On interrogera dans ce TP une base de données nommée *Mondial* [1], dont le schéma relationnel détaillé est donné en annexe de ce sujet. Lisez-le en entier avant d'écrire la moindre requête.

1 Rappels

Dans le modèle relationnel, une base de données est un ensemble de **relations**, représentées par des **tables**.

Chaque élément d'une relation, représenté par une ligne de la table, est appelé **enregistrement**.

Chaque **colonne** d'une table est appelée **attribut** de la relation associée.

Une **clé primaire** d'une relation est un ensemble d'attributs permettant d'identifier de manière unique un enregistrement. Ainsi, les clés primaires de deux lignes distinctes d'une même table auront toujours des valeurs différentes.

Une **clé étrangère** d'une relation est un ensemble d'attributs (clé) de cette relation qui fait référence à une clé primaire d'une autre relation.

Question 1. Quelle est la clé primaire de la table `country` ? de la table `continent` ? de la table `isMember` ?

Question 2. Déterminer deux clés étrangères de la table `encompasses` ? À quelles tables font-elles respectivement référence ?

2 Premières requêtes sur la base de données "*Mondial*"

Pour chaque question qui suivent, il faut désormais écrire une requête `SQL` répondant au problème posé.

2.1 Projection

Le schéma de requête ci-dessous permet de consulter l'ensemble de la table `table1`.

```
SELECT *
FROM table1
```

Question 3. Que contient la table `country` ?

Question 4. Que contient la table `borders` ?

Le schéma de requête ci-dessous permet d'afficher l'ensemble de la table `table1` en ne conservant que les colonnes (attributs) spécifiées.

```
SELECT attribut1, ..., attributn
FROM table1
```

Question 5. Établir une liste de tous les pays en précisant leurs capitales et nombres d'habitants respectifs.

Question 6. Établir une liste de tous les lacs de la base en précisant leurs superficies respectives.

Question 7. Établir la liste des groupes ethniques de la base. *On utilisera le mot-clé `DISTINCT` pour supprimer les éventuels enregistrements/lignes doublons.*

La clause `ORDER BY` permet de trier les lignes selon l'ordre ascendant **ASC** (par défaut) ou descendant `DESC`. Cette clause doit toujours être placé à la toute fin d'une requête (mais après la clause `LIMIT`).

```
SELECT attribut1, attribut2, attribut3
FROM table1
ORDER BY attribut2 ASC, attribut3 DESC
```

Question 8. Établir une liste des religions triée par ordre alphabétique

Question 9. Reprendre la question 6 en ordonnant les lacs dans l'ordre alphabétique.

Question 10. Reprendre la question 6 en ordonnant les lacs dans l'ordre décroissant des superficies.

[1] La base de données est consultable en ligne à l'adresse https://www.semwebtech.org/sqlfrontend/ mais via le langage `Oracle SQL` et non `MySQL` ou `SQLite`.

2.2 Sélection

Les schémas de requête ci-dessous permettent d'afficher l'ensemble de la table `table1` en ne conservant que les lignes (enregistrements) vérifiant la condition (booléenne) spécifiée.

```
SELECT *
FROM table1
WHERE condition
```

```
SELECT attribut1, ..., attributn
FROM table1
WHERE condition
```

Question 11. Déterminer toutes les villes de l'hémisphère Sud.

Question 12. Déterminer tous les aéroports de Londres (GB).

Question 13. Déterminer tous les aéroports (nom et ville) dont le décalage horaire avec Londres (GMT +0) est inférieur à 2h. *On triera les résultats par ordre croissant de décalage horaire, puis par ordre alphabétique des villes puis par ordre alphabétique des noms.*

3 Jointures

Les deux schémas de requête ci-dessous permettent de joindre deux tables sur une condition d'égalité de deux attributs (généralement une clé primaire et une clé étrangère). Lorsque deux attributs des deux tables jointes portent le même nom, il est parfois judicieux d'utiliser des alias de ces tables pour lever l'ambigüité, comme illustré dans le deuxième schéma avec le mot-clé `AS`.

```
SELECT *
FROM table1
JOIN table2
ON attribut1 = attribut2
```

```
SELECT *
FROM table1 AS t1
JOIN table2 AS t2
ON t1.attribut = t2.attribut
```

Question 14. Établir la liste de tous les continents ainsi que tous les pays qu'ils contiennent.
On ordonnera les enregistrements par ordre alphabétique des continents, puis, pour un même continent, par ordre alphabétique des pays.

