Package ‘cloudos’

August 9, 2021

Title R Client Library for CloudOS
Version 0.2.0
Description The 'CloudOS' client library for R makes it easy to interact with CloudOS <https://cloudos.lifebit.ai/> in the R environment for analysis.
License MIT + file LICENSE
Encoding UTF-8
RoxygenNote 7.1.1
Imports httr, utils, methods, dplyr (>= 1.0.0), jsonlite, rappdirs, ggplot2, tibble
Suggests httrtest, kableExtra, knitr, lintr, rmarkdown, testthat
URL https://github.com/lifebit-ai/cloudos
BugReports https://github.com/lifebit-ai/cloudos/issues
NeedsCompilation no
Author Sangram Keshari Sahu [aut, cre], Lifebit Biotech Ltd. [cph]
Maintainer Sangram Keshari Sahu <sangram@lifebit.ai>
Repository CRAN
Date/Publication 2021-08-09 14:30:02 UTC

R topics documented:

- .get_search_json .............................. 2
- .v1_query_to_v2 ................................. 2
- cb_apply_filter_dry_run ..................... 3
- cb_apply_query ................................. 4
- cb_create_cohort ............................... 5
- cb_extract_samples ........................... 6
- cb_get_cohort_phenotypes .................. 6
- cb_get_genotypic_table ..................... 7
- cb_get_participants_table .................. 7
.get_search_json

**Description**

*only used for v1 endpoint - creates v1 search json using the v2 style query*

**Usage**

```python
.get_search_json(my_cohort)
```

**Arguments**

- **my_cohort**
  A cohort object

---

.v1_query_to_v2

**Description**

*Convert a v1 style query (moreFields) to v2 style (query). v2 queries are a superset of v1 queries. A list of v1 phenotype queries are equivalent to a set of nested v2 AND operators containing those phenotypes. This function builds the nested AND query from the flat list of v1 phenotypes.*

**Usage**

```python
.v1_query_to_v2(cohort_more_fields)
```
cb_apply_filter_dry_run

Arguments

cohort more_fields
query information ('moreFields') from .get_cohort_info(cohort_id, cb_version="v1)

Description

This doesn’t update the database but mimics cb_apply_query

Usage

cb_apply_filter_dry_run(cohort, simple_query)

Arguments

cohort A cohort object. (Required) See constructor function cb_create_cohort or cb_load_cohort
simple_query A phenotype query using the "simple query" list structure (see cb_apply_query).

Value

A data frame.

Examples

## Not run:
my_cohort <- cb_load_cohort(cohort_id = "5f9af3793dd2dc6091cd17cd")
cb_apply_filter_dry_run(my_cohort, simple_query = list("22" = list("from" = "2015-05-13", "to" = "2016-04-29"),
                                     "50" = c("Father", "Mother")))

## End(Not run)
cb_apply_query

Apply a query to a cohort

Description

Updates a cohort by applying a new query.

Usage

cb_apply_query(
  cohort,
  simple_query,
  adv_query,
  column_ids,
  keep_query = TRUE,
  keep_columns = TRUE
)

Arguments

cohort       A cohort object. (Required) See constructor function `cb_create_cohort` or `cb_load_cohort`
simple_query A phenotype query using the "simple query" list structure (see example).
adv_query    A phenotype query using the "advanced query" nested list structure (see example). Advanced queries can include logical operators: 'AND', 'OR', 'NOT'.
column_ids   Phenotype IDs to be added as columns in the participant table.
keep_query   If True, combines the newly supplied query with the pre-existing query. Otherwise, pre-existing query is overwritten. (Default: TRUE)
keep_columns If True, pre-existing columns are retained and newly supplied columns are added. Otherwise, pre-existing columns are overwritten. (Default: TRUE)

Value

A confirmation string.

Examples

```r
## Not run:
my_cohort <- cb_load_cohort(cohort_id = "5f9af3793dd2dc6091cd17cd", cb_version = "v1")
cb_apply_query(my_cohort,
simple_query = list("22" = list("from" = "2015-05-13", "to" = "2016-04-29"),
  "50" = c("Father", "Mother"))
)

my_cohort <- cb_load_cohort(cohort_id = "5f9af3793dd2dc6091cd17cd", cb_version = "v2")
adv_query <- list(
  "operator" = "AND",
  "queries" = list(
```
cb_create_cohort

list( "id" = 22, "value" = list("from"="2015-05-13", "to"="2016-04-29")),
list(
  "operator" = "OR",
  "queries" = list(
    list("id" = 32, "value" = c("Cancer", "Rare Diseases")),
    list("id" = 14, "value" = "Yes")
  )
)
)

cb_apply_query(my_cohort, adv_query = adv_query)

