Interroger le système au lieu d'exporter à la main

Deux points d'accès qui répondent à des questions différentes, la syntaxe de dimensions qui adresse un cube, et les quatre choses à régler avant tout — authentification, pagination, limites de débit, et le fait que les tables analytiques sont périmées tant que personne ne les reconstruit.

Pierre Robentz Cassion

Leçon 5 sur 8

Ce que couvre cette leçon

- L'export mensuel est le problème
- Deux points d'accès, deux questions
- Adresser le cube
- La forme de la réponse
- Les quatre choses à régler d'abord
- Tirez aussi les métadonnées
- La suite

L'export mensuel est le problème

- Quelqu'un ouvre l'interface, choisit les unités d'organisation, les périodes, les éléments de données, clique sur exporter, et envoie un tableur par courriel.

Deux points d'accès, deux questions

Point d'accès	Renvoie	À employer pour
`/api/dataValueSets`	Les valeurs brutes stockées, telles que saisies	Reproduire le registre ; tout ce qui exige de voir ce qui a été tapé
`/api/analytics`	Des valeurs agrégées et calculées	Indicateurs, totaux remontés dans la hiérarchie, tout ce que le système calcule

Deux points d'accès, deux questions

- `dataValueSets` donne la vérité telle que stockée — une ligne par élément de données, unité d'organisation, période et...
- `analytics` donne la réponse que le système afficherait sur un tableau de bord — agrégée dans la hiérarchie,...
- Tirez les deux quand ils sont censés concorder — Là où ils divergent, l'écart est une règle d'agrégation dont vous...

Adresser le cube — Exemple

```
GET /api/analytics.json
  ?dimension=dx:PENTA3_UID;BCG_UID
  &dimension=ou:LEVEL-4;DISTRICT_UID
  &dimension=pe:202401;202402;202403
  &displayProperty=NAME
```

Adresser le cube

- `dx` — dimension de données : éléments de données, indicateurs, opérandes.
- `ou` — unités d'organisation. `LEVEL-4` désigne toutes les unités du niveau 4 ; y adjoindre l'UID d'un district…
- `pe` — périodes, explicites (`202401`) ou relatives (`LAST_12_MONTHS`).
- Préférez les périodes explicites aux relatives dans un script — `LAST_12_MONTHS` renvoie une fenêtre différente chaque…

Adresser le cube — En Python

```python
import requests

BASE = "https://play.dhis2.org/40/api"

def analytics(dx, ou, pe, session):
    response = session.get(
        f"{BASE}/analytics.json",
        params={
            "dimension": [f"dx:{';'.join(dx)}", f"ou:{ou}", f"pe:{';'.join(pe)}"],
            "displayProperty": "NAME",
            "skipMeta": "false",
        },
        timeout=120,
    )
    response.raise_for_status()
    return response.json()
```

Adresser le cube — En R

```r
library(httr2)

analytics <- function(dx, ou, pe) {
  request("https://play.dhis2.org/40/api/analytics.json") |>
    req_url_query(
      dimension = paste0("dx:", paste(dx, collapse = ";")),
      dimension = paste0("ou:", ou),
      dimension = paste0("pe:", paste(pe, collapse = ";")),
      displayProperty = "NAME",
      .multi = "explode"
    ) |>
    req_perform() |>
    resp_body_json()
}
```

La forme de la réponse — En Python

```python
def to_frame(payload):
    import pandas as pd

    columns = [h["name"] for h in payload["headers"]]
    frame = pd.DataFrame(payload["rows"], columns=columns)

    names = {uid: item["name"]
             for uid, item in payload["metaData"]["items"].items()}
    for column in ("dx", "ou", "pe"):
        if column in frame:
            frame[f"{column}_name"] = frame[column].map(names)
    return frame
```

La forme de la réponse — En R

```r
to_frame <- function(payload) {
  cols <- vapply(payload$headers, \(h) h$name, character(1))
  rows <- do.call(rbind, lapply(payload$rows, \(r) unlist(r)))
  as.data.frame(rows, stringsAsFactors = FALSE) |> setNames(cols)
}
```

La forme de la réponse

- Conservez les UID en plus des noms — Les noms changent ; les UID non

Les quatre choses à régler d'abord

- L'authentification — Employez un jeton d'accès personnel là où l'instance le permet, l'authentification de base sinon,...

Les quatre choses à régler d'abord — En Python

```
import os

session = requests.Session()
session.headers["Authorization"] = f"ApiToken {os.environ['DHIS2_TOKEN']}"
```

Les quatre choses à régler d'abord — En R

```
req_headers(request(url), Authorization = paste("ApiToken", Sys.getenv("DHIS2_TOKEN")))
```

Les quatre choses à régler d'abord

- La pagination — Les points d'accès de métadonnées paginent par défaut à 50

Les quatre choses à régler d'abord — En Python

```python
def paged(url, session, page_size=1000):
    page = 1
    while True:
        payload = session.get(url, params={"page": page, "pageSize": page_size},
                              timeout=120).json()
        yield payload
        pager = payload.get("pager", {})
        if page >= pager.get("pageCount", 1):
            return
        page += 1
```

Les quatre choses à régler d'abord — En R

```
# Same idea: loop until page == pageCount, and never assume one request is all of it.
```

Les quatre choses à régler d'abord

- Le débit et la taille — Demandez un district et une année à la fois, non le pays et cinq années
- Les tables analytiques sont périmées — C'est celle qui surprend

Les quatre choses à régler d'abord — En Python

```
info = session.get(f"{BASE}/system/info", timeout=60).json()
print(info.get("lastAnalyticsTableSuccess"))
```

Les quatre choses à régler d'abord — En R

```
resp <- request(paste0(base, "/system/info")) |> req_perform() |> resp_body_json()
resp$lastAnalyticsTableSuccess
```

Les quatre choses à régler d'abord

- Consignez cet horodatage avec chaque extraction — C'est la différence entre « les chiffres ont changé » et « les…

Tirez aussi les métadonnées — Exemple

```
/api/dataElements.json?fields=id,name,aggregationType,categoryCombo[id,name]
/api/organisationUnits.json?fields=id,name,level,parent[id]&paging=false
/api/dataSets.json?fields=id,name,periodType,organisationUnits~size
/api/indicators.json?fields=id,name,numerator,denominator,indicatorType[name]
```

La suite

- Vous savez poser une question au système.

Lire la leçon complète, avec le code exécutable

https://data-analysis.cassion.dev/fr/cours/routine-data-and-dhis2/dhis2-05-the-web-api