ToxValDB DCAP Prep

July 2, 2025

Type Package

Title Queries data from the ToxValDB database and prepares it for use in DCAP.

Version 2.0.1

Author Taylor Wall

Maintainer Taylor Wall <wall.taylor@epa.gov>

Description This package queries data from the ToxValDB database, filters to records that meet inclusion criteria, and prepares required fields for DCAP. To run the entire process, uses the function run_toxvaldb_dcap_prep(), which runs all underlying processes sequentially. All input and output files are stored and in the `data/` folder.

License BSD3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Imports DBI (>= 1.2.3),
digest (>= 0.6.37),
dplyr (>= 1.1.4),
jsonlite (>= 2.0.0),
magrittr (>= 2.0.3),
openxlsx (>= 4.2.8),
purrr (>= 1.0.4),
readr (>= 2.1.5),
readxl (>= 1.4.5),
renv (>= 1.1.4),
RMySQL (>= 0.11.0),
RSQLite (>= 2.3.9),
stats (>= 4.4.2),
stringi (>= 1.8.7),
stringr (>= 1.5.1),
tidyr (>= 1.3.1),
tidyselect (>= 1.2.1),
utils (>= 4.4.2),
writexl (>= 1.5.2)

Contents

`check_dcap_critical_effect_category_mappings` *check_dcap_critical_effect_category_mappings*

Description

Attempts to remap previously-mapped toxicological_effect_category values to DCAP entries.

Usage

```
check_dcap_critical_effect_category_mappings(
  toxval.db,
  get_suggestions = TRUE,
  input_file = "",
  output_dir = ""
)
```

Arguments

`toxval.db`	The version of ToxVal to use.
`get_suggestions`	Whether to provide mapping suggestions (Default: TRUE).
`input_file`	The file to pull missing toxicological_effect_category from (typically the filtered POD output file).
`output_dir`	The directory to write the output file.

Details

The output Excel files are as follows: - dcap_mappings_identified.xlsx: All toxicological_effect_categories that could be confidently remapped. - dcap_mappings_still_missing.xlsx: Full data for entries missing categorizations. - dcap_missing_categorization.xlsx: Just toxicological_effect, study_type values missing categorizations. - dcap_mapping_suggestions.xlsx: Mapping suggestions based on close, but not exact, matches.

Value

None; output is written to Excel files for additional review.

See Also

rename, pull, mutate, select, distinct, mutate-joins, case_when, filter, group_by read_excel separate_rows str_trim write_xlsx

---

`convert.fields.to.json`

convert.fields.to.json

Description

Generate a new "record" field as a JSON key-value dictionary of row field values.

Usage

```
convert.fields.to.json(in_dat)
```

Arguments

`in_dat`	Dataframe to use to generate the new "record" JSON field.

Value

Modified dataframe with new "record" field in JSON format.

See Also

summarise, select, bind toJSON, fromJSON

convert_get_conversion_factor *convert_get_conversion_factor*

Description

Function to provide equations needed to convert input values to desired units.

Usage

```
convert_get_conversion_factor(conv_factor = 1)
```

Arguments

conv_factor	Conversion factor to use (such as Molecular weight, tissue Density, etc.)

Value

List of conversion factors by input units and desired units.

convert_units *convert_units*

Description

Function to convert input values to desired units.

Usage

```
convert_units(x, units, desired, conv_factor = NA)
```

Arguments

x	Numeric to be converted.
units	Original units to be converted.
desired	Desired units to convert the input value into.
conv_factor	Conversion factor to use (such as Molecular weight, tissue Density, etc.).
overwrite_units	Boolean to overwrite the 'units' with desired units.

Value

Modified input "x" parameter converted to desired units.

Examples

```
## Not run:
if(interactive()){
 # Convert input "24" days into hours - expected result of "576" hours
 convert_units(x=24, units = "days", desired = "hours")
 }

## End(Not run)
```

`export.for.dcap` *export.for.dcap*

Description

Export records required for DCAP.

