

# Package ‘sejmRP’

May 9, 2026

**Title** An Information About Deputies and Votings in Polish Diet from Seventh to Eighth Term of Office

**Version** 1.3.4

**Date** 2017-03-28

**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>>. The package was developed as a result of an internship in MI2 Group - <<http://mi2.mini.pw.edu.pl>>, Faculty of Mathematics and Information Science, Warsaw University of Technology.

**BugReports** <http://github.com/mi2-warsaw/sejmRP/issues>

**Depends** R (>= 3.1.0)

**License** GPL-2

**LazyLoad** true

**LazyData** true

**Imports** DBI, dplyr, tidyr, cluster, factoextra, RPostgreSQL, rvest, stringi, XML, xml2

**RoxygenNote** 5.0.1

**NeedsCompilation** no

**Author** Piotr Smuda [aut, cre],  
Przemyslaw Biecek [aut],  
Tomasz Mikolajczyk [ctb]

**Repository** CRAN

**Date/Publication** 2017-03-28 17:29:47 UTC

## Contents

create_database . . . . .	2
deputies_add_new . . . . .	4
deputies_create_table . . . . .	5
deputies_get_data . . . . .	6

deputies_get_ids . . . . .	7
deputies_update_table . . . . .	8
get_deputies_table . . . . .	9
get_filtered_statements . . . . .	10
get_filtered_votes . . . . .	12
get_statements_table . . . . .	14
get_votes_table . . . . .	15
get_votings_table . . . . .	17
remove_database . . . . .	18
safe_html . . . . .	19
safe_readHTMLTable . . . . .	20
statements_create_table . . . . .	21
statements_get_statement . . . . .	22
statements_get_statements_data . . . . .	23
statements_get_statements_table . . . . .	24
statements_update_table . . . . .	25
votes_create_table . . . . .	26
votes_get_clubs_links . . . . .	27
votes_get_results . . . . .	28
votes_match_deputies_ids . . . . .	29
votes_update_table . . . . .	30
votings_create_table . . . . .	31
votings_get_date . . . . .	32
votings_get_meetings_links . . . . .	33
votings_get_meetings_table . . . . .	34
votings_get_votings_links . . . . .	35
votings_get_votings_table . . . . .	36
votings_update_table . . . . .	37

---

<b>Index</b>	<b>38</b>
--------------	-----------

---

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

---

**Description**

Function create\_database creates a database with four empty tables: deputies, votings, votes, statements.

**Usage**

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

**Arguments**

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

**Details**

Created 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',  
'Wstrzymal sie', 'Nieobecny',
  - 6) 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) 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.

**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      *Adding new deputies to table*

---

**Description**

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

**Usage**

```
deputies_add_new(dbname, user, password, host, type, id,  
nr_term_of_office = 8)
```

**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
<code>nr_term_of_office</code>	number of term of office of Polish Diet; default: 8

**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,  
nr_term_of_office = 8)
```

**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>nr_term_of_office</code>	number of term of office of Polish Diet; default: 8

**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, nr_term_of_office = 8)
```

### Arguments

`type`                    type of deputies which be add to table with deputies: active, inactive  
`nr_term_of_office`        number of term of office of Polish Diet; default: 8

### 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,  
  nr_term_of_office = 8, windows = .Platform$OS.type == '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>nr_term_of_office</code>	number of term of office of Polish Diet; default: 8
<code>windows</code>	information of used operation system; default: <code>.Platform\$OS.type == 'windows'</code>

### 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,  
                      nr_term_of_office = 8)
```

**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>nr_term_of_office</code>	number of term of office of Polish Diet; default: 8

**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 = .Platform$OS.type == 'windows')
```

### 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: <code>.Platform\$OS.type == 'windows'</code>

### 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] 983 3
names(deputies)
# [1] 'id_deputy' 'nr_term_of_office' 'surname_name'
## End(Not run)
```

---

```
get_filtered_statements
```

*Retrieve filtered statements from a database*

---

**Description**

Function `get_filtered_statements` reads filtered statements from a database.

**Usage**

