Package ‘cloudos’

October 12, 2022

Title R Client Library for CloudOS
Version 0.3.0
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 7.1.2
Imports httr, utils, methods, dplyr (>= 1.0.0), tidyr, jsonlite, rappdirs, ggplot2, tibble
Suggests httptest, 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], Ilya Levantis [aut], Henry Stewart [aut], Lifebit Biotech Ltd. [cph]
Maintainer Sangram Keshari Sahu <sangram@lifebit.ai>
Repository CRAN
Date/Publication 2022-02-09 10:20:02 UTC

R topics documented:

- .get_search_json ........................................... 2
- .v1_query_to_v2 ............................................ 2
- cb_apply_query ............................................. 3
- cb_create_cohort .......................................... 4
- cb_get_genotypic_table .................................... 4
- cb_get_participants_table ................................. 5
- cb_get_participants_table_long .......................... 6
### .get_search_json

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

**Description**

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

**Usage**

`.get_search_json(my_cohort)`

**Arguments**

- **my_cohort**
  - A cohort object

---

### .v1_query_to_v2

*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.*

**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**

`.v1_query_to_v2(cohort_more_fields)`
cb_apply_query

Arguments

cohort_more_fields
query information (‘moreFields’) from .get_cohort_info(cohort_id, cb_version="v1")

cb_apply_query Apply a query to a cohort

Description

Updates a cohort by applying a new query.

Usage

cb_apply_query(cohort, query, keep_query = TRUE)

Arguments

cohort A cohort object. (Required) See constructor function cb_create_cohort or cb_load_cohort
query A phenotype query defined using the phenotype function and logic operators (see example below)
keep_query If True, combines the newly supplied query with the pre-existing query. Otherwise, pre-existing query is overwritten. (Default: TRUE)

Value

The updated cohort object.

Examples

## Not run:
A <- phenotype(id = 13, from = "2016-01-21", to = "2017-02-13")
B <- phenotype(id = 4, value = "Cancer")
A_not_B <- A & !B
my_cohort <- cb_load_cohort(cohort_id = "612f37a57673ed0deaf1333", cb_version = "v2")
my_cohort <- cb_apply_query(my_cohort, query = A_not_B, keep_query = F)
## End(Not run)
cb_create_cohort  

Create Cohort

Description

Creates a new Cohort

Usage

```
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

```
## 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_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)
```
cb_get_participants_table

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.

Description

Get participant data table in a dataframe.

Usage

cb_get_participants_table(cohort, cols, page_number = "all", page_size = 5000)

Arguments

cohort  A cohort object. (Required) See constructor functions cb_create_cohort or cb_load_cohort
cols    Vector of phenotype IDs to fetch as columns in the dataframe. If omitted, columns saved in the cohort are fetched.
page_number  Number of page as integer or 'all' to fetch all data. (Optional) Default - 'all'
page_size  Number of entries in a page. (Optional) Default - 5000

Value

A dataframe.
cb_get_participants_table_long

*Get longform participant data table*

**Description**

Get participant data table in a longform dataframe.

**Usage**

```r
cb_get_participants_table_long(
  cohort,
  cols,
  broadcast = TRUE,
  page_number = 0,
  page_size = 100
)
```

**Arguments**

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

- **cols**
  Vector of phenotype IDs to fetch as columns in the dataframe. If omitted, columns saved in the cohort are fetched.

- **broadcast**
  Whether to broadcast single value phenotypes across rows. (Optional) Can be TRUE, FALSE or a vector of phenotype IDs to specify which phenotypes to broadcast. Default - TRUE

- **page_number**
  Number of page (can be ‘all’ to fetch all data). (Optional) Default - 0

- **page_size**
  Number of entries in a page. (Optional) Default - 10

**Value**

A tibble.

---

cb_get_phenotype_metadata

*Phenotype metadata*

**Description**

Get the metadata of a phenotype in the cohort browser

**Usage**

