INTRODUCTION

Dans ce notebook, nous allons analyser les données des dessins animées sorties sur certaines période

L'objectif sera d'identifier les facteurs qui ont générer les plus de revenus par prodution.

Tout au long de ce rapport nous nous poserons des questions pertinentes et essayerons d'y repondre avec du code et de la visualisation de données.

Le plan de notre analyse sont les suivants :

- Import des différentes librairies de Python
- Decouvertes et bref aperçu de nos données
- Nettoyage et Transformation de nos données
- Analyse exploratoire de données
- Visualisation

In [274…
```
# Importer les librairies de Python
import pandas as pd
import numpy as  np
import os
import seaborn as sns
import matplotlib.pyplot as plt

sns.set_style("darkgrid", {"axes.facecolor": "#121212","grid.color": "#FFD700",
sns.set_palette("husl", n_colors=8)
```

In [278…
```
# Decouvertes de données
dossier_actuel = os.getcwd()  # Montage de disque local
chemin_fichier = os.path.join(dossier_actuel, 'movie_animation.csv') # joindre l
df = pd.read_csv(chemin_fichier) # lire le fichier CSV
```

In [279…
```
df.head()
```

Out[279]:

	id	title	vote_average	vote_count	status	release_date	revenue	runtin
0	150540	Inside Out	7.922	19463	Released	2015-06-09	857611174	
1	14160	Up	7.949	18857	Released	2009-05-28	735099082	
2	12	Finding Nemo	7.824	18061	Released	2003-05-30	940335536	1(
3	354912	Coco	8.222	17742	Released	2017-10-27	800526015	1(
4	10681	WALL·E	8.078	17446	Released	2008-06-22	521311860	

5 rows × 23 columns

In [241…
```
# Obtenir la dimension du DataFrame
df.shape
```

Out[241]: (51945, 23)

In [242…
```
# Description du DataFrame et obtenir les nfo statistiques
df.describe()
```

Out[242]:

	id	vote_average	vote_count	revenue	runtime	bu
count	5.194500e+04	51945.000000	51945.000000	5.194500e+04	51945.000000	5.194500
mean	6.188122e+05	2.597083	40.395380	1.863464e+06	20.901415	5.759513
std	3.479748e+05	3.318748	482.705948	3.049824e+07	39.016002	7.606628
min	1.200000e+01	0.000000	0.000000	0.000000e+00	0.000000	0.000000
25%	3.300810e+05	0.000000	0.000000	0.000000e+00	3.000000	0.000000
50%	6.119150e+05	0.000000	0.000000	0.000000e+00	7.000000	0.000000
75%	9.159040e+05	6.000000	2.000000	0.000000e+00	19.000000	0.000000
max	1.238314e+06	10.000000	19463.000000	1.450027e+09	3720.000000	2.600000

In [243…

```
df.describe(include='object')
```

Out[243]:

	title	status	release_date	backdrop_path	
count	51944	51945	49808	15835	
unique	49381	6	16893	15731	
top	Little Red Riding Hood	Released	2017-01-01	/7tFOLDhZqKPactVlGfdjgVmYnlu.jpg	https://animatic
freq	15	51134	291	17	

In [244…

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51945 entries, 0 to 51944
Data columns (total 23 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   id                    51945 non-null  int64
 1   title                 51944 non-null  object
 2   vote_average          51945 non-null  float64
 3   vote_count            51945 non-null  int64
 4   status                51945 non-null  object
 5   release_date          49808 non-null  object
 6   revenue               51945 non-null  int64
 7   runtime               51945 non-null  int64
 8   adult                 51945 non-null  bool
 9   backdrop_path         15835 non-null  object
 10  budget                51945 non-null  int64
 11  homepage              8253 non-null   object
 12  imdb_id               29552 non-null  object
 13  original_language     51945 non-null  object
 14  original_title        51944 non-null  object
 15  overview              45866 non-null  object
 16  popularity            51945 non-null  float64
 17  poster_path           37934 non-null  object
 18  tagline               4678 non-null   object
 19  genres                51945 non-null  object
 20  production_companies  29398 non-null  object
 21  production_countries  39700 non-null  object
 22  spoken_languages      33818 non-null  object
dtypes: bool(1), float64(2), int64(5), object(15)
memory usage: 8.8+ MB
```

Nettoyage et tranformation de données

In [280…

