SejmRP package - how to use it

Piotr Smuda

Description

sejmRP package enables scraping data from Polish Diet's webpage sejm.gov.pl about votings and deputies from seventh to eighth term of office. All data is storaged in database.

Installation

To get started, install the latest version of **sejmRP** from CRAN:

```
install.packages("sejmRP")
```

Creative functions

- create_database
- remove_database
- deputies_create_table
- deputies_update_table
- deputies_add_new
- deputies_get_data
- deputies_get_ids
- votings_create_table
- votings_update_table
- votings_get_date
- votings_get_meetings_links
- votings_get_meetings_table
- votings_get_votings_links
- votings_get_votings_table
- votes_create_table
- votes_update_table
- votes_get_clubs_links
- votes_get_results
- votes_match_deputies_ids
- statements_create_table
- statements_update_table
- statements_get_statement
- statements_get_statements_data
- statements_get_statements_table

Access functions

- get_deputies_table
- get_votings_table
- get_votes_table

- get_statements_table
- get_filtered_votes

How to create and update tables in database

At the begining we need to create a database. For this purpose we use:

```
create_database(dbname, user, password, host)
```

where

- *dbname* is a name of database on the server in PostgreSQL,
- *user* is a username,
- *password* is a password to the database,
- *host* is an address of host or host's IP. These arguments we will use also in the future.

After that we will get four tables:

1. *deputies* with columns:
    1) *id_deputy* - deputy's id,
    2) *nr_term_of_office* - Polish Diet's number of term of office,
    3) *surname_name* - deputy's names and surnames,
2. *votings* with columns:
    1) *id_voting* - voting's id,
    2) *nr_term_of_office* - Polish Diet's number of term of office,
    3) *nr_meeting* - meeting's number,
    4) *date_meeting* - meeting's date,
    5) *nr_voting* - voting's number,
    6) *topic_voting* - voting's topic,
    7) *link_results* - link with voting's results,
3. *votes* with columns:
    1) *id_vote* - vote's id,
    2) *nr_term_of_office* - Polish Diet's number of term of office,
    3) *id_deputy* - deputy's id,
    4) *id_voting* - voting's id,
    5) *vote* - deputy's vote, one of: "Za","Przeciw","Wstrzymał się","Nieobecny",
    6) *club* - deputy's club,
4. *statements* with columns:
    1) *id_statement* - statement's id,
    2) *nr_term_of_office* - Polish Diet's number of term of office,
    3) *surname_name* - author of statement,
    4) *date_statement* - statement's date,
    5) *titles_order_points* - title of order points,
    6) *statement* - content of statement.

Now when we have empty database, we can start completing it with data. First of all we complete the *deputies* table. To do that we use:

```
deputies_create_table(dbname, user, password, host, nr_term_of_office = 8)
```

This function scraps active and inactive deputies' data from Polish Diet's webpage and put it to the table. If you want create *deputies* table with seventh term of office, choose *nr_term_of_office = 8*. In the case of you want get only deputies' data you can use:

```
deputies_get_data(type, nr_term_of_office = 8)
```

where you can choose between active and inactive deputies from eighth term of office. To update *deputies* table use:

```
deputies_update_table(dbname, user, password, host, nr_term_of_office = 8)
```

Remember to choose a proper value of *nr_term_of_office* variable. After that we should complete *votings* table with:

```
votings_create_table(dbname, user, password, host, nr_term_of_office = 8)
```

This function scraps all information about votings from http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=posglos&NrKadencji=7 for *nr_term_of_office = 7* and http://www.sejm.gov.pl/Sejm8.nsf/agent.xsp?symbol=posglos&NrKadencji=8 for *nr_term_of_office = 8*. If you want update *votings* table, try:

```
votings_update_table(dbname, user, password, host, nr_term_of_office = 8,
                     verbose = FALSE)
```

Again - remember to choose a proper value of *nr_term_of_office* variable. The last argument says function if you use want additional info be printed. If you are interested in extra information, you can use additional functions:

```
votings_get_meetings_table(
  page = "http://www.sejm.gov.pl/Sejm8.nsf/agent.xsp?symbol=posglos&NrKadencji=8")
votings_get_votings_table(page)
```

First of them enables downloading table with information about meetings during eighth diet's term of office. The second one does the same with votings during meeting.

Then we need to complete *votes* table. To do that we use:

```
votes_create_table(dbname, user, password, host, nr_term_of_office = 8,
                   windows = .Platform$OS.type == 'windows')
```

The last argument says function if you use Windows, because of encoding issue on this operating system.

To update *votes* table use:

```
votes_update_table(dbname, user, password, host, nr_term_of_office = 8,
                   windows = .Platform$OS.type == "windows",
                   verbose = FALSE)
```

If you want to know how deputies from chosen club voted try:

```
votes_get_results(page)
```

As *page* argument you should put page with this club's voting's results (example).

