Package 'wisdotcrashdatabase'

February 24, 2021

Title A package used for internal WisDOT crash database pulls and analysis

Version 0.0.0.9000

Description The goal of wisdotcrashdatabase is to make data pulls and data analysis on the crash database
much easier in an R environment. Importing data is done by one function and can output old and new db
into a single dataframe. There are functions to easily recall crash flags, to bin variables for analysis.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

URL https://github.com/jacciz/wisdotcrashdatabase

BugReports https://github.com/jacciz/wisdotcrashdatabase/issues

Suggests testthat (¿= 3.0.0)

Config/testthat/edition 3

Imports data.table,
dplyr,
expss,
fst,
magrittr,
stats,
stringr,
tidyr,
utils

Depends R (¿= 2.10)

R topics documented:

`county_rename` *Get full county name*

Description

This looks at the county code and returns a new column *countyname* of the full county name.

Usage

```
county_rename(any_df, county_col = "CNTYCODE")
```

Arguments

`any_df`	person, crash, or vehicle dataframe
`county_col`	column that has the county code

Value

A new column called *countyname*

Examples

```
## Not run: county_rename(vehicle17)
```

`get_age_groups` *Bin age groups by 5 or 10 years (old and new db)*

Description

This bins ages into age groups by 5 years or 10 years. If bin_by = "old_db_yr", this will allow for old and new db age groups to be combined. This just matches with the old db AGE.

Usage

```
get_age_groups(person_df, bin_by = "5_yr")
```

Arguments

`person_df`	person dataframe
`bin_by`	select either "5_yr", "10_yr", or "old_db_yr"

Value

A new column called *age_group_5yr*, *age_group_10yr* or *age_groups_both*

Examples

```
## Not run: get_age_groups(person17, bin_by = "10_yr"
```

`get_alc_drug_impaired_person` *Get flag for a suspected alcohol or drug person (old and new db)*

Description

This looks to see if a person was suspected of alcohol or drug use.

Usage

```
get_alc_drug_impaired_person(
  person_df,
  driver_only = "N",
  include_alc = "Y",
  include_drug = "Y"
)
```

Arguments

`person_df`	person dataframe
`driver_only`	Select for role as driver only ("Y" or"N")
`include_alc`	Select to include suspected alcohol ("Y" or"N")
`include_drug`	Select to include suspected drug ("Y" or"N")

Value

A new column called *drug_flag* or *alcohol_flag*. Values are "Y","N", and "U" for unknowm. If *driver_only* = "Y", then only drivers will return.

Examples

```
## Not run: get_alc_drug_impaired_person(person17, include_alc = "N")
```

get_crash_times *Get crash hour (new db)*

Description

Adds a new column that gives crash hour.

Usage

```
get_crash_times(dataframe)
```

Arguments

dataframe	dataframe

Value

A new column called *newtime* with crash hour. i.e. "12am"

Examples

```
## Not run: get_crash_times(crash17)
```

get_deerflag_crashes *Get flag for deer crashes (old and new db)*

Description

This finds if a crash involved a deer. Need **CRSHTYPE** and **ANMLTY**.

Usage

```
get_deerflag_crashes(crash_df)
```

Arguments

crash_df	crash dataframe

Value

A new column called *deer_flag* ("Y" or "N")

Examples

```
## Not run: get_deerflag_crashes(crash17)
```

get_driver_flags *Get driver flags - speed, distracted, teen, older*

Description

This adds a column(s) of certain crash flags. Driver flags are: distracted, speed, teen, and older. Speed can be for both old and new db. Rest are for new db only. Must have **DRVRPC** and **STATNM** for speed. **DISTACT** and **DRVRDS** for distracted. **AGE** for teen and older.

