24-01

데이터베이스(01)

Implementing a Simple Database Application

-Term project-

2171056 강승연

목차

1-A.개발 환경

Python 3.12.2

MySQL Server 8.0.37

Window 11

1-B.사용한 외부 라이브러리

Mysql.conector에서 DB 연결을 위해 connect, 에러 처리를 위해 Error를 사용했다.

Enum은 출력 형식 지정 시 헤더 종류를 선택할 때 비이상적 접근을 차단할 목적으로 사용했다.

Pandas는 csv 처리를 용이하게 하기 위해 사용했다.

Os는 data.csv가 pwd를 가져오기 위해 사용했다.

```
from mysql.connector import connect, Error
from enum import Enum
import pandas as pd
import os
```

1-C.요구사항

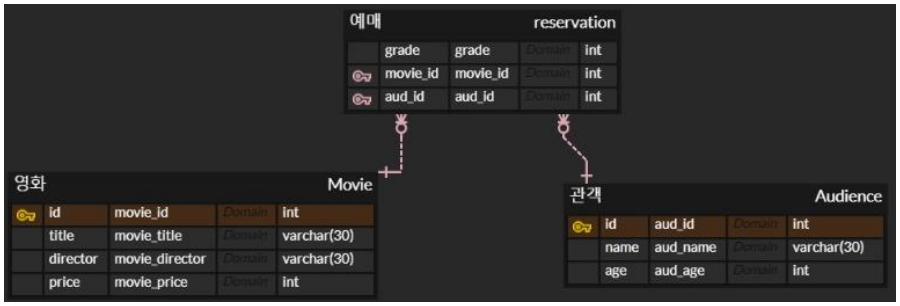
해당 과제는 Python 과 MySQL 을 이용하여 영화 예매 시뮬레이션 어플리케이션을 만드는 것을 목적으로 한다.

요구된 기능은 다음과 같다.

주어진 데이터로 데이터베이스를 초기화 할 수 있어야 한다.
영화 데이터 및 고객 데이터를 출력할 수 있어야 한다.
영화 데이터 및 고객 데이터를 추가/삭제할 수 있어야 한다.
고객이 영화를 예매할 수 있어야 한다.
고객이 영화에 평점을 부여할 수 있어야 한다.
영화 별로 영화를 예매한 고객의 명단 및 평점 정보를 출력할 수 있어야 한다.
고객 별로 예매한 영화의 정보 및 부여한 평점 정보를 출력할 수 있어야 한다.
프로그램 종료 및 데이터베이스 리셋과 생성이 가능해야 한다.

2. 데이터베이스 스키마 설계

해당 요구사항을 바탕으로 구상한 테이블은 다음과 같다.



예매를 Relation으로 영화와 관객 사이의 Many to Many 관계를 나타내었다. (영화-예매, 관객-예매는 One to Many)

영화는 id, title, director, price로 스키마가 구성되어 있다. 영화의 id에 나머지 영화의 모든 속성이 종속되어 functional dependency를 갖는다고 보았기 때문이다. 마찬가지의 의미로 관객은 id, name, age를 스키마로 가졌다.

예매는 movie의 primary key와 audience의 primary key를 가져야 하므로 movie id와 audience id를 키로 갖는다.

평점을 예매 둘 다 영화 id와 관객 id를 키로 가져 분리할까 고민했지만 현실을 가장 직관적으로 나타내기 위해 예매와 평점 사이의 관계가 존재해야 한다고 생각했다. 그래서 이 둘을 분리한 뒤 연결하는 것을 고민했으나, 해당 과제를 최대한 간단하게 구현하기 위해 예매에 평점을 넣어두었다. (예매한 영화에만 평점을 부여할 수 있기 때문이다. 뿐만 아니라 영화 출력 시 grade의 평균을 출력해야 하는 만큼 접근에 있어 성능을 고려하고 싶었다. 유연성 및 확장성을 고려할 필요가 없는 과제에서 다른 규칙을 만족하는 한 굳이 분리할 필요가 없다고 판단했다.)

그리고 기타 제약 사항들을 고려하여 제약을 걸 수 있는 한 최대한 데이터베이스 자체에 제약을 걸었다. CREATE에 사용된 SQL문은 다음과 같다.

```
CREATE TABLE movie (
  id INTEGER AUTO_INCREMENT PRIMARY KEY,
  title VARCHAR(80) NOT NULL,
  director VARCHAR(35) NOT NULL,
  price INTEGER NOT NULL,
  CONSTRAINT unique_title UNIQUE (title),
```

```
    CONSTRAINT price_range CHECK (price BETWEEN 0 AND 100000)
);

CREATE TABLE audience (
    id INTEGER AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(30) NOT NULL,
    age INTEGER NOT NULL,
    CONSTRAINT age_range CHECK (age BETWEEN 12 AND 110),
    CONSTRAINT unique_name_age UNIQUE (name, age)
);

CREATE TABLE reservation (
    mov_id INTEGER,
    aud_id INTEGER,
    grade INTEGER,
    PRIMARY KEY (mov_id, aud_id),
    CONSTRAINT grade_range CHECK (grade BETWEEN 1 AND 5),
    CONSTRAINT fk_mov FOREIGN KEY (mov_id) REFERENCES movie(id) ON DELETE CASCADE,
    CONSTRAINT fk_aud FOREIGN KEY (aud_id) REFERENCES audience(id) ON DELETE CASCADE
);
```

