Projet Final
Data

Membres du groupe

- BRIMESSE Wail GRP-1
- BENAHMED Rayan GRP-2
- FEDDILA Yanis GRP-2
- HASSAOUI Zaid GRP-2
- SKALLI Yanis GRP-3

Sommaire

Introduction

Dans le cadre de ce projet, nous avons réalisé une solution complète de traitement et d'analyse de données allant du scrapping d'un site web jusqu'à la visualisation des résultats après avoir nettoyé et stocké les informations dans une base de données.

Le site pour ce projet est **Books to Scrape**, une plateforme permettant de tester des techniques de scrapping sur des données de livres.

Nous avons listé plusieurs informations clés à récupérer pour chaque livre :

- Le titre
- Le prix (TTC et HT)
- La disponibilité
- Le stock
- La taxe
- La notation en étoiles
- La catégorie

Nous avons donc réalisé les étapes dans cet ordre :

1. Scraping des données
2. Nettoyage et structuration des données
3. Stockage des données dans une base de données
4. Analyse et virtualisation des données

Adopter cette structure étape par étape nous à permis de gagné en efficacité et en qualité de travail, tout en assurant une organisation claire et une meilleure gestion des données.

Travail réalisé

1 - Scraping des données

L'extraction des données a été réalisée à l'aide de **Requests** et **BeautifulSoup,** permettant d'envoyer des requêtes HTTP et de parser le HTML des pages. Un script a été mis en place pour parcourir jusqu'à 30 pages du site et récupérer les données de chaque livre.

2 - Nettoyage et Structuration des données

Une fois les données brutes extraites, elles ont été nettoyées et organisées à l'aide de **Pandas**. Plusieurs étapes ont été mises en place :

- Suppression des espaces inutiles et caractères spéciaux dans les prix
- Conversion des prix en valeurs numériques pour faciliter les analyses
- Gestion des valeurs manquantes

3 - Stockage des données dans une base de données

Pour garantir une meilleure gestion des données et leur réutilisation, elles ont été stockées dans une base de données **MySQL**. Une table dédiée a été créée avec les colonnes correspondantes, et un script Python utilisant MySQL Connector a permis d'insérer les données proprement dans la base.

4 - Analyse et Visualisation des données

Nous avons récupéré les données de la base **MySQL livre**, ajouté une colonne **vente**, et nettoyé les valeurs (prix, stock, etoiles).

Visualisations principales :

- **Répartition des livres** par **notation (etoiles)**, **catégorie** et **intervalle de prix**.
- **Analyse des stocks et des ventes** par **notation, catégorie et prix**.

Ces graphiques permettent de **visualiser les tendances des ventes et du stock**, aidant à optimiser l'approvisionnement et la stratégie de vente.

Base de données

La base de données **sprint_data** contient une table principale livres, qui stocke les informations extraites et nettoyées des livres du site **Books to Scrape**.

Structure de la Table Livres

CREATE TABLE livres (
id int(11) NOT NULL,
titre varchar(255) NOT NULL,
prix decimal(10,2) NOT NULL,
disponibilite varchar(50) NOT NULL,
etoiles int(11) DEFAULT NULL CHECK (etoiles between 0 and 5),
prix_ht decimal(10,2) NOT NULL,
prix_ttc decimal(10,2) NOT NULL,
taxe decimal(10,2) NOT NULL,
categorie varchar(100) NOT NULL,
stock int(11) NOT NULL,
vente int(11) DEFAULT NULL);

Remplissage de la Base

Nous avons intégré dans la base de données les données récupérées sur le site grâce aux commandes **SQL** suivantes :

LOAD DATA INFILE

'C:/Users/Rayan/PycharmProjects/PythonProject/.venv/books_toscrape_cleaned.csv'

INTO TABLE livres

FIELDS TERMINATED BY ';'

LINES TERMINATED BY '\n' IGNORE 1 ROWS

(titre, prix, disponibilite, etoiles, prix_ht, prix_ttc, taxe, categorie, stock);

Objectif

- Stocker les données de manière organisée après extraction et nettoyage.
- Faciliter le traitement des données.
- Préparer les données pour des visualisations ou exportations.

Code et Résultat

1 – Scraping des données

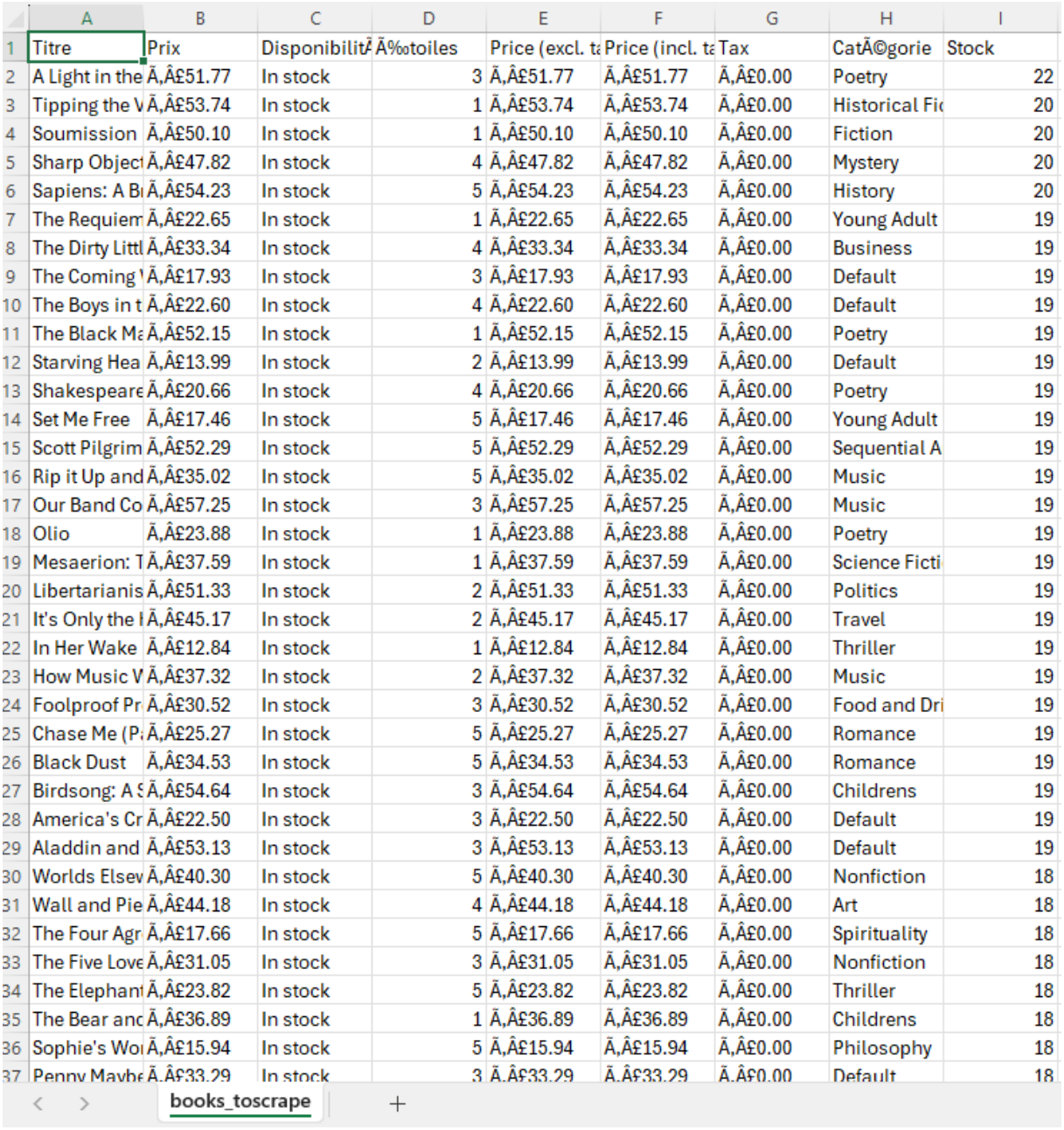
Code :

