

Package ‘inesss’

May 16, 2021

Title Institut National Excellence Sante Services Sociaux

Version 1.0.0

Description Cette librairie fournit des fonctionnalités pour une variété de tâches propices au domaine de la santé et des outils pour visualiser les résultats.

License GPL-3

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Imports askpass,
data.table,
DBI,
fs,
kableExtra,
knitr,
lubridate,
miniUI,
parallel,
odbc,
readxl,
rmarkdown,
rstudioapi,
shiny,
shinydashboard,
shinyFiles,
stringr,
testthat,
writexl

VignetteBuilder knitr

Depends R (>= 4.0.3)

R topics documented:

Charlson_Dx_CCI_INSPQ18	2
Charlson_Dx_UManitoba16	3
chunk_vec	4

CIM10	5
CIM9	5
CIM_correspond	6
Combine_Dx_CCI_INSPQ18	6
comorbidity	7
ComorbidityWeights	9
date_ymd	9
Elixhauser_Dx_CCI_INSPQ18	10
file_directory	11
I_APME_DEM_AUTOR_CRITR_ETEN_CM	12
Obstetrics_Dx	12
Pop_QC	13
query_naif_switch1	14
query_stat_gen1	17
replace_NA_in_dt	20
RLS_convert	20
RLS_list	21
RLS_tab_convert	21
rmNA	22
SQL_comorbidity	22
SQL_comorbidity_diagn	24
SQL_connexion	26
SQL_naif_switch1	26
SQL_obstetric	30
SQL_stats_SMED_NBR_JR_DUREE_TRAIT	31
SQL_stat_gen1	32
sunique	34
V_DEM_PAINT_MED_CM	35
V_DENOM_COMNE_MED	36
V_DES_COD	37
V_PRODU_MED	38

Index	39
--------------	-----------

Charlson_Dx_CCI_INSPQ18

Data - Codes diagnostics

Description

Codes SQL regex (se terminent par un '%') à utiliser lors de l'extraction des codes de diagnostics pour l'étude de la comorbidité.

Usage

```
data('Charlson_Dx_CCI_INSPQ18')
```

Format

```
list(Dx = list(CIM9,CIM10))
```

Details

L'attribut MaJ indique la dernière mise à jour ou la date de création du tableau.

aids : AIDS/HIV
 canc : Any tumor without metastasis
 cevd : Cerebrovascular disease
 chf : Congestive heart failure
 copd : Chronic pulmonary disease
 dementia : Dementia
 diab : Diabetes, complicated
 diabwc : Diabetes, uncomplicated
 ld : Liver disease
 metacanc : Metastatic cancer
 mi : Myocardial infarction
 para : Paralysis
 rend : Renal disease
 rheumd : Rheumatoid arth./collagen vascular disease
 ud : Ulcer disease
 valv : Valvular disease

Source

Validation of the Combined Comorbidity Index of Charlson and Elixhauser to Predict 30-Day Mortality Across ICD-9 and ICD-10. Voir PDF.

Charlson_Dx_UManitoba16

Data - Codes diagnostics

Description

Codes SQL regex (se terminent par un '%') à utiliser lors de l'extraction des codes de diagnostics pour l'étude de la comorbidité.

Usage

```
data('Charlson_Dx_UManitoba16')
```

Format

```
list(Dx = list(CIM9, CIM10))
```

Details

L'attribut MaJ indique la dernière mise à jour ou la date de création du tableau.

aids : HIV/AIDS
 canc : Cancer
 chf : Congestive Heart Failure
 cpd : Chronic Pulmonary Disease
 ctldr : Connective Tissue Disease - Rheumatic Disease

cvd : Cerebrovascular Disease
 dementia : Dementia
 diab : Diabetes with Chronic Complications
 diabwc : Diabetes without Chronic Complications
 ld1 : Mild Liver Disease
 ld2 : Moderate or Severe Liver Disease
 mc : Metastatic Carcinoma
 mi : Myocardial Infarction
 ph : Paraplegia and Hemiplegia
 pud : Peptic Ulcer Disease
 pvd : Peripheral Vascular Disease
 rd : Renal Disease

Source

CANCER DATA LINKAGE IN MANITOBA: EXPANDING THE INFRASTRUCTURE FOR RE-SEARCH page 72 du document.

chunk_vec

Tronquer un vecteur en plusieurs parties

Description

Divise le vecteur `x` en `n_chunks` parties ou le divise pour avoir au maximum `n_vals` valeurs dans chaque partie.

Usage

```
chunk_vec(x, n_chunks = NULL, n_vals = NULL)
```

Arguments

<code>x</code>	Vecteur à tronquer en plusieurs parties.
<code>n_chunks</code>	Diviser le vecteur en <code>n_chunks</code> parties.
<code>n_vals</code>	Chaque partie aura au maximum <code>n_vals</code> valeurs.

Details

Utiliser l'argument `n_chunks` ou `n_vals`, pas les deux.

Value

list ayant `n_chunks` éléments (ou `as.integer(length(x) / n_vals + 1L)`).

Examples

```

chunk_vec(x = 1:10, n_chunks = 3)
chunk_vec(x = 1:10, n_vals = 3)

```

CIM10*Data - Diagnostics CIM-10*

Description

Version légèrement modifiée par la RAMQ pour la facturation.

Usage

```
data('CIM10')
```

Format

Tableau de 2 variables et 15487 observations :

CODE Code de diagnostic CIM-10. character.

DIAGNOSTIC Description du code de diagnostic. character.

Source

[Répertoire des diagnostics.](#)

CIM9*Data - Diagnostics CIM-9*

Description

Version légèrement modifiée par la RAMQ pour la facturation.

Usage

```
data('CIM9')
```

Format

Tableau de 2 variables et 7184 observations :

CODE Code de diagnostic CIM-9. character.

DIAGNOSTIC Description du code de diagnostic. character.

Source

[Répertoire des diagnostics.](#)

CIM_correspond

Data - Correspondance entre CIM9 et CIM10

Description

Tableau de correspondance entre la CIM-9 et la CIM-10

Usage

```
data('CIM_correspond')
```

Format

Tableau de 4 variables et 25866 observations :

CIM9 Code de diagnostic CIM-9. character.

CIM9_DESC Description du code de diagnostic. character.

CIM10 Code de diagnostic CIM-10. character.

CIM10_DESC Description du code de diagnostic. character.

Source

[Répertoire des diagnostics.](#)

Combine_Dx_CCI_INSPQ18

Data - Codes diagnostics

Description

Codes SQL regex (se terminent par un '%') à utiliser lors de l'extraction des codes de diagnostics pour l'étude de la comorbidité.

Usage

```
data('Combine_Dx_CCI_INSPQ18')
```

Format

```
list(Dx = list(CIM9,CIM10))
```

Details

Contient les codes des datas Charlson_Dx_CCI_INSPQ18 et Elixhauser_Dx_CCI_INSPQ18.

L'attribut MaJ indique la dernière mise à jour ou la date de création du tableau.

aids : AIDS/HIV
alcohol : Alcohol abuse
blane : Blood loss anemia
canc : Any tumor without metastasis
carit : Cardiac arrhythmias
cevd : Cerebrovascular disease
chf : Congestive heart failure
coag : Coagulopathy
copd : Chronic pulmonary disease
dane : Deficiency anemia
dementia : Dementia
depre : Depression
diab : Diabetes, complicated
diabwc : Diabetes, uncomplicated
drug : Drug abuse
fed : Fluid and electrolyte disorders
hyp : Hypertension
hypothy : Hypothyroidism
ld : Liver disease
metacanc : Metastatic cancer
mi : Myocardial infarction
nd : Neurological disorders
obes : Obesity
para : Paralysis
pvd : Peripheral vascular disorders
psycho : Psychoses
pcd : Pulmonary circulation disorders
rend : Renal disease
rheumd : Rheumatoid arth./collagen vascular disease
ud : Ulcer disease
valv : Valvular disease
wloss : Weight loss

Source

Validation of the Combined Comorbidity Index of Charlson and Elixhauser to Predict 30-Day Mortality Across ICD-9 and ICD-10. Voir PDF.

comorbidity

Comorbidity

Description

Calcul des indicateurs de *Charlson*, *Elixhauser* et la combinaison des deux.

