Package: htce (via r-universe)

September 9, 2024

Title A set of internal tools for managing high-throughput assay data at NEB

Version 0.1.0

Description What the package does (one paragraph).

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1.9000

Depends R (>= 4.1)

Imports assertthat, data.table, DBI, duckdb, generics, glue, rlang, tibble, withr

Repository https://eric-hunt.r-universe.dev

RemoteUrl https://github.com/eric-hunt/htce

RemoteRef HEAD

RemoteSha 92ab8341be8dbe256ccb0cdc746ee2a27f92acfd

Contents

`address_to_well_id` *Retrieve a well_id from barcode and address strings*

Description

Retrieve a well_id from barcode and address strings

Usage

```
address_to_well_id(
  barcode,
  address,
  .db_loc = "./htCE.duckdb",
  .db_con = NULL,
  .pg_load = FALSE
)
```

Arguments

`barcode`	A string - the unique barcode identifying the physical plate
`address`	A string - the well identifier (e.g. A1, H12, etc.)
`.db_loc`	A string - local path to the DuckDB file; passed to `dkdb_collect()`
`.db_con`	A valid DBIConnection object; passed to `dkdb_collect()`
`.pg_load`	A Boolean - use the DuckDB Postgres extension?; passed to `dkdb_collect()`

Value

A double value representing the `well_id` integer primary key which corresponds to the barcode and address arguments

---

`barcode_to_plate_id` *Retrieve a plate_id from a barcode string*

Description

Retrieve a plate_id from a barcode string

Usage

```
barcode_to_plate_id(
  barcode,
  .db_loc = "./htCE.duckdb",
  .db_con = NULL,
  .pg_load = FALSE
)
```

Arguments

barcode	A string - the unique barcode identifying the physical plate
.db_loc	A string - local path to the DuckDB file; passed to dkdb_collect()
.db_con	A valid DBIConnection object; passed to dkdb_collect()
.pg_load	A Boolean - use the DuckDB Postgres extension?; passed to dkdb_collect()

Value

A double value representing the plate_id integer primary key which corresponds to the barcode argument

---

count_newlines *Return the number of lines in a file*

Description

Return the number of lines in a file

Usage

```
count_newlines(path, remove_header = TRUE, ignore_trailing = TRUE)
```

Arguments

path	A string - path to file
remove_header	A Boolean - should the first row be ignored?
ignore_trailing	A Boolean - should a trailing '\n' be ignored?

Value

An integer - the number of lines in the file

db_query *DuckDB Query*

Description

Execute or collect results from a query to a DuckDB database.

Usage

```
dkdb_execute(
  query_string,
  ...,
  .quiet = TRUE,
  .db_loc = "./htCE.duckdb",
  .db_con = NULL,
  .pg_install = FALSE,
  .pg_load = FALSE
)

dkdb_collect(
  query_string,
  ...,
  .quiet = TRUE,
  .db_loc = "./htCE.duckdb",
  .db_con = NULL,
  .pg_install = FALSE,
  .pg_load = FALSE,
  .return = NULL
)
```

Arguments

query_string	A string - SQL query
...	Key-value pairs - additional injected values passed to glue::glue_sql()
.quiet	A Boolean - suppress console messages?
.db_loc	A string - local path to the DuckDB file
.db_con	A valid DBIConnection object
.pg_install	A Boolean - install the DuckDB Postgres extension?
.pg_load	A Boolean - use the DuckDB Postgres extension?
.return	A string - how collected data should be returned, default is a data.frame (alias *df*, see DBI::dbGetQuery()), but can also be coerced to a tibble (alias *tbl*, see tibble::as_tibble()) or data.table (alias *dt*, see data.table::as.data.table())

Details

dkdb_execute executes a query without collecting results.

dkdb_collect collects the results from a query and returns them as a tibble.

