Package 'ENCODExplorer'

October 12, 2016

Name ENCODExplorer

Type Package

Title A compilation of ENCODE metadata

Version 1.4.3

Date 2015-02-25

Author Charles Joly Beauparlant <charles.joly-beauparlant@crchul.ulaval.ca>, Audrey Lemacon <lemacon.audrey@ulaval.ca> and Arnaud Droit <arnaud.droit@crchudequebec.ulaval.ca >

Author@R c(person(``Charles",``Joly Beauparlant", email = ``charles.joly- beauparlant@crchul.ulaval.ca", role = ``aut,cre"), person(``Audrey",``Lemacon", email = ``lemacon.audrey@ulaval.ca", role = ``aut"), person(``Arnaud",``Droit", email = ``arnaud.droit@crchudequebec.ulaval.ca", role = ``aut"), person(``Astrid- Louise",``Deschenes", email = ``Astrid-Louise.Deschenes@crchudequebec.ulaval.ca", role = ``ctb"))

Maintainer Charles Joly Beauparlant <charles.joly-beauparlant@crchul.ulaval.ca>

Description This package allows user to quickly access ENCODE project files metadata and give access to helper functions to query the ENCODE rest api, download ENCODE datasets and save the database in SQLite format.

License Artistic-2.0

BugReports https://github.com/CharlesJB/ENCODExplorer/issues

VignetteBuilder knitr

Depends R (>= 3.3)

Imports tools, jsonlite, RSQLite, parallel, RCurl

Suggests RUnit,BiocGenerics,knitr, curl, httr

LazyData true

biocViews Infrastructure, DataImport

RoxygenNote 5.0.1

NeedsCompilation no

R topics documented:

`accession_df` *Metadata from the ENCODE database in a R object. The tables were generated with the* `prepare_ENCODEdb` *function.*

Description

Metadata from the ENCODE database in a R object. The tables were generated with the `prepare_ENCODEdb` function.

Usage

```
data(accession_df)
```

Format

data frame

Value

data frame

See Also

`get_encode_types` to get a list of possible types. Note that some of the types are empty tables that are not included in the database created with `prepare_ENCODEdb` function.

clean_table *Clean a data.frame that was produced by extract_table*

Description

`data.frames` produced when converting JSON to `data.frame` with the `fromJSON` function will sometime have columns that are lists and/or columns that are `data.frames`.

Usage

```
clean_table(table)
```

Arguments

`table`	The table produced by the `extract_table` function.

Details

This function will either remove columns that are not relevant and convert columns to a vector.

Value

a `data.frame` corresponding to the cleaned version of the input `data.frame`.

downloadEncode *Download files from the Internet.*

Description

After processing to a basic search with the `searchEncode` function or a precise search thanks to the `queryEncode` function, you can proceed to the downloading of all the corresponding files.

Usage

```
downloadEncode(df = NULL, resultSet = NULL, resultOrigin = NULL,
  format = "all", dir = ".", force = TRUE)
```

Arguments

`df`	`list` of two `data.frame` containing ENCODE experiment and dataset metadata.
`resultSet`	the results set.
`resultOrigin`	name of the function used to generate the result set (`searchEncode` or `queryEncode`)
`format`	file format, default = all
`dir`	the name of the directory where the downloaded file will be saved. Default = current directory
`force`	Download file is it already exists and md5sums is valid? Default: TRUE.

Details

This function can be used to download a set of files by providing the results set, its origin (searchEncode or queryEncode), the file format and finally the destination directory.

Value

The downloaded file names, if download worked correctly.

Examples

```
resultSet <- queryEncode(biosample = "A549", file_format = "bam")
## Not run:
downloadEncode(resultSet = resultSet, dir = ".")

## End(Not run)
```

ENCODExplorer *ENCODExplorer*

Description

ENCODExplorer

---

encode_df *Metadata from the ENCODE database in a R object. The tables were generated with the* prepare_ENCODEdb *function.*

Description

Metadata from the ENCODE database in a R object. The tables were generated with the prepare_ENCODEdb function.

