# Sujet BAC 1 : Langage SQL

Sujets BAC - SQL

## 1. Exercice n°1 : Centres-Etrangers 2023

L'énoncé de cet exercice utilise les mots clefs du langage SQL suivants :

**Requête SQL**

```
SELECT, FROM, WHERE, JOIN...ON, UPDATE...SET, DELETE, INSERT INTO...VALUES, ORDER BY.
```

La clause `ORDER BY` suivie d'un attribut permet de trier les résultats par ordre croissant des valeurs de l'attribut.

Radio France souhaite créer une base de données relationnelle contenant les podcasts des émissions de radio. Pour cela elle utilise le langage `SQL`. Elle crée :

- une relation (ou table) **podcast** qui contient le thème et l'année de diffusion.
- une relation **emission** qui contient les émissions (**id_emission**, **nom**), la radio de diffusion et l'animateur.
- une relation **description** qui contient un résumé et la durée du podcast en minutes.
- Relation **podcast**

| id_podcast | theme | annee | id_emission |
|---|---|---|---|
| 1 | Le système d'enseignement supérieur français est-il juste et efficace ? | 2022 | 10081 |
| 2 | Trois innovations pour la croissance future (1/3) : La révolution blockchain. | 2021 | 10081 |
| 3 | Travailleurs de plateformes : vers un nouveau prolétariat ? | 2021 | 10175 |
| 4 | Le poids de la souveraineté numérique française | 2019 | 10183 |
| 40 | Le poids de la souveraineté numérique française | 2019 | 10183 |
| 5 | Dans le cloud en Islande, terre des data center | 2019 | 10212 |

- Relation **emission**

| id_emission | nom | radio | animateur |
|---|---|---|---|
| 10081 | Entendez-vous l'éco ? | France culture | Tiphaine De R. |
| 10175 | Le Temps du débat | France culture | Léa S. |
| 10183 | Soft power | France culture | Frédéric M. |
| 10212 | La tête au carré | France inter | Mathieu V. |

**id_podcast** de la relation **podcast** et **id_emission** de la relation **emission** sont des clés primaires.
L'attribut **id_emission** de la relation **podcast** fait directement référence à la clé primaire de la relation **emission**.

- Relation **description**

| id_description | resume | duree | id_emission |
|---|---|---|---|
| 101 | Autrefois réservé à une élite, l'enseignement supérieur français s'est profondément démocratisé : donne-t-il pour autant les mêmes chances à chacun ? | 4 | 10081 |
| 102 | Quelles sont leurs conditions de travail et quels sont leurs moyens de contestation ? | 58 | 10175 |
| 103 | La promesse de la blockchain, c'est la suppression des intermédiaires et la confiance à grande échelle. | 4 | 10081 |

1. Écrire le schéma relationnel de la relation **description**, en précisant les attributs et leurs types probables, la clé primaire et la ou les clé(s) étrangère(s) éventuelle(s).
2. a. Écrire ce qu'affiche la requête suivante appliquée aux extraits précédents :

   **❐ Requête SQL**

   ```sql
   SELECT theme, annee FROM podcast WHERE id_emission = 10081 ;
   ```

   b. Écrire une requête `SQL permettant d'afficher les thèmes des podcasts de l'année 2019.

   c. Écrire une requête `SQL` affichant la liste des thèmes et des années de diffusion des podcasts dans l'ordre chronologique des années.
3. a. Décrire simplement le résultat obtenu avec cette requête `SQL`.

   **❐ Requête SQL**

   ```sql
   SELECT DISTINCT theme FROM podcast ;
   ```

b. Écrire une requête `SQL` supprimant la ligne contenant **id_podcast=40** de la relation **podcast**.

4. a. Une erreur de saisie a été faite dans la relation **emission**. Écrire une requête `SQL` permettant de changer le nom de l'animateur de l'émission "Le Temps du débat" en "Emmanuel L.".
   b. Écrire une requête `SQL` permettant d'ajouter l'émission "Hashtag" sur la radio "France inter" avec "Mathieu V.". On lui donnera un **id_emission** égal à 12850.

   c. Écrire une requête permettant de lister les thèmes, le nom des émissions et le résumé des podcasts pour lesquels la durée est strictement inférieure à 5 minutes.