Finally we should complete *statements* table with:

```
statements_create_table(dbname, user, password, host, nr_term_of_office = 8)
```

To update *statements* table use:

```
statements_update_table(dbname, user, password, host, nr_term_of_office = 8,
                        verbose = FALSE)
```

Like before if you are interested in extra information, you can use additional functions:

```
statements_get_statements_table(page)
statements_get_statement(page, ...)
```

First of them enables downloading table with information about statements during chosen meeting (example). The second one gets statement's content (example).

How to read tables from database

First of all, we have to say that there are special parameters to read tables from database:

- *dbname* = 'sejmrp',
- *user* = 'reader',
- *password* = 'qux94874',
- *host* = 'services.mini.pw.edu.pl'.

If you are only interested in tables, try:

```
get_deputies_table(dbname = 'sejmrp', user = 'reader', password = 'qux94874',
  host = 'services.mini.pw.edu.pl', sorted_by_id = TRUE,
  windows = .Platform$OS.type == 'windows')
get_votings_table(dbname = 'sejmrp', user = 'reader', password = 'qux94874',
  host = 'services.mini.pw.edu.pl', sorted_by_id = TRUE,
  windows = .Platform$OS.type == 'windows')
get_votes_table(dbname = 'sejmrp', user = 'reader', password = 'qux94874',
  host = 'services.mini.pw.edu.pl', sorted_by_id = TRUE,
  windows = .Platform$OS.type == 'windows')
get_statements_table(dbname = 'sejmrp', user = 'reader', password = 'qux94874',
  host = 'services.mini.pw.edu.pl', sorted_by_id = TRUE,
  windows = .Platform$OS.type == 'windows')
```

where

- *sorted_by_id* informs function if table should be sorted by id,
- *windows* informs function if you use Windows operation system.

As you see all of the arguments are default, so you probably use this functions changing only *windows* argument, like:

```
get_deputies_table()
get_deputies_table(windows = FALSE)
```

We do not recommend change *sorted_by_id* to *FALSE*, because data can be unsorted and there may occur some problems during analysis.

There is also a function:

```
get_filtered_votes(dbname = 'sejmrp', user = 'reader', password = 'qux94874',
  host = 'services.mini.pw.edu.pl', windows = .Platform$OS.type == 'windows',
  clubs = character(0), dates = character(0), terms_of_office = integer(0),
  meetings = integer(0), votings = integer(0), deputies = character(0),
  topics = character(0))
```

that retrieves joined *deputies*, *votes* and *votings* tables with filtred data. As you see there are few possible filters:

1. *clubs* - names of clubs. This filter is a character vector with elements like for example: "PO", "PiS", "SLD". It is possible to choose more than one club.
2. *dates* - period of time. This filter is a character vector with two elements in date format "YYYY-MM-DD", where the first describes left boundary of period and the second right boundary. It is possible to choose only one day, just try the same date as first and second element of vector.
3. *terms_of_office* - range of terms of office's numbers. This filter is a integer vector with two elements, where the first describes a left boundary of range and the second a right boundary. It is possible to choose only one term of office, just try the same number as first and second element of vector.
4. *meetings* - range of meetings' numbers. This filter is a integer vector with two elements, where the first describes a left boundary of range and the second a right boundary. It is possible to choose only one meeting, just try the same number as first and second element of vector.
5. *votings* - range of votings' numbers. This filter is a integer vector with two elements, where the first describes a left boundary of range and the second a right boundary. It is possible to choose only one voting, just try the same number as first and second element of vector.
6. *deputies* - full names of deputies. This filter is a character vector with full names of deputies in format: "surname first_name second_name". If you are not sure if the deputy you were thinking about has second name, try "surname first_name" or just "surname". There is high probability that proper deputy will be chosen. It is possible to choose more than one deputy.
7. *topics* - text patterns. This filter is a character vector with text patterns of topics that you are interested about. Note that the votings' topics are written like sentences, so remember about case inflection of nouns and adjectives and use stems of words as patterns. For example if you want to find votings about education (in Polish: szkolnictwo) try "szkolnictw". It is possible to choose more than one pattern.

For example if you want to find every votings of deputies from PO and PiS during 2014 year try:

```
get_filtered_votes(clubs = c("PO", "PiS"), dates = c("2014-01-01", "2014-12-31"))
get_filtered_votes(windows = FALSE, clubs = c("PO", "PiS"),
  dates = c("2014-01-01", "2014-12-31")) #Linux/Mac OS
```

or if you are looking only for votings with referendum use:

```r
get_filtered_votes(topics = "referendum")
get_filtered_votes(windows = FALSE, topics = "referendum")  #Linux/Mac OS
```

Thanks to these filters there is a lot of possibilities of getting data.