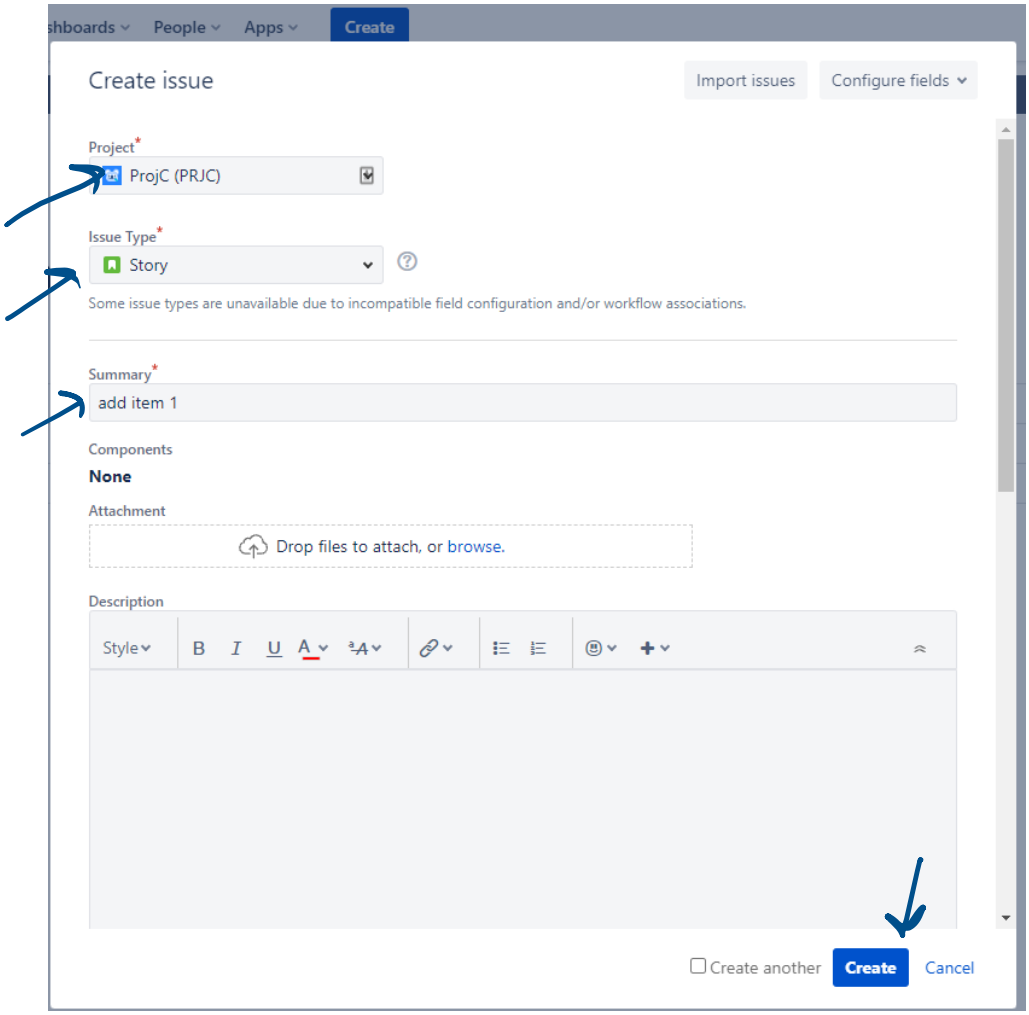
ECSE365L – Agile Software Development
Lab 08 – Quick Search, Basic Search, and JQL

NOTE: Add the faculty as your team member in your project

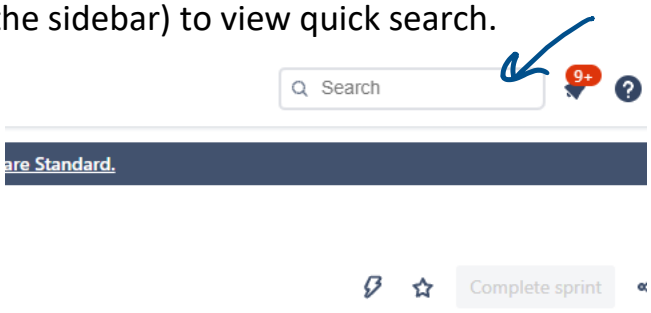
It is always better to search for some content from a huge pile of information. We have witnessed the advantages of search as well as different types of searches. On similar grounds, JIRA also has various types (quick searches, basic searches) of searches built-in. We will also experience how to work with the search results.

- Create a new company-managed SCRUM project '**ProjC**'.
- Create three issues (add item 1, add item 2, and add item 3) of issue type **Story** (check that while creating).



