

INTRODUCTION AUX BASES DE DONNÉES AVEC SQL

Enseignant :	Louis RAYNAL	Établissement :	ICES
Contact :	l-raynal@ices.fr	Année :	2023-2024
Public :	L3 Maths et M2 Bio-Santé		
Ressources :	https://github.com/LouisRaynal/coursSQL		

TP noté 1

Nom :
Prénom :
Filière :

Rappels : Toute forme de communication entre vous est interdite. Toute triche vous attribuera la note de 0, et peut engendrer une interdiction de participer à tout examen pendant 5 ans.

Documents autorisés : Polycopiés et scripts SQL des cours-TP, bases de données utilisées jusqu'à présent, vos notes de cours, fiches mémo., et autres ressources disponibles à l'adresse <https://github.com/LouisRaynal/coursSQL>.

Important : **Par exercice**, produire un fichier `.sql` comprenant vos requêtes SQL permettant de répondre aux différentes questions posées. Nommez ces fichiers `nom_prénom_exercice_1` et `nom_prénom_exercice_2`, (remplacez `nom` et `prénom` par votre nom et prénom).

A la fin du contrôle, envoyez vos deux fichiers .sql par e-mail à l'adresse l-raynal@ices.fr. Chaque requête doit être exécutable sans modifications de ma part. Ajoutez une ligne de commentaire comme ci-dessous avant chaque requête afin d'indiquer la question concernée.

```
-- Question X
```

Répondre aux questions nécessitant une réponse manuscrite sur le sujet.

Exercice 1

L'objectif de cet exercice est de créer une base de données pour l'ICES. Vous devrez stocker des informations sur les étudiants de l'ICES, les enseignements qui s'y déroulent, et les inscriptions pédagogiques des étudiants (pour déterminer quels étudiants sont inscrits à quels cours).

Il y a donc trois types d'entités qui vous intéressent et pour lesquels vous devrez stocker des informations : les étudiants, les enseignements et les inscriptions pédagogiques. Vous devrez donc créer trois tables que vous nommerez **Etudiants**, **Enseignements** et **Inscriptions**, afin de constituer votre base de données que vous nommerez **ICES**. L'ICES souhaite recueillir les informations ci-dessous.

Pour un étudiant, les informations suivantes :

- le numéro d'étudiant
- la classe
- le nom

- le prénom
- la date de naissance

Pour un enseignement, les informations suivantes :

- un identifiant unique de l'enseignement
- l'intitulé de l'enseignement
- le nom de l'enseignant
- une information indiquant si le cours est une option ou non

Pour une inscription pédagogique, les informations suivantes :

- le numéro de l'étudiant ayant effectué l'inscription
- l'identifiant du cours auquel l'étudiant s'inscrit
- la date et l'heure de l'inscription

Questions :

1. Complétez la liste des informations à recueillir afin que chaque table possède une clé primaire. Soulignez les informations qui auront le rôle de clés primaires.
2. Pour chaque information à stocker, indiquez le nom de colonne de votre choix que vous utiliserez dans les tables de la base de données.
3. Pour chaque information à stocker, écrire un type de colonne SQL approprié selon les données à stocker.
4. Pour chaque information à stocker, écrire les contraintes de colonne SQL appropriées (s'il y en a) qui seront utilisées dans vos différentes tables.
5. Quelles sont les informations qui seront des clés étrangères dans les tables de votre base ?

6. Dessinez le schéma relationnel de votre base ICES. (Le dessiner sur le sujet, dans l'espace après la question 12.)
7. Créez une base de données vide nommée **ICES**, qui sera stockée dans un fichier **ICES.db** **sur le bureau de votre ordinateur**. Rédigez des requêtes de création de table, afin de créer la structure des tables **Etudiants**, **Enseignements** et **Inscriptions** en utilisant vos réponses aux questions précédentes.
8. Rédigez des requêtes d'insertion de données afin d'ajouter **deux** étudiants dans la table **Etudiants**. (Une requête par étudiant est attendue).
9. Rédigez une seule requête d'insertion de données afin d'ajouter **deux** enseignements dans la table **Enseignements**.
10. Rédigez une requête d'insertion de données afin d'ajouter dans la table **Inscriptions** **une** inscription d'un étudiant à un enseignement de votre base, dont la date-heure d'inscription est le 26 octobre 2023 à 13h25.
11. Rédigez une requête de mise à jour de données afin de changer l'enseignement auquel l'étudiant s'est inscrit.
12. Rédigez une requête de suppression de lignes afin de supprimer l'étudiant n'ayant aucune inscription.

Exercice 2

Cet exercice utilise une base de données nommée **YouTube**, stockée dans le fichier **YouTube.db**, téléchargeable depuis le dossier **Examen1** à l'adresse <https://github.com/LouisRaynal/coursSQL>. En utilisant le navigateur web **Google Chrome**, commencez par télécharger cette base de données **sur votre bureau**, et connectez-vous-y depuis **SQLiteStudio**.

Cette base de données contient des vidéos du site YouTube, ainsi que certains commentaires de ces vidéos. Deux tables constituent la base de données : **Vidéos** et **Commentaires**.

