Package: wqbench (via r-universe)

September 17, 2024

Title Calculate Aquatic Life Benchmarks

Version 0.2.0

Description Download recent versions of the US EPA ECOTOX database. Clean, standardize and classify data so values are comparable. Use a SSD or deterministic method to determine the critical toxicity value and assessment factors to compute the aquatic life water quality benchmark for a compound.

License Apache License (== 2)

Imports bcdata, chk, chktemplate, DBI, dplyr, ggplot2, httr, readr, rlang, RSQLite, rvest, scales, ssdtools (>= 1.0.6.9011), stringr, tibble, tidyr, utils, withr

Suggests covr, knitr, openxlsx, rmarkdown, testthat (>= 3.0.0), wqbenchdata

Remotes poissonconsulting/chktemplate, bcgov/wqbenchdata, bcgov/ssdtools

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Depends R (>= 4.1)

LazyData true

VignetteBuilder knitr

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

RemoteUrl https://github.com/poissonconsulting/wqbench

RemoteRef HEAD

RemoteSha b2a5e02e2035d4168acedf50c8a8288ab943ef9e

Contents

`combine_bc_species` *Combine BC Species and DB Species*

Description

Internal to allow for testing

Usage

```
combine_bc_species(bc_species, db_species)
```

Arguments

`bc_species`	A data frame
`db_species`	A data frame

Value

A data frame

Examples

```
## Not run:
data <- combine_bc_species(bc_species, db_species)

## End(Not run)
```

`combine_bc_wqg` *Combine BC Water Quality Guidelines with DB Chemicals*

Description

Internal to allow for testing

Usage

```
combine_bc_wqg(bc_wqg, db_chemicals)
```

Arguments

bc_wqg	A data frame
db_chemicals	A data frame

Value

A data frame

Examples

```
## Not run:
data <- combine_bc_wqg(bc_wqg, db_chemicals)

## End(Not run)
```

---

combine_concentration_endpoints *Combine the Concentration Endpoints list to db endpoints*

Description

Internal to allow for testing

Usage

```
combine_concentration_endpoints(endpoint_concentration_pick, db_endpoint_code)
```

Arguments

endpoint_concentration_pick	A data frame
db_endpoint_code	A data frame

Value

A data frame

Examples

```
## Not run:
data <- combine_concentration_endpoints(
  endpoint_concentration_pick, db_endpoint_code
)

## End(Not run)
```

combine_conc_conversions *Combine Concentration Unit Conversation Values with DB Concentration Unit*

Description

Internal to allow for testing

Usage

```
combine_conc_conversions(concentration_std, db_concentration_unit_codes)
```

Arguments

concentration_std	A data frame
db_concentration_unit_codes	A data frame

Value

A data frame

Examples

```
## Not run:
data <- combine_conc_conversions(
  concentration_std, db_concentration_unit_codes
)

## End(Not run)
```

combine_duration_conversions *Combine Duration Unit Conversation Values and DB duration unit*

Description

Internal to allow for testing

Usage

```
combine_duration_conversions(duration_std, db_duration_unit_codes)
```

Arguments

duration_std A data frame
db_duration_unit_codes
A data frame

Value

A data frame

Examples

```
## Not run:
duration_unit_codes_std <- combine_duration_conversions(
  duration_std, db_duration_unit_codes
)

## End(Not run)
```

---

combine_lifestage *Combine Life Stage Groups and db life stage*

Description

Internal to allow for testing

Usage

```
combine_lifestage(lifestage_codes, db_lifestage_codes)
```

Arguments

lifestage_codes
A data frame
db_lifestage_codes
A data frame

Value

A data frame

Examples

```
## Not run:
lifestage_groups <- combine_lifestage(lifestage_codes, db_lifestage_codes)

## End(Not run)
```

combine_media *Combine Media Type Coding*

Description

Internal to allow for testing

Usage

```
combine_media(db_media_type)
```

Arguments

db_media_type	A data frame

Value

A data frame

Examples

```
## Not run:
media_type <- combine_media(db_media_type)

## End(Not run)
```

combine_trophic_group *Combine Trophic Groups and DB species*

Description

Internal to allow for testing

Usage

```
combine_trophic_group(trophic_groups, db_species)
```

Arguments

trophic_groups	A data frame
db_species	A data frame

Value

A data frame

Examples

```
## Not run:
species_trophic_group <- combine_trophic_group(trophic_groups, db_species)

## End(Not run)
```

`join_data` *Join Data*

Description

Internal to wqb_join_data() to allow for testing.

