

# Package ‘sejmRP’

September 4, 2015

**Title** An information about deputies and votings in Polish diet

**Version** 0.1

**Date** 2015-08-06

**Author** Piotr Smuda [aut, cre]

**Maintainer** Piotr Smuda <piotrsmuda@gmail.com>

**Description** Set of functions that access information about deputies and votings in Polish diet from webpage <http://www.sejm.gov.pl/>.

**Depends** R (>= 3.1.0)

**License** GPL-2

**LazyLoad** true

**LazyData** true

**Imports** RPostgreSQL, DBI, dplyr, XML, rvest, stringi

## R topics documented:

create_database . . . . .	1
deputies_add_new . . . . .	3
deputies_create_table . . . . .	4
deputies_get_data . . . . .	5
deputies_get_ids . . . . .	6
deputies_update_table . . . . .	7
get_deputies_table . . . . .	7
get_filtered_votes . . . . .	8
get_statements_table . . . . .	10
get_votes_table . . . . .	11
get_votings_table . . . . .	12
remove_database . . . . .	13
statements_create_table . . . . .	14
statements_get_statement . . . . .	15
statements_get_statements_data . . . . .	16
statements_update_table . . . . .	17
votes_create_table . . . . .	18
votes_get_clubs_links . . . . .	19

votes_get_results	20
votes_match_deputies_ids	21
votes_update_table	22
votes_create_table	23
votes_get_date	24
votes_get_meetings_links	24
votes_get_meetings_table	25
votes_get_votes_links	26
votes_get_votes_table	27
votes_update_table	28

---

create_database	<i>Creating database</i>
-----------------	--------------------------

---

## Description

Function `create_database` creates a database with four empty tables: `deputies`, `votes`, `statements`.

## Usage

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

## Arguments

<code>dbname</code>	name of database
<code>user</code>	name of user
<code>password</code>	password of database
<code>host</code>	name of host

## Details

Created tables:

1. `deputies` with columns:
  - 1) `id_deputy` - deputy's id,
  - 2) `surname_name` - deputy's names and surnames,
2. `votes` with columns:
  - 1) `id_voting` - voting's id,
  - 2) `nr_meeting` - meeting's number,
  - 3) `date_meeting` - meeting's date,
  - 4) `nr_voting` - voting's number,
  - 5) `topic_voting` - voting's topic,
  - 6) `link_results` - link with voting's results,
3. `votes` with columns:
  - 1) `id_vote` - vote's id,
  - 2) `id_deputy` - deputy's id,
  - 3) `id_voting` - voting's id,
  - 4) `vote` - deputy's vote, one of: "Za", "Przeciw", "Wstrzymal sie", "Nieobecny",
  - 5) `club` - deputy's club,
4. `statements` with columns:

- 1) id\_statement - statement's id, like:  
(meeting's number).(voting's number).(statement's number),
- 2) surname\_name - author of statement,
- 3) date\_statement - statement's date,
- 4) statement - content of statement.

**Value**

invisible NULL

**Note**

All information is stored in PostgreSQL database.

**Author(s)**

Piotr Smuda

**Examples**

```
## Not run:
create_database(dbname,user,password,host)
## End(Not run)
```

---

deputies_add_new	<i>Adding new deputies to table</i>
------------------	-------------------------------------

---

**Description**

Function `deputies_add_new` adds new deputies to a table with deputies.

**Usage**

```
deputies_add_new(dbname,user,password,host,type,id)
```

**Arguments**

<code>dbname</code>	name of database
<code>user</code>	name of user
<code>password</code>	password of database
<code>host</code>	name of host
<code>type</code>	type of deputies which be add to table with deputies: active, inactive
<code>id</code>	id of deputies from which we start add new deputies

**Details**

Function `deputies_add_new` adds new deputies to a table with deputies. Also there is a choice between types of deputies, because on the page of Polish diet deputies are splitted into *active* and *inactive*. In addition id of the last added deputy in *deputies* table is needed.

