find rows with many inventories (from one accession) in the Search Tool or dynamic folder: **@ taxonomy_genus.genus_name = 'Zea'** **AND (SELECT COUNT(*) FROM inventory i WHERE** **i.accession_id = accession.accession_id) > 32**
Subqueries	A subquery is a query within a query – the inner query is resolved first. Can be used in various ways, such as to search by specific owner Example: **@accession.owned_by IN (SELECT cooperator_id FROM cooperator WHERE last_name = 'Millard')** Example: A query using COUNT to find rows with many inventories (from one accession) in the Search Tool or dynamic folder: **@ taxonomy_genus.genus_name = 'Zea'** **AND (SELECT COUNT(*) FROM inventory i WHERE** **i.accession_id = accession.accession_id) > 32**

Wildcard / Operator / Reserved Words	Examples / Notes
DISTINCT [server >= 1.9.9.2]	The SELECT DISTINCT statement is used to return only distinct (different) values. Inside a table, a column often contains many duplicate values; and sometimes you only want to list the different (distinct) values. Example: List accession records with inventories having more than 2 different owners **WHERE taxonomy_genus.genus_name = 'Zea'** **AND (SELECT COUNT(distinct i.owned_by) FROM inventory i** **WHERE i.accession_id = accession.accession_id) > 2** In the Search Tool, change the first line to: **@taxonomy_genus.genus_name = 'Zea'**
LEN function [server >= 1.9.9.2]	The LEN function determines the string length. This could be used to find long plant names **WHERE LEN(accession_inv_name.plant_name) > 36**
EXCEPT [server >= 1.9.9.2]	Returns any distinct values from the query to the left of the EXCEPT operator that are not also returned from the right query. The following EXCEPT query is used to track orders not yet completed (**order_request.completed_date IS NULL**) when a curator has been alerted (**action_name_code = 'CURALERTED')** about an NC7 order **(site_id = 16)**, but he has not cleared it and the order is still pending (the curator hasn't cleared the order (**action_name_code = 'CURCLEARED').** **EXCEPT** **@site.site_id IN (16) AND @order_request.completed_date IS NULL** **AND @order_request_action.action_name_code = 'CURALERTED'** **AND @order_request_action.cooperator_id IN (122186)**

Wildcard / Operator / Reserved Words	**Examples / Notes**
INTERSECT [The GG server release must be >= 1.9.9.2]	The INTERSECT operator is used to combine like rows from two queries. It returns rows that are in common between both results. For example, using the search tool, find accessions with specific observation values for two different traits. Example: find *kernel color* **White** and *primary race* **Corn Belt Dent**. **@crop.name = 'Maize' AND @crop_trait_lang.title = 'Primary Race' AND @crop_trait_code_lang.title = 'Corn Belt Dent'** **INTERSECT** **@crop.name = 'Maize' AND @crop_trait_lang.title = 'KERNEL COLOR' AND @crop_trait_code_lang.title = 'White'** **INTERSECT** **@site.site_id IN (16) AND @inventory.is_distributable = 'Y' AND @inventory.is_available = 'Y'** A similar, but faster version of the query, using the trait IDs: **@crop_trait_observation.crop_trait_id = 89001 AND @crop_trait_code_lang.title = 'Corn Belt Dent'** **INTERSECT** **@crop_trait_observation.crop_trait_id = 89027 AND @crop_trait_code_lang.title = 'White'** **INTERSECT** **@site.site_id IN (16) AND @inventory.is_distributable = 'Y' AND @inventory.is_available = 'Y'**

Wildcard / Operator / Reserved Words	**Examples / Notes**
--DUMPSQL [server >= 1.9.9.2]	With –DUMPSQL, the search engine has an option to deliberately throw an error and show the SQL it generated when the first line of the query is this comment: **--DUMPSQL** (See DUMPSQL.) 

Basic Query

Search Now! Limit: 10000

Find: ● Default ○ accession

Matching ○ Any Word ● All Words ○ List of Items

Search Criteria

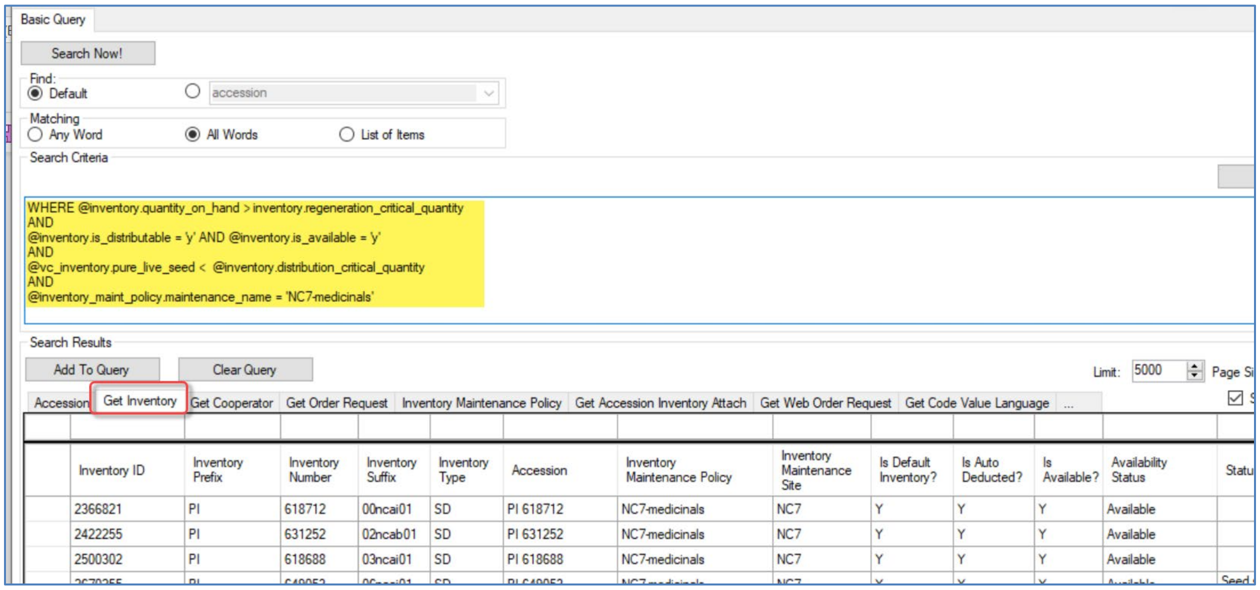
```
--DUMPSQL
WHERE taxonomy_genus.genus_name = 'Zea'
AND (SELECT COUNT(distinct i.owned_by) FROM inventory i WHERE i.accession_id = accession.accession_id) > 2
```

GRIN-Global v1.9.6.43

```
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 (SELECT COUNT(distinct i.owned_by) FROM inventory i WHERE i.accession_id =
accession.accession_id) > 2
```

OK

Maintenance Policy, such as in the following example:


Basic Query

Search Now!

