Package: chmsflow (via r-universe)

June 9, 2026

Type Package

Title Transforming and Harmonizing CHMS Variables

Version 0.1.0

Description Harmonizes variables from the Canadian Health Measures Survey (CHMS) across cycles 1-6 (2007-2019), producing consistent, analysis-ready variables for use with CHMS data. Recoding is data-driven through metadata tables and applied with recodeflow::rec_with_table() from the 'recodeflow' package. The recoding approach builds on sjmisc::rec() from the 'sjmisc' package (Ludecke 2018) <doi:10.21105/joss.00754>.

Depends R (>= 4.1.0)

Imports dplyr, haven, purrr, recodeflow

License MIT + file LICENSE

URL https://github.com/Big-Life-Lab/chmsflow, https://big-life-lab.github.io/chmsflow/

BugReports https://github.com/Big-Life-Lab/chmsflow/issues

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Suggests DT, kableExtra, knitr, quarto, readr, testthat (>= 3.0.0)

VignetteBuilder quarto

Config/build/clean-inst-doc FALSE

LazyData true

Repository https://big-life-lab.r-universe.dev

Date/Publication 2026-06-09 15:30:08 UTC

RemoteUrl https://github.com/big-life-lab/chmsflow

RemoteRef HEAD

RemoteSha 55550b80885b223e70ee618b02731f5a98b3d548

Contents

`adjust_dbp` *Adjusted diastolic blood pressure*

Description

This function adjusts diastolic blood pressure based on the respondent's diastolic average blood pressure across six measurements. The adjustment is made using specific correction factors. The adjusted diastolic blood pressure is returned as a numeric value.

Usage

```
adjust_dbp(bpmdpbpd)
```

Arguments

bpmdpbpd	numeric A numeric representing the respondent's diastolic average blood pressure (in mmHg) across six measurements.

Details

Blood pressure measurements in survey settings may require adjustment to account for measurement conditions and equipment differences. This function applies a standardized adjustment using the formula: dbp_adj_mmhg = 15.6 + (0.83 * bpmdpbpd).

```
  **Missing Data Codes:**
- `996`: Valid skip (e.g., measurement not taken). Handled as `haven::tagged_na("a")`.
- `997-999`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
```

Value

numeric The adjusted diastolic blood pressure as a numeric.

See Also

adjust_sbp() for systolic blood pressure adjustment, derive_hypertension() for hypertension classification

Examples

```
# Scalar usage: Single respondent
# Example: Adjust for a respondent with average diastolic blood pressure of 80 mmHg.
adjust_dbp(bpmdpbpd = 80)
# Output: 82

# Example: Adjust for a respondent with a non-response diastolic blood pressure of 996.
result <- adjust_dbp(bpmdpbpd = 996)
result # Shows: NA
haven::is_tagged_na(result, "a") # Shows: TRUE (confirms it's tagged NA(a))
format(result, tag = TRUE) # Shows: "NA(a)" (displays the tag)

# Multiple respondents
adjust_dbp(bpmdpbpd = c(80, 90, 100))
# Returns: c(82, 90.3, 98.6)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(dbp_adj_mmhg = adjust_dbp(bpmdpbpd)) |>
  head()
```

---

adjust_sbp *Adjusted systolic blood pressure*

Description

This function adjusts systolic blood pressure based on the respondent's systolic average blood pressure across six measurements. The adjustment is made using specific correction factors. The adjusted systolic blood pressure is returned as a numeric value.

Usage

```
adjust_sbp(bpmdpbps)
```

Arguments

bpmdpbps	numeric A numeric representing the respondent's systolic average blood pressure (in mmHg) across six measurements.

Details

Blood pressure measurements in survey settings may require adjustment to account for measurement conditions and equipment differences. This function applies a standardized adjustment using the formula: sbp_adj_mmhg = 11.4 + (0.93 * bpmdpbps).

```
  **Missing Data Codes:**
- `996`: Valid skip (e.g., measurement not taken). Handled as `haven::tagged_na("a")`.
- `997-999`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
```

Value

numeric The adjusted systolic blood pressure as a numeric.

See Also

`adjust_dbp()` for diastolic blood pressure adjustment, `derive_hypertension()` for hypertension classification

Examples

```
# Scalar usage: Single respondent
# Example: Adjust for a respondent with average systolic blood pressure of 120 mmHg.
adjust_sbp(bpmdpbps = 120)
# Output: 123

# Example: Adjust for a respondent with a non-response systolic blood pressure of 996.
result <- adjust_sbp(bpmdpbps = 996)
result # Shows: NA
haven::is_tagged_na(result, "a") # Shows: TRUE (confirms it's tagged NA(a))
format(result, tag = TRUE) # Shows: "NA(a)" (displays the tag)

# Multiple respondents
adjust_sbp(bpmdpbps = c(120, 130, 140))
# Returns: c(123, 132.3, 141.6)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(sbp_adj_mmhg = adjust_sbp(bpmdpbps)) |>
  head()
```

`aggregate_meds_by_person` *Aggregate medication variables to one row per person*

Description

Collapses long-format medication data (cycles 3-6) to one row per respondent by taking the maximum value of each medication variable across all rows. A result of 1 (taking the medication) takes precedence over 0. Tagged NAs are propagated only when all rows for a respondent are missing.

Usage

```
aggregate_meds_by_person(data, variables, by = "clinicid")
```

Arguments

`data`	data.frame Long-format medication data with multiple rows per respondent.
`variables`	character Variable names to aggregate.
`by`	character The respondent identifier column. Default is `"clinicid"`.

Value

data.frame One row per respondent with aggregated medication variables as numeric.

See Also

`recode_meds_cycles3to6()`, `recode_meds_cycles1to2()`

Examples

```
df <- data.frame(
  clinicid    = c(1, 1, 2, 2),
  any_htn_med = c(0, 1, 0, 0),
  diab_med    = c(1, 0, 0, 0)
)
aggregate_meds_by_person(df, variables = c("any_htn_med", "diab_med"))
```

---

calculate_exercise_daily_avg

Average daily minutes of moderate-to-vigorous physical activity (MVPA) from accelerometer data

---

Description

This function calculates the average daily minutes of moderate-to-vigorous physical activity (MVPA) across a week of accelerometer measurement. It takes seven parameters, each representing the MVPA minutes on a specific day (Day 1 to Day 7). The function computes the daily average across the week.

Usage

```
calculate_exercise_daily_avg(
  ammdmva1,
  ammdmva2,
  ammdmva3,
  ammdmva4,
  ammdmva5,
  ammdmva6,
  ammdmva7
)
```

Arguments

ammdmva1	numeric A numeric representing minutes of moderate-to-vigorous physical activity (MVPA) on Day 1 of accelerometer measurement.
ammdmva2	numeric A numeric representing minutes of MVPA on Day 2 of accelerometer measurement.
ammdmva3	numeric A numeric representing minutes of MVPA on Day 3 of accelerometer measurement.
ammdmva4	numeric A numeric representing minutes of MVPA on Day 4 of accelerometer measurement.
ammdmva5	numeric A numeric representing minutes of MVPA on Day 5 of accelerometer measurement.
ammdmva6	numeric A numeric representing minutes of MVPA on Day 6 of accelerometer measurement.
ammdmva7	numeric A numeric representing minutes of MVPA on Day 7 of accelerometer measurement.

Details

This function processes physical activity data from accelerometer measurements to create a weekly activity summary.

```
**Data Quality Requirements:**
- Requires complete 7-day data (missing days result in tagged NA)
- This conservative approach ensures reliable activity estimates
- Zero values are preserved (represent valid no-activity days)

**Missing Data Codes:**
- For all input variables:
  - `9996`: Valid skip. Handled as `haven::tagged_na("a")`.
- `9997-9999`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
```

Value

numeric The average daily minutes of MVPA across a week of accelerometer use. If inputs are invalid or out of bounds, the function returns a tagged NA.

See Also

calculate_exercise_weekly() for activity unit conversion, categorize_exercise() for activity level classification

Examples

```
# Scalar usage: Single respondent
# Example: Calculate the average minutes of exercise per day for a week of accelerometer data.
calculate_exercise_daily_avg(30, 40, 25, 35, 20, 45, 50)
# Output: 35

# Example: Respondent has non-response values for all inputs.
result <- calculate_exercise_daily_avg(9998, 9998, 9998, 9998, 9998, 9998, 9998)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
calculate_exercise_daily_avg(
  c(30, 20), c(40, 30), c(25, 35), c(35, 45),
  c(20, 25), c(45, 55), c(50, 60)
)
# Returns: c(35, 39.28571)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(avg_exercise = calculate_exercise_daily_avg(
    ammdmva1, ammdmva2,
    ammdmva3, ammdmva4, ammdmva5, ammdmva6, ammdmva7
  )) |>
  head()
```

---

calculate_exercise_weekly

Weekly minutes of moderate-to-vigorous physical activity (MVPA) from daily average

---

Description

This function takes the average daily minutes of moderate-to-vigorous physical activity (MVPA) across a week of accelerometer use as an input (mvpa_min) and calculates the equivalent weekly MVPA minutes. The result is returned as a numeric value.

Usage

```
calculate_exercise_weekly(mvpa_min)
```

Arguments

mvpa_min	numeric A numeric representing the average daily minutes of moderate-to-vigorous physical activity (MVPA) across a week of accelerometer use.

Details

The function multiplies the average daily MVPA minutes (mvpa_min) by 7 to obtain the equivalent weekly MVPA minutes.

```
**Missing Data Codes:**
- Propagates tagged NAs from the input `mvpa_min`.
```

Value

numeric The total weekly minutes of MVPA. If inputs are invalid or out of bounds, the function returns a tagged NA.

See Also

calculate_exercise_daily_avg(), categorize_exercise()

Examples

```
# Scalar usage: Single respondent
# Example: Convert average daily MVPA minutes to weekly MVPA minutes.
calculate_exercise_weekly(35)
# Output: 245

# Multiple respondents
calculate_exercise_weekly(c(35, 40, 20))
# Returns: c(245, 280, 140)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(avg_exercise = calculate_exercise_daily_avg(
    ammdmva1, ammdmva2,
    ammdmva3, ammdmva4, ammdmva5, ammdmva6, ammdmva7
  )) |>
  mutate(min_per_week = calculate_exercise_weekly(avg_exercise)) |>
  head()
```

`calculate_fv_daily_cycles1to2` *Daily fruit and vegetable consumption in a year - cycles 1-2*

Description

This function calculates the daily fruit and vegetable consumption in a year for respondent in the Canadian Health Measures Survey (CHMS) cycles 1-2. It takes seven parameters, each representing the number of times per year a specific fruit or vegetable item was consumed. The function then sums up the consumption frequencies of all these items and divides the total by 365 to obtain the average daily consumption of fruits and vegetables in a year.

Usage

```
calculate_fv_daily_cycles1to2(
  wsdd14y,
  gfvd17y,
  gfvd18y,
  gfvd19y,
  gfvd20y,
  gfvd22y,
  gfvd23y
)
```

Arguments

`wsdd14y`	numeric A numeric vector representing the number of times per year fruit juice was consumed.
`gfvd17y`	numeric A numeric vector representing the number of times per year fruit (excluding juice) was consumed.
`gfvd18y`	numeric A numeric vector representing the number of times per year tomato or tomato sauce was consumed.
`gfvd19y`	numeric A numeric vector representing the number of times per year lettuce or green leafy salad was consumed.
`gfvd20y`	numeric A numeric vector representing the number of times per year spinach, mustard greens, and cabbage were consumed.
`gfvd22y`	numeric A numeric vector representing the number of times per year potatoes were consumed.
`gfvd23y`	numeric A numeric vector representing the number of times per year other vegetables were consumed.

Details

The function calculates the total consumption of fruits and vegetables in a year by summing up the consumption frequencies of all the input items. It then divides the total by 365 to obtain the average daily consumption of fruits and vegetables in a year.

```
**Missing Data Codes:**
- For all input variables:
  - `9996`: Valid skip. Handled as `haven::tagged_na("a")`.
- `9997-9999`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
```

Value

numeric The average times per day fruits and vegetables were consumed in a year. If inputs are invalid or out of bounds, the function returns a tagged NA.

See Also

`calculate_fv_daily_cycles3to6()` for cycles 3-6 fruit and vegetable consumption, `categorize_diet_quality()` for overall diet quality

Examples

```
# Scalar usage: Single respondent
# Example: Calculate average daily fruit and vegetable consumption for a cycle 1-2 respondent.
calculate_fv_daily_cycles1to2(
  wsdd14y = 50, gfvd17y = 150, gfvd18y = 200, gfvd19y = 100, gfvd20y = 80,
  gfvd22y = 120, gfvd23y = 90
)
# Output: 2.164384

# Example: Respondent has non-response values for all inputs.
result <- calculate_fv_daily_cycles1to2(
  wsdd14y = 9998, gfvd17y = 9998, gfvd18y = 9998, gfvd19y = 9998, gfvd20y = 9998,
  gfvd22y = 9998, gfvd23y = 9998
)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
calculate_fv_daily_cycles1to2(
  wsdd14y = c(50, 60), gfvd17y = c(150, 160), gfvd18y = c(200, 210), gfvd19y = c(100, 110),
  gfvd20y = c(80, 90), gfvd22y = c(120, 130), gfvd23y = c(90, 100)
)
# Returns: c(2.164384, 2.356164)

# Database usage: Applied to survey datasets
library(dplyr)
cycle2 |>
  mutate(total_fv = calculate_fv_daily_cycles1to2(
    wsdd14y, gfvd17y, gfvd18y,
    gfvd19y, gfvd20y, gfvd22y, gfvd23y
  )) |>
  head()
```

`calculate_fv_daily_cycles3to6` *Daily fruit and vegetable consumption in a year - cycles 3-6*

Description

This function calculates the daily fruit and vegetable consumption in a year for respondents in the Canadian Health Measures Survey (CHMS) cycles 3-6. It takes eleven parameters, each representing the number of times per year a specific fruit or vegetable item was consumed. The function then sums up the consumption frequencies of all these items and divides the total by 365 to obtain the average daily consumption of fruits and vegetables in a year.

Usage

```
calculate_fv_daily_cycles3to6(
  wsdd34y,
  wsdd35y,
  gfvd17ay,
  gfvd17by,
  gfvd17cy,
  gfvd17dy,
  gfvd18y,
  gfvd19y,
  gfvd20y,
  gfvd22y,
  gfvd23y
)
```

Arguments

`wsdd34y`	numeric A numeric vector representing the number of times per year orange or grapefruit juice was consumed.
`wsdd35y`	numeric A numeric vector representing the number of times per year other fruit juices were consumed.
`gfvd17ay`	numeric A numeric vector representing the number of times per year citrus fruits were consumed.
`gfvd17by`	numeric A numeric vector representing the number of times per year strawberries were consumed (in summer).
`gfvd17cy`	numeric A numeric vector representing the number of times per year strawberries were consumed (outside summer).
`gfvd17dy`	numeric A numeric vector representing the number of times per year other fruits were consumed.
`gfvd18y`	numeric A numeric vector representing the number of times per year tomato or tomato sauce was consumed.

gfvd19y numeric A numeric vector representing the number of times per year lettuce or green leafy salad was consumed.

gfvd20y numeric A numeric vector representing the number of times per year spinach, mustard greens, and cabbage were consumed.

gfvd22y numeric A numeric vector representing the number of times per year potatoes were consumed.

gfvd23y numeric A numeric vector representing the number of times per year other vegetables were consumed.

Details

The function calculates the total consumption of fruits and vegetables in a year by summing up the consumption frequencies of all the input items. It then divides the total by 365 to obtain the average daily consumption of fruits and vegetables in a year.

```
**Missing Data Codes:**
- For all input variables:
  - `9996`: Valid skip. Handled as `haven::tagged_na("a")`.
- `9997-9999`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
```

Value

numeric The average times per day fruits and vegetables were consumed in a year. If inputs are invalid or out of bounds, the function returns a tagged NA.

See Also

calculate_fv_daily_cycles1to2() for cycles 1-2 fruit and vegetable consumption, categorize_diet_quality() for overall diet quality

Examples

```
# Scalar usage: Single respondent
# Example: Calculate average daily fruit and vegetable consumption for a cycle 3-6 respondent
calculate_fv_daily_cycles3to6(
  wsdd34y = 50, wsdd35y = 100, gfvd17ay = 150, gfvd17by = 80, gfvd17cy = 40,
  gfvd17dy = 200, gfvd18y = 100, gfvd19y = 80, gfvd20y = 60, gfvd22y = 120, gfvd23y = 90
)
# Output: 2.931507

# Example: Respondent has non-response values for all inputs.
result <- calculate_fv_daily_cycles3to6(
  wsdd34y = 9998, wsdd35y = 9998, gfvd17ay = 9998, gfvd17by = 9998, gfvd17cy = 9998,
 gfvd17dy = 9998, gfvd18y = 9998, gfvd19y = 9998, gfvd20y = 9998, gfvd22y = 9998, gfvd23y = 9998
)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
```

```
calculate_fv_daily_cycles3to6(
  wsdd34y = c(50, 60), wsdd35y = c(100, 110), gfvd17ay = c(150, 160), gfvd17by = c(80, 90),
  gfvd17cy = c(40, 50), gfvd17dy = c(200, 210), gfvd18y = c(100, 110), gfvd19y = c(80, 90),
   gfvd20y = c(60, 70), gfvd22y = c(120, 130), gfvd23y = c(90, 100)
)
# Returns: c(2.931507, 3.232877)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(total_fv = calculate_fv_daily_cycles3to6(
    wsdd34y, wsdd35y, gfvd17ay,
    gfvd17by, gfvd17cy, gfvd17dy, gfvd18y, gfvd19y, gfvd20y, gfvd22y, gfvd23y
  )) |>
  head()
```

`calculate_gfr` *Estimated glomerular filtration rate (GFR)*

Description

This function calculates the estimated glomerular filtration rate (GFR) according to Finlay's formula, where serum creatine is in mg/dL. The calculation takes into account the respondent's ethnicity, sex, and age.

Usage

```
calculate_gfr(lab_bcre, pgdcgt, clc_sex, clc_age)
```

Arguments

`lab_bcre`	numeric Blood creatine (umol/L). It should be a numeric between 14 and 785.
`pgdcgt`	integer Ethnicity (13 categories). It should be an integer between 1 and 13.
`clc_sex`	integer Sex (Male = 1, Female = 2). It should be an integer of either 1 or 2.
`clc_age`	numeric Age (years). It should be a numeric between 3 and 79.

Details

This function implements the Modification of Diet in Renal Disease (MDRD) equation to estimate glomerular filtration rate, a key indicator of kidney function.

```
**Clinical Significance:**
GFR estimates are essential for:
- Chronic kidney disease (CKD) classification
- Medication dosing adjustments
- Cardiovascular risk assessment
```

```
**Formula Application:**
Base: GFR = 175 x (creatinine^-1.154) x (age^-0.203)
Adjustments:
- Female: x 0.742
- Black ethnicity: x 1.210

**Unit Conversion:**
Serum creatinine converted from umol/L to mg/dL (/ 88.4)

**Missing Data Codes:**
- `lab_bcre`: `9996` (Not applicable), `9997-9999` (Missing)
- `pgdcgt`: `96` (Not applicable), `97-99` (Missing)
- `clc_sex`: `6` (Not applicable), `7-9` (Missing)
- `clc_age`: `96` (Not applicable), `97-99` (Missing)
```

Value

numeric The calculated GFR. If inputs are invalid or out of bounds, the function returns a tagged NA.

See Also

`categorize_ckd()` for CKD classification based on GFR values

Examples

```
# Scalar usage: Single respondent
# Example 1: Calculate gfr for a 45-year-old white female with serum creatine of 80 umol/L.
calculate_gfr(lab_bcre = 80, pgdcgt = 1, clc_sex = 2, clc_age = 45)
# Output: 67.27905

# Example 2: Respondent has non-response values for all inputs.
result <- calculate_gfr(lab_bcre = 9998, pgdcgt = 98, clc_sex = 8, clc_age = 98)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
calculate_gfr(
  lab_bcre = c(80, 70, 90), pgdcgt = c(1, 2, 1),
  clc_sex = c(2, 2, 1), clc_age = c(45, 35, 50)
)
# Returns: c(67.27905, 99.94114, 70.38001)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(gfr = calculate_gfr(lab_bcre, pgdcgt, clc_sex, clc_age)) |>
  head()
```

`calculate_household_income` *Adjusted total household income*

Description

This function calculates the adjusted total household income based on the respondent's income amount and actual household size, taking into account the weighted household size.

Usage

```
calculate_household_income(thi_01, dhhdhsz)
```

Arguments

`thi_01`	numeric A numeric representing the respondent's household income amount in dollars.
`dhhdhsz`	integer An integer representing the respondent's actual household size in persons.

