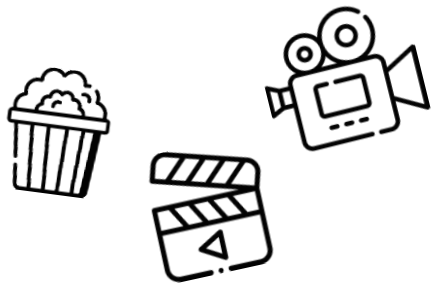


Мантатова Арюна, группа Data analyst

# Проектная работа по модулю SQL

**Анализ данных  
о фильмах с сайта IMDb**




# Описание проекта “Анализ данных о фильмах с сайта IMDB”

## Источник данных

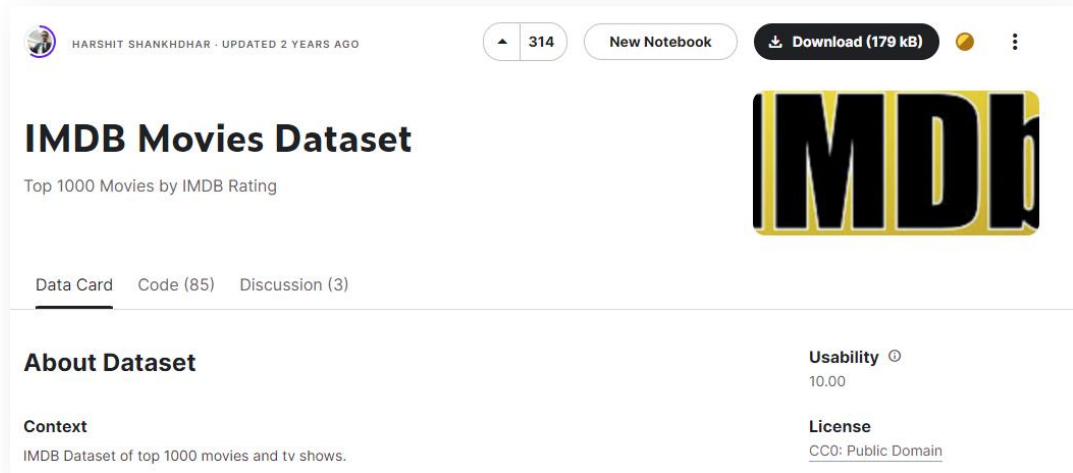
Датасет состоит из данные о 1000 фильмах с сайта IMDB. Источник – сайт Kaggle

Ссылка на датасет



## Атрибуты

Poster_Link	Ссылка на постер к фильму		
Series_Title	Название фильма	IMDB_Rating	Рейтинг IMDb
Released_Year	Год выпуска	Overview	Описание фильма
Certificate	Рейтинг возрастного ограничения	Meta_score	Баллы
Runtime	Длительность	Director	Режиссер
Genre	Жанр	Star1 - Star4	Имя знаменитого актера/актрисы
Gross	Кассовые сбора в USA (USD)	No_of_votes	Количество голосов



# Постановка задачи анализа

## Задача анализа

Определить какие факторы влияют на успешность фильмов в прокате

Для решения задачи, нужно ответить на следующие вопросы:

- 1 Фильмы с каким рейтингом заработали больше всего денег в прокате?
- 2 Фильмы каких режиссеров заработали больше всего денег в прокате?
- 3 Фильмы с каким сертификатом (возрастным рейтингом) заработали больше всего денег в прокате?
- 4 Фильмы какого жанра заработали больше всего денег в прокате?
- 5 Фильмы с какой длительностью заработали больше всего денег в прокате?

# Предобработка данных в Python

Загрузка данных в Jupiter notebook

```
In [1]: import pandas as pd
```

```
In [27]: # Загружаем данные  
df = pd.read_csv('imdb_top_1000.csv', sep=',')
```

```
In [28]: df.head()
```

Out[28]:

