Aperçu du projet

J'utiliserai l'analyse exploratoire des données pour générer des informations pour une partie prenante de l'entreprise.

Problème commercial

Une entreprise voit désormais toutes les grandes entreprises créer du contenu vidéo original et souhaite participer à ce projet. Elle a décidé de créer un nouveau studio de cinéma, mais n'y connaît rien en création cinématographique. Je suis chargé d'explorer les types de films qui cartonnent actuellement au box-office. Je dois ensuite traduire ces résultats en informations exploitables que le directeur du nouveau studio pourra utiliser pour l'aider à choisir le type de films à produire.

In [79]:
```
import sqlite3
import pandas as pd
import matplotlib.pyplot as plt
```

In [80]:
```
#Connexion avec la base de données
dfilm_path = 'data/im.db'
conn = sqlite3.connect(dfilm_path)
cursor = conn.cursor()
```

In [81]:
```
#Affichage des tables de la base de données
table_name_query = """SELECT name
                      AS 'Table Names'
                      FROM sqlite_master
                      WHERE type='table';"""

df = pd.read_sql(table_name_query, conn)
df
```

Out[81]:

	Table Names
0	movie_basics
1	directors
2	known_for
3	movie_akas
4	movie_ratings
5	persons
6	principals
7	writers

In [82]:
```
#affichage de la table movie_basics
first_query = """SELECT *
                      FROM movie_basics;"""

movie_basics = pd.read_sql(first_query, conn)
movie_basics.head()
```

Out[82]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	genres
0	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama
1	tt0066787	One Day Before the Rainy Season	Ashad Ka Ek Din	2019	114.0	Biography,Drama
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama
3	tt0069204	Sabse Bada Sukh	Sabse Bada Sukh	2018	NaN	Comedy,Drama
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy

In [83]:
```
#affichage de la table movie_ratings
second_query = """SELECT *
                      FROM movie_ratings;"""

movie_ratings = pd.read_sql(second_query, conn)
movie_ratings.head()
```

Out[83]:

	movie_id	averagerating	numvotes
0	tt10356526	8.3	31
1	tt10384606	8.9	559
2	tt1042974	6.4	20
3	tt1043726	4.2	50352
4	tt1060240	6.5	21

