

Devoir (1h50)*(Calculatrice non autorisée)***Exercice 1** (8 points)

Cet exercice porte sur les bases de données relationnelles et les requêtes SQL.

Dans cet exercice, on pourra utiliser les clauses du langage SQL pour :

- construire des requêtes d'interrogation à l'aide de `SELECT`, `FROM`, `WHERE` (avec les opérateurs logiques `AND`, `OR`), `JOIN ... ON` ;
- construire des requêtes d'insertion et de mise à jour à l'aide de `UPDATE`, `INSERT`, `DELETE` ;
- affiner les recherches à l'aide de `DISTINCT`, `ORDER BY`.

Dans un schéma relationnel, on utilisera les conventions suivantes :

- la clé primaire d'une relation est définie par son attribut souligné ;
- les attributs précédés de # sont les clés étrangères.

Le guitariste Slash possède une incroyable collection de guitares. Maud est une grande fan de Slash. Elle décide de faire un inventaire de la collection de guitares sous la forme d'une base de données relationnelle.

Partie A

Dans cette partie, Maud utilise la relation suivante :

inventaire (id, marque, modele, annee, num_ser, prix)

num_ser représente le numéro de série d'une guitare. Il est unique pour chaque guitare d'une même marque. Le prix est en euro.

Voici un extrait de la table inventaire.

inventaire					
id	marque	modele	annee	num_ser	prix
1	Gibson	Les Paul Goldtop	1956	@70562	100000
2	Gibson	Les Paul Goldtop	1988	81738349	20000
3	Gibson	Les Paul Standard	1959	@90663	250000
4	Gibson	Les Paul Standard	1987	81757532	25000
5	Fender	Telecaster	1952	000230	150000
6	Fender	Telecaster	1965	81345673	10000
7	Fender	Stratocaster	1956	001359	200000
8	Fender	Stratocaster	1965	81757532	15000

1. Expliquer pourquoi l'attribut `num_ser` ne peut pas être une clé primaire de la relation `inventaire`.
2. Donner, sous forme de tableau, le résultat de la requête suivante appliquée à l'extrait de table précédent.


```
SELECT marque, modele
FROM inventaire
WHERE annee = 1956
```
3. Écrire une requête SQL permettant d'obtenir toutes les années du modèle Les Paul Standard dans la collection.
4. Écrire une requête SQL permettant d'obtenir tous les modèles de guitares de la marque Gibson par ordre croissant de l'année dans la collection.
5. Maud a fait une erreur de saisie pour la guitare d'identifiant `id=1`. L'année est en réalité 1957. Écrire une requête SQL permettant de corriger cette erreur de saisie.

Partie B

Maud change de représentation pour l'inventaire de la collection. Dans cette partie, Maud utilise maintenant les trois relations suivantes :

`marque (id, nom)`

`modele (id, nom, #id_marque)`

`guitare (id, #id_modele, annee, num_ser, prix)`

Dans la relation `modele`, `#id_marque` est une clé étrangère reliée à la clé primaire `id` de la relation `marque`. Dans la relation `guitare`, `#id_modele` est une clé étrangère reliée à la clé primaire `id` de la relation `modele`.

Voici des extraits des trois tables `marque`, `modele`, `guitare`.

marque	
id	nom
1	Gibson
2	Fender

modele		
id	nom	id_marque
1	Les Paul Goldtop	1
2	Les Paul Standard	1
3	Telecaster	2
4	Stratocaster	2

guitare				
id	id_modele	annee	num_ser	prix
1	1	1956	@70562	100000
2	1	1988	81738349	20000
3	2	1959	@90663	250000
4	2	1987	81757532	25000
5	3	1952	000230	150000
6	3	1965	81345673	10000
7	4	1956	001359	200000
8	4	1965	81757532	15000

6. Expliquer brièvement, en justifiant, dans quel ordre les trois tables doivent être créées.
7. Écrire une requête SQL permettant d'obtenir le numéro de série et l'année de toutes les guitares Les Paul Standard de la collection.

Maud vient d'apprendre que Slash a fait cadeau d'une de ses guitares à un ami. Elle doit donc la retirer de sa base de données.

8. Écrire une requête SQL permettant de retirer de la collection la guitare d'identifiant `id=3`.

Slash a aussi acheté une guitare d'une marque qu'il n'avait pas encore dans sa collection. Maud décide de la rajouter.

9. Écrire l'ensemble des requêtes SQL permettant d'ajouter la guitare suivante :
 - marque : BC Rich
 - modèle : Mockingbird
 - année : 1992
 - numéro de série : 92R
 - prix : 5000.

On supposera que l'on peut attribuer la valeur 3 pour l'attribut `id` dans la table `marque` pour la marque BC Rich, que l'on peut attribuer la valeur 5 pour l'attribut `id` dans la table `modele` pour le modèle Mockingbird et que l'on peut attribuer la valeur 9 pour l'attribut `id` dans la table `guitare` pour cette guitare.