scrap.py

```python
import requests
from bs4 import BeautifulSoup
import pandas as pd

# URL de base pour accéder aux différentes pages
BASE_URL = "https://books.toscrape.com/catalogue/page-{}.html"

# Headers pour éviter d'être bloqué par le site
HEADERS = {
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36"
}

# Conversion des étoiles en notation numérique
STAR_RATINGS = {
    "One": 1,
    "Two": 2,
    "Three": 3,
    "Four": 4,
    "Five": 5
}

# Dictionnaire pour stocker les données des livres
books = {
    "Titre": [],
    "Prix": [],
    "Disponibilité": [],
    "Étoiles": [],
    "Price (excl. tax)": [],
    "Price (incl. tax)": [],
    "Tax": [],
    "Catégorie": [],
    "Stock": []
}

# Limite du nombre de pages à scraper
max_pages = 30

# Parcourir les pages une par une
for page in range(1, max_pages + 1):
    url = BASE_URL.format(page)
    response = requests.get(url, headers=HEADERS)

    if response.status_code != 200:
        print(f"Problème d'accès à la page {page}, arrêt du scraping.")
        break  # Si une page ne charge pas, on ne continue pas

    soup = BeautifulSoup(response.text, 'html.parser')

    # Récupérer tous les livres affichés sur la page
    items = soup.find_all("article", class_="product_pod")

    if not items:
        print(f"Arrêt du scraping : seulement {page - 1} pages trouvées.")
        break  # Si aucune donnée trouvée, inutile de continuer

    for item in items:
        try:
            title = item.find("h3").find("a")["title"]
            price = item.find("p", class_="price_color").text.strip()
```

```
            price = item.find("p", class_="price_color").text.strip()
            availability = item.find("p", class_="instock availability").text.strip()

            # Récupération des étoiles (notation de 1 à 5)
            star_class = item.find("p", class_="star-rating")["class"]
            star_rating = STAR_RATINGS.get(star_class[1], 0)  # Si pas trouvé, on met 0

            # Accéder à la page du produit pour récupérer les prix détaillés
            product_link = "https://books.toscrape.com/catalogue/" + item.find("h3").find("a")["href"]
            product_response = requests.get(product_link, headers=HEADERS)
            product_soup = BeautifulSoup(product_response.text, "html.parser")

            price_excl_tax = product_soup.find("th", string="Price (excl. tax)").find_next("td").text.strip()
            price_incl_tax = product_soup.find("th", string="Price (incl. tax)").find_next("td").text.strip()
            tax = product_soup.find("th", string="Tax").find_next("td").text.strip()

            # Récupérer la catégorie
            category = product_soup.find("ul", class_="breadcrumb").find_all("a")[2].text.strip()

            # Récupérer le nombre disponible dans le stock
            stock = product_soup.find("p", class_="instock availability").text.strip().replace("In stock (", "").replace(" available)", "")

            # Ajouter les données extraites dans le dictionnaire
            books["Titre"].append(title)
            books["Prix"].append(price)
            books["Disponibilité"].append(availability)
            books["Étoiles"].append(star_rating)
            books["Price (excl. tax)"].append(price_excl_tax)
            books["Price (incl. tax)"].append(price_incl_tax)
            books["Tax"].append(tax)
            books["Catégorie"].append(category)
            books["Stock"].append(stock)

        except Exception as e:
            print(f"Erreur lors de l'extraction d'un livre : {e}")
            continue

# Enregistrement des données dans un fichier CSV
df = pd.DataFrame(books)
df.to_csv("books_toscrape.csv", index=False, encoding='utf-8', sep=";")

print("Scraping terminé ! Données enregistrées dans books_toscrape.csv")
```

Résultat : Extrait de books_toscrape.csv



	A	B	C	D	E	F	G	H	I
1	Titre	Prix	DisponibilitÃ‰‰toiles		Price (excl. tâ	Price (incl. tâ	Tax	CatÃ©gorie	Stock
2	A Light in the	Ã‚Â£51.77	In stock	3	Ã‚Â£51.77	Ã‚Â£51.77	Ã‚Â£0.00	Poetry	22
3	Tipping the V	Ã‚Â£53.74	In stock	1	Ã‚Â£53.74	Ã‚Â£53.74	Ã‚Â£0.00	Historical Fic	20
4	Soumission	Ã‚Â£50.10	In stock	1	Ã‚Â£50.10	Ã‚Â£50.10	Ã‚Â£0.00	Fiction	20
5	Sharp Objec	Ã‚Â£47.82	In stock	4	Ã‚Â£47.82	Ã‚Â£47.82	Ã‚Â£0.00	Mystery	20
6	Sapiens: A Bı	Ã‚Â£54.23	In stock	5	Ã‚Â£54.23	Ã‚Â£54.23	Ã‚Â£0.00	History	20
7	The Requiem	Ã‚Â£22.65	In stock	1	Ã‚Â£22.65	Ã‚Â£22.65	Ã‚Â£0.00	Young Adult	19
8	The Dirty Littl	Ã‚Â£33.34	In stock	4	Ã‚Â£33.34	Ã‚Â£33.34	Ã‚Â£0.00	Business	19
9	The Coming '	Ã‚Â£17.93	In stock	3	Ã‚Â£17.93	Ã‚Â£17.93	Ã‚Â£0.00	Default	19
10	The Boys in t	Ã‚Â£22.60	In stock	4	Ã‚Â£22.60	Ã‚Â£22.60	Ã‚Â£0.00	Default	19
11	The Black Ma	Ã‚Â£52.15	In stock	1	Ã‚Â£52.15	Ã‚Â£52.15	Ã‚Â£0.00	Poetry	19
12	Starving Hea	Ã‚Â£13.99	In stock	2	Ã‚Â£13.99	Ã‚Â£13.99	Ã‚Â£0.00	Default	19
13	Shakespeare	Ã‚Â£20.66	In stock	4	Ã‚Â£20.66	Ã‚Â£20.66	Ã‚Â£0.00	Poetry	19
14	Set Me Free	Ã‚Â£17.46	In stock	5	Ã‚Â£17.46	Ã‚Â£17.46	Ã‚Â£0.00	Young Adult	19
15	Scott Pilgrim	Ã‚Â£52.29	In stock	5	Ã‚Â£52.29	Ã‚Â£52.29	Ã‚Â£0.00	Sequential A	19
16	Rip it Up and	Ã‚Â£35.02	In stock	5	Ã‚Â£35.02	Ã‚Â£35.02	Ã‚Â£0.00	Music	19
17	Our Band Co	Ã‚Â£57.25	In stock	3	Ã‚Â£57.25	Ã‚Â£57.25	Ã‚Â£0.00	Music	19
18	Olio	Ã‚Â£23.88	In stock	1	Ã‚Â£23.88	Ã‚Â£23.88	Ã‚Â£0.00	Poetry	19
19	Mesaerion: T	Ã‚Â£37.59	In stock	1	Ã‚Â£37.59	Ã‚Â£37.59	Ã‚Â£0.00	Science Ficti	19
20	Libertarianis	Ã‚Â£51.33	In stock	2	Ã‚Â£51.33	Ã‚Â£51.33	Ã‚Â£0.00	Politics	19
21	It's Only the H	Ã‚Â£45.17	In stock	2	Ã‚Â£45.17	Ã‚Â£45.17	Ã‚Â£0.00	Travel	19
22	In Her Wake	Ã‚Â£12.84	In stock	1	Ã‚Â£12.84	Ã‚Â£12.84	Ã‚Â£0.00	Thriller	19
23	How Music V	Ã‚Â£37.32	In stock	2	Ã‚Â£37.32	Ã‚Â£37.32	Ã‚Â£0.00	Music	19
24	Foolproof Pr	Ã‚Â£30.52	In stock	3	Ã‚Â£30.52	Ã‚Â£30.52	Ã‚Â£0.00	Food and Dri	19
25	Chase Me (P	Ã‚Â£25.27	In stock	5	Ã‚Â£25.27	Ã‚Â£25.27	Ã‚Â£0.00	Romance	19
26	Black Dust	Ã‚Â£34.53	In stock	5	Ã‚Â£34.53	Ã‚Â£34.53	Ã‚Â£0.00	Romance	19
27	Birdsong: A S	Ã‚Â£54.64	In stock	3	Ã‚Â£54.64	Ã‚Â£54.64	Ã‚Â£0.00	Childrens	19
28	America's Cr	Ã‚Â£22.50	In stock	3	Ã‚Â£22.50	Ã‚Â£22.50	Ã‚Â£0.00	Default	19
29	Aladdin and	Ã‚Â£53.13	In stock	3	Ã‚Â£53.13	Ã‚Â£53.13	Ã‚Â£0.00	Default	19
30	Worlds Elsev	Ã‚Â£40.30	In stock	5	Ã‚Â£40.30	Ã‚Â£40.30	Ã‚Â£0.00	Nonfiction	18
31	Wall and Pie	Ã‚Â£44.18	In stock	4	Ã‚Â£44.18	Ã‚Â£44.18	Ã‚Â£0.00	Art	18
32	The Four Agr	Ã‚Â£17.66	In stock	5	Ã‚Â£17.66	Ã‚Â£17.66	Ã‚Â£0.00	Spirituality	18
33	The Five Love	Ã‚Â£31.05	In stock	3	Ã‚Â£31.05	Ã‚Â£31.05	Ã‚Â£0.00	Nonfiction	18
34	The Elephant	Ã‚Â£23.82	In stock	5	Ã‚Â£23.82	Ã‚Â£23.82	Ã‚Â£0.00	Thriller	18
35	The Bear and	Ã‚Â£36.89	In stock	1	Ã‚Â£36.89	Ã‚Â£36.89	Ã‚Â£0.00	Childrens	18
36	Sophie's Wo	Ã‚Â£15.94	In stock	5	Ã‚Â£15.94	Ã‚Â£15.94	Ã‚Â£0.00	Philosophy	18
37	Penny Maybe	Ã‚Â£33.29	In stock	3	Ã‚Â£33.29	Ã‚Â£33.29	Ã‚Â£0.00	Default	18

