Package 'rPDBapi'

May 16, 2024

Type Package

Title A Comprehensive Interface for Accessing the Protein Data Bank

Version 1.2

Description Streamlines the interaction with the RCSB Protein Data Bank (PDB) <https://www.rcsb.org/>. This interface offers an intuitive and
powerful tool for searching and retrieving a diverse range of data types from the PDB. It includes advanced functionalities like
BLAST and sequence motif queries. Built upon the existing XML-based API of the PDB, it simplifies the creation of custom requests,
thereby enhancing usability and flexibility for researchers.

License GPL (>= 2)

Encoding UTF-8

Imports dplyr, purrr, httr, jsonlite, xml2, bio3d, magrittr, methods

RoxygenNote 7.3.1

NeedsCompilation no

Author Selcuk Korkmaz [aut, cre] (<https://orcid.org/0000-0003-4632-6850>),
Bilge Eren Yamasan [aut] (<https://orcid.org/0000-0002-6525-2503>)

Maintainer Selcuk Korkmaz <selcukorkmaz@gmail.com>

Repository CRAN

Date/Publication 2024-05-16 15:00:06 UTC

R topics documented:

add_property 2
autoresolve_sequence_type 3
ChemicalOperator 3
ComparisonOperator 4
ContainsPhraseOperator 4
ContainsWordsOperator 5
data_fetcher 5
DefaultOperator 6
describe_chemical 6

ExactMatchOperator 7
ExistsOperator 7
FastaSequence 8
fetch_data 9
find_papers 9
find_results 10
generate_json_query 11
get_fasta_from_rcsb_entry 11
get_info 12
get_pdb_file 13
infer_search_service 14
InOperator 14
parse_fasta_text_to_list 15
perform_search 15
QueryGroup 16
QueryNode 17
query_search 17
RangeOperator 18
RequestOptions 19
return_data_as_dataframe 20
ScoredResult 20
search_graphql 21
SeqMotifOperator 21
SequenceOperator 22
StructureOperator 23
walk_nested_dict 23

Index **25**

`add_property` *Add or Merge Properties for RCSB PDB Data Fetching*

Description

This function is designed for handling properties related to fetching data from the Protein Data Bank (PDB). It takes a dictionary, where keys represent properties and values are lists of subproperties. If a property already exists in the input list, the function merges the subproperties, ensuring each subproperty is unique and maintains character vector format.

Usage

```
add_property(property)
```

Arguments

`property`	A dictionary where keys are the properties (like 'cell', 'exptl') and values are lists of subproperties (like 'volume', 'angle_beta', 'method'). Each subproperty should be in character vector format. Full list of properties can be found at https://data.rcsb.org/#data-schema.

Value

A modified list with updated properties where subproperties are merged if a property already exists.

Examples

```
properties <- list(cell = c("length_a", "length_b", "length_c"), exptl = c("method"))
add_property(properties)
```

`autoresolve_sequence_type` *Automatically Determine the Sequence Type*

Description

This function determines the type of a given sequence (DNA, RNA, or PROTEIN) based on its characters.

Usage

```
autoresolve_sequence_type(sequence)
```

Arguments

`sequence`	A string representing the sequence to be analyzed.

Value

A string indicating the resolved sequence type.

`ChemicalOperator` *Chemical Operator for SMILES/InChI Searches*

Description

Creates an object representing a chemical search operator using SMILES or InChI descriptors.

Usage

```
ChemicalOperator(descriptor, matching_criterion = "graph-strict")
```

Arguments

`descriptor`	A valid SMILES or InChI string.
`matching_criterion`	The criterion for matching, one of the values from DescriptorMatchingCriterion. Defaults to "graph-strict".

Value

A list representing a chemical operator.

`ComparisonOperator` *Create a Comparison Search Operator*

Description

Constructs a ComparisonOperator object for search operations that perform comparison checks on attribute values.

Usage

```
ComparisonOperator(attribute, value, comparison_type)
```

Arguments

`attribute`	The attribute to be compared.
`value`	The value to compare against.
`comparison_type`	A string specifying the type of comparison (e.g., 'equal', 'greater_than').

Value

An object of class 'ComparisonOperator'.

`ContainsPhraseOperator`
Create a Contains Phrase Search Operator

Description

Constructs a ContainsPhraseOperator object for search operations that look for attributes containing a specific phrase.

Usage

```
ContainsPhraseOperator(attribute, value)
```

Arguments

`attribute`	The attribute to be evaluated.
`value`	The phrase to search for in the attribute.