```
get_filtered_statements(dbname = 'sejmrp', user = 'reader',
  password = 'qux94874', host = 'services.mini.pw.edu.pl',
  windows = .Platform$OS.type == 'windows', terms_of_office = integer(0),
  deputies = character(0), dates = character(0), topics = character(0),
  content = character(0), max_rows = Inf)
```

**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>windows</code>	information of used operation system; default: <code>.Platform\$OS.type == 'windows'</code>
<code>terms_of_office</code>	range of terms of office's numbers that will be taken to filter data from database; default: <code>integer(0)</code>
<code>deputies</code>	full names of deputies that will be taken to filter data from database; default: <code>character(0)</code>
<code>dates</code>	period of time that will be taken to filter data from database; default: <code>character(0)</code>
<code>topics</code>	text patterns that will be taken to filter data from database; default: <code>character(0)</code>
<code>content</code>	text patterns that will be taken to filter data from database; default: <code>character(0)</code>
<code>max_rows</code>	maximum number of rows to download; default: <code>Inf</code>

**Details**

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

Possible filters:

1. `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.
2. `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.
3. `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.
4. `topics` - text patterns. This filter is a character vector with text patterns of topics in order points. Note that the order points 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 order points about education (in Polish: szkolnictwo) try 'szkolnictw'. It is possible to choose more than one pattern.
5. `content` - text patterns. This filter is a character vector with text patterns in statements. Note that strings with statements are sentences, so remember about case inflection of nouns and adjectives and use stems of words as patterns. For example if you want to find order points 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 ( $\leq$  ~150 MB) it may take few seconds / minutes to download all statements.

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)**

Tomasz Mikolajczyk, Piotr Smuda

**Examples**

```
## Not run:
filtered_statements <- get_filtered_statements()
dim(filtered_statements)
# [1] 2568      6
names(filtered_statements)
[1] 'id_statement' 'nr_term_of_office' 'surname_name' 'date_statement'
[5] 'titles_order_points' 'statement'
object.size(filtered_statements)
# 6488552 bytes
## 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 = .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), max_rows = Inf)
```

**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: <code>.Platform\$OS.type == 'windows'</code>
clubs	names of clubs that will be taken to filter data from database; default: <code>character(0)</code>
dates	period of time that will be taken to filter data from database; default: <code>character(0)</code>
terms_of_office	range of terms of office's numbers that will be taken to filter data from database; default: <code>integer(0)</code>
meetings	range of meetings' numbers that will be taken to filter data from database; default: <code>integer(0)</code>

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)
max_rows	maximum number of rows to download; default: Inf

### 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. 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.

If you did not choose any filter, the whole database will be downloaded. Note that, due to data size (<= ~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] 2826483      9
names(filtered_votes)
[1] 'surname_name' 'nr_term_of_office' 'club' 'vote' 'id_voting'
[6] 'nr_meeting' 'nr_voting' 'date_meeting' 'topic_voting'
object.size(filtered_votes)
# 148694336 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 = .Platform$OS.type == 'windows')
```

**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: .Platform\$OS.type == 'windows'

**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] 43432 6
names(statements)
# [1] 'id_statement' 'nr_term_of_office' 'surname_name'
# [4] 'date_statement' 'titles_order_points' 'statement'
## End(Not run)
```

---

`get_votes_table`*Importing votes table from a database*

---

**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 = .Platform$OS.type == 'windows')
```

**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: .Platform\$OS.type == 'windows'

**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] 2826483 6
names(votes)
# [1] 'id_vote' 'nr_term_of_office' 'id_deputy' 'id_voting' 'vote' 'club'
object.size(votes)
# 90474040 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 = .Platform$OS.type == 'windows')
```

### 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: <code>.Platform\$OS.type == 'windows'</code>

### 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] 6212 7
names(votings)
# [1] 'id_voting' 'nr_term_of_office' 'nr_meeting'
# [4] 'date_meeting' 'nr_voting' 'topic_voting'
# [7] '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)
```

---

safe_html	<i>Safe html scrapping</i>
-----------	----------------------------

---

**Description**

Function safe\_html tries to download the URL several times.