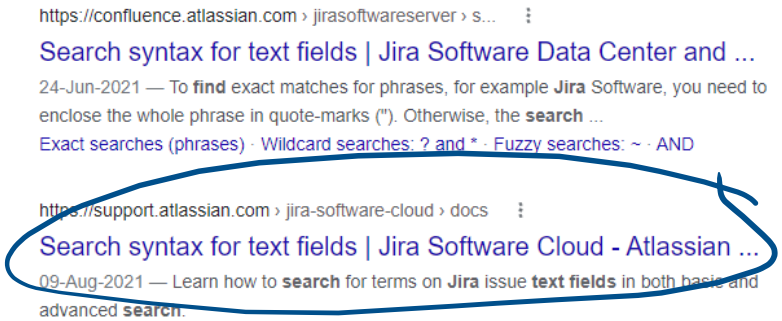
Perform quick searches

1. Click in the search box in the upper right (or if you see it, click on the search icon in the sidebar) to view quick search.



2. Search for "**WI**". As you type, the search results will change. This searches issues with fields of types text, board names, filter names, and project names. Press Enter. You will be taken to the Filters section with the associated text-based search of issues.
3. Use quick search to search for "**item 2**" → Search for "item 2" and verify that search terms are not case-sensitive.
4. Search for "**item AND 2**". The results should be the same as the previous search. The terms of a query are joined with AND by default.
5. Search for "**item NOT 1**". The NOT keyword should exclude the "**add item 1**" issues.
6. Search for "**item not 1**". This should return the "**add item 1**" issues. This is because "**not**" is in lowercase, and it is such a common word that it is excluded from the search (a reserved or stop word).
    - This is the same as searching for "item 1".

7. In another browser window or tab, perform a general web search for "**Jira search syntax for text fields**" → click on the Atlassian documentation.



8. Scroll down to the "**Reserved words**" heading and verify that "**and**" and "**not**" are reserved words for searches of text fields.

Reserved words

To keep the search index size and search performance optimal in Jira, the following English *reserved words* (also known as '*stop words*') are ignored from the search index and hence, Jira's text search features:

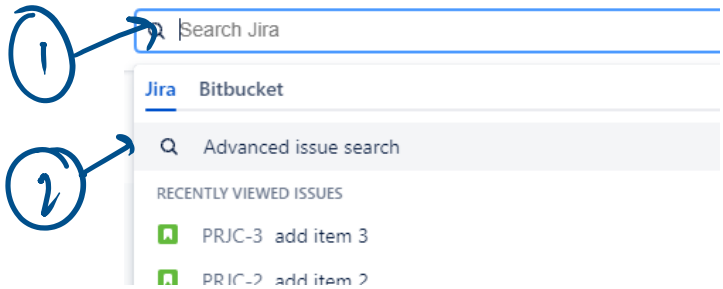
"a", "an", "abort", "access", "add", "after", "alias", "all", "alter", "and", "any", "are", "as", "asc", "at", "audit", "avg", "be", "before", "begin", "between", "boolean", "break", "but", "by", "byte", "catch", "cf", "char", "character", "check", "checkpoint", "collate", "collation", "column", "commit", "connect", "continue", "count", "create", "current", "date", "decimal", "declare", "decrement", "default", "defaults", "define", "delete", "delimiter", "desc", "difference", "distinct", "divide", "do", "double", "drop", "else", "empty", "encoding", "end", "equals", "escape", "exclusive", "exec", "execute", "exists", "explain", "false", "fetch", "file", "field", "first", "float", "for", "from", "function", "go", "goto", "grant", "greater", "group", "having", "identified", "if", "immediate", "in", "increment", "index", "initial", "inner", "inout", "input", "insert", "int", "integer", "intersect", "intersection", "into", "is", "isempty", "isnull", "it", "join", "last", "left", "less", "like", "limit", "lock", "long", "max", "min", "minus", "mode", "modify", "modulo", "more", "multiply", "next", "no", "noaudit", "not", "notin", "nowait", "null", "number", "object", "of", "on", "option", "or", "order", "outer", "output", "power", "previous", "prior", "privileges", "public", "raise", "raw", "remainder", "rename", "resource", "return", "returns", "revoke", "right", "row", "rowid", "rownum", "rows", "select", "session", "set", "share", "size", "sqrt", "start", "strict", "string", "subtract", "such", "sum", "synonym", "table", "that", "the", "their", "then", "there", "these", "they", "this", "to", "trans", "transaction", "trigger", "true", "uid", "union", "unique", "update", "user", "validate", "values", "view", "was", "when", "whenever", "where", "while", "will", "with"

Be aware that this can sometimes lead to unexpected results. For example, suppose one issue contains the text phrase "VSX will crash" and another issue contains the phrase "VSX will not crash". A text search for "VSX will crash" will return both of these issues. This is because the words `will` and `not` are part of the reserved words list.

