TD 1 : Algèbre relationnelle

Logistique et Algèbre relationnelle

2024-09-20

L3 MIASHS/Ingémath
Université Paris Cité
Année 2024
Course Homepage
Moodle

Récapitulatif

Schémas

Les schémas sont une abstraction spécifique à PostgreSQL. Les schémas permettent de faire cohabiter sur une même base de données (ou catalogue dans le jargon PostgreSQL) plusieurs ensembles d'informations de nature différentes. On peut indiquer à `pgcli`, `psql` ou un autre client (`dbeaver` ou autre) quels schémas on veut utiliser *par défaut*.

Dans ce TP, nous allons nous concentrer sur le schéma `world` qui contient des informations concernant des villes et des pays/territoires du monde entier. Au cours de ce semestre, nous serons amenés à utiliser d'autres schémas : `pagila` qui contient des informations concernant des films, …

Pour lister et modifier les schémas de votre environnement de travail :

```
bd_2023-24> SHOW search_path ;                  -- lister
bd_2023-24> SET search_path TO world, public ;  -- modifier
bd_2023-24> SHOW search_path ;       -- visualiser
```

Lister les schémas du catalogue `bd_2023-24`.

```
\dn
```

Pour lister les tables des schemas inscrits sur votre `search_path`.

```
bd_2023-24> \d
+----------+---------------------+--------+----------+
| Schema   | Name                | Type   | Owner    |
|----------+---------------------+--------+----------|
| world    | city                | table  | postgres |
| world    | country             | table  | postgres |
| world    | countrylanguage     | table  | postgres |
+----------+---------------------+--------+----------+
```

Vue d'ensemble du schéma `world`

Suggestion

Si vous travaillez avec `pgcli`/`psql`, utilisez en local votre éditeur préféré (emacs, vi, sublime text, visual studio code, …), transférez votre script sql grâce à `scp`, et chargez le script dans votre session `psql`/`pgcli` à l'aide de `\\i`.

Tables

Pour voir la définition (c'est-à-dire les différentes colonnes) d'une table :

```
bd_2023-24> \d world.country

+--------------------+---------------+-------------+
| Column             | Type          | Modifiers   |
|--------------------+---------------+-------------|
| countrycode        | character(3)  |  not null   |
| name_country       | text          |  not null   |
| continent          | text          |  not null   |
| region             | text          |  not null   |
| surfacearea        | real          |  not null   |
| indepyear          | smallint      |             |
| population_country | integer       |  not null   |
| lifeexpectancy     | real          |             |
| gnp                | numeric(10,2) |             |
| gnpold             | numeric(10,2) |             |
| localname          | text          |  not null   |
| governmentform     | text          |  not null   |
| headofstate        | text          |             |
| capital            | integer       |             |
| code2              | character(2)  |  not null   |
+--------------------+---------------+-------------+
```

Explorer les possibilités de `psql` ou de `pgcli`

```
bd_2023-24> \?
```

Dans la deuxième partie du TP, on s'intéresse aux requêtes, c'est-à-dire, les moyens d'extraire une information pertinente d'une base de données.

Écriture d'une requête

Pour extraire des informations d'une base de données, on utilise l'algèbre relationnelle (pour la théorie) et le langage SQL (pour la pratique).

L'*algèbre relationnelle* est un ensemble d'opération sur les tables. Chaque opération prend en argument une ou plusieurs tables et produit une nouvelle table. Nous commençons par introduire deux opérations importantes qui opèrent sur une table à la fois :

- *Projection* : $\pi_{\texttt{liste de colonnes}} R$. Cette opération ne garde que les colonnes mentionnées de la table R. Par exemple $\pi_{\texttt{name,continent}}\texttt{world.country}$ est la table avec deux colonnes `name, continent` et une ligne pour chaque ligne de la table `world.country`.
- *Sélection* : $\sigma_{\texttt{condition}} R$. Cette opération ne garde que les lignes qui respectent la condition indiquée. Par exemple $\sigma_{\texttt{lifeexpectancy < 50}}\texttt{world.country}$ renvoie la table contenant les pays ayant une espérance de vie de moins de 50 ans.

