Package ‘dbparser’

June 8, 2020

Title 'DrugBank' Database XML Parser

Version 1.1.2

Description This tool is for parsing the 'DrugBank' XML database <http://drugbank.ca/>. The parsed data are then returned in a proper 'R' dataframe with the ability to save them in a given database.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports DBI, dplyr, odbc, purrr, readr, RMariaDB, RSQLite, tibble, tools, XML

RoxygenNote 7.1.0

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

URL `https://docs.ropensci.org/dbparser`, `https://github.com/ropensci/dbparser`

BugReports `https://github.com/ropensci/dbparser/issues`

NeedsCompilation no

Author Mohammed Ali [aut, cre],
Ali Ezzat [aut],
Hao Zhu [rev],
Emma Mendelsohn [rev]

Maintainer Mohammed Ali `<moh_fcis@yahoo.com>`

Repository CRAN

Date/Publication 2020-06-08 17:00:02 UTC

R topics documented:

carriers 3
carriers_actions 5

carriers_articles 7
carriers_links 9
carriers_polypeptide 10
carriers_polypeptides_go 12
carriers_polypeptides_pfams 14
carriers_polypeptides_syn 16
carriers_polypeptide_ext_id 18
carriers_textbooks 20
dbparser 21
drug 22
drug_affected_organisms 24
drug_ahfs_codes 26
drug_all 27
drug_articles 29
drug_atc_codes 31
drug_books 33
drug_calc_prop 34
drug_categories 36
drug_classification 38
drug_dosages 40
drug_element 41
drug_element_options 43
drug_exp_prop 44
drug_external_links 45
drug_ex_identity 47
drug_food_interactions 49
drug_groups 51
drug_interactions 53
drug_intern_brand 54
drug_links 56
drug_manufacturers 58
drug_mixtures 60
drug_packagers 61
drug_patents 63
drug_pathway 65
drug_pathway_drugs 67
drug_pathway_enzyme 68
drug_pdb_entries 70
drug_prices 72
drug_products 73
drug_reactions 75
drug_reactions_enzymes 77
drug_salts 78
drug_sequences 80
drug_snp_adverse_reactions 82
drug_snp_effects 84
drug_syn 85
enzymes 87

enzymes_actions 89
enzymes_articles 91
enzymes_links 92
enzymes_polypeptide 94
enzymes_polypeptide_ext_ident 96
enzymes_polypeptide_go 97
enzymes_polypeptide_pfams 99
enzymes_polypeptide_syn 101
enzymes_textbooks 103
get_drugbank_exported_date 105
get_drugbank_metadata 105
get_drugbank_version 106
read_drugbank_xml_db 106
targets 107
targets_actions 108
targets_articles 110
targets_links 112
targets_polypeptide 113
targets_polypeptide_ext_ident 115
targets_polypeptide_go 117
targets_polypeptide_pfams 119
targets_polypeptide_syn 121
targets_textbooks 123
transporters 124
transporters_actions 126
transporters_articles 128
transporters_links 129
transporters_polypeptide 131
transporters_polypeptide_go 133
transporters_polypeptide_pfams 135
transporters_polypeptide_syn 137
transporters_polypep_ex_ident 139
transporters_textbooks 140

Index **143**

`carriers` *Extracts the drug carriers element and return data as tibble.*

Description

`carriers` returns tibble of drug carriers elements.

Usage

```
carriers(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the carriers element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug carriers node attributes date frame

See Also

Other carriers: `carriers_actions()`, `carriers_articles()`, `carriers_links()`, `carriers_polypeptide_ext_id()`, `carriers_polypeptides_go()`, `carriers_polypeptides_pfams()`, `carriers_polypeptides_syn()`, `carriers_polypeptide()`, `carriers_textbooks()`

Examples

```
## Not run:
# return only the parsed tibble
carriers()

# will throw an error, as database_connection is NULL
carriers(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
```

```
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
carriers(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
carriers(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`carriers_actions` *Extracts the drug carriers actions element and return data as tibble.*

Description

`carriers_actions` returns tibble of drug carriers actions elements.

Usage

```
carriers_actions(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true

`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the carriers actions element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug carriers actions node attributes date frame

See Also

Other carriers: `carriers_articles()`, `carriers_links()`, `carriers_polypeptide_ext_id()`, `carriers_polypeptides_go()`, `carriers_polypeptides_pfams()`, `carriers_polypeptides_syn()`, `carriers_polypeptide()`, `carriers_textbooks()`, `carriers()`

Examples

```
## Not run:
# return only the parsed tibble
carriers_actions()

# will throw an error, as database_connection is NULL
carriers_actions(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
carriers_actions(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current location and
# return parsed tibble.
# If the csv exist before read it and return its data.
carriers_actions(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_actions(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location and
# return parsed tibble.
```

```
# If the csv exist before read it and return its data.
carriers_actions(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current location and
# return parsed tibble.
# If the csv exist override it and return it.

carriers_actions(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`carriers_articles` *Extracts the drug carriers articles element and return data as tibble.*

Description

`carriers_articles` returns tibble of drug carriers articles elements.

Usage

```
carriers_articles(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the carriers articles element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug carriers_articles node attributes date frame

See Also

Other carriers: `carriers_actions()`, `carriers_links()`, `carriers_polypeptide_ext_id()`, `carriers_polypeptides_go()`, `carriers_polypeptides_pfams()`, `carriers_polypeptides_syn()`, `carriers_polypeptide()`, `carriers_textbooks()`, `carriers()`

Examples

```
## Not run:
# return only the parsed tibble
carriers_articles()

# will throw an error, as database_connection is NULL
carriers_articles(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
carriers_articles(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_articles(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
# current location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_articles(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_articles(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
carriers_articles(
  save_csv = TRUE, csv_path = TRUE,
```

```
  override = TRUE
)

## End(Not run)
```

`carriers_links` *Extracts the drug carriers links element and return data as tibble.*

Description

`drug_groups` returns tibble of drug carriers links elements.

Usage

```
carriers_links(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the carriers links element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug carriers_links node attributes date frame

See Also

Other carriers: `carriers_actions()`, `carriers_articles()`, `carriers_polypeptide_ext_id()`, `carriers_polypeptides_go()`, `carriers_polypeptides_pfams()`, `carriers_polypeptides_syn()`, `carriers_polypeptide()`, `carriers_textbooks()`, `carriers()`

Examples

```
## Not run:
# return only the parsed tibble
carriers_links()

# will throw an error, as database_connection is NULL
carriers_links(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
carriers_links(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_links(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_links(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location and
# return parsed tibble.
# If the csv exist before read it and return its data.
carriers_links(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current location and
# return parsed tibble.
# If the csv exist override it and return it.
carriers_links(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`carriers_polypeptide` *Extracts the drug carriers polypeptides element and return data as tibble.*

Description

`carriers_polypeptides` returns tibble of drug carriers polypeptides elements.

Usage

```
carriers_polypeptide(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the carriers polypeptides element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug carriers polypeptides node attributes date frame

See Also

Other carriers: `carriers_actions()`, `carriers_articles()`, `carriers_links()`, `carriers_polypeptide_ext_id()`, `carriers_polypeptides_go()`, `carriers_polypeptides_pfams()`, `carriers_polypeptides_syn()`, `carriers_textbooks()`, `carriers()`

Examples

```
## Not run:
# return only the parsed tibble
carriers_polypeptide()

# will throw an error, as database_connection is NULL
carriers_polypeptide(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
```

```
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
carriers_polypeptide(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_polypeptide(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
# current location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_polypeptide(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_polypeptide(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
carriers_polypeptide(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`carriers_polypeptides_go` *Extracts the drug carriers polypeptides go classifiers element and return data as tibble.*

Description

`carriers_polypeptides_go` returns tibble of drug carriers polypeptides go classifiers elements.

Usage

```
carriers_polypeptides_go(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the carriers polypeptides go classifiers element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug carriers polypeptides go classifiers node attributes date frame

See Also

Other carriers: `carriers_actions`(), `carriers_articles`(), `carriers_links`(), `carriers_polypeptide_ext_id`(), `carriers_polypeptides_pfams`(), `carriers_polypeptides_syn`(), `carriers_polypeptide`(), `carriers_textbooks`(), `carriers`()

Examples

```
## Not run:
# return only the parsed tibble
carriers_polypeptides_go()

# will throw an error, as database_connection is NULL
carriers_polypeptides_go(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
carriers_polypeptides_go(save_table = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_polypeptides_go(save_csv = TRUE)
```

```
# save in database, save parsed tibble as csv if it does not exist in
current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_polypeptides_go(
  save_table = TRUE,
  save_csv = TRUE
)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_polypeptides_go(
  save_csv = TRUE,
  csv_path = TRUE
)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
carriers_polypeptides_go(
  save_csv = TRUE,
  csv_path = TRUE, override = TRUE
)

## End(Not run)
```

`carriers_polypeptides_pfams` *Extracts the drug carriers polypeptides pfams element and return data as tibble.*

Description

`carriers_polypeptides_pfams` returns tibble of drug carriers polypeptides pfams elements.

Usage

```
carriers_polypeptides_pfams(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.

`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the carriers polypeptides pfams element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug carriers polypeptides pfams node attributes date frame

See Also

Other carriers: `carriers_actions()`, `carriers_articles()`, `carriers_links()`, `carriers_polypeptide_ext_id()`, `carriers_polypeptides_go()`, `carriers_polypeptides_syn()`, `carriers_polypeptide()`, `carriers_textbooks()`, `carriers()`

Examples

```
## Not run:
# return only the parsed tibble
carriers_polypeptides_pfams()

# will throw an error, as database_connection is NULL
carriers_polypeptides_pfams(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
carriers_polypeptides_pfams(save_table = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_polypeptides_pfams(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
#  current location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_polypeptides_pfams(save_table = TRUE, save_csv = TRUE,
```

```
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_polypeptides_pfams(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
carriers_polypeptides_pfams(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`carriers_polypeptides_syn` *Extracts the drug carriers polypeptides syn element and return data as tibble.*

Description

`carriers_polypeptides_syn` returns tibble of drug carriers polypeptides syn elements.

