Package: rrlite (via r-universe)

August 29, 2024

Title R Bindings to rlite

Version 0.4.0

Description R bindings to rlite. rlite is a ``self-contained, serverless, zero-configuration, transactional redis-compatible database engine. rlite is to Redis what SQLite is to SQL.".

Depends R (>= 3.1.0)

License GPL-2

LazyData true

Suggests testthat

Imports redux (>= 0.5.0)

SystemRequirements GNU make

RoxygenNote 5.0.1

Repository https://ropensci.r-universe.dev

RemoteUrl https://github.com/ropensci/rrlite

RemoteRef master

RemoteSha 65b83997f8e461201d7830bc59cae4a2f83143f7

Contents

`hirlite` *Interface to rlite*

Description

Create an interface to rlite, with a generated interface to all rlite commands (using `Redux`).

Usage

```
hirlite(...)

rlite_available(...)
```

Arguments

`...`	Named configuration options passed to `redis_config`, used to create the environment (notable keys include `host`, `port`, and the environment variable `REDIS_URL`). In addition to the `Redux` treatment of the configuration, `RLITE_URL` takes precendence over `REDIS_URL`, and a host of `localhost` or `127.0.0.1` will be treated as an in-memory database (`:memory:`).

Examples

```
r <- hirlite()
r$PING()
r$SET("foo", "bar")
r$GET("foo")
```

`rlite_config` *rlite configuration*

Description

rlite configuration settings. Based on the `redis_config` function but with additional tweaks for rlite. The differences between this configuration and `redis_config` is that:

Usage

```
rlite_config(...)
```

Arguments

`...`	Arguments passed to `redis_config`; see that file for more information.

Details

- RLITE_URL takes precendence over REDIS_URL if both are present (otherwise REDIS_URL will still be used).
- A host of localhost or 127.0.0.1, which is redis_config's default, will map to a filename of :memory: for a transient in-memory store.

The port entry will be ignored, but the password and db entries will be used if present. path is equivalent to host.

---

rlite_connection *Create a rlite connection*

Description

Create a rlite connection. This function is designed to be used in other packages, and not directly by end-users. However, it is possible and safe to use. See the hirlite package for the user friendly interface.

Usage

```
rlite_connection(config = rlite_config())
```

Arguments

config	Configuration parameters as generated by rlite_config

Details

This function creates a list of functions, appropriately bound to a pointer to a rlite connection. This is designed for package authors to use so without having to ever deal with the actual pointer itself (which cannot be directly manipulated from R anyway).

The returned list has elements, all of which are functions:

config() The configuration information

reconnect() Attempt reconnection of a connection that has been closed, through serialisation/deserialiation or through loss of internet connection.

command(cmd) Run a Redis command. The format of this command will be documented elsewhere.

pipeline(cmds) Run a pipeline of Redis commands.

subscribe(channel, pattern, callback, envir) Subscribe to a channel or pattern specifying channels. Here, channel must be a character vector, pattern a logical indicating if channel should be interpreted as a pattern, callback is a function to apply to each recieved message, returning TRUE when subscription should stop, and envir is the environment in which to evaluate callback. See below.

Subscriptions

The callback function must take a single argument; this will be the recieved message with named elements `type` (which will be message), `channel` (the name of the channel) and `value` (the message contents). If `pattern` was `TRUE`, then an additional element `pattern` will be present (see the Redis docs). The callback must return `TRUE` or `FALSE`; this indicates if the client should continue quit (i.e., `TRUE` means return control to R, `FALSE` means keep going).

Because the `subscribe` function is blocking and returns nothing, so all data collection needs to happen as a side-effect of the callback function.

There is currently no way of interrupting the client while it is waiting for a message.

Index