Package ‘mudata2’

October 13, 2022

Title Interchange Tools for Multi-Parameter Spatiotemporal Data

Version 1.1.2

Maintainer Dewey Dunnington <dewey@fishandwhistle.net>

Description Formatting and structuring multi-parameter spatiotemporal data is often a time-consuming task. This package offers functions and data structures designed to easily organize and visualize these data for applications in geology, paleolimnology, dendrochronology, and paleoclimate. See Dunnington and Spooner (2018) <doi:10.1139/facets-2017-0026>.

Imports ggplot2, dplyr (>= 0.7), jsonlite (>= 1.2), tibble, magrittr, stringr, readr, tidyr, lubridate, rlang, tidyselect, withr, glue, fs

Depends R (>= 3.2.0)

Suggests testthat (>= 2.1.0), sf (>= 0.5.5), covr, hms, knitr, markdown

License GPL-2

URL https://paleolimbot.github.io/mudata2,
https://github.com/paleolimbot/mudata2

BugReports https://github.com/paleolimbot/mudata2/issues

LazyData true

RoxygenNote 7.0.2

Encoding UTF-8

VignetteBuilder knitr

NeedsCompilation no

Author Dewey Dunnington [aut, cre] (<https://orcid.org/0000-0002-9415-4582>)

Repository CRAN

Date/Publication 2020-03-20 20:20:03 UTC
alta_lake

Altas Lake Gravity Core Data

Description

Bulk geochemistry of a gravity core from Alta Lake, Whistler, British Columbia, Canada.

Usage

alta_lake

Format

A mudata object
as_mudata

References

Examples
print(alta_lake)

---

as_mudata  Coerce objects to mudata

Description
Coerce objects to mudata

Usage
as_mudata(x, ...)

Arguments
x  An object
...
Passed to other methods

Value
A mudata object or an error
distinct_params

Get distinct params, locations, and datasets from a mudata object

Description

Get distinct params, locations, and datasets from a mudata object

Usage

distinct_params(x, ...)

## Default S3 method:
distinct_params(x, table = "data", ...)

distinct_locations(x, ...)

## Default S3 method:
distinct_locations(x, table = "data", ...)

distinct_datasets(x, ...)

## Default S3 method:
distinct_datasets(x, table = "data", ...)

distinct_columns(x, ...)

## S3 method for class 'mudata'
src_tbls(x, ...)

Arguments

x A mudata object  
... Passed to other methods  
table The table to use to calculate the distinct values. Using the "data" table is safest, but for large datasets that are not in memory, using the meta table (params, locations, or datasets) may be useful.

Value

A character vector of distinct parameter names
filter_datasets

Examples

distinct_params(kentvillegreenwood)
distinct_locations(kentvillegreenwood)
distinct_datasets(kentvillegreenwood)

filter_datasets

Subset a mudata object by complex expression

Description

These methods allow more complex selection criteria than select_datasets and family, which only use the identifier values. These methods first subset the required table using the provided expression, then subset other tables to ensure internal consistency.

Usage

filter_datasets(.data, ...)

## Default S3 method:
filter_datasets(.data, ...)

filter_data(.data, ...)

## Default S3 method:
filter_data(.data, ...)

filter_locations(.data, ...)

## Default S3 method:
filter_locations(.data, ...)

filter_params(.data, ...)

## Default S3 method:
filter_params(.data, ...)

Arguments

.data A mudata object

... Objects passed to filter on the appropriate table

Value

A subsetted mudata object

See Also

filter, select_locations
Examples

# select only locations with a latitude above 45
ns_climate %>%
  filter_locations(latitude > 45)

# select only params measured in mm
ns_climate %>%
  filter_params(unit == "mm")

# select only june temperature from ns_climate
library(lubridate)
ns_climate %>%
  filter_data(month(date) == 6)

is_mudata

Test if an object is a mudata object

Description

Test if an object is a mudata object

Usage

is_mudata(x)

is.mudata(x)

Arguments

x An object

Value

TRUE if the object is a mudata object, FALSE otherwise

Examples

is_mudata(kentvillegreenwood)
kentvillegreenwood

**Kentville/Greenwood Climate Data**

**Description**

Climate data for Kentville and Greenwood (Nova Scotia) for July and August of 1999.