Usage

```
comorbidity(
  dt,
  ID,
  DIAGN,
  DATE_DX,
  SOURCE,
  n1 = 30,
  n2 = 730,
  Dx_table = "Comorbidity_Dx_CCI_INSPQ18",
  scores = "CCI_INSPQ_2018_CIM10",
  confirm_sourc = list(MEDECHO = 1, BDCU = 2, SMOD = 2),
  exclu_diagn = NULL,
  keep_confirm_data = FALSE
)
```

Arguments

dt	Dataset ayant au moins les quatre (4) colonnes ID, DIAGN, DATE_DX et SOURCE.
ID	Nom de la colonne indiquant le numéro de l'utilisateur, de l'individu.
DIAGN	Nom de la colonne indiquant le code d'un diagnostic. Voir <code>names(inesss::Comorbidity_diagn_codes)</code> .
DATE_DX	Nom de la colonne indiquant la date du diagnostic.
SOURCE	Nom de la colonne indiquant la provenance du diagnostic.
n1, n2	Nombre de jours dans le but de construire l'intervalle [n1,n2]. Pour qu'un code de diagnostic soit confirmé, il faut que <i>DIAGNi</i> soit suivi de <i>DIAGNj</i> (où $i < j$) et que le nombre de jours entre les deux soit dans l'intervalle [n1,n2].
Dx_table	Nom du dataset contenant la liste des codes de diagnostics à l'étude. <ul style="list-style-type: none"> 'Combine_Dx_CCI_INSPQ18' 'Charlson_Dx_CCI_INSPQ18' 'Elixhauser_Dx_CCI_INSPQ18' 'Charlson_Dx_UManitoba16'
scores	Nom de la table à utiliser pour le calcul des indicateurs. Voir les éléments de la liste <code>ComorbidityWeights</code> . <ul style="list-style-type: none"> 'CCI_INSPQ_2018_CIM9' 'CCI_INSPQ_2018_CIM10' 'UManitoba_2016'
confirm_sourc	list indiquant la <i>confiance</i> des SOURCE. Si une SOURCE doit être confirmée par une autre dans l'intervalle [n1,n2], inscrire 2, sinon 1. Inscrire les sources sous le format : <code>confirm_sourc = list(source1 = 1, source2 = 2, source3 = 2, ...)</code> . <code>confirm_sourc</code> doit contenir toutes les valeurs uniques de la colonne SOURCE.
exclu_diagn	Vecteur contenant le nom du ou des diagnostics à exclure de l'analyse. Voir la liste de <code>Dx_table</code> pour connaître les valeurs permises.
keep_confirm_data	TRUE ou FALSE. Place en attribut le data <code>confirm_data</code> qui indique la date de repérage et la date de confirmation d'un diagnostic.

Details

`confirm_sourc` : Dans l'exemple `confirm_sourc = list(source1=1, source2=2, source3=2, ...)`, la source3 pourrait confirmer la source2 et vice-versa.

Value

`data.table`

ComorbidityWeights	<i>Data - Poids des codes de diagnostics</i>
--------------------	--

Description

Data - Poids des codes de diagnostics

Usage

```
data('ComorbidityWeights')
```

Format

`list` contenant trois (3) tables indiquant la description, le code et le poids :

CCI_INSPQ_2018_CIM9

CCI_INSPQ_2018_CIM10

UManitoba_2016

Details

L'attribut `MaJ` indique la dernière mise à jour ou la date de création du tableau.

Source

Voir la source des datas [Combine_Dx_CCI_INSPQ18](#), [Charlson_Dx_CCI_INSPQ18](#), [Elixhauser_Dx_CCI_INSPQ18](#) et [Charlson_Dx_UManitoba16](#)

<code>date_ymd</code>	<i>Date</i>
-----------------------	-------------

Description

Retourne une date au format AAAA-MM-JJ. Utile dans des `for loop`, car `dd` peut prendre la valeur 'last' (au lieu d'un nombre), donc pas besoin de savoir si le dernier jour du mois est le 28 ou le 29 en février, ou un 30 ou un 31 pour les autres mois.

Usage

```
date_ymd(yyyy, mm, dd)
```

Arguments

yyyy	Nombre entier indiquant l'année.
mm	Nombre entier compris entre 1 et 12, où 1 indique janvier et 12 décembre.
dd	Nombre compris entre 1 et 31 selon les mois. Pour remplacer le dernier jour du mois (28, 29, 30, 31), il est possible d'inscrire dd = 'last'.

Value

lubridate::as_date

Examples

```
date_ymd(2020, 1, 15)
date_ymd(2020, 10, 31)
date_ymd(2020, 6, 'last')
for (yr in 1996:2004) {
  print(date_ymd(yyyy = yr, mm = 2, dd = 'last'))
}
```

Elixhauser_Dx_CCI_INSPQ18

Data - Codes diagnostics

Description

Codes SQL regex (se terminent par un '%') à utiliser lors de l'extraction des codes de diagnostics pour l'étude de la comorbidité.

Usage

```
data('Elixhauser_Dx_CCI_INSPQ18')
```

Format

```
list(Dx = list(CIM9, CIM10))
```

Details

L'attribut MaJ indique la dernière mise à jour ou la date de création du tableau.

aids : AIDS/HIV
 alcohol : Alcohol abuse
 blane : Blood loss anemia
 canc : Any tumor without metastasis
 carit : Cardiac arrhythmias
 chf : Congestive heart failure
 coag : Coagulopathy
 copd : Chronic pulmonary disease
 dane : Deficiency anemia
 depre : Depression
 diab : Diabetes, complicated

diabwc : Diabetes, uncomplicated
drug : Drug abuse
fed : Fluid and electrolyte disorders
hyp : Hypertension
hypothy : Hypothyroidism
ld : Liver disease
metacanc : Metastatic cancer
nd : Neurological disorders
obes : Obesity
para : Paralysis
pcd : Pulmonary circulation disorders
psycho : Psychoses
pvd : Peripheral vascular disorders
rend : Renal disease
rheumd : Rheumatoid arth./collagen vascular disease
ud : Ulcer disease
valv : Valvular disease
wloss : Weight loss

Source

Validation of the Combined Comorbidity Index of Charlson and Elixhauser to Predict 30-Day Mortality Across ICD-9 and ICD-10. Voir PDF.

file_directory	<i>Utils</i>
----------------	--------------

Description

Emplacement sur le disque dur où le script est sauvegardé.

Usage

file_directory()

Details

Si le script n'est pas sauvegardé, retourne NULL.

Value

CHR. Emplacement du dossier qui contient le script R.

I_APME_DEM_AUTOR_CRITR_ETEN_CM

Data - Demandes d'autorisation de Patient-Médicament d'exceptions.

Description

Data - Demandes d'autorisation de Patient-Médicament d'exceptions.

Usage

```
data('I_APME_DEM_AUTOR_CRITR_ETEN_CM')
```

Format

```
list
```

DES_COURT_INDCN_RECNU Valeurs uniques de la description courte complète de l'indication reconnue de PME.

- DES_COURT_INDCN_RECNU : Description courte complète de l'indication reconnue. character.
- DEBUT : Première année (APME_DAT_STA_DEM_PME) où la description courte complète a été inscrite. integer.
- FIN : Dernière année (APME_DAT_STA_DEM_PME) où la description courte complète a été inscrite. integer.

NO_SEQ_INDCN_RECNU_PME Indique la première et la dernière année d'utilisation.

- NO_SEQ_INDCN_RECNU : Numéro de séquence d'indication reconnue - PME. integer.
- DD_TRAIT_DEM : Date de début de traitement demandée. character.
- DF_TRAIT_DEM : Date de fin de traitement demandée. character.
- DD_AUTOR : Date de début de l'autorisation PME. character.
- DF_AUTOR : Date de fin de l'autorisation PME. character.
- DD_APLIC_AUTOR : Date de début de l'applicabilité de l'autorisation de PME. character.
- DF_APLIC_AUTOR : Date de fin de l'applicabilité de l'autorisation de PME. character.
- DAT_STA_DEM : Date de création ou de mise à jour du statut d'une demande d'autorisation correspondant à l'attribution du dernier statut de la demande. character.

Source

Dictionnaire EI

Obstetrics_Dx

Data - Codes diagnostics gestationnels

Description

Codes SQL regex (se terminent par un '%') à utiliser lors de l'extraction des codes de diagnostics gestationnels pour l'étude de la comorbidité.

Usage

```
data('Obstetrics_Dx')
```

Format

```
list(Dx = list(CIM9,CIM10))
```

Details

L'attribut MaJ indique la dernière mise à jour ou la date de création du tableau.

Pop_QC	<i>Data - Estimations et projections de population comparables (1996-2041)</i>
--------	--

Description

Tableau de la population québécoise par niveau géographique.

Ce fichier présente une série continue de données populationnelles comparables composée de la série des estimations (1996-2019) et de la série des projections (2020-2041) de population. Ces données tiennent compte de l'évolution de la population selon les plus récentes données observées de naissances, décès et mouvements migratoires.

Il est à noter que ces données de population sont présentées sur la base du découpage territorial du réseau de la santé et des services sociaux, soit pour les territoires suivants : le Québec, les réseaux universitaires intégrés de santé et de services sociaux (RUISSS), les régions sociosanitaires (RSS), les réseaux territoriaux de services (RTS), les réseaux locaux de services (RLS) et les centres locaux de services communautaires (CLSC).

Usage

```
data('Pop_QC')
```

Format

Tableau de 8 variables et 2 595 320 observations :

GEO Niveau géographique : Québec, RUISSS, RSS, RTS, RLS, CLSC. character.

CODE Code du territoire. integer.

AN Année. integer.

TYPE Type de données : Estimations ou Projections. character.

STATUT Donnée révisée ou provisoire. NA indique que la donnée n'a pas été changée depuis la dernière publication. character.

SEXE character.

AGE integer.

POP Population. integer.

Details

Attention AGE = 90 équivaut à *90 ans et plus*.

La classe des colonnes est `character` lorsque c'est du texte ou `integer` lorsque c'est un nombre.

Mise en ligne : 25 février 2016.

Dernière modification : 24 avril 2020.

Publication no : EstimProjComp-ISQ.

La fiche d'information et technique de cette base de données est disponible avec le fichier Excel (voir *Source*).

L'attribut `MaJ` indique la dernière mise à jour ou la date de création du tableau.

Source

MSSS Données de population.