Usage

```
join_data(
  db_results,
  db_tests,
  db_endpoint_codes,
  db_species,
  db_lifestage_codes,
  db_chemicals,
  db_duration_unit_codes,
  db_concentration_unit_codes,
  db_references,
  db_effect_codes,
  db_media_type_codes,
  db_meta_data_download
)
```

Arguments

`db_results`	A data frame
`db_tests`	A data frame
`db_endpoint_codes`	A data frame
`db_species`	A data frame
`db_lifestage_codes`	A data frame
`db_chemicals`	A data frame
`db_duration_unit_codes`	A data frame
`db_concentration_unit_codes`	A data frame
`db_references`	A data frame

db_effect_codes
: A data frame

db_media_type_codes
: A data frame

db_meta_data_download
: A data frame

Value

Invisible data frame

Examples

```
## Not run:
data <- wqb_join_data(
  db_results, db_tests, db_endpoint_codes, db_species, db_lifestage_codes,
  db_chemicals, db_duration_unit_codes, db_concentration_unit_codes,
  db_references, db_effect_codes, db_media_type_codes, db_meta_data_download
)

## End(Not run)
```

---

read_bc_species *Read in BC Species CSV*

Description

Internal to allow for testing

Usage

```
read_bc_species(bc_species_file_path, db_species)
```

Arguments

bc_species_file_path
: A file path

db_species
: A data frame

Value

A data frame

Examples

```
## Not run:
data <- read_bc_species(bc_species_file_path, db_species)

## End(Not run)
```

read_bc_wqg *Read BC Water Quality Guidelines*

Description

Internal to allow for testing

Usage

```
read_bc_wqg(db_chemicals)
```

Arguments

db_chemicals	A data frame

Value

A data frame

Examples

```
## Not run:
chemicals_bc_wqg <- read_bc_wqg(db_chemicals)

## End(Not run)
```

read_concentration_endpoints *Read the Concentration Endpoints*

Description

Internal to allow for testing

Usage

```
read_concentration_endpoints(conc_endpoints_file_path, db_endpoint_code)
```

Arguments

conc_endpoints_file_path	A file path
db_endpoint_code	A data frame

Value

A data frame

Examples

```
## Not run:
endpoint_concentration <- read_concentration_endpoints(
  conc_endpoints_file_path, db_endpoint_code
)

## End(Not run)
```

`read_conc_conversions` *Read Concentration Unit Conversation Values*

Description

Internal to allow for testing

Usage

```
read_conc_conversions(concentration_std_file_path, db_concentration_unit_codes)
```

Arguments

`concentration_std_file_path`	A file path
`db_concentration_unit_codes`	A data frame

Value

A data frame

Examples

```
## Not run:
concentration_unit_codes_std <- read_conc_conversions(
  concentration_std_file_path, db_concentration_unit_codes
)

## End(Not run)
```

read_duration_conversions *Read Duration Unit Conversation*

Description

Internal to allow for testing

Usage

```
read_duration_conversions(duration_std_file_path, db_duration_unit_codes)
```

Arguments

duration_std_file_path	A file path
db_duration_unit_codes	A data frame

Value

A data frame

Examples

```
## Not run:
duration_unit_codes_std <- read_duration_conversions(
  duration_std_file_path, db_duration_unit_codes
)

## End(Not run)
```

read_lifestage *Read Life Stage Groups*

Description

Internal to allow for testing

Usage

```
read_lifestage(lifestage_file_path, db_lifestage_codes)
```

Arguments

lifestage_file_path
: A data frame

db_lifestage_codes
: A data frame

Value

A data frame

Examples

```
## Not run:
lifestage_groups <- read_lifestage(lifestage_file_path, db_lifestage_codes)

## End(Not run)
```