Details

This function applies equivalence scales to adjust household income for household size, allowing for meaningful income comparisons across different household compositions.

```
**Equivalence Scale Logic:**
- First adult: Weight = 1.0 (full weight)
- Second adult: Weight = 0.4 (economies of scale)
- Additional members: Weight = 0.3 each (further economies)

**Missing Data Codes:**
- `thi_01`:
  - `99999996`: Valid skip. Handled as `haven::tagged_na("a")`.
- `99999997-99999999`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
- `dhhdhsz`:
  - `96`: Valid skip. Handled as `haven::tagged_na("a")`.
- `97-99`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
```

Value

numeric The calculated adjusted total household income as a numeric. If inputs are invalid or out of bounds, the function returns a tagged NA.

See Also

`categorize_income_quintile()` for income classification, `is_lowest_income_quintile()` for poverty indicators

Examples

```
# Scalar usage: Single respondent
# Example 1: Respondent with $50,000 income and a household size of 3.
calculate_household_income(thi_01 = 50000, dhhdhsz = 3)
# Output: 29411.76

# Example 2: Respondent has non-response values for all inputs.
result <- calculate_household_income(thi_01 = 99999998, dhhdhsz = 98)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
calculate_household_income(thi_01 = c(50000, 75000, 90000), dhhdhsz = c(3, 2, 1))
# Returns: c(29411.76, 53571.43, 90000)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(adj_hh_income = calculate_household_income(thi_01, dhhdhsz)) |>
  head()
```

---

calculate_nonhdl *Non-HDL cholesterol level*

Description

This function calculates a respondent's non-HDL cholesterol level by subtracting their HDL cholesterol level from their total cholesterol level. It first checks whether the input values `lab_chol` (total cholesterol) and `lab_hdl` (HDL cholesterol) are within valid ranges.

Usage

```
calculate_nonhdl(lab_chol, lab_hdl)
```

Arguments

`lab_chol`	numeric A numeric representing a respondent's total cholesterol level in mmol/L.
`lab_hdl`	numeric A numeric representing a respondent's HDL cholesterol level in mmol/L.

Details

The function calculates the non-HDL cholesterol level by subtracting the HDL cholesterol level from the total cholesterol level.

```
**Missing Data Codes:**
- `lab_chol`:
  - `99.96`: Valid skip. Handled as `haven::tagged_na("a")`.
- `99.97-99.99`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
- `lab_hdl`:
  - `9.96`: Valid skip. Handled as `haven::tagged_na("a")`.
- `9.97-9.99`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
```

Value

numeric The calculated non-HDL cholesterol level (in mmol/L). If inputs are invalid or out of bounds, the function returns a tagged NA.

See Also

categorize_nonhdl()

Examples

```
# Scalar usage: Single respondent
# Example: Respondent has total cholesterol of 5.0 mmol/L and HDL cholesterol of 1.5 mmol/L.
calculate_nonhdl(lab_chol = 5.0, lab_hdl = 1.5)
# Output: 3.5

# Example: Respondent has non-response values for cholesterol.
result <- calculate_nonhdl(lab_chol = 99.98, lab_hdl = 1.5)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
calculate_nonhdl(lab_chol = c(5.0, 6.0, 7.0), lab_hdl = c(1.5, 1.0, 2.0))
# Returns: c(3.5, 5.0, 5.0)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(non_hdl = calculate_nonhdl(lab_chol, lab_hdl)) |>
  head()
```

---

calculate_pack_years *Smoking pack-years*

Description

This function calculates an individual's smoking pack-years based on various CHMS smoking variables. Pack years is a measure used by researchers to quantify lifetime exposure to cigarette use.

Usage

```
calculate_pack_years(
  smkdsty,
  clc_age,
  smk_54,
  smk_52,
  smk_31,
  smk_41,
  smk_53,
  smk_42,
  smk_21,
  smk_11
)
```

Arguments

smkdsty	integer An integer representing the smoking status of the respondent.
clc_age	numeric A numeric representing the respondent's age.
smk_54	numeric A numeric representing the respondent's age when they stopped smoking daily.
smk_52	numeric A numeric representing the respondent's age when they first started smoking daily.
smk_31	integer An integer representing the number of cigarettes smoked per day for daily smokers.
smk_41	numeric A numeric representing the number of cigarettes smoked per day for occasional smokers.
smk_53	numeric A numeric representing the number of cigarettes smoked per day for former daily smokers.
smk_42	numeric A numeric representing the number of days in past month the respondent smoked at least 1 cigarette (for occasional smokers).
smk_21	numeric A numeric representing the respondent's age when they first started smoking occasionally.
smk_11	integer An integer representing whether the respondent has smoked at least 100 cigarettes in their lifetime.

Details

Pack-years is a standardized measure of lifetime cigarette exposure used in epidemiological research and clinical practice. The calculation varies by smoking pattern:

```
**Smoking Patterns:**
- Daily smokers: Consistent daily consumption over time period
- Occasional smokers: Variable consumption adjusted for frequency
- Former smokers: Historical consumption during smoking periods
```

```
**Minimum Values:**
The function applies minimum pack-year values (0.0137 or 0.007) to prevent
underestimation of health risks for light smokers.

**Missing Data Codes:**
- `smkdsty`: `96` (Not applicable), `97-99` (Missing)
- `clc_age`: `96` (Not applicable), `97-99` (Missing)
- Other variables: Handled within the formula logic.
```

Value

numeric A numeric representing the pack years for the respondent's smoking history. If inputs are invalid or out of bounds, the function returns a tagged NA.

See Also

https://big-life-lab.github.io/cchsflow/reference/calculate_pack_years.html

Examples

```
# Scalar usage: Single respondent
# A former occasional smoker who smoked at least 100 cigarettes in their lifetime.
calculate_pack_years(
  smkdsty = 5, clc_age = 50, smk_54 = 40, smk_52 = 18, smk_31 = NA,
  smk_41 = 15, smk_53 = NA, smk_42 = 3, smk_21 = 25, smk_11 = 1
)
# Output: 0.0137

# Example: Respondent has non-response values for all inputs.
result <- calculate_pack_years(
  smkdsty = 98, clc_age = 998, smk_54 = 98, smk_52 = 98, smk_31 = 98,
  smk_41 = 98, smk_53 = 98, smk_42 = 98, smk_21 = 98, smk_11 = 8
)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
calculate_pack_years(
  smkdsty = c(1, 5, 6),
  clc_age = c(40, 50, 60),
  smk_52 = c(20, 18, NA),
  smk_31 = c(30, NA, NA),
  smk_54 = c(NA, 40, NA),
  smk_41 = c(NA, 15, NA),
  smk_53 = c(NA, NA, NA),
  smk_42 = c(NA, 3, NA),
  smk_21 = c(NA, 25, NA),
  smk_11 = c(NA, 1, NA)
)
# Returns: c(30, 0.0137, 0)
```

```
# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(pack_years = calculate_pack_years(
    smkdsty, clc_age, smk_54, smk_52,
    smk_31, smk_41, smk_53, smk_42, smk_21, smk_11
  )) |>
  head()
```

`calculate_waist_height_ratio` *Waist-to-height ratio (WHtR)*

Description

This function calculates the Waist-to-Height Ratio (WHtR) by dividing the waist circumference by the height of the respondent.

Usage

```
calculate_waist_height_ratio(hwm_11cm, hwm_14cx)
```

Arguments

`hwm_11cm`	numeric A numeric representing the height of the respondent in centimeters.
`hwm_14cx`	numeric A numeric representing the waist circumference of the respondent in centimeters.

Details

This function calculates the Waist-to-Height Ratio (WHtR), an indicator of central obesity.

```
**Missing Data Codes:**
- `hwm_11cm`:
  - `999.96`: Valid skip. Handled as `haven::tagged_na("a")`.
- `999.97-999.99`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
- `hwm_14cx`:
  - `999.6`: Valid skip. Handled as `haven::tagged_na("a")`.
- `999.7-999.9`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
```

Value

numeric The WHtR. If inputs are invalid or out of bounds, the function returns a tagged NA.

Examples

```
# Scalar usage: Single respondent
# Example 1: Calculate WHtR for a respondent with height = 170 cm and waist circumference = 85 cm.
calculate_waist_height_ratio(hwm_11cm = 170, hwm_14cx = 85)
# Output: 0.5 (85/170)

# Example 2: Calculate WHtR for a respondent with missing height.
result <- calculate_waist_height_ratio(hwm_11cm = 999.98, hwm_14cx = 85)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
calculate_waist_height_ratio(hwm_11cm = c(170, 180, 160), hwm_14cx = c(85, 90, 80))
# Returns: c(0.5, 0.5, 0.5)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(whtr = calculate_waist_height_ratio(hwm_11cm, hwm_14cx)) |>
  head()
```

---

categorize_ckd *Chronic kidney disease (CKD) derived variable*

Description

This function categorizes individuals' glomerular filtration rate (GFR) into stages of Chronic Kidney Disease (CKD).

Usage

```
categorize_ckd(gfr)
```

Arguments

gfr	numeric A numeric representing the glomerular filtration rate.

Details

This function applies the Kidney Disease: Improving Global Outcomes (KDIGO) guideline to classify Chronic Kidney Disease (CKD) based on GFR.

```
**Missing Data Codes:**
- Propagates tagged NAs from the input `gfr`.
```

Value

integer The CKD stage:

- 1: GFR of 60 or below (indicating CKD)
- 2: GFR above 60 (not indicating CKD)
- `haven::tagged_na("a")`: Not applicable
- `haven::tagged_na("b")`: Missing

References

Kidney Disease: Improving Global Outcomes (KDIGO) CKD Work Group. (2013). KDIGO 2012 clinical practice guideline for the evaluation and management of chronic kidney disease. Kidney international supplements, 3(1), 1-150.

See Also

`calculate_gfr()`

Examples

```
# Scalar usage: Single respondent
# Example 1: Categorize a GFR of 45
categorize_ckd(45)
# Output: 1

# Example 2: Categorize a GFR of 75
categorize_ckd(75)
# Output: 2

# Example 3: Respondent has a non-response value for GFR.
result <- categorize_ckd(haven::tagged_na("b"))
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
categorize_ckd(c(45, 75, 60))
# Returns: c(1, 2, 1)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(gfr = calculate_gfr(lab_bcre, pgdcgt, clc_sex, clc_age)) |>
  mutate(ckd = categorize_ckd(gfr)) |>
  head()
```

`categorize_diet_quality` *Categorical diet indicator*

Description

This function categorizes individuals' diet quality based on their total fruit and vegetable consumption.

Usage

```
categorize_diet_quality(fv_daily)
```

Arguments

`fv_daily`	numeric A numeric vector representing the average times per day fruits and vegetables were consumed in a year.

Details

This function categorizes diet quality based on the widely recognized "5-a-day" recommendation for fruit and vegetable intake.

```
**Missing Data Codes:**
- Propagates tagged NAs from the input `fv_daily`.
```

Value

integer A categorical indicating the diet quality:

- 1: Good diet (fv_daily >= 5)
- 2: Poor diet (fv_daily < 5)
- `haven::tagged_na("a")`: Valid skip
- `haven::tagged_na("b")`: Missing

See Also

`calculate_fv_daily_cycles1to2()`, `calculate_fv_daily_cycles3to6()`

Examples

```
# Scalar usage: Single respondent
# Example 1: Categorize a fv_daily value of 3 as poor diet
categorize_diet_quality(3)
# Output: 2

# Example 2: Categorize a fv_daily value of 7 as good diet
categorize_diet_quality(7)
```

```
# Output: 1

# Multiple respondents
categorize_diet_quality(c(3, 7, 5))
# Returns: c(2, 1, 1)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(total_fv = calculate_fv_daily_cycles3to6(
    wsdd34y, wsdd35y, gfvd17ay,
    gfvd17by, gfvd17cy, gfvd17dy, gfvd18y, gfvd19y, gfvd20y, gfvd22y, gfvd23y
  )) |>
  mutate(diet_quality = categorize_diet_quality(total_fv)) |>
  head()
```

`categorize_exercise` *Categorical weekly moderate-to-vigorous physical activity (MVPA) indicator*

Description

This function categorizes individuals' weekly moderate-to-vigorous physical activity (MVPA) levels against the 150 minutes/week guideline.

Usage

```
categorize_exercise(exercise_min_week)
```

Arguments

`exercise_min_week`
: numeric A numeric representing an individual's minutes of moderate-to-vigorous physical activity (MVPA) per week.

Details

This function applies the national physical activity guideline of 150 minutes of moderate-to-vigorous physical activity (MVPA) per week.

```
**Missing Data Codes:**
- Propagates tagged NAs from the input `exercise_min_week`.
```

Value

integer A categorical indicating the MVPA category:

- 1: Meets or exceeds the recommended 150 minutes of MVPA per week (exercise_min_week >= 150)
- 2: Below the recommended 150 minutes of MVPA per week (exercise_min_week < 150)
- `haven::tagged_na("a")`: Not applicable
- `haven::tagged_na("b")`: Missing

See Also

`calculate_exercise_weekly()`

Examples

```
# Scalar usage: Single respondent
# Example 1: Categorize 180 minutes of MVPA per week as meeting the recommendation
categorize_exercise(180)
# Output: 1

# Example 2: Categorize 120 minutes of MVPA per week as below the recommendation
categorize_exercise(120)
# Output: 2

# Multiple respondents
categorize_exercise(c(180, 120, 150))
# Returns: c(1, 2, 1)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(avg_exercise = calculate_exercise_daily_avg(
    ammdmva1, ammdmva2,
    ammdmva3, ammdmva4, ammdmva5, ammdmva6, ammdmva7
  )) |>
  mutate(min_per_week = calculate_exercise_weekly(avg_exercise)) |>
  mutate(pa_category = categorize_exercise(min_per_week)) |>
  head()
```

`categorize_income_quintile` *Categorical adjusted household income*

Description

This function categorizes individuals' adjusted household income based on specified income ranges.

Usage

```
categorize_income_quintile(adj_hh_income)
```

Arguments

adj_hh_income	numeric A numeric representing the adjusted household income.

Details

This function segments adjusted household income into quintiles, providing a standardized measure of socioeconomic status.

```
**Missing Data Codes:**
- Propagates tagged NAs from the input `adj_hh_income`.
```

Value

integer The income category:

- 1: Below or equal to $21,500
- 2: Above $21,500 and up to $35,000
- 3: Above $35,000 and up to $50,000
- 4: Above $50,000 and up to $70,000
- 5: Above $70,000
- `haven::tagged_na("a")`: Not applicable
- `haven::tagged_na("b")`: Missing

See Also

`calculate_household_income()`, `is_lowest_income_quintile()`

Examples

```
# Scalar usage: Single respondent
# Example 1: Categorize a household income of $25,000
categorize_income_quintile(25000)
# Output: 2

# Example 2: Categorize a household income of $45,000
categorize_income_quintile(45000)
# Output: 3

# Multiple respondents
categorize_income_quintile(c(25000, 45000, 80000))
# Returns: c(2, 3, 5)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
```

```
mutate(adj_hh_income = calculate_household_income(thi_01, dhhdhsz)) |>
mutate(income_category = categorize_income_quintile(adj_hh_income)) |>
head()
```

`categorize_nonhdl` *Categorical non-HDL cholesterol level*

Description

This function categorizes individuals' non-HDL cholesterol levels based on a threshold value.

Usage

```
categorize_nonhdl(nonhdl)
```

Arguments

nonhdl	numeric A numeric representing an individual's non-HDL cholesterol level.

Details

This function categorizes non-HDL cholesterol levels into 'High' or 'Normal' based on a 4.3 mmol/L threshold.

```
**Missing Data Codes:**
- Propagates tagged NAs from the input `nonhdl`.
```

Value

integer A categorical indicating the non-HDL cholesterol category:

- 1: High non-HDL cholesterol (nonhdl >= 4.3)
- 2: Normal non-HDL cholesterol (nonhdl < 4.3)
- `haven::tagged_na("a")`: Not applicable
- `haven::tagged_na("b")`: Missing

See Also

`calculate_nonhdl()`

Examples

```
# Scalar usage: Single respondent
# Example 1: Categorize a nonhdl value of 5.0 as high non-HDL cholesterol
categorize_nonhdl(5.0)
# Output: 1

# Example 2: Categorize a nonhdl value of 3.8 as normal non-HDL cholesterol
categorize_nonhdl(3.8)
# Output: 2

# Multiple respondents
categorize_nonhdl(c(5.0, 3.8, 4.3))
# Returns: c(1, 2, 1)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(non_hdl = calculate_nonhdl(lab_chol, lab_hdl)) |>
  mutate(non_hdl_category = categorize_nonhdl(non_hdl)) |>
  head()
```

---

cycle1 *Canadian Health Measures Survey (CHMS) Cycle 1*

Description

This is dummy data representing the second cycle of the Canadian Health Measures Survey (CHMS). The CHMS survey is conducted by Statistics Canada.

Usage

```
data(cycle1)
```

Source

Statistics Canada

Examples

```
data(cycle1)
str(cycle1)
```

cycle1_meds *Canadian Health Measures Survey (CHMS) Cycle 1 Medications*

Description

This dummy data representing the medication portion of the second cycle of the Canadian Health Measures Survey (CHMS). The CHMS survey is conducted by Statistics Canada.

Usage

```
data(cycle1_meds)
```

Format

A data frame with X rows and Y columns.

Source

Statistics Canada

Examples

```
data(cycle1_meds)
str(cycle1_meds)
```

cycle2 *Canadian Health Measures Survey (CHMS) Cycle 2*

Description

This is dummy data representing the second cycle of the Canadian Health Measures Survey (CHMS). The CHMS survey is conducted by Statistics Canada.

Usage

```
data(cycle2)
```

Source

Statistics Canada

Examples

```
data(cycle2)
str(cycle2)
```

cycle2_meds *Canadian Health Measures Survey (CHMS) Cycle 2 Medications*

Description

This dummy data representing the medication portion of the second cycle of the Canadian Health Measures Survey (CHMS). The CHMS survey is conducted by Statistics Canada.

Usage

```
data(cycle2_meds)
```

Source

Statistics Canada

Examples

```
data(cycle2_meds)
str(cycle2_meds)
```

cycle3 *Canadian Health Measures Survey (CHMS) Cycle 3*

Description

This is dummy data representing the third cycle of the Canadian Health Measures Survey (CHMS). The CHMS survey is conducted by Statistics Canada.

Usage

```
data(cycle3)
```

Source

Statistics Canada

Examples

```
data(cycle3)
str(cycle3)
```

cycle3_meds *Canadian Health Measures Survey (CHMS) Cycle 3 Medications*

Description

This dummy data representing the medication portion of the third cycle of the Canadian Health Measures Survey (CHMS). The CHMS survey is conducted by Statistics Canada.

Usage

```
data(cycle3_meds)
```

Source

Statistics Canada

Examples

```
data(cycle3_meds)
str(cycle3_meds)
```

cycle4 *Canadian Health Measures Survey (CHMS) Cycle 4*

Description

This is dummy data representing the fourth cycle of the Canadian Health Measures Survey (CHMS). The CHMS survey is conducted by Statistics Canada.

Usage

```
data(cycle4)
```

Source

Statistics Canada

Examples

```
data(cycle4)
str(cycle4)
```

cycle4_meds *Canadian Health Measures Survey (CHMS) Cycle 4 Medications*

Description

This dummy data representing the medication portion of the third cycle of the Canadian Health Measures Survey (CHMS). The CHMS survey is conducted by Statistics Canada.

Usage

```
data(cycle4_meds)
```

Source

Statistics Canada

Examples

```
data(cycle4_meds)
str(cycle4_meds)
```

cycle5 *Canadian Health Measures Survey (CHMS) Cycle 5*

Description

This is dummy data representing the fifth cycle of the Canadian Health Measures Survey (CHMS). The CHMS survey is conducted by Statistics Canada.

Usage

```
data(cycle5)
```

Source

Statistics Canada

Examples

```
data(cycle5)
str(cycle5)
```

cycle5_meds *Canadian Health Measures Survey (CHMS) Cycle 5 Medications*

Description

This dummy data representing the medication portion of the third cycle of the Canadian Health Measures Survey (CHMS). The CHMS survey is conducted by Statistics Canada.

Usage

```
data(cycle5_meds)
```

Source

Statistics Canada

Examples

```
data(cycle5_meds)
str(cycle5_meds)
```

cycle6 *Canadian Health Measures Survey (CHMS) Cycle 6*

Description

This is dummy data representing the fifth cycle of the Canadian Health Measures Survey (CHMS). The CHMS survey is conducted by Statistics Canada.

Usage

```
data(cycle6)
```

Source

Statistics Canada

Examples

```
data(cycle6)
str(cycle6)
```

cycle6_meds *Canadian Health Measures Survey (CHMS) Cycle 6 Medications*

Description

This dummy data representing the medication portion of the third cycle of the Canadian Health Measures Survey (CHMS). The CHMS survey is conducted by Statistics Canada.