Fichier Excel utilisé.

query_naif_switch1	<i>Code SQL</i>
--------------------	-----------------

Description

Générateur de code SQL pour la méthode `naif_switch1`.

Usage

```
query_naif_switch1(
  debut,
  fin,
  type_Rx = "DENOM",
  codes,
  group_by = "DENOM",
  type_Rx_retro = NULL,
  rx_retrospect_a_exclure = NULL,
  njours_sans_conso = 365,
  code_serv = c("1", "AD"),
  code_serv_filtre = "Exclusion",
  code_list = NULL,
  code_list_filtre = "Inclusion",
  age_date = NULL,
  ...
)
```

Arguments

<code>debut</code>	Date de début de la période d'étude au format AAAA-MM-JJ (une seule valeur).
<code>fin</code>	Date de fin de la période d'étude au format AAAA-MM-JJ (une seule valeur).

type_Rx	Type de code à analyser. Une valeur parmi : <ul style="list-style-type: none"> 'DENOM' : Code de dénomination commune (SMED_COD_DENOM_COMNE). 'DIN' : Code d'identification du médicament (SMED_COD_DIN).
codes	Le ou les codes à analyser. Voir <i>Details</i> .
group_by	Regrouper (aggréger) les résultats par : <ul style="list-style-type: none"> 'AHFS' : Résultats par code de classe AHFS. 'DENOM' : Résultats par code de dénomination commune. 'DIN' : Résultats par code d'identification du médicament. 'CodeList' : Résultats par code de catégories de liste de médicaments. 'CodeServ' : Résultats par code de service. 'Teneur' : Résultats par teneur du médicament. 'Format' : Résultats par format d'acquisition du médicament. 'Age' : Résultats par âge à une date précise. Voir argument age_date. L'âge est calculé à partir de la date de naissance disponible dans la vue V_FICH_ID_BEN_CM.
type_Rx_retro	Type de code à exclure. Si NULL, prend la valeur de type_Rx. Une valeur parmi : <ul style="list-style-type: none"> 'AHFS' : Code identifiant la classe de médicaments telle que déterminée par l'<i>American Hospital Formulary Service</i>. 'DENOM' : Code de dénomination commune (SMED_COD_DENOM_COMNE). 'DIN' : Code d'identification du médicament (SMED_COD_DIN).
rx_retrospect_a_exclure	Traitement(s) à inclure dans la période rétrospective. Voir <i>Details</i> . Un individu qui a au moins un traitement durant la période rétrospective ne sera pas considéré comme <i>naïf</i> ou <i>switch</i> .
njours_sans_conso	Nombre de jours qu'un individu ne doit pas avoir reçu de traitements avant sa date de référence (date index) pour être considéré <i>naïf</i> ou <i>switch</i> .
code_serv	Vecteur de type character comprenant le ou les codes de service (SMED_COD_SERV_1) à exclure ou à inclure, sinon inscrire NULL.
code_serv_filtre	'Inclusion' ou 'Exclusion' des codes de service code_serv. Inscrire code_serv = NULL s'il n'y a pas de filtre à appliquer.
code_list	Vecteur de type character comprenant le ou les codes de catégories de listes de médicaments (SMED_COD_CATG_LISTE_MED) à exclure ou à inclure, sinon inscrire NULL.
code_list_filtre	'Inclusion' ou 'Exclusion' des codes de catégories de liste de médicaments code_list. Inscrire code_list = NULL s'il n'y a pas de filtre à appliquer.
age_date	Date à laquelle on calcule l'âge si group_by contient 'Age'. Si NULL, aura pour valeur debut.

Details

rx_retrospect_a_exclure :

La période rétrospective est construite à partir des dates de références (date index) et de l'argument njours_sans_conso : [INDEX-njours_sans_conso; INDEX-1].

Inscrire les codes AHFS sous la forme de six (6) caractères. Par exemple, inscrire 040812 revient

à chercher la classe 04, la sous-classe 08 et la sous-sous-classe 12. Il est aussi possible de chercher seulement la classe AHFS 04 en inscrivant 04----. Puisque les deux premiers caractères indiquent la classe, ceux du milieu la sous-classe et les deux derniers la sous-sous-classe, remplacer une paire de caractères revient à rechercher toutes les classes (ou sous-classe, ou sous-sous-classe).

```
code_serv_filtre, code_list_filtre :
'Exclusion' inclus les NULL
'Inclusion' exclus les NULL.
```

Value

Chaîne de caractères à utiliser dans une requête SQL.

Examples

```
### Avantages d'utiliser cat()
# Sans cat()
query_naif_switch1(debut = '2018-01-01', fin = '2018-12-31',
                    type_Rx = 'DIN', codes = c(707503, 707600),
                    group_by = 'DIN')

# Avec cat()
cat(query_naif_switch1(debut = '2018-01-01', fin = '2018-12-31',
                       type_Rx = 'DIN', codes = c(707503, 707600),
                       group_by = 'DIN'))

### group_by
# Aucun
cat(query_naif_switch1(debut = '2018-01-01', fin = '2018-12-31',
                       type_Rx = 'DIN', codes = c(707503, 707600),
                       group_by = NULL))

# Tous les group_by
cat(query_naif_switch1(debut = '2018-01-01', fin = '2018-12-31',
                       type_Rx = 'DIN', codes = c(707503, 707600),
                       group_by = c("DENOM", "DIN", "CodeList", "CodeServ", "Teneur", "Format", "Age"))))

# Age a une date autre que debut
cat(query_naif_switch1(debut = '2018-01-01', fin = '2018-12-31',
                       type_Rx = 'DIN', codes = c(707503, 707600),
                       group_by = c('DENOM', 'Age'),
                       age_date = '2018-06-15'))

### DENOM
cat(query_naif_switch1(debut = '2018-01-01', fin = '2018-12-31',
                       type_Rx = 'DENOM', codes = c(39, 47092, 47135),
                       group_by = 'DENOM'))

cat(query_naif_switch1(debut = '2018-01-01', fin = '2018-12-31',
                       type_Rx = 'DENOM', codes = c(39, 47092, 47135),
                       group_by = c('DENOM', 'DIN'))))

### DIN
### Voir sections plus haut 1) Avantages d'utiliser cat() ou 2) group_by

### Exclusions Rx retrospectif
# AHFS
cat(query_naif_switch1(debut = '2018-01-01', fin = '2018-12-31',
                       type_Rx = 'DENOM', codes = c(47092, 47135),
                       group_by = 'DENOM',
```



```

        type_Rx_retro = 'AHFS', rx_retrospect_a_exclure = '040408'))
cat(query_naif_switch1(debut = '2018-01-01', fin = '2018-12-31',
        type_Rx = 'DENOM', codes = c(47092, 47135),
        group_by = 'DENOM',
        type_Rx_retro = 'AHFS',
        rx_retrospect_a_exclure = c('04----', '08--16', '122436'))))

# DENOM
cat(query_naif_switch1(debut = '2018-01-01', fin = '2018-12-31',
        type_Rx = 'DENOM', codes = c(47092, 47135),
        group_by = 'DENOM',
        type_Rx_retro = 'DENOM',
        rx_retrospect_a_exclure = c(47092, 47135, 47136)))

# DIN
cat(query_naif_switch1(debut = '2018-01-01', fin = '2018-12-31',
        type_Rx = 'DENOM', codes = 47092,
        group_by = 'DENOM',
        type_Rx_retro = 'DIN',
        rx_retrospect_a_exclure = c(2083523, 2084082, 2240331, 2453312)))

### Age
cat(query_naif_switch1(debut = '2018-01-01', fin = '2018-12-31',
        type_Rx = 'DENOM', codes = c(39, 47092, 47135),
        group_by = c('DENOM', 'DIN', 'Age'), age_date = '2018-01-01'))

### Exclusion VS Inclusion
cat(query_stat_gen1(debut = '2018-01-01', fin = '2018-12-31',
        type_Rx = 'DENOM', codes = c(47092, 47135), group_by = 'DENOM',
        code_serv_filtre = 'Exclusion', code_serv = c('1', 'AD'))))
cat(query_stat_gen1(debut = '2018-01-01', fin = '2018-12-31',
        type_Rx = 'DENOM', codes = c(47092, 47135), group_by = 'DENOM',
        code_serv_filtre = 'Inclusion', code_serv = c('1', 'AD'))))

```

query_stat_gen1

Code SQL

Description

Générateur de code SQL pour la méthode stat_gen1.