	Poster_Link	Series_Title	Released_Year	Certificate	Runtime	Genre	IMDB_Rating	Overview	Meta_score	Director	Star1
0	<a href="https://m.media-amazon.com/images/M/MV5BMDFkYT...">https://m.media-amazon.com/images/M/MV5BMDFkYT...</a>	The Shawshank Redemption	1994	A	142 min	Drama	9.3	Two imprisoned men bond over a number of years...	80.0	Frank Darabont	Tim Robbins
1	<a href="https://m.media-amazon.com/images/M/MV5BM2MyNj...">https://m.media-amazon.com/images/M/MV5BM2MyNj...</a>	The Godfather	1972	A	175 min	Crime, Drama	9.2	An organized crime dynasty's aging patriarch t...	100.0	Francis Ford Coppola	Marlon Brando
2	<a href="https://m.media-amazon.com/images/M/MV5BMTMxNT...">https://m.media-amazon.com/images/M/MV5BMTMxNT...</a>	The Dark Knight	2008	UA	152 min	Action, Crime, Drama	9.0	When the menace known as the Joker wreaks havo...	84.0	Christopher Nolan	Christian Bale

# Предобработка данных в Python

Проверяем данные на наличие дубликатов. Вывод: дубликатов нет

```
In [29]: df.shape
```

```
Out[29]: (1000, 16)
```

```
In [30]: # Проверяем наличие дубликатов в данных
```

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

Проверяем тип данных.

Вывод: Runtime и Gross нужно преобразовать в числовой формат

```
In [31]: df.dtypes
```

```
Out[31]: Poster_Link      object
Series_Title      object
Released_Year      object
Certificate      object
Runtime      object
Genre      object
IMDB_Rating      float64
Overview      object
Meta_score      float64
Director      object
Star1      object
Star2      object
Star3      object
Star4      object
No_of_Votes      int64
Gross      object
dtype: object
```

# Предобработка данных в Python

Проверяем количество пустых значений

Вывод: Пустые ячейки Gross нужно удалить, так как наличие данных в этом поле критично

```
In [32]: df.count()
```

```
Out[32]: Poster_Link    1000  
Series_Title    1000  
Released_Year    1000  
Certificate      899  
Runtime          1000  
Genre            1000  
IMDB_Rating      1000  
Overview         1000  
Meta_score       843  
Director         1000  
Star1            1000  
Star2            1000  
Star3            1000  
Star4            1000  
No_of_Votes      1000  
Gross            831  
dtype: int64
```

```
In [33]: # удаляю фильмы без данных по выручке  
df = df[~df.Gross.isna()]
```

# Предобработка данных в Python

Приводим поля Runtime и Gross к числовому формату

```
In [28]: df.head()
```

Out[28]:

title	Released_Year	Certificate	Runtime	Genre	IMDB_Rating	Overview	Meta_score	Director	Star1	Star2	Star3	Star4	No_of_Votes	Gross
The ink ion	1994	A	142 min	Drama	9.3	Two imprisoned men bond over a number of years...	80.0	Frank Darabont	Tim Robbins	Morgan Freeman	Bob Gunton	William Sadler	2343110	28,341,469

..

```
In [35]: # Приводим поле Gross к числовому формату, убираем запятые
```

```
df['Gross'] = df.Gross.str.replace(',', '').astype(int)
```

```
In [39]: # Приводим поле Runtime к числовому формату и убираем min в данных
```

```
df['Runtime'] = df.Runtime.apply(lambda x : x.split(' ')[0]).astype(int)
```

# Предобработка данных в Python

Проверили данные в поле Released\_Year, удалили строку с неправильными данными. Удалили поля, которые не нужны для анализа

```
In [44]: # удаляем строку с Released_Year = PG
df.Released_Year.sort_values()
```

```
Out[44]: 127      1921
         194      1924
         193      1925
         462      1925
         320      1926
         ...
         195      2019
         466      2019
         475      2019
         334      2019
         966       PG
         Name: Released_Year, Length: 831, dtype: object
```

```
In [45]: df = df.query('Released_Year != "PG"')
```

```
In [46]: # убираем колонки, которые не нужны для анализа
```

```
df = df.drop(columns=['Poster_Link', 'Overview', 'Star2', 'Star3', 'Star4'])
```



# Загрузка таблицы в базу данных SQLite

```
In [58]: from sqlalchemy import create_engine
```

```
In [59]: # Путь к файлу  
database_path = r'C:\Users\ryuna\Documents\IT\MathsHub\SQL\проект\imbd_db'
```

```
In [60]: # Создаю engine  
engine = create_engine(f'sqlite:/// {database_path}')
```

```
In [61]: # Подключаюсь к базе данных  
connection = engine.connect()
```

```
In [62]: # Название новой таблицы в базеданных  
table_name = 'project_movies'
```

```
In [63]: # Загружаю датафрейм в базу данных  
df.to_sql(table_name, engine, if_exists='replace', index=False)
```

```
Out[63]: 830
```

```
In [64]: # Закрываю соединение  
connection.close()
```

