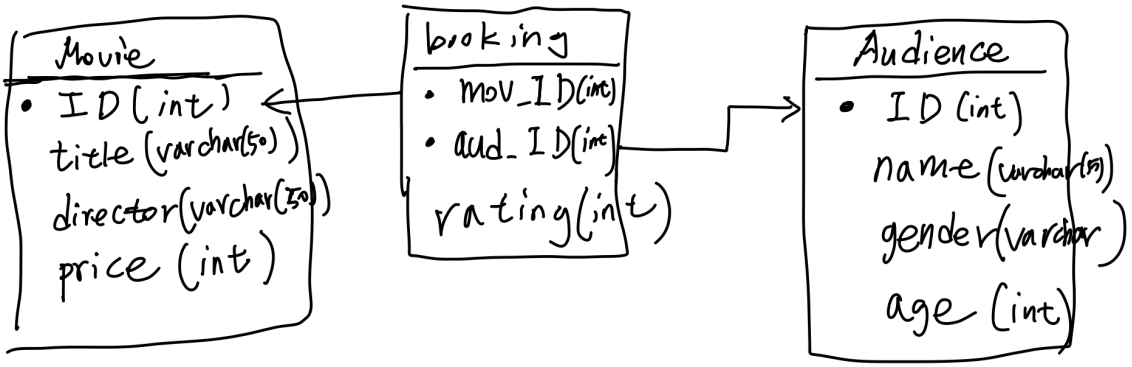
DB Project 2 Report

2017-16140
김준오

1. DB 스키마 설계

표현해야 하는 정보는 영화, 관객, 그리고 예매와 평점이다. 영화 데이터에서 표현해야 하는 정보는 영화 제목, 감독 이름, 가격이고, 관객에 대해 표현해야 하는 정보는 이름, 성별, 나이이다. 또한 예매를 한 영화에 대해서만 평점을 매길 수 있으므로 평점 정보는 예매에 묶어서 표현 가능하다. 이를 관계형 모델로 설계한 것이 다음과 같다.



2. 모듈 및 알고리즘 구현

가장 먼저 선행되어야 하는 작업은 데이터베이스를 초기화하는 일이다. CSV 파일로부터 Raw Data를 읽어와 데이터베이스의 각 테이블에 삽입해야 하는데, CSV 파일의 Row 한 줄은 Booking 테이블에 들어갈 Record 1개를 의미한다. 즉, Movie record와 Audience record들을 먼저 추출한 다음, 그 ID를 이용해서 다시 Booking record들을 삽입해야 했다.

추천 시스템을 구현하기 위해 우선 각 유저의 각 영화 별 평점을 벡터로 갖는 user_item_matrix와, 입력 받은 유저 – 다른 모든 유저 사이의 cosine_similarity를 저장하는 user_similarity_vector를 모두 0으로 초기화 한 상태로 준비하였다. 그런 다음 평점을 매긴 적이 있는 user들에 한해서 해당 user의 user_item_matrix의 벡터를 평점의 평균값으로 채우고, 입력 받은 유저 – 다른 모든 유저 간의 cosine similarity를 계산하였다. 마지막으로 그렇게 계산된 cosine similarity를 weight으로 하는 wetight average를 계산하고, 입력 받은 유저가 직접 평점을 매긴 적 없는 영화들 중 가장 가중치가 큰 영화를 선택하도록 구현하였다.

그 외의 구현들은 MySQL 쿼리를 활용하여 구현되었다.

3. 컴파일 및 실행 방법

같은 디렉토리 내에 data.csv와 run.py가 존재하는 상황에서 python3 run.py 명령어를 실행한다. 이때 첫 실행의 경우 db와 연결되지 않아 오류가 발생할 수 있지만, 다시 시도하면 연결이 된 상태로 프로그램이 진행 된다.

4. 가정한 것들

동일한 제목의 영화, 동일한 이름, 성별, 나이 set을 가진 관객은 다시 추가되지 않을 것이라고 가정한 채로 작업하였다. 모든 것을 ID로 쿼리하기 때문에 추가된다 하더라도 ID로 구별될 수 있

지만, 실제 사용 시에는 혼동이 올 수 있을 것 같다.

앞서도 말했지만 첫 실행 시 연결에 실패하면서 프로그램이 종료된다. 다시 실행하면 연결이 된 상태로 한동안은 실패하지 않고 바로 작동한다.

reset 작동 시 모든 table을 drop하고, 다시 create 한 다음 데이터를 처리해서 초기화를 진행하게 된다.

5. 느낀 점

직접 사용해보지 않은 상태에서 페이퍼만으로 요구 사항을 충분히 만족하는 스키마를 설계하기란 상당히 어렵다는 것을 알게 되었다. 굉장히 간단한 스펙임에도 설계하고 또 구현하면서 조금씩 수정이 가해졌는데, 규모가 커지면 더욱 복잡해질 것이고 수정 사항도 많아질 것이라는 생각이 들었다. 처음부터 완벽하게 하는 것도 좋지만, 협업자 간의 소통이나 문서화가 이를 보완하는 것 또한 매우 중요한 점 중 하나가 될 것이라고 느껴졌다.