# Package 'tsdb'

September 22, 2023

**Type** Package

**Title** Terribly-Simple Data Base for Time Series

**Version** 1.1-0

**Date** 2023-02-20

**Maintainer** Enrico Schumann <es@enricoschumann.net>

**Description** A terribly-simple data base for numeric time series, written purely in R, so no external database-software is needed. Series are stored in plain-text files (the most-portable and enduring file type) in CSV format. Timestamps are encoded using R's native numeric representation for 'Date'/'POSIXct', which makes them fast to parse, but keeps them accessible with other software. The package provides tools for saving and updating series in this standardised format, for retrieving and joining data, for summarising files and directories, and for coercing series from and to other data types (such as 'zoo' series).

**License** GPL-3

**Imports** datetimeutils, fastmatch, utils, zoo

**Suggests** data.table, tinytest

**URL** http://enricoschumann.net/R/packages/tsdb/, https://github.com/enricoschumann/tsdb, https://gitlab.com/enricoschumann/tsdb

**NeedsCompilation** no

**Author** Enrico Schumann [aut, cre] (<https://orcid.org/0000-0001-7601-6576>)

## R topics documented:

---

tsdb-package *Terribly-Simple Database for Time Series*

---

## Description

A terribly-simple data base for numeric time series, written purely in R, so no external database-software is needed. Series are stored in plain-text files (the most-portable and enduring file type) in CSV format; timestamps are encoded using R's native numeric representation for `Date`/`POSIXct`, which makes them fast to parse, but keeps them accessible with other software. The package provides tools for saving and updating series in this standardised format, for retrieving and joining data, for summarising files and directories, and for coercing series from and to other data types (such as 'zoo' series).

## Details

See the functions `ts_table` and `as.ts_table` for creating a `ts_table`.

See `write_ts_table` and `read_ts_tables` for storing and loading a `ts_table` (or several).

For getting started, see the tutorial at `https://gitlab.com/enricoschumann/tsdb/blob/master/README.org` or `https://github.com/enricoschumann/tsdb/blob/master/README.org` .

## Author(s)

Enrico Schumann

## See Also

`ts_table` and `as.ts_table` for creating a `ts_table`

`write_ts_table` and `read_ts_tables` for storing and loading a `ts_table`

`as.ts_table` *Coerce to ts_table*

## Description

Coerce objects to `ts_table`

## Usage

```
as.ts_table(x, ...)

## S3 method for class 'zoo'
as.ts_table(x, columns, ...)
```

## Arguments

| | |
|---|---|
| `x` | object to be coerced to ts_table |
| `columns` | character |
| `...` | arguments to be passed to other methods |

## Details

A generic function for coercing objects to class `ts_table`.

## Value

a `ts_table`

## Author(s)

Enrico Schumann

## See Also

`read_ts_tables`

## Examples

```
library("zoo")
as.ts_table(zoo(1:5, Sys.Date()-5:1),  ## note that the "columns"
            columns = "value")         ## must be specified
```

file_info *Information about Data File*

## Description

Provides information about data stored in file: columns, number of observations, range of timestamps.

## Usage

```
file_info(dir, file)
```

## Arguments

| | |
|---|---|
| dir | character |
| file | character |

## Details

Provide information, such as number of entries, of specified files.

It is recommended that code that uses the returned information to alter or write tables, should explicitly check whether a table exists (column `exists` in the returned `data.frame`). For instance, a value of `NA` for `min.timestamp` would occur for a non-existing file, but also if the file could not be read for some reason.

## Value

An object of type `file_info`, which is a `data.frame` with information such as whether a file exists, minimum and maximum timestamp, and more.

## Author(s)

Enrico Schumann

## See Also

`ts_table`

## Examples

```
ts <- ts_table(1:3, as.Date("2018-12-3") + 1:3, columns = "A")
d <- tempdir()
write_ts_table(ts, file = "temp", dir = d)
file_info(d, "temp")
```

read_ts_tables *Read Time-Series Data from Files*

## Description

Read time-series data from files and merge them.

## Usage

```
read_ts_tables(file, dir, t.type = "guess",
               start, end, columns,
               return.class = NULL,
               drop.weekends = FALSE,
               column.names = "%dir%/%file%::%column%",
               backend = "csv",
               read.fn = NULL,
               frequency = "1 sec",
               timestamp)
```

## Arguments

| | |
|---|---|
| file | character |
| dir | character |
| t.type | character: guess, Date or POSIXct |
| start | a timestamp: either of classes Date or POSIXct (possibly including timezone information), or a character string. Strings are passed to as.Date/as.POSIXct. Note in particular that a string of the form "YYYY-MM-DD HH:MM:SS", when passed to as.POSIXct, will be interpreted as a datetime in the current timezone.<br>It is best to always specify start: if start is missing, the function will use the first timestamp of the first time-series it reads. |
| end | a timestamp: either of classes Date or POSIXct (possibly including timezone information), or a character string. Strings are passed to as.Date/as.POSIXct. Note in particular that a string of the form "YYYY-MM-DD HH:MM:SS", when passed to as.POSIXct, will be interpreted as a datetime in the current timezone.<br>It is best to always specify end: if end is missing, the function will use the current time (which may not be appropriate: for instance, when forecasts are stored). |
| columns | character. |
| return.class | NULL (default) or character: if NULL, a list is returned. Also supported are zoo, data.frame and ts_table. |
| drop.weekends | logical |
| column.names | character: a format string for column names; may contain %dir%, %file%, and %column%. It is only used when return.class is data.frame or zoo. |
| backend | character: currently, only 'csv' is fully supported |

| | |
|---|---|
| read.fn | NULL or character: use 'fread' to use fread from package **data.table** |
| frequency | character; used compute a regular grid between start and end. The argument is only used when t.type is POSIXct (or guessed to be POSIXct) and no timestamp is specified. If set to NA, the function will first read all files and compute timestamp as the union of all files' timestamps. |
| timestamp | a vector of timestamps: if specified, only data at the times in timestamp are selected |

## Details

Read time-series data from CSV files.

