SQL : Exercices BAC

Thème 2 : Base de données

BAC Langage SQL

1. Exercice n°1 : Métropole J1 : Base de données cinématographique

SQL

- 3 relations dans une base de données sur le cinéma
- 2 tables : `individu` et `realisation`

On pourra utiliser les mots clés SQL suivants : `SELECT, FROM, WHERE, JOIN, ON, INSERT, INTO, VALUES, UPDATE, SET, AND`.

Nous allons étudier une base de données traitant du cinéma dont voici le schéma relationnel qui comporte 3 relations :

- la relation `individu (id_ind, nom, prenom, naissance)`
- la relation `realisation (id_rea, titre, annee, type)`
- la relation `emploi (id_emp, description, #id_ind, #id_rea)`

Les clés primaires sont soulignées et les clés étrangères sont précédées d'un `#`.
Ainsi `emploi.id_ind` est une clé étrangère faisant référence à `individu.id_ind`.

Voici un extrait des tables `individu` et `realisation` :

- Extrait de `individu`

id_ind	nom	prenom	naissance
105	'Hulka'	'Daniel'	'01-06-1968'

id_ind	nom	prenom	naissance
403	'Travis'	'Daniel'	'10-03-1968'
688	'Crog'	'Daniel'	'07-07-1968'
695	'Pollock'	'Daniel'	'24-08-1968'

-Extrait de `realisation`

id_rea	titre	annee	type
105	'Casino Imperial'	2006	'action'
325	'Ciel tombant'	2012	'action'
655	'Fantôme'	2015	'action'
950	'Mourir pour attendre'	2021	'action'

1. On s'intéresse ici à la récupération de données dans une relation.

1.a. Décrire ce que renvoie la requête ci-dessous :

🗔 **Requête SQL**

```
SELECT nom, prenom, naissance
FROM individu
WHERE nom = 'Crog';
```

1.b. Fournir une requête SQL permettant de récupérer le titre et la clé primaire de chaque film dont la date de sortie est strictement supérieure à 2020.

2. Cette question traite de la modification de relations.

2.a. Dire s'il faut utiliser la requête 1 ou la requête 2 proposées ci-dessous pour modifier la date de naissance de Daniel Crog. Justifier votre réponse en expliquant pourquoi la requête refusée ne pourra pas fonctionner.

🗔 **Requête SQL 1**

```
UPDATE individu
SET naissance = '02-03-1968'
WHERE id_ind = 688 AND nom = 'Crog' AND prenom = 'Daniel';
```

🗔 **Requête SQL 2**

```
INSERT INTO individu
VALUES (688, 'Crog', 'Daniel', '02-03-1968');
```

2.b. Expliquer si la relation **`individu`** peut accepter (ou pas) deux individus portant le même nom, le même prénom et la même date de naissance.

3. Cette question porte sur la notion de clés étrangères.

3.a. Recopier sur votre copie les demandes ci-dessous, dans leur intégralité, et les compléter correctement pour qu'elles ajoutent dans la relation emploi les rôles de Daniel Crog en tant que James Bond dans le film nommé `'Casino Impérial'` puis dans le film `'Ciel tombant'`.

Requête SQL

```
INSERT INTO emploi
VALUES (5400, 'Acteur(James Bond)', ... , ... );

INSERT INTO emploi
VALUES (5401, 'Acteur(James Bond)', ... , ...);
```

3.b. On désire rajouter un nouvel emploi de Daniel Crog en tant que James Bond dans le film `'Docteur Yes'`.
Expliquer si l'on doit d'abord créer l'enregistrement du film dans la relation **`realisation`** ou si l'on doit d'abord créer le rôle dans la relation **`emploi`**.

4. Cette question traite des jointures.

4.a. Recopier sur votre copie la requête SQL ci-dessous, dans son intégralité, et la compléter de façon à ce qu'elle renvoie le nom de l'acteur, le titre du film et l'année de sortie du film, à partir de tous les enregistrements de la relation **`emploi`** pour lesquels la description de l'emploi est `'Acteur(James Bond)'`.

Requête SQL

```
SELECT ...
FROM emploi
JOIN individu ON ...
JOIN realisation ON ...
WHERE emploi.description = 'Acteur(James Bond)';
```

4.b. Fournir une requête SQL permettant de trouver toutes les descriptions des emplois de Denis Johnson (Denis est son prénom et Johnson est son nom).
On veillera à n'afficher que la description des emplois et non les films associés à ces emplois.

2. Exercice n°2 : D'après 2022, Métropole, J2

SQL

- 2 relations dans une base de données sur la musique
- 2 tables : **`morceaux`** et **`interpretes`**

On pourra utiliser les mots clés SQL suivants : `SELECT, FROM, WHERE, JOIN, ON, INSERT, INTO, VALUES, UPDATE, SET, AND`.

La clause `ORDER BY` suivie d'un attribut permet de trier les résultats par ordre croissant de l'attribut. L'instruction `COUNT(*)` renvoie le nombre de lignes d'une requête.

Un musicien souhaite créer une base de données relationnelle contenant ses morceaux et interprètes préférés. Pour cela il utilise le langage SQL.

Il crée une table **`morceaux`** qui contient entre autres attributs les titres des morceaux et leur année de sortie :

- Table **`morceaux`**

`id_morceau`	**`titre`**	**`annee`**	**`id_interprete`**
1	Like a Rolling Stone	1965	1
2	Respect	1967	2
3	Imagine	1970	3
4	Hey Jude	1968	4
5	Smells Like Teen Spirit	1991	5
6	I Want To hold Your Hand	1963	4

Il crée la table **`interpretes`** qui contient les interprètes et leur pays d'origine :

- Table **`interpretes`**

`id_interprete`	**`nom`**	**`pays`**
1	Bob Dylan	États-Unis
2	Aretha Franklin	États-Unis
3	John Lennon	Angleterre
4	The Beatles	Angleterre
5	Nirvana	États-Unis

`id_morceau` de la table **`morceaux`** et `id_interprete` de la table **`interpretes`** sont des clés primaires.