중복(UNIQUE) 제약과 범위가 제약되는 경우, 그리고 Foreign Key를 데이터베이스 상에서 제약했고, 이외에는 프로그램 상에서 제약했다. 그리고 오류 처리는 해당 제약조건에 이름을 붙여 해당 이름의 문자열을 substring()으로 찾는 방법을 이용했다.

3-A.전체 프로그램 구조

전체 프로그램의 구조는 간단하다.

```
project/
 ├──── run.py
 └──── query.py
```

Run.py는 메인 프로그램으로 실제 프로그램의 실행과 관련된 함수, 그리고 main함수가 존재한다.

Query.py는 메인 프로그램을 위한 DB 조작 모듈과 SQL문들을 미리 작성해 두었다.

3-B.핵심 모듈 - Query.py

Query.py는 아래와 같은 구성을 갖는다.

1. DB 조작 모듈 Repository
2. 출력 형식 지정 함수 print_table과 사전에 출력 형식을 지정해둔 Enum
3. SQL문

Import

```
from mysql.connector import connect, Error
from enum import Enum
import pandas as pd
```

1. Repository class는 메인 프로그램을 위한 DB 조작 모듈이다.

DB와 연결하고 SQL문을 실행할 목적으로 만든 모듈이다. 실제로 메인 프로그램에서는 이 클래스의 인스턴스를 통해 모든 SQL문을 실행한다.

```
class Repository:
    # 데이터베이스와의 연결 생성
    def __init__(self):
        db_name ='movie_reserving'
        self.connection = connect(
            host ='127.0.0.1',
            user ='DB',
            password ='0309',
            db =db_name,
            charset ='utf8'
        )
        if not self.connection.is_connected() :
            print("[ERROR]MySQL Server 와의 연결에 문제가 있습니다. ")
            exit()

    # sql 문을 실행시키고 결과와 에러를 반환한다.
    def execute_sql(self, sql, commit = True):
        if not self.connection.is_connected():
            self.__init__()

        cursor = self.connection.cursor()

        try :
            cursor.execute(sql)
            result = cursor.fetchall()
            if commit == True :
```

```
                self.connection.commit()
            return result
        except Error as e:
            self.connection.rollback()
            return e.msg
        finally:
            cursor.close()

    # data frame 을 input 으로, 해당 data frame 을 DB 에 Insert 한다.
    def init_with_csv(self, table_header, df):
        data = ""
        for x in df.to_records(index=False) :
            t = str(tuple(x))
            data += ',' + t
        data = data[1:]
        sql = f"INSERT INTO {table_header} VALUES {data};"
        self.execute_sql(sql)

    # connection 을 종료한다.
    def close_connection(self):
        self.connection.close()
```

2. 출력 형식 지정 함수

형식에 맞게 출력해야 하는 경우에는 함수를 미리 만들어 사용했다.

헤더를 출력해야 해 사전에 헤더의 양식도 정의해 두었다. 헤더를 숫자 자료형 혹은 문자열로 선택할 시 잘못된 접근이 생길 가능성이 있어 enum으로 정의하여 사용했다.

아쉬운 점은 data와 header 형식이 일치하지 않을 시 오류가 발생한다는 점이다.

```
# 형식에 맞게 출력한다.
def print_table(data, header_query):
    # 중략
# print_table 출력 시 사용되는 헤더 목록이다.
header_format_movie = [5, 80, 35, 10, 5, 5]
header_select_movie = ["id", "title", "director", "price", "reservation",
"avg. rating"]
# 중략

# 헤더를 Dictionary 로 만들어 Enum 으로 접근하기 쉽게 만든다.
header = {
    1 : {"format": header_format_movie, "element": header_select_movie},
    2 : {"format": header_format_audience, "element": header_select_audience},
    3 : {"format": header_format_reserve_aud, "element":
header_select_reservation_by_audience},
    4 : {"format": header_format_reserve_mov, "element":
header_select_reservation_by_movie}
```

```
}

# Enum 으로 Header 를 접근할 수 있게 하여 오접근을 막는다.
class header_type(Enum):
    movie = 1
    audience = 2
    reserve_aud = 3
    reserve_mov = 4
```

3. SQL문

SQL을 사전에 문자열과 함수로 작성했다. 해당 SQL문을 Repository 모듈로 실행시키면 메인 프로그램에서는 SQL문을 실제 DB에 실행시킨 것과 동일한 결과를 얻는다.