**Value**

invisible NULL

**Note**

All information is stored in PostgreSQL database.

**Author(s)**

Piotr Smuda

**Examples**

```
## Not run:
deputies_add_new(dbname,user,password,host,"active",id)
deputies_add_new(dbname,user,password,host,"inactive",id)
## End(Not run)
```

---

```
deputies_create_table
```

*Creating table with deputies*

---

**Description**

Function `deputies_create_table` creates a table with deputies.

**Usage**

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

**Arguments**

<code>dbname</code>	name of database
<code>user</code>	name of user
<code>password</code>	password of database
<code>host</code>	name of host

**Value**

invisible NULL

**Note**

Use only this function for first time, when the *deputies* table is empty. Then use `deputies_update_table`.  
All information is stored in PostgreSQL database.

**Author(s)**

Piotr Smuda

## Examples

```
## Not run:
deputies_create_table(dbname,user,password,host)
## End(Not run)
```

---

deputies\_get\_data    *Getting data about deputies*

---

## Description

Function `deputies_get_data` gets data about deputies.

## Usage

```
deputies_get_data(type)
```

## Arguments

`type`                      type of deputies which be add to table with deputies: active, inactive

## Details

Function `deputies_get_data` gets deputies' ids and personal data like name and surname. Also there is a choice between types of deputies, because on the page of Polish diet deputies are splitted into *active* and *inactive*.

## Value

data frame with two columns: `id_deputy`, `surname_name`

## Note

All information is stored in PostgreSQL database.

## Author(s)

Piotr Smuda

## Examples

```
## Not run:
deputies_get_data("active")
deputies_get_data("inactive")
## End(Not run)
```

---

deputies_get_ids	<i>Getting deputies' ids</i>
------------------	------------------------------

---

## Description

Function `deputies_get_ids` gets deputies' ids from *deputies* table.

## Usage

```
deputies_get_ids(dbname, user, password, host, windows)
```

## Arguments

<code>dbname</code>	name of database
<code>user</code>	name of user
<code>password</code>	password of database
<code>host</code>	name of host
<code>windows</code>	information of used operation system; default: TRUE

## Details

Function `deputies_get_ids` gets deputies' ids from *deputies* table. As result of this function you get named character vector with ids, where their names are names and surnames of deputies. Because of encoding issue on Windows operation system, you need to select if you use Windows.

## Value

named character vector

## Note

All information is stored in PostgreSQL database.

## Author(s)

Piotr Smuda

## Examples

```
## Not run:  
deputies_get_ids(dbname, user, password, host, TRUE)  
deputies_get_ids(dbname, user, password, host, FALSE)  
## End(Not run)
```

---

deputies\_update\_table  
*Updating table with deputies*

---

**Description**

Function `deputies_update_table` updates a table with deputies.

**Usage**

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

**Arguments**

<code>dbname</code>	name of database
<code>user</code>	name of user
<code>password</code>	password of database
<code>host</code>	name of host

**Value**

invisible NULL

**Note**

All information is stored in PostgreSQL database.

**Author(s)**

Piotr Smuda

**Examples**

```
## Not run:  
deputies_update_table(dbname, user, password, host)  
## End(Not run)
```

---

`get_deputies_table` *Importing deputies table from a database*

---

**Description**

Function `get_deputies_table` imports deputies table from a database.

**Usage**

```
get_deputies_table(dbname = 'sejmrp', user = 'reader',  
  password = 'qux94874', host = 'services.mini.pw.edu.pl',  
  sorted_by_id = TRUE, windows = TRUE)
```

**Arguments**

dbname	name of database; default: 'sejmrp'
user	name of user; default: 'reader'
password	password of database; default: 'qux94874'
host	name of host; default: 'services.mini.pw.edu.pl'
sorted_by_id	information if table should be sorted by id; default: TRUE
windows	information of used operation system; default: TRUE

**Details**

Function `get_deputies_table` imports deputies table from a database. The result of this function is a data frame with deputies' data. Because of encoding issue on Windows operation system, you need to select if you use Windows.

