Title Integration of multiple microRNA-target databases with their disease and drug associations

Version 1.24.0

Description A collection of microRNAs/targets from external resources, including validated microRNA-target databases (miRecords, miRTarBase and TarBase), predicted microRNA-target databases (DIANA-microT, ElMMo, MicroCosm, miRanda, miRDB, PicTar, PITA and TargetScan) and microRNA-disease/drug databases (miR2Disease, Pharmaco-miR VerSe and PhenomiR).

URL https://github.com/KechrisLab/multiMiR

BugReports https://github.com/KechrisLab/multiMiR/issues

Depends R (>= 3.4)

Imports stats, XML, RCurl, purrr (>= 0.2.2), tibble (>= 1.2), methods, BiocGenerics, AnnotationDbi, dplyr,

Suggests BiocStyle, edgeR, knitr, rmarkdown, testthat (>= 1.0.2)

VignetteBuilder knitr

License MIT + file LICENSE

LazyData true

NeedsCompilation no

biocViews miRNAData, Homo_sapiens_Data, Mus_musculus_Data, Rattus_norvegicus_Data, OrganismData

RoxygenNote 6.0.1

Encoding UTF-8

git_url https://git.bioconductor.org/packages/multiMiR

git_branch RELEASE_3_18

git_last_commit 65060e9

git_last_commit_date 2023-10-24

Repository Bioconductor 3.18

Date/Publication 2024-04-03
Author  Yuanbin Ru [aut],
        Matt Mulvahill [cre, aut],
        Spencer Mahaffey [aut],
        Katerina Kechris [aut, cph, ths]

Maintainer  Matt Mulvahill <matt.mulvahill@gmail.com>

R topics documented:

add.multimir.links ........................................ 3
all_tables .................................................. 3
as.mmquery .................................................. 4
as_mmsql_components ...................................... 5
build_mmsql ............................................... 5
default_cutoff ........................................... 6
deprecate_arg ............................................ 6
extract_mmquery ........................................... 7
get.multimir.cutoffs ...................................... 7
get_multimir ............................................... 8
list_multimir .............................................. 10
mmquery_bioc-class ....................................... 11
multiMiR ................................................... 12
multimir.summary .......................................... 13
multimir_dbInfo .......................................... 13
multimir_switchDBVersion .................................. 15
null_to_df ................................................ 16
pad .......................................................... 16
parens_quote .............................................. 17
parens_wrap ............................................... 17
parse_orgs ............................................... 17
parse_response ........................................... 18
query_multimir ........................................... 18
quote_wrap ................................................ 18
remove_empty_strings ..................................... 19
remove_table .............................................. 19
search_multimir .......................................... 20
split_by ................................................... 21
sql_org ..................................................... 21
sql_validated ............................................. 22
submit_request ........................................... 22

Index 23
add.multimir.links

Add External Database Link for Each of the multiMiR Result Entry

Description
This is an internal multiMiR function that is not intended to be used directly. Please use get_multimir.

Usage
add.multimir.links(x, org)

Arguments
x  table/dataset returned by multimir db
org  Organism (see get_multimir)

Value
The input data frame x with a column added for the external database links.

all_tables  Functions defining the category each table belongs to.

Description
One of three types: predicted, validated, or diseasedrug. Additionally two functions define characteristics of tables: those without a target column tables_wo_target and those with conserved target sites conserved_tables.

Usage
all_tables()
validated_tables()
predicted_tables()
diseasedrug_tables()
tables_wo_target()
conserved_tables()
reverse_table_lookup(.table)
table_types()
Arguments

.table  a table name

Value

Returns dataset that names that belong to the category of the function name (e.g. validated_tables() returns tables with validated miRNA-target interactions). reverse_table_lookup() does the opposite; it returns the category a given .table belongs to.

Examples

all_tables()
validated_tables()
predicted_tables()
diseasedrug_tables()
predicted_tables() %in% all_tables() # TRUE
table_types()

---

as.mmquery  S3 constructor and methods for object returned by get_multimir().

Description

This package’s primary user-facing object. Contains the SQL statement and the returned data query, as well as a summary table depending on specified option.