Input이 없는 경우 단순 문자열로, Input이 있는 경우 함수로 만들어 문자열을 반환하도록 구성했다.

```
### Input 없는 SQL 문###

sql_drop_db = """DROP TABLE reservation;
DROP TABLE movie;
DROP TABLE audience;
"""

#중략
### Input 있는 SQL 문###
def sql_insert_movie(title, director, price):
    return f"INSERT INTO movie(title, director, price) VALUES ('{title}',
'{director}', {price});"

def sql_insert_audience(name, age):
    return f"INSERT INTO audience(name, age) VALUES ('{name}', {age});"

#중략
```

3-C.메인 프로그램 - run.py

Run.py에서는 query.py를 이용해 실제 요구사항을 구현하고, 이를 main에서 선택해 실행 가능하도록 만들었다.

Import

```
import os
import query
import pandas as pd
```

0. 성공/오류 여부 확인 함수

```
# 정상작동 확인 및 성공 메시지 출력
def check_complete(result, msg):
    if len(result) == 0:
        print(msg)
        return True
    return False

# Error 여부 확인 및 오류 메시지 출력
def check_error(result, error, msg):
    if result.find(error) != -1 :
        print(msg)
```

Select과 같이 반환값이 존재하는 경우를 제외한 쿼리는 반환값이 없을 때 정상적인 작동으로 간주하였다.

오류를 체크할 때는 에러의 종류를 함께 받아 해당 에러가 반환값에 존재하는지 확인했다.

1. DB 초기화

```
#1. initialize database
def create_table(repo):
    repo.execute_sql(query.sql_create_table, commit = False)

# table 들을 create 하고 csv 파일로 초기화함
def init_db(repo, file_path):
    create_table(repo)

    init_db = pd.read_csv(file_path)
```

```
    movie = init_db.iloc[:,0:3]
    movie.drop_duplicates(keep='first', inplace=True)
    audience = init_db.iloc[:,3:5]
    audience.drop_duplicates(keep='first', inplace=True)

    repo.init_with_csv('movie(title, director, price)', movie)
    repo.init_with_csv('audience(name, age)', audience)

    for index, row in init_db.iterrows():
        mov_id =
int(repo.execute_sql(query.sql_get_mov_id(str(row.iloc[0])))[0][0])
        aud_id = int(repo.execute_sql(query.sql_get_aud_id(str(row.iloc[3]),
str(row.iloc[4])))[0][0])
        repo.execute_sql(query.sql_insert_reservation(mov_id, aud_id))


    print("Database successfully initialized")
```

데이터베이스 초기화 시에는 pandas 라이브러리를 이용했다. Csv 파일을 data frame 으로 변환한 뒤 sql문으로 movie, audience 데이터를 전부 삽입했다.

Reservation의 경우 movie id 와 audience id를 사용해야 했기에 sql문으로 쿼리하여 id를 가져와 삽입했다.

2. 이외 SQL문/에러 체크를 요하는 기능(2~10)

```
#2. print all movies
def print_all_movies(repo):
    result = repo.execute_sql(query.sql_select_movie)
    query.print_table(result, query.header_type.movie)
```

위와 같이 query.py에 저장된 sql문을 실행시켜 기능을 구현했다.

Input이 필요한 경우에는 wrapper 함수를 만들었다. 입력으로 받는 변수가 전부 str 혹은 int라서 int 형식을 wrapper 함수에서 함께 검증한다.

```
#7. remove a user
def remove_user_input(repo):
    aud_id = input("User ID: ")
    if check_int(aud_id, "User id must be integer") : return
    remove_user(repo, aud_id)

def remove_user(repo, aud_id):
```

```
    #사전에 존재하는지 check
    check = repo.execute_sql(query.sql_id_check('audience', aud_id))
    if check_complete(check, f'User {aud_id} does not exist') : return
    result = repo.execute_sql(query.sql_delete("audience", aud_id))
    check_complete(result, 'One user successfully removed')
```

일부 검증을 위해 sql 실행이 추가로 필요한 경우 아래와 같이 check를 통해 구현했다.

```
def insert_user(repo, name, age):
    result = repo.execute_sql(query.sql_insert_audience(name, age))
    if check_complete(result, 'One user successfully inserted') : return
    check_error(result, 'age_range', 'User age should be from 12 to 110')
    check_error(result, 'unique_name_age', f'User ({name}, {age}) already
exists')
```

그리고 에러 검사의 경우 아래와 같이 check_error 함수를 사용했고 성공 여부의 경우 check_complete를 사용했다.

Check_error 의 경우 DB에서 constraint로 발생한 오류만 처리할 수 있어 constraint로 제약하기 어려운 오류의 경우에는 프로그램 차원에서 처리했다.

```
def book_movie(repo, mov_id, aud_id):
    check = repo.execute_sql(query.sql_check_reservation_full(mov_id))
    if (int(check[0][0]) >= 10) :
        print(f'Movie {mov_id} has already been fully booked')
        return
    result = repo.execute_sql(query.sql_insert_reservation(mov_id, aud_id))
    if check_complete(result, 'Movie successfully booked') : return
    check_error(result, 'PRIMARY', f'User {aud_id} has already booked movie
{mov_id}')
    check_error(result, 'fk_mov', f'Movie {mov_id} does not exist')
    check_error(result, 'fk_aud', f'User {aud_id} does not exist')

def remove_movie(repo, mov_id):
    #사전에 존재하는지 check
    check = repo.execute_sql(query.sql_id_check('movie', mov_id))
    if (not str(check[0]).isdigit()): print(f'Movie {mov_id} is in the wrong
format');return
    if check_complete(check, f'Movie {mov_id} does not exist'): return
    result = repo.execute_sql(query.sql_delete("movie", mov_id))
    check_complete(result, 'One movie successfully removed')
```

3. 메인 함수

DB와 연결 및 쿼리문 사용을 위해 Repository 클래스의 인스턴스를 미리 생성해두고 기능을 실행시키는 역할을 맡았다.

```
def main():
    repo = query.Repository()

    while True :
        print(actions)
        action = int(input('Select your action: '))
        if action == 1 : init_db(repo, file_path) # 1. initialize database
        elif action == 2 : print_all_movies(repo) # 2. print all movies
        elif action == 3 : print_all_users(repo) # 3. print all users
        elif action == 4 : insert_movie_input(repo) # 4. insert a new movie
        elif action == 5 : insert_user_input(repo) # 5. insert a new user
        elif action == 6 : remove_movie_input(repo) # 6. remove a movie
        elif action == 7 : remove_user_input(repo) # 7. remove a user
        elif action == 8 : book_movie_input(repo) # 8. book a movie
        elif action == 9 : rate_movie_input(repo) # 9. rate a movie
        elif action == 10 : print_book_movie_input(repo) # 10. print all users
who booked for a movie
        elif action == 11 : print_book_user_input(repo) # 11. print all movies
booked by a user
        elif action == 12 : print("Bye!"); break # 12. exit
        elif action == 13 : reset_input(repo) # 13. reset database
        else : print("Invalid action")

    repo.close_connection()

if __name__ == "__main__":
    main()
```

3-D.실행 방법

Mysql.connector로 데이터베이스를 생성할 수 없어 사전에 데이터베이스 생성이 필요하다.

Connection 정보는 다음과 같다. (Query.py)

```
        db_name ='movie_reserving'
        self.connection = connect(
            host ='127.0.0.1',
            user ='DB',
            password ='0309',
            db =db_name,
            charset ='utf8'
```

```
        )
```

사전에 mysql 서버에서 DB 유저를 만든 뒤, pw를 0309로 설정한다.

만약 기존 유저와 pw를 사용하고 싶다면 코드를 수정한다.

데이터베이스 또한 사전에 생성해 두어야 한다. 마찬가지로 이름은 movie_reserving 이나, 자율적으로 변경할 수 있다.

해당 connection 코드 이외에서 위 정보가 사용되는 일은 없으므로 해당 위치에서만 변경하면 정상적으로 작동한다.

그리고 DB 서버 실행을 위해 mysql을 실행시킨다. 명령 프롬프트로 아래와 같이 진행하면 DB 서버가 실행된다. 마지막에 mysql> 이 커서가 된다면 정상적으로 실행이 완료된 것이다.

```
C:\Users\syeon>mysql -u DB -p
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 341
Server version: 8.0.37 MySQL Community Server - GPL

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

DB까지 설정을 마쳤다면 run.py를 실행시킨 뒤, 최초로 13을 실행시켜 데이터베이스를 초기화한다. 이전에 다른 명령어를 실행 시 crash가 생길 수 있다.

Table 및 데이터 초기화가 완료됐기 때문에 이 이후 프로그램 실행 시에는 13을 먼저 실행시킬 필요가 없다. 프로그램이 제공하는 기능 내에서 데이터베이스를 drop할 수 있는 기능이 제공되지 않기 때문이다.

이후로는 자유롭게 프로그램을 사용할 수 있다.

만약 MySQL 서버와의 연결에 문제가 있을 경우 에러문구 “[ERROR] MySQL Server 와의 연결에 문제가 있습니다. “를 출력하고 프로그램을 종료한다.

4. 실행 예시

실행 예시는 과제 실행 예시와 동일하게 진행했다.

다만, 5번과 6번 기능의 순서가 달라 5번과 6번 기능을 바꿔 실행했고 1번 기능 대신 13번으로 DB를 최초 초기화하였다.