```
# Suppression des colonnes indésirables
df.drop(['backdrop_path', 'poster_path', 'id', 'overview', 'homepage', 'tagline'
```

In [281…

```
df.head()
```

Out[281]:

	title	vote_average	vote_count	status	release_date	revenue	runtime	adult
0	Inside Out	7.922	19463	Released	2015-06-09	857611174	95	False
1	Up	7.949	18857	Released	2009-05-28	735099082	96	False
2	Finding Nemo	7.824	18061	Released	2003-05-30	940335536	100	False
3	Coco	8.222	17742	Released	2017-10-27	800526015	105	False
4	WALL·E	8.078	17446	Released	2008-06-22	521311860	98	False

In [282…

```
#  Recherche de doublons
df.loc[df.duplicated()]
```

Out[282]:

	title	vote_average	vote_count	status	release_date	revenue	runtime	a
26479	Entre Elle & Lui - 11 A la fin du 1er trimestre	0.0	0	Released	NaN	0	0	
29741	Haïku	0.0	0	Released	2001-01-01	0	1	
45068	The Head Saves The Earth	0.0	0	Released	1995-09-12	0	110	

In [283…

```
# Supprimer les doublons
df.drop([26479, 29741, 45068], inplace=True)
```

In [284…

```
df.loc[df.duplicated()]
```

Out[284]:

title	vote_average	vote_count	status	release_date	revenue	runtime	adult	budge

In [285…
```
# Changer le format des dates en yyyy-mm-dd
df['release_date'] = pd.to_datetime(df['release_date'], format='%Y-%m-%d')
```

Spilter le Genre pour les infos

In [288…
```
genre = {}
for row in df["genres"].str.split(","):
    for word in row:
