RESUME

TP : **PYTHON**

THEME : **Analyse des ventes d'un magasin : quincaillerie**

RÉDIGÉ PAR : **BOCOVI Shalom**

CONTACT : **70 62 45 67**

PROFESSEUR : **Mr LOUKOUME**

Lomé, le 06 Avril 2025

INTRODUCTION :

L'analyse des ventes est un élément clé pour toute entreprise souhaitant optimiser ses performances commerciales. Dans ce travail pratique, nous nous intéressons à l'analyse des ventes d'un magasin de produits alimentaires en nous focalisant sur leur répartition selon les jours de la semaine. Cette étude a pour objectif principal d'identifier les jours où les ventes sont les plus importantes, ce qui permettra de mieux comprendre le comportement des clients et d'ajuster, si nécessaire, les stratégies commerciales (promotions, horaires d'ouverture, gestion des stocks...). pour ce faire, nous utilisons une base de données **SQLite** contenant les informations sur les ventes, les produits et les clients. À l'aide du langage **Python** et des bibliothèques **Pandas**, **NumPy**, **Matplotlib** et **Seaborn**, nous extrayons les données pertinentes, les analysons puis les visualisons sous forme de graphique.
L'analyse porte principalement sur la colonne **date**, à partir de laquelle nous extrayons le jour de la semaine pour agréger les ventes et détecter les tendances journalières.

Étape 1 : Modélisation de la base de données

Produits

id (INTEGER, PRIMARY KEY)

nom (TEXT) categorie (TEXT)

prix (REAL)

Clients

id (INTEGER, PRIMARY KEY)

nom (TEXT) ville (TEXT)

Ventes

- id_vente (INTEGER, PRIMARY KEY)
- id_produit (INTEGER, FOREIGN KEY → Produits)
- id_client (INTEGER, FOREIGN KEY → Clients)
- date (TEXT)

- quantite (INTEGER)
  Relations :

- Une **vente** est liée à un seul **produit** et un seul **client**.
- Un **client** peut effectuer plusieurs **ventes**.
- Un **produit** peut être vendu plusieurs fois.

Diagramme simplifié (E-A)

id	nom	categorie	prix
1	Produit_1	Alimentation	139.95
2	Produit_2	Alimentation	79.01
3	Produit_3	Alimentation	31.17
4	Produit_4	Alimentation	149.98
5	Produit_5	Maison	78.45
6	Produit_6	Maison	182.73
7	Produit_7	Loisirs	164.83
8	Produit_8	Loisirs	59.02
9	Produit_9	Électronique	178.03
10	Produit_10	Maison	100.5
11	Produit_11	Maison	63.16
12	Produit_12	Alimentation	128.38
13	Produit_13	Vêtements	140.75
14	Produit_14	Maison	19.76
15	Produit_15	Électronique	183.9
16	Produit_16	Alimentation	109.05
17	Produit_17	Maison	166.99
18	Produit_18	Loisirs	179.7
19	Produit_19	Alimentation	92.1
20	Produit_20	Vêtements	76.35
21	Produit_21	Alimentation	117.61

id	nom	ville
1	Client_1	Lyon
2	Client_2	Lyon
3	Client_3	Toulouse
4	Client_4	Toulouse
5	Client_5	Marseille
6	Client_6	Paris
7	Client_7	Paris
8	Client_8	Toulouse
9	Client_9	Toulouse
10	Client_10	Bordeaux
11	Client_11	Bordeaux
12	Client_12	Bordeaux
13	Client_13	Paris
14	Client_14	Marseille
15	Client_15	Toulouse
16	Client_16	Bordeaux
17	Client_17	Marseille
18	Client_18	Lyon
19	Client_19	Bordeaux
20	Client_20	Bordeaux
21	Client_21	Bordeaux

