Package: tarflow.iquizoo (via r-universe)

May 19, 2026

Title Setup ``targets'' Workflows for ``iquizoo'' Data Processing

Version 3.12.8

Description For ``iquizoo'' data processing, there is already a package called ``preproc.iquizoo'', but eventually the use of it is relied on a workflow. This package is used to build such workflows based on tools provided by ``targets'' package which mimics the logic of ``make'', automating the building processes.

License Apache License (>= 2)

URL https://iquizoo.github.io/tarflow.iquizoo/, https://github.com/iquizoo/tarflow.iquizoo

BugReports https://github.com/iquizoo/tarflow.iquizoo/issues

Depends R (>= 4.1.0)

Imports cachem, cli, DBI, dplyr, glue, jsonlite, memoise, rlang (>= 1.0.0), RMariaDB, tarchetypes, targets, tidyr, tidyselect

Suggests bit64, covr, data.iquizoo (>= 2024.7.14), digest, lifecycle, preproc.iquizoo (>= 2.6.0), roxygen2, testthat (>= 3.0.0), tibble

Remotes iquizoo/data.iquizoo, iquizoo/preproc.iquizoo

Config/testthat/edition 3

Config/testthat/parallel true

Config/testthat/start-first targets

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Config/pak/sysreqs cmake libglpk-dev make libicu-dev libmysqlclient-dev libuv1-dev libxml2-dev

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

Date/Publication 2025-06-23 03:03:44 UTC

RemoteUrl https://github.com/iquizoo/tarflow.iquizoo

RemoteRef HEAD

RemoteSha 47125ffccdec49cf8bfa3d5755be1b1e1b478dea

Contents

`check_source` *Check if the database based on the given data source is ready*

Description

Check if the database based on the given data source is ready

Usage

```
check_source(group = getOption("tarflow.group"))
```

Arguments

`group`	Section identifier in the `default.file`. See `RMariaDB::MariaDB()` for more information.

Value

TRUE if the database is ready, FALSE otherwise.

clean_users_props *Clean users properties*

Description

Clean users properties

Usage

```
clean_users_props(users, props)
```

Arguments

users	A data.frame contains the users properties.
props	A character vector of the users properties to keep.

Value

A data.frame contains the cleaned users properties.

fetch_data *Fetch data from iQuizoo database*

Description

This function is a wrapper of `fetch_iquizoo()`, which is used as a helper function to fetch data from the iQuizoo database.

Usage

```
fetch_data(
  project_id,
  game_id,
  ...,
  what = c("raw_data", "scores"),
  query = NULL,
  suffix_format = "%Y0101"
)
```

Arguments

project_id	The project id to be bound to the query.
game_id	The game id to be bound to the query.
...	Further arguments passed to fetch_iquizoo().
what	What to fetch. Can be either "raw_data" or "scores".
query	A parameterized SQL query. A default query file is stored in the package, which is often enough for most cases. You can also specify your own query file by this argument. See details for more information.
suffix_format	The format of the date suffix. See details for more information.

Details

The data essentially means one of the two types of data: raw data or scores. The raw data is the original data collected from the game, while the scores are the scores calculated by the iQuizoo system. While scores can also be calculated from the raw data, the pre-calculated scores are used to for some quick analysis.

The data is separated by project date, so the table name is suffixed by the project date, which is automatically fetched from the database by this function. You could set the format of the date suffix by suffix_format, although currently you should not need to change it because it probably will not change in the future. Finally, this suffix should be substituted into the query, which should contain an expression to inject the table name, i.e., "{table_name}".

Value

A data.frame contains the fetched data.

---

fetch_iquizoo *Fetch result of query from iQuizoo database*

Description

Fetch result of query from iQuizoo database

Usage

```
fetch_iquizoo(query, ..., params = NULL, group = getOption("tarflow.group"))
```

Arguments

query	A character string containing SQL.
...	Further arguments passed to DBI::dbConnect().
params	The parameters to be bound to the query. Default to NULL, see DBI::dbGetQuery() for more details.
group	Section identifier in the default.file. See RMariaDB::MariaDB() for more information.

Value

A data.frame contains the fetched data.

See Also

`fetch_iquizoo_mem()` for a memoised version of this function.