Usage

```

query_stat_gen1(
  debut,
  fin,
  type_Rx = "DENOM",
  codes,
  group_by = "DENOM",
  code_serv = c("1", "AD"),
  code_serv_filtre = "Exclusion",
  code_list = NULL,
  code_list_filtre = "Inclusion",
  age_date = NULL,
  ...
)

```

Arguments

debut	Date de début de la période d'étude au format AAAA-MM-JJ (une seule valeur).
fin	Date de fin de la période d'étude au format AAAA-MM-JJ (une seule valeur).
type_Rx	Typeype de code à analyser. Une valeur parmi : <ul style="list-style-type: none"> 'AHFS' : Code identifiant la classe de médicaments telle que déterminée par l'<i>American Hospital Formulary Service</i>. 'DENOM' : Code de dénomination commune (SMED_COD_DENOM_COMNE). 'DIN' : Code d'identification du médicament (SMED_COD_DIN).
codes	Le ou les codes à analyser. Voir <i>Details</i> .
group_by	Regrouper (aggréger) les résultats par : <ul style="list-style-type: none"> 'AHFS' : Résultats par code de classe AHFS. 'DENOM' : Résultats par code de dénomination commune. 'DIN' : Résultats par code d'identification du médicament. 'CodeList' : Résultats par code de catégories de liste de médicaments (SMED_COD_CATG_LISTE_MED). 'CodeServ' : Résultats par code de service (SMED_COD_SERV_1). 'Teneur' : Résultats par teneur du médicament (SMED_COD_TENR_MED) incluant les valeurs absentes. 'Format' : Résultats par format d'acquisition du médicament (SMED_COD_FORMA_ACQ_MED) incluant les valeurs absentes. 'Age' : Résultats par âge à une date précise. Voir argument <i>age_date</i>. L'âge est calculé à partir de la date de naissance disponible dans la vue V_FICH_ID_BEN_CM.
code_serv	Vecteur de type character comprenant le ou les codes de service (SMED_COD_SERV_1) à exclure ou à inclure, sinon inscrire NULL.
code_serv_filtre	'Inclusion' ou 'Exclusion' des codes de service code_serv. Inscrire code_serv = NULL s'il n'y a pas de filtre à appliquer.
code_list	Vecteur de type character comprenant le ou les codes de catégories de listes de médicaments (SMED_COD_CATG_LISTE_MED) à exclure ou à inclure, sinon inscrire NULL.
code_list_filtre	'Inclusion' ou 'Exclusion' des codes de catégories de liste de médicaments code_list. Inscrire code_list = NULL s'il n'y a pas de filtre à appliquer.
age_date	Date à laquelle on calcule l'âge si group_by contient 'Age'. Si NULL, aura pour valeur debut.

Details

codes :

Si type_Rx='AHFS' : codes sous la forme de 6 caractères où les deux premiers caractères représente la classe AHFS, les deux du milieu la sous-classe AHFS et les deux derniers la sous-sous-classe AHFS. Il est possible de remplacer une paire de caractères ({1, 2}, {3, 4} ou {5, 6}) par '--' pour rechercher toutes les types de classes. Par exemple, '04--12' indique qu'on recherche la classe AHFS 04, toutes les sous-classes AHFS et la sous-sous-classe 12.

Sinon inscrire les codes sous la forme d'un nombre entier.

code_serv_filtre, code_list_filtre :

'Exclusion' inclus les NULL

'Inclusion' exclus les NULL.

Value

Chaîne de caractères à utiliser dans une requête SQL.

Examples

```
### Avantages d'utiliser cat()
# Sans cat()
query_stat_gen1(debut = '2018-01-01', fin = '2018-12-31',
                type_Rx = 'DENOM', codes = c(39, 47092, 47135),
                group_by = 'DENOM')

# Avec cat()
cat(query_stat_gen1(debut = '2018-01-01', fin = '2018-12-31',
                    type_Rx = 'DENOM', codes = c(39, 47092, 47135),
                    group_by = 'DENOM'))

### group_by
# Aucun
cat(query_stat_gen1(debut = '2018-01-01', fin = '2018-12-31',
                    type_Rx = 'DENOM', codes = c(39, 47092, 47135),
                    group_by = NULL))

# Tous les group_by
cat(query_stat_gen1(debut = '2018-01-01', fin = '2018-12-31',
                    type_Rx = 'DENOM', codes = c(39, 47092, 47135),
                    group_by = c('AHFS', 'DENOM', 'DIN', 'CodeList', 'CodeServ', 'Teneur', 'Format', 'Age'))))

### AHFS
cat(query_stat_gen1(debut = '2018-01-01', fin = '2018-12-31',
                    type_Rx = 'AHFS', codes = c('040412', '08----'),
                    group_by = 'AHFS'))
cat(query_stat_gen1(debut = '2018-01-01', fin = '2018-12-31',
                    type_Rx = 'AHFS', codes = '04--12',
                    group_by = 'AHFS'))
cat(query_stat_gen1(debut = '2018-01-01', fin = '2018-12-31',
                    type_Rx = 'AHFS', codes = '04--12',
                    group_by = c('AHFS', 'DENOM'))))

### DENOM
cat(query_stat_gen1(debut = '2018-01-01', fin = '2018-12-31',
                    type_Rx = 'DENOM', codes = c(39, 47092, 47135),
                    group_by = 'DENOM'))
cat(query_stat_gen1(debut = '2018-01-01', fin = '2018-12-31',
                    type_Rx = 'DENOM', codes = c(39, 47092, 47135),
                    group_by = c('DENOM', 'DIN'))))

### DIN
cat(query_stat_gen1(debut = '2018-01-01', fin = '2018-12-31',
                    type_Rx = 'DIN', codes = c(30848, 585092),
                    group_by = 'DIN'))

### Age
cat(query_stat_gen1(debut = '2018-01-01', fin = '2018-12-31',
                    type_Rx = 'DIN', codes = c(30848, 585092),
                    group_by = c('DIN', 'Age'), age_date = '2018-01-01'))

### Exclusion et Inclusion - code_serv et code_list
cat(query_stat_gen1(debut = '2018-01-01', fin = '2018-12-31',
```

```

type_Rx = 'DENOM', codes = c(47092, 47135),
group_by = 'DENOM',
code_serv_filtre = 'Exclusion', code_serv = c('1', 'AD'),
code_list_filtre = 'Inclusion', code_list = c('40', '41'))

```

replace_NA_in_dt	<i>Utils</i>
------------------	--------------

Description

Remplace les NAs dans un tableau par by.

Usage

```
replace_NA_in_dt(dt, by)
```

Arguments

dt	Tableau contenant des NAs.
by	Valeur de remplacement.

RLS_convert	<i>Conversion RLS</i>
-------------	-----------------------

Description

Le projet de loi n°10 a modifié la plupart des codes RLS (voir la table RLS_tab_convert). Cette fonction permet de convertir les RLS d'une table si elle contient des codes qui existaient avant la loi 10.

Usage

```
RLS_convert(dt, rls_colname)
```

Arguments

dt	Table pouvant contenir des RLS à convertir.
rls_colname	Nom de la colonne contenant les codes de RLS.

Value

data.table

Examples

```

dt = data.frame(id = 1:5,
                age = c(45, 65, 78, 15, 35),
                sexe = c("F", "M", "F", "F", "M"),
                rls = c(1204, 215, 1611, 1503, 1610))
rls_colname = "rls"
dt_convert <- RLS_convert(dt, rls_colname = "rls")

```

RLS_list	<i>Data - Liste des RLS</i>
----------	-----------------------------

Description

Vecteur contenant la liste des 93 RLS plus 3 valeurs utiles lors d'analyse : 1001, 1701, 1801.

Usage

```
data('RLS_list')
```

Format

Vecteur integer de 96 nombres.

Details

L'attribut MaJ indique la dernière mise à jour ou la date de création du tableau.

RLS_tab_convert	<i>Data - Correspondance RLS Loi 10</i>
-----------------	---

Description

Établir la correspondance des RLS avant et après l'adoption de la loi 10.

Usage

```
data('RLS_tab_convert')
```

Format

Tableau de 2 variables et 84 observations :

RLS14 Code de RLS **avant** l'adoption de la loi 10. integer.

RLS15 Code de RLS **après** l'adoption de la loi 10. integer.

Details

Certains RLS ne peuvent être convertis, car leur valeur se retrouve avant et après l'adoption de la loi 10.

`attr(RLS_tab_convert, "RLS_exclus")` indique les quatre (4) RLS exclus : 611, 612, 1611, 1612.

`attr(RLS_tab_convert, "RLS_exclus_value")` renvoie un tableau indiquant les valeurs avant et après l'adoption de la loi 10 pour ces quatre (4) RLS.

L'attribut MaJ indique la dernière mise à jour ou la date de création du tableau.

Source

Correspondance Etablissement Public Loi 10.

Fichier Excel utilisé.

rmNA

*Utils***Description**

Supprime les NAs du vecteur. Renvoie NULL si aucune valeur.

Usage

```
rmNA(x)
```

Arguments

x Vecteur.

Examples

```
rmNA(c(4, 6, 8, NA, 78, 4, NaN))
```

SQL_comorbidity

*Comorbidity***Description**

Extraction des codes de diagnostics CIM pour ensuite calculer les indicateurs de Charlson et Elixhauser.

