Package: etn (via r-universe)

October 28, 2024

Title Access Data from the European Tracking Network

Version 2.2.1

Description Package with functions to access and process data from the European Tracking Network hosted by VLIZ.

License MIT + file LICENSE

URL https://github.com/inbo/etn, https://inbo.github.io/etn

BugReports https://github.com/inbo/etn/issues

Depends R (>= 3.4.0)

Imports assertthat, DBI, dplyr, glue, jsonlite, lubridate, methods, odbc, readr, stringr

Suggests formattable, leaflet, kableExtra, knitr, rmarkdown, testthat (>= 3.0.0), tidyr, frictionless, withr

LazyData true

Encoding UTF-8

VignetteBuilder knitr

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Config/testthat/edition 3

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

RemoteUrl https://github.com/inbo/etn

RemoteRef HEAD

RemoteSha cb0a15c52499ddb5e0486918b71d9e39514ee206

Contents

connect_to_etn 2
download_acoustic_dataset 3
get_acoustic_deployments 5
get_acoustic_detections 6

get_acoustic_projects 8
get_acoustic_receivers 9
get_animals 10
get_animal_projects 11
get_cpod_projects 12
get_tags 12
list_acoustic_project_codes 13
list_acoustic_tag_ids 14
list_animal_ids 14
list_animal_project_codes 15
list_cpod_project_codes 15
list_deployment_ids 16
list_receiver_ids 16
list_scientific_names 17
list_station_names 17
list_tag_serial_numbers 18
list_values 18
write_dwc 19

Index **21**

`connect_to_etn` *Connect to the ETN database*

Description

Connect to the ETN database using username and password.

Usage

```
connect_to_etn(username = Sys.getenv("userid"), password = Sys.getenv("pwd"))
```

Arguments

`username`	Character. Username to use for the connection.
`password`	Character. Password to use for the connection.

Value

ODBC connection to ETN database.

Examples

```
## Not run:
# Connect to the ETN database using your rstudio.lifewatch.be username and
# password, and save as the default connection variable "con"
con <- connect_to_etn()

# Connect to the ETN database using non-default username and password
con <- connect_to_etn(username = "my_username", password = "my_password")

## End(Not run)
```

`download_acoustic_dataset` *Download acoustic data package*

Description

Download all acoustic data related to an **animal project** as a data package that can be deposited in a research data repository. Includes option to filter on scientific names.

Usage

```
download_acoustic_dataset(
  connection = con,
  animal_project_code,
  scientific_name = NULL,
  directory = animal_project_code
)
```

Arguments

`connection`	A connection to the ETN database. Defaults to `con`.
`animal_project_code`	Character. Animal project you want to download data for. Required.
`scientific_name`	Character (vector). One or more scientific names. Defaults to no all (all scientific names, include "Sync tag", etc.).
`directory`	Character. Relative path to local download directory. Defaults to creating a directory named after animal project code. Existing files of the same name will be overwritten.

Details

The data are downloaded as a **Frictionless Data Package** containing:

file	description
`animals.csv`	Animals related to an `animal_project_code`, optionally filtered on `scientific_name`(s), as returned

`tags.csv`	Tags associated with the selected animals, as returned by `get_tags()`.
`detections.csv`	Acoustic detections for the selected animals, as returned by `get_acoustic_detections()`.
`deployments.csv`	Acoustic deployments for the `acoustic_project_code`(s) found in detections, as returned by `get_acc`
`receivers.csv`	Acoustic receivers for the selected deployments, as returned by `get_acoustic_receivers()`.
`datapackage.json`	A Frictionless Table Schema metadata file describing the fields and relations of the above csv files. This

The function will report the number of records per csv file, as well as the included scientific names and acoustic projects. Warnings will be raised for:

- Animals with multiple tags
- Tags associated with multiple animals
- Deployments without acoustic project: these deployments will not be listed in `deployments.csv` and will therefore raise a foreign key validation error.
- Duplicate detections: detections with the duplicate `detection_id`. These are removed by the function in `detections.csv`.

Important: The data are downloaded *as is* from the database, i.e. no quality or consistency checks are performed by this function. We therefore recommend to verify the data before publication. A consistency check can be performed by validation tools of the Frictionless Framework, e.g. `frictionless validate datapackage.json` on the command line using frictionless-py.