Usage

```
data(encode_df)
```

Format

A list of data frames

Value

A list of data frames

See Also

get_encode_types to get a list of possible types. Note that some of the types are empty tables that are not included in the database created with prepare_ENCODEdb function.

export_ENCODEdb_accession

Extract accession numbers from all the datasets of RSQLite databse in a `data.frame`

Description

Extract accession numbers from all the datasets of RSQLite databse in a `data.frame`

Usage

```
export_ENCODEdb_accession(df = NULL, database_filename)
```

Arguments

`df`	`list` of two `data.frame` containing ENCODE experiment and dataset metadata. Default
`database_filename`	The name of the file to save the database into.

Value

a `data.frame` composed of 3 fields : accession, files (`list` of files accessions) and dataset_type.

Examples

```
    database_filename <- system.file("extdata/ENCODEdb.sqlite",
                                                                    package = "ENCODEdb")
    ## Not run:
        export_ENCODEdb_accession(database_filename = database_filename)

## End(Not run)
```

export_ENCODEdb_matrix

Extract essential informations from the RSQLite databse in a `list` *of* `data.frames`

Description

Extract essential informations from the RSQLite databse in a `list` of `data.frames`

Usage

```
export_ENCODEdb_matrix(database_filename, mc.cores = 1)
```

Arguments

database_filename	The name of the file to save the database into.
mc.cores	The number of cores to use. Default 1

Value

a list containing two elements. The first one 'experiment' is a data.frame containing essential informations for each file part of an experiment ; the second one 'dataset' is a data.frame containing essential informations for each file part of a dataset.

Examples

```
    database_filename <- system.file("extdata/ENCODEdb.sqlite",
                                                                    package = "ENCODEdb")
    ## Not run:
        export_ENCODEdb_matrix(database_filename = database_filename)

## End(Not run)
```

---

extract_table *Extract a data.frame corresponding to a table in ENCODE database*

Description

Extract a data.frame corresponding to a table in ENCODE database

Usage

```
extract_table(type)
```

Arguments

type	The type of table to extract from ENCODE rest api.

Value

a data.frame corresponding to the table asked. If no match is found, returns an empty data.frame

get_encode_types *A list of known tables from ENCODE database.*

Description

The type (table) names are extracted from the schema list from ENCODE-DCC github repository: https://github.com/ENCODE-DCC/encoded/tree/master/src/encoded/schemas

Usage

```
get_encode_types()
```

Details

The data is extracted using the github api: https://developer.github.com/guides/getting-started/

Value

a vector of `character` with the names of the known tables in the ENCODE database.

get_schemas *Extract the schemas from ENCODE's github*

Description

The JSONs are fetched from: https://github.com/ENCODE-DCC/encoded/tree/master/src/encoded/schemas

Usage

```
get_schemas()
```

Details

The data is extracted using the github api: https://developer.github.com/guides/getting-started/

The data is then downloaded using the `jsonlite` package.

Value

a `list` of schemas.

prepare_ENCODEdb *Create the RSQLite databse for the tables in ENCODE*

Description

Create the RSQLite databse for the tables in ENCODE

Usage

```
prepare_ENCODEdb(database_filename = "inst/extdata/ENCODEdb.sqlite",
  types = get_encode_types(), overwrite = FALSE)
```

Arguments

database_filename	The name of the file to save the database into.
types	The names of the tables to extract from ENCODE rest api.
overwrite	Should tables already present in database be overwrited Default: FALSE.

Value

is a list with selected tables from ENCODE that were used to create the RSQLite database.

Examples

```
prepare_ENCODEdb(database_filename = "platform.sql", types = "platform")
file.remove("platform.sql")
    ## Not run:
        prepare_ENCODEdb("ENCODEdb.sqlite")

## End(Not run)
```

queryEncode *Produce a subset of data following predefined criteria*

Description

After running the prepare_ENCODEDb function, this function will allow you to extract a subset of data encording to the following criteria : accession, assay name, biosample, dataset accession, file accession, file format, laboratory, donor organism, target and treatment.