Question 15. Établir la liste des pays qui sont ou contiennent une ile. *On ordonnera les enregistrements par ordre alphabétique des pays.*

Question 16. Établir la liste de tous les aéroports du monde, en précisant la ville et le pays où ils sont situés. *On triera les villes dans l'ordre alphabétique des pays, puis, pour un même pays, par ordre alphabétique du nom des aéroports.*

Question 17. Établir la liste des aéroports situé en France, en précisant la ville. *On s'interdira de chercher "à la main" le code de la France et on triera les enregistrements par ordre alphabétique des villes.*

Question 18. Déterminer toutes les villes du Vietnam (stockées dans la base). *On s'interdira de chercher "à la main" le code du pays et on triera les villes dans l'ordre décroissant du nombre d'habitants.*

Le schéma de requête ci-dessous illustre comment joindre successivement trois tables. Attention : l'ordre des tables peut avoir de l'importance.

```
SELECT *
FROM table1
JOIN table2 ON condition1
JOIN table3 ON condition2
```

Question 19. Établir la liste de tous les sommets du monde dépassant 5800 mètres, en précisant le pays où ils sont situés et leur altitude. *On triera les résultats par altitudes décroissantes.*

Question 20. Établir la liste des couples de pays frontaliers.

4 Fonctions d'agrégation

Les fonctions d'agrégation au programme sont les fonctions `MIN`, `MAX`, `SUM`, `AVG` (calcul de moyenne) et la fonction de comptage `COUNT`. Le schéma de requête ci-dessous renvoie une table ne contenant qu'une seule ligne (et une seule colonne) : le résultat de l'appel de la fonction `f` choisie.

```
SELECT f(attribut1)
FROM table1
```

Question 21. Combien y a-t-il de villes répertoriées dans la base ? *On renommera la colonne réponse.*

Question 22. Combien y a-t-il de frontières répertoriées dans la base ?

Question 23. Combien de pays contiennent ou sont une ile ? *Attention : un pays peut contenir plusieurs iles.*

Question 24. Quelle est la superficie maximale des provinces de la base ?

Le schéma de requête ci-dessous calcule, pour chaque valeur de l'attribut `attribut1`, `f(attribut2)`. La clause `GROUP BY` permet de réaliser ces calculs "par paquets". Une requête contenant la clause `GROUP BY` affiche une table contenant autant de lignes que de valeurs distinctes de l'attribut `attribut1`.

```
SELECT attribut1, f(attribut2)
FROM table1
GROUP by attribut1
```

Question 25. Quelle est le nombre d'habitants de chaque continent ?

Question 26. Déterminer, pour chaque pays, le nombre de montagnes situées dans le pays.

La clause `HAVING` permet de réaliser une opération de sélection, i.e. de ne conserver que les enregistrements vérifiant la condition de la clause, après application d'une fonction d'agrégation. Cette condition doit nécessairement porter sur le résultat de l'application de la fonction d'agrégation. Le schéma de requête ci-dessous illustre l'utilisation de cette clause.

```
SELECT attribut1, fonction(attribut2) AS alias
FROM table1
WHERE condition — clause optionnelle
GROUP by attribut1
HAVING condition2 — sur le nouvel attribut alias résultat d'une agrégation
```

Question 27. Établir la liste des pays du monde où sont localisés au moins 10 sommets dans la base.

Question 28. Établir la liste des pays où au moins 11 langues/dialectes sont parlées.

5 Requêtes imbriquées

Il existe deux types de requêtes imbriquées : celles pour lesquelles on crée une table temporaire sur laquelle on réalise une opération (projection, sélection, etc) et celles dans lesquelles on va chercher des enregistrements via la clause `WHERE`. Les deux schémas ci-dessous illustrent ces deux types de requêtes imbriquées.

```
SELECT *
FROM (
   SELECT *
   FROM table
   ...
) AS nom_table_temporaire
```

```
SELECT *
FROM table
WHERE (attrib1, attrib2) = (
    SELECT attrib3, attrib4
    FROM table 2
)
```

```
SELECT *
FROM table
WHERE (attrib1, attrib2) IN (
    SELECT attrib3,attrib4
    FROM table 2
)
```

On remarquera que dans le premier cas, on donne un nom à la table temporaire créée, tandis que dans le second cas, le nombre d'attributs de table construite doit être égal au nombre d'attributs des enregistrements recherchés.

Question 29. Déterminer la ou les provinces de superficie maximale, en précisant cette superficie.

Question 30. Déterminer le minimum des altitudes des aéroports ainsi que le(s) pays où il(s) se situ(ent).

Question 31. Déterminer le(s) pays membres du plus grand nombre d'organisations, en précisant ce nombre.

Le schéma de requête ci-dessous utilise la clause `LIMIT`. Placée à la fin d'une requête, celle-ci permet d'afficher les `p` premières occurrences de la table construite. Il est pratique de la combiner avec la clause `ORDER BY`.

```
SELECT attribut1, ..., attributn
FROM table1
ORDER BY attribut2
LIMIT p
```

Question 32. Déterminer les trois aéroports de plus hautes altitudes ainsi que le(s) pays où il(s) se situ(ent).

Question 33. Reprendre la question 29 en utilisant la classe `LIMIT`.

Question 34. Reprendre la question 31 en utilisant la classe `LIMIT`.

Question 35. Déterminer le nombre moyen de volcans actifs par continent.