Corrigé du TP 3 : Bases de données

Nous utilisons dans cette séance de TP, nous allons utiliser la base de données Mondial, qui contient des informations sur la géographie mondiale.
Celle-ci est décrite à l'adresse : https://www.dbis.informatik.uni-goettingen.de/Mondial/ .
On peut effectuer des requêtes via l'interface web : http ://www.semwebtech.org/sqlfrontend/ .
Cette base de données contient de nombreuses tables. Pour avoir une idée de ces tables et de la façon dont elles sont déclarées, vous pouvez ouvrir (dans un éditeur de texte) le fichier mondial-schema.sql.
On donne ci-dessous les informations sur une partie des tables de cette base de données.
La clef primaire de chaque table est soulignée (elle peut être constituée de plusieurs attributs. Dans les tables où il apparaît, l'attribut Country est une clef étrangère référençant la clef primaire Code dans la table Country.

- **Continent** : Name , Area ;
- **Country** : Name, Code , Capital, Province, Area, Population ; (Province est la région de la capitale) ;
- **Encompasses** : Country, Continent , Percentage ;
- **City** : Name , Country, Province , Population, Longitude, Latitude ;
- **Borders** : Country1 ,Country2 , length ; (Country1 $<$ Country2 pour l'ordre lexicographique)
- **Organization** : Abbreviation , Name, City, Country, Province, Established ; (Established est la date de fondation)
- **IsMember** : Country,Organization , type
- **Population** : Country , Population_growth, infant_mortality ;
- **Economy** : Country , GDP, Agriculture, Service, Industry, Inflation ; (GDP est le PIB) ;
- **Language** : Name , Country , Percentage ;
- **Mountain** : Name , Mountains, Elevation, Type, Longitude, Latitude ;
- **Geo_Mountain** : Mountain , Country, Province ;
- **Sea** : Name , depth
- **Geo_Sea** : Sea, Country, Province ;
- **Lake** : Name , Area, Depth, Elevation, Type, River, Longitude, Latitude ;
- **Geo_Lake** : Lake, Country, Province ;
- **River** : Name , River, Lake, Sea, Length ; (la rivière se jette dans une rivière, un lac ou la mer ; un attribut non renseigné est égal à NULL) ;
- **Geo_River** : River, Country, Province ;

Répondre aux questions suivantes en effectuant les requêtes SQL adaptées.

1. (a) Quelle est la capitale de Bangladesh ?

```
SELECT capital
FROM country
WHERE name = 'Bangladesh'
```

   (b) Donner la liste des continents avec leur surface.

```
SELECT *
FROM continent
```

   (c) Donner la liste des continents avec leur surface, classés du plus étendu au moins étendu.

```
SELECT *
FROM continent
ORDER BY area DESC
```

(d) Quels sont les différents types de montagnes ?

```
SELECT DISTINCT Type
FROM Mountain
```

(e) Quels sont les montagnes de plus de 8000 mètres de hauteur ?

```
SELECT *
FROM Mountain
WHERE elevation > 8000
```

(f) Quels sont les volcans ('volcano') de plus de 5000 m ?

```
SELECT *
FROM Mountain
WHERE type = 'volcano' and elevation > 5000
```

(g) Donner le nombre de pays de plus de 100 millions d'habitants.

```
SELECT COUNT(*)
FROM country
WHERE population > 100000000
```

(h) Donner le nombre moyen d'habitants des pays de plus de 100 millions d'habitants.

```
SELECT AVG(population)
FROM country
WHERE population > 100000000
```

2. (a) Déterminer les pays (leurs noms) dont la population décroît.

```
SELECT country.name
FROM population JOIN country
                ON population.country = country.code
WHERE population_growth < 0
```

(b) Établir la liste des pays avec leur continent.
Remarques :

- on a besoin uniquement des tables country et encompasses ;
- un pays peut apparaître plusieurs fois.

```
SELECT country.name, encompasses.continent
FROM   encompasses  JOIN country
           ON encompasses.country = country.code ;
```

(c) Déterminer la liste des pays avec leur continent et le pourcentage de la surface de ce continent occupé par ce pays (là on a besoin de trois tables).

```
SELECT country.name, continent.name, 100*country.area/continent.area
FROM   (encompasses  JOIN country  ON encompasses.country = country.code) JOIN
            ON encompasses.continent = continent.name
```

(d) Déterminer les pays d'Europe dont la population décroît, rangés selon l'importance de cette décroissance. On fera également figurer l'évolution de la population.

```
SELECT country.name ,  population_growth
FROM (population JOIN country
                 ON population.country = country.code)
                 JOIN encompasses ON encompasses.country = country.code
WHERE encompasses.continent = 'Europe' AND  population_growth < 0
ORDER BY population_growth
```

3. (a) Dessiner le schéma relationnel des trois premières tables de cette base.

   (b) Pourquoi a-t-on besoin de la table **Encompasses** ? On pourra faire pour l'expliquer un schéma UML des deux tables **Continent** et **Country**.

   Réponse :
   Les cardinalités des relations entre les tables **Continent** et **Country** sont du type `1..*` (un pays peut être sur plusieurs continents); ce qui ne permet pas de les associer par une clef étrangère.

   (c) Établir la liste des pays avec le nombre de continents dont ils font partie.

```
SELECT  country.name , COUNT(*)
FROM encompasses JOIN country ON encompasses.country = country.code
GROUP BY country.name
```

   (d) Déterminer les pays faisant partie de plusieurs continents.

```
SELECT  country.name , COUNT(*)
FROM encompasses JOIN country ON encompasses.country = country.code
GROUP BY country.name
HAVING COUNT(*) > 1
```

4. (a) Donner le nombre de fleuves qui se jettent dans la mer noire ('Black Sea').

```
SELECT COUNT(*)
FROM River
WHERE Sea = 'Black Sea'
```

   (b) Donner pour chaque mer le nombre de fleuves qui se jettent dedans.

```
SELECT Sea , COUNT(*)
FROM River
GROUP BY Sea
```

   (c) Donner les organisations regroupant plus de 100 pays, avec la population moyenne de ceux-ci par organisation.

```

SELECT Abbreviation , COUNT(*) , AVG(Country.population)
FROM (Organization JOIN IsMember ON Organization.Abbreviation =  IsMember.Organ
          JOIN Country ON IsMember.Country = Country.Code
GROUP BY Abbreviation
HAVING COUNT(*) > 100
```

5. (a) Déterminer les noms de pays qui sont aussi des noms de villes.

```
SELECT country.name
FROM country

INTERSECT

SELECT city.name
FROM city
```

(b) Déterminer les pays ayant une frontière avec la France (on identifiera celle-ci par son code : 'F').

```
SELECT Country1
FROM Borders
WHERE Country2 = 'F'

UNION

SELECT Country2
FROM Borders
WHERE Country1 = 'F'
```

Rem : on obtient ainsi les identifiants de ces pays ; on peut obtenir leurs noms en faisant une jointure.