# Transformer les données avec data.table : : COMPENDIUM

## Les bases

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

**dt[i, j, by]**

Objet data.table **dt**,

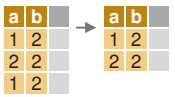
## Manipuler les colonnes avec j

### EXTRAIRE

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


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

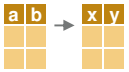
### LIGNES UNIQUES



**unique(**dt, by = c("a", "b")**)** – extraire 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")**)** – compter le nombre de lignes uniques basées sur les colonnes spécifiées dans “by”.

### RENOMMER DES COLONNES



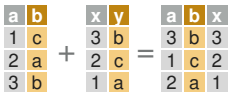
**setnames(**dt, c("a", "b"), c("x", "y")**)** – renommer les anciennes colonnes (a, b) en (x, y).

### DÉFINIR DES CLÉS

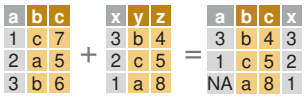
**setkey(**dt, a, b**)** – définir des clés pour permettre des recherches rapides et répétées dans les colonnes spécifiées en utilisant “dt[.(value), ]” ou pour fusionner sans indiquer les colonnes à utiliser avec “dt_a[dt_b]”.

# Combiner des data.tables

### JOINTURE

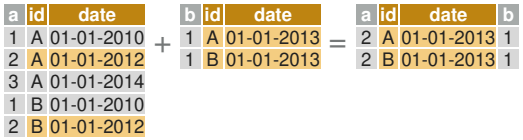


dt_a[dt_b, **on = .(b = y)**] – combiner les data.tables sur la base des lignes d’égales valeurs.



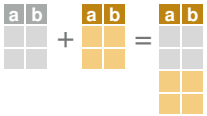
dt_a[dt_b, **on = .(b = y, c > z)**] – combiner les data.tables sur la base des lignes de valeurs égales *et différentes*.

### JOINTURE GLISSANTE

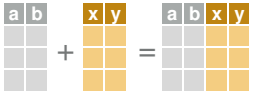


dt_a[dt_b, **on = .(id = id, date = date), roll = TRUE**] – combiner les data.tables pour les lignes qui correspondent dans les colonnes id, mais ne garder que la correspondance précédente la plus récente avec la data.table de gauche en fonction des colonnes de date. Utiliser “roll = -Inf” pour inverser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

### LIER



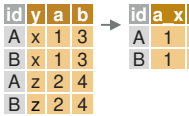
**rbind(**dt_a, dt_b**)** – combiner les lignes de deux data.tables.



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

# Restructurer un data.table

### RESTRUCTURER EN LARGEUR

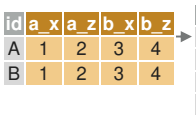


**dcast(** dt,
id ~ y,
value.var = c("a", "b") **)**

Restructurer une data.table d’un format long en format large.

| | |
|---|---|
| dt | Un data.table. |
| id ~ y | Formule avec pour le membre de gauche : colonnes ID contenant les IDs d’entrées multiples. Et pour membre de droite : les colonnes avec les valeurs à distribuer dans l’entête des colonnes. |
| value.var | Colonnes des valeurs à mettre dans les cellules. |

### RESTRUCTURER EN LONGUEUR



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

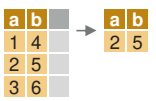
Restructurer un data.table d’un format large en format long.

| | |
|---|---|
| dt | Un 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). |

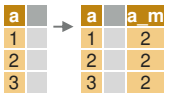
**measure(**out_name1, out_name2, sep="_", pattern="([ab])_(.*)"**)**
**sep**(séparateur) ou pattern (expression régulière) utilisés pour spécifier les colonnes à restructurer en analysant les noms des colonnes d’entrée.
out_name1, out_name2: noms des colonnes de sortie (crée une colonne à valeur unique), ou value.name (crée des colonnes de valeurs pour chaque partie unique du nom de la colonne restructurée).

# Fonction appliquée aux colonnes

### APPLIQUER UNE FONCTION À PLUSIEURS COLONNES



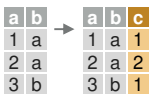
dt[, **lapply(.SD, mean)**, **.SDcols = c("a", "b")**] – appliquer une fonction – telle que mean(), as.character(), which.max() – aux colonnes indiquées dans .SDcols avec lapply() et le symbole .SD . Fonctionne aussi avec les groupes.



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

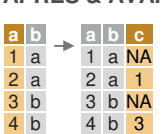
# Lignes séquentielles

### IDS DE LIGNES



dt[, **c := 1:.N**, **by = b**] – évaluer, au sein des groupes, une colonne avec des IDs de lignes séquentielles.

### APRÈS & AVANT



dt[, **c := shift(a, 1)**, **by = b**] – dupliquer, au sein des groupes, une colonne avec les lignes *suivantes* de la valeur spécifiée.

dt[, **c := shift(a, 1, type = "lead")**, **by = b**] – dupliquer, au sein des groupes, une colonne avec les lignes *précédentes* de la valeur spécifiée.

# Lire & écrire des fichiers

### IMPORTER

**fread(**"file.csv"**)** – lire les données d’un fichier de type .csv ou .tsv, dans R.

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

### EXPORTER

**fwrite(**dt, "file.csv"**)** – écrire les données dans un fichier depuis R.