Usage

```
carriers_polypeptides_syn(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the carriers polypeptides syn element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug carriers polypeptides syn node attributes date frame

See Also

Other carriers: `carriers_actions()`, `carriers_articles()`, `carriers_links()`, `carriers_polypeptide_ext_id()`, `carriers_polypeptides_go()`, `carriers_polypeptides_pfams()`, `carriers_polypeptide()`, `carriers_textbooks()`, `carriers()`

Examples

```
## Not run:
# return only the parsed tibble
carriers_polypeptides_syn()

# will throw an error, as database_connection is NULL
carriers_polypeptides_syn(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
carriers_polypeptides_syn(save_table = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_polypeptides_syn(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_polypeptides_syn(
  save_table = TRUE,
  save_csv = TRUE
)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_polypeptides_syn(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
```

```
# If the csv exist override it and return it.
carriers_polypeptides_syn(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

---

carriers_polypeptide_ext_id

Extracts the drug carriers polypeptides external identifiers element and return data as tibble.

Description

`carriers_polypeptide_ext_id` returns tibble of drug carriers polypeptides external identifiers elements.

Usage

```
carriers_polypeptide_ext_id(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the carriers polypeptides external identifiers element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug carriers polypeptides external identifiers node attributes date frame

See Also

Other carriers: `carriers_actions()`, `carriers_articles()`, `carriers_links()`, `carriers_polypeptides_go()`, `carriers_polypeptides_pfams()`, `carriers_polypeptides_syn()`, `carriers_polypeptide()`, `carriers_textbooks()`, `carriers()`

Examples

```
## Not run:
# return only the parsed tibble
carriers_polypeptide_ext_id()

# will throw an error, as database_connection is NULL
carriers_polypeptide_ext_id(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
carriers_polypeptide_ext_id(save_table = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_polypeptide_ext_id(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_polypeptide_ext_id(
  save_table = TRUE,
  save_csv = TRUE
)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_polypeptide_ext_id(
  save_csv = TRUE,
  csv_path = TRUE
)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
carriers_polypeptide_ext_id(
  save_csv = TRUE, csv_path = TRUE, override = TRUE
)

## End(Not run)
```

`carriers_textbooks` *Extracts the drug carriers textbooks element and return data as tibble.*

Description

`carriers_textbooks` returns tibble of drug carriers textbooks elements.

Usage

```
carriers_textbooks(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the carriers textbooks element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug carriers textbooks node attributes date frame

See Also

Other carriers: `carriers_actions()`, `carriers_articles()`, `carriers_links()`, `carriers_polypeptide_ext_id()`, `carriers_polypeptides_go()`, `carriers_polypeptides_pfams()`, `carriers_polypeptides_syn()`, `carriers_polypeptide()`, `carriers()`

Examples

```
## Not run:
# return only the parsed tibble
carriers_textbooks()

# will throw an error, as database_connection is NULL
carriers_textbooks(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
carriers_textbooks(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_textbooks(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
# current location and return parsed tibble.
# If the csv exist before read it and return its data.
carriers_textbooks(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location and
# return parsed tibble.
# If the csv exist before read it and return its data.
carriers_textbooks(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current location and
# return parsed tibble.
# If the csv exist override it and return it.
carriers_textbooks(
  save_csv = TRUE, csv_path = TRUE, override =
    TRUE
)

## End(Not run)
```

`dbparser` *dbparser: A package for reading and parsing* **DrugBank** *xml database.*

Description

The main purpose of the ‘dbparser‘ package is to parse [DrugBank](http://DrugBank.ca/) database which is downloadable in XML format from [this link](https://www.DrugBank.ca/releases/latest).

Details

The parsed data can then be explored and analyzed as desired by the user with the ability to save parsed data into desired database as well.

To achieve this purpose, ‘dbparser“ package provides three main categories of functions:

- xml db reader,

- **DrugBank** elements parsers,

- and database related methods.

For more information kindly check the reference/index (https://docs.ropensci.org/dbparser/reference/index.html)

xml db reader functions

Reads **DrugBank** xml database and build drug elements full tree in memory

parsers functions

Each parser function is responsible of parsing certain drug element and returning its tibble with the ability to save it in a predefined database.

Check this tutorial (https://docs.ropensci.org/dbparser/articles/dbparser.html)

database functions

To open a connection to given database in order to store parsed **DrugBank** elements database.

Check this tutorial (https://docs.ropensci.org/dbparser/articles/Database_Saving.html)

`drug` *Extracts the main drug elements and return data as tibble.*

Description

`drug` returns tibble of drugs main elements.

Usage

```
drug(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the main element of drug node in drugbank xml database with the option to save it in a user defined database. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug main node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`

Examples

```
## Not run:
# return only the parsed tibble
drug()

# will throw an error, as database_connection is NULL
drug(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug(save_csv = TRUE)
```

```
# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug(save_table = TRUE, save_csv = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
#  and return parsed tibble.
# If the csv exist before read it and return its data.
drug(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_affected_organisms` *Extracts the drug affected organisms element and return data as tibble.*

Description

`drug_affected_organisms` returns tibble of drug affected organisms elements.

Usage

```
drug_affected_organisms(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the affected organisms element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug affected organisms node attributes tibble

See Also

Other drugs: `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_affected_organisms()

# will throw an error, as database_connection is NULL
drug_affected_organisms(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_affected_organisms(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_affected_organisms(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_affected_organisms(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_affected_organisms(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
```

```
# If the csv exist override it and return it.
drug_affected_organisms(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`drug_ahfs_codes` *Extracts the drug ahfs codes element and return data as tibble.*

Description

`drug_ahfs_codes` returns tibble of drug ahfs codes elements.

Usage

```
drug_ahfs_codes(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the ahfs codes element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug ahfs codes node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_ahfs_codes()

# will throw an error, as database_connection is NULL
drug_ahfs_codes(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_ahfs_codes(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_ahfs_codes(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
# current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_ahfs_codes(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location and
# return parsed tibble.
# If the csv exist before read it and return its data.
drug_ahfs_codes(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist override it and return it.
drug_ahfs_codes(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_all` *extracts the all drug elements and return data as list of tibbles.*

Description

this functions extracts all element of drug nodes in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. it takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. it must be called after `read_drugbank_xml_db` function like any other parser function. if `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Usage

```
drug_all(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Value

all drug elements tibbles

See Also

Other common: `drug_element_options()`, `drug_element()`

Examples

```
## Not run:
# return only the parsed tibble
drug_all()

# will throw an error, as database_connection is NULL
drug_all(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_all(save_table = TRUE, database_connection = sqlite_con)
```

```
# save parsed tibble as csv if it does not exist in current location,
# and return parsed tibble.
# if the csv exist before read it and return its data.
drug_all(save_csv = TRUE)

# save in database, save parsed tibble as csv,
# if it does not exist in current location and return parsed tibble.
# if the csv exist before read it and return its data.
drug_all(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location,
# and return parsed tibble.
# if the csv exist before read it and return its data.
drug_all(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current location and
# return parsed tibble.
# if the csv exist override it and return it.
drug_all(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_articles` *Extracts the drug articles element and return data as tibble.*

Description

`drug_articles` returns tibble of drug articles elements.

Usage

```
drug_articles(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation

`database_connection`
DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the articles element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug articles node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_articles()

# will throw an error, as database_connection is NULL
drug_articles(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_articles(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_articles(save_csv = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_articles(save_table = TRUE, database_connection = sqlite_con)
# save in database, save parsed tibble as csv if it
# does not exist in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_articles(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)
```

```
# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_articles(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_articles(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_atc_codes` *Extracts the drug atc codes element and return data as tibble.*

Description

`drug_atc_codes` returns tibble of drug atc codes elements.

Usage

```
drug_atc_codes(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the atc codes element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug atc_codes node attributes date frame

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_atc_codes()

# will throw an error, as database_connection is NULL
drug_atc_codes(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_atc_codes(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current location and
# return parsed tibble.
# If the csv exist before read it and return its data.
drug_atc_codes(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
current
#  location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_atc_codes(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location and
# return parsed tibble.
# If the csv exist before read it and return its data.
drug_atc_codes(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current location and
# return parsed tibble.
# If the csv exist override it and return it.
drug_atc_codes(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_books` *Extracts the drug books element and return data as tibble.*

Description

`drug_books` returns tibble of drug books elements.

Usage

```
drug_books(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the books element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug books node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, drug_intern_brand(), drug_links(), drug_manufacturers(), drug_mixtures(), drug_packagers(), drug_patents(), drug_pdb_entries(), drug_products(), drug_salts(), drug_snp_adverse_reactions(), drug_snp_effects(), drug_syn(), drug()

Examples

```
## Not run:
# return only the parsed tibble
drug_books()

# will throw an error, as database_connection is NULL
drug_books(save_table = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_books(save_csv = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_books(save_table = TRUE, database_connection = sqlite_con)

# save in database, save parsed tibble as csv if it does not
# exist in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_books(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given l
# ocation and return parsed tibble.
# If the csv exist before read it and return its data.
drug_books(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_books(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

---

drug_calc_prop *Extracts the drug calculated properties element and return data as tibble.*

Description

`drug_calc_prop` returns tibble of drug calculated properties elements.

Usage

```
drug_calc_prop(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the calculated properties element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug calculated properties node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_calc_prop()
```

```
# will throw an error, as database_connection is NULL
drug_calc_prop(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_calc_prop(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_calc_prop(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_calc_prop(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_calc_prop(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_calc_prop(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`drug_categories` *Extracts the drug categories element and return data as tibble.*

Description

`drug_categories` returns tibble of drug categories elements.

Usage

```
drug_categories(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the categories element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug categories node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_categories()

# will throw an error, as database_connection is NULL
drug_categories(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_categories(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_categories(save_csv = TRUE)
```

```
# save in database, save parsed tibble as csv if it does not exist
#  in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_categories(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_categories(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_categories(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_classification` *Extracts the drug classifications element and return data as tibble.*

Description

`drug_classification` returns tibble of drug classifications elements.

Usage

```
drug_classification(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the classifications element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug classifications node attributes date frame

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_classification()

# will throw an error, as database_connection is NULL
drug_classification(save_table = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_classification(save_csv = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_classification(save_table = TRUE, database_connection = sqlite_con)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_classification(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
#  and return parsed tibble.
# If the csv exist before read it and return its data.
drug_classification(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
```

```
# If the csv exist override it and return it.
drug_classification(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_dosages` *Extracts the drug dosages element and return data as tibble.*

Description

`drug_dosages` returns tibble of drug dosages elements.

Usage

```
drug_dosages(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the dosages element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug dosages node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_dosages()

# will throw an error, as database_connection is NULL
drug_dosages(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_dosages(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_dosages(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_dosages(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
#  and return parsed tibble.
# If the csv exist before read it and return its data.
drug_dosages(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_dosages(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_element` *extracts the given drug elements and return data as list of tibbles.*

Description

`drug_element` returns list of tibbles of drugs selected elements.

