Анализ базы данных книжного сервиса

Содержание

В условиях коронавирусных ограничений возросла популярность чтения. Наш стартап разрабатывает инновационное приложение для любителей чтения.

Цель исследования: сформулировать ценностное предложение для нового продукта.

Задачи исследования: проанализировать базу данных крупного сервиса для чтения книг по подписке и ответить на основные вопросы:

- сколько книг вышло после 1 января 2000 года;
- для каждой книги посчитать количество обзоров и среднюю оценку;
- определить издательство, которое выпустило наибольшее число книг толще 50 страниц (исключит из анализа брошюры);
- определить автора с самой высокой средней оценкой книг (учитывать только книги с 50 и более оценками);
- посчитать среднее количество обзоров от пользователей, которые поставили больше 50 оценок.

Примерный план исследования:

- исследовать таблицы, вывести первые строки;
- выполнить SQL-запросы, результаты которых отвечают на вопросы исследования;
- сделать выводы.

Библиотеки, используемые в исследовании

In [1]:
```
import pandas as pd
from sqlalchemy import create_engine
```

Загрузка и обзор данных

In [2]:
```
# определение функции обзора данных
# ================================================
# на вход подаётся датафрейм df
# на выходе:
#     - 10 случайных строк df
#     - информация df.info()
#     - количество явных дубликатов в строках df
#     - процент пропусков данных в столбцах df
# ================================================
def data_observe(df):
    row_num = 5    # количество отображаемых строк таблицы

    print('Размерность данных (row, col):', df.shape)
    print('============================\n')

    print('Произвольные строки таблицы:')
    print('============================')
    if len(df) >= row_num:
        display(df.sample(row_num))
    else:
        display(df)

    print('\nИнформация о таблице:')
    print('=====================')
    df.info()

    print('\nКоличество явных дубликатов в таблице:')
    print('======================================')
    print(df.duplicated().sum())

    print('\nПроцент пропусков в столбцах:')
    print('============================')
    display(pd.DataFrame(
                round((df.isna().mean()*100),2), columns=['NaNs, %'])
                .sort_values(by='NaNs, %', ascending=False
            )
```

```
        .style.format('{:.2f}')
        .background_gradient('coolwarm')
    )
```

Установка соединения с базой данных

In [3]:
```
# устанавливаем параметры
db_config = {'user': 'user',           # имя пользователя
             'pwd': 'slHke57#2',               # пароль
             'host': 'book.service.net',
             'port': 6432,                         # порт подключения
             'db': 'books-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'})
```

Обзор таблицы `books`

Познакомимся с данными таблицы `books`:

In [4]:
```
# выведем первые 10 строк таблицы
query = '''
SELECT *
FROM books
LIMIT 10
'''

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

Out[4]:

	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
4	5	125	1776	386	2006-07-04	268
5	6	257	1st to Die (Women's Murder Club #1)	424	2005-05-20	116
6	7	258	2nd Chance (Women's Murder Club #2)	400	2005-05-20	116
7	8	260	4th of July (Women's Murder Club #4)	448	2006-06-01	318
8	9	563	A Beautiful Mind	461	2002-02-04	104
9	10	445	A Bend in the Road	341	2005-04-01	116

Согласно описанию данных, таблица 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.

Обзор таблицы `authors`

Познакомимся с данными таблицы `authors`:

In [5]:
```
# выведем первые 10 строк таблицы
query = '''
SELECT *
FROM authors
LIMIT 10
'''

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

Out[5]:

	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
5	6	Alan Paton
6	7	Albert Camus/Justin O'Brien
7	8	Aldous Huxley
8	9	Aldous Huxley/Christopher Hitchens
9	10	Aleksandr Solzhenitsyn/H.T. Willetts

Согласно описанию данных, таблица содержит данные об авторах:

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

Обзор таблицы `publishers`

Познакомимся с данными таблицы `publishers`:

In [6]:
```
# выведем первые 10 строк таблицы
query = '''
SELECT *
FROM publishers
LIMIT 10
'''

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

Out[6]:

	publisher_id	publisher
0	1	Ace
1	2	Ace Book
2	3	Ace Books
3	4	Ace Hardcover
4	5	Addison Wesley Publishing Company
5	6	Aladdin
6	7	Aladdin Paperbacks
7	8	Albin Michel
8	9	Alfred A. Knopf
9	10	Alfred A. Knopf Books for Young Readers

Согласно описанию данных, таблица содержит данные об издательствах:

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

Обзор таблицы `ratings`

Познакомимся с данными таблицы `ratings`:

In [7]:
```
# выведем первые 10 строк таблицы
query = '''
SELECT *
FROM ratings
LIMIT 10
'''

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

Out[7]:

	rating_id	book_id	username	rating
0	1	1	ryanfranco	4
1	2	1	grantpatricia	2
2	3	1	brandtandrea	5
3	4	2	lorichen	3
4	5	2	mariokeller	2
5	6	3	johnsonamanda	4
6	7	3	scotttamara	5
7	8	3	lesliegibbs	5
8	9	4	abbottjames	5
9	10	4	valenciaanne	4

Согласно описанию данных, таблица содержит данные о пользовательских оценках книг:

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

Обзор таблицы `reviews`

Познакомимся с данными таблицы `reviews`:

In [8]:
```
# выведем первые 10 строк таблицы
query = '''
SELECT *
FROM reviews
LIMIT 10
'''

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

Out[8]:

	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...
5	6	3	lesliegibbs	Analysis no several cause international.
6	7	4	valenciaanne	One there cost another. Say type save. With pe...
7	8	4	abbottjames	Within enough mother. There at system full rec...
8	9	5	npowers	Thank now focus realize economy focus fly. Ite...
9	10	5	staylor	Game push lot reduce where remember. Including...

Согласно описанию данных, таблица содержит данные о пользовательских обзорах на книги:

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

Ответы на вопросы исследования

Сколько книг вышло после 1 января 2000 года

In [9]:
```
# составим запрос
query = '''
SELECT COUNT(book_id)
FROM books
WHERE publication_date > '2000-01-01'
'''

# выполним запрос
pd.io.sql.read_sql(query, con = engine)
```

Out[9]:

	count
0	819

После 1 января 2000 года вышло 819 книг.

Для каждой книги посчитать количество обзоров и среднюю оценку;

In [10]:
```
# составим запрос
query = '''
WITH
books_review_counts AS
  (SELECT book_id,
          COUNT(review_id) AS review_count
   FROM reviews
   GROUP BY book_id),

books_ratings AS
  (SELECT book_id,
          AVG(rating) AS avg_rating
   FROM ratings
   GROUP BY book_id)

SELECT b.book_id,
       b.title,
       rc.review_count,
       br.avg_rating
FROM books AS b
LEFT JOIN books_review_counts AS rc ON b.book_id = rc.book_id
LEFT JOIN books_ratings AS br ON b.book_id = br.book_id
'''

# выполним запрос
pd.io.sql.read_sql(query, con = engine)
```

Out[10]:

	book_id	title	review_count	avg_rating
0	652	The Body in the Library (Miss Marple #3)	2.0	4.500000
1	273	Galápagos	2.0	4.500000
2	51	A Tree Grows in Brooklyn	5.0	4.250000
3	951	Undaunted Courage: The Pioneering First Missio...	2.0	4.000000
4	839	The Prophet	4.0	4.285714
...	...	...	...	...
995	672	The Cat in the Hat and Other Dr. Seuss Favorites	NaN	5.000000
996	83	Anne Rice's The Vampire Lestat: A Graphic Novel	NaN	3.666667
997	221	Essential Tales and Poems	NaN	4.000000
998	387	Leonardo's Notebooks	NaN	4.000000
999	808	The Natural Way to Draw	NaN	3.000000

1000 rows × 4 columns

Мы посчитали количество обзоров и средний рейтинг для каждой книги. Отметим, что не обо всех книгах читатели оставили отзывы.

Определить издательство, которое выпустило наибольшее число книг толще 50 страниц (исключит из анализа брошюры);

In [11]:
```
# составим запрос
query = '''
WITH
books_50_publishers AS
  (SELECT publisher_id,
          COUNT(book_id) AS books_published
   FROM books
   WHERE num_pages > 50
   GROUP BY publisher_id)

SELECT p.publisher,
       bp.books_published
FROM publishers AS p
INNER JOIN books_50_publishers AS bp ON p.publisher_id = bp.publisher_id
ORDER BY books_published DESC
LIMIT 1
'''

# выполним запрос
pd.io.sql.read_sql(query, con = engine)
```

Out[11]:

	publisher	books_published
0	Penguin Books	42

Больше всего книг, содержащих более 50 страниц, выпустило издательство *Penguin Books*.

Определить автора с самой высокой средней оценкой книг (учитывать только книги с 50 и более оценками);

In [12]:
```
# составим запрос
query = '''
WITH
-- посчитаем оценки для книг и выберем с >= 50 оценок
fifty_rating_books AS
  (SELECT book_id
   FROM ratings
   GROUP BY book_id
   HAVING COUNT(rating_id) >= 50),

-- посчитаем средние оценки выбранных книг
selected_books_ratings AS
  (SELECT book_id,
          AVG(rating) AS avg_rating
   FROM ratings
   WHERE book_id IN (SELECT * FROM fifty_rating_books)
   GROUP BY book_id)

SELECT a.author,
       AVG(sbr.avg_rating) AS avg_rating
FROM selected_books_ratings AS sbr
LEFT JOIN books AS b ON sbr.book_id = b.book_id
LEFT JOIN authors AS a ON a.author_id = b.author_id
GROUP BY a.author
ORDER BY avg_rating DESC
LIMIT 1
'''

# выполним запрос
pd.io.sql.read_sql(query, con = engine)
```

Out[12]:

	author	avg_rating
0	J.K. Rowling/Mary GrandPré	4.283844

Авторами с самой высокой средней оценкой книг является писательский дуэт *J.K. Rowling / Mary GrandPré*.

Посчитать среднее количество обзоров от пользователей, которые поставили больше 50 оценок.

In [13]:
```
# составим запрос
query = '''
WITH
-- посчитаем количество оценок для каждого пользователя и выберем > 50
fifty_rating_users AS
(SELECT username
FROM ratings
GROUP BY username
HAVING COUNT(rating_id) > 50),

--посчитаем количество обзоров у выделенных пользователей
review_counts AS
(SELECT username,
        COUNT(review_id)
FROM reviews
WHERE username IN (SELECT * FROM fifty_rating_users)
GROUP BY username)

SELECT AVG(count)
FROM review_counts
'''

# выполним запрос
pd.io.sql.read_sql(query, con = engine)
```

Out[13]:

	avg
0	24.333333

Читатели, поставившие более 50 оценок, пишут в среднем по 24 обзора (отзыва).

Выводы

Основываясь на изучении базы данных книжного сервиса, для нового продукта целесообразно, в первую очередь, использовать книги, изданные после 2000 года издательством *Penguin Books*, имеющие не менее 24 отзывов в базе. Особое внимание следует обратить на произведения писательско-оформительского дуэта *J.K. Rowling / Mary GrandPré*.