Transformer les données avec data.table : : COMPENDIUM

Les bases

data.table est un package très rapide et performant avec la mémoire pour transformer des données avec R. Il convertit les objets data frame natifs de R en data.table avec des fonctionnalités nouvelles et étendues. Les bases pour travailler avec data.table sont:

dt[i, j, by]

Utiliser data.table **dt**,
Extraire des lignes avec **i**
et manipuler les colonnes avec **j**,
grouper avec **by**.

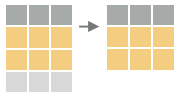
Les data.tables sont aussi des data frames – les fonctions qui opèrent sur des data frames opèrent également avec les data.tables.

Créer une data.table

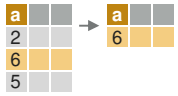
data.table(a = c(1, 2), b = c("a", "b")**)** – crée une data.table en partant de rien. Equivaut à data.frame().

setDT(df**)*** ou **as.data.table(**df**)** – convertit une data frame ou une liste en data.table.

Extraire des lignes avec i



dt[**1:2**,] – extrait les lignes en fonction des numéros de lignes.



dt[**a > 5**,] – extrait les lignes en fonction des valeurs contenues dans une ou plusieurs colonnes.

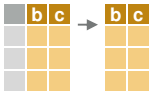
OPERATEURS LOGIQUES A UTILISER DANS i

<	<=	is.na()	%in%	|	**%like%**
>	>=	!is.na()	!	&	**%between%**

Manipuler les colonnes avec j

EXTRAIRE

dt[, **c(2)**] – extraire les colonnes par numéro. Préfixer les numéros de colonne avec "-" pour les ignorer.



dt[, **.(b, c)**] – extraire les colonnes par le nom.

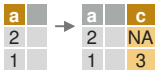
SOMMER

dt[, **.(x = sum(a))**] – créer une data.table avec de nouvelles colonnes basées sur le total des valeurs des lignes.

Les fonctions somme telles que mean(), median(), min(), max(), etc. peuvent être utilisées pour sommer les lignes.

CALCULER DES COLONNES*

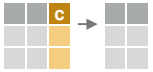
dt[, **c := 1 + 2**] – calcul d'une colonne basé sur une expression.



dt[**a == 1**, **c := 1 + 2**] – calcul d'une colonne basé sur une expression mais seulement sur un sous-ensemble de lignes.

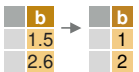
dt[, **`:=`(c = 1 , d = 2)**] – calcul de plusieurs colonnes basé sur des expressions séparées.

SUPPRIMER UNE COLONNE



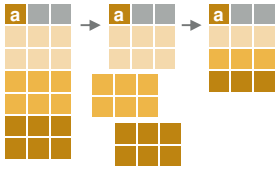
dt[, **c := NULL**] – supprime une colonne.

CONVERTIR LE TYPE D'UNE COLONNE



dt[, **b := as.integer(b)**] – conversion du type d'une colonne en utilisant as.integer(), as.numeric(), as.character(), as.Date(), etc..

Grouper avec by



dt[, j, **by = .(a)**] – groupe les lignes par valeurs des colonnes spécifiées.

dt[, j, **keyby = .(a)**] – groupe *et trie simultanément* les lignes par valeur des colonnes spécifiées.

OPERATIONS COMMUNES DE GROUPEMENT

dt[, **.(c = sum(b))**, **by = a**] – somme les lignes dans les groupes.

dt[, **c := sum(b)**, **by = a**] – crée une nouvelle colonne et calcule les lignes dans les groupes.

dt[, **.SD[1]**, **by = a**] – extrait la première ligne des groupes.

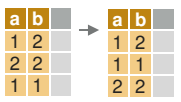
dt[, **.SD[.N]**, **by = a**] – extrait la dernière ligne des groupes.

Chaînage

dt[…**][**…**]** – réalise une séquence d'opérations de data.table en *chaînant* plusieurs "[]".

Fonctions pour les data.tables

TRI

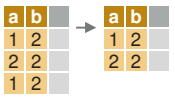


setorder(dt, a, **-**b**)** – trie une data.table en fonction des colonnes spécifiées. Préfixer les noms des colonnes avec "**-**" pour trier dans l'ordre descendant.

* FONCTIONS SET ET :=

Les fonctions de data.table préfixées par "set" et l'opérateur ":=" fonctionnent sans le "<-" pour modifier les données sans faire de copies en mémoire. Par exemple la fonction"setDT(df)" est plus efficace que sont analogue "df <- as.data.table(df)".

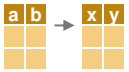
LIGNES UNIQUES



unique(dt, by = c("a", "b")**)** – extrait des lignes uniques basées sur les colonnes spécifiées dans "by". Ne pas utiliser "by" pour avoir toutes les colonnes.

uniqueN(dt, by = c("a", "b")**)** – compte le nombre de lignes uniques basées sur les colonnes spécifiées dans "by".

RENOMMER LES COLONNES



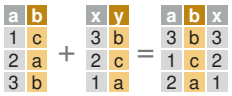
setnames(dt, c("a", "b"), c("x", "y")**)** – renomme les colonnes.