Usage

```
data(cycle6_meds)
```

Source

Statistics Canada

Examples

```
data(cycle6_meds)
str(cycle6_meds)
```

derive_alcohol_risk *Low risk drinking score*

Description

This function calculates a low drink score (step 1 only) for a respondent using Canada's Low-Risk Alcohol Drinking Guideline. The score is based solely on the number of standard drinks consumed per week and the respondent's sex. (Step 2, which would add additional points based on other drinking habits, is not included.).

Usage

```
derive_alcohol_risk(clc_sex, alc_11, alcdwky)
```

Arguments

clc_sex	integer An integer indicating the respondent's sex (1 for male, 2 for female).
alc_11	integer An integer indicating whether the respondent drank alcohol in the past year (1 for "Yes", 2 for "No").
alcdwky	integer An integer representing the number of standard drinks consumed by the respondent in a week.

Details

The scoring is determined by first allocating points (referred to as `step1`) based on the weekly alcohol consumption and the respondent's sex:

- If the respondent drank in the past year (alc_11 == 1):
  - For alcdwky <= 10, assign 0 points.
  - For alcdwky > 10 and <= 15: assign 0 points for males (clc_sex == 1) and 1 point for females (clc_sex == 2).
  - For alcdwky > 15 and <= 20: assign 1 point for males and 3 points for females.
  - For alcdwky > 20: assign 3 points.
- For respondents who did not drink in the past year (alc_11 == 2), 0 points are assigned.

These `step1` points are then mapped to the final categorical score as follows:

- 0 points -> score of 1 ("Low risk"),
- 1-2 points -> score of 2 ("Marginal risk"),
- 3-4 points -> score of 3 ("Medium risk"),
- 5-9 points -> score of 4 ("High risk").

This function implements Canada's Low-Risk Alcohol Drinking Guidelines (Step 1 only) to assess alcohol consumption risk. The scoring system considers both the quantity of alcohol consumed and biological sex differences in alcohol metabolism.

Risk Categories:

- Low risk (0 points): Safe consumption levels
- Marginal risk (1-2 points): Slightly elevated risk
- Medium risk (3-4 points): Moderate health concerns
- High risk (5-9 points): Significant health risks

Sex-Based Differences: Women generally have lower tolerance thresholds due to physiological differences in alcohol metabolism, reflected in the sex-specific point allocations.

Non-response Handling: Invalid inputs or survey non-response values result in tagged NA ("b").

Value

integer The low drink score, with:

- 1 for "Low risk" (0 points),
- 2 for "Marginal risk" (1-2 points),
- 3 for "Medium risk" (3-4 points), and
- 4 for "High risk" (5-9 points). If inputs are invalid or out of bounds, the function returns a tagged NA.

Note

This function implements only Step 1 of the guidelines. Step 2 (additional drinking pattern assessments) is not included due to data limitations in the survey.

References

Canada's Low-Risk Alcohol Drinking Guidelines, Health Canada

See Also

`derive_alcohol_risk_detailed()` for extended categorization including former/never drinkers

Examples

```
# Scalar usage: Single respondent
# Example: A male respondent who drank in the past year and consumes 3 standard drinks per week.
derive_alcohol_risk(clc_sex = 1, alc_11 = 1, alcdwky = 3)
# Expected output: 1 (Low risk)

# Missing data examples showing tagged NA patterns
result <- derive_alcohol_risk(clc_sex = 1, alc_11 = 6, alcdwky = 5)
result # Shows: NA
haven::is_tagged_na(result, "a") # Shows: TRUE (confirms it's tagged NA(a))
format(result, tag = TRUE) # Shows: "NA(a)" (displays the tag)

result <- derive_alcohol_risk(clc_sex = 1, alc_11 = 7, alcdwky = 5)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

result <- derive_alcohol_risk(clc_sex = 1, alc_11 = 1, alcdwky = NA)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
derive_alcohol_risk(clc_sex = c(1, 2, 1), alc_11 = c(1, 1, 2), alcdwky = c(3, 12, NA))
# Returns: c(1, 2, 1)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(alc_risk_score = derive_alcohol_risk(clc_sex, alc_11, alcdwky)) |>
  head()
```

---

derive_alcohol_risk_detailed

Low risk drinking score - former/never categories

Description

Computes a categorical alcohol consumption score based on Canada's Low-Risk Alcohol Drinking Guidelines (Step 1), while distinguishing between never, former, light, moderate, and heavy

drinkers. The function uses information about weekly consumption, past-year use, lifetime drinking, and history of heavy drinking.

Usage

```
derive_alcohol_risk_detailed(clc_sex, alc_11, alcdwky, alc_17, alc_18)
```

Arguments

clc_sex	integer Respondent's sex (1 = male, 2 = female).
alc_11	integer Whether the respondent drank alcohol in the past year (1 = Yes, 2 = No).
alcdwky	integer Number of standard drinks consumed in a typical week (0-84).
alc_17	integer Whether the respondent ever drank alcohol in their lifetime (1 = Yes, 2 = No).
alc_18	integer Whether the respondent regularly drank more than 12 drinks per week (1 = Yes, 2 = No).

Details

Step 1: Assign points based on weekly alcohol consumption.

- If the respondent drank in the past year (alc_11 == 1):
  - 0 to 10 drinks/week: 0 points
  - 11 to 15 drinks/week: 0 points for males, 1 point for females
  - 16 to 20 drinks/week: 1 point for males, 3 points for females
  - More than 20 drinks/week: 3 points for males, 5 points for females
- If they did not drink in the past year (alc_11 == 2): 0 points

Step 2: Determine the final categorical score.

- If the point score from Step 1 is 0, the final category is determined based on lifetime and past-year drinking habits:
  - A score of 1 (Never drinker) is assigned if the respondent either never drank alcohol in their lifetime or is a former drinker who did not regularly consume more than 12 drinks a week.
  - A score of 2 (Low-risk drinker) is assigned if the respondent drank in the past year (but still scored 0 points) or is a former drinker with a history of regularly consuming more than 12 drinks a week.
- If the point score from Step 1 is 1 or 2, the respondent is classified as a Moderate drinker (Score = 3).
- If the point score from Step 1 is 3 or more, the respondent is classified as a Heavy drinker (Score = 4). If inputs are invalid or out of bounds, the function returns a tagged NA.

Value

integer Score: 1 = Never drank, 2 = Low-risk (former or light) drinker, 3 = Moderate drinker (1–2 points), 4 = Heavy drinker (3–4 points). If inputs are invalid or out of bounds, the function returns a tagged NA.

Note

This function uses only Step 1 of the guidelines, as Step 2 information is unavailable in CHMS.

References

Canada's Low-Risk Alcohol Drinking Guidelines, Health Canada

See Also

`derive_alcohol_risk()` for basic risk scoring without drinking history

Examples

```
# Scalar usage: Single respondent
# Example: Male, drinks 3 drinks/week, drank in past year, no history of heavy drinking
derive_alcohol_risk_detailed(
  clc_sex = 1, alc_11 = 1, alcdwky = 3,
  alc_17 = 1, alc_18 = 2
)
# Expected output: 2

# Missing data examples showing tagged NA patterns
result <- derive_alcohol_risk_detailed(
  clc_sex = 1, alc_11 = 6, alcdwky = 5,
  alc_17 = 1, alc_18 = 2
)
result # Shows: NA
haven::is_tagged_na(result, "a") # Shows: TRUE (confirms it's tagged NA(a))
format(result, tag = TRUE) # Shows: "NA(a)" (displays the tag)

result <- derive_alcohol_risk_detailed(
  clc_sex = 1, alc_11 = 7, alcdwky = 5,
  alc_17 = 1, alc_18 = 2
)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

result <- derive_alcohol_risk_detailed(
  clc_sex = 1, alc_11 = 1, alcdwky = NA,
  alc_17 = 1, alc_18 = 2
)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
derive_alcohol_risk_detailed(
  clc_sex = c(1, 2, 1), alc_11 = c(1, 1, 2),
  alcdwky = c(3, 12, NA), alc_17 = c(1, 1, 1), alc_18 = c(2, 2, 1)
)
# Returns: c(2, 3, 2)
```

```
# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(alc_detailed_risk_score = derive_alcohol_risk_detailed(
    clc_sex, alc_11, alcdwky, alc_17, alc_18
  )) |>
  head()
```

`derive_cvd_family_history`

Cardiovascular Disease (CVD) family history

Description

This function evaluates a respondent's family history of cardiovascular disease (CVD), based on data about diagnoses of heart disease and stroke in immediate family members and the ages at which these diagnoses occurred. It identifies premature CVD if any diagnosis occurred before age 60.

Usage

```
derive_cvd_family_history(fmh_11, fmh_12, fmh_13, fmh_14)
```

Arguments

`fmh_11`	integer An integer: Indicates whether an immediate family member was diagnosed with heart disease. - 1 for "Yes" - 2 for "No".
`fmh_12`	numeric A numeric: Represents the youngest age at diagnosis of heart disease in an immediate family member.
`fmh_13`	integer An integer: Indicates whether an immediate family member was diagnosed with stroke. - 1 for "Yes" - 2 for "No".
`fmh_14`	numeric A numeric: Represents the youngest age at diagnosis of stroke in an immediate family member.

Details

This function assesses family history of premature cardiovascular disease (CVD), a significant risk factor for personal CVD development.

```
**Missing Data Codes:**
- `fmh_11`, `fmh_13`:
  - `6`: Valid skip. Handled as `haven::tagged_na("a")`.
- `7-9`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
- `fmh_12`, `fmh_14`:
  - `996`: Valid skip. Handled as `haven::tagged_na("a")`.
- `997-999`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
```

Value

integer The CVD family history:

- 1: "Yes" – Family history of premature CVD exists (diagnosis before age 60).
- 2: "No" – No family history of premature CVD.
- `haven::tagged_na("a")`: Not applicable
- `haven::tagged_na("b")`: Missing

See Also

`derive_cvd_personal_history()`

Examples

```
# Scalar usage: Single respondent
# Example 1: Premature CVD due to heart disease diagnosis at age 50
derive_cvd_family_history(fmh_11 = 1, fmh_12 = 50, fmh_13 = 2, fmh_14 = NA)
# Output: 1

# Example 2: Respondent has non-response values for all inputs.
result <- derive_cvd_family_history(fmh_11 = 8, fmh_12 = 998, fmh_13 = 8, fmh_14 = 998)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
derive_cvd_family_history(
  fmh_11 = c(1, 2, 1), fmh_12 = c(50, NA, 70),
  fmh_13 = c(2, 1, 2), fmh_14 = c(NA, 55, NA)
)
# Returns: c(1, 1, 2)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
 mutate(cvd_family_history = derive_cvd_family_history(fmh_11, fmh_12, fmh_13, fmh_14)) |>
  head()
```

---

derive_cvd_personal_history

Cardiovascular disease (CVD) personal history

Description

This function determines a respondent's cardiovascular disease (CVD) personal history based on the presence or absence of specific conditions related to heart disease, heart attack, and stroke.

Usage

```
derive_cvd_personal_history(ccc_61, ccc_63, ccc_81)
```

Arguments

ccc_61	integer An integer representing the respondent's personal history of heart disease. 1 for "Yes" if the person has heart disease, 2 for "No" if the person does not have heart disease.
ccc_63	integer An integer representing the respondent's personal history of heart attack. 1 for "Yes" if the person had a heart attack, 2 for "No" if the person did not have a heart attack.
ccc_81	integer An integer representing the respondent's personal history of stroke. 1 for "Yes" if the person had a stroke, 2 for "No" if the person did not have a stroke.

Details

This function synthesizes self-reported data on major cardiovascular events (heart disease, heart attack, stroke) into a single binary indicator.

```
**Missing Data Codes:**
- For all input variables:
  - `6`: Valid skip. Handled as `haven::tagged_na("a")`.
- `7-9`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
```

Value

integer The CVD personal history: - 1: "Yes" if the person had heart disease, heart attack, or stroke. - 2: "No" if the person had neither of the conditions. - `haven::tagged_na("a")`: Not applicable - `haven::tagged_na("b")`: Missing

See Also

`derive_cvd_family_history()`

Examples

```
# Scalar usage: Single respondent
# Determine CVD personal history for a person with heart disease (ccc_61 = 1).
derive_cvd_personal_history(ccc_61 = 1, ccc_63 = 2, ccc_81 = 2)
# Output: 1

# Example: Respondent has non-response values for all inputs.
result <- derive_cvd_personal_history(ccc_61 = 8, ccc_63 = 8, ccc_81 = 8)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
derive_cvd_personal_history(ccc_61 = c(1, 2, 2), ccc_63 = c(2, 1, 2), ccc_81 = c(2, 2, 1))
```

```
# Returns: c(1, 1, 1)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(cvd_personal_history = derive_cvd_personal_history(ccc_61, ccc_63, ccc_81)) |>
  head()
```

`derive_diabetes_status` *Diabetes derived variable*

Description

This function evaluates diabetes status using a comprehensive approach that combines laboratory measurements, self-reported diagnosis, and medication usage to create an inclusive diabetes classification.

Usage

```
derive_diabetes_status(diab_a1c, ccc_51, diab_med)
```

Arguments

`diab_a1c`	integer An integer indicating whether the respondent has diabetes based on HbA1c level. 1 for "Yes", 2 for "No".
`ccc_51`	integer An integer indicating whether the respondent self-reported diabetes. 1 for "Yes", 2 for "No".
`diab_med`	integer An integer indicating whether the respondent is on diabetes medication. 1 for "Yes", 0 for "No".

Details

This function classifies diabetes status based that considers:

```
**Data Sources:**
- Laboratory: HbA1c levels indicating diabetes (diab_a1c)
- Self-report: Participant-reported diabetes diagnosis (ccc_51)
- Medication: Current diabetes medication usage (diab_med)

**Classification Logic:**
- ANY positive indicator results in diabetes classification
- ALL negative indicators required for "no diabetes" classification
- Sophisticated missing data handling preserves available information

**Missing Data Codes:**
```

```
- `diab_a1c`, `diab_med`:
  - Tagged NA "a": Valid skip.
  - Tagged NA "b": Don't know, refusal, or not stated.
- `ccc_51`:
  - `6`: Valid skip. Handled as `haven::tagged_na("a")`.
- `7-9`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
```

Value

integer The inclusive diabetes status: - 1 ("Yes") if any of diab_a1c, ccc_51, or diab_med is 1. - 2 ("No") if all of diab_a1c, ccc_51, and diab_med are 2 or 0. - haven::tagged_na("a"): Not applicable - haven::tagged_na("b"): Missing

See Also

Related health condition functions: derive_hypertension(), calculate_gfr()

Examples

```
# Scalar usage: Single respondent
# Example: Determine the inclusive diabetes status for a respondent with diabetes based on HbA1c.
derive_diabetes_status(diab_a1c = 1, ccc_51 = 2, diab_med = 0)
# Output: 1 (Inclusive diabetes status is "Yes").

# Example: Determine the inclusive diabetes status for a respondent no diabetes all around.
derive_diabetes_status(diab_a1c = 2, ccc_51 = 2, diab_med = 0)
# Output: 2 (Inclusive diabetes status is "No").

# Example: Determine inclusive diabetes status when only one parameter is NA.
derive_diabetes_status(diab_a1c = 2, ccc_51 = NA, diab_med = 1)
# Output: 1 (Based on `diab_med`, inclusive diabetes status is "Yes").

# Example: Respondent has non-response values for all inputs.
result <- derive_diabetes_status(haven::tagged_na("b"), 8, haven::tagged_na("b"))
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
derive_diabetes_status(diab_a1c = c(1, 2, 2), ccc_51 = c(2, 1, 2), diab_med = c(0, 0, 1))
# Returns: c(1, 1, 1)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(diab_a1c = 1, diab_med = 0) |>
  mutate(diabetes_status = derive_diabetes_status(diab_a1c, ccc_51, diab_med)) |>
  head()
```

`derive_hypertension` *Hypertension derived variable*

Description

This function determines the hypertension status of a respondent based on their systolic and diastolic blood pressure measurements and medication usage.

Usage

```
derive_hypertension(
  bpmdpbps,
  bpmdpbpd,
  any_htn_med,
  ccc_32 = 2,
  cvd_status = 2,
  diab_status = 2,
  ckd_status = 2
)
```

Arguments

`bpmdpbps`	integer An integer representing the systolic blood pressure measurement of the respondent.
`bpmdpbpd`	integer An integer representing the diastolic blood pressure measurement of the respondent.
`any_htn_med`	integer An integer indicating whether the respondent is on medication for hypertension. • 1: Yes • 0: No
`ccc_32`	integer An optional integer indicating whether the respondent is actually on medication for hypertension. • 1: Yes • 2: No (default)
`cvd_status`	integer An optional integer indicating the presence of cardiovascular disease, affecting medication status. • 1: Yes • 2: No (default)
`diab_status`	integer An optional integer indicating the presence of diabetes, affecting blood pressure thresholds. • 1: Yes • 2: No (default)
`ckd_status`	integer An optional integer indicating the presence of chronic kidney disease, affecting blood pressure thresholds.

- 1: Yes
- 2: No (default)

Details

This function implements clinical guidelines for hypertension classification:

```
  **Blood Pressure Thresholds:**
  - General population: >= 140/90 mmHg indicates hypertension
- Diabetes or CKD patients: >= 130/80 mmHg indicates hypertension (lower threshold)

  **Medication Logic:**
  - Anyone taking hypertension medication is classified as hypertensive
- Medication status may be adjusted based on comorbidities (diabetes, CKD, cardiovascular disease)

  **Missing Data Codes:**
  - `bpmdpbps`, `bpmdpbpd`:
    - `996`: Valid skip. Handled as `haven::tagged_na("a")`.
  - `997-999`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
  - `any_htn_med`:
    - Tagged NA "a": Valid skip.
    - Tagged NA "b": Don't know, refusal, or not stated.
  - `ccc_32`, `cvd_status`, `diab_status`, `ckd_status`:
    - `6`: Valid skip. Handled as `haven::tagged_na("a")`.
  - `7-9`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
```

Value

integer The hypertension status:

- 1: High blood pressure (BP >= 140/90 mmHg (or >= 130/80 mmHg if diabetes or CKD) or on hypertension medication)
- 2: Normal blood pressure (BP < 140/90 mmHg (or < 130/80 mmHg if diabetes or CKD) and not on hypertension medication)
- `haven::tagged_na("a")`: Not applicable
- `haven::tagged_na("b")`: Missing

See Also

`adjust_sbp()`, `adjust_dbp()` for blood pressure adjustment, `derive_hypertension_adj()` for adjusted BP classification

Examples

```
# Scalar usage: Single respondent
# Example 1: Respondent has systolic BP = 150, diastolic BP = 95, and on medication.
derive_hypertension(bpmdpbps = 150, bpmdpbpd = 95, any_htn_med = 1)
# Output: 1 (High blood pressure due to systolic BP, diastolic BP, and medication usage).
```

```
# Example 2: Respondent has systolic BP = 120, diastolic BP = 80, and not on medication.
derive_hypertension(bpmdpbps = 120, bpmdpbpd = 80, any_htn_med = 0)
# Output: 2 (Normal blood pressure as BP is below 140/90 mmHg and not on medication).

# Example 3: Respondent has non-response BP values of 996 for both systolic and diastolic.
result <- derive_hypertension(bpmdpbps = 996, bpmdpbpd = 996, any_htn_med = 0)
result # Shows: NA
haven::is_tagged_na(result, "a") # Shows: TRUE (confirms it's tagged NA(a))
format(result, tag = TRUE) # Shows: "NA(a)" (displays the tag)

# Multiple respondents
derive_hypertension(
  bpmdpbps = c(150, 120, 135), bpmdpbpd = c(95, 80, 85),
  any_htn_med = c(1, 0, 1), diab_status = c(2, 2, 1)
)
# Returns: c(1, 2, 1)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(any_htn_med = 0) |>
  mutate(htn = derive_hypertension(bpmdpbps, bpmdpbpd, any_htn_med)) |>
  select(clinicid, bpmdpbps, bpmdpbpd, htn) |>
  head()
```

`derive_hypertension_adj` *Hypertension derived variable with adjusted blood pressures*

Description

This function determines the hypertension status of a respondent based on their adjusted systolic and diastolic blood pressure measurements and medication usage.

