Correction du TP 1 SQL

Thème Bases de données

Terminale NSI

1 Exercice 2

1. Écrire une requête qui affiche toutes les lignes de la table `communes`.

```
SELECT * FROM communes ;
```

2. Écrire une requête qui affiche toutes les lignes de la table `communes` ordonnées par population décroissante.

```
SELECT *
FROM communes
ORDER BY population DESC ;
```

3. Écrire une requête qui renvoie uniquement les noms de toutes les lignes de la table `communes` .

```
SELECT nom
FROM communes ;
```

4. Écrire une requête qui renvoie uniquement les noms de toutes les lignes de la table `communes` mais sans doublons.

```
SELECT DISTINCT nom
FROM communes ;
```

5. Écrire une requête qui affiche toutes les lignes de la table `communes` dont la valeur de la colonne `nom` est égale à `'Belleville'`.

```
SELECT *
FROM communes
WHERE nom = 'Belleville' ;
```

6. Écrire une requête qui affiche uniquement les colonnes `nom` et `codeDepartement` de la table `communes` dont la valeur de la colonne `nom` contient `'Belleville'` éventuellement suivie d'une autre partie.

```
SELECT nom, codeDepartement
FROM communes
WHERE nom LIKE 'Belleville%' ;
```

7. Écrire une requête qui affiche le nom et la population de tous les communes de la table `communes` dont le département est `'2A'` et la population est supérieure ou égale à 10000.

```
SELECT *
FROM communes
WHERE codeDepartement = '2A' AND population >= 10000;
```

8. Écrire une requête qui affiche dans l'ordre décroissant de la population, le nom et la population de tous les communes de la table `communes` dont le département est `'2A'` ou `'2B'` et la population est supérieure ou égale à 10000.

```
SELECT *
FROM communes
WHERE (codeDepartement = '2A' OR codeDepartement = '2B') AND population >=
     10000
ORDER BY population DESC;
```

2 Exercice 3

1. Écrire une requête avec la fonction d'agrégation `COUNT` qui renvoie le nombre total de communes françaises.

```
SELECT COUNT(*)
FROM communes ;
```

2. Écrire une requête avec la fonction d'agrégation `COUNT` et le mot clef `DISTINCT` qui renvoie le nombre total de noms distincts de communes françaises

```
SELECT COUNT(DISTINCT nom)
FROM communes;
```

3. Écrire une requête avec la fonction d'agrégation `SUM` qui renvoie la population totale du département `'69'`.

```
SELECT SUM(population)
FROM communes
WHERE codeDepartement = '69' ;
```

4. Avec les fonctions d'agrégation `SUM`, `MIN`, `MAX`, `AVG` et le mot clef `AS`, écrire une requête qui, pour l'ensemble des communes françaises, renvoie une ligne avec quatre colonnes ;

   - `pop_mini` contenant la population minimale ;
   - `pop_maxi` contenant la population maximale ;
   - `pop_total` contenant la somme de toutes les populations de communes ;
   - `pop_moy` contenant la population moyenne par commune.

```
SELECT MIN(population) AS pop_mini, MAX(population) AS pop_maxi, SUM(
    population) AS pop_totale, AVG(population) AS pop_moy
FROM communes ;
```

5. Pour déterminer toutes les communes dont la population est minimale par rapport à l'ensemble des communes françaises, on peut utiliser une *sous-requête* :

```
SELECT * FROM communes
WHERE population = (SELECT MIN(population) FROM communes);
```

À l'aide d'une *sous-requête*, écrire une requête qui renvoie toutes les communes du département `'92'` dont la population est minimale par rapport au sous-ensemble des communes du `'92'`.

```
SELECT *
FROM communes
WHERE codeDepartement = '92'
AND population = (
    SELECT MIN(population)
    FROM communes
    WHERE codeDepartement = '92'
    ) ;
```

Il s'agit de villages mémoires entièrement détruits pendant la première guerre mondiale https://fr.wikipedia.org/wiki/Villages_fran%C3%A7ais_d%C3%A9truits_durant_la_Premi%C3%A8re_Guerre_mondiale

3 Exercice 3 *Jointures simples*

1. Écrire une requête qui renvoie une nouvelle table constituée de trois colonnes :
   - `nom_com` contenant le nom de la commune ;
   - `nom_dep` contenant le nom de son département ;
   - `pop` contenant sa population.

