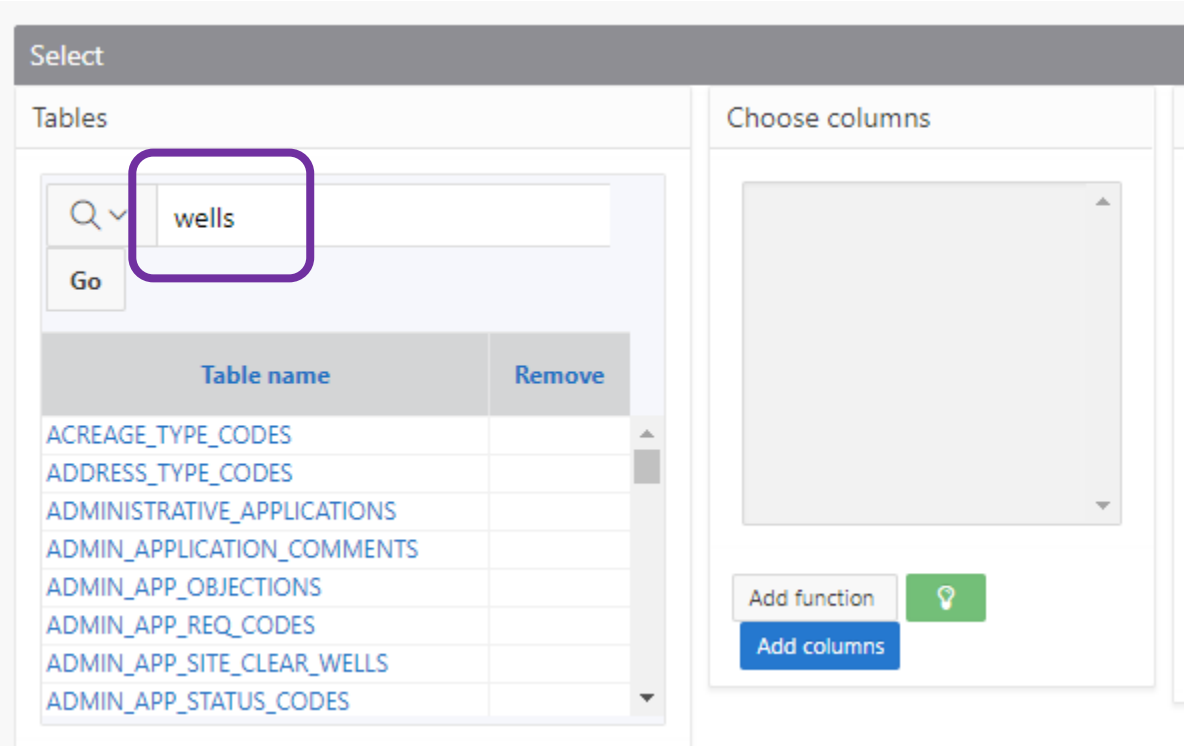
DODA

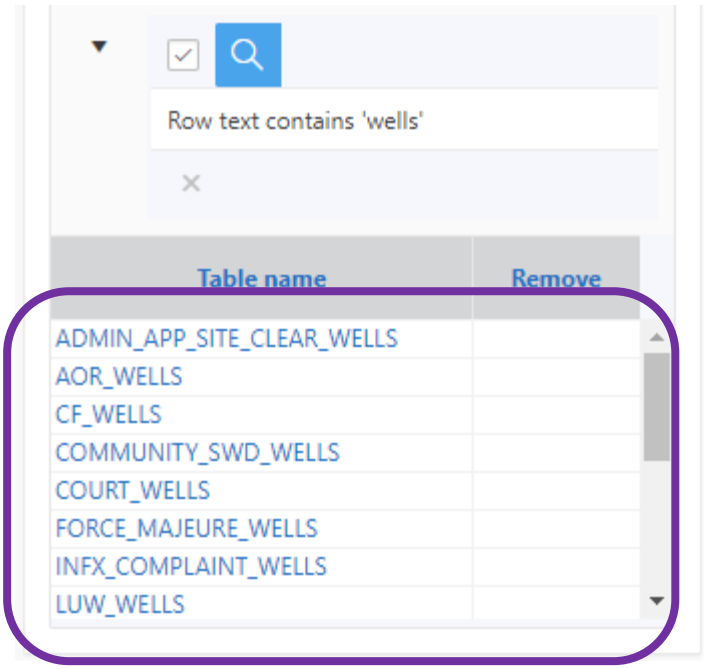
Ad-hoc Query Building – Beginner's Guide

Familiar building simple queries in DODA? If so, skip down to the following pages for more complex options and explanations of some of the fields in DODA:

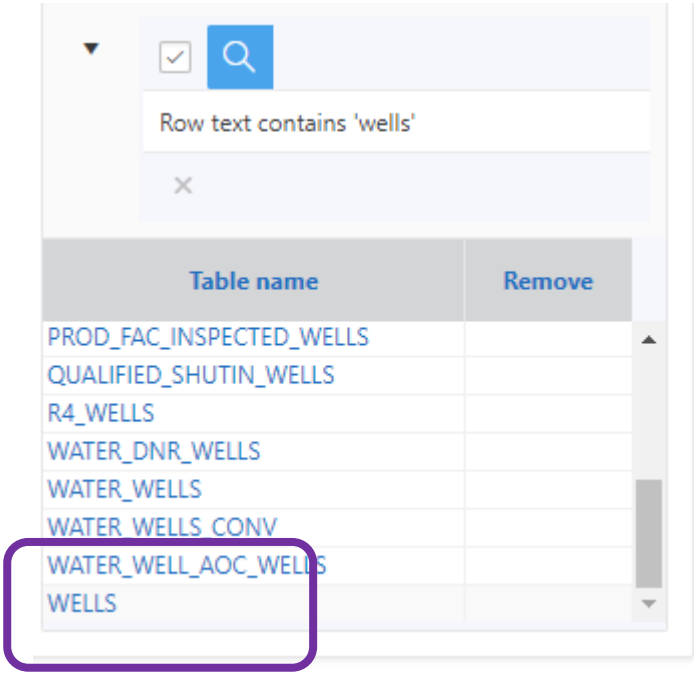
Are the above bullet points Greek to you? If so, follow these steps! To begin building an ad-hoc query, start by selecting the table of interest. When building queries, it is useful to consider the parent-child relationship of the data. For instance, if my goal is to identify wells by status code, I will start with reference to wells then pull in the associated status code description table (ie, the status code description does not stand alone; we need a well record to make sense of the status description). In ad-hoc, users can search for the table name or scroll through the table names. To search, key in a portion of the table name…



…then select *Go* in the application or select Enter on your keyboard.

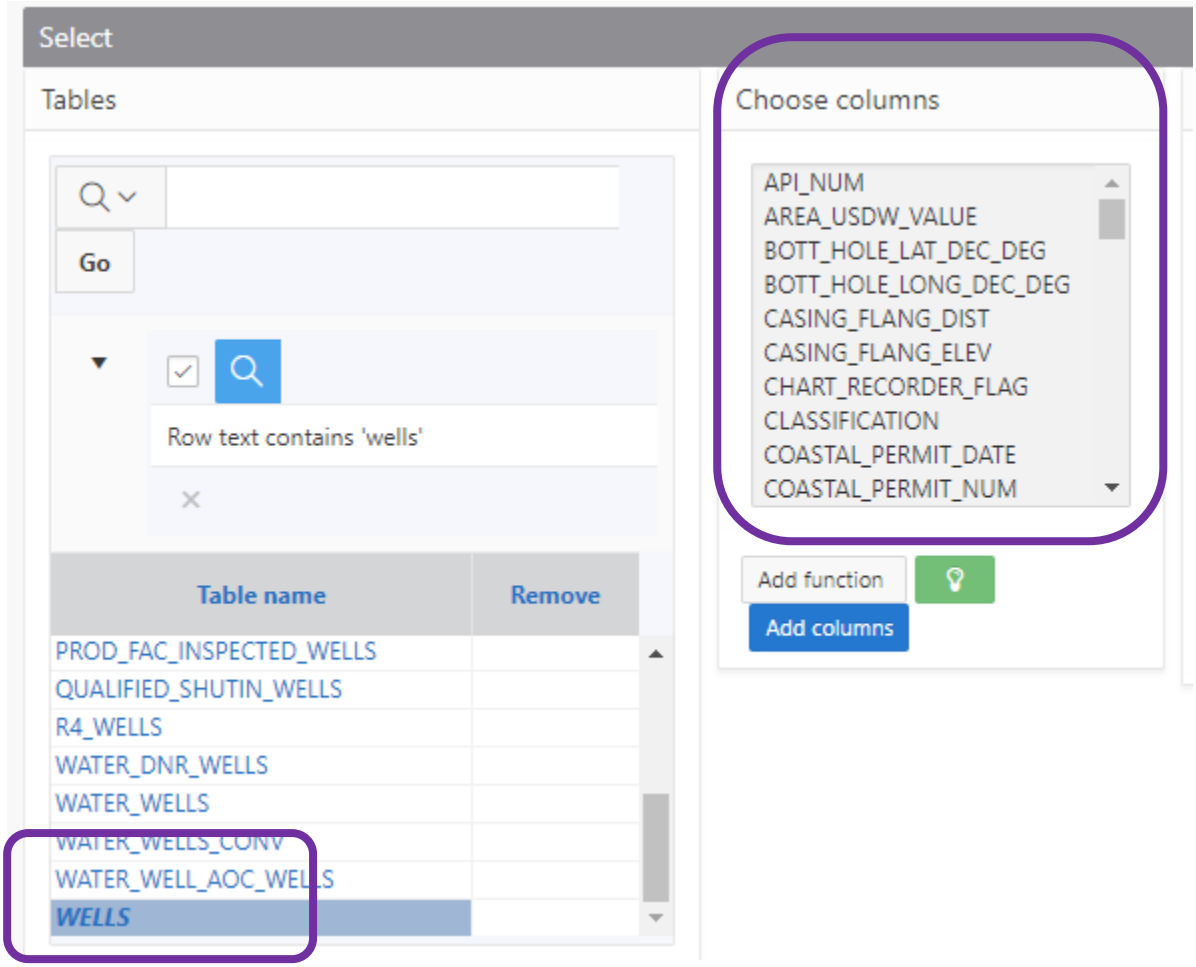


All tables with "wells" in the table name populates. My target is WELLS, so I will scroll down and select that table name.

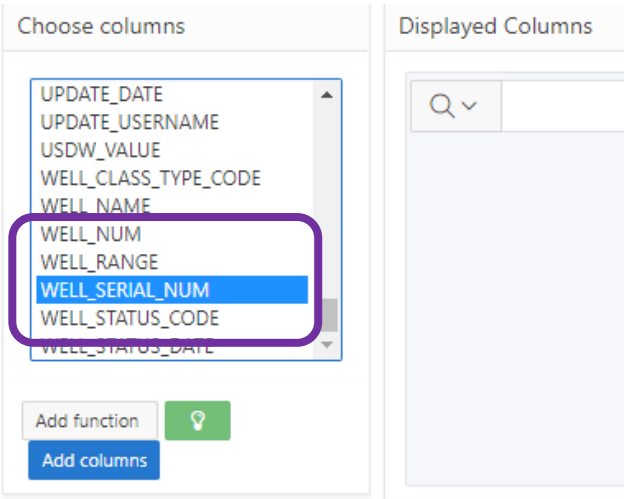


- Note- not all tables are immediately available in DODA. If you do not see a known table, put in a ticket to have the table available in DODA.

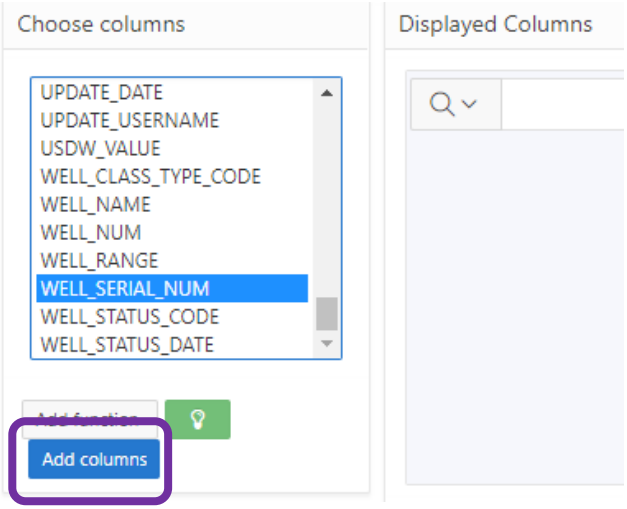
Once the table name is selected, the table name changes to ***bold, italic*** to reflect the selection and columns within the table display in the field to the right.



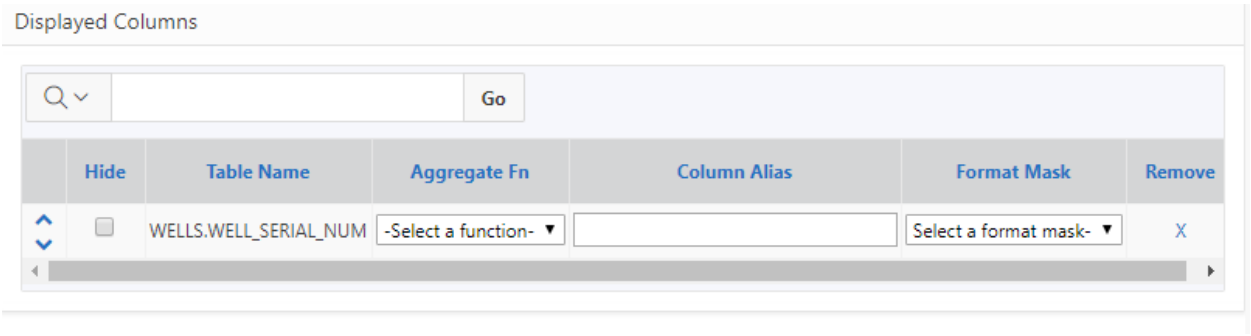
Scroll to the desired column; select with cursor to highlight…



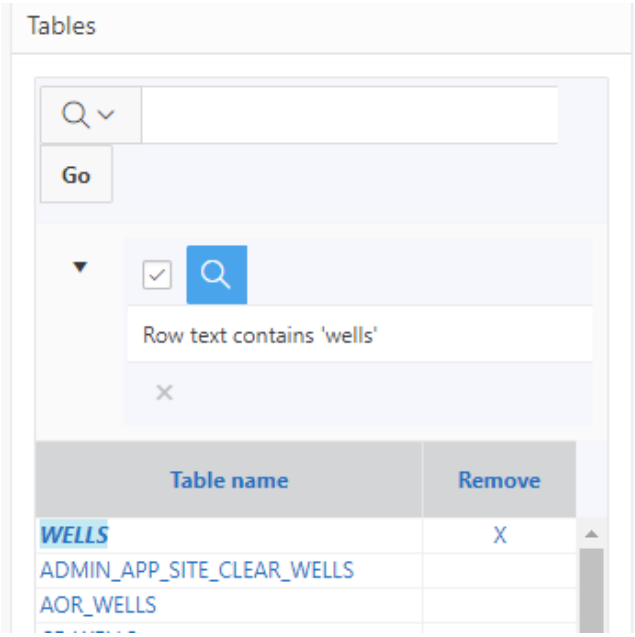
…then select *Add columns* to add the column to the query.



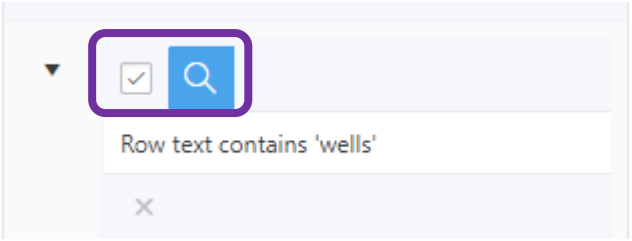
Once selected and added to the query, the column populates in the *Displayed Columns* field and can be utilized to display in the results or be utilized as a condition to develop the query.



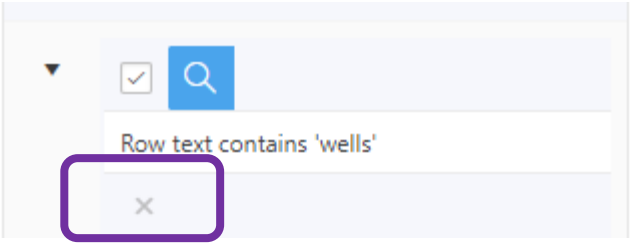
- Note- the selection of the column moves the table name to the top of the *Tables* field, reflecting the table association in the query.