Usage

```
SQL_comorbidity(
  conn,
  dt,
  ID,
  DATE_INDEX,
  Dx_table = "Combine_Dx_CCI_INSPQ18",
  CIM = c("CIM9", "CIM10"),
  scores = "CCI_INSPQ_2018_CIM10",
  lookup = 2,
  n1 = 30,
  n2 = 730,
  dt_source = c("V_DIAGN_SEJ_HOSP_CM", "V_SEJ_SERV_HOSP_CM", "V_EPISO_SOIN_DURG_CM",
    "I_SMOD_SERV_MD_CM"),
  dt_desc = list(V_DIAGN_SEJ_HOSP_CM = "MEDECHO", V_SEJ_SERV_HOSP_CM = "MEDECHO",
    V_EPISO_SOIN_DURG_CM = "BDCU", I_SMOD_SERV_MD_CM = "SMOD"),
  confirm_sourc = list(MEDECHO = 1, BDCU = 2, SMOD = 2),
  obstetric_exclu = TRUE,
  exclu_diagn = NULL,
  verbose = TRUE,
  keep_confirm_data = FALSE
)
```

Arguments

conn	Variable contenant la connexion entre R et Teradata. Voir SQL_connexion .
dt	Tableau ayant au moins deux colonnes : ID et DATE_INDEX.
ID	Nom de la colonne contenant l'identifiant unique de l'utilisateur.
DATE_INDEX	Nom de la colonne contenant la date index de chaque usager.
Dx_table	Nom du dataset contenant la liste des codes de diagnostics à l'étude. <ul style="list-style-type: none"> • 'Combine_Dx_CCI_INSPQ18' • 'Charlson_Dx_CCI_INSPQ18' • 'Elixhauser_Dx_CCI_INSPQ18' • 'Charlson_Dx_UManitoba16'
CIM	'CIM9', 'CIM10' ou les deux. Permet de filtrer les codes de diagnostics selon le numéro de révision de la <i>Classification statistique internationale des maladies et des problèmes de santé connexes</i> (CIM).
scores	Nom de la table à utiliser pour le calcul des indicateurs. Voir les éléments de la liste ComorbidityWeights. <ul style="list-style-type: none"> • 'CCI_INSPQ_2018_CIM9' • 'CCI_INSPQ_2018_CIM10' • 'UManitoba_2016'
lookup	Nombre entier. Années à analyser avant la date indexe de chaque individu.
n1	Nombre de jours dans le but de construire l'intervalle [n1,n2]. Pour qu'un code de diagnostic soit confirmé, il faut que <i>DIAGNi</i> soit suivi de <i>DIAGNj</i> (où $i < j$) et que le nombre de jours entre les deux soit dans l'intervalle [n1,n2].
n2	Nombre de jours dans le but de construire l'intervalle [n1,n2]. Pour qu'un code de diagnostic soit confirmé, il faut que <i>DIAGNi</i> soit suivi de <i>DIAGNj</i> (où $i < j$) et que le nombre de jours entre les deux soit dans l'intervalle [n1,n2].
dt_source	Vecteur comprenant la ou les bases de données où aller chercher l'information. Voir <i>Details</i> .
dt_desc	list décrivant les bases de données demandées dans dt_source au format list(BD = 'MaDescription'). Voir <i>Details</i> .
confirm_sourc	list indiquant la <i>confiance</i> des SOURCE. Si une SOURCE doit être confirmée par une autre dans l'intervalle [n1,n2], inscrire 2, sinon 1. Inscrive les sources sous le format : confirm_sourc = list(source1 = 1, source2 = 2, source3 = 2, ...). confirm_sourc doit contenir toutes les valeurs uniques de la colonne SOURCE.
obstetric_exclu	TRUE ou FALSE. Si l'on doit exclure (TRUE) les diabètes et les hypertension de type gestationnel. Voir <i>Détails</i> .
exclu_diagn	Vecteur contenant le nom du ou des diagnostics à exclure de l'analyse. Voir la liste de Dx_table pour connaître les valeurs permises.
verbose	TRUE ou FALSE. Affiche le temps qui a été nécessaire pour extraire les diagnostics d'une source (dt_source). Utile pour suivre le déroulement de l'extraction.
keep_confirm_data	TRUE ou FALSE. Place en attribut le data confirm_data qui indique la date de repérage et la date de confirmation d'un diagnostic.

Details

dt : Si un ID a plus d'une date index, seule la première, la plus ancienne, sera conservée.

obstetric_exclu : Lorsqu'un cas de diabète ou d'hypertension a lieu 120 jours avant ou 180 jours après un évènement obstétrique, on les considère de type gestationnel. Ces cas sont alors exclus de l'analyse.

dt_source :

- **V_DIAGN_SEJ_HOSP_CM** : Cette structure contient tous les diagnostics associés à un séjour hospitalier.
- **V_SEJ_SERV_HOSP_CM** : Cette structure contient les séjours dans un service effectués par l'individu hospitalisé.
- **V_EPISO_SOIN_DURG_CM** : Cette structure contient les épisodes de soins des départements d'urgence de la province.
- **I_SMOD_SERV_MD_CM** : Cette vue retourne différentes informations se rapportant aux Services rendus à l'acte par des médecins.

Value

data.table :

- ID : Colonne contenant l'identifiant unique de l'utilisateur.
- Charlson : Indicateur, seulement si method contient 'Charlson'.
- Elixhauser : Indicateur, seulement si method contient 'Elixhauser'.
- Combined : Indicateur, seulement si method contient 'Charlson' et 'Elixhauser'.
- Tous les diagnostics ainsi que leur poids (score).

SQL_comorbidity_diagn *Extraction - Codes diagn comorbidity*

Description

Extraction SQL des diagnostics pour l'étude de la comorbidité.

Usage

```
SQL_comorbidity_diagn(
  conn,
  cohort,
  debut,
  fin,
  Dx_table = "Combine_Dx_CCI_INSPQ18",
  CIM = c("CIM9", "CIM10"),
  dt_source = c("V_DIAGN_SEJ_HOSP_CM", "V_SEJ_SERV_HOSP_CM", "V_EPISO_SOIN_DURG_CM",
    "I_SMOD_SERV_MD_CM"),
  dt_desc = list(V_DIAGN_SEJ_HOSP_CM = "MEDECHO", V_SEJ_SERV_HOSP_CM = "MEDECHO",
    V_EPISO_SOIN_DURG_CM = "BDCU", I_SMOD_SERV_MD_CM = "SMOD"),
  exclu_diagn = NULL,
  verbose = TRUE
)
```


Arguments

conn	Variable contenant la connexion entre R et Teradata. Voir SQL_connexion .
cohort	Cohorte d'étude. Vecteur comprenant les numéros d'identification des individus à conserver.
debut	Date de début de la période d'étude au format AAAA-MM-JJ.
fin	Date de fin de la période d'étude au format AAAA-MM-JJ.
Dx_table	Nom du dataset contenant la liste des codes de diagnostics à l'étude. <ul style="list-style-type: none"> • 'Combine_Dx_CCI_INSPQ18' • 'Charlson_Dx_CCI_INSPQ18' • 'Elixhauser_Dx_CCI_INSPQ18' • 'Charlson_Dx_UManitoba16'
CIM	'CIM9', 'CIM10' ou les deux. Permet de filtrer les codes de diagnostics selon le numéro de révision de la <i>Classification statistique internationale des maladies et des problèmes de santé connexes</i> (CIM).
dt_source	Vecteur comprenant la ou les bases de données où aller chercher l'information. Voir <i>Details</i> .
dt_desc	list décrivant les bases de données demandées dans dt_source au format list(BD = 'MaDescription'). Voir <i>Details</i> .
exclu_diagn	Vecteur contenant le nom du ou des diagnostics à exclure de l'analyse. Voir la liste de Dx_table pour connaître les valeurs permises.
verbose	TRUE ou FALSE. Affiche le temps qui a été nécessaire pour extraire les diagnostics d'une source (dt_source). Utile pour suivre le déroulement de l'extraction.

Details

dt_source :

- **V_DIAGN_SEJ_HOSP_CM** : Cette structure contient tous les diagnostics associés à un séjour hospitalier.
- **V_SEJ_SERV_HOSP_CM** : Cette structure contient les séjours dans un service effectués par l'individu hospitalisé.
- **V_EPISO_SOIN_DURG_CM** : Cette structure contient les épisodes de soins des départements d'urgence de la province.
- **I_SMOD_SERV_MD_CM** : Cette vue retourne différentes informations se rapportant aux Services rendus à l'acte par des médecins.

Value

data.table de 4 variables :

- ID : Numéro d'identification de l'utilisateur.
- DATE_DX : Date de diagnostic.
- DIAGN : Code descriptif des diagnostics provenant de diagn_codes.
- SOURCE : Indique d'où provient l'information. Une valeur parmi dt_source.

SQL_connexion	<i>Connexion Teradata</i>
---------------	---------------------------

Description

Connexion entre R et SQL Teradata.

Usage

```
SQL_connexion(uid, pwd = NULL, dsn = "PEI_PRD", encoding = "latin1")
```

Arguments

uid	Identifiant.
pwd	Mot de passe. Si NULL, le mot de passe est demandé lors de l'exécution.
dsn	Data Source Name. Par défaut 'PEI_PRD'.
encoding	'latin1' ou 'UTF-8'. Encodage de la base de données. Par défaut 'latin1'.

Value

Connexion Teradata, sinon NULL.

Examples

```
## Not run:
conn <- SQL_connexion('abc007')
conn <- SQL_connexion(uid = 'abc007', pwd = 'MonMotDePasse', dsn = 'PEI_PRD')

## End(Not run)
```

SQL_naif_switch1	<i>Naïfs et Switchs</i>
------------------	-------------------------

Description

Statistiques générales pour un ou des médicaments à partir d'une cohorte consommant ce(s) médicament(s) pour la première fois.

Un individu est considéré *naïf* lorsqu'il a un traitement pour la première fois et qu'il n'a jamais eu d'autres traitements *de la même famille*.

Un individu est considéré *switch* lorsqu'il a un traitement pour la première fois, mais qu'il a eu un autre traitement dans le passé appartenant à *la même famille*.

Vue utilisée : **V_DEM_PAINT_MED_CM**.

