Package 'altair'

January 19, 2019

Title Fonctions pour l'analyse des bases de paye au format XML

Version 0.1

Description Analyse, traite et transforme les données des bases de paye conformes aux spécifications de la convention-cadre nationale de dématérialisation (http://xemelios.org)

Depends R (>= 3.2.3),
data.table (>= 1.9.5),
zeallot

License CeCILL v.2.1

LazyData true

RoxygenNote 6.1.1

ByteCompile true

Encoding UTF-8

R topics documented:

`adm` *Quotité administrative*

Description

Quotité administrative

Usage

```
adm(quotite)
```

Arguments

`quotite`	quotite formelle (Temps.de.travail / 100)

`againToBar` *Group of functions page title*

Description

Group of functions Description section

Usage

```
againToBar(x)

againNotToBar(x, y)

againTheQuestion(z)
```

Arguments

`x`	a param for toBar and notToBar
`y`	a param just for notToBar
`z`	a param just for theQuestion. Could put here, but easiest if all params are described in the same place, with page documentation, so none get repeated.

Details

Group of functions Details paragraph.

Value

Hard to have one return section for all functions, might want to have a manual list here.

After Arguments and Value sections

Despite its location, this actually comes after the Arguments and Value sections. Also, don't need to use null, could annotate first function, and then using function name as the groupBy name is more intuitive.

Another section

Probably better if all sections come first, uless have one section per function. Makes it easier to see the information flow.

`agrégat_annuel` *Affichage du tableau des agrégats des primes A et B, pour chaque année de période*

Description

Affichage du tableau des agrégats des primes A et B, pour chaque année de période

Usage

```
agrégat_annuel(résultat, verbeux)
```

Arguments

`résultat`	Résultat retourné par la fonction test_prime
`verbeux`	[FALSE] Le résultat n'est affiché que si `verbeux` vaut `TRUE`

Examples

```
agrégat_annuel(test_prime(prime_IAT, prime_B = prime_IFTS, Paie_I, verbeux = FALSE), verb
```

`ajuster.médiane.temps.complet`
Ajuster la médiane des templs complets en heures

Description

Ajuster la médiane des templs complets en heures

Usage

```
ajuster.médiane.temps.complet()
```

altair — *Altaïr: un paquet pour analyser les bases de paye XML.*

Description

Altaïr: un paquet pour analyser les bases de paye XML.

Démographie

effectifs : tableau des effectifs, ETP et EQTP

pyramide des âges : de 15 à 68 ans, verticale, hommes/femmes.

analyser — *Analyse des contraintes principales associées à une indemnité*

Description

Analyse des contraintes principales associées à une indemnité

Usage

```
analyser(prime, Paie_I, verbeux)
```

Arguments

`prime` Prime au format liste comportant les arguments :

- **nom** Nom de la prime en majuscules. Une expression régulière en décrivant le libellé doit être enregistrée dans l'espace global sous le nom : expression.rég.nom
- **catégorie** "A", "B", "C" ou tout vecteur d'une à deux lettres comprises dans ces trois valeurs. Décrit les catégories statutaires auxquelles la prime est attribuable.
- **restreint_fonctionnaire** Booléen. Par défaut FALSE. Préciser TRUE si la prime est uniquement attrubuable aux fonctionnaires. Dans certains cas (mais pas pour tous), la prime peut aussi être attribuable aux non-titulaires, sous réserve d'un acte réglementaire interne à l'organisme.
- **dossier** Chaîne de caractères. Sous-dossier du dossier Bases dans lequel le fichier auxiliaire CSV doit être généré. Par exemple : "Reglementation".
- **expr.rég.** Chaîne de caractères. Expression régulière filtrant sur champs `Grade`, décriuvant une contrainte limitant l'accès de la prime à un certain sous-ensemble de grades.
- **indice** Liste. Couple d'un caractère "+" ou "-" et d'un entier, ou triplet correspondant au couple augmenté d'un vecteur d'une ou deux lettres statutaires. Exemple : list("+", 350, c("A","B")). La liste décrit un critère limitatif pour la prime : elle ne peut être attribuée qu'aux indices supérieurs ("+") ou inférieurs ("-") au nombre donné en deuxième position pour les fonctionnaires de catégorie précisée en troisième prosition.

`Paie_I` Base data.table des indemnités comportant les colonnes :

- Nom
- Prenom
- Matricule
- Annee
- Mois
- Debut
- Fin
- Code
- Libelle
- Montant
- Type
- Emploi
- Grade
- Indice
- Statut
- Categorie

`verbeux` [FALSE] Le résultat des tableaux "non titulaires" et "catégories" n'est affiché que si `verbeux` vaut `TRUE`

Value

Liste constituée de :

Paye La base data.table de paye correspondant à la prime en premier argument, toutes primes confondues.

Lignes Les lignes de paye correspondant à la prime en premier argument seulement.

K Codes de paye correspondant à la prime.

manquant Booléen. TRUE si absence de résultat, FALSE sinon.

Note

Sauvegarde deux fichiers dans le sous-dossier prime$dossier :

prime$nom.non.tit.csv Recense les attributaires non titulaires prime$nom.cat.A (ou AB ou B ou BC...) Recense les attributaires de catégorie A, B, C ou toute combinaison de ces lettres.

`année_comparaison` *Annee de comparaison avec les données nationales.*

Description

Calcule l'année à laquelle la pyramide des âges va être comparée aux données nationales, pour un versant donné de la focntion publique.

Usage

```
année_comparaison(versant)
```

Arguments

versant	Chaîne de caractères parmi "FPT", "FPH", "TIT_FPT", "TIT_FPH", "NONTIT_FPT", "NONTIT_FPH".

Value

Une liste dont les composantes sont :

année l'année de comparaison (entier).

pyr la base de données nationales correspondante (data.table).

---

anyNameButFunctionNameIsUnique

Group of functions page title

Description

Group of functions Description section

Usage

```
alsoToBar(x)

alsoNotToBar(x, y)

alsoTheQuestion(z)
```

Arguments

x	a param for toBar and notToBar
y	a param just for notToBar
z	a param just for theQuestion. Could put here, but easiest if all params are described in the same place, with page documentation, so none get repeated.

Details

Group of functions Details paragraph.

Value

Hard to have one return section for all functions, might want to have a manual list here.

After Arguments and Value sections

Despite its location, this actually comes after the Arguments and Value sections. Also, don't need to use null, could annotate first function, and then using function name as the groupBy name is more intuitive.

Another section

Probably better if all sections come first, uless have one section per function. Makes it easier to see the information flow.

`calcul.nb.jours.mois` *Calcul du nombre de jours dans le mois*

Description

Calcul du nombre de jours dans le mois

Usage

```
calcul.nb.jours.mois(Mois, année)
```

Author(s)

Fabrice Nicol

`calculer_indice_complexité` *Diagnostic des tables de jointure*

Description

Calcule le nombre minimum de variables nécessaires pour apparier les bases de paye et un tableau de correspondance entre codes de paye et comptabilité

Usage

```
calculer_indice_complexité()
```

Value

NULL si les variables `Code, Libelle, Type` suffisent, sinon le tableau des multiplicités associées aux combinaisons de ces variables auxquelles il est impossible d'associer un compte unique.

`calcul_astreintes` *Constrôle des astreintes pour les emplois de responsabilité supérieure*

Description

Vérification du non-paiement des astreintes à des emplois de directeur général.

Usage

```
calcul_astreintes()
```

Details

- Filtre la paye en retenant les astreintes.
- Retient les cumuls avec les NBI non nulles pour la catégorie A et un emploi ou un grade de directeur général
- Affiche le nombre d'agents détectés dans ces cas
- Crée la table `Controle_astreintes` ayant pour colonnes : `"Code.astreinte", "Libelle.astrein`
- Crée la table `Cum_astreintes` ayant pour colonne : `Montant annuel astreintes`, et son total pour la période.
- Sauve les bases `Controle_astreintes` et `libelles.astreintes` dans le répertoire *Reglementation*
- Affiche si besoin la table `Cum_astreintes` au format `Tableau.vertical2`

Note

Les emplois sont filtrés à l'aide de l'expression rationnelle Perl suivante, sans prise en compte de la casse :
`d(?:\.|ir)\w*\s*\bg(?:\.|\w*n\.?\w*)\s*\b(?:des?)\s*\bs\w.*`

Author(s)

Fabrice Nicol

`calcul_HS` *Calcul des heures supplémentaires*

Description

Fonction permettant de calculer les heures supplémentaires par mois.