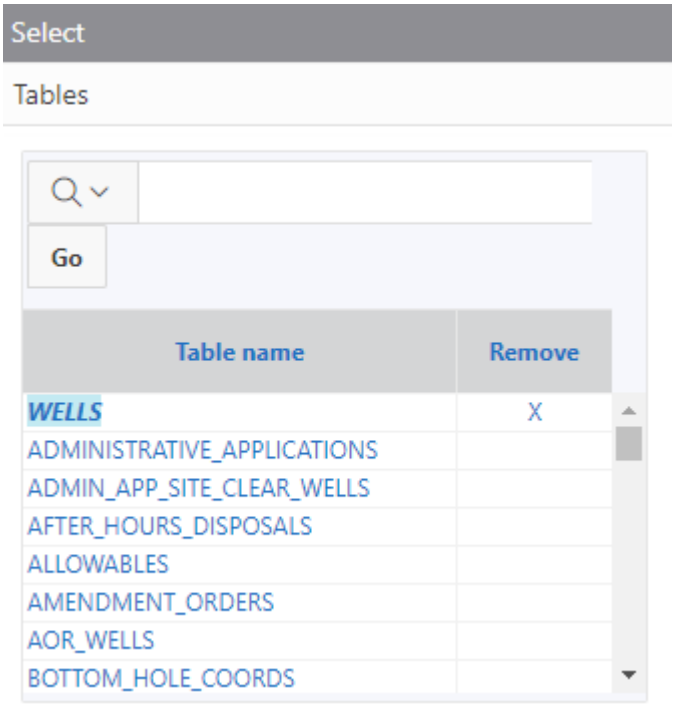
In the *Tables* field, if we clear the 'wells' filter by selecting the display option to clear…



…or removing the filter altogether by selecting the X…

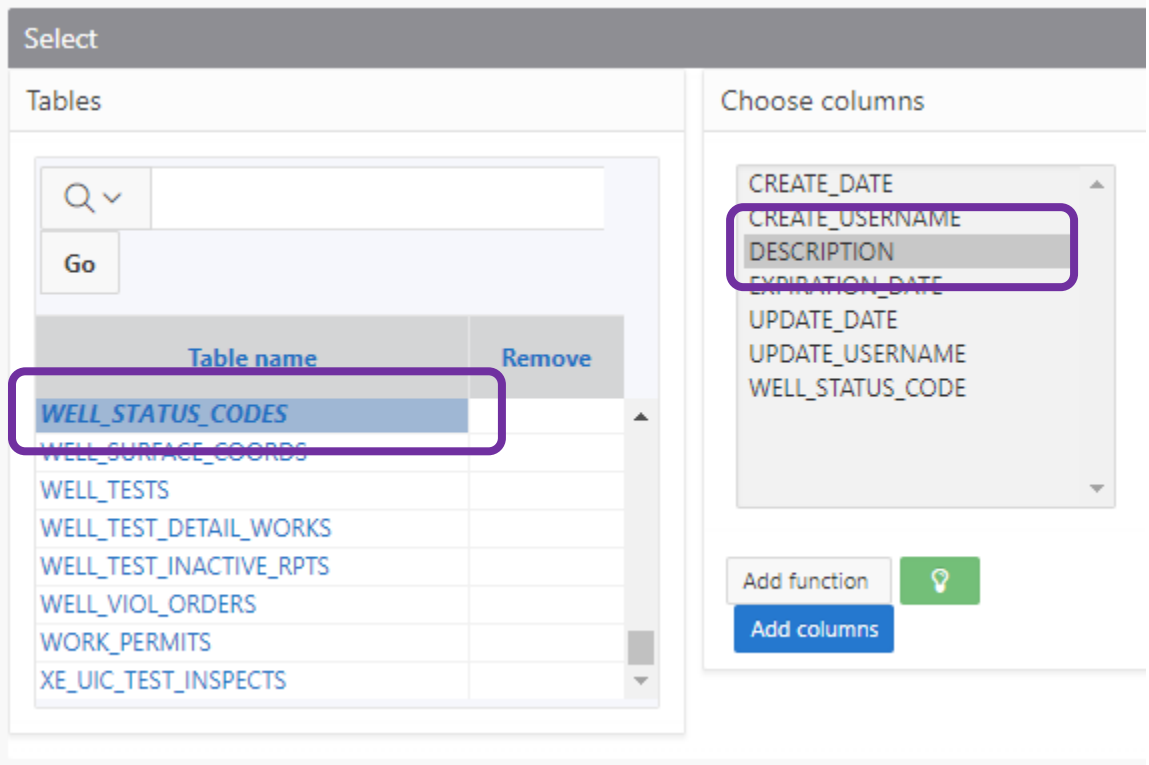


…other tables display. The tables that display are those that have an established join in DODA to other tables based on the tables selected.

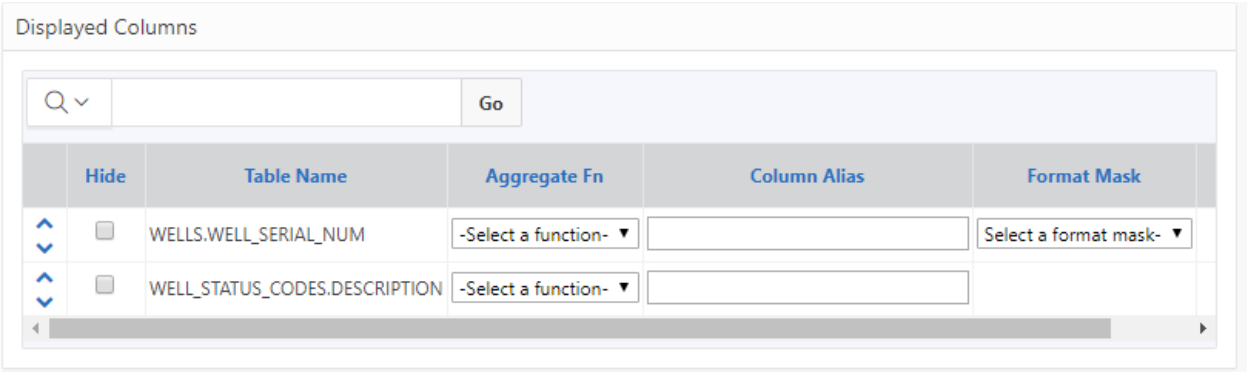


- Note – a join is the relationship between tables. Joins are set up by DODA Administrators. If a join does not exist, put in a ticket for the join to be made.
    - In order for a join to be established, there must be a common column in the tables to join the data. WELL_SERIAL_NUM is referenced in many tables, which is why so many joins exist for the column; hence the multiple results in the *Tables* field.

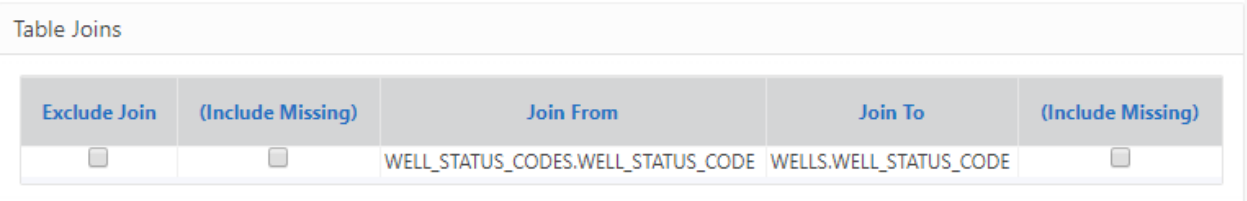
In addition to WELLS.WELL_SERIAL_NUM, I need status code for my query. In the available tables, I see WELL_STATUS_CODES. I select the table name to display the available columns, then select DESCRIPTION to add to my query…



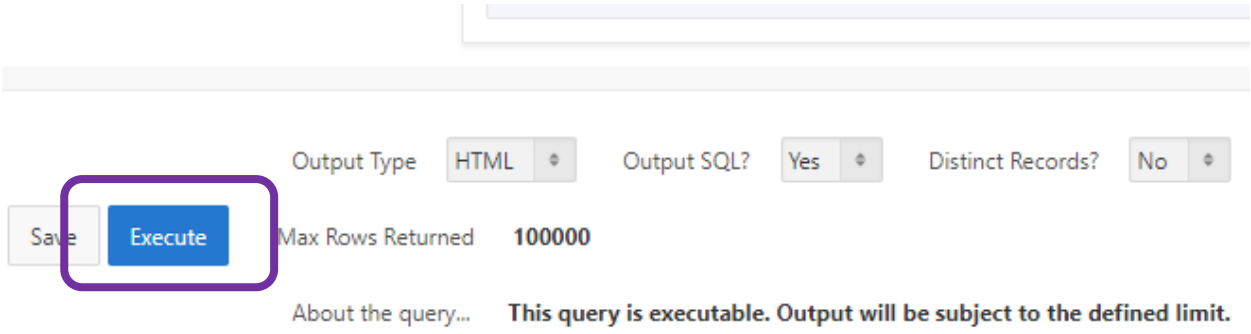
…then, select *Add columns*. The WELL_STATUS_CODES.DESCRIPTION is now in the *Displayed Columns* field…



…and in the middle of the page, the join is displayed.

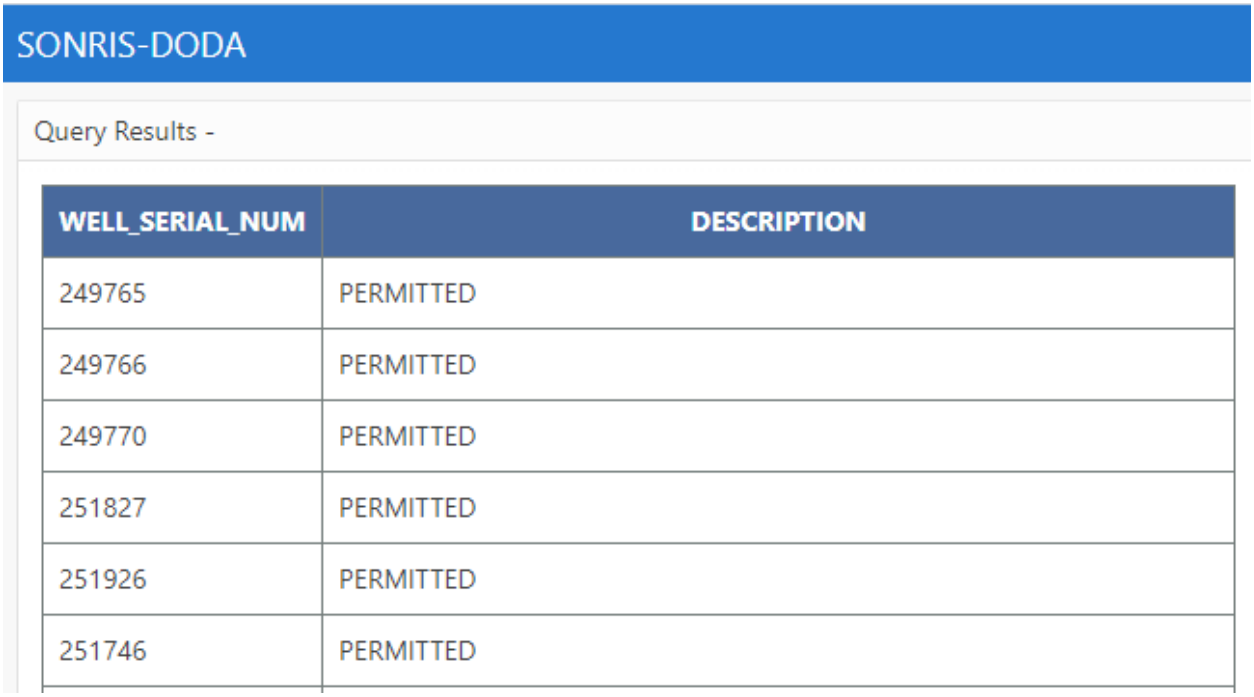


Scroll to the bottom of the ad-hoc page. Select *Execute* to execute the query.



- Note – default export in ad-hoc is HTML, but Excel and CSV are other options. Excel maintains the database format of the columns (ie, keeps characters as characters even if values display) whereas CSV reduces the values to numbers. Results will populate in a new window. DODA is limited to 100,000 rows of data in the results. Please note that DODA can evaluate more than 100,000 rows of data by adding conditions and functions discussed later in this tutorial.
    - Once a query has been executed as HTML, make sure you aren't using the results window as another ad-hoc builder. If multiple windows are open by a user at a time for ad-hoc use, unexpected errors will occur.
    - If the *Execute* option is selected from the ad-hoc page as HTML while the results window is open, the new execution will override the existing results. Another results window will not open.
    - *About the query…* notifies the user if the query is executable or not.

Results populate in a new window.



WELL_SERIAL_NUM	DESCRIPTION
249765	PERMITTED
249766	PERMITTED
249770	PERMITTED
251827	PERMITTED
251926	PERMITTED
251746	PERMITTED

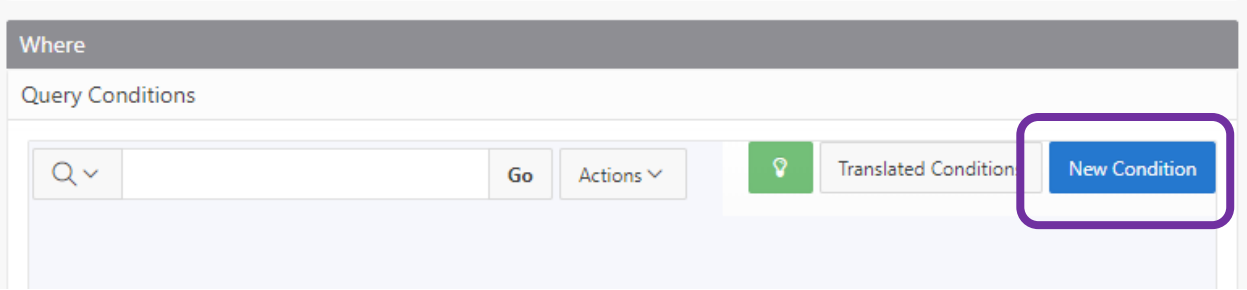
Scroll to the bottom of the page for the number of rows returned.