   Les lignes doivent être classées d'abord par ordre croissant de nom de département puis par ordre décroissant de population.

```
SELECT communes.nom AS nom_com, departements.nom AS nom_dep, communes.
    population AS pop
FROM communes JOIN departements ON communes.codeDepartement =
    departements.code
ORDER BY nom_dep ASC, pop DESC ;
```

2. `LENGTH(chaine)` est une fonction `sqlite` permettant d'afficher la longueur d'une chaîne de caractères. Voir https://www.sqlitetutorial.net/sqlite-functions/sqlite-length/

   Écrire une requête qui renvoie une nouvelle table contenant les communes dont le nom a une longueur égale à la longueur du nom de leur département et constituée de deux colonnes :

- `nom_com` contenant le nom de la commune ;
- `nom_dep` contenant le nom de son département.

Les lignes doivent être classées par ordre décroissant de nom de commune puis croissant de nom de département.

```
SELECT communes.nom AS nom_com, departements.nom AS nom_dep
FROM communes JOIN departements ON communes.codeDepartement =
     departements.code
WHERE LENGTH(nom_com) = LENGTH(nom_dep)
ORDER BY nom_com DESC, nom_dep ASC;
```

3. Précisez la requête précédente en sélectionnant uniquement les communes de plus de 100000 d'habitants et en ordonnant les résultats par population décroissante.

```
SELECT communes.nom AS nom_com, departements.nom AS nom_dep, communes.
     population AS pop
FROM communes JOIN departements ON communes.codeDepartement =
     departements.code
WHERE (LENGTH(nom_com) = LENGTH(nom_dep)) AND (pop >= 100000)
ORDER BY pop DESC ;
```

4 Exercice 4 *Jointures multiples*

1. Écrire une requête qui renvoie une nouvelle table constituée de quatre colonnes :
   - `nom_com` contenant le nom de la commune ;
   - `nom_reg` contenant le nom de sa région ;
   - `nom_dep` contenant le nom de son département ;
   - `pop` contenant sa population.

   Les lignes doivent être classées d'abord par ordre croissant de nom de région, puis de département et enfin par ordre décroissant de population.

```
SELECT communes.nom AS nom_com, departements.nom AS nom_dep, communes.
     population AS pop, regions.nom AS nom_reg
FROM communes JOIN departements ON communes.codeDepartement =
     departements.code
JOIN regions ON departements.codeRegion = regions.code
ORDER BY nom_reg ASC, nom_dep ASC, pop DESC ;
```

2. Écrire une requête qui renvoie le nombre de communes de la région `Auvergne-Rhône-Alpes`.

```
SELECT COUNT(*)
FROM communes JOIN departements ON communes.codeDepartement =
     departements.code
JOIN regions ON departements.codeRegion = regions.code
```

```
WHERE regions.nom = 'Auvergne-Rhône-Alpes';
```

3. Écrire une requête qui renvoie la population totale de la région `Auvergne-Rhône-Alpes`.

```
SELECT SUM(communes.population)
FROM communes JOIN departements ON communes.codeDepartement =
     departements.code
JOIN regions ON departements.codeRegion = regions.code
WHERE regions.nom = 'Auvergne-Rhône-Alpes';
```

4. Dans la table `regions`, la valeur de la clef primaire de la région Île-de-France est 11.

    1. Écrire une requête qui renvoie le nombre de communes de plus de 50000 habitants dans cette région.

```
SELECT communes.nom AS nom_com, departements.nom AS nom_dep,
     communes.population AS pop
FROM communes JOIN departements ON communes.codeDepartement =
     departements.code
JOIN regions ON departements.codeRegion = regions.code
WHERE regions.code = '11' AND pop > 50000
ORDER BY pop DESC ;
```

    2. Écrire une requête qui renvoie le nom, le département et la population de la ou les commune(s) dont la population est minimale dans cette région.

```
SELECT communes.nom AS nom_com, departements.nom AS nom_dep,
     communes.population AS pop
FROM communes JOIN departements ON communes.codeDepartement =
     departements.code
JOIN regions ON departements.codeRegion = regions.code
WHERE regions.code = '11'
AND pop = (
    SELECT MIN(communes.population)
    FROM communes JOIN departements ON communes.codeDepartement =
         departements.code
    JOIN regions ON departements.codeRegion = regions.code
    WHERE regions.code = '11'
    );
```

5. Écrire une requête qui renvoie le nom, la population, le département et la région des communes dont la population totale est supérieure à celle de la région `Mayotte`. On utilisera une sous-requête.

```
SELECT communes.nom AS nom_com, departements.nom AS nom_dep, communes.
     population AS pop
FROM communes JOIN departements ON communes.codeDepartement =
     departements.code
JOIN regions ON departements.codeRegion = regions.code
WHERE pop >= (
```

```
    SELECT SUM(communes.population)
    FROM communes JOIN departements ON communes.codeDepartement =
         departements.code
    JOIN regions ON departements.codeRegion = regions.code
    WHERE regions.nom = 'Mayotte'
    );
```