Usage

```
SQL_naif_switch1(
  conn = NULL,
  debut,
  fin,
  type_Rx = "DENOM",
  codes,
  group_by = "DENOM",
  type_Rx_retro = NULL,
  rx_retrospect_a_exclure = NULL,
  njours_sans_conso = 365,
  code_serv = c("1", "AD"),
  code_serv_filtre = "Exclusion",
  code_list = NULL,
  code_list_filtre = "Inclusion",
  age_date = NULL,
  ...
)
```

Arguments

conn	Variable contenant la connexion entre R et Teradata. Voir SQL_connexion .
debut	Date de début de la période d'étude au format AAAA-MM-JJ (une seule valeur).
fin	Date de fin de la période d'étude au format AAAA-MM-JJ (une seule valeur).
type_Rx	Type de code à analyser. Une valeur parmi : <ul style="list-style-type: none"> 'DENOM' : Code de dénomination commune (SMED_COD_DENOM_COMNE). 'DIN' : Code d'identification du médicament (SMED_COD_DIN).
codes	Le ou les codes à analyser. Voir <i>Details</i> .
group_by	Regrouper (aggréger) les résultats par : <ul style="list-style-type: none"> 'AHFS' : Résultats par code de classe AHFS. 'DENOM' : Résultats par code de dénomination commune. 'DIN' : Résultats par code d'identification du médicament. 'CodeList' : Résultats par code de catégories de liste de médicaments. 'CodeServ' : Résultats par code de service. 'Teneur' : Résultats par teneur du médicament. 'Format' : Résultats par format d'acquisition du médicament. 'Age' : Résultats par âge à une date précise. Voir argument <code>age_date</code>. L'âge est calculé à partir de la date de naissance disponible dans la vue <code>V_FICH_ID_BEN_CM</code>.
type_Rx_retro	Type de code à exclure. Si NULL, prend la valeur de <code>type_Rx</code> . Une valeur parmi : <ul style="list-style-type: none"> 'AHFS' : Code identifiant la classe de médicaments telle que déterminée par l'<i>American Hospital Formulary Service</i>. 'DENOM' : Code de dénomination commune (SMED_COD_DENOM_COMNE). 'DIN' : Code d'identification du médicament (SMED_COD_DIN).
rx_retrospect_a_exclure	Traitement(s) à inclure dans la période rétrospective. Voir <i>Details</i> . Un individu qui a au moins un traitement durant la période rétrospective ne sera pas considéré comme <i>naïf</i> ou <i>switch</i> .

njours_sans_conso	Nombre de jours qu'un individu ne doit pas avoir reçu de traitements avant sa date de référence (date index) pour être considéré <i>naïf</i> ou <i>switch</i> .
code_serv	Vecteur de type character comprenant le ou les codes de service (SMED_COD_SERV_1) à exclure ou à inclure, sinon inscrire NULL.
code_serv_filtre	'Inclusion' ou 'Exclusion' des codes de service code_serv. Inscrire code_serv = NULL s'il n'y a pas de filtre à appliquer.
code_list	Vecteur de type character comprenant le ou les codes de catégories de listes de médicaments (SMED_COD_CATG_LISTE_MED) à exclure ou à inclure, sinon inscrire NULL.
code_list_filtre	'Inclusion' ou 'Exclusion' des codes de catégories de liste de médicaments code_list. Inscrire code_list = NULL s'il n'y a pas de filtre à appliquer.
age_date	Date à laquelle on calcule l'âge si group_by contient 'Age'. Si NULL, aura pour valeur debut.

Details

rx_retrospect_a_exclure :

La période rétrospective est construite à partir des dates de références (index) et de l'argument `njours_sans_conso` : `[INDEX -njours_sans_conso; INDEX -1]`.

code_serv_filtre, code_list_filtre :

'Exclusion' inclus les NULL

'Inclusion' exclus les NULL.

Value

`data.table`

- **DATE_DEBUT** : Indique la ou les dates de début de la période d'étude.
- **DATE_FIN** : Indique la ou les dates de fin de la période d'étude.
- **AHFS_CLA** : Seulement si `group_by` contient 'AHFS'. Code de la classe AHFS.
- **AHFS_SCLA** : Seulement si `group_by` contient 'AHFS'. Code de la sous-classe AHFS.
- **AHFS_SSCLA** : Seulement si `group_by` contient 'AHFS'. Code de la sous-sous-classe AHFS.
- **NOM_AHFS** : Seulement si `group_by` contient 'AHFS'. Nom de la classe AHFS.
- **DENOM** : Seulement si `group_by` contient 'DENOM'. Code de dénomination commune.
- **NOM_DENOM** : Seulement si `group_by` contient 'DENOM'. Nom de la dénomination commune.
- **DIN** : Seulement si `group_by` contient 'DIN'. Code d'identification du médicament.
- **NOM_MARQ_COMRC** : Seulement si `group_by` contient 'DIN'. Nom de la marque commerciale.
- **CODE_SERV** : Seulement si `group_by` contient 'CodeServ'. Code de service,
- ****CODE_LIST** : Seulement si `group_by` contient 'CodeList'. Code de catégorie de listes de médicaments.
- **TENEUR** : Seulement si `group_by` contient 'Teneur'. Teneur du médicament.
- **FORMAT_ACQ** : Seulement si `group_by` contient 'Format'. Format d'acquisition du médicament.
- **AGE** : Seulement si `group_by` contient 'Age'. Age de l'individu à la date `age_date`.

- MNT_MED : Montant autorisé par la RAMQ pour le médicament ou le produit. Il comprend la part du grossiste (s'il y a lieu) et la part du manufacturier. Voir la variable SMED_MNT_AUTOR_MED.
- MNT_SERV : Montant de frais de service autorisé par la RAMQ à la date du service. Voir la variable SMED_MNT_AUTOR_FRAIS_SERV.
- MNT_TOT : Somme des variables MNT_MED et MNT_SERV.
- COHORTE : Nombre d'individus unique.
- NBRE_RX : Nombre de demandes de paiement.
- QTE_MED : Quantité totale des médicaments ou des fournitures dispensés. Voir la variable SMED_QTE_MED.
- DUREE_TX : Durée de traitement totale des prescriptions en jours. Voir la variable SMED_NBR_JR_DUREE_TRAIT.

Examples

```
## Not run:
conn <- SQL_connexion(askpass::askpass('Utilisateur :'), askpass::askpass('Mot de passe :'))

### group_by
# Aucun
ex01 <- SQL_naif_switch1(
  conn, debut = c('2018-01-01', '2019-01-01'), fin = c('2018-12-31', '2019-12-31'),
  type_Rx = 'DENOM', codes = c(39, 47092, 47135), group_by = NULL
)
# Tous les group_by
ex02 <- SQL_naif_switch1(
  conn, debut = c('2018-01-01', '2019-01-01'), fin = c('2018-12-31', '2019-12-31'),
  type_Rx = 'DENOM', codes = c(39, 47092, 47135),
  group_by = c('AHFS', 'DENOM', 'DIN', 'CodeList', 'CodeServ', 'Teneur', 'Format', 'Age')
)

### DENOM
ex03 <- SQL_naif_switch1(
  conn, debut = c('2018-01-01', '2019-01-01'), fin = c('2018-12-31', '2019-12-31'),
  type_Rx = 'DENOM', codes = c(39, 47092, 47135), group_by = 'DENOM'
)

### DIN
ex04 <- SQL_naif_switch1(
  conn, debut = c('2018-01-01', '2019-01-01'), fin = c('2018-12-31', '2019-12-31'),
  type_Rx = 'DIN', codes = c(30848, 585092), group_by = 'DIN'
)

### Exclusions Rx retrospectif
# AHFS
ex05 <- SQL_naif_switch1(
  conn, debut = c('2018-01-01', '2019-01-01'), fin = c('2018-12-31', '2019-12-31'),
  type_Rx = 'DENOM', codes = c(47092, 47135), group_by = 'DENOM',
  type_Rx_retro = 'AHFS', rx_retrospect_a_exclure = c('04---', '08--16', '122436')
)

# DENOM
ex06 <- SQL_naif_switch1(
  conn, debut = c('2018-01-01', '2019-01-01'), fin = c('2018-12-31', '2019-12-31'),
  type_Rx = 'DENOM', codes = c(47092, 47135), group_by = 'DENOM',
  type_Rx_retro = 'DENOM', rx_retrospect_a_exclure = c(47092, 47135, 47136)
```

```

)
# DIN
ex07 <- SQL_naif_switch1(
  conn, debut = c('2018-01-01', '2019-01-01'), fin = c('2018-12-31', '2019-12-31'),
  type_Rx = 'DENOM', codes = 47092, group_by = c('DENOM', 'DIN'),
  type_Rx_retro = 'DIN', rx_retrospect_a_exclure = c(2083523, 2084082, 2240331, 2453312)
)

### Age
ex08 <- SQL_naif_switch1(
  conn, debut = c('2018-01-01', '2019-01-01'), fin = c('2018-12-31', '2019-12-31'),
  type_Rx = 'DIN', codes = c(30848, 585092), group_by = c('DIN', 'Age'), age_date = '2018-06-05'
)

### Exclusion VS Inclusion
ex09 <- SQL_naif_switch1(
  conn, debut = c('2018-01-01', '2019-01-01'), fin = c('2018-12-31', '2019-12-31'),
  type_Rx = 'DENOM', codes = c(39, 47092, 47135), group_by = 'DENOM',
  code_serv = c('1', 'AD'), code_serv_filtre = 'Exclusion',
  code_list = c('40', '41'), code_list_filtre = 'Inclusion'
)

## End(Not run)

```

SQL_obstetric

Extraction - Codes diagn obstetriques

Description

Extraction des événements obstétriques.