Usage

```
calcul_HS()
```

Details

`Base.IHTS` Filtre la paye sur les IHTS

`lignes.IHTS.tot` Lignes IHTS sommées par mois, en tenant compte des rappels ultérieurs :

```
ihts.tot  = ihts.cum.rappels  + ihts.cum.hors.rappels  + ihts.cum.rap
nihts.tot = nihts.cum.rappels + nihts.cum.hors.rappels + nihts.cum.ra
```

calcul_NBI *Vérification des payes de NBI, compte tenu des rappels et des quotités*

Description

Vérification des payes de NBI, compte tenu des rappels et des quotités

Usage

```
calcul_NBI()
```

Details

lignes_NBI	Sélectionne les lignes de paye de NBI
NBI_dec	Détecte les payes dont le nombre de points de NBI n'est pas entier
NBI.aux.non.titulaires	Détecte les NBI versées aux non titulaires
cumuls.nbi	Cumule les points de NBI par année

Détecte les rappels dont les années ou les mois sont inconnus. Dans de tels cas, suppose que l'année ou le mois manquant est l'année ou le mois en cours.

Variables globales
Affiche
Sauvegarde

Principales formules

```
nbi.eqtp.tot            = (nbi.cum.rappels + nbi.cum.hors.rappels) / adm.quotit
cumul.annuel.indiciaire = sum(nbi.cum.indiciaire, na.rm = TRUE)
cumul.annuel.montants   = sum(nbi.eqtp.tot, na.rm = TRUE)
```

See Also

proratisation_NBI catégories_NBI

calcul_rmpp *RMPP Calcul de la rémunération moyenne des personnes en place*

Description

RMPP Calcul de la rémunération moyenne des personnes en place

Usage

```
calcul_rmpp()
```

Note

Filtrage : on enlève les personnels présents depuis moins d'un seuil de troncature (ex. 120 jours) dans l'année et les élus (paramètre seuil.troncature) Filtrage pour l'étude des variations : on enlève les valeurs manquantes des variations, les centiles extrêmaux, les rémunérations nettes négatives ou proche de zéro. On exige un statut explicite en fin de période. Paramétrable par : minimum.positif, quantile.cut

---

charges.eqtp *Tableau des coûts moyens de personnel par grade.*

Description

Elabore un tableau des charges de personnel (coût) par grade et par année.

Usage

```
charges.eqtp(Base = Paie, grade = NULL, classe = NULL,
  service = NULL, libellés = NULL, agr = FALSE, période = NULL,
  variation = FALSE, statut = NULL, catégorie = NULL,
  exclure.codes = NULL, quotite.nulle = FALSE)
```

Arguments

`Base`	Base des bulletins de paye, comportant pour l'ensemble de la période 1. les variables charactère suivantes : • `Annee` • `Matricule` • `Statut` • `Grade` 2. les variables numériques : `quotite` réel entre 0 et 1 .
`grade`	Grade particulier. Tous les grades en l'absence de spécification.
`classe`	Liste de vecteurs de grades, chaque vecteur représentant une classe aggrégée, ou bien vecteur de chaîne de caractères représentant des expressions rationnelles sur les grades. Tous les grades en l'absence de spécification. La casse est ignorée pour les expressions rationnelles.
`service`	Services. Vecteur de chaînes de caractères exactes. Tous les services en l'absence de spécification.
`libellés`	Vecteur de libellés des agrégations de grades par expression régulière. Doit avoir la même dimension que le vecteur de regexp.
`agr`	Booléen (défaut FALSE). Si TRUE, l'expression régulière précédente conduit à agréger les grades décrits par le vecteur d'expressions régulières précédent : une ligne par composante du vecteur.
`période`	Vecteur des années considérées.
`variation`	Booléen Insérer une colonne des variations (défaut FALSE).

`statut` Restreindre le tableau au vecteur des statuts en paramètres. Expressions exactes. Tous statuts par défaut.

`catégorie` Categorie statutaire (vecteur de lettres parmi 'A', 'B', 'C'). Par défaut A, B, C ou indéterminée.

`exclure.codes` Codes de paye à exclure pour le calcul du coût salarial (vecteur de chaînes de caractères).

Value

Un tableau des effectifs mis en forme avec les grades en ligne et autant de colonnes numériques que d'années de période, plus une colonne de libellés.

Examples

```
charges.eqtp()
```

---

charges.personnel *Tableau des charges de personnel par grade.*

Description

Elabore un tableau des charges de personnel (coût) par grade et par année.

Usage

```
charges.personnel(Base = Paie, grade = NULL, classe = NULL,
  service = NULL, libellés = NULL, agr = FALSE, période = NULL)
```

Arguments

`Base` Base des bulletins de paye, comportant pour l'ensemble de la période

1. les variables charactère suivantes :
   - `Annee`
   - `Matricule`
   - `Statut`
   - `Grade`
2. les variables numériques :
   `quotite` réel entre 0 et 1

.

`grade` Grade particulier. Tous les grades en l'absence de spécification.

`classe` Liste de vecteurs de grades, chaque vecteur représentant une classe aggrégée, ou bien vecteur de chaîne de caractères représentant des expressions rationnelles sur les grades. Tous les grades en l'absence de spécification. La casse est ignorée pour les expressions rationnelles.

`service` Services. Vecteur de chaînes de caractères exactes. Tous les services en l'absence de spécification.

`libellés` Vecteur de libellés des agrégations de grades par expression régulière. Doit avoir la même dimension que le vecteur de regexp.

agr	Booléen (défaut FALSE). Si TRUE, l'expression régulière précédente conduit à agréger les grades décrits par le vecteur d'expressions régulières précédent : une ligne par composante du vecteur.
période	Vecteur des années considérées.
variation	Booléen Insérer une colonne des variations (défaut FALSE).
statut	Restreindre le tableau au vecteur des statuts en paramètres. Expressions exactes. Tous statuts par défaut.
catégorie	Categorie statutaire (vecteur de lettres parmi 'A', 'B', 'C'). Par défaut A, B, C ou indéterminée.

Value

Un tableau des effectifs mis en forme avec les grades en ligne et autant de colonnes numériques que d'années de période, plus une colonne de libellés.

Examples

```
charges.personnel()
```

---

chemin *Chemin complet d'un fichier dans le dossier* `chemin.dossier.données`

Description

Chemin complet d'un fichier dans le dossier `chemin.dossier.données`

Usage

```
chemin(fichier)
```

Arguments

fichier	Nom de fichier

---

conditionnel *Insertion conditionnelle de texte dans le rapport*

Description

Insertion conditionnelle de texte dans le rapport

Usage

```
conditionnel(msg = "", path = "")
```

Arguments

msg	Partie du rapport (commençant par #')
path	Chemin local de la base CSV sur le dossier d'exportation

Value

Le message msg dans le rapport

Author(s)

Fabrice Nicol

`correspondance_grade_catégorie`
Correspondance grade-catégorie

Description

Etablit la correspondance entre le grade et la catégorie statutaire à partir d'un tableau importé

Usage

```
correspondance_grade_catégorie()
```

Value

La `data.table grade.categorie` résultant de la lecture du fichier **grades.catégories.csv** sous le répertoire **Données** classe de références (RefClass).

Note

Requiert l'utilisation d'une table de jointure importée **grades.catégories.csv** sous le répertoire **Données**. A défaut, tente une association approximative à partir d'expressions rationnelles appliquées aux grades.

Author(s)

Fabrice Nicol

`correspondance_paye_budget`
Correspondance paye-budget

Description

Etablit la correspondance entre paye et comptabilité administrative (comptes 64 et 65)

Usage

```
correspondance_paye_budget()
```

Value

La `data.table code.libellé` résultant de la lecture du fichier **paye_budget.csv** sous le répertoire **Données**

Note

Requiert l'utilisation d'une table de jointure importée **paye_budget.csv** sous le répertoire **Données**. A défaut, tente une association approximative à partir d'expressions rationnelles appliquées aux libellés de paye.

`cumul_astreintes_IHTS` *Contrôle du cumul des astreintes et des IHTS*

Description

Détection du paiement le même mois d'indemnités d'astreintes et d'IHTS

Usage

```
cumul_astreintes_IHTS()
```

Author(s)

Fabrice Nicol

`distribution_smpt` *distribution_smpt*

Description

distribution_smpt

Usage

```
distribution_smpt(Filtre)
```

Arguments

`Filtre`	Filtre fonctionnel booléen décrivant une sélection sur les lignes (caractéristiques des personnels)

Value

Quartiles du SMPT

`effectifs` *Tableau des effectifs.*

Description

Elabore un tableau des effectifs et équivalents temps plein travaillés par type de personnel et par année.