# Анализ данных с помощью SQL

1 Топ-10 фильмов с наибольшей выручкой

```
SELECT Series_Title, Gross  
FROM project_movies  
ORDER BY Gross DESC  
LIMIT 10
```

ABC Series_Title ▼	123 Gross ▼
Star Wars: Episode VII - The Force Awakens	936 662 225
Avengers: Endgame	858 373 000
Avatar	760 507 625
Avengers: Infinity War	678 815 482
Titanic	659 325 379
The Avengers	623 279 547
Incredibles 2	608 581 744
The Dark Knight	534 858 444
Rogue One	532 177 324
The Dark Knight Rises	448 139 099

# Анализ данных с помощью SQL

2 Фильмы с каким рейтингом заработали больше всего денег в прокате?

```
SELECT
  CASE
    WHEN IMDB_Rating >= 7.5 AND IMDB_Rating < 8 THEN '7.5 - 8'
    WHEN IMDB_Rating >= 8 AND IMDB_Rating < 8.5 THEN '8 - 8.5'
    WHEN IMDB_Rating >= 8.5 AND IMDB_Rating < 9 THEN '8.5 - 9'
    WHEN IMDB_Rating >= 9 AND IMDB_Rating < 9.5 THEN '9 - 9.5'
  ELSE ''
  END as IMDB_rating_bucket,
  ROUND(AVG(Gross),0) as avg_gross
FROM project_movies
GROUP BY IMDB_rating_bucket
ORDER BY avg_gross DESC
```

IMDB_rating_bucket	avg_gross
9 - 9.5	151 965 265
8.5 - 9	126 910 259
7.5 - 8	64 546 661
8 - 8.5	63 539 272

Фильмы с рейтингом от 9 до 9,5 в среднем зарабатывают в прокате больше денег, чем фильмы с более низким рейтингом.

Фильмы с рейтингом от 8,5 до 9,5 зарабатывают денег в прокате в среднем в два раза больше, чем фильмы с рейтингом от 7,5 до 8,5

# Анализ данных с помощью SQL

## 2 Фильмы с каким рейтингом заработали больше всего денег в прокате?

Проверим, сколько человек в среднем проголосовало за фильмы в разбивке по рейтингам

```
-- Проверим, сколько человек проголосовало за фильмы в разбивке по рейтингам
SELECT
  CASE
    WHEN IMDB_Rating >= 7.5 AND IMDB_Rating < 8 THEN '7.5 - 8'
    WHEN IMDB_Rating >= 8 AND IMDB_Rating < 8.5 THEN '8 - 8.5'
    WHEN IMDB_Rating >= 8.5 AND IMDB_Rating < 9 THEN '8.5 - 9'
    WHEN IMDB_Rating >= 9 AND IMDB_Rating < 9.5 THEN '9 - 9.5'
  ELSE ''
  END as IMDB_rating_bucket,
  ROUND(AVG(No_of_Votes),0) as avg_votes
FROM project_movies
GROUP BY IMDB_rating_bucket
ORDER BY avg_votes DESC
```

ABC IMDB_rating_bucket ▼	123 avg_votes ▼
9 - 9.5	1 617 301
8.5 - 9	1 039 062
8 - 8.5	339 980
7.5 - 8	214 866

В среднем один фильм с рейтингом 8.5 – 9.5 оценило от 1 млн до 1,6 млн человек

Следовательно, большее количество зрителей оценивают фильмы с высоким рейтингом.  
Рейтинги можно считать валидными

# Анализ данных с помощью SQL

3 Фильмы каких режиссеров заработали больше всего денег в прокате?

```
SELECT
    Director,
    ROUND(AVG(Gross), 0) as avg_gross
FROM project_movies
GROUP BY Director
ORDER BY avg_gross DESC
LIMIT 10
```

ABC Director ▼	123 avg_gross ▼
Anthony Russo	551 259 851
Gareth Edwards	532 177 324
J.J. Abrams	474 390 302
Josh Cooley	434 038 008
Roger Allers	422 783 777
Tim Miller	363 070 709
James Gunn	361 494 851
James Cameron	349 647 320
Byron Howard	341 268 248
David Yates	326 317 907