**Usage**

kentvillegreenwood

**Format**

A `mudata` object

**Source**


**Examples**

`print(kentvillegreenwood)`

---

long_lake

**Long Lake Lake Gravity/Percussion Core Data**

**Description**

Bulk geochemistry of a gravity core from Long Lake, Cumberland Marshes Region, Nova Scotia-New Brunswick Border Region, Canada.

**Usage**

long_lake

**Format**

A `mudata` object

**References**

Create a mudata object

Create a mudata object, which is a collection of five tables: data, locations, params, datasets, and columns. You are only required to provide the data table, which must contain columns "param" and "value", but will more typically contain columns "location", "param", "datetime" (or "date"), and "value". See ns_climate, kentvillegreenwood, alta_lake, long_lake, and second_lake_temp for examples of data in this format.

Usage

mudata(
    data,
    locations = NULL,  # The locations table, which is a data frame containing the columns (at least) "dataset", and "location". If omitted, it will be created automatically using all unique dataset/location combinations.
    params = NULL,  # The params table, which is a data frame containing the columns (at least) "dataset", and "param". If omitted, it will be created automatically using all unique dataset/param combinations.
    datasets = NULL,  # The datasets table, which is a data frame containing the column (at least) "dataset". If omitted, it will be generated automatically using all unique datasets.
    columns = NULL,  # A data.frame/tibble containing columns "param" and "value" (at least), but more typically columns "location", "param", "datetime" (or "date", depending on the type of data), and "value".
    x_columns = NULL,
    ...,
    more_tbls = NULL,
    dataset_id = "default",
    location_id = "default",
    validate = TRUE
)

Arguments

data  A data.frame/tibble containing columns "param" and "value" (at least), but more typically columns "location", "param", "datetime" (or "date", depending on the type of data), and "value".

locations

params

datasets
The columns table, which is a data frame containing the columns (at least) "dataset", "table", and "column". If omitted, it will be created automatically using all dataset/table/column combinations.

A vector of column names from the data table that in combination with "dataset", "location", and "param" identify unique rows. These will typically be guessed using the column names between "param" and "value".

More tbls (as named arguments) to be included in the mudata object

The dataset to use if a "dataset" column is omitted.

The location if a "location" column is omitted.

Pass FALSE to skip validation of input tables using validate_mudata.

An object of class "mudata", which is a list with components data, locations, params, datasets, columns, and any other tables provided in more_tbls. All list components must be tbls.


Examples

# use the data table from kentvillegreenwood as a template
kg_data <- tbl_data(kentvillegreenwood)
# create mudata object using just the data table
mudata(kg_data)

# create a mudata object starting from a parameter-wide data frame
library(tidyr)
library(dplyr)

gather(Ca, Ti, V, key = "param", value = "param_value") %>%
group_by(core, param, depth) %>%
summarise(value = mean(param_value), sd = mean(param_value)) %>%
rename(location = core)

# create mudata object
mudata(datatable)
mudata_prepare_column  *Prepare mudata table columns for writing*

**Description**

This set of generics is similar to `output_column` in that it converts columns to a form suitable to writing. `mudata_prepare_column` in combination with is intended to be opposites with `mudata_parse_column` except for date/time vectors that are not in UTC (`mudata_parse_column` assumes UTC, and `mudata_prepare_column` always converts to UTC with a message).

**Usage**

```r
mudata_prepare_column(x, format = NA, ...)  
mudata_prepare_tbl(x, format = NA, ...)  
## Default S3 method:  
mudata_prepare_tbl(x, format = NA, ...)  
## S3 method for class 'tbl'  
mudata_prepare_tbl(x, format = NA, ...)  
## S3 method for class 'data.frame'  
mudata_prepare_tbl(x, format = NA, ...)  
## Default S3 method:  
mudata_prepare_column(x, format = NA, ...)  
## S3 method for class 'POSIXt'  
mudata_prepare_column(x, format = NA, ...)  
## S3 method for class 'sfc'  
mudata_prepare_column(x, format = NA, ...)  
## S3 method for class 'hms'  
mudata_prepare_column(x, format = NA, ...)  
## S3 method for class 'list'  
mudata_prepare_column(x, format = NA, ...)  
mudata_parse_column(x, type_str = NA_character_, ...)  
mudata_parse_tbl(x, type_str = NA_character_, ...)  
```

**Arguments**

- `x`  
  A an object
mutate_data

format       csv, json, or NA for unknown,
...          Passed to methods
type_str     A type string, generated by the internal generate_type_str

Details

Type strings are currently internal, and are in the columns table in the "type" column. They are usually one of "character", "date", "datetime", "double", "integer", "json", and "wkt". They can also contain simple arguments, like "wkt(epsg=4326)" (actually, "wkt" is the only type string that should have arguments). You should generally not mess with these (in fact, the "type" column in the columns table is overwritten right before read by default, so it is hard to mess this up).