Usage

```
effectifs(période, Bulletins = Bulletins.paie,
  personnels = Analyse.remunerations, Analyse.v = Analyse.variations)
```

Arguments

`période`	Vecteur des années de la période sous revue
`Bulletins`	Base des bulletins de paye, comportant pour l'ensemble de la période 1. les variables charactère suivantes : • `Matricule` • `Statut` • `Grade` 2. les variables numériques : `quotite` réel entre 0 et 1 `nb.mois` entier entre 0 et 12 3. la variable booléenne : `permanent` 12 bulletins sur l'année .
`Analyse`	Base des analyses de rémunérations, comptant les variables : `Filtre_actif, Filtre_annexe`
`Analyse`	Base des analyses de variations de rémunérations, comptant les variables : `temps.complet, est`

Value

Un tableau des effectifs mis en forme de 22 lignes et autant de colonnes numériques que d'années de période, plus une colonne de libellés.

Examples

```
effectifs(2010:2015)
```

`Eliminer.duplications`
Eliminer les doublons

Description

Eliminer les doublons

Usage

```
Eliminer.duplications()
```

`eqtp.grade` *Tableau des EQTP par grade.*

Description

Elabore un tableau des équivalents temps plein travaillés par grade et par année.

Usage

```
eqtp.grade(Base = Bulletins.paie, grade = NULL, classe = NULL,
  service = NULL, libellés = NULL, agr = FALSE, période = NULL,
  variation = FALSE, statut = NULL, catégorie = NULL)
```

Arguments

`Base`	Base des bulletins de paye, comportant pour l'ensemble de la période 1. les variables charactère suivantes : • `Annee` • `Matricule` • `Statut` • `Grade` 2. les variables numériques : `quotite` réel entre 0 et 1 .
`grade`	Grade particulier. Tous les grades en l'absence de spécification.
`classe`	Liste de vecteurs de grades, chaque vecteur représentant une classe aggrégée, ou bien vecteur de chaîne de caractères représentant des expressions rationnelles sur les grades. Tous les grades en l'absence de spécification. La casse est ignorée pour les expressions rationnelles.
`service`	Services. Vecteur de chaînes de caractères exactes. Tous les services en l'absence de spécification.
`libellés`	Vecteur de libellés des agrégations de grades par expression régulière. Doit avoir la même dimension que le vecteur de regexp.
`agr`	Booléen (défaut FALSE). Si TRUE, l'expression régulière précédente conduit à agréger les grades décrits par le vecteur d'expressions régulières précédent : une ligne par composante du vecteur.
`période`	Vecteur des années considérées.
`variation`	Booléen Insérer une colonne des variations (défaut FALSE).
`statut`	Restreindre le tableau au vecteur des statuts en paramètres. Expressions exactes. Tous statuts par défaut.
`catégorie`	Categorie statutaire (vecteur de lettres parmi 'A', 'B', 'C'). Par défaut A, B, C ou indéterminée.

Value

Un tableau des effectifs mis en forme avec les grades en ligne et autant de colonnes numériques que d'années de période, plus une colonne de libellés.

Examples

```
eqtp.grade()
```

`essayer` *Essaye d'exécuter une portion de code et en cas d'erreur continue l'exécution du script en renvoyant un message d'erreur non bloquant.*

Description

Essaye d'exécuter une portion de code et en cas d'erreur continue l'exécution du script en renvoyant un message d'erreur non bloquant.

Usage

```
essayer(X, Y, abort = FALSE, prof = profiler, times = 5,
  label = "")
```

Arguments

`abort`	Arrêt du programme en cas d'erreur.
`prof`	Profiler ou pas.
`times`	Nombre de tests de benchmark.
`code`	Portion de code.
`message`	Message d'erreur.

Value

Valeur retournée par X en cas de succès, sinon objet de classe "try-error" retourné par try(code)

Author(s)

Fabrice Nicol

Examples

```
if (exists("e")) rm(e)
essayer({ a <- 1/e}, "division par inconnu")  # affichage du message d'erreur
```

`exporter_tableau` *Apparier la base des lignes de paye et une table de jointure*

Description

Ajoute une ou plusieurs colonnes à la base des lignes de paye, étant donnée une table de jointure comportant des clés d'appariement de la base de paye et des vecteurs à apparier.

Usage

```
exporter_tableau(table.jointure, requis,
  clés = intersect(names(table.jointure), names(Paie)),
  calculer.indice.complexité = FALSE)
```

Arguments

`table.jointure`
: Tableau au format `data.table` indiquant la correspondance entre des clés appartenant à une `data.table` et un ou plusieurs vecteurs à rajouter à cette base.

`requis`
: Vecteur des noms des colonnes qui sont attendues dans le tableau de jointure, autre que les clés d'appariement, pour ajout à la base de paye.

`clés`
: Vecteur des noms de clés d'appariement. Par défaut, les noms de colonnes communs à la base des lignes de paye et à la table de jointure.

`calculer.indice.complexité`
: Pour l'appariement avec la comptabilité, vérifier s'il est éventuellement possible d'apparier sur les seules clés `Code, Libelle, Type`

Details

La table de jointure doit satisfaire une condition d'unicité de la valeur associée à chaque combinaison de clés.

Value

Base des lignes de paye `Paie` appariée avec la table de jointure restreinte aux variables : `Annee, Code, Libelle,` et aux colonnes ajoutées par la table de jointure

Note

Pour chaque combinaison de valeurs des clés, il doit y avoir une et une seule valeur des colonnes supplémentaires apportées par la table de jointure.

Effet de bord : Base des lignes de paye `Paie` appariée avec la table de jointure.

`extraire.nir` *Traitement du NIR (numéro d'inscription au répertoire des personnes physiques).*

Description

Extrait la répartition par âge et sexe des individus ayant un NIR.

Usage

```
extraire.nir(Base, année)
```

Arguments

`Base`	data.table contenant au moins une variable nommée Nir décrivant le NIR.
`année`	Annee civile à la fin de laquelle est évalué l'âge de l'individu.

Value

Une base data.table ayant la forme suivante (les bornes d'âge ne sont pas impératives) :

age	Hommes	Femmes
15	0	1
16	NA	2
17	1	3
18	409	52
...	...	...
68	2216	NA

Examples

```
extraire.nir(Base, 2012)
```

`extraire_paye` *Extraire les matricules et Nir des mois de décembre, sauf pour les élus Pour les statuts listés dans L si L non null*

Description

Extraire les matricules et Nir des mois de décembre, sauf pour les élus Pour les statuts listés dans L si L non null

Usage

```
extraire_paye(an, L, out)
```

Arguments

an	Annee
L	Vecteur des statuts considérés
out	Data.table extraite de Bulletins.paie de deux colonnes (Matricule et Nir), filtrée des doublons.

Author(s)

Fabrice Nicol

Examples

```
extraire_paye(2012, c("TITULAIRE", "STAGIAIRE"), "Bulletins.début.psr")
```

---

file2Latin *Conversion d'un fichier en ISO-8859-15*

Description

Conversion d'un fichier de UTF-8 en ISO-8859-15

Usage

```
file2Latin(nom, encodage.in = "UTF-8")
```

Arguments

nom	Chemin du fichier à encoder
encodage.in	(= encodage.entrée) Encodage du fichier de lecture

Value

Lancement dun appel système à iconv -t ISO-8859-15

Author(s)

Fabrice Nicol

`file2utf8` *Conversion d'un fichier de ISO-8859-15 en UTF-8*

Description

Conversion d'un fichier de ISO-8859-15 en UTF-8

Usage

```
file2utf8(nom, encodage.in = "ISO-8859-15")
```

Arguments

`nom`	Chemin du fichier à encoder
`encodage.in`	(= encodage.entrée) Encodage du fichier de lecture

Value

Lancement dun appel système à iconv -t UTF-8

Author(s)

Fabrice Nicol

`filtre` *Sélection du filtre correspondant à une chaîne de caractères associée*

Description

Sélection du filtre correspondant à une chaîne de caractères associée

Usage

```
filtre(x)
```

Arguments

`x`	Chaîne de caractères associée figurant dans la colonne `type` de la table `codes`

Value

Cette fonction renvoie un ensemble de valeurs fixes ou une expression régulière, sauf si x n'est pas dans la colonne `type` de la table `codes` (renvoie NULL)

Author(s)

Fabrice Nicol

filtrer_Paie *Filtrage d'une base de paye*

Description

Filtre la base par une expression régulière sur libellés de paye ou par valeurs fixes sur une variable donnée. Si le filtrage a une portée, l'ensemble des lignes de la portée (exemple "Mois") est conservé.