Preparation des donnees

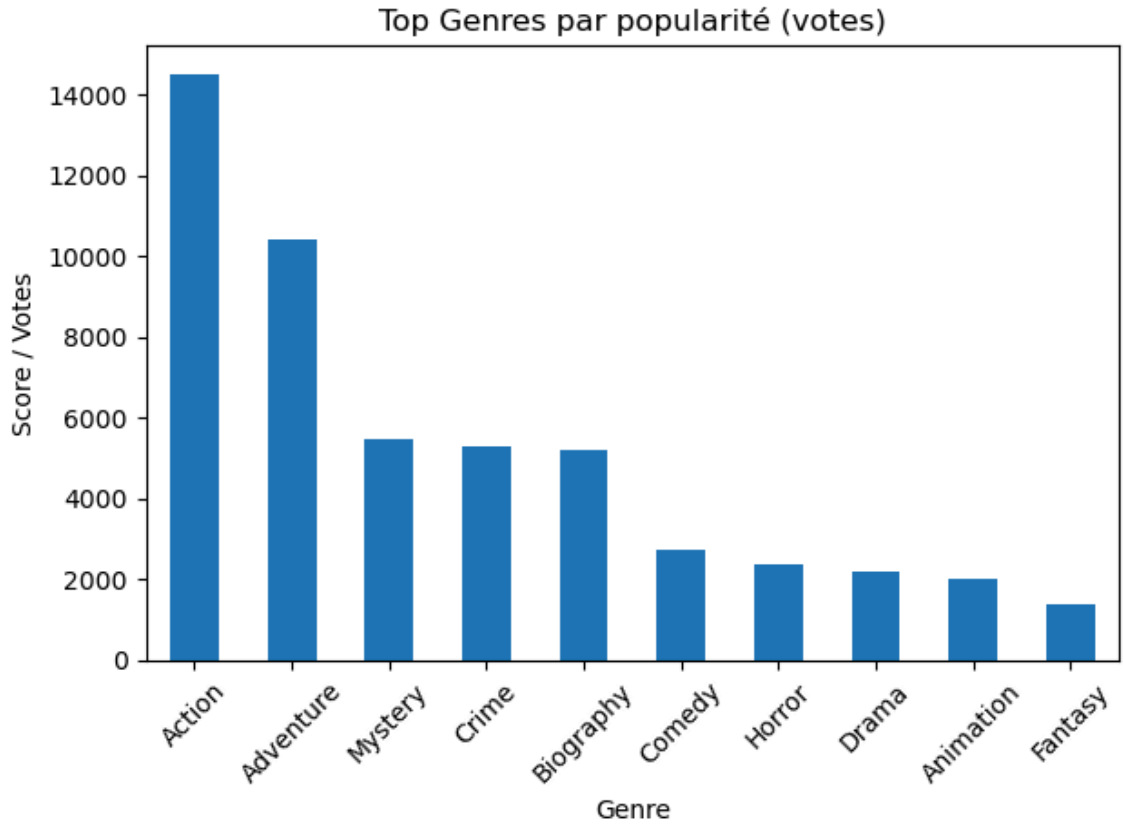
fusionnez les tables

In [84]:
```
#Fusion des tables
df = pd.merge(movie_basics, movie_ratings, on='movie_id')
```

