TP 2

KHERIMICHE Abdelkarim (0123456)
ABDELKRIM.KHERMICHE@ofppt-edu.ma

October 18, 2025

1. Introduction

Cette collection de plus **50 exercices SQL** propose un ensemble complet de questions conçues pour tester et renforcer vos compétences en SQL. Les exercices couvrent un large éventail de sujets, allant des requêtes de base aux techniques avancées de gestion de bases de données.

Chaque exercice comprend :

— une courte description du contexte,
— la requête SQL correspondante,
— le résultat attendu,
— une explication claire en français.

2. Création de la base de données

2.1. Table `Sales`

La table `Sales` enregistre les informations sur les ventes de produits, incluant la quantité vendue, la date et le prix total. Elle sert de base transactionnelle pour l'analyse des tendances de ventes.

```
CREATE TABLE Sales (
    sale_id INT PRIMARY KEY,
    product_id INT,
    quantity_sold INT,
    sale_date DATE,
    total_price DECIMAL(10, 2),
    FOREIGN KEY (product_id) REFERENCES Products(product_id)
);

INSERT INTO Sales (sale_id, product_id, quantity_sold, sale_date, total_price) VALUES
(1, 101, 5, '2024-01-01', 2500.00),
(2, 102, 3, '2024-01-02', 900.00),
(3, 103, 2, '2024-01-02', 60.00),
(4, 104, 4, '2024-01-03', 80.00),
(5, 105, 6, '2024-01-03', 90.00);
```

Résultat attendu : Table `Sales` correctement créée et remplie.

—

2.2. Table `Products`

La table `Products` contient les détails des produits (nom, catégorie, prix unitaire).

```
CREATE TABLE Products (
    product_id INT PRIMARY KEY,
    product_name VARCHAR(100),
    category VARCHAR(50),
    unit_price DECIMAL(10, 2)
);

INSERT INTO Products (product_id, product_name, category, unit_price) VALUES
(101, 'Laptop', 'Electronics', 500.00),
(102, 'Smartphone', 'Electronics', 300.00),
(103, 'Headphones', 'Electronics', 30.00),
(104, 'Keyboard', 'Electronics', 20.00),
(105, 'Mouse', 'Electronics', 15.00);
```

3. Questions pratiques SQL de selection

1. Récupérer toutes les colonnes de la table `Sales`.
2. Récupérer `product_name` et `unit_price` de la table `Products`.
3. Récupérer `sale_id` et `sale_date` de la table `Sales`.
4. Filtrer la table `Sales` pour afficher uniquement les ventes dont `total_price` est supérieur à 100 $.
5. Filtrer la table `Products` pour afficher uniquement les produits appartenant à la catégorie `Electronics`.
6. Récupérer `sale_id` et `total_price` des ventes effectuées le 3 janvier 2024.
7. Récupérer `product_id` et `product_name` des produits dont le `unit_price` est supérieur à 100 $.
8. Calculer le chiffre d'affaires total généré par toutes les ventes.
9. Calculer le prix unitaire moyen des produits.
10. Calculer la quantité totale vendue dans la table `Sales`.
11. Compter le nombre de ventes par jour.
12. Récupérer le `product_name` et le `unit_price` du produit ayant le prix unitaire le plus élevé.
13. Récupérer les ventes dont la quantité vendue (`quantity_sold`) est supérieure à 4.
14. Récupérer `product_name` et `unit_price` des produits, triés par prix décroissant.
15. Calculer le total des ventes en arrondissant les valeurs à deux décimales.
16. Calculer le prix moyen des ventes dans la table `Sales`.
17. Récupérer `sale_id` et `sale_date` de la table `Sales`, en formatant la date au format `AAAA-MM-JJ`.
18. Calculer le chiffre d'affaires total généré par les produits appartenant à la catégorie `Electronics`.
19. Récupérer `product_name` et `unit_price` des produits dont le prix est compris entre 20 $ et 600 $.
20. Récupérer `product_name` et `category` de la table `Products`, triés par catégorie en ordre croissant.

4. Exercices SQL de niveau intermédiaire

Ces exercices de niveau intermédiaire visent à approfondir la compréhension des concepts SQL au-delà des requêtes de base, en abordant des notions telles que les jointures, les sous-requêtes, les fonctions d'agrégation et les fonctions de fenêtre.

1. Calculer la quantité totale vendue de produits appartenant à la catégorie `Electronics`.
2. Récupérer le nom du produit et le prix total, en calculant ce dernier comme le produit de `quantity_sold` par `unit_price`.

3. Identifier le produit le plus fréquemment vendu dans la table `Sales`.
4. Trouver les produits qui n'ont pas été vendus dans la table `Products`.
5. Calculer le chiffre d'affaires total généré par les ventes pour chaque catégorie de produits.
6. Trouver la catégorie de produits ayant le prix unitaire moyen le plus élevé.
7. Identifier les produits dont le total des ventes dépasse 30 $.
8. Compter le nombre de ventes effectuées chaque mois.
9. Récupérer les détails des ventes pour les produits dont le nom contient le mot "Smart".
10. Déterminer la quantité moyenne vendue pour les produits dont le prix unitaire est supérieur à 100 $.
11. Récupérer le nom du produit et le chiffre d'affaires total pour chaque produit.
12. Lister toutes les ventes avec les noms de produits correspondants.
13. Calculer le pourcentage du chiffre d'affaires total par catégorie de produits et afficher les trois principales.
14. Classer les produits selon leur chiffre d'affaires total à l'aide d'une fonction de classement.
15. Calculer le total cumulé du chiffre d'affaires pour chaque catégorie de produit.
16. Catégoriser les ventes en "High", "Medium" ou "Low" selon le montant total de la vente.
17. Identifier les ventes dont la quantité vendue est supérieure à la moyenne de toutes les ventes.
18. Extraire le mois et l'année de chaque date de vente et compter le nombre de ventes pour chaque mois.
19. Calculer le nombre de jours écoulés entre la date actuelle et la date de vente pour chaque enregistrement.
20. Identifier les ventes effectuées en semaine et celles réalisées pendant le week-end.

5. Exercices SQL de niveau avancé

Cette section avancée se concentre sur des requêtes SQL complexes utilisant des fonctionnalités telles que les fonctions de fenêtre, les vues, les contraintes, les transactions et les procédures stockées. Ces exercices permettent d'affiner vos compétences pour des analyses de données sophistiquées et une meilleure maîtrise du langage SQL.

1. Lister les trois produits ayant la plus forte contribution au chiffre d'affaires total.
2. Créer une vue `Total_Sales` affichant le montant total des ventes pour chaque produit, avec leurs noms et catégories.
3. Récupérer les détails des produits (nom, catégorie, prix unitaire) pour ceux dont la quantité vendue est supérieure à la moyenne globale.

4. Expliquer l'importance de l'indexation dans une base de données SQL et donner un exemple où un index améliore la performance.
5. Ajouter une contrainte de clé étrangère à la table `Sales` qui référence la colonne `product_id` de la table `Products`.
6. Créer une vue `Top_Products` listant les trois produits les plus vendus selon la quantité totale vendue.
7. Mettre en œuvre une transaction qui déduit automatiquement la quantité vendue du stock de la table `Products` lors de l'enregistrement d'une vente.
8. Créer une requête listant le nom des produits accompagné du nombre total de ventes réalisées.
9. Trouver toutes les ventes dont le prix total est supérieur au prix moyen de toutes les ventes.
10. Analyser l'impact de l'indexation de la colonne `sale_date` sur les performances des requêtes filtrant cette colonne.
11. Ajouter une contrainte de vérification (`CHECK`) sur la colonne `quantity_sold` pour s'assurer qu'elle est toujours supérieure à zéro.
12. Créer une vue `Product_Sales_Info` affichant les informations produits et le nombre total de ventes pour chacun.
13. Développer une procédure stockée `Update_Unit_Price` permettant de mettre à jour le prix unitaire d'un produit selon son identifiant.
14. Implémenter une transaction qui insère un nouveau produit dans la table `Products` et ajoute la vente correspondante dans la table `Sales`.
15. Écrire une requête qui calcule le chiffre d'affaires total généré par chaque catégorie de produits pour l'année 2024.