Usage

```
filtrer_Paie(x, portée = NULL, Base = Paie, Var = "Code",
  indic = FALSE)
```

Author(s)

Fabrice Nicol

FR *Ajoute le séparateur des milliers et sans décimale en français*

Description

Ajoute le séparateur des milliers et sans décimale en français

Usage

```
FR(x)
```

Arguments

x	Vecteur de valeurs numériques

Value

Vecteur de chaînes de caractères modifiées

Author(s)

Fabrice Nicol

Examples

```
FR(1235345.23) #1 235 345,2
```

`importer` *Importer les données*

Description

Importer les données

Usage

```
importer()
```

`importer_base_ifse` *Importer le base externe IFSE*

Description

Importer le base externe IFSE

Usage

```
importer_base_ifse()
```

`importer_base_logements` *Importer la base des logements de fonction*

Description

Importer la base des logements de fonction

Usage

```
importer_base_logements()
```

`importer_matricules` *Importer la base externe des correspondance ente matricules, catégories et grades*

Description

Importer la base externe des correspondance ente matricules, catégories et grades

Usage

```
importer_matricules()
```

`incrément` *Incrémente le numéro de tableau courant*

Description

Incrémente le numéro de tableau courant

Usage

```
incrément()
```

Value

Valeur incrémentée de numéro.tableau

Author(s)

Fabrice Nicol

`incrémenter.chapitre` *Incrémente le numéro de chapitre courant*

Description

Incrémente le numéro de chapitre courant

Usage

```
incrémenter.chapitre()
```

Value

Valeur incrémentée de chapitre

Author(s)

Fabrice Nicol

insérer_script *Insérer un script auxiliaire, indexé par une variable globale*

Description

Insérer un script auxiliaire, indexé par une variable globale

Usage

```
insérer_script(chemin = NULL, index = 1, variable = "année",
  gen = générer.rapport, incrémenter = FALSE, fonction = NULL)
```

Arguments

chemin	Chemin du script R
index	Vecteur numérique contenant les valeurs de la variable globale.
variable	Vecteur de caractères contenant le nom de la variable globale dans le script auxiliaire.
gen	Si FALSE alors se contente de sourcer le script auxiliaire selon encodage.code.source. Sinon intègre le rapport auxiliaire au format du rapport principal.
incrémenter	INcrémenter le chapitre de présentation du script
fonction	Appeler une liste de fonctions à argument vide
seq	Exécuter le script en mode séquentiel (si TRUE, resp. si FALSE, en mode parallèle)

Value

Valeur de la dernière variable globale variable instanciée. Effets de bord en sortie.

Author(s)

Fabrice Nicol

longueur.non.na *Longueur d'un vecteur, ou nombre de lignes d'une table, lorsque l'on a retiré les éléments ou les lignes NA*

Description

Longueur d'un vecteur, ou nombre de lignes d'une table, lorsque l'on a retiré les éléments ou les lignes NA

Usage

```
longueur.non.na(v)
```

Arguments

v	Vecteur ou data.frame/data.table

Value

Nombre d'éléments ou de lignes.

Author(s)

Fabrice Nicol

`net.eqtp` *Tableau des rémunérations nettes moyennes par grade.*

Description

Elabore un tableau des rémunérations nettes moyennes du personnel par grade et par année (hors SFT).

Usage

```
net.eqtp(Base = Paie, grade = NULL, classe = NULL, service = NULL,
  libellés = NULL, agr = FALSE, période = NULL,
  variation = FALSE, statut = NULL, catégorie = NULL,
  exclure.codes = NULL, quotite.nulle = FALSE)
```

Arguments

`Base`	Base des bulletins de paye, comportant pour l'ensemble de la période 1. les variables charactère suivantes : • `Annee` • `Matricule` • `Statut` • `Grade` 2. les variables numériques : `quotite` réel entre 0 et 1 .
`grade`	Grade particulier. Tous les grades en l'absence de spécification.
`classe`	Liste de vecteurs de grades, chaque vecteur représentant une classe aggrégée, ou bien vecteur de chaîne de caractères représentant des expressions rationnelles sur les grades. Tous les grades en l'absence de spécification. La casse est ignorée pour les expressions rationnelles.
`service`	Services. Vecteur de chaînes de caractères exactes. Tous les services en l'absence de spécification.
`libellés`	Vecteur de libellés des agrégations de grades par expression régulière. Doit avoir la même dimension que le vecteur de regexp.
`agr`	Booléen (défaut FALSE). Si TRUE, l'expression régulière précédente conduit à agréger les grades décrits par le vecteur d'expressions régulières précédent : une ligne par composante du vecteur.
`période`	Vecteur des années considérées.
`variation`	Booléen Insérer une colonne des variations (défaut FALSE).

`statut` Restreindre le tableau au vecteur des statuts en paramètres. Expressions exactes. Tous statuts par défaut.

`catégorie` Categorie statutaire (vecteur de lettres parmi 'A', 'B', 'C'). Par défaut A, B, C ou indéterminée.

`exclure.codes` Codes de paye à exclure pour le calcul du coût salarial (vecteur de chaînes de caractères).

Value

Un tableau des effectifs mis en forme avec les grades en ligne et autant de colonnes numériques que d'années de période, plus une colonne de libellés.

Examples

```
net.eqtp()
```

---

newline *Saut de ligne dans les rapports d'analyse*

Description

Saut de ligne dans les rapports d'analyse

Usage

```
newline()
```

Value

Aucun

Author(s)

Fabrice Nicol

---

newpage *Saut de page dans les rapports d'analyse*

Description

Saut de page dans les rapports d'analyse

Usage

```
newpage()
```

Value

Aucun

Author(s)

Fabrice Nicol

`non.null` *Sélectionne les éléments non nuls d'un vecteur*

Description

Sélectionne les éléments non nuls d'un vecteur

Usage

```
non.null(X)
```

Author(s)

Fabrice Nicol

`noria` *Décomposition de l'évolution des salaires, RMPP, SMPT et GVT*

Description

Elabore des tableaux permettant de relier l'évolution de la RMPP, du salaire moyen et des effets d'entrées-sorties.

Usage

```
noria(Bulletins = Bulletins.paie,
  Base = Analyse.variations.par.exercice, classe = "",
  champ = "brut", filtre = "", fichier = "", dec = ",",
  sep = ";", encoding = "UTF-8", afficher.tableau = TRUE,
  controle.quotite = FALSE)
```

Arguments

`Bulletins` Base mensuelle des bulletins de paie, comportant pour l'ensemble de la période

1. les variables charactère suivantes :
   - `Annee`
   - `Matricule`
   - `Statut`
   - `Grade`
   - `Categorie`
2. les variables numériques :

   `quotite` réel entre 0 et 1, quotite mensuelle

`Base` Base de paye, ou extraction de cette base, comportant pour l'ensemble de la période

1. les variables charactère suivantes :
   - `Annee`
   - `Matricule`
   - `Statut`
   - `Categorie`
2. les variables numériques :

   `quotite.moyenne` réel entre 0 et 1, quotite annuelle moyenne (somme des quotites divisée par 12).

   `Montant.net.eqtp` salaire net moyen annuel EQTP

   `Montant.brut.eqtp` salaire brut moyen annuel en EQTP

.

`classe` Valeur caractère définissant une sous-population.

`champ` "brut" ou "net" selon que le tableau est calculé sur rémunérations brutes ou nettes.

`filtre` "A", "B", "C" pour les catégories statutaires correspondantes, ou bien un vecteur de libellés de statuts possibles (exemple c("TITULAIRE", "STAGIAIRE"). Vecteur de chaînes de caractères exactes.

`fichier` Si absent, seules les bases de paye seront utilisées. En première année sous revue, est réputé absent le Matricule ayant une quotite non uniformément nulle sur l'année, qui n'a pas de quotite positive dans les trois premiers mois En dernière année sous revue, est réputé absent le Matricule ayant une quotite non uniformément nulle sur l'année, qui n'a pas de quotite positive dans les trois derniers mois Cette modélisation est adoptée par défaut si un fichier des entrées sorties n'est pas passé en paramètre fichier. Sont exclus les vacataires et assistantes maternelles détectées. Si présent, fichier indique le chemin complet du fichier des entrées-sorties, comportant pour l'ensemble de la période :

1. à titre obligatoire :

   `Matricule` Matricule de l'agent, au même format que dans les autres bases

   `Entrée` Date d'entrée de l'agent en service, de la forme numérique XX/XX/20XX

   `Sortie` Date de sortie de l'agent du service, de la forme numérique XX/XX/20XX
2. à titre optionnel :

   `Classe` une variable qualitative de type caractère définissant une sous-population.

`dec` Paramètre `dec` de la fonction `data.table::fread`.

`sep` Paramètre `sep` de la fonction `data.table::fread`.

`encoding` Paramètre `encoding` de la fonction `data.table::fread`.

`afficher.tableau`
Si TRUE, affiche quatre tableaux correspondant à la valeur de retour sur la sortie standard.

`controle.quotite`
Si TRUE, calcule la RMPP comme dans la définition (quotites identiques sur deux exercices, à 0,1 point près). Si FALSE, relâche cette hypothèse.

Details

La RMPP ici utilisée se distingue de la définition officielle pour la première année. On considère en effet en début de période que les salaires considérés sont ceux des agents présents toute la première année et toute la seconde avec la même quotite. Ce devrait être en principe "et toute l'année

précédente", mais celle-ci n'est pas documentée et l'approximation reste valable pour l'évaluation du GVT.