id	nom	categorie	prix
1	Produit_1	Alimentation	139.95
2	Produit_2	Alimentation	79.01
3	Produit_3	Alimentation	31.17
4	Produit_4	Alimentation	149.98
5	Produit_5	Maison	78.45
6	Produit_6	Maison	182.73
7	Produit_7	Loisirs	164.83
8	Produit_8	Loisirs	59.02
9	Produit_9	Électronique	178.03
10	Produit_10	Maison	100.5
11	Produit_11	Maison	63.16
12	Produit_12	Alimentation	128.38
13	Produit_13	Vêtements	140.75
14	Produit_14	Maison	19.76
15	Produit_15	Électronique	183.9
16	Produit_16	Alimentation	109.05
17	Produit_17	Maison	166.99
18	Produit_18	Loisirs	179.7
19	Produit_19	Alimentation	92.1
20	Produit_20	Vêtements	76.35
21	Produit_21	Alimentation	117.61
22	Produit_22	Électronique	127.82

:

Étape 2 : Création de la base de données et insertion des données simulées

1. **La création de la base de donnée**

une **connexion SQLite** est établie et le **schéma de la base de données** est chargé à partir d'un fichier externe schema.sql qui contient les requêtes CREATE TABLE :

```
conn = sqlite3.connect('ventes_magasin.db')
cursor = conn.cursor()

# Création des tables à partir du fichier schema.sql
with open('schema.sql', 'r', encoding='utf-8') as f:
    cursor.executescript(f.read())
```

Insertion des produits

50 produits sont générés aléatoirement avec :

- Un nom du type Produit_1, Produit_2, ..., Produit_50
- Une **catégorie** choisie parmi : Électronique, Vêtements, Alimentation, Maison, Loisirs • Un **prix unitaire**

```
produits = []
for i in range(1, 51):
    nom = f"Produit_{i}"
    categorie = random.choice(categories)
    prix = round(random.uniform(10, 200), 2)
    produits.append((i, nom, categorie, prix))
```

Ces produits sont ensuite insérés dans la table Produits :

```
cursor.executemany("INSERT INTO Produits VALUES (?, ?, ?, ?)", produits)
```

Insertion des clients

50 clients sont générés avec un nom et une ville aléatoire parmi Paris, Lyon, Marseille, Toulouse, Bordeaux :

```
clients = []
for i in range(1, 51):
    nom = f"Client_{i}"
    ville = random.choice(villes)
    clients.append((i, nom, ville))
```

Puis insérés dans la table Clients :

```
cursor.executemany("INSERT INTO Produits VALUES (?, ?, ?, ?)", produits)
```

Insertion des ventes

Le script génère ensuite **50 ventes aléatoires** :

- Sélection aléatoire d'un **produit** et d'un **client**
- Quantité aléatoire entre 1 et 5
- Date de la vente située entre aujourd'hui et les 30 derniers jours

```
for i in range(1, 51):
    id_produit = random.randint(1, 50)
    id_client = random.randint(1, 50)
    quantite = random.randint(1, 5)
    date = (datetime.now() - timedelta(days=random.randint(1, 30))).strftime('%Y-%m-%d')
    cursor.execute("INSERT INTO Ventes VALUES (?, ?, ?, ?, ?)", (i, id_produit, id_client, date, quantite))

# Sauvegarde et fermeture
```

Les données sont enregistrées avec conn.commit() et la connexion est fermée. Un message de succès s'affiche :

```
conn.commit()
conn.close()

print("✅ Base de données créée avec succès avec 50 produits, 50 clients et 50 ventes !")
```

2. Création des tables

- Table Produits :

```
CREATE TABLE IF NOT EXISTS Produits (
    id INTEGER PRIMARY KEY,
    nom TEXT,
    categorie TEXT,
    prix REAL
);
```

- Table Clients :

```
CREATE TABLE IF NOT EXISTS Clients (
    id INTEGER PRIMARY KEY,
    nom TEXT,
    ville TEXT
);
```

- Tables Ventes :

```sql
CREATE TABLE IF NOT EXISTS Ventes (
    id INTEGER PRIMARY KEY,
    id_produit INTEGER,
    id_client INTEGER,
    date TEXT,
    quantite INTEGER
);
```