        word = word.strip()
        if word in genre:
            genre[word] += 1
        else:
            genre[word] = 1
genres_df = pd.DataFrame.from_dict([genre]).T.sort_values(by=0, ascending=False)
```

In [289…
```
# Suprimmer Animation dans le colonnes genre
genres_df.drop('Animation', inplace=True)
```

In [290…
```
genres_df
```

Out[290]:

	0
Comedy	7878
Family	7480
Fantasy	3896
Adventure	3539
Drama	2849
Science Fiction	2539
Action	2428
Documentary	1897
Music	1697

In [291…
```
# Afficher les nombres des languages originals
df1['original_language'].value_counts()
```

Out[291]:
```
original_language
en    27549
ja     5230
ru     2787
fr     2714
es     1387
      ...
cr        1
af        1
te        1
zu        1
ne        1
Name: count, Length: 102, dtype: int64
```

In [292…
```
# Afficher les statues
df1['status'].value_counts()
```

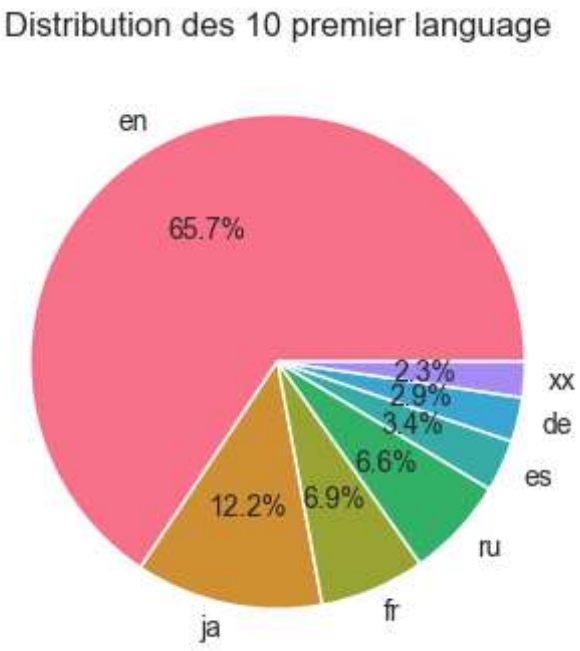
Out[292]:
```
status
Released           49634
In Production        113
Post Production       49
Planned                9
Canceled               1
Rumored                1
Name: count, dtype: int64
```

Visualisation de données

In [293…
```
# Versions originals
origin_group_lang = df.groupby('original_language').size().sort_values(ascending
origin_group_lang
```

Out[293]:
```
original_language
en    28582
ja     5330
fr     2997
ru     2871
es     1472