```
============================================================
1. initialize database
2. print all movies
3. print all users
4. insert a new movie
5. insert a new user
6. remove a movie
7. remove a user
8. book a movie
9. rate a movie
10. print all users who booked for a movie
11. print all movies booked by a user
12. exit
13. reset database
============================================================
Select your action: 13
Are you really gonna reset this database? (y/n): y
Database successfully initialized
============================================================
1. initialize database
2. print all movies
3. print all users
4. insert a new movie
5. insert a new user
6. remove a movie
7. remove a user
8. book a movie
9. rate a movie
10. print all users who booked for a movie
11. print all movies booked by a user
12. exit
13. reset database
============================================================
Select your action: 2
------------------------------------------------------------------------------------------------------------------------
--------------------------------
id      title                                                                                                  director
price       reservationavg. rating
------------------------------------------------------------------------------------------------------------------------
--------------------------------
1       1917                                                                                                  Sam Mendes
10000       4       None
2       Alien                                                                                                 Ridley Scott
15000       3       None
3       American History X                                                                                    Tony Kaye
10000       1       None
4       Anand                                                                                                 Hrishikesh Mukherjee
16000       2       None
5       Andhadhun                                                                                             Sriram Raghavan
20000       2       None
6       Avengers: Endgame                                                                                     Anthony Russo
15000       1       None
7       Avengers: Infinity War                                                                                Anthony Russo
15000       2       None
```

8	Ayla: The Daughter of War			Can Ulkay
17000	2	None		
9	Capharnaüm			Nadine Labaki
15000	1	None		
10	Casablanca			Michael Curtiz
25000	3	None		
11	City Lights			Charles Chaplin
25000	2	None		
12	Coco			Lee Unkrich
17000	1	None		
13	Django Unchained			Quentin Tarantino
25000	1	None		
14	Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb			Stanley Kubrick
17000	2	None		
15	Eternal Sunshine of the Spotless Mind			Michel Gondry
10000	3	None		
16	Fight Club			David Fincher
10000	1	None		
17	Gisaengchung			Bong Joon Ho
8500	1	None		
18	Gladiator			Ridley Scott
17000	1	None		
19	Good Will Hunting			Gus Van Sant
8500	2	None		
20	Goodfellas			Martin Scorsese
15000	1	None		
21	Hotaru no haka			Isao Takahata
10000	1	None		
22	Il buono, il brutto, il cattivo			Sergio Leone
17000	1	None		
23	Inglourious Basterds			Quentin Tarantino
15000	3	None		
24	It's a Wonderful Life			Frank Capra
17000	2	None		
25	La vita e bella			Roberto Benigni
20000	1	None		
26	Miracle in cell NO.7			Mehmet Ada _ztekin
15000	2	None		
27	Modern Times			Charles Chaplin
20000	1	None		
28	Mononoke-hime			Hayao Miyazaki
20000	1	None		
29	Oldeuboi			Chan-wook Park
10000	3	None		
30	Once Upon a Time in America			Sergio Leone
8500	1	None		
31	Once Upon a Time in the West			Sergio Leone
8500	1	None		
32	Paths of Glory			Stanley Kubrick
17000	1	None		
33	Raiders of the Lost Ark			Steven Spielberg
10000	2	None		
34	Requiem for a Dream			Darren Aronofsky
10000	1	None		
35	Saving Private Ryan			Steven Spielberg
17000	2	None		
36	Sen to Chihiro no kamikakushi			Hayao Miyazaki
17000	1	None		
37	Shichinin no samurai			Akira Kurosawa
20000	2	None		
38	Snatch			Guy Ritchie
10000	1	None		
39	Soorarai Pottru			Sudha Kongara
20000	1	None		
40	Spider-Man: Into the Spider-Verse			Bob Persichetti

```
25000       3       None
41      Star Wars                                                                                                George Lucas
15000       3       None
42      Star Wars: Episode V - The Empire Strikes Back                                                           Irvin Kershner
20000       1       None
43      The Dark Knight                                                                                          Christopher Nolan
17000       1       None
44      The Dark Knight Rises                                                                                    Christopher Nolan
17000       2       None
45      The Departed                                                                                             Martin Scorsese
20000       3       None
46      The Godfather: Part II                                                                                   Francis Ford Coppola
25000       1       None
47      The Great Dictator                                                                                       Charles Chaplin
15000       3       None
48      The Green Mile                                                                                           Frank Darabont
15000       2       None
49      The Intouchables                                                                                         Olivier Nakache
25000       1       None
50      The Lion King                                                                                            Roger Allers
8500        1       None
51      The Lives of Others                                                                                      Florian Henckel von
Donnersmarck    20000       2       None
52      The Lord of the Rings: The Fellowship of the Ring                                                        Peter Jackson
20000       1       None
53      The Lord of the Rings: The Return of the King                                                            Peter Jackson
20000       1       None
54      The Matrix                                                                                               Lana Wachowski
8500        2       None
55      The Shining                                                                                              Stanley Kubrick
20000       2       None
56      The Usual Suspects                                                                                       Bryan Singer
15000       2       None
57      Tumbbad                                                                                                  Rahi Anil Barve
10000       1       None
58      Vikram Vedha                                                                                                 Gayatri
17000       2       None
59      WALL·E                                                                                                   Andrew Stanton
10000       1       None
60      Witness for the Prosecution                                                                              Billy Wilder
25000       1       None
-------------------------------------------------------------------------------------------------------------------------------
------------------------------
============================================================
1. initialize database
2. print all movies
3. print all users
4. insert a new movie
5. insert a new user
6. remove a movie
7. remove a user
8. book a movie
9. rate a movie
10. print all users who booked for a movie
11. print all movies booked by a user
12. exit
13. reset database
============================================================
Select your action: 3
--------------------------------------------
id          name                            age
--------------------------------------------
1           Ava                             51
2           Mason                            22
3           Charlotte                       54
```

4	Sophia	22
5	Emily	72
6	Jacob	29
7	Aria	18
8	Elijah	72
9	Olivia	18
10	Harper	37
11	John	17
12	Wyatt	17
13	Mason	33
14	Olivia	36
15	Owen	49
16	Mateo	71
17	Ethan	62
18	Ellie	26
19	Noah	22
20	Wyatt	67
21	Sebastian	21
22	Noah	35
23	Wyatt	81
24	William	81
25	Ava	63
26	Daniel	47
27	Sebastian	66
28	Abigail	62
29	Mia	69
30	Camila	17
31	Olivia	65
32	Abigail	86
33	Michael	36
34	Harper	46
35	Samuel	71
36	Jacob	72
37	Michael	90
38	Layla	62
39	Alexander	66
40	Joseph	23
41	Scarlett	50
42	Nora	19
43	Daniel	53
44	Mason	36
45	James	38
46	Sofia	46
47	Ava	67
48	Elizabeth	56
49	Layla	77
50	Mason	49
51	Mateo	73
52	Mia	70
53	Olivia	71
54	Ethan	67
55	Henry	47
56	Joseph	57
57	Emma	27
58	Scarlett	54
59	Luna	66
60	Jackson	23
61	Sophia	57
62	Penelope	43
63	Jackson	63
64	Camila	32
65	Aria	26
66	Elizabeth	88
67	Luna	85
68	Aria	35