**Value**

data frame

**Note**

Default parameters use privileges of 'reader'. It can only SELECT data from database. All information is stored in PostgreSQL database.

**Author(s)**

Piotr Smuda

**Examples**

```
## Not run:
deputies <- get_deputies_table()
dim(deputies)
# [1] 512  2
names(deputies)
# [1] "id_deputy"      "surname_name"
## End(Not run)
```

---

`get_filtered_votes` *Retrieve filtered votes from a database*

---

**Description**

Function `get_filtered_votes` reads filtered votes from a database.

**Usage**

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



## Arguments

dbname	name of database; default: 'sejmrp'
user	name of user; default: 'reader'
password	password of database; default: 'qux94874'
host	name of host; default: 'services.mini.pw.edu.pl'
windows	information of used operation system; default: TRUE
clubs	names of clubs that will be taken to filter data from database; default: character(0)
dates	period of time that will be taken to filter data from database; default: character(0)
meetings	range of meetings' numbers that will be taken to filter data from database; default: integer(0)
votings	range of votings' numbers that will be taken to filter data from database; default: integer(0)
deputies	full names of deputies that will be taken to filter data from database; default: character(0)
topics	text patterns that will be taken to filter data from database; default: character(0)

## Details

Function `get_filtered_votes` reads filtered votes from a database. The result of this function is an invisible data frame with statements' data.

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. 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.
4. 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.
5. 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.
6. 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.

If you did not choose any filter, the whole database will be downloaded. Note that, due to data size ( $\leq \sim 150$  MB) it may take few seconds / minutes to download all votes.

Because of encoding issue on Windows operation system, you also need to select if you use Windows.

**Value**

data frame with NULL

**Note**

Default parameters use privileges of 'reader'. It can only SELECT data from database.  
All information is stored in PostgreSQL database.

**Author(s)**

Piotr Smuda

**Examples**

```
## Not run:
filtered_votes <- get_filtered_votes()
dim(filtered_votes)
# [1] 2741899      8
names(filtered_votes)
[1] "surname_name" "club" "vote" "id_voting" "nr_meeting"
[6] "nr_voting" "date_meeting" "topic_voting"
object.size(filtered_votes)
# 144250888 bytes
## End(Not run)
```

---

get\_statements\_table

*Importing statements table from a database*

---

**Description**

Function `get_statements_table` imports statements table from a database.

**Usage**

```
get_statements_table(dbname = 'sejmrp', user = 'reader',
  password = 'qux94874', host = 'services.mini.pw.edu.pl',
  sorted_by_id = TRUE, windows = TRUE)
```

**Arguments**

<code>dbname</code>	name of database; default: 'sejmrp'
<code>user</code>	name of user; default: 'reader'
<code>password</code>	password of database; default: 'qux94874'
<code>host</code>	name of host; default: 'services.mini.pw.edu.pl'
<code>sorted_by_id</code>	information if table should be sorted by id; default: TRUE
<code>windows</code>	information of used operation system; default: TRUE

**Details**

Function `get_statements_table` imports statements table from a database. The result of this function is a data frame with statements' data. Because of encoding issue on Windows operation system, you need to select if you use Windows.

**Value**

data frame

**Note**

Default parameters use privileges of 'reader'. It can only SELECT data from database. All information is stored in PostgreSQL database.

**Author(s)**

Piotr Smuda

**Examples**

```
## Not run:
statements <- get_statements_table()
dim(statements)
# [1] 42169      4
names(statements)
# [1] "id_statement" "surname_name" "date_statement" "statement"
## End(Not run)
```

---

get_votes_table	<i>Importing votes table from a database</i>
-----------------	--

---

**Description**

Function `get_votes_table` imports votes table from a database.