books_toscrape

2 – Nettoyage des données

Code :

nettoyage.py

```python
import pandas as pd

# Charger les données avec le bon séparateur et encodage
def charger_donnees(fichier):
    try:
        df = pd.read_csv(fichier, sep=";", encoding="utf-8")
        # Vérifier les valeurs manquantes
        print("Valeurs manquantes par colonne :")
        print(df.isnull().sum())
        return df
    except Exception as e:
        print(f"Erreur lors du chargement des données : {e}")
        return None


# Nettoyer les données
def nettoyer_donnees(df):
    if df is None or df.empty:
        return None

    try:
        # Colonnes numériques à nettoyer
        colonnes_numeriques = ["Prix", "Price (excl. tax)", "Price (incl. tax)", "Tax", "Stock"]

        for col in colonnes_numeriques:
            if col in df.columns:
                # Supprimer les symboles non numérique et convertir en nombres
                df[col] = df[col].astype(str).str.replace("Â£", "", regex=False)
                df[col] = df[col].str.replace("[^0-9.]", "", regex=True)  # Garder uniquement les chiffres et les points
                df[col] = pd.to_numeric(df[col], errors="coerce")  # Convertir en nombres, avec gestion des erreurs

                # Remplacer les valeurs manquantes par la moyenne de la colonne
                moyenne = df[col].mean()
                df[col] = df[col].fillna(moyenne)

        # Nettoyage de la colonne "Stock" en extrayant uniquement le nombre disponible
        if "Stock" in df.columns:
            df["Stock"] = df["Stock"].astype(str).str.extract("(\\d+)").astype(float)

        # Convertir la colonne "Stock" en entiers pour supprimer le ".0"
        if "Stock" in df.columns:
            df["Stock"] = df["Stock"].astype(int)

        return df
    except Exception as e:
        print(f"Erreur lors du nettoyage des données : {e}")
        return None


# Sauvegarder les données nettoyées
def sauvegarder_donnees(df, fichier_sortie):
    if df is None or df.empty:
        print("Erreur lors de la sauvegarde : aucune donnée à enregistrer.")
        return

    try:
        # Vérifier les valeurs manquantes avant la sauvegarde
        print("Valeurs manquantes après nettoyage :")
        print(df.isnull().sum())
```

```
        print(df.isnull().sum())

        df.to_csv(fichier_sortie, sep=";", encoding="utf-8", index=False)
        print("Donnée sauvegardée avec succès !")
    except Exception as e:
        print(f"Erreur lors de la sauvegarde : {e}")


#Exécution du programme
if __name__ == "__main__":
    fichier_entree = "books_toscrape.csv"
    fichier_sortie = "books_toscrape_cleaned.csv"

    df = charger_donnees(fichier_entree)
    df = nettoyer_donnees(df)
    sauvegarder_donnees(df, fichier_sortie)
```

Résultat : le csv nettoyé :

	A	B	C	D	E	F	G	H	I
1	Titre	Prix	DisponibilitÃ‰Ã‰‰toiles		Price (excl. tâ	Price (incl. tâ	Tax	CatÃ©gorie	Stock
2	A Light in the	51.77	In stock	3	51.77	51.77	0.0	Poetry	22
3	Tipping the V	53.74	In stock	1	53.74	53.74	0.0	Historical Fi	20
4	Soumission	50.1	In stock	1	50.1	50.1	0.0	Fiction	20
5	Sharp Objec	47.82	In stock	4	47.82	47.82	0.0	Mystery	20
6	Sapiens: A Bı	54.23	In stock	5	54.23	54.23	0.0	History	20
7	The Requiem	22.65	In stock	1	22.65	22.65	0.0	Young Adult	19
8	The Dirty Littl	33.34	In stock	4	33.34	33.34	0.0	Business	19
9	The Coming '	17.93	In stock	3	17.93	17.93	0.0	Default	19
10	The Boys in t	22.6	In stock	4	22.6	22.6	0.0	Default	19
11	The Black Ma	52.15	In stock	1	52.15	52.15	0.0	Poetry	19
12	Starving Hea	13.99	In stock	2	13.99	13.99	0.0	Default	19
13	Shakespeare	20.66	In stock	4	20.66	20.66	0.0	Poetry	19
14	Set Me Free	17.46	In stock	5	17.46	17.46	0.0	Young Adult	19
15	Scott Pilgrim	52.29	In stock	5	52.29	52.29	0.0	Sequential A	19
16	Rip it Up and	35.02	In stock	5	35.02	35.02	0.0	Music	19
17	Our Band Co	57.25	In stock	3	57.25	57.25	0.0	Music	19
18	Olio	23.88	In stock	1	23.88	23.88	0.0	Poetry	19
19	Mesaerion: T	37.59	In stock	1	37.59	37.59	0.0	Science Ficti	19
20	Libertarianis	51.33	In stock	2	51.33	51.33	0.0	Politics	19
21	It's Only the H	45.17	In stock	2	45.17	45.17	0.0	Travel	19
22	In Her Wake	12.84	In stock	1	12.84	12.84	0.0	Thriller	19
23	How Music V	37.32	In stock	2	37.32	37.32	0.0	Music	19
24	Foolproof Pr	30.52	In stock	3	30.52	30.52	0.0	Food and Dri	19
25	Chase Me (Pı	25.27	In stock	5	25.27	25.27	0.0	Romance	19
26	Black Dust	34.53	In stock	5	34.53	34.53	0.0	Romance	19
27	Birdsong: A S	54.64	In stock	3	54.64	54.64	0.0	Childrens	19
28	America's Cr	22.5	In stock	3	22.5	22.5	0.0	Default	19
29	Aladdin and	53.13	In stock	3	53.13	53.13	0.0	Default	19
30	Worlds Elsev	40.3	In stock	5	40.3	40.3	0.0	Nonfiction	18
31	Wall and Pie	44.18	In stock	4	44.18	44.18	0.0	Art	18
32	The Four Agr	17.66	In stock	5	17.66	17.66	0.0	Spirituality	18
33	The Five Love	31.05	In stock	3	31.05	31.05	0.0	Nonfiction	18
34	The Elephant	23.82	In stock	5	23.82	23.82	0.0	Thriller	18
35	The Bear and	36.89	In stock	1	36.89	36.89	0.0	Childrens	18
36	Sophie's Wo	15.94	In stock	5	15.94	15.94	0.0	Philosophy	18
37	Penny Maybe	33.29	In stock	3	33.29	33.29	0.0	Default	18