Value

An object of class 'ContainsPhraseOperator'.

ContainsWordsOperator *Create a Contains Words Search Operator*

Description

Constructs a ContainsWordsOperator object for search operations that look for attributes containing certain words.

Usage

```
ContainsWordsOperator(attribute, value)
```

Arguments

attribute	The attribute to be evaluated.
value	The words to search for in the attribute.

Value

An object of class 'ContainsWordsOperator'.

data_fetcher *Fetch RCSB PDB Data Based on Specified Criteria*

Description

This function fetches data based on a given identifier (ID), data type, and a set of properties. It can return the data either in its original format or as a dataframe. The function integrates several steps including validating IDs, generating a JSON query, fetching the data, and formatting the response.

Usage

```
data_fetcher(
  id = NULL,
  data_type = "ENTRY",
  properties = NULL,
  return_as_dataframe = TRUE
)
```

Arguments

id	An identifier or a list of identifiers for the data to be fetched.
data_type	A string specifying the type of data to fetch. Default is "ENTRY".
properties	A list or dictionary of properties to be included in the data fetching process.
return_as_dataframe	A boolean indicating whether to return the response as a dataframe. Default is TRUE.

Value

Depending on the value of 'return_as_dataframe', this function returns either a dataframe or data in its original format.

Examples

```
properties <- list(cell = c("length_a", "length_b", "length_c"), exptl = c("method"))
data_fetcher(
id = c("4HHB"),
data_type = "ENTRY",
properties = properties,
return_as_dataframe = TRUE
)
```

DefaultOperator *Create a Default Search Operator*

Description

Constructs a DefaultOperator object for use in general search operations within the RCSB PDB. This operator is a basic search operator with a single value.

Usage

```
DefaultOperator(value)
```

Arguments

value	The value to be used in the search operation.

Value

An object of class 'DefaultOperator' representing the default search operator.

---

describe_chemical *Describe Chemical Compound from RCSB PDB*

Description

This function retrieves the description of a chemical compound from the RCSB PDB based on its ID.

Usage

```
describe_chemical(chem_id)
```

Arguments

chem_id	A string representing the 3-character chemical ID.

Value

A dictionary containing the chemical description.

Examples

```
chem_desc <- describe_chemical('NAG')
chem_desc$rcsb_chem_comp_descriptor$smiles
```

ExactMatchOperator *Create an Exact Match Search Operator*

Description

Constructs an ExactMatchOperator object for precise search operations in the RCSB PDB. It matches an exact attribute value.

Usage

```
ExactMatchOperator(attribute, value)
```

Arguments

attribute	The attribute to match.
value	The exact value to search for.

Value

An object of class 'ExactMatchOperator'.

ExistsOperator *Create an Existence Search Operator*

Description

Constructs an ExistsOperator object for search operations to check the existence of an attribute.

Usage

```
ExistsOperator(attribute)
```

Arguments

attribute	The attribute whose existence is to be checked.

Value

An object of class 'ExistsOperator'.

FastaSequence *Create a FASTA Sequence Data Structure*

Description

This function constructs a FASTA sequence data structure from given parameters. It is typically used to format and store FASTA sequence information.

Usage

```
FastaSequence(entity_id, chains, sequence, fasta_header)
```

Arguments

entity_id	A string representing the entity ID of the sequence.
chains	A character vector of chain identifiers associated with the sequence.
sequence	A string representing the nucleotide or amino acid sequence.
fasta_header	A string representing the header of the FASTA sequence.

Value

A list representing the FASTA sequence, including entity ID, chains, the sequence itself, and the FASTA header.

Examples

```
fasta_data <- FastaSequence("1XYZ", c("A", "B"), "MVLSPADKT", "header_info")
```

`fetch_data` *Fetch Data from RCSB PDB Using a JSON Query*

Description

This function sends a JSON query to the RCSB Protein Data Bank (PDB) and fetches the corresponding data. It checks for errors in the response and warns if there are discrepancies in the number of IDs found.

Usage

```
fetch_data(json_query, data_type, ids)
```

Arguments

`json_query`	A JSON string representing the query to be sent to the PDB.
`data_type`	A string indicating the type of data to be fetched (not directly used in the function but may be relevant for context).
`ids`	A vector of identifiers to fetch data for.

Value

A list containing the data fetched from the PDB, with the names of the list elements set to the corresponding IDs. If an error is encountered in the data fetching process, the function returns 'NULL'.