Maud souhaite connaître la valeur totale des modèles Stratocaster de la collection. Son ami David lui conseille de regarder la fonction `SUM`. La syntaxe pour utiliser cette fonction SQL peut être similaire à celle-ci :

```
SELECT SUM(nom_colonne)
FROM tab
```

Cette requête SQL permet de calculer la somme des valeurs contenues dans la colonne `nom_colonne` de la table `tab`.

10. Écrire une requête SQL permettant de calculer la valeur totale des modèles Stratocaster de la collection de Slash.

Exercice 2 (12 points)

Cet exercice porte sur le codage binaire, les bases de données relationnelles et les requêtes SQL.

Cet exercice est composé de deux parties peu dépendantes entre elles.

Lorsqu'il y a un accident, les pompiers essaient d'intervenir sur les lieux le plus rapidement possible avec les équipes et le matériel adéquats. On s'intéresse à l'étude d'un système informatique simplifié permettant de répondre à certaines de leurs problématiques.

Chaque pompier possède des aptitudes opérationnelles qui lui permettent de tenir un rôle. Lors du départ d'un véhicule (on parle d'agrès), il faut *a minima* un conducteur et un chef d'agrès.

Pour simplifier, on considère que

- un Véhicule Tout Usage (VTU) ne requiert que le duo conducteur et chef d'agrès, donc deux pompiers ;
- un Véhicule de Secours et d'Assistance aux Victimes (VSAV) requiert, en plus du duo conducteur et chef d'agrès, un équipier, donc trois pompiers ;
- un Fourgon Pompe Tonne (FPT) requiert, en plus du duo conducteur et chef d'agrès, un chef d'équipe et un équipier, donc quatre pompiers.

On souhaite entrer dans une base de données l'ensemble de ces informations afin de pouvoir conserver un historique des interventions.

Partie A – Encodage binaire

Afin de gagner de la place mémoire, on décide de coder l'ensemble des aptitudes sur un entier de 8 bits plutôt que d'écrire en toutes lettres "équipier", "chef d'équipe", etc. Cet ensemble d'aptitudes formera la qualification du pompier considéré.

- Un personnel non formé est codé par 0 ;
- l'aptitude "équipier" est codée par le bit de rang 0 (celui de poids le plus faible), soit 2^0 ;
- l'aptitude "chef d'équipe" est codée par le bit de rang 1, soit 2^1 ;
- l'aptitude "chef d'agrès" est codée par le bit de rang 2, soit 2^2 ;
- enfin, l'aptitude "conducteur" est codée par le bit de rang 3, soit 2^3 .

Remarque : un chef d'équipe étant nécessairement équipier, on lui ajoute cette aptitude dans son codage. De même, un chef d'agrès est nécessairement un chef d'équipe et un équipier : on lui ajoute ces aptitudes dans son codage.

1. Justifier que la qualification décimale 11 correspond à un chef d'équipe conducteur.
2. Déterminer le codage décimal de la qualification chef d'agrès conducteur.
3. Expliquer pourquoi dans la situation décrite, un pompier ne peut pas avoir de qualification dont le codage décimale est 4.
4. Avec ce codage sur un octet, indiquer combien de nouvelles aptitudes peuvent être définies.

On considère que chacune des quatre aptitudes aurait pu être codée par une chaîne de 10 caractères dont chaque caractère utilise 1 octet en mémoire.

5. Choisir, avec justification, l'économie mémoire que le codage sur un entier de 8 bits permet de faire comparée au codage basé sur les chaînes de caractères : environ 10 %, 30 %, 50 %, 98 %.

Partie B

On considère maintenant la base de données relationnelle suivante :

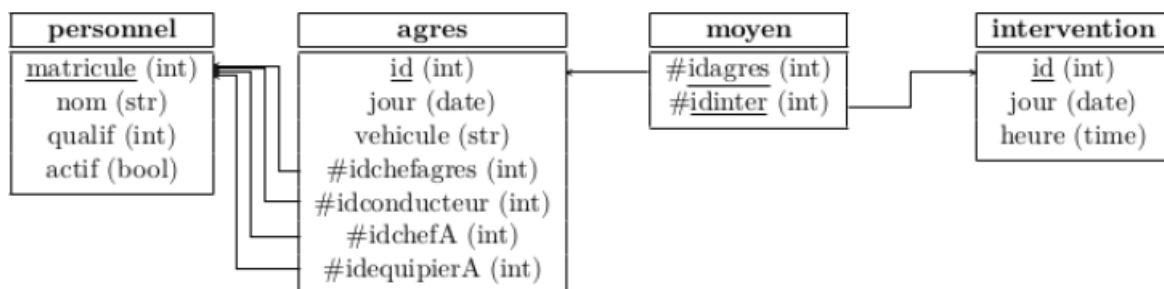


Figure 1. Schéma relationnel

La table *personnel* est composée :

- du matricule unique du pompier ;
- de son nom ;
- de sa qualification (selon le codage vu avant) ;
- d'un attribut qui précise s'il est actif (1) ou inactif (0).