Value

Si afficher.tableau = TRUE :

1. Affiche un tableau des entrées-sorties, comportant les variables numériques suivantes :

   `Annee` Annees de la période sous revue, puis le libellé "Total".

   `Effectifs` Effectifs en janvier, hors quotites nulles, vacataires et assistantes maternelles identifiés.

   `ETPT` ETPT calculés sur base de paye, postes actifs non annexes (voir fonction effectifs).

   `ETPT entrants` ETPT des entrants de l'année, par année.

   `ETPT sortants` ETPT des sortants de l'année, par année.

   `Entrants` Entrants (effectifs physiques) de l'année, par année, puis Total.

   `Sortants` Sortants (effectifs physiques) de l'année, par année, puis Total.

   `Var. effectifs` différence du nombre d'entrants et de sortants, par année, puis Total.

   `Taux de rotation` ratio égal à la moyenne du nombre d'entrants et de sortants rapportée à l'effectif en janvier

2. Un deuxième tableau détaille l'effet des entrées sorties sur la masse des rémunérations brutes ou nettes.
   Des colonnes indiquent la valeur de l'effet en point de pourcentage du salaire moyen (% SMPT) :

   `Annee` Annees de la période sous revue, puis le libellé "Total".

   `Effet noria` Cumul des économies (si négatif) ou des coûts (si positif) sur salaires (nets ou bruts) dus au remplacement des sortants par les entrants, par année, puis Total.

   `Effet var. effectifs` Cumul des économies (si négatif) ou des coûts (si positif) sur salaires (nets ou bruts) dus aux variations d'effectifs.
   Est positif si le nombre d'entrants excède le nombre de sortants, par année, puis Total.

   `Effet vacances` Cumul des économies (si négatif) ou des coûts (si positif) sur salaires (nets ou bruts) dus aux vacances d'emplois.
   Est positif si les entrants intègrent l'organisme en moyenne plus tôt dans l'année que les sortants ne le quittent, par année, puis Total.

   `Total` Total des trois effets liés aux entrées-sorties (GVT négatif et variation d'effectifs vacances comprises)

3. Un troisième tableau détaille la décomposition du salaire moyen par tête (SMPT) en ses composantes stables (RMPP) et variables (flux d'entrées-sorties) Les coefficients des colonnes comprises entre la colonne "RMPP" et la colonne "SMTP" sont des variations relatives exprimées en pourcentage. Pour passer de la colonne "RMPP" à la colonne "SMPT", il suffit de multiplier successivement par chaque coefficient sur la même ligne, augmenté de 1.

   `Annee` Annees de la période sous revue

   `RMPP` Rémunération moyenne des personnes en place, présentes deux années complètes consécutives.
   La vérification de la permanence de la quotite sur les deux années n'est pas implémentée.

   `Entrée n - 1` Effet relatif en % des entrées de personnels en n - 1 présents en n et des variations de quotite entre n - 1 et n. Pour la première année, n - 1 est remplacé par n + 1 et "entrées" par "sorties".

   `Noria` Effet de noria. Effet relatif en % du remplacement des sortants par les entrants en faisant l'hypothèse que les entrants sont aussi nombreux que les sortants.

`Var. effectifs` Effet relatif en % sur le salaire moyen distribué des variations d'effectifs.

`Effet vacances` Effet relatif en % sur le salaire moyen distribué de la différence entre le nombre d'entrants et le nombre de sortants.

`Total E/S` Effet relatif en % sur le salaire moyen distribué des quatre effets précédents liés aux entrées-sorties.

`Ajustement` Correction à ajouter en point de pourcentage au coefficient Total E/S, dues aux approximations utilisées pour le calcul des effets d'entrées-sorties.

`SMPT` Salaire moyen par tête, résultant de l'application multiplicative des corefficients de variation précédents.

4. Un quatrième tableau est déduit des précédents et indique la variation de la RMPP, du SMPT, et les effets des entrées-sorties en points de pourcentage :

   `Var. RMPP` Variation relative de la RMPP.

   `Var. effets E/S` Effet des entrées-sorties : s'obtient par la différence entre la valeur du Total E/S du tableau précédent,
   corrigée par le coefficient d'ajustement, pour l'année n + 1, et celle de l'année n.

   `Cumul` Cumul des deux effets de variation de la RMPP et des effets d'entrées-sorties. Si ce cumul est égal à celui de la variation du SMPT, alors l'ensemble des calculs est correct.

   `Var. SMPT` Variation du SMPT.

5. Le retour de la fonction est un tableau du GVT négatif au format `data.table`, comportant les variables numériques suivantes (les totaux de colonnes figurent seulement dans l'affichage) :

   `période` Annees de la période sous revue, puis le libellé "Total".

   `effectifs.janvier` Effectifs physiques en janvier.

   `etpt.ent` ETPT entrants dans l'année, correspondant à des agents présents en décembre et pas en janvier.

   `etpt.sort` ETPT sortants dans l'année, correspondant à des agents présents en janvier et pas en décembre.

   `nent` Nombre d'entrants, par année, puis Total.

   `nsort` Nombre de sortants, par année, puis Total.

   `variation.effectifs` Différence du nombre d'entrants et de sortants, par année, puis Total.

   `taux.rotation` Ratio égal à la moyenne du nombre d'entrants et de sortants rapportée à l'effectifs de l'année précédente.

   `effet.noria` Cumul des économies (si négatif) ou des coûts (si positif) sur salaires (nets ou bruts) dus au remplacement des sortants par les entrants, par année, puis Total.

   `effet.variation.effectifs` Cumul des économies (si négatif) ou des coûts (si positif) sur salaires (nets ou bruts) dus aux variations d'effectifs.
   Est positif si le nombre d'entrants excède le nombre de sortants, par année, puis Total.

   `effet.vacances` Cumul des économies (si négatif) ou des coûts (si positif) sur salaires (nets ou bruts) dus aux vacances d'emplois.
   Est positif si les entrants intègrent l'organisme en moyenne plus tôt dans l'année que les sortants ne le quittent, par année, puis Total.

   `effet.total.entrées.sorties` Cumul des effets précédents, par année, puis Total.

   `rmpp.salaire` Rémunération moyenne des personnes en place

   `smpt.salaire` Salaire moyen par tête

   `rmpp.etpt` ETPT correpondant au calcul de la RMPP

   `smpt.etpt` ETPT correpondant au calcul du SMPT

`coef.entrées.ant` Effet relatif en % des entrées de personnels au cours de l'année précédente.

`coef.noria` Effet de noria relatif

`coef.var.effectifs` Effet relatif en % sur le salaire moyen distribué des variations d'effectifs

`coef.vacances` Effet relatif en % sur le salaire moyen distribué de la différence entre le nombre d'entrants et le nombre de sortants.

`coef.smpt` Produit des variations relatives correspondant aux coefficients précédents.

`var.rmpp` Variation de la RMPP.

`effet.es` Effet des entrées-sorties sur la variation du SMPT.

`var.smpt` Variations du SMPT.

`qualité` Indice de qualité des calculs, égal à l'erreur relative résultant de la modélisation, entre le cumul des effets des variables coef.x.y exprimés en masse de salaires, partant d'un salaire moyen fictif égal à la RMPP, et la somme des salaires effectivement versés au même nombre d'agents.

`delta.coef` Correction à ajouter en point de pourcentage au coefficient Total E/S, dues aux approximations utilisées pour le calcul des effets d'entrées-sorties

References

https://www.fonction-publique.gouv.fr/files/files/statistiques/rapports_annuels/2016/DGAFP_RA2016_web_signet.pdf Rapport annuel sur l'état de la fonction publique 2017 Rapport annuel sur l'état de la fonction publique 2018

	RMPP	Effet E/S	SMTP
brut	+3	-1,3	+1,7
net	+2,7	-1,3	+1,3

Voir aussi : outil 2BPSS, Minefi, Direction du budget

Examples

```
noria()
```

Annee	Effectifs	ETPT	ETPT entrants	ETPT sortants	Entrants	Sortants	Var.
2011	803	789,6	14,8	10,0	30,0	23,0	
2012	992	941,1	14,1	13,6	24,0	25,0	
2013	989	938,8	11,5	16,7	27,0	32,0	
2014	975	958,1	17,5	18,1	38,0	37,0	

```
Lecture :
L'organisme comptait 939 ETPT en 2013, dont 11,5 ETPT entrant et 16,7 ETPT sortants
correspondant respectivement à 27 entrants physiques et 32 sortants physiques.
Le taux de rotation était de 3,0 % au cours de cette année.
```