`fetch_iquizoo_mem` *Memoised version of* `fetch_iquizoo()`

Description

This function is a memoised version of `fetch_iquizoo()`. It is useful when the same query is called multiple times or you want to cache the result. See `memoise::memoise()` and `fetch_iquizoo()` for more details.

Usage

```
fetch_iquizoo_mem(cache = NULL)
```

Arguments

cache	The cache to be used. Default cache could be configured by setting the environment variable `TARFLOW_CACHE` to `"disk"` or `"memory"`. If set `TARFLOW_CACHE` to `"disk"`, the cache will be stored in disk at `~/.cache/tarflow.iquizoo` with a maximal age of 7 days. If set `TARFLOW_CACHE` to `"memory"`, the cache will be stored in memory. You can also set `cache` to a custom cache, see `memoise::memoise()` for more details.

Value

A memoised version of `fetch_iquizoo()`.

See Also

`fetch_iquizoo()` for the original function.

get_users_props_names *Get the names of the user properties.*

Description

Get the names of the user properties.

Usage

```
get_users_props_names()
```

Value

A character vector of the names.

parse_data *Parse Raw Data*

Description

Raw data fetched from iQuizoo database is stored in json string format. This function is used to parse raw json string data as `data.frame()` and store them in a list column.

Usage

```
parse_data(data, col_raw_json = "game_data", name_raw_parsed = "raw_parsed")
```

Arguments

`data`	The raw data.
`col_raw_json`	The column name storing raw json string data.
`name_raw_parsed`	The name used to store parsed data.

Value

A data.frame contains the parsed data.

setup_option_file *Setup MySQL database connection option file*

Description

This function will create a MySQL option file at the given path. To ensure it works, set these environment variables before calling this function:

- `MYSQL_HOST`: The host name of the MySQL server.
- `MYSQL_USER`: The user name of the MySQL server.
- `MYSQL_PASSWORD`: The password of the MySQL server.

Usage

```
setup_option_file(path = NULL, overwrite = FALSE, quietly = FALSE)
```

Arguments

`path`	The path to the option file. Default location is operating system dependent. On Windows, it is `C:/my.cnf`. On other systems, it is `~/.my.cnf`.
`overwrite`	Whether to overwrite the existing option file.
`quietly`	A logical indicates whether message should be suppressed.

Value

NULL (invisible).

setup_templates *Set up templates used to fetch data*

Description

If you want to extract data based on your own parameters, you should use this function to set up your own SQL templates. Note that the SQL queries should be parameterized.

Usage

```
setup_templates(
  contents = NULL,
  users = NULL,
  raw_data = NULL,
  scores = NULL,
  progress_hash = NULL
)
```

Arguments

contents	The SQL template file used to fetch contents. At least project_id and game_id columns should be included in the fetched data based on the template. project_id will be used as the only parameter in users and project templates, while all three will be used in raw_data and scores templates.
users	The SQL template file used to fetch users. Usually you don't need to change this.
raw_data	The SQL template file used to fetch raw data. See fetch_data() for details. Usually you don't need to change this.
scores	The SQL template file used to fetch scores. See fetch_data() for details. Usually you don't need to change this.
progress_hash	The SQL template file used to fetch progress hash. Usually you don't need to change this.

Value

A S3 object of class tarflow.template with the options.

---

tar_fetch_data *Generate a set of targets for fetching data*

Description

This target factory is the main part of the tar_prep_iquizoo function. It fetches the raw data and scores for each project and task/game combination.

Usage

```
tar_fetch_data(
  contents,
  what = c("raw_data", "scores"),
  templates = setup_templates(),
  check_progress = TRUE
)
```

Arguments

contents	The contents structure used as the configuration of data fetching.
what	What to fetch.
templates	The SQL template files used to fetch data. See setup_templates() for details.
check_progress	Whether to check the progress hash. If set as TRUE, Before fetching the data, the progress hash objects named as progress_hash_{project_id} will be depended on, which are typically generated by tar_prep_hash(). If the projects are finalized, set this argument as FALSE.

Value

A list of target objects.

`tar_fetch_users` *Generate a set of targets for fetching user information*

Description

The user information is used to identify the users involved in the project.

