Package ‘cubelyr’

May 8, 2026

Title A Data Cube 'dplyr' Backend

Version 1.0.2

Description An implementation of a data cube extracted out of 'dplyr' for backward compatibility.

License MIT + file LICENSE

URL https://github.com/hadley/cubelyr

BugReports https://github.com/hadley/cubelyr/issues

Depends R (>= 3.3)

Imports dplyr, glue, pillar, purrr, rlang, tibble, tidyselect

Suggests covr, testthat (>= 2.1.0)

Encoding UTF-8

LazyData true

RoxygenNote 7.2.1

NeedsCompilation no

Author Hadley Wickham [aut, cre], RStudio [cph]

Maintainer Hadley Wickham <hadley@rstudio.com>

Repository CRAN

Date/Publication 2022-11-07 15:50:02 UTC

Contents

as.table.tbl_cube *Coerce a* tbl_cube *to other data structures*

Description

Supports conversion to tables, data frames, tibbles.

For a cube, the data frame returned by `tibble::as_tibble()` resulting data frame contains the dimensions as character values (and not as factors).

Usage

```
## S3 method for class 'tbl_cube'
as.table(x, ..., measure = 1L)

## S3 method for class 'tbl_cube'
as.data.frame(x, ...)

## S3 method for class 'tbl_cube'
as_tibble(x, ...)
```

Arguments

x	a tbl_cube
...	Passed on to individual methods; otherwise ignored.
measure	A measure name or index, default: the first measure

Value

A table, data frame, or tibble, as appropriate.

as.tbl_cube *Coerce an existing data structure into a* tbl_cube

Description

Coerce an existing data structure into a tbl_cube

Usage

```
as.tbl_cube(x, ...)

## S3 method for class 'array'
as.tbl_cube(
  x,
  dim_names = names(dimnames(x)),
```

```
  met_name = deparse(substitute(x)),
  ...
)

## S3 method for class 'table'
as.tbl_cube(x, dim_names = names(dimnames(x)), met_name = "Freq", ...)

## S3 method for class 'matrix'
as.tbl_cube(
  x,
  dim_names = names(dimnames(x)),
  met_name = deparse(substitute(x)),
  ...
)

## S3 method for class 'data.frame'
as.tbl_cube(x, dim_names = NULL, met_name = guess_met(x), ...)
```

Arguments

x	an object to convert. Built in methods will convert arrays, tables and data frames.
...	Passed on to individual methods; otherwise ignored.
dim_names	names of the dimensions. Defaults to the names of the dimnames().
met_name	a string to use as the name for the measure.

Value

A tbl_cube.

---

nasa *NASA spatio-temporal data*

Description

This data comes from the ASA 2007 data expo, https://community.amstat.org/jointscsg-section/dataexpo/dataexpo2006. The data are geographic and atmospheric measures on a very coarse 24 by 24 grid covering Central America. The variables are: temperature (surface and air), ozone, air pressure, and cloud cover (low, mid, and high). All variables are monthly averages, with observations for Jan 1995 to Dec 2000. These data were obtained from the NASA Langley Research Center Atmospheric Sciences Data Center (with permission; see important copyright terms below).

Usage

```
nasa
```

Format

A tbl_cube with 41,472 observations.

Dimensions

- `lat`, `long`: latitude and longitude
- `year`, `month`: month and year

Measures

- `cloudlow`, `cloudmed`, `cloudhigh`: cloud cover at three heights
- `ozone`
- `surftemp` and `temperature`
- `pressure`

Examples

```
nasa
```

`tbl_cube` *A data cube tbl*

Description

A cube tbl stores data in a compact array format where dimension names are not needlessly repeated. They are particularly appropriate for experimental data where all combinations of factors are tried (e.g. complete factorial designs), or for storing the result of aggregations. Compared to data frames, they will occupy much less memory when variables are crossed, not nested.

Usage

```
tbl_cube(dimensions, measures)
```

Arguments

`dimensions`	A named list of vectors. A dimension is a variable whose values are known before the experiment is conducted; they are fixed by design (in **reshape2** they are known as id variables). `tbl_cubes` are dense which means that almost every combination of the dimensions should have associated measurements: missing values require an explicit NA, so if the variables are nested, not crossed, the majority of the data structure will be empty. Dimensions are typically, but not always, categorical variables.
`measures`	A named list of arrays. A measure is something that is actually measured, and is not known in advance. The dimension of each array should be the same as the length of the dimensions. Measures are typically, but not always, continuous values.

Details

tbl_cube support is currently experimental and little performance optimisation has been done, but you may find them useful if your data already comes in this form, or you struggle with the memory overhead of the sparse/crossed of data frames. There is no support for hierarchical indices (although I think that would be a relatively straightforward extension to storing data frames for indices rather than vectors).

Value

A new data cube with class tbl_cube.

Implementation

Manipulation functions:

- select() (M)
- summarise() (M), corresponds to roll-up, but rather more limited since there are no hierarchies.
- filter() (D), corresponds to slice/dice.
- mutate() (M) is not implemented, but should be relatively straightforward given the implementation of summarise.
- arrange() (D?) Not implemented: not obvious how much sense it would make

Joins: not implemented. See vignettes/joins.graffle for ideas. Probably straightforward if you get the indexes right, and that's probably some straightforward array/tensor operation.

See Also

as.tbl_cube() for ways of coercing existing data structures into a tbl_cube.

Examples

```
library(dplyr)
# The built in nasa dataset records meterological data (temperature,
# cloud cover, ozone etc) for a 4d spatio-temporal dataset (lat, long,
# month and year)
nasa
head(as.data.frame(nasa))

titanic <- as.tbl_cube(Titanic)
head(as.data.frame(titanic))

admit <- as.tbl_cube(UCBAdmissions)
head(as.data.frame(admit))

as.tbl_cube(esoph, dim_names = 1:3)

# Some manipulation examples with the NASA dataset --------------------------
```

```
# select() operates only on measures: it doesn't affect dimensions in any way
select(nasa, cloudhigh:cloudmid)
select(nasa, matches("temp"))

# filter() operates only on dimensions
filter(nasa, lat > 0, year == 2000)
# Each component can only refer to one dimensions, ensuring that you always
# create a rectangular subset
## Not run: filter(nasa, lat > long)

# Arrange is meaningless for tbl_cubes

by_loc <- group_by(nasa, lat, long)
summarise(by_loc, pressure = max(pressure), temp = mean(temperature))
```

Index