Usage

```
drug_element(
  elements_options = c("all"),
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`elements_options`	list, options of elements to be parsed. default is "all"
`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown. @return list of selected drug elements tibbles

Details

this functions extracts selected element of drug nodes in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. it takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. it must be called after `read_drugbank_xml_db` function like any other parser function. if `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

drug_element_options can be called to know the valid options for this method

See Also

Other common: `drug_all()`, `drug_element_options()`

Examples

```
## Not run:
# return only the parsed tibble
drug_element()

# will throw an error, as database_connection is NULL
drug_element(save_table = TRUE)

# save parsed tibble as csv if it does not exist in current location and
# return parsed tibble.
```

```
# if the csv exist before read it and return its data.
drug_element(save_csv = TRUE)


sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_element(save_table = TRUE, database_connection = sqlite_con)

# save in database, save parsed tibble as csv if it does not
# exist in current location and return parsed tibble.
# if the csv exist before read it and return its data.
drug_element(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location and
# return parsed tibble.
# if the csv exist before read it and return its data.
drug_element(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# if the csv exist override it and return it.
drug_element(save_csv = TRUE, csv_path = TRUE, override = TRUE)
drug_element(c("drug_ahfs_codes", "drug_carriers"), save_table = TRUE)
drug_element(save_table = FALSE)
drug_element(c("drug_ahfs_codes", "drug_carriers"))

## End(Not run)
```

`drug_element_options` *returns* `drug_element` *valid options.*

Description

returns `drug_element` valid options.

Usage

```
drug_element_options()
```

Value

list of `drug_element` valid options

See Also

Other common: `drug_all()`, `drug_element()`

Examples

```
## Not run:
drug_element_options()

## End(Not run)
```

`drug_exp_prop` *Extracts the drug experimental properties element and return data as tibble.*

Description

`drug_exp_prop` returns tibble of drug experimental properties elements.

Usage

```
drug_exp_prop(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the experimental properties element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug experimental properties node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_exp_prop()

# will throw an error, as database_connection is NULL
drug_exp_prop(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_exp_prop(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_exp_prop(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_exp_prop(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_exp_prop(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_exp_prop(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`drug_external_links` *Extracts the drug external links element and return data as tibble.*

Description

`drug_external_links` returns tibble of drug external links elements.

Usage

```
drug_external_links(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the external links element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug external links node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_external_links()

# will throw an error, as database_connection is NULL
drug_external_links(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_external_links(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current location and
# return parsed tibble.
# If the csv exist before read it and return its data.
drug_external_links(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
# current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_external_links(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_external_links(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist override it and return it.
drug_external_links(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_ex_identity` *Extracts the drug external identifiers element and return data as tibble.*

Description

`drug_ex_identity` returns tibble of external identifiers groups elements.

Usage

```
drug_ex_identity(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
```

```
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the external identifiers element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug external identifiers node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_ex_identity()

# will throw an error, as database_connection is NULL
drug_ex_identity(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_ex_identity(save_table = TRUE, database_connection = sqlite_con)
```

```
# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_ex_identity(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
# current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_ex_identity(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location and
# return parsed tibble.
# If the csv exist before read it and return its data.
drug_ex_identity(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current location and
# return parsed tibble.
# If the csv exist override it and return it.
drug_ex_identity(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`drug_food_interactions` *Extracts the drug food interactions element and return data as tibble.*

Description

`drug_food_interactions` returns tibble of drug food interactions elements.

Usage

```
drug_food_interactions(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true

`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the food interactions element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug food interactions node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_food_interactions()

# will throw an error, as database_connection is NULL
drug_food_interactions(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_food_interactions(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_food_interactions(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_food_interactions(save_table = TRUE, save_csv = TRUE,
```

```
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_food_interactions(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current location
#  and return parsed tibble.
# If the csv exist override it and return it.
drug_food_interactions(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`drug_groups` *Extracts the drug groups element and return data as tibble.*

Description

`drug_groups` returns tibble of drug groups elements.

Usage

```
drug_groups(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the groups element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug groups node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_groups()

# will throw an error, as database_connection is NULL
drug_groups(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_groups(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_groups(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_groups(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_groups(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
```

```
# If the csv exist override it and return it.
drug_groups(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_interactions` *Extracts the drug interactions element and return data as tibble.*

Description

`drug_interactions` returns tibble of drug interactions elements.

Usage

```
drug_interactions(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the interactions element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug interactions node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_interactions()

# will throw an error, as database_connection is NULL
drug_interactions(save_table = TRUE)


# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_interactions(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_interactions(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_interactions(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_interactions(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_interactions(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_intern_brand` *Extracts the drug international brands and return data as tibble.*

Description

`drug_intern_brand` returns tibble of drug products elements.

Usage

```
drug_intern_brand(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the international brands element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug international brands node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_intern_brand()

# will throw an error, as database_connection is NULL
drug_intern_brand(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_intern_brand(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_intern_brand(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
#  current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_intern_brand(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_intern_brand(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_intern_brand(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`drug_links` *Extracts the drug links element and return data as tibble.*

Description

`drug_links` returns tibble of drug links elements.

Usage

```
drug_links(
  save_table = FALSE,
```

```
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the links element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug links node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_links()

# will throw an error, as database_connection is NULL
drug_links(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
```

```
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_links(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_links(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not
# exist in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_links(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_links(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_links(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_manufacturers` *Extracts the drug manufacturers element and return data as tibble.*

Description

`drug_manufacturers` returns tibble of drug manufacturers elements.

Usage

```
drug_manufacturers(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true

`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the manufacturers element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug manufacturers node attributes date frame

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_manufacturers()

# will throw an error, as database_connection is NULL
drug_manufacturers(save_table = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_manufacturers(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_manufacturers(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_manufacturers(save_csv = TRUE, csv_path = TRUE)
```

```
# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_manufacturers(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_mixtures` *Extracts the drug mixtures element and return data as tibble.*

Description

`drug_mixtures` returns tibble of drug mixtures elements.

Usage

```
drug_mixtures(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the mixtures element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug mixtures node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_mixtures()

# will throw an error, as database_connection is NULL
drug_mixtures(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_mixtures(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_mixtures(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
# current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_mixtures(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_mixtures(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist override it and return it.
drug_mixtures(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_packagers` *Extracts the drug packagers element and return data as tibble.*

Description

`drug_packagers` returns tibble of drug packagers elements.

Usage

```
drug_packagers(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the packagers element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug packagers node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_packagers()
```

```
# will throw an error, as database_connection is NULL
drug_packagers(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_packagers(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_packagers(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not
# exist in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_packagers(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_packagers(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_packagers(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_patents` *Extracts the drug patents element and return data as tibble.*

Description

`drug_patents` returns tibble of drug patents elements.

Usage

```
drug_patents(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the patents element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug patents node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_patents()

# will throw an error, as database_connection is NULL
drug_patents(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_patents(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_patents(save_csv = TRUE)
```

```
# save in database, save parsed tibble as csv if it does not exist
# 'in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_patents(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_patents(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_patents(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_pathway` *Extracts the drug pathway element and return data as tibble.*

Description

`drug_pathway` returns tibble of drug pathway elements.

Usage

```
drug_pathway(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the groups element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug pathway node attributes date frame

Examples

```
## Not run:
# return only the parsed tibble
drug_pathway()

# will throw an error, as database_connection is NULL
drug_pathway(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_pathway(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_pathway(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not
#  exist in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_pathway(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_pathway(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_pathway(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_pathway_drugs` *Extracts the drug pathway drugs element and return data as tibble.*

Description

`drug_pathway_drugs` returns tibble of drug pathway drugs elements.

Usage

```
drug_pathway_drugs(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the pathway drugs element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug pathway drugs node attributes date frame

Examples

```
## Not run:
# return only the parsed tibble
drug_pathway_drugs()
```

```
# will throw an error, as database_connection is NULL
drug_pathway_drugs(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_pathway_drugs(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_pathway_drugs(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_pathway_drugs(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_pathway_drugs(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_pathway_drugs(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_pathway_enzyme` *Extracts the drug pathway enzyme element and return data as tibble.*

Description

`drug_pathway_enzyme` returns tibble of drug pathway enzyme elements.

Usage

```
drug_pathway_enzyme(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the pathway enzyme element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug pathway enzyme node attributes date frame

Examples

```
## Not run:
# return only the parsed tibble
drug_pathway_enzyme()

# will throw an error, as database_connection is NULL
drug_pathway_enzyme(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_pathway_enzyme(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_pathway_enzyme(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
# current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_pathway_enzyme(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
```

```
drug_pathway_enzyme(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist override it and return it.
drug_pathway_enzyme(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_pdb_entries` *Extracts the drug pdb entries element and return data as tibble.*

Description

`drug_pdb_entries` returns tibble of drug pdb entries elements.

Usage

```
drug_pdb_entries(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the pdb entries element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug pdb entries node attributes tibble

See Also

Other drugs: `drug_affected_organisms`(), `drug_ahfs_codes`(), `drug_articles`(), `drug_atc_codes`(), `drug_books`(), `drug_calc_prop`(), `drug_categories`(), `drug_classification`(), `drug_dosages`(), `drug_ex_identity`(), `drug_exp_prop`(), `drug_external_links`(), `drug_food_interactions`(), `drug_groups`(), `drug_interactions`(), `drug_intern_brand`(), `drug_links`(), `drug_manufacturers`(), `drug_mixtures`(), `drug_packagers`(), `drug_patents`(), `drug_products`(), `drug_salts`(), `drug_snp_adverse_reacti`
`drug_snp_effects`(), `drug_syn`(), `drug`()

Examples

```
## Not run:
# return only the parsed tibble
drug_pdb_entries()

# will throw an error, as database_connection is NULL
drug_pdb_entries(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_pdb_entries(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_pdb_entries(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_pdb_entries(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_pdb_entries(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_pdb_entries(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_prices` *Extracts the drug prices element and return data as tibble.*

Description

`drug_prices` returns tibble of drug prices elements.

Usage

```
drug_prices(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the prices element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug prices node attributes date frame

Examples

```
## Not run:
# return only the parsed tibble
drug_prices()
```

```
# will throw an error, as database_connection is NULL
drug_prices(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_prices(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_prices(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_prices(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_prices(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_prices(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_products` *Extracts the drug products element and return data as tibble.*

Description

`drug_products` returns tibble of drug products elements.

Usage

```
drug_products(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the products element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug products node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_products()

# will throw an error, as database_connection is NULL
drug_products(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_products(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_products(save_csv = TRUE)
```

```
# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_products(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_products(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_products(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_reactions` *Extracts the drug reactions element and return data as tibble.*

Description

`drug_reactions` returns tibble of drug reactions elements.