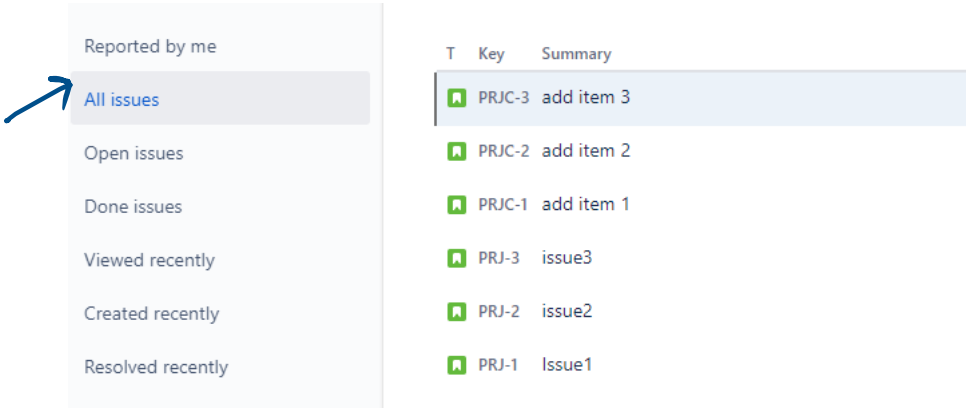
Perform basic searches

1. Click the search box in the upper right and select **Advanced issue search**.

   This takes you to the Filters section of your site



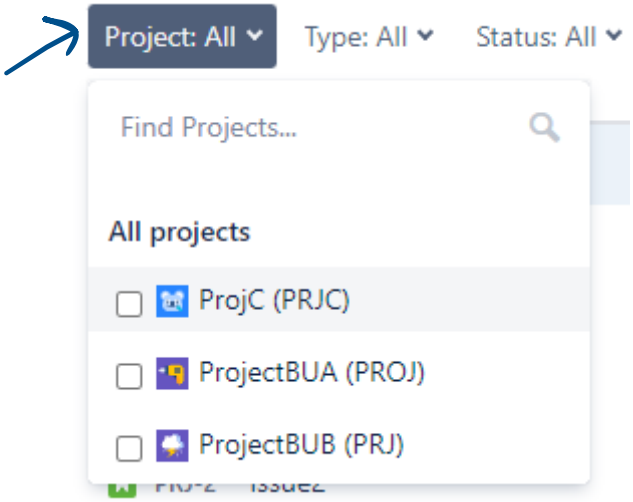
2. Click on the **All issues** tab on the left. You should be viewing all of the issues of the projects.



3. Check that you are in the basic search.

   - You should see a row of interface elements under **All issues** and a **Switch to JQL** link to the right.
   - If you see a **Basic link**, click on it to change from JQL to basic search.