de     1266
xx     1011
dtype: int64
```

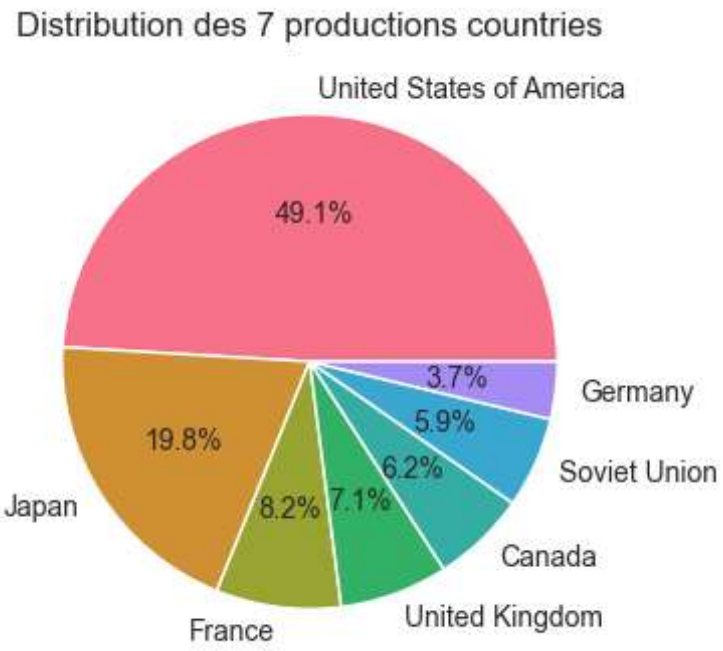
In [294…
```
plt.figure(figsize=(4,4))
plt.title("Distribution des 10 premier language")
plt.pie(origin_group_lang, labels=lang_origin_group.index, autopct='%1.1f%%')
plt.show()
```



In [295…
```
# Productions par Pays
prod_countr_group = df.groupby('production_countries').size().sort_values(ascend
prod_countr_group
```

Out[295]:
```
production_countries
United States of America    12836
Japan                        5182
France                       2136
United Kingdom               1854
Canada                       1616
Soviet Union                 1556
Germany                       980
dtype: int64
```

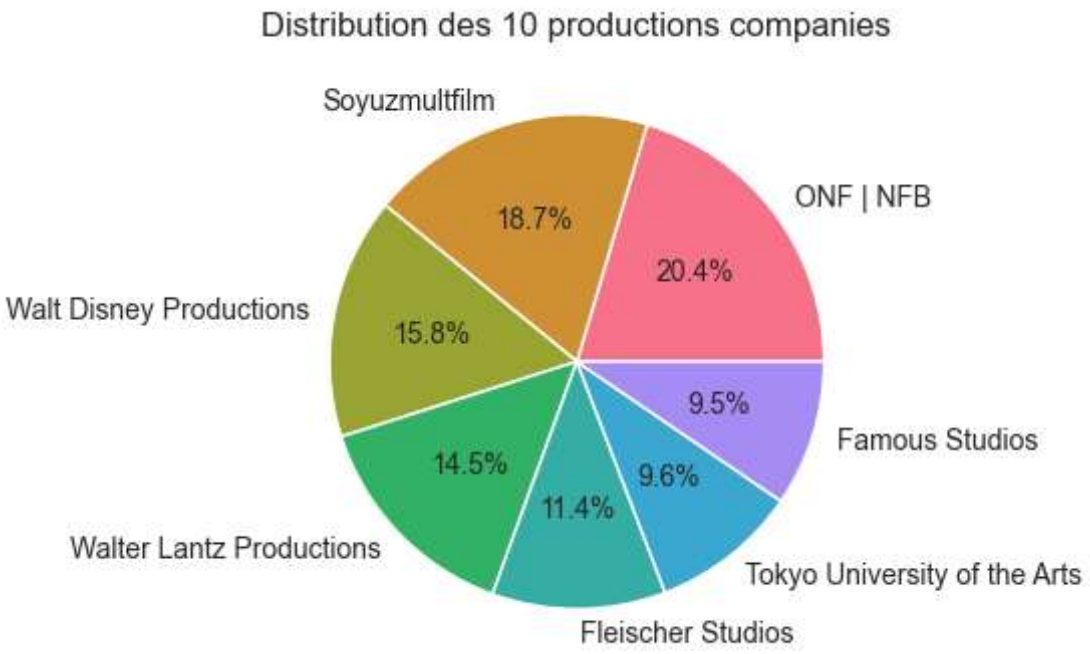
In [296…
```
plt.figure(figsize=(4,4))
plt.title("Distribution des 7 productions countries")
plt.pie(prod_countr_group, labels=prod_countr_group.index, autopct='%1.1f%%')
plt.show()
```



In [297…
```
# Meuilleurs Producteurs
prod_comp_group = df.groupby('production_companies').size().sort_values(ascendin
prod_comp_group
```

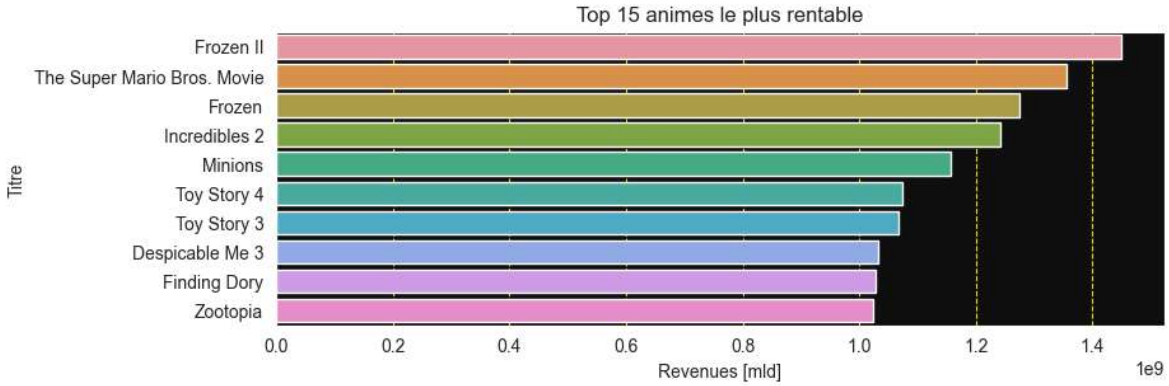
Out[297]:
```
production_companies
ONF | NFB                       717
Soyuzmultfilm                   657
Walt Disney Productions         555
Walter Lantz Productions        511
Fleischer Studios               399
Tokyo University of the Arts    339
