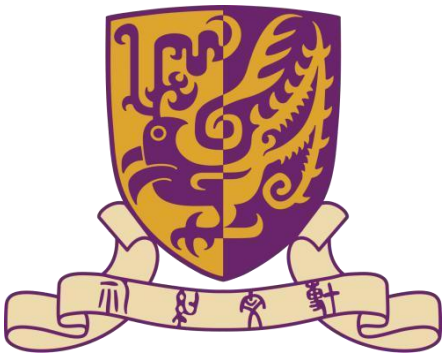

THE CHINESE UNIVERSITY OF HONG KONG, SHENZHEN

# Report for
# DOUBAN DATABASE

Quan Liu

Xin Shu

Yanglin Zhang

Muhan Lin

Date: May 1, 2021

# Contents

# 1 INTRODUCTION

Man is a unity of natural existence, social existence, and spiritual existence. Human needs are multi-faceted and multi-layered. Among them, material needs are basic needs, first and limited. Spiritual and cultural needs are advanced needs, second and unlimited. The effective satisfaction of material needs will promote the transformation of spiritual and cultural needs from non-dominant needs to dominant needs.

With the rapid economic development, people's material life has been satisfied, and people are paying more and more attention to cultural life. As a non-profit cultural industry analysis organization, we want to understand people's preferences for cultural types by studying movies and books data over the past 60 years. Thus, it is better to help people enrich their cultural life. At the same time, cultural industry companies can also get some suggestions from our results.

We will use python Spider to obtain valuable data from the website, store it in a MySQL database. Then we will use these data to draw some realistic conclusions and implement some simple functions. Finally, we will show the results through a website.

# 2 DESIGN OF DATABASE

To design the database, we need to determine the ER diagram and schema. The original tables contain a large number of functional dependence from non-key attributes to other attributes. To construct it better, we decide to design the schema and do normalization firstly and then convert it to ER diagram.

To make schema be in 1NF, every attribute that is not atomic is rearranged. In general, those attributes correspond to rules such as "a movie participation can have multi-professions", "a book is written by several authors". However, the new schema often contains non-key attributes that are not functionally dependent on PK. In addition, there are substantial redundancies caused by repeated information.

To make schema be in 2NF, the invalid tables are broken into several sub-tables. After testing, all the decomposition is lossless. In our case, most tables in this step are automatically in 3NF and BCNF.

In the end, we convert schema satisfying BCNF to the ER diagram, which is shown in the appendix at the end of this report. All tables are divided, according to their functions, into three parts: Book, Movie and User. Those three parts are also the main objects in this project.

# 3 IMPLEMENTATION

The basic idea of implementation has two steps: the first one is to using web crawler to get raw data then create a database; the second one is to using DJANGO to connect the database and HTML.

## 3.1 WEB CRAWLER

We following the steps shown below to crawl across the Douban website to get raw data. The invalid data are filtered in this step, like emoji or text unknown by utf-8.

- Register several DouBan accounts
- Store the username and password in a dictionary, then use urllib.parse to get the encoded dictionary
- Simulate login using the encoded dictionary and get cookies using the request library of Python3
- Prepare some proxies that can be used to avoid IP blocking due to crawlers
- Prepare some headers, randomly select proxy and header combinations
- Use get() in request library to get HTML data. The parameters in get() consist of the URL, proxy and header
- Use the bs4 library to parse the HTML data
- Get the HTML tags of the content to be crawled by examining the web page, and then use the tags to get the required data

### 3.1.1 PROBLEM-DEALING

To deal with 2 million data and import them into the database, the following problems are fixed:

- The format of date-type data are not unified. E.g. there are ”xxxx-xx-xx” ”xxxx/xx/xx”. To fix to, we use python to reset all date-type data be like ”xxxx-xx-xx”
- There are ”0000-00-00” in the data. It represents ”null” for this place but is not acceptable by MySQL when importing. To fix it, we set this attribute with the default value ”null”.
- The speed of the built-in function ”table data import wizard” in MySQL is too slow to import data. To fix it, we use the SQL command ”load data infile” to accelerate the rate from 10 thousand per hour to 100 thousand per second.
- Exclude ”line break” for search data in MySQL more conveniently.