Value

An atomic vector

mutate_data Modify mudata tables

Description

Modify mudata tables

Usage

mutate_data(x, ...)
mutable_params(x, ...)
mutable_locations(x, ...)
mutable_datasets(x, ...)
mutable_columns(x, ...)
mutable_tbl(x, ...)

## Default S3 method:
mutable_tbl(x, tbl, ...)

Arguments

x          A mudata object
...        Passed to mutate
tbl         The table name to modify
**new_mudata**

**Value**
A modified mudata object

**Examples**
```
library(lubridate)
second_lake_temp %>%
mutate_data(datetime = with_tz(datetime, "America/Halifax"))
```

**Description**
Validates a mudata object by calling `rlang::abort` when an error is found; creates a mudata object from a list. Validation is generally performed when objects are created using `mudata`, or when objects are read/written using `read_mudata` and `write_mudata`.

**Usage**
```
new_mudata(md, x_columns)
validate_mudata(
  md,
  check_unique = TRUE,
  check_references = TRUE,
  action = abort
)
```

**Arguments**
- `md` An object of class 'mudata'
- `x_columns` The x_columns attribute (see `mudata`).
- `check_unique` Check if columns identify unique values in the appropriate tables
- `check_references` Check the referential integrity of the mudata object
- `action` The function to be called when errors are detected in `validate_mudata`

**Examples**
```
validate_mudata(kentvillegreenwood)
new_mudata(kentvillegreenwood, x_columns = "date")
```
**ns_climate**

**Nova Scotia Long-Term Climate Data**

**Description**

Monthly climate data for locations in Nova Scotia with records longer than 80 years.

**Usage**

```r
ns_climate
```

**Format**

A `mudata` object

**Source**


**Examples**

```r
print(ns_climate)
```

---

**parallel_gather**

*Melt multiple sets of columns in parallel*

**Description**

Essentially this is a wrapper around `gather` that is able to `bind_cols` with several gather operations. This is useful when a wide data frame contains uncertainty or flag information in paired columns.

**Usage**

```r
parallel_gather(x, key, ..., convert = FALSE, factor_key = FALSE)
```

**Arguments**

- `x` A data.frame
- `key` Column name to use to store variables, which are the column names of the first gather operation.
- `...` Named arguments in the form `new_col_name = c(old, col, names)`. All named arguments must have the same length (i.e., gather the same number of columns).
- `convert` Convert types (see `gather`)
- `factor_key` Control whether the key column is a factor or character vector.
Value

A gathered data frame.

See Also

gather

Examples

# gather paired value/error columns using
# parallel_gather
parallel_gather(pocmajsum, key = "param",
value = c(Ca, Ti, V),
    sd = c(Ca_sd, Ti_sd, V_sd))

# identical result using only tidyverse functions
library(dplyr)
library(tidyr)
gathered_values <- pocmajsum %>%
    select(core, depth, Ca, Ti, V) %>%
    gather(Ca, Ti, V,
        key = "param", value = "value")
gathered_sds <- pocmajsum %>%
    select(core, depth, Ca_sd, Ti_sd, V_sd) %>%
    gather(Ca_sd, Ti_sd, V_sd,
        key = "param_sd", value = "sd")

bind_cols(
    gathered_values,
    gathered_sds %>% select(sd)
)

---

pocmaj  

**Pockwock Lake/Lake Major Elemental Sample Data**

Description

A small example data.frame used to test structure methods.

Usage

pocmaj

Format

A data.frame containing multi-qualifier concentration data
pocmajsum

**Pre-summarised Sample Data**

**Description**
A small example data.frame of pre-summarised data; a summarised version of the pocmaj dataset.

**Usage**

pocmajsum

**Format**
A data.frame containing multi-qualifier data

---

print.mudata

**Print a mudata object**

**Description**
Print a mudata object

**Usage**

```r
## S3 method for class 'mudata'
print(x, ..., width = NULL)
## S3 method for class 'mudata'
summary(object, ...)
```

**Arguments**

- `x, object` A mudata object
- `...` Passed to other methods
- `width` The number of characters to use as console width

**Value**

print returns x (invisibly); summary returns a data frame with summary information.

