

Entraînement avant le défi 2

Consignes de recyclage fictives : versions, clés et valeurs manquantes

Exemple distinct et fictif

Un commerce fictif reçoit quatre lots. Une liste fictive fixe une consigne (montant remboursable) en dollars par unité pour certains produits ; elle a deux versions. Un produit absent d'une version n'est pas consigné dans cette version : consigne de 0 \$. Une quantité inconnue reste inconnue. Dans cet exemple, D est présent dans les deux versions avec une quantité à confirmer.

```
library(tibble)
library(knitr)
lots <- tibble(lot_id = c("A", "B", "C", "D"), code = c("001", "001", "002", "003"),
  quantite = c(10, 20, 5, NA))
consignes <- tibble(version = c("v1", "v1", "v2", "v2", "v2"),
  code = c("001", "003", "001", "002", "003"), consigne = c(0.10, 0.25, 0.20, 0.05, 0.40))
```

lot_id	code	quantite
A	001	10
B	001	20
C	002	5
D	003	NA

version	code	consigne
v1	001	0.10
v1	003	0.25
v2	001	0.20
v2	002	0.05
v2	003	0.40

À faire

1. Donner la clé de chaque table. Pourquoi A et B restent-ils deux lots ?
2. Construire une ligne par lot avec la consigne totale en v1 et en v2, sans prendre une consigne dans l'autre version. Un produit absent d'une version donne 0 \$; la quantité inconnue de D donne un montant inconnu.
3. Exécuter `lots |> dplyr::left_join(consignes, by = "code")`. Combien de lignes obtenez-vous, et pourquoi ? Que se passerait-il si l'on additionnait ensuite par lot ?
4. Calculer la variation totale entre v1 et v2 pour les lots de quantité connue, et vérifier un montant à la main.
5. Écrire deux contrôles avec `stopifnot()` : clé unique de `consignes` par version et code, une ligne par lot dans votre résultat.
6. Faire une figure où un montant de 0 \$ et un montant inconnu se distinguent, puis un court rapport HTML avec `embed-resources: true` et une valeur calculée en ligne.

Pour vous préparer à l'entretien : avec votre code visible, prévoyez l'effet d'un passage de la consigne v2 du code 001 de 0,20 \$ à 0,30 \$, puis vérifiez votre prévision dans une copie.