## 3.2 MYSQL

There are two steps to construct the database: the first one is to create schema according to the design; the second one is to import data into the database.

Creating schema and tables is easy to achieve. One of the details is to set the character format as utf-8 to make schema compatible with Chinese.

### 3.2.1 INDEX

To make searching more efficient, indices are used. The following list shows details of the indices.

Index using Hash: following three attributes have been used frequently, so using hash can improve efficiency and save time.

- book_id
- movie_id
- user_id

Index using BTree: used for attributes whose using frequency is less than that of the attributes mentioned in the "Hash" part but is still high.

- all PKs except attributes mentioned in Hash.
- movie_person_address
- book_person_address
- movie_person_language
- book_person_language

## 3.3 HTML

To make the database visible and user-friendly, our team designed a web page for data search. The development tools include html, css and JavaScript.The home page consists of navigation bars and a content part.

The navigation bars show six classes of the data analysis functions of this database:

- 1. Movie Popularity Info
- 2. Excellent Movie Practitioner Info
- 3. Popular Book Info

- 4. Excellent Publication Practitioner Info
- 5. Self-defined Query
- 6. Recommendation

The first four classes contain all standard query functions. After a user clicks one of the first four classes in the navigation part, the content part lists the functions in this class. When it comes to a class containing too many functions, these functions are divided further into groups on the content page with subtitles for convenient search. There are brief description and a button "Check the Result" for each function. These functions are realized by sample Mysql queries written by our team. To fetch the results of these standard queries, users do not need to write Mysql instructions. They only need to click buttons on the web page. As for the result display, if a result checking button for the ith function is clicked, the JavaScript function ***jump(i)*** will be triggered and redirect the website address to *http://127.0.0.1/8000/searchi/*, where the corresponding data will be shown.

If a user wants to enter customized queries, he or she can click "Self-defined Query" in the navigation part. The content part will show a input pane and a submission button. The user can type in one Mysql query here and click the button to execute it.

If a user clicks "Recommendation" in the navigation part, the content part will show a input pane and prompt the user to enter their favorite book names or movie names. After the user clicks the button to submit the input, the html interface will send this request to the backend to get similar recommendations according to this given name.

When implementing the forms in class 5 and 6, we declared the action (where the web directory will change to after submission) and the method (how the backend can receive the submitted request). For example, if the form is declared as ⟨***form action="/search/" method="get"***⟩, the web page will jumped to *http://127.0.0.1/8000/search/* automatically after submission button is clicked, and the backend should use ***get*** method to receive the request. We also assigned each input pane a name for later request retrieve.

## 3.4 DJANGO

To receive requests from frontend and render responses on the web page, our team adopted Django, a python package, to connect the user interface and the database.

A URL was implemented in Django. Website directories and functions are stored in pairs as rules in url.py. When our website is visited, the URL will check the rules to execute the python function corresponding to the visited website directory.

If a request is sent by a form, the website address will redirect to a given address. Then the URL checks the rules and execute the python function corresponding to this new address. In this function, we first used ***request.GET(name)*** method to receive the request from the input

pane with the given name. Then the program can execute necessary operations to respond to the request. For example, we can connect to our Douban Database and execute some queries.

The communication between python and Mysql database is completed with pymysql package. First the python program connects to Mysql database with ***pymysql.connect*** and get the cursor. Then operations can be executed with ***cursor.execute*** and committed with ***db.commit*** or rolled back with ***db.rollback***.The result is fetched with ***cursor.fetchall*** as a two-dimensional tuple. Our program then transforms the tuple into a string.

The function carried out by the URL finally shows the execution result at its corresponding website address with ***HttpResponse(result)***. That is how we showed users the response to their request.

# 4 RESULTS

We have realized the interaction between users, web pages and the database. The user should first go to *http://127.0.0.1/8000/index/*, which is the address of our home page. The home page is shown in Figure 1. We can see the navigation bars on the left and the content page on the right. Now the selected function class is 1.