**Usage**

```
get_votes_table(dbname = 'sejmrp', user = 'reader',
  password = 'qux94874', host = 'services.mini.pw.edu.pl',
  sorted_by_id = TRUE, windows = TRUE)
```

**Arguments**

dbname	name of database; default: 'sejmrp'
user	name of user; default: 'reader'
password	password of database; default: 'qux94874'
host	name of host; default: 'services.mini.pw.edu.pl'
sorted_by_id	information if table should be sorted by id; default: TRUE
windows	information of used operation system; default: TRUE

**Details**

Function `get_votes_table` imports votes table from a database. The result of this function is a data frame with votes' data. Because of encoding issue on Windows operation system, you need to select if you use Windows.

**Value**

data frame

**Note**

Default parameters use privileges of 'reader'. It can only SELECT data from database.

All information is stored in PostgreSQL database.

**Author(s)**

Piotr Smuda

**Examples**

```
## Not run:
votes <- get_votes_table()
dim(votes)
# [1] 2741899      5
names(votes)
# [1] "id_vote"      "id_deputy"    "id_voting"    "vote"         "club"
# ' object.size(votes)
# 87767112 bytes
## End(Not run)
```

---

`get_votings_table`    *Importing votings table from a database*

---

**Description**

Function `get_votings_table` imports votings table from a database.

**Usage**

```
get_votings_table(dbname = 'sejmrp', user = 'reader',
  password = 'qux94874', host = 'services.mini.pw.edu.pl',
  sorted_by_id = TRUE, windows = TRUE)
```

**Arguments**

<code>dbname</code>	name of database; default: 'sejmrp'
<code>user</code>	name of user; default: 'reader'
<code>password</code>	password of database; default: 'qux94874'
<code>host</code>	name of host; default: 'services.mini.pw.edu.pl'
<code>sorted_by_id</code>	information if table should be sorted by id; default: TRUE
<code>windows</code>	information of used operation system; default: TRUE

**Details**

Function `get_votings_table` imports votings table from a database. The result of this function is a data frame with votings' data. Because of encoding issue on Windows operation system, you need to select if you use Windows.

**Value**

data frame

**Note**

Default parameters use privileges of 'reader'. It can only SELECT data from database.  
All information is stored in PostgreSQL database.

**Author(s)**

Piotr Smuda

**Examples**

```
## Not run:
votings <- get_votings_table()
dim(votings)
# [1] 5969    6
names(votings)
# [1] "id_voting"      "nr_meeting"     "date_meeting"
# [4] "nr_voting"      "topic_voting"   "link_results"
## End(Not run)
```

---

remove_database	<i>Removing database</i>
-----------------	--------------------------

---

**Description**

Function `remove_database` remove whole database.

**Usage**

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

**Arguments**

dbname	name of database
user	name of user
password	password of database
host	name of host

**Value**

invisible NULL

**Note**

All information is stored in PostgreSQL database.

**Author(s)**

Piotr Smuda

**Examples**

```
## Not run:
remove_database(dbname,user,password,host)
## End(Not run)
```

---

```
statements_create_table
```

*Creating table with deputies' statements*

---

**Description**

Function `statements_create_table` creates a table with deputies' statements.

**Usage**

```
statements_create_table(dbname,user,password,host,
  home_page="http://www.sejm.gov.pl/Sejm7.nsf/")
```

**Arguments**

<code>dbname</code>	name of database
<code>user</code>	name of user
<code>password</code>	password of database
<code>host</code>	name of host
<code>home_page</code>	main page of polish diet: <a href="http://www.sejm.gov.pl/Sejm7.nsf/">http://www.sejm.gov.pl/Sejm7.nsf/</a>

**Value**

invisible NULL

**Note**

Use only this function for first time, when the *statements* table is empty. Then use `statements_update_table`.  
All information is stored in PostgreSQL database.

