Package 'uwinutils'

November 30, 2021

Type Package

Title A suite of useful functions to query / report from the UWIN database

Version 0.1.0

Author Mason Fidino

Maintainer Mason Fidino <mfidino@lpzoo.org>

Description This is really only helpful to the handful of folks who have access to the backend of the UWIN database. However, it will hopefully make automated reporting much easier.

License GPL-2

Encoding UTF-8

LazyData true

Depends R (>= 3.5.1),
magrittr,
RMariaDB,
doSNOW,
progress,
foreach

Imports dplyr,
rstudioapi,
lubridate,
stringr

RoxygenNote 7.1.1

R topics documented:

`b2utc` *Back to UTC*

Description

`b2utc` changes the datetime object returned from SQL to UTC.

Usage

```
b2utc(x)
```

Arguments

`x`	a datetime vector

Value

A datetime vector in UTC format

`connect2db` *Connect to UWIN database*

Description

`connect2db` connects to the UWIN camera trapping database. It takes no arguments but will request for the password to the UWIN database when a connection is attempted.

Usage

```
connect2db()
```

Value

If the password is correctly input `connect2db` will return a `MariaDBConnection` called `uwidb.` to the global environment.

Examples

```
## Not run:
connect2db()

## End(Not run)
```

data_source *Connect to UWIN database*

Description

data_source checks the current files in the uwin-dataset folder and let's you know if they were generated from an access db, the UWIN web app, or if it's precleaned.

Usage

```
data_source(x)
```

Details

#' @param x The filepath to a given folder of data within a city.

Value

A character that says where the data came from.

Examples

```
## Not run:
data_source(x = "./chil/1_JA16/")

## End(Not run)
```

gsutil_copy *Copy images from google cloud*

Description

gsutil_copy takes in the object from images_of and copies them to a specified folder.

Usage

```
gsutil_copy(images_to_copy = NULL, output_folder = NULL, ncore = 2)
```

Arguments

images_to_copy	The output data.frame from images_of.
ncore	a numeric scalar. The number of cores to use to copy images from google cloud. It defaults to 2.
output_to_folder	A character vector that denotes the full path of a folder to copy the images to. If the folder does not exist it will be created. A final slash cannot be placed at the end of the character. For example, "C:/users/data" works but "C:/users/data/" does not.

gsutil_delete *Delete images from google cloud*

Description

gsutil_delete takes in the object from images_of and deletes them from the google cloud.

Usage

```
gsutil_delete(images_to_delete = NULL, ncore = 2, all = FALSE)
```

Arguments

images_to_delete	a data.frame with a column titled 'filepath'. the filepath is where on the cloud these photos should be deleted from
ncore	a numeric scalar. The number of cores to use to copy images from google cloud. It defaults to 2.
all	Delete all photos in a given folder? Defaults to FALSE.

gsutil_move *Move images between folders on Google cloud*

Description

gsutil_move requires paths from (where the images are located) and to where the images should go on google cloud.

Usage

```
gsutil_move(from = NULL, to = NULL)
```

Arguments

from	paths of files or folders to move
to	paths of where files should go. If from and to are folders, this will move all files between them. If files, then the file will be moved (which can be used to rename images if needed).

images_of *Images of a species at a given study area*

Description

images_of queries the UWIN database for a selected species and study area.

Usage

```
images_of(species = NULL, studyArea = NULL, db = uwidb)
```

Arguments

species	A character vector of the species to be queried. If left as NULL then you can select the species you want in a pop up list. Multiple species can be selected by holding CTRL when clicking on them.
studyArea	The four letter capitalized study area abbreviation for a city. If left as NULL you can select the study area from a pop up list. Only one study area may be selected at a time.
db	The MariaDB connection to the UWIN database. Defaults to 'uwidb'

Value

a data.frame with the following columns: - filepath: the location of the image on google cloud - locationName: the site name the image was taken - photoName: the name of the image - commonName: the species tagged in the images - numIndividuals: The number of individuals of the tagged species

Examples

```
## Not run:
my_images <- images_of()

## End(Not run)
```

MODIFY *Modify the records in the UWIN database.*

Description

MODIFY can be used, for example, to update or delete records in the UWIN database.