```r
cb_get_phenotype_metadata(phenoid, cb_version = "v2")
```
cb_get_phenotype_statistics

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pheno_id</td>
<td>A phenotype ID. (Required)</td>
</tr>
<tr>
<td>cb_version</td>
<td>cohort browser version. (Default: &quot;v2&quot;) [ &quot;v1&quot;</td>
</tr>
</tbody>
</table>

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,
  max_depth = Inf,
  page_number = "all",
  page_size = 1000
)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cohort</td>
<td>A cohort object. (Required) See constructor function <code>cb_create_cohort</code> or <code>cb_load_cohort</code></td>
</tr>
<tr>
<td>pheno_id</td>
<td>A phenotype ID. (Required)</td>
</tr>
<tr>
<td>max_depth</td>
<td>The maximum depth to descend in a 'nested list' phenotype. (Default: Inf)</td>
</tr>
<tr>
<td>page_number</td>
<td>For internal use.</td>
</tr>
<tr>
<td>page_size</td>
<td>For internal use.</td>
</tr>
</tbody>
</table>

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.
\textit{cb\textunderscore participant\textunderscore count} 

\textbf{See Also} 
\begin{itemize}
  \item \textit{cb\textunderscore create\textunderscore cohort} for creating a new cohort.
\end{itemize}

\textbf{cb\_participant\_count} \hspace{1cm} Participant Count

\textbf{Description}

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

\textbf{Usage}

\begin{verbatim}
cb\_participant\_count(cohort, query = list(), keep_query = TRUE)
\end{verbatim}

\textbf{Arguments}

\begin{enumerate}
  \item cohort \hspace{1cm} A cohort object. (Required) See constructor function \textit{cb\textunderscore create\textunderscore cohort} or \textit{cb\textunderscore load\textunderscore cohort}
  \item query \hspace{1cm} A phenotype query defined using the \textit{codephenotype} function and logic operators (see example below)
  \item keep_query \hspace{1cm} Apply newly specified query on top of existing query (Default: TRUE)
\end{enumerate}

\textbf{Value}

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

\textbf{cb\_plot\_filters} \hspace{1cm} Plot filters

\textbf{Description}

Get a list of \textit{ggplot} objects, each plot having one filter.

\textbf{Usage}

\begin{verbatim}
cb\_plot\_filters(cohort)
\end{verbatim}

\textbf{Arguments}

\begin{enumerate}
  \item cohort \hspace{1cm} A cohort object. (Required) See constructor function \textit{cb\textunderscore create\textunderscore cohort} or \textit{cb\textunderscore load\textunderscore cohort}
\end{enumerate}

\textbf{Value}

A list of \textit{ggplot} objects
cb_search_phenotypes

Examples

```r
## 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**

```r
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**

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

all_phenos <- cb_search_phenotypes(term = "")

## End(Not run)
```
cb_set_columns

Set the columns in a cohort

Description
Updates a cohort by applying a new query.

Usage
cb_set_columns(cohort, column_ids, keep_columns = TRUE)

Arguments
- cohort: A cohort object. (Required) See constructor function cb_create_cohort or cb_load_cohort
- column_ids: Vector of phenotype IDs to be added as columns in the participant table.
- keep_columns: If True, pre-existing columns are retained and newly supplied columns are added. Otherwise, pre-existing columns are overwritten. (Default: TRUE)

Value
The updated cohort object.

Examples
## Not run:
my_cohort <- cb_load_cohort(cohort_id = "612f37a57673ed0ddef1333", cb_version = "v2")

my_cohort <- cb_set_columns(my_cohort, c(1, 99, 38), keep_columns = F)

## 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

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>base_url</td>
<td>Base URL for cloudos</td>
</tr>
<tr>
<td>token</td>
<td>API key or token</td>
</tr>
<tr>
<td>team_id</td>
<td>team/workspace ID</td>
</tr>
</tbody>
</table>

Value

None

---

cloudos_whoami  

whoami

Description

To check the current configuration

Usage

cloudos_whoami()

Value

None

---

cohort-class  

cohort class

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.
query_phenotype_ids  IDs of phenotypes used in the query.
columns  All the columns.
um_participants  number of participants in the cohort.
cb_version  cohort browser version.
**phenotype**

---

**Description**

Defines a single phenotype

**Usage**

```
phenotype(id, value, from, to, instance = "0")
```

**Arguments**

- **id**
  A single phenotype id. Possible phenotypes can be explored using the `codecb_search_phenotypes` function
- **value**
  The categorical value of the phenotype id defined
- **from**
  For continuous phenotypes, the lower bound of the desired value range
- **to**
  For continuous phenotypes, the upper bound of the desired value range
- **instance**
  The instance number of the phenotype, default 0

**Value**

A single phenotypes definition that can be combined using &, | and ! operators

**Examples**

```r
## Not run:
continuous_phenotype <- phenotype(id = 13, from = "2016-01-21", to = "2017-02-13")
categorical_phenotype <- phenotype(id = 4, value = "Cancer")
## End(Not run)
```
Index

.get_search_json, 2
.v1_query_to_v2, 2

cb_apply_query, 3
cb_create_cohort, 3, 4, 5–7, 9, 11, 12
cb_get_genotypic_table, 4
cb_get_participants_table, 5
cb_get_participants_table_long, 6
cb_get_phenotype_metadata, 6
cb_get_phenotype_statistics, 7
cb_list_cohorts, 8
cb_load_cohort, 3–7, 8, 9, 11, 12
cb_participant_count, 9
cb_plot_filters, 9
cb_search_phenotypes, 10, 13
cb_set_columns, 11
cloudos_configure, 11
cloudos_whoami, 12
cohort, 4, 8
cohort-class, 12

phenotype, 3, 9, 13