132908	DRY AND PLUGGED
132912	DRY AND PLUGGED
132914	DRY AND PLUGGED
100000 rows returned.	**Max row limit=100000**

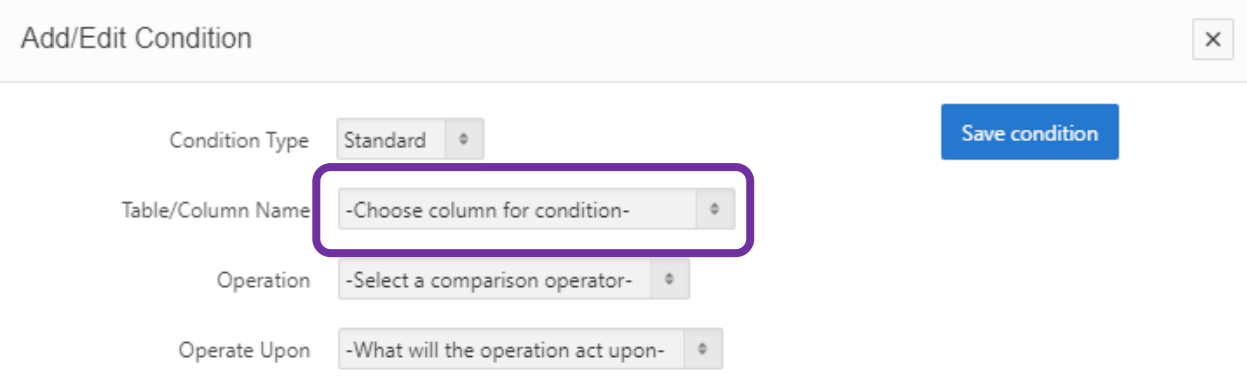
Query Name: adhoc

SELECT /**/ WELLS.WELL_SERIAL_NUM, WELL_STATUS_CODES.DESCRIPTION FROM WELLS, WELL_STATUS_CODES WHERE 1=1 AND WELL_STATUS_CODES.WELL_STATUS_CODE=WELLS.WELL_STATUS_CODE

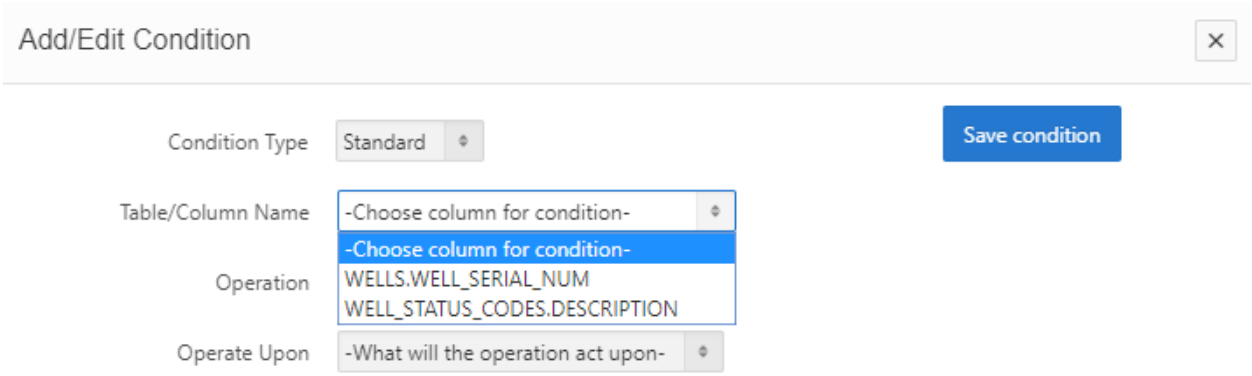
In this query, the maximum number of rows displayed. This implies there are more rows in the dataset than allowed to be displayed in DODA. To limit our results, we can add a condition. On the ad-hoc page, select *New Condition.*



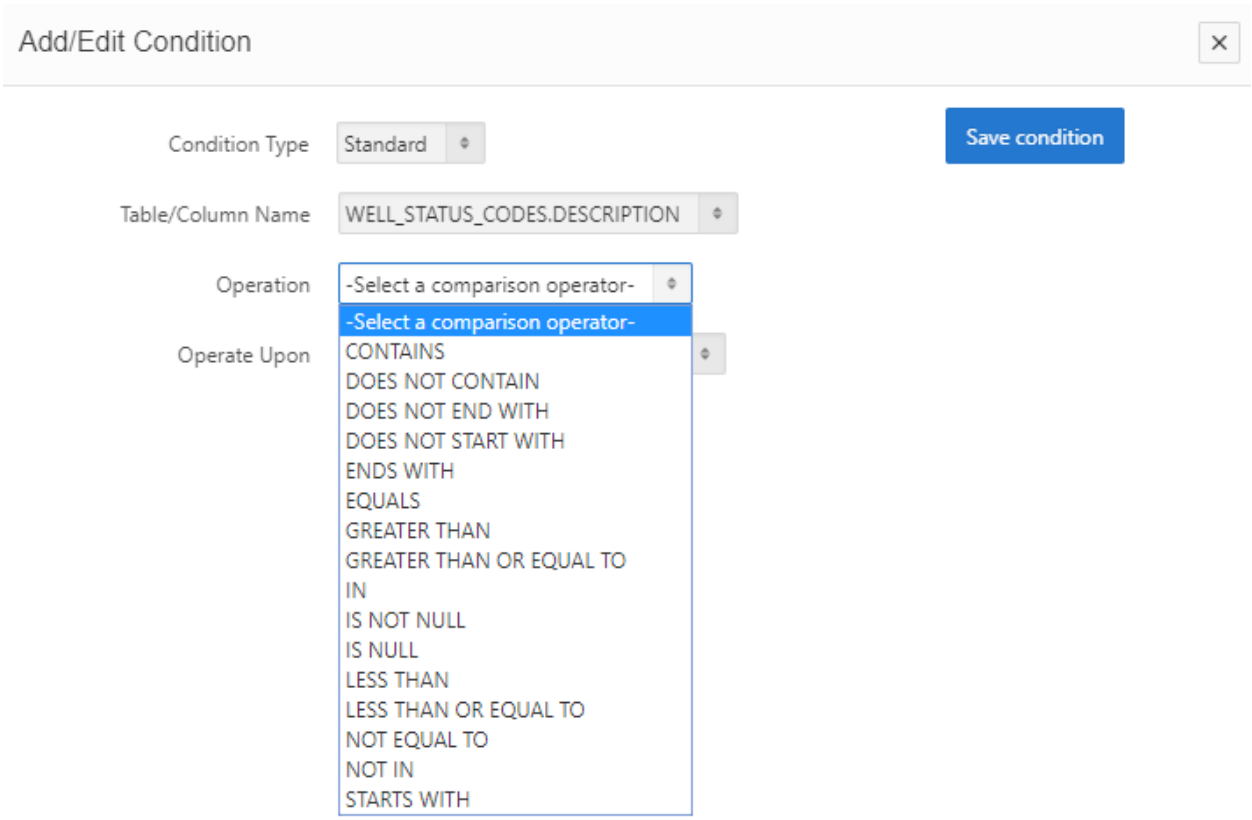
The default condition is *Standard*. *Standard* conditions utilize DODA to create the required SQL function. A column must be selected in the *Displayed Columns* field to build a standard condition.



In the *Table/Column Name* dropdown, our selections are displayed…



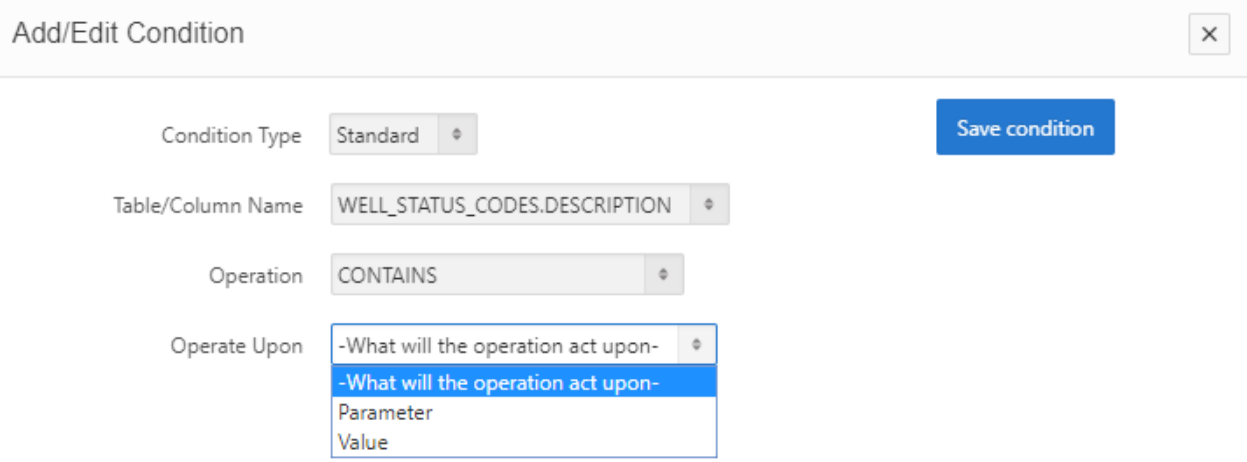
…we can select either to build our condition. We will select WELL_STATUS_CODES.DESCRIPTION to limit our results to one status code. Selecting *Operation* yields multiple built-in functions.



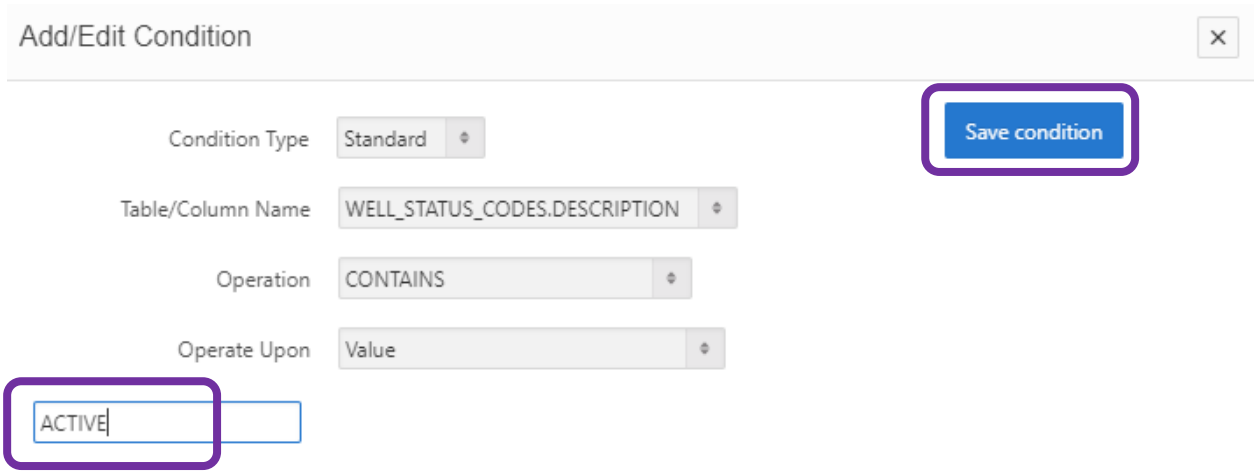
- Note – to query multiple status codes at once, "IN" must be utilized and the items must be separated by a colon.

In this case, we will utilize CONTAINS. This operator applies the % "wildcards" utilized in SONRIS forms.

Next, we will select *Operate Upon*. The options for this condition are Parameter and Value. When building queries, it is best to select Value to ensure the query is set up appropriately.

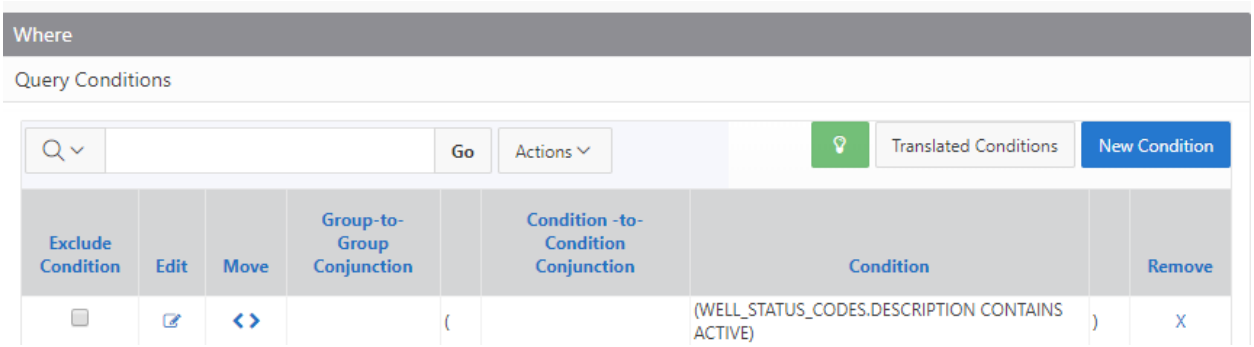


After selecting Value, we now have a field to key in. Our query will be wells with current status description that includes ACTIVE. After keying in the value, select *Save condition.*



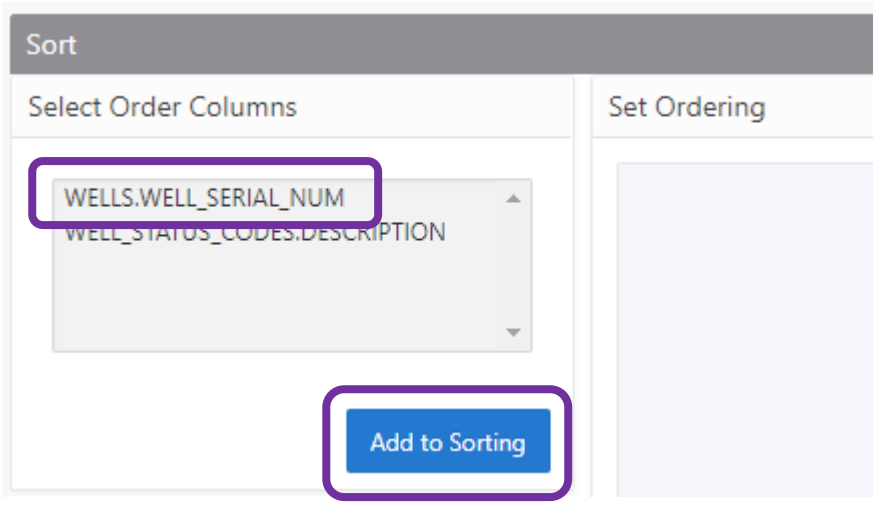
- Note – some database characters are case sensitive and some are not. Keying in the value "active" yields no results; however, capitalizing the value gives expected results.

Ad hoc window now displays the following:

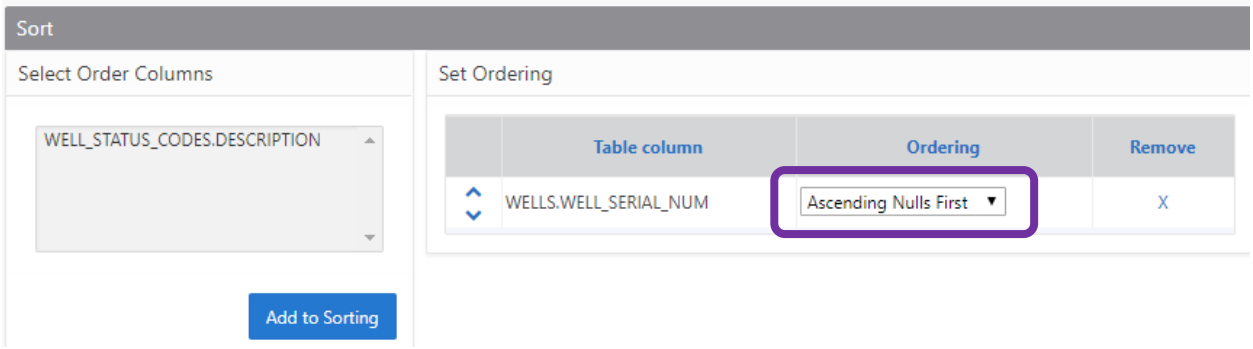


The condition can be edited, excluded, removed, or additional conditions can be added.

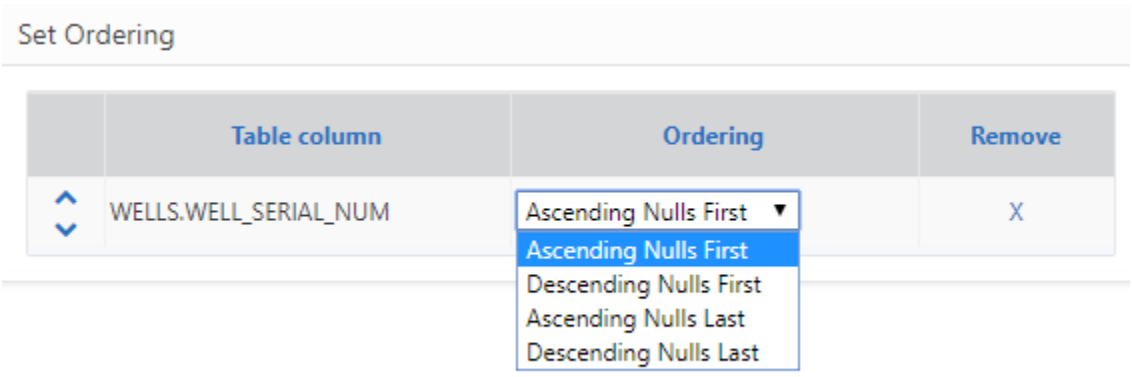
Scrolling down further, there is a *Sort* option. Select the item to be sorted, then *Add to Sorting*.



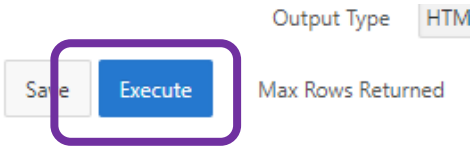
The default sort is displayed.



Selecting from the *Ordering* dropdown yields additional options.



Once the sort is selected, scroll down to the bottom of the window and select *Execute* to execute the query.



The results now display all well serial numbers with a status code description containing the word "ACTIVE" and is sorted by well serial number in an ascending fashion.