**Examples**

```r
print(kentvillegreenwood)
summary(kentvillegreenwood)
```
rename_locations

rename_locations

rbind.mudata  Combine mudata objects

Description

This implementation of `rbind` combines component tables using `bind_rows` and `distinct`. When combined object use different datasets, or when subsets of the same object are recombined, this function works well. When this is not the case, it may be necessary to modify the tables such that when they are passed to `bind_rows` and `distinct`, no duplicate information exists. This should be picked up by `validate_mudata`.

Usage

```r
## S3 method for class 'mudata'
rbind(..., validate = TRUE)
```

Arguments

- `...`: `mudata` objects to combine
- `validate`: Flag to validate the final object using `validate_mudata`.

Value

A `mudata` object

Examples

```r
rbind(
  kentvillegreenwood %>%
  select_params(maxtemp) %>%
  select_locations(starts_with("KENT")),
  kentvillegreenwood %>%
  select_params(mintemp) %>%
  select_locations(starts_with("GREEN"))
)
```

rename_locations  Rename identifiers in a mudata object

Description

These functions rename locations, datasets, params, and columns, making sure internal consistency is maintained. These functions use dplyr syntax for renaming (i.e. the `rename` function). This syntax can also be used while subsetting using `select_locations` and family.
rename_locations

Usage

rename_locations(.data, ...)

## Default S3 method:
rename_locations(.data, ...)

rename_params(.data, ...)

## Default S3 method:
rename_params(.data, ...)

rename_datasets(.data, ...)

## Default S3 method:
rename_datasets(.data, ...)

classify_datasets(.data, ...)

## Default S3 method:
classify_datasets(.data, ...)

rename_columns(.data, ...)

## Default S3 method:
rename_columns(.data, ...)

Arguments

.data A mudata object

... Variables to rename in the form new_var = old_var

Value

A modified mudata object

See Also

rename, select_locations

Examples

rename_datasets(kentvillegreenwood, avalley = ecclimate)
rename_locations(kentvillegreenwood, Greenwood = starts_with("GREENWOOD"))
rename_params(kentvillegreenwood, max_temp = maxtemp)
rename_columns(kentvillegreenwood, lon = longitude, lat = latitude)
select_datasets

---

**second_lake_temp**  
*Second Lake Thermistor String Data*

**Description**

Temperatures at multiple depths in the water column for a season at Second Lake, Lower Sackville, Nova Scotia, Canada.

**Usage**

```r
second_lake_temp
```

**Format**

A `mudata` object

**References**


**Examples**

```r
print(second_lake_temp)
```

---

**select_datasets**  
*Subset a mudata object by identifier*

**Description**

These functions use `dplyr`-like selection syntax to quickly subset a `mudata` object by param, location, or dataset. Params, locations, an datasets can also be renamed using keyword arguments, identical to `dplyr` selection syntax.

**Usage**

```r
select_datasets(.data, ...)
```

```r
## Default S3 method:
select_datasets(.data, ..., .factor = FALSE)
```

```r
select_locations(.data, ...)
```

```r
## Default S3 method:
```
select_locations(.data, ..., .factor = FALSE)

select_params(.data, ...)

## Default S3 method:
select_params(.data, ..., .factor = FALSE)

**Arguments**

- `.data` A mudata object
- `...` Quoted names, bare names, or helpers like `starts_with`, `contains`, `ends_with`, `one_of`, or `matches`.
- `.factor` If TRUE, the new object will keep the order specified by converting columns to factors. This may be useful for specifying order when using ggplot2.

**Value**

A subsetted mudata object.

**See Also**

`select`, `rename_locations`, `distinct_locations`, `filter_locations`

**Examples**

# renaming can be handy when locations are verbosely named
ns_climate %>%
  select_locations(sable_island = starts_with("SABLE"),
                  nappan = starts_with("NAPPAN"),
                  baddeck = starts_with("BADDECK")) %>%
  select_params(ends_with("temp"))

# can also use quoted values
long_lake %>%
  select_params("Pb", "As", "Cr")

# can also use negative values to remove params/datasets/locations
long_lake %>%
  select_params(-Pb)

# to get around non-standard evaluation, use one_of()
my_params <- c("Pb", "As", "Cr")
long_lake %>%
  select_params(one_of(my_params))
subset.mudata

Description

This object uses standard evaluation to subset a `mudata` object using character vectors of datasets, params, and locations. The result is subsetted such that all rows in the data table are documented in the other tables (provided) they were to begin with. It is preferred to use `select_locations`, `select_params`, and `select_datasets` to subset a `mudata` object, or `filter_data`, `filter_locations`, `filter_params`, and `filter_datasets` to subset by row while maintaining internal consistency.