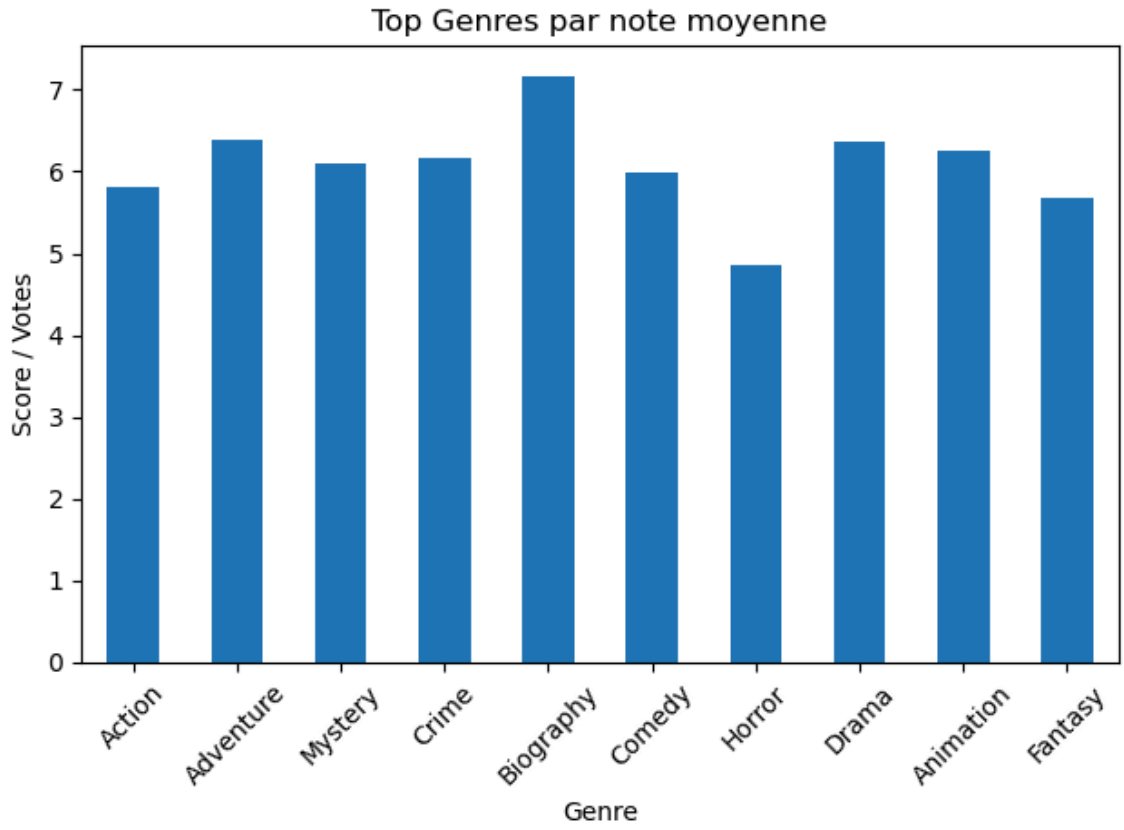
Genres

Le genre qui rapporte le plus au box-office

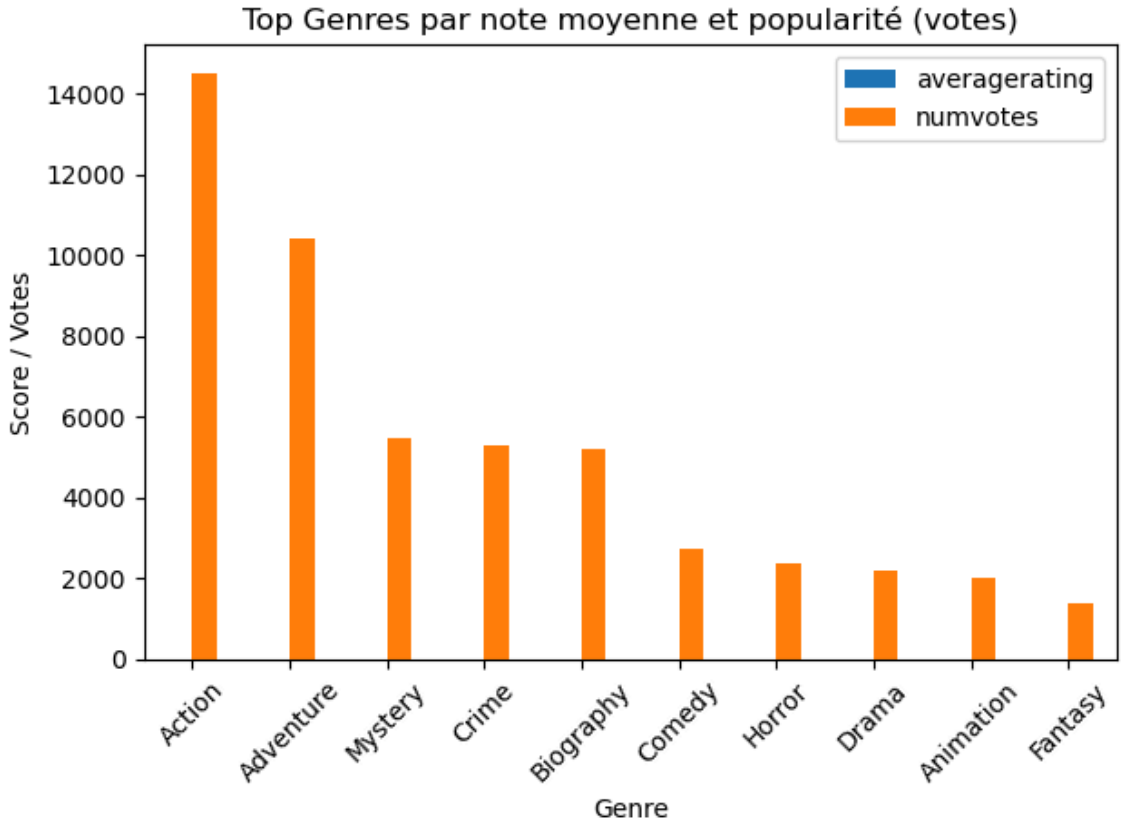
In [85]:
```
#Supprimer les lignes sans genres ou notes
df = df.dropna(subset=['genres', 'averagerating', 'numvotes'])

#Extraire le genre principal
df['main_genre'] = df['genres'].apply(lambda x: x.split(',')[0])

genre_stats = df.groupby('main_genre').agg({
    'averagerating': 'mean',
    'numvotes': 'mean',
    'movie_id': 'count'
}).rename(columns={'movie_id': 'num_movies'}).sort_values(by='numvotes', ascending=False)

#Visualisation
genre_stats = genre_stats.head(10)  # Top 10 genres
genre_stats['numvotes'].plot(kind='bar')
plt.title("Top Genres par popularité (votes)")
plt.xlabel("Genre")
plt.ylabel("Score / Votes")
plt.xticks(rotation=45)
plt.tight_layout()
plt.savefig("./images/Top_genres1.png", dpi=150)
plt.show()
```



