Package ‘atsd’

January 29, 2018

Title Support Querying Axibase Time-Series Database

Version 1.2.0

Date 2018-01-29

Author Axibase Corporation

Maintainer Mikhail Zvagelsky <mikhail.zvagelsky@axibase.com>

URL https://github.com/axibase/atsd-api-r/

Description Provides functions for retrieving time-series and related meta-data such as entities, metrics, and tags from the Axibase Time-Series Database (ATSD). ATSD is a non-relational clustered database used for storing performance measurements from IT infrastructure resources: servers, network devices, storage systems, and applications.

Depends R (>= 3.1.2)

License Apache License 2.0

Imports RCurl (>= 1.95.4.5), httr (>= 0.6.1)

Suggests zoo, knitr, chron, timeDate, testthat, rmarkdown

VignetteBuilder knitr

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2018-01-29 10:04:43 UTC

R topics documented:

atsd *Support querying Axibase Time-Series Database.*

Description

The package lets you query Axibase Time-Series Database (ATSD) for time-series data and forecasts, and save your series into ATSD. List of package functions:

`create_entity` – creates a new entity and it's tags or replaces the tags of an existing entity.

`get_metrics` – get information about the metrics collected by ATSD.

`get_entities` – get information about the entities collected by ATSD.

`get_series_tags` – get unique time series tags for the metric.

`query` – get time-series data and forecasts from ATSD.

`save_series` – save time series in ATSD.

`set_connection`, `save_connection`,`show_connection` - are used to manage connection with ATSD: set up and store the url, user name, and password, configure cryptographic protocol and enforce SSL certificate validation in the case of https connection.

`to_zoo` - convert a time-series data frame to 'zoo' object for manipulating irregular time-series with built-in functions in zoo package.

`update_entity` – update tags and enabled status of an entity.

Type `browseVignettes(package = "atsd")` to view the complete package documentation and usage examples.

Author(s)

Axibase, api@axibase.com

`create_entity` *Create an entity with specified tags or replace the tags of an existing entity.*

Description

This method creates a new entity and it's tags or replaces the tags of an existing entity. If only a subset of tags is provided for an existing entity, the remaining tags will be deleted.

Usage

```
create_entity(entity, tag_names = character(0), tag_values = character(0),
  enabled = TRUE, verbose = FALSE)
```

Arguments

`entity`	Required argument, the name of new entity. To modify some of tags of existing entity and do not change remaining tags use the `update_entity` function.
`tag_names`	Optional argument, a character vector of names of tags.
`tag_values`	Optional argument, a character vector of values of tags. This vector should has the same length as the `tag_names` vector.
`enabled`	Optional boolean argument. If `enabled = TRUE` the entity will be enabled, if `enabled = FALSE` the entity will be disabled. The default value is `enabled = TRUE`.
`verbose`	Optional boolean argument, `FALSE` by default. If `verbose = FALSE` then console output will be suppressed.

Value

codeTRUE if creation/replace was successful, `FALSE` — otherwise.

`get_entities` *Get information about entities from Axibase Time-Series Database.*

Description

This function fetches a list of entities from ATSD, and convert it to a data frame.

Usage

```
get_entities(expression = "", active = "", tags = "*", limit = "",
  verbose = TRUE)
```

Arguments

`expression`	Optional string argument. Select entities matching particular name pattern and/or user-defined entity tags. The syntax of the `expression` argument is explained in the package vignette. Type `browseVignettes(package = "atsd")` to see the vignette.
`active`	Optional string argument: "true" or "false". Filter entities by `lastInsertTime`. If `active = "true"`, only entities with positive lastInsertTime are included in the response.
`tags`	Optional string argument. User-defined entity tags to be included in the response. By default, all the tags will be included.
`limit`	Optional integer argument. If limit > 0, the response shows the top-N entities ordered by name.
`verbose`	Optional boolean argument. If `verbose = FALSE` then all console output will be suppressed. By default, `verbose = TRUE`.