Click the third button to get ”Popular movie Types from audiences’ view by scores”, and the result is shown as Figure 2 at *http://127.0.0.1/8000/search3/*.

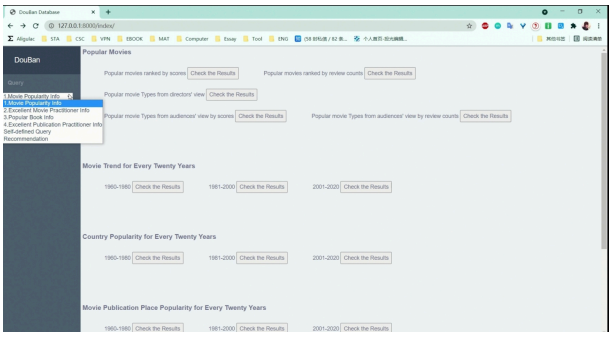


**FIGURE 1:** THE HOME PAGE

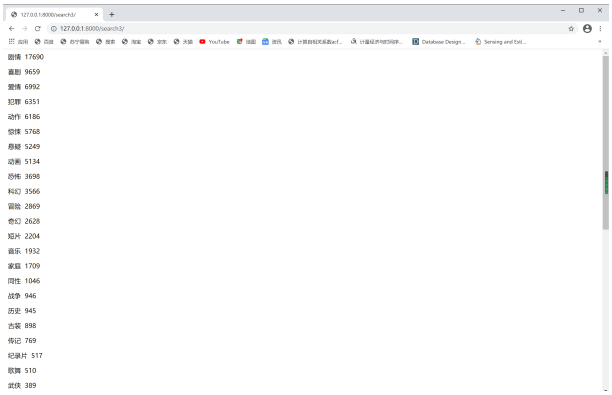

**FIGURE 2:** THE RESULT OF THE THIRD SEARCH FUNCTION

Then choose the ”Self-defined Query” class, and enter the query as shown by Figure 3. The execution result is shown as Figure 4. That implies that our database is real-time.

Choose the ”Recommendation” class in the navigation part. Enter book names as shown in Figure 5. The recommendations are shown as Figure 6.