SONRIS-DODA

Query Results -

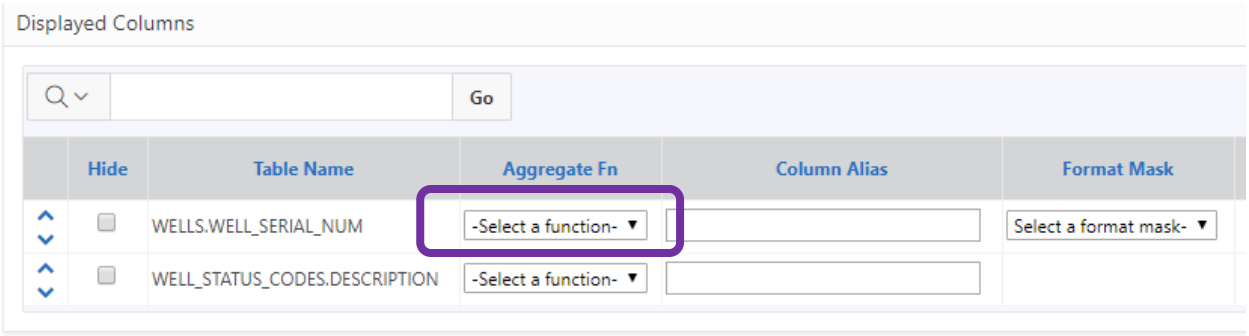
WELL_SERIAL_NUM	DESCRIPTION
233	ACTIVE - PRODUCING
257	ACTIVE - PRODUCING
594	ACTIVE - PRODUCING
725	ACTIVE - PRODUCING
801	ACTIVE - PRODUCING
901	ACTIVE - PRODUCING
1071	ACTIVE - PRODUCING

At the bottom of the results window, the results displayed are less than the maximum rows returned.

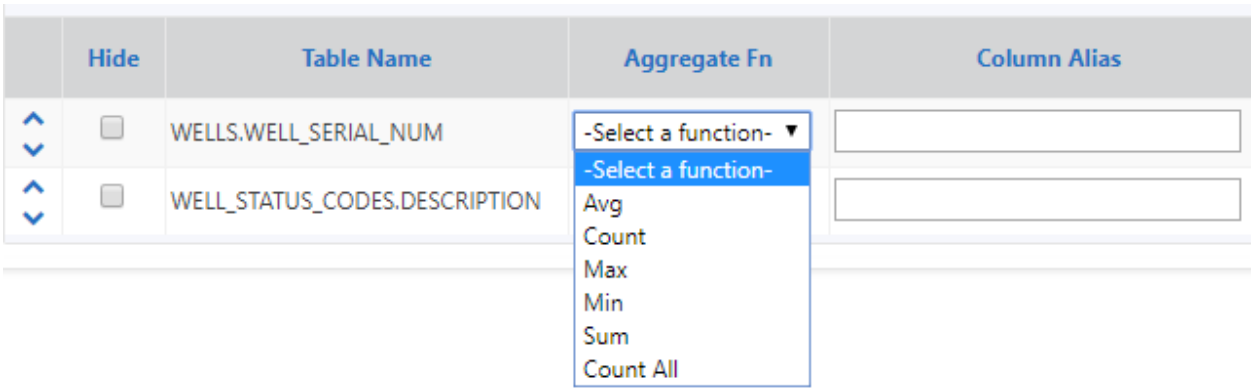
999996	ACTIVE - PRODUCING
999997	PA-35 TEMPORARY INACTIVE W
999998	ACTIVE - PRODUCING
999999	ACTIVE - PRODUCING
39486 rows returned.	**Max row limit=100000**

In addition to displaying results, DODA can also perform calculations in lieu of providing thousands of rows of data. Utilizing the same ad-hoc window, we can amend our query to display the number of well serial numbers that meet this condition.

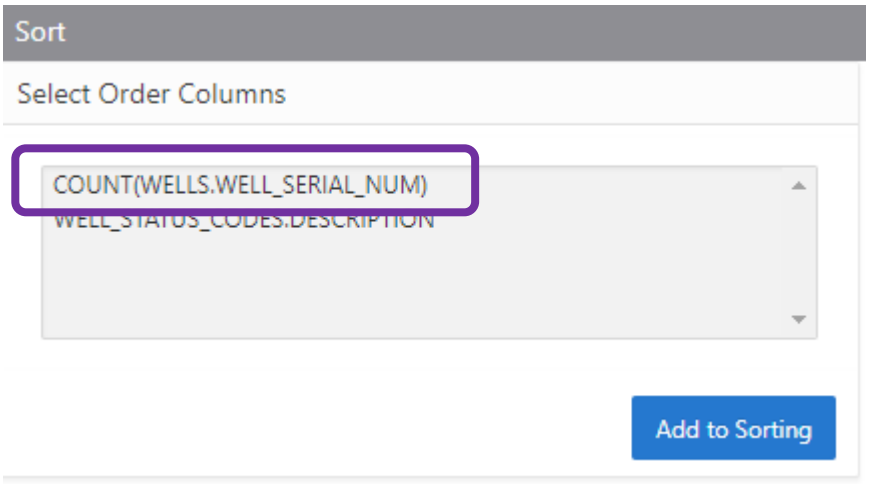
Scroll to the top of the ad-hoc page to the *Displayed Columns* portion. Select the dropdown for *Aggregate Fn* for the WELLS.WELL_SERIAL_NUM column.



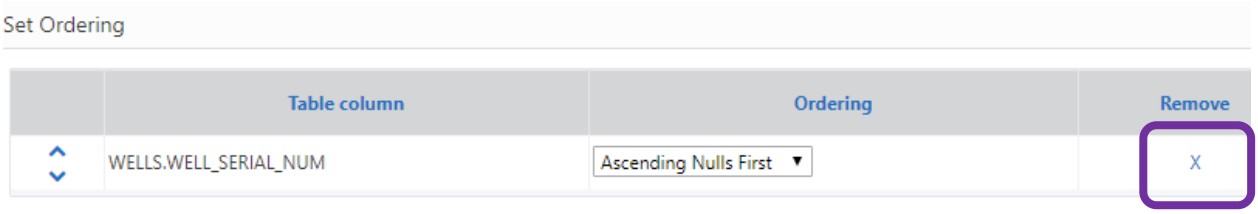
Multiple options display. To complete this task, we will select *Count.*



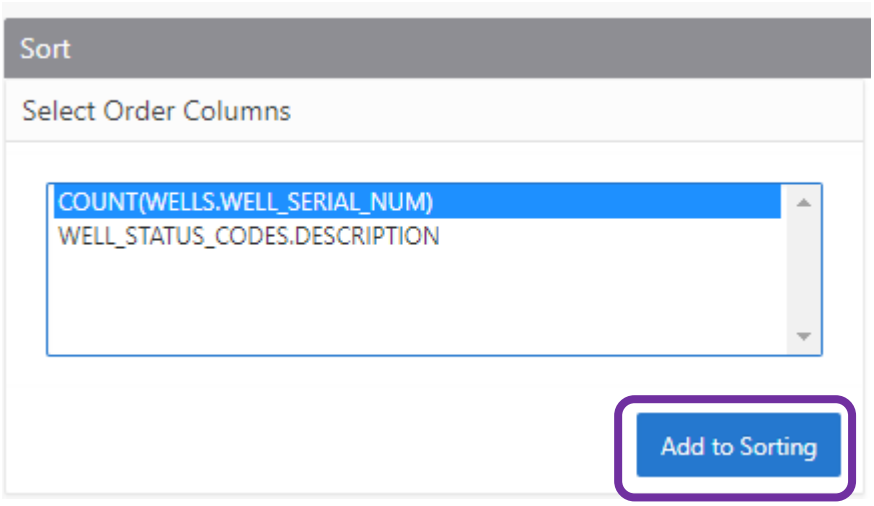
Scroll down to the bottom of the window. Since our WELLS.WELL_SERIAL_NUM count is now a count, the *Sort* reference to WELLS.WELL_SERIAL_NUM no longer applies to this query. See the change in the column reference:



The previous sort still displays to the right, but the results will not sort in that manner since WELLS.WELL_SERIAL_NUM has been amended to a count. We can remove the sort by selecting the X…



…and replace with the updated column from the available columns in the query.



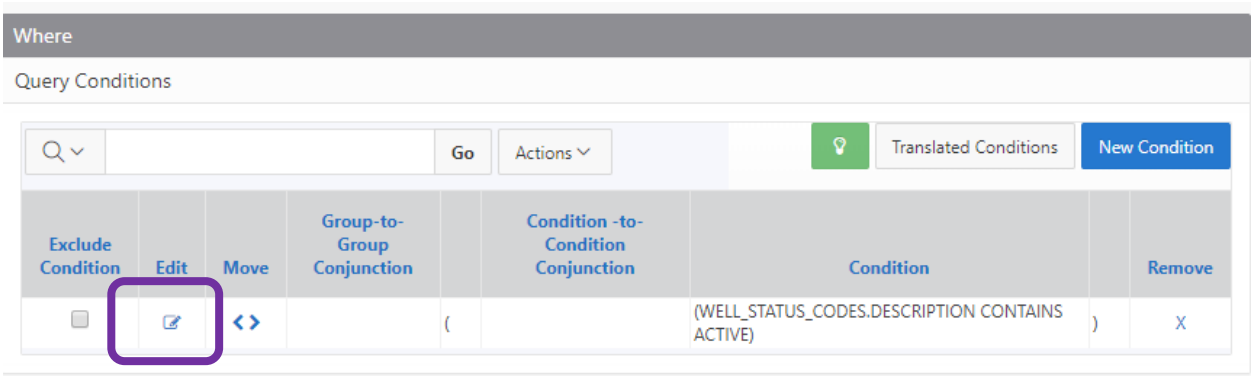
Leaving the default as Ascending, select execute; the results displays as follows:

SONRIS-DODA

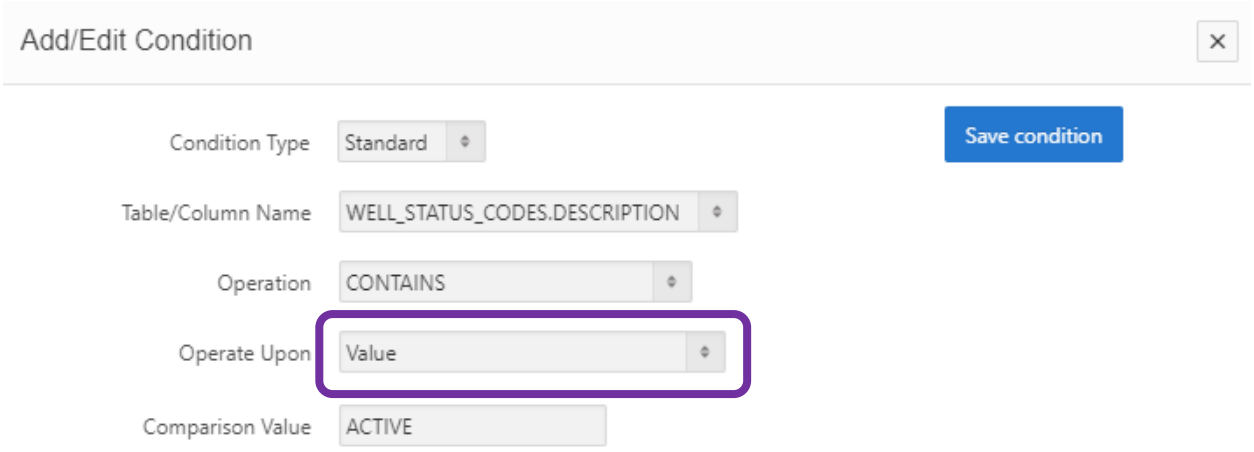
Query Results -

WELL_SERIAL_NUM_COUNT	DESCRIPTION
1	ACTIVE PRODUCING/CYCLIC INJCT
5	INACTIVE INJECTION WELL (COMMERCIAL OR OTHER)
5	INACTIVE WELL, NO RESP. PARTY
1191	PA-35 TEMPORARY INACTIVE WELL TO BE OMITTED FROM PROD.REPORT
4633	ACTIVE- INJECTION
33651	ACTIVE - PRODUCING
6 rows returned.	**Max row limit=100000**

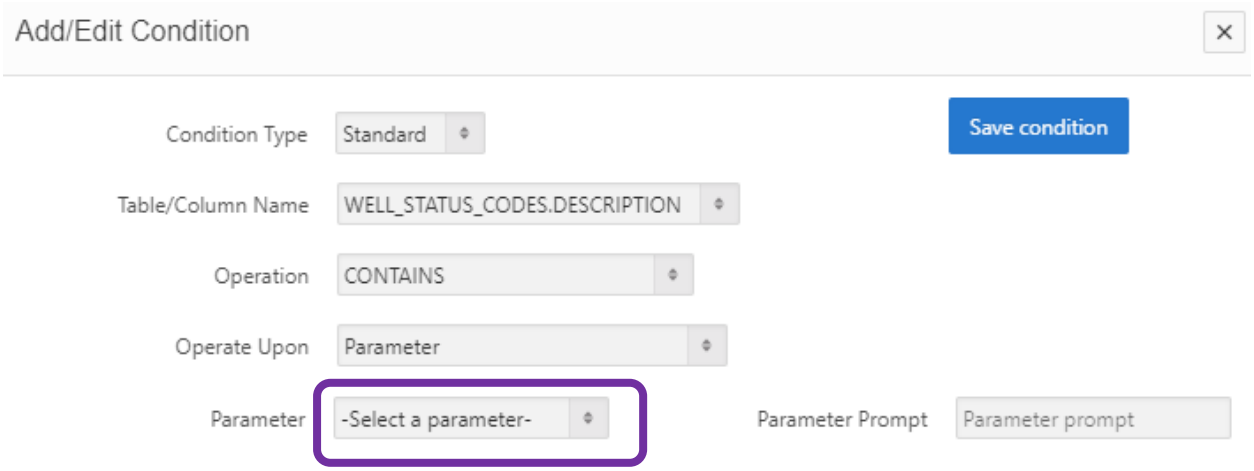
If we are satisfied with this query, we can modify the condition from *Standard* to *Parameter* to receive a count of well serial number by any type of well status code description. Return to the ad-hoc window and scroll up to the *Where* clause. Select *Edit.*



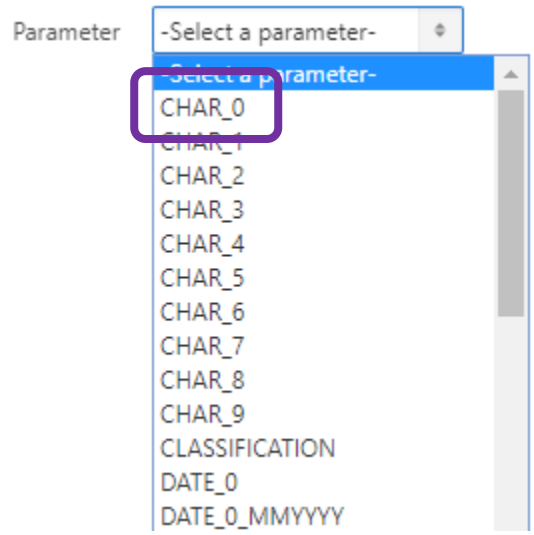
To change to a parameter, select *Operate Upon* to change from Value to Parameter.



Parameter prompts appear and replace the *Comparison Value* field. Select the *Parameter* drop down to define the type…

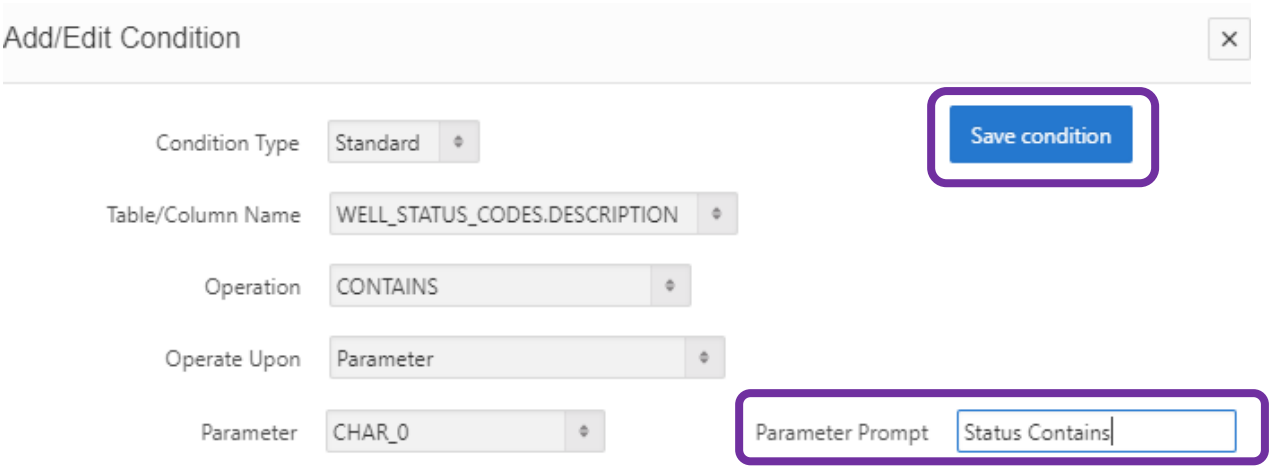


A list of available parameter options displays.