`find_papers` *Search for and Retrieve Paper Titles from PDB*

Description

This function searches for papers in the Protein Data Bank (PDB) using a specified search term. It retrieves the titles of papers up to a specified maximum number of results. The function assumes the presence of 'query_search' and 'get_info' functions to perform the search and fetch paper details.

Usage

```
find_papers(search_term, max_results = 10)
```

Arguments

`search_term`	A string specifying the term to search for in the PDB.
`max_results`	An integer indicating the maximum number of paper titles to retrieve. Defaults to 10.

Value

A named list where each element's name is a PDB ID and its value is the title of the corresponding paper.

Examples

```
find_papers("CRISPR")
```

`find_results` *Retrieve Specific Fields for Search Results from RCSB PDB*

Description

This function performs a search in the Protein Data Bank (PDB) using a provided search term and retrieves information for a specified field (e.g., citation) for each search result. It relies on 'query_search' and 'get_info' functions for searching and retrieving detailed information.

Usage

```
find_results(search_term, field = "citation")
```

Arguments

`search_term`	A string specifying the term to search for in the PDB.
`field`	A string indicating the specific field to retrieve for each search result. Default is "citation". Other options are 'audit_author', 'cell', 'diffrn', 'diffrn_detector', 'diffrn_radiation', 'diffrn_source', 'entry', 'exptl', 'exptl_crystal', 'exptl_crystal_grow', 'pdbx_sgproject', 'pdbx_audit_revision_details', 'pdbx_audit_revision_history', 'pdbx_database_related', 'pdbx_database_status', 'rcsb_accession_info', 'rcsb_entry_container_identifie 'rcsb_entry_info', 'rcsb_primary_citation', 'refine', 'refine_hist', 'refine_ls_restr', 'reflns', 'reflns_shell', 'software', 'struct', 'struct_keywords', 'symmetry', 'rcsb_id'

Value

A named list where each element's name is a PDB ID and its value is the information for the specified field from the corresponding search result.

Examples

```
find_results("crispr", field = "citation")
```

generate_json_query *Generate a JSON Query for RCSB PDB Data Retrieval*

Description

This function constructs a JSON query for retrieving data from the RCSB Protein Data Bank (PDB). It requires input parameters like IDs, data type, and properties to tailor the query for specific data retrieval needs.

Usage

```
generate_json_query(ids, data_type, properties)
```

Arguments

ids	A vector of identifiers for which data needs to be retrieved.
data_type	A string indicating the type of data to be queried, such as 'ENTRY', 'POLYMER_ENTITY', etc.
properties	A list of properties to be included in the query. Each element of the list should be a character vector representing properties for the respective data type.

Value

A string representing the generated JSON query formatted for PDB data retrieval.

Examples

```
ids <- c("1XYZ", "2XYZ")
properties <- list(cell = c("volume", "angle_beta"), exptl = c("method"))
json_query <- generate_json_query(ids, "ENTRY", properties)
json_query
```

---

get_fasta_from_rcsb_entry *Retrieve FASTA Sequences from PDB Entry*

Description

This function fetches FASTA sequences from the RCSB Protein Data Bank (PDB) for a specified entry ID. It sends an HTTP request to the PDB and processes the response to extract FASTA sequences.

Usage

```
get_fasta_from_rcsb_entry(rcsb_id, verbosity = TRUE)
```

Arguments

rcsb_id	A string representing the PDB ID for which the FASTA sequence is to be retrieved.
verbosity	A boolean flag indicating whether to print status messages during the function execution. Defaults to TRUE.

Value

A list of FASTA sequences associated with the provided RCSB entry ID.

Examples

```
get_fasta_from_rcsb_entry(c("4HHB"), verbosity = TRUE)
```

---

get_info *Retrieve Information for a Given PDB ID*

Description

This function looks up all information about a given PDB ID using the REST API. It handles JSON data and HTTP requests and converts old entry identifiers.

Usage

```
get_info(pdb_id, url_root = "https://data.rcsb.org/rest/v1/core/entry/")
```

Arguments

pdb_id	A 4-character string specifying a PDB entry of interest.
url_root	The root URL for the specific request type. Default is 'https://data.rcsb.org/rest/v1/core/entry/'.

Value

An ordered dictionary (list in R) object corresponding to entry information. Returns NULL if retrieval fails.

Examples

```
get_info(pdb_id = "4HHB")
```

get_pdb_file *Download PDB Files from RCSB Database*

Description

This function downloads PDB files from the RCSB database, supporting different file types and optional compression. It warns to consider compression for CIF files for faster download.

