

Package ‘bcdata’

June 25, 2020

Title Search and Retrieve Data from the BC Data Catalogue

Version 0.2.0

Description Search, query, and download tabular and 'geospatial' data from the British Columbia Data Catalogue (<<https://catalogue.data.gov.bc.ca/>>). Search catalogue data records based on keywords, data licence, sector, data format, and B.C. government organization. View metadata directly in R, download many data formats, and query 'geospatial' data available via the B.C. government Web Feature Service ('WFS') using 'dplyr' syntax.

License Apache License (== 2.0)

URL <https://bcgov.github.io/bcdata/>,
<https://catalogue.data.gov.bc.ca/>,
<https://github.com/bcgov/bcdata>

BugReports <https://github.com/bcgov/bcdata/issues>

Imports cli, crul (>= 0.7.4), dbplyr (>= 1.3.0), dplyr (>= 0.8.1),
tibble, glue, jsonlite (>= 1.6), leaflet, leaflet.extras,
methods, purrr (>= 0.2), readr (>= 1.3), readxl, rlang (>=
0.3.1), sf (>= 0.7), tidyselect (>= 0.2.5), utils, xml2

Suggests covr, ggplot2, knitr, rmarkdown, testthat

VignetteBuilder knitr

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

NeedsCompilation no

Author Andy Teucher [aut, cre] (<<https://orcid.org/0000-0002-7840-692X>>),
Sam Albers [aut, ctb] (<<https://orcid.org/0000-0002-9270-7884>>),
Stephanie Hazlitt [aut, ctb] (<<https://orcid.org/0000-0002-3161-2304>>),
Province of British Columbia [cph]

Maintainer Andy Teucher <andy.teucher@gov.bc.ca>

Repository CRAN

Date/Publication 2020-06-25 07:00:02 UTC

R topics documented:

bcdc_browse	2
bcdc_describe_feature	3
bcdc_get_data	4
bcdc_get_record	6
bcdc_list	7
bcdc_options	7
bcdc_preview	8
bcdc_query_geodata	9
bcdc_read_functions	10
bcdc_search	10
bcdc_search_facets	11
bcdc_tidy_resources	12
CQL	13
cql_geom_predicates	14
Index	16

bcdc_browse

Load the B.C. Data Catalogue URL into an HTML browser

Description

This is a wrapper around `utils::browseURL` with the URL for the B.C. Data Catalogue as the default

Usage

```
bcdc_browse(
  query = NULL,
  browser = getOption("browser"),
  encodeIfNeeded = FALSE
)
```

Arguments

query	Default (NULL) opens a browser to <code>https://catalogue.data.gov.bc.ca</code> . This argument will also accept a B.C. Data Catalogue record ID or name to take you directly to that page. If the provided ID or name doesn't lead to a valid webpage, <code>bcdc_browse</code> will search the data catalogue for that string.
browser	a non-empty character string giving the name of the program to be used as the HTML browser. It should be in the PATH, or a full path specified. Alternatively, an R function to be called to invoke the browser. Under Windows NULL is also allowed (and is the default), and implies that the file association mechanism will be used.

`encodeIfNeeded` Should the URL be encoded by [URLencode](#) before passing to the browser? This is not needed (and might be harmful) if the browser program/function itself does encoding, and can be harmful for 'file://' URLs on some systems and for 'http://' URLs passed to some CGI applications. Fortunately, most URLs do not need encoding.

Value

A browser is opened with the B.C. Data Catalogue URL loaded if the session is interactive. The URL used is returned as a character string.

See Also

[browseURL](#)

Examples

```
## Take me to the B.C. Data Catalogue home page
bcdc_browse()

## Take me to the B.C. airports catalogue record
bcdc_browse("bc-airports")

#' ## Take me to the B.C. airports catalogue record
bcdc_browse("76b1b7a3-2112-4444-857a-afccf7b20da8")

## Take me to the catalogue search results for 'fish'
bcdc_browse("fish")
```

`bcdc_describe_feature` *Describe the attributes of a Web Service feature*

Description

Describe the attributes of column of a record accessed through the Web Service. This can be a useful tool to examine a layer before issuing a query with `bcdc_query_geodata`.

Usage

```
bcdc_describe_feature(record)
```

Arguments

`record` either a `bcdc_record` object (from the result of `bcdc_get_record()`), a character string denoting the name or ID of a resource (or the URL) or a BC Geographic Warehouse (BCGW) name.

It is advised to use the permanent ID for a record or the BCGW name rather than the human-readable name to guard against future name changes of the record. If you use the human-readable name a warning will be issued once per session. You can silence these warnings altogether by setting an option: `options("silence_named_get_data_warning" = TRUE)` - which you can set in your `.Rprofile` file so the option persists across sessions.

Value

`bcdc_describe_features` returns a tibble describing the attributes of a B.C. Data Catalogue record. The tibble returns the following columns:

- `col_name`: attributes of the feature
- `sticky`: whether a column can be separated from the record in a Web Service call via the `dplyr::select` method
- `remote_col_type`: class of what is return by the web feature service
- `local_col_type`: the column class in R

Examples

```
bcdc_describe_feature("bc-airports")
bcdc_describe_feature("WHSE_IMAGERY_AND_BASE_MAPS.GSR_AIRPORTS_SVW")
```

bcdc_get_data
Download and read a resource from a B.C. Data Catalogue record

Description

Download and read a resource from a B.C. Data Catalogue record

Usage

```
bcdc_get_data(record, resource = NULL, verbose = TRUE, ...)
```

Arguments

record	<p>either a <code>bcdc_record</code> object (from the result of <code>bcdc_get_record()</code>), a character string denoting the name or ID of a resource (or the URL) or a BC Geographic Warehouse (BCGW) name.</p> <p>It is advised to use the permanent ID for a record or the BCGW name rather than the human-readable name to guard against future name changes of the record. If you use the human-readable name a warning will be issued once per session. You can silence these warnings altogether by setting an option: <code>options("silence_named_get_data_warning" = TRUE)</code> - which you can set in your <code>.Rprofile</code> file so the option persists across sessions.</p>
resource	<p>optional argument used when there are multiple data files within the same record. See examples.</p>
verbose	<p>When more than one resource is available for a record, should extra information about those resources be printed to the console? Default TRUE</p>
...	<p>arguments passed to other functions. Tabular data is passed to a function to handle the import based on the file extension. <code>bcdc_read_functions()</code> provides details on which functions handle the data import. You can then use this information to look at the help pages of those functions. See the examples for a workflow that illustrates this process. For spatial Web Service data the ... arguments are passed to <code>bcdc_query_geodata()</code>.</p>

Value

An object of a type relevant to the resource (usually a tibble or an sf object)

Examples

```
# Using the record and resource ID:
bcdc_get_data(record = '76b1b7a3-2112-4444-857a-afccf7b20da8',
              resource = '4d0377d9-e8a1-429b-824f-0ce8f363512c')
bcdc_get_data('1d21922b-ec4f-42e5-8f6b-bf320a286157')

# Using a `bcdc_record` object obtained from `bcdc_get_record`:
record <- bcdc_get_record('1d21922b-ec4f-42e5-8f6b-bf320a286157')
bcdc_get_data(record)

# Using a BCGW name
bcdc_get_data("WHSE_IMAGERY_AND_BASE_MAPS.GSR_AIRPORTS_SVW")

## Example of correcting import problems

## Some initial problems reading in the data
bcdc_get_data('d7e6c8c7-052f-4f06-b178-74c02c243ea4')

## From bcdc_get_record we realize that the data is in xlsx format
bcdc_get_record('8620ce82-4943-43c4-9932-40730a0255d6')

## bcdc_read_functions let's us know that bcdat
```

```
## uses readxl::read_excel to import xlsx files
bcdc_read_functions()

## bcddata let's you know that this resource has
## multiple worksheets
bcdc_get_data('8620ce82-4943-43c4-9932-40730a0255d6')

## we can control what is read in from an excel file
## using arguments from readxl::read_excel

bcdc_get_data('8620ce82-4943-43c4-9932-40730a0255d6', sheet = 'Regional Districts')
```

bcdc_get_record *Show a single B.C. Data Catalogue record*

Description

Show a single B.C. Data Catalogue record

Usage

```
bcdc_get_record(id)
```

Arguments

`id` the human-readable name, permalink ID, or URL of the record.

It is advised to use the permanent ID for a record rather than the human-readable name to guard against future name changes of the record. If you use the human-readable name a warning will be issued once per session. You can silence these warnings altogether by setting an option: `options("silence_named_get_record_warning" = TRUE)` - which you can put in your `.Rprofile` file so the option persists across sessions.

Value

A list containing the metadata for the record

Examples

```
bcdc_get_record("https://catalogue.data.gov.bc.ca/dataset/bc-airports")
bcdc_get_record("bc-airports")
bcdc_get_record("https://catalogue.data.gov.bc.ca/dataset/76b1b7a3-2112-4444-857a-afccf7b20da8")
bcdc_get_record("76b1b7a3-2112-4444-857a-afccf7b20da8")
```

bcdc_list	<i>Return a full list of the names of B.C. Data Catalogue records</i>
-----------	---

Description

Return a full list of the names of B.C. Data Catalogue records

Usage

```
bcdc_list()
```

Value

A character vector of the names of B.C. Data Catalogue records

bcdc_options	<i>Retrieve options used in bcddata, their value if set and the default value.</i>
--------------	--

Description

This function retrieves bcddata specific options that can be set. These options can be set using `option({name of the option} = {value of the option})`. The default options are purposefully set conservatively to hopefully ensure successful requests. Resetting these options may result in failed calls to the data catalogue. Options in R are reset every time R is re-started. See examples for addition ways to restore your initial state.

Usage

```
bcdc_options()
```

Details

`bcddata.max_geom_pred_size` is the maximum size of an object used for a geometric operation. Objects that are bigger than this value will have a bounding box drawn and apply the geometric operation on that simpler polygon. Users can try to increase the maximum geometric predicate size and see if the bcddata catalogue accepts their request.

`bcddata.chunk_limit` is an option useful when dealing with very large data sets. When requesting large objects from the catalogue, the request is broken up into smaller chunks which are then re-combined after they've been downloaded. bcddata does this all for you but using this option you can set the size of the chunk requested. On faster internet connections, a bigger chunk limit could be useful while on slower connections, it is advisable to lower the chunk limit. Chunks must be less than 10000.

Examples

```
## Save initial conditions
original_options <- options()

## See initial options
bcdc_options()

options(bcddata.max_geom_pred_size = 1E6)

## See updated options
bcdc_options()

## Reset initial conditions
options(original_options)
```

bcdc_preview

Get map from the B.C. Web Service

Description

Get map from the B.C. Web Service

Usage

```
bcdc_preview(record)
```

Arguments

record	either a <code>bcdc_record</code> object (from the result of <code>bcdc_get_record()</code>), a character string denoting the name or ID of a resource (or the URL) or a BC Geographic Warehouse (BCGW) name.
--------	--

It is advised to use the permanent ID for a record or the BCGW name rather than the human-readable name to guard against future name changes of the record. If you use the human-readable name a warning will be issued once per session. You can silence these warnings altogether by setting an option: `options("silence_named_get_data_warning" = TRUE)` - which you can set in your `.Rprofile` file so the option persists across sessions.

Examples

```
bcdc_preview("regional-districts-legally-defined-administrative-areas-of-bc")
bcdc_preview("points-of-well-diversion-applications")

# Using BCGW name
bcdc_preview("WHSE_LEGAL_ADMIN_BOUNDARIES.ABMS_REGIONAL_DISTRICTS_SP")
```


bcdc_query_geodata *Query data from the B.C. Web Service*

Description

Queries features from the B.C. Web Service. The data must be available as a Web Service. See `bcdc_get_record(record)$resources`. If the record is greater than 10000 rows, the response will be paginated. If you are querying layers of this size, expect that the request will take quite a while.

Usage

```
bcdc_query_geodata(record, crs = 3005)
```

Arguments

record	either a <code>bcdc_record</code> object (from the result of <code>bcdc_get_record()</code>), a character string denoting the name or ID of a resource (or the URL) or a BC Geographic Warehouse (BCGW) name. It is advised to use the permanent ID for a record or the BCGW name rather than the human-readable name to guard against future name changes of the record. If you use the human-readable name a warning will be issued once per session. You can silence these warnings altogether by setting an option: <code>options("silence_named_get_data_warning" = TRUE)</code> - which you can set in your <code>.Rprofile</code> file so the option persists across sessions.
crs	the epsg code for the coordinate reference system. Defaults to 3005 (B.C. Albers). See https://epsg.io .

Details

Note that this function doesn't actually return the data, but rather an object of class `bcdc_promise`, which includes all of the information required to retrieve the requested data. In order to get the actual data as an `sf` object, you need to run `collect()` on the `bcdc_promise`. This allows further refining the call to `bcdc_query_geodata()` with `filter()` and/or `select()` statements before pulling down the actual data as an `sf` object with `collect()`. See examples.

Value

A `bcdc_promise` object. This object includes all of the information required to retrieve the requested data. In order to get the actual data as an `sf` object, you need to run `collect()` on the `bcdc_promise`.

Examples

```
# Returns a bcdc_promise, which can be further refined using filter/select:
```

```

bcdc_query_geodata("bc-airports", crs = 3857)

# To obtain the actual data as an sf object, collect() must be called:
bcdc_query_geodata("bc-airports", crs = 3857) %>%
  filter(PHYSICAL_ADDRESS == 'Victoria, BC') %>%
  collect()

bcdc_query_geodata("ground-water-wells") %>%
  filter(OBSERVATION_WELL_NUMBER == 108) %>%
  select(WELL_TAG_NUMBER, INTENDED_WATER_USE) %>%
  collect()

## A moderately large layer
bcdc_query_geodata("bc-environmental-monitoring-locations")
bcdc_query_geodata("bc-environmental-monitoring-locations") %>%
  filter(PERMIT_RELATIONSHIP == "DISCHARGE")

## A very large layer
bcdc_query_geodata("terrestrial-protected-areas-representation-by-biogeoclimatic-unit")

## Using a BCGW name
bcdc_query_geodata("WHSE_IMAGERY_AND_BASE_MAPS.GSR_AIRPORTS_SVW")

```

bcdc_read_functions *Formats supported and loading functions*

Description

Provides a tibble of formats supported by bcddata and the associated function that reads that data into R. This function is meant as a resource to determine which parameters can be passed through the bcdc_get_data function to the reading function. This is particularly important to know if the data requires using arguments from the read in function.

Usage

```
bcdc_read_functions()
```

bcdc_search *Search the B.C. Data Catalogue*

Description

Search the B.C. Data Catalogue

Usage

```
bcdc_search(
  ...,
  license_id = NULL,
  download_audience = "Public",
  type = NULL,
  res_format = NULL,
  sector = NULL,
  organization = NULL,
  n = 100
)
```

Arguments

- ... search terms
- license_id the type of license (see `bcdc_search_facets("license_id")`).
- download_audience download audience (see `bcdc_search_facets("download_audience")`). Default "Public"
- type type of resource (see `bcdc_search_facets("type")`)
- res_format format of resource (see `bcdc_search_facets("res_format")`)
- sector sector of government from which the data comes (see `bcdc_search_facets("sector")`)
- organization government organization that manages the data (see `bcdc_search_facets("organization")`)
- n number of results to return. Default 100

Value

A list containing the records that match the search

Examples

```
bcdc_search("forest")
bcdc_search("regional district", type = "Geographic", res_format = "fgdb")
```

bcdc_search_facets *Get the valid values for a facet (that you can use in `bcdc_search()`)*

Description

Get the valid values for a facet (that you can use in `bcdc_search()`)

Usage

```
bcdc_search_facets(
  facet = c("license_id", "download_audience", "type", "res_format", "sector",
            "organization")
)
```

Arguments

facet the facet(s) for which to retrieve valid values. Can be one or more of: "license_id", "download_audience", "type", "res_format", "sector", "organization"

Value

A data frame of values for the selected facet

Examples

```
bcdc_search_facets("type")
```

bcdc_tidy_resources *Provide a data frame containing the metadata for all resources from a single B.C. Data Catalogue record*

Description

Returns a rectangular data frame of all resources contained within a record. This is particularly useful if you are trying to construct a vector of multiple resources in a record. The data frame also provides useful information on the formats, availability and types of data available.

Usage

```
bcdc_tidy_resources(record)
```

Arguments

record either a bcdc_record object (from the result of bcdc_get_record()), a character string denoting the name or ID of a resource (or the URL) or a BC Geographic Warehouse (BCGW) name.

It is advised to use the permanent ID for a record or the BCGW name rather than the human-readable name to guard against future name changes of the record. If you use the human-readable name a warning will be issued once per session. You can silence these warnings altogether by setting an option: options("silence_named_get_data_warning" = TRUE) - which you can set in your .Rprofile file so the option persists across sessions.

Value

A data frame containing the metadata for all the resources for a record

Examples

```
airports <- bcdc_get_record("bc-airports")
airports <- bcdc_tidy_resources(airports)
```

CQL

CQL escaping

Description

Write a CQL expression to escape its inputs, and return a CQL/SQL object. Used when writing filter expressions in `bcdc_query_geodata()`.

Usage

```
CQL(...)
```

Arguments

... Character vectors that will be combined into a single CQL statement.

Details

See [the CQL/ECQL for Geoserver website](#).

Value

An object of class `c("CQL", "SQL")`

Examples

```
CQL("FOO > 12 & NAME LIKE 'A&'")
```

 cql_geom_predicates *CQL Geometry Predicates*

Description

Functions to construct a CQL expression to be used to filter results from `bcdc_query_geodata()`. See [the geoserver CQL documentation for details](#). The `sf` object is automatically converted in a bounding box to reduce the complexity of the Web Service call. Subsequent in-memory filtering may be needed to achieve exact results.

Usage

`EQUALS(geom)`

`DISJOINT(geom)`

`INTERSECTS(geom)`

`TOUCHES(geom)`

`CROSSES(geom)`

`WITHIN(geom)`

`CONTAINS(geom)`

`OVERLAPS(geom)`

`BBOX(coords, crs = NULL)`

```
DWITHIN(
  geom,
  distance,
  units = c("meters", "feet", "statute miles", "nautical miles", "kilometers")
)
```

Arguments

<code>geom</code>	an <code>sf/sfc/sfg</code> or <code>bbox</code> object (from the <code>sf</code> package)
<code>coords</code>	the coordinates of the bounding box as four-element numeric vector <code>c(xmin, ymin, xmax, ymax)</code> , a <code>bbox</code> object from the <code>sf</code> package (the result of running <code>sf::st_bbox()</code> on an <code>sf</code> object), or an <code>sf</code> object which then gets converted to a bounding box on the fly.
<code>crs</code>	(Optional) A numeric value or string containing an SRS code. If <code>coords</code> is a <code>bbox</code> object with non-empty <code>crs</code> , it is taken from that. (For example, <code>'EPSG:3005'</code> or just <code>3005</code> . The default is to use the CRS of the queried layer)

distance	numeric value for distance tolerance
units	units that distance is specified in. One of "feet", "meters", "statute miles", "nautical miles", "kilometers"

Value

a CQL expression to be passed on to the WFS call

Index

BBOX (cql_geom_predicates), 14
bcdc_browse, 2
bcdc_describe_feature, 3
bcdc_get_data, 4
bcdc_get_record, 6
bcdc_list, 7
bcdc_options, 7
bcdc_preview, 8
bcdc_query_geodata, 9
bcdc_query_geodata(), 5, 13, 14
bcdc_read_functions, 10
bcdc_read_functions(), 5
bcdc_search, 10
bcdc_search(), 11
bcdc_search_facets, 11
bcdc_tidy_resources, 12
browseURL, 3

collect(), 9
CONTAINS (cql_geom_predicates), 14
CQL, 13
cql_geom_predicates, 14
CROSSES (cql_geom_predicates), 14

DISJOINT (cql_geom_predicates), 14
DWITHIN (cql_geom_predicates), 14

EQUALS (cql_geom_predicates), 14

filter(), 9

INTERSECTS (cql_geom_predicates), 14

OVERLAPS (cql_geom_predicates), 14

select(), 9

TOUCHES (cql_geom_predicates), 14

urlencode, 3

WITHIN (cql_geom_predicates), 14