Annee	Effet noria	% SMPT	Effet var. effectifs	% SMPT	Effet vacances	% SMPT
2011	-64 887,1	-0,2	97 655,0	0,4	-45 805,1	-0,2
2012	-10 714,7	-0,0	-16 499,3	-0,0	90 945,2	0,3
2013	-102 531,4	-0,3	-61 836,1	-0,2	-50 050,9	-0,1

```
| 2014  |  -88 795,8  |  -0,3  |       11 441,7       |  0,0   |   -46 453,6    |  -0,1
```

```
Lecture :
L'effet de l'effet de noria en 2013 était de -0,3 % en valeur relative en proportion de
la masse des rémunérations brutes 2013 (autrement dit en points de SMPT).
Le total des effets d'entrées-sorties était de -0,6 point de SMPT, soit une économie
sur rémunérations brutes versées de 214 418,3 euros.
```

```
| Annee |   RMPP   | Entrées n - 1 | Noria | Var. effectifs | Vacances | Total E/S | Aju
|-------|----------|---------------|-------|----------------|----------|-----------|----
| 2011  | 34 849,0 |      0,00     | -0,24 |      0,36      |  -0,17   |   -0,05   |   -
| 2012  | 35 910,5 |     -0,40     | -0,03 |     -0,05      |   0,27   |   -0,21   |   -
| 2013  | 36 493,9 |     -0,63     | -0,30 |     -0,18      |  -0,15   |   -1,25   |
| 2014  | 36 576,0 |     -0,81     | -0,26 |      0,03      |  -0,13   |   -1,16   |   -
```

```
Lecture :
Le salaire brut moyen 2013 serait égal à la RMPP (rémunération des personnes
présentes tout au long de 2012 et 2013)
sans les effets d'entrées et de sorties en 2012 et 2013.
Ces effets interviennent en 2012 (seulement les entrées, première colonne) et en 2013
(colonnes suivantes).
L'effet relatif des entrées 2012 est de -0,63 % : les entrants en 2012 encore présents e
que les présents-présents (sur toute les années 2012 et 2013) en moyenne.
L'effet de noria 2013 (emplacement des sortants 2013 par un même nombre d'entrants) est
les sortants avaient en moyenne des salaires plus élevés que les entrants.
L'effet relatif de la variation des effectifs 2013 est de -0,18 % : il y a eu un peu plu
L'effet des vacances d'emploi est de -0,15 % : tous les sortants ne sont pas remplacés a
Le total de ces effets d'entrées-sorties est de -1,25 %, auquel s'ajoute un redressement
lié aux hypothèses de modélisation de 0,004 points de pourcentage.
```

```
|   Annee   | Var. RMPP | Var. effets E/S| Cumul | Var. SMPT |
|-----------|-----------|----------------|-------|-----------|
| 2011-2012 |   3,05    |     -0,58      |  2,45 |   2,45    |
| 2012-2013 |   1,62    |      0,07      |  1,70 |   1,70    |
| 2013-2014 |   0,22    |     -0,82      | -0,60 |  -0,60    |
```

```
Lien avec les prévisions budgétaires et le GVT :
Le GVT positif se déduit de la variation de la RMPP en retranchant l'effet, estimé par c
des mesures catégorielles et générales. Le GVT négatif total, défini comme la somme de l
et des différents effets d'entrées-sorties (en n et n-1) est égal à la colonne "Variatic
```

```
noria(champ = "net", classe = "SPP", filtre = "A")
```

```
Lecture : Produit des tableaux analogues aux précédents pour la rémunération des agents S
```

`positive` *Sélectionne les éléments positifs d'un vecteur*

Description

Sélectionne les éléments positifs d'un vecteur

Usage

```
positive(X)
```

Author(s)

Fabrice Nicol

`produire_pyramides` *Produire les pyramides des âges par versant de la fonction publique*

Description

Produire les pyramides des âges par versant de la fonction publique

Usage

```
produire_pyramides(Filtre_bulletins, titre, versant = "", envir)
```

Arguments

`Filtre_bulletins`	Fonction permettant de filtrer les bulletins sur les lignes de data.table
`titre`	Titre de la pyramide.
`versant`	Versant de la fonction publique ("FPT" ou "FPH")
`envir`	Environnement de stockage des caractéristiques des âges (`nom.fichier.après`, `nom.fichier.avant`, `res` quartiles de distribution des âges)

Note

- Prérequis :

```
année.fin.comp            Année de la fin de la comparaison
début.période.sous.revue  Début de la période sous revue
fin.période.sous.revue    Fin de la période sous revue
```

- Appelle : année_comparaison, extraire.nir, pyramides
- Crée les noms de base noms `data.table` :
  - `envir$nom.fichier.avant`
  - `envir$nom.fichier.après`
- Sauve : Ces deux bases.
  - Format : 3 colonnes numériques
  - Noms de colonnes : `age`, `Hommes`, `Femmes`

See Also

extraire.nir

`pyramide_ages` *Pyramide des âges.*

Description

Elabore une pyramide des âges verticale avec superposition du début et de la fin de la période sous revue.

Usage

```
pyramide_ages(Avant, Après = NULL, titre = "",
  date.début = début.période.sous.revue,
  date.fin = fin.période.sous.revue, versant = "",
  couleur_H = "darkslateblue", couleur_F = "firebrick4")
```

Arguments

`Avant`	data.table/data.frame décrivant la situation en début de période Cette base doit avoir la forme suivante (bornes d'âges non impératifs):

age	Hommes	Femmes
15	0	1
16	NA	2
17	1	3
18	409	52
...	...	...
68	2216	NA

dans laquelle "age" peut être soit un vecteur de nom de lignes soit une colonne.

`Après`	data.table/data.frame décrivant la situation en fin de période. Même format que `Avant`.
`titre`	Titre du graphique.
`date.début`	date du début de la période.
`date.fin`	date de fin de période.
`versant`	Si non renseigné, sans effet. Si renseigné par "FPT" (resp. "FPH"), le deuxième argument `Après` ne doit pas être renseigné. Il est automatiquement remplacé par une base de données disponible dans le répertoire `data/` du paquet, correspondant à l'année la plus proche du versant de la fonction publique correspondant. La pyramide superposée représente celle qu'aurait l'organisme si la distribution de ses âges était celle du versant mentionné de la fonction publique.
`couleur_H`	couleur utilisée pour représenter les hommes (partie droite de la pyramide). Par défaut `darkslateblue`
`couleur_F`	couleur utilisée pour représenter les femmes (partie gauche de la pyramide). Par défaut `firebrick4`
`envir`	environnement

Value

Une liste de deux vecteurs numériques représentant chacun des axes (gauche puis droit). Un graphique comprenat une pyramide, une légende et éventuellement un titre.

Examples

```
pyramide_ages(df1, NULL, "Pyramide des âges", 2008, 2012, versant = "FPT", comparer = TRU
```

`quotites` *Calcul des quotites*

Description

Calcul des quotites

Usage

```
quotites()
```

Arguments

`Bulletins.paie`	Fichier des bulletins de paye
`Paie`	Base des lignes de paye

`Read.csv` *Lecture d'une série de bases CSV*

Description

Appelle read.csv.skip sur chaque base d'une série de chemins et empile les retours en lignes

Usage

```
Read.csv(base.string, fichiers, charger = TRUE, colClasses = NA,
  skip = 0, drop = NULL,
  séparateur.liste = séparateur.liste.entrée,
  séparateur.décimal = séparateur.décimal.entrée, rapide = FALSE,
  convertir.encodage = TRUE, encodage = encodage.entrée)
```

Arguments

`base.string`	Vecteur de caractères du nom de l'objet data.table retourné
`fichiers`	Vecteur de chemins de fichiers
`charger`	Booléen : TRUE pour charger les bases, FALSE sinon (sans effet)
`colClasses`	Vecteur de classes ("numeric" ou "character", etc.) caractérisant les colonnes
`skip`	Sauter les N premières lignes
`drop`	Rang de la colonne à supprimer
`séparateur.liste`	Séparateur des champs CSV
`séparateur.décimal`	Séparateur décimal
`rapide`	Accélération parallèle ou pas
`convertir.encodage`	convertir d'encodage (basculer entre Latin-1 et UTF-8)
`encodage`	Encodage de la base d'entrée.

Value

Objet `data.table` résultant de l'empilement des bases lues.

Author(s)

Fabrice Nicol

Examples

```
test <- data.table(datasets::cars)
res  <- try(Read.csv("base",
                     "test.csv",
                      colClasses = c("integer", "integer"),
                      séparateur.liste = ";",
                      séparateur.décimal = ",",
                      convertir.encodage = FALSE,
                      encodage = "UTF-8",
                      rapide = TRUE),
          silent = FALSE)
if (inherits(res, 'try-error'))
  stop("Problème de lecture de la base de la table bulletins-lignes de Paie")
```

`read.csv.skip` *Lecture d'une base CSV*

Description

Lecture d'un fichier CSV et conversion en data.table. Si sécuriser.types.sortie = TRUE, forçage des types en sortie.