Usage

```
get_pdb_file(
  pdb_id,
  filetype = "cif",
  rm.insert = FALSE,
  rm.alt = TRUE,
  compression = TRUE,
  save = FALSE,
  path = NULL
)
```

Arguments

pdb_id	A 4-character string specifying a PDB entry of interest.
filetype	The file type: 'pdb', 'cif', 'xml', or 'structfact'. Default is 'pdb'.pdb is the older file format and cif is the newer replacement. xlm is also can be obtained and parsed. structfact retrieves structure factors (only available for certain PDB entries).
rm.insert	Logical, if TRUE PDB insert records are ignored.
rm.alt	Logical, if TRUE PDB alternate records are ignored.
compression	Logical indicating whether to request the data as a compressed version. Default is TRUE
save	Logical, if TRUE saves PDB file to the desired path.
path	The path where the file should be saved. If NULL, the file is saved in a temporary directory.

Value

Returns a list of class "pdb" with the following components:

atom	a data.frame containing all atomic coordinate ATOM and HETATM data, with a row per ATOM/HETATM and a column per record type.
xyz	a numeric matrix of class "xyz" containing the ATOM and HETATM coordinate data.
calpha	logical vector with length equal to nrow(atom) with TRUE values indicating a C-alpha "elety".
call	the matched call.

Examples

```
pdb_file <-  get_pdb_file(pdb_id = "4HHB", filetype = "cif")
```

`infer_search_service` *Infer the Appropriate Search Service*

Description

Determines the appropriate search service for a given search operator in RCSB PDB queries.

Usage

```
infer_search_service(search_operator)
```

Arguments

`search_operator`	A query operator object.

Value

The inferred search service.

`InOperator` *Create an Inclusion Search Operator*

Description

Constructs an InOperator object for search operations where the attribute value must be within a specified set.

Usage

```
InOperator(attribute, value)
```

Arguments

`attribute`	The attribute to be evaluated.
`value`	The set of values to include in the search.

Value

An object of class 'InOperator'.

parse_fasta_text_to_list *Parse Raw FASTA Text into a List of Sequences*

Description

This function parses raw FASTA text into a structured list format. It splits the FASTA text into individual sequences and extracts relevant information like entity ID, chains, sequence, and header.

Usage

```
parse_fasta_text_to_list(raw_fasta_text)
```

Arguments

`raw_fasta_text` A string containing the raw FASTA text to be parsed.

Value

A list of FASTA sequences, each as a separate list element with entity ID, chains, sequence, and header.

Examples

```
raw_fasta_text <- ">1XYZ|Chains A, B\nMVLSPADKT...\n>2XYZ|Chain C\nGVLSADFT..."
fasta_list <- parse_fasta_text_to_list(raw_fasta_text)
```

perform_search *Perform a Search in the RCSB PDB*

Description

This function facilitates searching the RCSB Protein Data Bank (PDB) using a specified search operator. It allows various configurations like return type, additional request options, and verbosity control.

Usage

```
perform_search(
  search_operator,
  return_type = "ENTRY",
  request_options = NULL,
  return_with_scores = FALSE,
  return_raw_json_dict = FALSE,
  verbosity = TRUE
)
```

Arguments

search_operator	An object specifying the search criteria.
return_type	A string specifying the type of data to return, defaulting to 'ENTRY'.
request_options	Additional options for the search request, default is NULL.
return_with_scores	A boolean indicating whether to return search results with scores, default is FALSE.
return_raw_json_dict	A boolean indicating whether to return raw JSON response, default is FALSE.
verbosity	A boolean flag indicating whether to display verbose messages during execution, default is TRUE.

Value

The search results, which can vary based on the return type and options specified.

Examples

```
search_operator = InOperator(value=c("Mus musculus", "Homo sapiens"),
attribute="rcsb_entity_source_organism.taxonomy_lineage.name")
return_type = "NON_POLYMER_ENTITY"
results = perform_search(search_operator, return_type)
results
```

QueryGroup *Create a Grouped Query Object*

Description

Constructs a grouped query object for performing complex searches in RCSB PDB. It groups multiple query objects using a specified logical operator.

Usage

```
QueryGroup(queries, logical_operator)
```

Arguments

queries	A list of query objects to be grouped together.
logical_operator	A string specifying the logical operator (e.g., 'AND', 'OR') to combine the queries.

Value

A list representing the grouped query object.