## Value

When return.class is NULL, a list:

| | |
|---|---|
| data | a numeric matrix |
| timestamp | Date or POSIXct |
| columns | character |
| file.path | character |

Otherwise an object of class as specified by argument return.class.

## Author(s)

Enrico Schumann

## See Also

write_ts_table

## Examples

```
t1 <- ts_table(1:3, as.Date("2018-12-3") + 1:3, columns = "A")
t2 <- ts_table(4:5, as.Date("2018-12-3") + 1:2, columns = "A")

d <- tempdir()  ## this is just an example.
                ## Actual (valuable) data should never
                ## be stored in a tempdir!

write_ts_table(t1, dir = d, file = "t1")
write_ts_table(t2, dir = d, file = "t2")

read_ts_tables(c("t1", "t2"),
               dir = d, columns = "A",
               return.class = "zoo",
               column.names = "%file%.%column%")
```

ts_table *Create ts_table*

## Description

Create a ts_table.

## Usage

```
ts_table(data, timestamp, columns)
```

## Arguments

| | |
|---|---|
| data | numeric |
| timestamp | Date or POSIXct |
| columns | column names |

## Details

Create a time-series table (ts_table). A ts_table is a numeric matrix, so there is always a dim attribute. For a ts_table x, you get the number of observations with dim(x)[1L].

Attached to this matrix are several attributes:

**timestamp** a vector: the numeric representation of the timestamp

**t.type** character: the class of the original timestamp, either Date or POSIXct

**columns** a character vector that provides the columns names

There may be other attributes as well, but these three are always present.

Timestamps must be of class Date or POSIXct (POSIXlt is converted). A tzone attribute is dropped.

A ts_table is not meant as a time-series class. For most computations (plotting, calculation of statistics, etc.), the ts_table must first be coerced to zoo, xts, a data.frame or a similar data structure. Methods that perform such coercions are responsible for converting the numeric timestamp vector to an actual timestamp. For this, they may use the function ttime ('translate time').

## Value

a ts_table

## Author(s)

Enrico Schumann

## See Also

as.ts_table

## Examples

```
ts_table(1:5, Sys.Date() - 5:1, columns = "value")
```

---

`ttime` *Translate Timestamps*

---

## Description

Translate a vector of timestamps.

## Usage

```
ttime(x, from = "datetime", to = "numeric", tz = "",
      strip.attr = TRUE, format = "%Y-%m-%d")
```

## Arguments

| | |
|---|---|
| `x` | numeric |
| `from` | character: `datetime`, `numeric` or `character` |
| `to` | character: `numeric`, `Date` or `POSIXct` |
| `tz` | character |
| `strip.attr` | logical: strip attributes; in particular, timezone information |
| `format` | character |

## Details

`ttime` ('translate time') converts timestamps between formats.

## Author(s)

Enrico Schumann

## See Also

`ts_table`

## Examples

```
ttime(Sys.Date())
ttime(17397, from = "numeric", to = "Date")
```

write_ts_table *Write Time-Series Data to File*

## Description

Write time-series data to files.

## Usage

```
write_ts_table(ts, dir, file, add = FALSE, overwrite = FALSE,
               replace.file = FALSE, backend = "csv")
```

## Arguments

| | |
|---|---|
| ts | a ts_table |
| dir | character |
| file | character |
| add | logical: if TRUE, add data with timestamps that are not in a file. |
| overwrite | logical: overwrite existing file when data differs. overwrite implies add. |
| replace.file | logical: if TRUE, an existing file is deleted and replaced by a new file (i.e. containing ts) |
| backend | a string; currently, only csv is supported |

## Details

The function takes a ts_table and writes it to a file.

If the file already exists and both add and overwrite are FALSE (the default), nothing is written.

When add is TRUE, the function checks if ts contains timestamps not yet in the file and, if there are any, writes only those data.

When overwrite is TRUE, the function merges all observations in the file with those in ts and writes the result back to the file. If ts contains timestamps that were already in the file, the data in the file are overwritten. Note that no data will be removed from the file: timestamps not in ts remain unchanged in the file.

## Value

Invisibly, the number of data rows written to a file.

## Author(s)

Enrico Schumann

## See Also

read_ts_tables

## Examples

```
t1 <- ts_table(1:3, as.Date("2018-12-3") + 1:3, columns = "A")
t2 <- ts_table(4:5, as.Date("2018-12-3") + 1:2, columns = "A")

d <- tempdir()  ## this is just an example.
                ## Actual (valuable) data should never
                ## be stored in a tempdir!

write_ts_table(t1, dir = d, file = "t1")
write_ts_table(t2, dir = d, file = "t2")

read_ts_tables(c("t1", "t2"),
               dir = d, columns = "A",
               return.class = "zoo",
               column.names = "%file%.%column%")
```

# Index