Examples

```
## Not run:
# Set default connection variable
con <- connect_to_etn()

# Download data for the 2012_leopoldkanaal animal project (all scientific names)
download_acoustic_dataset(animal_project_code = "2012_leopoldkanaal")
#> Downloading data to directory `2012_leopoldkanaal`:
#> * (1/6): downloading animals.csv
#> * (2/6): downloading tags.csv
#> * (3/6): downloading detections.csv
#> * (4/6): downloading deployments.csv
#> * (5/6): downloading receivers.csv
#> * (6/6): adding datapackage.json as file metadata
#>
#> Summary statistics for dataset `2012_leopoldkanaal`:
#> * number of animals:           104
#> * number of tags:              103
#> * number of detections:        2215243
#> * number of deployments:       1968
#> * number of receivers:         454
#> * first date of detection:     2012-07-04
#> * last date of detection:      2021-09-02
#> * included scientific names:   Anguilla anguilla
#> * included acoustic projects:  albert, Apelafico, bpns, JJ_Belwind, leopold, MOBEIA, pc4c, SPAWNSEIS, ws2, zeesc
#>
#> Warning message:
```

```
#> In download_acoustic_dataset(animal_project_code = "2012_leopoldkanaal") :
#> Found tags associated with multiple animals: 1145373

## End(Not run)
```

`get_acoustic_deployments` *Get acoustic deployment data*

Description

Get data for deployments of acoustic receivers, with options to filter results.

Usage

```
get_acoustic_deployments(
  connection = con,
  deployment_id = NULL,
  receiver_id = NULL,
  acoustic_project_code = NULL,
  station_name = NULL,
  open_only = FALSE
)
```

Arguments

`connection`	A connection to the ETN database. Defaults to `con`.
`deployment_id`	Integer (vector). One or more deployment identifiers.
`receiver_id`	Character (vector). One or more receiver identifiers.
`acoustic_project_code`	Character (vector). One or more acoustic project codes. Case-insensitive.
`station_name`	Character (vector). One or more deployment station names.
`open_only`	Logical. Restrict deployments to those that are currently open (i.e. no end date defined). Defaults to `FALSE`.

Value

A tibble with acoustic deployment data, sorted by `acoustic_project_code`, `station_name` and `deploy_date_time`. See also field definitions.

Examples

```
# Set default connection variable
con <- connect_to_etn()

# Get all acoustic deployments
get_acoustic_deployments(con)

# Get specific acoustic deployment
get_acoustic_deployments(con, deployment_id = 1437)

# Get acoustic deployments for a specific receiver
get_acoustic_deployments(con, receiver_id = "VR2W-124070")

# Get open acoustic deployments for a specific receiver
get_acoustic_deployments(con, receiver_id = "VR2W-124070", open_only = TRUE)

# Get acoustic deployments for a specific acoustic project
get_acoustic_deployments(con, acoustic_project_code = "demer")

# Get acoustic deployments for two specific stations
get_acoustic_deployments(con, station_name = c("de-9", "de-10"))
```

---

get_acoustic_detections

Get acoustic detections data

Description

Get data for acoustic detections, with options to filter results. Use `limit` to limit the number of returned records.

Usage

```
get_acoustic_detections(
  connection = con,
  start_date = NULL,
  end_date = NULL,
  acoustic_tag_id = NULL,
  animal_project_code = NULL,
  scientific_name = NULL,
  acoustic_project_code = NULL,
  receiver_id = NULL,
  station_name = NULL,
  limit = FALSE
)
```

Arguments

connection	A connection to the ETN database. Defaults to con.
start_date	Character. Start date (inclusive) in ISO 8601 format (yyyy-mm-dd, yyyy-mm or yyyy).
end_date	Character. End date (exclusive) in ISO 8601 format (yyyy-mm-dd, yyyy-mm or yyyy).
acoustic_tag_id	Character (vector). One or more acoustic tag ids.
animal_project_code	Character (vector). One or more animal project codes. Case-insensitive.
scientific_name	Character (vector). One or more scientific names.
acoustic_project_code	Character (vector). One or more acoustic project codes. Case-insensitive.
receiver_id	Character (vector). One or more receiver identifiers.
station_name	Character (vector). One or more deployment station names.
limit	Logical. Limit the number of returned records to 100 (useful for testing purposes). Defaults to FALSE.

Value

A tibble with acoustic detections data, sorted by acoustic_tag_id and date_time. See also field definitions.