Usage

```
derive_hypertension_adj(
  sbp_adj_mmhg,
  dbp_adj_mmhg,
  any_htn_med,
  ccc_32 = 2,
  cvd_status = 2,
  diab_status = 2,
  ckd_status = 2
)
```

Arguments

sbp_adj_mmhg	integer An integer representing the adjusted systolic blood pressure measurement of the respondent.
dbp_adj_mmhg	integer An integer representing the adjusted diastolic blood pressure measurement of the respondent.
any_htn_med	integer An integer indicating whether the respondent is on medication for hypertension. • 1: Yes • 0: No
ccc_32	integer An optional integer indicating whether the respondent is actually on medication for hypertension. • 1: Yes • 2: No (default)
cvd_status	integer An optional integer indicating the presence of cardiovascular disease, affecting medication status. • 1: Yes • 2: No (default)
diab_status	integer An optional integer indicating the presence of diabetes, affecting blood pressure thresholds. • 1: Yes • 2: No (default)
ckd_status	integer An optional integer indicating the presence of chronic kidney disease, affecting blood pressure thresholds. • 1: Yes • 2: No (default)

Details

This function implements clinical guidelines for hypertension classification using adjusted blood pressure values:

```
  **Blood Pressure Thresholds:**
  - General population: >= 140/90 mmHg indicates hypertension
- Diabetes or CKD patients: >= 130/80 mmHg indicates hypertension (lower threshold)

  **Medication Logic:**
  - Anyone taking hypertension medication is classified as hypertensive
- Medication status may be adjusted based on comorbidities (diabetes, CKD, cardiovascular disease)

  **Missing Data Codes:**
  - `sbp_adj_mmhg`, `dbp_adj_mmhg`:
    - `996`: Valid skip. Handled as `haven::tagged_na("a")`.
  - `997-999`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
  - `any_htn_med`:
```

```
    - Tagged NA "a": Valid skip.
    - Tagged NA "b": Don't know, refusal, or not stated.
  - `ccc_32`, `cvd_status`, `diab_status`, `ckd_status`:
    - `6`: Valid skip. Handled as `haven::tagged_na("a")`.
  - `7-9`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
```

Value

integer The hypertension status:

- 1: High blood pressure (adjusted BP >= 140/90 mmHg (or >= 130/80 mmHg if diabetes or CKD) or on hypertension medication)
- 2: Normal blood pressure (adjusted BP < 140/90 mmHg (or < 130/80 mmHg if diabetes or CKD) and not on hypertension medication)
- `haven::tagged_na("a")`: Not applicable
- `haven::tagged_na("b")`: Missing

See Also

`derive_hypertension()` for unadjusted BP classification

Examples

```
# Scalar usage: Single respondent
# Example 1: Respondent has adjusted SBP = 150, adjusted DBP = 95, and on medication.
derive_hypertension_adj(sbp_adj_mmhg = 150, dbp_adj_mmhg = 95, any_htn_med = 1)
# Output: 1 (High blood pressure due to adjusted SBP, adjusted DBP, and medication usage).

# Example 2: Respondent has adjusted SBP = 120, adjusted DBP = 80, and not on medication.
derive_hypertension_adj(sbp_adj_mmhg = 120, dbp_adj_mmhg = 80, any_htn_med = 2)
# Output: 2 (Normal blood pressure as adjusted BP is below 140/90 mmHg and not on medication).

# Example 3: Respondent has non-response BP values of 996 for both systolic and diastolic.
result <- derive_hypertension_adj(sbp_adj_mmhg = 996, dbp_adj_mmhg = 996, any_htn_med = 0)
result # Shows: NA
haven::is_tagged_na(result, "a") # Shows: TRUE (confirms it's tagged NA(a))
format(result, tag = TRUE) # Shows: "NA(a)" (displays the tag)

# Multiple respondents
derive_hypertension_adj(
  sbp_adj_mmhg = c(150, 120, 135), dbp_adj_mmhg = c(95, 80, 85),
  any_htn_med = c(1, 0, 1), diab_status = c(2, 2, 1)
)
# Returns: c(1, 2, 1)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(sbp_adj_mmhg = adjust_sbp(bpmdpbps), dbp_adj_mmhg = adjust_dbp(bpmdpbpd)) |>
  mutate(any_htn_med = 0) |>
  mutate(htn_adj = derive_hypertension_adj(sbp_adj_mmhg, dbp_adj_mmhg, any_htn_med)) |>
```

```
  select(clinicid, sbp_adj_mmhg, dbp_adj_mmhg, htn_adj) |>
  head()
```

`derive_hypertension_control` *Controlled hypertension derived variable*

Description

This function determines the controlled hypertension status of a respondent based on their systolic and diastolic blood pressure measurements and medication usage.

Usage

```
derive_hypertension_control(
  bpmdpbps,
  bpmdpbpd,
  any_htn_med,
  ccc_32 = 2,
  cvd_status = 2,
  diab_status = 2,
  ckd_status = 2
)
```

Arguments

`bpmdpbps`	integer An integer representing the systolic blood pressure measurement of the respondent.
`bpmdpbpd`	integer An integer representing the diastolic blood pressure measurement of the respondent.
`any_htn_med`	integer An integer indicating whether the respondent is on medication for hypertension. • 1: Yes • 0: No
`ccc_32`	integer An optional integer indicating whether the respondent is actually on medication for hypertension. • 1: Yes • 2: No (default)
`cvd_status`	integer An optional integer indicating the presence of cardiovascular disease, affecting medication status. • 1: Yes • 2: No (default)
`diab_status`	integer An optional integer indicating the presence of diabetes, affecting blood pressure thresholds.

- 1: Yes
- 2: No (default)

`ckd_status` integer An optional integer indicating the presence of chronic kidney disease, affecting blood pressure thresholds.

- 1: Yes
- 2: No (default)

Details

This function assesses whether a respondent's hypertension is controlled:

```
**Control Thresholds:**
- General population: < 140/90 mmHg
- Diabetes or CKD patients: < 130/80 mmHg

**Logic:**
- Only applies to respondents taking hypertension medication.
- If BP is below the threshold, hypertension is "controlled" (1).
- If BP is at or above the threshold, it is "not controlled" (2).

**Missing Data Codes:**
- `bpmdpbps`, `bpmdpbpd`:
  - `996`: Valid skip. Handled as `haven::tagged_na("a")`.
- `997-999`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
- `any_htn_med`:
  - Tagged NA "a": Valid skip.
  - Tagged NA "b": Don't know, refusal, or not stated.
- `ccc_32`, `cvd_status`, `diab_status`, `ckd_status`:
  - `6`: Valid skip. Handled as `haven::tagged_na("a")`.
- `7-9`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
```

Value

integer The hypertension status:

- 1: Hypertension controlled (BP < 140/90 mmHg (or < 130/80 mmHg if diabetes or CKD) when on hypertension medication)
- 2: Hypertension not controlled (BP >= 140/90 mmHg (or >= 130/80 mmHg if diabetes or CKD) when on hypertension medication)
- `haven::tagged_na("a")`: Not applicable
- `haven::tagged_na("b")`: Missing

See Also

`derive_hypertension_control_adj()` for controlled status with adjusted BP

Examples

```
# Scalar usage: Single respondent
# Example 1: Respondent has systolic BP = 150, diastolic BP = 95, and on medication.
derive_hypertension_control(bpmdpbps = 150, bpmdpbpd = 95, any_htn_med = 1)
# Output: 2 (Hypertension not controlled due to high SBP and SBP despite medication usage).

# Example 2: Respondent has systolic BP = 120, diastolic BP = 80, and on medication.
derive_hypertension_control(bpmdpbps = 120, bpmdpbpd = 80, any_htn_med = 1)
# Output: 1 (Hypertension controlled as BP is below 140/90 mmHg and on medication).

# Example 3: Respondent has non-response BP values of 996 for both systolic and diastolic.
result <- derive_hypertension_control(bpmdpbps = 996, bpmdpbpd = 996, any_htn_med = 0)
result # Shows: NA
haven::is_tagged_na(result, "a") # Shows: TRUE (confirms it's tagged NA(a))
format(result, tag = TRUE) # Shows: "NA(a)" (displays the tag)

# Multiple respondents
derive_hypertension_control(
  bpmdpbps = c(150, 120, 135), bpmdpbpd = c(95, 80, 85),
  any_htn_med = c(1, 1, 1), diab_status = c(2, 2, 1)
)
# Returns: c(2, 1, 2)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(any_htn_med = 1) |>
  mutate(ctrl = derive_hypertension_control(bpmdpbps, bpmdpbpd, any_htn_med)) |>
  select(clinicid, bpmdpbps, bpmdpbpd, ctrl) |>
  head()
```

---

derive_hypertension_control_adj

Controlled hypertension derived variable with adjusted blood pressures

---

Description

This function determines the controlled hypertension status of a respondent based on their adjusted systolic and diastolic blood pressure measurements and medication usage.

Usage

```
derive_hypertension_control_adj(
  sbp_adj_mmhg,
  dbp_adj_mmhg,
  any_htn_med,
  ccc_32 = 2,
```

```
  cvd_status = 2,
  diab_status = 2,
  ckd_status = 2
)
```

Arguments

sbp_adj_mmhg	integer An integer representing the adjusted systolic blood pressure measurement of the respondent.
dbp_adj_mmhg	integer An integer representing the adjusted diastolic blood pressure measurement of the respondent.
any_htn_med	integer An integer indicating whether the respondent is on medication for hypertension. • 1: Yes • 0: No
ccc_32	integer An optional integer indicating whether the respondent is actually on medication for hypertension. • 1: Yes • 2: No (default)
cvd_status	integer An optional integer indicating the presence of cardiovascular disease, affecting medication status. • 1: Yes • 2: No (default)
diab_status	integer An optional integer indicating the presence of diabetes, affecting blood pressure thresholds. • 1: Yes • 2: No (default)
ckd_status	integer An optional integer indicating the presence of chronic kidney disease, affecting blood pressure thresholds. • 1: Yes • 2: No (default)

Details

This function assesses whether a respondent's hypertension is controlled using adjusted BP values:

```
**Control Thresholds:**
- General population: < 140/90 mmHg
- Diabetes or CKD patients: < 130/80 mmHg

**Logic:**
- Only applies to respondents taking hypertension medication.
- If adjusted BP is below the threshold, hypertension is "controlled" (1).
- If adjusted BP is at or above the threshold, it is "not controlled" (2).
```

```
**Missing Data Codes:**
- `sbp_adj_mmhg`, `dbp_adj_mmhg`:
  - `996`: Valid skip. Handled as `haven::tagged_na("a")`.
- `997-999`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
- `any_htn_med`:
  - Tagged NA "a": Valid skip.
  - Tagged NA "b": Don't know, refusal, or not stated.
- `ccc_32`, `cvd_status`, `diab_status`, `ckd_status`:
  - `6`: Valid skip. Handled as `haven::tagged_na("a")`.
- `7-9`: Don't know, refusal, or not stated. Handled as `haven::tagged_na("b")`.
```

Value

integer The hypertension status:

- 1: Hypertension controlled (BP < 140/90 mmHg (or < 130/80 mmHg if diabetes or CKD) when on hypertension medication)
- 2: Hypertension not controlled (BP >= 140/90 mmHg (or >= 130/80 mmHg if diabetes or CKD) when on hypertension medication)
- `haven::tagged_na("a")`: Not applicable
- `haven::tagged_na("b")`: Missing

See Also

`derive_hypertension_control()` for controlled status with unadjusted BP

Examples

```
# Scalar usage: Single respondent
# Example 1: Respondent has adjusted SBP = 150, adjusted DBP = 95, and on medication.
derive_hypertension_control_adj(sbp_adj_mmhg = 150, dbp_adj_mmhg = 95, any_htn_med = 1)
# Output: 2 (Hypertension not controlled due to high adjusted SBP and DBP despite medication usage).

# Example 2: Respondent has adjusted SBP = 120, adjusted DBP = 80, and on medication.
derive_hypertension_control_adj(sbp_adj_mmhg = 120, dbp_adj_mmhg = 80, any_htn_med = 1)
# Output: 1 (Hypertension controlled as adjusted BP is below 140/90 mmHg and on medication).

# Example 3: Respondent has non-response BP values of 996 for both systolic and diastolic.
result <- derive_hypertension_control_adj(sbp_adj_mmhg = 996, dbp_adj_mmhg = 996, any_htn_med = 0)
result # Shows: NA
haven::is_tagged_na(result, "a") # Shows: TRUE (confirms it's tagged NA(a))
format(result, tag = TRUE) # Shows: "NA(a)" (displays the tag)

# Multiple respondents
derive_hypertension_control_adj(
  sbp_adj_mmhg = c(150, 120, 135), dbp_adj_mmhg = c(95, 80, 85),
  any_htn_med = c(1, 1, 1), diab_status = c(2, 2, 1)
)
# Returns: c(2, 1, 2)
```

```
# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(sbp_adj_mmhg = adjust_sbp(bpmdpbps), dbp_adj_mmhg = adjust_dbp(bpmdpbpd)) |>
  mutate(any_htn_med = 1) |>
 mutate(ctrl_adj = derive_hypertension_control_adj(sbp_adj_mmhg, dbp_adj_mmhg, any_htn_med)) |>
  select(clinicid, sbp_adj_mmhg, dbp_adj_mmhg, ctrl_adj) |>
  head()
```

`is_ace_inhibitor` *ACE inhibitors*

Description

This function checks if a given medication is an ACE inhibitor. This function processes multiple inputs efficiently.

Usage

```
is_ace_inhibitor(meucatc, npi_25b)
```

Arguments

`meucatc`	character ATC code of the medication.
`npi_25b`	integer Time when the medication was last taken.

Details

Identifies ACE inhibitors based on ATC codes starting with "C09".

```
**Missing Data Codes:**
- `meucatc`: `9999996` (Not applicable), `9999997-9999999` (Missing)
- `npi_25b`: `6` (Not applicable), `7-9` (Missing)
```

Value

numeric 1 if medication is an ACE inhibitor, 0 otherwise. If inputs are invalid or out of bounds, the function returns a tagged NA.

Examples

```
# Scalar usage: Single respondent
is_ace_inhibitor("C09AB03", 2)
# Returns: 1

# Example: Respondent has non-response values for all inputs.
result <- is_ace_inhibitor("9999998", 8)
result # Shows: NA
```

```
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
is_ace_inhibitor(c("C09AB03", "C01AA05"), c(2, 1))
# Returns: c(1, 0)

# Database usage: Applied to survey datasets
library(dplyr)
cycle3_meds |>
  mutate(ace_inhibitor = is_ace_inhibitor(meucatc, npi_25b)) |>
  head()
```

---

is_ace_med_cycles1to2 *ACE inhibitors - cycles 1-2*

Description

This function checks if a person is taking ACE inhibitors based on the provided Anatomical Therapeutic Chemical (ATC) codes for medications and the Canadian Health Measures Survey (CHMS) response for the time when the medication was last taken.

Usage

```
is_ace_med_cycles1to2(
  atc_101a = NULL,
  atc_102a = NULL,
  atc_103a = NULL,
  atc_104a = NULL,
  atc_105a = NULL,
  atc_106a = NULL,
  atc_107a = NULL,
  atc_108a = NULL,
  atc_109a = NULL,
  atc_110a = NULL,
  atc_111a = NULL,
  atc_112a = NULL,
  atc_113a = NULL,
  atc_114a = NULL,
  atc_115a = NULL,
  atc_201a = NULL,
  atc_202a = NULL,
  atc_203a = NULL,
  atc_204a = NULL,
  atc_205a = NULL,
  atc_206a = NULL,
  atc_207a = NULL,
```

```
  atc_208a = NULL,
  atc_209a = NULL,
  atc_210a = NULL,
  atc_211a = NULL,
  atc_212a = NULL,
  atc_213a = NULL,
  atc_214a = NULL,
  atc_215a = NULL,
  atc_131a = NULL,
  atc_132a = NULL,
  atc_133a = NULL,
  atc_134a = NULL,
  atc_135a = NULL,
  atc_231a = NULL,
  atc_232a = NULL,
  atc_233a = NULL,
  atc_234a = NULL,
  atc_235a = NULL,
  mhr_101b = NULL,
  mhr_102b = NULL,
  mhr_103b = NULL,
  mhr_104b = NULL,
  mhr_105b = NULL,
  mhr_106b = NULL,
  mhr_107b = NULL,
  mhr_108b = NULL,
  mhr_109b = NULL,
  mhr_110b = NULL,
  mhr_111b = NULL,
  mhr_112b = NULL,
  mhr_113b = NULL,
  mhr_114b = NULL,
  mhr_115b = NULL,
  mhr_201b = NULL,
  mhr_202b = NULL,
  mhr_203b = NULL,
  mhr_204b = NULL,
  mhr_205b = NULL,
  mhr_206b = NULL,
  mhr_207b = NULL,
  mhr_208b = NULL,
  mhr_209b = NULL,
  mhr_210b = NULL,
  mhr_211b = NULL,
  mhr_212b = NULL,
  mhr_213b = NULL,
  mhr_214b = NULL,
  mhr_215b = NULL,
```

```
  mhr_131b = NULL,
  mhr_132b = NULL,
  mhr_133b = NULL,
  mhr_134b = NULL,
  mhr_135b = NULL,
  mhr_231b = NULL,
  mhr_232b = NULL,
  mhr_233b = NULL,
  mhr_234b = NULL,
  mhr_235b = NULL
)
```

Arguments

atc_101a	character ATC code of respondent's first prescription medication.
atc_102a	character ATC code of respondent's second prescription medication.
atc_103a	character ATC code of respondent's third prescription medication.
atc_104a	character ATC code of respondent's fourth prescription medication.
atc_105a	character ATC code of respondent's fifth prescription medication.
atc_106a	character ATC code of respondent's sixth prescription medication.
atc_107a	character ATC code of respondent's seventh prescription medication.
atc_108a	character ATC code of respondent's eighth prescription medication.
atc_109a	character ATC code of respondent's ninth prescription medication.
atc_110a	character ATC code of respondent's tenth prescription medication.
atc_111a	character ATC code of respondent's eleventh prescription medication.
atc_112a	character ATC code of respondent's twelfth prescription medication.
atc_113a	character ATC code of respondent's thirteenth prescription medication.
atc_114a	character ATC code of respondent's fourteenth prescription medication.
atc_115a	character ATC code of respondent's fifteenth prescription medication.
atc_201a	character ATC code of respondent's first over-the-counter medication.
atc_202a	character ATC code of respondent's second over-the-counter medication.
atc_203a	character ATC code of respondent's third over-the-counter medication.
atc_204a	character ATC code of respondent's fourth over-the-counter medication.
atc_205a	character ATC code of respondent's fifth over-the-counter medication.
atc_206a	character ATC code of respondent's sixth over-the-counter medication.
atc_207a	character ATC code of respondent's seventh over-the-counter medication.
atc_208a	character ATC code of respondent's eighth over-the-counter medication.
atc_209a	character ATC code of respondent's ninth over-the-counter medication.
atc_210a	character ATC code of respondent's tenth over-the-counter medication.
atc_211a	character ATC code of respondent's eleventh over-the-counter medication.
atc_212a	character ATC code of respondent's twelfth over-the-counter medication.

atc_213a character ATC code of respondent's thirteenth over-the-counter medication.

atc_214a character ATC code of respondent's fourteenth over-the-counter medication.

atc_215a character ATC code of respondent's fifteenth over-the-counter medication.

atc_131a character ATC code of respondent's first new prescription medication.

atc_132a character ATC code of respondent's second new prescription medication.

atc_133a character ATC code of respondent's third new prescription medication.

atc_134a character ATC code of respondent's fourth new prescription medication.

atc_135a character ATC code of respondent's fifth new prescription medication.

atc_231a character ATC code of respondent's first new over-the-counter medication.

atc_232a character ATC code of respondent's second new over-the-counter medication.

atc_233a character ATC code of respondent's third new over-the-counter medication.

atc_234a character ATC code of respondent's fourth new over-the-counter medication.

atc_235a character ATC code of respondent's fifth new over-the-counter medication.

mhr_101b integer Response for when the first prescription medication was last taken (1 = Today, . . . , 6 = Never).

mhr_102b integer Response for when the second prescription medication was last taken (1-6).

mhr_103b integer Response for when the third prescription medication was last taken (1-6).

mhr_104b integer Response for when the fourth prescription medication was last taken (1-6).

mhr_105b integer Response for when the fifth prescription medication was last taken (1-6).

mhr_106b integer Response for when the sixth prescription medication was last taken (1-6).

mhr_107b integer Response for when the seventh prescription medication was last taken (1-6).

mhr_108b integer Response for when the eighth prescription medication was last taken (1-6).

mhr_109b integer Response for when the ninth prescription medication was last taken (1-6).

mhr_110b integer Response for when the tenth prescription medication was last taken (1-6).

mhr_111b integer Response for when the eleventh prescription medication was last taken (1-6).

mhr_112b integer Response for when the twelfth prescription medication was last taken (1-6).

mhr_113b integer Response for when the thirteenth prescription medication was last taken (1-6).

mhr_114b integer Response for when the fourteenth prescription medication was last taken (1-6).

mhr_115b integer Response for when the fifteenth prescription medication was last taken (1-6).

mhr_201b integer Response for when the first over-the-counter medication was last taken (1-6).

mhr_202b integer Response for when the second over-the-counter medication was last taken (1-6).

mhr_203b integer Response for when the third over-the-counter medication was last taken (1-6).

mhr_204b integer Response for when the fourth over-the-counter medication was last taken (1-6).

mhr_205b integer Response for when the fifth over-the-counter medication was last taken (1-6).

mhr_206b integer Response for when the sixth over-the-counter medication was last taken (1-6).

mhr_207b integer Response for when the seventh over-the-counter medication was last taken (1-6).

mhr_208b integer Response for when the eighth over-the-counter medication was last taken (1-6).

mhr_209b integer Response for when the ninth over-the-counter medication was last taken (1-6).

mhr_210b integer Response for when the tenth over-the-counter medication was last taken (1-6).

mhr_211b integer Response for when the eleventh over-the-counter medication was last taken (1-6).

mhr_212b integer Response for when the twelfth over-the-counter medication was last taken (1-6).

mhr_213b integer Response for when the thirteenth over-the-counter medication was last taken (1-6).

mhr_214b integer Response for when the fourteenth over-the-counter medication was last taken (1-6).

mhr_215b integer Response for when the fifteenth over-the-counter medication was last taken (1-6).

mhr_131b integer Response for when the first new prescription medication was last taken (1-6).

mhr_132b integer Response for when the second new prescription medication was last taken (1-6).

mhr_133b integer Response for when the third new prescription medication was last taken (1-6).

mhr_134b integer Response for when the fourth new prescription medication was last taken (1-6).

mhr_135b integer Response for when the fifth new prescription medication was last taken (1-6).

mhr_231b integer Response for when the first new over-the-counter medication was last taken (1-6).

mhr_232b integer Response for when the second new over-the-counter medication was last taken (1-6).

mhr_233b integer Response for when the third new over-the-counter medication was last taken (1-6).

mhr_234b integer Response for when the fourth new over-the-counter medication was last taken (1-6).

mhr_235b integer Response for when the fifth new over-the-counter medication was last taken (1-6).