4. Click on the **Project** dropdown in the row of interface elements to view the issues of any one of your projects.

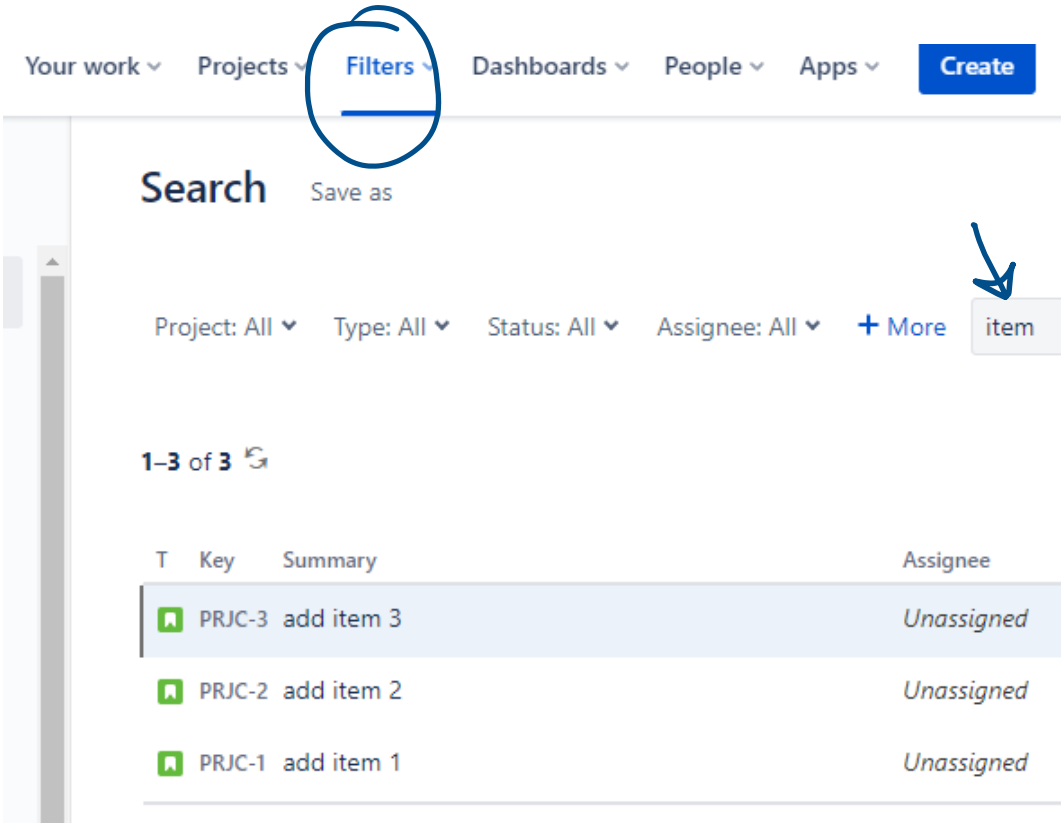


5. Use the "**Contains text**" box in the basic search row to further limit your results.

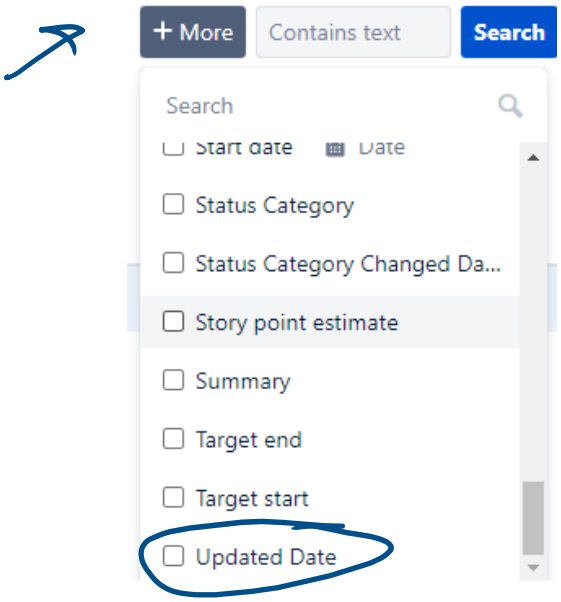
   - Press Enter or click on the search hourglass to perform the search. Verify that the NOT keyword works in the basic search.

6. Use quick search (like you did earlier in the lab) to type in "**item**" and click Enter.

   - You should be brought to the *Filters area with a basic search. Verify that the text that you entered is in the textbox.

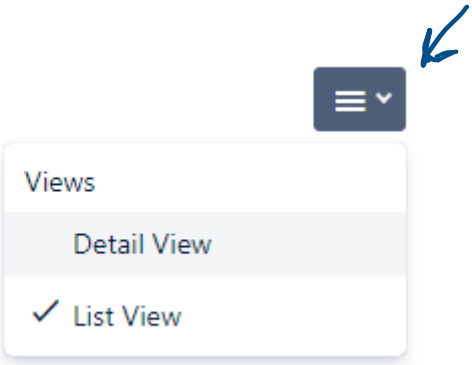


7. Clear the existing search by clicking **Search issues** or **All issues** in the sidebar on the left.
8. In basic search, click on the **More** dropdown and search for issues that have been updated (**Updated Date** field) in the last hour, day and week. Your results depend on when you performed your previous labs.

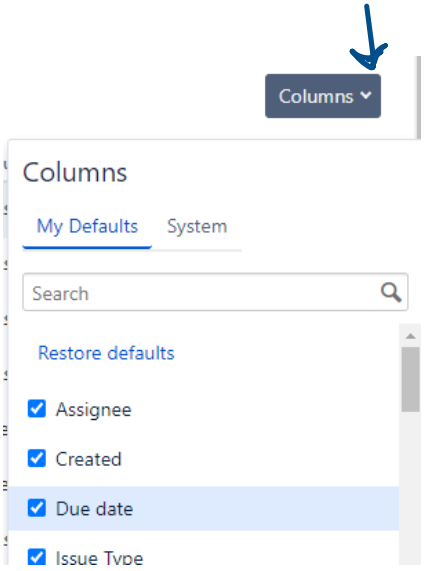


Work with search results

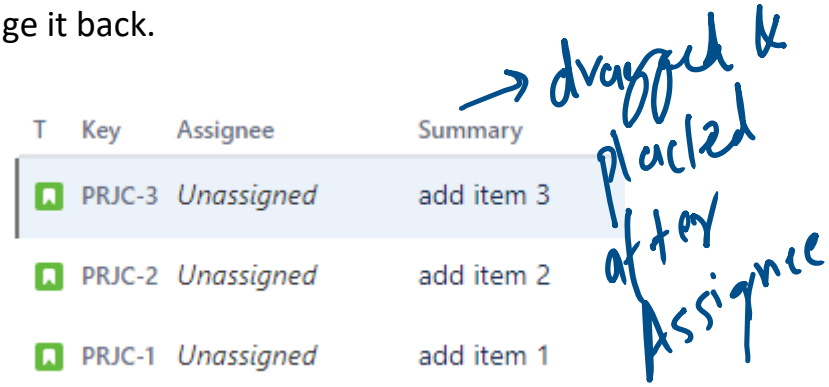
1. Toggle between **List View** and **Detail View** using the Change View icon to the right of the basic search elements.