Usage

as.mmquery(a_list)

## S3 method for class 'mmquery'
print(x)

Value

An mmquery object.
as_mmsql_components

Description

The collection object has a defined set of components that match the multiMiR database and defined options in `get_multimir()`. Conceptually this is split into two parts, the relatively straightforward SELECT, FROM, and ON portion of the query and the more complex filtering and sorting operations: WHERE and ORDER BY. The latter have their own classes, the former are resolved as strings (or character vectors) in the functions defining handling of each sql table (`sql_prefix`).

Usage

```r
as_mmsql_components(.select = NULL, .from = NULL, .on = NULL,
                      .where_list = NULL, .orderby = NULL, typeattr = NULL)
```

```r
as_where_list(...) 
```

```r
as_where(.vars, .connect = NULL, .operator, .value = "%s")
```

```r
is_where(x)
```

```r
as_orderby(.vars, .order)
```

Value

`as_mmsql_components`: A collection of components that make up a SQL query. `as_where_list`, `as_where`, `as_orderby`: Individual components of a SQL query.

build_mmsql

Constructors for parts of SQL queries Expand_query converts a mmyquery object to a SQL query string

Description

Constructors for parts of SQL queries Expand_query converts a mmyquery object to a SQL query string

Usage

```r
build_mmsql(.table, org, mirna = NULL, target = NULL, disease.drug = NULL,
predicted.site = NULL, predicted.cutoff.type = NULL,
predicted.cutoff = NULL, limit = NULL)
```
expand_query(x)
expand_select(x)
expand_from(x)
expand_on(x)
expand_where_list(x)
expand_where(x)
expand_orderby(x)
expand_limit(x)
merge_order(list)

Value
A complete SQL statement and related information.

| default_cutoff | If null, set default predicted.cutoff |

Description
If null, set default predicted.cutoff

Usage
default_cutoff(predicted.cutoff.type, predicted.cutoff)

Value
The default cutoff value.

depricate_arg | Internal function for sending deprecation messages |

Description
Internal function for sending deprecation messages

Usage
depricate_arg(name = c("url", "schema.file", "db.tables", "cutoff.file"))
**extract_mmquery**

**Arguments**

- **name**  
  Name of a deprecated function argument.

**Value**

A message indicating deprecated arg and new version.

```r
extract_mmquery(outlist, org, .args, summary = FALSE, use.tibble = FALSE)
```

**Description**

Creates all objects needed for the legacy S3 return object and the new S4 object.

**get.multimir.cutoffs**  
*Load Pre-calculated Prediction Score Cutoffs in the multiMiR Package*

**Description**

This is an internal multiMiR function that is not intended to be used directly. Please set prediction score cutoff in `get_multimir`.

```r
get.multimir.cutoffs(name = NULL, cutoff.file = NULL)
```

**Arguments**

- **cutoff.file**  
  Deprecated. Set path to cutoffs file with the global option `multimir.cutoffs`.

**Value**

Cutoff values object from remote database.
get_multimir 

Get microRNA-target Interactions from the multiMiR Package

Description

The main function to retrieve predicted and validated miRNA-target interactions and their disease and drug associations from the multiMiR package.

Usage

```r
get_multimir(url = NULL, org = "hsa", mirna = NULL, target = NULL,
  disease.drug = NULL, table = "validated", predicted.cutoff = NULL,
  predicted.cutoff.type = "p", predicted.site = "conserved",
  summary = FALSE, add.link = FALSE, use.tibble = FALSE, limit = NULL,
  legacy.out = FALSE)
```