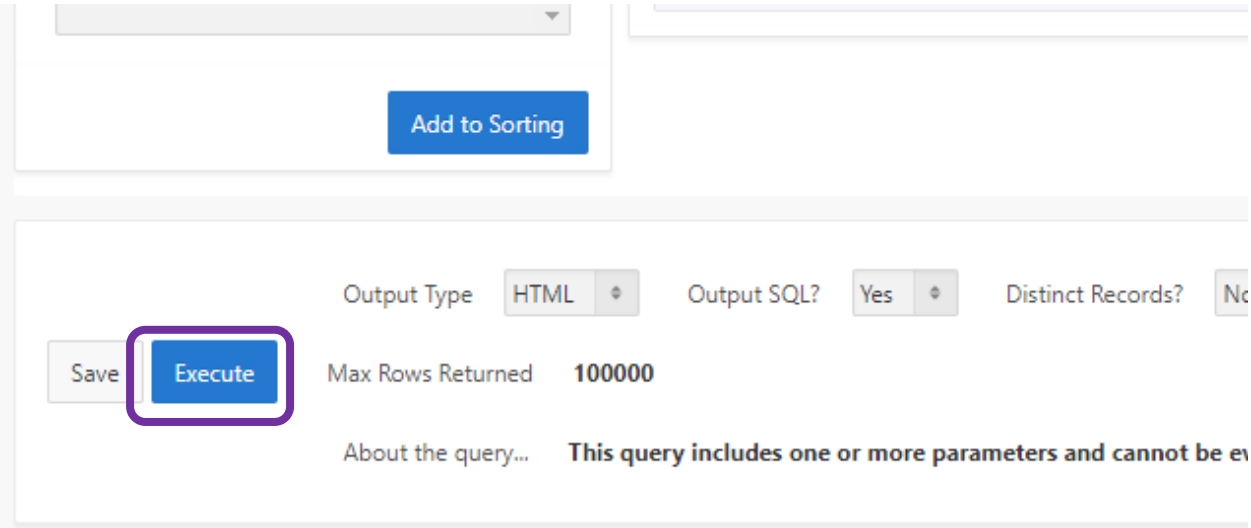


- Note – the parameter selection must reflect the data type. WELL_STATUS_DESCRIPTION is a character. As such, we will select CHAR_0 as our parameter.

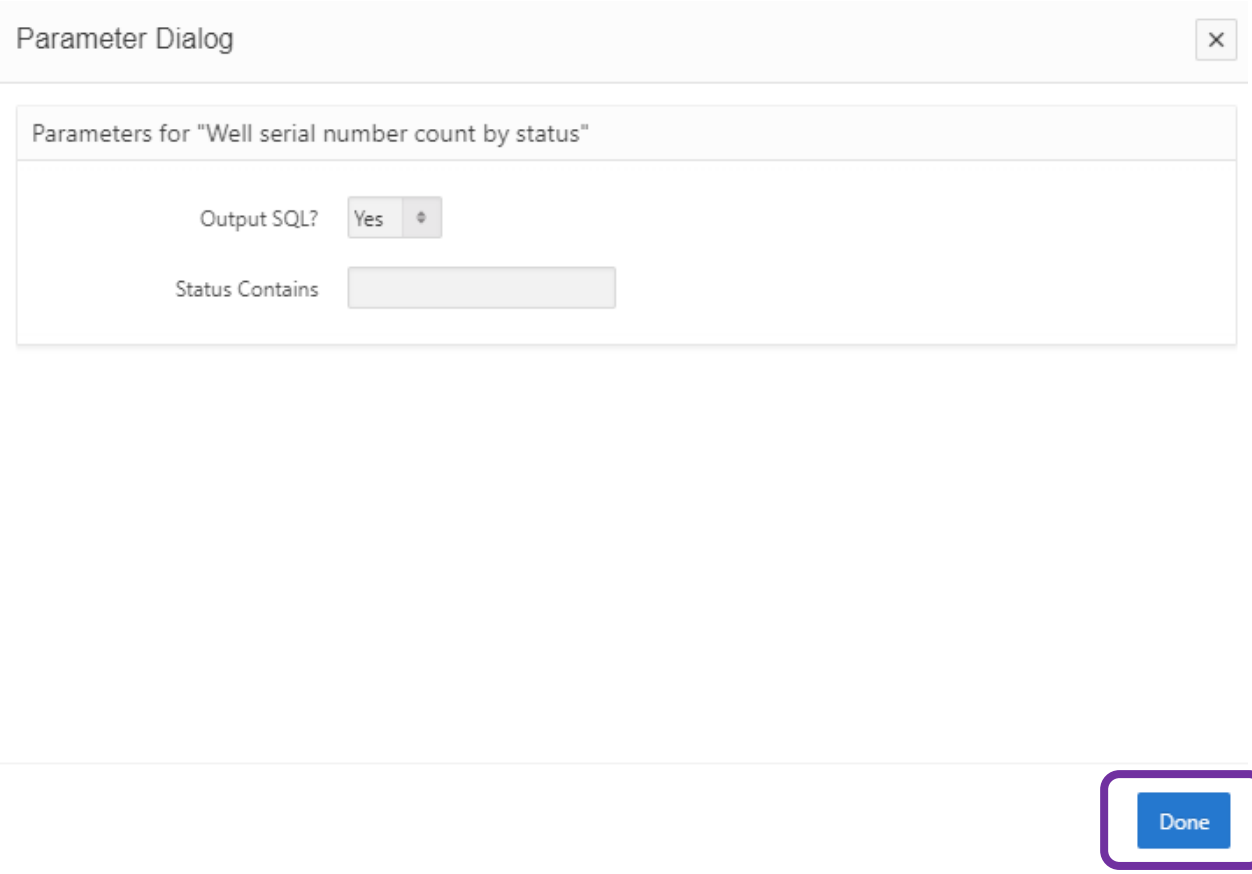
The *Parameter Prompt* field is what you decide will display to prompt the user to execute the query. Once completed, select *Save condition*.



Once selected, scroll down to the bottom of the window and select *Execute.*



The *Parameter Dialog* displays.



The default parameter setting is optional (can be set to required upon save), so I can execute the query by keying a value in the "Status Contains" field or leaving the field null. I'm choosing to leave the field null and simply selecting *Done*.

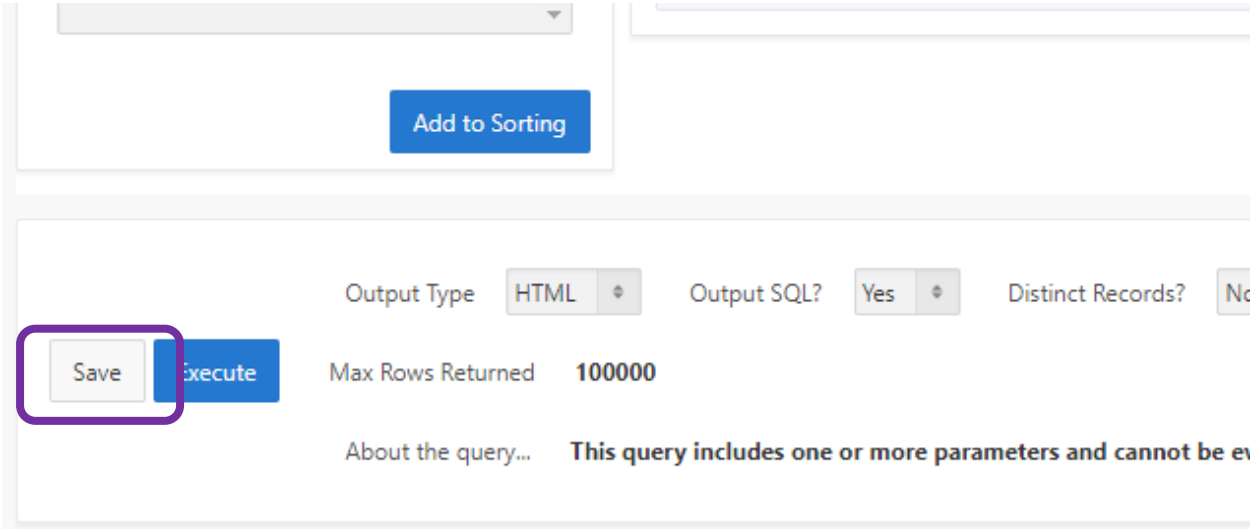
The results display in the *Results* window.

SONRIS-DODA

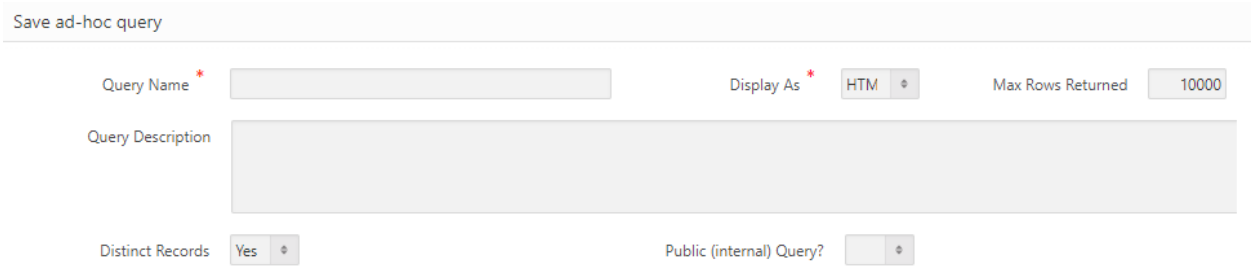
Query Results - Well serial number count by status

WELL_SERIAL_NUM_COUNT	DESCRIPTION
1	MULTIPLE COMPLETED/PA-35 WELL
1	ACTIVE PRODUCING/CYCLIC INJCT
4	SHUT-IN WAITING ON MARKET
5	NON-WELL/ FOR UIC MANIFEST ONLY
5	INACTIVE WELL, NO RESP. PARTY
5	INACTIVE INJECTION WELL (COMMERCIAL OR OTHER)
9	SL-STATE JUR,BHL-FED JUR

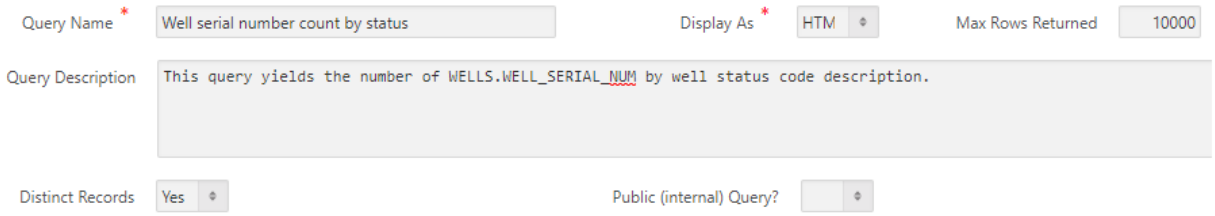
Since the query is satisfactory, it can be saved for future use. Return to the Ad-hoc builder page, scroll down to the bottom of the window and select *Save*.



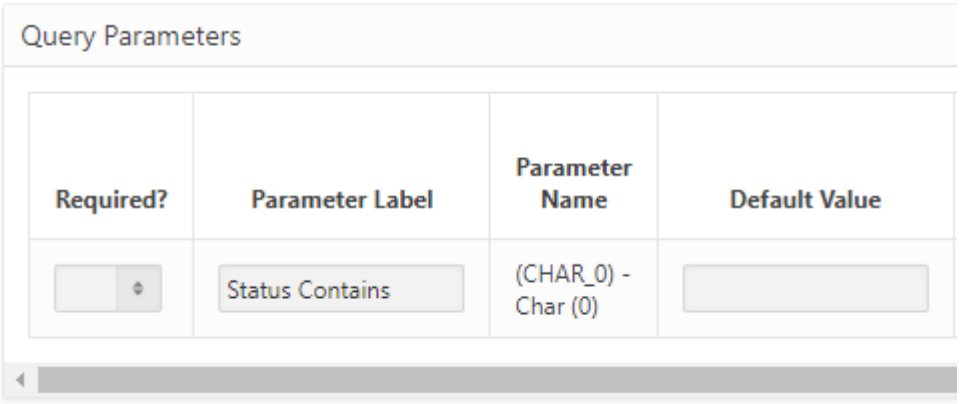
The *Save ad-hoc query* view appears. Items marked with a red asterisk are required.



- Note – *Display As* is the default display of the results. Users will have the option to toggle between CSV, Excel, and HTML. The default *Max Rows Returned* is 10,000. This field can be increased to 100,000. *Query Description* is a useful tool to help users identify what the query accomplishes without executing the query.

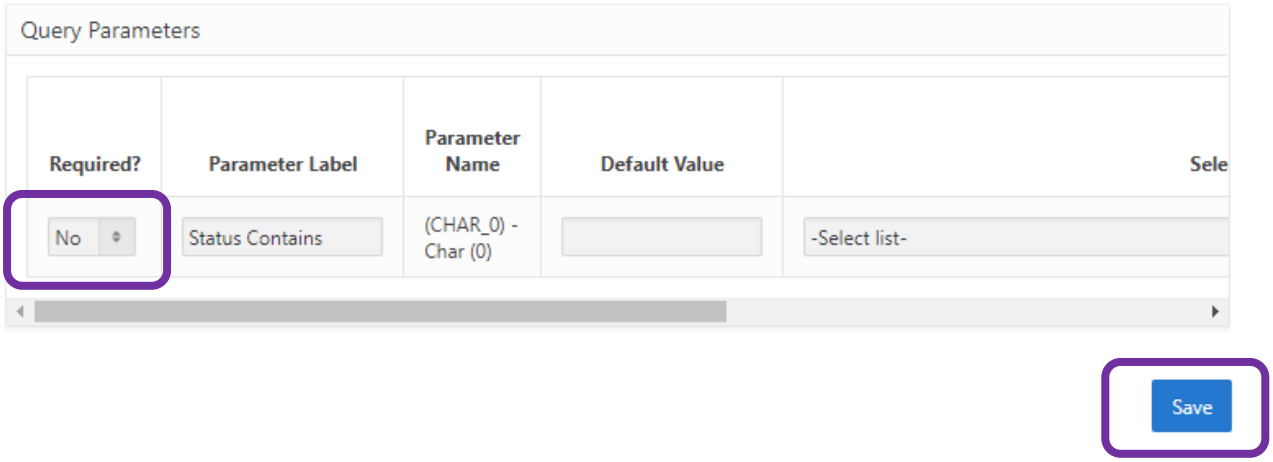


Beneath these fields is the *Query Parameters* section. This section will only display if a parameter is included in the query.

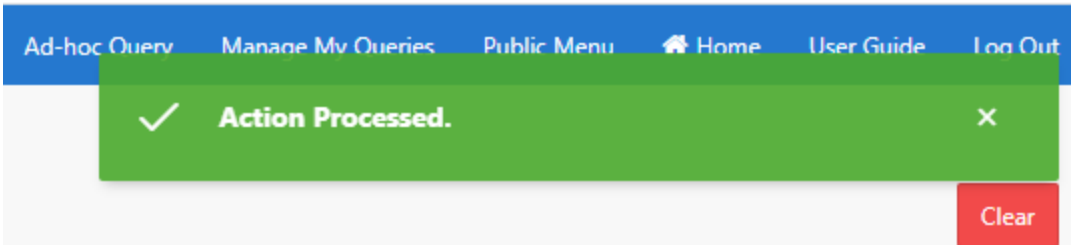


- Note – users can set the parameter as optional or required.