Usage

```
read.csv.skip(x, encodage = encodage.entrée, classes = NA,
  drop = NULL, skip = 0, rapide = FALSE,
  séparateur.liste = séparateur.liste.entrée,
  séparateur.décimal = séparateur.décimal.entrée,
  convertir.encodage = TRUE)
```

Arguments

`encodage`	Encodage de la base lue. Valeur par défaut : encodage.entrée
`classes`	Les classes ("character", "numeric") des variables en colonnes
`drop`	Rang de la colonne à supprimer
`skip`	Nombre de lignes à sauter en début de fichier (défaut aucune).
`rapide`	Booléen (= FALSE). Si TRUE, et si convertir.encodage est TRUE, convertir en UTF-8 avant lecture.
`séparateur.liste`	= séparateur.liste.entrée,
`séparateur.décimal`	= séparateur.décimal.entrée,
`convertir.encodage`	(= TRUE) convertir en encodage UTF-8 avant lecture

Value

Une base data.table

Author(s)

Fabrice Nicol

Examples

```
read.csv.skip(Base, séparateur.décimal = ",")
```

Redresser.heures — *Redresser les heures de travail (variable* Heures*) en tenant compte des traitements.*

Description

Redresser les heures de travail (variable `Heures`) en tenant compte des traitements.

Usage

```
Redresser.heures()
```

---

roxy — *Documentation*

Description

Produit la documentation

Usage

```
roxy(pwd = "mimine")
```

Arguments

`pwd`	Mot de passe

Note

Installation de la bibliothèque `altair.linux` dans /usr/local/lib

rémunérations_eqtp *Calculer la rémunération nette et brute en EQTP*

Description

Calculer la rémunération nette et brute en EQTP

Usage

```
rémunérations_eqtp(DT)
```

Arguments

DT	data.table (Bulletins.paie)

Résumé *Statistiques descriptives de base (minimum, maximum et quartiles)*

Description

Lecture d'un fichier CSV et conversion en data.table. Si sécuriser.types.sortie = TRUE, forçage des types en sortie.

Usage

```
Résumé(x, y, align = "r", extra = 0, type = "pond")
```

Arguments

x	Encodage de la base lue. Valeur par défaut : encodage.entrée
y	Les classes ("character", "numeric") des variables en colonnes
align	Rang de la colonne à supprimer
extra	Nombre de lignes à sauter en début de fichier (défaut aucune).
type	Booléen (= FALSE). Si TRUE, et si convertir.encodage est TRUE, convertir en UTF-8 avant lecture.

Value

Une base data.table

Author(s)

Fabrice Nicol

Examples

```
read.csv.skip(Base, séparateur.décimal = ",")
```

`Sauv.base` *Sauvegarde d'une base*

Description

Sauvegarde d'une base data.table sous forme de fichier CSV Si sécuriser.types.sortie = TRUE, forçage des types en sortie.

Usage

```
Sauv.base(chemin.dossier = "", nom = "", nom.sauv = nom,
  Latin = TRUE, sep = séparateur.liste.sortie,
  dec = séparateur.décimal.sortie, environment = .GlobalEnv)
```

Arguments

`chemin.dossier`	Chemin du dossier dans lequel la base sera sauvegardée
`nom`	Nom de l'objet à sauvegarder
`nom.sauv`	Chaine de caractères du nom du fichier .csv sans l'extension
`Latin`	(= convertir.latin) Convertir en encodage latin ISO-8859-15
`sep`	(= séparateur.liste.sortie)
`dec`	(= séparateur.décimal.sortie),
`environment`	(= .GlobalEnv) environnement,

Value

Valeur booléenne de file.exists(file.path(chemin.dossier, nom.sauv

Author(s)

Fabrice Nicol

Examples

```
Sauv.base("données", Base, "BaseDonnée", sep = ";", dec = ",")
```

`sauv.bases` *Sauvegarde de plusieurs bases*

Description

Sauvegarde d'une ou plusieurs base de type data.table sous forme de fichier CSV

Usage

```
sauv.bases(chemin.dossier, env, ...)
```

Arguments

chemin.dossier	Chemin du dossier dans lequel la base sera sauvegardée
env	Environnement
...	Autres noms d'objets à sauvegarder
nom	Nom de l'objet à sauvegarder

Value

Liste de booléens résultant de l'application de Sauv.base

Author(s)

Fabrice Nicol

Examples

```
envir <- environment()
# Générer Bulletins.paie et Paie dans envir
sauv.bases("données",  env = envir, c("Bulletins.paie", "Paie"))
```

---

sauvebase *Sauvegarde une base dans le dossier des bases*

Description

Sauvegarde paramétrée par environnement

Usage

```
sauvebase(x, y, z, env)
```

Arguments

x	Objet à sauvegarder (vecteur de caractères)
y	Nom du fichier de sauvegarde CSV.
z	Nom du sous-dossier du dossier des bases.
env	Environnement

SFT_sans_enfant *SFT sans enfant*

Description

SFT sans enfant

Usage

```
SFT_sans_enfant()
```

smpt *SMPT*

Description

SMPT

Usage

```
smpt(Filtre, type = "smpt net")
```

Arguments

Filtre	Filtre fonctionnel booléen décrivant une sélection sur les lignes (caractéristiques des personnels)
type	"smpt net", "smpt brut", "rmpp net", "rmpp brut"

Value

Tableau d'évolution du SMPT

Tableau *Tableau*

Description

Présentation de tables sous forme de tableau d'une seule ligne de données. Le tableau ne peut pas présenter plus d'une seule ligne.

Usage

```
Tableau(x, ...)
```

Arguments

x	Vecteur de nom de lignes pour le tableau
...	paramètres. Si sep.milliers est un paramètre, sa valeur est utiisée comme séparateur des milliers. Par défault, le séparateur blanc.

Value

Base de données data.table mise en forme de tableau par la fonction knitr::kable comportant l'ensemble des paramètres mis en colonnes centrées, avec x comme noms de lignes.

Author(s)

Fabrice Nicol

Examples

```
Tableau(c("a", "b", "c", "d"), 1, 2, 3, 4)
```

```
| a | b | c | d |
|:-:|:-:|:-:|:-:|
| 1 | 2 | 3 | 4 |
```

Tableau.vertical *Tableau de plusieurs lignes*

Description

Présentation de tables sous forme de tableau de plusieurs lignes de données.

Usage

```
Tableau.vertical(colnames, rownames, extra = "", ...)
```

Arguments

colnames	Vecteur de nom de colonnes pour le tableau
rownames	Vecteur de nom de lignes pour le tableau
extra	(= "") Si la valeur de ce paramètre est "variation", la variation relative est calculée entre le début et la fin de la période en lignes
...	paramètres fonctionnels uniquement. Si un paramètre n'est pas une fonction, le tableau est vide dans son ensemble

Value

Base de données data.table mise en forme de tableau par la fonction knitr::kable comportant l'ensemble des paramètres mis en colonnes centrées, avec x comme noms de lignes.

Author(s)

Fabrice Nicol

Examples

```
Tableau(c("a", "b", "c", "d"), 1, 2, 3, 4)
```

```
| a | b | c | d |
|:-:|:-:|:-:|:-:|
| 1 | 2 | 3 | 4 |
```

Tableau.vertical2 *Tableau vertical 2*

Description

Tableau vertical 2

Usage

```
Tableau.vertical2(colnames, rownames, ...)
```

Author(s)

Fabrice Nicol

tableau_cumuls *Affichage du tableau des cumuls de primes*

Description

Affichage du tableau des cumuls de primes

Usage

```
tableau_cumuls(résultat)
```

Arguments

résultat	Résultat retourné par la fonction test_prime

See Also

Other Tableau de primes: tableau_NAS

Examples

```
tableau_cumuls(test_prime(prime_IAT, prime_B = prime_IFTS, Paie_I, verbeux = FALSE))
```

```
|Matricule |Annee |       Grade          |                  Régime                      |
|----------|------|----------------------|----------------------------------------------|
|010843    |2009  |ANIMATEUR TERRITORIAL |   IFTS 1 mois-IAT 10 mois-Cumul 1 mois       |
|010843    |2010  |ANIMATEUR TERRITORIAL |   IFTS 11 mois-IAT 0 mois-Cumul 1 mois       |
|010854    |2009  |REDACTEUR TERRITORIAL |   IFTS 9 mois-IAT 2 mois-Cumul 1 mois        |
```