Value

A data frame. Each row of the data frame corresponds to an entity and its tags: `name`, `enabled`, `lastInsertTime` and user-defined entity tags as requested by the 'tags' argument. For more information look at the package vignette: `browseVignettes(package = "atsd")`.

See Also

Visit http://axibase.com/axibase-time-series-database/ for information about ATSD.

Examples

```
## Not run:
# get all entities and include all their tags in the data frame
get_entities()

# get all active entities and include all their tags in the data frame
get_entities(active = "true")

# Get the top 2 entities whose 'name'  and user-defined entity tag, 'app',
# match to the expression. Include the tag, 'app', into response
# and exclude oter user-defined entity tags.
get_entities(expression = "name like 'nur*' and lower(tags.app) like '*hbase*'",
             tags = "app", limit = 2)

## End(Not run)
```

`get_metrics` *Get information about metrics from Axibase Time-Series Database.*

Description

This function fetches a list of metrics and their tags from ATSD, and converts it to a data frame.

Usage

```
get_metrics(expression = "", active = "", tags = "*", limit = 0,
  verbose = TRUE)
```

Arguments

`expression`	Optional string argument. Select metrics matching particular name pattern and/or user-defined metric tags. The syntax of the `expression` argument is explained in the package vignette. Type `browseVignettes(package = "atsd")` to see the vignette.
`active`	Optional string argument: "true" or "false". Filter metrics by `lastInsertTime`. If `active = "true"`, only metrics with positive lastInsertTime are included in the response.
`tags`	Optional string argument. User-defined metric tags to be included in the response. By default, all the tags will be included.
`limit`	Optional integer argument. If limit > 0, the response shows the top-N metrics ordered by name.
`verbose`	Optional boolean argument. If `verbose = FALSE` then all console output will be suppressed. By default, `verbose = TRUE`.

Value

A data frame. Each row of the data frame corresponds to a metric and its tags: `name`, `counter`, `lastInsertTime` and user-defined metric tags as requested by the 'tags' argument. For more information view the package vignette: `browseVignettes(package = "atsd")`.

See Also

Visit `http://axibase.com/axibase-time-series-database/` for information about ATSD.

Examples

```
## Not run:
# get all metrics and include all their tags in the data frame
get_metrics()

# get the top 100 active metrics which have tag, 'table',
# include this tag into response and exclude oter user-defined metric tags
get_metrics(expression = "tags.table != ''", active = "true",
            tags = "table", limit = 100)

# get metrics which have user-defined metric tag, 'table',
# and whose name starts with 'cpu'
get_metrics(expression = "name like 'cpu*' and tags.table != ''")

# more complitcated expressions
get_metrics(expression = "likeAll(name, list('*disk*,*use*'))")
get_metrics(expression = "(name like 'cpu*' or tags.source = '') and tags.table like 'BC*'")

## End(Not run)
```

get_series_tags *Get unique series tags for the metric.*

Description

The function determines time series collected for a given metric. For each time series it lists tags associated with the series, and last time the series was updated. The list of fetched time series is based on data stored on disk for the last 24 hours.

Usage

```
get_series_tags(metric, entity = NA, verbose = TRUE)
```

Arguments

metric	Required string argument. The name of the metric you want to get data for. For example, metric = "disk_used_percent". To obtain a list of metrics collected by ATSD use the get_metrics function.
entity	Optional string argument. The name of the entity you want to get data for. If not provided, then data for all entities will be fetched for the specified metric. Obtain the list of entities and their tags with the get_entities function.
verbose	Optional boolean argument. If verbose = FALSE then all console output will be suppressed. By default, verbose = TRUE.

Value

A data frame. Each row of the data frame corresponds to a time series, and contains the series unique tags, and last time the series was updated. For more information view the package vignette: browseVignettes(package = "atsd").