Find: (● Default) ○ accession

Matching: ○ Any Word (● All Words) ○ List of Items

Search Criteria

```
WHERE @inventory.quantity_on_hand > inventory.regeneration_critical_quantity
AND
@inventory.is_distributable = 'y' AND @inventory.is_available = 'y'
AND
@vc_inventory.pure_live_seed <  @inventory.distribution_critical_quantity
AND
@inventory_maint_policy.maintenance_name = 'NC7-medicinals'
```

Search Results

Add To Query | Clear Query | Limit: 5000 | Page Si

Accession | Get Inventory | Get Cooperator | Get Order Request | Inventory Maintenance Policy | Get Accession Inventory Attach | Get Web Order Request | Get Code Value Language | ...

Inventory ID	Inventory Prefix	Inventory Number	Inventory Suffix	Inventory Type	Accession	Inventory Maintenance Policy	Inventory Maintenance Site	Is Default Inventory?	Is Auto Deducted?	Is Available?	Availability Status	Statu
2366821	PI	618712	00ncai01	SD	PI 618712	NC7-medicinals	NC7	Y	Y	Y	Available	
2422255	PI	631252	02ncab01	SD	PI 631252	NC7-medicinals	NC7	Y	Y	Y	Available	
2500302	PI	618688	03ncai01	SD	PI 618688	NC7-medicinals	NC7	Y	Y	Y	Available	

The code above:

WHERE @inventory.quantity_on_hand > inventory.regeneration_critical_quantity

AND

@inventory.is_distributable = 'y' AND @inventory.is_available = 'y'

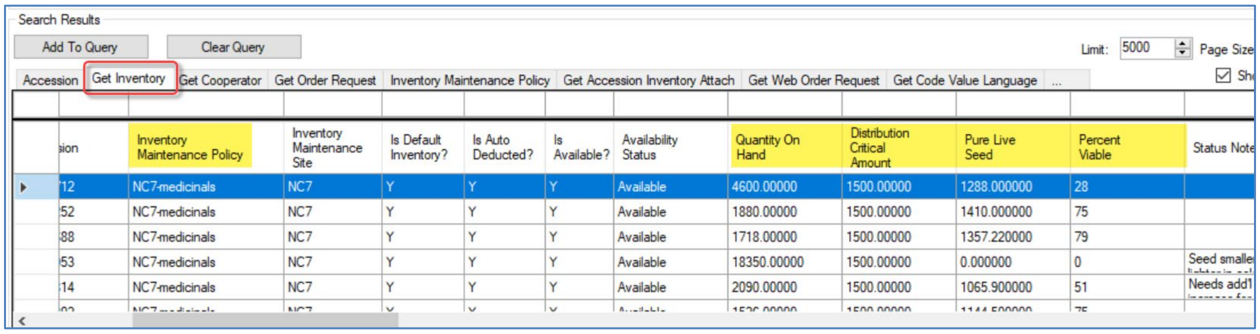
AND

@vc_inventory.pure_live_seed < @inventory.distribution_critical_quantity

AND

@inventory_maint_policy.maintenance_name = 'NC7-medicinals'

In the query above, 13 inventory lots were identified as having quantities of viable seeds that were less than the desired distribution quantities.


Search Results

Add To Query | Clear Query | Limit: 5000 | Page Size

Accession | Get Inventory | Get Cooperator | Get Order Request | Inventory Maintenance Policy | Get Accession Inventory Attach | Get Web Order Request | Get Code Value Language | ...

ion	Inventory Maintenance Policy	Inventory Maintenance Site	Is Default Inventory?	Is Auto Deducted?	Is Available?	Availability Status	Quantity On Hand	Distribution Critical Amount	Pure Live Seed	Percent Viable	Status Note
12	NC7-medicinals	NC7	Y	Y	Y	Available	4600.00000	1500.00000	1288.000000	28	
52	NC7-medicinals	NC7	Y	Y	Y	Available	1880.00000	1500.00000	1410.000000	75	
88	NC7-medicinals	NC7	Y	Y	Y	Available	1718.00000	1500.00000	1357.220000	79	
53	NC7-medicinals	NC7	Y	Y	Y	Available	18350.00000	1500.00000	0.000000	0	Seed smalle
14	NC7-medicinals	NC7	Y	Y	Y	Available	2090.00000	1500.00000	1065.900000	51	Needs add1

Full Text Indexing

A full text index will have an entry in a generated index for each term or word found in a specified table field. These indexes are established by the genebank's GG administrator for specific fields in the database; additional fields can be indexed over time. This feature provides significant changes to the Public Website users' searches.

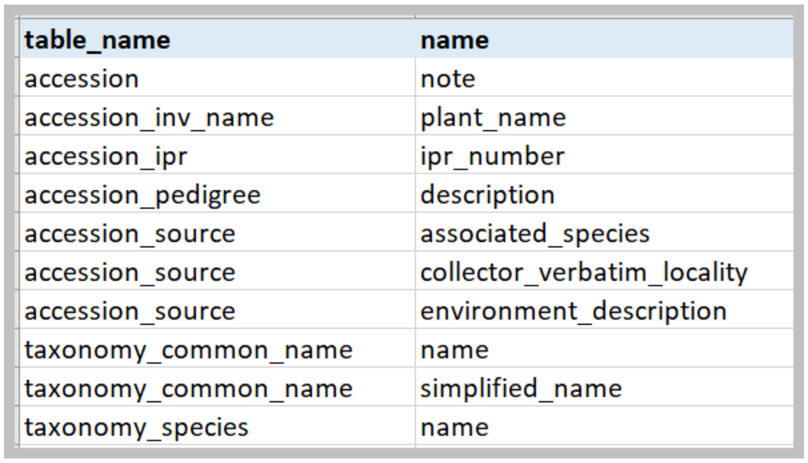
Administrator's Note: Full text indexing requires the GG administrator to use SQL Server's Full Text Indexing methodology. See also Appendix A.