DEFINIR DES CLES

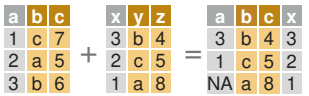
setkey(dt, a, b**)** – définit les clés pour permettre des recherches rapides et répétées dans les colonnes spécifiées en utilisant "dt[.(value),]" ou pour fusionner sans spécifier les colonnes en utilisant "dt_a[dt_b]".

Combiner des data.tables

JOINTURE

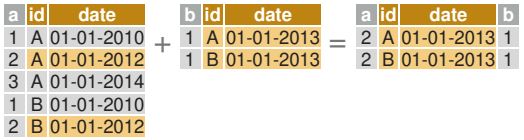


dt_a[dt_b, **on = .(b = y)**] – joint les data.tables sur les lignes d'égale valeur.



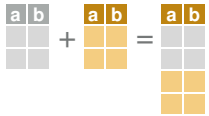
dt_a[dt_b, **on = .(b = y, c > z)**] – joint les data.tables sur les lignes des valeurs égales *et différentes*.

JOINTURE AVEC ROLL

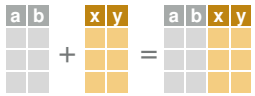


dt_a[dt_b, **on = .(id = id, date = date), roll = TRUE**] – joint les data.tables pour les lignes qui correspondent dans les colonnes id mais ne garde que la correspondance précédente la plus récente avec la data.table de gauche en fonction des colonnes de date. "roll = -Inf" inverse la direction.

LIER



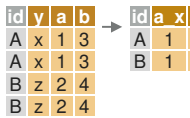
rbind(dt_a, dt_b**)** – combine les rangées de deux data.tables.



cbind(dt_a, dt_b**)** – combine les colonnes de deux data.tables.

Formater une data.table

FORMATER EN LARGEUR

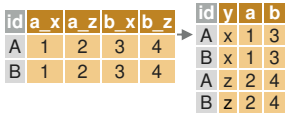


dcast(dt,
id ~ y,
value.var = c("a", "b")**)**

Restructure une data.table d'un format long en format large.

dt	Une data.table.
id ~ y	Formule avec pour membre gauche : colonnes ID contenant les IDs des entrées multiples. Et pour membre droit : les colonnes avec les valeurs à distribuer dans les entêtes des colonnes.
value.var	Colonnes des valeurs à mettre dans les cellules.

FORMATER EN LONGUEUR



melt(dt,
measure.vars = measure (
value.name, y, sep="_"))

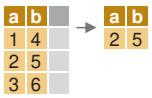
Restructure une data.table d'un format large en format long.

dt	Une data.table.
measure.vars	Colonnes des valeurs à mettre dans les cellules, souvent en utilisant measure() ou patterns ().
id.vars	Vecteur de caractères des noms des colonnes ID (optionnel).
variable.name, value.name	Noms des colonnes de sortie (optionnel).

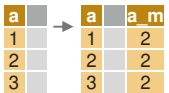
measure(out_name1, out_name2, sep="_", pattern="([ab])_(.*)"**)**
sep(séparateur) ou pattern (expression régulière) sont utilisés pour spécifier les colonnes à reformater, et pour analyser les noms dans la colonne d'entrée.
out_name1, out_name2: noms des colonnes de sortie (crée une colonne à valeur unique), ou value.name (crée des colonnes de valeurs pour chaque partie unique du nom de colonne reformatée).

Fonction appliquée aux colonnes

APPLIQUER UNE FONCTION A PLUSIEURS COLONNES



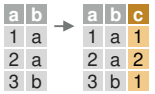
dt[, **lapply(.SD, mean)**, **.SDcols = c("a", "b")**] – applique une fonction – telle que mean(), as.character(), which.max() – aux colonnes spécifiées dans .SDcols avec lapply() et le symbole .SD . Fonctionne aussi avec les groupes.



cols <- c("a")
dt[, **paste0(cols, "_m") := lapply(.SD, mean)**, **.SDcols = cols**] – applique une fonction aux colonnes spécifiées et assigne le résultat avec les noms des variables suffixés aux données originales.

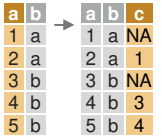
Lignes séquentielles

IDS DE LIGNES



dt[, **c := 1:.N**, **by = b**] – dans les groupes, évalue une colonne avec des IDs de lignes séquentielles.

EN TETE & EN QUEUE



dt[, **c := shift(a, 1)**, **by = b**] – dans les groupes, duplique une colonne avec les lignes *en queue* de la valeur spécifiée.

dt[, **c := shift(a, 1, type = "lead")**, **by = b**] – dans les groupes, duplique une colonne avec les lignes *en tête* de la valeur spécifiée.

Lire & écrire des fichiers

IMPORTER

fread("file.csv"**)** – lire les données d'un fichier à plat comme un fichier .csv ou .tsv, dans R.

fread("file.csv", select = c("a", "b")**)** – lire des colonnes spécifiques d'un fichier à plat, dans R.

EXPORTER

fwrite(dt, "file.csv"**)** – écrire les données dans un fichier à plat à partir de R.