L'attribut `id_interprete` de la table **`morceaux`** fait directement référence à la clé primaire de la table **`interpretes`**.

1.a. Écrire le résultat de la requête suivante :

Requête SQL

```sql
SELECT titre
FROM morceaux
WHERE id_interprete = 4;
```

1.b. Écrire une requête permettant d'afficher les noms des interprètes originaires d'Angleterre.

1.c. Écrire le résultat de la requête suivante :

Requête SQL

```sql
SELECT titre, annee
FROM morceaux
ORDER BY annee;
```

1.d. Écrire une requête permettant de calculer le nombre de morceaux dans la table **`morceaux`**.

1.e. Écrire une requête affichant les titres des morceaux par ordre alphabétique.

2.a. Citer, en justifiant, la clé étrangère de la table **`morceaux`**.

2.b. Écrire un schéma relationnel des tables **`interpretes`** et **`morceaux`**.

2.c. Expliquer pourquoi la requête suivante produit une erreur :

Requête SQL

```sql
INSERT INTO interpretes
VALUES (1, 'Trust', 'France');
```

3.a. Une erreur de saisie a été faite. Écrire une requête SQL permettant de changer l'année du titre « Imagine » en 1971.

3.b. Écrire une requête SQL permettant d'ajouter l'interprète « The Who » venant d'Angleterre à la table **`interpretes`**. On lui donnera un `id_interprete` égal à 6.

3.c. Écrire une requête SQL permettant d'ajouter le titre « My Generation » de « The Who » à la table **`morceaux`**. Ce titre est sorti en 1965 et on lui donnera un `id_morceau` de 7 ainsi que l'`id_interprete` qui conviendra.

4. Écrire une requête permettant de lister les titres des interprètes venant des États-Unis.

3. Exercice n°3 : Métropole, Candidats libres, J2 2021

SQL

- 2 relations dans une base de données sur un CDI
- 3 tables : **`Livres`**, **`Emprunts`** et **`Eleves`**

L'énoncé de cet exercice utilise les mots du langage SQL suivants :
`SELECT FROM, WHERE, JOIN ON, INSERT INTO VALUES, UPDATE, SET, DELETE, COUNT, AND, OR`.
On considère dans cet exercice une gestion simplifiée des emprunts des ouvrages d'un CDI. La base de données utilisée sera constituée de trois relations (ou tables) nommées `Eleves`, `Livres` et `Emprunts` selon le schéma relationnel suivant :

- **`Livres`** (`isbn (CHAR 13)`, `titre (CHAR)`, `auteur (CHAR)`)
- **`Emprunts`** (`idEmprunt (INT)`, `#idEleve (INT)`, `#isbn (CHAR 13)`, `dateEmprunt (DATE)`, `dateRetour (Date)`)
- **`Eleves`** (`idEleve (INT)`, `nom (CHAR)`, `prenom (CHAR)`, `classe (CHAR)`)

Dans ce schéma relationnel, un attribut souligné indique qu'il s'agit d'une clé primaire.

Le symbole # devant un attribut indique qu'il s'agit d'une clé étrangère. Ainsi, l'attribut `idEleve` de la relation `Emprunts` est une clé étrangère qui fait référence à la clé primaire `idEleve` de la relation `Eleves`. De même l'attribut `isbn` de la relation `Emprunts` est une clé étrangère qui fait référence à la clé primaire `isbn` de la relationcompléter `Livres`.

1. Expliquer pourquoi le code `SQL` ci-dessous provoque une erreur.

Requête SQL

```
INSERT INTO Eleves VALUES (128, 'Dupont', 'Jean', 'T1') ;
INSERT INTO Eleves VALUES (200, 'Dupont', 'Jean', 'T1') ;
INSERT INTO Eleves VALUES (128, 'Dubois', 'Jean', 'T2') ;
```

2. Dans la définition de la relation `Emprunts`, qu'est-ce qui assure qu'on ne peut pas enregistrer un emprunt pour un élève qui n'a pas encore été inscrit dans la relation `Eleves` ?

3. Écrire une requête `SQL` qui renvoie les titres des ouvrages de Molière détenus par le CDI.

4. Décrire le résultat renvoyé par la requête ci-dessous.

Requête SQL

```
SELECT COUNT(*)
FROM Eleves
WHERE classe = 'T2' ;
```

5. Camille a emprunté le livre « *Les misérables* ». Le code ci-dessous a permis d'enregistrer cet emprunt.

Requête SQL

```
INSERT INTO Emprunts
VALUES (640, 192, '9782070409228', '2020-09-15', NULL);
```

Camille a restitué le livre le 30 septembre 2020. Recopier et compléter la requête ci-dessous de manière à mettre à jour la date de retour dans la base de données.

Requête SQL

```
UPDATE Emprunts
SET ......................
WHERE ...................... ;
```

6. Décrire le résultat renvoyé par la requête ci-dessous.

Requête SQL

```
SELECT DISTINCT nom, prenom
FROM Eleves, Emprunts
WHERE Eleves.idEleve = Emprunts.idEleve
AND Eleves.classe = 'T2' ;
```

7. Écrire une requête SQL qui permet de lister les noms et prénoms des élèves qui ont emprunté le livre « *Les misérables* ».