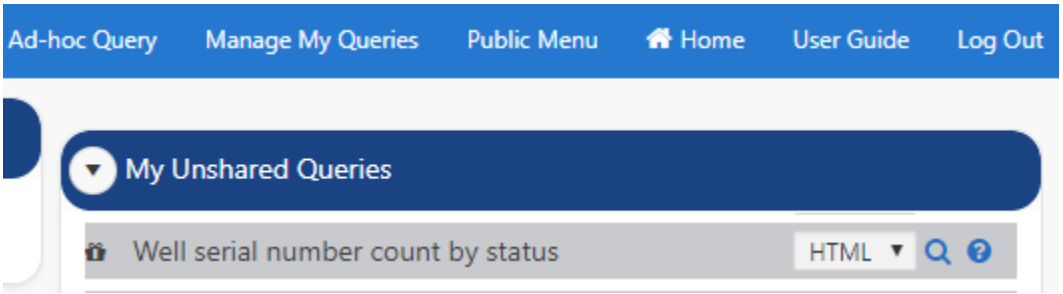
Once fields are completed, select *Save.*



Upon save, the window reverts back to the ad-hoc page with confirmation of the save:

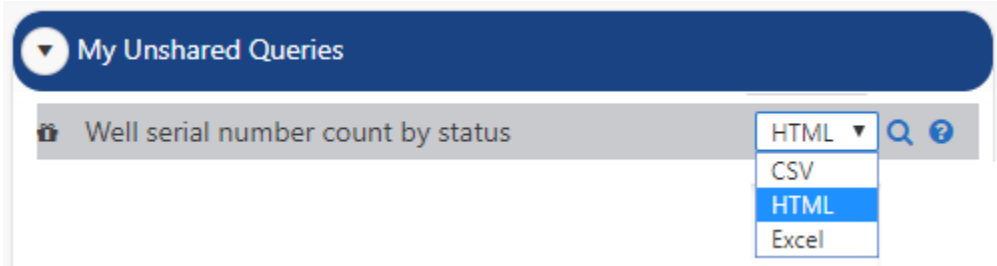


Now that the query is saved, it displays on my homepage under *My Unshared Queries.*

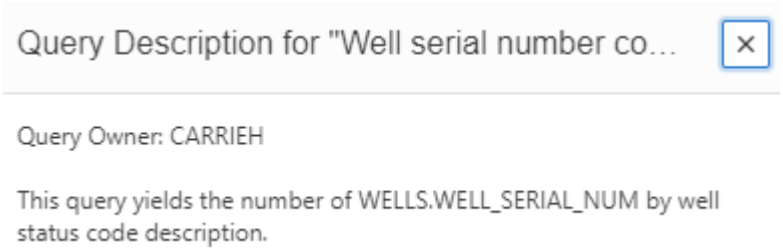


- Note – select the *User Guide* in the header to review the "Manage My Queries" section for detailed steps on sharing, copying, editing, and deleting saved queries.

From my homepage, I can execute the query as the default HTML or change to CSV or Excel by selecting the dropdown menu, then the magnifying glass to execute the query.



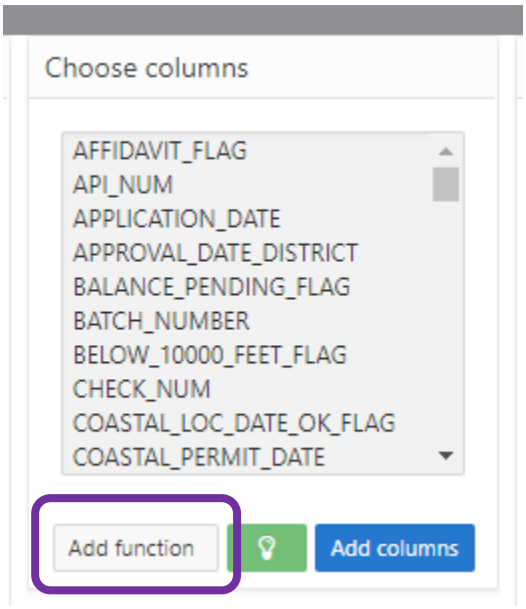
Selecting the question mark displays the description written on the save portion of the screen.



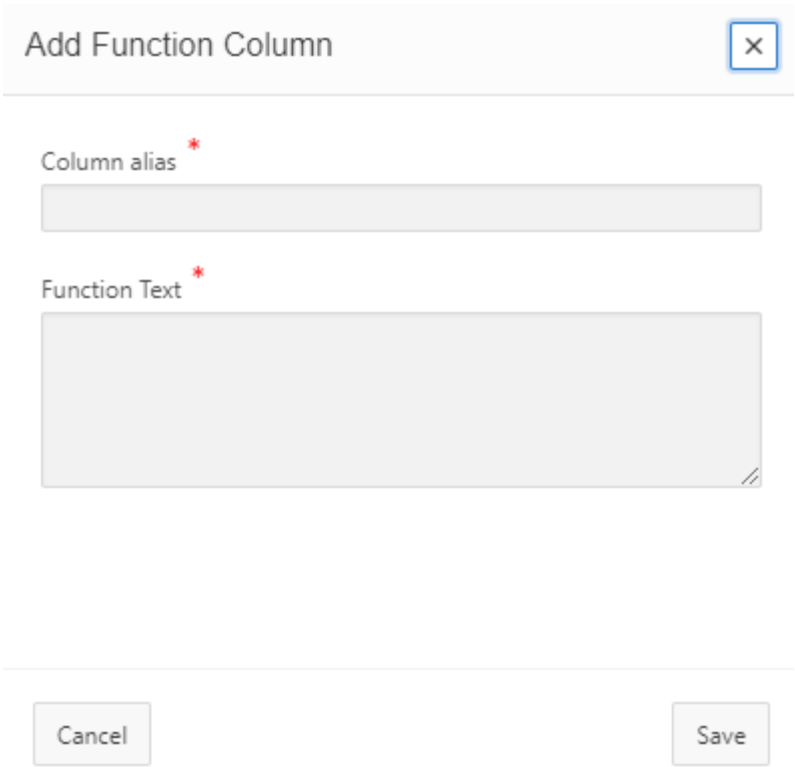
That ends the introduction to building simple queries in DODA. There are much more possibilities in DODA than discussed in the prior pages. The remainder of this document includes depictions of more complex use of DODA and will be updated based on advice given to users.

Columns, Aggregates, and Conditions

EXAMPLE 1: Use of CASE function to determine EXPIRATION_DATE. EXPIRATION_DATE is a calculation in SONRIS- APPROVAL_DATE + 179 if a six-month permit; APPROVAL_DATE + 364 if one-year permit. To get this calculation in DODA, we will use a CASE function and addition. I have columns selected in my desired table (DRILL_PERM_EVALS). In the *Choose columns* field, I will select *Add function*.



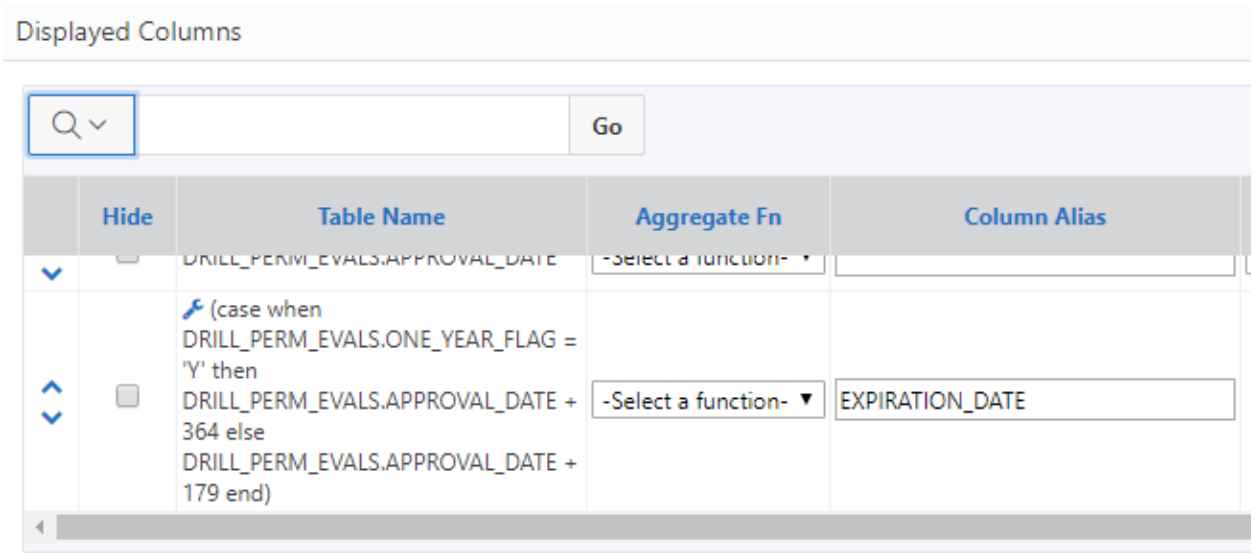
The following prompt displays:



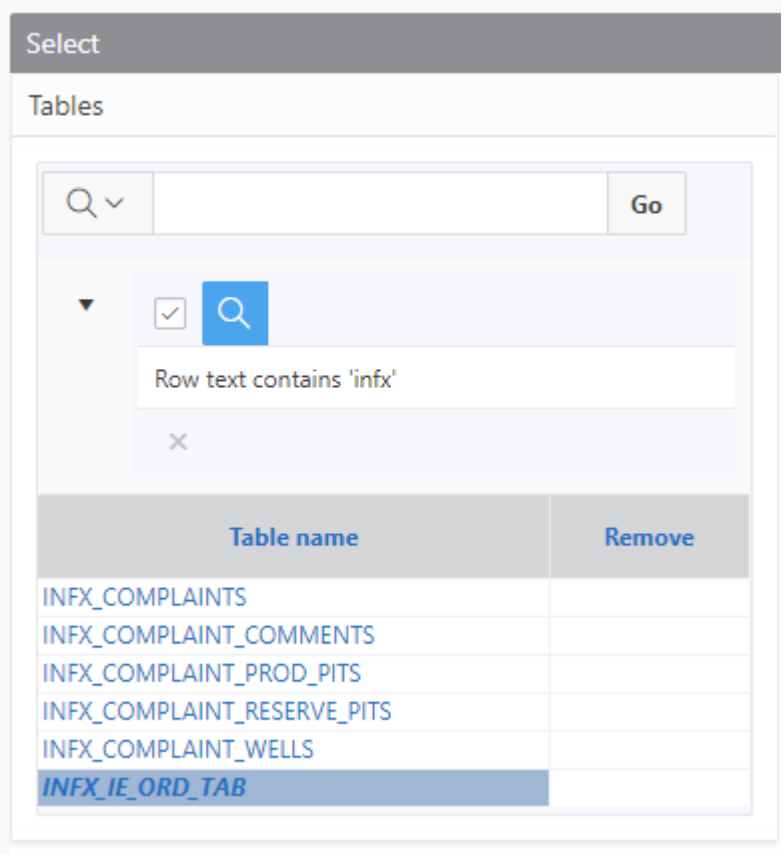
In the DRILL_PERM_EVALS table, there is a column named ONE_YEAR_FLAG. Values of Y reflect one-year permit; values of N reflect six-month permit. Our function will be "if one-year flag = Y, add 364 to approval date; otherwise, add 179 days." The SQL function will include CASE (if this, then that) and addition. Resulting function will be:

- Note – the *Column alias* is user defined. Whatever is keyed in here will be the column name in the results.

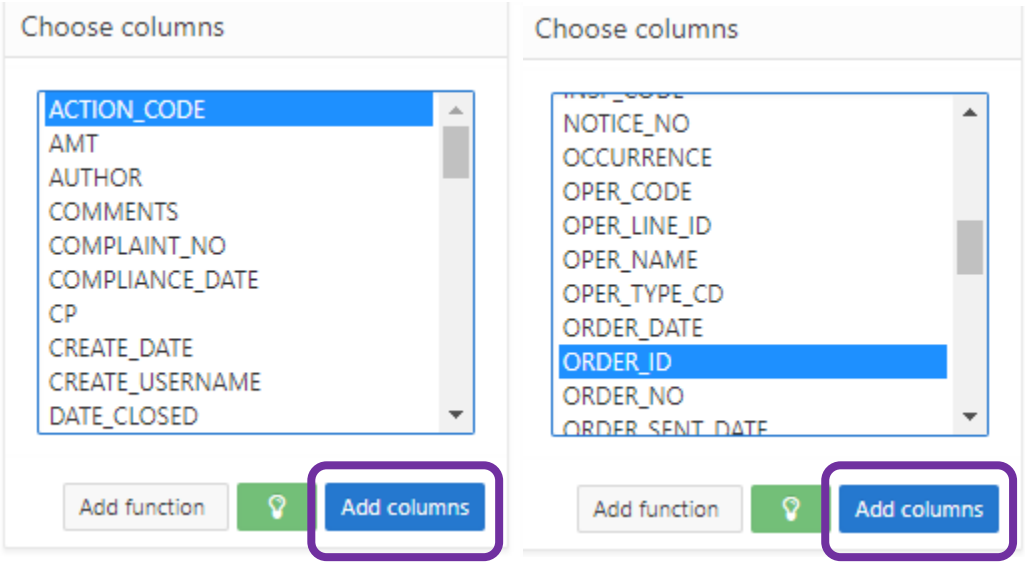
Select *Save*; the function now displays in our columns.



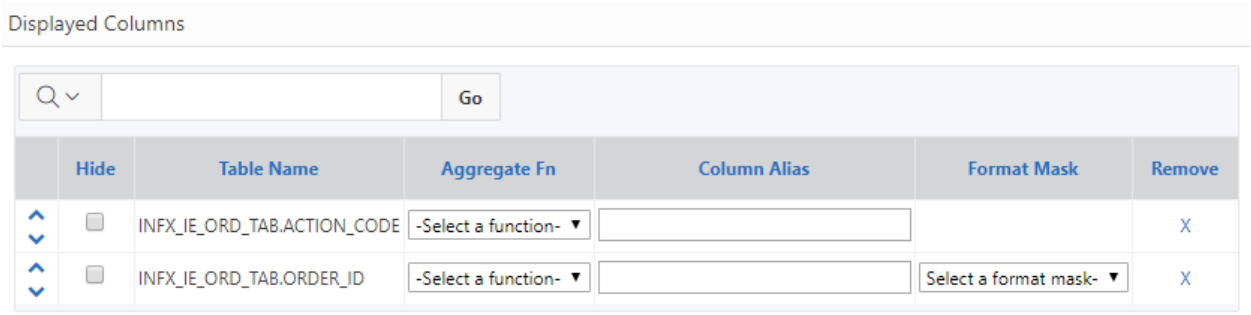
EXAMPLE 2: Goal- to determine the number of Compliance Orders by Action Code. This can be done by using the built-in Aggregate Functions in DODA. First, I select my table, INFX_IE_ORD_TAB.



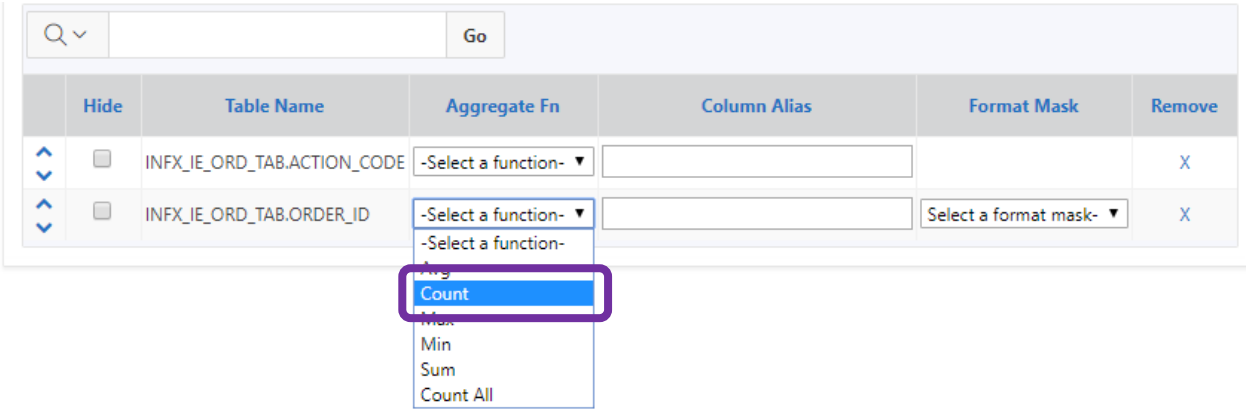
Next, I select the columns I'm interested in, ACTION_CODE to display the types of actions and ORDER_ID to develop my count.