**Author(s)**

Piotr Smuda, Tomasz Mikolajczyk

**Examples**

```
## Not run:
statements_create_table(dbname,user,password,host)
## End(Not run)
```

---

`statements_get_statement`*Getting statements*

---

## Description

Function `statements_get_statement` gets statement's content.

## Usage

```
statements_get_statement (page)
```

## Arguments

<code>page</code>	deputy's statement's page
-------------------	---------------------------

## Details

Function `statements_get_statement` gets statement's content. Example of page with deputy's statement: <http://www.sejm.gov.pl/Sejm7.nsf/wypowiedz.xsp?posiedzenie=15&dzien=1&wyp=008>

## Value

character vector

## Note

All information is stored in PostgreSQL database.

## Author(s)

Piotr Smuda, Tomasz Mikolajczyk

## Examples

```
## Not run:
page <- paste0("http://www.sejm.gov.pl/Sejm7.nsf/",
               "wypowiedz.xsp?posiedzenie=15&dzien=1&wyp=008")
statements_get_statement (page)
## End(Not run)
```

---

```
statements_get_statements_data
```

*Getting data about statements*

---

## Description

Function `statements_get_statements_data` gets data about statements.

## Usage

```
statements_get_statements_data(statements_links,
                               home_page="http://www.sejm.gov.pl/Sejm7.nsf/")
```

## Arguments

<code>statements_links</code>	list of elements of XMLNodeSet class with statements' ids, links and their's authors
<code>home_page</code>	main page of polish diet: <a href="http://www.sejm.gov.pl/Sejm7.nsf/">http://www.sejm.gov.pl/Sejm7.nsf/</a>

## Details

Function `statements_get_statements_data` gets data about statements like author, page with content of statement and it's id.

## Value

data frame with three columns: names, `statements_links`, ids

## Note

All information is stored in PostgreSQL database.

## Author(s)

Piotr Smuda, Tomasz Mikolajczyk

## Examples

```
## Not run:
page <- html(paste0("http://www.sejm.gov.pl/Sejm7.nsf/",
                    "wypowiedz.xsp?posiedzenie=15&dzien=1&wyp=0"))
page <- html_nodes(page, ".stenogram")
statements_links <- html_nodes(page, "h2 a")
statements_get_statements_data(statements_links,
                               home_page="http://www.sejm.gov.pl/Sejm7.nsf/")
## End(Not run)
```



---

`statements_update_table`*Updating table with deputies' statements*

---

**Description**

Function `statements_update_table` updates a table with deputies' statements.

**Usage**

```
statements_update_table (dbname, user, password, host,  
    home_page="http://www.sejm.gov.pl/Sejm7.nsf/")
```

**Arguments**

<code>dbname</code>	name of database
<code>user</code>	name of user
<code>password</code>	password of database
<code>host</code>	name of host
<code>home_page</code>	main page of polish diet: <a href="http://www.sejm.gov.pl/Sejm7.nsf/">http://www.sejm.gov.pl/Sejm7.nsf/</a>

**Value**

invisible NULL

**Note**

All information is stored in PostgreSQL database.

**Author(s)**

Piotr Smuda, Tomasz Mikolajczyk

**Examples**

```
## Not run:  
statements_update_table (dbname, user, password, host)  
## End(Not run)
```

---

votes\_create\_table *Creating table with votes*

---

### Description

Function votes\_create\_table creates a table with votes.

### Usage

```
votes_create_table(dbname, user, password, host,  
  home_page="http://www.sejm.gov.pl/Sejm7.nsf/", windows=TRUE)
```

### Arguments

dbname	name of database
user	name of user
password	password of database
host	name of host
home_page	main page of polish diet: <a href="http://www.sejm.gov.pl/Sejm7.nsf/">http://www.sejm.gov.pl/Sejm7.nsf/</a>
windows	information of used operation system; default: TRUE

### Value

invisible NULL

### Note

Use only this function for first time, when the *votes* table is empty. Then use votes\_update\_table.