Value

A data.frame (default), tibble, or data.table containing the collected query results

---

dkdb_lookup *Lookup something by a set of attributes and return the* id *or a Boolean value indicating if it already exists*

Description

Lookup something by a set of attributes and return the id or a Boolean value indicating if it already exists

Usage

```
dkdb_lookup(
  ...,
  .table = NULL,
  .return = "id",
  .db_loc = "./htCE.duckdb",
  .db_con = NULL,
  .pg_load = FALSE
)
```

Arguments

...	Key-value pairs - unique set of injected arguments (using glue::glue_sql()) for component lookup
.table	A string - the table to query in the database schema
.return	A string - should the returned value be a component_id ("id") or a Boolean from an EXISTS operator subquery ("exists")
.db_loc	A string - local path to the DuckDB file; passed to dkdb_collect()
.db_con	A valid DBIConnection object; passed to dkdb_collect()
.pg_load	A Boolean - use the DuckDB Postgres extension?; passed to dkdb_collect()

Value

A primary key id as a double (.return = "id") or a Boolean value returned from an EXISTS operator subquery (.return = "exixts")

existing_barcodes *Title*

Description

Title

Usage

```
existing_barcodes(
  source_only = FALSE,
  .table = "plates",
  .db_loc = "./htCE.duckdb",
  .db_con = NULL
)
```

Arguments

source_only	A Boolean - only retrieve source plate barcodes
.table	A string - the database table to query ({.table}.barcode)
.db_loc	A string - local path to the DuckDB file; passed to dkdb_collect()
.db_con	A valid DBIConnection object; passed to dkdb_collect()

Value

A named character vector of barcodes that exist in the database

fatools_matrix *Title*

Description

Title

Usage

```
fatools_matrix(m)
```

Arguments

m

`find_skips` *Find and return skip values*

Description

Returns a named list of skip values for importing Echo transfer report metadata and transfer data.

Usage

```
find_skips(file_path)
```

Arguments

`file_path`	A string - the path to the .csv delimited file to read

Value

A named list of skip values to use during Echo transfer report import

`make_submission_files` *Create CE submission files*

Description

Creates tab-delimited files for submitting CE plates to the NEB sequencing core via the LIMS system.

Usage

```
make_submission_files(
  initials = "EH",
  datetime = NULL,
  plate_code = 5,
  num_wells = 96,
  num_plates = 12,
  dest_dir = NULL,
  .pad_well_num = FALSE
)
```

Arguments

initials	A string - your initials
datetime	A date - use base::as.Date to convert a string to a date
plate_code	An integer - a code representing the type of plate
num_wells	An integer - 96 or 384, the number of wells on the submission plate
num_plates	An integer - the number of plate submission files to create
dest_dir	A string - the directory to create the tab-delimited submission files
.pad_well_num	A Boolean - passed to make_well_order() *pad* argument

---

make_well_order *Create a vector of well addresses*

Description

Create a vector of well addresses

Usage

```
make_well_order(num_wells, pad = FALSE)
```

Arguments

num_wells	An integer - the number of wells in the plate format (96 or 384)
pad	A Boolelan - whether or not to pad the column numbers with zeros (e.g. A1 vs. A01)

Value

A character vector of well addresses in top to bottom, left to right order (e.g. for 96 wells, A1 -> H1, A2...H11, A12 -> H12)

---

new_fatools_matrix *Title*

Description

Title

Usage

```
new_fatools_matrix(m = matrix())
```

Arguments

m	

tidy.fatools_matrix *Title*

Description

Title

Usage

```
## S3 method for class 'fatools_matrix'
tidy(x, ...)
```

Arguments

x

validate_fatools_matrix *Title*

Description

Title

Usage

```
validate_fatools_matrix(m)
```

Arguments

m

with_duckdb_connection *DuckDB Connection*

Description

DuckDB connection handling à la `withr::local_db_connection`.

Usage

```
with_duckdb_connection(
  db_file_path,
  read_only = FALSE,
  .local_envir = parent.frame()
)
```

Arguments

`db_file_path`	a string - path to a DuckDB file
`read_only`	a Boolean - passed to `DBI::dbConnect()`; default is `FALSE`
`.local_envir`	an environment - passed to `withr::defer()`; default is `parent.frame()`

Value

A database connection of class *DBIConnection*.

See Also

`DBI::dbConnect()`, `withr::defer()`, `withr::local_db_connection()`, duckdb/duckdb#5525

Index