Example:

Releases pre- 1.9.9.2	Release 1.9.9.2
PW: '%weedy red rice%'	PW: weedy red rice

NPGS:

In Release 1.9.9.2 and later, the following fields are now set to full text indexing:



table_name	name
accession	note
accession_inv_name	plant_name
accession_ipr	ipr_number
accession_pedigree	description
accession_source	associated_species
accession_source	collector_verbatim_locality
accession_source	environment_description
taxonomy_common_name	name
taxonomy_common_name	simplified_name
taxonomy_species	name

Some stop words (such as "the" and "and") that are both common and typically not meaningful are ignored by the search. (sample stop words)

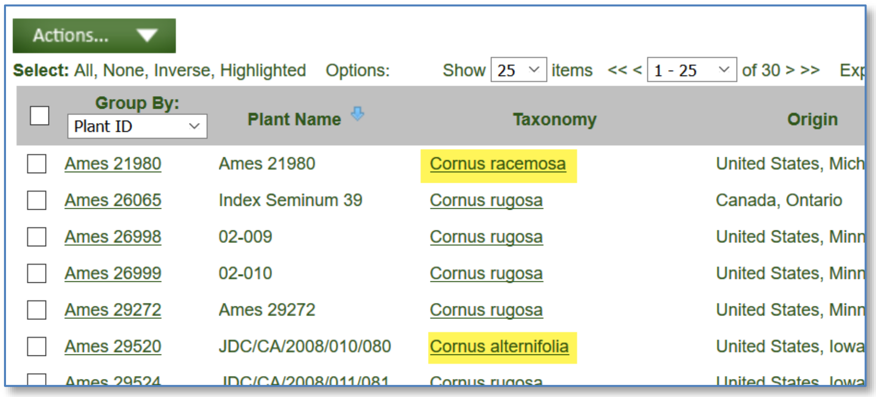
How would you know what fields are indexed? When logged into the Public Website, run the following SQL:

```
SELECT DISTINCT
  object_name(fic.[object_id])as table_name,
  [name]
FROM
  sys.fulltext_index_columns fic
  INNER JOIN sys.columns c
```

```
ON c.[object_id] = fic.[object_id]
AND c.[column_id] = fic.[column_id]
```

Considerations

A Public Website search for **%Cornus rugosa%** may find accessions which at first glance in the list may seem like not a valid match. In this example, the following displays in NPGS's database:

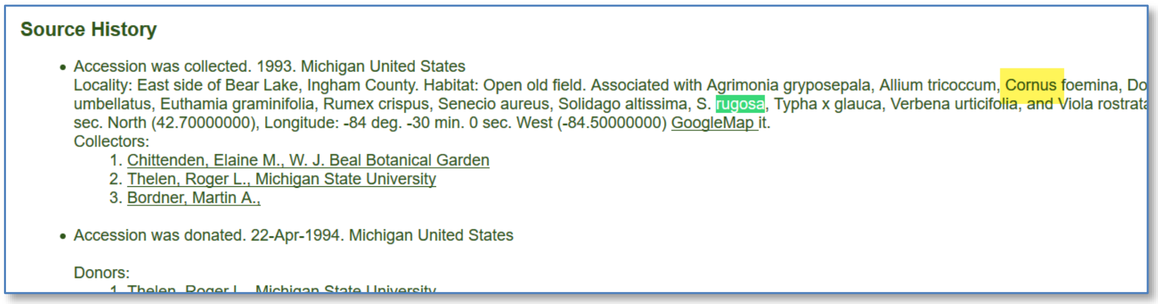


Actions...

Select: All, None, Inverse, Highlighted Options: Show 25 items << < 1 - 25 of 30 > >> Exp

Group By: Plant ID	Plant Name	Taxonomy	Origin
Ames 21980	Ames 21980	Cornus racemosa	United States, Mich
Ames 26065	Index Seminum 39	Cornus rugosa	Canada, Ontario
Ames 26998	02-009	Cornus rugosa	United States, Minn
Ames 26999	02-010	Cornus rugosa	United States, Minn
Ames 29272	Ames 29272	Cornus rugosa	United States, Minn
Ames 29520	JDC/CA/2008/010/080	Cornus alternifolia	United States, Iowa
Ames 29524	JDC/CA/2008/011/081	Cornus rugosa	United States, Iowa

Using the Ames 21980 accession, the detail page shows:



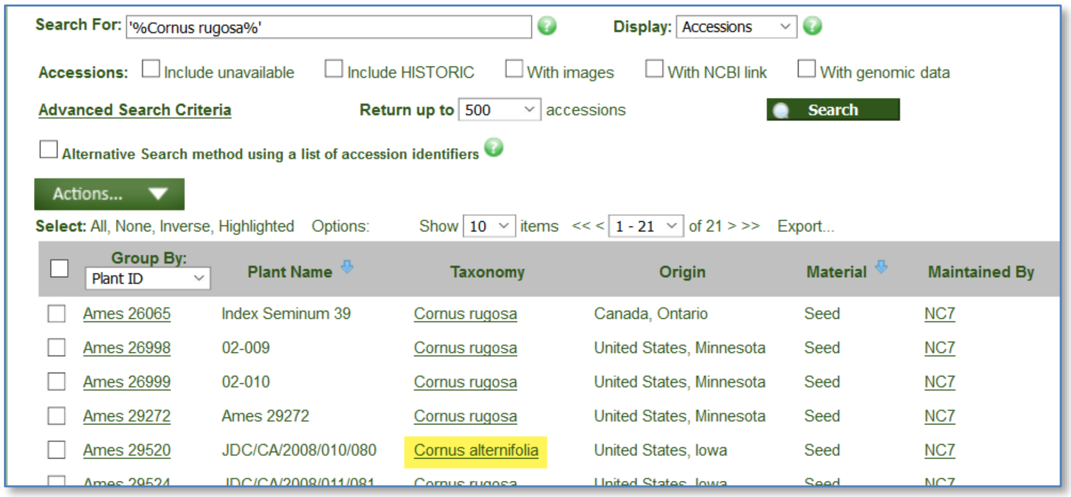
Source History

- Accession was collected. 1993. Michigan United States
  Locality: East side of Bear Lake, Ingham County. Habitat: Open old field. Associated with Agrimonia gryposepala, Allium tricoccum, Cornus foemina, Do
  umbellatus, Euthamia graminifolia, Rumex crispus, Senecio aureus, Solidago altissima, S. rugosa, Typha x glauca, Verbena urticifolia, and Viola rostrata
  sec. North (42.70000000), Longitude: -84 deg. -30 min. 0 sec. West (-84.50000000) GoogleMap it.
  Collectors:
  1. Chittenden, Elaine M., W. J. Beal Botanical Garden
  2. Thelen, Roger L., Michigan State University
  3. Bordner, Martin A.,
- Accession was donated. 22-Apr-1994. Michigan United States

