Là où naissent la plupart des erreurs d'indicateurs

group_by() et summarise() — le regroupement qui survit à l'appel, la clé manquante qui devient son propre groupe, et les trois comportements par défaut exactement inverses de ceux de pandas.

Pierre Robentz Cassion

Leçon 6 sur 8

R et le tidyverse pour les données de programme — Les verbes

Ce que couvre cette leçon

- Un indicateur, ce sont deux nombres
- `n()` et les deux façons de compter
- Le regroupement survit, et c'est l'erreur la plus fréquente
- Une clé manquante devient son propre groupe
- Niveaux de facteur inutilisés : `count()` écarte, `table()` conserve
- `mutate()` groupé : une statistique de groupe sur chaque ligne
- `count()` quand c'est tout ce que vous voulez
- Deux clés, et le tableau qu'attend un rapport
- Écrire le résultat
- Les trois renversements, en un seul endroit
- Ce qui vient ensuite

Un indicateur, ce sont deux nombres — En R

```r
library(dplyr)

by_commune <- muac |>
  mutate(
    assessed = !is.na(muac_mm) | !is.na(oedema),
    gam = assessed & (muac_mm < 125 | oedema == "true")
  ) |>
  group_by(commune) |>
  summarise(
    screened  = n(),
    assessed  = sum(assessed, na.rm = TRUE),
    gam_cases = sum(gam, na.rm = TRUE),
    .groups = "drop"
  ) |>
  mutate(gam_rate = round(gam_cases / assessed, 3))
```

`n()` et les deux façons de compter — En R

```
summarise(muac, rows = n(), measured = sum(!is.na(muac_mm)))
```

`n()` et les deux façons de compter — En R

```r
sum(c(TRUE, NA))
#> [1] NA

sum(c(TRUE, NA), na.rm = TRUE)
#> [1] 1
```

Le regroupement survit, et c'est l'erreur la plus fréquente — En R

```r
out <- muac |>
  mutate(month = format(screening_date, "%Y-%m")) |>
  group_by(commune, month) |>
  summarise(n = n())
```

Le regroupement survit, et c'est l'erreur la plus fréquente — Exemple

```
`summarise()` has regrouped the output.
i Summaries were computed grouped by commune and month.
i Output is grouped by commune.
i Use `summarise(.groups = "drop_last")` to silence this message.
```

Le regroupement survit, et c'est l'erreur la plus fréquente — En R

```r
class(out)[1]
#> [1] "grouped_df"

dplyr::group_vars(out)
#> [1] "commune"
```

Le regroupement survit, et c'est l'erreur la plus fréquente — En R

```r
out |> mutate(share = n / sum(n))      # part dans la commune, non dans le district
out |> arrange(desc(n))                # trie a l'interieur de chaque commune
out |> slice_max(n, n = 5)             # top 5 par commune, non au total
```

Le regroupement survit, et c'est l'erreur la plus fréquente

- Dites ce que vous voulez, à chaque fois

Le regroupement survit, et c'est l'erreur la plus fréquente — En R

```r
summarise(..., .groups = "drop")        # resultat degroupe -- la reponse habituelle
summarise(..., .groups = "drop_last")   # garder tout sauf la derniere (le defaut)
summarise(..., .groups = "keep")        # tout garder
```

Le regroupement survit, et c'est l'erreur la plus fréquente — En R

```
muac |>
  summarise(n = n(), .by = c(commune, month))
```

Une clé manquante devient son propre groupe — En R

```r
d <- tibble::tibble(g = c("a", NA, "a"), v = c(1, 2, 3))

d |> group_by(g) |> summarise(s = sum(v), .groups = "drop")
```

Une clé manquante devient son propre groupe — Exemple

```
# A tibble: 2 × 2
  g         s
  <chr> <dbl>
1 a         4
2 <NA>      2
```

Une clé manquante devient son propre groupe — En R

```
d |> filter(!is.na(g)) |> group_by(g) |> summarise(s = sum(v), .groups = "drop")
```

Une clé manquante devient son propre groupe — En R

```
stopifnot(!any(is.na(muac$commune)))
```

Niveaux de facteur inutilisés : `count()` écarte, `table()` conserve — En R

```r
d <- tibble::tibble(k = factor(c("a", "b", "a"), levels = c("a", "b", "c")))

nrow(count(d, k))                 #> 2   -- le niveau vide a disparu
nrow(count(d, k, .drop = FALSE)) #> 3   -- conserve
length(table(d$k))                #> 3   -- conserve
```

Niveaux de facteur inutilisés : `count()` écarte, `table()` conserve

- Passez `.drop` explicitement — dans tout ce qui produit un tableau rapporté

`mutate()` groupé : une statistique de groupe sur chaque ligne — En R

```r
muac |>
  group_by(commune) |>
  mutate(
    commune_mean = mean(muac_mm, na.rm = TRUE),
    vs_commune   = muac_mm - commune_mean
  ) |>
  ungroup()
```

`mutate()` groupé : une statistique de groupe sur chaque ligne

- Noter le `ungroup()` — Un tableau groupé qui s'échappe dans la suite d'un script est la même erreur que ci-dessus,...

`count()` quand c'est tout ce que vous voulez — En R

```r
muac |> count(commune, outcome, sort = TRUE)
muac |> count(commune, wt = muac_mm)          # sommer une colonne plutot que compter
```

Deux clés, et le tableau qu'attend un rapport — En R

```r
by_commune_month <- muac |>
  mutate(month = format(screening_date, "%Y-%m")) |>
  summarise(cases = sum(gam, na.rm = TRUE), .by = c(commune, month))

wide <- by_commune_month |>
  tidyr::pivot_wider(names_from = month, values_from = cases, values_fill = 0)
```

Écrire le résultat — En R

```r
readr::write_csv(by_commune, here::here("outputs", "tables", "gam_by_commune.csv"))
```

Écrire le résultat — En R

```r
definitions <- tibble::tibble(
  indicateur   = "MAG",
  numerateur   = "PB < 125 mm ou oedemes bilateraux prenant le godet",
  denominateur = "enfants avec une mesure de PB ou une evaluation d'oedemes saisie"
)

readr::write_csv(definitions, here::here("outputs", "tables", "definitions.csv"))
```

Les trois renversements, en un seul endroit

	pandas	**R / dplyr**
Valeur manquante dans `mean()`	ignorée en silence	renvoie `NA` jusqu'à `na.rm`
Clé de groupe manquante	écartée	conservée comme groupe à part
Catégorie inutilisée	écartée par `observed=True`	conservée par `table()`, écartée par `count()`

Ce qui vient ensuite

- Le tableau agrège correctement et les deux moitiés de chaque taux proviennent d'un seul appel.

Lire la leçon complète, avec le code exécutable

https://data-analysis.cassion.dev/fr/cours/r-for-programme-data/r-06-grouping