---

read_trophic_group *Read Trophic Groups and DB species*

Description

Internal to allow for testing

Usage

```
read_trophic_group(trophic_groups_file_path, db_species)
```

Arguments

trophic_groups_file_path
: A file path

db_species
: A data frame

Value

A data frame

Examples

```
## Not run:
species_trophic_group <- read_trophic_group(trophic_groups_file_path, db_species)

## End(Not run)
```

template *Template for Adding your own Data*

Description

Template for Adding your own Data

Usage

```
template
```

Format

A data.frame with columns:

name Row description.

latin_name The latin name of the test species.

endpoint Toxicity endpoint.

effect The effect that was being tested.

lifestage The lifestage the species was during the test.

effect_conc_mg.L Contaminant concentration that corresponds to the endpoint.

effect_conc_std_mg.L The effect concentration standardized to include the acute to chronic ratio to extrapolate acute and/or effect concentrations to chronic and/or no-effect concentrations in mg/L.

trophic_group Trophic group of species.

ecological_group Identification of salmonids and planktonic invertebrates. If neither of these, listed as "other".

species_present_in_bc Species is present in British Columbia if entry = TRUE

wqb_add_bc_species *Add BC Species*

Description

Read in a list of British Columbia species and add a column to the `species` table in the database to indicate if the species is present in British Columbia.

Usage

```
wqb_add_bc_species(database, quiet = FALSE)
```

Arguments

database	A string to the location of the database.
quiet	Turn off message when quiet set to TRUE.

Details

The BC species data is contained in a csv file in the extdata folder of the package. This file can be edited by adding new species or removing species. Do not add new columns, rename columns or rename the file. The file must only contain a single column named latin_name.

The latin_name column must consist of the genus and species separated by a space. The latin_name column in the bc-species.csv file is matched to the latin_name column in the species table of the database. A new column species_present_in_bc is added to the species table that codes each species as TRUE if it matches a value in the bc-species.csv or FALSE if there is no match.

Value

Invisible data frame

Examples

```
## Not run:
bc_species <- wqb_add_bc_species(
  database = "ecotox_ascii_09_15_2022.sqlite"
)

bc_species <- wqb_add_bc_species(
  database = "ecotox_db/ecotox_ascii_09_15_2022.sqlite"
)

## End(Not run)
```

---

wqb_add_bc_wqg *Add BC Water Quality Guidelines*

Description

Read in the British Columbia water quality guidelines (wqg) and add a column to the chemicals tables in the database to indicate if the chemical is present in the British Columbia water quality guidelines.

Usage

```
wqb_add_bc_wqg(database, quiet = FALSE)
```

Arguments

database	A string to the location of the database.
quiet	Turn off message when quiet set to TRUE.

Details

The wqg data is stored in the BC Data Catalogue.

The `CAS_number` column in the bc wqg data is matched to the `cas_number` column in the chemicals table of the database. A new column `present_in_bc_wqg` is added to the chemicals table that codes each chemical as TRUE if the chemical is present in wqg or FALSE if the chemical is not present in wqg.

Value

Invisible data frame

Examples

```
## Not run:
chem_bc_wqg <- wqb_add_bc_wqg(
  database = "ecotox_ascii_09_15_2022.sqlite"
)

chem_bc_wqg <- wqb_add_bc_wqg(
  database = "ecotox_db/ecotox_ascii_09_15_2022.sqlite"
)

## End(Not run)
```

---

wqb_add_concentration_endpoints

Add the Concentration Endpoints

Description

Read in the concentration endpoints that have been selected to keep in the final data set. Add a column to the endpoint_codes table in the database to mark the corresponding endpoints.

Usage

```
wqb_add_concentration_endpoints(database, quiet = FALSE)
```

Arguments

`database`	A string to the location of the database.
`quiet`	Turn off message when quiet set to TRUE.

Details

The list of concentration endpoints is contained in a csv file in the extdata folder of the package. The csv file can be edited by adding or removing rows. To add new rows get the `code` and `description` values from the `endpoint_codes` table in the database and paste them into the csv file.

Do not add new columns, rename columns or rename the file. The file must only contain the `code` and `description` column.

The `code` values in the endpoint-concentration file are matched to the `code` values in the endpoint_code table in the database. A new column `concentration_flag` is added to the endpoint_code table that codes each endpoint as TRUE if the endpoint is present in endpoint-concentration.csv file.

