JGI-KBase search api documentation (Updated)

Note : New updates are highlighted in red

API Methods

Base URL = https://jgi-kbase.nersc.gov/

Methods	URL	Arguments
POST	/query	-d '{ "query" : <query string> [**"userid": <KBase userid>**, "size" : <default = 10> , "page" : <page number default = 0>] }'
POST	/fetch	-d '{ "ids":<list or comma delimit string of ids of docs to be retrieved>, "path" : <abs. path to destination dir on dtn.chicago.kbase.us>, [**"userid": <KBase userid>**] }'
GET	/status	id=<job_id returned by fetch call>
POST	/summary	TBA

Authentication

API is accesible only via the shared authetication token in the "Authorization" request header

```
curl -H "Content-Type: application/json" -X POST —user 'KBase:TOKEN'  https://jgi-kbase.nersc.gov/
```

Success Response

1) **Query** :

- Query accepts json input. These are fields it accepts :
  - "query" (Required) : Explained below.
  - "filter"   (Optional)   : Explained below.
  - "userid" (Optional) : KBase userid. If there is a valid corresponding JGI identity, then user will be able to search or fetch his private data. Else, only public data is provided.
  - "size"   (Optional)   : # of search hits returned. Default = 10.

  - “page” (Optional)   : Page number of the search results. Default = 0.
- **query input : It can either be a str or dict**
  - If it is a string - Query performs a full text search on the _all ES index (which is a “catch-all” text index of the entire document). Multiple words in the string can be combined with boolean logic (explained below). Default operator is OR
  - ***For logical operations, query string supports standard ES “Simple Query String" syntax :*** (https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-simple-query-string-query.html

    - `+` signifies AND operation
    - `|` signifies OR operation
    - `-` negates a single token
    - `"` wraps a number of tokens to signify a phrase for searching
    - `*` at the end of a term signifies a prefix query
    - `(` and `)` signify precedence
    - `~N` after a word signifies edit distance (fuzziness)
    - `~N` after a phrase signifies slop amount

In order to search for any of these special characters, they will need to be escaped with \

Default operator is OR

  - Query accepts dict with field-value pairs : Each value string follows same boolean logic syntax as above.
  - Use “operator” key in the dict to mention boolean logic between field value pairs. It accepts “AND”, “OR” , “NOT”. Default operator is “OR”.

Examples :

All text search -

```
-X POST https://jgi-kbase.nersc.gov/query -d ‘ { “query” : “coli” } ’  | jq .total
710
-X POST  https://jgi-kbase.nersc.gov/query -d ‘ { “query” : ”coli + fasta” } ' | jq .total
25      ## Searches for Coli AND fasta
-X POST https://jgi-kbase.nersc.gov/query -d ‘ { “query” : "coli fasta” } ' | jq .total
35279  ## Since default operator is OR
-X POST https://jgi-kbase.nersc.gov/query -d ‘ { “query” : "coli + cox@biochem.wisc.edu” } ' | jq .total
541
-X POST https://jgi-kbase.nersc.gov/query -d ‘ { “query” : "fasta + coli + cox@biochem.wisc.edu” } ' | jq .total
17
```

Field-value pairs -

```
-d  ‘ { “query” : { “file_name” : "CAGATC*” , “file_type” : “fastq” , “operator” : “AND” } } ’
```

```
-d  ‘ { “query” : { “file_name” : "CAGATC*” , “file_type” : ”fastq” } } '
```

- **Filter input : Must be a dict**
    - Query results can be further filtered on these fields : *proposal_id, project_id, project_title, pi_name, project_date*.
    - Filter values must be exact match to the value of the field in the indices.
    - Filter are cached by elasticsearch and filters don’t recalculate search scores. So they are more efficient.
    - Filter field-value pairs are tied by “OR” operator by default.
    - Use “operator” key to change boolean to “AND”
    - project_id, proposal_id are numbers. To filter on multiple values in that field (equal is mysql IN), provide comma delimited string or array of values
    - Examples :
        - “filter” : { “project_id” : ”1021271,1020138” , “operator” : “and” , “pi_name” : “Gary” }
        - “filter” : { “project_id” : [1021271 , 1020138] , “pi_name” : “Gary” } #Default is OR
        - “filter” : { “proposal_id” : ”102,321” , “operator” : “or” , “pi_name” : “Gary” }
    - To filter on any other fields, you must provide full name of the field. Example :
        - “filter” : { “metadata.library_name” : “HCSN” }
- Return value :
    - JSON response consisting of following fields :

      hits : Array of results returned
      hits._source : Actual source document (entire metadata jamo record)
      hits._index : Name of ES index searched on
      hits._score : ES search result score
      hits._id : Unique record id, which is used to fetch files.
      total : Total number of results matching this search.

    - Identifying public data vs private : If “_source" contains “_es_public_data” field then the record is public data.

- A max of 10,000 results can be accessed for a single search (i.e max page number is 1000 for the default page size of 10).

2) **Fetch** :

- Accepts JSON input. Options are :
    - “ids” (required) - List or comma delimit string of ids of docs to be retrieved
    - “path” (required) - Abs. path to destination dir on dtn.chicago.kbase.us
    - **“userid” (optional) - KBase userid. If there is a valid corresponding JGI identity, then user will be able to search or fetch his private data. Else, only public data is provided.**
- Checks to see if the user has access to the requested ids. Only downloaded eligible

files. If user doesn't have permission to access few ids, response msg will list the invalid ids.

- Returns a job id if the inputs ids provided are correct (else returns a 400 error response)

```
  Example :  -X POST https://jgi-kbase.nersc.gov/fetch -d '{
                                    "ids" : "51eee03d067c014b2c665cb7 ,
51edd9c8067c014b2c6625e9" , "path" : "/data/sdm"
                                    }'
                    { "id" : JOB_1 }
```

- If the file requested is not on disk, system has to wait for the tape restore to happen. So time taken for file transfer can vary based on volume of data and the current restore requests queue length.

3) **Status**:

- Input - Job_id that is returned by fetch.
- Returns status of the job. Shows # of files in various states in :
    - In queue state - waiting in restore queue,
    - restore_in_progress state - being pulled from tape to disk,
    - copy_in_progress state - waiting for scp to complete.
    - *Failed states include* :
    - restore_failed - failed to restore from tape,
    - scp_failed - scp transfer failed.

- Example : -X GET https://jgi-kbase.nersc.gov/status?id="JOB_1"

```
      "Transfer Complete. Transfered 0 files. Scp failed for files =
[u'587b58df7ded5e4229d88737']"
```