Arguments

- `url`: Deprecated. The URL for queries is now defined by the package options `multimir.url` and `multimir.queries`.
- `org`: a character string for the organism. Three organisms are supported so far: human ("hsa" (default), "human", or "Homo Sapiens"), mouse ("mmu", "mouse", or "Mus musculus"), and rat ("rno", "rat", or "Rattus norvegicus"). The organism is case insensitive.
- `mirna`: 'NULL' (default) or a character string or character vector for the mature miRNA(s). It can be the mature miRNA accession number (i.e. "MIMAT0000072"), mature miRNA ID (i.e. "hsa-miR-199a-3p"), or a combination of both (i.e. c("MIMAT0000065", "hsa-miR-30a-5p")). The character is case insensitive. *See note about the length of list supported.
- `target`: 'NULL' (default) or a character string or character vector for the target gene(s). It can be the gene symbol (i.e. c("TP53", "KRAS"), Entrez gene ID (i.e. c(578, 3845)), Ensembl gene ID (i.e. "ENSG00000171791"), or a combination of any of these identifiers (i.e. c("TP53", 3845, "ENSG00000171791")). The character is case insensitive. *See note about the length of list supported.
- `disease.drug`: 'NULL' (default) or a character string or character vector for the disease(s) and/or drug(s) (i.e. c("bladder cancer", "cisplatin")). The character is case insensitive.
- `table`: a character string indicating which table(s) in multiMiR to search. Each table contains data from an external database. Options include "validated" (default, to search all validated tables "mirecords", "mirtarbase", and "tarbase"), "predicted" (to search all predicted tables "diana_microt", "elmmo", "microcosm", "mirtarbase", and "tarbase"), "all" (to search all tables). *See note about the length of list supported.
- `predicted.cutoff`: the minimum p-value for validated predictions. The default is 0.05.
- `predicted.cutoff.type`: the type of p-value cutoff. The default is "p" (p-value).
- `predicted.site`: the site of interaction. The default is "conserved".
- `summary`: a logical argument determining whether to return a summary of the results. The default is FALSE.
- `add.link`: a logical argument determining whether to add links to the source data. The default is FALSE.
- `use.tibble`: a logical argument determining whether to return the data in a tibble format. The default is FALSE.
- `limit`: the number of results to return. The default is NULL.
- `legacy.out`: a logical argument determining whether to use the legacy output format. The default is FALSE.

Notes:

- The length of the list supported is up to 10,000 entries.

Examples

```r
get_multimir(url = NULL, org = "hsa", mirna = NULL, target = NULL,
  disease.drug = NULL, table = "validated", predicted.cutoff = NULL,
  predicted.cutoff.type = "p", predicted.site = "conserved",
  summary = FALSE, add.link = FALSE, use.tibble = FALSE, limit = NULL,
  legacy.out = FALSE)
```
get_multimir

get_multimir() has been deprecated and replaced with the get_multimir() version.

get_multimir is the main and recommended function to retrieve information from the multiMiR package. Input to the function must contain at least one of the followings: miRNA(s), target gene(s), and disease and drug term(s).

The setting of predicted.site is applicable to three ("miranda", "pita", and "targetscan") of the eight predicted tables. If predicted.site is "conserved", the function will search conserved target sites annotated by TargetScan, target sites with conservation scores greater than or equal to 0.57 (in human and rat; or 0.566 in mouse) in miRanda, and/or sites with conservation scores greater than or equal to 0.9 in PITA.

Although the summary (if summary=TRUE) can be used to find results that are recorded by combinations of different databases, please note that for predicted interactions a combination approach may not be as effective as a single algorithm because of age or quality of the tool.

Note: The length of the list supported has been increased from version 1.0.1. The size is now limited to 20MB which should accommodate most requests. There is a possibility for technical reasons that the query could fail even if the list is under this limit. If this occurs it is recommended that you break up the list into smaller batches and submit them sequentially.
list_multimir

List microRNAs, Genes, Drugs Or Diseases in the multiMiR Package

Description

list_multimir lists all the unique microRNAs, target genes, drugs, or diseases in the web server of the multiMiR package.

Usage

list_multimir(x = c("mirna", "gene", "drug", "disease"), limit = NULL, url = NULL)

Arguments

- **x**: a character string indicating what to list. This must be one of the strings "mirna" (default), "gene", "drug", or "disease". This can be abbreviated and is case insensitive.
- **limit**: a positive integer. Limits the number of records returned from each table. Useful in testing potentially large queries.
- **url**: Deprecated. Use global option `multimir.url` instead.
Details

`list.multimir()` has been deprecated and replaced with the `list_multimir()` version.

Value

`list_multimir` returns a data frame with information of microRNAs (microRNA unique ID, organism, mature microRNA accession number, and mature microRNA ID), target genes (gene unique ID, organism, gene symbol, Entrez gene ID, and Ensembl gene ID), drugs (drug names), and diseases (disease name).