Usage

```
queryEncode(df = NULL, set_accession = NULL, assay = NULL,
  biosample = NULL, dataset_accession = NULL, file_accession = NULL,
  file_format = NULL, lab = NULL, organism = NULL, target = NULL,
  treatment = NULL, project = NULL, file_status = "released",
  status = "released", fixed = TRUE, quiet = FALSE)
```

Arguments

df	list of two data.frame containing ENCODE experiment and dataset metadata
set_accession	character string to select the experiment or dataset accession
assay	character string to select the assay type
biosample	character string to select the biosample name
dataset_accession	character string to select the dataset accession
file_accession	character string to select the file accesion
file_format	character string to select the file format
lab	character string to select the laboratory
organism	character string to select the donor organism
target	character string to select the experimental target
treatment	character string to select the treatment
project	character string to select the project
file_status	character string to select the file status ("released", "revoked", "all"). Default "released"
status	character string to select the dataset/experiment status
fixed	logical. If TRUE, pattern is a string to be matched as it is.
quiet	logical enables to switch off the result summary information when setting at TRUE.

Details

By default, the query can be made on an exact match term. This behaviour can be modified by setting the fixed argument at TRUE

Value

a list of two data.frames containing data about ENCODE experiments and datasets

Examples

```
queryEncode(biosample = "A549", file_format = "bam")
```

resolveEncodeAccession

Return a `data.frame` *containing basic datasets information from an accession number*

Description

Return a `data.frame` containing basic datasets information from an accession number

Usage

```
resolveEncodeAccession(accession)
```

Arguments

`accession`	character, dataset accession number

Value

a `data.frame` containing basic datasets information for the requested accession number

Examples

```
res <- resolveEncodeAccession(accession = 'ENCSR361ONJ')$accession
```

searchEncode

Simulate a query on ENCODE website and return the result as a `data.frame`

Description

`data.frames` produced when converting JSON to `data.frame` with the `fromJSON` function will sometime have columns that are lists and/or columns that are `data.frames`.

Usage

```
searchEncode(searchTerm = NULL, limit = 10, quiet = FALSE)
```

Arguments

`searchTerm`	a search term
`limit`	the maximum number of return entries, default 10.
`quiet`	logical value enables to switch off the result summary information when setting at TRUE. will return all the result. It can generate large results set.

Details

This function simulates a basic query on ENCODE website

Value

a `data.frame` corresponding Every object that matches the search term

Examples

```
searchEncode("ChIP-Seq+H3K4me1")
```

`searchToquery` *Convert searchEncode output in queryEncode output.*

Description

After processing to a basic search with the `searchEncode` function you can convert your result in a queryEncode output. Thus you can benefit from the collected metadata.

Usage

```
searchToquery(df = NULL, searchResults, quiet = FALSE)
```

Arguments

`df`	`list` of two `data.frame` containing ENCODE experiment and dataset metadata.
`searchResults`	the results set generated from `searchEncode`
`quiet`	logical enables to switch off the result summary information when setting at TRUE.

Details

The output is compatible with the dowload function.

Value

a `list` of two `data.frames` containing data about ENCODE experiments and datasets

Examples

```
search_res <- searchEncode(searchTerm = "switchgear elavl1", limit = "1")
res <- searchToquery(searchResults = search_res, quiet = TRUE)
```

`update_ENCODExplorer` *Create or update all the needed data for ENCODExplorer*

Description

This function creates or updates ENCODExplorer data according the following steps : 1) Create the RSQLite databse for the tables in ENCODE 2) Extract essential informations from the RSQLite databse in encode_df 3) Extract accession numbers from all the datasets of RSQLite databse in accession_df 4) if overwrite = TRUE, the new encode_df will overwrite the former one else return the newly generated objets.

Usage

```
update_ENCODExplorer(database_filename = "inst/extdata/ENCODEdb.sqlite",
  overwrite = FALSE, mc.cores = 1)
```

Arguments

`database_filename`	The name of the file to save the database into.
`overwrite`	Should tables already present in database be overwrited
`mc.cores`	The number of cores to use. Default 1 Default: `FALSE`.

Value

none if `overwrite` is set to TRUE or return a `list` containg two objects encode_df and accession_df.

Examples

```
    ## Not run:
        update_ENCODExplorer("ENCODEdb.sqlite")

## End(Not run)
```

Index