Examples

```
## Not run:
# get all time series and their tags collected by ATSD for the "disk_used_percent" metric
get_series_tags(metric = "disk_used_percent")
```

```
# get all time series and their tags for the "disk_used_percent" metric
# end "nurswgvml007" entity
get_series_tags(metric = "disk_used_percent", entity = "nurswgvml007")

## End(Not run)
```

`query` *Fetch time-series historic data or forecasts from Axibase Time-Series Database.*

Description

This function fetches time-series from ATSD and creates a data frame from it.

Usage

```
query(metric, entity = NA, entity_group = NA, entity_expression = NA,
  tags = character(), selection_interval, end_time = NA,
  aggregate_interval = NA, aggregate_statistics = "Avg",
  interpolation = "None", export_type = "History", verbose = TRUE)
```

Arguments

`metric`	Required string argument. The name of the metric you want to get data for. For example, `metric = "disk_used_percent"`. To obtain a list of metrics collected by ATSD use the `get_metrics` function.
`entity`	Optional string argument. The name of the entity you want to get data for. If not provided, then data for all entities will be fetched for the specified metric. Obtain the list of entities and their tags with the `get_entities` function.
`entity_group`	Optional string argument. You could specify a group of entities and extract data for entities from this group. For example, `entity_group = "HP Servers"`.
`entity_expression`	Optional string argument. Select entities matching particular name pattern and/or user-defined entity tags. The syntax of the `entity_expression` argument is explained in the package vignette. Type `browseVignettes(package = "atsd")` to see the vignette.
`tags`	Optional string vector argument. List of user-defined series tags to filter the fetched time-series data, for example, `c("disk_name=sda1", "mount_point=/")`.
`selection_interval`	Required string argument. This is the time interval for which the data will be selected. Specify it as "n-unit", where unit is a Second, Minute, Hour, Day, Week, Month, Quarter, or Year and n is the number of units, for example, "3-Week" or "12-Hour".

`end_time`	Optional string argument. The end time of the selection interval, for example, `end_time = "date('2014-12-27')"`. If not provided, the current time will be used. Specify the date and time, or use one of the supported expressions: `https://github.com/axibase/atsd/blob/master/shared/calendar.md#keywords`. For example, `'current_day'` would set the end of selection interval to 00:00:00 of the current day.
`aggregate_interval`	Optional string argument. The length of the aggregation interval. The period of produced time-series will be equal to the `aggregate_interval`. The value for each period is computed by the `aggregate_statistics` function applied to all samples of the original time-series within the period. The format of the `aggregate_interval` is the same as for the `selection_interval` argument, for example, "1-Minute".
`aggregate_statistics`	Optional string vector argument. The statistic function used for aggregation. List of available functions: "Avg", "Min", "Max", "Sum", "Count", "StDev", "WAvg", "WTAvg", "Percentile 50", "Percentile 75", "Percentile 90", "Percentile 95", "Percentile 99", "Percentile 99.5", "Percentile 99.9". Multiple values are supported, for example, c("Min", "Avg", "StDev"). The default value is "Avg".
`interpolation`	Optional string argument If aggregation is enabled, then the values for the periods without data will be computed by one of the following interpolation functions: "None", "Linear", "Step". The default value is "None".
`export_type`	Optional string argument. It can take one of two values: "History" or "Forecast". The default value is "History". For example, `export_type = "Forecast"`.
`verbose`	Optional boolean argument. If `verbose = FALSE` then all console output will be suppressed. By default, `verbose = TRUE`.

Details

The function has only two required arguments: `metric` and `selection_interval`.
Type `browseVignettes(package = "atsd")` to view the complete package documentation and usage examples.

Value

The function returns a data frame. It could be empty if no data match your query or if your request could not be processed by ATSD server. In any case you will get a console diagnostic message with a short description of the server response.

See Also

Visit `http://axibase.com/axibase-time-series-database/` for information about ATSD.