  Donors:
  1. Thelen, Roger L., Michigan State University

The search is basically asking for *either* word to be found, ***Cornus***, or ***rugosa***. Any words specified between the %...%

When the same string is used, but in quotes – **'%Cornus rugosa%'** – the list of records will not include that record:



Search For: '%Cornus rugosa%' Display: Accessions

Accessions: ☐ Include unavailable ☐ Include HISTORIC ☐ With images ☐ With NCBI link ☐ With genomic data

Advanced Search Criteria Return up to 500 accessions Search

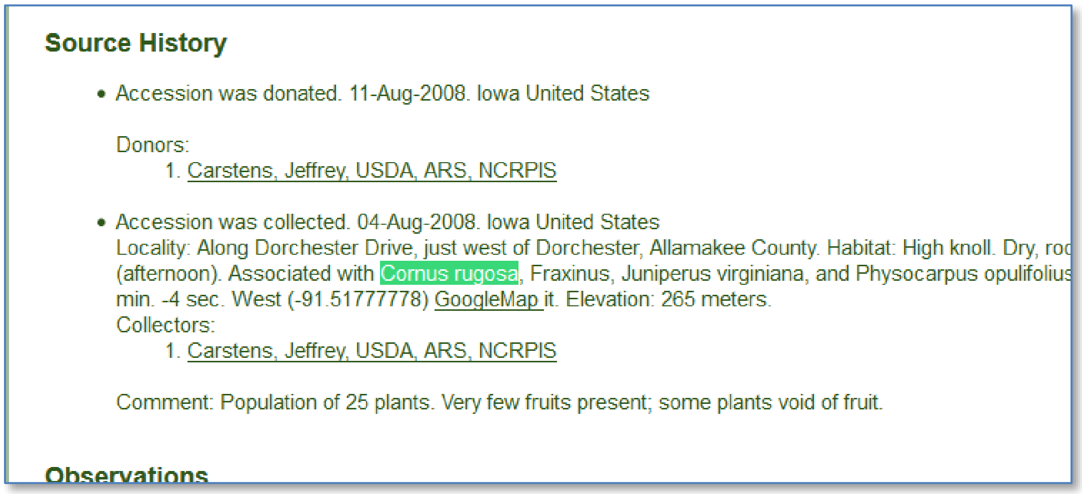
☐ Alternative Search method using a list of accession identifiers

Actions...

Select: All, None, Inverse, Highlighted Options: Show 10 items << < 1 - 21 of 21 > >> Export...

Group By: Plant ID	Plant Name	Taxonomy	Origin	Material	Maintained By
Ames 26065	Index Seminum 39	Cornus rugosa	Canada, Ontario	Seed	NC7
Ames 26998	02-009	Cornus rugosa	United States, Minnesota	Seed	NC7
Ames 26999	02-010	Cornus rugosa	United States, Minnesota	Seed	NC7
Ames 29272	Ames 29272	Cornus rugosa	United States, Minnesota	Seed	NC7
Ames 29520	JDC/CA/2008/010/080	Cornus alternifolia	United States, Iowa	Seed	NC7
Ames 29521	JDC/CA/2008/011/081	Cornus rugosa	United States, Iowa	Seed	NC7

Using the Ames 29520 accession, the detail page shows:



Source History

- Accession was donated. 11-Aug-2008. Iowa United States

  Donors:
  1. Carstens, Jeffrey, USDA, ARS, NCRPIS

- Accession was collected. 04-Aug-2008. Iowa United States
  Locality: Along Dorchester Drive, just west of Dorchester, Allamakee County. Habitat: High knoll. Dry, roc
  (afternoon). Associated with Cornus rugosa, Fraxinus, Juniperus virginiana, and Physocarpus opulifolius
  min. -4 sec. West (-91.51777778) GoogleMap it. Elevation: 265 meters.
  Collectors:
  1. Carstens, Jeffrey, USDA, ARS, NCRPIS

  Comment: Population of 25 plants. Very few fruits present; some plants void of fruit.

Observations

Using the quotes ensures that the full term is searched, in this case, ***Cornus rugosa***.

Two examples, with and without quotes - and the number of found records:

Search string	Records Found	What the Search Engine is Looking For
'yellow rain'	0	the two words **yellow rain** - exactly as entered
yellow rain	66	either word, **yellow**, or **rain**, in any of the fields that are searched
'rain'	638	any occurrence of the word **rain** in any of the fields being searched
rain	638	any occurrence of the word **rain** in any of the fields being searched

Extended SQL Support

WHERE

SQL WHERE clauses work in the Search Tool. However, since the search engine doesn't use table aliases, use full table names when constructing statements.

@taxonomy_genus.genus_name LIKE 'Glycine%'   equals

WHERE taxonomy_genus.genus_name LIKE 'Glycine%'

In the following example, a comment (text preceded with --) is also illustrated.

The following code can be used in the Search Tool:

```
-- Find accessions owned by Esther which are active, but not available
WHERE accession.owned_by=107186
AND accession.status_code = 'ACTIVE'
AND NOT EXISTS (SELECT * FROM inventory WHERE accession_id = accession.accession_id
           AND is_distributable = 'Y' AND is_available = 'Y' and owned_by=107186)
```

NOT

…now allowed in freeform queries:

Ex: **Bahamas AND NOT gossypium**
Ex: **Malus NOT (KAZ or Canada or USA or GBR)**

BETWEEN

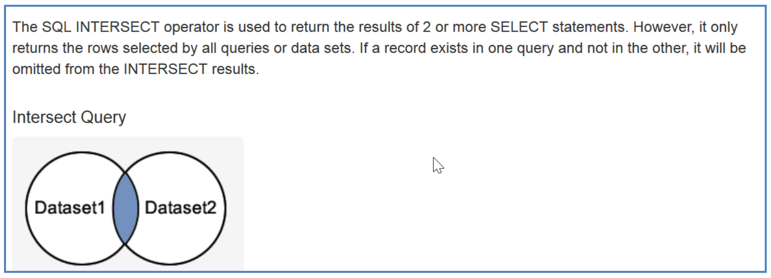
@accession.accession_number_part2 BETWEEN 500000 AND 500050

@order_request.ordered_date BETWEEN '2015-01-31' AND '2015-03-01'

INTERSECT

https://www.techonthenet.com/sql/intersect.php

The INTERSECT operator is used to combine like rows from two queries. It returns rows that are in common between both results.