Usage

```

SQL_obstetric(
  conn,
  cohort,
  debut,
  fin,
  CIM = c("CIM9", "CIM10"),
  dt_source = c("V_DIAGN_SEJ_HOSP_CM", "V_SEJ_SERV_HOSP_CM", "V_EPISO_SOIN_DURG_CM",
    "I_SMOD_SERV_MD_CM"),
  dt_desc = list(V_DIAGN_SEJ_HOSP_CM = "MED-ECHO", V_SEJ_SERV_HOSP_CM = "MED-ECHO",
    V_EPISO_SOIN_DURG_CM = "BDCU", I_SMOD_SERV_MD_CM = "SMOD"),
  verbose = TRUE
)

```

Arguments

conn	Variable contenant la connexion entre R et Teradata. Voir SQL_connexion .
cohort	Cohorte d'étude. Vecteur comprenant les numéros d'identification des individus à conserver.
debut	Date de début de la période d'étude au format AAAA-MM-JJ.

fin	Date de fin de la période d'étude au format AAAA-MM-JJ.
CIM	'CIM9', 'CIM10' ou les deux. Permet de filtrer les codes de diagnostics selon le numéro de révision de la <i>Classification statistique internationale des maladies et des problèmes de santé connexes</i> (CIM).
dt_source	Vecteur comprenant la ou les bases de données où aller chercher l'information. Voir <i>Details</i> .
dt_desc	list décrivant les bases de données demandées dans dt_source au format list(BD = 'MaDescription'). Voir <i>Details</i> .
verbose	TRUE ou FALSE. Affiche le temps qui a été nécessaire pour extraire les diagnostics d'une source (dt_source). Utile pour suivre le déroulement de l'extraction.

Details

dt_source :

- **V_DIAGN_SEJ_HOSP_CM** : Cette structure contient tous les diagnostics associés à un séjour hospitalier.
- **V_SEJ_SERV_HOSP_CM** : Cette structure contient les séjours dans un service effectués par l'individu hospitalisé.
- **V_EPISO_SOIN_DURG_CM** : Cette structure contient les épisodes de soins des départements d'urgence de la province.
- **I_SMOD_SERV_MD_CM** : Cette vue retourne différentes informations se rapportant aux Services rendus à l'acte par des médecins.

SQL_stats_SMED_NBR_JR_DUREE_TRAIT
Statistiques

Description

Statistiques

Usage

```
SQL_stats_SMED_NBR_JR_DUREE_TRAIT(
  conn,
  debut,
  fin,
  by_code_serv = TRUE,
  include_dureeTx_0 = FALSE
)
```

Arguments

conn	Variable contenant la connexion entre R et Teradata. Voir SQL_connexion .
debut	Date de début de la période d'étude au format AAAA-MM-JJ.
fin	Date de fin de la période d'étude au format AAAA-MM-JJ.
by_code_serv	TRUE ou FALSE. Grouper les résultats par code de services. Par défaut TRUE.
include_dureeTx_0	TRUE ou FALSE. Inclure les durées de traitements égale à zéro. Par défaut FALSE.

SQL_stat_gen1

Statistiques générales

Description

Statistiques d'un ou de plusieurs codes de médicaments selon certains critères.

Vue utilisée : [V_DEM_PAINT_MED_CM](#).

Usage

```
SQL_stat_gen1(
  conn = NULL,
  debut,
  fin,
  type_Rx = "DENOM",
  codes,
  group_by = "DENOM",
  code_serv = c("1", "AD"),
  code_serv_filtre = "Exclusion",
  code_list = NULL,
  code_list_filtre = "Inclusion",
  age_date = NULL
)
```

Arguments

conn	Variable contenant la connexion entre R et Teradata. Voir SQL_connexion .
debut	Vecteur contenant la ou les dates de début des périodes d'étude au format AAAA-MM-JJ.
fin	Vecteur contenant la ou les dates de fin des périodes d'étude au format AAAA-MM-JJ.
type_Rx	Type de code à analyser. Une valeur parmi : <ul style="list-style-type: none"> 'AHFS' : Code identifiant la classe de médicaments telle que déterminée par l'<i>American Hospital Formulary Service</i>. 'DENOM' : Code de dénomination commune. 'DIN' : Code d'identification du médicament.
codes	Le ou les codes à analyser. Voir <i>Details</i> .
group_by	Équivalent du <i>group by</i> SQL. Regrouper (aggréger) les résultats par : <ul style="list-style-type: none"> 'AHFS' : Code identifiant la classe de médicaments telle que déterminée par l'<i>American Hospital Formulary Service</i>. 'DENOM' : Code de dénomination commune. 'DIN' : Code d'identification du médicament. 'CodeList' : Code de catégorie de liste de médicament. 'CodeServ' : Code de service. 'Teneur' : Teneur du médicament. 'Format' : Format d'acquisition du médicament. 'Age' : Âge à une date précise. Combiner avec l'argument <code>age_date</code>.
code_serv	Le ou les codes de services à exclure ou inclure, sinon inscrire NULL. <i>character</i> .

code_serv_filtre	'Exclusion' ou 'Inclusion' des codes de services.
code_list	Le ou les codes de catégorie de listes de médicaments à exclure ou inclure, sinon inscrire NULL. character.
code_list_filtre	'Exclusion' ou 'Inclusion' des codes de catégories de listes de médicaments.
age_date	Date à laquelle on calcul l'âge des individus. À utiliser seulement si group_by contient 'Age'.

Details

debut, fin :

debut et fin doivent contenir le même nombre de valeurs.

codes :

Si type_Rx='AHFS' : codes sous la forme de 6 caractères où les deux premiers caractères représente la classe AHFS, les deux du milieu la sous-classe AHFS et les deux derniers la sous-sous-classe AHFS. Il est possible de remplacer une paire de caractères ({1, 2}, {3, 4} ou {5, 6}) par '--' pour rechercher toutes les types de classes. Par exemple, '04--12' indique qu'on recherche la classe AHFS 04, toutes les sous-classes AHFS et la sous-sous-classe 12.

Sinon : inscrire les codes sous la forme d'un nombre entier.

code_serv_filtre, code_list_filtre :

'Exclusion' inclus les NULL

'Inclusion' exclus les NULL.

Nom des médicaments :

Que ce soit pour les codes AHFS (NOM_AHFS), les DENOM (NOM_DENOM) ou les DIN (NOM_MARQ_COMRC), **le nom inscrit est toujours celui le plus récent.**

Value

data.table

Examples

```
## Not run:
conn <- SQL_connexion(askpass::askpass('Utilisateur :'), askpass::askpass('Mot de passe :'))

### group_by
# Aucun
ex01 <- SQL_stat_gen1(
  conn, debut = c('2018-01-01', '2019-01-01'), fin = c('2018-12-31', '2019-12-31'),
  type_Rx = 'DENOM', codes = c(39, 47092, 47135), group_by = NULL
)
# Tous les group_by
ex02 <- SQL_stat_gen1(
  conn, debut = c('2018-01-01', '2019-01-01'), fin = c('2018-12-31', '2019-12-31'),
  type_Rx = 'DENOM', codes = c(39, 47092, 47135),
  group_by = c('AHFS', 'DENOM', 'DIN', 'CodeList', 'CodeServ', 'Teneur', 'Format', 'Age')
)

### AHFS
```

```

ex03 <- SQL_stat_gen1(
  conn, debut = c('2018-01-01', '2019-01-01'), fin = c('2018-12-31', '2019-12-31'),
  type_Rx = 'AHFS', codes = c('04----', '08--12', '122426'), group_by = 'AHFS'
)

### DENOM
ex04 <- SQL_stat_gen1(
  conn, debut = c('2018-01-01', '2019-01-01'), fin = c('2018-12-31', '2019-12-31'),
  type_Rx = 'DENOM', codes = c(39, 47092, 47135), group_by = c('DENOM', 'DIN')
)

### DIN
ex05 <- SQL_stat_gen1(
  conn, debut = c('2018-01-01', '2019-01-01'), fin = c('2018-12-31', '2019-12-31'),
  type_Rx = 'DIN', codes = c(30848, 585092), group_by = 'DIN'
)

### Age
ex06 <- SQL_stat_gen1(
  conn, debut = c('2018-01-01', '2019-01-01'), fin = c('2018-12-31', '2019-12-31'),
  type_Rx = 'DIN', codes = c(30848, 585092), group_by = c('DIN', 'Age'), age_date = '2018-01-01'
)

### Exclusion et Inclusion code_serv et code_list
ex07 <- SQL_stat_gen1(
  conn, debut = c('2018-01-01', '2019-01-01'), fin = c('2018-12-31', '2019-12-31'),
  type_Rx = 'DENOM', codes = c(39, 47092, 47135), group_by = 'DENOM',
  code_serv = c('1', 'AD'), code_serv_filtre = 'Exclusion',
  code_list = c('40', '41'), code_list_filtre = 'Inclusion'
)

## End(Not run)

```

sunique

Utils

Description

Combinaison de sort() et unique().