Value

Invisible data frame

Examples

```
## Not run:
wqb_add_concentration_endpoints(
  database = "ecotox_ascii_09_15_2022.sqlite"
)

wqb_add_concentration_endpoints(
  database = "ecotox_db/ecotox_ascii_09_15_2022.sqlite"
)

## End(Not run)
```

---

wqb_add_conc_conversions

Add Concentration Unit Conversation Values

Description

Read in the concentration-conversion file and code units that can be converted and which ones will be removed. Three columns will be added to the concentration_unit_codes table in the database, indicate which units are being kept, the conversion factor, and the units it is being converted to. Currently units are converted to mg/L or equivalent.

Usage

```
wqb_add_conc_conversions(database, quiet = FALSE)
```

Arguments

`database`	A string to the location of the database.
`quiet`	Turn off message when quiet set to TRUE.

Details

The list of units to be converted are contained in a csv file in the extdata folder of the package. The csv file can be edited by adding or removing rows. To add new rows get the code and description values from the concentration_unit_codes table in the database and paste them into the csv file.

Do not add new columns, rename columns or rename the file. The file must only contain the columns: code, description, conc_conversion_flag, conc_conversion_value_multiplier and conc_conversion_unit.

The conc_conversion_flag column indicates which units are being converted and which are being removed because they can not be converted or are not in the aquatic portion of the data.The conc_conversion_value_multiplier column contains the value need to convert the unit into the unit listed in the conc_conversion_unit column.

The code values in the concentration-conversion file are matched to the code values in the concentration_unit_codes table in the database.

Value

Invisible data frame

Examples

```
## Not run:
concentration_unit_code_standardization <- wqb_add_conc_conversions(
  database = "ecotox_ascii_09_15_2022.sqlite"
)

concentration_unit_code_standardization <- wqb_add_conc_conversions(
  database = "ecotox_db/ecotox_ascii_09_15_2022.sqlite"
)

## End(Not run)
```

---

wqb_add_duration_conversions

Add Duration Unit Conversation Values

Description

Read in the duration-conversion file and code unit values that can be converted and which ones will be removed. Two columns will be added to the duration_unit_codes table in the database to indicate which units are being kept and the conversion factor. Currently duration units are converted to hours.

Usage

```
wqb_add_duration_conversions(database, quiet = FALSE)
```

Arguments

database	A string to the location of the database.
quiet	Turn off message when quiet set to TRUE.

Details

The list of units to be converted are contained in a csv file in the extdata folder of the package. The csv file can be edited by adding or removing rows. To add new rows get the code and description values from the duration_unit_codes table in the database and paste them into the csv file.

Do not add new columns, rename columns or rename the file. The file must only contain the columns: code, description, duration_units_to_keep and duration_value_multiplier_to_hours.

The code values in the duration-conversion file are matched to the code values in the duration_unit_codes table in the database.

The duration_units_to_keep column indicates which units are being converted and which are being removed because they can not be converted or are not in the aquatic portion of the data. The duration_value_multiplier_to_hours column contains the value need to convert the unit into hours.

Value

Invisible data frame

Examples

```
## Not run:
duration_unit_code_standardization <- wqb_add_duration_conversions(
  database = "ecotox_ascii_09_15_2022.sqlite"
)

duration_unit_code_standardization <- wqb_add_duration_conversions(
  database = "ecotox_db/ecotox_ascii_09_15_2022.sqlite"
)

## End(Not run)
```

---

wqb_add_lifestage *Add Life Stage Groups*

Description

Read in the life stage simple groups and add a column in the lifestage_code table in the database to mark the corresponding values.

Usage

```
wqb_add_lifestage(database, quiet = FALSE)
```

Arguments

database	A string to the location of the database.
quiet	Turn off message when quiet set to TRUE.

Details

Only life stages related to fish or amphibians have been coded. The purpose of the coding is to be able to simplify the many life stages into three categories: els (early life stage), juveniles and adults. Not all life stages have been coded into these three groups.

The life stage data is contained in a csv file in the extdata folder of the package. The csv file can be edited by adding or removing rows. To add new rows get the code and description values from the lifestage_code table in the dataset and paste them into the csv file and then add the value to the simple_lifestage column.

Do not add new columns, rename columns or rename the file. The file must only contain the code, description_lifestage and simple_lifestage column.

The code values in the lifestage-codes.csv file are matched to the code values in the lifestage_code table in the database. Any codes that match are coded in a new column called simple_lifestage.