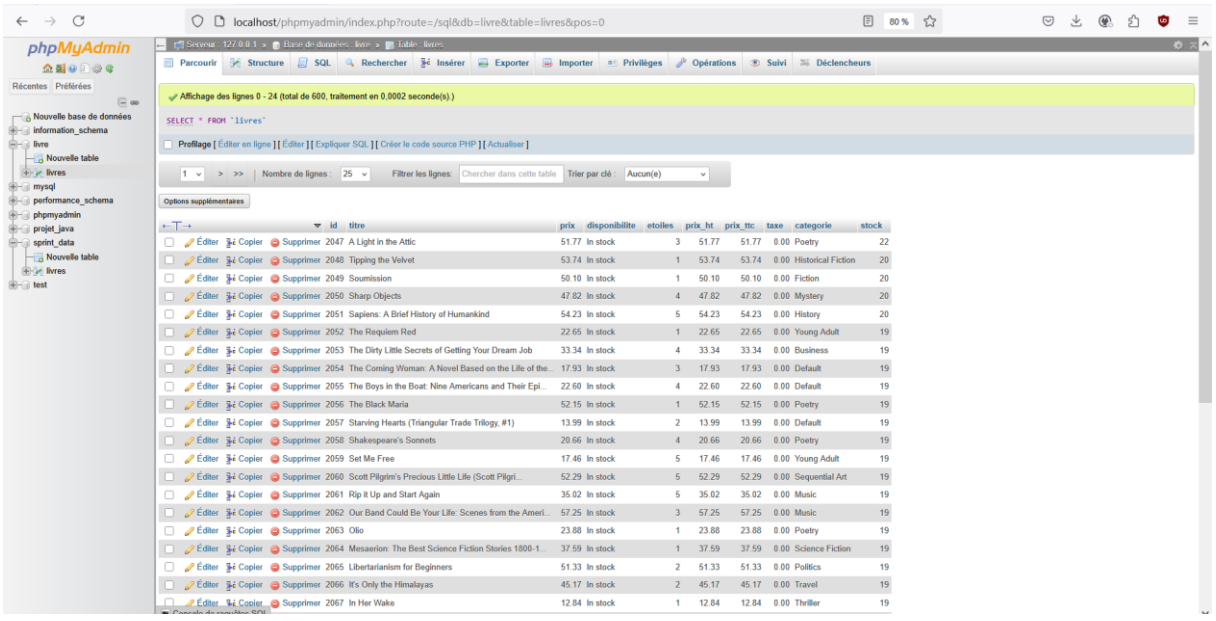
books_toscrape_cleaned

3 – Base de données

Code : Insertion des livres dans la base de données

```sql
--

CREATE TABLE `livres` (
  `id` int(11) NOT NULL,
  `titre` varchar(255) NOT NULL,
  `prix` decimal(10,2) NOT NULL,
  `disponibilite` varchar(50) NOT NULL,
  `etoiles` int(11) DEFAULT NULL CHECK (`etoiles` between 0 and 5),
  `prix_ht` decimal(10,2) NOT NULL,
  `prix_ttc` decimal(10,2) NOT NULL,
  `taxe` decimal(10,2) NOT NULL,
  `categorie` varchar(100) NOT NULL,
  `stock` int(11) NOT NULL,
  `vente` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Déchargement des données de la table `livres`
--

INSERT INTO `livres` (`id`, `titre`, `prix`, `disponibilite`, `etoiles`, `prix_ht`, `prix_ttc`, `taxe`, `categorie`, `stock`, `vente`) VALUES
(1024, 'A Light in the Attic', 51.77, 'In stock', 3, 51.77, 51.77, 0.00, 'Poetry', 22, 18),
(1025, 'Tipping the Velvet', 53.74, 'In stock', 1, 53.74, 53.74, 0.00, 'Historical Fiction', 20, 20),
(1026, 'Soumission', 50.10, 'In stock', 1, 50.10, 50.10, 0.00, 'Fiction', 20, 20),
(1027, 'Sharp Objects', 47.82, 'In stock', 4, 47.82, 47.82, 0.00, 'Mystery', 20, 20),
(1028, 'Sapiens: A Brief History of Humankind', 54.23, 'In stock', 5, 54.23, 54.23, 0.00, 'History', 20, 20),
(1029, 'The Requiem Red', 22.65, 'In stock', 1, 22.65, 22.65, 0.00, 'Young Adult', 19, 21),
(1030, 'The Dirty Little Secrets of Getting Your Dream Job', 33.34, 'In stock', 4, 33.34, 33.34, 0.00, 'Business', 19, 21),
(1031, 'The Coming Woman: A Novel Based on the Life of the Infamous Feminist, Victoria Woodhull', 17.93, 'In stock', 3, 17.93, 17.93, 0.00, 'Default', 19, 21),
(1032, 'The Boys in the Boat: Nine Americans and Their Epic Quest for Gold at the 1936 Berlin Olympics', 22.60, 'In stock', 4, 22.60, 22.60, 0.00, 'Default', 19, 21),
(1033, 'The Black Maria', 52.15, 'In stock', 1, 52.15, 52.15, 0.00, 'Poetry', 19, 21),
(1034, 'Starving Hearts (Triangular Trade Trilogy, #1)', 13.99, 'In stock', 2, 13.99, 13.99, 0.00, 'Default', 19, 21),
(1035, 'Shakespeare\'s Sonnets', 20.66, 'In stock', 4, 20.66, 20.66, 0.00, 'Poetry', 19, 21),
(1036, 'Set Me Free', 17.46, 'In stock', 5, 17.46, 17.46, 0.00, 'Young Adult', 19, 21),
(1037, 'Scott Pilgrim\'s Precious Little Life (Scott Pilgrim #1)', 52.29, 'In stock', 5, 52.29, 52.29, 0.00, 'Sequential Art', 19, 21),
(1038, 'Rip It Up and Start Again', 35.02, 'In stock', 5, 35.02, 35.02, 0.00, 'Music', 19, 21),
(1039, 'Our Band Could Be Your Life: Scenes from the American Indie Underground, 1981-1991', 57.25, 'In stock', 3, 57.25, 57.25, 0.00, 'Music', 19, 21),
(1040, 'Olio', 23.88, 'In stock', 1, 23.88, 23.88, 0.00, 'Poetry', 19, 21),
(1041, 'Mesaerion: The Best Science Fiction Stories 1800-1849', 37.59, 'In stock', 1, 37.59, 37.59, 0.00, 'Science Fiction', 19, 21),
(1042, 'Libertarianism for Beginners', 51.33, 'In stock', 2, 51.33, 51.33, 0.00, 'Politics', 19, 21),
(1043, 'It\'s Only the Himalayas', 45.17, 'In stock', 2, 45.17, 45.17, 0.00, 'Travel', 19, 21),
(1044, 'In Her Wake', 12.84, 'In stock', 1, 12.84, 12.84, 0.00, 'Thriller', 19, 21),
(1045, 'How Music Works', 37.32, 'In stock', 2, 37.32, 37.32, 0.00, 'Music', 19, 21),
(1046, 'Foolproof Preserving: A Guide to Small Batch Jams, Jellies, Pickles, Condiments, and More: A Foolproof Guide to Making Small Batch Jams, Jellies, Pickles, Condiments, and More', 30.52, 'In stock', 3, 30.52, 30.52, 0.00, 'Food and Drink', 19, 21),
(1047, 'Chase Me (Paris Nights #2)', 25.27, 'In stock', 5, 25.27, 25.27, 0.00, 'Romance', 19, 21),
(1048, 'Black Dust', 34.53, 'In stock', 5, 34.53, 34.53, 0.00, 'Romance', 19, 21),
(1049, 'Birdsong: A Story in Pictures', 54.64, 'In stock', 3, 54.64, 54.64, 0.00, 'Childrens', 19, 21),
(1050, 'America\'s Cradle of Quarterbacks: Western Pennsylvania\'s Football Factory from Johnny Unitas to Joe Montana', 22.50, 'In stock', 3, 22.50, 22.50, 0.00, 'Default', 19, 21),
(1051, 'Aladdin and His Wonderful Lamp', 53.13, 'In stock', 3, 53.13, 53.13, 0.00, 'Default', 19, 21);
```

Résultat :



4 – Analyse et Visualisation

Code :