Usage

```
drug_reactions(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the groups element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug reactions node attributes date frame

Examples

```
## Not run:
# return only the parsed tibble
drug_reactions()

# will throw an error, as database_connection is NULL
drug_reactions(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_reactions(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_reactions(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_reactions(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_reactions(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_reactions(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_reactions_enzymes` *Extracts the drug reactions enzymes element and return data as tibble.*

Description

`drug_reactions_enzymes` returns tibble of drug reactions enzymes elements.

Usage

```
drug_reactions_enzymes(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the reactions enzymes element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug reactions enzymes node attributes date frame

Examples

```
## Not run:
# return only the parsed tibble
drug_reactions_enzymes()

# will throw an error, as database_connection is NULL
drug_reactions_enzymes(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_reactions_enzymes(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in
# current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_reactions_enzymes(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not
# exist in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_reactions_enzymes(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in
#  given location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_reactions_enzymes(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist i
# n current location and return parsed tibble.
# If the csv exist override it and return it.
drug_reactions_enzymes(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`drug_salts` *Extracts the drug salts and return data as tibble.*

Description

`drug_salts` returns tibble of drug products elements.

Usage

```
drug_salts(
  save_table = FALSE,
```

```
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in t he new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the salts element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug salts node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_salts()

# will throw an error, as database_connection is NULL
drug_salts(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
```

```
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_salts(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_salts(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
# current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_salts(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_salts(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_salts(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`drug_sequences` *Extracts the drug sequences element and return data as tibble.*

Description

`drug_sequences` returns tibble of drug sequences elements.

Usage

```
drug_sequences(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true

`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the sequences element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug sequences node attributes date frame

Examples

```
## Not run:
# return only the parsed tibble
drug_sequences()

# will throw an error, as database_connection is NULL
drug_sequences(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_sequences(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_sequences(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_sequences(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_sequences(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_sequences(save_csv = TRUE, csv_path = TRUE, override = TRUE)
```

```
## End(Not run)
```

`drug_snp_adverse_reactions` *Extracts the drug snp adverse drug reactions element and return data as tibble.*

Description

`drug_snp_adverse_reactions` returns tibble of drug snp adverse drug reactions elements.

Usage

```
drug_snp_adverse_reactions(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the groups element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug snp adverse drug reactions node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_effects()`, `drug_syn()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_snp_adverse_reactions()

# will throw an error, as database_connection is NULL
drug_snp_adverse_reactions(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_snp_adverse_reactions(save_table = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_snp_adverse_reactions(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
# current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_snp_adverse_reactions(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_snp_adverse_reactions(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist override it and return it.
drug_snp_adverse_reactions(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`drug_snp_effects` *Extracts the drug snp effects element and return data as tibble.*

Description

`drug_snp_effects` returns tibble of snp effects groups elements.

Usage

```
drug_snp_effects(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the snp effects element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug snp effects node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, drug_groups(), drug_interactions(), drug_intern_brand(), drug_links(), drug_manufacturers(), drug_mixtures(), drug_packagers(), drug_patents(), drug_pdb_entries(), drug_products(), drug_salts(), drug_snp_adverse_reactions(), drug_syn(), drug()

Examples

```
## Not run:
# return only the parsed tibble
drug_snp_effects()

# will throw an error, as database_connection is NULL
drug_snp_effects(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_snp_effects(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_snp_effects(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_snp_effects(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_snp_effects(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist override it and return it.
drug_snp_effects(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

---

drug_syn *Extracts the drug syn element and return data as tibble.*

Description

drug_syn returns tibble of drug syn elements.

Usage

```
drug_syn(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the syn element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug syn node attributes tibble

See Also

Other drugs: `drug_affected_organisms()`, `drug_ahfs_codes()`, `drug_articles()`, `drug_atc_codes()`, `drug_books()`, `drug_calc_prop()`, `drug_categories()`, `drug_classification()`, `drug_dosages()`, `drug_ex_identity()`, `drug_exp_prop()`, `drug_external_links()`, `drug_food_interactions()`, `drug_groups()`, `drug_interactions()`, `drug_intern_brand()`, `drug_links()`, `drug_manufacturers()`, `drug_mixtures()`, `drug_packagers()`, `drug_patents()`, `drug_pdb_entries()`, `drug_products()`, `drug_salts()`, `drug_snp_adverse_reactions()`, `drug_snp_effects()`, `drug()`

Examples

```
## Not run:
# return only the parsed tibble
drug_syn()
```

```
# will throw an error, as database_connection is NULL
drug_syn(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
drug_syn(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_syn(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
drug_syn(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
drug_syn(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
drug_syn(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`enzymes` *Extracts the drug enzymes element and return data as tibble.*

Description

`enzymes` returns tibble of drug enzymes elements.

Usage

```
enzymes(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the enzymes element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug enzymes node attributes date frame

See Also

Other enzymes: `enzymes_actions()`, `enzymes_articles()`, `enzymes_links()`, `enzymes_polypeptide_ext_ident()`, `enzymes_polypeptide_go()`, `enzymes_polypeptide_pfams()`, `enzymes_polypeptide_syn()`, `enzymes_polypeptide()`, `enzymes_textbooks()`

Examples

```
## Not run:
# return only the parsed tibble
enzymes()

# will throw an error, as database_connection is NULL
enzymes(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
enzymes(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
```

```
# If the csv exist before read it and return its data.
enzymes(save_table = TRUE, save_csv = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
enzymes(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`enzymes_actions` *Extracts the drug enzymes actions element and return data as tibble.*

Description

`enzymes_actions` returns tibble of drug enzymes actions elements.

Usage

```
enzymes_actions(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the enzymes actions element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug enzymes actions node attributes date frame

See Also

Other enzymes: `enzymes_articles()`, `enzymes_links()`, `enzymes_polypeptide_ext_ident()`, `enzymes_polypeptide_go()`, `enzymes_polypeptide_pfams()`, `enzymes_polypeptide_syn()`, `enzymes_polypeptide()`, `enzymes_textbooks()`, `enzymes()`

Examples

```
## Not run:
# return only the parsed tibble
enzymes_actions()

# will throw an error, as database_connection is NULL
enzymes_actions(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
enzymes_actions(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_actions(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_actions(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_actions(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
enzymes_actions(save_csv = TRUE, csv_path = TRUE, override = TRUE)
```

```
## End(Not run)
```

---

enzymes_articles *Extracts the drug enzymes articles element and return data as tibble.*

Description

`enzymes_articles` returns tibble of drug enzymes articles elements.

Usage

```
enzymes_articles(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the enzymes articles element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug enzymes articles node attributes date frame

See Also

Other enzymes: `enzymes_actions()`, `enzymes_links()`, `enzymes_polypeptide_ext_ident()`, `enzymes_polypeptide_go()`, `enzymes_polypeptide_pfams()`, `enzymes_polypeptide_syn()`, `enzymes_polypeptide()`, `enzymes_textbooks()`, `enzymes()`

Examples

```
## Not run:
# return only the parsed tibble
enzymes_articles()

# will throw an error, as database_connection is NULL
enzymes_articles(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
enzymes_articles(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_articles(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_articles(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_articles(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
enzymes_articles(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`enzymes_links` *Extracts the drug enzymes links element and return data as tibble.*

Description

`enzymes_links` returns tibble of drug enzymes links elements.

Usage

```
enzymes_links(
  save_table = FALSE,
```

```
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the enzymes links element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug enzymes links node attributes date frame

See Also

Other enzymes: `enzymes_actions()`, `enzymes_articles()`, `enzymes_polypeptide_ext_ident()`, `enzymes_polypeptide_go()`, `enzymes_polypeptide_pfams()`, `enzymes_polypeptide_syn()`, `enzymes_polypeptide()`, `enzymes_textbooks()`, `enzymes()`

Examples

```
## Not run:
# return only the parsed tibble
enzymes_links()

# will throw an error, as database_connection is NULL
enzymes_links(save_table = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_links(save_csv = TRUE)
```

```
# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_links(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_links(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
enzymes_links(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`enzymes_polypeptide` *Extracts the drug enzymes polypeptides element and return data as tibble.*

Description

`enzymes_polypeptide` returns tibble of drug enzymes polypeptides elements.

Usage

```
enzymes_polypeptide(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the enzymes polypeptides element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug enzymes polypeptides node attributes date frame

See Also

Other enzymes: `enzymes_actions()`, `enzymes_articles()`, `enzymes_links()`, `enzymes_polypeptide_ext_ident()`, `enzymes_polypeptide_go()`, `enzymes_polypeptide_pfams()`, `enzymes_polypeptide_syn()`, `enzymes_textbooks()`, `enzymes()`

Examples

```
## Not run:
# return only the parsed tibble
enzymes_polypeptide()

# will throw an error, as database_connection is NULL
enzymes_polypeptide(save_table = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_polypeptide(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not
# exist in current location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_polypeptide(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_polypeptide(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
enzymes_polypeptide(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`enzymes_polypeptide_ext_ident` *Extracts the drug enzymes polypeptides external identifiers element and return data as tibble.*

Description

`enzymes_polypeptide_ext_ident` returns data frame of drug enzymes polypeptides external identifiers elements.

Usage

```
enzymes_polypeptide_ext_ident(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the enzymes polypeptides external identifiers element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug enzymes polypeptides external identifiers node attributes date frame

See Also

Other enzymes: `enzymes_actions()`, `enzymes_articles()`, `enzymes_links()`, `enzymes_polypeptide_go()`, `enzymes_polypeptide_pfams()`, `enzymes_polypeptide_syn()`, `enzymes_polypeptide()`, `enzymes_textbooks()`, `enzymes()`

Examples

```
## Not run:
# return only the parsed tibble
enzymes_polypeptide_ext_ident()

# will throw an error, as database_connection is NULL
enzymes_polypeptide_ext_ident(save_table = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_polypeptide_ext_ident(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
#  current location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_polypeptide_ext_ident(
  save_table = TRUE,
  save_csv = TRUE
)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_polypeptide_ext_ident(
  save_csv = TRUE,
  csv_path = TRUE
)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
enzymes_polypeptide_ext_ident(
  save_csv = TRUE, csv_path = TRUE, override = TRUE
)

## End(Not run)
```

`enzymes_polypeptide_go` *Extracts the drug groups element and return data as tibble.*

Description

`enzymes_polypeptide_go` returns data frame of drug enzymes polypeptides go classifiers elements.

Usage

```
enzymes_polypeptide_go(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the enzymes polypeptides go classifiers element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug enzymes polypeptides go classifiers node attributes date frame

See Also

Other enzymes: `enzymes_actions()`, `enzymes_articles()`, `enzymes_links()`, `enzymes_polypeptide_ext_ident()`, `enzymes_polypeptide_pfams()`, `enzymes_polypeptide_syn()`, `enzymes_polypeptide()`, `enzymes_textbooks()`, `enzymes()`

Examples

```
## Not run:
# return only the parsed tibble
enzymes_polypeptide_go()

# will throw an error, as database_connection is NULL
enzymes_polypeptide_go(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
enzymes_polypeptide_go(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_polypeptide_go(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
#  current location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_polypeptide_go(
  save_table = TRUE,
  save_csv = TRUE
)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_polypeptide_go(
  save_csv = TRUE,
  csv_path = TRUE
)

# save parsed tibble as csv if it does not exist in
# current location and return parsed tibble.
# If the csv exist override it and return it.
enzymes_polypeptide_go(
  save_csv = TRUE,
  csv_path = TRUE, override = TRUE
)

## End(Not run)
```

`enzymes_polypeptide_pfams` *Extracts the drug enzymes polypeptides pfams element and return data as tibble.*

Description

`enzymes_polypeptide_pfams` returns tibble of drug enzymes polypeptides pfams elements.

Usage

```
enzymes_polypeptide_pfams(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the enzymes polypeptides pfams element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug groups node attributes date frame

See Also

Other enzymes: `enzymes_actions()`, `enzymes_articles()`, `enzymes_links()`, `enzymes_polypeptide_ext_ident()`, `enzymes_polypeptide_go()`, `enzymes_polypeptide_syn()`, `enzymes_polypeptide()`, `enzymes_textbooks()`, `enzymes()`

Examples

```
## Not run:
# return only the parsed tibble
enzymes_polypeptide_pfams()

# will throw an error, as database_connection is NULL
enzymes_polypeptide_pfams(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
```

```
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
enzymes_polypeptide_pfams(save_table = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_polypeptide_pfams(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_polypeptide_pfams(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_polypeptide_pfams(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in
# current location and return parsed tibble.
# If the csv exist override it and return it.
enzymes_polypeptide_pfams(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`enzymes_polypeptide_syn` *Extracts the drug enzymes polypeptides syn element and return data as tibble.*

Description

`enzymes_polypeptide_syn` returns tibble of drug enzymes polypeptides syn elements.

Usage

```
enzymes_polypeptide_syn(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the enzymes polypeptides syn element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug enzymes polypeptides syn node attributes date frame

See Also

Other enzymes: `enzymes_actions()`, `enzymes_articles()`, `enzymes_links()`, `enzymes_polypeptide_ext_ident()`, `enzymes_polypeptide_go()`, `enzymes_polypeptide_pfams()`, `enzymes_polypeptide()`, `enzymes_textbooks()`, `enzymes()`

Examples

```
## Not run:
# return only the parsed tibble
enzymes_polypeptide_syn()

# will throw an error, as database_connection is NULL
enzymes_polypeptide_syn(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
enzymes_polypeptide_syn(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_polypeptide_syn(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does
# not exist in current location and return parsed tibble.
```

```
# If the csv exist before read it and return its data.
enzymes_polypeptide_syn(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in
# given location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_polypeptide_syn(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
enzymes_polypeptide_syn(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`enzymes_textbooks` *Extracts the drug enzymes textbooks element and return data as tibble.*

Description

`enzymes_textbooks` returns tibble of drug enzymes textbooks elements.

Usage

```
enzymes_textbooks(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the enzymes textbooks element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug enzymes textbooks node attributes date frame

See Also

Other enzymes: `enzymes_actions()`, `enzymes_articles()`, `enzymes_links()`, `enzymes_polypeptide_ext_ident()`, `enzymes_polypeptide_go()`, `enzymes_polypeptide_pfams()`, `enzymes_polypeptide_syn()`, `enzymes_polypeptide()`, `enzymes()`

Examples

```
## Not run:
# return only the parsed tibble
enzymes_textbooks()

# will throw an error, as database_connection is NULL
enzymes_textbooks(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
enzymes_textbooks(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_textbooks(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_textbooks(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
enzymes_textbooks(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
enzymes_textbooks(
  save_csv = TRUE, csv_path = TRUE,
```

```
  override = TRUE
)

## End(Not run)
```

`get_drugbank_exported_date` *Return uploaded drugbank database exported date*

Description

`get_drugbank_exported_date` returns uploaded drugbank database exported date.

Usage

```
get_drugbank_exported_date()
```

Value

drugbank exported date

Examples

```
## Not run:
get_drugbank_exported_date()

## End(Not run)
```

`get_drugbank_metadata` *Return uploaded drugbank database metadata*

Description

`get_drugbank_metadata` returns uploaded drugbank database version and exported date.

Usage

```
get_drugbank_metadata()
```

Value

drugbank metadata

Examples

```
## Not run:
get_drugbank_metadata()

## End(Not run)
```

`get_drugbank_version` *Return uploaded drugbank database version*

Description

`get_drugbank_version` returns uploaded drugbank database version.

Usage

```
get_drugbank_version()
```

Value

drugbank version

Examples

```
## Not run:
get_drugbank_version()

## End(Not run)
```

`read_drugbank_xml_db` *Reads* **DrugBank** *xml database and load it into memory.*

Description

`read_drugbank_xml_db` loads **DrugBank** xml database full tree into memory.

Usage

```
read_drugbank_xml_db(drugbank_db_path)
```

Arguments

`drugbank_db_path`	**string**, full path for the **DrugBank** xml or zip file.

Details

This functions reads **DrugBank** xml database and load it into memory for later processing. Hence; this method **must** be called before any other function in the package and it needs to be called one time only.

It takes one single mandatory argument which is the location of DrugBank db.

Value

TRUE when the loading process into memory to be used by parser methods is completed successfully and **FALSE** otherwise.

Examples

```
## Not run:
read_drugbank_xml_db("db_full_path")
read_drugbank_xml_db(drugbank_db_path = "db_full_path")

## End(Not run)
```

`targets` *Extracts the drug targ element and return data as tibble.*

Description

`targets` returns tibble of drug targ elements.

Usage

```
targets(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the target element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug target node attributes date frame

See Also

Other targets: `targets_actions()`, `targets_articles()`, `targets_links()`, `targets_polypeptide_ext_ident()`, `targets_polypeptide_go()`, `targets_polypeptide_pfams()`, `targets_polypeptide_syn()`, `targets_polypeptide()`, `targets_textbooks()`

Examples

```
## Not run:
# return only the parsed tibble
targets()

# will throw an error, as database_connection is NULL
targets(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
targets(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
targets(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not
# exist in current location and return parsed tibble.
# If the csv exist before read it and return its data.
targets(save_table = TRUE, save_csv = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
targets(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
#  location and return parsed tibble.
# If the csv exist override it and return it.
targets(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`targets_actions` *Extracts the drug targ actions element and return data as tibble.*

Description

`targets_actions` returns tibble of drug targ actions elements.

Usage

```
targets_actions(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the targ actions element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug targ actions node attributes date frame

See Also

Other targets: `targets_articles`(), `targets_links`(), `targets_polypeptide_ext_ident`(), `targets_polypeptide_go`(), `targets_polypeptide_pfams`(), `targets_polypeptide_syn`(), `targets_polypeptide`(), `targets_textbooks`(), `targets`()

Examples

```
## Not run:
# return only the parsed tibble
targets_actions()

# will throw an error, as database_connection is NULL
targets_actions(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
```

```
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
targets_actions(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_actions(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not
# exist in current location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_actions(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_actions(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
targets_actions(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`targets_articles` *Extracts the drug targ articles element and return data as tibble.*

Description

`targets_articles` returns tibble of drug targ articles elements.

Usage

```
targets_articles(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true

`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the targ articles element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug targ articles node attributes date frame

See Also

Other targets: `targets_actions()`, `targets_links()`, `targets_polypeptide_ext_ident()`, `targets_polypeptide_go()`, `targets_polypeptide_pfams()`, `targets_polypeptide_syn()`, `targets_polypeptide()`, `targets_textbooks()`, `targets()`

Examples

```
## Not run:
# return only the parsed tibble
targets_articles()

# will throw an error, as database_connection is NULL
targets_articles(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
targets_articles(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_articles(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not
# exist in current location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_articles(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
```

```
targets_articles(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
targets_articles(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`targets_links` *Extracts the drug targ links element and return data as tibble.*

Description

`targets_links` returns tibble of drug targ links elements.

Usage

```
targets_links(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the targ links element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug targ_links node attributes date frame

See Also

Other targets: `targets_actions`(), `targets_articles`(), `targets_polypeptide_ext_ident`(), `targets_polypeptide_go`(), `targets_polypeptide_pfams`(), `targets_polypeptide_syn`(), `targets_polypeptide`(), `targets_textbooks`(), `targets`()

Examples

```
## Not run:
# return only the parsed tibble
targets_links()

# will throw an error, as database_connection is NULL
targets_links(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
targets_links(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_links(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not
# exist in current location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_links(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_links(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in
#  current location and return parsed tibble.
# If the csv exist override it and return it.
targets_links(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`targets_polypeptide` *Extracts the drug targ polypeptides element and return data as tibble.*

Description

`targets_polypeptide` returns tibble of drug targ polypeptides elements.

Usage

```
targets_polypeptide(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the targ polypeptides element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug targ polypeptides node attributes date frame

See Also

Other targets: `targets_actions()`, `targets_articles()`, `targets_links()`, `targets_polypeptide_ext_ident()`, `targets_polypeptide_go()`, `targets_polypeptide_pfams()`, `targets_polypeptide_syn()`, `targets_textbooks()`, `targets()`

Examples

```
## Not run:
# return only the parsed tibble
targets_polypeptide()
```

```
# will throw an error, as database_connection is NULL
targets_polypeptide(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
targets_polypeptide(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_polypeptide(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not
# exist in current location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_polypeptide(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_polypeptide(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
targets_polypeptide(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

---

targets_polypeptide_ext_ident

Extracts the drug targ polypeptides external identifiers element and return data as tibble.

Description

`targets_polypeptide_ext_ident` returns tibble of drug targ polypeptides external identifiers elements.

Usage

```
targets_polypeptide_ext_ident(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
```

```
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the targ polypeptides external identifiers element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug targ polypeptides external identifiers node attributes date frame

See Also

Other targets: `targets_actions`(), `targets_articles`(), `targets_links`(), `targets_polypeptide_go`(), `targets_polypeptide_pfams`(), `targets_polypeptide_syn`(), `targets_polypeptide`(), `targets_textbooks`(), `targets`()

Examples

```
## Not run:
# return only the parsed tibble
targets_polypeptide_ext_ident()

# will throw an error, as database_connection is NULL
targets_polypeptide_ext_ident(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
targets_polypeptide_ext_ident(save_table = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current location
```

```
# and return parsed tibble.
# If the csv exist before read it and return its data.
targets_polypeptide_ext_ident(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_polypeptide_ext_ident(
  save_table = TRUE,
  save_csv = TRUE
)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
targets_polypeptide_ext_ident(
  save_csv = TRUE,
  csv_path = TRUE
)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist override it and return it.
targets_polypeptide_ext_ident(
  save_csv = TRUE, csv_path = TRUE, override = TRUE
)

## End(Not run)
```

`targets_polypeptide_go` *Extracts the drug targ polypeptides go classifiers element and return data as tibble.*

Description

`targets_polypeptide_go` returns tibble of drug targ polypeptides go classifiers elements.

Usage

```
targets_polypeptide_go(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the targ polypeptides go classifiers element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug targ polypeptides go classifiers node attributes date frame

See Also

Other targets: `targets_actions`(), `targets_articles`(), `targets_links`(), `targets_polypeptide_ext_ident`(), `targets_polypeptide_pfams`(), `targets_polypeptide_syn`(), `targets_polypeptide`(), `targets_textbooks`(), `targets`()

Examples

```
## Not run:
# return only the parsed tibble
targets_polypeptide_go()

# will throw an error, as database_connection is NULL
targets_polypeptide_go(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
targets_polypeptide_go(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist before read it and return its data.
targets_polypeptide_go(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
```

```
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_polypeptide_go(
  save_table = TRUE,
  save_csv = TRUE
)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_polypeptide_go(
  save_csv = TRUE,
  csv_path = TRUE
)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
targets_polypeptide_go(
  save_csv = TRUE,
  csv_path = TRUE, override = TRUE
)

## End(Not run)
```

`targets_polypeptide_pfams` *Extracts the drug targ polypeptides pfams element and return data as tibble.*

Description

`targets_polypeptide_pfams` returns tibble of drug targ polypeptides pfams elements.

Usage

```
targets_polypeptide_pfams(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true

`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the targ polypeptides pfams element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug targ polypeptides pfams node attributes date frame

See Also

Other targets: `targets_actions()`, `targets_articles()`, `targets_links()`, `targets_polypeptide_ext_ident()`, `targets_polypeptide_go()`, `targets_polypeptide_syn()`, `targets_polypeptide()`, `targets_textbooks()`, `targets()`

Examples

```
## Not run:
# return only the parsed tibble
targets_polypeptide_pfams()

# will throw an error, as database_connection is NULL
targets_polypeptide_pfams(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
targets_polypeptide_pfams(save_table = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist before read it and return its data.
targets_polypeptide_pfams(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_polypeptide_pfams(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)
```

```
# save parsed tibble as csv if it does not exist in given location and
#  return parsed tibble.
# If the csv exist before read it and return its data.
targets_polypeptide_pfams(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist override it and return it.
targets_polypeptide_pfams(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`targets_polypeptide_syn` *Extracts the drug targ polypeptides syn element and return data as tibble.*

Description

`targets_polypeptide_syn` returns data frame of drug targ polypeptides syn elements.

Usage

```
targets_polypeptide_syn(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the targ polypeptides syn element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug targ polypeptides syn node attributes date frame

See Also

Other targets: `targets_actions`(), `targets_articles`(), `targets_links`(), `targets_polypeptide_ext_ident`(), `targets_polypeptide_go`(), `targets_polypeptide_pfams`(), `targets_polypeptide`(), `targets_textbooks`(), `targets`()

Examples

```
## Not run:
# return only the parsed tibble
targets_polypeptide()

# will throw an error, as database_connection is NULL
targets_polypeptide_syn(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
targets_polypeptide_syn(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_polypeptide_syn(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_polypeptide_syn(save_table = TRUE, save_csv = TRUE,
 database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location and
# return parsed tibble.
# If the csv exist before read it and return its data.
targets_polypeptide_syn(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist override it and return it.
targets_polypeptide_syn(
  save_csv = TRUE, csv_path = TRUE,
```

```
  override = TRUE
)

## End(Not run)
```

`targets_textbooks` *Extracts the drug targ textbooks element and return data as tibble.*

Description

`targets_textbooks` returns tibble of drug targ textbooks elements.

Usage

```
targets_textbooks(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the targ textbooks element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug targ textbooks node attributes date frame

See Also

Other targets: `targets_actions()`, `targets_articles()`, `targets_links()`, `targets_polypeptide_ext_ident()`, `targets_polypeptide_go()`, `targets_polypeptide_pfams()`, `targets_polypeptide_syn()`, `targets_polypeptide()`, `targets()`

Examples

```
## Not run:
# return only the parsed tibble
targets_textbooks()

# will throw an error, as database_connection is NULL
targets_textbooks(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
targets_textbooks(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_textbooks(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not
# exist in current location and return parsed tibble.
# If the csv exist before read it and return its data.
targets_textbooks(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
targets_textbooks(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
targets_textbooks(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`transporters` *Extracts the drug transporters element and return data as tibble.*

Description

`transporters` returns tibble of drug transporters elements.

Usage

```
transporters(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the transporters element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug transporters node attributes date frame

See Also

Other transporters: `transporters_actions()`, `transporters_articles()`, `transporters_links()`, `transporters_polypep_ex_ident()`, `transporters_polypeptide_go()`, `transporters_polypeptide_pfams()`, `transporters_polypeptide_syn()`, `transporters_polypeptide()`, `transporters_textbooks()`

Examples

```
## Not run:
# return only the parsed tibble
transporters()

# will throw an error, as database_connection is NULL
transporters(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
```

```
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
transporters_textbooks(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in
# current location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not
# exist in current location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters(save_table = TRUE, save_csv = TRUE,
database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
transporters(save_csv = TRUE, csv_path = TRUE, override = TRUE)

## End(Not run)
```

`transporters_actions` *Extracts the drug transporters actions element and return data as tibble.*

Description

`transporters actions` returns tibble of drug transporters actions elements.

Usage

```
transporters_actions(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true

`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the transporters actions element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug transporters actions node attributes date frame

See Also

Other transporters: `transporters_articles()`, `transporters_links()`, `transporters_polypep_ex_ident()`, `transporters_polypeptide_go()`, `transporters_polypeptide_pfams()`, `transporters_polypeptide_syn()`, `transporters_polypeptide()`, `transporters_textbooks()`, `transporters()`

Examples

```
## Not run:
# return only the parsed tibble
transporters_actions()

# will throw an error, as database_connection is NULL
transporters_actions(save_table = TRUE)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_actions(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_actions(save_table = TRUE, save_csv = TRUE)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_actions(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current location
```

```
#  and return parsed tibble.
# If the csv exist override it and return it.
transporters_actions(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`transporters_articles` *Extracts the drug transporters articles element and return data as tibble.*

Description

`transporters articles` returns tibble of drug transporters articles elements.

Usage

```
transporters_articles(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the transporters articles element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug transporters articles node attributes date frame

See Also

Other transporters: `transporters_actions()`, `transporters_links()`, `transporters_polypep_ex_ident()`, `transporters_polypeptide_go()`, `transporters_polypeptide_pfams()`, `transporters_polypeptide_syn()`, `transporters_polypeptide()`, `transporters_textbooks()`, `transporters()`

Examples

```
## Not run:
# return only the parsed tibble
transporters_articles()

# will throw an error, as database_connection is NULL
transporters_articles(save_table = TRUE)

# save parsed tibble as csv if it does not exist in current location
# and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_articles(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_articles(save_table = TRUE, save_csv = TRUE)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_articles(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
transporters_articles(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`transporters_links` *Extracts the drug transporters links element and return data as tibble.*

Description

`transporters_links` returns tibble of drug transporters_links elements.

Usage

```
transporters_links(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the transporters links element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug transporters links node attributes date frame

See Also

Other transporters: `transporters_actions()`, `transporters_articles()`, `transporters_polypep_ex_ident()`, `transporters_polypeptide_go()`, `transporters_polypeptide_pfams()`, `transporters_polypeptide_syn()`, `transporters_polypeptide()`, `transporters_textbooks()`, `transporters()`

Examples

```
## Not run:
# return only the parsed tibble
transporters_links()

# will throw an error, as database_connection is NULL
transporters_links(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
```

```
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
transporters_textbooks(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_links(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_links(save_table = TRUE, save_csv = TRUE)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_links(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
transporters_links(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`transporters_polypeptide` *Extracts the drug transporters polypeptides element and return data as tibble.*

Description

`transporters_polypeptide` returns tibble of transporters polypeptides groups elements.

Usage

```
transporters_polypeptide(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the transporters polypeptides element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug transporters polypeptides node attributes date frame

See Also

Other transporters: `transporters_actions()`, `transporters_articles()`, `transporters_links()`, `transporters_polypep_ex_ident()`, `transporters_polypeptide_go()`, `transporters_polypeptide_pfams()`, `transporters_polypeptide_syn()`, `transporters_textbooks()`, `transporters()`

Examples

```
## Not run:
# return only the parsed tibble
transporters_polypeptide()

# will throw an error, as database_connection is NULL
transporters_polypeptide(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
transporters_textbooks(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_polypeptide(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not
# exist in current location and return parsed tibble.
```

```
# If the csv exist before read it and return its data.
transporters_polypeptide(save_table = TRUE, save_csv = TRUE)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_polypeptide(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
transporters_polypeptide(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`transporters_polypeptide_go` *Extracts the drug transporters polypeptides go classifiers element and return data as tibble.*

Description

`transporters_polypeptide_go` returns tibble of drug transporters polypeptides go classifiers elements.

Usage

```
transporters_polypeptide_go(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the transporters polypeptides go classifiers element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug transporters polypeptides go classifiers node attributes date frame

See Also

Other transporters: `transporters_actions()`, `transporters_articles()`, `transporters_links()`, `transporters_polypep_ex_ident()`, `transporters_polypeptide_pfams()`, `transporters_polypeptide_syn()`, `transporters_polypeptide()`, `transporters_textbooks()`, `transporters()`

Examples

```
## Not run:
# return only the parsed tibble
transporters_polypeptide_go()

# will throw an error, as database_connection is NULL
transporters_polypeptide_go(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
transporters_textbooks(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_polypeptide_go(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist in
#  current location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_polypeptide_go(
  save_table = TRUE,
  save_csv = TRUE
)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_polypeptide_go(
  save_csv = TRUE,
  csv_path = TRUE
)
```

```
# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
transporters_polypeptide_go(
  save_csv = TRUE, csv_path = TRUE, override = TRUE
)

## End(Not run)
```

`transporters_polypeptide_pfams` *Extracts the drug transporters polypeptides pfams element and return data as tibble.*

Description

`transporters_polypeptide_pfams` returns tibble of drug transporters polypeptides pfams elements.

Usage

```
transporters_polypeptide_pfams(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the transporters polypeptides pfams element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug transporters polypeptides pfams node attributes date frame

See Also

Other transporters: `transporters_actions()`, `transporters_articles()`, `transporters_links()`, `transporters_polypep_ex_ident()`, `transporters_polypeptide_go()`, `transporters_polypeptide_syn()`, `transporters_polypeptide()`, `transporters_textbooks()`, `transporters()`

Examples

```
## Not run:
# return only the parsed tibble
transporters_polypeptide_pfams()

# will throw an error, as database_connection is NULL
transporters_polypeptide_pfams(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
transporters_textbooks(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_polypeptide_pfams(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not
# exist in current location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_polypeptide_pfams(
  save_table = TRUE,
  save_csv = TRUE
)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_polypeptide_pfams(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
```

```
# If the csv exist override it and return it.
transporters_polypeptide_pfams(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

`transporters_polypeptide_syn`
Extracts the drug transporters polypeptides syn element and return data as tibble.

Description

`transporters_polypeptide_syn` returns data frame of drug transporters polypeptides syn elements.

Usage

```
transporters_polypeptide_syn(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the transporters polypeptides syn element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug transporters polypeptides syn node attributes date frame

See Also

Other transporters: `transporters_actions()`, `transporters_articles()`, `transporters_links()`, `transporters_polypep_ex_ident()`, `transporters_polypeptide_go()`, `transporters_polypeptide_pfams()`, `transporters_polypeptide()`, `transporters_textbooks()`, `transporters()`

Examples

```
## Not run:
# return only the parsed tibble
transporters_polypeptide_syn()

# will throw an error, as database_connection is NULL
transporters_polypeptide_syn(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
transporters_textbooks(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_polypeptide_syn(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not
# exist in current location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_polypeptide_syn(
  save_table = TRUE,
  save_csv = TRUE
)

# save parsed tibble as csv if it does not exist in given
# location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_polypeptide_syn(
  save_csv = TRUE,
  csv_path = TRUE
)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
transporters_polypeptide_syn(
  save_csv = TRUE,
  csv_path = TRUE, override = TRUE
)

## End(Not run)
```

`transporters_polypep_ex_ident` *Extracts the drug transporters polypeptides external identifiers element and return data as tibble.*

Description

`transporters_polypep_ex_ident` returns tibble of drug transporters polypeptides external identifiers elements.

Usage

```
transporters_polypep_ex_ident(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the transporters polypeptides external identifiers element of drug node in **DrugBank** xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug transporters polypeptides external identifiers node attributes date frame

See Also

Other transporters: transporters_actions(), transporters_articles(), transporters_links(), transporters_polypeptide_go(), transporters_polypeptide_pfams(), transporters_polypeptide_syn(), transporters_polypeptide(), transporters_textbooks(), transporters()

Examples

```
## Not run:
# return only the parsed tibble
transporters_polypep_ex_ident()

# will throw an error, as database_connection is NULL
transporters_polypep_ex_ident(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
transporters_textbooks(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_polypep_ex_ident(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_polypep_ex_ident(
  save_table = TRUE, save_csv = TRUE
)

# save parsed tibble as csv if it does not exist in given location
# and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_polypep_ex_ident(
  save_csv = TRUE, csv_path = TRUE
)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
transporters_polypep_ex_ident(
  save_csv = TRUE, csv_path = TRUE, override = TRUE
)

## End(Not run)
```

`transporters_textbooks` *Extracts the drug transporters textbooks element and return data as tibble.*

Description

`transporters_textbooks` returns tibble of drug transporters textbooks elements.

Usage

```
transporters_textbooks(
  save_table = FALSE,
  save_csv = FALSE,
  csv_path = ".",
  override_csv = FALSE,
  database_connection = NULL
)
```

Arguments

`save_table`	boolean, save table in database if true.
`save_csv`	boolean, save csv version of parsed tibble if true
`csv_path`	location to save csv files into it, default is current location, save_csv must be true
`override_csv`	override existing csv, if any, in case it is true in the new parse operation
`database_connection`	DBI connection object that holds a connection to user defined database. If `save_table` is enabled without providing value for this function an error will be thrown.

Details

This functions extracts the transporters textbooks element of drug node in drugbank xml database with the option to save it in a predefined database via passed database connection. It takes two optional arguments to save the returned tibble in the database `save_table` and `database_connection`. It must be called after `read_drugbank_xml_db` function like any other parser function. If `read_drugbank_xml_db` is called before for any reason, so no need to call it again before calling this function.

Value

drug transporters textbooks node attributes date frame

See Also

Other transporters: `transporters_actions()`, `transporters_articles()`, `transporters_links()`, `transporters_polypep_ex_ident()`, `transporters_polypeptide_go()`, `transporters_polypeptide_pfams()`, `transporters_polypeptide_syn()`, `transporters_polypeptide()`, `transporters()`

Examples

```
## Not run:
# return only the parsed tibble
transporters_textbooks()
```

```
# will throw an error, as database_connection is NULL
transporters_textbooks(save_table = TRUE)

# save in database in SQLite in memory database and return parsed tibble
sqlite_con <- DBI::dbConnect(RSQLite::SQLite())
transporters_textbooks(save_table = TRUE, database_connection = sqlite_con)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_textbooks(save_csv = TRUE)

# save in database, save parsed tibble as csv if it does not exist
# in current location and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_textbooks(save_table = TRUE, save_csv = TRUE)

# save parsed tibble as csv if it does not exist in given location
#  and return parsed tibble.
# If the csv exist before read it and return its data.
transporters_textbooks(save_csv = TRUE, csv_path = TRUE)

# save parsed tibble as csv if it does not exist in current
# location and return parsed tibble.
# If the csv exist override it and return it.
transporters_textbooks(
  save_csv = TRUE, csv_path = TRUE,
  override = TRUE
)

## End(Not run)
```

Index

carriers, 3, 6, 8, 10, 11, 13, 15, 17, 19, 20
carriers_actions, 4, 5, 8, 10, 11, 13, 15, 17, 19, 20
carriers_articles, 4, 6, 7, 10, 11, 13, 15, 17, 19, 20
carriers_links, 4, 6, 8, 9, 11, 13, 15, 17, 19, 20
carriers_polypeptide, 4, 6, 8, 10, 10, 13, 15, 17, 19, 20
carriers_polypeptide_ext_id, 4, 6, 8, 10, 11, 13, 15, 17, 18, 20
carriers_polypeptides_go, 4, 6, 8, 10, 11, 12, 15, 17, 19, 20
carriers_polypeptides_pfams, 4, 6, 8, 10, 11, 13, 14, 17, 19, 20
carriers_polypeptides_syn, 4, 6, 8, 10, 11, 13, 15, 16, 19, 20
carriers_textbooks, 4, 6, 8, 10, 11, 13, 15, 17, 19, 20

dbparser, 21
drug, 22, 25, 27, 30, 32, 34, 35, 37, 39, 41, 45, 46, 48, 50, 52, 54, 55, 57, 59, 61, 62, 64, 71, 74, 79, 83, 85, 86
drug_affected_organisms, 23, 24, 27, 30, 32, 33, 35, 37, 39, 41, 45, 46, 48, 50, 52, 54, 55, 57, 59, 61, 62, 64, 71, 74, 79, 83, 84, 86
drug_ahfs_codes, 23, 25, 26, 30, 32, 33, 35, 37, 39, 41, 45, 46, 48, 50, 52, 54, 55, 57, 59, 61, 62, 64, 71, 74, 79, 83, 84, 86
drug_all, 27, 42, 43
drug_articles, 23, 25, 27, 29, 32, 33, 35, 37, 39, 41, 45, 46, 48, 50, 52, 54, 55, 57, 59, 61, 62, 64, 71, 74, 79, 83, 84, 86
drug_atc_codes, 23, 25, 27, 30, 31, 33, 35, 37, 39, 41, 45, 46, 48, 50, 52, 54, 55, 57, 59, 61, 62, 64, 71, 74, 79, 83, 84, 86
drug_books, 23, 25, 27, 30, 32, 33, 35, 37, 39, 41, 45, 46, 48, 50, 52, 54, 55, 57, 59, 61, 62, 64, 71, 74, 79, 83, 84, 86
drug_calc_prop, 23, 25, 27, 30, 32, 33, 34, 37, 39, 41, 45, 46, 48, 50, 52, 54, 55, 57, 59, 61, 62, 64, 71, 74, 79, 83, 84, 86
drug_categories, 23, 25, 27, 30, 32, 33, 35, 36, 39, 41, 45, 46, 48, 50, 52, 54, 55, 57, 59, 61, 62, 64, 71, 74, 79, 83, 84, 86
drug_classification, 23, 25, 27, 30, 32, 33, 35, 37, 38, 41, 45, 46, 48, 50, 52, 54, 55, 57, 59, 61, 62, 64, 71, 74, 79, 83, 84, 86
drug_dosages, 23, 25, 27, 30, 32, 33, 35, 37, 39, 40, 45, 46, 48, 50, 52, 54, 55, 57, 59, 61, 62, 64, 71, 74, 79, 83, 84, 86
drug_element, 28, 41, 43
drug_element_options, 28, 42, 43
drug_ex_identity, 23, 25, 27, 30, 32, 33, 35, 37, 39, 41, 45, 46, 47, 50, 52, 54, 55, 57, 59, 61, 62, 64, 71, 74, 79, 83, 84, 86
drug_exp_prop, 23, 25, 27, 30, 32, 33, 35, 37, 39, 41, 44, 46, 48, 50, 52, 54, 55, 57, 59, 61, 62, 64, 71, 74, 79, 83, 84, 86
drug_external_links, 23, 25, 27, 30, 32, 33, 35, 37, 39, 41, 45, 45, 48, 50, 52, 54, 55, 57, 59, 61, 62, 64, 71, 74, 79, 83, 84, 86
drug_food_interactions, 23, 25, 27, 30, 32, 33, 35, 37, 39, 41, 45, 46, 48, 49, 52, 54, 55, 57, 59, 61, 62, 64, 71, 74, 79, 83, 84, 86
drug_groups, 23, 25, 27, 30, 32, 33, 35, 37, 39, 41, 45, 46, 48, 50, 51, 54, 55, 57, 59, 61, 62, 64, 71, 74, 79, 83, 85, 86
drug_interactions, 23, 25, 27, 30, 32, 33,

35, *37*, *39*, *41*, *45*, *46*, *48*, *50*, *52*, 53, *55*, *57*, *59*, *61*, *62*, *64*, *71*, *74*, *79*, *83*, *85*, *86*
`drug_intern_brand`, *23*, *25*, *27*, *30*, *32*, *34*, *35*, *37*, *39*, *41*, *45*, *46*, *48*, *50*, *52*, *54*, 54, *57*, *59*, *61*, *62*, *64*, *71*, *74*, *79*, *83*, *85*, *86*
`drug_links`, *23*, *25*, *27*, *30*, *32*, *34*, *35*, *37*, *39*, *41*, *45*, *46*, *48*, *50*, *52*, *54*, *55*, 56, *59*, *61*, *62*, *64*, *71*, *74*, *79*, *83*, *85*, *86*
`drug_manufacturers`, *23*, *25*, *27*, *30*, *32*, *34*, *35*, *37*, *39*, *41*, *45*, *46*, *48*, *50*, *52*, *54*, *55*, *57*, 58, *61*, *62*, *64*, *71*, *74*, *79*, *83*, *85*, *86*
`drug_mixtures`, *23*, *25*, *27*, *30*, *32*, *34*, *35*, *37*, *39*, *41*, *45*, *46*, *48*, *50*, *52*, *54*, *55*, *57*, *59*, 60, *62*, *64*, *71*, *74*, *79*, *83*, *85*, *86*
`drug_packagers`, *23*, *25*, *27*, *30*, *32*, *34*, *35*, *37*, *39*, *41*, *45*, *46*, *48*, *50*, *52*, *54*, *55*, *57*, *59*, *61*, 61, *64*, *71*, *74*, *79*, *83*, *85*, *86*
`drug_patents`, *23*, *25*, *27*, *30*, *32*, *34*, *35*, *37*, *39*, *41*, *45*, *46*, *48*, *50*, *52*, *54*, *55*, *57*, *59*, *61*, *62*, 63, *71*, *74*, *79*, *83*, *85*, *86*
`drug_pathway`, 65
`drug_pathway_drugs`, 67
`drug_pathway_enzyme`, 68
`drug_pdb_entries`, *23*, *25*, *27*, *30*, *32*, *34*, *35*, *37*, *39*, *41*, *45*, *46*, *48*, *50*, *52*, *54*, *55*, *57*, *59*, *61*, *62*, *64*, 70, *74*, *79*, *83*, *85*, *86*
`drug_prices`, 72
`drug_products`, *23*, *25*, *27*, *30*, *32*, *34*, *35*, *37*, *39*, *41*, *45*, *46*, *48*, *50*, *52*, *54*, *55*, *57*, *59*, *61*, *62*, *64*, *71*, 73, *79*, *83*, *85*, *86*
`drug_reactions`, 75
`drug_reactions_enzymes`, 77
`drug_salts`, *23*, *25*, *27*, *30*, *32*, *34*, *35*, *37*, *39*, *41*, *45*, *46*, *48*, *50*, *52*, *54*, *55*, *57*, *59*, *61*, *62*, *64*, *71*, *74*, 78, *83*, *85*, *86*
`drug_sequences`, 80
`drug_snp_adverse_reactions`, *23*, *25*, *27*, *30*, *32*, *34*, *35*, *37*, *39*, *41*, *45*, *46*, *48*, *50*, *52*, *54*, *55*, *57*, *59*, *61*, *62*, *64*, *71*, *74*, *79*, 82, *85*, *86*
`drug_snp_effects`, *23*, *25*, *27*, *30*, *32*, *34*, *35*, *37*, *39*, *41*, *45*, *46*, *48*, *50*, *52*, *54*, *55*, *57*, *59*, *61*, *62*, *64*, *71*, *74*, *79*, *83*, 84, *86*
`drug_syn`, *23*, *25*, *27*, *30*, *32*, *34*, *35*, *37*, *39*, *41*, *45*, *46*, *48*, *50*, *52*, *54*, *55*, *57*, *59*, *61*, *62*, *64*, *71*, *74*, *79*, *83*, *85*, 85

`enzymes`, 87, *90*, *91*, *93*, *95*, *97*, *98*, *100*, *102*, *104*
`enzymes_actions`, *88*, 89, *91*, *93*, *95*, *97*, *98*, *100*, *102*, *104*
`enzymes_articles`, *88*, *90*, 91, *93*, *95*, *97*, *98*, *100*, *102*, *104*
`enzymes_links`, *88*, *90*, *91*, 92, *95*, *97*, *98*, *100*, *102*, *104*
`enzymes_polypeptide`, *88*, *90*, *91*, *93*, 94, *97*, *98*, *100*, *102*, *104*
`enzymes_polypeptide_ext_ident`, *88*, *90*, *91*, *93*, *95*, 96, *98*, *100*, *102*, *104*
`enzymes_polypeptide_go`, *88*, *90*, *91*, *93*, *95*, *97*, 97, *100*, *102*, *104*
`enzymes_polypeptide_pfams`, *88*, *90*, *91*, *93*, *95*, *97*, *98*, 99, *102*, *104*
`enzymes_polypeptide_syn`, *88*, *90*, *91*, *93*, *95*, *97*, *98*, *100*, 101, *104*
`enzymes_textbooks`, *88*, *90*, *91*, *93*, *95*, *97*, *98*, *100*, *102*, 103

`get_drugbank_exported_date`, 105
`get_drugbank_metadata`, 105
`get_drugbank_version`, 106

`read_drugbank_xml_db`, *4*, *6*, *8*, *9*, *11*, *13*, *15*, *17*, *18*, *20*, *23*, *25*, *26*, *28*, *30*, *31*, *33*, *35*, *37*, *39*, *40*, *42*, *44*, *46*, *48*, *50*, *52*, *53*, *55*, *57*, *59*, *60*, *62*, *64*, *66*, *67*, *69*, *70*, *72*, *74*, *76*, *77*, *79*, *81*, *82*, *84*, *86*, *88*, *90*, *91*, *93*, *95*, *96*, *98*, *100*, *102*, *104*, 106, *107*, *109*, *111*, *112*, *114*, *116*, *118*, *120*, *122*, *123*, *125*, *127*, *128*, *130*, *132*, *134*, *136*, *137*, *139*, *141*

`targets`, 107, *109*, *111*, *113*, *114*, *116*, *118*, *120*, *122*, *124*
`targets_actions`, *108*, 108, *111*, *113*, *114*, *116*, *118*, *120*, *122*, *124*
`targets_articles`, *108*, *109*, 110, *113*, *114*, *116*, *118*, *120*, *122*, *124*
`targets_links`, *108*, *109*, *111*, 112, *114*, *116*, *118*, *120*, *122*, *124*

targets_polypeptide, *108*, *109*, *111*, *113*, 113, *116*, *118*, *120*, *122*, *124*
targets_polypeptide_ext_ident, *108*, *109*, *111*, *113*, *114*, 115, *118*, *120*, *122*, *124*
targets_polypeptide_go, *108*, *109*, *111*, *113*, *114*, *116*, 117, *120*, *122*, *124*
targets_polypeptide_pfams, *108*, *109*, *111*, *113*, *114*, *116*, *118*, 119, *122*, *124*
targets_polypeptide_syn, *108*, *109*, *111*, *113*, *114*, *116*, *118*, *120*, 121, *124*
targets_textbooks, *108*, *109*, *111*, *113*, *114*, *116*, *118*, *120*, *122*, 123
transporters, 124, *127*, *129*, *130*, *132*, *134*, *136*, *138*, *140*, *141*
transporters_actions, *125*, 126, *129*, *130*, *132*, *134*, *136*, *138*, *140*, *141*
transporters_articles, *125*, *127*, 128, *130*, *132*, *134*, *136*, *138*, *140*, *141*
transporters_links, *125*, *127*, *129*, 129, *132*, *134*, *136*, *138*, *140*, *141*
transporters_polypep_ex_ident, *125*, *127*, *129*, *130*, *132*, *134*, *136*, *138*, 139, *141*
transporters_polypeptide, *125*, *127*, *129*, *130*, 131, *134*, *136*, *138*, *140*, *141*
transporters_polypeptide_go, *125*, *127*, *129*, *130*, *132*, 133, *136*, *138*, *140*, *141*
transporters_polypeptide_pfams, *125*, *127*, *129*, *130*, *132*, *134*, 135, *138*, *140*, *141*
transporters_polypeptide_syn, *125*, *127*, *129*, *130*, *132*, *134*, *136*, 137, *140*, *141*
transporters_textbooks, *125*, *127*, *129*, *130*, *132*, *134*, *136*, *138*, *140*, 140