Value

Invisible data frame

Examples

```
## Not run:
lifestage_codes <- wqb_add_lifestage(
  database = "ecotox_ascii_09_15_2022.sqlite"
)

lifestage_codes <- wqb_add_lifestage(
  database = "ecotox_db/ecotox_ascii_09_15_2022.sqlite"
)

## End(Not run)
```

---

wqb_add_media *Add Media Type Coding of Salt or Fresh Water*

Description

Code the media type information into three categories: salt water, fresh water and not reported.

Usage

```
wqb_add_media(database, quiet = FALSE)
```

Arguments

database	A string to the location of the database.
quiet	Turn off message when quiet set to TRUE.

Details

Read in the media_type_codes table from the database and code the 24 groups into three categories: salt water, fresh water or not reported. The new categories are added to a column in the table called media_type_group.

Value

Invisible data frame

Examples

```
## Not run:
media_info <- wqb_add_media(
  database = "ecotox_ascii_09_15_2022.sqlite"
)

media_info <- wqb_add_media(
  database = "ecotox_db/ecotox_ascii_09_15_2022.sqlite"
)

## End(Not run)
```

---

wqb_add_trophic_group *Add Trophic Groups to Database*

Description

Read in the trophic and ecological groups and and add a column to the species table in the database that lists the groups for each species.

Usage

```
wqb_add_trophic_group(database, quiet = FALSE)
```

Arguments

database	A string to the location of the database.
quiet	Turn off message when quiet set to TRUE.

Details

The trophic group data is contained in a csv file in the extdata folder of the package. This file can be edited by adding or removing groups and classes. Do not add new columns, rename columns or rename the file.

The trophic groups file must contain the columns: `class`, `order`, `trophic_group`, and `ecological_group`. The `class` and `order` columns are matched to the `class` and `tax_order` columns in the `species` table of the database and then adds the `trophic_group` and `ecological_group` columns to the species table.

Value

Invisible data frame

Examples

```
## Not run:
trophic_group <- wqb_add_trophic_group(
  database = "ecotox_ascii_09_15_2022.sqlite"
)

trophic_group <- wqb_add_trophic_group(
  database = "ecotox_db/ecotox_ascii_09_15_2022.sqlite"
)

## End(Not run)
```

---

wqb_af *Determine the Assessment Factors*

Description

Determines the species variation factor, ecological assessment factor and B.C. species factor.

Usage

```
wqb_af(data)
```

Arguments

`data`	A data frame

Details

This is a wrapper function that calls `wqb_af_bc_species()`, `wqb_af_ecological()` and `wqb_af_variation()`.

Value

A data frame

Examples

```
## Not run:
data <- wqb_af(data)

## End(Not run)
```

`wqb_af_bc_species` *Determine the BC Species Assessment Factor*

Description

Determines an assessment factor based on the number of BC species present.

Usage

```
wqb_af_bc_species(data)
```

Arguments

`data`	A data frame

Details

Check the resource document for how the assessment factor was calculated.

Value

A data frame

Examples

```
## Not run:
data <- wqb_af_bc_species(data)

## End(Not run)
```

wqb_af_ecological *Determine the Ecological Assessment Factors*

Description

Determines the assessment factor based on the presence of certain ecological groups.

Usage

```
wqb_af_ecological(data)
```

Arguments

data	A data frame

Details

Check the resource document for how the assessment factor was calculated.

Value

A data frame

Examples

```
## Not run:
data <- wqb_af_ecological(data)

## End(Not run)
```

wqb_af_variation *Determine the Species Variation Assessment Factor*

Description

Determine the assessment factor based on the number of trophic groups and species.

Usage

```
wqb_af_variation(data)
```

Arguments

data	A data frame

Details

Check the resource document for how the assessment factor was calculated.

Value

A data frame

Examples

```
## Not run:
data <- wqb_af_variation(data)

## End(Not run)
```

`wqb_aggregate` *Aggregate Data for each Species*

Description

Aggregate data to select the most sensitive value for each species. The priority of effect level depends on the benchmark method that will be used.

Usage

```
wqb_aggregate(data)
```

Arguments

`data`	A data frame

Details

Check the resource document for the rules used to aggregate the data down to a single value per species. This is Step 5.