Author(s)

Yuanbin Ru <ruyuanbin@gmail.com>

Examples

```r
miRNAs <- list_multimir("mirna", limit = 10)
genes <- list_multimir("gene", limit = 10)
drugs <- list_multimir("drug", limit = 10)
diseases <- list_multimir("disease", limit = 10)
```

Description

This package’s primary user-facing object. Contains the SQL statement and the returned data query, as well as a summary table depending on specified option. Note that the returned data is now contained in a single dataframe. To filter to a specific type of association or interaction, select on the type variable.

Usage

```r
as.mmquery_bioc(.list)
## S4 method for signature 'mmquery_bioc'
columns(x)

## S4 method for signature 'mmquery_bioc'
keys(x, keytype, ...)

## S4 method for signature 'mmquery_bioc'
keytypes(x)

## S4 method for signature 'mmquery_bioc'
select(x, keys, columns, keytype, ...)

## S4 method for signature 'mmquery_bioc'
show(object)
```
Arguments

- `.list` a list of returned dataframes, summary
- `x`, `object` An mmquery_bioc object.
- `keytype` allows the user to discover which keytypes can be passed in to select or keys and the keytype argument
- `...` additional arguments
- `keys` A result of the keys() function. For the mmquery_bioc class this is a character vector of microRNA's in the returned mmquery_bioc object.
- `columns` lists the columns that can be returned for the mmquery_bioc object.

Value

an s4 object of class mmquery_bioc. Contains queried data, a summary dataset, and associated input parameters.

Slots

data  A dataframe containing validated and predicted microRNA-target interactions and disease/drug associations found.

queries  A list of queries submitted to the multiMiR SQL server.

summary  A summary dataframe of the returned microRNA dataframes

tables  A character vector of the microRNA relationship types returned (validated, predicted, disease.drug, or all).

org  The selected organism (hsa/human, mmu/mouse, rno/rat).

predicted.cutoff  An integer giving a prediction score cutoff.

predicted.cutoff.type  A character indicating the type of prediction score cutoff (p = percentage, n = number, character() = none)

predicted.site  A character string indicating the type of predicted target sites to searched.

Description

This package provides an interface to the multiMiR database of microRNA-target interactions, and disease and drug associations. See http://multimir.org and the vignette ('multiMiR') for more details.

References

[Add reference here]
Summarize microRNA/target Information from the multiMiR Package

Description

This is an internal multiMiR function that is not intended to be used directly. Please use \texttt{get\_multimir}.

Usage

\texttt{multimir\_summary(result, pair.index = 2:6, order.by = "all.sum")}

Arguments

- \texttt{result}
- \texttt{pair.index}
- \texttt{order.by}

Value

Summary of objects queries from database

---

\texttt{multimir\_dbInfo}

Collect Information About the Web Server And Database of the multiMiR Package

---

Description

Functions for collecting and displaying information about the web server and database of the multiMiR package.

Usage

\texttt{multimir\_dbInfo(url = NULL)}

\texttt{multimir\_dbInfoVersions(url = NULL)}

\texttt{multimir\_dbSchema(schema.file = NULL)}

\texttt{multimir\_dbTables(url = NULL)}

\texttt{multimir\_dbCount(url = NULL)}
Arguments

url                  Deprecated. Use global option multimir.url instead.
schema.file          Deprecated. Option exists as multimir.schema, but it should not need to be set directly.

Details

multimir.url is a global option containing the URL of the multiMiR web server. Set using
options("multimir.url" = ...)

multimir_dbCount returns counts of records in the tables in the multiMiR database. Each table
contains data from an external miRNA/target database.
multimir_dbInfo returns other information about the multiMiR database. This includes informa-
tion of external miRNA/target databases in multiMiR.
multimir_dbInfoVersions returns other information about the multiMiR database versions avail-
able. This provides a list of available options if switching to previous version is desired.
multimir_dbSchema prints the schema definition of the multiMiR database.
multimir_dbTables returns the vector of tables in the multiMiR database and saves it to the global
option multimir.tables.list. This function is automatically run when get_multimir is called if
the multimir.tables.list is NULL.