2. In List View → click on the **Columns** dropdown to change the columns that are displayed in the results → Click **Restore defaults** to undo what you have changed.

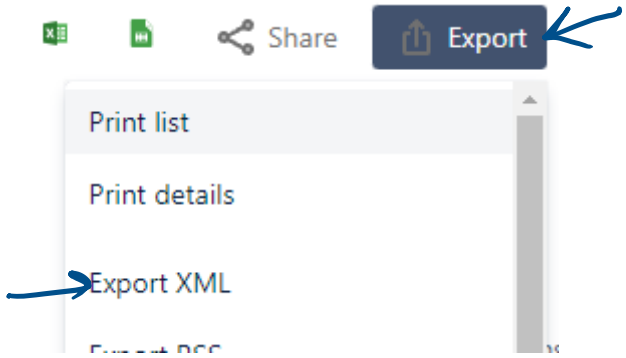


3. Reorder the first two columns by dragging and dropping the column header → Change it back.

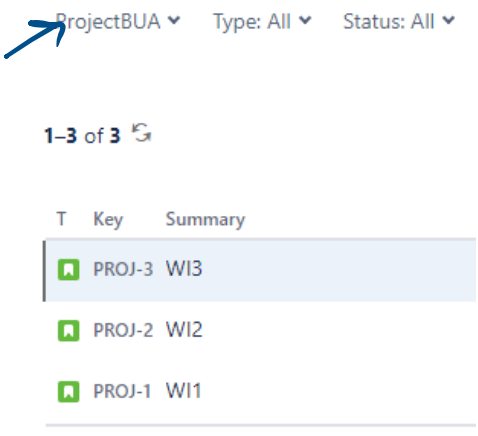


4. Click on a column header to sort by that column → click on the column header again to reverse the sorting.
5. In the basic search "**Contains text**" box, enter "**item 2**" and hit Enter. Using the Share icon in the upper right, email yourself a copy of the search results. You should receive an email with a link to the underlying JQL query.

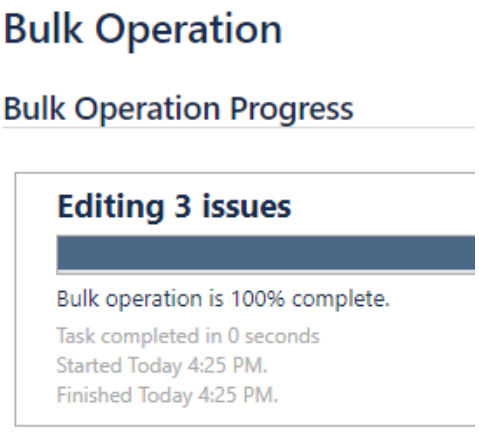
   Click on the link in the email and you should see your search in a new browser window. Close the new browser window.
6. In the original browser window, click on the **Export** icon in the upper right. Select **Export XML**. You should see the XML search results.



7. View the issues' field names and values under an **item**. If the XML is displayed in your browser window, click the browser's back button to navigate back to Jira.
8. Change the Assignee of all the issues of the project:
   a. Search for all issues of your ***Lab5*** project.



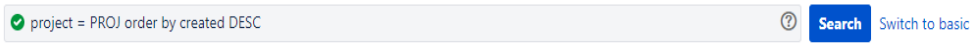
b. Click on the More icon (the three dots) in the upper right and select **Bulk change all X issue(s).**
c. First, select the Unassigned issues. (If they are all assigned, you can change this exercise to unassigning them all.)
d. Second, select Edit Issues.
e. Third, click Change Assignee and click Assign to me.
f. Fourth, click Confirm.



g. Verify that your bulk changes were made.

Create a basic search and view the JQL query

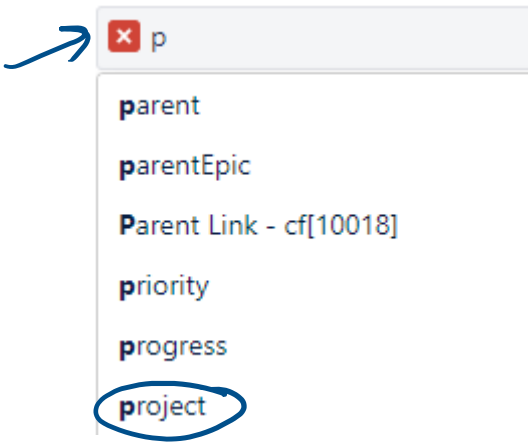
1. Open basic search by clicking on Filters.
   - If needed, click on the Switch to basic link to view the basic search.
2. In basic search → search for all issues of ***Lab05*** project.
3. Click on the *Switch to JQL link* to enter advanced/JQL search.