Examples

```
## Not run:
# Create data frame which contains time series for the given metric
# and all entities for the last 1 hour.
```

```
dfr <- query(metric = "disk_used_percent", selection_interval = "1-Hour")

dfr <- query( export_type = "Forecast",
              metric = "disk_used_percent",
              entity_group = "Linux",
              tags = c("mount_point=/boot", "file_system=/dev/sda1"),
              selection_interval = "1-Week",
              aggregate_statistics = c("Avg", "Min", "Max"),
              aggregate_interval = "1-Minute",
              interpolation = "Linear")

# Example of the end_time argument usage.
dfr <- query( metric = "cpu_usage",
              entity = "host-383",
              selection_interval = "1-Day",
              end_time = "date('2015-02-10 10:15:03')")

## End(Not run)
```

---

save_connection *Write connection parameters to configuration file.*

Description

The function writes the connection parameters into configuration file.

Usage

```
save_connection(url = NA, user = NA, password = NA, verify = NA,
  encryption = NA)
```

Arguments

url	Optional string argument. The url of ATSD with the port number.
user	Optional string argument. The user name.
password	Optional string argument. The user's password.
verify	Optional string argument – "yes" or "no". `verify = "yes"` ensures validation of ATSD SSL certificate and `verify = "no"` suppresses the validation (applicable in the case of 'https' protocol).
encryption	Optional string argument. Cryptographic protocol used by ATSD https server. Possible values are: "default", "ssl2", "ssl3", and "tls1" (In most cases, use "ssl3" or "tls1".)

Details

If you call `save_connection()` without arguments, then the current values of the connection parameters (including NAs) will be written to the configuration file. If you provide some arguments, they will be written into the configuration file. If configuration file is absent it will be created in the atsd package folder.

See Also

For more information about the configuration file view the package vignette: `browseVignettes(package = "atsd")`.

Examples

```
## Not run:
# Write the current values of the connection parameters to the configuration file
save_connection()

# Write the user name and the password to the configuration file
save_connection(user = "user001", password = "123456")

# Write all parameters nedeed for https connection to the configuration file
save_connection(url = "https://my.company.com:8443", user = "user001", password = "123456",
                verify = "no", encryption = "ssl3")

## End(Not run)
```

---

save_series — *Save time series into ATSD.*

Description

Save time series from given data frame into ATSD. The data frame should have a column with time stamps and at least one numeric column with values of a metric.

Usage

```
save_series(dfr, time_col = 1, time_format = "%Y-%m-%d %H:%M:%S",
  tz = "GMT", metric_col, metric_name = character(),
  entity_col = numeric(), entity = NA, tags_col = numeric(), tags = NA,
  verbose = TRUE)
```

Arguments

`dfr`	Required argument, a data frame. The data frame should have a column with timestamps and at least one numeric column with values of a metric.
`time_col`	Optional numeric or character argument, default value is 1. Number or name of the column with timestamps. For example, time_col = 1, or time_col = "Timestamp". Read "Time stamps format" section for supported time stamps formats.
`time_format`	Optional string argument, indicates format of time stamps. This argument is used in the case when time stamps format is not clear from their class. The value of this argument can be one of strings: "ms" (for epoch milliseconds), "sec" (for epoch seconds), or format string, for example "%Y-%m-%d %H:%M:%S". This format string will be used to convert provided time stamps to epoch milliseconds before storing time stamps into ATSD. Read "Time stamps format" section for details.