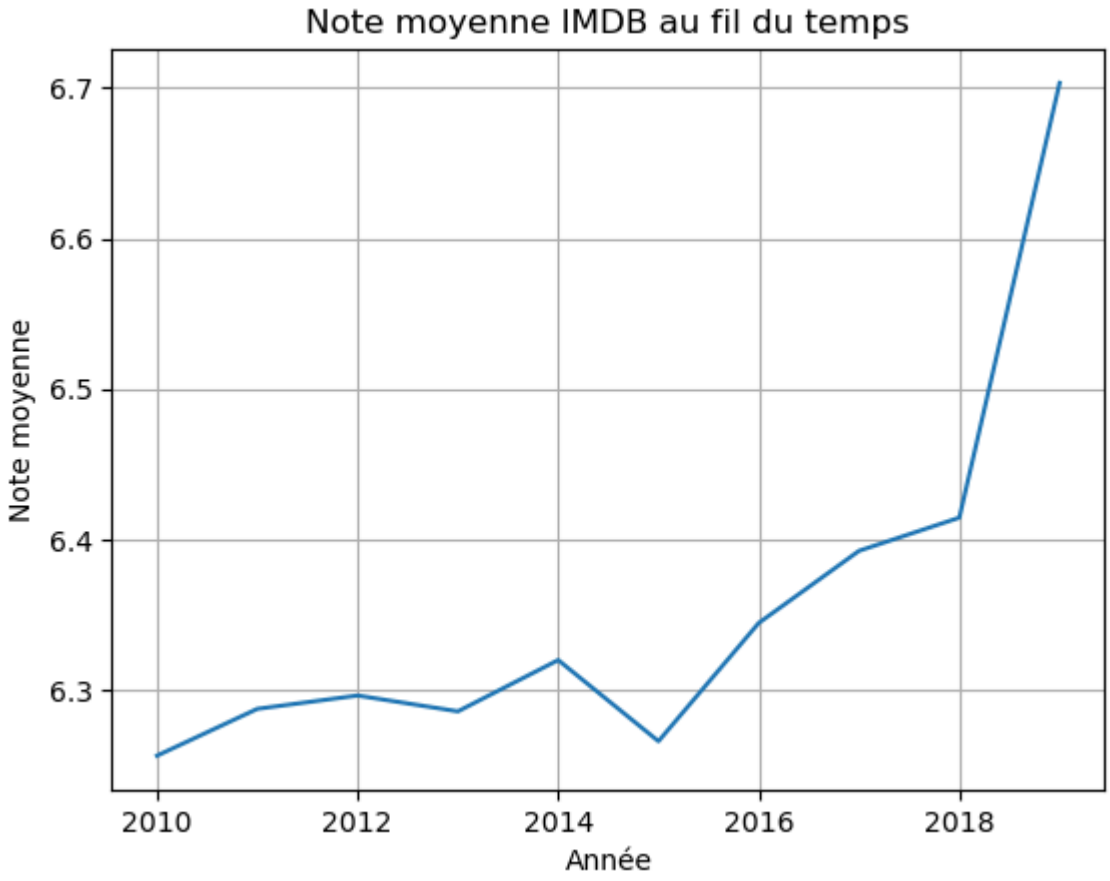
In [86]:
```
#Visualisation
genre_stats = genre_stats.head(10)  # Top 10 genres
genre_stats['averagerating'].plot(kind='bar')
plt.title("Top Genres par note moyenne")
plt.xlabel("Genre")
plt.ylabel("Score / Votes")
plt.xticks(rotation=45)
plt.tight_layout()
plt.savefig("./images/Top_genres2.png", dpi=150)
plt.show()
```



In [87]:
```
#Visualisation
genre_stats = genre_stats.head(10)  # Top 10 genres
genre_stats[['averagerating','numvotes']].plot(kind='bar')
plt.title("Top Genres par note moyenne et popularité (votes)")
plt.xlabel("Genre")
plt.ylabel("Score / Votes")
plt.xticks(rotation=45)
plt.tight_layout()
plt.savefig("./images/Top_genres.png", dpi=150)
plt.show()
```



In [88]:
```
#Garder les films avec une année de sortie connue
df['start_year'] = pd.to_numeric(df['start_year'], errors='coerce')
df_by_year = df.dropna(subset=['start_year'])

#Moyenne des notes par année
yearly = df_by_year.groupby('start_year')['averagerating'].mean()

yearly.plot()
plt.title("Note moyenne IMDB au fil du temps")
plt.xlabel("Année")
plt.ylabel("Note moyenne")
plt.grid(True)
plt.show()
```



In [ ]: