Package 'bcdata'

December 17, 2019

Title Search and Retrieve Data from the BC Data Catalogue

Version 0.1.2

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 (>= 0.7.5), 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 7.0.2

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], Province of British Columbia [cph]

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

Repository CRAN

Date/Publication 2019-12-17 06:30:03 UTC

R topics documented:

`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 `https://catalogue.data.gov.bc.ca`. 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, bcdc_browse 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 a Web Service feature*

Description

Describe the columns from a Web Service feature. The column name, whether a column can be separated from the record in Web Service (nillable) and the type of column are returned. This can be a helpful 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.

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	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.
resource	optional argument used when there are multiple data files within the same record. See examples.
verbose	When more than one resource is available for a record, should extra information about those resources be printed to the console? Default TRUE

... arguments passed to other functions. Tabular data is passed to a function to handle the import based on the file extension. `bcdc_read_functions()` 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 `bcdc_query_geodata()`.

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('d7e6c8c7-052f-4f06-b178-74c02c243ea4')

## bcdc_read_functions let's us know that bcdata
## uses readxl::read_excel to import xlsx files
bcdc_read_functions()

## If you read the help page for readxl::read_excel,
## it seems likely that we need to skip the first row:
bcdc_get_data('d7e6c8c7-052f-4f06-b178-74c02c243ea4', skip = 1)
```

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

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 *Retrieve options used in bcdata, their value if set and the default value.*

Description

This function retrieves bcdata 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

`bcdata.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 bcdata catalogue accepts their request.

`bcdata.chunk_limit` is an option useful when dealing with very large datasets. 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. bcdata 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(bcdata.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 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.

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 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.
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, WATERSHED_CODE) %>%
  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 bcdata 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")
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

geom	an sf/sfc/sfg object
coords	the coordinates of the bounding box as four-element numeric vector `c(xmin,ymin,xmax,ymax)`, or a bbox object from the `sf` package (the result of running `sf::st_bbox()` on an `sf` object).
crs	(Optional) A string containing an SRS code (For example, 'EPSG:1234'. 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

filter.bcdc_promise *Filter a query from Web Service call*

Description

Filter a query from Web Service using dplyr methods. This filtering is accomplished lazily so that the full sf object is not read into memory until `collect()` has been called.

Usage

```
## S3 method for class 'bcdc_promise'
filter(.data, ...)
```

Arguments

`.data`	object of class `bcdc_promise` (likely passed from `bcdc_query_geodata()`)
`...`	Logical predicates with which to filter the results. Multiple conditions are combined with `&`. Only rows where the condition evaluates to `TRUE` are kept. Accepts normal R expressions as well as any of the special CQL geometry functions such as `WITHIN()` or `INTERSECTS()`. If you know `CQL` and want to write a `CQL` query directly, write it enclosed in quotes, wrapped in the `CQL()` function. e.g., `CQL("ID = '42'")`

Examples

```
 crd <- bcdc_query_geodata("regional-districts-legally-defined-administrative-areas-of-bc") %>%
    filter(ADMIN_AREA_NAME == "Cariboo Regional District") %>%
    collect()

ret1 <- bcdc_query_geodata("fire-perimeters-historical") %>%
  filter(FIRE_YEAR == 2000, FIRE_CAUSE == "Person", INTERSECTS(crd)) %>%
  collect()
```

mutate.bcdc_promise *Throw an informative error when attempting mutate on a Web Service call*

Description

The CQL syntax to generate WFS calls does not current allow arithmetic operations. Therefore this function exists solely to generate an informative error that suggests an alternative approach to use mutate with bcdata

Usage

```
## S3 method for class 'bcdc_promise'
mutate(.data, ...)
```

Arguments

.data	object of class bcdc_promise (likely passed from bcdc_query_geodata())
...	One or more unquoted expressions separated by commas. See details.

Examples

```
## Not run:

## Mutate columns
bcdc_query_geodata("bc-airports") %>%
  mutate(LATITUDE * 100)

## End(Not run)
```

---

select.bcdc_promise *Select columns from Web Service call*

Description

Similar to a dplyr::select call, this allows you to select which columns you want the Web Service to return. A key difference between dplyr::select and bcdata::select is the presence of "sticky" columns that are returned regardless of what columns are selected. If any of these "sticky" columns are selected only "sticky" columns are return. bcdc_describe_feature is one way to tell if columns are sticky in advance of issuing the Web Service call.

Usage

```
## S3 method for class 'bcdc_promise'
select(.data, ...)
```

Arguments

.data	object of class bcdc_promise (likely passed from bcdc_query_geodata())
...	One or more unquoted expressions separated by commas. See details.

Examples

```
feature_spec <- bcdc_describe_feature("bc-airports")
## Columns that can selected:
feature_spec[feature_spec$nillable == TRUE,]

## Select columns
bcdc_query_geodata("bc-airports") %>%
  select(DESCRIPTION, PHYSICAL_ADDRESS)

## Select "sticky" columns
bcdc_query_geodata("bc-airports") %>%
  select(LOCALITY)
```

Index