There is a possibility that someone's voice reader broke during voting and this situation is treated like this deputy was absent. Even if deputy made a decision, he's/she's vote is "Nieobecny".

All information is stored in PostgreSQL database.

### Author(s)

Piotr Smuda

### Examples

```
## Not run:  
home_page <- http://www.sejm.gov.pl/Sejm7.nsf/  
votes_create_table(dbname, user, password, host, home_page, TRUE)  
votes_create_table(dbname, user, password, host, home_page, FALSE)  
## End(Not run)
```

---

`votes_get_clubs_links`*Getting links with voting's results for each club*

---

## Description

Function `votes_get_clubs_links` gets links with voting's results for each club from voting's page.

## Usage

```
votes_get_clubs_links(home_page="http://www.sejm.gov.pl/Sejm7.nsf/",  
                      page)
```

## Arguments

<code>home_page</code>	main page of polish diet: <a href="http://www.sejm.gov.pl/Sejm7.nsf/">http://www.sejm.gov.pl/Sejm7.nsf/</a>
<code>page</code>	voting's page

## Details

Function `votes_get_clubs_links` gets links with voting's results for each club from voting's page. Example of a voting's page: <http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=glosowania&NrKadencji=7&NrPosiedzenia=1&NrGlosowania=1>

## Value

data frame with two columns: club, links

## Note

All information is stored in PostgreSQL database.

## Author(s)

Piotr Smuda

## Examples

```
## Not run:  
home_page <- "http://www.sejm.gov.pl/Sejm7.nsf/"  
page <- paste0("http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?",  
              "symbol=glosowania&NrKadencji=7&NrPosiedzenia=1&NrGlosowania=1")  
votes_get_clubs_links(home_page, page)  
## End(Not run)
```

---

votes\_get\_results    *Getting voting's results for each club*

---

## Description

Function `votes_get_results` gets voting's results for each club.

## Usage

```
votes_get_results(page)
```

## Arguments

page	club's voting's results page
------	------------------------------

## Details

Function `votes_get_results` gets voting's results for each club. Example of page with voting's results of PO club: <http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=klubglos&IdGlosowania=37494&KodKlubu=PO>

## Value

data frame with two columns: deputy, vote

## Note

All information is stored in PostgreSQL database.

## Author(s)

Piotr Smuda

## Examples

```
## Not run:
page <- paste0("http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?",
               "symbol=klubglos&IdGlosowania=37494&KodKlubu=PO")
votes_get_results(page)
votes_get_results(page)
## End(Not run)
```

---

votes\_match\_deputies\_ids

*Matching deputies to theirs' ids*


---

## Description

Function `votes_match_deputies_ids` matches deputies from voting's results page to theirs' ids from *deputies* table.

## Usage

```
votes_match_deputies_ids(dbname, user, password, host, page, windows=TRUE)
```

## Arguments

<code>dbname</code>	name of database
<code>user</code>	name of user
<code>password</code>	password of database
<code>host</code>	name of host
<code>page</code>	club's voting's results page
<code>windows</code>	information of used operation system; default: TRUE

## Details

Function `votes_match_deputies_ids` matches deputies from voting's results page to theirs' ids from *deputies* table. The result of this function is a data frame with deputies' data, ids and votes. Because of encoding issue on Windows operation system, you need to select if you use Windows. Example of page with voting's results of PO club: <http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=klubglos&IdGlosowania=37494&KodKlubu=PO>

## Value

data frame with three columns: deputy, vote, id

## Note

All information is stored in PostgreSQL database.