## End(Not run)

---

**cb_create_cohort**

*Create Cohort*

**Description**

Creates a new Cohort

**Usage**

```r
cb_create_cohort(cohort_name, cohort_desc, filters = "", cb_version = "v2")
```

**Arguments**

- **cohort_name**: New cohort name to be created. (Required)
- **cohort_desc**: New cohort description to be created. (Optional)
- **filters**: WIP - details will be added.
- **cb_version**: cohort browser version. ["v1" | "v2"] (Optional) Default - "v2"

**Value**

A `cohort` object.

**See Also**

- `cb_load_cohort` for loading a available cohort.

**Examples**

```r
## Not run:
my_cohort <- cb_create_cohort(cohort_name = "Cohort-R",
  cohort_desc = "This cohort is for testing purpose, created from R.")

## End(Not run)
```
**cb_extract_samples**  
*Extract participants - WIP*

**Description**
Extracts selected participants.

**Usage**
```
cb_extract_samples(raw_data)
```

**Arguments**
- `raw_data` A JSON string for selected participants. (Required)

**Value**
A dataframe.

---

**cb_get_cohort_phenotypes**  
*Get data for phenotypes associated with a cohort*

**Description**
Get a dataframe with distribution data for each phenotype associated with a cohort. Associated phenotypes are those found in the "Overview" section of the Cohort Browser Web UI.

**Usage**
```
cb_get_cohort_phenotypes(cohort)
```

**Arguments**
- `cohort` A cohort object. (Required) See constructor function `cb_create_cohort` or `cb_load_cohort`

**Value**
A list of data frames.
cb_get_genotypic_table

Get genotypic table

Description
Get Genotypic table in a dataframe. Optionally genotypic filters can be applied as well.

Usage
cb_get_genotypic_table(cohort, size = 10, geno_filters_query)

Arguments
- cohort: A cohort object. (Required) See constructor functions cb_create_cohort or cb_load_cohort
- size: Number of entries from database. (Optional) Default - 10 (Optional)
- geno_filters_query: Genotypic filter query (Optional)

Value
A dataframe.

cb_get_participants_table

Get participant data table

Description
Get participant data table in a dataframe.

Usage
cb_get_participants_table(cohort, page_number = 0, page_size = 10)

Arguments
- cohort: A cohort object. (Required) See constructor functions cb_create_cohort or cb_load_cohort
- page_number: Number of page. (Optional) Default - 0
- page_size: Number of entries in a page. (Optional) Default - 10

Value
A dataframe.
cb_get_phenotype_metadata

Phenotype metadata

Description

Get the metadata of a phenotype in the cohort browser

Usage

```r
cb_get_phenotype_metadata(pheno_id, cb_version = "v2")
```

Arguments

- **pheno_id**: A phenotype ID. (Required)
- **cb_version**: cohort browser version. (Default: "v2") ["v1" | "v2"]

Value

A data frame.

---

cb_get_phenotype_statistics

Get distribution of a phenotype in a cohort

Description

Retrieve a data frame containing the distribution data for a specific phenotype within a cohort.

Usage

```r
cb_get_phenotype_statistics(cohort, pheno_id)
```

Arguments

- **cohort**: A cohort object. (Required) See constructor function `cb_create_cohort` or `cb_load_cohort`
- **pheno_id**: A phenotype ID. (Required)

Value

A data frame holding distribution data.
cb_list_cohorts

List cohorts

Description
Extracts the data frame with limited cohort data columns.

Usage
   cb_list_cohorts(size = 10, cb_version = "v2")

Arguments
  size               Number of cohort entries from database. (Optional) Default - 10
  cb_version         cohort browser version. ["v1" | "v2"] (Optional) Default - "v2"

Value
A data frame with available cohorts.

Examples
   ## Not run:
   cohorts_list()
   ## End(Not run)

---

cb_load_cohort

Get cohort information

Description
Get all the details about a cohort including applied query.

Usage
   cb_load_cohort(cohort_id, cb_version = "v2")

Arguments
  cohort_id         Cohort id (Required)
  cb_version        cohort browser version (Optional) ["v1" | "v2"]

Value
A cohort object.
cb_plot_filters

See Also
cb_create_cohort for creating a new cohort.

cb_participant_count  Participant Count

Description
Returns the number of participants in a cohort if the supplied query were to be applied.