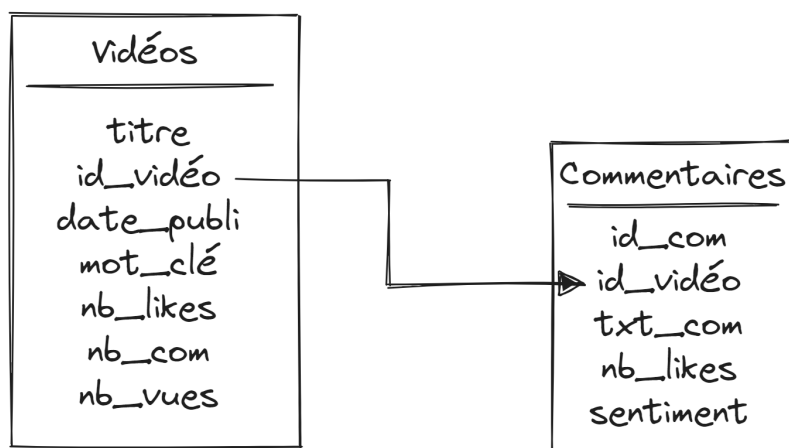
La table **Vidéos** contient les informations suivantes pour chaque vidéo :

- **titre** - le titre de la vidéo ;
- **id_vidéo** - l'identifiant unique à chaque vidéo ;
- **date_publi** - la date de publication de la vidéo ;
- **mot_clé** - un mot clé associé à chaque vidéo ;
- **nb_likes** - le nombre de "J'aime" que la vidéo a reçu. Une valeur égale à -1 signifie que ce nombre n'est pas public ;
- **nb_com** - le nombre de commentaires que la vidéo a reçu. Une valeur égale à -1 signifie que les commentaires ne sont pas activés pour cette vidéo ;
- **nb_vues** - le nombre de vues de la vidéo.

La table **Commentaires** contient les informations suivantes pour chaque commentaire :

- **id_com** - l'identifiant unique de chaque commentaire ;
- **id_vidéo** - l'identifiant de la vidéo dont est issu le commentaire ;
- **txt_com** - le texte que contient le commentaire ;
- **nb_likes** - le nombre de "J'aime" que le commentaire a reçu ;
- **sentiment** - le sentiment auquel se rapporte le commentaire, (**positif**, **négatif** ou **neutre**).

Le schéma relationnel de cette base de données est représenté ci-dessous.



Questions :

Rédigez des requêtes SQL afin d'accomplir les tâches ci-dessous. Il vous faudra rédiger **une requête par question**.

1. Vérifiez qu'il n'existe aucune vidéo avec un titre vide.
2. Affichez les mots clés différents utilisés pour les vidéos, triez ces mots clés par ordre alphabétique.
3. Affichez la date de publication et le titre des vidéos dont le mot clé n'est pas parmi les suivants : **music**, **sports**, **movies**. Utilisez une seule condition.
4. Affichez toutes les informations de la vidéo dont le titre possède trois caractères et se termine par la lettre **d**.
5. Affichez toutes les informations sur les vidéos dont les commentaires sont désactivés ou dont le nombre de "J'aime" n'est pas public.
6. Affichez le titre de la 5ième vidéo avec le plus de commentaires. Affichez également une colonne correspondant à la somme de son nombre de "J'aime" et de son nombre de vues, **en milliers**.
7. Affichez le texte des 10 commentaires les plus courts, ainsi que le sentiment de ces commentaires.
8. Affichez toutes les colonnes des vidéos publiées au mois de septembre.
9. Affichez toutes les colonnes de la table **Commentaires** en ne gardant que les lignes dont le texte du commentaire contient le smiley :) ainsi que l'un des deux smileys suivants : :D ou ^^ . Ordonnez le résultat selon le nombre de "J'aime" décroissant, puis selon l'ordre alphabétique du texte des commentaires.
10. Pour chaque sentiment de commentaire, calculez combien de commentaires ont ce sentiment, ainsi que le nombre moyen de "J'aime" (utiliser la fonction **AVG()** pour le calculer la moyenne d'une colonne). Nommez ces deux colonnes calculées avec un nom de votre choix. N'affichez à la fin les résultats que pour les sentiments dont le nombre moyen de "J'aime" est supérieur ou égal à 1000.
11. Identifiez le titre des deux vidéos dont le titre commence ou se termine par **data science**, avec 30000 "J'aime" ou plus, et avec une année de publication hors 2019.
12. La fonction **SUBSTR(c, n, m)** prend comme arguments une expression **c** s'évaluant en une chaîne de caractères, puis deux entiers **n** et **m**. Cette fonction retourne **m** caractères de la chaîne à partir de la position **n**.

```
-- Par exemple
SELECT SUBSTR('Bonjour', 1, 3)
;
-- Retourne la chaîne de caractères 'Bon', car il s'agit des 3 caractères de la chaîne 'Bonjour' à partir de la position 1.
```

Sachant que vous pouvez utiliser une valeur **n** négative afin de partir de la fin de la chaîne de caractère, utilisez la fonction **SUBSTR** pour afficher le titre des vidéos dont le premier caractère est le même que le dernier.

13. Pour chaque vidéo, affichez son titre ainsi que le nombre de commentaires présents **dans la table Commentaires** pour cette vidéo. Triez les résultats par nombre de commentaires décroissant.