**Correction**

## 2. Exercice n°2 : France 2023 - J1

*Cet exercice porte sur la notion de base de données relationnelle et le langage SQL.*

On pourra utiliser les mots-clés `SQL` suivants :

**Requête SQL**

```
AND, FROM, INSERT, INTO, JOIN, ON, SELECT, SET, UPDATE, VALUES, WHERE.
```

Un grand magasin de meubles propose à ses clients un large choix de meubles. Les informations correspondantes sont rangées dans une base de données composée de trois relations.

Voici le schéma de deux de ces relations :
- `Clients (id, nom, prenom, adresse, ville)`
- `Commandes (id, #idClient, #idMeuble, quantite, date)`

Dans ce schéma :
- la clé primaire de chaque relation est définie par les attributs soulignés ;
- les attributs précédés de # sont les clés étrangères.

La troisième relation est appelée **Meubles** et concerne les meubles du magasin.
Le tableau de la figure 1 ci-dessous en présente un extrait :

| id | intitule | prix | stock | description |
|---|---|---|---|---|
| 62 | 'skap' | 69.99 | 2 | 'Armoire blanche 3 portes' |
| 63 | 'skap' | 69.99 | 3 | 'Armoire noire 3 portes' |
| 74 | 'stol' | 39.99 | 10 | 'Chaise en bois avec tissu bleu' |
| 98 | 'hylla' | 99.99 | 0 | 'Bibliothèque 5 étages blanche' |

Figure 1 - Extrait de la relation Meubles

1. Dans cette question, on s'intéresse au modèle relationnel.
   a. Donner la caractéristique qu'un attribut doit avoir pour être choisi comme clé primaire.
   b. Expliquer le rôle des deux clés étrangères de la relation **Commandes**.
   c. Donner le schéma relationnel de la relation **Meubles** en précisant la clé primaire et les éventuelles clés étrangères.
2. En vous basant uniquement sur les données du tableau de la figure 1, donner le résultat de la requête suivante :

   **Requête SQL**

   ```
   SELECT id, stock, description
   FROM Meubles
   WHERE intitule = 'skap' ;
   ```

3. Donner la requête `SQL` permettant d'afficher les noms et prénoms des clients habitant à Paris.
4. Le magasin vient de recevoir des meubles dont l'intitulé est **hylla** et dont l'attribut **id** dans la relation **Meubles** vaut 98. Le stock de ces meubles est alors de 50.
   Recopier et compléter la requête SQL ci-dessous qui permet de mettre à jour la base de données.

   **Requête SQL**

   ```
   UPDATE ...
   SET ...
   WHERE ...
   ```

5. Le magasin vient d'ajouter à son catalogue un nouveau meuble dont les caractéristiques sont les suivantes :

   | id | intitule | prix | stock | description |
   |---|---|---|---|---|
   | 65 | 'matta' | 95.99 | 25 | 'Tapis vert à pois rouges' |

   Donner la requête `SQL` qui permet d'ajouter cet article à la relation **Meubles**.
6. Donner la requête SQL permettant de récupérer le nom et le prénom des différents clients qui ont passé une commande le 30 avril 2021.
   On précise que, dans la relation **Commandes**, les dates sont des chaînes de caractères, par exemple **'21/08/2002'**.

**Correction**

## 3. Exercice n°3 : France 2023 J2

*Cet exercice porte sur les bases de données et le langage **SQL**.*

On considère une gestion simplifiée des voyages dans l'espace. La base de données utilisée est constituée de quatre relations nommées **Astronaute**, **Fusee**, **Equipe** et **Vol**. Voici le contenu des tables **Astronaute**, **Fusee**, **Equipe** et **Vol**.

Les clés primaires sont soulignées et les clefs étrangères sont précédées d'un # :

- **Astronaute**

| id_astronaute | nom | prenom | nationalite | nb_vols |
|---|---|---|---|---|
| 1 | 'PESQUET' | 'Thomas' | 'français' | 2 |
| 2 | 'AMSTRONG' | 'Neil' | 'américain' | 8 |
| 3 | 'MAURER' | 'Mathias' | 'allemand' | 1 |
| 4 | 'MCARTHUR' | 'Megan' | 'américain' | 5 |

- **Fusee**

| id_fusee | modele | constructeur | nb_places |
|---|---|---|---|
| 1 | 'Falcon 9' | 'SpaceX' | 6 |
| 2 | 'Starship' | 'SpaceX' | 100 |
| 3 | 'Soyouz' | 'TsSKB Progress' | 2 |
| 4 | 'SLS' | 'Boeing' | 6 |

- **Equipe**

| id_vol | #id_astronaute |
|---|---|
| 1 | 1 |
| 1 | 2 |
| 1 | 3 |
| 2 | 1 |
| 2 | 3 |
| 3 | 1 |

| id_vol | #id_astronaute |
|---|---|
| 3 | 2 |
| 3 | 4 |
| 4 | 2 |
| 4 | 4 |

- **Vol**

| id_vol | #id_fusee | Date |
|---|---|---|
| 1 | 1 | '12/09/2022' |
| 2 | 4 | '25/10/2022' |
| 3 | 3 | '18/11/2022' |
| 4 | 2 | '23/12/2022' |

On pourra utiliser les mots clés suivants :

**Requête SQL**

```
COUNT, FROM, INSERT, INTO, JOIN ON, ORDER BY, SELECT, VALUES, WHERE.
```

- Le mot clé `COUNT` permet de récupérer le nombre d'enregistrements issu de la requête.
  Par exemple, la requête suivante renvoie la valeur 4.

  **Requête SQL**

  ```
  SELECT COUNT (*) FROM Astronaute;
  ```

- Le mot clé `ORDER BY` permet de trier les éléments par ordre alphabétique. Par exemple, la requête suivante :

  **Requête SQL**

  ```
  SELECT modele FROM Fusee ORDER BY modele;
  ```

renvoie la table :

| 'Falcon 9' |
|---|

| 'Falcon 9' |
| --- |
| 'SLS' |
| 'Soyouz' |
| 'Starship' |

1. On s'intéresse ici à la notion de clés primaire et étrangère.
   a. Donner la définition d'une clé primaire.
   b. Dans la table **Astronaute**, la clé primaire est **id_astronaute**.
   Expliquer pourquoi cette requête SQL renvoie une erreur :

   **Requête SQL**

   ```
   INSERT INTO Astronaute
   VALUES (3, 'HAIGNERE', 'Claudie', 'français', 3) ;
   ```

   c. Le schéma relationnel de la table **Astronaute** est :
   `Astronaute (id_astronaute: INT, nom : TEXT, prenom : TEXT, nationalite : TEXT, nb_vols : INT)`
   Écrire le schéma relationnel de la table **Fusee** en précisant le domaine de chaque attribut.
2. On s'intéresse ici à la récupération d'informations issues de la base de données.
   a. Écrire le résultat que la requête suivante renvoie :

   **Requête SQL**

   ```
   SELECT COUNT(*)
   FROM Fusee
   WHERE constructeur = 'SpaceX' ;
   ```

   b. Écrire une requête SQL qui renvoie le modèle et le constructeur des fusées ayant au moins quatre places.
   c. Écrire une requête SQL qui renvoie les noms et prénoms des astronautes dans l'ordre alphabétique du nom.
3. a. Recopier et compléter les requêtes `SQL` suivantes permettant d'ajouter un cinquième vol avec la fusée 'Soyouz' le 12/04/2023 avec l'équipage composé de PESQUET Thomas et MCARTHUR Megan. On ne s'intéresse pas ici à la mise à jour qui suivra.

   **Requête SQL**

   ```
   INSERT INTO Vol VALUES(...);
   INSERT INTO Equipe VALUES(...);
   INSERT INTO ... VALUES(...);
   ```

   b. Écrire une requête SQL permettant d'obtenir le nom et le prénom des astronautes ayant décollé le '25/10/2022'.

**Correction**

# 4. Exercice n°4 : 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' |
| 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' |

| id_rea | titre | annee | type |
|---|---|---|---|
| 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.

**Correction**

## 5. Exercice n°5 : 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 |
|---|---|---|---|

| 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**

```
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**

```
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**

```
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.

**Correction**

## 6. Exercice n°6 : 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**

```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* ».

**Correction**