Usage
cb_participant_count(cohort, simple_query, adv_query, keep_query = TRUE)

Arguments
- cohort: A cohort object. (Required) See constructor function cb_create_cohort or cb_load_cohort
- simple_query: A phenotype query using the "simple query" list structure (see cb_apply_query).
- adv_query: A phenotype query using the "advanced query" nested list structure (see cb_apply_query).
- keep_query: Apply newly specified query on top of existing query (Default: TRUE)

Value
A list with count of participants in the cohort and the total no. of participants in the dataset.

cb_plot_filters  Plot filters

Description
Get a list of ggplot objects, each plot having one filter.

Usage
cb_plot_filters(cohort)

Arguments
- cohort: A cohort object. (Required) See constructor function cb_create_cohort or cb_load_cohort

Value
A list of ggplot objects
cb_search_phenotypes

Examples

## Not run:
my_cohort <- cb_load_cohort(cohort_id = "5f9af3793dd2dc6091cd17cd")
plot_list <- cb_plot_filters(cohort = my_cohort)

plot_list[[1]]

library(ggpubr)
ggpubr::ggarrange(plotlist = plot_list)

## End(Not run)

cb_search_phenotypes

Search available phenotypes

Description

Search for phenotypes in the Cohort Browser that match your term and return a tibble containing the metadata information for each matching phenotype. Use `term = ""` to return all phenotypes.

Usage

cb_search_phenotypes(term, cb_version = "v2")

Arguments

term A term to search. (Required)

cb_version cohort browser version (Optional) [ "v1" | "v2" ]

Value

A tibble with phenotype metadata

Examples

## Not run:
cancer_phenos <- cb_search_phenotypes(term = "cancer")

all_phenos <- cb_search_phenotypes(term = "")

## End(Not run)
cloudos_configure  Configure cloudos

Description
On a system for the first time the cloudos configuration needed to be done. This function can help do that.

Usage
cloudos_configure(base_url, token, team_id)

Arguments
- base_url: Base URL for cloudos
- token: API key or token
- team_id: team/workspace ID

Value
None

cloudos_whoami  whoami

Description
To check the current configuration

Usage
cloudos_whoami()

Value
None
Description

This class creates a cohort object, which holds the information related to a cohort: cohort ID, name, description, query, table columns. This class is used in functions which carry out operations related to specific cohorts. A cohort class object can be created using constructor functions `cb_create_cohort` or `cb_load_cohort`.

Slots

- `id` cohort ID.
- `name` cohort name.
- `desc` cohort description.
- `phenotype_filters` phenotypes displayed in the cohort overview.
- `query` applied query.
- `columns` All the columns
- `cb_version` cohort browser version
# Index

<table>
<thead>
<tr>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>.get_search_json</td>
<td>2</td>
</tr>
<tr>
<td>.v1_query_to_v2</td>
<td>2</td>
</tr>
<tr>
<td>cb_apply_filter_dry_run</td>
<td>3</td>
</tr>
<tr>
<td>cb_apply_query</td>
<td>3, 4, 10</td>
</tr>
<tr>
<td>cb_create_cohort</td>
<td>3, 4, 5, 6–8, 10, 13</td>
</tr>
<tr>
<td>cb_extract_samples</td>
<td>6</td>
</tr>
<tr>
<td>cb_get_cohort_phenotypes</td>
<td>6</td>
</tr>
<tr>
<td>cb_get_genotypic_table</td>
<td>7</td>
</tr>
<tr>
<td>cb_get_participants_table</td>
<td>7</td>
</tr>
<tr>
<td>cb_get_phenotype_metadata</td>
<td>8</td>
</tr>
<tr>
<td>cb_get_phenotype_statistics</td>
<td>8</td>
</tr>
<tr>
<td>cb_list_cohorts</td>
<td>9</td>
</tr>
<tr>
<td>cb_load_cohort</td>
<td>3–8, 9, 10, 13</td>
</tr>
<tr>
<td>cb_participant_count</td>
<td>10</td>
</tr>
<tr>
<td>cb_plot_filters</td>
<td>10</td>
</tr>
<tr>
<td>cb_search_phenotypes</td>
<td>11</td>
</tr>
<tr>
<td>cloudos_configure</td>
<td>12</td>
</tr>
<tr>
<td>cloudos_whoami</td>
<td>12</td>
</tr>
<tr>
<td>cohort</td>
<td>5, 9</td>
</tr>
<tr>
<td>cohort-class</td>
<td>13</td>
</tr>
</tbody>
</table>