Examples

```
# Set default connection variable
con <- connect_to_etn()

# Get limited sample of acoustic detections
get_acoustic_detections(con, limit = TRUE)

# Get all acoustic detections from a specific animal project
get_acoustic_detections(con, animal_project_code = "2014_demer")

# Get 2015 acoustic detections from that animal project
get_acoustic_detections(
  con,
  animal_project_code = "2014_demer",
  start_date = "2015",
  end_date = "2016",
)

# Get April 2015 acoustic detections from that animal project
get_acoustic_detections(
  con,
  animal_project_code = "2014_demer",
  start_date = "2015-04",
```

```
  end_date = "2015-05",
)

# Get April 24, 2015 acoustic detections from that animal project
get_acoustic_detections(
  con,
  animal_project_code = "2014_demer",
  start_date = "2015-04-24",
  end_date = "2015-04-25",
)

# Get acoustic detections for a specific tag at two specific stations
get_acoustic_detections(
  con,
  acoustic_tag_id = "A69-1601-16130",
  station_name = c("de-9", "de-10")
)

# Get acoustic detections for a specific species, receiver and acoustic project
get_acoustic_detections(
  con,
  scientific_name = "Rutilus rutilus",
  receiver_id = "VR2W-124070",
  acoustic_project_code = "demer"
)
```

`get_acoustic_projects` *Get acoustic project data*

Description

Get data for acoustic projects, with options to filter results.

Usage

```
get_acoustic_projects(connection = con, acoustic_project_code = NULL)
```

Arguments

`connection`	A connection to the ETN database. Defaults to `con`.
`acoustic_project_code`	Character (vector). One or more acoustic project codes. Case-insensitive.

Value

A tibble with acoustic project data, sorted by `project_code`. See also field definitions.

Examples

```
# Set default connection variable
con <- connect_to_etn()

# Get all acoustic projects
get_acoustic_projects(con)

# Get a specific acoustic project
get_acoustic_projects(con, acoustic_project_code = "demer")
```

`get_acoustic_receivers` *Get acoustic receiver data*

Description

Get data for acoustic receivers, with options to filter results.

Usage

```
get_acoustic_receivers(connection = con, receiver_id = NULL, status = NULL)
```

Arguments

`connection`	A connection to the ETN database. Defaults to `con`.
`receiver_id`	Character (vector). One or more receiver identifiers.
`status`	Character. One or more statuses, e.g. `available` or `broken`.

Value

A tibble with acoustic receiver data, sorted by `receiver_id`. See also field definitions. Values for `owner_organization` will only be visible if you are member of the group.

Examples

```
# Set default connection variable
con <- connect_to_etn()

# Get all acoustic receivers
get_acoustic_receivers(con)

# Get lost and broken acoustic receivers
get_acoustic_receivers(con, status = c("lost", "broken"))

# Get a specific acoustic receiver
get_acoustic_receivers(con, receiver_id = "VR2W-124070")
```

`get_animals` *Get animal data*

Description

Get data for animals, with options to filter results. Associated tag information is available in columns starting with `tag` and `acoustic_tag_id`. If multiple tags are associated with a single animal, the information is comma-separated.

Usage

```
get_animals(
  connection = con,
  animal_id = NULL,
  tag_serial_number = NULL,
  animal_project_code = NULL,
  scientific_name = NULL
)
```

Arguments

`connection`	A connection to the ETN database. Defaults to `con`.
`animal_id`	Integer (vector). One or more animal identifiers.
`tag_serial_number`	Character (vector). One or more tag serial numbers.
`animal_project_code`	Character (vector). One or more animal project codes. Case-insensitive.
`scientific_name`	Character (vector). One or more scientific names.

Value

A tibble with animals data, sorted by `animal_project_code`, `release_date_time` and `tag_serial_number`. See also field definitions.

Examples

```
# Set default connection variable
con <- connect_to_etn()

# Get all animals
get_animals(con)

# Get specific animals
get_animals(con, animal_id = 305) # Or string value "305"
get_animals(con, animal_id = c(304, 305, 2827))
```

```
# Get animals from specific animal project(s)
get_animals(con, animal_project_code = "2014_demer")
get_animals(con, animal_project_code = c("2014_demer", "2015_dijle"))

# Get animals associated with a specific tag_serial_number
get_animals(con, tag_serial_number = "1187450")

# Get animals of specific species (across all projects)
get_animals(con, scientific_name = c("Rutilus rutilus", "Silurus glanis"))

# Get animals of a specific species from a specific project
get_animals(con, animal_project_code = "2014_demer", scientific_name = "Rutilus rutilus")
```

`get_animal_projects` *Get animal project data*

Description

Get data for animal projects, with options to filter results.

Usage

```
get_animal_projects(connection = con, animal_project_code = NULL)
```

Arguments

`connection`	A connection to the ETN database. Defaults to `con`.
`animal_project_code`	Character (vector). One or more animal project codes. Case-insensitive.