В среднем один фильм таких режиссеров как A. Russo, G. Edwards, JJ. Abrams зарабатывает больше денег в прокате, чем фильмы других режиссеров

# Анализ данных с помощью SQL

## 3 Фильмы каких режиссеров заработали больше всего денег в прокате?

```
SELECT
  Director,
  Series_Title,
  Gross
FROM project_movies
WHERE Director IN ('Anthony Russo', 'Gareth Edwards', 'J.J. Abrams')
ORDER BY Director, Gross
```

ABC Director ▼	ABC Series_Title ▼	123 Gross ▼
Anthony Russo	Captain America: The Winter Soldier	259 766 572
Anthony Russo	Captain America: Civil War	408 084 349
Anthony Russo	Avengers: Infinity War	678 815 482
Anthony Russo	Avengers: Endgame	858 373 000
Gareth Edwards	Rogue One	532 177 324
J.J. Abrams	Star Trek Into Darkness	228 778 661
J.J. Abrams	Star Trek	257 730 019
J.J. Abrams	Star Wars: Episode VII - The Force Awakens	936 662 225

Все три режиссера снимают фильмы в жанре фантастика

Предполагаем, что фантастические фильмы зарабатывают больше всего денег в прокате в США

# Анализ данных с помощью SQL

4 Проверим какой жанр у топ-10 фильмов с наибольшей выручкой в прокате

ABC Series_Title	ABC Genre	123 Gross	
Star Wars: Episode VII - The Force Awakens	Action, Adventure, Sci-Fi	936 662 225	Фантастика
Avengers: Endgame	Action, Adventure, Drama	858 373 000	Фантастика
Avatar	Action, Adventure, Fantasy	760 507 625	Фантастика
Avengers: Infinity War	Action, Adventure, Sci-Fi	678 815 482	Фантастика
Titanic	Drama, Romance	659 325 379	
The Avengers	Action, Adventure, Sci-Fi	623 279 547	Фантастика
Incredibles 2	Animation, Action, Adventure	608 581 744	
The Dark Knight	Action, Crime, Drama	534 858 444	Фантастика
Rogue One	Action, Adventure, Sci-Fi	532 177 324	Фантастика
The Dark Knight Rises	Action, Adventure	448 139 099	Фантастика

-- Топ-10 фильмов с наибольшей выручкой

```
SELECT Series_Title, Genre, Gross
FROM project_movies
ORDER BY Gross DESC
LIMIT 10
```

Большинство фильмов (8 из 10) относятся к жанру фантастика. Возможно фильмы именно этого жанра зарабатывают больше денег в прокате из-за спецэффектов. Таким образом, люди предпочитают смотреть эти фильмы в кинотеатре, а не дома.

В поле Genre указывается большее одного значения, поэтому я анализирую только топ-10 фильмов, а не все

# Анализ данных с помощью SQL

5 Фильмы с каким сертификатом (возрастным рейтингом) заработали больше всего денег в прокате?

```
SELECT
    Certificate,
    ROUND(AVG(Gross),0) as avg_gross
FROM project_movies
GROUP BY Certificate
ORDER BY avg_gross DESC
LIMIT 10
```

ABC Certificate	123 avg_gross
UA	131 145 955
U	86 811 448
A	63 887 133
G	49 596 326
PG-13	36 947 857
U/A	26 020 957
R	25 793 599
PG	17 888 273
Approved	6 949 014
Passed	5 435 554

## **U (Unrestricted Public Exhibition)** -

This U certification is the basic one it means that anybody can see the movie. It can have mild violence or intimate scene but those are overseen

## **U/A (Parental Guidance for children below the age of 12 years)-**

These may contain some moderate intimate or violence scene.

## **A (Restricted to adults)** -

Strong intimate scenes, abusive language that's why the window covers only adults.