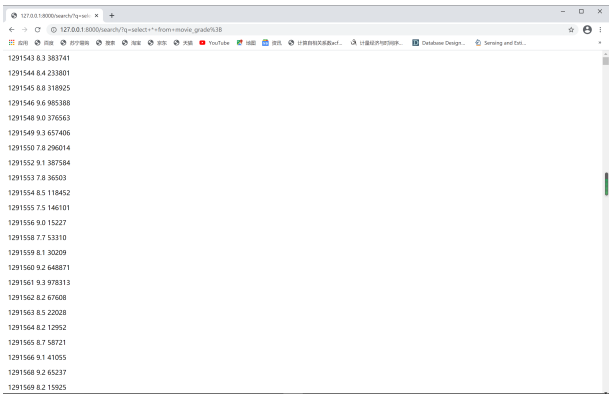

**FIGURE 3:** THE RESULT OF THE SELF-DEFINED QUERY 'SELECT * FROM MOVIE_GRADE;'

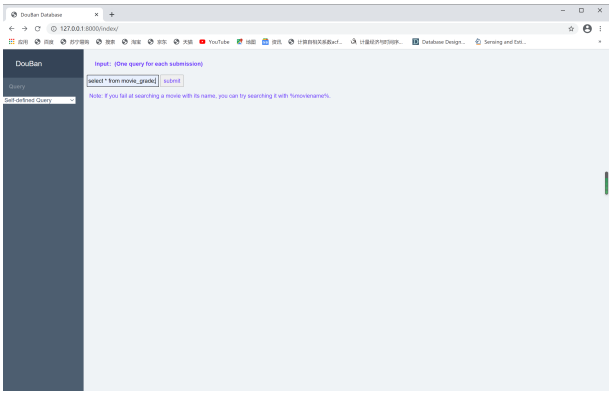

**FIGURE 4:** THE RESULT OF THE THIRD SEARCH FUNCTION

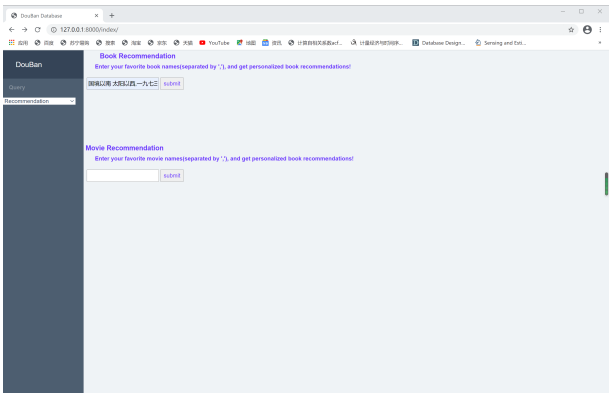

**FIGURE 5:** REQUEST RECOMMENDATION

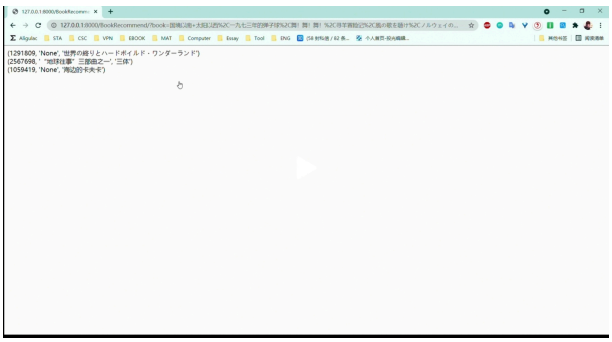

**FIGURE 6:** RECOMMENDATIONS

# 5 DATA ANALYSIS

## 5.1 BASIC FUNCTIONS AND RESULTS

We wrote 24 queries to gain information of movies and books. Detailed query information is shown in appendix B.

In general, we have implemented several basic functions, which are:

- movie part:
  - movie rank
  - movie country
  - movie duration
  - personal information of commercially valuable directors, actors and writers
- book part:
  - book rank

- district of readers
- personal information of commercially valuable writers, publishers and translators

## 5.2 DEMO MYSQL QUERY

```
## 23. common books read by shanghai people
#
drop table if exists books_shanghai_read;
create table books_shanghai_read as
select origin_name, subtitle, count(book_name.book_id) as num_of_people_read
from book_name join user_book on book_name.book_id=user_book.book_id
join user_info on user_book.user_id=user_info.user_id
where address = '上海'
group by book_name.book_id
order by count(book_name.book_id) desc;
select * from books_shanghai_read;
```

**FIGURE 7:** DEMO MYSQL QUERY

This query is to extract popular books read by Shanghai people.

It is constructed by joining three tables (book_name, user_book, user_info) on their common attributes, grouping by book name and order by the number of readers.

## 5.3 INTERPRETATIONS AND SUGGESTIONS

After data visualization, we get several descriptive conclusions, and based on these, we give out several useful suggestions in both the cultural and commercial industry.

### 5.3.1 OUTCOME 1

The first outcome is about movie's popularity versus quality. From the graph, we can see that disaster has high number of movie review count with a rather low movie score.