QueryNode *Create a Query Node Object*

Description

Constructs a query node, which can be a terminal node or a grouped node, for complex RCSB PDB searches. This function is used to structure queries for the search system.

Usage

```
QueryNode(search_operator, logical_operator = NULL)
```

Arguments

search_operator	A search operator or group object.
logical_operator	A string specifying the logical operator, default is NULL. Used only if the search_operator is a group.

Value

A list representing the query node.

query_search *Search Query Function*

Description

This function performs a search query against the RCSB Protein Data Bank using their REST API. It allows for various types of searches based on the provided parameters.

Usage

```
query_search(
  search_term,
  query_type = "full_text",
  return_type = "entry",
  scan_params = NULL,
  num_attempts = 1,
  sleep_time = 0.5
)
```

Arguments

search_term	A string specifying the term to search in the database.
query_type	A string specifying the type of query to perform. Supported values include "full_text", "PubmedIdQuery", "TreeEntityQuery", "ExpTypeQuery", "AdvancedAuthorQuery", "OrganismQuery", "pfam", and "uniprot". Default is "full_text".
return_type	A string specifying the type of search result to return. Possible values are "entry" (default) and "polymer_entity".
scan_params	Additional parameters for the scan, provided as a list. This is 'NULL' by default and typically only used for advanced queries.
num_attempts	An integer specifying the number of attempts to try the query in case of failure.
sleep_time	A numeric value specifying the time in seconds to wait between attempts.

Value

Depending on the return_type, it either returns a list of PDB IDs (if "entry") or the full response from the API.

Examples

```
# Get a list of PDBs for a specific search term
pdbs <- query_search("ribosome")

# Search by PubMed ID Number
pdbs <- query_search(search_term = 27499440, query_type = "PubmedIdQuery")

# Search by source organism using NCBI TaxId
pdbs <- query_search(search_term = "6239", query_type = "TreeEntityQuery")
```

RangeOperator *Create a Range Search Operator*

Description

Constructs a RangeOperator object for search operations that specify a range for attribute values.

Usage

```
RangeOperator(
  attribute,
  from_value,
  to_value,
  include_lower = TRUE,
  include_upper = TRUE,
  negation = FALSE
)
```

Arguments

attribute	The attribute to be evaluated within a range.
from_value	The starting value of the range.
to_value	The ending value of the range.
include_lower	Boolean to include the lower bound in the range.
include_upper	Boolean to include the upper bound in the range.
negation	Boolean to negate the range condition.

Value

An object of class 'RangeOperator'.

RequestOptions *Define Options for Search Requests*

Description

Sets various options for RCSB PDB search requests, such as pagination and sorting.

Usage

```
RequestOptions(
  result_start_index = NULL,
  num_results = NULL,
  sort_by = "score",
  desc = TRUE
)
```

Arguments

result_start_index	An integer specifying the starting index for result pagination, default is NULL.
num_results	An integer specifying the number of results to return, default is NULL.
sort_by	A string indicating the attribute to sort by, default is 'score'.
desc	A boolean indicating whether sorting should be in descending order, default is TRUE.

Value

A list of request options.

return_data_as_dataframe *Transform Response Data into a Dataframe*

Description

This function converts response data from the RCSB Protein Data Bank (PDB) into a structured dataframe. It handles null or empty responses and flattens the data for dataframe conversion.

Usage

```
return_data_as_dataframe(response, data_type, ids)
```

Arguments

response	A list containing the response data from a PDB query.
data_type	A string indicating the type of data contained in the response. This parameter is not directly used in the function but might be relevant for context.
ids	A vector of identifiers corresponding to the response data.

Value

A dataframe constructed from the response data. Returns NULL if the response is null or empty.

ScoredResult *Create a Scored Result Object*

Description

Constructs a scored result object typically used in search results to associate an entity ID with its score.

Usage

```
ScoredResult(entity_id, score)
```

Arguments

entity_id	A string representing the entity ID.
score	A numeric value representing the score associated with the entity.

Value

A list representing the scored result.

search_graphql *Perform a GraphQL Query to RCSB PDB*

Description

Executes a GraphQL query against the RCSB Protein Data Bank (PDB). It sends the query in JSON format and handles the HTTP response, including error checking.

Usage

```
search_graphql(graphql_json_query)
```

Arguments

graphql_json_query	A list containing the GraphQL query in JSON format.

Value

The parsed content of the response from the PDB.