В среднем больше зарабатывают фильмы с рейтингом UA (дети до 12 лет могут смотреть фильм только с разрешения взрослых)



# Анализ данных с помощью SQL

6 Фильмы с какой длительностью заработали больше всего денег в прокате?

```
SELECT
  CASE
    WHEN Runtime < 60 THEN 'менее часа'
    WHEN Runtime >= 60 AND Runtime < 90 THEN 'от часа до 1,5 часов'
    WHEN Runtime >= 90 AND Runtime < 120 THEN 'от 1,5 часов до 2 часов'
    WHEN Runtime >= 120 AND Runtime < 180 THEN 'от 2 часов до 2,5 часов'
    ELSE 'более 2,5 часов'
  END as runtime_bucket,
  ROUND(AVG(Gross),0) as avg_gross
FROM project_movies
GROUP BY runtime_bucket
ORDER BY avg_gross DESC
```

runtime_bucket	avg_gross
более 2,5 часов	90 285 738
от 2 часов до 2,5 часов	81 356 901
от 1,5 часов до 2 часов	55 036 895
от часа до 1,5 часов	37 327 065
менее часа	977 375

```
-- Проверим сколько фильмов в каждом бачете
SELECT
  CASE
    WHEN Runtime < 60 THEN 'менее часа'
    WHEN Runtime >= 60 AND Runtime < 90 THEN 'от часа до 1,5 часов'
    WHEN Runtime >= 90 AND Runtime < 120 THEN 'от 1,5 часов до 2 часов'
    WHEN Runtime >= 120 AND Runtime < 180 THEN 'от 2 часов до 2,5 часов'
    ELSE 'более 2,5 часов'
  END as runtime_bucket,
  COUNT(Gross) as count_movies
FROM project_movies
GROUP BY runtime_bucket
ORDER BY count_movies DESC
```

runtime_bucket	count_movies
от 2 часов до 2,5 часов	390
от 1,5 часов до 2 часов	352
от часа до 1,5 часов	49
более 2,5 часов	38
менее часа	1

В среднем больше зарабатывают фильмы с длительностью от 2 до 2,5 часов.  
Фильмов в этой категории больше, чем фильмов с длительностью более 2,5 часов

# Основные выводы

Наибольшую выручку в прокате в США заработали следующие фильмы:

- 1 Фильмы с рейтингом от 8,5 до 9,5 зарабатывают денег в прокате в среднем в два раза больше, чем фильмы с рейтингом от 7,5 до 8,5
- 2 В среднем один фильм таких режиссеров как A. Russo, G. Edwards, JJ. Abrams зарабатывает больше денег в прокате, чем фильмы других режиссеров
- 3 Фильмы фантастического жанра зарабатывают больше в прокате, чем фильмы других жанров
- 4 Фильмы с рейтингом UA (дети до 12 лет могут смотреть фильм только с разрешения взрослых) в среднем больше зарабатывают больше денег в прокате, чем фильмы с другим возрастным рейтингом
- 5 В среднем больше зарабатывают фильмы с длительностью от 2 до 2,5 часов.

# Считывание данных из базы с помощью Python

```
In [30]: # импортируем библиотеку
import sqlite3
```

```
In [10]: #подключение к бд
con = sqlite3.connect(r'C:\Users\ryuna\Documents\IT\MathsHub\SQL\проект\imbd_db', timeout=10)
cur = con.cursor()
```

```
In [17]: #считывание данных из таблицы

avg_gross_by_year_df = pd.read_sql(sql='select * from avg_gross_by_year', con=con)
avg_gross_by_year_df
```

Сортируем полученные данные по возрастанию года выпуска фильма

```
: avg_gross_by_year_df = avg_gross_by_year_df.sort_values(by='Released_Year')
avg_gross_by_year_df.head()
```

```
:
```

