Project Description

You've been given a database of one of new apps for book lovers. It contains data on books, publishers, authors, and customer ratings and reviews of books. This information will be used to generate a value proposition for a new product.

Description of the data

books

Contains data on books:

- book_id (int)
- author_id (int)
- title (varchar)
- num_pages — number of pages (int)
- publication_date (datetime)
- publisher_id (id)

authors

Contains data on authors:

- author_id (int)
- author (varchar)

publishers

Contains data on publishers:

- publisher_id (int)
- publisher (varchar)

ratings

Contains data on user ratings:

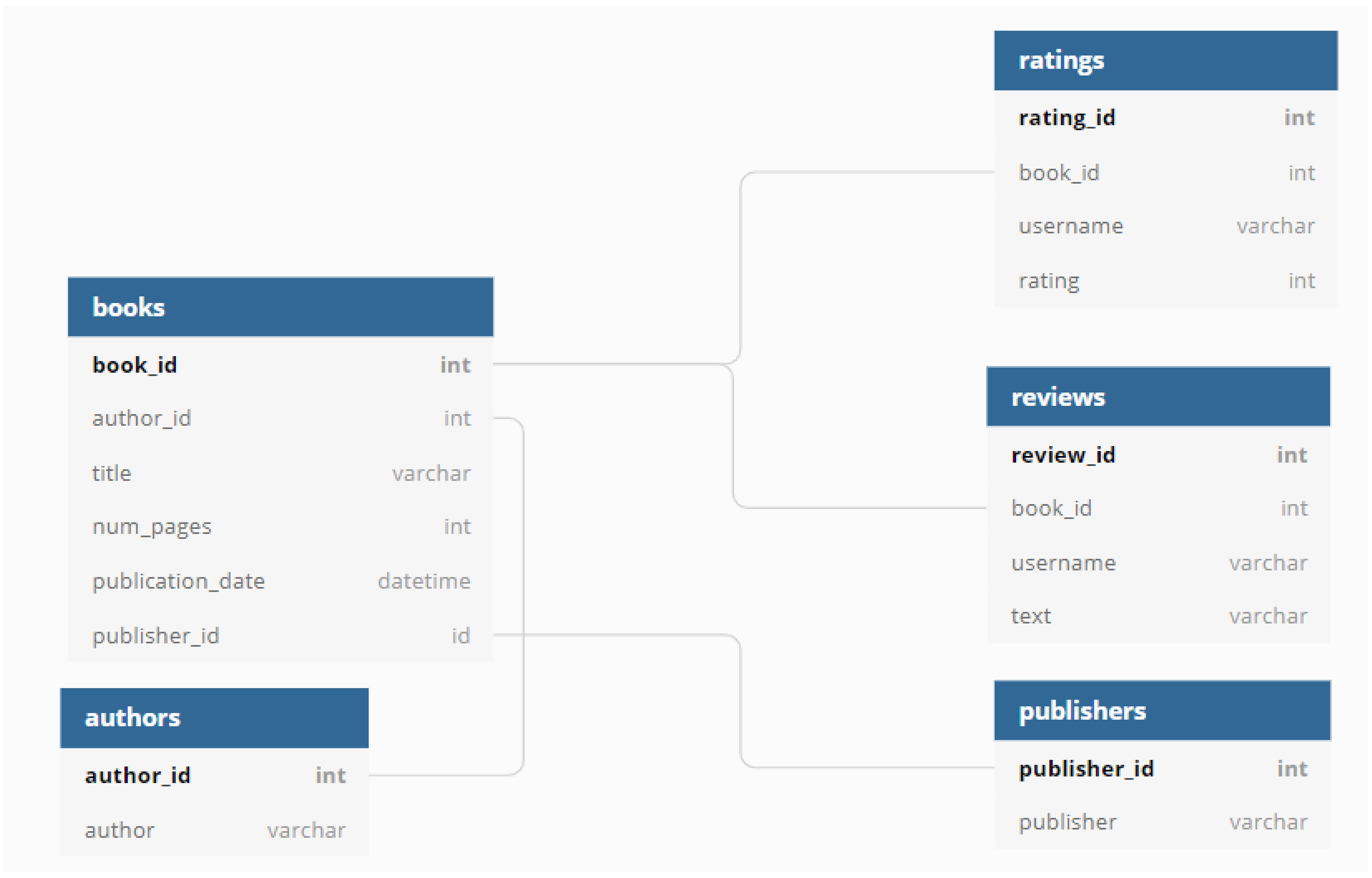
- rating_id (int)
- book_id (int)
- username — the name of the user who rated the book (varchar)
- rating (int)

reviews

Contains data on customer reviews:

- review_id (int)
- book_id (int)
- username — the name of the user who reviewed the book (varchar)
- text — the text of the review (varchar)

Data diagram



Tasks

1) Find the number of books released after January 1, 2000.

2) Find the number of user reviews and the average rating for each book.

3) Identify the publisher that has released the greatest number of books with more than 50 pages (this will help you exclude brochures and similar publications from your analysis).

4) Identify the author with the highest average book rating (look only at books with at least 50 ratings).

5) Find the average number of text reviews among users who rated more than 50 books.

Table of Contents

Description the study goals

Explore the data to define values of the parameters that will be used to generate a value proposition for a new product.

Opening the tables

First of all before doing the tasks we need to create engine:

In [1]:
```
!pip install psycopg2

# import libraries
import pandas as pd
from sqlalchemy import create_engine


db_config = {'user': 'praktikum_student',         # user name
             'pwd': 'Sdf4$2;d-d30pp', # password
             'host': 'rc1b-wcoijxj3yxfsf3fs.mdb.yandexcloud.net',
             'port': 6432,              # connection port
             'db': 'data-analyst-final-project-db'}          # the name of the data base

connection_string = 'postgresql://{}:{}@{}:{}/{}'.format(db_config['user'],
                                                         db_config['pwd'],
                                                           db_config['host'],
                                                           db_config['port'],
                                                           db_config['db'])

engine = create_engine(connection_string, connect_args={'sslmode':'require'})
```

```
Requirement already satisfied: psycopg2 in c:\programdata\anaconda3\lib\site-packages (2.8.6)
```

Let's take a look at the data:

In [2]:
```
query_books = '''SELECT * FROM books'''
books = pd.io.sql.read_sql(query_books, con = engine)
books.head()
```

Out[2]:

	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

In [3]:
```
query_authors = '''SELECT * FROM authors'''
authors = pd.io.sql.read_sql(query_authors, con = engine)
authors.head()
```

Out[3]:

	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 [4]:
```
query_publishers = '''SELECT * FROM publishers'''
publishers = pd.io.sql.read_sql(query_publishers, con = engine)
publishers.head()
```

Out[4]:

	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 [5]:
```
query_ratings = '''SELECT * FROM ratings'''
ratings = pd.io.sql.read_sql(query_ratings, con = engine)
ratings.head()
```

Out[5]:

	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

```
In [6]: query_reviews = '''SELECT * FROM reviews'''
        reviews = pd.io.sql.read_sql(query_reviews, con = engine)
        reviews.head()
```

Out[6]:

	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...

Allright. Now we can start doing the tasks.

SQL queries of the tasks and conclusions

Task 1: Find the number of books released after January 1, 2000.

```
In [7]: # SQL query
        query_task_1 = '''

        SELECT
            COUNT(title) AS number_of_books_since_2000

        FROM
            books

        WHERE publication_date > '2000-01-01' '''

        number_of_books = pd.io.sql.read_sql(query_task_1, con = engine)
        number_of_books
```

Out[7]:

	number_of_books_since_2000
0	819

Conclusion: 819 books were released after January 1, 2000.

Task 2: Find the number of user reviews and the average rating for each book.

```
In [8]: query_task_2 = '''
        SELECT
            AVG(ratings.rating) AS avg_rating,
            query.number_of_reviews AS number_of_reviews,
            query.title AS title
        FROM
            (SELECT
                 COUNT(reviews.text) AS number_of_reviews,
                 books.title AS title,
                 books.book_id AS book_id
             FROM
                 books INNER JOIN reviews ON books.book_id = reviews.book_id
             GROUP BY books.title, books.book_id) AS query INNER JOIN ratings ON ratings.book_id = query.book_id
        GROUP BY query.number_of_reviews, query.title
        ORDER BY query.number_of_reviews DESC
        '''

        number_of_user_reviews = pd.io.sql.read_sql(query_task_2, con = engine)
        number_of_user_reviews.head(5)
```

Out[8]:

	avg_rating	number_of_reviews	title
0	3.662500	7	Twilight (Twilight #1)
1	3.750000	6	The Giver (The Giver #1)
2	4.125000	6	The Hobbit or There and Back Again
3	3.825581	6	The Catcher in the Rye
4	4.081081	6	The Curious Incident of the Dog in the Night-Time

Task 3: Identify the publisher that has released the greatest number of books with more than 50 pages (this will help you exclude brochures and similar publications from your analysis).

```
In [9]: query_task_3 = '''

        SELECT
            COUNT(books_over_50_pages.title) AS number_of_books_over_50_pages,
            publishers.publisher AS publisher

        FROM
           (SELECT
                *
            FROM
                books
            WHERE
                num_pages > 50) AS books_over_50_pages INNER JOIN publishers
                ON publishers.publisher_id = books_over_50_pages.publisher_id

        GROUP BY publisher
        ORDER BY
            number_of_books_over_50_pages DESC

        '''

        rating_of_publishers = pd.io.sql.read_sql(query_task_3, con = engine)
        rating_of_publishers.head(5)
```

Out[9]:

	number_of_books_over_50_pages	publisher
0	42	Penguin Books
1	31	Vintage
2	25	Grand Central Publishing
3	24	Penguin Classics
4	19	Ballantine Books

Conclusion: Penguin Books is our leader.

Task 4: Identify the author with the highest average book rating (look only at books with at least 50 ratings).

First query will define the books with the highest average ratings:

In [10]:
```
# for a book

query_task_4 = '''

SELECT
    query2.avg_rating AS avg_rating_for_book,
    query2.title AS title,
    authors.author AS author
FROM
    (SELECT
        *
    FROM
        (SELECT
             AVG(ratings.rating) AS avg_rating,
             COUNT(ratings.rating) AS number_of_ratings,
             books.title AS title,
             books.book_id AS book_id,
             books.author_id AS author_id
         FROM
            books INNER JOIN ratings ON ratings.book_id = books.book_id
         GROUP BY
             books.title,
             books.book_id,
             books.author_id) AS query
    WHERE
        number_of_ratings > 50) AS query2 INNER JOIN authors ON query2.author_id = authors.author_id
GROUP BY
    author,
    query2.avg_rating,
    query2.title
ORDER BY
    query2.avg_rating DESC
'''

rating_of_publishers = pd.io.sql.read_sql(query_task_4, con = engine)
rating_of_publishers
```

Out[10]:

	avg_rating_for_book	title	author
0	4.414634	Harry Potter and the Prisoner of Azkaban (Harr...	J.K. Rowling/Mary GrandPré
1	4.391892	The Fellowship of the Ring (The Lord of the Ri...	J.R.R. Tolkien
2	4.287500	Harry Potter and the Chamber of Secrets (Harry...	J.K. Rowling/Mary GrandPré
3	4.264151	The Book Thief	Markus Zusak/Cao Xuân Việt Khương
4	4.246575	Harry Potter and the Half-Blood Prince (Harry ...	J.K. Rowling/Mary GrandPré
5	4.192308	Little Women	Louisa May Alcott
6	4.186667	Harry Potter and the Order of the Phoenix (Har...	J.K. Rowling/Mary GrandPré
7	4.125000	The Hobbit or There and Back Again	J.R.R. Tolkien
8	4.080645	The Lightning Thief (Percy Jackson and the Oly...	Rick Riordan
9	3.901408	Lord of the Flies	William Golding
10	3.830508	The Da Vinci Code (Robert Langdon #2)	Dan Brown
11	3.825581	The Catcher in the Rye	J.D. Salinger
12	3.789474	The Alchemist	Paulo Coelho/Alan R. Clarke/Özdemir İnce
13	3.787879	Romeo and Juliet	William Shakespeare/Paul Werstine/Barbara A. M...
14	3.750000	The Giver (The Giver #1)	Lois Lowry
15	3.729730	Animal Farm	George Orwell/Boris Grabnar/Peter Škerl
16	3.678571	Angels & Demons (Robert Langdon #1)	Dan Brown
17	3.662500	Twilight (Twilight #1)	Stephenie Meyer
18	3.622951	Of Mice and Men	John Steinbeck

And this one takes average ratings of all books of each author and finds average rating for authors:

```
In [11]: query_task_4 = '''

SELECT
    AVG(avg_rating_for_book) AS avg_rating_for_author,
    author
FROM
    (SELECT
         query2.avg_rating AS avg_rating_for_book,
         query2.title AS title,
         authors.author AS author
     FROM
         (SELECT
             *
         FROM
             (SELECT
                  AVG(ratings.rating) AS avg_rating,
                  COUNT(ratings.rating) AS number_of_ratings,
                  books.title AS title,
                  books.book_id AS book_id,
                  books.author_id AS author_id
              FROM
                 books INNER JOIN ratings ON ratings.book_id = books.book_id
              GROUP BY
                  books.title,
                  books.book_id,
                  books.author_id) AS query
         WHERE
             number_of_ratings > 50) AS query2 INNER JOIN authors ON query2.author_id = authors.author_id
     GROUP BY
         author,
         query2.avg_rating,
         query2.title
     ORDER BY
         query2.avg_rating DESC) AS query3
GROUP BY
    author
ORDER BY
    avg_rating_for_author DESC

'''

rating_of_publishers = pd.io.sql.read_sql(query_task_4, con = engine)
rating_of_publishers
```

Out[11]:

	avg_rating_for_author	author
0	4.283844	J.K. Rowling/Mary GrandPré
1	4.264151	Markus Zusak/Cao Xuân Việt Khương
2	4.258446	J.R.R. Tolkien
3	4.192308	Louisa May Alcott
4	4.080645	Rick Riordan
5	3.901408	William Golding
6	3.825581	J.D. Salinger
7	3.789474	Paulo Coelho/Alan R. Clarke/Özdemir İnce
8	3.787879	William Shakespeare/Paul Werstine/Barbara A. M...
9	3.754540	Dan Brown
10	3.750000	Lois Lowry
11	3.729730	George Orwell/Boris Grabnar/Peter Škerl
12	3.662500	Stephenie Meyer
13	3.622951	John Steinbeck

Conclusion: J.K. Rowling is No1

Task 5: Find the average number of text reviews among users who rated more than 50 books.

In [12]:
```
query_task_5 = '''

SELECT
    AVG(CAST(query2.number_of_text_reviews_per_user AS real)) AS avg_number_of_text_reviews_per_user
FROM
    (SELECT
         reviews.username AS username,
         COUNT(reviews.text) AS number_of_text_reviews_per_user
     FROM
         books INNER JOIN reviews ON reviews.book_id = books.book_id

     WHERE username IN (SELECT
                            query1.username AS user_name
                        FROM
                            (SELECT
                                 COUNT(books.title) AS number_of_rated_books_per_user,
                                 ratings.username AS username
                             FROM
                                 books INNER JOIN ratings ON ratings.book_id = books.book_id
                             GROUP BY
                                 ratings.username) AS query1
                        WHERE number_of_rated_books_per_user > 50)
     GROUP BY reviews.username) AS query2

'''
rating_of_publishers = pd.io.sql.read_sql(query_task_5, con = engine)
rating_of_publishers
```

Out[12]:

	avg_number_of_text_reviews_per_user
0	24.333333

Conclusion: The average number of text reviews among users who rated more than 50 books is 24

Requirements

In [13]:
```
pip freeze > requirements.txt
```

```
Note: you may need to restart the kernel to use updated packages.
```