Ces deux opérations peuvent être écrites en SQL ainsi :

```
SELECT colonne1, colonne2, ...
FROM table
WHERE condition;
```

Cette opération affiche les colonnes de `table` indiquées après le `SELECT` qui respectent la `condition`.

C'est la transcription de

$$\Pi_{\texttt{colonne1, colonne2}}\left(\sigma_{\texttt{condition}}\left(\texttt{table}\right)\right)$$

Par exemple :

```
SELECT name_country, lifeexpectancy
FROM world.country
WHERE lifeexpectancy < 50 and continent = 'Asia';
```

affichera le nom et l'espérance de vie des pays d'Asie ayant une espérance de vie inférieure à 50 ans. Remarquez l'utilisation des apostrophes simples pour délimiter les chaînes de caractères ('Asia').

Requêtes monotables

Ecrivez des requêtes en algèbre relationnelle et en SQL (dans `pgcli/psql, ...`) pour extraire les informations suivantes du schéma `world` :

- Quelles sont les régions ? (25 lignes)
- Quelles sont les régions situées en Europe ? (6 lignes)
- Quels sont les pays situés en Europe du sud ? (15 lignes)
- Quelles sont les capitales des pays situés en Europe de l'Ouest ? (quel est le type de la colonne `capital` ?) (9 lignes)
- A partir de la table `countrylanguage`, quels sont les langues qui sont officielles dans au moins un pays ? (102 lignes)
- Quels sont les codes des pays où le français est langue officielle ? (18 lignes) Même question pour les langues de votre choix ?
- Quelle est la date d'indépendance de la France ?
- Quelles sont les dates d'indépendance des pays d'Europe ? (46 lignes)
- Quelles sont les villes françaises de plus de 200 000 habitants ? (10 lignes)
- Pour chaque pays européen, calculer la densité, le GNP par habitant, et l'espérance de vie, ordonner par densité décroissante. (46 lignes)
- Quels sont les pays où l'espérance de vie n'est pas inférieure à 77 ans et le pnb par habitant n'est pas supérieur à (0.010) ? (10 lignes)
- Quels sont les pays tels que la condition (espérance de vie supérieure ou égale à 77 ans ou PNB par habitant inférieur à (0.01)) n'est pas vérifée ? (16 lignes)
- Quels sont les pays où une langue est officielle sans être parlée par au moins la moitié de la population ? (92 lignes)
- Quels sont les pays qui ont au moins une langue officielle ? (190 lignes)
- Quels sont les noms des pays qui comptent plus de 100 000 000 d'habitants ? (10 lignes)

Requêtes multi-tables

On peut aussi combiner plusieurs tables. Pour ce TP, nous allons seulement présenter le produit cartésien de deux tables : $T \times S$ est la table dont les colonnes sont les colonnes de S et les colonnes de T. Ces lignes contient tous les couples (l_1, l_2) où l_1 est une ligne de T et l_2 est une ligne de S. En SQL, on écrira :

```
SELECT col1,col2
FROM table1,table2
WHERE condition;
```

Par exemple,

```
SELECT language
FROM world.country as c, world.countrylanguage as l
WHERE c.countrycode = l.countrycode and c.continent = 'Europe';
```

affichera les langues parlées en Europe. Remarquez l'utilisation des `as` pour donner de nouveaux noms aux tables et l'utilisation de `c.countrycode` pour lever l'ambiguité sur des noms de colonnes qui seraient éventuellement les mêmes.

Avec ça, écrivez des requêtes pour les questions suivantes :

- Quels sont les noms des capitales Sud-Américaines ? (14 lignes)
- Quels sont les noms des pays où le français est langue officielle ? (18 lignes)
- Quelles sont les pays où l'espagnol est langue officielle et la forme de gouvernement est `Federal Republic` ? (3 lignes)
- Quels sont les pays qui ont au moins deux langues officielles ? (38 lignes)
- Quels sont les pays qui n'ont pas de langue officielle ? (49 lignes)
- Quels sont les pays qui comportent au moins deux villes de plus de 1 000 000 habitants ? (32 lignes)
- Quelles sont les régions qui ne comportent qu'une seule forme de gouvernement ? (3 lignes)
- Quelles sont les régions où on ne trouve pas de monarchie ? (9 lignes)