Value

multimir_dbCount: a data frame with the count of records in each of the tables in the multiMiR
database.
multimir_dbInfo: a data frame with information about the multiMiR database.
multimir_dbInfoVersions: a data frame with information about the multiMiR database versions.
multimir_dbSchema: none (invisible NULL).
multimir_dbTables: a data frame with table names in the multiMiR database.

Examples

this_url <- getOption("multimir.url")
this_url
options(multimir.url = this_url)
db_ver <- multimir_dbInfoVersions()
db_count <- multimir_dbCount()
db_info <- multimir_dbInfo()
multimir_dbSchema()
db_tables <- multimir_dbTables()
multimir_switchDBVersion

Manage Database Version to use

Description

Functions for managing the database version used to complete requests on the web server.

Usage

```
multimir_switchDBVersion(db_version, url = NULL)
```

Arguments

- `db_version`: A character string containing the full version number for the database version to use for all package functions. The default will be the most recent version.
- `url`: Deprecated. Use global option `multimir.url` instead.

Details

`url` is a character string containing the URL of the multiMiR web server. Optional as it is set when the package is loaded.

`multimir_dbInfoVersions` returns other information about the multiMiR database versions available. This provides a list of available options if switching to previous version is desired.

`multimir_switchDBVersion` returns other information about the multiMiR database versions available. This provides a list of available options if switching to previous version is desired.

Value

- `multimir_dbInfoVersions`: a data frame with information about the multiMiR database versions.
- `multimir_switchDBVersion`: none (invisible `NULL`).

Examples

```
multimir_dbInfoVersions()
multimir_switchDBVersion(db_version="2.0.0")
```
null_to_df

Replace nulls with an empty object of each type

Usage
null_to_df(x)
null_to_num(x)
null_to_char(x)

Arguments
x
input object

Value
an empty data.frame, numeric, or character vector.

pad

Pad single space on each side of an input

Usage
pad(x)

Value
Input value wrapped in single spaces.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Usage</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>parens_quote</td>
<td>Prep certain names for use in SQL query by adding parens</td>
<td>parens_quote(x)</td>
<td>The input value wrapped in quotes and then parentheses.</td>
</tr>
<tr>
<td>parens_wrap</td>
<td>Collapse a vector to a single comma-separated string and wrap in parentheses</td>
<td>parens_wrap(x)</td>
<td>The input vector converted to a comma-separated string wrapped in parentheses.</td>
</tr>
<tr>
<td>parse_orgs</td>
<td>Each org can be specified in one of 3 ways – this standardizes the argument into the 3 char abbreviation.</td>
<td>parse_orgs(org)</td>
<td>A standardized, abbreviated form of the input org.</td>
</tr>
</tbody>
</table>
quote_wrap

---

**parse_response**

*Parse the Result Returned by the multiMiR Web Server*

**Description**

This is an internal multiMiR function that is not intended to be used directly. Please use `get_multimir`.

**Usage**

```
parse_response(HTML.response)
```

**Value**

The queried table portion of the HTML response.

---

**query_multimir**

*Wrapper for search_multimir for adding feature (printing notification to console)*

**Description**

Wrapper for `search_multimir` for adding feature (printing notification to console)

**Usage**

```
query_multimir(x, org, add.link, use.tibble)
```

**Value**

The queried multimir data with the addition of a requested feature.

---

**quote_wrap**

*Internal function for adding single quotes around a string*

**Description**

Internal function for adding single quotes around a string

**Usage**

```
quote_wrap(x)
```

**Arguments**

```
x
```

a string to be wrapped in single quotes.
Value

The input wrapped in single quotes.

remove_empty_strings  Remove empty strings from character vector.

Description

The WHERE clauses for target and mirna use allow for multiple arguments always separated by 'OR' and several columns are checked for each value (mirna id, acc; target symbol, entrez, ensemble). If empty strings "" are present in the get_multimir arguments, Targets and miRNA with empty values in one of these columns will be incorrectly returned. – thus purge empty strings first.

Usage

remove_empty_strings(x)

Arguments

x  A character vector

Value

A character vector with empty strings removed

remove_table  Remove tables x from a vector of table names.

Description

Typically used when a set of arguments don’t apply to a table or would return an error/empty response

Usage

remove_table(tables, x)

Arguments

tables  A character vector.

x  A second character vector to remove from the first (tables).