   - View the JQL query associated with the basic search. You can use this technique of switching from basic to advanced/JQL search to help "write" JQL queries.
4. Click on column headers in the list view to sort the results. Notice the changes to the query.

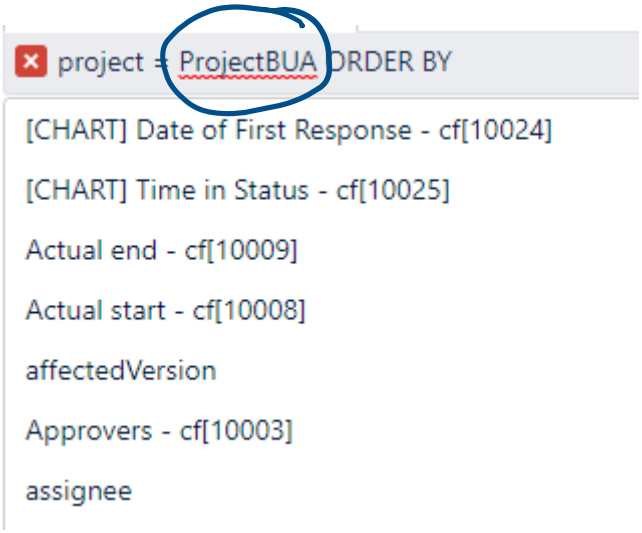
Create JQL queries with the help of autocomplete and column sorting

1. Enter advanced/JQL search (if necessary).
2. Clear the current JQL query.
3. Create and execute a query that finds all issues in the ***lab05*** project:
   a. With the JQL textbox selected, press p and select project from the autocomplete dropdown.



b. Press the space bar to view operator autocomplete.
c. Select the equals (=) operator.
d. Press the space bar to view value autocomplete.
e. Select ***Lab05*** project.
f. Press Enter to execute the query.

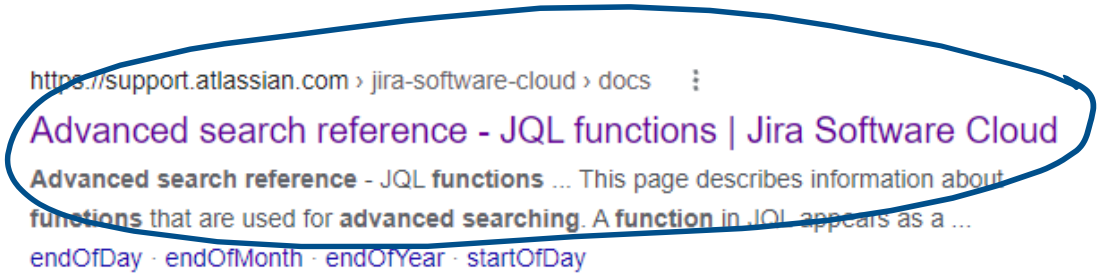
4. Add an **ORDER BY** clause to the query by clicking on the Summary heading in the list view.



Use functions as values

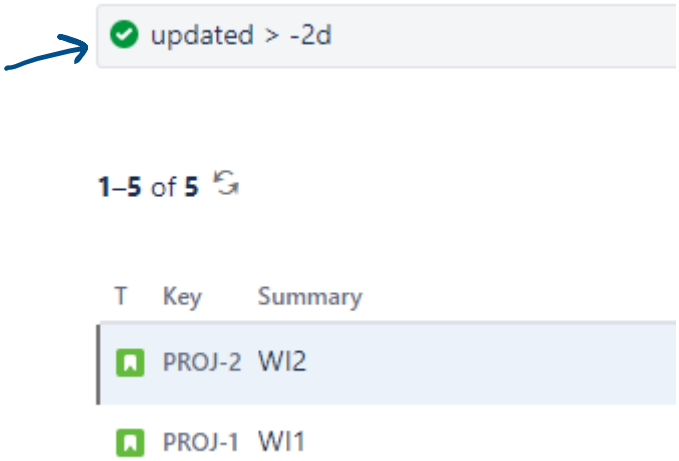
1. In advanced/JQL search, use autocomplete to find all issues assigned to you using the currentUser() function. ***assignee = currentUser()***
2. Find all issues that were created since the ***startOfWeek()***. You will use the > operator.

   - In a separate browser window, perform a web search for ***Jira advanced searching functions reference***. → Click on the Atlassian documentation link → View the available advanced/JQL searching functions.



