Анализ базы данных книг

Коронавирус застал мир врасплох, изменив привычный порядок вещей. В свободное время жители городов больше не выходят на улицу, не посещают кафе и торговые центры. Зато стало больше времени для книг. Это заметили стартаперы — и бросились создавать приложения для тех, кто любит читать.

Ваша компания решила быть на волне и купила крупный сервис для чтения книг по подписке. Ваша первая задача как аналитика — проанализировать базу данных. В ней — информация о книгах, издательствах, авторах, а также пользовательские обзоры книг. Эти данные помогут сформулировать ценностное предложение для нового продукта.

Описание данных

Таблица `books`

Содержит данные о книгах:

- `book_id` — идентификатор книги;
- `author_id` — идентификатор автора;
- `title` — название книги;
- `num_pages` — количество страниц;
- `publication_date` — дата публикации книги;
- `publisher_id` — идентификатор издателя.

Таблица `authors`

Содержит данные об авторах:

- `author_id` — идентификатор автора;
- `author` — имя автора.

Таблица `publishers`

Содержит данные об издательствах:

- `publisher_id` — идентификатор издательства;
- `publisher` — название издательства;

Таблица `ratings`

Содержит данные о пользовательских оценках книг:

- `rating_id` — идентификатор оценки;
- `book_id` — идентификатор книги;
- `username` — имя пользователя, оставившего оценку;
- `rating` — оценка книги.

Таблица `reviews`

Содержит данные о пользовательских обзорах на книги:

- `review_id` — идентификатор обзора;
- `book_id` — идентификатор книги;
- `username` — имя пользователя, написавшего обзор;
- `text` — текст обзора.

План работы

Подключение к БД. Просмотр таблиц

In [1]:
```
#!/usr/bin/python
# -*- coding: utf-8 -*-

!pip install psycopg2-binary==2.8.4 -U

import pandas as pd
from sqlalchemy import create_engine
# устанавливаем параметры
db_config = {'user': 'praktikum_student', # имя пользователя
 'pwd': 'Sdf4$2;d-d30pp', # пароль
 'host': 'rc1b-wcoijxj3yxfsf3fs.mdb.yandexcloud.net',
 'port': 6432, # порт подключения
 'db': 'data-analyst-final-project-db'} # название базы данных
connection_string = 'postgresql://{}:{}@{}:{}/{}'.format(db_config['user'],
 db_config['pwd'],
 db_config['host'],
 db_config['port'],
 db_config['db'])
# сохраняем коннектор
engine = create_engine(connection_string, connect_args={'sslmode':'require'})
```

```
Requirement already up-to-date: psycopg2-binary==2.8.4 in c:\users\admin\anaconda3\l
ib\site-packages (2.8.4)
```

Посмотрим на имеющиеся таблицы

In [2]:
```
query = '''SELECT
         *
         FROM books
         LIMIT 5;
         '''
```

In [3]:
```
pd.io.sql.read_sql(query, con = engine)
```

Out[3]:

	book_id	author_id	title	num_pages	publication_date	publisher_id
0	1	546	'Salem's Lot	594	2005-11-01	93
1	2	465	1 000 Places to See Before You Die	992	2003-05-22	336
2	3	407	13 Little Blue Envelopes (Little Blue Envelope...	322	2010-12-21	135
3	4	82	1491: New Revelations of the Americas Before C...	541	2006-10-10	309

	book_id	author_id	title	num_pages	publication_date	publisher_id
4	5	125	1776	386	2006-07-04	268

In [4]:
```
query = '''SELECT *
FROM authors
LIMIT 5;'''
pd.io.sql.read_sql(query, con = engine)
```

Out[4]:

	author_id	author
0	1	A.S. Byatt
1	2	Aesop/Laura Harris/Laura Gibbs
2	3	Agatha Christie
3	4	Alan Brennert
4	5	Alan Moore/David Lloyd

In [5]:
```
query = '''
SELECT *
FROM publishers
LIMIT 5;'''
pd.io.sql.read_sql(query, con = engine)
```

Out[5]:

	publisher_id	publisher
0	1	Ace
1	2	Ace Book
2	3	Ace Books
3	4	Ace Hardcover
4	5	Addison Wesley Publishing Company

In [6]:
```
query = '''
SELECT *
FROM publishers
;'''
pd.io.sql.read_sql(query, con = engine).info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 340 entries, 0 to 339
Data columns (total 2 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   publisher_id  340 non-null    int64 
 1   publisher     340 non-null    object
dtypes: int64(1), object(1)
memory usage: 5.4+ KB
```

In [7]:
```
query = """
SELECT *
FROM ratings
LIMIT 5;"""
pd.io.sql.read_sql(query, con = engine)
```

Out[7]:

	rating_id	book_id	username	rating
0	1	1	ryanfranco	4

	rating_id	book_id	username	rating
1	2	1	grantpatricia	2
2	3	1	brandtandrea	5
3	4	2	lorichen	3
4	5	2	mariokeller	2

In [8]:
```
query = """
SELECT *
FROM ratings;"""
pd.io.sql.read_sql(query, con = engine).info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6456 entries, 0 to 6455
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   rating_id  6456 non-null   int64
 1   book_id    6456 non-null   int64
 2   username   6456 non-null   object
 3   rating     6456 non-null   int64
dtypes: int64(3), object(1)
memory usage: 201.9+ KB
```

In [9]:
```
query = '''
SELECT *
FROM reviews
LIMIT 5;'''
pd.io.sql.read_sql(query, con = engine)
```

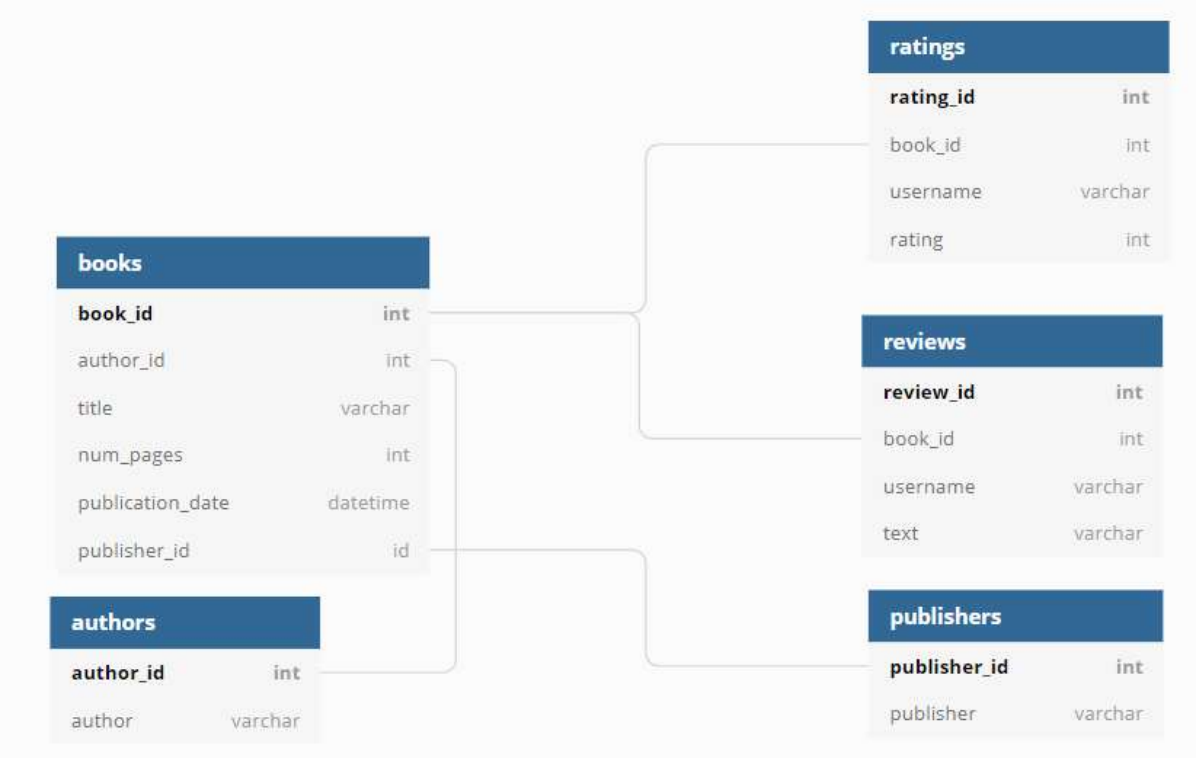
Out[9]:

	review_id	book_id	username	text
0	1	1	brandtandrea	Mention society tell send professor analysis. ...
1	2	1	ryanfranco	Foot glass pretty audience hit themselves. Amo...
2	3	2	lorichen	Listen treat keep worry. Miss husband tax but ...
3	4	3	johnsonamanda	Finally month interesting blue could nature cu...
4	5	3	scotttamara	Nation purpose heavy give wait song will. List...

In [10]:
```
query = '''
SELECT *
FROM reviews
;'''
pd.io.sql.read_sql(query, con = engine).info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2793 entries, 0 to 2792
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   review_id  2793 non-null   int64
 1   book_id    2793 non-null   int64
 2   username   2793 non-null   object
 3   text       2793 non-null   object
dtypes: int64(2), object(2)
memory usage: 87.4+ KB
```

Все таблицы связаны между собой



Краткий анализ

Посчитаем, сколько книг вышло после 1 января 2000 года

In [11]:
```
query = '''
SELECT
COUNT(book_id)
FROM books
WHERE
CAST(publication_date AS date) > '2000-01-01'
;
'''
pd.io.sql.read_sql(query, con = engine)
```

Out[11]:

	count
0	819

Вышло 819 книг

Для каждой книги посчитаем количество отзывов и среднюю оценку

In [12]:
```
query = '''
SELECT

SUBQ1.title,
SUM(SUBQ1.n_reviews),
SUBQ2.avg_rating

FROM
(
SELECT
books.title,
books.book_id,
COUNT(reviews.review_id) AS n_reviews
FROM
books
INNER JOIN reviews ON reviews.book_id = books.book_id
```

```
GROUP BY
books.title,
books.book_id

)
SUBQ1
INNER JOIN (
SELECT
books.title,
books.book_id,
AVG(ratings.rating) AS avg_rating
FROM
books
INNER JOIN ratings ON ratings.book_id = books.book_id
GROUP BY
books.title,
books.book_id
) SUBQ2 on SUBQ2.book_id = SUBQ1.book_id
GROUP BY
SUBQ1.title,

SUBQ2.avg_rating
ORDER BY SUM(SUBQ1.n_reviews)


;'''
pd.io.sql.read_sql(query, con = engine)
```

Out[12]:

	title	sum	avg_rating
0	Ten Apples Up On Top!	1.0	4.000000
1	To Green Angel Tower (Memory Sorrow and Thor...	1.0	4.500000
2	Debt of Honor (Jack Ryan #7)	1.0	3.000000
3	Death: The High Cost of Living	1.0	3.000000
4	Creepshow	1.0	4.500000
...	...	...	...
989	Outlander (Outlander #1)	6.0	4.125000
990	The Glass Castle	6.0	4.206897
991	Harry Potter and the Prisoner of Azkaban (Harr...	6.0	4.414634
992	Water for Elephants	6.0	3.977273
993	Twilight (Twilight #1)	7.0	3.662500

994 rows × 3 columns

Больше всего отзывов у первой части первой части книги "Сумерки", но средняя оценка не очень высокая. Наверное, поэтому на втором месте по количеству отзывов не вторая часть :)

Определим издательство, которое выпустило наибольшее число книг толще 50 страниц — так исключим из анализа брошюры

In [13]:

```
query = '''
SELECT
publishers.publisher,
COUNT(books.book_id)
```

```
FROM
books
INNER JOIN publishers ON publishers.publisher_id = books.publisher_id
WHERE
books.num_pages > 50
GROUP BY
publishers.publisher
ORDER BY
COUNT(books.book_id) DESC
LIMIT 5;

;'''
pd.io.sql.read_sql(query, con = engine)
```

Out[13]:

	publisher	count
0	Penguin Books	42
1	Vintage	31
2	Grand Central Publishing	25
3	Penguin Classics	24
4	Bantam	19

Penguin Books выпустило большего всего книг

Определим автора с самой высокой средней оценкой книг — будем учитывать только книги с 50 и более оценками

In [14]:

```
query = '''
SELECT
authors.author,
AVG(SUBQ1.rating)

FROM
(
SELECT
books.book_id,
books.author_id,
AVG(ratings.rating) AS rating
FROM
books
INNER JOIN ratings on ratings.book_id = books.book_id
GROUP BY
books.book_id

HAVING COUNT(ratings.rating) >= 50
) SUBQ1
INNER JOIN authors ON authors.author_id = SUBQ1.author_id
GROUP BY
authors.author
ORDER BY
AVG(SUBQ1.rating) DESC
LIMIT 5
;'''
pd.io.sql.read_sql(query, con = engine)
```

Out[14]:

	author	avg
0	J.K. Rowling/Mary GrandPré	4.283844
1	Markus Zusak/Cao Xuân Việt Khương	4.264151

	author	avg
2	J.R.R. Tolkien	4.258446
3	Louisa May Alcott	4.192308
4	Rick Riordan	4.080645

Самые высокие средние оценки у книг Дж.К. Роулинг

Посчитаем среднее количество обзоров от пользователей, которые поставили больше 50 оценок

В среднем активный пользователь оставляет около 24 обзоров

In [16]:
```
query = '''
SELECT
AVG(SUBQ.n_reviews)

FROM
(
SELECT
username,
COUNT(review_id) AS n_reviews
FROM
reviews
GROUP BY
username

) SUBQ

WHERE SUBQ.username IN
(
SELECT
username
FROM
ratings
GROUP BY
username
HAVING COUNT(rating) > 50
)

;
'''
pd.io.sql.read_sql(query, con = engine)
```

Out[16]:

	avg
0	24.333333

Вывод

Мы знаем самую популярную книгу, автора с самыми высокими оценками, среднее количество обзоров. Можем рекомендовать популярные книги, авторов, давать рекламу популярных издательств