3. Insertion de données simulées

```python
Produits= [
      ('Marteau', 'outillage', 17.50),
      ('Scie', 'otillage', 25.00),
      ('Tournevis', 'outillage', 8.00),
      (' clé à molette', 'outillage', 15.75),
      ('Peinture', 'peinture', 20.50),
      ('pince', 'outillage', 9.30),
      ('vis', 'quincaillerie', 5.00)
]
cursor.executemany("' INSERT INTO Produits (nom_produit, categorie, prix_unitaire)
                    VALUES(?, ?, ?)'", produits)

Clients=[
        ('Dupont', 'dupont@example.com', '0123456789'),
        ('Martin' , 'martin@example.com','0987654321'),
        ('Durand', 'durant@example.com', '0145789632')
]
cursor.executemany("'INSERT INTO Clients ( non_client, email, telephone)
                    VALUES(?, ?, ?)'", Clients)

Ventes= [
          (1, 1, '2025-01-01', 2, 200.0),
          (2, 2, '2025-01-02', 3, 150.0),
          (3, 3, '2025-0-03', 1, 200.à)
]
cursor.executemany("'INSERT INTO Vents ( id_produits, id_client, date_vente, quantite_vendre, montant_total)
                    VALUES(?, ?, ?), Ventes)
```

Étape 3 : Extraction des données avec Python et Analyse des staistique avec Pandas et Nympy

Le script commence par se connecter à la base SQLite (ventes_magasin.db) et extrait les trois tables suivantes à l'aide de pandas.read_sql() :

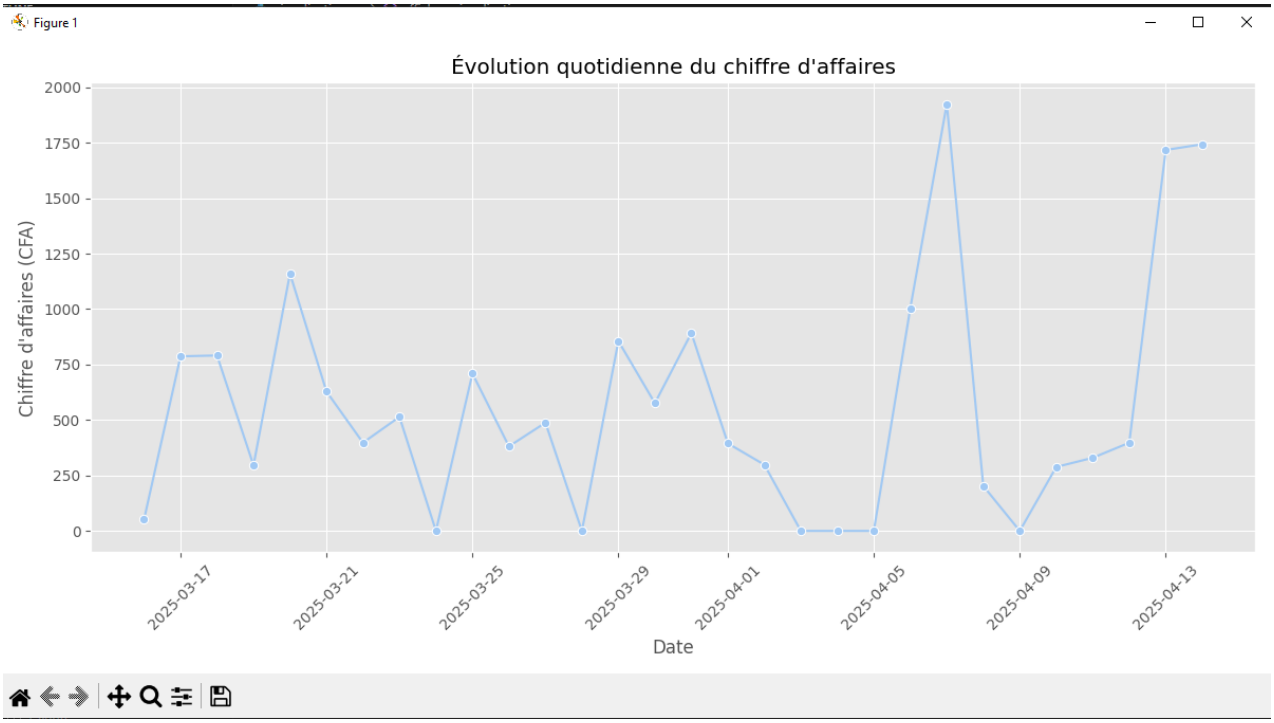
- Produits : informations des produits (nom, catégorie, prix)
- Clients : liste des clients avec leurs villes
- Ventes : chaque ligne correspond à une transaction

Les données sont ensuite fusionnées (pd.merge()) pour constituer un **DataFrame global**. Une nouvelle colonne montant_total est calculée :

```
# Calcul du montant total pour chaque vente
donnees_completes['montant_total'] = donnees_completes['quantite'] * donnees_completes['prix_unitaire']
```

La colonne date_vente est convertie au format datetime pour permettre l'analyse temporelle.

Étape 5 : La Virsualisation des données avec Matplotlib et Seaborn



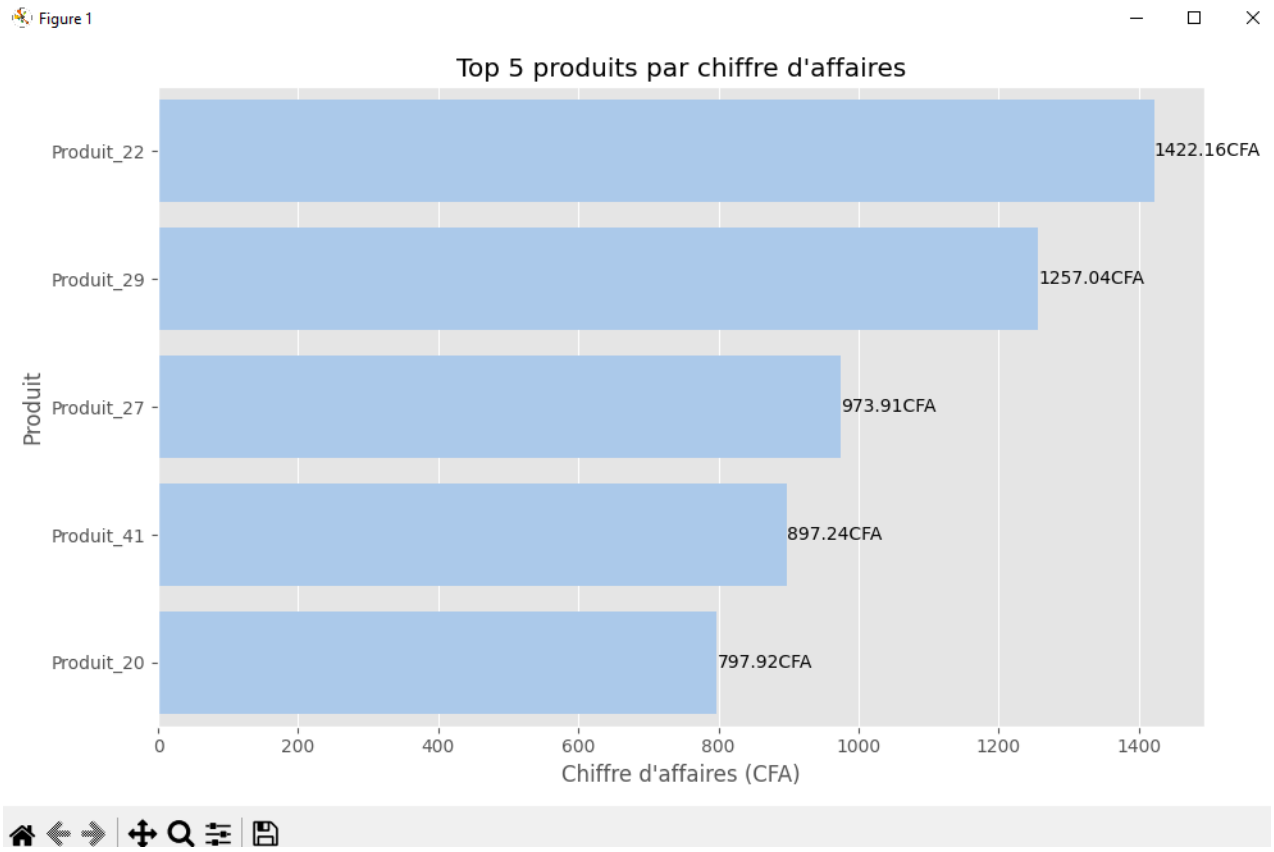
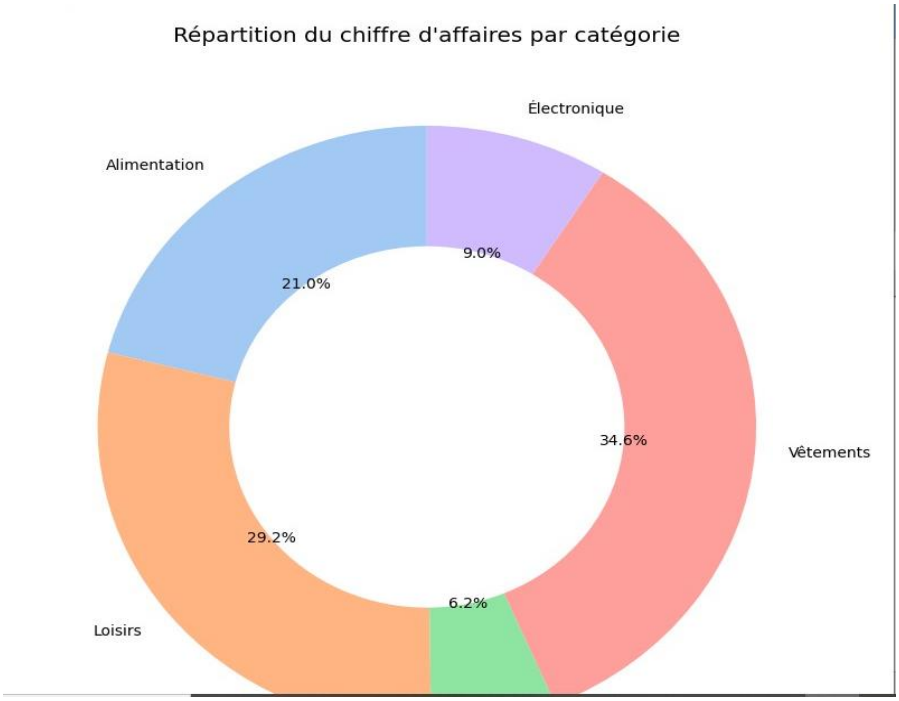


Figure 1
Top 5 produits par chiffre d'affaires
Produit_22
Produit_29
Produit_27
Produit_41
Produit_20
Produit
1422.16CFA
1257.04CFA
973.91CFA
897.24CFA
797.92CFA
0
200
400
600
800
1000
1200
1400
Chiffre d'affaires (CFA)


Répartition du chiffre d'affaires par catégorie
Électronique
Alimentation
9.0%
21.0%
34.6%
Vêtements
29.2%
6.2%
Loisirs

CONCLUSION

En conclusion ce TP permettre de mettre en pratique des compétences essentielles en gestion de bases de données, en analyse statistique avec Pandas et NumPy, ainsi qu'en visualisation des données avec Matplotlib et Seaborn. Nous avons remis un rapport structuré qui présente notre démarche, nos résultats, ainsi que nos visualisations et analyses. De par ce que nous venons de faire, nous sommes convaincus de la réussite de ce TP, donc l'accomplissement de cet objectif.