

Matricule : Groupe : Nom :

Partie I : QCM (5 points)

1. Quel est le rôle d'une sous-requête ?
A. Créer une table temporaire B. Exécuter une requête imbriquée dans une autre.
C. Supprimer une ligne D. Gérer les utilisateurs
2. Quelle requête retourne les employés dont le salaire est supérieur au salaire moyen ?
A. `SELECT * FROM employes WHERE salaire > (SELECT AVG(salaire) FROM employes);`
B. `SELECT * FROM employes WHERE salaire = MAX(salaire);`
C. `SELECT * FROM employes WHERE salaire IN (SELECT salaire FROM employes);`
D. `SELECT AVG(salaire) FROM employes;`
3. Quel type de jointure retourne uniquement les lignes avec correspondance dans les deux tables ?
A. FULL JOIN B. LEFT JOIN C. RIGHT JOIN D. INNER JOIN
4. Soit `clients(id, nom)` et `commandes(id, client_id#, date)`.
Comment afficher tous les clients, même ceux sans commande ?
A. INNER JOIN B. LEFT JOIN C. RIGHT JOIN D. Les deux B et C
5. Quel mot-clé est utilisé pour lier deux tables dans une jointure ?
A. CONNECT B. WITH C. ON D. AS
6. Une sous-requête corrélée est une sous-requête qui :
A. Utilise une table externe B. Est indépendante
C. Fait référence à une colonne de la requête principale D. Ne peut jamais être imbriquée
7. Quelle requête est correcte pour une jointure explicite ?
A. `SELECT * FROM A JOIN B ON A.id = B.id;` B. `SELECT * FROM A AND B WHERE A.id = B.id;`
C. `SELECT * FROM A CONNECT B USING (id);` D. `SELECT * FROM A WITH B ON A.id = B.id;`

Partie II : (15 points)

Une entreprise commerciale utilise une base de données « **MagasinBD** » pour gérer ses clients, produits, employés et ventes. Cette base de données permet de suivre les ventes réalisées par les clients.

La base de données « **MagasinBD** » contient six tables principales :

- **Client** (id, nom, tel, email, adresse, ville, pays)
- **Produit** (id, nom, idCategorie#, prixUnitaire, enPromotion)
enPromotion: indique si le produit est en promotion (1 = oui, 0 = non)
- **Catégorie** (id, nom, description)
- **Vente** (id, idClient#, idEmploye#, prixTotal, date, dateExpedition, adresseExpedition)
- **ProduitVendu** (id, idVente#, idProduit#, prixUnitaire, quantite)
- **Employé** (id, nom, prenom, dateNaissance, Adresse, ville, pays)

Exercice1

1. Afficher les noms de tous les produits vendus en 2023.

2. Afficher les noms des clients ayant effectué au moins un achat.

3. Afficher la quantité minimale vendue de chaque produit par l'employé Sidina.

4. Afficher les clients ayant acheté un produit dont le prix unitaire est supérieur à celui de tous les autres produits de la même catégorie.

5. Pour chaque produit, afficher le(s) client(s) en ayant(s) acheté la plus grande quantité

6. Afficher les employés ayant vendu au moins un produit à un client vivant dans la même ville qu'eux.

Exercice 2

Complétez les requêtes Mysql suivantes. Ecrivez seulement les parties manquantes marquées par des pointillés :

- 1- Pour chaque client ayant effectué au moins un achat, la requête affiche l'identifiant de chaque achat réalisé par ce client ainsi que le pourcentage du montant dépensé pour cet achat par rapport au total des dépenses de ce client.

```
SELECT c.nom, v.id, v.date, ROUND(prixTotal * 100.0 /  
(  
.....  
.....  
.....  
.....  
.....  
.....  
)) AS pourcentage  
FROM vente v JOIN client c ON .....
```

- 2- La requête suivante affiche le nom de chaque catégorie et le nombre de produits dans cette catégorie dont le prix unitaire est supérieur au prix unitaire moyen des produits dans la même catégorie.

```
SELECT c.nom AS nomCategorie, ..... AS nbProduits  
FROM Categorie c JOIN Produit p ON c.id = p.idCategorie  
WHERE p.prixUnitaire > (  
.....  
.....  
.....  
.....  
.....  
.....  
)  
GROUP BY .....
```


- 3- La requête suivante affiche pour chaque produit, son nom, la quantité maximale dans laquelle il a été vendue, ainsi que le nombre de ventes correspondant à cette quantité.

```
SELECT p.nom, quantite, COUNT(p.id) FROM produitVendu pv JOIN produit p ON
```

.....

```
WHERE quantite = (
```

```
  SELECT MAX(quantite) FROM produitVendu
```

```
  WHERE .....
```

.....

.....

```
) GROUP BY .....;
```

- 4- Afficher les employés ayant effectué au moins une vente dont le prix total dépasse la moyenne des ventes des employés de leur propre ville.

```
SELECT DISTINCT e.nom, e.prenom FROM Employe e
```

```
JOIN .....
```

```
WHERE prixTotal > (
```

```
  SELECT AVG(v2.prixTotal)
```

.....

.....

.....

.....

```
);
```

Bonne Chance!