`tableau_NAS` *Affichage du tableau des cumuls de primes et du logement par NAS*

Description

Affichage du tableau des cumuls de primes et du logement par NAS

Usage

```
tableau_NAS(résultat)
```

Arguments

`résultat`	Résultat retourné par la fonction test_prime

See Also

Other Tableau de primes: `tableau_cumuls`

Examples

```
tableau_NAS(test_prime(prime_IFTS, Paie_I, verbeux = FALSE, NAS = "non"))
```

```
|Matricule |Annee | Mois |       Grade              |       Emploi             |    Montant
|----------|------|------|--------------------------|--------------------------|------------
|010843    |2009  |   8  |  ATTACHE PRINCIPAL       |   CHEF DE DIVISION       |    785,25
|010843    |2009  |   9  |  ATTACHE PRINCIPAL       |   CHEF DE DIVISION       |    785,25
|010854    |2009  |   9  |  ADMINISTRATEUR          |   CHEF DE SERVICE        |    995,20
```

`tester.homogeneite.matricules` *Teste si, dans une base, la proportion d'enregistrements Noms-Prenoms dont les matricules ne sont pas identiques reste inférieure à une marge de tolérance fixée (taux.tolérance.homonymie)*

Description

Teste si, dans une base, la proportion d'enregistrements Noms-Prenoms dont les matricules ne sont pas identiques reste inférieure à une marge de tolérance fixée (taux.tolérance.homonymie)

Usage

```
tester.homogeneite.matricules(Base)
```

Arguments

`Base`	Base à tester

Author(s)

Fabrice Nicol

test_avn *Teste les logements par NAS*

Description

Teste les logements par NAS

Usage

```
test_avn(avantage, Paie, base.logements = NULL)
```

Arguments

avantage	Vecteur de caractères indiquant le type de logement (actuellement seul "NAS" est actif)
Paie	Base de Paye principale comportant les variables Matricule, Annee, Mois, Statut, Gra
base.logements	La base de logements facultative importée par l'onglet Extra de l'interface graphique

Value

base de type data.table comportant les enregistrements identifiés comme problématiques.

test_plafonds *Teste les plafonds d'une prime*

Description

Teste les plafonds d'une prime

Usage

```
test_plafonds(plafonds, Lignes, logements = NULL)
```

Arguments

plafonds	Base data.table comportant les colonnes caractères Grade, Groupe et Logement suivies de la colonne numérique Plafond. Logement doit contenir le codage NAS pour les personnels logés par nécessité absolue de service.
Lignes	Lignes de paye limitées à des montants indemnitaires fléchés (ex: IFSE) et comportant les variables Matricule, Annee, Mois, Statut, Grade, Emploi, Type, C
logements	La base de logements importée par l'onglet Extra de l'interface graphique, comportant la variable Logement et le codage NAS pour les personnels logés par nécessité absolue de service. A défaut tous les agents sont considérés non logés.

Value

Liste constituée du coût des dépassements par année et d'une base de type data.table comportant les bulletins de paye comportant une ligne IFSE identifiée comme problématique.

`test_prime` *Teste les primes et indemnités*

Description

Analyse les contraintes relatives aux non titulaires, à la catégorie statutaire, au grade, à l'indice, aux cumuls avec d'autres indemnités.

Usage

```
test_prime(prime, prime_B, Paie_I = NULL, Paie_B = NULL,
  Lignes_B = NULL, verbeux = FALSE)
```

Arguments

`prime` Prime au format liste comportant les arguments :

nom Nom de la prime en majuscules. Une expression régulière en décrivant le libellé doit être enregistrée dans l'espace global sous le nom : expression.rég.nom

catégorie "A", "B", "C" ou tout vecteur d'une à deux lettres comprises dans ces trois valeurs. Décrit les catégories statutaires auxquelles la prime est attribuable.

restreint_fonctionnaire Booléen. Par défaut FALSE. Préciser TRUE si la prime est uniquement attrubuable aux fonctionnaires. Dans certains cas (mais pas pour tous), la prime peut aussi être attribuable aux non-titulaires, sous réserve d'un acte réglementaire interne à l'organisme.

dossier Chaîne de caractères. Sous-dossier du dossier Bases dans lequel le fichier auxiliaire CSV doit être généré. Par exemple : "Reglementation".

expr.rég. Chaîne de caractères. Expression régulière filtrant sur champs `Grade`, décriuvant une contrainte limitant l'accès de la prime à un certain sous-ensemble de grades.

indice Liste. Couple d'un caractère "+" ou "-" et d'un entier, ou triplet correspondant au couple augmenté d'un vecteur d'une ou deux lettres statutaires. Exemple : list("+", 350, c("A","B")). La liste décrit un critère limitatif pour la prime : elle ne peut être attribuée qu'aux indices supérieurs ("+") ou inférieurs ("-") au nombre donné en deuxième position pour les fonctionnaires de catégorie précisée en troisième prosition.

NAS Si vaut "non", la prime est incompatible avec le logement par nécessité absolue de service (NAS). Si vaut un nombre, la prime doit être inférieure à ce seuil pour bénéficier d'un logement par NAS.

`prime_B` Prime au format liste comportant les mêmes types d'arguments. Les cumuls de `prime` et de `prime_B` seront analysés.

`Paie_I` Base data.table des indemnités comportant les colonnes :

- Nom
- Prenom
- Matricule
- Annee
- Mois

- Debut
- Fin
- Code
- Libelle
- Montant
- Type
- Emploi
- Grade
- Indice
- Statut
- Categorie

`verbeux` [FALSE] Le résultat des tableaux "non titulaires" et "catégories" n'est affiché que si `verbeux` vaut `TRUE`

Value

Liste constituée de :

Paye La base data.table de paye correspondant à la prime en premier argument, toutes primes confondues.

Lignes Les lignes de paye correspondant à la prime en premier argument seulement.

K Codes de paye correspondant à la prime.

manquant Booléen. TRUE si absence de résultat, FALSE sinon.

`NAS` Base des cumuls irréguliers de `prime` et d'un logement par NAS, si `NAS` vaut "non", sinon NULL

Note

Sauvegarde deux fichiers dans le sous-dossier prime$dossier :

prime$nom.non.tit.csv Recense les attributaires non titulaires prime$nom.cat.A (ou AB ou B ou BC...) Recense les attributaires de catégorie A, B, C ou toute combinaison de ces lettres.

`toBar` *Group of functions page title*

Description

Group of functions Description section

Usage

```
toBar(x)

notToBar(x, y)

theQuestion(z)
```

Arguments

x	a param for toBar and notToBar
y	a param just for notToBar
z	a param just for theQuestions
z	a param just for theQuestion. Could put here, but easiest if all params are described in the same place, with page documentation, so none get repeated.

Details

Group of functions Details paragraph.

Value

Hard to have one return section for all functions, might want to have a manual list here, But if doing a manual list, that kind of avoids the purpose of the Functions section. Probably want to describe in aggregate here and in specific with each function.

Functions

- `toBar`: Description for the base function in the group.
  Returns no value. Not indented.
- `notToBar`: Description for another function in the toBar group
- `theQuestion`: Final function in toBar group. Still on same line with name even though new paragraph. Indenting does nothing here either.

After function section

Despite its location, this actually comes after the function section.

Another section after function section

Probably better if all sections come first, uless have one section per function. Makes it easier to see the information flow.

`type.données` *Contrôle le type des données d'entrée*

Description

Contrôle le type des données d'entrée

Usage

```
type.données(colonnes)
```

Arguments

colonnes	Vecteur des noms de colonnes de la table globale

%a% — *Assignation dans l'environnement d'appel*

Description

Assignation dans l'environnement d'appel

Usage

```
x %a% y
```

Arguments

x	Chaîne de caractères entre guillemets du nom de la variable assignée (gauche), comme pour `assign`
y	Valeur. Variable assignante (droite)

Examples

```
"x" %a% median(1:100); cat(x)
```

%+% — *Concaténation de chaîne de caractères*

Description

Equivalent de paste0(,)

Usage

```
x %+% y
```

Arguments

x	Vecteur de caractères
y	Vecteur de caractères

Value

Vecteur de caractères concaténé.

Author(s)

Fabrice Nicol

Examples

```
x <- "abc"
y <- "def"
cat(x %+% y) # "abcdef"
```

%s% *Prise en compte du pluriel*

Description

Prise en compte du pluriel

Usage

```
mot %s% N
```

Arguments

N	Entier : pluriel si N > 1.
Mot	Mot à pluraliser éventuellement

Value

Concaténation du mot et de "s" si N > 1

Author(s)

Fabrice Nicol

Examples

```
cat("patient" %s% 2)
```

évolution_agrégat *Affichage du tableau des variations des agrégats des primes A et B sur l'ensemble de la période*

Description

Affichage du tableau des variations des agrégats des primes A et B sur l'ensemble de la période

Usage

```
évolution_agrégat(résultat, verbeux)
```

Arguments

résultat	Résultat retourné par la fonction test_prime
verbeux	[FALSE] Le résultat n'est affiché que si `verbeux` vaut `TRUE`

Index