analyse.py

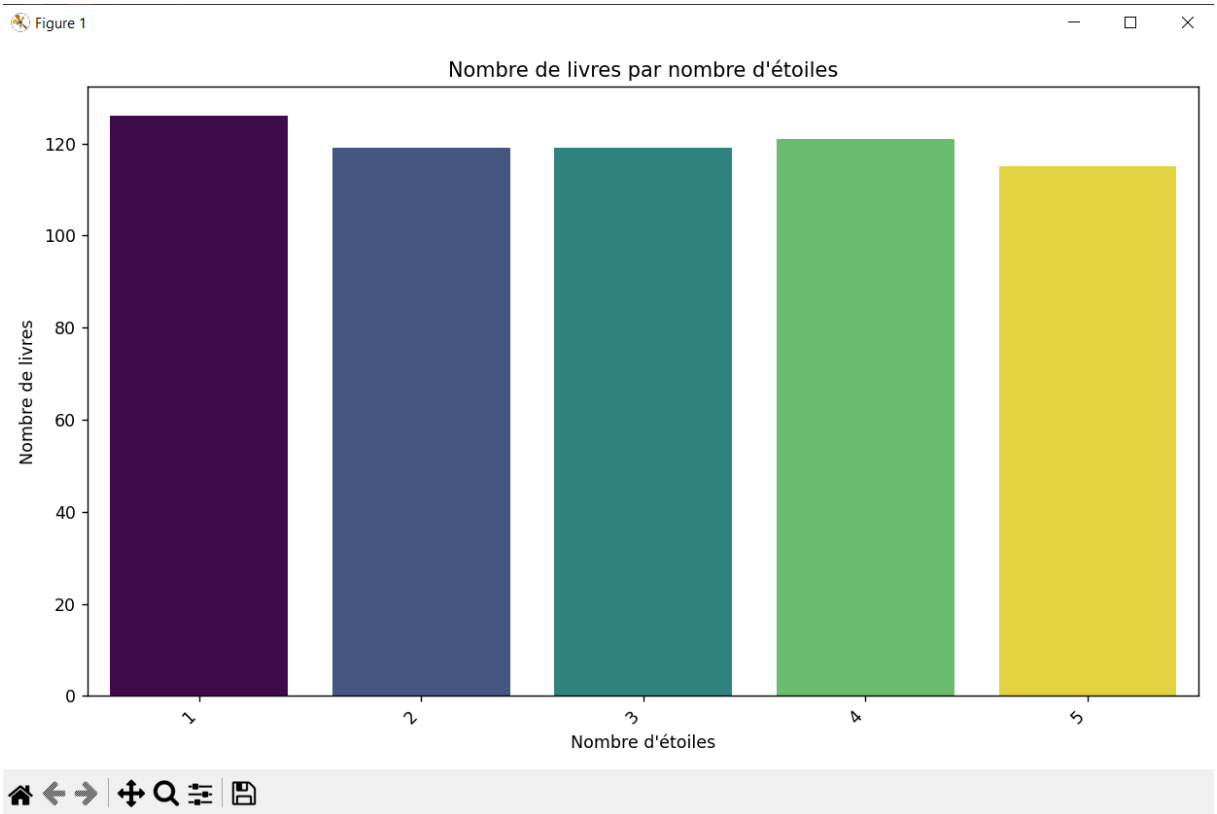
```python
import mysql.connector
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

# Connexion a la bdd mysql
try:
    conn = mysql.connector.connect(
        host="127.0.0.1",       # @serv
        user="root",
        password="",
        database="livre"        # Nom bdd
    )
    print("Connexion à la base de données réussie !")
except mysql.connector.Error as err:
    print(f"Erreur de connexion à la base de données : {err}")
    exit()

# Récup data table livre
cursor = conn.cursor()
query = "SELECT * FROM livres"
cursor.execute(query)

# Convertir les résultats en DataFrame pandas
columns = [col[0] for col in cursor.description]  # recuperation des noms des col
data = cursor.fetchall()  # recuperation des lignes
df = pd.DataFrame(data, columns=columns)

# Ajout d la colonne vente au DataFrame
df['vente'] = 40 - df['stock']  # Calcul de la colonne vente

# MAJ de la bdd MySQL avec la nouvelle col vente
try:
    # Ajout la colonne vente à la table livres si elle n'existe pas déjà
    cursor.execute("ALTER TABLE livres ADD COLUMN IF NOT EXISTS vente INT")

    # MAJ chaque ligne avec la valeur de vente
    for index, row in df.iterrows():
        update_query = f"UPDATE livres SET vente = {row['vente']} WHERE id = {row['id']}"
        cursor.execute(update_query)

    # Valider les modif dans la bdd
    conn.commit()
    print("Colonne 'vente' ajoutée et mise à jour dans la base de données !")
except mysql.connector.Error as err:
    print(f"Erreur lors de la mise à jour de la base de données : {err}")
finally:
    # Fermer le curseur et la connexion
    cursor.close()
    conn.close()

# Vérif et convertir les types de données
df['etoiles'] = df['etoiles'].astype(int)
df['stock'] = df['stock'].astype(int)
df['prix'] = df['prix'].astype(float)
df['vente'] = df['vente'].astype(int)

# Palette de couleurs commune
```

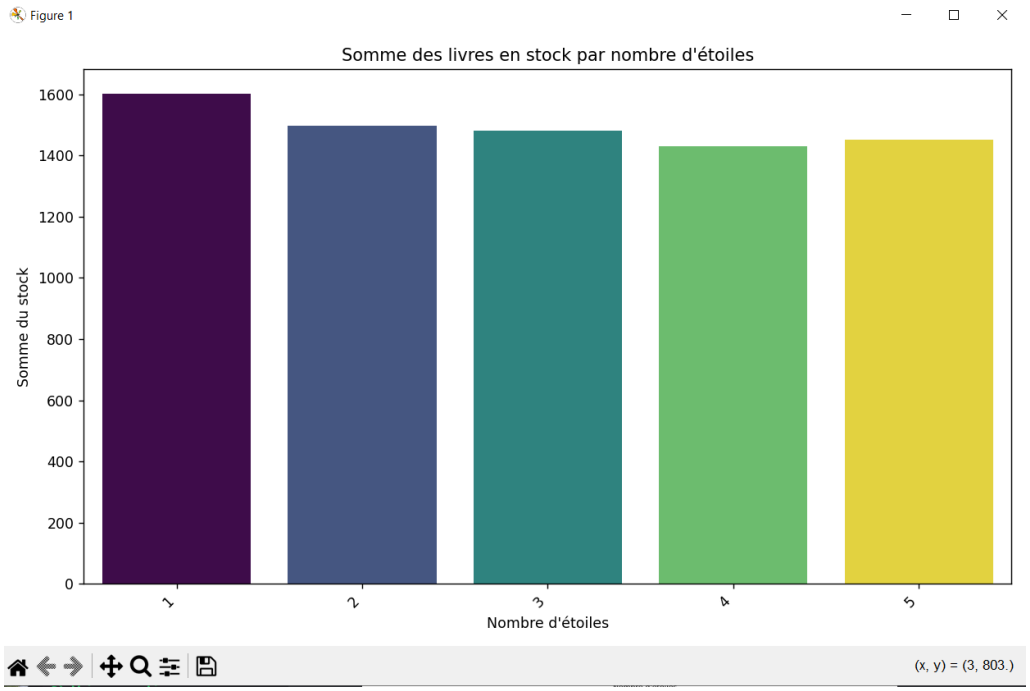
```python
plt.show()
plt.close()

# Figure 4 : Somme des livres en stock par catégorie (trié par ordre alphabétique)
stock_par_catégorie = df.groupby('categorie')['stock'].sum().reset_index()

stock_par_catégorie = stock_par_catégorie.sort_values(by='categorie')

plt.figure(figsize=(12, 6))
sns.barplot(x='categorie', y='stock', data=stock_par_catégorie, hue='categorie', palette=palette, legend=False)
plt.title('Somme des livres en stock par catégorie')
plt.xlabel('Catégorie')
plt.ylabel('Somme du stock')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
plt.close()

# Figure 4bis : Somme des ventes par catégorie (trié par ordre alphabétique)
ventes_par_catégorie = df.groupby('categorie')['vente'].sum().reset_index()

ventes_par_catégorie = ventes_par_catégorie.sort_values(by='categorie')

plt.figure(figsize=(12, 6))
sns.barplot(x='categorie', y='vente', data=ventes_par_catégorie, hue='categorie', palette=palette, legend=False)
plt.title('Somme des ventes par catégorie')
plt.xlabel('Catégorie')
plt.ylabel('Somme des ventes')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
plt.close()

# Figure 5 : Nombre de livres par intervalle de prix (graphique en barres)
# Creer des intervalle de prix en excluant les deux premier
bins = np.arange(10, df['prix'].max() + 5, 5)  # Commence à 10 au lieu de 0
df['Prix Intervalle'] = pd.cut(df['prix'], bins=bins)

# Compte le nombre de livres par intervalle de prix
prix_counts = df['Prix Intervalle'].value_counts().sort_index().reset_index()
prix_counts.columns = ['Prix Intervalle', 'count']

# Filtre pour exclure les intervalles vides (au cas où des fois que)
prix_counts = prix_counts.dropna()

plt.figure(figsize=(12, 6))
sns.barplot(x='Prix Intervalle', y='count', data=prix_counts, hue='Prix Intervalle', palette=palette, legend=False)
plt.title('Nombre de livres par intervalle de prix')
plt.xlabel('Intervalle de prix')
plt.ylabel('Nombre de livres')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
plt.close()
```