```
69          Hazel                       18
70          Mila                        63
71          Evelyn                      76
72          Jackson                     90
73          Charlotte                   66
74          Jack                        38
75          Henry                       80
76          Madison                     39
77          Avery                       87
78          Logan                       21
79          Camila                      35
80          Owen                        68
81          Eleanor                     20
82          Jacob                       83
83          Noah                        36
84          Mason                       52
85          Aiden                       18
86          Penelope                    48
87          John                        51
88          Wyatt                       86
89          Nora                        44
90          Alexander                   90
91          James                       73
92          Eleanor                     61
93          Samuel                      82
94          Ellie                       46
95          Liam                        34
96          Nora                        82
97          Benjamin                    61
98          Olivia                      78
99          Mason                       70
100         Logan                       88
-------------------------------------------
============================================================
1. initialize database
2. print all movies
3. print all users
4. insert a new movie
5. insert a new user
6. remove a movie
7. remove a user
8. book a movie
9. rate a movie
10. print all users who booked for a movie
11. print all movies booked by a user
12. exit
13. reset database
============================================================
Select your action: 4
Movie title: Alien2
Movie director: Ridley Scott
Movie price: 5000
One movie successfully inserted
============================================================
1. initialize database
2. print all movies
3. print all users
4. insert a new movie
5. insert a new user
6. remove a movie
7. remove a user
8. book a movie
9. rate a movie
10. print all users who booked for a movie
11. print all movies booked by a user
```