Usage

```r
## S3 method for class 'mudata'
subset(x, ..., datasets = NULL, params = NULL, locations = NULL)
```

Arguments

- `x`  
  The object to subset
- `...`  
  Used to `filter` the data table
- `datasets`  
  Vector of datasets to include
- `params`  
  Vector of parameters to include
- `locations`  
  Vector of locations to include

Value

A subsetted `mudata` object

See Also

`select_locations`, `select_params`, `select_datasets`, `filter_data`, `filter_locations`, `filter_params`, and `filter_datasets`

Examples

```r
subset(kentvillegreenwood, params = c("mintemp", "maxtemp"))
```
**Description**

Access components of a mudata object

**Usage**

```r
tbl_data(x)

## Default S3 method:
tbl_data(x)

tbl_data_wide(x, ...)

## Default S3 method:
tbl_data_wide(x, key = "param", value = "value", ...)

tbl_params(x)

## Default S3 method:
tbl_params(x)

tbl_locations(x)

## Default S3 method:
tbl_locations(x)

tbl_datasets(x)

## Default S3 method:
tbl_datasets(x)

tbl_columns(x)

tbl_columns(x)

## S3 method for class 'mudata'
tbl(src, which, ...)

x_columns(x)

## Default S3 method:
x_columns(x)
```
Arguments

- `x, src` A mudata object
- ... Passed to other methods
- `key, value` Passed to `spread`
- `which` Which tbl to extract

Value

The appropriate component

Examples

```r
tbl_data(kentvillegreenwood)
```

update_columns_table  Update the columns table

Description

Update the columns table

Usage

```r
update_columns_table(md, quiet = FALSE)
```

Arguments

- `md` A mudata object
- `quiet` Suppress changes to existing types

Value

A mudata object
update_datasets  

Add documentation to mudata objects

Description

Add documentation to mudata objects

Usage

update_datasets(x, ...)

## Default S3 method:
update_datasets(x, datasets, ...)

update_locations(x, ...)

## Default S3 method:
update_locations(x, locations, datasets, ...)

update_params(x, ...)

## Default S3 method:
update_params(x, params, datasets, ...)

update_columns(x, ...)

## Default S3 method:
update_columns(x, columns, tables, datasets, ...)

Arguments

x  
A mudata object

...  
Key/value pairs (values of length 1)

datasets  
One or more datasets to update

locations  
One or more locations to update

params  
One or more params to update

columns  
One or more columns to update (columns table)

tables  
One or more tables to update (columns table)

Value

A modified version of x
Examples

```r
kentvillegreenwood %>%
  update_datasets("ecclimate", new_key = "new_value") %>%
tbl_datasets()
```

write_mudata

Read/Write mudata objects

Description

These functions will read and write mudata objects to disk using a directory (which contains one .csv file for each table in the object), a ZIP archive (which is a zipped version of the directory format), or a JSON file. The base read/write functions attempt to guess which of these types to use based on the file extension: use the specific read/write function to avoid this.

Usage

```r
write_mudata(md, filename, ...)

read_mudata(filename, ...)

write_mudata_zip(
  md,
  filename,
  overwrite = FALSE,
  validate = TRUE,
  update_columns = TRUE,
  ...
)

read_mudata_zip(filename, validate = TRUE, ...)

write_mudata_dir(
  md,
  filename,
  overwrite = FALSE,
  validate = TRUE,
  update_columns = TRUE,
  ...
)

read_mudata_dir(filename, validate = TRUE, ...)

write_mudata_json(
  md,
  filename,
  overwrite = FALSE,
```
write_mudata

    validate = TRUE,
    update_columns = TRUE,
    pretty = TRUE,
    ...
  )

to_mudata_json(md, validate = TRUE, update_columns = TRUE, pretty = FALSE, ...)

read_mudata_json(filename, validate = TRUE, ...)

from_mudata_json(txt, validate = TRUE, ...)

Arguments

  md       A mudata object
  filename File to read/write (can also be a directory)
  ...      Passed to read/write functions
  overwrite Pass TRUE to overwrite if the file/directory already exists.
  validate Flag to validate mudata object after read or before write
  update_columns Update the columns table "type" column to reflect the internal R types of columns (recommended).
  pretty   Produce pretty or minified JSON output
  txt      JSON text from which to read a mudata object.