My columns now display as follows:



I'm going to select the -*Select a function*- dropdown on the INFX_IE_ORD_TAB.ORDER_ID row to display available functions, and select COUNT



Scrolling down to the bottom of the page, I select *Execute*; results display the number of ORDER_IDs by ACTION_CODE

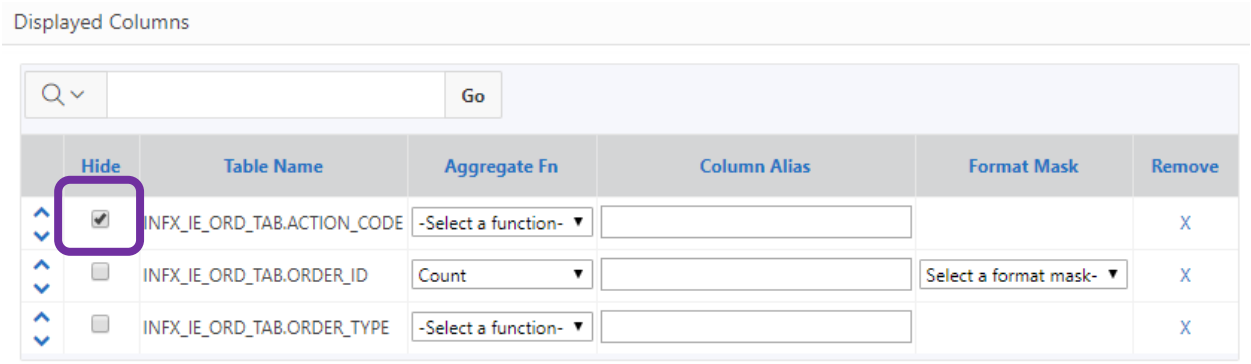
Query Results -

ACTION_CODE	ORDER_ID_COUNT
	1150
UIC-14	20
Survey	131
29B	9583
P&Apit	65
29D	140
UIC-9	1

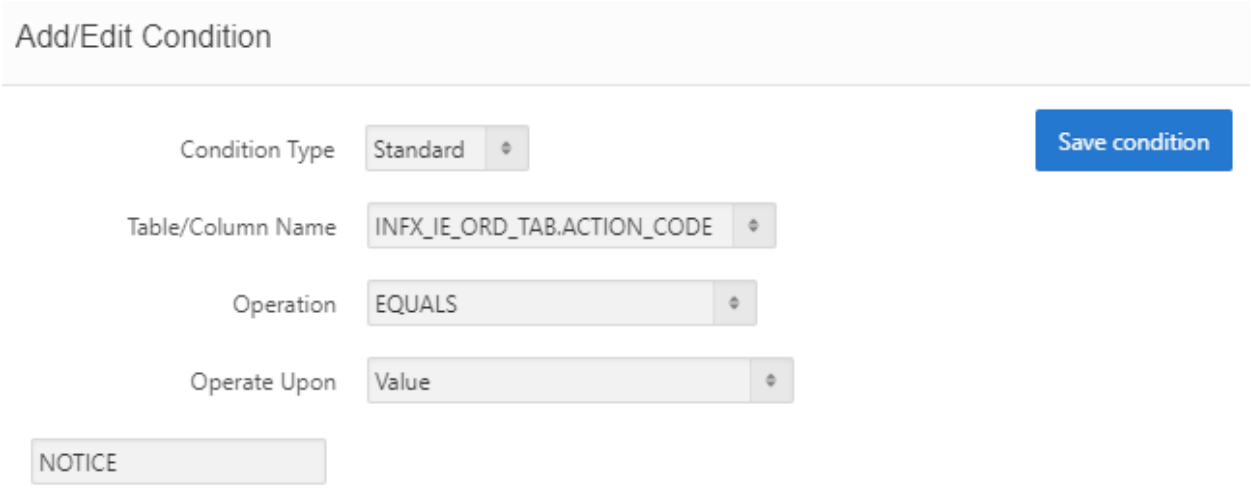
Results continued…

FORMS	308
FUT	58
NOTICE	7087
BOND	430
29Bpit	72
GW	53
29B-NI	5087

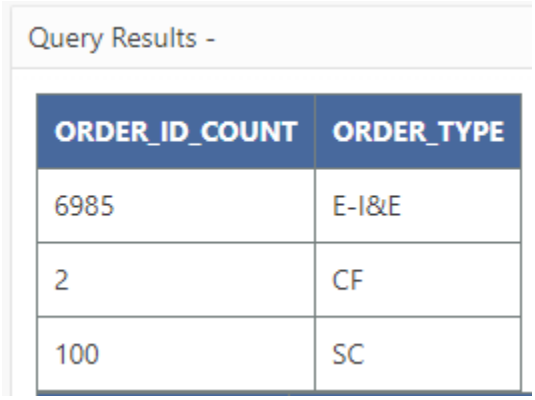
One of the ACTION_CODE results is NOTICE with a large count of Orders. To further review this result, we can add more columns and conditions. In my *Choose columns* field, I will add ORDER_TYPE. I will select *Hide* for the ACTION_CODE since it should not be displayed in the results but is required to set up the condition.



In the *Query Conditions* field, I will select *New Condition* and define my condition as ACTION_CODE = NOTICE



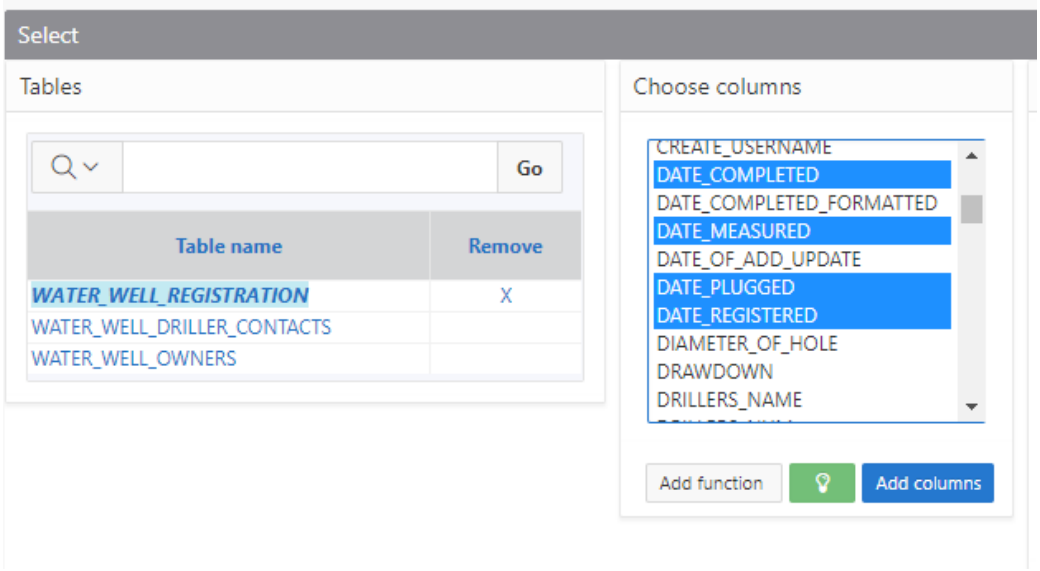
Select *Save Condition*, then scroll to the bottom of the screen to select *Execute*. The results now display the count of ORDER_IDs with ACTION_CODE = NOTICE by ORDER_TYPE.



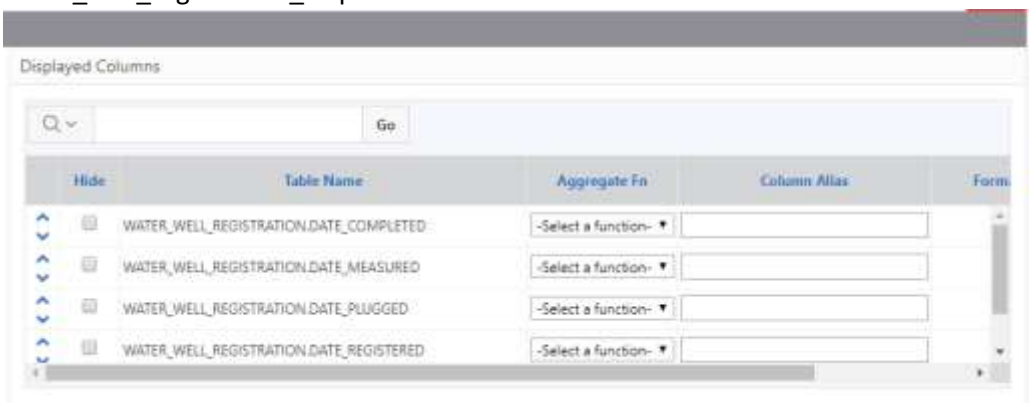
ORDER_ID_COUNT	ORDER_TYPE
6985	E-I&E
2	CF
100	SC

EXAMPLE 3: Goal- count of water well registrations

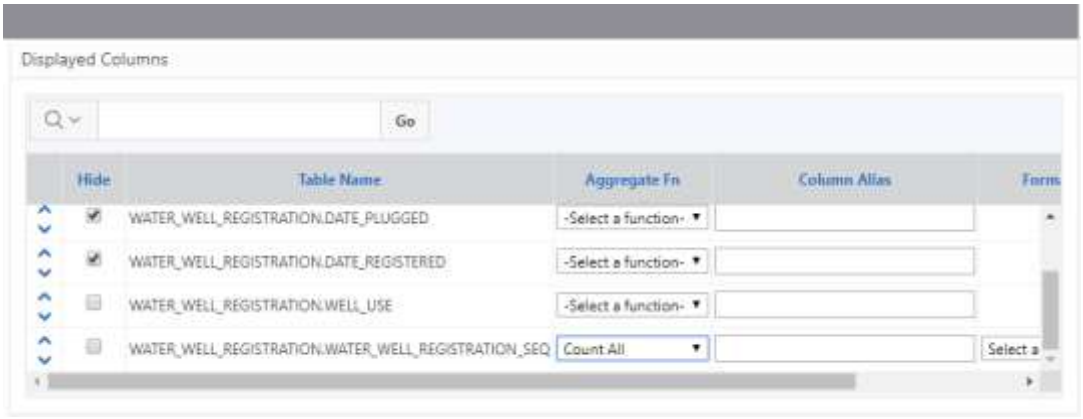
a. Select your table and choose some columns that we might want to use to conditionally include/exclude data rows from the returned dataset.



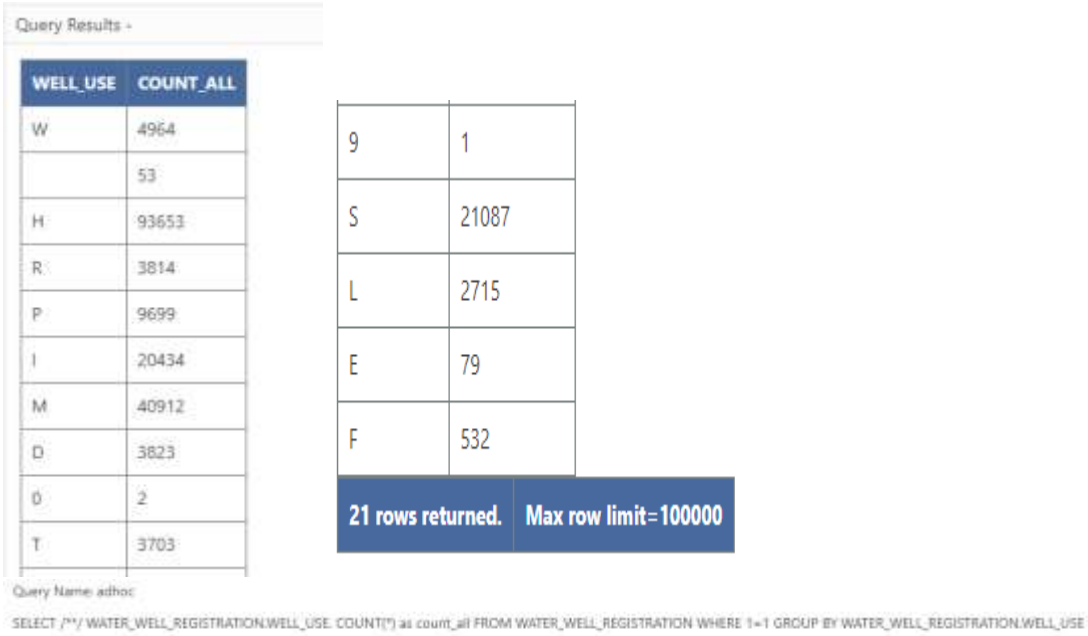
b. Click Add Columns to add the columns to the list of displayed columns. I added date_completed, date_measured, date_plugged, date_registered, well_use, and water_well_registration_seq.



c. As a first pass, turn off the display for date_completed, date_measured, date_plugged, date_registered by clicking the “Hide” column checkbox. Set the aggregate function Count All on the water_well_registration_seq and leave well_use alone.

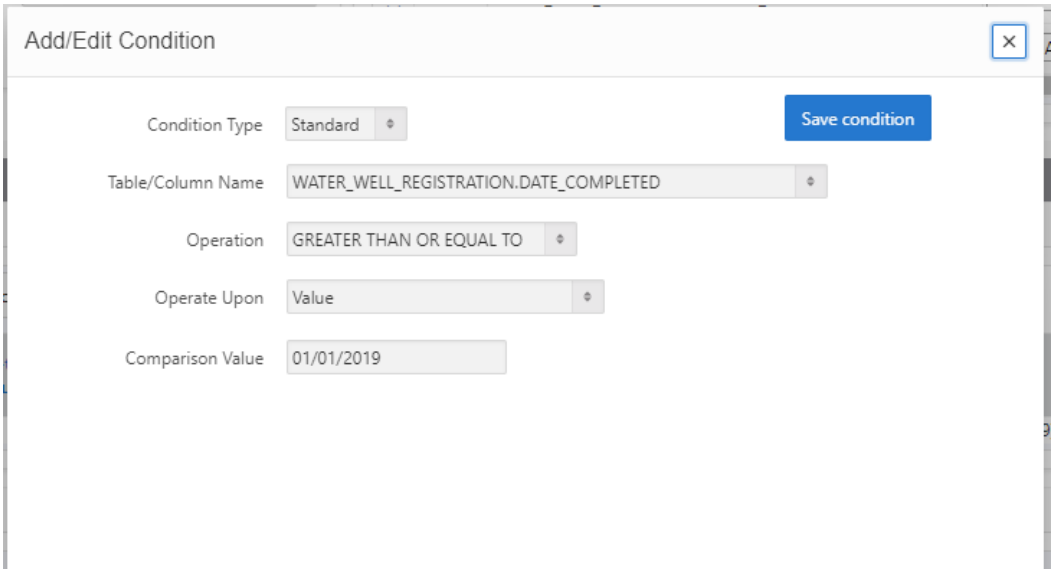


d. Click the Execute button at the bottom of the screen. This resulting query displays a count of registrations for each distinct value of well_use.



DODA will automatically group by the non-hidden non-aggregated columns.

e. We can expand on this by adding some conditions to the query. For example, let's say we want to see the breakdown of well use for wells completed after 01/01/2019. To do this we can simply add a condition. In order to use a column as part of a condition it must be included in the Displayed Columns section. If the field is not desired for display, simply use the "Hide" checkbox to tell DODA not to display the field (like I did for each of the date fields in our initial iteration of the query). Click the New Condition button in the Where region of the screen to open the Add/Edit Condition dialog. Adding the condition is a simple as selecting the column name, picking the type of comparison we want, and indicating a comparison value. Click Save Condition when done.



f. Click the Execute button at the bottom of the screen. This resulting query displays a count of registrations for each distinct value of well_use having a completion date >= 01/01/2019.