Usage

```
sunique(x, decreasing = FALSE, na.last = FALSE)
```

Arguments

x	Vecteur à trier et supprimer doublons.
decreasing	Ordre décroissant = TRUE, sinon FALSE.
na.last	Afficher les NA à la fin = TRUE, sinon FALSE. NA n'affiche pas les valeurs NA.

Examples

```
x <- sample(c(1:10, NA, NaN))
x

sunique(x)
sunique(x, na.last = TRUE)
sunique(x, decreasing = TRUE, na.last = NA)
```

V_DEM_PAINT_MED_CM	<i>Data</i>
--------------------	-------------

Description

Base de données sur les demandes de paiement de médicaments.

Usage

```
data('V_DEM_PAINT_MED_CM')
```

Format

list :

DENOM_DIN_AHFS Valeurs uniques des combinaisons 1) codes de dénomination communes, 2) codes DIN et 3) codes de classe AHFS.

- DENOM : Code de dénomination commune. character.
- DIN : Code d'identification du médicament. integer.
- AHFS_CLA : Classe AHFS. character.
- AHFS_SCLA : Sous-classe AHFS. character.
- AHFS_SSCLA : Sous-sous-classe AHFS. character.
- NOM_DENOM : Description du code DENOM. character.
- MARQ_COMRC : Nom de la marque commerciale. character.
- AHFS_NOM_CLA : Nom de la classe AHFS. character.
- DEBUT : Première année où la combinaison a été inscrite. integer.
- FIN : Dernière année où la combinaison a été inscrite. integer.

COD_AHFS Codes de classe AHFS.

- AHFS_CLA : Classe AHFS. character.
- AHFS_SCLA : Sous-classe AHFS. character.
- AHFS_SSCLA : Sous-sous-classe AHFS. character. - AHFS_NOM_CLA : Nom de la classe AHFS. character.
- DEBUT : Première année où le code a été inscrit. integer.
- FIN : Dernière année où le code a été inscrit. integer.

COD_DENOM_COMNE Codes de dénominations communes qui existent dans la base de données **V_DEM_PAINT_MED_CM**.

- DENOM : Code de dénomination commune. character.
- NOM_DENOM : Description du code DENOM.
- DEBUT : Première année où le code a été inscrit dans la base de données. integer.
- FIN : Dernière année où le code a été inscrit dans la base de données. integer.

COD_DIN Description des codes d'identification du médicament :

- DIN : Code d'identification du médicament. integer.
- DEBUT : Première année où le code a été inscrit dans la base de données. integer.
- FIN : Dernière année où le code a été inscrit dans la base de données. integer.

COD_SERV Description et années d'utilisation des codes de service. NA indique que le code n'a pas été utilisé.

- COD_SERV : Code de service. character.
- SERV_1 : Première et dernière année que le code de service a été inscrit dans la colonne **SMED_COD_SERV_1**. character.
- SERV_2 : Première et dernière année que le code de service a été inscrit dans la colonne **SMED_COD_SERV_2**. character.
- SERV_3 : Première et dernière année que le code de service a été inscrit dans la colonne **SMED_COD_SERV_3**. character.
- COD_SERV_DESC : Description du code de service. character.

COD_STA_DECIS Codes de statut de décision qui existent dans la base de données **V_DEM_PAINT_MED_CM**.

- COD_STA_DECIS: Code de statut de décision. character.
- COD_STA_DESC : Description du code de statut de décision. character.
- DEBUT : Première année où le code a été inscrit dans la base de données. integer.
- FIN : Dernière année où le code a été inscrit dans la base de données. integer.

Details

L'attribut MaJ indique la dernière mise à jour ou la date de création

Source

Dictionnaire EI

V_DENOM_COMNE_MED	<i>Data</i>
-------------------	-------------

Description

Description des codes de dénomination commune.

Usage

```
data('V_DENOM_COMNE_MED')
```

Format

Tableau de 7 variables :

DENOM Code de dénomination commune (NMED_COD_DENOM_COMNE). character.

DATE_DEBUT Date à laquelle cette dénomination commune est apparue pour la première fois (NMED_DD_DENOM_COMNE). Date.

DATE_FIN Date à laquelle la dénomination commune a cessé d'être utilisée (NMED_DF_DENOM_COMNE). Date.

NOM_DENOM Nom de la dénomination commune du médicament (NMED_NOM_DENOM_COMNE). character.

NOM_DENOM_SYNON Synonyme du nom de la dénomination commune du médicament (NMED_NOM_DENOM_COMNE_SYNON). character.

NOM_DENOM_ANGLAIS Nom anglais de la dénomination commune du médicament (NMED_NOM_ANGL_DENOM_COMNE). character.

NOM_DENOM_SYNON_ANGLAIS Synonyme du nom anglais de la dénomination commune du médicament (NMED_NOM_ANGL_DENOM_SYNON). character.

Details

L'attribut MaJ indique la dernière mise à jour ou la date de création du tableau.

Source

Dictionnaire EI.

V_DES_COD	<i>Data</i>
-----------	-------------

Description

Domaine de valeurs pour les différents codes de l'environnement informationnel.

Usage

```
data('V_DES_COD')
```

Format

Tableau de 5 variables :

CODE Valeurs codifiées que peut prendre un élément (CODE_VAL_COD). character.

TYPE_CODE Nom identifiant un élément de données (CODE_NOM_COD). character.

CODE_DESC Description du code (CODE_DES). character.

DATE_DEBUT Date de début de la période d'application (CODE_DD_DES_COD). Date.

DATE_FIN Date de fin de la période d'application (CODE_DF_DES_COD). Date.

Details

L'attribut MaJ indique la dernière mise à jour ou la date de création du tableau.

Source

Dictionnaire EI.

V_PRODU_MED	<i>Data</i>
-------------	-------------

Description

Produit qui peut faire l'objet d'une facturation. Règle générale, c'est un médicament conçu par un fabricant.

Usage

```
data('V_PRODU_MED')
```

Format

Tableau de 5 variables :

- NOM_MARQ_COMRC** Nom sous lequel est commercialisé un produit pharmaceutique.
- DENOM : Code de dénomination commune (NMED_COD_DENOM_COMNE). character.
 - DIN : Code d'identification du médicament (NMED_COD_DIN). integer.
 - NOM_MARQ_COMRC : Nom sous lequel est commercialisé un produit pharmaceutique (NMED_NOM_MARQ_COMRC). character.
 - DATE_DEBUT : Date d'entrée en vigueur de la mise à jour à laquelle est relié l'ajout ou la modification de cette occurrence (NMED_DD_PRODU_MED). Date.
 - DATE_FIN : Date d'entrée en vigueur de la mise à jour **moins un jour** de l'occurrence suivante (NMED_DF_PRODU_MED). Date.

Details

L'attribut MaJ indique la dernière mise à jour ou la date de création du tableau.

Source

Dictionnaire EI.

Index

* datasets

- Charlson_Dx_CCI_INSPQ18, [2](#)
 - Charlson_Dx_UManitoba16, [3](#)
 - CIM10, [5](#)
 - CIM9, [5](#)
 - CIM_correspond, [6](#)
 - Combine_Dx_CCI_INSPQ18, [6](#)
 - ComorbidityWeights, [9](#)
 - Elixhauser_Dx_CCI_INSPQ18, [10](#)
 - I_APME_DEM_AUTOR_CRITR_ETEN_CM, [12](#)
 - Obstetrics_Dx, [12](#)
 - Pop_QC, [13](#)
 - RLS_list, [21](#)
 - RLS_tab_convert, [21](#)
 - V_DEM_PAIMT_MED_CM, [35](#)
 - V_DENOM_COMNE_MED, [36](#)
 - V_DES_COD, [37](#)
 - V_PRODU_MED, [38](#)
-
- Charlson_Dx_CCI_INSPQ18, [2](#), [9](#)
 - Charlson_Dx_UManitoba16, [3](#), [9](#)
 - chunk_vec, [4](#)
 - CIM10, [5](#)
 - CIM9, [5](#)
 - CIM_correspond, [6](#)
 - Combine_Dx_CCI_INSPQ18, [6](#), [9](#)
 - comorbidity, [7](#)
 - ComorbidityWeights, [9](#)
-
- date_ymd, [9](#)
-
- Elixhauser_Dx_CCI_INSPQ18, [9](#), [10](#)
-
- file_directory, [11](#)
-
- I_APME_DEM_AUTOR_CRITR_ETEN_CM, [12](#)
-
- Obstetrics_Dx, [12](#)
-
- Pop_QC, [13](#)
-
- query_naif_switch1, [14](#)
 - query_stat_gen1, [17](#)
-
- replace_NA_in_dt, [20](#)
-
- RLS_convert, [20](#)
 - RLS_list, [21](#)
 - RLS_tab_convert, [21](#)
 - rmNA, [22](#)
-
- SQL_comorbidity, [22](#)
 - SQL_comorbidity_diagn, [24](#)
 - SQL_connexion, [23](#), [25](#), [26](#), [27](#), [30–32](#)
 - SQL_naif_switch1, [26](#)
 - SQL_obstetric, [30](#)
 - SQL_stat_gen1, [32](#)
 - SQL_stats_SMED_NBR_JR_DUREE_TRAIT, [31](#)
 - sunique, [34](#)
-
- V_DEM_PAIMT_MED_CM, [35](#)
 - V_DENOM_COMNE_MED, [36](#)
 - V_DES_COD, [37](#)
 - V_PRODU_MED, [38](#)