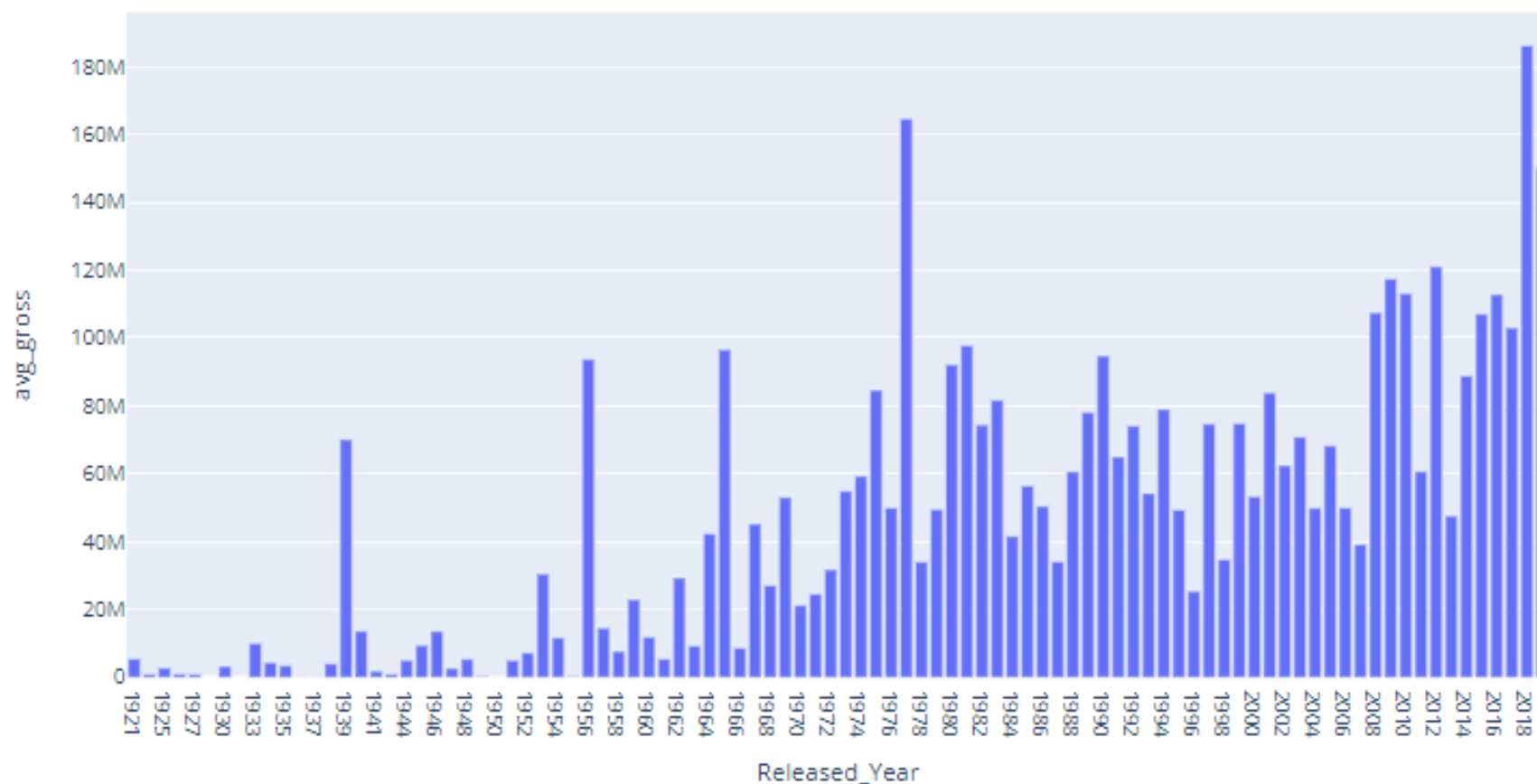
	Released_Year	avg_gross
72	1921	5450000.0
85	1924	977375.0
80	1925	2750485.0
83	1926	1033895.0
86	1927	887853.0

# Визуализация данных с помощью Python

Строим график с средней выручкой фильмов по годам

```
In [ ]: import plotly.express as px
```

```
In [35]: fig = px.bar(avg_gross_by_year_df, x='Released_Year', y='avg_gross')  
fig.show()
```



В целом в среднем фильмы стали зарабатывать больше денег в прокате

Возможные причины: рост благосостояния населения, улучшение качества фильмов, рост числа кинотеатров, инфляция

# Визуализация данных с помощью Python

Строим график с средней выручкой фильмов по категориям рейтингов

```
: # график средней выручки фильма по категории рейтинга  
# импортируем данные из БД
```

```
with_rating_bucket_df = pd.read_sql(sql='select * from df_with_rating_bucket', con=con)  
with_rating_bucket_df.head()
```