La table *agrès* est composée :

- de son identifiant ;
- du jour où l'agrès se tient prêt à intervenir ;
- du type de véhicule ;
- de l'identifiant du chef d'agrès ;
- de l'identifiant du conducteur ;
- de l'identifiant du chef d'équipe si le véhicule le nécessite, sinon le champ est à NULL ;
- de l'identifiant de l'équipier si le véhicule le nécessite, sinon le champ est à NULL.

La table *moyen* est composée :

- de l'identifiant de l'agrès appelé sur intervention ;
- de l'identifiant de l'intervention.

La table *intervention* est composée :

- de son identifiant ;
- du jour de début d'intervention (sauf pour les longues interventions où il correspond au jour de l'agrès) ;
- de l'heure de début d'intervention.

On rappelle que `COUNT(*)` permet de compter le nombre de lignes extraites lors d'une requête. Par exemple, pour afficher le nombre de personnes dans la table `personnel`, on exécute la requête :

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

`DISTINCT` permet de retirer les doublons des réponses. Par exemple, pour afficher tous les noms distincts de la table `personnel`, on exécute la requête :

```
SELECT DISTINCT(nom) FROM personnel;
```

On considère l'extrait de la base de données ci-dessous :

personnel			
matricule	nom	qualif	actif
10	'Sam'	3	1
16	'Charlot'	1	0
31	'Red'	23	0
83	'Vaillante'	7	1
2501	'Marco'	1	1
2674	'Aicha'	23	1
3004	'Fatou'	7	1
4044	'Abdel'	19	1
4671	'Mamadou'	17	1
5301	'Zoe'	17	1
7450	'Medhi'	3	1
8641	'Gaia'	1	1
8678	'Kevin'	17	0
8682	'Marie'	1	1
9153	'Fred'	23	1

moyen	
idagres	idinter
0	0
2	3
2	4
3	4
4	4
9	5
17	6
22	7
23	8
24	8
24	9

intervention		
id	jour	heure
0	'2023-11-21'	'12:32:21'
1	'2023-11-22'	'22:20:00'
2	'2023-12-17'	'23:17:30'
3	'2024-02-15'	'01:44:06'
4	'2024-02-15'	'12:15:00'
5	'2024-03-02'	'04:58:12'
6	'2024-03-27'	'13:07:18'
7	'2024-05-31'	'05:17:12'
8	'2024-06-11'	'05:38:17'
9	'2024-06-11'	'15:08:56'
10	'2024-06-18'	'07:42:33'

agres						
id	jour	vehicule	idchefagres	idconducteur	idchefA	idequipierA
0	'2023-11-21'	'VSAV'	83	9153	NULL	10
2	'2024-02-15'	'VSAV'	2674	4044	NULL	8641
3	'2024-02-15'	'FPT'	9153	5301	8682	2501
4	'2024-02-15'	'VSAV'	83	4671	NULL	7450
7	'2024-02-29'	'VSAV'	9153	3004	NULL	2501
9	'2024-03-02'	'FPT'	2674	5301	8682	8641
12	'2024-03-21'	'VTU'	3004	5301	NULL	NULL
17	'2024-03-27'	'VSAV'	3004	8682	NULL	10
18	'2024-03-27'	'VSAV'	9153	5301	NULL	10
22	'2024-05-31'	'FPT'	9153	4044	7450	8641
23	'2024-06-11'	'VTU'	83	2674	NULL	NULL
24	'2024-06-11'	'VSAV'	3004	4044	NULL	7450

6. Expliquer la différence entre une clé primaire et une clé étrangère.
7. Expliquer pourquoi la requête suivante génère une erreur pour l'extrait de données.

```
INSERT INTO moyen (idagres, idinter) VALUES (1, 5);
```

8. Proposer une requête SQL qui met à jour l'heure de l'intervention du « 15 février 2024 de 01 heure 44 minutes et 06 secondes » à « 10 heures 44 minutes et 06 secondes ».
9. Préciser le résultat de la requête suivante pour l'extrait de données.

```
SELECT nom FROM personnel  
WHERE actif = 0;
```

10. Proposer une requête SQL qui permet d'afficher les noms des personnels conducteurs actifs. On notera qu'un conducteur possède un attribut `qualif` supérieur ou égal à 8 .
11. Écrire l'affichage obtenu après exécution des deux requêtes ci-dessous sur l'extrait de la base de données. Expliquer ce que chacune des requêtes affiche en général.

Requête A

```
SELECT COUNT(*) FROM agres  
WHERE jour = '2024-03-27';
```

Requête B

```
SELECT COUNT(*) FROM moyen AS m  
INNER JOIN agres AS a ON a.id = m.idagres  
WHERE a.jour = '2024-03-27';
```

12. Proposer une requête qui renvoie sans répétition tous les noms des chefs d'agres assignés à un véhicule le 15 février 2024.
13. Proposer une requête qui renvoie sans répétition tous les noms des chefs d'agres engagés en intervention le 11 juin 2024.