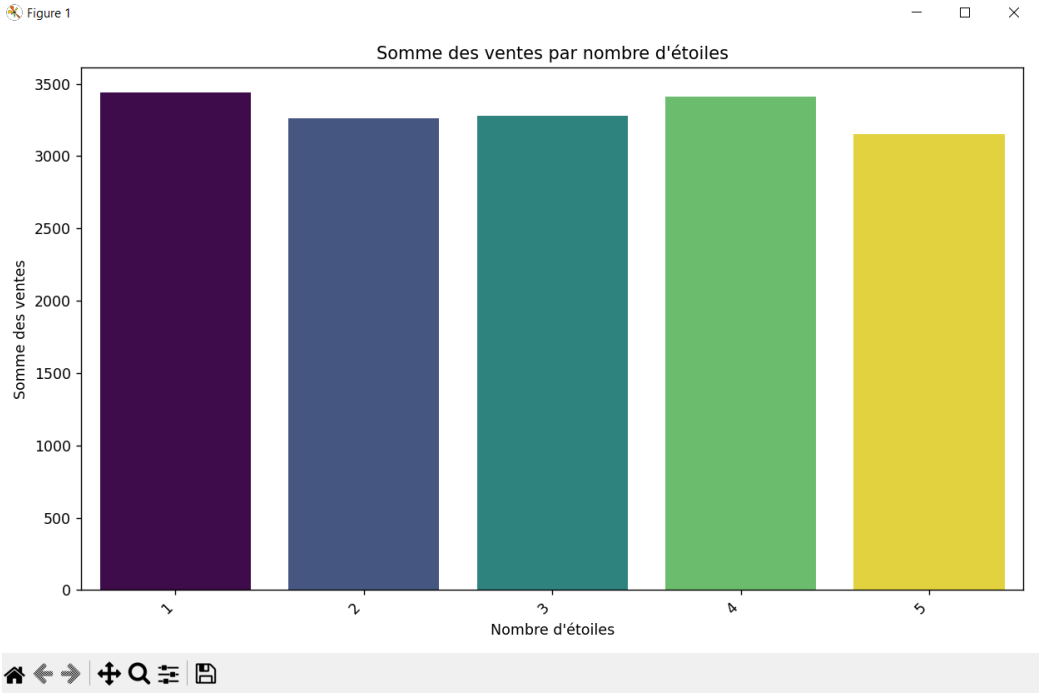
```python
# Palette de couleurs commune
palette = 'viridis'

# Figure 1 : Nombre de livres par nombre d'étoiles (graphique en barres)
etoiles_counts = df['etoiles'].value_counts().sort_index().reset_index()
etoiles_counts.columns = ['etoiles', 'count']

plt.figure(figsize=(10, 6))
sns.barplot(x='etoiles', y='count', data=etoiles_counts, hue='etoiles', palette=palette, legend=False)
plt.title('Nombre de livres par nombre d\'étoiles')
plt.xlabel('Nombre d\'étoiles')
plt.ylabel('Nombre de livres')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
plt.close()

# Figure 2 : Somme des livres en stock par nombre d'étoiles
stock_par_étoiles = df.groupby('etoiles')['stock'].sum().reset_index()

plt.figure(figsize=(10, 6))
sns.barplot(x='etoiles', y='stock', data=stock_par_étoiles, hue='etoiles', palette=palette, legend=False)
plt.title('Somme des livres en stock par nombre d\'étoiles')
plt.xlabel('Nombre d\'étoiles')
plt.ylabel('Somme du stock')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
plt.close()

# Figure 2bis : Somme des ventes par nombre d'étoiles
ventes_par_étoiles = df.groupby('etoiles')['vente'].sum().reset_index()

plt.figure(figsize=(10, 6))
sns.barplot(x='etoiles', y='vente', data=ventes_par_étoiles, hue='etoiles', palette=palette, legend=False)
plt.title('Somme des ventes par nombre d\'étoiles')
plt.xlabel('Nombre d\'étoiles')
plt.ylabel('Somme des ventes')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
plt.close()

# Figure 3 : Nombre de livres par catégorie (graphique en barres, trié par ordre alphabétique)
categorie_counts = df['categorie'].value_counts().reset_index()
categorie_counts.columns = ['categorie', 'count']

# Trier les catégories par ordre alphabétique
categorie_counts = categorie_counts.sort_values(by='categorie')

plt.figure(figsize=(12, 6))
sns.barplot(x='categorie', y='count', data=categorie_counts, hue='categorie', palette=palette, legend=False)
plt.title('Nombre de livres par catégorie ')
plt.xlabel('Catégorie')
plt.ylabel('Nombre de livres')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

```python

# Figure 6 : Somme des livres en stock par intervalle de prix
# Utiliser les mêmes intervalles que pour la Figure 5
stock_par_prix = df.groupby('Prix Intervalle', observed=True)['stock'].sum().reset_index()

# Filtrer pour exclure les intervalles vides (au cas où)
stock_par_prix = stock_par_prix.dropna()

plt.figure(figsize=(12, 6))
sns.barplot(x='Prix Intervalle', y='stock', data=stock_par_prix, hue='Prix Intervalle', palette=palette, legend=False, dodge=False)
plt.title('Somme des livres en stock par intervalle de prix')
plt.xlabel('Intervalle de prix')
plt.ylabel('Somme du stock')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
plt.close()

# Figure 6bis : Somme des ventes par intervalle de prix
ventes_par_prix = df.groupby('Prix Intervalle', observed=True)['vente'].sum().reset_index()

