# Exercice 1

- Ecrire les requêtes SQL qui permettent de:
  - Augmentez de 10% le prix de tous les produits en rupture de stock.
  - Mettre à jour le prix de tous les produits de la catégorie: "Electronique" de 20% si leur quantité en stock est inférieure à 10.
  - Réduire le prix des produits dont le prix dépasse par 1000 la moyenne des prix de tous les produits par 10%.
- - **Augmenter de 10% le prix de tous les produits en rupture de stock:**

```
UPDATE Produits
SET Prix = Prix * 1.1
WHERE QuantiteEnStock = 0;
```

- **Mettre à jour le prix de tous les produits de la catégorie "Electronique" de 20% si leur quantité en stock est inférieure à 10:**

```
UPDATE Produits
SET Prix = Prix * 1.2
WHERE IDCategorie = (SELECT IDCategorie FROM Categories WHERE Nom =
'Electronique')
AND QuantiteEnStock < 10;
```

- **Réduire le prix des produits dont le prix dépasse par 1000 la moyenne des prix de tous les produits par 10%:**

```
UPDATE Produits
SET Prix = Prix * 0.9
WHERE Prix > (SELECT AVG(Prix) FROM Produits) + 1000;
```

# Exercice 2

- Ecrire les requêtes SQL qui permettent de
  - Insérer le client dont le nom est "Dupont", le prénom est "Jean", et email est "jean.dupont@gmail.com" dans la table "Clients".
  - Supprimer la ligne de commande dont le id=10
  - Supprimer toutes les commandes passées il y a plus d'un mois.
  - Ajouter une nouvelle catégorie de produits nommée "Mobilier Moderne". Ensuite, affecter cette nouvelle catégorie aux produits dont les id sont: 1,2 et 3
  - Ajouter une commande passée aujourd'hui dans la table "Commandes" du client dont l'email est "marie.martin@exemple.com"(on suppose que les emails sont uniques dans la

table commandes) . L'état de cette commande est "en cours"

- **Insérer un nouveau client:**

```
INSERT INTO Clients (Nom, Prenom, Email)
VALUES ('Dupont', 'Jean', 'jean.dupont@gmail.com');
```

- **Supprimer une ligne de commande:**

```
DELETE FROM LignesCommande WHERE IDLigneCommande = 10;
```

- **Supprimer les commandes de plus d'un mois:**
- On suppose qu'il n'y a aucun conflit avec la table LignesCommande (les id correspondant ne sont pas référencés dans la table LignezsCommande)

```
DELETE FROM Commandes
WHERE DateCommande < DATEADD(month, -1, GETDATE());
```

- **Ajouter une nouvelle catégorie et l'affecter à des produits:**

```
INSERT INTO Categories (Nom) VALUES ('Mobilier Moderne');
UPDATE Produits
SET IDCategorie = (SELECT IDCategorie FROM Categories WHERE Nom =
'Mobilier Moderne')
WHERE IDProduit IN (1, 2, 3);
```

- **Ajouter une nouvelle commande:**

```
INSERT INTO Commandes (IDClient, DateCommande, EtatCommande)
SELECT IDClient, GETDATE(), 'en cours'
FROM Clients
WHERE Email = 'marie.martin@gmail.com';
```

# Exercice 3

- Ecrire les requêtes SQL qui permettent de
  - Créer une vue à partir de la table clients pour touts les clients dont le nom commence par la lettre "a". Le nom de la vue sera Clients_A.
  - Créer une vue "Chiffre_affaire" des produits. Pour chaque nom de produit, vous calculer la valeur de son chiffre d'affaire.
  - Créer la vue "Clients_régulier" des clients ayant passé plus de 10 commandes.
- **Vue des clients commençant par "A":**

```
CREATE VIEW Clients_A AS
SELECT * FROM Clients WHERE Nom LIKE 'A%';
```

- **Vue du chiffre d'affaires par produit:**

```
CREATE VIEW Chiffre_affaire AS
SELECT Produits.Nom, SUM(LignesCommande.Quantite * Produits.Prix) AS
ChiffreAffaires
FROM Produits
INNER JOIN LignesCommande ON Produits.IDProduit =
LignesCommande.IDProduit
GROUP BY Produits.Nom;
```

- **Vue des clients réguliers:**

```
CREATE VIEW Clients_regulier AS
SELECT Clients.IDClient, Clients.Nom, Clients.Prenom, COUNT(
Clients.Nom) as "count"
FROM Clients
INNER JOIN Commandes on
commandes.IDClient = Clients.IDClient
GROUP BY Clients.IDClient, Clients.Nom, Clients.Prenom
HAVING COUNT( Clients.Nom) > 10
```

# Exercice 4

- Ecrire les requêtes SQL qui permettent de:
  - Retourner la liste des vues qui existent.
  - La liste des bases de données crées aujourd'hui.
  - La liste des champs dont le nom commence par "id", de toutes les tables de base (les vues ne sont pas incluses) .
- **Liste des vues:**

```
SELECT table_name FROM INFORMATION_SCHEMA.VIEWS;
```

```
SELECT TABLE_NAME
FROM INFORMATION_SCHEMA.TABLES
WHERE TABLE_TYPE = 'VIEW'
```

- **bases de données créées aujourd'hui:**

```
SELECT *
FROM sys.databases
```

```
WHERE create_date >= CAST(GETDATE() AS DATE);
```

- **Champs commençant par "id":**

```
SELECT TABLE_NAME, COLUMN_NAME
FROM INFORMATION_SCHEMA.COLUMNS
WHERE COLUMN_NAME LIKE 'id%'
and TABLE_NAME not in (select TABLE_NAME from
INFORMATION_SCHEMA.VIEWS);
```

# Exercice 5

- Ecrire les requêtes SQL qui permettent de:
  - Créez une vue nommée `VueCommandeDetail` qui affiche pour chaque ligne de commande : le nom et prénom du client, le nom du produit, la quantité commandée et le prix total de la ligne (quantité * prix).
  - Trouvez les clients qui n'ont jamais passé de commande.
  - Affichez pour chaque commande, le nom et prénom du client, la date de commande et l'état de la commande sous forme textuelle: "En cours de traitement" pour l'état "en cours". "Commande Expédiée " pour "expediee". Et "Livrée au Client" pour "livre".
  - Augmenter les prix des produits selon leur catégorie. Pour les produits dont le idCatégorie=5, augmentez le prix de 10%. Pour les produits dont le idCatégorie=10, augmenter le prix de 15%.
  - Trouver les clients ayant commandé au moins un produit de la catégorie "Mobilier"
  - Trouver les clients ayant dépensé plus de 2000 €
  - Listez les clients n'ayant acheté aucun produit de la catégorie "Electronique"
- **Création de la vue "VueCommandeDetail"**

```
CREATE VIEW VueCommandeDetail AS
SELECT
    Clients.Nom AS NomClient,
        Clients.Prenom as PrenomClient,
    Produits.Nom AS NomProduit,
    LignesCommande.Quantite,
    LignesCommande.Quantite * Produits.Prix AS PrixTotal
FROM
    LignesCommande
INNER JOIN Commandes  ON LignesCommande.IDCommande = Commandes.IDCommande
INNER JOIN Clients  ON Commandes.IDClient = Clients.IDClient
INNER JOIN Produits  ON LignesCommande.IDProduit = Produits.IDProduit;
```

- **Clients n'ayant jamais commandé:**

- ```
  SELECT * FROM Clients
  ```
  ```

```
WHERE IDClient NOT IN (SELECT IDClient FROM Commandes);
```

- **États de commande en texte:**

```
SELECT Clients.Nom, Clients.prenom, Commandes.DateCommande,
       CASE
           WHEN Commandes.EtatCommande = 'en cours' THEN 'En cours de
traitement'
           WHEN Commandes.EtatCommande = 'expediee' THEN 'Commande
Expédiée'
           ELSE 'Livrée au Client'
       END AS EtatCommande
FROM Commandes
INNER JOIN Clients ON Commandes.IDClient = Clients.IDClient;
```

- **Augmenter les prix par catégorie:**

```
UPDATE Produits
SET Prix = CASE
    WHEN IDCategorie = 5 THEN Prix * 1.1
    WHEN IDCategorie = 10 THEN Prix * 1.15
    ELSE Prix
END;
```

- **Clients ayant commandé du mobilier:**
- première méthode:

```
SELECT DISTINCT clients.*
FROM Clients
INNER JOIN Commandes  ON clients.IDClient = Commandes.IDClient
INNER JOIN LignesCommande  ON Commandes.IDCommande =
lignescommande.IDCommande
INNER JOIN Produits ON lignescommande.IDProduit = Produits.IDProduit
WHERE Produits.IDCategorie = (SELECT IDCategorie FROM Categories WHERE
Nom = 'Mobilier');
    ```
```

- deuxième méthode:

```
SELECT DISTINCT clients.*
FROM Clients
INNER JOIN Commandes  ON clients.IDClient = Commandes.IDClient
INNER JOIN LignesCommande  ON Commandes.IDCommande =
lignescommande.IDCommande
INNER JOIN Produits ON lignescommande.IDProduit = Produits.IDProduit
 INNER JOIN Categories ON Produits.IDCategorie = Categories.IDCategorie
 WHERE Categories.Nom = 'Mobilier';
```
```

```
```

- **Clients ayant dépensé plus de 2000€:**

```
SELECT Clients.IDClient, Clients.Nom, Clients.Prenom
FROM Clients
INNER JOIN Commandes co ON Clients.IDClient = co.IDClient
inner join LignesCommande on LignesCommande.IDCommande = Co.IDCommande
inner join produits on LignesCommande.IDProduit= Produits.IDProduit

GROUP BY Clients.IDClient, Clients.Nom, Clients.Prenom
HAVING SUM(LignesCommande.Quantite * Produits.Prix) > 2000;
```

- **Clients n'ayant pas acheté d'électronique:**

```
SELECT Clients.IDClient, Clients.Nom, Clients.Prenom
    FROM Clients
    left outer JOIN Commandes co ON Clients.IDClient = co.IDClient
        left outer join LignesCommande on LignesCommande.IDCommande =
Co.IDCommande
        left outer join produits on LignesCommande.IDProduit=
Produits.IDProduit
        left outer join Categories on Categories.IDCategorie=
Produits.IDCategorie and Categories.Nom ='Electronique'
        group by Clients.IDClient, Clients.Nom, Clients.Prenom
        having max(Categories.nom) is NULL;
```