Value

A tibble with animal project data, sorted by `project_code`. See also field definitions.

Examples

```
# Set default connection variable
con <- connect_to_etn()

# Get all animal projects
get_animal_projects(con)

# Get a specific animal project
get_animal_projects(con, animal_project_code = "2014_demer")
```

get_cpod_projects *Get cpod project data*

Description

Get data for cpod projects, with options to filter results.

Usage

```
get_cpod_projects(connection = con, cpod_project_code = NULL)
```

Arguments

connection	A connection to the ETN database. Defaults to con.
cpod_project_code	Character (vector). One or more cpod project codes. Case-insensitive.

Value

A tibble with animal project data, sorted by project_code. See also field definitions.

Examples

```
# Set default connection variable
con <- connect_to_etn()

# Get all animal projects
get_cpod_projects(con)

# Get a specific animal project
get_cpod_projects(con, cpod_project_code = "cpod-lifewatch")
```

get_tags *Get tag data*

Description

Get data for tags, with options to filter results. Note that there can be multiple records (acoustic_tag_id) per tag device (tag_serial_number).

Usage

```
get_tags(
  connection = con,
  tag_type = NULL,
  tag_subtype = NULL,
  tag_serial_number = NULL,
  acoustic_tag_id = NULL
)
```

Arguments

connection	A connection to the ETN database. Defaults to con.
tag_type	Character (vector). acoustic or archival. Some tags are both, find those with acoustic-archival.
tag_subtype	Character (vector). animal, built-in, range or sentinel.
tag_serial_number	Character (vector). One or more tag serial numbers.
acoustic_tag_id	Character (vector). One or more acoustic tag identifiers, i.e. identifiers found in get_acoustic_detections().

Value

A tibble with tags data, sorted by tag_serial_number. See also field definitions. Values for owner_organization and owner_pi will only be visible if you are member of the group.

Examples

```
# Set default connection variable
con <- connect_to_etn()

# Get all tags
get_tags(con)

# Get archival tags, including acoustic-archival
get_tags(con, tag_type = c("archival", "acoustic-archival"))

# Get tags of specific subtype
get_tags(con, tag_subtype = c("built-in", "range"))

# Get specific tags (note that these can return multiple records)
get_tags(con, tag_serial_number = "1187450")
get_tags(con, acoustic_tag_id = "A69-1601-16130")
get_tags(con, acoustic_tag_id = c("A69-1601-16129", "A69-1601-16130"))
```

---

list_acoustic_project_codes

List all available acoustic project codes

Description

List all available acoustic project codes

Usage

```
list_acoustic_project_codes(connection = con)
```

Arguments

connection	A connection to the ETN database. Defaults to con.

Value

A vector of all unique project_code of type = "acoustic" in project.sql.

---

list_acoustic_tag_ids *List all available acoustic tag ids*

Description

List all available acoustic tag ids

Usage

```
list_acoustic_tag_ids(connection = con)
```

Arguments

connection	A connection to the ETN database. Defaults to con.

Value

A vector of all unique acoustic_tag_id in acoustic_tag_id.sql.

---

list_animal_ids *List all available animal ids*

Description

List all available animal ids

Usage

```
list_animal_ids(connection = con)
```

Arguments

connection	A connection to the ETN database. Defaults to con.

Value

A vector of all unique id_pk present in common.animal_release.

list_animal_project_codes

List all available animal project codes

Description

List all available animal project codes

Usage

```
list_animal_project_codes(connection = con)
```

Arguments

connection	A connection to the ETN database. Defaults to con.

Value

A vector of all unique project_code of type = "animal" in project.sql.

list_cpod_project_codes

List all available cpod project codes

Description

List all available cpod project codes

Usage

```
list_cpod_project_codes(connection = con)
```

Arguments

connection	A connection to the ETN database. Defaults to con.

Value

A vector of all unique project_code of type = "cpod" in project.sql.

list_deployment_ids *List all available receiver ids*

Description

List all available receiver ids

Usage

```
list_deployment_ids(connection = con)
```

Arguments

connection	A connection to the ETN database. Defaults to con.

Value

A vector of all unique id_pk present in acoustic.deployments.

list_receiver_ids *List all available receiver ids*

Description

List all available receiver ids

Usage

```
list_receiver_ids(connection = con)
```

Arguments

connection	A connection to the ETN database. Defaults to con.

Value

A vector of all unique receiver present in acoustic.receivers.

list_scientific_names *List all available scientific names*

Description

List all available scientific names

Usage

```
list_scientific_names(connection = con)
```

Arguments

connection	A connection to the ETN database. Defaults to con.