## Author(s)

Piotr Smuda

## Examples

```
## Not run:
page <- paste0("http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?",
               "symbol=klubglos&IdGlosowania=37494&KodKlubu=PO")
votes_match_deputies_ids(dbname, user, password, host, page, TRUE)
votes_match_deputies_ids(dbname, user, password, host, page, FALSE)
## End(Not run)
```

---

votes\_update\_table *Updating table with votes*

---

### Description

Function votes\_update\_table updates a table with votes.

### Usage

```
votes_update_table(dbname, user, password, host,  
  home_page="http://www.sejm.gov.pl/Sejm7.nsf/", windows=TRUE)
```

### Arguments

dbname	name of database
user	name of user
password	password of database
host	name of host
home_page	main page of polish diet: <a href="http://www.sejm.gov.pl/Sejm7.nsf/">http://www.sejm.gov.pl/Sejm7.nsf/</a>
windows	information of used operation system; default: TRUE

### Value

invisible NULL

### Note

There is a possibility that someone's voice reader broke during voting and this situation is treated like this deputy was absent. Even if deputy made a decision, he's/she's vote is "Nieobecny".

All information is stored in PostgreSQL database.

### Author(s)

Piotr Smuda

### Examples

```
## Not run:  
home_page <- http://www.sejm.gov.pl/Sejm7.nsf/  
votes_update_table(dbname, user, password, host, home_page, TRUE)  
votes_update_table(dbname, user, password, host, home_page, FALSE)  
## End(Not run)
```

---

votings\_create\_table

*Creating table with votings*


---

## Description

Function `votings_create_table` creates a table with votings.

## Usage

```
votings_create_table(dbname, user, password, host,
  home_page="http://www.sejm.gov.pl/Sejm7.nsf/", page=
  "http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=posglos&NrKadencji=7")
```

## Arguments

<code>dbname</code>	name of database
<code>user</code>	name of user
<code>password</code>	password of database
<code>host</code>	name of host
<code>home_page</code>	main page of polish diet: <a href="http://www.sejm.gov.pl/Sejm7.nsf/">http://www.sejm.gov.pl/Sejm7.nsf/</a>
<code>page</code>	page with votings in polish diet: <a href="http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=posglos&amp;NrKadencji=7">http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=posglos&amp;NrKadencji=7</a>

## Value

invisible NULL

## Note

Use only this function for first time, when the *votings* table is empty. Then use `votings_update_table`.  
All information is stored in PostgreSQL database.

## Author(s)

Piotr Smuda

## Examples

```
## Not run:
votings_create_table(dbname, user, password, host)
## End(Not run)
```

---

votings_get_date	<i>Getting date of meeting</i>
------------------	--------------------------------

---

**Description**

Function `votings_get_date` gets a date of meeting.

**Usage**

```
votings_get_date(page)
```

**Arguments**

page	meeting's page
------	----------------

**Details**

Example of a meeting's page: <http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=listaglos&IdDnia=1179>

**Value**

date in format YYYY-MM-DD as character

**Note**

All information is stored in PostgreSQL database.

**Author(s)**

Piotr Smuda

**Examples**

```
## Not run:  
page <- "http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=listaglos&IdDnia=1179"  
votings_get_date(page)  
## End(Not run)
```

---

votings_get_meetings_links	<i>Getting meetings' links</i>
----------------------------	--------------------------------

---

**Description**

Function `votings_get_meetings_links` gets meetings' links.

**Usage**

```
votings_get_meetings_links(  
  home_page="http://www.sejm.gov.pl/Sejm7.nsf/", page=  
  "http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=posglos&NrKadencji=7")
```



### Arguments

home_page	main page of polish diet: <a href="http://www.sejm.gov.pl/Sejm7.nsf/">http://www.sejm.gov.pl/Sejm7.nsf/</a>
page	page with votings in polish diet: <a href="http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=posglos&amp;NrKadencji=7">http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=posglos&amp;NrKadencji=7</a>

### Value

character vector

### Note

All information is stored in PostgreSQL database.