`tz`	Optional string argument. By default, `tz = "GMT"`. Specify time zone, when time stamps are strings formatted as set in the `time_format` argument. For example, `tz = "Australia/Darwin"`. View the "TZ" column of the time zones table for the list of possible values.
`metric_col`	Required argument. Numeric or character vector. Specifies numbers or names of the columns where metrics values are stored. For example, `metric_col = c(2, 3, 4)`, or `metric_col = c("Value", "Avg")` If `metric_name` argument is not given, then names of columns, in low case, are used as names of metrics for saving into ATSD.
`metric_name`	Optional argument. Character vector. Specifies names of metrics. The series pointed by `metric_col` argument are saved in ATSD along with metric names, provided by the `metric_name`. So the number and order of names in the `metric_name` should match to columns in `metric_col`. If `metric_name` argument is not provided, then names of columns, in low case, are used as names of metrics for saving into ATSD.
`entity_col`	Optional argument, should be provided if the entity argument is not given. Number or name of a column with entities. Several entities in the column are allowed. For example, entity_col = 4, or entity_col = "server001".
`entity`	Optional character argument, should be provided if the entity_col argument is not given. Name of the entity.
`tags_col`	Optional argument. Numeric or character vector. Lists numbers or names of the columns containing values of tags. So the name of a column is a tag name, and values in the column are the tag values.
`tags`	Optional argument. Character vector. Lists tags and their values in "tag=value" format. Each provided tag stick to each series. Whitespace symbols are ignored.
`verbose`	Optional boolean argument. If `verbose = FALSE` then all console output will be suppressed. By default, `verbose = TRUE`.

Time stamps format

The list of allowed time stamps types.
– Numeric, in epoch milliseconds or epoch seconds. In that case `time_format = "ms"` or `time_format = "sec"` should be used, and time zone argument `tz` is ignored.
– Object of one of types "Date", "POSIXct", "POSIXlt", "chron" from the chron package or "timeDate" from the timeDate package. In that case arguments `time_format` and `tz` are ignored.
– String, for example, "2015-01-03 10:07:15". In that case `time_format` argument should specify which format string is used for the time stamps. For example, `time_format = "%Y-%m-%d %H:%M:%S"`. Type `?strptime` to see list of format symbols. This format string will be used to convert provided time stamps to epoch milliseconds before store time stamps in ATSD. So time zone, as written in `tz` argument, and standard origin "1970-01-01 00:00:00" are used for conversion. In fact conversion is done with use of command: `as.POSIXct(time_stamp, format = time_format, origin="1970-01-01", tz = tz)`.

Note that time stamps will be stored in epoch milliseconds. So if you put some data into ATSD and then get it back, the time stamps will refer to the same time but in GMT time zone. For example, if you save time stamp `"2015-02-15 10:00:00"` with `tz = "Australia/Darwin"` in ATSD, and then fetch it back, you will get time stamp `"2015-02-15 00:30:00"` because Australia/Darwin time zone has +09:30 shift relatively GMT zone.

Entity specification

You can provide entity name in one of 'entity' or 'entity_col' arguments. In the first case all series will have the same entity. In the second case, if the column of the data frame, specified by 'entity_col', contains several entities, then that entities will be saved along with corresponding series.

Tags specification

The 'tags_col' argument points which columns of the data frame keep tags of time series. The name of each column specified by tags_col argument is a tag name, and the values in the column are the tag values.

Before storing in ATSD the data frame will be split to several data frames, each of them has unique entity and unique list of tags values. This entity and tags are stored in ATSD along with time series from such data frame. NA's and missing values in time series will be ignored.

In 'tags' argument you can specify tags which are the same for all rows (records) of the data frame. So each series value saved in ATSD will have tags, provided in the 'tags' argument.

`set_connection` *Set up parameters of a connection with ATSD.*

Description

The function overrides the connection parameters for the duration of the current R session without changing the configuration file.

