```
////////////////////////////////////////////////
/// TP1 SQLite -- nj jan 2021
////////////////////////////////////////////////

En utilisant la base de données chinook.db (de https://www.sqlitetutorial.net), écrire
les requêtes répondant aux questions.

Toutes les ressources utiles pour le TP sont dans le répertoire TP1-ressources.

Le diagramme d'organisation des tables est présenté
* en noir et blanc dans le fichier sqlite-sample-database-diagram.pdf
* avec des couleurs dans le fichier sqlite-sample-database-diagram-color.pdf

Les résultats des questions de ce TP étant donnés, vous pouvez comparer votre résultats
avec le résultat attendu à l'aide de la commande diff.
Pour la question XXX, vous avez la réponse res-XXX.txt.
Pour éviter les erreurs de manipulation, les fichiers res-XXX.txt sont en lecture seule.

L'objet du TP est de donner les commandes permettant d'obtenir les réponses res-XXX.txt.
En cas de mauvaise réponse, inutile de donner la commande que vous avez trouvé.

Pour mémoire, la commande wc permet de compter les caractères, les mots et les lignes.
La commande 'cat FILE | wc -c' compte les caractères dans le fichier FILE.
La commande 'cat FILE | wc -w' compte les mots dans le fichier FILE.
La commande 'cat FILE | wc -l' compte les lignes dans le fichier FILE.

Ainsi la commande 'sqlite3 BASE "UNE_REQUETE_SQLITE" | wc -l' compte les lignes dans le
résultat de la requête UNE_REQUETE_SQLITE sur la base BASE.

Ensuite on pourra rediriger votre résultat dans out.txt et faire la différence avec le
fichier res-XXX.txt avec la commande diff.

Exemple :
01) afficher l'ensemble de la table tracks
On suppose SELECT * FROM tracks;
$> sqlite3 chinook.db "SELECT * FROM tracks;" > out.txt
$> diff -q out.txt res-01.txt
$>
Ce qui valide la réponse.

Autre exemple :
02) le nombre de tables de la base chinook
On suppose la commande suivante
$> sqlite3 chinook.db ".tables" | wc -w > out.txt
$> diff -q out.txt res-02.txt
$>
Ce qui valide la réponse.

On suppose par erreur la commande suivante
$> sqlite3 chinook.db ".tables" > out.txt
$> diff -q out.txt res-02.txt
Les fichiers out.txt et res-02.txt sont différents
$>
Ce qui montre que la réponse n'est pas la bonne.

Autre exemple dans le cas d'un SELECT
On peut compter les réponses d'un SELECT avec SELECT COUNT ou avec SELECT ... | wc -l
Les deux lignes suivantes place le meme resultat dans out.txt
$> sqlite3 chinook.db "SELECT * FROM tracks;" | wc -l > out.txt
$> sqlite3 chinook.db "SELECT COUNT(*) FROM tracks;" > out.txt

////////////////////////////////////////////////
/// questions sur des commandes SQLite
////////////////////////////////////////////////

03) afficher les champs de la table tracks

04) le nombre d'index de la table tracks
```

```
////////////////////////////////////////////////
/// questions sur des SELECT
////////////////////////////////////////////////

05) le nombre de lignes dans la table tracks

06) afficher l'ensemble de la table tracks;

07) le nombre de lignes dans la table tracks dont Mozart est le compositeur

08) afficher les noms de la table tracks dont Mozart est le compositeur

09) afficher les oeuvres de Mozart de la table tracks par ordre croissant
(chaque ligne contient les champs name et composer)

10) afficher les oeuvres de Mozart de la table tracks par ordre décroissant
(chaque ligne contient les champs name et composer)

11) afficher les oeuvres de Mozart de la table tracks par ordre croissant de deuxieme
colonne
(chaque ligne contient tous les champs de la table tracks)

12) afficher les oeuvres sans compositeur

13) après consultation de la structure de la table employees,
afficher les villes de la table employees:
a) simplement
b) par ordre croissant des villes
c) par ordre croissant des villes et sans doublons

14) afficher les enregistrements de la table tracks les enregistrements ayant une durée
supérieure à 5 minutes parmi les 9 premiers albums
(chaque ligne contient trackid, albumid, nom, temps en millisec)

15) afficher les Noms, Prénoms de la table employees dont les prenoms commencent par M

16) afficher l'enregistrement de la table employees avec l'id 5

17) afficher les enregistrements de la table employees avec les id 1 à 3
a) en utilisant WHERE ... IN ...
b) en utilisant WHERE ... BETWEEN ... AND ...

18) afficher les albums avec plus de 20 tracks (en utilisant COUNT, GROUP_BY et HAVING)
(chaque ligne contient albumid et nombre de tracks de l'album)

////////////////////////////////////////////////
/// questions sur des JOINTURES SIMPLES : INNER JOIN
////////////////////////////////////////////////

19) afficher les 10 premiers titres d'album et noms d'artiste des tables albums et
artistes

20) afficher les 10 premiers noms d'album et genre des tables tracks et genres

21) ajouter à la requète précédente le type de média en utilisant la table media_types

////////////////////////////////////////////////
/// questions sur des JOINTURES LEFT JOIN
////////////////////////////////////////////////

22) compter les artistId distincts dans artists

23) compter les artistId distincts dans albums

(comme les résultats de 21 et 22 différent, il existe des artistes sans album)

24) afficher les artistes sans album et dont le nom commence par A

25) compter les genreId distincts dans genres
```

26) compter les genreId distincts dans tracks

(comme les résultats de 24 et 25 sont identiques, il n'existe pas de genre sans track)

27) afficher les genres sans track

28) compter les trackId distincts dans tracks

29) compter les trackId distincts dans invoice_items

(comme les résultats de 27 et 28 différent, il existe des tracks sans facture)

30) afficher les tracks sans facture et dont le nom commence par Z

//
/// JOINTURES CROSS JOIN
//

31) afficher les noms
a) de la table media_types
b) de la table genres

32) afficher la combinaison des tables media_types et genres

//
/// JOINTURES SELF JOIN (en utilisant INNER JOIN)
//

33) afficher les noms des villes ayant plus d'un employé