### Author(s)

Piotr Smuda

### Examples

```
## Not run:
votings_get_meetings_links()
## End(Not run)
```

---

votings\_get\_meetings\_table  
*Getting meetings' table*

---

### Description

Function `votings_get_meetings_table` gets meetings' table.

### Usage

```
votings_get_meetings_table(page=
  "http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=posglos&NrKadencji=7")
```

### Arguments

page	page with votings in polish diet: <a href="http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=posglos&amp;NrKadencji=7">http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=posglos&amp;NrKadencji=7</a>
------	---

### Details

Function `votings_get_meetings_table` gets meetings' table. The result of this function is a data frame with three columns, where the first includes numbers of meetings, the second theirs' dates in Polish and the third is with numbers of votings on each meeting.

### Value

data frame with three unnamed columns

**Note**

All information is stored in PostgreSQL database.

**Author(s)**

Piotr Smuda

**Examples**

```
## Not run:
votings_get_meetings_table()
## End (Not run)
```

---

```
votings_get_votings_links
      Getting votings' links
```

---

**Description**

Function `votings_get_votings_links` gets votings' links from meeting's page.

**Usage**

```
votings_get_votings_links(home_page="http://www.sejm.gov.pl/Sejm7.nsf/",
  page)
```

**Arguments**

<code>home_page</code>	main page of polish diet: <a href="http://www.sejm.gov.pl/Sejm7.nsf/">http://www.sejm.gov.pl/Sejm7.nsf/</a>
<code>page</code>	meeting's page

**Details**

Example of a meeting's page: <http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=listaglos&IdDnia=1179>

**Value**

character vector

**Note**

All information is stored in PostgreSQL database.

**Author(s)**

Piotr Smuda

**Examples**

```
## Not run:
home_page <- "http://www.sejm.gov.pl/Sejm7.nsf/"
page <- "http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=listaglos&IdDnia=1179"
votings_get_votings_links(home_page, page)
## End (Not run)
```

---

votings\_get\_votings\_table  
*Getting votings' table*

---

## Description

Function `votings_get_votings_table` gets votings' table from meeting's page.

## Usage

```
votings_get_votings_table (page)
```

## Arguments

page	meeting's page
------	----------------

## Details

Function `votings_get_votings_table` gets votings' table from meeting's page. Example of a meeting's page: <http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=listaglos&IdDnia=1179>. The result of this function is a data frame with three columns, where the first includes numbers of votings, the second voting's time and the third is with voting's topics.

## Value

data frame with three columns: Nr, Godzina (Time), Temat (Topic)

## Note

All information is stored in PostgreSQL database.

## Author(s)

Piotr Smuda

## Examples

```
## Not run:
page <- "http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=listaglos&IdDnia=1179"
votings_get_votings_table (page)
## End (Not run)
```

---

`votings_update_table`*Updating table with votings*

---

**Description**

Function `votings_update_table` updates table with votings.

**Usage**

```
votings_update_table(dbname,user,password,host,  
  home_page="http://www.sejm.gov.pl/Sejm7.nsf/",page=  
  "http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=posglos&NrKadencji=7")
```

**Arguments**

<code>dbname</code>	name of database
<code>user</code>	name of user
<code>password</code>	password of database
<code>host</code>	name of host
<code>home_page</code>	main page of polish diet: <a href="http://www.sejm.gov.pl/Sejm7.nsf/">http://www.sejm.gov.pl/Sejm7.nsf/</a>
<code>page</code>	page with votings in polish diet: <a href="http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=posglos&amp;NrKadencji=7">http://www.sejm.gov.pl/Sejm7.nsf/agent.xsp?symbol=posglos&amp;NrKadencji=7</a>

**Value**

invisible NULL

**Note**

All information is stored in PostgreSQL database.

**Author(s)**

Piotr Smuda

**Examples**

```
## Not run:  
votings_update_table(dbname,user,password,host)  
## End(Not run)
```