Query Results -

WELL_USE	COUNT_ALL
W	4601
	45
H	85886
R	3601
P	8533
I	18966
M	39033
D	3615
0	2
N	3596
Z	2118
O	1844
9	1
S	20350
L	2703
E	75
F	532

21 rows returned. **Max row limit=100000**

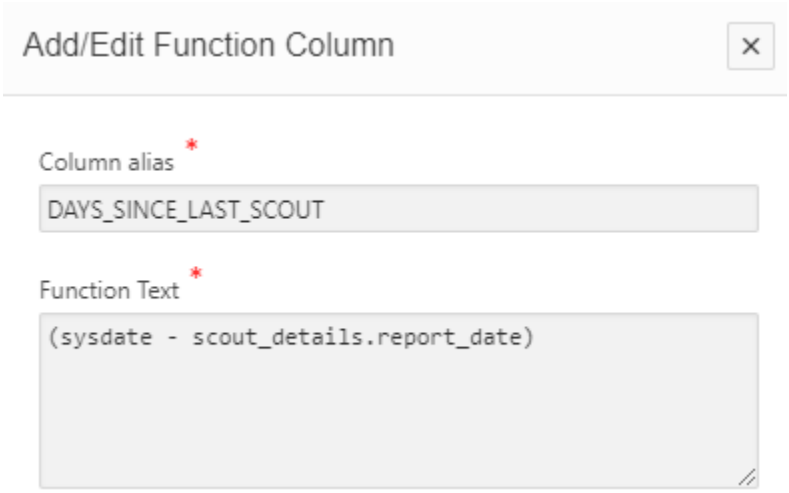
Query Name: adhoc

SELECT /**/ WATER_WELL_REGISTRATION.WELL_USE, COUNT(*) as count_all FROM WATER_WELL_REGISTRATION WHERE 1=1 AND ((WATER_WELL_REGISTRATION.DATE_COMPLETED >= '01/01/2019')) GROUP BY WATER_WELL_REGISTRATION.WELL_USE

Displayed Columns – Add function

Note- aggregates are not allowed in the *Displayed Columns – Add function* feature. For instance, the following equation IPS_INVOICES.INV_AMOUNT – SUM(IPS_PAYMENTS.PAYMENT_AMOUNT) yields error "ORA-00934: group function is not allowed here." The aggregate function SUM is prohibited. To resolve, users can request for the query to be made into a predefined query (which are not editable by DODA users) OR request a view that conducts the function.

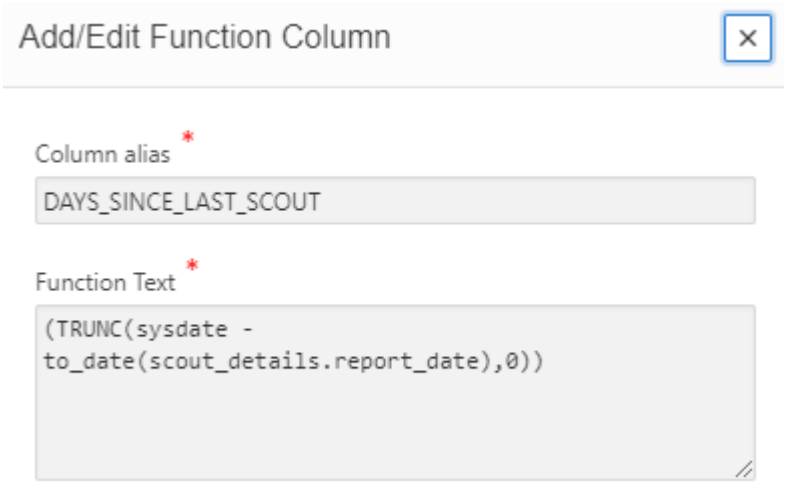
Displayed Columns – Add function can be utilized for simple arithmetic. Goal in this instance is to determine the number of days since last scout report.



Some date fields in the database are not truncated to days and results display decimals.

DAYS_SINCE_LAST_SCOUT	OC	SN	NUM	REPORT_DATE	SCOUT_STATUS
119.503159722222222222222222222222222222	A1760	252116	002-ALT	13-MAR-20	31
119.503159722222222222222222222222222222	A1760	252115	001-ALT	13-MAR-20	31
38.503159722222222222222222222222222222	G2380	252104	001	02-JUN-20	31
30.503159722222222222222222222222222222	E6963	252174	001	10-JUN-20	31
4.503159722222222222222222222222222222	60046	252163	003-ALT	06-JUL-20	07
4.503159722222222222222222222222222222	60046	252162	002-ALT	06-JUL-20	05

To resolve, use the truncate feature. The following formula truncates the date difference to whole days:

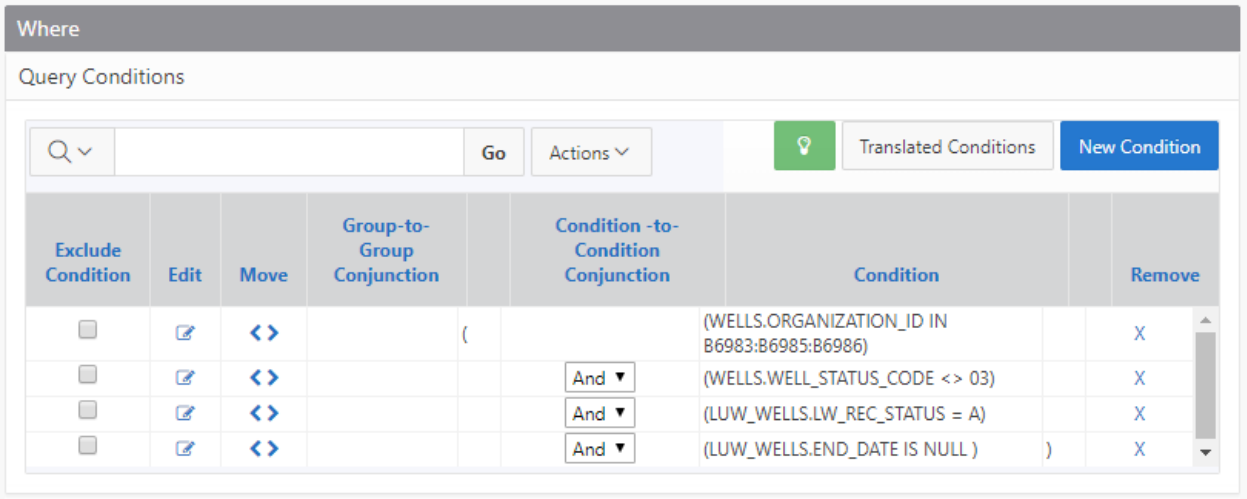


Results:

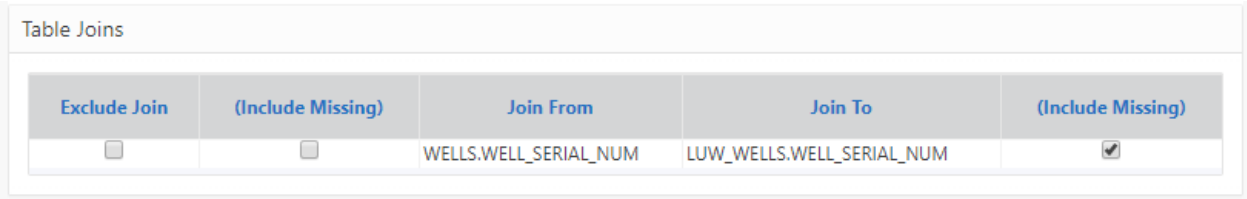
DAYS_SINCE_LAST_SCOUT	OC	SN	NUM	REPORT_DATE	SCOUT_STATUS
119	A1760	252116	002-ALT	13-MAR-20	31
119	A1760	252115	001-ALT	13-MAR-20	31
38	G2380	252104	001	02-JUN-20	31
30	E6963	252174	001	10-JUN-20	31
4	60046	252163	003-ALT	06-JUL-20	07
4	60046	252162	002-ALT	06-JUL-20	05

Conditions

Sample of multiple conditions in one query.

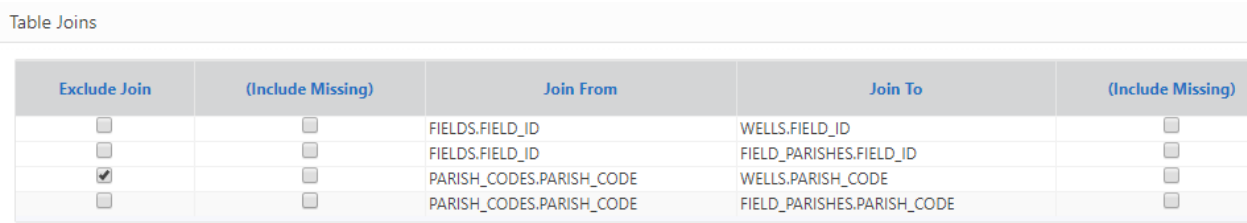


Joins



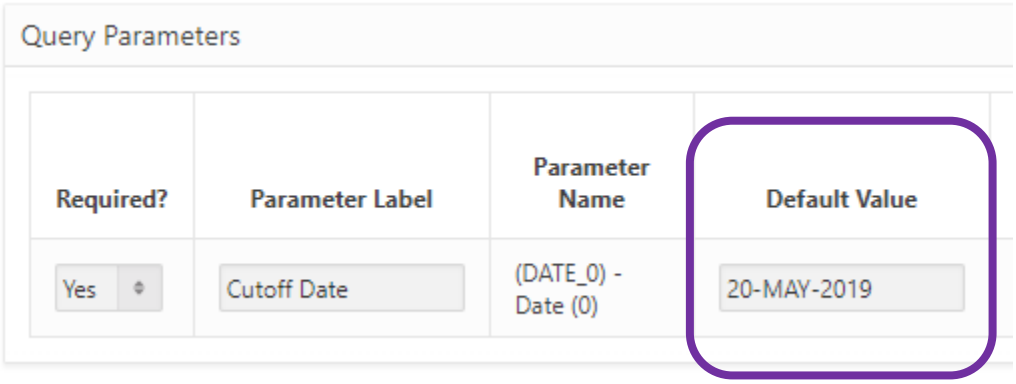
A join can be regular or an outer join. An outer join instructs the query to display results from that table if there are no results in that table. In the snippet above, the join is between WELLS and LUW_WELLS by WELL_SERIAL_NUM. All well serial numbers are in the WELLS table, but not all well serial numbers are in the LUW_WELLS table. Marking the *(Include Missing)* box on the LUW_WELLS table instructs the query to set an outer join to the LUW_WELLS table. So, well serial numbers with and without LUW_WELL records will display. If the box was unselected, this would be a regular join. Only well serial numbers that are in WELLS and LUW_WELLS will display.

Exclude Join is used when the same column displays in multiple tables as a join. See example below.



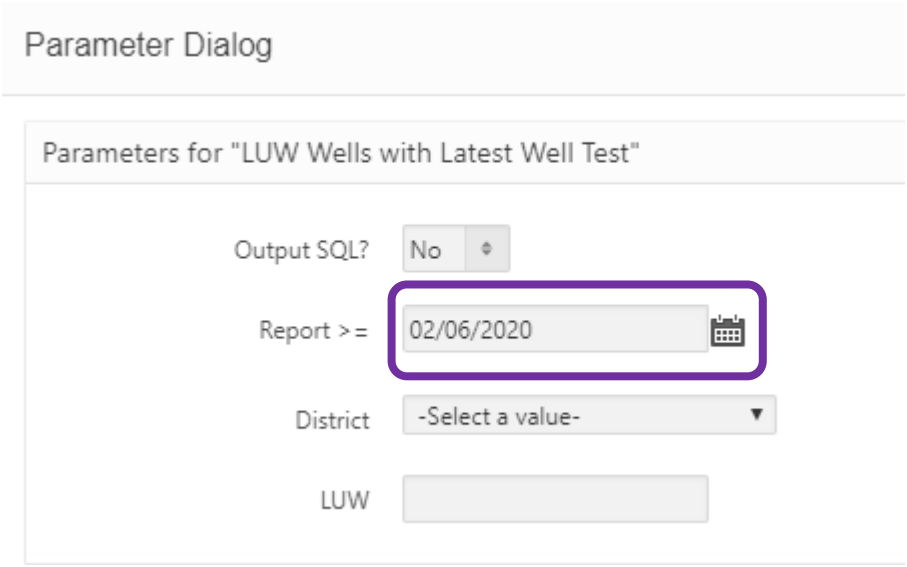
There is a join to PARISH_CODES by WELLS and by FIELD_PARISHES. The box checked directs DODA to ignore the join between PARISH_CODES and WELLS. The results displayed will include the results in the relationship between PARISH_CODES and FIELD_PARISHES.

Parameters

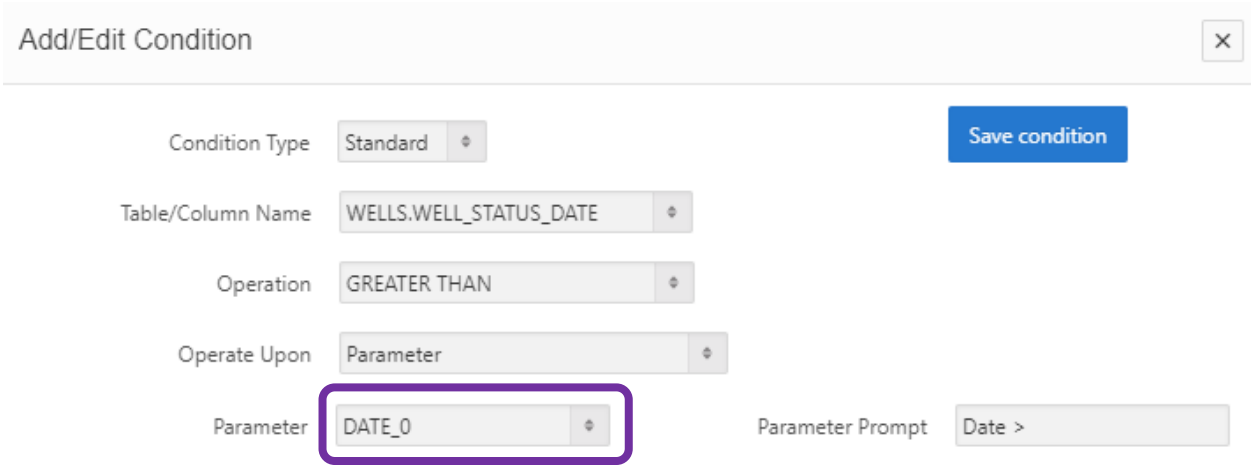


If a parameter is a date, on the save page, the snippet above displays the format required to set as a default value.

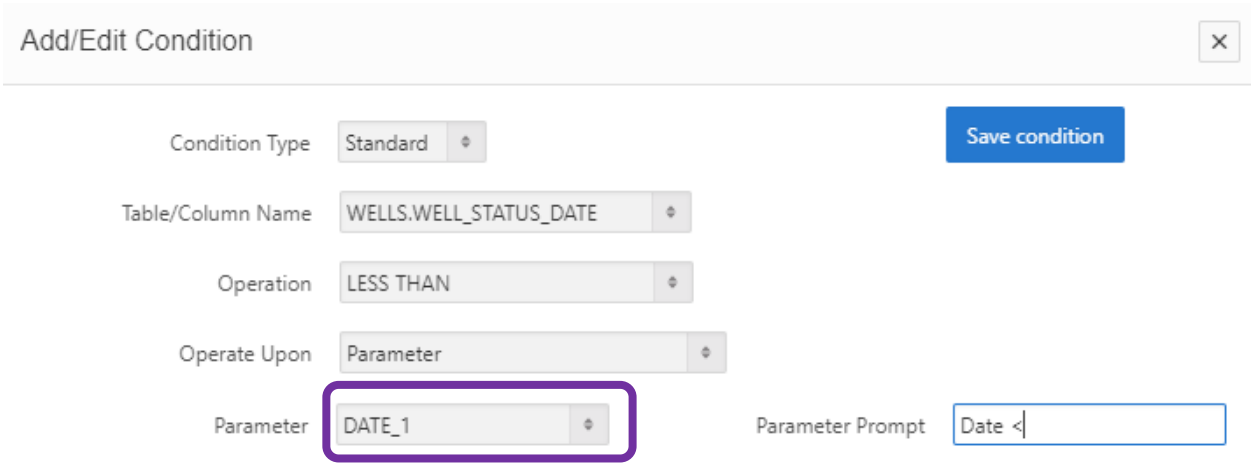
This is noted because on the parameter prompt pages, the date format is different:



If a query includes multiple parameters of the same type (ie, multiple character fields or multiple date fields), each parameter must have its own unique identifier. For instance, setting up parameters for dates- greater than and less than prompts. The first date will be DATE_0…



…and the second date will be DATE_1.



The difference of the two is noted on the save page.