Usage

```
set_connection(url = NA, user = NA, password = NA, verify = NA,
  encryption = NA, file = NA)
```

Arguments

`url`	Optional string argument. The url of ATSD with the port number.
`user`	Optional string argument. The user name.
`password`	Optional string argument. The user's password.
`verify`	Optional string argument – "yes" or "no". `verify = "yes"` ensures validation of ATSD SSL certificate and `verify = "no"` suppresses the validation (applicable in the case of 'https' protocol).
`encryption`	Optional string argument. Cryptographic protocol used by ATSD https server. Possible values are: "default", "ssl2", "ssl3", and "tls1" (In most cases, use "ssl3" or "tls1".)
`file`	Optional string argument. The absolute path to the file from which the connection parameters could be read. The file should be formatted as the package configuration file, see the Details section below.

Details

The function overrides the connection parameters for the duration of the current R session without changing the configuration file. If called without arguments the function sets the connection parameters from the configuration file. If the file argument is provided the function use it. In both cases the current values of the parameters became the same as in the file. The file should be a plain text file formatted as the following:

```
# the url of ATSD including port number
url=http://host_name:port_number
# the user name
user=atsd_user_name
# the user's password
password=atsd_user_password
# validate ATSD SSL certificate: yes, no
verify=no
# cryptographic protocol used by ATSD https server:
# default, ssl2, ssl3, tls1
encryption=ssl3
```

In case the `file` argument is not provided, but some of other arguments are specified, the only specified parameters will be changed.

See Also

To see the current values of the connection parameters use the `show_connection` function. To change the configuration file use the `save_connection` function.

Examples

```
# Modify the user
set_connection(user = "user001")

# Modify the cryptographic protocol
set_connection(encryption = "tls1")

# Set up url, user and password
set_connection(url = "http://my.company.com:8088", user = "user001", password = "123456")

# Set up parameters of https connection
set_connection(url = "https://my.company.com:8443", user = "user001", password = "123456",
               verify = "no", encryption = "ssl3")

## Not run:
# Set up parameters from a file
set_connection(file = "/home/user001/atsd_https_connection.txt")

# Set up parameters from the configuration file
set_connection()

## End(Not run)
```

`show_connection` *Show connection parameters.*

Description

The function shows the current values of the connection parameters `url, user, password, verify` and `encryption`. They are used to arrange a connection with ATSD.

Usage

```
show_connection()
```

See Also

You could change the connection parameters with the `set_connection` function and save that changes to the configuration file with the `save_connection` function.

`to_zoo` *Build zoo object from data frame.*

Description

The function builds a zoo object from given data frame. The `timestamp` argument provides a column of the data frame which is used as index for the zoo object. The `value` argument gives series to be saved in the zoo object. If several columns are listed in `value` argument the multivariate zoo object will be built. Information from other columns is ignored. To use this function the 'zoo' package should be installed. To install the 'zoo' package type: `install.packages("zoo")`.

Usage

```
to_zoo(dfr, timestamp = "Timestamp", value = "Value")
```

Arguments

`dfr`	The data frame with columns for time stamps and for values.
`timestamp`	Name or number of a column with time stamps. By default, `timestamp = "Timestamp"`.
`value`	Vector of names or numbers of columns with series values. By default, `value = "Value"`.

update_entity *Update tags and enabled status of an entity.*

Description

Update specified tags and enabled status of an existing entity. Tags that are not specified are left unchanged.

Usage

```
update_entity(entity, tag_names = character(0), tag_values = character(0),
  enabled = NA, verbose = FALSE)
```

Arguments

`entity`	Required argument, an entity name. The entity should exist into ATSD. In case you want to create new entity use the `create_entity` function.
`tag_names`	Optional argument, a character vector of names of tags.
`tag_values`	Optional argument, a character vector of values of tags. This vector should has the same length as the `tag_names` vector.
`enabled`	Optional boolean argument. If `enabled = TRUE` the entity will be enabled, if `enabled = FALSE` the entity will be disabled, in the default case `enabled = NA` the enabled status of entity will not be changed.
`verbose`	Optional boolean argument, `FALSE` by default. If `verbose = FALSE` then console output will be suppressed.

Value

codeTRUE if update success, `FALSE` — otherwise.

Index