The SQL INTERSECT operator is used to return the results of 2 or more SELECT statements. However, it only returns the rows selected by all queries or data sets. If a record exists in one query and not in the other, it will be omitted from the INTERSECT results.

Intersect Query

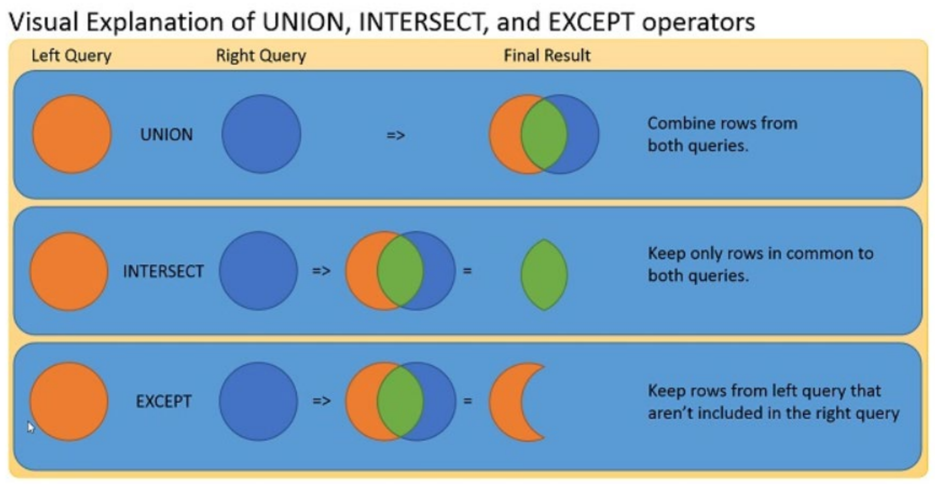
For example, using the search tool, find accessions with specific observation values for two different traits. Example: find *kernel color* **White** and *primary race* **Corn Belt Dent**.

```
@crop.name = 'Maize' AND @crop_trait_lang.title = 'Primary Race' AND
@crop_trait_code_lang.title = 'Corn Belt Dent'
INTERSECT
@crop.name = 'Maize' AND @crop_trait_lang.title = 'KERNEL COLOR' AND
@crop_trait_code_lang.title = 'White'
INTERSECT
@site.site_id IN (16) AND @inventory.is_distributable = 'Y' AND @inventory.is_available = 'Y'
```

A similar, but faster version of the query, using the trait IDs:

```
@crop_trait_observation.crop_trait_id = 89001 AND @crop_trait_code_lang.title = 'Corn Belt
Dent'
INTERSECT
@crop_trait_observation.crop_trait_id = 89027 AND @crop_trait_code_lang.title = 'White'
INTERSECT
@site.site_id IN (16) AND @inventory.is_distributable = 'Y' AND @inventory.is_available = 'Y'
```

Besides INTERSECT, UNION and EXCEPT can be used to fine tune searches.


Visual Explanation of UNION, INTERSECT, and EXCEPT operators
Left Query
Right Query
Final Result
UNION
=>
Combine rows from both queries.
INTERSECT
=>
=
Keep only rows in common to both queries.
EXCEPT
=>
=
Keep rows from left query that aren't included in the right query

INTERSECT Example: Looking for aronia accessions that have an available inventory and have an inventory with an image attached, available or not. That requires fancier SQL, such as an INTERSECT.

This search produces incorrect results:

```
@taxonomy_genus.genus_name = 'aronia'
```

and @accession.status_code = 'ACTIVE'
and @accession.is_web_visible = 'Y'
AND @inventory.is_distributable = 'Y'
AND @inventory.is_available = 'Y'
AND @accession_inv_attach.category_code = 'IMAGE'

With INTERSECT, the search produces correct results:

@taxonomy_genus.genus_name = 'aronia'
and @accession.status_code = 'ACTIVE'
and @accession.is_web_visible = 'Y'
INTERSECT @inventory.is_distributable = 'Y' AND @inventory.is_available = 'Y'
INTERSECT @accession_inv_attach.category_code = 'IMAGE'

EXCEPT FUNCTION

Returns any distinct values from the query to the left of the EXCEPT operator that are not also returned from the right query.

The following EXCEPT query is used to track the orders when a curator has been alerted (**action_name_code = 'CURALERTED')** about an NC7 order **(site_id = 16)**, but he has not cleared it and the order is still pending (the curator hasn't cleared the order (**action_name_code = 'CURCLEARED').**

@site.site_id IN (16) AND @order_request.completed_date IS NULL AND
@order_request_action.action_name_code = 'CURALERTED' AND
@order_request_action.cooperator_id IN (122186)

EXCEPT

@order_request_action.action_name_code = 'CURCLEARED' AND
@order_request_action.cooperator_id IN (122186)

-- Millard is 122186

LEN function

The LEN function determines the string length. This could be used to find long plant names

WHERE LEN(accession_inv_name.plant_name) > 36

DateDiff function to find recent viabilities

```
WHERE datediff(day, inventory_viability.tested_date, getdate() ) < 180

@inventory_viability.inventory_viability_id LIKE '%'
AND datediff(day, inventory_viability.tested_date, getdate() ) < 180
```

Subqueries

A subquery is a query within a query – the inner query is resolved first.