```
12. exit
13. reset database
============================================================
Select your action: 6
Moive id: My User Name
Movie id must be integer
============================================================
1. initialize database
2. print all movies
3. print all users
4. insert a new movie
5. insert a new user
6. remove a movie
7. remove a user
8. book a movie
9. rate a movie
10. print all users who booked for a movie
11. print all movies booked by a user
12. exit
13. reset database
============================================================
Select your action: 5
User name: My User Name
User age: 19
One user successfully inserted
============================================================
1. initialize database
2. print all movies
3. print all users
4. insert a new movie
5. insert a new user
6. remove a movie
7. remove a user
8. book a movie
9. rate a movie
10. print all users who booked for a movie
11. print all movies booked by a user
12. exit
13. reset database
============================================================
Select your action: 2
------------------------------------------------------------------------------------------------------------------------
--------------------------------
id      title                                                                                                   director
price       reservationavg. rating
------------------------------------------------------------------------------------------------------------------------
--------------------------------
1       1917                                                                                                    Sam Mendes
10000       4       None
2       Alien                                                                                                   Ridley Scott
15000       3       None
3       American History X                                                                                      Tony Kaye
10000       1       None
4       Anand                                                                                                   Hrishikesh Mukherjee
16000       2       None
5       Andhadhun                                                                                               Sriram Raghavan
20000       2       None
6       Avengers: Endgame                                                                                       Anthony Russo
15000       1       None
7       Avengers: Infinity War                                                                                  Anthony Russo
15000       2       None
8       Ayla: The Daughter of War                                                                               Can Ulkay
17000       2       None
9       Capharnaüm                                                                                              Nadine Labaki
15000       1       None
```

10	Casablanca			Michael Curtiz
25000	3	None		
11	City Lights			Charles Chaplin
25000	2	None		
12	Coco			Lee Unkrich
17000	1	None		
13	Django Unchained			Quentin Tarantino
25000	1	None		
14	Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb			Stanley Kubrick
17000	2	None		
15	Eternal Sunshine of the Spotless Mind			Michel Gondry
10000	3	None		
16	Fight Club			David Fincher
10000	1	None		
17	Gisaengchung			Bong Joon Ho
8500	1	None		
18	Gladiator			Ridley Scott
17000	1	None		
19	Good Will Hunting			Gus Van Sant
8500	2	None		
20	Goodfellas			Martin Scorsese
15000	1	None		
21	Hotaru no haka			Isao Takahata
10000	1	None		
22	Il buono, il brutto, il cattivo			Sergio Leone
17000	1	None		
23	Inglourious Basterds			Quentin Tarantino
15000	3	None		
24	It's a Wonderful Life			Frank Capra
17000	2	None		
25	La vita e bella			Roberto Benigni
20000	1	None		
26	Miracle in cell NO.7			Mehmet Ada _ztekin
15000	2	None		
27	Modern Times			Charles Chaplin
20000	1	None		
28	Mononoke-hime			Hayao Miyazaki
20000	1	None		
29	Oldeuboi			Chan-wook Park
10000	3	None		
30	Once Upon a Time in America			Sergio Leone
8500	1	None		
31	Once Upon a Time in the West			Sergio Leone
8500	1	None		
32	Paths of Glory			Stanley Kubrick
17000	1	None		
33	Raiders of the Lost Ark			Steven Spielberg
10000	2	None		
34	Requiem for a Dream			Darren Aronofsky
10000	1	None		
35	Saving Private Ryan			Steven Spielberg
17000	2	None		
36	Sen to Chihiro no kamikakushi			Hayao Miyazaki
17000	1	None		
37	Shichinin no samurai			Akira Kurosawa
20000	2	None		
38	Snatch			Guy Ritchie
10000	1	None		
39	Soorarai Pottru			Sudha Kongara
20000	1	None		
40	Spider-Man: Into the Spider-Verse			Bob Persichetti
25000	3	None		
41	Star Wars			George Lucas
15000	3	None		
42	Star Wars: Episode V - The Empire Strikes Back			Irvin Kershner

```
20000       1       None
43      The Dark Knight                                                                 Christopher Nolan
17000       1       None
44      The Dark Knight Rises                                                           Christopher Nolan
17000       2       None
45      The Departed                                                                    Martin Scorsese
20000       3       None
46      The Godfather: Part II                                                          Francis Ford Coppola
25000       1       None
47      The Great Dictator                                                              Charles Chaplin
15000       3       None
48      The Green Mile                                                                  Frank Darabont
15000       2       None
49      The Intouchables                                                                Olivier Nakache
25000       1       None
50      The Lion King                                                                   Roger Allers
8500        1       None
51      The Lives of Others                                                             Florian Henckel von
Donnersmarck    20000       2       None
52      The Lord of the Rings: The Fellowship of the Ring                               Peter Jackson
20000       1       None
53      The Lord of the Rings: The Return of the King                                   Peter Jackson
20000       1       None
54      The Matrix                                                                      Lana Wachowski
8500        2       None
55      The Shining                                                                     Stanley Kubrick
20000       2       None
56      The Usual Suspects                                                              Bryan Singer
15000       2       None
57      Tumbbad                                                                         Rahi Anil Barve
10000       1       None
58      Vikram Vedha                                                                    Gayatri
17000       2       None
59      WALL·E                                                                          Andrew Stanton
10000       1       None
60      Witness for the Prosecution                                                     Billy Wilder
25000       1       None
61      Alien2                                                                          Ridley Scott
5000        1       None
-------------------------------------------------------------------------------------------------------------------------
------------------------------
============================================================
1. initialize database
2. print all movies
3. print all users
4. insert a new movie
5. insert a new user
6. remove a movie
7. remove a user
8. book a movie
9. rate a movie
10. print all users who booked for a movie
11. print all movies booked by a user
12. exit
13. reset database
============================================================
Select your action: 3
--------------------------------------------
id          name                            age
--------------------------------------------
1           Ava                             51
2           Mason                           22
3           Charlotte                       54
4           Sophia                          22
5           Emily                           72
```

6	Jacob	29
7	Aria	18
8	Elijah	72
9	Olivia	18
10	Harper	37
11	John	17
12	Wyatt	17
13	Mason	33
14	Olivia	36
15	Owen	49
16	Mateo	71
17	Ethan	62
18	Ellie	26
19	Noah	22
20	Wyatt	67
21	Sebastian	21
22	Noah	35
23	Wyatt	81
24	William	81
25	Ava	63
26	Daniel	47
27	Sebastian	66
28	Abigail	62
29	Mia	69
30	Camila	17
31	Olivia	65
32	Abigail	86
33	Michael	36
34	Harper	46
35	Samuel	71
36	Jacob	72
37	Michael	90
38	Layla	62
39	Alexander	66
40	Joseph	23
41	Scarlett	50
42	Nora	19
43	Daniel	53
44	Mason	36
45	James	38
46	Sofia	46
47	Ava	67
48	Elizabeth	56
49	Layla	77
50	Mason	49
51	Mateo	73
52	Mia	70
53	Olivia	71
54	Ethan	67
55	Henry	47
56	Joseph	57
57	Emma	27
58	Scarlett	54
59	Luna	66
60	Jackson	23
61	Sophia	57
62	Penelope	43
63	Jackson	63
64	Camila	32
65	Aria	26
66	Elizabeth	88
67	Luna	85
68	Aria	35
69	Hazel	18
70	Mila	63