plt.figure(figsize=(12, 6))
sns.barplot(x='Prix Intervalle', y='vente', data=ventes_par_prix, hue='Prix Intervalle', palette=palette, legend=False, dodge=False)
plt.title('Somme des ventes par intervalle de prix')
plt.xlabel('Intervalle de prix')
plt.ylabel('Somme des ventes')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
plt.close()
```

Résultats : A savoir, on part du principe que chaque livre a été acheté par le site au fournisseur en 40 exemplaires !

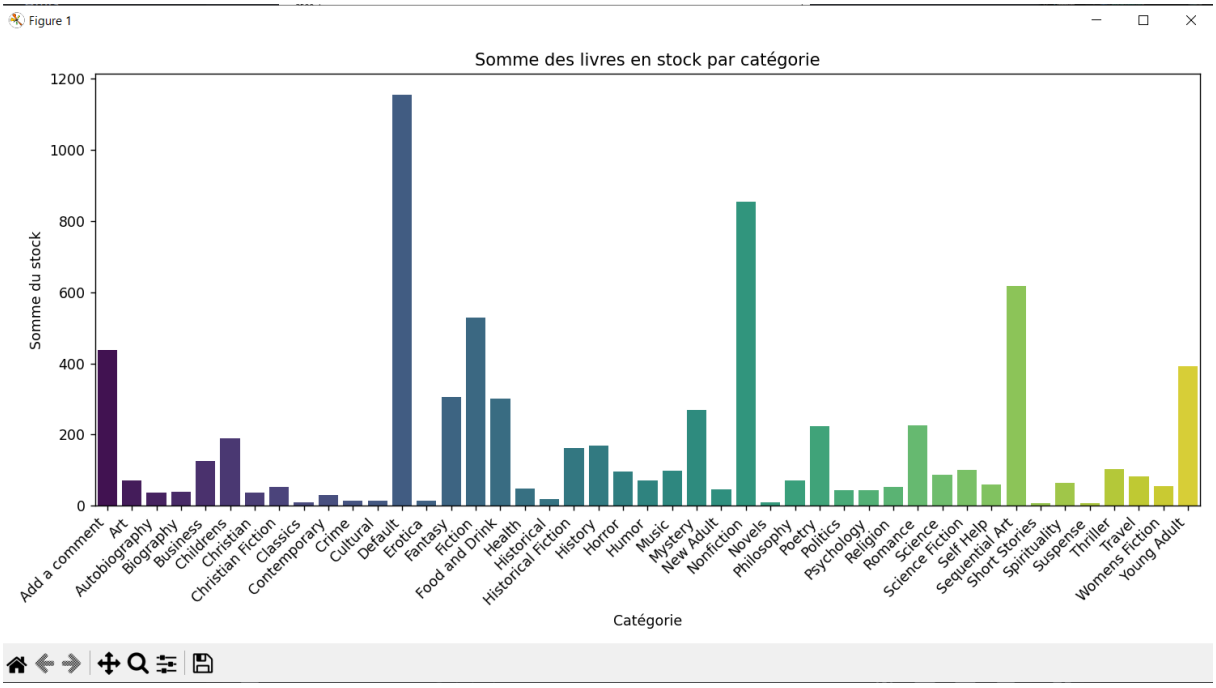


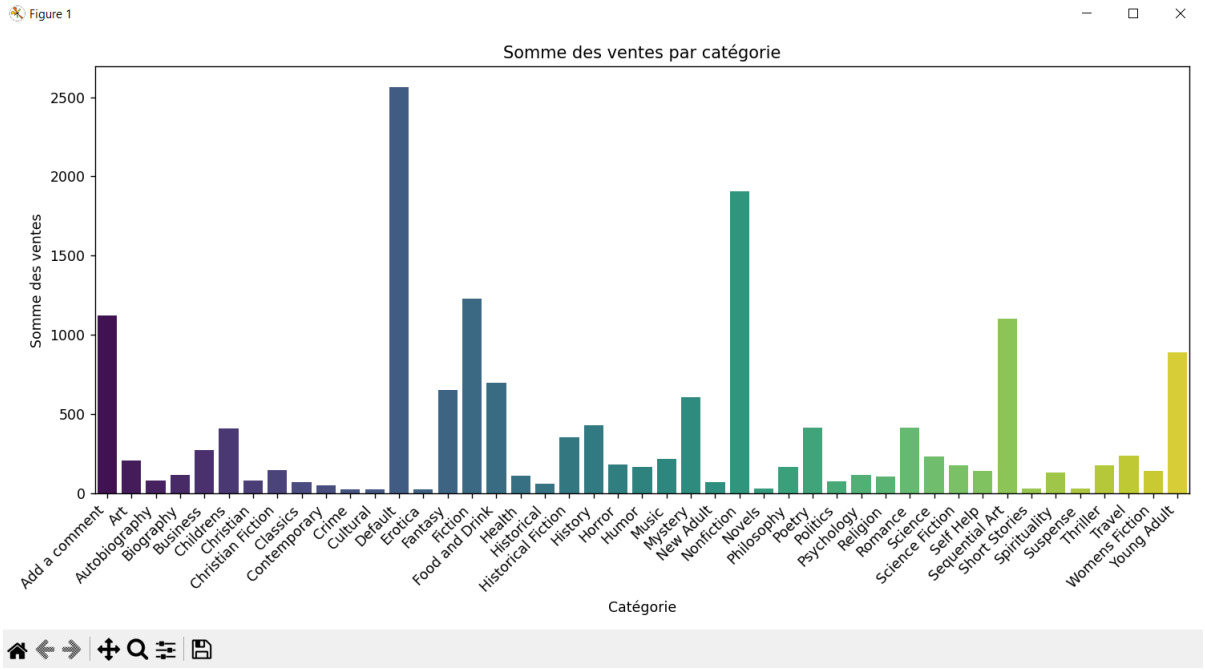


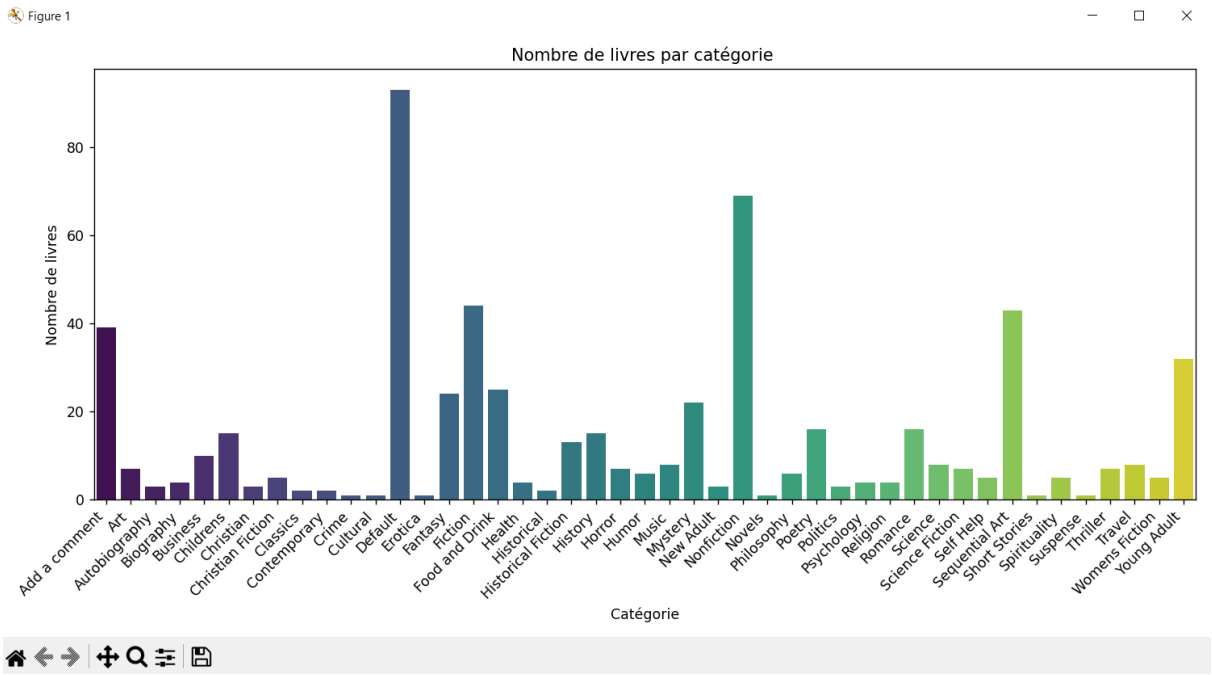


Ici, le premier graphique nous montre qu'il y a environ 125 livres (unique) qui ont 1 étoile et environ 115 livres (unique) qui ont 5 étoiles. On peut donc partir du principe qu'au total la somme de la commande passé de livre était de 125x40(=5000) (pour les livres 1 étoile et de 115x40(=4500) pour les livres 5 étoiles, (il y a donc là un ratio de 9/10ieme) et si on va sur le graphique numéro 3 nous pouvons simplement voir que les ordres de grandeur sont les mêmes que sur le graphique 1. On peut quand

même dire que les livres 4 étoiles se sont surement un peu mieux vendu que les autres d'après le graphique numéro 2 car on sait que les livres ont été acheté dans les mêmes quantité peu importe le nombre d'étoile, et l'ordre de grandeur du graphique numéro ne se réplique pas sur le deuxième graphique donc on sait que les 4 étoiles se sont un peu mieux vendu, et la raison pour laquelle cette interprétation ne saute pas aux yeux sur le graphique 3, c'est parce qu'on peu croire que l'ordre de grandeur est respecté mais si on regarde attentivement le premier graphique le nombre de livre unique est de 125 pour les 1 étoile et de 120 pour les 4 étoiles ce qui nous fait un ratio de 0,96 (120/125) alors qu'au niveau du graphique 2, on voit environ 1600 livres en stock pour les 1 étoile et 1400 livres en stock pour les 4 étoiles donc un ratio de 0,87 ! Il y a donc eu une légère surperformance au niveau de vente de ces livres 4 étoiles, mais rien de vraiment significatif. En ayant l'œil, on peut directement le voir sur le graphique 3 car les 4 étoiles ont légèrement surperformé les attentes mais ce n'est pas évident à avoir, il faut avoir le coup d'œil !

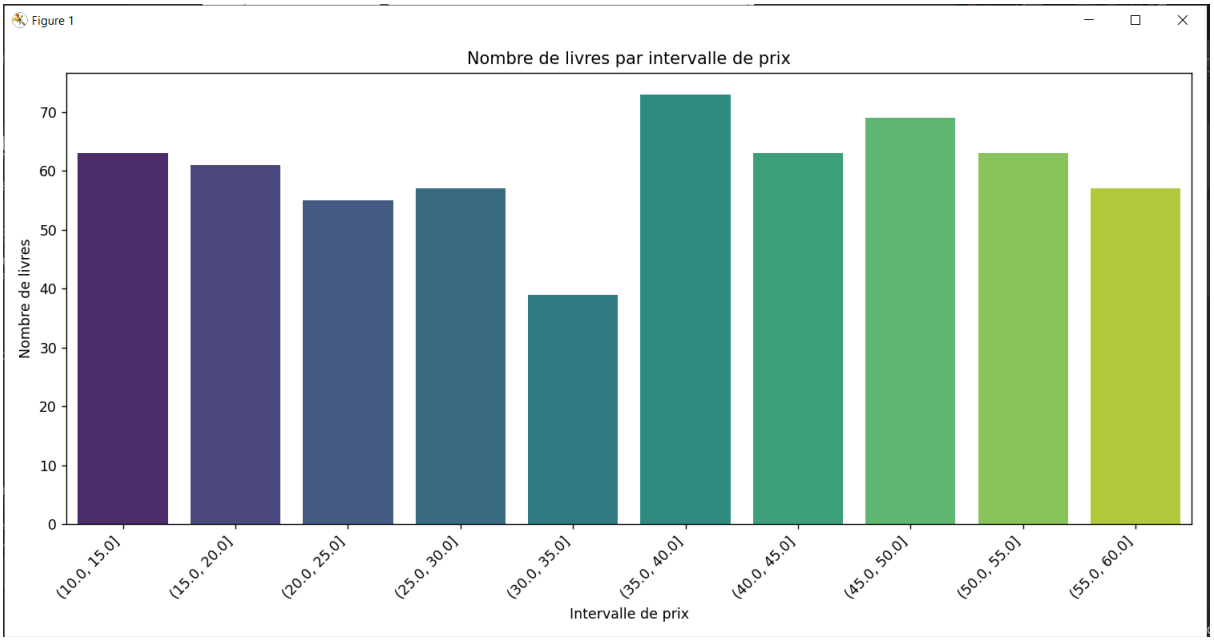


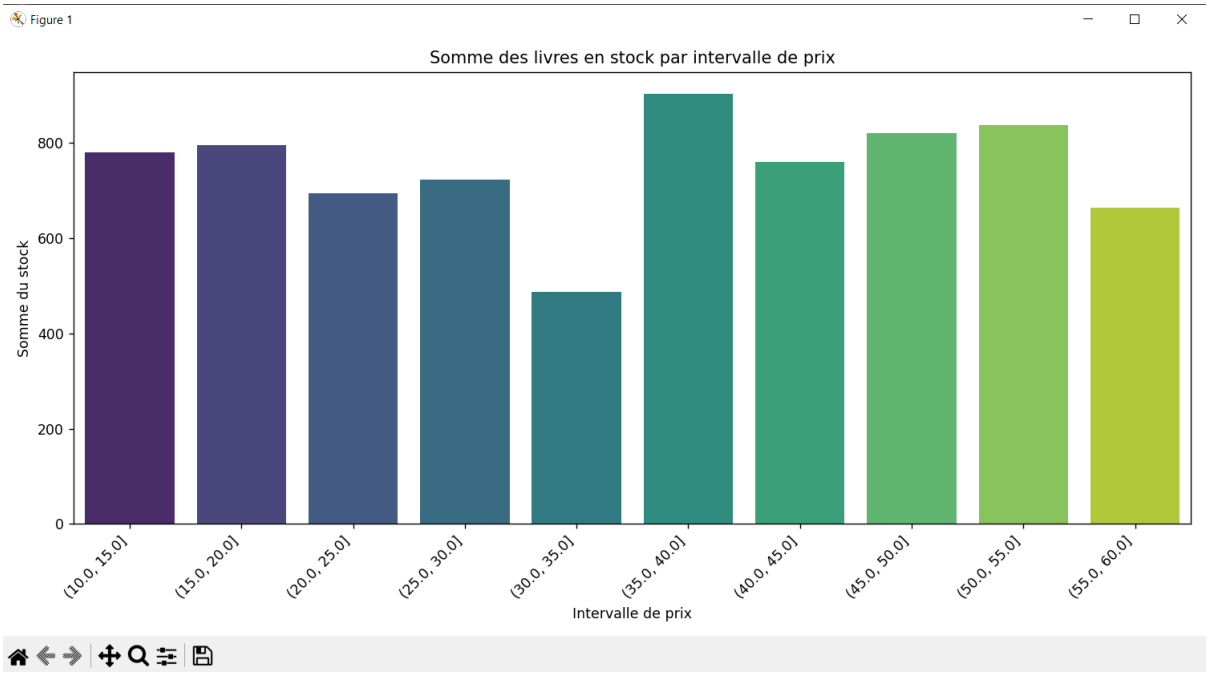


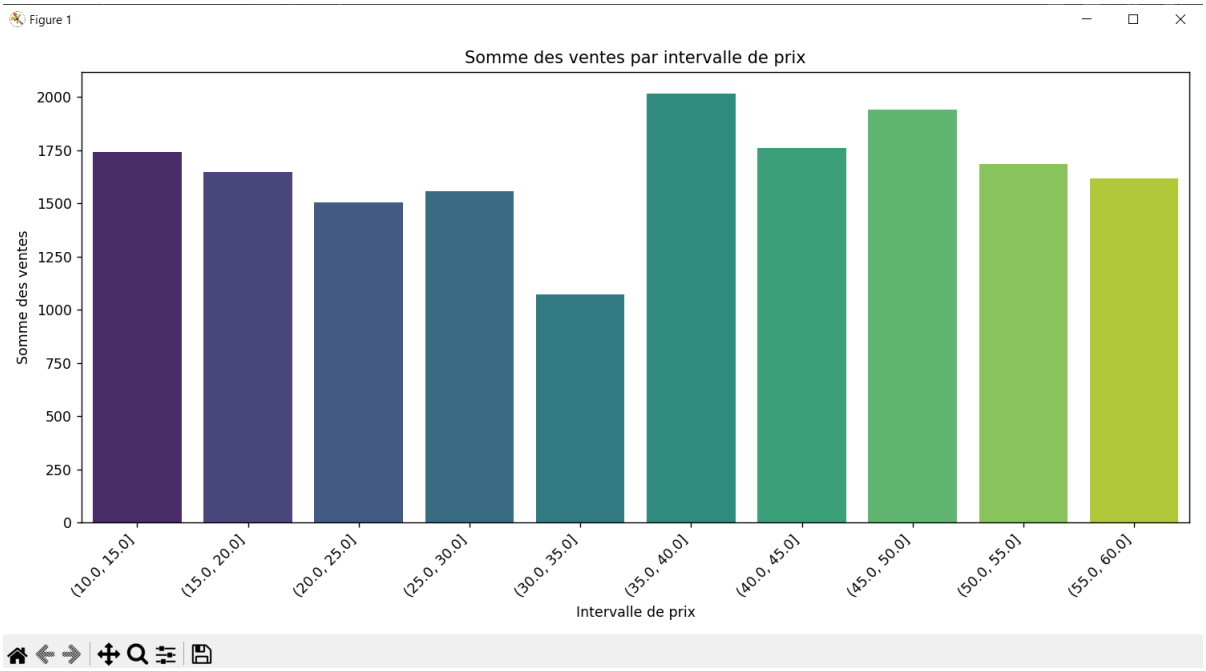


Au niveau du graphique 1 on peut voir que certaines catégories, comme "Default", "Young Adult" et "Philosophy", sont beaucoup plus représentées que d'autres, ce qui peut indiquer des tendances éditoriales. En ce qui concerne le graphique 2, les stocks sont plus élevés pour certaines catégories populaires, ce qui suggère que l'offre suit la demande du marché de la part de camarade de chez bookstoscrape ! Et d’après le graphique 3, on observe que les catégories ayant le plus de ventes