Can be used in various ways, such as to search by specific owner

Ex: **@accession.owned_by IN (SELECT cooperator_id FROM cooperator WHERE last_name = 'Millard')**

Ex2: A nested subquery for site name:

```
@accession.owned_by IN
  (SELECT cooperator_id FROM cooperator WHERE site_id =
    (SELECT site_id FROM site WHERE site_short_name = 'NC7'))
```

Ex3: A query using COUNT to find rows with many inventories (from one accession)

in the Search Tool or dynamic folder:

```
@ taxonomy_genus.genus_name = 'Zea'
AND (SELECT COUNT(*) FROM inventory i WHERE i.accession_id = accession.accession_id) > 32
```

in a SQL query:

```
WHERE taxonomy_genus.genus_name = 'Zea'
AND (SELECT COUNT(*) FROM inventory i WHERE i.accession_id = accession.accession_id) > 32
```

Ex4: A query using a dataview's calculated field COUNT to determine the number of orders with a specified number of items for a specified year:

```
@order_request.completed_date LIKE  '%2019%'
AND @site.site_short_name = 'NC7'
AND (SELECT count(*) FROM order_request_item WHERE order_request_id =
order_request.order_request_id) >=250
```

DISTINCT

The SELECT DISTINCT statement is used to return only distinct (different) values. Inside a table, a column often contains many duplicate values; and sometimes you only want to list the different (distinct) values.

Example: List More than 2 inventory owners

```
WHERE taxonomy_genus.genus_name = 'Zea'
AND (SELECT COUNT(distinct i.owned_by) FROM inventory i WHERE i.accession_id =
accession.accession_id) > 2
```

NOT EXISTS

Similar to EXCEPT…

The EXISTS operator is used to test for the existence of any record in a subquery. The EXISTS operator returns true if the subquery returns one or more records. If a subquery returns any rows at all, EXISTS *subquery* is TRUE, and NOT EXISTS *subquery* is FALSE.

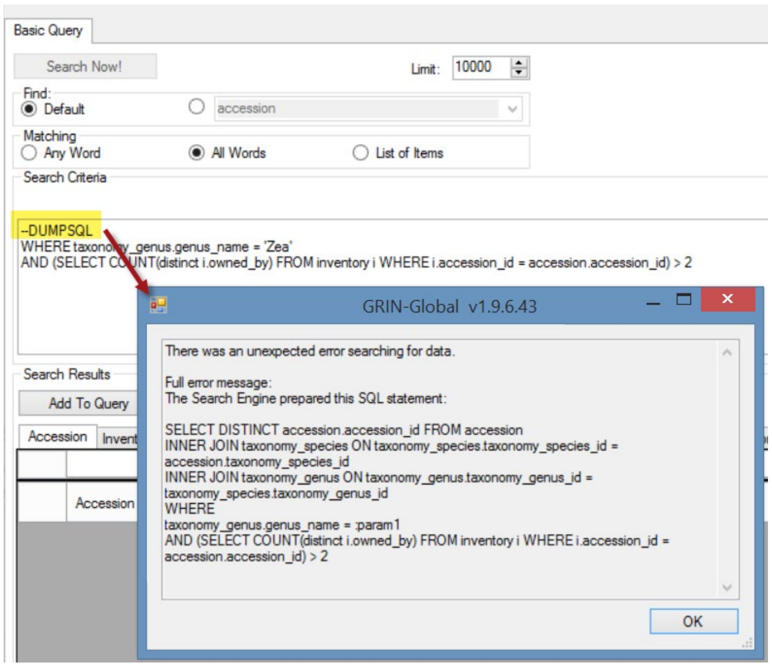
SELECT *column_name(s)*
FROM *table_name*
WHERE EXISTS
(SELECT *column_name* FROM *table_name* WHERE *condition*);

Example: Find records without a recent viability test

```
WHERE
inventory.inventory_id IS NOT NULL  /* necessary if resolving outside inventory */
AND NOT EXISTS
(SELECT * FROM inventory_viability iv
  WHERE iv.inventory_id = inventory.inventory_id    -- link subquery to main select
  AND datediff(day, iv.tested_date, getdate() ) < 365)
```

Displaying the SQL: --DUMPSQL

- SE4 has an option to deliberately throw an error and show the SQL it generated when the first line of the query is this comment: **--DUMPSQL**



```
--DUMPSQL
WHERE taxonomy_genus.genus_name = 'Zea'
AND (SELECT COUNT(distinct i.owned_by) FROM inventory i WHERE i.accession_id = accession.accession_id) > 2
```

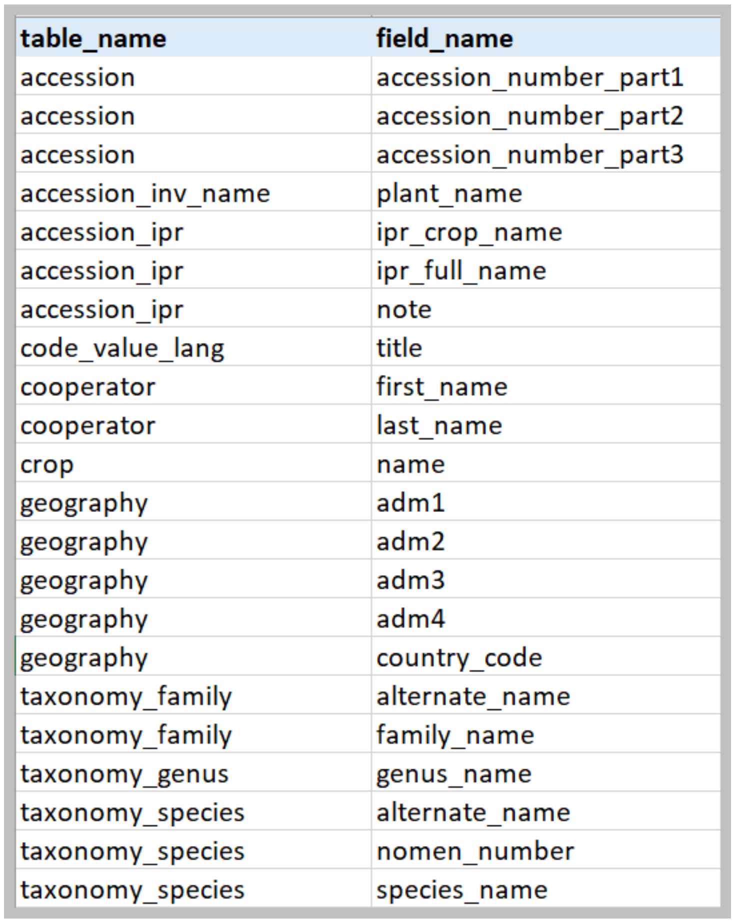
```
There was an unexpected error searching for data.

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 (SELECT COUNT(distinct i.owned_by) FROM inventory i WHERE i.accession_id =
accession.accession_id) > 2
```

Appendix A: Fields used in the GG Searches

The GRIN-Global administrator can determine which fields are to be searched using two different approaches. GG "Auto" fields may be designated in the **sys_search_autofield** table. The second method requires the GG administrator to use SQL Server's Full Text Indexing methodology.