Value

A data frame

Examples

```
## Not run:
data <- wqb_aggregate(data)

## End(Not run)
```

wqb_benchmark_method *Determine Benchmark Method*

Description

Determine if a species sensitivity distribution (SSD) or deterministic method will be used to determine the aquatic life water quality benchmark value for the data. A column called `method` will be added to the data indicating which method will be used.

Usage

```
wqb_benchmark_method(data)
```

Arguments

`data`	A data frame

Details

Check the resource document for the rules used to determine which benchmark method will be used. This is Step 4.

Value

A data frame

Examples

```
## Not run:
data <- wqb_benchmark_method(data)
data <- wqb_benchmark_method(aggregated_data)

## End(Not run)
```

wqb_check_add_data *Check the Uploaded Data*

Description

Checks the uploaded data for the basic requirements to ensure the data matches the downloaded Ecotox data.

Usage

```
wqb_check_add_data(data, template)
```

Arguments

data	A data frame. The data you want to check.
template	A data frame. The format the data should be in, in the chktemplate format.

Details

The values for the endpoint, trophic_group, and ecological_group columns are checked against the data tables used to build the database. To update the allowed values the corresponding csv file needs to be updated.

Value

A data frame

Examples

```
## Not run:
data <- wqb_check_add_data(data, template)

## End(Not run)
```

---

wqb_classify_duration *Classify Duration*

Description

Classify each test as either acute or chronic in the duration_class column.

Usage

```
wqb_classify_duration(data, quiet = FALSE)
```

Arguments

data	A data frame
quiet	Turn off message when quiet set to TRUE.

Details

Check the resource document for the rules used to determine the classification for each trophic group. This is Step 2.

Value

A data frame

Examples

```
## Not run:
classified_data <- wqb_classify_duration(compiled_data)

## End(Not run)
```

`wqb_clean_data` *Compile Data*

Description

Join database tables together and start filtering and cleaning data.

Usage

```
wqb_clean_data(data, quiet = FALSE)
```

Arguments

`data`	A data frame.
`quiet`	Turn off message when quiet set to TRUE.

Details

Check the resource document more details on the data added, filter conditions and cleaning steps. This is part of Step 1.

Value

Invisible data frame

Examples

```
## Not run:
data <- wqb_clean_data(data)

## End(Not run)
```

wqb_create_data_set *Create the Data Set for the App*

Description

This function downloads the data, creates and adds all the data to the database, compiles, standardizes, and classifies the data. The output of this function is fed into the app.

Usage

```
wqb_create_data_set(
  file_path = "~/Ecotoxicology/ecotox",
  version = 1,
  folder_path = "~/Ecotoxicology/ecotox_db/",
  quiet = FALSE,
  ask = TRUE,
  save_rds = TRUE
)
```

Arguments

file_path	A string of the file path location to save the downloaded files. The default is your current working directory.
version	An integer to indicate which version you want to download. The default is 1 which downloads the most recent version.
folder_path	Folder path to write to.
quiet	Turn off message when quiet set to TRUE.
ask	Turn off question when set to FALSE.
save_rds	Saves the data set as a rds file in the folder_path. Turn off when set to FALSE.

Value

A data frame

Examples

```
## Not run:
wqb_create_data_set(
  file_path = "~/Ecotoxicology/ecotox",
  version = 1,
  folder_path = "~/Ecotoxicology/ecotox_db/"
)

## End(Not run)
```

wqb_create_epa_ecotox *Create the ECOTOX Database*

Description

Create a SQLite database from the US EPA ECOTOX downloaded files.

Usage

```
wqb_create_epa_ecotox(folder_path = ".", data_path, quiet = FALSE, ask = TRUE)
```

Arguments

folder_path	Folder path to write to.
data_path	Folder path to the downloaded ECOTOX folder
quiet	Turn off message when quiet set to TRUE.
ask	Turn off question when set to FALSE.

Details

This functions reads in the text files in the folder and writes them to the database.

This function will overwrite a database if already present.

Value

Invisible string of the file path of the database.

Examples

```
## Not run:
wqb_create_epa_ecotox(data_path = "ecotox_ascii_12_15_2022")

## End(Not run)
```

wqb_download_epa_ecotox *Download ECOTOX Data Files*

Description

Download the ECOTOX data files from their FTP website. The default is to download the most recent version of the data.