Examples

```
graphql_json_query <- list(query = "{entries(entry_ids:
                          [\"4LZA\", \"5RU3\"]){cell {volume, angle_beta},
                          exptl {method}}}")
result <- search_graphql(graphql_json_query)
result
```

SeqMotifOperator *Create a Sequence Motif Operator for Searches*

Description

Constructs a SeqMotifOperator object for use in sequence motif searches within the RCSB PDB. This operator allows specifying a pattern, sequence type, and pattern type.

Usage

```
SeqMotifOperator(pattern, sequence_type, pattern_type)
```

Arguments

pattern	A string representing the motif pattern to search for.
sequence_type	A string indicating the type of sequence ('DNA', 'RNA', or 'PROTEIN').
pattern_type	A string indicating the type of pattern ('SIMPLE', 'PROSITE', or 'REGEX').

Value

An object of class 'SeqMotifOperator' representing the sequence motif operator.

Examples

```
seq_motif_operator <- SeqMotifOperator(pattern = "A[TU]G",
                                       sequence_type = "DNA",
                                       pattern_type = "REGEX")
seq_motif_operator
```

SequenceOperator *Create a Sequence Operator for Sequence-Based Searches*

Description

Constructs a SequenceOperator object for use in sequence-based searches within the RCSB PDB. This operator allows specifying a sequence, its type, and cutoffs for e-value and identity.

Usage

```
SequenceOperator(
  sequence,
  sequence_type = NULL,
  evalue_cutoff = 100,
  identity_cutoff = 0.95
)
```

Arguments

sequence	A string representing the sequence to search for.
sequence_type	Optional: a string indicating the type of sequence ('DNA', 'RNA', or 'PROTEIN'). If not provided, the type is autoresolved.
evalue_cutoff	A numeric value for the e-value cutoff in the search, default is 100.
identity_cutoff	A numeric value for the identity cutoff in the search, default is 0.95.

Value

An object of class 'SequenceOperator' representing the sequence operator.

StructureOperator *Create a Structure Operator for Structure-Based Searches*

Description

Constructs a StructureOperator object for use in structure-based searches within the RCSB PDB. This operator allows specifying a PDB entry ID, assembly ID, and the search mode.

Usage

```
StructureOperator(
  pdb_entry_id,
  assembly_id = 1,
  search_mode = "STRICT_SHAPE_MATCH"
)
```

Arguments

pdb_entry_id	A string representing the PDB entry ID to search for.
assembly_id	An integer representing the assembly ID, default is 1.
search_mode	A string indicating the search mode, such as 'STRICT_SHAPE_MATCH', default is 'STRICT_SHAPE_MATCH'.

Value

An object of class 'StructureOperator' representing the structure search operator.

walk_nested_dict *Recursively Walk Through a Nested Dictionary*

Description

This function performs a recursive search through a nested dictionary-like structure in R, looking for a specific term and collecting its values. It's useful for extracting specific pieces of data from complex, deeply nested results.

Usage

```
walk_nested_dict(my_result, term, outputs = list(), depth = 0, maxdepth = 25)
```

Arguments

`my_result`	The nested dictionary-like structure to search through.
`term`	The term to search for within the nested dictionary.
`outputs`	An initially empty list to store the results of the search, default is an empty list.
`depth`	The current depth of the recursion, default is 0.
`maxdepth`	The maximum depth to recurse, default is 25. If exceeded, the function issues a warning and returns NULL.

Value

A list of values associated with the term found in the nested dictionary. Returns NULL if the term is not found or if maximum recursion depth is exceeded.

Index

add_property, 2
autoresolve_sequence_type, 3

ChemicalOperator, 3
ComparisonOperator, 4
ContainsPhraseOperator, 4
ContainsWordsOperator, 5

data_fetcher, 5
DefaultOperator, 6
describe_chemical, 6

ExactMatchOperator, 7
ExistsOperator, 7

FastaSequence, 8
fetch_data, 9
find_papers, 9
find_results, 10

generate_json_query, 11
get_fasta_from_rcsb_entry, 11
get_info, 12
get_pdb_file, 13

infer_search_service, 14
InOperator, 14

parse_fasta_text_to_list, 15
perform_search, 15

query_search, 17
QueryGroup, 16
QueryNode, 17

RangeOperator, 18
RequestOptions, 19
return_data_as_dataframe, 20

ScoredResult, 20
search_graphql, 21
SeqMotifOperator, 21
SequenceOperator, 22
StructureOperator, 23

walk_nested_dict, 23