Autofields

The following fields were designated by the National Plant Germplasm System (NPGS) GG administrator to be used for text box searches. (Every GG genebank can determine what fields are to be included.)

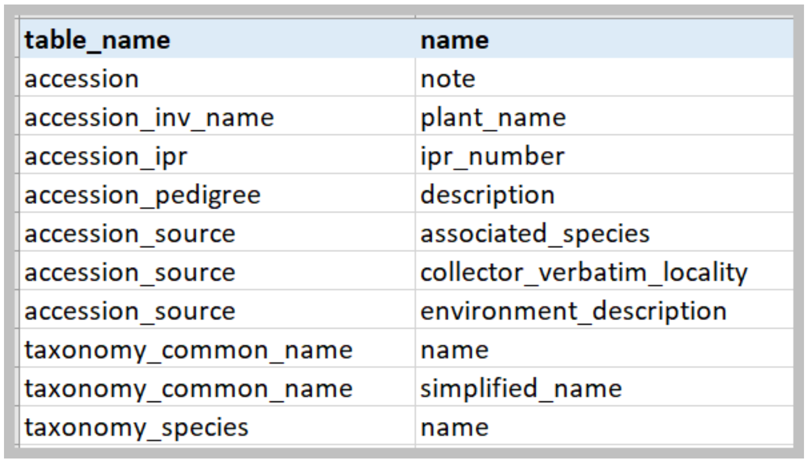


table_name	field_name
accession	accession_number_part1
accession	accession_number_part2
accession	accession_number_part3
accession_inv_name	plant_name
accession_ipr	ipr_crop_name
accession_ipr	ipr_full_name
accession_ipr	note
code_value_lang	title
cooperator	first_name
cooperator	last_name
crop	name
geography	adm1
geography	adm2
geography	adm3
geography	adm4
geography	country_code
taxonomy_family	alternate_name
taxonomy_family	family_name
taxonomy_genus	genus_name
taxonomy_species	alternate_name
taxonomy_species	nomen_number
taxonomy_species	species_name

SQL to List the "Autofields" Used in the Search Box

```
SELECT table_name, field_name
  FROM sys_search_autofield ssa
  JOIN sys_table_field stf ON stf.sys_table_field_id = ssa.sys_table_field_id
  JOIN sys_table st ON st.sys_table_id = stf.sys_table_id
  ORDER BY 1,2
```

Full Text Indexing
The fields listed below were indexed by the National Plant Germplasm System (NPGS) GG administrator.



table_name	name
accession	note
accession_inv_name	plant_name
accession_ipr	ipr_number
accession_pedigree	description
accession_source	associated_species
accession_source	collector_verbatim_locality
accession_source	environment_description
taxonomy_common_name	name
taxonomy_common_name	simplified_name
taxonomy_species	name

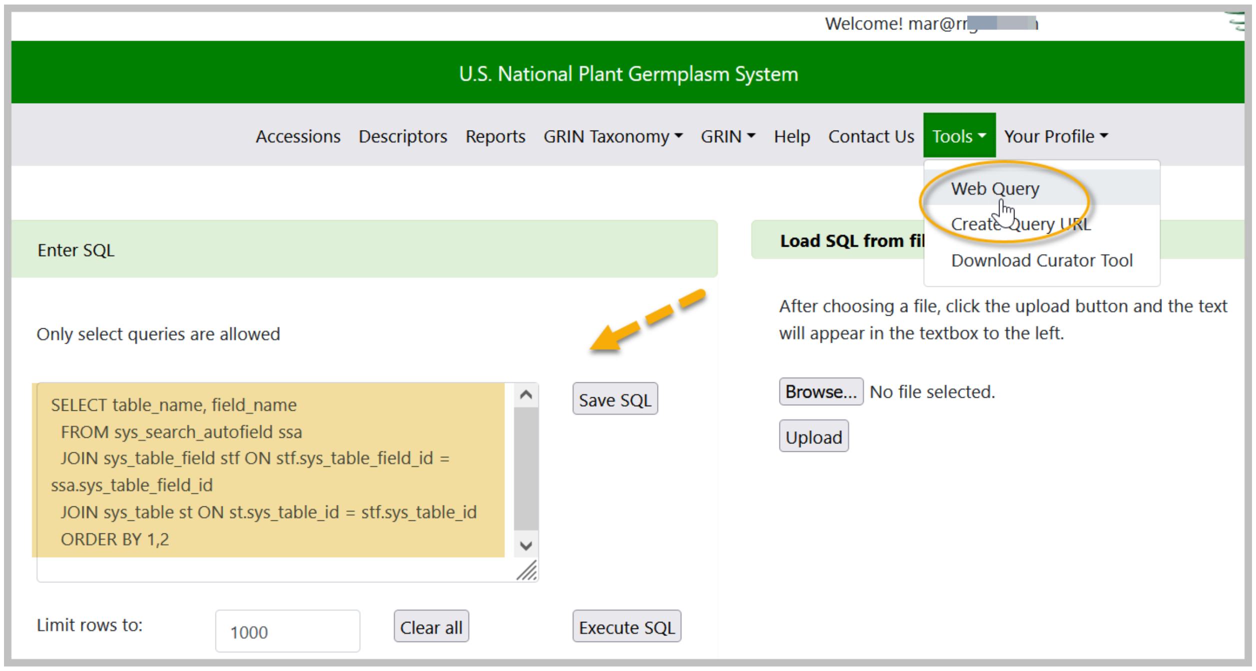
SQL to List the Fields Having Full Text Indexes

```
SELECT DISTINCT
   object_name(fic.[object_id])as table_name,
   [name]
FROM
   sys.fulltext_index_columns fic
   INNER JOIN sys.columns c
      ON c.[object_id] = fic.[object_id]
      AND c.[column_id] = fic.[column_id]
```

Appendix B: SQL Queries on the Public Website

Overview

Genebank staff who have had their Public Website account connected to their Curator Tool account by their GG administrator, when logged into the Public Website, will have the **Tools** option visible on the menu. From there, select **Web Query** to display the box for inputting SQL. Log in; select **Tools** | **Web Query** You can copy or type valid SQL in the box as shown:



Only select queries are allowed

```
SELECT table_name, field_name
  FROM sys_search_autofield ssa
  JOIN sys_table_field stf ON stf.sys_table_field_id =
ssa.sys_table_field_id
  JOIN sys_table st ON st.sys_table_id = stf.sys_table_id
  ORDER BY 1,2
```

After choosing a file, click the upload button and the text will appear in the textbox to the left.