correspondent souvent aux plus représentées il y a donc corrélation entre les volumes d’achats, les ventes et donc très logiquement les stocks.







On peut commencer par analyser ce graphique 3 qui démontrer clairement que les livres avec un prix plus élevé se sont vendus dans de plus grande quantité en général (prix supérieur à 35) mais que cela correspond aussi au volume de livre unique et donc d'exemplaire au global de ces tranches de prix respective !

Sur cette séquence de graphique on peut voir 3 tranches intéressante de 40 £ à 55 £. Car sachant le nombre d'exemplaire unique de 40 £ à 45 £ et de 50 £ à 55 £ à 65 livres on sait qu'il y a eu 2600 livres en tout des ces catégories respectives achetés donc moins que dans la catégorie des livres qui ont un prix compris entre 50 £ et 55 £ qui eux sont au nombre de 70 exemplaires uniques donc 70x40=2800 exemplaires sans considération du livre. En l'occurrence on peut voir sur le graphique 2 que les stocks des livres de 50 £ à 55 £ sont plus élevé que celle des 40 £ à 45 £ et cela correspond parfaitement avec le graphique 3 qui montre bien que les ventes sont moins élevées pour une quantité initiale de livre pourtant supérieur.

IMPORTANT !

Les analyses faites ici ne disent clairement rien de concret car les données entrer sur le site sont purement fictives, elles ne sont représentatives d'aucun comportement humain par conséquent on peut voir que peu importe la manière dont on souhaite analyser ces données, elles ont été entrées de manière très lisse dans ce site fictif par conséquent il est difficile de dégager de quelconque tendance !

Conclusion

Ce projet nous a permis de mettre en place un processus complet de scraping, nettoyage, stockage et analyse de données. En partant de l'extraction automatisée des livres sur **Books to Scrape**, nous avons structuré et stocké ces informations dans une base de données **MySQL**, garantissant ainsi une gestion efficace et une exploitation optimale des données.

Grâce au nettoyage des valeurs extraites et à la conversion des types, nous avons pu obtenir une base de données fiable, évitant ainsi les erreurs lors de l'analyse. Enfin, les visualisations graphiques générées ont permis d'identifier des tendances intéressantes concernant les prix ou la notation moyenne des livres.

Ce projet illustre bien l'importance d'une approche méthodique et structurée dans la gestion des données.

En plus des aspects techniques, ce travail nous a apporté une meilleure compréhension des défis liés à la collecte et au traitement des données en conditions réelles, ainsi qu'une méthodologie efficace pour exploiter ces informations de manière pertinente.

Enfin, ce projet nous aura permis de constituer une base solide pour nos futures missions en entreprise, où la gestion et l'analyse des données joueront un rôle clé.