**Usage**

```
safe_html(page, time = 60, attempts = 10)
```

**Arguments**

page	requested URL
time	sleep interval after each failure
attempts	max number of tries (if there is a problem with connection)

**Details**

Function safe\_html performs 10 (by default) attempts to download the URL and waits 60sec (by default) after each failure

**Value**

character vector

**Author(s)**

Przemyslaw Biecek

**Examples**

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

safe\_readHTMLTable     *Safe html table scrapping*

---

### **Description**

Function safe\_readHTMLTable tries to download the table from given URL several times.

### **Usage**

```
safe_readHTMLTable(..., time = 60, attempts = 10)
```

### **Arguments**

...	arguments that will be passed to readHTMLTable
time	sleep interval after each failure
attempts	max number of tries (if there is a problem with connection)

### **Details**

Function safe\_readHTMLTable performs 10 (by default) attempts to download the URL and waits 60sec (by default) after each failure

### **Value**

character vector

### **Author(s)**

Przemyslaw Biecek

### **Examples**

```
## Not run:  
page <- paste0('http://www.sejm.gov.pl/Sejm7.nsf/',  
              'posiedzenie.xsp?posiedzenie=99&dzien=2')  
safe_readHTMLTable(page)  
## 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,  
nr_term_of_office = 8)
```

**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>nr_term_of_office</code>	number of term of office of Polish Diet; default: 8

**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
<code>...</code>	other arguments, that will be passed to <code>safe_html()</code>

**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/')
```

## Arguments

<code>statements_links</code>	list of elements of <code>XMLNodeSet</code> 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/">http://www.sejm.gov.pl/</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 <- safe_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\_get\_statements\_table

*Getting statements' table*

---

### **Description**

Function statements\_get\_statements\_table gets statements' table from meeting's page.

### **Usage**

```
statements_get_statements_table(page)
```

### **Arguments**

page                    meeting's page

### **Details**

Function statements\_get\_statements\_table gets statements' table. from meeting's page. Example of a meeting's page: <http://www.sejm.gov.pl/Sejm7.nsf/posiedzenie.xsp?posiedzenie=99&dzien=2>  
The result of this function is a data frame with three columns, where the first includes author of statement, the second the number of order point and the third is a title of order point.

### **Value**

data frame with three unnamed columns

### **Note**

All information is stored in PostgreSQL database.

### **Author(s)**

Piotr Smuda

### **Examples**

```
## Not run:  
page <- 'http://www.sejm.gov.pl/Sejm7.nsf/posiedzenie.xsp?posiedzenie=99&dzien=2'  
statements_get_statements_table(page)  
## 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,  
nr_term_of_office = 8, verbose = FALSE)
```

**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>nr_term_of_office</code>	number of term of office of Polish Diet; default: 8
<code>verbose</code>	if TRUE then additional info will be printed

**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,  
nr_term_of_office = 8, windows = .Platform$OS.type == 'windows')
```

### Arguments

dbname	name of database
user	name of user
password	password of database
host	name of host
nr_term_of_office	number of term of office of Polish Diet; default: 8
windows	information of used operation system; default: .Platform\$OS.type == 'windows'

### 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:  
votes_create_table(dbname, user, password, host, 7, TRUE)  
votes_create_table(dbname, user, password, host, 7, 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/Sejm8.nsf/',  
  page)
```

### Arguments

home_page	main page of polish diet: <a href="http://www.sejm.gov.pl/Sejm8.nsf/">http://www.sejm.gov.pl/Sejm8.nsf/</a>
page	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)
## 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,
  nr_term_of_office = 8, windows = .Platform$OS.type == 'windows')
```

**Arguments**

dbname	name of database
user	name of user
password	password of database
host	name of host
page	club's voting's results page
nr_term_of_office	number of term of office of Polish Diet; default: 8
windows	information of used operation system; default: .Platform\$OS.type == 'windows'