Usage

```
tar_fetch_users(
  contents,
  subset_users_props = get_users_props_names(),
  templates = setup_templates(),
  check_progress = TRUE
)
```

Arguments

`contents`	The contents structure used as the configuration of data fetching.
`subset_users_props`	The subset of user properties to be fetched. See `get_users_props_names()` for all the available properties.
`templates`	The SQL template files used to fetch data. See `setup_templates()` for details.
`check_progress`	Whether to check the progress hash. Set it as `FALSE` if the project is finalized.

Value

A list of target objects.

`tar_prep_hash` *Generate a set of targets for fetching progress hash*

Description

The progress hash stores the progress of the project, which is used to check whether the project is updated.

Usage

```
tar_prep_hash(contents, templates = setup_templates())
```

Arguments

contents	The contents structure used as the configuration of data fetching.
templates	The SQL template files used to fetch data. See setup_templates() for details.

Details

These objects are named as progress_hash_{project_id} for each project.

Value

A list of target objects.

---

tar_prep_iquizoo *Generate a set of targets for pre-processing of iQuizoo data*

Description

This target factory prepares a set of target objects used to fetch data from iQuizoo database, separated into static branches so that each is for a specific project and task/game combination. Further pre-processing on the fetched data can also be added if requested.

Usage

```
tar_prep_iquizoo(
  params,
  contents,
  ...,
  what = c("raw_data", "scores"),
  action_raw_data = c("all", "parse", "none"),
  combine = NULL,
  subset_users_props = get_users_props_names(),
  templates = setup_templates(),
  check_progress = TRUE,
  cache = NULL
)
```

Arguments

params, contents

Used as the configuration of data fetching. These two arguments are mutually exclusive. If params is specified, it will be used as parameters to be bound to the query, see DBI::dbBind() for more details. The default template requires specifying organization_name, project_name, course_name and game_name, in that order. Set the column as NA to skip that parameter. If contents is specified, it should be a data.frame and will be used directly as the configuration of data fetching. Note contents should at least contain project_id and game_id names.

`...`	For future usage. Should be empty.
`what`	What to fetch. There are basically two types of data, i.e., raw data and scores. The former is the logged raw data for each trial of the tasks/games, and further actions on the fetched raw data can be specified by `action_raw_data`. The latter is the scores calculated by iQuizoo server.
`action_raw_data`	The action to be taken on the fetched raw data. There are two consecutive actions, i.e., raw data parsing and pre-processing. The former will parse the `json` formatted raw data into `data.frame()`s and wrap them into one list column, see `parse_data()` for more details. The latter will calculate indices based on the parsed data, see `preproc.iquizoo::preproc_data()` for more details. If set as `"none"`, neither will be done. If set as `"parse"`, only raw data parsing will be done. If set as `"all"`, both parsing and pre-processing will be done. If `what` is set as `"scores"`, this argument will be ignored.
`combine`	Specify which targets to be combined. Note you should only specify names from `c("scores", "raw_data", "raw_data_parsed", "indices")`. If `NULL`, none will be combined.
`subset_users_props`	The subset of user properties to be fetched. See `get_users_props_names()` for all the available properties.
`templates`	The SQL template files used to fetch data. See `setup_templates()` for details.
`check_progress`	Whether to check the progress hash. Set it as `FALSE` if the project is finalized.
`cache`	The cache to be used in `fetch_iquizoo_mem()`.

Value

A list of target objects.

`tar_prep_raw` *Generate a set of targets for wrangling and pre-processing raw data*

Description

This target factory is the main part of the `tar_prep_iquizoo` function. It wrangles the raw data into a tidy format and calculates indices based on the parsed data.

Usage

```
tar_prep_raw(
  contents,
  action_raw_data = c("parse", "preproc"),
  name_data = "raw_data",
  name_parsed = "raw_data_parsed",
  name_indices = "indices"
)
```

Arguments

`contents`	The contents structure used as the configuration of data fetching.
`action_raw_data`	The action to be taken on the fetched raw data.
`name_data`	The name of the raw data target.
`name_parsed`	The name of the parsed data target.
`name_indices`	The name of the indices target.

Value

A list of target objects.

`use_targets_pipeline` *Create standard data fetching targets pipeline script*

Description

This function creates a standard data fetching targets pipeline script for you to fill in.

Usage

```
use_targets_pipeline()
```

Value

NULL (invisible). This function is called for its side effects.

Index