Value

Character vector tables excluding the strings matching those in x.
search_multimir  

Search the multiMiR Database Given a MySQL Query

Description

This is a function for directly querying the multiMiR database with MySQL queries. Given a MySQL query, it searches and retrieves results from the multiMiR database on the multiMiR web server. To use search_multimir directly, users will need to be familiar with MySQL and multiMiR table structures. Users are advised to use get_multimir instead.

Usage

search_multimir(query)

search.multimir(query)

Arguments

query  
a character string for the MySQL query.

Details

search.multimir() has been deprecated and replaced with the search_multimir() version.

Value

search_multimir returns a data frame containing results from the multiMiR web server.

Examples

## show all tables in the multiMiR database
tables <- search_multimir(query="show tables")

## show the structure of table diana_microt
microt <- search_multimir(query="describe diana_microt")

## search for validated target genes of hsa-miR-18a-3p in miRecords
qry <- paste("SELECT m.mature_mirna_acc, m.mature_mirna_id,",
" t.target_symbol, t.target_entrez, t.target_ensembl,",
" i.experiment, i.support_type, i.pubmed_id",
" FROM mirna AS m INNER JOIN mirecords AS i INNER JOIN target",
" AS t ON (m.mature_mirna_uid=i.mature_mirna_uid AND",
" i.target_uid=t.target_uid),"
" WHERE m.mature_mirna_id='hsa-miR-18a-3p'")
result <- search_multimir(query = qry)
split_by

Split, order and sort lists by their components.

Description
Copied from purrr:v0.2.2

Usage
split_by(.x, .f, ...)

Arguments
- .x: A list or atomic vector.
- .f: A function, formula, or atomic vector.
- ...: Additional arguments passed on to .f.

Value
A list split by .f

sql_org

Functions defining the WHERE clauses.

Description
Functions defining filtering by organism (org), disease/drug, conserved, and cutoff. Filtering by mirna and target are defined within their sql_... functions.

Usage
sql_org(.table, org)

where_org(.table, org)

where_diseasedrug(.table, disease.drug)

where_conserved(.table, org, predicted.site)

where_cutoff(.table, score_var, score_cutoff)

create_cutoff_name(.table, org, predicted.site)

cutoff_to_score(.table, cutoff_name, predicted.cutoff.type, predicted.cutoff)
Value

The \texttt{WHERE} portion of a SQL query

\begin{itemize}
  \item \texttt{sql_validated} \textit{Generate mmsql\_components objects for each of the three types of tables, as well as the mirna and target tables.}
\end{itemize}

\section*{Description}

The three types of tables are predicted, validated, and diseasedrug (disease/drug). Additionally, mirna and target portions of the SQL statements are defined, including their filter clauses (\texttt{WHERE}).

\section*{Usage}

\begin{itemize}
  \item \texttt{sql_validated}(\texttt{.table})
  \item \texttt{sql_predicted}(\texttt{.table, org, predicted.site, predicted.cutoff.type, predicted.cutoff})
  \item \texttt{sql_diseasedrug}(\texttt{.table, disease.drug})
  \item \texttt{sql_mirna}(\texttt{mirna})
  \item \texttt{sql_target}(\texttt{.table, target})
\end{itemize}

\section*{Value}

Components of a SQL query specific to each table type.

\begin{itemize}
  \item \texttt{submit_request} \textit{General workhorse function for submitting and returning queries}
\end{itemize}

\section*{Description}

This is an internal multiMiR function that is not intended to be used directly. Please use \texttt{get\_multimir}.

\section*{Usage}

\begin{itemize}
  \item \texttt{submit\_request(url = full\_url("multimir.queries"), query, ...)}
\end{itemize}