Usage

```
export.for.dcap(
  toxval.db,
  include.pesticides = FALSE,
  include.drugs = FALSE,
  include.epa_dws = TRUE,
  include.food_add = FALSE,
  run_name = Sys.Date()
)
```

Arguments

`toxval.db`	Database version.
`include.pesticides`	Flag to include pesticides in output or not, default FALSE.
`include.drugs`	Flag to include drugs in output or not, default FALSE.
`include.epa_dws`	Flag to include EPA DWS in output or not, default TRUE.
`include.food_add`	Flag to include food additives in output or not, default FALSE.
`run_name`	The desired name for the output directory, default current date.

Details

Exports all of the data required for the DCAP calculations.

Value

Write a file with the results: ToxValDB for DCAP toxval.db Sys.Date().xlsx

See Also

createStyle, write.xlsx

Examples

```
## Not run:
if(interactive()){
 export.for.dcap(toxval.db = "res_toxval_v96_1")
 }

## End(Not run)
```

---

filter.pods *filter.pods*

Description

Filter values for DCAP according to specified POD rules.

Usage

```
## S3 method for class 'pods'
filter(toxval.db, run_name = Sys.Date())
```

Arguments

toxval.db	Database name.
run_name	The desired name for the output directory, default current date.

Value

None; Filtered results are recorded in Excel file.

See Also

read.xlsx, createStyle, write.xlsx

Examples

```
## Not run:
if(interactive()){
 filter.pods(toxval.db = "res_toxval_v96_1")
 }

## End(Not run)
```

`fix.replace.unicode` *fix.replace.unicode*

Description

A function to check all character fields and handle unicode symbols, either by removing them or replacing them with alphabetic equivalents.

Usage

```
fix.replace.unicode(df)
```

Arguments

`df`	Character vector to check/replace unicode symbols.

Value

Returns a modified version of the input vector with unicode replacements.

See Also

[stri_escape_unicode][stringi::stri_escape_unicode] [str_extract][stringr::str_extract]

`get.conceptual_model.by.toxicological_effect_category` *get.conceptual_model.by.toxicological_effect_category*

Description

Get the conceptual model based on toxicological_effect_category.

Usage

```
get.conceptual_model.by.toxicological_effect_category(df, run_name)
```

Arguments

`df`	Input dataframe of study_type and toxicological_effect data.
`run_name`	The desired name for the output directory, default current date.

Value

DataFrame map of models by toxicological_effect and study_type.

See Also

select, distinct, mutate, context, case_when, mutate-joins, rename, across, reexports, group_by, na_if separate_rows, replace_na str_trim read_delim, cols

---

`get_chemical_index` *get_chemical_index*

Description

Generate a chemical index based on ToxVal DTXSID list and QSAR Ready Smiles.

Usage

```
get_chemical_index(input_toxval_file, input_qsar_file)
```

Arguments

`input_toxval_file`
: Filepath to input ToxVal file with DTXSID values.

`input_qsar_file`
: Filepath to input QSAR Ready Smiles file with DTXSID values.

Value

Combined chemical index dataframe. A CSV file is also written to the input subfolder.

See Also

read_delim, cols, write_delim select, mutate-joins, distinct, c("rowwise", "rowwise"), mutate, group_by map digest

---

`global_vars` *global_vars*

Description

Global variable list to use acros scripts.

Usage

```
global_vars()
```

Value

Named list of global variables.

Examples

```
## Not run:
if(interactive()){
 # Get vector of DCAP sources
 global_vars()$dcap_sources
 }

## End(Not run)
```

`init.current.run.directory` *init.current.run.directory*

Description

Initialize output directory for current run.

Usage

```
init.current.run.directory(run_name = Sys.Date())
```

Arguments

`run_name`	The desired name for the output directory, default current date.

Value

None. new folder directory is created.