Usage

```
get_driver_flags(person_df, flags)
```

Arguments

person_df	person dataframe
flags	select either/all ("distracted","speed","teen","older")

Value

Returns only drivers. Adds a column of selected flag(s) with *speed_flag*, *distracted_flag*, *teendriver_flag*, and *olderdriver_flag*. ("Y" or "N")

Examples

```
## Not run: get_driver_flags(person_df, flags = c("teen", "distracted"))
```

get_motorcycle_persons *Get motorcyclists (old and new db)*

Description

This finds if a person was a motorcyclist (driver or passenger).

Usage

```
get_motorcycle_persons(person_df, vehicle_df)
```

Arguments

person_df	person dataframe
vehicle_df	vehicle dataframe

Value

Only motorcyclists in a crash

Examples

```
## Not run: get_motorcycle_persons(person17, vehicle17)
```

get_seatbelt_flag_by_unit *Get seatbelt flag (new db)*

Description

Finds if a person in a unit was not wearing a seatbelt. For example, a passenger not wearing a seatbelt, every person in that unit would get a seat belt flag. This includes the drivers and other passengers, if any. Need **SFTYEQP**, **EYEPROT** and **HLMTUSE**.

Usage

```
get_seatbelt_flag_by_unit(person_df)
```

Arguments

person_df	person dataframe

Value

A new column *seatbelt_flag_unit* ("Y" or "N")

Examples

```
## Not run: get_seatbelt_flag_by_unit(person17)
```

import_db_data *Import crash, vehicle, person from crash database*

Description

This imports all data based on crash db type, years selected, and columns selected. It combines old and new crash data into a single dataframe. It renames columns of the old db to match db and renames some variables, such as CRSHSVR, to match new db. Note: if an old db is imported, all columns will be automatically selected.

Usage

```
import_db_data(filepath, db_type, years_old = c(), years = c(), columns = c())
```

Arguments

filepath	path where CSVs are stored (must all be in this folder)
db_type	Type of database - any one of "crash", "vehicle", or "person"
years_old	Year(s) of old db data c("16"). Must be "16" or lower.
years	Year(s) of new db data c("20", "21"). Must be "17" or higher.
columns	Columns to be imported. For the new db these columns will always be imported (if applicable): "CRSHNMBR", "CRSHDATE", "CNTYCODE ,"CRSHSVR", "UNITNMBR", "ROLE","VEHTYPE","WISINJ". Columns with multiples, like DRVRPC and ANMLTY, only the first part without the number should be inputted. For old db, all columns will be imported.

Value

dataframe of either crash, vehicle or person

Examples

```
import_db_data(filepath = "C:/CSV/csv_from_sas/fst/", db_type = "crash",
  years_old = c("15", "16"), years = c("17","18"),  columns = c("DRVRPC"))
## Not run: import_db_data(csv_path, "person", years = "20")
```

---

import_narrative *Import crash narratives*

Description

Import crash narratives.

Usage

```
import_narrative(filepath, years)
```

Arguments

filepath	path where CSVs are stored (must all be in this folder)
years	Year(s) of new db data c("20", "21"). Must be "17" or higher.

Value

dataframe of crash narratives

Examples

```
import_narrative(filepath = "C:/CSV/csv_from_sas/from_sas_csv/", years = c("17","18"))
```

---

relabel_person_variables

Relabels WISINJ and ROLE in person

Description

This bins certain variables by recategorizing and making a new column. This is useful when working with data from an old and new database or when wanting to have fewer categories. "wisinj" bins *WISINJ* into "No Injury", "Injured", and "Killed". "bikeped" bins *ROLE* of bicyclists and pedestrians.

Usage

```
relabel_person_variables(person_df, relabel_by = "wisinj")
```

Arguments

person_df	person dataframe
relabel_by	either by "wisinj" or "bikeped"

Value

A new column of either/all *inj* or *bike_ped_role*

Examples

```
## Not run: system.file("extdata", "17person.fst", package = "fst") %>%
  relabel_person_date(relabel_by = "bikeped")
## End(Not run)
```

Index