	Series_Title	Released_Year	Certificate	Runtime	Genre	IMDB_Rating	Meta_score	Director	Star1	No_of_Votes	Gross	IMDB_rating_bucket
0	The Shawshank Redemption	1994	A	142	Drama	9.3	80.0	Frank Darabont	Tim Robbins	2343110	28341469	9 - 9.5
1	The Godfather	1972	A	175	Crime, Drama	9.2	100.0	Francois Ford Coppola	Marlon Brando	1620367	134966411	9 - 9.5
2	The Dark Knight	2008	UA	152	Action, Crime, Drama	9.0	84.0	Christopher Nolan	Christian Bale	2303232	534858444	9 - 9.5
3	The Godfather: Part II	1974	A	202	Crime, Drama	9.0	90.0	Francois Ford Coppola	Al Pacino	1129952	57300000	9 - 9.5
4	12 Angry Men	1957	U	96	Crime, Drama	9.0	96.0	Sidney Lumet	Henry Fonda	689845	4360000	9 - 9.5

```
: rating_df = with_rating_bucket_df.groupby(['IMDB_rating_bucket'], as_index=False).agg({'Gross': 'mean'}).round(0)  
rating_df
```

	IMDB_rating_bucket	Gross
0	7.5 - 8	64546661.0
1	8 - 8.5	63539272.0
2	8.5 - 9	126910259.0
3	9 - 9.5	151965265.0

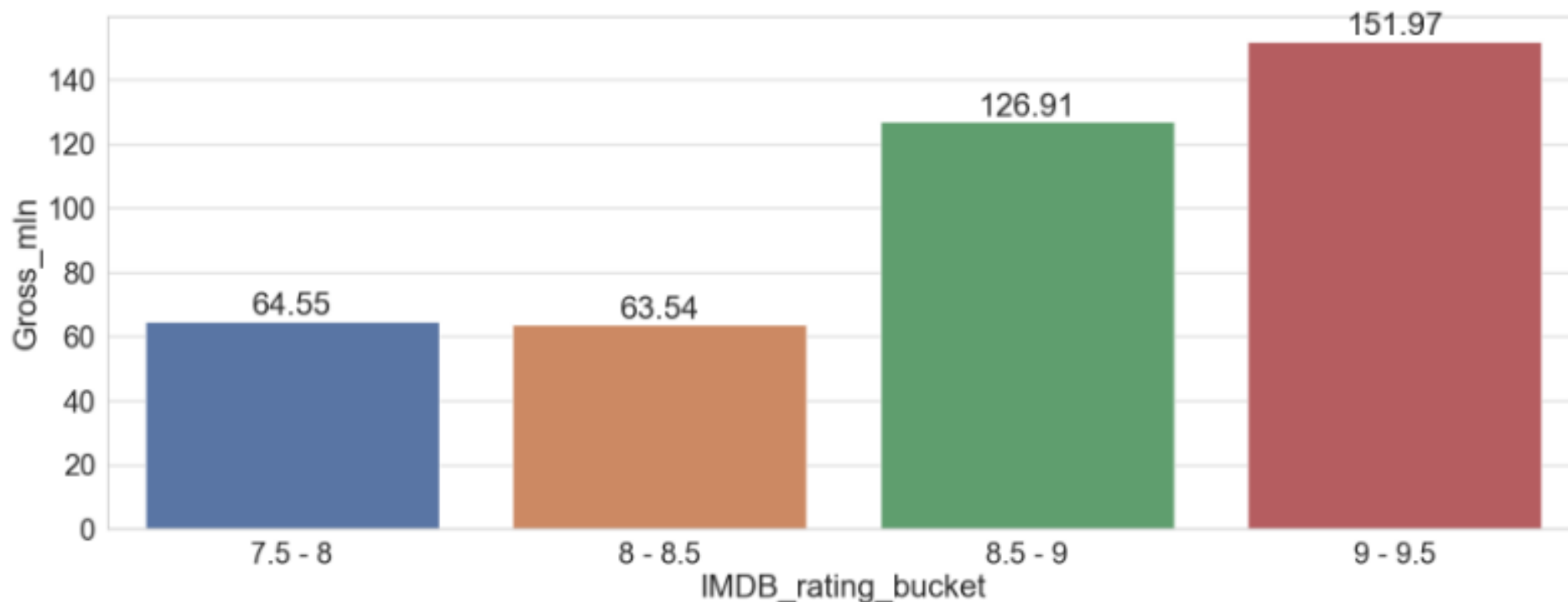
# Визуализация данных с помощью Python

Строим график с средней выручкой фильмов по категориям рейтингов

```
rating_df['Gross_mln'] = (rating_df.Gross / 1000000).round(2)

ax = sns.barplot(data=rating_df, x='IMDB_rating_bucket', y='Gross_mln')

for i in ax.containers:
    ax.bar_label(i,)
```



# Визуализация данных с помощью Python

Строим график с топ-10 фильмов по выручке

Предобработка данных

```
: # топ-10 фильмов по выручке в прокате

# импорт данных из БД
top_df = pd.read_sql(sql='select * from project_movies', con=con)
```

```
: # сортировка по убыванию выручки, выбираем топ 10
top_df = top_df.sort_values('Gross', ascending=False).head(10)
top_df.head(2)
```

```
:
```

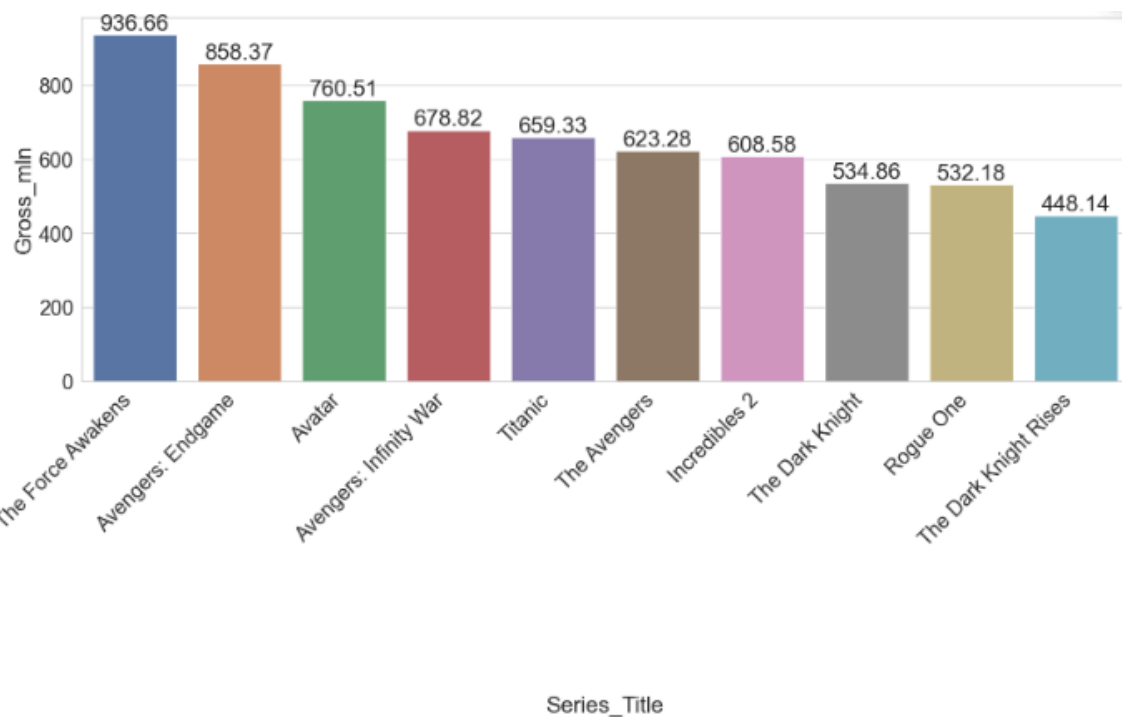
	Series_Title	Released_Year	Certificate	Runtime	Genre	IMDB_Rating	Meta_score	Director	Star1	No_of_Votes	Gross
388	Star Wars: Episode VII - The Force Awakens	2015	U	138	Action, Adventure, Sci-Fi	7.9	80.0	J.J. Abrams	Daisy Ridley	860823	936662225
52	Avengers: Endgame	2019	UA	181	Action, Adventure, Drama	8.4	78.0	Anthony Russo	Joe Russo	809955	858373000

```
: # создаем колонку с данными в млн долл
top_df['Gross_mln'] = (top_df.Gross / 1000000).round(2)
```

# Визуализация данных с помощью Python

Строим график с топ-10 фильмов по выручке

```
ax = sns.barplot(data=top_df, x='Series_Title', y='Gross_mln')  
  
for i in ax.containers:  
    ax.bar_label(i,  
                )  
  
ax.set_xticklabels(top_df.Series_Title, rotation=45, horizontalalignment='right')
```





**Спасибо за внимание!**

