Préliminaires

Ouvrez le programme `Sqlite Man` (ou un nom s'approchant) ainsi que la base de données `airports.sqlite` dans ce programme.

Rappel sur les requêtes

La structure principale est la suivante :

```
SELECT [DISTINCT]..., [COUNT/SUM...]
FROM ... [JOIN .. ON ..]
WHERE...
[GROUP BY... HAVING ...]
[ORDER BY...]
```

Méthodologie pour les requêtes simples

1. Identifier dans la question tous les champs (ou attributs, ou colonnes) qui doivent être affichés ou sur lesquels portent des conditions.
   Toutes les données à afficher/retourner doivent figurer dans la clause SELECT.
2. Lister la ou les tables qui contiennent les champs identifiés, construire la clause FROM contenant éventuellement une ou plusieurs jointures.
3. Écrire, si besoin la clause WHERE qui permet de filtrer les données en imposant des conditions sur un ou plusieurs champs. On peut utiliser les connecteurs logiques AND, OR ou la négation OR pour les conditions complexes.
4. La clause ORDER BY permet d'ordonner les lignes retournée en les triant (par ordre croissant) suivant une ou des colonnes. Pour trier par ordre décroissant on utilise ORDER BY colonne DESC

Écriture des jointures

Pour rappel :

1. identifier d'abord par un schéma les liens entre les tables : certaines colonnes sont des références à une clé primaire d'une autre table.
2. Un jointure s'écrit

   `FROM table1 JOIN table2 ON table1.primaire = table2.cle_etrangere`

   où il faut remplacer les noms des tables, ainsi que les noms de la clé primaire (notée `primaire` ici) et de la colonne y faisant référence (notée `cle_etrangere` ici)

Remarquons que nous avons utilisé des noms qualifiés (de la forme *table.colonne*) pour éviter toute ambiguïté.

Conseil : utiliser un fichier texte pour sauvegarder les requêtes qui fonctionnent.

I Explorer la base de données

Exercice 1
Lister les attributs (ou colonnes) des tables `countries` et `regions`. Identifier la clé primaire puis donner pour chaque table un autre attribut unique.

Exercice 2
Trouver les liens permettant de faire des jointures entre les 3 tables de cette base.

II Première requêtes

Exercice 3
Trouver tous les aéroport se situant en France (le code iso est "FR"), puis tous les grands aéroports allemands (code iso "DE").

Exercice 4
Donner les noms, latitudes et longitudes des aéroports sud-américains (trouver le code de contient dans la base).

Exercice 5
En utilisant la commande `SELECT DISTINCT`, trouver tous les types possibles pour les aéroports.
Combien y-a-t-il de bases d'hydravion (dans une seconde requête) ? On utilisera la fonction `COUNT(*)`.

Exercice 6
Donner de nombre d'aéroports dont la latitude est dans $[0, 30]$.

Exercice 7
Donner de nombre d'aéroports situés dans la bande définie par :

$$\text{latitude} \in \left[\frac{\text{longitude}}{2} - 1, \frac{\text{longitude}}{2} + 1\right]$$

On pourra utiliser la fonction `abs`

Exercice 8
Donner le nombre de pays dont le code de pays commence par un A.
Indication : les relations $\leqslant$ et $\geqslant$ entre chaînes de caractères se basent sur l'ordre alphabétique.

III Quelques jointures

Exercice 9
Trouver tous les aéroports de Tanzanie, en utilisant une condition sur le champ `name` de la table `countries` (le nom en question est "Tanzania").

Exercice 10
Trouver tous les aéroports (via une jointure) dans la région dont le nom est “Ohio”.

Exercice 11
Donner le nom du pays, de la région et la ville pour chaque base d'hydravion.

Exercice 12
Plus délicat. Donner les noms des aéroports, noms des pays et villes pour les couples d'aéroports de même latitude. On veut obtenir deux colonnes pour les noms d'aéroports, deux pour les pays, deux pour les villes.

IV Grouper les données

Pour rappel, lorsque l'on utilise une fonction d'agrégation, on peut regrouper les données suivant un attribut (ou colonne).

```
SELECT count(*) c, ...
FROM ...
WHERE ...
GROUP BY nom_colonne
```

Dans ce cas, les lignes partageant la même valeur dans la colonne citée seront regroupées et comptée ensemble. On obtient alors, dans la table résultat, une ligne par compte effectué c'est à dire une ligne par valeur différente dans la colonne nommée `nom_colonne`.

Exercice 13
Trouver le nombre d'aéroport par pays. On donnera ce nombre, plus le code iso du pays.

Exercice 14
Trouver, pour les pays en possédant, le nombre de de base d'hydravion en donnant en plus le nom du pays (et pas seulement son code).

Exercice 15
Donner la liste des noms de régions et le nombre d'aéroport par région, classée par ordre de nombres d'aéroports décroissant. On pourra utiliser `ORDER BY ... DESC`

Exercice 16
Donner pour chaque pays (affiché par son nom), la latitude et longitude moyenne de ses aéroports. On pourra utiliser `avg(colonne)`.

V Conditions calculées

Le résultat du calcul d'un fonction d'agrégation ne peut pas être utilisé directement dans une clause `WHERE`. Pour donner des conditions sur une colonne `count`, par exemple, on doit énoncer cette condition dans une clause `HAVING`

```
SELECT count(*) c, ...
FROM ...
WHERE ...
GROUP BY nom_colonne
HAVING c>=...
```

par exemple.

Exercice 17
Donner les pays qui possèdent entre 0 et 10 aéroports.

Exercice 18
Donner le nombre moyen d'aéroport par pays (en affichant le nom du pays, encore). Indication : on peut utiliser le fait que la valeur de retour d'une requête est une table pour utiliser une table calculée dans la clause FROM.
En déduire les pays qui possèdent moins d'aéroports que la moyenne. Indication : une requête scalaire (table de retour de taille 1×1) peut être utilisée comme une donnée.