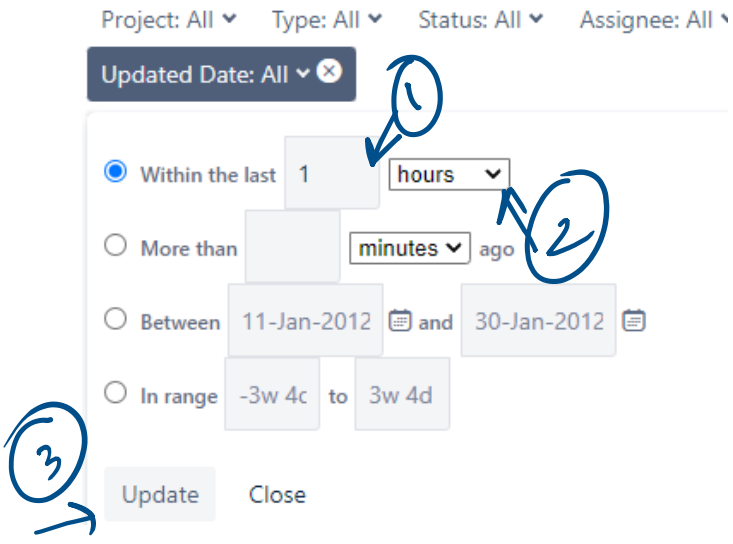
Use time unit qualifiers

1. In advanced/JQL search, enter the query ***updated > -2d*** to find issues that were updated in the past 48 hours.



2. Modify the previous query to find issues updated in the past two hours.
3. Using basic search (you may need to clear the existing advanced/JQL search and press ***Enter*** to enable the basic search link).

- Find all issues that were updated in the past hour.

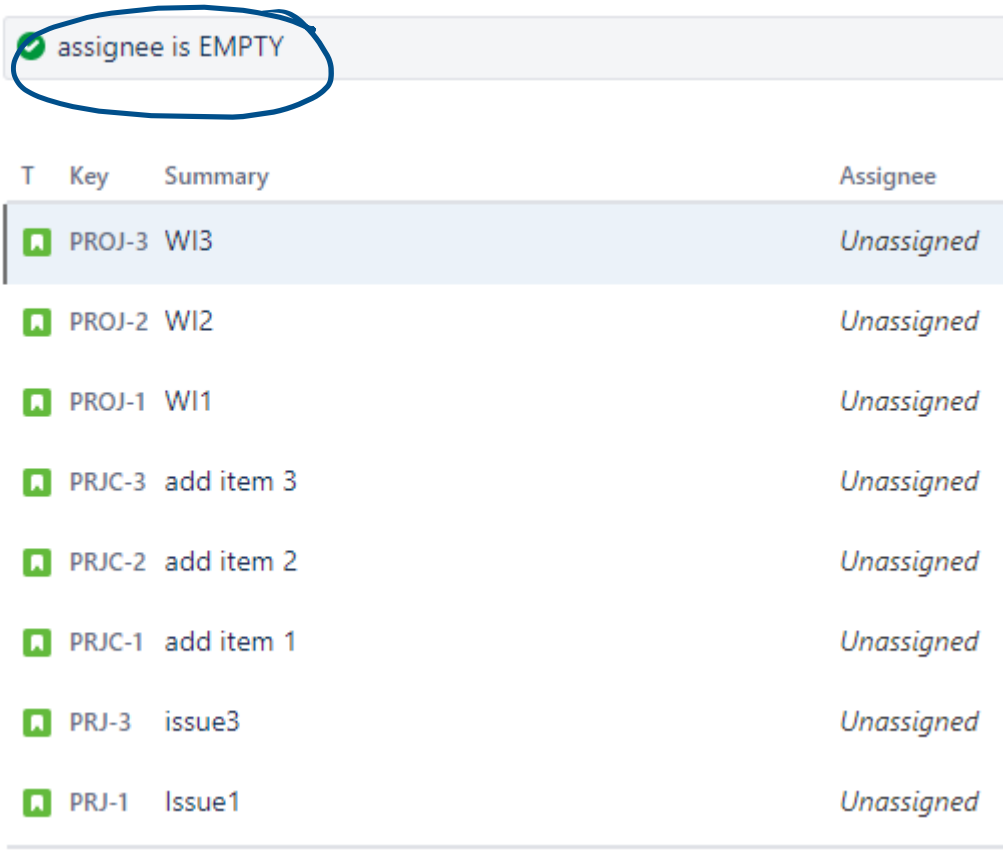


- Switch to advanced/JQL search and notice that a time unit qualifier is used in the query. This is a helpful way to write queries with time unit qualifiers.