Usage

```
wqb_download_epa_ecotox(
  file_path = ".",
  version = 1,
  ask = TRUE,
  quiet = FALSE
)
```

Arguments

file_path	A string of the file path location to save the downloaded files. The default is your current working directory.
version	An integer to indicate which version you want to download. The default is 1 which downloads the most recent version.
ask	Turn off question when set to FALSE.
quiet	Turn off message when quiet set to TRUE.

Details

You have the option of downloading older version of the data but only up to the four most recent version. The most recent version is set as 1 and the oldest version is version 4.

The downloaded folder will contain various files that are needed to build the database.

You must have a working internet connection to run this function successfully.

Value

Invisible string of the file path the downloaded files were saved.

References

US EPA ECOTOX website: https://cfpub.epa.gov/ecotox/

Olker, J. H., Elonen, C. M., Pilli, A., Anderson, A., Kinziger, B., Erickson, S., Skopinski, M., Pomplun, A., LaLone, C. A., Russom, C. L., & Hoff, D. (2022). The ECOTOXicology Knowledgebase: A Curated Database of Ecologically Relevant Toxicity Tests to Support Environmental Research and Risk Assessment. Environmental Toxicology and Chemistry, 41(6):1520-1539. https://doi.org/10.1002/etc.5324

Examples

```
## Not run:
wqb_download_epa_ecotox()

wqb_download_epa_ecotox("data_download")

# pull previous version of the database
wqb_download_epa_ecotox("data_download", version = 2)

## End(Not run)
```

wqb_filter_chemical *Filter to a Single Chemical*

Description

Filter to a Single Chemical

Usage

```
wqb_filter_chemical(data, cas_num)
```

Arguments

data	A data frame
cas_num	A string of the cas number.

Value

A data frame

Examples

```
## Not run:
data <- wqb_filter_chemical(data, "129909906")
data <- wqb_filter_chemical(data, "1000984359")

## End(Not run)
```

wqb_generate_ctv *Generate Critical Toxicity Value*

Description

Determine the critical toxicity value for the species. The function will apply the method listed in the method column.

Usage

```
wqb_generate_ctv(data, dists = ssdtools::ssd_dists_bcanz())
```

Arguments

data	A data frame
dists	A character vector of the distributions to fit.

Details

The ctv_est_mg.L is the estimate of the critical toxicity value for the species. The ctv_lcl_mg.L is the lower confidence limit. The ctv_ucl_mg.L is the upper confidence limit.

The Deterministic method will always produce missing values in the ctv_lcl_mg.L and ctv_ucl_mg.L columns.

Value

A data frame

Examples

```
## Not run:
ctv <- wqb_generate_ctv(data)

## End(Not run)
```

---

wqb_join_data *Join Data*

Description

Join database tables together and start filtering and cleaning data.

Usage

```
wqb_join_data(database, quiet = FALSE)
```

Arguments

database	A string to the location of the database.
quiet	Turn off message when quiet set to TRUE.

Details

Check the resource document more details on the data added, filter conditions and cleaning steps. This is part of Step 1.

Value

Invisible data frame

Examples

```
## Not run:
data_compiled <- wqb_join_data(
  database = "ecotox_ascii_09_15_2022.sqlite"
)

data_compiled <- wqb_join_data(
  database = "ecotox_db/ecotox_ascii_09_15_2022.sqlite"
)

## End(Not run)
```

`wqb_method_det` *Deterministic Method to Generate Critical Toxicity Value for the Chemical*

Description

Use the deterministic method to generate the critical toxicity value for the data set.

Usage

```
wqb_method_det(data)
```

Arguments

`data`	A data frame

Details

Check the resource document for more details . This is Step 1.

Value

A data frame

Examples

```
## Not run:
bench_iodine <- wqb_method_det(data)

## End(Not run)
```

wqb_method_ssd *SSD Method to Calculate Critical Toxicity Value for the Chemical*

Description

Use a species sensitivity distribution to calculate the critical toxicity value for the data set. The critical toxicity value is the hazardous concentration for 5% of species (HC5) when the SSD method is used.

Usage

```
wqb_method_ssd(data, fit, nboot = 1000)
```

Arguments

data	A data frame
fit	The fit
nboot	The number of bootstrap samples. Default value of 1000.