**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, 7, TRUE)
votes_match_deputies_ids(dbname, user, password, host, page, 7, 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,
                  nr_term_of_office = 8, windows = .Platform$OS.type == 'windows',
                  verbose = FALSE)
```

**Arguments**

dbname	name of database
user	name of user
password	password of database
host	name of host
nr_term_of_office	number of term of office of Polish Diet; default: 8
windows	information of used operation system; default: .Platform\$OS.type == 'windows'
verbose	if TRUE then additional info will be printed

**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:  
votes_update_table(dbname, user, password, host, 7, TRUE)  
votes_update_table(dbname, user, password, host, 7, 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,  
nr_term_of_office = 8)
```

**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>nr_term_of_office</code>	number of term of office of Polish Diet; default: 8

**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  
*Getting meetings' links*

---

**Description**

Function `votings_get_meetings_links` gets meetings' links.

**Usage**

```
votings_get_meetings_links(  
  home_page = 'http://www.sejm.gov.pl/Sejm8.nsf/', page =  
  'http://www.sejm.gov.pl/Sejm8.nsf/agent.xsp?symbol=posglos&NrKadencji=8')
```

**Arguments**

<code>home_page</code>	main page of polish diet: <a href="http://www.sejm.gov.pl/Sejm8.nsf/">http://www.sejm.gov.pl/Sejm8.nsf/</a>
<code>page</code>	page with votings in polish diet: <a href="http://www.sejm.gov.pl/Sejm8.nsf/agent.xsp?symbol=posglos&amp;NrKadencji=8">http://www.sejm.gov.pl/Sejm8.nsf/agent.xsp?symbol=posglos&amp;NrKadencji=8</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/Sejm8.nsf/agent.xsp?symbol=posglos&NrKadencji=8')
```

**Arguments**

<code>page</code>	page with votings in polish diet: <a href="http://www.sejm.gov.pl/Sejm8.nsf/agent.xsp?symbol=posglos&amp;NrKadencji=8">http://www.sejm.gov.pl/Sejm8.nsf/agent.xsp?symbol=posglos&amp;NrKadencji=8</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/Sejm8.nsf/',  
page)
```

**Arguments**

<code>home_page</code>	main page of polish diet: <a href="http://www.sejm.gov.pl/Sejm8.nsf/">http://www.sejm.gov.pl/Sejm8.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,  
nr_term_of_office = 8, verbose = FALSE)
```

**Arguments**

dbname	name of database
user	name of user
password	password of database
host	name of host
nr_term_of_office	number of term of office of Polish Diet; default: 8
verbose	if TRUE then additional info will be printed

**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)
```

# Index

[create\\_database](#), 2

[deputies\\_add\\_new](#), 4  
[deputies\\_create\\_table](#), 5  
[deputies\\_get\\_data](#), 6  
[deputies\\_get\\_ids](#), 7  
[deputies\\_update\\_table](#), 8

[get\\_deputies\\_table](#), 9  
[get\\_filtered\\_statements](#), 10  
[get\\_filtered\\_votes](#), 12  
[get\\_statements\\_table](#), 14  
[get\\_votes\\_table](#), 15  
[get\\_votings\\_table](#), 17

[remove\\_database](#), 18

[safe\\_html](#), 19  
[safe\\_readHTMLTable](#), 20  
[statements\\_create\\_table](#), 21  
[statements\\_get\\_statement](#), 22  
[statements\\_get\\_statements\\_data](#), 23  
[statements\\_get\\_statements\\_table](#), 24  
[statements\\_update\\_table](#), 25

[votes\\_create\\_table](#), 26  
[votes\\_get\\_clubs\\_links](#), 27  
[votes\\_get\\_results](#), 28  
[votes\\_match\\_deputies\\_ids](#), 29  
[votes\\_update\\_table](#), 30  
[votings\\_create\\_table](#), 31  
[votings\\_get\\_date](#), 32  
[votings\\_get\\_meetings\\_links](#), 33  
[votings\\_get\\_meetings\\_table](#), 34  
[votings\\_get\\_votings\\_links](#), 35  
[votings\\_get\\_votings\\_table](#), 36  
[votings\\_update\\_table](#), 37