TP 11 - Découverte d'une base de données CORRIGÉ

Nous allons explorer une base de données concernant les communes françaises. L'objectif du TP est d'effectuer des requêtes simples pour extraire des informations utiles de cette base de données.

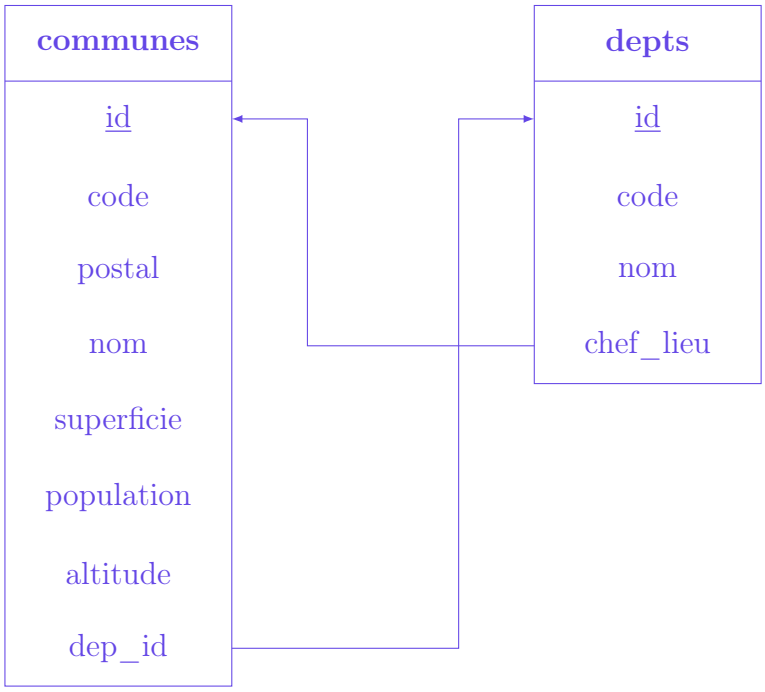
1 Mise en place

1. Télécharger le logiciel portable SQLiteStudio sur le site : https ://sqlitestudio.pl/. Aucune installation n'est nécessaire : une fois le fichier téléchargé, il suffit de le décompresser et de lancer l'exécutable "sqlitestudio".

2. Télécharger la base de données "communes_francaises.sqlite" sur le site "cahier de prepa". Créer une copie de ce fichier. L'ouvrir avec sqlitestudio (database -> connect to the database)

3. Observer (on regardera notamment les ongles "structure", "contraintes" et "DDL") et décrire : les tables contenues dans cette base de données, leurs attributs, les schémas relationnels, les clés primaires et étrangères.

   réponse : La base contient deux tables intitulées "communes" et "depts", selon le schéma suivant :



   Les unités de la table communes ne sont pas précisées ; on devine toutefois que

   - "superficie" est exprimé en hectares ($1 hectare = 10^{-2} km^2$)
   - "population" est exprimé en milliers
   - "altitude est exprimée en mètres.

2 Premières requêtes

Toujours dans SQLiteStudio, ouvrir l'éditeur SQL (onglet "tools"). Les requêtes sont écrites dans l'éditeur (une requête par ligne) et exécutées (icône en forme de triangle bleu)

Rappel : requêtes SQL typiques :

```
SELECT attribut(s)
FROM table1
JOIN table2 ON ... JOIN table3 ON ... (* jointures *)
WHERE condition (* sélection *)
GROUP BY attribut(s) (* agrégation *)
HAVING condition (*séléction post-agrégation *)
ORDER BY attribut(s) (ASC | DESC) (* ordonnancement *)
LIMIT n (* limitation du nombre de résultats affichés à n *)
OFFSET p (* ignorer les p premiers résultats *)
```

Il est possible d'insérer des commentaires : texte précédé de −−

Répondre aux questions suivantes en interrogeant une des deux tables de la base de donnée (pas besoin de jointure ici)

1. Afficher toutes les entrées de la table "dpts"

   réponse :
   SELECT * FROM dpts ;

2. Afficher les noms des communes du département du var

   réponse :
   SELECT nom FROM communes
   WHERE dep_id=84 ;

3. Afficher les noms des communes du département du var qui ont au moins mille habitants

   réponse :
   SELECT nom,population FROM communes
   WHERE dep_id=84 AND population >=1 ;

4. Afficher les 10 communes du var qui ont la plus grande superficie

   réponse :
   SELECT nom,superficie FROM communes
   WHERE dep_id=84
   ORDER BY superficie DESC
   LIMIT 10 ;

5. Afficher les communes du var par ordre d'altitude décroissante

   réponse :
   SELECT nom,altitude FROM communes
   WHERE dep_id=84
   ORDER BY altitude DESC ;

6. Quelle est la superficie moyenne d'une commune française ?

   réponse :
   SELECT AVG(superficie) FROM communes ;

7. Combien y-a-il de communes dans le département du Var ?

   réponse :
   ```
   SELECT COUNT() FROM communes
       WHERE dep_id=84;
   ```

8. Combien y a-t-il d'habitants dans le var ? et dans les 20 communes les plus peuplées du var ?

   réponse :
   ```
   SELECT SUM(population) FROM communes
       WHERE dep_id=84;
   ```
   Le deuxième point est plus délicat ; on peut créer la table des population des 20 communes les plus peuplées, et lui appliquer la fonction SUM ; on utilise donc deux SELECT imbriqués (sous-requête) :

   ```
   SELECT SUM(population)
           FROM (SELECT population FROM communes
                   WHERE dep_id=84
                   ORDER BY population DESC
                   LIMIT 20)
   ```

9. Afficher les communes de France dont la densité est comprise entre 1000 et 2000 habitants/km^2

   réponse :
   ```
   SELECT nom,population*100000/superficie AS densite FROM communes
           WHERE densite BETWEEN 1000 AND 2000;
   ```

3 Utilisation de jointures

Pour les questions suivantes nous allons interroger les deux tables en même temps (jointure)

1. Afficher la liste des 100 communes les plus peuplées de France et le département correspondant.

   réponse :
   ```
   SELECT communes.nom,depts.nom, population
           FROM communes JOIN depts ON communes.dep_id=depts.id
           ORDER BY population DESC LIMIT 100;
   ```

2. Afficher la liste des département et leur population rangée par ordre croissant

   réponse :
   ```
   SELECT depts.nom, SUM(population) AS pop_dep
           FROM communes JOIN depts ON communes.dep_id=depts.id
           GROUP BY depts.id
           ORDER BY pop_dep ASC
   ```

3. Afficher les dix départements ayant la plus grande superficie (calculée comme la somme des superficies de ses communes).

   réponse :
   ```
   SELECT depts.nom, SUM(superficie) AS sup_dep
                   FROM communes JOIN depts ON communes.dep_id=depts.id
                   GROUP BY depts.id
                   ORDER BY sup_dep DESC
                   LIMIT 10
   ```

4. Quel chef-lieu de département a l'altitude la plus élevée ?

   réponse :
   ```
   SELECT chef_lieu, communes.nom, altitude
                   FROM communes JOIN depts ON communes.id=chef_lieu
                   ORDER BY altitude DESC
                   LIMIT 1
   ```

5. Afficher les départements qui ont au moins une commune située à une altitude supérieure à 2000 m, ainsi que le nombre de ces communes.

   réponse :
   ```
   SELECT depts.nom, COUNT(communes.nom) AS nombre
                   FROM depts JOIN communes ON communes.dep_id=depts.id
                   WHERE altitude>=2000
                   GROUP BY depts.nom
   ```