Value

A vector of all unique scientific_name present in common.animal_release.

list_station_names *List all available station names*

Description

List all available station names

Usage

```
list_station_names(connection = con)
```

Arguments

connection	A connection to the ETN database. Defaults to con.

Value

A vector of all unique station_name present in acoustic.deployments.

list_tag_serial_numbers *List all available tag serial numbers*

Description

List all available tag serial numbers

Usage

```
list_tag_serial_numbers(connection = con)
```

Arguments

connection	A connection to the ETN database. Defaults to con.

Value

A vector of all unique tag_serial_numbers present in common.tag_device.

list_values *List all unique values from a data.frame column*

Description

Get a vector with all unique values found in a given column of a data.frame. Concatenated values (A,B) in the column can be returned as single values (A and B).

Usage

```
list_values(.data, column, split = ",")
```

Arguments

.data	Data frame. Data.frame to select column from.
column	Character or integer. Quoted or unqoted column name or column position.
split	Character (vector). Character or regular expression(s) passed to strsplit() to split column values before returning unique values. Defaults to ,.

Value

A vector of the same type as the given column.

Examples

```
# Set default connection variable
con <- connect_to_etn()
library(dplyr) # For %>%

# List unique scientific_name from a dataframe containing animal information
df <- get_animals(con, animal_project_code = "2014_demer")
list_values(df, "scientific_name")

# Or using pipe and unquoted column name
df %>% list_values(scientific_name)

# Or using column position
df %>% list_values(8)

# tag_serial_number can contain comma-separated values
df <- get_animals(con, animal_id = 5841)
df$tag_serial_number

# list_values() will split those and return unique values
list_values(df, tag_serial_number)

# Another expression can be defined to split values (here ".")
list_values(df, tag_serial_number, split = "\\.")
```

---

write_dwc *Transform ETN data to Darwin Core*

Description

Transforms and downloads data from a European Tracking Network **animal project** to Darwin Core. The resulting CSV file(s) can be uploaded to an IPT for publication to OBIS and/or GBIF. A `meta.xml` or `eml.xml` file are not created.

Usage

```
write_dwc(
  connection = con,
  animal_project_code,
  directory = ".",
  rights_holder = NULL,
  license = "CC-BY"
)
```

Arguments

`connection`	Connection to the ETN database.
`animal_project_code`	Animal project code.

`directory`	Path to local directory to write file(s) to. If `NULL`, then a list of data frames is returned instead, which can be useful for extending/adapting the Darwin Core mapping before writing with `readr::write_csv()`.
`rights_holder`	Acronym of the organization owning or managing the rights over the data.
`license`	Identifier of the license under which the data will be published. • `CC-BY` (default). • `CC0`.

Value

CSV file(s) written to disk or list of data frames when `directory = NULL`.

Transformation details

Data are transformed into an Occurrence core. This **follows recommendations** discussed and created by Peter Desmet, Jonas Mortelmans, Jonathan Pye, John Wieczorek and others. See the SQL file(s) used by this function for details.

Key features of the Darwin Core transformation:

- Deployments (animal+tag associations) are parent events, with capture, surgery, release, recapture (human observations) and acoustic detections (machine observations) as child events. No information about the parent event is provided other than its ID, meaning that data can be expressed in an Occurrence Core with one row per observation and `parentEventID` shared by all occurrences in a deployment.
- The release event often contains metadata about the animal (sex, lifestage, comments) and deployment as a whole.
- Acoustic detections are downsampled to the **first detection per hour**, to reduce the size of high-frequency data. Duplicate detections (same animal, tag and timestamp) are excluded. It is possible for a deployment to contain no detections, e.g. if the tag malfunctioned right after deployment.

Index

connect_to_etn, 2

download_acoustic_dataset, 3

get_acoustic_deployments, 5
get_acoustic_detections, 6
get_acoustic_detections(), *13*
get_acoustic_projects, 8
get_acoustic_receivers, 9
get_animal_projects, 11
get_animals, 10
get_cpod_projects, 12
get_tags, 12

list_acoustic_project_codes, 13
list_acoustic_tag_ids, 14
list_animal_ids, 14
list_animal_project_codes, 15
list_cpod_project_codes, 15
list_deployment_ids, 16
list_receiver_ids, 16
list_scientific_names, 17
list_station_names, 17
list_tag_serial_numbers, 18
list_values, 18

readr::write_csv(), *20*

strsplit(), *18*

write_dwc, 19