Details

A wrapper on the ssdtools package function `ssdtools::ssd_hc_bcanz()` that only returns the HC5 concentration.

Value

A data frame

Examples

```
## Not run:
hc5 <- wqb_method_ssd(data, fit)

## End(Not run)
```

wqb_plot *Plot Data*

Description

Plot the standardized data to see the concentrations per species for each endpoint.

Usage

```
wqb_plot(data)
```

Arguments

data	A data frame

Examples

```
## Not run:
wqb_plot(data)

## End(Not run)
```

---

wqb_plot_det *Plot the Deterministic Method Results*

Description

Plot deterministic results to see the standardized data, critical toxicity value and aquatic life benchmark.

Usage

```
wqb_plot_det(data)
```

Arguments

data	A data frame

Examples

```
## Not run:
wqb_plot_det(data)

## End(Not run)
```

---

wqb_plot_ssd *Plot Species Sensitivity Distribution*

Description

Plot results to see the species sensitivity distribution. The dashed line shows the HC5 value.

Usage

```
wqb_plot_ssd(data, fit)
```

Arguments

data	A data frame
fit	The fit from ssd

Details

This is a wrapper on `ssdtools::predict()` and `ssdtools::ssd_plot()`.

Examples

```
## Not run:
wqb_plot_ssd(data, fit)

## End(Not run)
```

---

wqb_ssd_fit *Fit BCANZ Distributions*

Description

Wrapper to `ssdtools::ssd_fit_bcanz()`. The sp_aggre_conc_mg.L values are the concentrations used.

Usage

```
wqb_ssd_fit(data, dists = ssdtools::ssd_dists_bcanz())
```

Arguments

data	A data frame
dists	A character vector of the distributions to fit

Value

A data frame

Examples

```
## Not run:
fit <- wqb_ssd_fit(data)

## End(Not run)
```

wqb_ssd_hc5 *Run SSD to get Hazard Concentrations*

Description

Wrapper to the `ssdtools::ssd_hc_bcanz()` function and selects only the row which is 5%.

Usage

```
wqb_ssd_hc5(fit, nboot = 1000)
```

Arguments

`fit`	The fit from ssd
`nboot`	A count of the number of bootstrap samples to use to estimate the SE and confidence limits. Default value of 1000.

Details

The number of bootstrap samples is set to 1000 so the estimates are quick to generate. Check out `ssdtools::ssd_hc_bcanz()` for more details.

Value

A data frame

Examples

```
## Not run:
hc5 <- wqb_ssd_hc5(fit)

## End(Not run)
```

wqb_standardize_effect *Standardize type of Effect*

Description

Determine and apply factor needed to standardize the endpoints.

Usage

```
wqb_standardize_effect(data, quiet = FALSE)
```

Arguments

data	A data frame
quiet	Turn off message when quiet set to TRUE.

Details

Check the resource document for details the rules used to determine standardized the endpoints. This is Step 3.

Value

A data frame

Examples

```
## Not run:
standardized_effect_data <- wqb_standardize_effect(data)
standardized_effect_data <- wqb_standardize_effect(classified_data)

## End(Not run)
```

---

wqb_summary_af *Summary Table of the Assessment Factor Values*

Description

Create summary table of the species variation factor, ecological assessment factor and B.C. species factor.

Usage

```
wqb_summary_af(data)
```

Arguments

data	A data frame

Value

A data frame

Examples

```
## Not run:
summary_af <- wqb_summary_af(data)

## End(Not run)
```

wqb_summary_trophic_groups

Summary Table of Assessment Factors Information

Description

Summary Table of Assessment Factors Information

Usage

```
wqb_summary_trophic_groups(data)
```

Arguments

data	A data frame

Value

A data frame

Examples

```
## Not run:
summary_af <- wqb_summary_trophic_groups(data)

## End(Not run)
```

wqb_summary_trophic_species

Summary Table of Number of Species per Trophic Group

Description

Summary Table of Number of Species per Trophic Group

Usage

```
wqb_summary_trophic_species(data)
```

Arguments

data	A data frame

Value

A data frame.

Examples

```
## Not run:
summary_trophic <- wqb_summary_trophic_species(data)

## End(Not run)
```

Index