Usage

```
MODIFY(sql = NULL, report = FALSE, db = uwidb)
```

Arguments

sql	A SQL statement input as a character vector to be sent to the UWIN database.
report	Logical. Whether or not to report the number of rows affected. Defaults to FALSE.
db	The MariaDB connection to the UWIN database. Defaults to 'uwidb'

---

progress_of *The progress made for each season's worth of data*

Description

progress_of queries the UWIN database for a selected study area.

Usage

```
progress_of(studyArea = NULL, db = uwidb)
```

Arguments

studyArea	The four letter capitalized study area abbreviation for a city. If left as NULL you can select the study area from a pop up list. Only one study area may be selected at a time.
db	The MariaDB connection to the UWIN database. Defaults to 'uwidb'

Value

a list with the following elements:

- assignedIncomplete (data.frame): This is a report on photo groups that have been assigned but have yet to be finished. The columns in assingedIncomplete are:

  - User: The name of the person who has photo groups to classify.
  - email: The users email.
  - yearMonth: The year and month of visits in the database that have photos.r
  - countAssignedIncomplete: The number of photo groups assigned to a user per yearMonth that have not been completed.

- fullComplete (data.frame): This is a progress report on the images that have been classified as 'complete' in the cloud database, which depends on how many users are necessary to consider an image 'complete' (either one or two, depending on a study area). The columns in fullComplete are:

  - yearMonth: The year and month of visits in the database that have photos.
  - percentComplete: The percent of images in a yearMonth that are considered 100
  - TotalPhotos: The number of photos tied to a yearMonth.
  - NotAssigned: Photos in a yearMonth that have not been assigned to a user to tag.
  - ToValidate: The number of photos that need to go through Validation per yearMonth. The Valdidation is on the UWIN cloud db.

- pendingComplete (data.frame): This is a progress report on the images that have been tagged by at least one user on the cloud db, which could be used to generate an occupancy query (assuming that 1 viewer is okay). The columns in pendingComplete are:

  - yearMonth: The year and month of visits in the database that have photos.
  - percentComplete: The percent of images in a yearMonth that have been tagged at least once.
  - TotalPhotos: The number of photos tied to a yearMonth.
  - NotAssigned: Photos in a yearMonth that have not been assigned to a user to tag.

Examples

```
## Not run:
my_images <- progress_of("CHIL")

## End(Not run)
```

SELECT *Apply a select query to the UWIN database.*

Description

SELECT will send select SQL queries to the UWIN database and return the output.

Usage

```
SELECT(sql = NULL, db = uwidb)
```

Arguments

sql	A SQL statement input as a character vector to be sent to the UWIN database.
db	The MariaDB connection to the UWIN database. Defaults to 'uwidb'

Value

A data.frame with the output from the SQL statement.

Examples

```
## Not run:
my_sql <- 'SELECT * FROM Visits;'
SELECT(my_sql)

## End(Not run)
```

sqlpaste *Wrapper function for paste for prettier sql statements*

Description

sqlpaste will add a new line to the end of every paste statement. If the last object is a sole ";", it will be tacked onto the element before it.

Usage

```
sqlpaste(...)
```

Arguments

...	one or more ‘R‘ objects, to be converted to character vectors.

Value

A nicely formatted SQL statement.

Examples

```
## Not run:
tmp_qry <- sqlpaste(
"select * from tbldf",
"where tbldf.column == 'value'"
)
response <- SELECT(tmp_qry)

## End(Not run)
```

sql_IN *Concatenate SQL IN statement*

Description

sql_IN will create an IN statement from a vector of values.

Usage

```
sql_IN(x, add_quotes = TRUE)
```

Arguments

x	A vector of values to create an IN statement with.
add_quotes	Whether to put single quotes around values.

Value

A sql_IN statement with single quotes around the values

Examples

```
my_in <- c("hello", "world")
sql_IN(my_in)

my_in2 <- c(1, 2, 3)
sql_IN(my_in2, FALSE)
```

Index