```
71          Evelyn                    76
72          Jackson                   90
73          Charlotte                 66
74          Jack                      38
75          Henry                     80
76          Madison                   39
77          Avery                     87
78          Logan                     21
79          Camila                    35
80          Owen                      68
81          Eleanor                   20
82          Jacob                     83
83          Noah                      36
84          Mason                     52
85          Aiden                     18
86          Penelope                  48
87          John                      51
88          Wyatt                     86
89          Nora                      44
90          Alexander                 90
91          James                     73
92          Eleanor                   61
93          Samuel                    82
94          Ellie                     46
95          Liam                      34
96          Nora                      82
97          Benjamin                  61
98          Olivia                    78
99          Mason                     70
100         Logan                     88
101         My User Name              19
-----------------------------------------
============================================================
1. initialize database
2. print all movies
3. print all users
4. insert a new movie
5. insert a new user
6. remove a movie
7. remove a user
8. book a movie
9. rate a movie
10. print all users who booked for a movie
11. print all movies booked by a user
12. exit
13. reset database
============================================================
Select your action: 8
Movie ID: 3
User ID: 1
Movie successfully booked
============================================================
1. initialize database
2. print all movies
3. print all users
4. insert a new movie
5. insert a new user
6. remove a movie
7. remove a user
8. book a movie
9. rate a movie
10. print all users who booked for a movie
11. print all movies booked by a user
12. exit
13. reset database
```

```
============================================================
Select your action: 9
Movie ID: 1
User ID: 1
Ratings (1~5): 2
Movie successfully rated
============================================================
1. initialize database
2. print all movies
3. print all users
4. insert a new movie
5. insert a new user
6. remove a movie
7. remove a user
8. book a movie
9. rate a movie
10. print all users who booked for a movie
11. print all movies booked by a user
12. exit
13. reset database
============================================================
Select your action: 9
Movie ID: 2
User ID: 1
Ratings (1~5): 4
User 1 has not booked movie 2 yet
============================================================
1. initialize database
2. print all movies
3. print all users
4. insert a new movie
5. insert a new user
6. remove a movie
7. remove a user
8. book a movie
9. rate a movie
10. print all users who booked for a movie
11. print all movies booked by a user
12. exit
13. reset database
============================================================
Select your action: 9
Movie ID: 2
User ID: 2
Ratings (1~5): 4
User 2 has not booked movie 2 yet
============================================================
1. initialize database
2. print all movies
3. print all users
4. insert a new movie
5. insert a new user
6. remove a movie
7. remove a user
8. book a movie
9. rate a movie
10. print all users who booked for a movie
11. print all movies booked by a user
12. exit
13. reset database
============================================================
Select your action: 9
Movie ID: 3
User ID: 1
Ratings (1~5): 4
```

```
Movie successfully rated
============================================================
1. initialize database
2. print all movies
3. print all users
4. insert a new movie
5. insert a new user
6. remove a movie
7. remove a user
8. book a movie
9. rate a movie
10. print all users who booked for a movie
11. print all movies booked by a user
12. exit
13. reset database
============================================================
Select your action: 10
Movie ID: 1
--------------------------------------------------
id          name                            age   rating
--------------------------------------------------
1           Ava                             51    2
2           Mason                           22    None
3           Charlotte                       54    None
4           Sophia                          22    None
--------------------------------------------------
============================================================
1. initialize database
2. print all movies
3. print all users
4. insert a new movie
5. insert a new user
6. remove a movie
7. remove a user
8. book a movie
9. rate a movie
10. print all users who booked for a movie
11. print all movies booked by a user
12. exit
13. reset database
============================================================
Select your action: 11
User ID: 1
------------------------------------------------------------------------------------------------------------------------
----------------------
id          title                                                                                          director
rating
------------------------------------------------------------------------------------------------------------------------
----------------------
1           1917                                                                                           Sam Mendes
2
3           American History X                                                                             Tony Kaye
4
------------------------------------------------------------------------------------------------------------------------
----------------------
============================================================
1. initialize database
2. print all movies
3. print all users
4. insert a new movie
5. insert a new user
6. remove a movie
7. remove a user
8. book a movie
9. rate a movie
```

```
10. print all users who booked for a movie
11. print all movies booked by a user
12. exit
13. reset database
============================================================
Select your action: 12
Bye!
```