Details

These functions are designed to make sure that the read/write operations are as lossless as possible. Some exceptions to this are if date/time columns are not in UTC (in which case they will be converted to UTC before writing), and if table names have characters that are not filesystem safe (allowed characters are [A-Za-z0-9_.-] and others will be stripped).

Examples

# read/write to directory
outfile <- tempfile(fileext=".mudata")
write_mudata(kentvillegreenwood, outfile)
md <- read_mudata(outfile)
unlink(outfile)

# read/write to zip
outfile <- tempfile(fileext=".zip")
write_mudata(kentvillegreenwood, outfile)
md <- read_mudata(outfile)
unlink(outfile)

# read/write to JSON
outfile <- tempfile(fileext=".json")
write_mudata(kentvillegreenwood, outfile)
md <- read_mudata(outfile)
unlink(outfile)
Index

* datasets
  alta_lake, 2
  kentvillegreenwood, 7
  long_lake, 7
  ns_climate, 13
  pocmaj, 14
  pocmajsum, 15
  second_lake_temp, 18
  # alta_lake, 2, 8
  as.mudata (as_mudata), 3
  as_mudata, 3
  bind_cols, 13
  bind_rows, 16
  contains, 19
  distinct, 16
  distinct_columns (distinct_params), 4
  distinct_datasets (distinct_params), 4
  distinct_locations, 19
  distinct_locations (distinct_params), 4
  distinct_params, 4
  ends_with, 19
  filter, 5, 20
  filter_data, 20
  filter_data (filter_datasets), 5
  filter_datasets, 5, 20
  filter_locations, 19, 20
  filter_locations (filter_datasets), 5
  filter_params, 20
  filter_params (filter_datasets), 5
  from_mudata_json (write_mudata), 24
  gather, 13, 14
  is.mudata (is_mudata), 6
  is_mudata, 6
  kentvillegreenwood, 7, 8
  list, 9, 12
  long_lake, 7, 8
  matches, 19
  mudata, 2, 3, 5, 7, 8, 12, 13, 16, 18, 20
  mudata_parse_column
    (mudata_prepare_column), 10
  mudata_parse_tbl
    (mudata_prepare_column), 10
  mudata_prepare_column, 10
  mudata_prepare_tbl
    (mudata_prepare_column), 10
  mutate, 11
  mutate_columns (mutate_data), 11
  mutate_data, 11
  mutate_datasets (mutate_data), 11
  mutate_locations (mutate_data), 11
  mutate_params (mutate_data), 11
  mutate_tbl (mutate_data), 11
  new_mudata, 12
  ns_climate, 8, 13
  one_of, 19
  output_column, 10
  parallel_gather, 13
  pocmaj, 14, 15
  pocmajsum, 15
  print.mudata, 15
  rbind, 16
  rbind.mudata, 16
  read_mudata, 12
  read_mudata (write_mudata), 24
  read_mudata_dir (write_mudata), 24
  read_mudata_json (write_mudata), 24
  read_mudata_zip (write_mudata), 24
  rename, 16, 17
rename_columns (rename_locations), 16
rename_datasets (rename_locations), 16
rename_locations, 16, 19
rename_params (rename_locations), 16
rlang::abort, 12
second_lake_temp, 8, 18
select, 19
select_datasets, 5, 18, 20
select_locations, 5, 16, 17, 20
select_locations (select_datasets), 18
select_params, 20
select_params (select_datasets), 18
spread, 22
src_tbls.mudata (distinct_params), 4
starts_with, 19
subset.mudata, 20
summary.mudata (print.mudata), 15
tbl.mudata (tbl_data), 21
tbl_columns (tbl_data), 21
tbl_data, 21
tbl_data_wide (tbl_data), 21
tbl_datasets (tbl_data), 21
tbl_locations (tbl_data), 21
tbl_params (tbl_data), 21
tibble, 8
to_mudata_json (write_mudata), 24
update_columns (update_datasets), 23
update_columns_table, 22
update_datasets, 23
update_locations (update_datasets), 23
update_params (update_datasets), 23
validate_mudata, 9, 16
validate_mudata (new_mudata), 12
write_mudata, 12, 24
write_mudata_dir (write_mudata), 24
write_mudata_json (write_mudata), 24
write_mudata_zip (write_mudata), 24
x_columns (tbl_data), 21