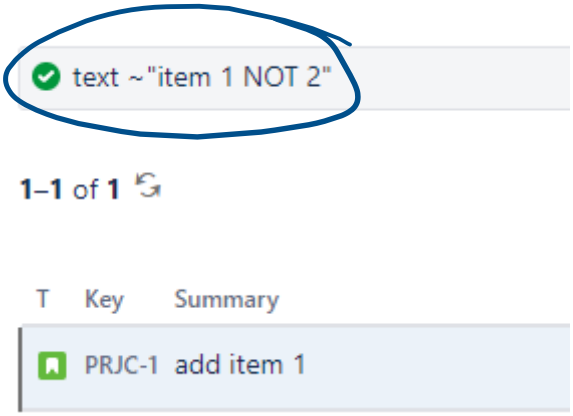
4. In advanced/JQL search, find all issues that were updated yesterday or today. (Hint: use the ***startOfDay()*** function with an argument of a time unit qualifier of ***-1d*** .)

Use various operators

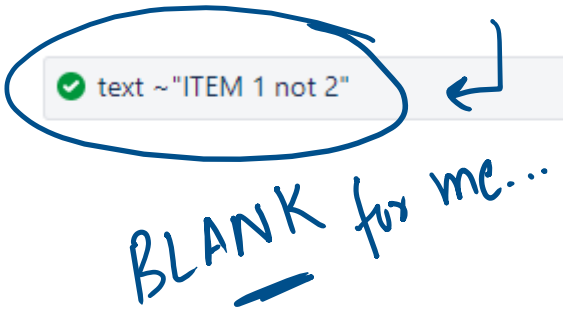
1. In advanced/JQL search, enter ***assignee*** and press the space bar to view the available operators.
2. Select the ***is*** operator and press the space bar. Notice that the only valid value is ***EMPTY***. Select it.
3. Execute the query to find all unassigned issues.



4. Execute the query ***text ~ "item 1 NOT 2"*** to find issues with text fields that contain ***item*** and ***1*** but not ***2***.

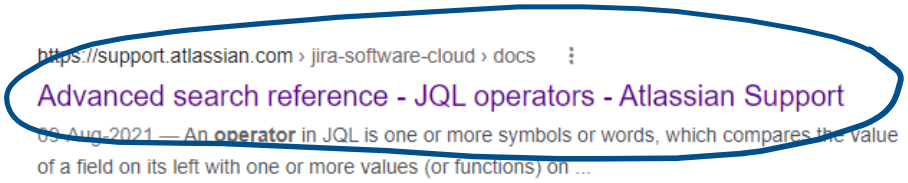


5. Modify the previous query to capitalize ***ITEM*** and verify that text strings are not case-sensitive.
6. Modify the previous query to change ***not*** to lowercase and verify that the query results are different.



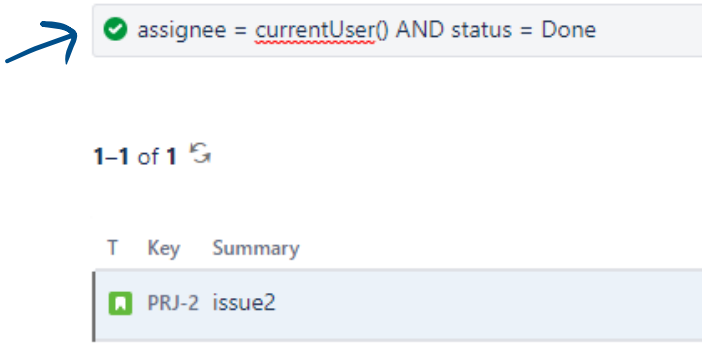
   - The NOT keyword in text field searches is case-sensitive. This query is the same as ***item 1 2*** and probably returns no issues because no issues have a 1 and a 2.

7. In a separate browser window, perform a web search for ***Jira advanced searching operators reference*** → Select the Atlassian documentation → Explore the reference.



Use Boolean operators

1. In advanced/JQL search, use a Boolean operator to find issues with an ***assignee*** of ***currentUser()*** and ***status*** of ***Done***.



2. Use the ***NOT*** Boolean operator to find issues that do not have a ***status*** of ***Backlog***. Verify that this query is equivalent to ***status != Backlog***.
3. Use the ***OR*** operator to find issues with a ***status*** of ***Selected for Development*** or ***In Progress***.
4. Create a query that is equivalent to the previous query using the ***in*** operator. (See answer at the end of the lab.)
5. Create a query containing multiple Boolean operators that returns different results depending on if you use parentheses in the query. Examples:

   - NOT (project = projectBUA OR project = projectBUB)
   - (status = Done OR status = "To Do") AND summary ~ "item 1"

Answer of #4 ***status in ("Selected for Development", "In Progress")***

Submit an MS Word file that contains your final board from Lab05 (mine was ProjecjBUA) as your assignment on LMS.

Ref: Altassian