You can open a .txt or Word file in which SQL has been stored and cut in paste into the query box, or use the PW feature to Create a query SQL text file.

3 Basic Components

In general, in GRIN-Global, most SQL statements will use these three words.

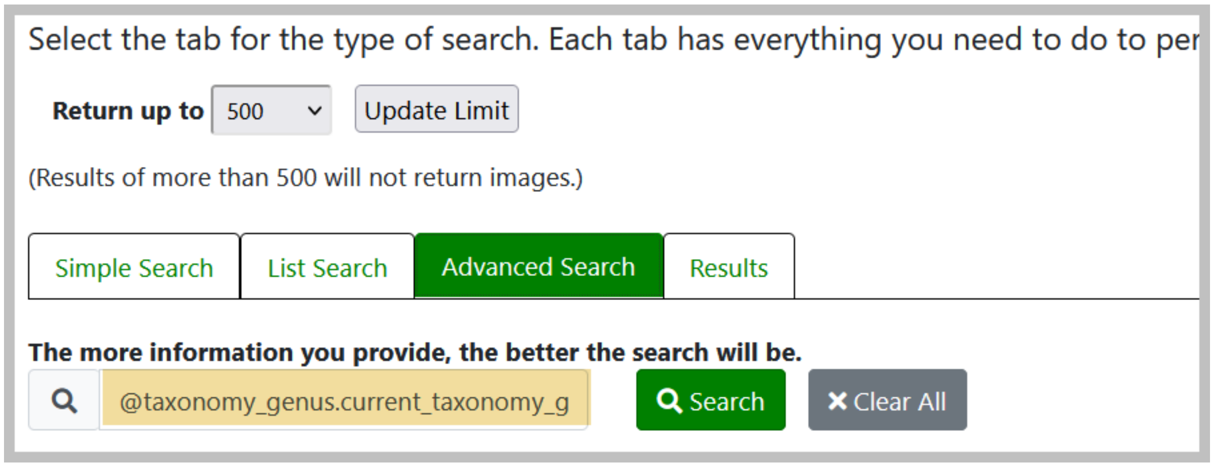
SELECT – what columns to display
FROM – what tables to search
WHERE – what criteria

In a valid SQL command, indicate what data you want to display and the conditions. In the GRIN-Global Public Website, a user cannot modify data – only read. Statements such as INSERT or DELETE do not work on the PW page.

Online there are multiple documents, tutorials, and examples on how to use SQL queries on the Public Website. See https://www.grin-global.org/sql_examples.htm.

Public Website Searches Using the @

On the Public Website, you can also use @ search constructs. While it is not user friendly, if you know the actual table and field names, using these searches provides more search capabilities on the Public Website. Also, internal genebank staff can share these constructs with external users when appropriate.



Select the tab for the type of search. Each tab has everything you need to do to per

Return up to 500 Update Limit

(Results of more than 500 will not return images.)

The more information you provide, the better the search will be.

@taxonomy_genus.current_taxonomy_g

Appendix C: Administrator Notes on Sorting Search Results

There are three levels of sort on the output of Public Website searches:

1. Highest weighted field hits first (genus hits before others)
2. Accessions with PI prefixes are listed before Non-PIs*
3. Most recently received accessions are listed first

* Organizations other than NPGS that are running GRIN-Global may set the preferred prefix from "PI" to their organizations preferred prefix.

If there are more than 500 (or whatever your maximum limit is set to) accessions that are genus hits on PI numbers, the most recent of those is first. If there are less than 500 PI records for the genus you are searching, going to see recent non-PI genus hits further down the list and recent PI non-genus hits even further down. That is, not all recent accessions will be at the top because the other sorts have a higher precedence.

If as administrator of a GG system you want to change any of that behavior, you'll need to know how the sorting is controlled.

The first sort is by the weights of the freeform text fields, controlled by the **get_search_autofields** dataview. The weights assigned to autosearch fields can be adjusted in the following CASE clause:

```
  , CASE

      WHEN field_name IN ('genus_name', 'title')  THEN 0

      WHEN field_name IN ('species_name', 'adm1',
                'accession_number_part1')         THEN 1

      ELSE 2

     END AS weight
```

Note that "title" refers to the country name from the code lang translation. So hitting on genus name or country name are equally weighted, then species name, state name, or accession prefix for the next level, then the rest of the autosearch fields and finally the full text index hits (not controlled by the dataview).

The other two levels of sort are controlled by the PW dataviews web_search_overview_2 and web_search_overview_noimages_2 with an ORDER BY clause at the end of the dataview:

```
-- Put PI numbers first, then sort by date received

ORDER BY CASE WHEN a.accession_number_part1 = 'PI' THEN 0 ELSE 1 END,
COALESCE(a.initial_received_date, a.created_date) DESC, pi_number
```

Actually there is a fourth level of sort by PI number if the received date is exactly the same. Another system could change that ORDER BY to whatever suits them.

The Search Tool retrieves the data in a different fashion. So the sort order as described above for the Public Website doesn’t apply.

Appendix D: Document Change Notes

– June 26, 2025
- Additional information regarding IS NULL

– August 27, 2024
- editing / wording changes

– July 26, 2024
- editing / minor wording changes

– December 21, 2022
- added Note regarding BETWEEN
- also, corrected BETWEEN example

– June 21, 2022
- added Appendix B and details regarding PW sort priority preferences

– February 25, 2022
- added a dynamic query section with a link to the online Dynamic Query guide

– January 10, 2022
- mainly formatting changes

– July 12, 2021
- enhanced the section regarding calculating the actual quantities of viable seeds

– April 20, 2021
- formatted the table headings for the reserved words; therefore the headings are now included in the TOC

– February 2, 2021
- elaborated on the three search types; added screen examples

– October 1, 2020
- added note on comments
- enhanced notes on using search text on the Public Website

– September 20, 2020
- enhanced List Search notes

– August 12, 2020
- expanded information on BETWEEN

– February 29, 2020

- added use case searching using Live Seed (a calculated field)

– April 24, 2019

- changed example and wording for the WHERE clause

– December 17, 2018

- changed example and wording for the WHERE clause