\section*{Value}

Table requested in query.
Index

* database
  get_multimir, 8
  list_multimir, 10
  multimir_dbInfo, 13
  multimir_switchDBVersion, 15
  search_multimir, 20
* diseasedrug
  sql_org, 21
  sql_validated, 22
* disease
  sql_org, 21
  sql_validated, 22
* drug
  sql_org, 21
  sql_validated, 22
* internal
  add.multimir.links, 3
  as.mmquery, 4
  as_mmsql_components, 5
  build_mmsql, 5
  default_cutoff, 6
  deprecate_arg, 6
  extract_mmquery, 7
  get.multimir.cutoffs, 7
  multimir.summary, 13
  null_to_df, 16
  pad, 16
  parens_quote, 17
  parens_wrap, 17
  parse_orgs, 17
  parse_response, 18
  query_multimir, 18
  quote_wrap, 18
  remove_empty_strings, 19
  remove_table, 19
  split_by, 21
  sql_org, 21
  sql_validated, 22
  submit_request, 22
* predicted
  sql_org, 21
  sql_validated, 22
* tables
  all_tables, 3
  sql_org, 21
  sql_validated, 22
* types
  sql_org, 21
  sql_validated, 22
* validated
  sql_org, 21
  sql_validated, 22

add.multimir.links, 3
all_tables, 3
all_tables, (all_tables), 3
as.mmquery, 4
as.mmquery_bioc (mmquery_bioc-class), 11
as_mmsql_components, 5
as_mmsql_components,
  (as_mmsql_components), 5
as_orderby (as_mmsql_components), 5
as_orderby, (as_mmsql_components), 5
as_where (as_mmsql_components), 5
as_where, (as_mmsql_components), 5
as_where_list (as_mmsql_components), 5
as_where_list, (as_mmsql_components), 5
build_mmsql, 5

columns_mmquery_bioc-method
  (mmquery_bioc-class), 11
conserved_tables (all_tables), 3
create_cutoff_name (sql_org), 21
create_score (sql_org), 21

default_cutoff, 6
deprecate_arg, 6
diseasedrug_tables (all_tables), 3
diseasedrug_tables, (all_tables), 3
expand_from (build_mmsql), 5
expand_limit (build_mmsql), 5
expand_on (build_mmsql), 5
expand_orderby (build_mmsql), 5
expand_query (build_mmsql), 5
expand_select (build_mmsql), 5
expand_where (build_mmsql), 5
expand_where_list (build_mmsql), 5
extract_mmquery, 7
get_multimir (get_multimir), 8
gget_multimir, 8
is_where (as_mmsql_components), 5
is_where_list (as_mmsql_components), 5
keys, mmquery.bioc-method
(mmmquery.bioc-class), 11
keytypes, mmquery.bioc-method
(mmmquery.bioc-class), 11
list.multimir (list_multimir), 10
list_multimir, 10
merge_order (build_mmsql), 5
mmquery.bioc-class, 11
multiMiR, 12
multiMiR-package (multiMiR), 12
multiMiR-summary, 13
multiMiR-dbCount (multiMiR-dbInfo), 13
multiMiR-dbInfo, 13
multiMiR-dbInfoVersions
(multiMiR-dbInfo), 13
multiMiR-dbSchema (multiMiR-dbInfo), 13
multiMiR-dbTables (multiMiR-dbInfo), 13
multiMiR-switchDBVersion, 15

null_to_char (null_to_df), 16
null_to_df, 16
null_to_num (null_to_df), 16

pad, 16
parens_quote, 17
parens_wrap, 17
parse_orgs, 17
parse_response, 18
predicted_tables (all_tables), 3
predicted_tables, (all_tables), 3
print.mmquery (as.mmquery), 4
query_multimir, 18
quote_wrap, 18

remove_empty_strings, 19
remove_table, 19
reverse_table_lookup (all_tables), 3
search.multimir (search_multimir), 20
search_multimir, 20
select, mmquery.bioc-method
(mmmquery.bioc-class), 11
show, mmquery.bioc-method
(mmmquery.bioc-class), 11
split_by, 21
sql_diseasedrug (sql_validated), 22
sql_mirror (sql_validated), 22
sql.org, 21
sql.org, (sql.org), 21
sql_predicted (sql_validated), 22
sql_target (sql_validated), 22
sql_validated, 22
submit_request, 22

table_types (all_tables), 3
tables_wo_target (all_tables), 3
tables_wo_target, (all_tables), 3

validated_tables (all_tables), 3
validated_tables, (all_tables), 3

where_conserved (sql_org), 21
where_conserved, (sql_org), 21
where_cutoff (sql_org), 21
where_diseasedrug (sql_org), 21
where_diseasedrug, (sql_org), 21
where_org (sql_org), 21
where_org, (sql_org), 21