`printCurrentFunction` *printCurrentFunction*

Description

Print the name of the current function to assist with logging and debugging.

Usage

```
printCurrentFunction(comment.string = NA)
```

Arguments

`comment.string`	An optional string to be printed

Value

None. Console output generated.

See Also

flush.console

---

runQuery *runQuery*

Description

Runs a database query and returns a result set.

Usage

```
runQuery(query = NULL, db, do.halt = TRUE, verbose = FALSE)
```

Arguments

query	A properly formatted SQL query as a string.
db	The name of the database.
do.halt	If TRUE, halt on errors or warnings, default TRUE.
verbose	If TRUE, print diagnostic information, default FALSE.

Value

Dataframe of query results.

See Also

character(0), MySQLDriver-class flush.console

Examples

```
## Not run:
if(interactive()){
 runQuery(query = "SELECT * FROM toxval LIMIT 1", db = "res_toxval_v96_1")
 }

## End(Not run)
```

run_toxvaldb_dcap_prep *run_toxvaldb_dcap_prep*

Description

Run all functions to pull records from ToxValDB and prep for DCAP analysis.

Usage

```
run_toxvaldb_dcap_prep(
  toxval.db,
  run_name = Sys.Date(),
  run.export = TRUE,
  include.pesticides = FALSE,
  include.drugs = FALSE,
  include.epa_dws = TRUE,
  include.food_add = FALSE
)
```

Arguments

toxval.db	Database name.
run_name	The desired name for the output directory, default current date.
run.export	Whether to run the export.for.dcap function, default TRUE.
include.pesticides	Flag to include pesticides in output or not
include.drugs	Flag to include drugs in output or not
include.epa_dws	Flag to include EPA DWS in output or not
include.food_add	Flag to include food additives in output or not

Value

None. Functions are run in sequence to generate output files.

Examples

```
## Not run:
if(interactive()){
 run_toxvaldb_dcap_prep(toxval.db = "res_toxval_v96_1")
 }

## End(Not run)
```

`source_hash_vectorized` *source_hash_vectorized*

Description

Generate the hash key for an input table based on hashing columns. The hash key helps identify duplicate records in a dataframe.

Usage

```
source_hash_vectorized(res, hashing_cols)
```

Arguments

`res`	The data frame to receive a hash key.
`hashing_cols`	Optional list of columns to use for generating hash key field.

Value

Modified input dataframe with new source_hash field.

See Also

`digest` `distinct`

`split.species.list` *Split species list*

Description

Split species lists into multiple rows.

Usage

```
## S3 method for class 'species.list'
split(df)
```

Arguments

`df`	Input dataframe with species lists to split.

Value

Modified input "df" with split species lists into multiple rows.

See Also

mutate, case_when, filter, bind_rows, group_by, summarise separate_rows str_trim, case

---

toxval.record.dedup *toxval.record.dedup*

Description

Perform deduplication on data based on select identifier fields. Non-identifier fields will be collapsed based on the input delimiter.

Usage

```
toxval.record.dedup(
  res,
  dedup_fields = NULL,
  hashing_cols = NULL,
  delim = " |::| "
)
```

Arguments

res	Dataframe to process and collapse duplicate record fields.
dedup_fields	vector containing field names to deduplicate, default NULL (all fields but hashing cols).
hashing_cols	vector containing field names of hashing columns, default NULL.
delim	String used to separate collapsed values, default ' |::| '.

Value

Dataframe containing deduplicated data, with duplicate records collapsed by delimiter in non-identifier fields.

See Also

select, group_by, summarise, context, filter, mutate, across, reexports, na_if, distinct

`%>%` *Pipe operator*

Description

See `magrittr::%>%` for details.

Usage

```
lhs %>% rhs
```

Arguments

`lhs`	A value or the magrittr placeholder.
`rhs`	A function call using the magrittr semantics.

Value

The result of calling ‘rhs(lhs)‘.

Index