Famous Studios                  335
dtype: int64
```

In [298…
```
plt.figure(figsize=(4,4))
plt.title("Distribution des 10 productions companies")
plt.pie(prod_comp_group, labels=prod_comp_group.index, autopct='%1.1f%%')
plt.show()
```



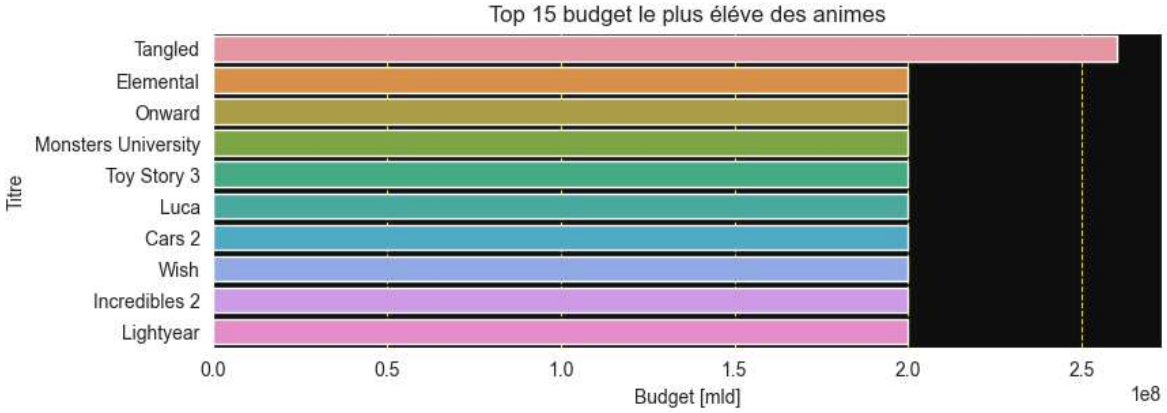
In [299…
```
# Anime le plus rentable
grossing_movies = df[['title', 'revenue']].sort_values(by='revenue', ascending=F
```

In [300…
```
plt.figure(figsize=(9, 3))
plt.title("Top 15 animes le plus rentable")
sns.barplot(data=grossing_movies, y='title', x='revenue', orient='h')
plt.xlabel('Revenues [mld]')
plt.ylabel("Titre")
plt.show()
```



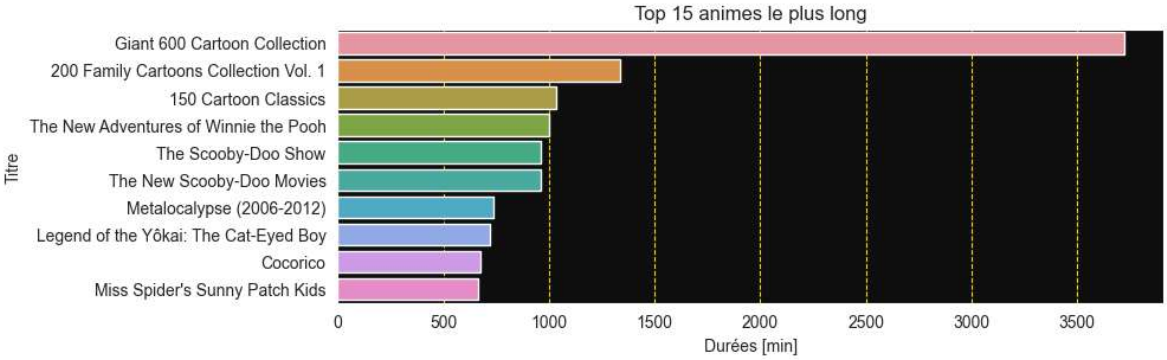
In [301…
```
# Budget plus élevé
budget_movies = df[['title', 'budget']].sort_values(by='budget', ascending=False
```

In [302…
```
plt.figure(figsize=(9, 3))
plt.title("Top 15 budget le plus éléve des animes")
sns.barplot(data=budget_movies, y='title', x='budget', orient='h')
plt.xlabel('Budget [mld]')
plt.ylabel("Titre")
plt.show()
```



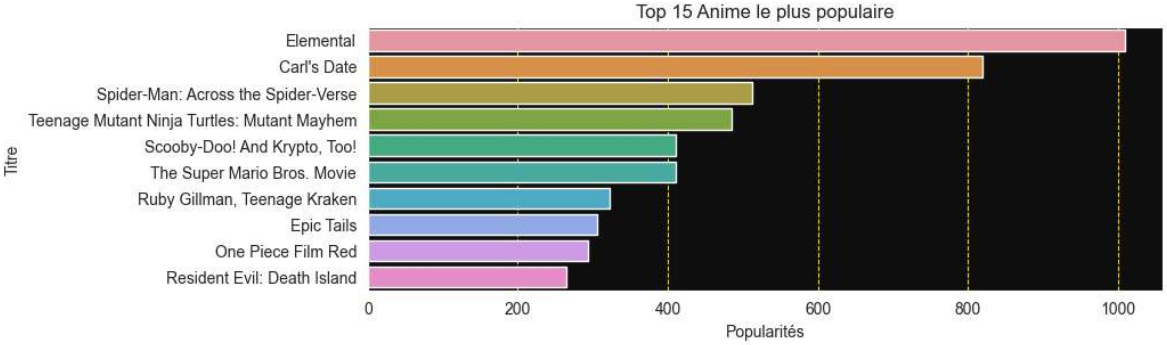
In [303…
```
# Plus long Anime en min
movies_long = df[['title', 'runtime']].sort_values(by='runtime', ascending=False
```

In [304…
```
plt.figure(figsize=(9, 3))
plt.title("Top 15 animes le plus long")
sns.barplot(data=movies_long, y='title', x='runtime', orient='h')
plt.xlabel('Durées [min]')
plt.ylabel("Titre")