Details

The function identifies ACE inhibitors based on ATC codes starting with "C09". It checks all medication variables provided in the input data frame.

```
  **Missing Data Codes:**
- The function handles tagged NAs from the `is_ace_inhibitor` function and propagates them.
```

Value

numeric Returns 1 if the person is taking ACE inhibitors, 0 otherwise. If all medication information is missing, it returns a tagged NA.

See Also

is_ace_inhibitor

Examples

```
# This is a wrapper function and is not intended to be called directly by the user.
# See `is_ace_inhibitor` for usage examples.
```

---

is_any_antihtn_med *Any anti-hypertensive medications*

Description

This function checks if a given medication is any anti-hypertensive drug. This function processes multiple inputs efficiently.

Usage

```
is_any_antihtn_med(meucatc, npi_25b)
```

Arguments

meucatc character ATC code of the medication.

npi_25b integer Time when the medication was last taken.

Details

Identifies anti-hypertensive drugs based on ATC codes starting with "C02", "C03", "C07", "C08", or "C09", excluding specific sub-codes.

```
**Missing Data Codes:**
- `meucatc`: `9999996` (Not applicable), `9999997-9999999` (Missing)
- `npi_25b`: `6` (Not applicable), `7-9` (Missing)
```

Value

numeric 1 if medication is an anti-hypertensive drug, 0 otherwise. If inputs are invalid or out of bounds, the function returns a tagged NA.

Examples

```
# Scalar usage: Single respondent
is_any_antihtn_med("C07AB02", 4)
# Returns: 1

# Example: Respondent has non-response values for all inputs.
result <- is_any_antihtn_med("9999998", 8)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
is_any_antihtn_med(c("C07AB02", "C07AA07"), c(4, 2))
# Returns: c(1, 0)

# Database usage: Applied to survey datasets
library(dplyr)
cycle3_meds |>
  mutate(any_antihtn = is_any_antihtn_med(meucatc, npi_25b)) |>
  head()
```

---

is_any_htn_med_cycles1to2

Any anti-hypertensive medications - cycles 1-2

Description

This function checks if a person is taking any anti-hypertensive medication based on the provided Anatomical Therapeutic Chemical (ATC) codes for medications and the Canadian Health Measures Survey (CHMS) response for the time when the medication was last taken.

Usage

```
is_any_htn_med_cycles1to2(
  atc_101a = NULL,
  atc_102a = NULL,
  atc_103a = NULL,
  atc_104a = NULL,
  atc_105a = NULL,
  atc_106a = NULL,
  atc_107a = NULL,
  atc_108a = NULL,
  atc_109a = NULL,
  atc_110a = NULL,
  atc_111a = NULL,
  atc_112a = NULL,
  atc_113a = NULL,
  atc_114a = NULL,
  atc_115a = NULL,
  atc_201a = NULL,
  atc_202a = NULL,
  atc_203a = NULL,
  atc_204a = NULL,
  atc_205a = NULL,
  atc_206a = NULL,
  atc_207a = NULL,
  atc_208a = NULL,
  atc_209a = NULL,
  atc_210a = NULL,
  atc_211a = NULL,
  atc_212a = NULL,
  atc_213a = NULL,
  atc_214a = NULL,
  atc_215a = NULL,
  atc_131a = NULL,
  atc_132a = NULL,
  atc_133a = NULL,
  atc_134a = NULL,
  atc_135a = NULL,
  atc_231a = NULL,
  atc_232a = NULL,
  atc_233a = NULL,
  atc_234a = NULL,
  atc_235a = NULL,
  mhr_101b = NULL,
  mhr_102b = NULL,
  mhr_103b = NULL,
  mhr_104b = NULL,
  mhr_105b = NULL,
  mhr_106b = NULL,
```

```
  mhr_107b = NULL,
  mhr_108b = NULL,
  mhr_109b = NULL,
  mhr_110b = NULL,
  mhr_111b = NULL,
  mhr_112b = NULL,
  mhr_113b = NULL,
  mhr_114b = NULL,
  mhr_115b = NULL,
  mhr_201b = NULL,
  mhr_202b = NULL,
  mhr_203b = NULL,
  mhr_204b = NULL,
  mhr_205b = NULL,
  mhr_206b = NULL,
  mhr_207b = NULL,
  mhr_208b = NULL,
  mhr_209b = NULL,
  mhr_210b = NULL,
  mhr_211b = NULL,
  mhr_212b = NULL,
  mhr_213b = NULL,
  mhr_214b = NULL,
  mhr_215b = NULL,
  mhr_131b = NULL,
  mhr_132b = NULL,
  mhr_133b = NULL,
  mhr_134b = NULL,
  mhr_135b = NULL,
  mhr_231b = NULL,
  mhr_232b = NULL,
  mhr_233b = NULL,
  mhr_234b = NULL,
  mhr_235b = NULL
)
```

Arguments

atc_101a	character ATC code of respondent's first prescription medication.
atc_102a	character ATC code of respondent's second prescription medication.
atc_103a	character ATC code of respondent's third prescription medication.
atc_104a	character ATC code of respondent's fourth prescription medication.
atc_105a	character ATC code of respondent's fifth prescription medication.
atc_106a	character ATC code of respondent's sixth prescription medication.
atc_107a	character ATC code of respondent's seventh prescription medication.
atc_108a	character ATC code of respondent's eighth prescription medication.

`atc_109a` character ATC code of respondent's ninth prescription medication.

`atc_110a` character ATC code of respondent's tenth prescription medication.

`atc_111a` character ATC code of respondent's eleventh prescription medication.

`atc_112a` character ATC code of respondent's twelfth prescription medication.

`atc_113a` character ATC code of respondent's thirteenth prescription medication.

`atc_114a` character ATC code of respondent's fourteenth prescription medication.

`atc_115a` character ATC code of respondent's fifteenth prescription medication.

`atc_201a` character ATC code of respondent's first over-the-counter medication.

`atc_202a` character ATC code of respondent's second over-the-counter medication.

`atc_203a` character ATC code of respondent's third over-the-counter medication.

`atc_204a` character ATC code of respondent's fourth over-the-counter medication.

`atc_205a` character ATC code of respondent's fifth over-the-counter medication.

`atc_206a` character ATC code of respondent's sixth over-the-counter medication.

`atc_207a` character ATC code of respondent's seventh over-the-counter medication.

`atc_208a` character ATC code of respondent's eighth over-the-counter medication.

`atc_209a` character ATC code of respondent's ninth over-the-counter medication.

`atc_210a` character ATC code of respondent's tenth over-the-counter medication.

`atc_211a` character ATC code of respondent's eleventh over-the-counter medication.

`atc_212a` character ATC code of respondent's twelfth over-the-counter medication.

`atc_213a` character ATC code of respondent's thirteenth over-the-counter medication.

`atc_214a` character ATC code of respondent's fourteenth over-the-counter medication.

`atc_215a` character ATC code of respondent's fifteenth over-the-counter medication.

`atc_131a` character ATC code of respondent's first new prescription medication.

`atc_132a` character ATC code of respondent's second new prescription medication.

`atc_133a` character ATC code of respondent's third new prescription medication.

`atc_134a` character ATC code of respondent's fourth new prescription medication.

`atc_135a` character ATC code of respondent's fifth new prescription medication.

`atc_231a` character ATC code of respondent's first new over-the-counter medication.

`atc_232a` character ATC code of respondent's second new over-the-counter medication.

`atc_233a` character ATC code of respondent's third new over-the-counter medication.

`atc_234a` character ATC code of respondent's fourth new over-the-counter medication.

`atc_235a` character ATC code of respondent's fifth new over-the-counter medication.

`mhr_101b` integer Response for when the first prescription medication was last taken (1 = Today, . . . , 6 = Never).

`mhr_102b` integer Response for when the second prescription medication was last taken (1-6).

`mhr_103b` integer Response for when the third prescription medication was last taken (1-6).

mhr_104b integer Response for when the fourth prescription medication was last taken (1-6).

mhr_105b integer Response for when the fifth prescription medication was last taken (1-6).

mhr_106b integer Response for when the sixth prescription medication was last taken (1-6).

mhr_107b integer Response for when the seventh prescription medication was last taken (1-6).

mhr_108b integer Response for when the eighth prescription medication was last taken (1-6).

mhr_109b integer Response for when the ninth prescription medication was last taken (1-6).

mhr_110b integer Response for when the tenth prescription medication was last taken (1-6).

mhr_111b integer Response for when the eleventh prescription medication was last taken (1-6).

mhr_112b integer Response for when the twelfth prescription medication was last taken (1-6).

mhr_113b integer Response for when the thirteenth prescription medication was last taken (1-6).

mhr_114b integer Response for when the fourteenth prescription medication was last taken (1-6).

mhr_115b integer Response for when the fifteenth prescription medication was last taken (1-6).

mhr_201b integer Response for when the first over-the-counter medication was last taken (1-6).

mhr_202b integer Response for when the second over-the-counter medication was last taken (1-6).

mhr_203b integer Response for when the third over-the-counter medication was last taken (1-6).

mhr_204b integer Response for when the fourth over-the-counter medication was last taken (1-6).

mhr_205b integer Response for when the fifth over-the-counter medication was last taken (1-6).

mhr_206b integer Response for when the sixth over-the-counter medication was last taken (1-6).

mhr_207b integer Response for when the seventh over-the-counter medication was last taken (1-6).

mhr_208b integer Response for when the eighth over-the-counter medication was last taken (1-6).

mhr_209b integer Response for when the ninth over-the-counter medication was last taken (1-6).

mhr_210b integer Response for when the tenth over-the-counter medication was last taken (1-6).

mhr_211b integer Response for when the eleventh over-the-counter medication was last taken (1-6).

mhr_212b integer Response for when the twelfth over-the-counter medication was last taken (1-6).

mhr_213b integer Response for when the thirteenth over-the-counter medication was last taken (1-6).

mhr_214b integer Response for when the fourteenth over-the-counter medication was last taken (1-6).

mhr_215b integer Response for when the fifteenth over-the-counter medication was last taken (1-6).

mhr_131b integer Response for when the first new prescription medication was last taken (1-6).

mhr_132b integer Response for when the second new prescription medication was last taken (1-6).

mhr_133b integer Response for when the third new prescription medication was last taken (1-6).

mhr_134b integer Response for when the fourth new prescription medication was last taken (1-6).

mhr_135b integer Response for when the fifth new prescription medication was last taken (1-6).

mhr_231b integer Response for when the first new over-the-counter medication was last taken (1-6).

mhr_232b integer Response for when the second new over-the-counter medication was last taken (1-6).

mhr_233b integer Response for when the third new over-the-counter medication was last taken (1-6).

mhr_234b integer Response for when the fourth new over-the-counter medication was last taken (1-6).

mhr_235b integer Response for when the fifth new over-the-counter medication was last taken (1-6).

Details

The function identifies anti-hypertensive drugs based on ATC codes starting with "C02", "C03", "C07", "C08", or "C09", excluding specific sub-codes. It checks all medication variables provided in the input data frame.

```
  **Missing Data Codes:**
- The function handles tagged NAs from the `is_any_antihtn_med` function and propagates them.
```

Value

numeric Returns 1 if the person is taking any anti-hypertensive medication, 0 otherwise. If all medication information is missing, it returns a tagged NA.

See Also

is_any_antihtn_med

Examples

```
# This is a wrapper function and is not intended to be called directly by the user.
# See `is_any_antihtn_med` for usage examples.
```

---

is_bb_med_cycles1to2 *Beta blockers - cycles 1-2*

Description

This function checks if a person is taking beta blockers based on the provided Anatomical Therapeutic Chemical (ATC) codes for medications and the Canadian Health Measures Survey (CHMS) response for the time when the medication was last taken.

Usage

```
is_bb_med_cycles1to2(
  atc_101a = NULL,
  atc_102a = NULL,
  atc_103a = NULL,
  atc_104a = NULL,
  atc_105a = NULL,
  atc_106a = NULL,
  atc_107a = NULL,
  atc_108a = NULL,
  atc_109a = NULL,
  atc_110a = NULL,
  atc_111a = NULL,
  atc_112a = NULL,
  atc_113a = NULL,
  atc_114a = NULL,
  atc_115a = NULL,
  atc_201a = NULL,
  atc_202a = NULL,
  atc_203a = NULL,
  atc_204a = NULL,
  atc_205a = NULL,
  atc_206a = NULL,
  atc_207a = NULL,
  atc_208a = NULL,
  atc_209a = NULL,
  atc_210a = NULL,
  atc_211a = NULL,
  atc_212a = NULL,
```

```
  atc_213a = NULL,
  atc_214a = NULL,
  atc_215a = NULL,
  atc_131a = NULL,
  atc_132a = NULL,
  atc_133a = NULL,
  atc_134a = NULL,
  atc_135a = NULL,
  atc_231a = NULL,
  atc_232a = NULL,
  atc_233a = NULL,
  atc_234a = NULL,
  atc_235a = NULL,
  mhr_101b = NULL,
  mhr_102b = NULL,
  mhr_103b = NULL,
  mhr_104b = NULL,
  mhr_105b = NULL,
  mhr_106b = NULL,
  mhr_107b = NULL,
  mhr_108b = NULL,
  mhr_109b = NULL,
  mhr_110b = NULL,
  mhr_111b = NULL,
  mhr_112b = NULL,
  mhr_113b = NULL,
  mhr_114b = NULL,
  mhr_115b = NULL,
  mhr_201b = NULL,
  mhr_202b = NULL,
  mhr_203b = NULL,
  mhr_204b = NULL,
  mhr_205b = NULL,
  mhr_206b = NULL,
  mhr_207b = NULL,
  mhr_208b = NULL,
  mhr_209b = NULL,
  mhr_210b = NULL,
  mhr_211b = NULL,
  mhr_212b = NULL,
  mhr_213b = NULL,
  mhr_214b = NULL,
  mhr_215b = NULL,
  mhr_131b = NULL,
  mhr_132b = NULL,
  mhr_133b = NULL,
  mhr_134b = NULL,
  mhr_135b = NULL,
```

```
  mhr_231b = NULL,
  mhr_232b = NULL,
  mhr_233b = NULL,
  mhr_234b = NULL,
  mhr_235b = NULL
)
```

Arguments

atc_101a	character ATC code of respondent's first prescription medication.
atc_102a	character ATC code of respondent's second prescription medication.
atc_103a	character ATC code of respondent's third prescription medication.
atc_104a	character ATC code of respondent's fourth prescription medication.
atc_105a	character ATC code of respondent's fifth prescription medication.
atc_106a	character ATC code of respondent's sixth prescription medication.
atc_107a	character ATC code of respondent's seventh prescription medication.
atc_108a	character ATC code of respondent's eighth prescription medication.
atc_109a	character ATC code of respondent's ninth prescription medication.
atc_110a	character ATC code of respondent's tenth prescription medication.
atc_111a	character ATC code of respondent's eleventh prescription medication.
atc_112a	character ATC code of respondent's twelfth prescription medication.
atc_113a	character ATC code of respondent's thirteenth prescription medication.
atc_114a	character ATC code of respondent's fourteenth prescription medication.
atc_115a	character ATC code of respondent's fifteenth prescription medication.
atc_201a	character ATC code of respondent's first over-the-counter medication.
atc_202a	character ATC code of respondent's second over-the-counter medication.
atc_203a	character ATC code of respondent's third over-the-counter medication.
atc_204a	character ATC code of respondent's fourth over-the-counter medication.
atc_205a	character ATC code of respondent's fifth over-the-counter medication.
atc_206a	character ATC code of respondent's sixth over-the-counter medication.
atc_207a	character ATC code of respondent's seventh over-the-counter medication.
atc_208a	character ATC code of respondent's eighth over-the-counter medication.
atc_209a	character ATC code of respondent's ninth over-the-counter medication.
atc_210a	character ATC code of respondent's tenth over-the-counter medication.
atc_211a	character ATC code of respondent's eleventh over-the-counter medication.
atc_212a	character ATC code of respondent's twelfth over-the-counter medication.
atc_213a	character ATC code of respondent's thirteenth over-the-counter medication.
atc_214a	character ATC code of respondent's fourteenth over-the-counter medication.
atc_215a	character ATC code of respondent's fifteenth over-the-counter medication.

atc_131a character ATC code of respondent's first new prescription medication.

atc_132a character ATC code of respondent's second new prescription medication.

atc_133a character ATC code of respondent's third new prescription medication.

atc_134a character ATC code of respondent's fourth new prescription medication.

atc_135a character ATC code of respondent's fifth new prescription medication.

atc_231a character ATC code of respondent's first new over-the-counter medication.

atc_232a character ATC code of respondent's second new over-the-counter medication.

atc_233a character ATC code of respondent's third new over-the-counter medication.

atc_234a character ATC code of respondent's fourth new over-the-counter medication.

atc_235a character ATC code of respondent's fifth new over-the-counter medication.

mhr_101b integer Response for when the first prescription medication was last taken (1 = Today, . . . , 6 = Never).

mhr_102b integer Response for when the second prescription medication was last taken (1-6).

mhr_103b integer Response for when the third prescription medication was last taken (1-6).

mhr_104b integer Response for when the fourth prescription medication was last taken (1-6).

mhr_105b integer Response for when the fifth prescription medication was last taken (1-6).

mhr_106b integer Response for when the sixth prescription medication was last taken (1-6).

mhr_107b integer Response for when the seventh prescription medication was last taken (1-6).

mhr_108b integer Response for when the eighth prescription medication was last taken (1-6).

mhr_109b integer Response for when the ninth prescription medication was last taken (1-6).

mhr_110b integer Response for when the tenth prescription medication was last taken (1-6).

mhr_111b integer Response for when the eleventh prescription medication was last taken (1-6).

mhr_112b integer Response for when the twelfth prescription medication was last taken (1-6).

mhr_113b integer Response for when the thirteenth prescription medication was last taken (1-6).

mhr_114b integer Response for when the fourteenth prescription medication was last taken (1-6).

mhr_115b integer Response for when the fifteenth prescription medication was last taken (1-6).

mhr_201b integer Response for when the first over-the-counter medication was last taken (1-6).

`mhr_202b` integer Response for when the second over-the-counter medication was last taken (1-6).

`mhr_203b` integer Response for when the third over-the-counter medication was last taken (1-6).

`mhr_204b` integer Response for when the fourth over-the-counter medication was last taken (1-6).

`mhr_205b` integer Response for when the fifth over-the-counter medication was last taken (1-6).

`mhr_206b` integer Response for when the sixth over-the-counter medication was last taken (1-6).

`mhr_207b` integer Response for when the seventh over-the-counter medication was last taken (1-6).

`mhr_208b` integer Response for when the eighth over-the-counter medication was last taken (1-6).

`mhr_209b` integer Response for when the ninth over-the-counter medication was last taken (1-6).

`mhr_210b` integer Response for when the tenth over-the-counter medication was last taken (1-6).

`mhr_211b` integer Response for when the eleventh over-the-counter medication was last taken (1-6).

`mhr_212b` integer Response for when the twelfth over-the-counter medication was last taken (1-6).

`mhr_213b` integer Response for when the thirteenth over-the-counter medication was last taken (1-6).

`mhr_214b` integer Response for when the fourteenth over-the-counter medication was last taken (1-6).

`mhr_215b` integer Response for when the fifteenth over-the-counter medication was last taken (1-6).

`mhr_131b` integer Response for when the first new prescription medication was last taken (1-6).

`mhr_132b` integer Response for when the second new prescription medication was last taken (1-6).

`mhr_133b` integer Response for when the third new prescription medication was last taken (1-6).

`mhr_134b` integer Response for when the fourth new prescription medication was last taken (1-6).

`mhr_135b` integer Response for when the fifth new prescription medication was last taken (1-6).

`mhr_231b` integer Response for when the first new over-the-counter medication was last taken (1-6).

`mhr_232b` integer Response for when the second new over-the-counter medication was last taken (1-6).

mhr_233b	integer Response for when the third new over-the-counter medication was last taken (1-6).
mhr_234b	integer Response for when the fourth new over-the-counter medication was last taken (1-6).
mhr_235b	integer Response for when the fifth new over-the-counter medication was last taken (1-6).

Details

The function identifies beta blockers based on ATC codes starting with "C07", excluding specific sub-codes. It checks all medication variables provided in the input data frame.

```
  **Missing Data Codes:**
- The function handles tagged NAs from the `is_beta_blocker` function and propagates them.
```

Value

numeric Returns 1 if the respondent is taking beta blockers, 0 otherwise. If all medication information is missing, returns a tagged NA.

See Also

is_beta_blocker

Examples

```
# This is a wrapper function and is not intended to be called directly by the user.
# See `is_beta_blocker` for usage examples.
```

---

is_beta_blocker *Beta blockers*

Description

This function determines whether a given medication is a beta blocker. This function processes multiple inputs efficiently.

Usage

```
is_beta_blocker(meucatc, npi_25b)
```

Arguments

meucatc	character ATC code of the medication.
npi_25b	integer Time when the medication was last taken.

Details

Identifies beta blockers based on ATC codes starting with "C07", excluding specific sub-codes.

```
**Missing Data Codes:**
- `meucatc`: `9999996` (Not applicable), `9999997-9999999` (Missing)
- `npi_25b`: `6` (Not applicable), `7-9` (Missing)
```

Value

numeric 1 if medication is a beta blocker, 0 otherwise. If inputs are invalid or out of bounds, the function returns a tagged NA.

Examples

```
# Scalar usage: Single respondent
is_beta_blocker("C07AA13", 3)
# Returns: 1

# Example: Respondent has non-response values for all inputs.
result <- is_beta_blocker("9999998", 8)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
is_beta_blocker(c("C07AA13", "C07AA07"), c(3, 4))
# Returns: c(1, 0)

# Database usage: Applied to survey datasets
library(dplyr)
cycle3_meds |>
  mutate(beta_blocker = is_beta_blocker(meucatc, npi_25b)) |>
  head()
```

---

is_calcium_channel_blocker

Calcium channel blockers

Description

This function checks if a given medication is a calcium channel blocker. This function processes multiple inputs efficiently.

Usage

```
is_calcium_channel_blocker(meucatc, npi_25b)
```

Arguments

meucatc	character ATC code of the medication.
npi_25b	integer Time when the medication was last taken.

Details

Identifies calcium channel blockers based on ATC codes starting with "C08".

```
**Missing Data Codes:**
- `meucatc`: `9999996` (Not applicable), `9999997-9999999` (Missing)
- `npi_25b`: `6` (Not applicable), `7-9` (Missing)
```

Value

numeric 1 if medication is a calcium channel blocker, 0 otherwise. If inputs are invalid or out of bounds, the function returns a tagged NA.

Examples

```
# Scalar usage: Single respondent
is_calcium_channel_blocker("C08CA05", 1)
# Returns: 1

# Example: Respondent has non-response values for all inputs.
result <- is_calcium_channel_blocker("9999998", 8)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
is_calcium_channel_blocker(c("C08CA05", "C01AA05"), c(1, 2))
# Returns: c(1, 0)

# Database usage: Applied to survey datasets
library(dplyr)
cycle3_meds |>
  mutate(ccb = is_calcium_channel_blocker(meucatc, npi_25b)) |>
  head()
```

---

is_ccb_med_cycles1to2 *Calcium channel blockers - cycles 1-2*

Description

This function checks if a person is taking calcium channel blockers based on the provided Anatomical Therapeutic Chemical (ATC) codes for medications and the Canadian Health Measures Survey (CHMS) response for the time when the medication was last taken.

Usage

```
is_ccb_med_cycles1to2(
  atc_101a = NULL,
  atc_102a = NULL,
  atc_103a = NULL,
  atc_104a = NULL,
  atc_105a = NULL,
  atc_106a = NULL,
  atc_107a = NULL,
  atc_108a = NULL,
  atc_109a = NULL,
  atc_110a = NULL,
  atc_111a = NULL,
  atc_112a = NULL,
  atc_113a = NULL,
  atc_114a = NULL,
  atc_115a = NULL,
  atc_201a = NULL,
  atc_202a = NULL,
  atc_203a = NULL,
  atc_204a = NULL,
  atc_205a = NULL,
  atc_206a = NULL,
  atc_207a = NULL,
  atc_208a = NULL,
  atc_209a = NULL,
  atc_210a = NULL,
  atc_211a = NULL,
  atc_212a = NULL,
  atc_213a = NULL,
  atc_214a = NULL,
  atc_215a = NULL,
  atc_131a = NULL,
  atc_132a = NULL,
  atc_133a = NULL,
  atc_134a = NULL,
  atc_135a = NULL,
  atc_231a = NULL,
  atc_232a = NULL,
  atc_233a = NULL,
  atc_234a = NULL,
  atc_235a = NULL,
  mhr_101b = NULL,
  mhr_102b = NULL,
  mhr_103b = NULL,
  mhr_104b = NULL,
  mhr_105b = NULL,
  mhr_106b = NULL,
```

```
  mhr_107b = NULL,
  mhr_108b = NULL,
  mhr_109b = NULL,
  mhr_110b = NULL,
  mhr_111b = NULL,
  mhr_112b = NULL,
  mhr_113b = NULL,
  mhr_114b = NULL,
  mhr_115b = NULL,
  mhr_201b = NULL,
  mhr_202b = NULL,
  mhr_203b = NULL,
  mhr_204b = NULL,
  mhr_205b = NULL,
  mhr_206b = NULL,
  mhr_207b = NULL,
  mhr_208b = NULL,
  mhr_209b = NULL,
  mhr_210b = NULL,
  mhr_211b = NULL,
  mhr_212b = NULL,
  mhr_213b = NULL,
  mhr_214b = NULL,
  mhr_215b = NULL,
  mhr_131b = NULL,
  mhr_132b = NULL,
  mhr_133b = NULL,
  mhr_134b = NULL,
  mhr_135b = NULL,
  mhr_231b = NULL,
  mhr_232b = NULL,
  mhr_233b = NULL,
  mhr_234b = NULL,
  mhr_235b = NULL
)
```

Arguments

atc_101a	character ATC code of respondent's first prescription medication.
atc_102a	character ATC code of respondent's second prescription medication.
atc_103a	character ATC code of respondent's third prescription medication.
atc_104a	character ATC code of respondent's fourth prescription medication.
atc_105a	character ATC code of respondent's fifth prescription medication.
atc_106a	character ATC code of respondent's sixth prescription medication.
atc_107a	character ATC code of respondent's seventh prescription medication.
atc_108a	character ATC code of respondent's eighth prescription medication.

atc_109a character ATC code of respondent's ninth prescription medication.

atc_110a character ATC code of respondent's tenth prescription medication.

atc_111a character ATC code of respondent's eleventh prescription medication.

atc_112a character ATC code of respondent's twelfth prescription medication.

atc_113a character ATC code of respondent's thirteenth prescription medication.

atc_114a character ATC code of respondent's fourteenth prescription medication.

atc_115a character ATC code of respondent's fifteenth prescription medication.

atc_201a character ATC code of respondent's first over-the-counter medication.

atc_202a character ATC code of respondent's second over-the-counter medication.

atc_203a character ATC code of respondent's third over-the-counter medication.

atc_204a character ATC code of respondent's fourth over-the-counter medication.

atc_205a character ATC code of respondent's fifth over-the-counter medication.

atc_206a character ATC code of respondent's sixth over-the-counter medication.

atc_207a character ATC code of respondent's seventh over-the-counter medication.

atc_208a character ATC code of respondent's eighth over-the-counter medication.

atc_209a character ATC code of respondent's ninth over-the-counter medication.

atc_210a character ATC code of respondent's tenth over-the-counter medication.

atc_211a character ATC code of respondent's eleventh over-the-counter medication.

atc_212a character ATC code of respondent's twelfth over-the-counter medication.

atc_213a character ATC code of respondent's thirteenth over-the-counter medication.

atc_214a character ATC code of respondent's fourteenth over-the-counter medication.

atc_215a character ATC code of respondent's fifteenth over-the-counter medication.

atc_131a character ATC code of respondent's first new prescription medication.

atc_132a character ATC code of respondent's second new prescription medication.

atc_133a character ATC code of respondent's third new prescription medication.

atc_134a character ATC code of respondent's fourth new prescription medication.

atc_135a character ATC code of respondent's fifth new prescription medication.

atc_231a character ATC code of respondent's first new over-the-counter medication.

atc_232a character ATC code of respondent's second new over-the-counter medication.

atc_233a character ATC code of respondent's third new over-the-counter medication.

atc_234a character ATC code of respondent's fourth new over-the-counter medication.

atc_235a character ATC code of respondent's fifth new over-the-counter medication.

mhr_101b integer Response for when the first prescription medication was last taken (1 = Today, . . . , 6 = Never).

mhr_102b integer Response for when the second prescription medication was last taken (1-6).

mhr_103b integer Response for when the third prescription medication was last taken (1-6).

mhr_104b integer Response for when the fourth prescription medication was last taken (1-6).

mhr_105b integer Response for when the fifth prescription medication was last taken (1-6).

mhr_106b integer Response for when the sixth prescription medication was last taken (1-6).

mhr_107b integer Response for when the seventh prescription medication was last taken (1-6).

mhr_108b integer Response for when the eighth prescription medication was last taken (1-6).

mhr_109b integer Response for when the ninth prescription medication was last taken (1-6).

mhr_110b integer Response for when the tenth prescription medication was last taken (1-6).

mhr_111b integer Response for when the eleventh prescription medication was last taken (1-6).

mhr_112b integer Response for when the twelfth prescription medication was last taken (1-6).

mhr_113b integer Response for when the thirteenth prescription medication was last taken (1-6).

mhr_114b integer Response for when the fourteenth prescription medication was last taken (1-6).

mhr_115b integer Response for when the fifteenth prescription medication was last taken (1-6).

mhr_201b integer Response for when the first over-the-counter medication was last taken (1-6).

mhr_202b integer Response for when the second over-the-counter medication was last taken (1-6).

mhr_203b integer Response for when the third over-the-counter medication was last taken (1-6).

mhr_204b integer Response for when the fourth over-the-counter medication was last taken (1-6).

mhr_205b integer Response for when the fifth over-the-counter medication was last taken (1-6).

mhr_206b integer Response for when the sixth over-the-counter medication was last taken (1-6).

mhr_207b integer Response for when the seventh over-the-counter medication was last taken (1-6).

mhr_208b integer Response for when the eighth over-the-counter medication was last taken (1-6).

mhr_209b integer Response for when the ninth over-the-counter medication was last taken (1-6).

mhr_210b integer Response for when the tenth over-the-counter medication was last taken (1-6).

mhr_211b integer Response for when the eleventh over-the-counter medication was last taken (1-6).

mhr_212b integer Response for when the twelfth over-the-counter medication was last taken (1-6).

mhr_213b integer Response for when the thirteenth over-the-counter medication was last taken (1-6).

mhr_214b integer Response for when the fourteenth over-the-counter medication was last taken (1-6).

mhr_215b integer Response for when the fifteenth over-the-counter medication was last taken (1-6).

mhr_131b integer Response for when the first new prescription medication was last taken (1-6).

mhr_132b integer Response for when the second new prescription medication was last taken (1-6).

mhr_133b integer Response for when the third new prescription medication was last taken (1-6).

mhr_134b integer Response for when the fourth new prescription medication was last taken (1-6).

mhr_135b integer Response for when the fifth new prescription medication was last taken (1-6).

mhr_231b integer Response for when the first new over-the-counter medication was last taken (1-6).

mhr_232b integer Response for when the second new over-the-counter medication was last taken (1-6).

mhr_233b integer Response for when the third new over-the-counter medication was last taken (1-6).

mhr_234b integer Response for when the fourth new over-the-counter medication was last taken (1-6).

mhr_235b integer Response for when the fifth new over-the-counter medication was last taken (1-6).

Details

The function identifies calcium channel blockers based on ATC codes starting with "C08". It checks all medication variables provided in the input data frame.

```
  **Missing Data Codes:**
- The function handles tagged NAs from the `is_calcium_channel_blocker` function and propagates them
```

Value

numeric Returns 1 if the person is taking calcium channel blockers, 0 otherwise. If all medication information is missing, it returns a tagged NA.

See Also

is_calcium_channel_blocker

Examples

```
# This is a wrapper function and is not intended to be called directly by the user.
# See `is_calcium_channel_blocker` for usage examples.
```

---

is_diab_med_cycles1to2

Diabetes medications - cycles 1-2

Description

This function checks if a person is taking diabetes drugs based on the provided Anatomical Therapeutic Chemical (ATC) codes for medications and the Canadian Health Measures Survey (CHMS) response for the time when the medication was last taken.

Usage

```
is_diab_med_cycles1to2(
  atc_101a = NULL,
  atc_102a = NULL,
  atc_103a = NULL,
  atc_104a = NULL,
  atc_105a = NULL,
  atc_106a = NULL,
  atc_107a = NULL,
  atc_108a = NULL,
  atc_109a = NULL,
  atc_110a = NULL,
  atc_111a = NULL,
  atc_112a = NULL,
  atc_113a = NULL,
  atc_114a = NULL,
  atc_115a = NULL,
  atc_201a = NULL,
  atc_202a = NULL,
  atc_203a = NULL,
  atc_204a = NULL,
  atc_205a = NULL,
  atc_206a = NULL,
  atc_207a = NULL,
  atc_208a = NULL,
  atc_209a = NULL,
  atc_210a = NULL,
  atc_211a = NULL,
```

```
  atc_212a = NULL,
  atc_213a = NULL,
  atc_214a = NULL,
  atc_215a = NULL,
  atc_131a = NULL,
  atc_132a = NULL,
  atc_133a = NULL,
  atc_134a = NULL,
  atc_135a = NULL,
  atc_231a = NULL,
  atc_232a = NULL,
  atc_233a = NULL,
  atc_234a = NULL,
  atc_235a = NULL,
  mhr_101b = NULL,
  mhr_102b = NULL,
  mhr_103b = NULL,
  mhr_104b = NULL,
  mhr_105b = NULL,
  mhr_106b = NULL,
  mhr_107b = NULL,
  mhr_108b = NULL,
  mhr_109b = NULL,
  mhr_110b = NULL,
  mhr_111b = NULL,
  mhr_112b = NULL,
  mhr_113b = NULL,
  mhr_114b = NULL,
  mhr_115b = NULL,
  mhr_201b = NULL,
  mhr_202b = NULL,
  mhr_203b = NULL,
  mhr_204b = NULL,
  mhr_205b = NULL,
  mhr_206b = NULL,
  mhr_207b = NULL,
  mhr_208b = NULL,
  mhr_209b = NULL,
  mhr_210b = NULL,
  mhr_211b = NULL,
  mhr_212b = NULL,
  mhr_213b = NULL,
  mhr_214b = NULL,
  mhr_215b = NULL,
  mhr_131b = NULL,
  mhr_132b = NULL,
  mhr_133b = NULL,
  mhr_134b = NULL,
```

```
  mhr_135b = NULL,
  mhr_231b = NULL,
  mhr_232b = NULL,
  mhr_233b = NULL,
  mhr_234b = NULL,
  mhr_235b = NULL
)
```

Arguments

atc_101a	character ATC code of respondent's first prescription medication.
atc_102a	character ATC code of respondent's second prescription medication.
atc_103a	character ATC code of respondent's third prescription medication.
atc_104a	character ATC code of respondent's fourth prescription medication.
atc_105a	character ATC code of respondent's fifth prescription medication.
atc_106a	character ATC code of respondent's sixth prescription medication.
atc_107a	character ATC code of respondent's seventh prescription medication.
atc_108a	character ATC code of respondent's eighth prescription medication.
atc_109a	character ATC code of respondent's ninth prescription medication.
atc_110a	character ATC code of respondent's tenth prescription medication.
atc_111a	character ATC code of respondent's eleventh prescription medication.
atc_112a	character ATC code of respondent's twelfth prescription medication.
atc_113a	character ATC code of respondent's thirteenth prescription medication.
atc_114a	character ATC code of respondent's fourteenth prescription medication.
atc_115a	character ATC code of respondent's fifteenth prescription medication.
atc_201a	character ATC code of respondent's first over-the-counter medication.
atc_202a	character ATC code of respondent's second over-the-counter medication.
atc_203a	character ATC code of respondent's third over-the-counter medication.
atc_204a	character ATC code of respondent's fourth over-the-counter medication.
atc_205a	character ATC code of respondent's fifth over-the-counter medication.
atc_206a	character ATC code of respondent's sixth over-the-counter medication.
atc_207a	character ATC code of respondent's seventh over-the-counter medication.
atc_208a	character ATC code of respondent's eighth over-the-counter medication.
atc_209a	character ATC code of respondent's ninth over-the-counter medication.
atc_210a	character ATC code of respondent's tenth over-the-counter medication.
atc_211a	character ATC code of respondent's eleventh over-the-counter medication.
atc_212a	character ATC code of respondent's twelfth over-the-counter medication.
atc_213a	character ATC code of respondent's thirteenth over-the-counter medication.
atc_214a	character ATC code of respondent's fourteenth over-the-counter medication.
atc_215a	character ATC code of respondent's fifteenth over-the-counter medication.

atc_131a character ATC code of respondent's first new prescription medication.

atc_132a character ATC code of respondent's second new prescription medication.

atc_133a character ATC code of respondent's third new prescription medication.

atc_134a character ATC code of respondent's fourth new prescription medication.

atc_135a character ATC code of respondent's fifth new prescription medication.

atc_231a character ATC code of respondent's first new over-the-counter medication.

atc_232a character ATC code of respondent's second new over-the-counter medication.

atc_233a character ATC code of respondent's third new over-the-counter medication.

atc_234a character ATC code of respondent's fourth new over-the-counter medication.

atc_235a character ATC code of respondent's fifth new over-the-counter medication.

mhr_101b integer Response for when the first prescription medication was last taken (1 = Today, . . . , 6 = Never).

mhr_102b integer Response for when the second prescription medication was last taken (1-6).

mhr_103b integer Response for when the third prescription medication was last taken (1-6).

mhr_104b integer Response for when the fourth prescription medication was last taken (1-6).

mhr_105b integer Response for when the fifth prescription medication was last taken (1-6).

mhr_106b integer Response for when the sixth prescription medication was last taken (1-6).

mhr_107b integer Response for when the seventh prescription medication was last taken (1-6).

mhr_108b integer Response for when the eighth prescription medication was last taken (1-6).

mhr_109b integer Response for when the ninth prescription medication was last taken (1-6).

mhr_110b integer Response for when the tenth prescription medication was last taken (1-6).

mhr_111b integer Response for when the eleventh prescription medication was last taken (1-6).

mhr_112b integer Response for when the twelfth prescription medication was last taken (1-6).

mhr_113b integer Response for when the thirteenth prescription medication was last taken (1-6).

mhr_114b integer Response for when the fourteenth prescription medication was last taken (1-6).

mhr_115b integer Response for when the fifteenth prescription medication was last taken (1-6).

mhr_201b integer Response for when the first over-the-counter medication was last taken (1-6).

mhr_202b integer Response for when the second over-the-counter medication was last taken (1-6).

mhr_203b integer Response for when the third over-the-counter medication was last taken (1-6).

mhr_204b integer Response for when the fourth over-the-counter medication was last taken (1-6).

mhr_205b integer Response for when the fifth over-the-counter medication was last taken (1-6).

mhr_206b integer Response for when the sixth over-the-counter medication was last taken (1-6).

mhr_207b integer Response for when the seventh over-the-counter medication was last taken (1-6).

mhr_208b integer Response for when the eighth over-the-counter medication was last taken (1-6).

mhr_209b integer Response for when the ninth over-the-counter medication was last taken (1-6).

mhr_210b integer Response for when the tenth over-the-counter medication was last taken (1-6).

mhr_211b integer Response for when the eleventh over-the-counter medication was last taken (1-6).

mhr_212b integer Response for when the twelfth over-the-counter medication was last taken (1-6).

mhr_213b integer Response for when the thirteenth over-the-counter medication was last taken (1-6).

mhr_214b integer Response for when the fourteenth over-the-counter medication was last taken (1-6).

mhr_215b integer Response for when the fifteenth over-the-counter medication was last taken (1-6).

mhr_131b integer Response for when the first new prescription medication was last taken (1-6).

mhr_132b integer Response for when the second new prescription medication was last taken (1-6).

mhr_133b integer Response for when the third new prescription medication was last taken (1-6).

mhr_134b integer Response for when the fourth new prescription medication was last taken (1-6).

mhr_135b integer Response for when the fifth new prescription medication was last taken (1-6).

mhr_231b integer Response for when the first new over-the-counter medication was last taken (1-6).

mhr_232b integer Response for when the second new over-the-counter medication was last taken (1-6).

mhr_233b	integer Response for when the third new over-the-counter medication was last taken (1-6).
mhr_234b	integer Response for when the fourth new over-the-counter medication was last taken (1-6).
mhr_235b	integer Response for when the fifth new over-the-counter medication was last taken (1-6).

Details

The function identifies diabetes drugs based on ATC codes starting with "A10". It checks all medication variables provided in the input data frame.

```
  **Missing Data Codes:**
- The function handles tagged NAs from the `is_diabetes_med` function and propagates them.
```

Value

numeric Returns 1 if the person is taking any diabetes drugs, 0 otherwise. If all medication information is missing, it returns a tagged NA.

See Also

is_diabetes_med

Examples

```
# This is a wrapper function and is not intended to be called directly by the user.
# See `is_diabetes_med` for usage examples.
```

---

is_diabetes_med *Diabetes medications*

Description

This function checks if a given medication is a diabetes drug. This function processes multiple inputs efficiently.

Usage

```
is_diabetes_med(meucatc, npi_25b)
```

Arguments

meucatc	character ATC code of the medication.
npi_25b	integer Time when the medication was last taken.

Details

Identifies diabetes drugs based on ATC codes starting with "A10".

```
**Missing Data Codes:**
- `meucatc`: `9999996` (Not applicable), `9999997-9999999` (Missing)
- `npi_25b`: `6` (Not applicable), `7-9` (Missing)
```

Value

numeric 1 if medication is a diabetes drug, 0 otherwise. If inputs are invalid or out of bounds, the function returns a tagged NA.

Examples

```
# Scalar usage: Single respondent
is_diabetes_med("A10BB09", 3)
# Returns: 1

# Example: Respondent has non-response values for all inputs.
result <- is_diabetes_med("9999998", 8)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
is_diabetes_med(c("A10BB09", "C09AA02"), c(3, 2))
# Returns: c(1, 0)

# Database usage: Applied to survey datasets
library(dplyr)
cycle3_meds |>
  mutate(diabetes_med = is_diabetes_med(meucatc, npi_25b)) |>
  head()
```

---

is_diur_med_cycles1to2

Diuretics - cycles 1-2

Description

This function checks if a person is taking diuretics based on the provided Anatomical Therapeutic Chemical (ATC) codes for medications and the Canadian Health Measures Survey (CHMS) response for the time when the medication was last taken.

Usage

```
is_diur_med_cycles1to2(
  atc_101a = NULL,
  atc_102a = NULL,
  atc_103a = NULL,
  atc_104a = NULL,
  atc_105a = NULL,
  atc_106a = NULL,
  atc_107a = NULL,
  atc_108a = NULL,
  atc_109a = NULL,
  atc_110a = NULL,
  atc_111a = NULL,
  atc_112a = NULL,
  atc_113a = NULL,
  atc_114a = NULL,
  atc_115a = NULL,
  atc_201a = NULL,
  atc_202a = NULL,
  atc_203a = NULL,
  atc_204a = NULL,
  atc_205a = NULL,
  atc_206a = NULL,
  atc_207a = NULL,
  atc_208a = NULL,
  atc_209a = NULL,
  atc_210a = NULL,
  atc_211a = NULL,
  atc_212a = NULL,
  atc_213a = NULL,
  atc_214a = NULL,
  atc_215a = NULL,
  atc_131a = NULL,
  atc_132a = NULL,
  atc_133a = NULL,
  atc_134a = NULL,
  atc_135a = NULL,
  atc_231a = NULL,
  atc_232a = NULL,
  atc_233a = NULL,
  atc_234a = NULL,
  atc_235a = NULL,
  mhr_101b = NULL,
  mhr_102b = NULL,
  mhr_103b = NULL,
  mhr_104b = NULL,
  mhr_105b = NULL,
  mhr_106b = NULL,
```

```
  mhr_107b = NULL,
  mhr_108b = NULL,
  mhr_109b = NULL,
  mhr_110b = NULL,
  mhr_111b = NULL,
  mhr_112b = NULL,
  mhr_113b = NULL,
  mhr_114b = NULL,
  mhr_115b = NULL,
  mhr_201b = NULL,
  mhr_202b = NULL,
  mhr_203b = NULL,
  mhr_204b = NULL,
  mhr_205b = NULL,
  mhr_206b = NULL,
  mhr_207b = NULL,
  mhr_208b = NULL,
  mhr_209b = NULL,
  mhr_210b = NULL,
  mhr_211b = NULL,
  mhr_212b = NULL,
  mhr_213b = NULL,
  mhr_214b = NULL,
  mhr_215b = NULL,
  mhr_131b = NULL,
  mhr_132b = NULL,
  mhr_133b = NULL,
  mhr_134b = NULL,
  mhr_135b = NULL,
  mhr_231b = NULL,
  mhr_232b = NULL,
  mhr_233b = NULL,
  mhr_234b = NULL,
  mhr_235b = NULL
)
```

Arguments

atc_101a	character ATC code of respondent's first prescription medication.
atc_102a	character ATC code of respondent's second prescription medication.
atc_103a	character ATC code of respondent's third prescription medication.
atc_104a	character ATC code of respondent's fourth prescription medication.
atc_105a	character ATC code of respondent's fifth prescription medication.
atc_106a	character ATC code of respondent's sixth prescription medication.
atc_107a	character ATC code of respondent's seventh prescription medication.
atc_108a	character ATC code of respondent's eighth prescription medication.

atc_109a character ATC code of respondent's ninth prescription medication.

atc_110a character ATC code of respondent's tenth prescription medication.

atc_111a character ATC code of respondent's eleventh prescription medication.

atc_112a character ATC code of respondent's twelfth prescription medication.

atc_113a character ATC code of respondent's thirteenth prescription medication.

atc_114a character ATC code of respondent's fourteenth prescription medication.

atc_115a character ATC code of respondent's fifteenth prescription medication.

atc_201a character ATC code of respondent's first over-the-counter medication.

atc_202a character ATC code of respondent's second over-the-counter medication.

atc_203a character ATC code of respondent's third over-the-counter medication.

atc_204a character ATC code of respondent's fourth over-the-counter medication.

atc_205a character ATC code of respondent's fifth over-the-counter medication.

atc_206a character ATC code of respondent's sixth over-the-counter medication.

atc_207a character ATC code of respondent's seventh over-the-counter medication.

atc_208a character ATC code of respondent's eighth over-the-counter medication.

atc_209a character ATC code of respondent's ninth over-the-counter medication.

atc_210a character ATC code of respondent's tenth over-the-counter medication.

atc_211a character ATC code of respondent's eleventh over-the-counter medication.

atc_212a character ATC code of respondent's twelfth over-the-counter medication.

atc_213a character ATC code of respondent's thirteenth over-the-counter medication.

atc_214a character ATC code of respondent's fourteenth over-the-counter medication.

atc_215a character ATC code of respondent's fifteenth over-the-counter medication.

atc_131a character ATC code of respondent's first new prescription medication.

atc_132a character ATC code of respondent's second new prescription medication.

atc_133a character ATC code of respondent's third new prescription medication.

atc_134a character ATC code of respondent's fourth new prescription medication.

atc_135a character ATC code of respondent's fifth new prescription medication.

atc_231a character ATC code of respondent's first new over-the-counter medication.

atc_232a character ATC code of respondent's second new over-the-counter medication.

atc_233a character ATC code of respondent's third new over-the-counter medication.

atc_234a character ATC code of respondent's fourth new over-the-counter medication.

atc_235a character ATC code of respondent's fifth new over-the-counter medication.

mhr_101b integer Response for when the first prescription medication was last taken (1 = Today, . . . , 6 = Never).

mhr_102b integer Response for when the second prescription medication was last taken (1-6).

mhr_103b integer Response for when the third prescription medication was last taken (1-6).

mhr_104b integer Response for when the fourth prescription medication was last taken (1-6).

mhr_105b integer Response for when the fifth prescription medication was last taken (1-6).

mhr_106b integer Response for when the sixth prescription medication was last taken (1-6).

mhr_107b integer Response for when the seventh prescription medication was last taken (1-6).

mhr_108b integer Response for when the eighth prescription medication was last taken (1-6).

mhr_109b integer Response for when the ninth prescription medication was last taken (1-6).

mhr_110b integer Response for when the tenth prescription medication was last taken (1-6).

mhr_111b integer Response for when the eleventh prescription medication was last taken (1-6).

mhr_112b integer Response for when the twelfth prescription medication was last taken (1-6).

mhr_113b integer Response for when the thirteenth prescription medication was last taken (1-6).

mhr_114b integer Response for when the fourteenth prescription medication was last taken (1-6).

mhr_115b integer Response for when the fifteenth prescription medication was last taken (1-6).

mhr_201b integer Response for when the first over-the-counter medication was last taken (1-6).

mhr_202b integer Response for when the second over-the-counter medication was last taken (1-6).

mhr_203b integer Response for when the third over-the-counter medication was last taken (1-6).

mhr_204b integer Response for when the fourth over-the-counter medication was last taken (1-6).

mhr_205b integer Response for when the fifth over-the-counter medication was last taken (1-6).

mhr_206b integer Response for when the sixth over-the-counter medication was last taken (1-6).

mhr_207b integer Response for when the seventh over-the-counter medication was last taken (1-6).

mhr_208b integer Response for when the eighth over-the-counter medication was last taken (1-6).

mhr_209b integer Response for when the ninth over-the-counter medication was last taken (1-6).

mhr_210b integer Response for when the tenth over-the-counter medication was last taken (1-6).

mhr_211b integer Response for when the eleventh over-the-counter medication was last taken (1-6).

mhr_212b integer Response for when the twelfth over-the-counter medication was last taken (1-6).

mhr_213b integer Response for when the thirteenth over-the-counter medication was last taken (1-6).

mhr_214b integer Response for when the fourteenth over-the-counter medication was last taken (1-6).

mhr_215b integer Response for when the fifteenth over-the-counter medication was last taken (1-6).

mhr_131b integer Response for when the first new prescription medication was last taken (1-6).

mhr_132b integer Response for when the second new prescription medication was last taken (1-6).

mhr_133b integer Response for when the third new prescription medication was last taken (1-6).

mhr_134b integer Response for when the fourth new prescription medication was last taken (1-6).

mhr_135b integer Response for when the fifth new prescription medication was last taken (1-6).

mhr_231b integer Response for when the first new over-the-counter medication was last taken (1-6).

mhr_232b integer Response for when the second new over-the-counter medication was last taken (1-6).

mhr_233b integer Response for when the third new over-the-counter medication was last taken (1-6).

mhr_234b integer Response for when the fourth new over-the-counter medication was last taken (1-6).

mhr_235b integer Response for when the fifth new over-the-counter medication was last taken (1-6).

Details

The function identifies diuretics based on ATC codes starting with "C03", excluding specific sub-codes. It checks all medication variables provided in the input data frame.

```
  **Missing Data Codes:**
- The function handles tagged NAs from the `is_diuretic` function and propagates them.
```

Value

numeric Returns 1 if the person is taking diuretics, 0 otherwise. If all medication information is missing, it returns a tagged NA.

See Also

is_diuretic

Examples

```
# This is a wrapper function and is not intended to be called directly by the user.
# See `is_diuretic` for usage examples.
```

---

is_diuretic *Diuretics*

Description

This function checks if a given medication is a diuretic. This function processes multiple inputs efficiently.

Usage

```
is_diuretic(meucatc, npi_25b)
```

Arguments

meucatc	character ATC code of the medication.
npi_25b	integer Time when the medication was last taken.

Details

Identifies diuretics based on ATC codes starting with "C03", excluding specific sub-codes.

```
**Missing Data Codes:**
- `meucatc`: `9999996` (Not applicable), `9999997-9999999` (Missing)
- `npi_25b`: `6` (Not applicable), `7-9` (Missing)
```

Value

numeric 1 if medication is a diuretic, 0 otherwise. If inputs are invalid or out of bounds, the function returns a tagged NA.

Examples

```
# Scalar usage: Single respondent
is_diuretic("C03AA03", 3)
# Returns: 1

# Example: Respondent has non-response values for all inputs.
result <- is_diuretic("9999998", 8)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
```

```
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
is_diuretic(c("C03AA03", "C03BA08"), c(3, 2))
# Returns: c(1, 0)

# Database usage: Applied to survey datasets
library(dplyr)
cycle3_meds |>
  mutate(diuretic = is_diuretic(meucatc, npi_25b)) |>
  head()
```

`is_lowest_income_quintile` *Lowest income quintile indicator*

Description

This function checks if an individual's income category corresponds to the lowest income quintile.

Usage

```
is_lowest_income_quintile(income_quintile)
```

Arguments

`income_quintile`
: integer A categorical vector indicating the income category as defined by the categorize_income_quintile function.

Details

This function identifies individuals in the lowest income quintile, a common indicator for socioeconomic disadvantage.

```
**Missing Data Codes:**
- Propagates tagged NAs from the input `income_quintile`.
```

Value

integer Whether the individual is in the lowest income quintile:

- 1: In the lowest income quntile
- 2: Not in the lowest income quntile
- `haven::tagged_na("a")`: Not applicable
- `haven::tagged_na("b")`: Missing

See Also

categorize_income_quintile()

Examples

```
# Scalar usage: Single respondent
# Example 1: Check if an income category of 3 is in the lowest quintile
is_lowest_income_quintile(3)
# Output: 2

# Example 2: Check if an income category of 1 is in the lowest quintile
is_lowest_income_quintile(1)
# Output: 1

# Multiple respondents
is_lowest_income_quintile(c(3, 1, 5))
# Returns: c(2, 1, 2)

# Database usage: Applied to survey datasets
library(dplyr)
cycle4 |>
  mutate(adj_hh_income = calculate_household_income(thi_01, dhhdhsz)) |>
  mutate(income_category = categorize_income_quintile(adj_hh_income)) |>
  mutate(in_lowest_quintile = is_lowest_income_quintile(income_category)) |>
  head()
```

---

is_misc_htn_med_cycles1to2

Other anti-hypertensive medications - cycles 1-2

Description

This function checks if a person is taking another type of anti-hypertensive medication based on the provided Anatomical Therapeutic Chemical (ATC) codes for medications and the Canadian Health Measures Survey (CHMS) response for the time when the medication was last taken.

Usage

```
is_misc_htn_med_cycles1to2(
  atc_101a = NULL,
  atc_102a = NULL,
  atc_103a = NULL,
  atc_104a = NULL,
  atc_105a = NULL,
  atc_106a = NULL,
  atc_107a = NULL,
  atc_108a = NULL,
```

```
  atc_109a = NULL,
  atc_110a = NULL,
  atc_111a = NULL,
  atc_112a = NULL,
  atc_113a = NULL,
  atc_114a = NULL,
  atc_115a = NULL,
  atc_201a = NULL,
  atc_202a = NULL,
  atc_203a = NULL,
  atc_204a = NULL,
  atc_205a = NULL,
  atc_206a = NULL,
  atc_207a = NULL,
  atc_208a = NULL,
  atc_209a = NULL,
  atc_210a = NULL,
  atc_211a = NULL,
  atc_212a = NULL,
  atc_213a = NULL,
  atc_214a = NULL,
  atc_215a = NULL,
  atc_131a = NULL,
  atc_132a = NULL,
  atc_133a = NULL,
  atc_134a = NULL,
  atc_135a = NULL,
  atc_231a = NULL,
  atc_232a = NULL,
  atc_233a = NULL,
  atc_234a = NULL,
  atc_235a = NULL,
  mhr_101b = NULL,
  mhr_102b = NULL,
  mhr_103b = NULL,
  mhr_104b = NULL,
  mhr_105b = NULL,
  mhr_106b = NULL,
  mhr_107b = NULL,
  mhr_108b = NULL,
  mhr_109b = NULL,
  mhr_110b = NULL,
  mhr_111b = NULL,
  mhr_112b = NULL,
  mhr_113b = NULL,
  mhr_114b = NULL,
  mhr_115b = NULL,
  mhr_201b = NULL,
```

```
  mhr_202b = NULL,
  mhr_203b = NULL,
  mhr_204b = NULL,
  mhr_205b = NULL,
  mhr_206b = NULL,
  mhr_207b = NULL,
  mhr_208b = NULL,
  mhr_209b = NULL,
  mhr_210b = NULL,
  mhr_211b = NULL,
  mhr_212b = NULL,
  mhr_213b = NULL,
  mhr_214b = NULL,
  mhr_215b = NULL,
  mhr_131b = NULL,
  mhr_132b = NULL,
  mhr_133b = NULL,
  mhr_134b = NULL,
  mhr_135b = NULL,
  mhr_231b = NULL,
  mhr_232b = NULL,
  mhr_233b = NULL,
  mhr_234b = NULL,
  mhr_235b = NULL
)
```

Arguments

atc_101a	character ATC code of respondent's first prescription medication.
atc_102a	character ATC code of respondent's second prescription medication.
atc_103a	character ATC code of respondent's third prescription medication.
atc_104a	character ATC code of respondent's fourth prescription medication.
atc_105a	character ATC code of respondent's fifth prescription medication.
atc_106a	character ATC code of respondent's sixth prescription medication.
atc_107a	character ATC code of respondent's seventh prescription medication.
atc_108a	character ATC code of respondent's eighth prescription medication.
atc_109a	character ATC code of respondent's ninth prescription medication.
atc_110a	character ATC code of respondent's tenth prescription medication.
atc_111a	character ATC code of respondent's eleventh prescription medication.
atc_112a	character ATC code of respondent's twelfth prescription medication.
atc_113a	character ATC code of respondent's thirteenth prescription medication.
atc_114a	character ATC code of respondent's fourteenth prescription medication.
atc_115a	character ATC code of respondent's fifteenth prescription medication.
atc_201a	character ATC code of respondent's first over-the-counter medication.

atc_202a character ATC code of respondent's second over-the-counter medication.

atc_203a character ATC code of respondent's third over-the-counter medication.

atc_204a character ATC code of respondent's fourth over-the-counter medication.

atc_205a character ATC code of respondent's fifth over-the-counter medication.

atc_206a character ATC code of respondent's sixth over-the-counter medication.

atc_207a character ATC code of respondent's seventh over-the-counter medication.

atc_208a character ATC code of respondent's eighth over-the-counter medication.

atc_209a character ATC code of respondent's ninth over-the-counter medication.

atc_210a character ATC code of respondent's tenth over-the-counter medication.

atc_211a character ATC code of respondent's eleventh over-the-counter medication.

atc_212a character ATC code of respondent's twelfth over-the-counter medication.

atc_213a character ATC code of respondent's thirteenth over-the-counter medication.

atc_214a character ATC code of respondent's fourteenth over-the-counter medication.

atc_215a character ATC code of respondent's fifteenth over-the-counter medication.

atc_131a character ATC code of respondent's first new prescription medication.

atc_132a character ATC code of respondent's second new prescription medication.

atc_133a character ATC code of respondent's third new prescription medication.

atc_134a character ATC code of respondent's fourth new prescription medication.

atc_135a character ATC code of respondent's fifth new prescription medication.

atc_231a character ATC code of respondent's first new over-the-counter medication.

atc_232a character ATC code of respondent's second new over-the-counter medication.

atc_233a character ATC code of respondent's third new over-the-counter medication.

atc_234a character ATC code of respondent's fourth new over-the-counter medication.

atc_235a character ATC code of respondent's fifth new over-the-counter medication.

mhr_101b integer Response for when the first prescription medication was last taken (1 = Today, . . . , 6 = Never).

mhr_102b integer Response for when the second prescription medication was last taken (1-6).

mhr_103b integer Response for when the third prescription medication was last taken (1-6).

mhr_104b integer Response for when the fourth prescription medication was last taken (1-6).

mhr_105b integer Response for when the fifth prescription medication was last taken (1-6).

mhr_106b integer Response for when the sixth prescription medication was last taken (1-6).

mhr_107b integer Response for when the seventh prescription medication was last taken (1-6).

mhr_108b integer Response for when the eighth prescription medication was last taken (1-6).

`mhr_109b` integer Response for when the ninth prescription medication was last taken (1-6).

`mhr_110b` integer Response for when the tenth prescription medication was last taken (1-6).

`mhr_111b` integer Response for when the eleventh prescription medication was last taken (1-6).

`mhr_112b` integer Response for when the twelfth prescription medication was last taken (1-6).

`mhr_113b` integer Response for when the thirteenth prescription medication was last taken (1-6).

`mhr_114b` integer Response for when the fourteenth prescription medication was last taken (1-6).

`mhr_115b` integer Response for when the fifteenth prescription medication was last taken (1-6).

`mhr_201b` integer Response for when the first over-the-counter medication was last taken (1-6).

`mhr_202b` integer Response for when the second over-the-counter medication was last taken (1-6).

`mhr_203b` integer Response for when the third over-the-counter medication was last taken (1-6).

`mhr_204b` integer Response for when the fourth over-the-counter medication was last taken (1-6).

`mhr_205b` integer Response for when the fifth over-the-counter medication was last taken (1-6).

`mhr_206b` integer Response for when the sixth over-the-counter medication was last taken (1-6).

`mhr_207b` integer Response for when the seventh over-the-counter medication was last taken (1-6).

`mhr_208b` integer Response for when the eighth over-the-counter medication was last taken (1-6).

`mhr_209b` integer Response for when the ninth over-the-counter medication was last taken (1-6).

`mhr_210b` integer Response for when the tenth over-the-counter medication was last taken (1-6).

`mhr_211b` integer Response for when the eleventh over-the-counter medication was last taken (1-6).

`mhr_212b` integer Response for when the twelfth over-the-counter medication was last taken (1-6).

`mhr_213b` integer Response for when the thirteenth over-the-counter medication was last taken (1-6).

`mhr_214b` integer Response for when the fourteenth over-the-counter medication was last taken (1-6).

mhr_215b integer Response for when the fifteenth over-the-counter medication was last taken (1-6).

mhr_131b integer Response for when the first new prescription medication was last taken (1-6).

mhr_132b integer Response for when the second new prescription medication was last taken (1-6).

mhr_133b integer Response for when the third new prescription medication was last taken (1-6).

mhr_134b integer Response for when the fourth new prescription medication was last taken (1-6).

mhr_135b integer Response for when the fifth new prescription medication was last taken (1-6).

mhr_231b integer Response for when the first new over-the-counter medication was last taken (1-6).

mhr_232b integer Response for when the second new over-the-counter medication was last taken (1-6).

mhr_233b integer Response for when the third new over-the-counter medication was last taken (1-6).

mhr_234b integer Response for when the fourth new over-the-counter medication was last taken (1-6).

mhr_235b integer Response for when the fifth new over-the-counter medication was last taken (1-6).

Details

The function identifies other anti-hypertensive drugs based on ATC codes starting with "C02", excluding a specific sub-code. It checks all medication variables provided in the input data frame.

```
  **Missing Data Codes:**
- The function handles tagged NAs from the `is_other_antihtn_med` function and propagates them.
```

Value

numeric Returns 1 if the person is taking another type of anti-hypertensive medication, 0 otherwise. If all medication information is missing, it returns a tagged NA.

See Also

is_other_antihtn_med

Examples

```
# This is a wrapper function and is not intended to be called directly by the user.
# See `is_other_antihtn_med` for usage examples.
```

`is_nsaid` *Non-steroidal anti-inflammatory drugs (NSAIDs)*

Description

This function checks if a given medication is an NSAID. This function processes multiple inputs efficiently.

Usage

```
is_nsaid(meucatc, npi_25b)
```

Arguments

`meucatc`	character ATC code of the medication.
`npi_25b`	integer Time when the medication was last taken.

Details

Identifies NSAIDs based on ATC codes starting with "M01A".

```
**Missing Data Codes:**
- `meucatc`: `9999996` (Not applicable), `9999997-9999999` (Missing)
- `npi_25b`: `6` (Not applicable), `7-9` (Missing)
```

Value

numeric 1 if medication is an NSAID, 0 otherwise. If inputs are invalid or out of bounds, the function returns a tagged NA.

Examples

```
# Scalar usage: Single respondent
is_nsaid("M01AB05", 1)
# Returns: 1

# Example: Respondent has non-response values for all inputs.
result <- is_nsaid("9999998", 8)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
is_nsaid(c("M01AB05", "A10BB09"), c(1, 3))
# Returns: c(1, 0)

# Database usage: Applied to survey datasets
library(dplyr)
cycle3_meds |>
```

```
  mutate(nsaid = is_nsaid(meucatc, npi_25b)) |>
  head()
```

`is_nsaid_med_cycles1to2` *Non-steroidal anti-inflammatory drugs (NSAIDs) - cycles 1-2*

Description

This function checks if a person is taking any NSAIDs based on the provided Anatomical Therapeutic Chemical (ATC) codes for medications and the Canadian Health Measures Survey (CHMS) response for the time when the medication was last taken.

Usage

```
is_nsaid_med_cycles1to2(
  atc_101a = NULL,
  atc_102a = NULL,
  atc_103a = NULL,
  atc_104a = NULL,
  atc_105a = NULL,
  atc_106a = NULL,
  atc_107a = NULL,
  atc_108a = NULL,
  atc_109a = NULL,
  atc_110a = NULL,
  atc_111a = NULL,
  atc_112a = NULL,
  atc_113a = NULL,
  atc_114a = NULL,
  atc_115a = NULL,
  atc_201a = NULL,
  atc_202a = NULL,
  atc_203a = NULL,
  atc_204a = NULL,
  atc_205a = NULL,
  atc_206a = NULL,
  atc_207a = NULL,
  atc_208a = NULL,
  atc_209a = NULL,
  atc_210a = NULL,
  atc_211a = NULL,
  atc_212a = NULL,
  atc_213a = NULL,
  atc_214a = NULL,
  atc_215a = NULL,
```

```
  atc_131a = NULL,
  atc_132a = NULL,
  atc_133a = NULL,
  atc_134a = NULL,
  atc_135a = NULL,
  atc_231a = NULL,
  atc_232a = NULL,
  atc_233a = NULL,
  atc_234a = NULL,
  atc_235a = NULL,
  mhr_101b = NULL,
  mhr_102b = NULL,
  mhr_103b = NULL,
  mhr_104b = NULL,
  mhr_105b = NULL,
  mhr_106b = NULL,
  mhr_107b = NULL,
  mhr_108b = NULL,
  mhr_109b = NULL,
  mhr_110b = NULL,
  mhr_111b = NULL,
  mhr_112b = NULL,
  mhr_113b = NULL,
  mhr_114b = NULL,
  mhr_115b = NULL,
  mhr_201b = NULL,
  mhr_202b = NULL,
  mhr_203b = NULL,
  mhr_204b = NULL,
  mhr_205b = NULL,
  mhr_206b = NULL,
  mhr_207b = NULL,
  mhr_208b = NULL,
  mhr_209b = NULL,
  mhr_210b = NULL,
  mhr_211b = NULL,
  mhr_212b = NULL,
  mhr_213b = NULL,
  mhr_214b = NULL,
  mhr_215b = NULL,
  mhr_131b = NULL,
  mhr_132b = NULL,
  mhr_133b = NULL,
  mhr_134b = NULL,
  mhr_135b = NULL,
  mhr_231b = NULL,
  mhr_232b = NULL,
  mhr_233b = NULL,
```

```
  mhr_234b = NULL,
  mhr_235b = NULL
)
```

Arguments

atc_101a	character ATC code of respondent's first prescription medication.
atc_102a	character ATC code of respondent's second prescription medication.
atc_103a	character ATC code of respondent's third prescription medication.
atc_104a	character ATC code of respondent's fourth prescription medication.
atc_105a	character ATC code of respondent's fifth prescription medication.
atc_106a	character ATC code of respondent's sixth prescription medication.
atc_107a	character ATC code of respondent's seventh prescription medication.
atc_108a	character ATC code of respondent's eighth prescription medication.
atc_109a	character ATC code of respondent's ninth prescription medication.
atc_110a	character ATC code of respondent's tenth prescription medication.
atc_111a	character ATC code of respondent's eleventh prescription medication.
atc_112a	character ATC code of respondent's twelfth prescription medication.
atc_113a	character ATC code of respondent's thirteenth prescription medication.
atc_114a	character ATC code of respondent's fourteenth prescription medication.
atc_115a	character ATC code of respondent's fifteenth prescription medication.
atc_201a	character ATC code of respondent's first over-the-counter medication.
atc_202a	character ATC code of respondent's second over-the-counter medication.
atc_203a	character ATC code of respondent's third over-the-counter medication.
atc_204a	character ATC code of respondent's fourth over-the-counter medication.
atc_205a	character ATC code of respondent's fifth over-the-counter medication.
atc_206a	character ATC code of respondent's sixth over-the-counter medication.
atc_207a	character ATC code of respondent's seventh over-the-counter medication.
atc_208a	character ATC code of respondent's eighth over-the-counter medication.
atc_209a	character ATC code of respondent's ninth over-the-counter medication.
atc_210a	character ATC code of respondent's tenth over-the-counter medication.
atc_211a	character ATC code of respondent's eleventh over-the-counter medication.
atc_212a	character ATC code of respondent's twelfth over-the-counter medication.
atc_213a	character ATC code of respondent's thirteenth over-the-counter medication.
atc_214a	character ATC code of respondent's fourteenth over-the-counter medication.
atc_215a	character ATC code of respondent's fifteenth over-the-counter medication.
atc_131a	character ATC code of respondent's first new prescription medication.
atc_132a	character ATC code of respondent's second new prescription medication.
atc_133a	character ATC code of respondent's third new prescription medication.

`atc_134a` character ATC code of respondent's fourth new prescription medication.

`atc_135a` character ATC code of respondent's fifth new prescription medication.

`atc_231a` character ATC code of respondent's first new over-the-counter medication.

`atc_232a` character ATC code of respondent's second new over-the-counter medication.

`atc_233a` character ATC code of respondent's third new over-the-counter medication.

`atc_234a` character ATC code of respondent's fourth new over-the-counter medication.

`atc_235a` character ATC code of respondent's fifth new over-the-counter medication.

`mhr_101b` integer Response for when the first prescription medication was last taken (1 = Today, . . . , 6 = Never).

`mhr_102b` integer Response for when the second prescription medication was last taken (1-6).

`mhr_103b` integer Response for when the third prescription medication was last taken (1-6).

`mhr_104b` integer Response for when the fourth prescription medication was last taken (1-6).

`mhr_105b` integer Response for when the fifth prescription medication was last taken (1-6).

`mhr_106b` integer Response for when the sixth prescription medication was last taken (1-6).

`mhr_107b` integer Response for when the seventh prescription medication was last taken (1-6).

`mhr_108b` integer Response for when the eighth prescription medication was last taken (1-6).

`mhr_109b` integer Response for when the ninth prescription medication was last taken (1-6).

`mhr_110b` integer Response for when the tenth prescription medication was last taken (1-6).

`mhr_111b` integer Response for when the eleventh prescription medication was last taken (1-6).

`mhr_112b` integer Response for when the twelfth prescription medication was last taken (1-6).

`mhr_113b` integer Response for when the thirteenth prescription medication was last taken (1-6).

`mhr_114b` integer Response for when the fourteenth prescription medication was last taken (1-6).

`mhr_115b` integer Response for when the fifteenth prescription medication was last taken (1-6).

`mhr_201b` integer Response for when the first over-the-counter medication was last taken (1-6).

`mhr_202b` integer Response for when the second over-the-counter medication was last taken (1-6).

`mhr_203b` integer Response for when the third over-the-counter medication was last taken (1-6).

`mhr_204b` integer Response for when the fourth over-the-counter medication was last taken (1-6).

`mhr_205b` integer Response for when the fifth over-the-counter medication was last taken (1-6).

`mhr_206b` integer Response for when the sixth over-the-counter medication was last taken (1-6).

`mhr_207b` integer Response for when the seventh over-the-counter medication was last taken (1-6).

`mhr_208b` integer Response for when the eighth over-the-counter medication was last taken (1-6).

`mhr_209b` integer Response for when the ninth over-the-counter medication was last taken (1-6).

`mhr_210b` integer Response for when the tenth over-the-counter medication was last taken (1-6).

`mhr_211b` integer Response for when the eleventh over-the-counter medication was last taken (1-6).

`mhr_212b` integer Response for when the twelfth over-the-counter medication was last taken (1-6).

`mhr_213b` integer Response for when the thirteenth over-the-counter medication was last taken (1-6).

`mhr_214b` integer Response for when the fourteenth over-the-counter medication was last taken (1-6).

`mhr_215b` integer Response for when the fifteenth over-the-counter medication was last taken (1-6).

`mhr_131b` integer Response for when the first new prescription medication was last taken (1-6).

`mhr_132b` integer Response for when the second new prescription medication was last taken (1-6).

`mhr_133b` integer Response for when the third new prescription medication was last taken (1-6).

`mhr_134b` integer Response for when the fourth new prescription medication was last taken (1-6).

`mhr_135b` integer Response for when the fifth new prescription medication was last taken (1-6).

`mhr_231b` integer Response for when the first new over-the-counter medication was last taken (1-6).

`mhr_232b` integer Response for when the second new over-the-counter medication was last taken (1-6).

`mhr_233b` integer Response for when the third new over-the-counter medication was last taken (1-6).

`mhr_234b` integer Response for when the fourth new over-the-counter medication was last taken (1-6).

`mhr_235b` integer Response for when the fifth new over-the-counter medication was last taken (1-6).

Details

The function identifies NSAIDs based on ATC codes starting with "M01A". It checks all medication variables provided in the input data frame.

```
  **Missing Data Codes:**
- The function handles tagged NAs from the `is_nsaid` function and propagates them.
```

Value

numeric Returns 1 if the person is taking any NSAIDs, 0 otherwise. If all medication information is missing, it returns a tagged NA.

See Also

`is_nsaid`

Examples

```
# This is a wrapper function and is not intended to be called directly by the user.
# See `is_nsaid` for usage examples.
```

---

is_other_antihtn_med *Other anti-hypertensive medications*

Description

This function checks if a given medication is another anti-hypertensive drug. This function processes multiple inputs efficiently.

Usage

```
is_other_antihtn_med(meucatc, npi_25b)
```

Arguments

`meucatc`	character ATC code of the medication.
`npi_25b`	integer Time when the medication was last taken.

Details

Identifies other anti-hypertensive drugs based on ATC codes starting with "C02", excluding a specific sub-code.

```
**Missing Data Codes:**
- `meucatc`: `9999996` (Not applicable), `9999997-9999999` (Missing)
- `npi_25b`: `6` (Not applicable), `7-9` (Missing)
```

Value

numeric 1 if medication is another anti-hypertensive drug, 0 otherwise. If inputs are invalid or out of bounds, the function returns a tagged NA.

Examples

```
# Scalar usage: Single respondent
is_other_antihtn_med("C02AC04", 3)
# Returns: 1

# Example: Respondent has non-response values for all inputs.
result <- is_other_antihtn_med("9999998", 8)
result # Shows: NA
haven::is_tagged_na(result, "b") # Shows: TRUE (confirms it's tagged NA(b))
format(result, tag = TRUE) # Shows: "NA(b)" (displays the tag)

# Multiple respondents
is_other_antihtn_med(c("C02AC04", "C02KX01"), c(3, 2))
# Returns: c(1, 0)

# Database usage: Applied to survey datasets
library(dplyr)
cycle3_meds |>
  mutate(other_antihtn = is_other_antihtn_med(meucatc, npi_25b)) |>
  head()
```

---

recode_after_meds *Recode variables that depend on derived medication variable inputs*

Description

Wraps `recodeflow::rec_with_table()` for use after derived medication variables (e.g., `any_htn_med`, `diab_med`) have been recoded and merged into the main cycle dataset. Use this instead of `rec_with_table()` when deriving variables whose inputs include derived medication variables: it automatically excludes medication-specific rows from `variable_details` so that pre-computed medication columns are passed through via the `copy` entries rather than re-derived from raw ATC/MHR columns.

Usage

```
recode_after_meds(
  data,
  variables,
  by = "clinicid",
  database_name = NULL,
  variable_details = chmsflow::variable_details
)
```

Arguments

data	data.frame Main cycle data with derived medication variables already merged.
variables	character Variable names to recode.
by	character Respondent identifier column. Default is "clinicid".
database_name	character Name of the database in variable_details. Defaults to the name of the data argument.
variable_details	data.frame Variable details table. Defaults to chmsflow::variable_details.

Value

data.frame Recoded data frame returned by recodeflow::rec_with_table().

See Also

recode_meds_cycles3to6(), recode_meds_cycles1to2()

Examples

```
cycle3 <- recode_meds_cycles3to6(cycle3, cycle3_meds, c("any_htn_med", "diab_med"))
cycle3_diab <- recode_after_meds(
  cycle3,
  c("lab_hba1", "diab_a1c", "diab_med", "ccc_51", "diab_status")
)
head(cycle3_diab[, c("clinicid", "diab_status")])
```

---

recode_meds_cycles1to2

Recode medication variables for cycles 1-2 (wide format)

Description

Recodes medication variables from cycles 1-2 wide-format data (one row per respondent, up to 80 ATC/MHR columns), and merges into the main cycle dataset. Wraps recodeflow::rec_with_table() and converts factor outputs to numeric.

Usage

```
recode_meds_cycles1to2(
  data,
  meds_data,
  variables,
  by = "clinicid",
  meds_database_name = NULL,
  variable_details = chmsflow::variable_details
)
```

Arguments

data	data.frame Main cycle data to merge medication variables into.
meds_data	data.frame Wide-format medication data (cycles 1-2). Must contain clinicid, ATC code columns atc_101a-atc_115a, atc_131a-atc_135a, atc_201a-atc_215a, atc_231a-atc_235a, and matching time-last-taken columns mhr_101b-mhr_115b, mhr_131b-mhr_135b, mhr_201b-mhr_215b, mhr_231b-mhr_235b. Column names are normalized to lowercase before recoding, so uppercase variants (e.g., CLINICID, ATC_101A) are accepted.
variables	character Medication variable names to derive (e.g., "any_htn_med").
by	character Respondent identifier column. Default is "clinicid".
meds_database_name	character Name of the meds database in variable_details. Defaults to the name of the meds_data argument. Override when passing a transformed object (e.g., head()).
variable_details	data.frame Variable details table. Defaults to chmsflow::variable_details.

Value

data.frame data with derived medication variables merged in as numeric columns.

See Also

recode_meds_cycles3to6(), aggregate_meds_by_person()

Examples

```
result <- recode_meds_cycles1to2(
  cycle1,
  cycle1_meds,
  c("any_htn_med", "diab_med")
)
head(result[, c("clinicid", "any_htn_med", "diab_med")])
```

---

recode_meds_cycles3to6

Recode medication variables for cycles 3-6 (long format)

Description

Recodes medication variables from cycles 3-6 long-format data (one row per medication per respondent), aggregates to one row per respondent, and merges into the main cycle dataset. Wraps recodeflow::rec_with_table() and aggregate_meds_by_person().

Usage

```
recode_meds_cycles3to6(
  data,
  meds_data,
  variables,
  by = "clinicid",
  meds_database_name = NULL,
  variable_details = chmsflow::variable_details
)
```

Arguments

data	data.frame Main cycle data to merge medication variables into.
meds_data	data.frame Long-format medication data (cycles 3-6) with columns clinicid, meucatc, and npi_25b.
variables	character Medication variable names to derive (e.g., "any_htn_med").
by	character Respondent identifier column. Default is "clinicid".
meds_database_name	character Name of the meds database in variable_details. Defaults to the name of the meds_data argument. Override when passing a transformed object (e.g., head()).
variable_details	data.frame Variable details table. Defaults to chmsflow::variable_details.

Value

data.frame data with derived medication variables merged in as numeric columns.

See Also

recode_meds_cycles1to2(), aggregate_meds_by_person()

Examples

```
result <- recode_meds_cycles3to6(
  cycle3,
  cycle3_meds,
  c("any_htn_med", "diab_med")
)
head(result[, c("clinicid", "any_htn_med", "diab_med")])
```

variable_details *variable-details.csv*

Description

This dataset provides details on how variables are recoded in chmsflow.

Usage

```
data(variable_details)
```

Source

See https://big-life-lab.github.io/chmsflow/articles/variable_details.html for more details.

Examples

```
data(variable_details)
str(variable_details)
```

variables *variables.csv*

Description

This dataset lists all the variables that are present in chmsflow.

Usage

```
data(variables)
```

Source

See https://big-life-lab.github.io/chmsflow/articles/variables_sheet.html for more details.

Examples

```
data(variables)
str(variables)
```

Index