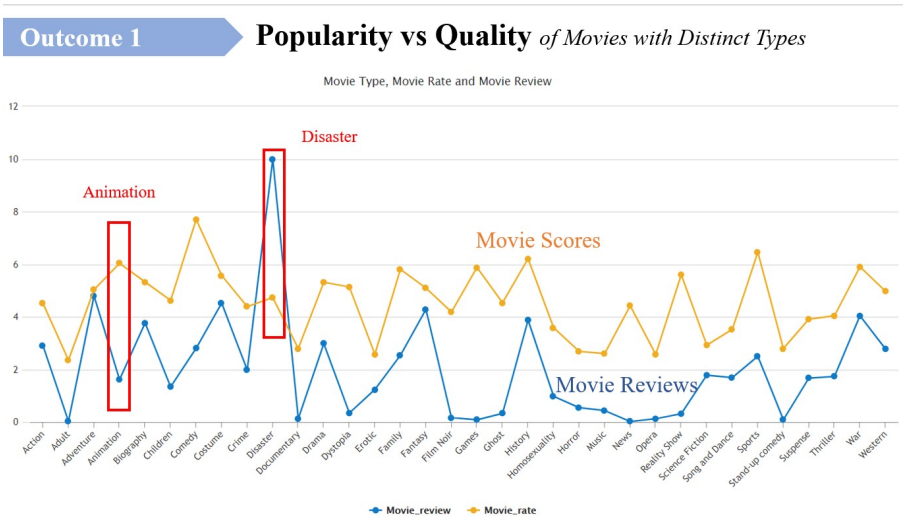


**FIGURE 8:** OUTCOME 1

At the same time, Animation has a high movie score with surprisingly low number of review counts.

We can see that not only for these two movie types, other movie types also have this problem. We think that the phenomena maybe due to some complicated reasons. Therefore, we suggest people working in cultural industry should realize the phenomena and gain some insight into it.

### 5.3.2 OUTCOME 2

The next outcome is about the movie language popularity trend every twenty years from 1960 to 2020.

We can see that for the sixty years, English and Japanese has occupied stable percentage of the movie market.

While in 1980s, Cantonese became popular and in the last twenty years, Chinese and Korean became more popular.

This fits the historical fact that China and Korea have developed fast in the last twenty years, not only economically, but also in the aspect of cultural export.

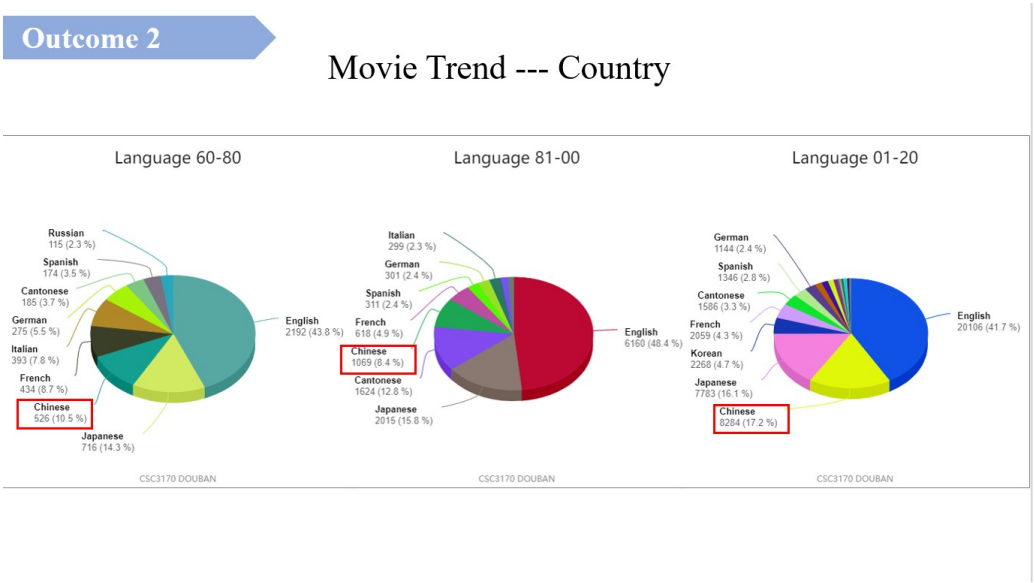


**FIGURE 9:** OUTCOME 2

### 5.3.3 OUTCOME 3

Third outcome is about the relationship between district and books.

We get the information of popular books read by Beijing, Shanghai and Guangzhou people. We can see that people in different districts have different book preferences.

Therefore, we suggest bookstores in different districts should have different book reprinting policies according to local people's preferences.

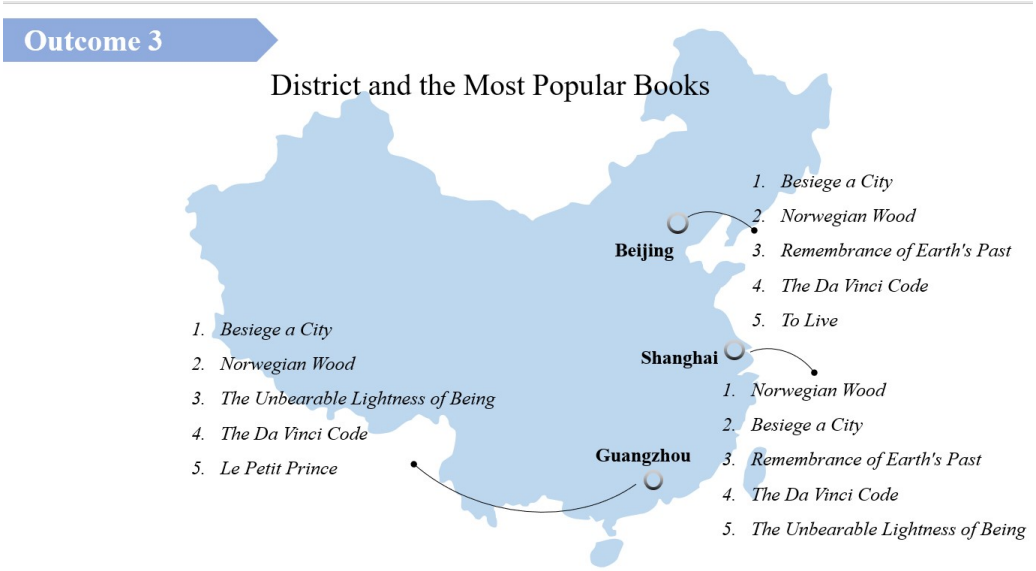


**FIGURE 10:** OUTCOME 3

## 5.4 RECOMMENDATION

We use the association rule-based recommendation algorithm – Apriori to recommend books and movies to users. First of all, import all the books or movies that the user has seen into the program using read.transactions(). Then, use apriori() in arules library to get the association rules. Then, get the right number of rules by adjusting the minimum support, the minimum confidence and the minimum rule length. After that, use is.redundant() to remove redundant rules. The rules for de-duplication are: rule1 X->Y with confidence cf1, rule2 X' -> Y with confidence cf2 where X' is a subset of X, rule1 is said to be redundant if rule2 has a higher confidence than rule1 i.e cf2 > cf1 (where X' is a subset of X). In addition, use plot() in arulesViz library to get a distribution plot of rules' support and confidence (see Figure 11 and 12). Finally, export the left and right ends of the rule to a csv file.

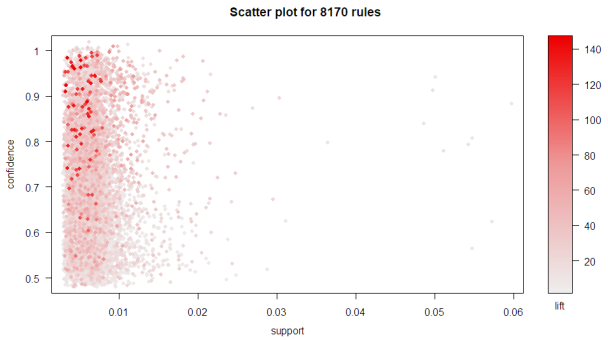


**FIGURE 11:** REQUEST RECOMMENDATION

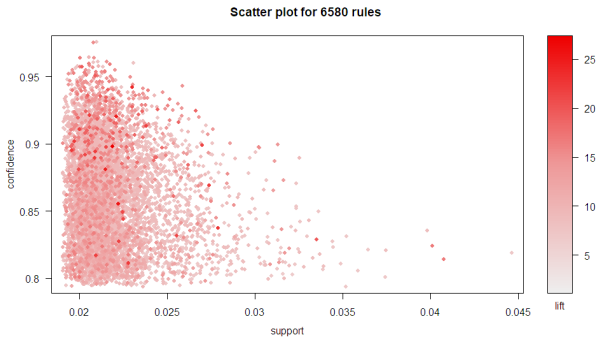


**FIGURE 12:** RECOMMENDATIONS

We implement the Apriori algorithm by following steps. Firstly, read the rules file using Python. Then, store the left and right sides of the rule into two lists. The input side is a list of

book or movie IDs, get all subsets of this list, and then query in the first list to get the recommended books if available and vice versa. Finally, the recommended books will be removed from the duplicates to get the final recommendation.

# 6 CONCLUSION

We used a large amount of DouBan data to do some business value analysis as well as to study cultural industry trends. The structure of our database satisfies the BCNF form to make the redundancy as small as possible, but at the same time retains the dependency, making the database have a very clear structure.

As a result, we use a user-friendly website that makes it easy for users to query the results of our research. We have a lot of data analysis outcomes. First, we studied the popularity of different types of movies and their quality. Second, we analyzed the film production in different countries over the past six decades and analyzed the development of different countries. Third, we analyze people's preferences for books and movies in different regions so that publishers can adjust their strategies. Finally, we use the Apriori algorithm to achieve personalized recommendations for users.

The results of these analyses will help people better understand the trends in the cultural industry over the past six decades, as well as help cultural industry companies better understand the preferences of people of different regions and ages for movies and books.

# 7 SELF-EVALUATION

This part evaluates this project from advantages and disadvantages and gives suggestions for the future.

From implementations and functions, the project has the following advantages:

- Stores a huge amount of data that has been filtered according to designed rules.
- Mines data thoroughly like classification, association.
- Is user-friendly like webs that are easy to use
- Has various functions such as providing the characteristics of users and their preferences, and recommendations to help users find the books and movies in which they have interest in a short time.

To further improve our system, we analyzed our disadvantages:

- Has a relatively low reaction speed. The main reason is that running MySQL codes waste lots of time for its properties. Thus we propose that replacing SQL with Python would fix this problem and get better performance.

In the further, except for overcoming the drawbacks, we are going to analyze and compare the data from other platforms to get a more comprehensive realization of people's cultural life.

# 8 CONTRIBUTION

Everyone contributes 25% of project.

# References

[1] Douban.com. 2021. Douban. [online] Available at:<https://www.douban.com/> [Accessed 1 May 2021]

[2] A. Silberschatz, H. Korth, and S. Sudarshan, Database System Concepts, 7th edition, McGraw-Hill (2020)

[3] R. Elmasri and S. B. Navathe, Fundamentals of Database Systems, 7th edition, Addison-Wesley (2016)

[4] Tan, P., Steinbach, M., Kumar, V. and Karpatne, A., 2019. Introduction to Data Mining, Global Edition. 2nd ed. Harlow, United Kingdom: Pearson Education Limited, pp.451-510.

[5] Zaki, M. J. (2004). Mining non-redundant association rules. Data mining and knowledge discovery, 9(3), 223-248.

# APPENDIX A: ER DIAGRAM

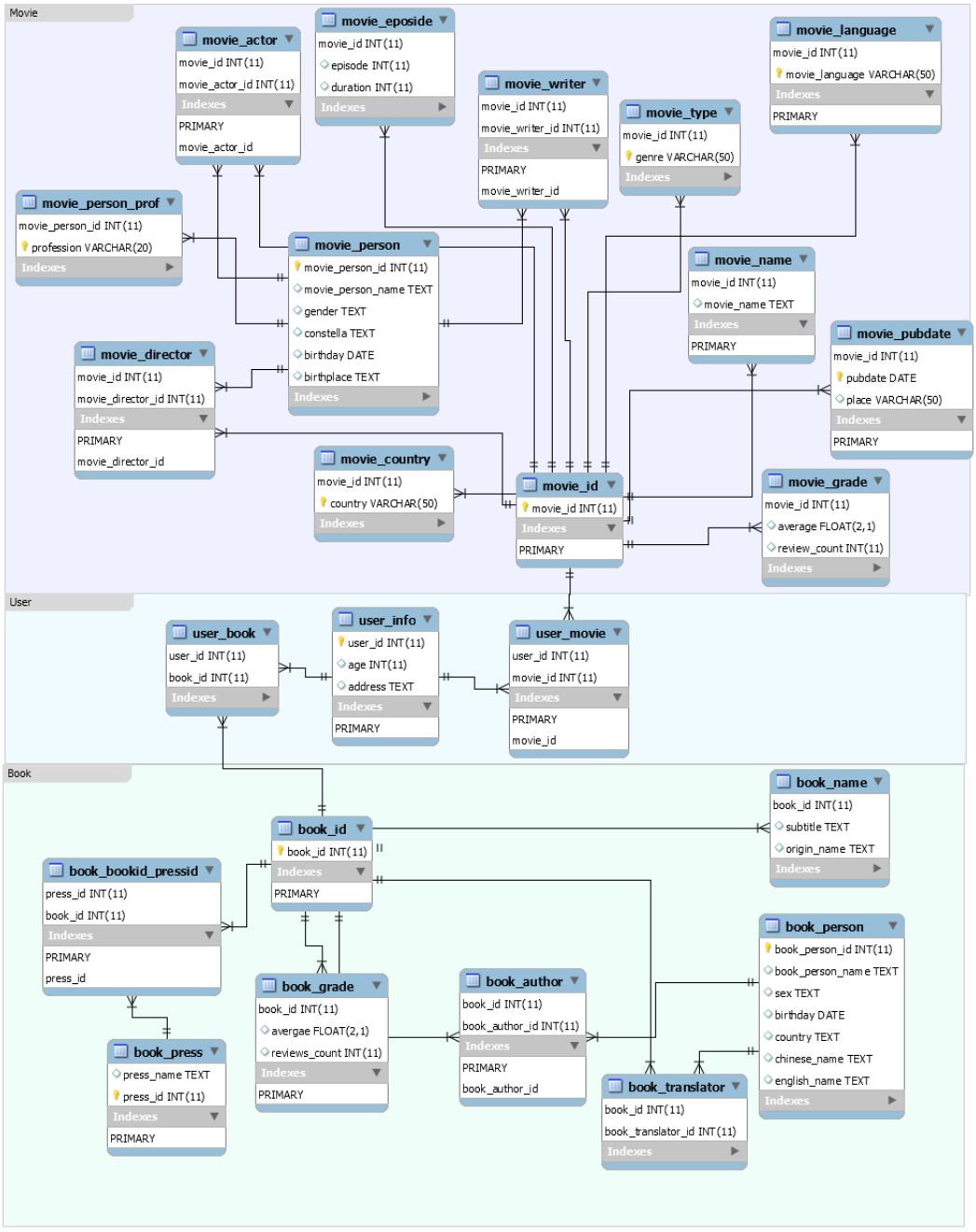


**FIGURE 13:** ER DIAGRAM

# APPENDIX B: DETAILED FUNCTIONS

For movie:

1. movie popularity ranked by credits
2. movie popularity ranked by the number of review counts

3. movie type popularity from director's view
4. movie type popularity from audience's view by credit
5. movie type popularity from audience's view by the number of review counts
6. the trend of movie for every twenty years, lasting 60 years:
    - language popularity from 1960-1980,1981-2000,2001-2020.
    - country popularity from 1960-1980,1981-2000,2001-2020.
    - publication place popularity from 1960-1980,1981-2000,2001-2020.
7. personal information of commercial valuable directors (directing most in top 1000 ranked movies)
8. personal information of commercial valuable actors (acting most in top 1000 ranked movies)
9. personal information of commercial valuable writers (writing most in top 1000 ranked movies)
10. movie duration information
11. information of all-around movie persons
12. relationship between age and movie type:
    - average audiences' age for every movie type
    - favourite movie type for different group of ages

For Book:

1. book popularity ranked by credits and the number of review counts
2. author's popularity (writing the most in top 1000 books)
3. publisher's rank by the number of books published
4. publisher popularity (publishing most in top 1000 books)
5. average book price published of each publisher
6. average book price of every writer
7. number of writers connected to each publisher
8. most valuable writer for each publisher (write most number of books out of the books published by that publisher)

9. the rank of publishing year (ranked by the number of books published for a certain year)
10. the rank of writers' productivity (total page number)
11. the rank of translators' popularity (number of books translated)
12. relationship between districts and popular books:
    - popular books read by Shanghai people
    - popular books read by Beijing people
    - popular books read by Guangzhou people