plt.show()
```



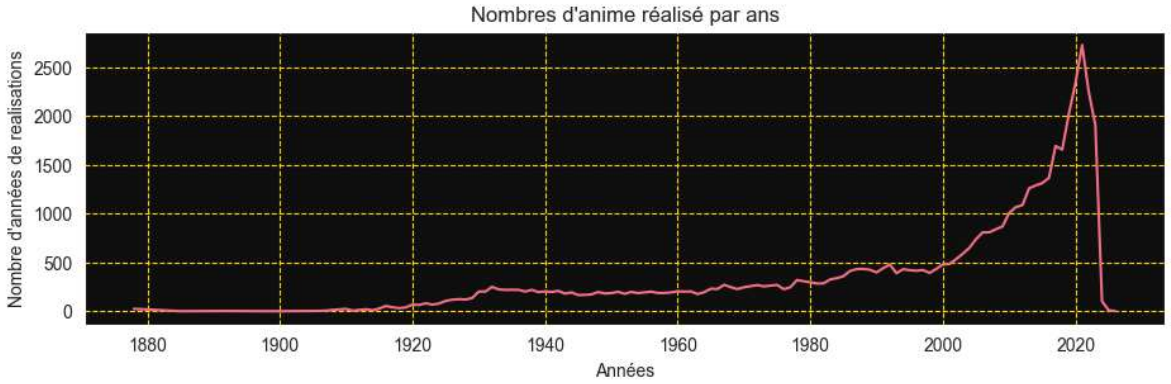
In [310…
```
# Les plus populaire
popular_movies = df[['title', 'popularity']].sort_values(by='popularity', ascend
```

In [306…
```
plt.figure(figsize=(9, 3))
plt.title("Top 15 Anime le plus populaire")
sns.barplot(data=popular_movies, y='title', x='popularity', orient='h')
plt.xlabel('Popularités')
plt.ylabel("Titre")
plt.show()
```

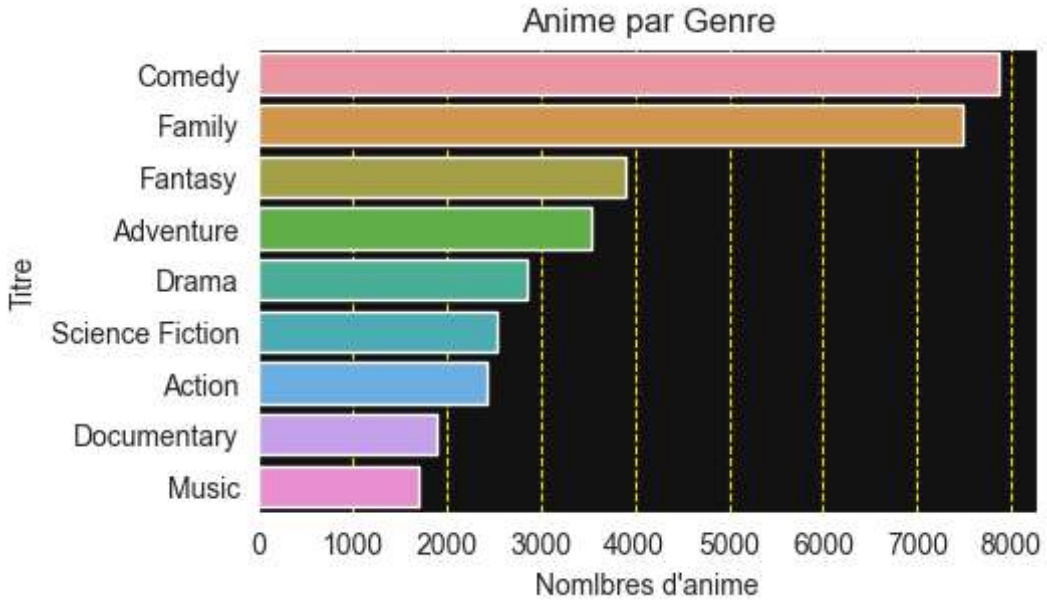


In [307…
```
# Nombres de film réalisé par années
release_year = df.groupby(df.release_date.dt.year)['title'].count()
```

In [308…
```
plt.figure(figsize=(11,3))
release_year_group.plot()
plt.title("Nombres d'anime réalisé par ans")
plt.ylabel("Nombre d'années de realisations")
plt.xlabel("Années")
plt.show()
```



In [309…
```
# Nombres d'animé par genre
plt.figure(figsize=(5,3))
plt.title("Anime par Genre")
sns.barplot(data=genres_df, y=genres_df.index, x=0, orient='h')
plt.xlabel('Nomlbres d\'anime')
plt.ylabel('Titre')
plt.show()
```



CONCLUSION

Dans notre analyse, nous pouvons identifier à partir de notre analyse les informations suivantes :

-Le nombre de productions par Pays

-Meuilleurs compagnie Producteurs

-les Animes qui on générées plus de revenus

-Budget plus élevé pour la production

-Durée des animées le plus long (en minute)

-Les plus populaire

-Nombres d'animé par genre

In [ ]: