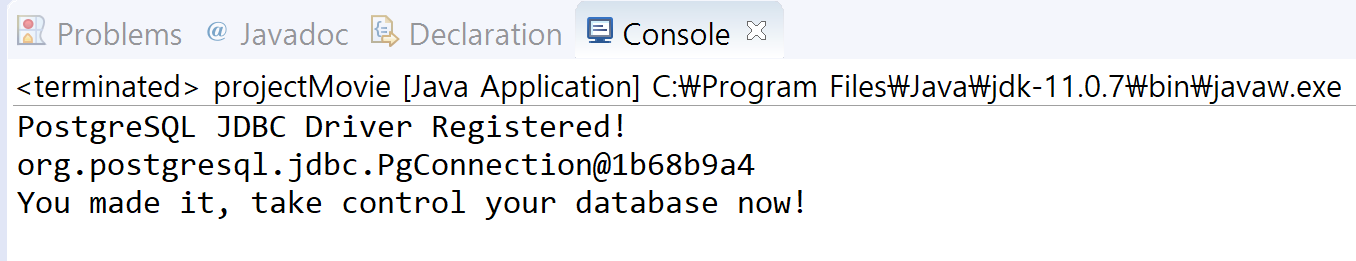
12181726이현진 데이터베이스 002분반 Project1

[1] 구현 처음부터 마지막까지 모두 성공

[2] 구현 과정은 4.에서 캡쳐한 출력결과와 같이 설명한다.

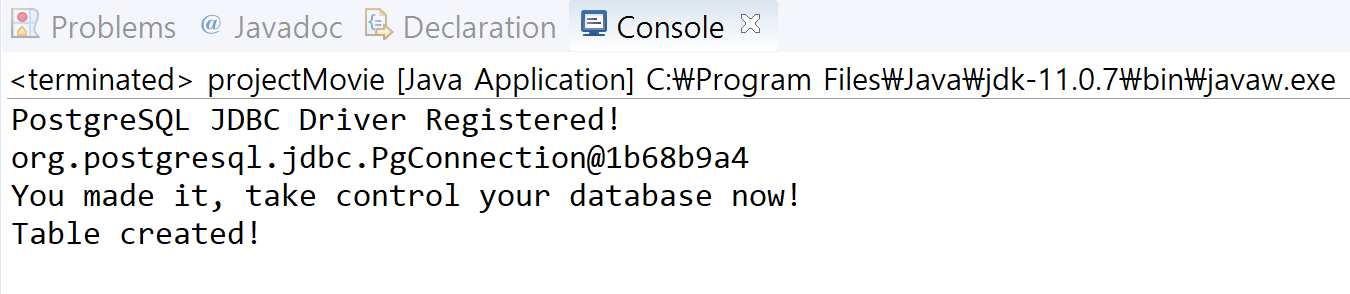
[3] 한 문제씩 try/catch block을 만들어 실행(컴파일)하고 실행이 완료된 try/catch는 /\*\*/로 주석처리하였다. 이는 다음 문제 실행에 영향을 미치지 않도록 하기위함과, 문제단위씩 exeption이나 error 캐치하여 troubleshooting을 하기 위함이다.

[4] 0. 연결성공

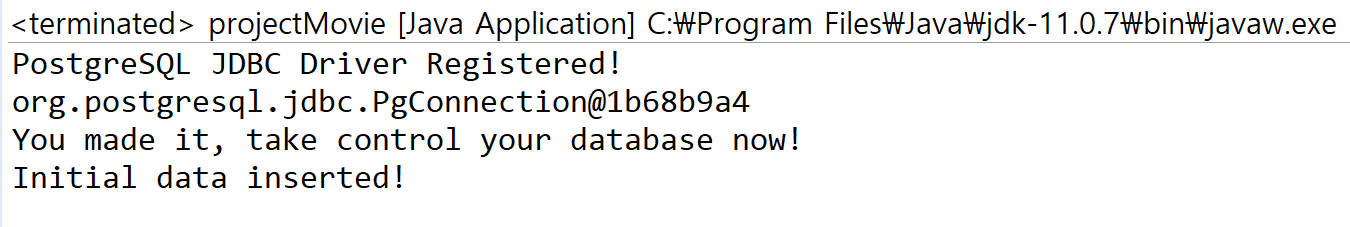


1. Create table

Director, actor, movie, customer부터 create(이들이 다른 table들의 가장 기본적인 referenced table이 된다. 다음으로 award, genre를 create(이들 primary key역시 다른 table에서 foreign key가 참조한다.). award는 movieObtain, actorObtain, directorObtain이 referencing하고, genre는 movieGenre가 referencing하도록 설계했다. 그 뒤에 나머지 table들을 create하였다.

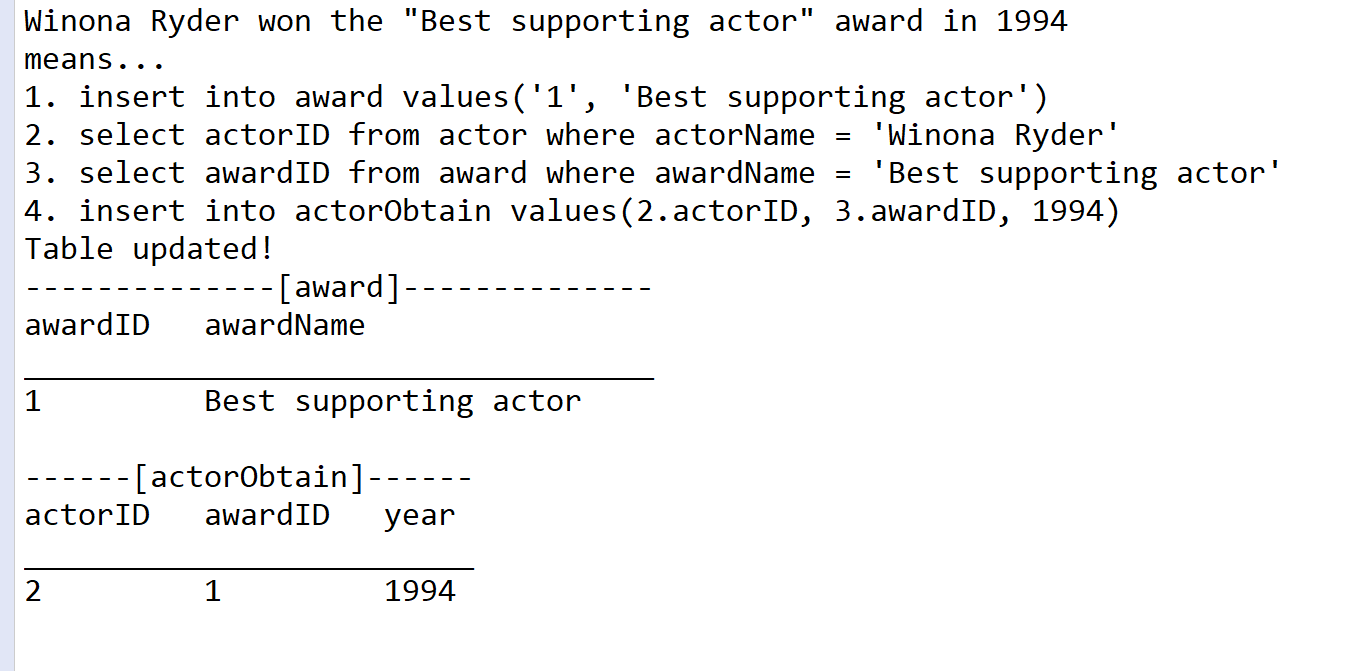


and initialize with data. award, movieObtain, actorObtain, directorObtain, customerRate는 표 상으로 아직 넣을 데이터가 없다.

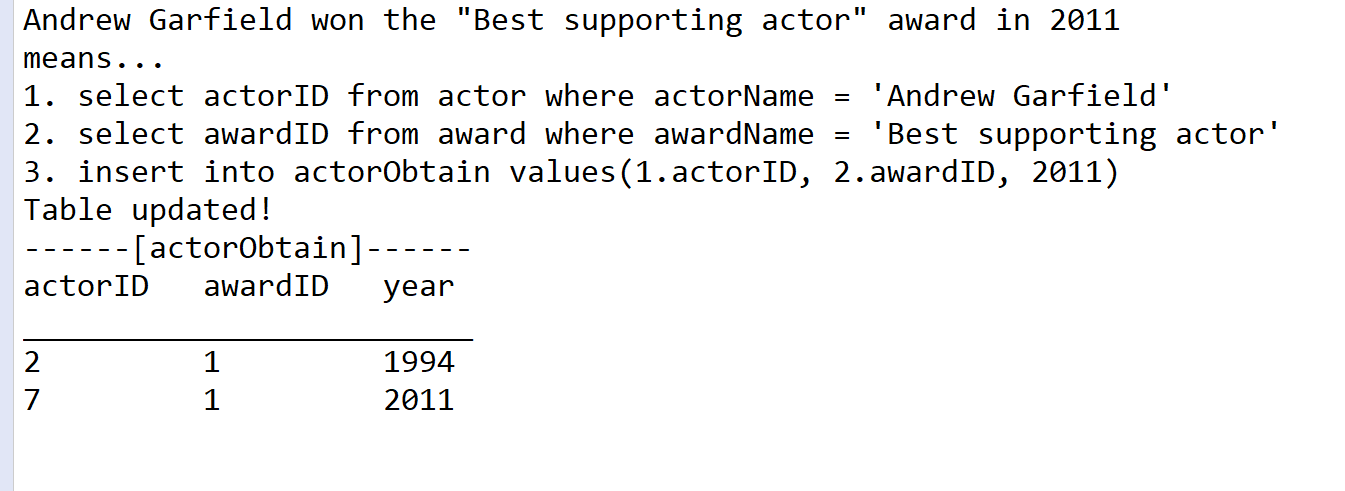


2. insert the proper data. Award, movieObtain, actorObtain, directorObtain을 update하는 과정이다. 각 문제마다 새로운 award가 들어오면 award에 넣고, 지정하는 awardID와 수상자ID를 select하여 문자타입으로 저장한 후, 이에 맞는 각 Obtain table에 값을 update해주는 과정의 반복이다.

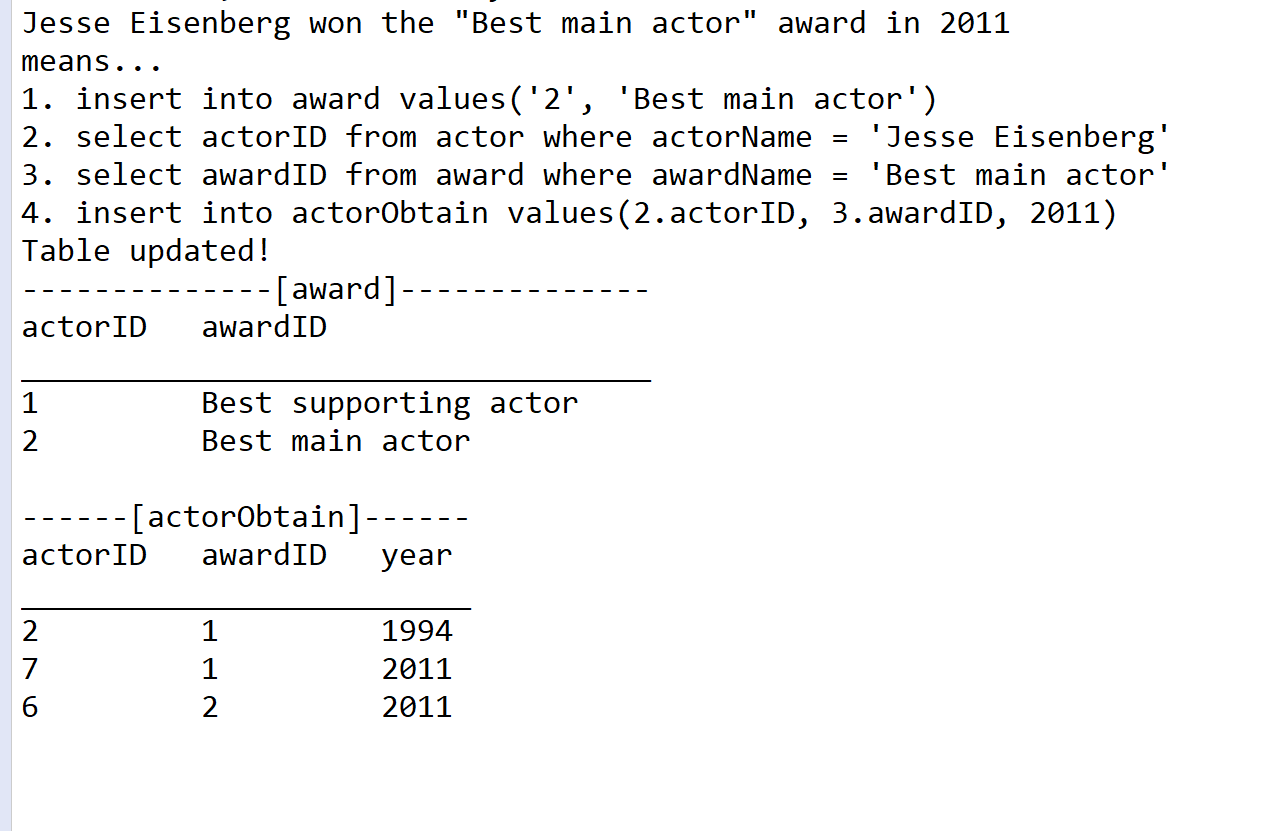
2.1 실행결과



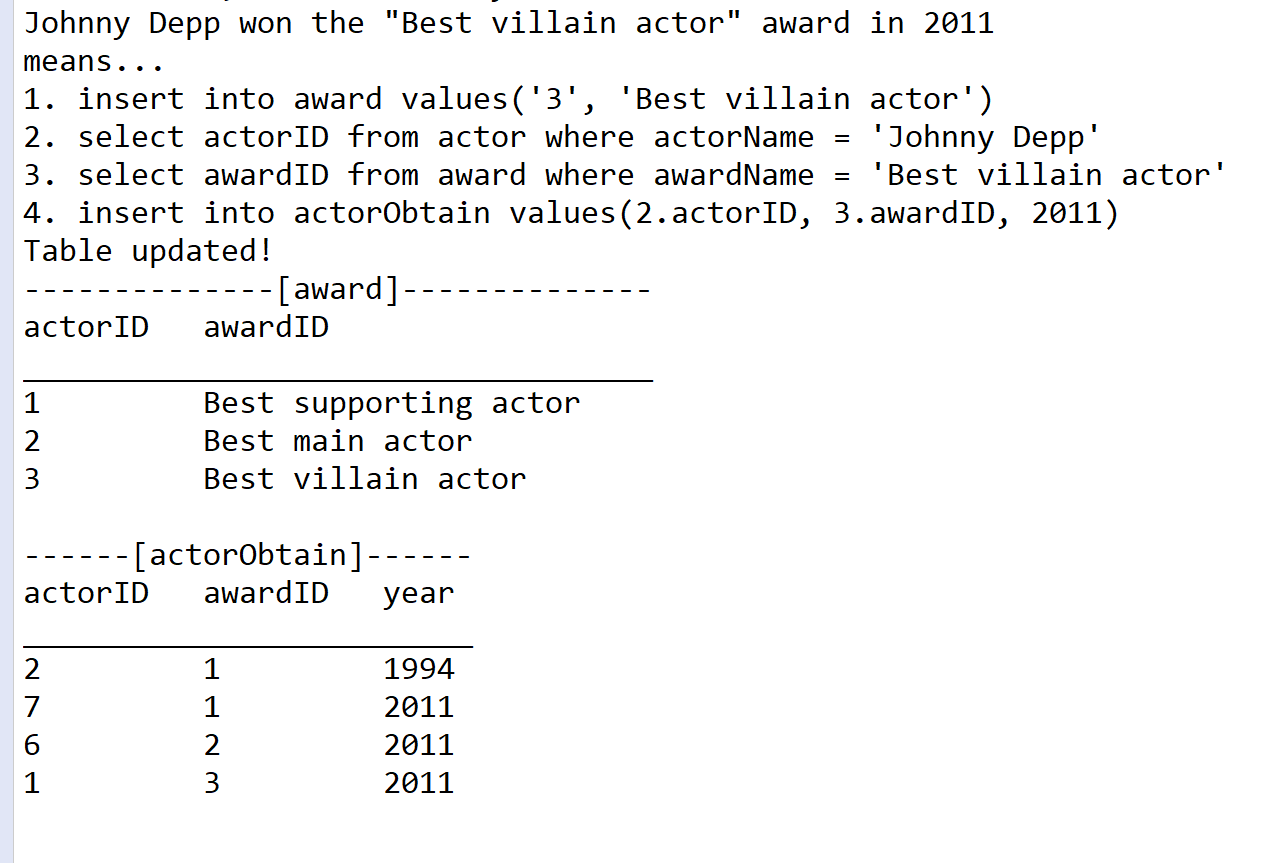
2.2 실행결과



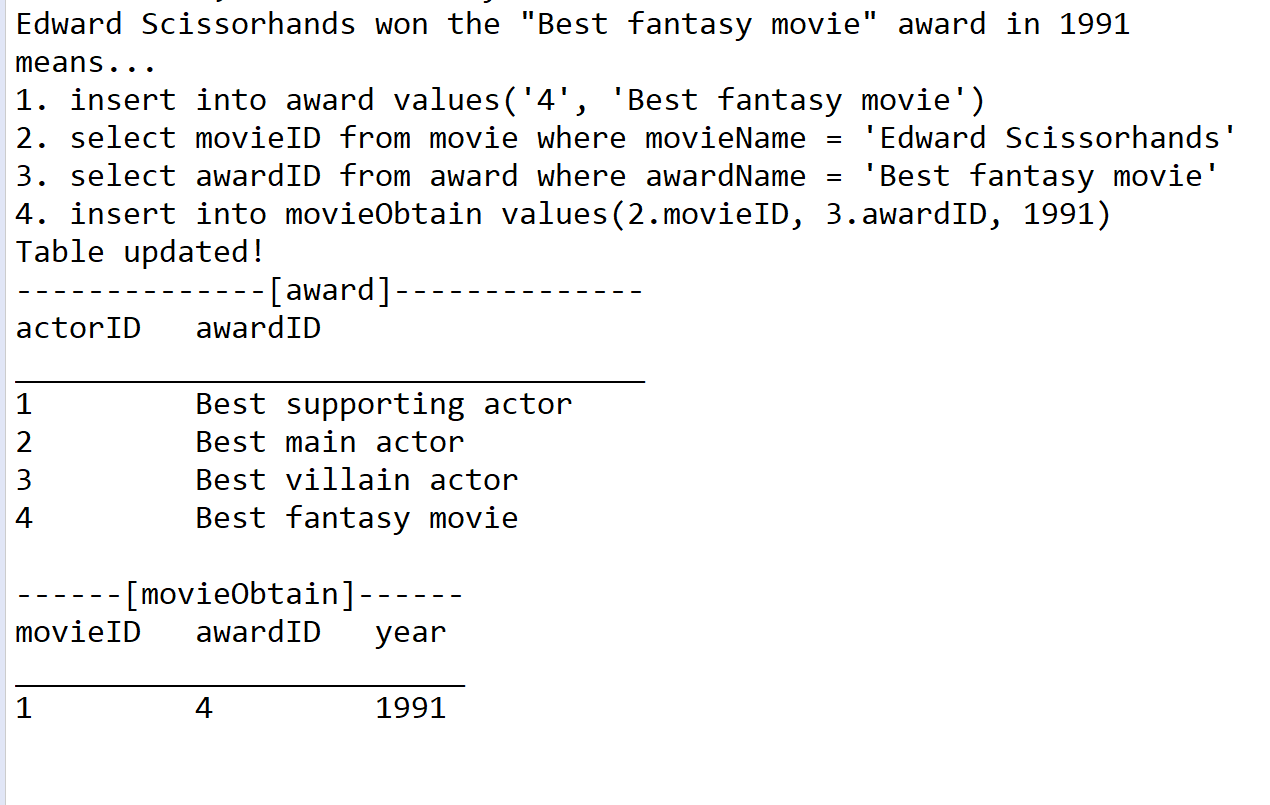
2.3 실행결과



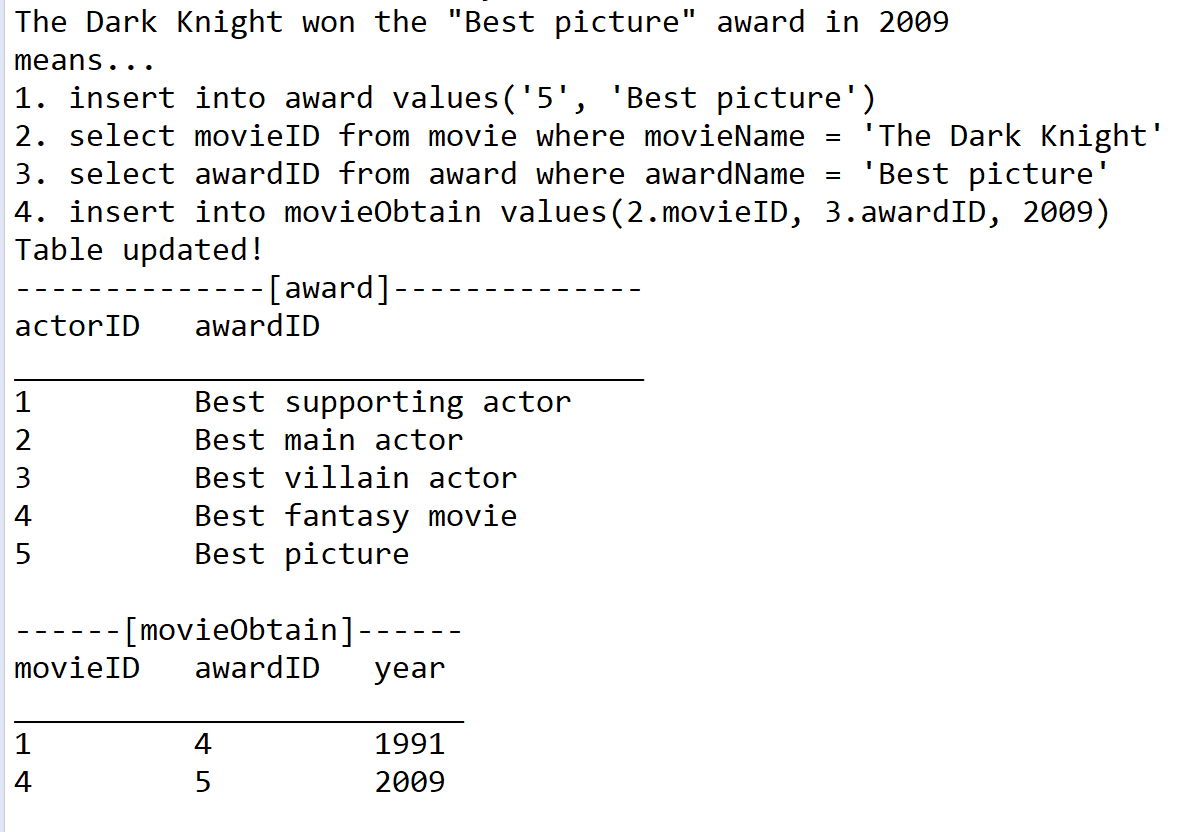
2.4 실행결과



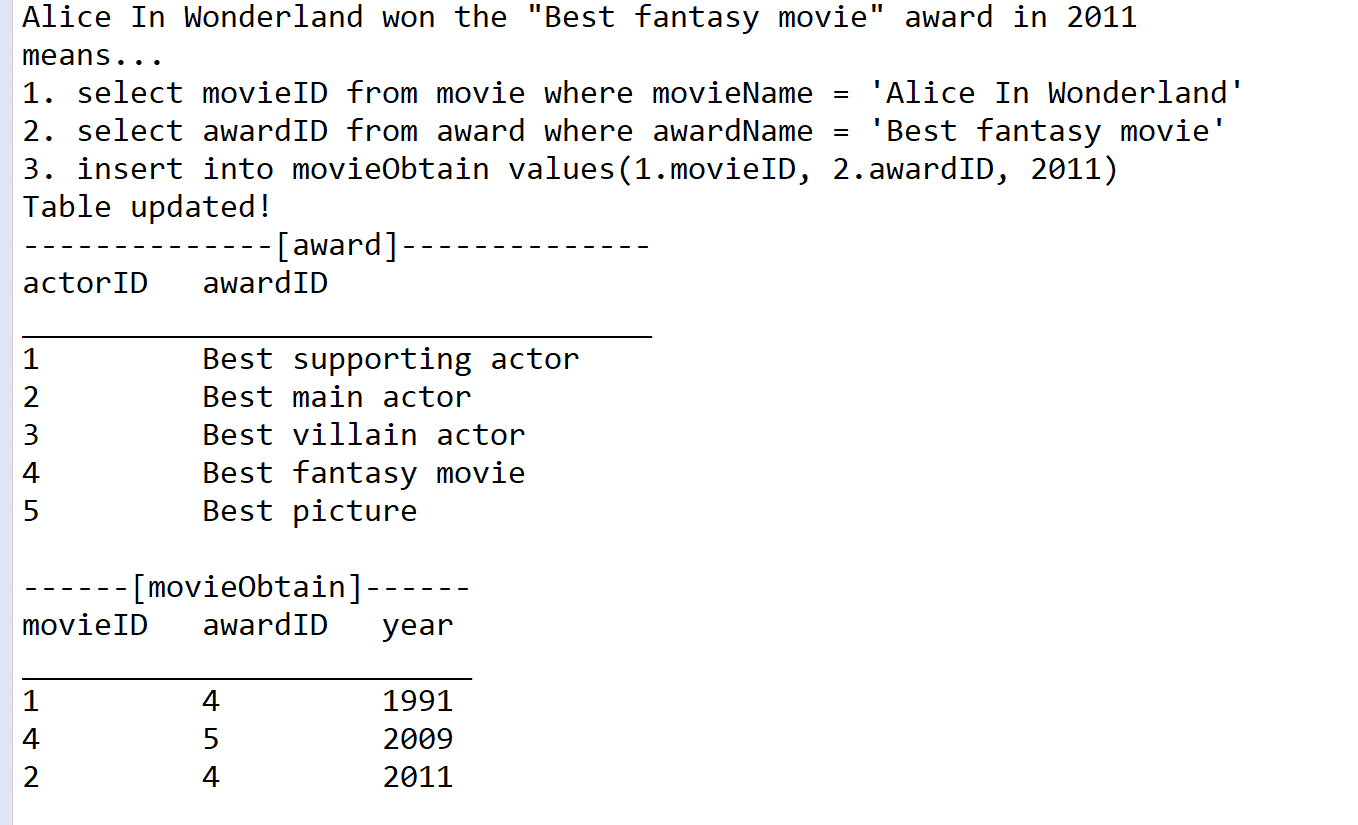
2.5 실행결과



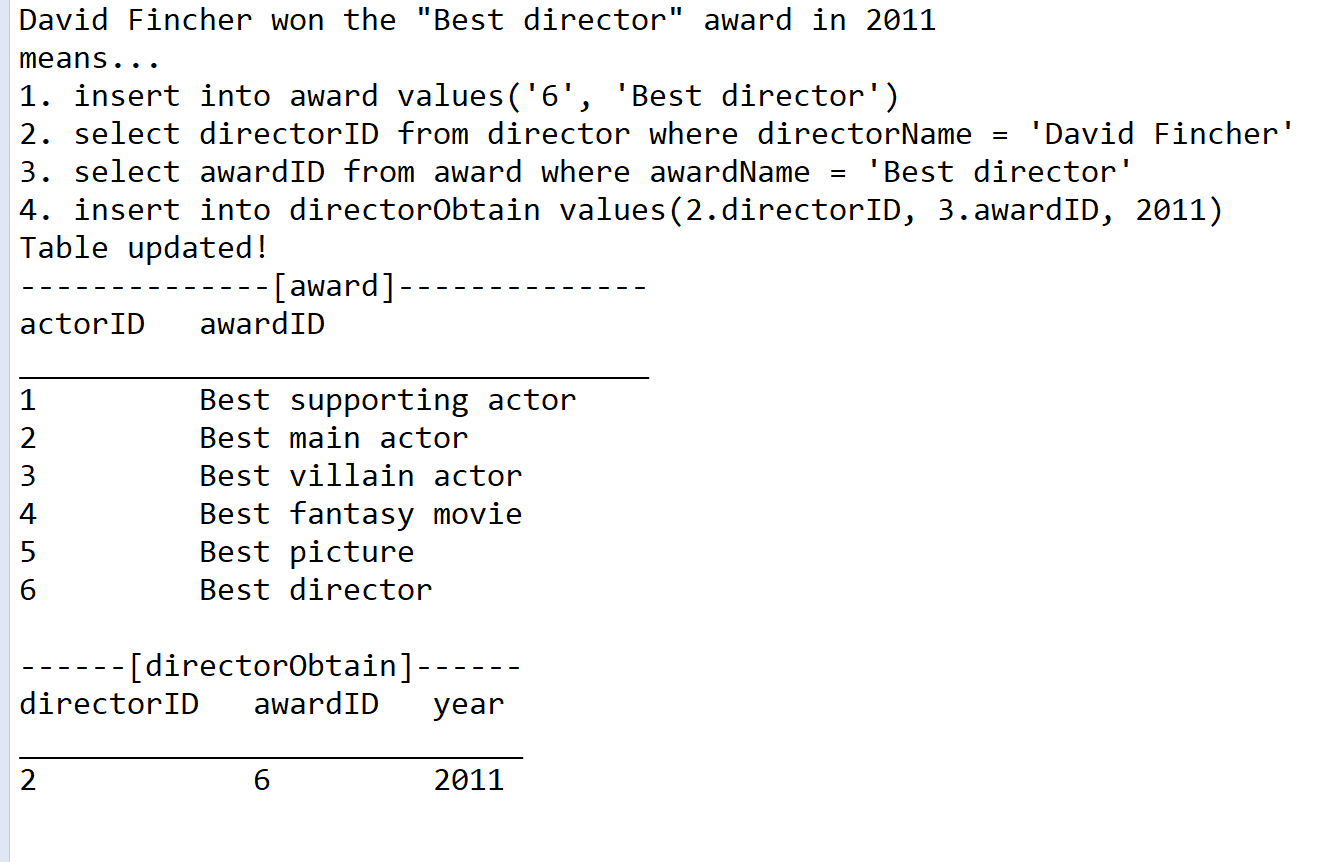
2.6 실행결과



2.7 실행결과

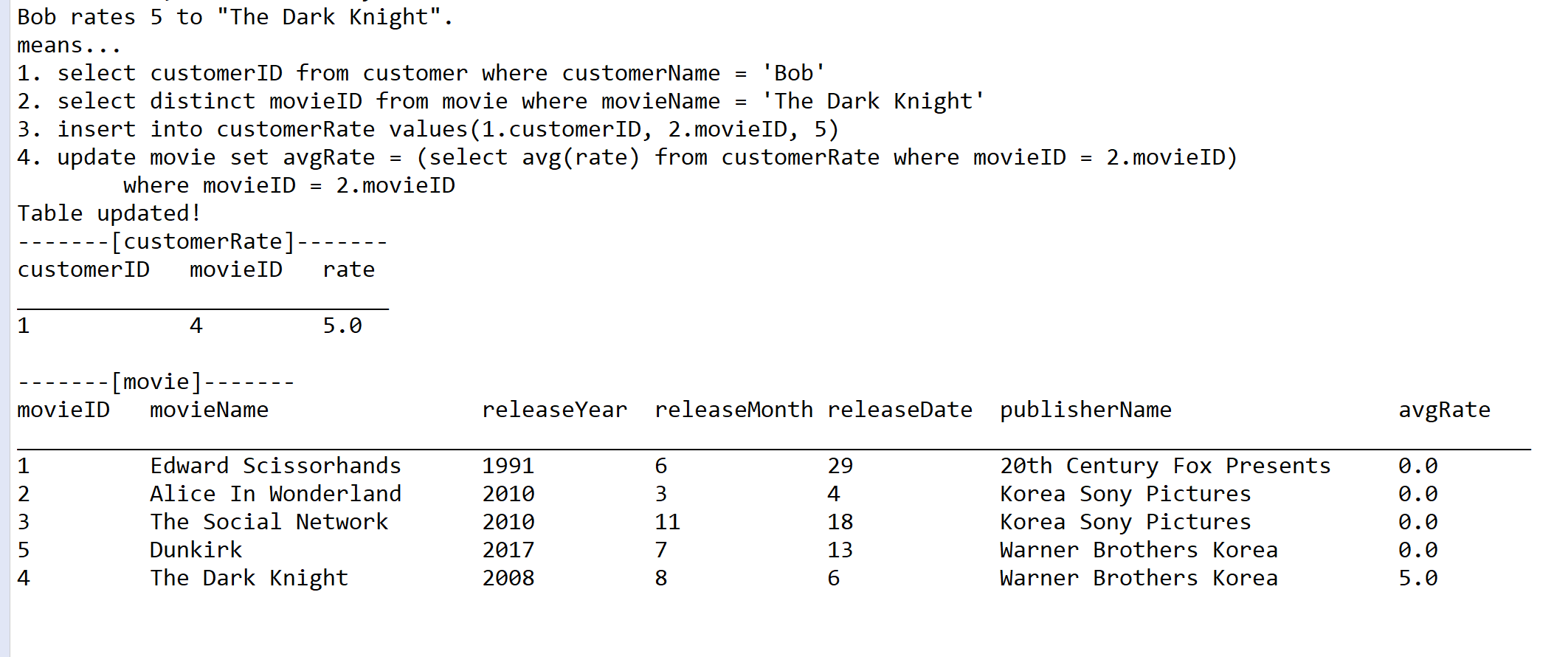


2.8 실행결과

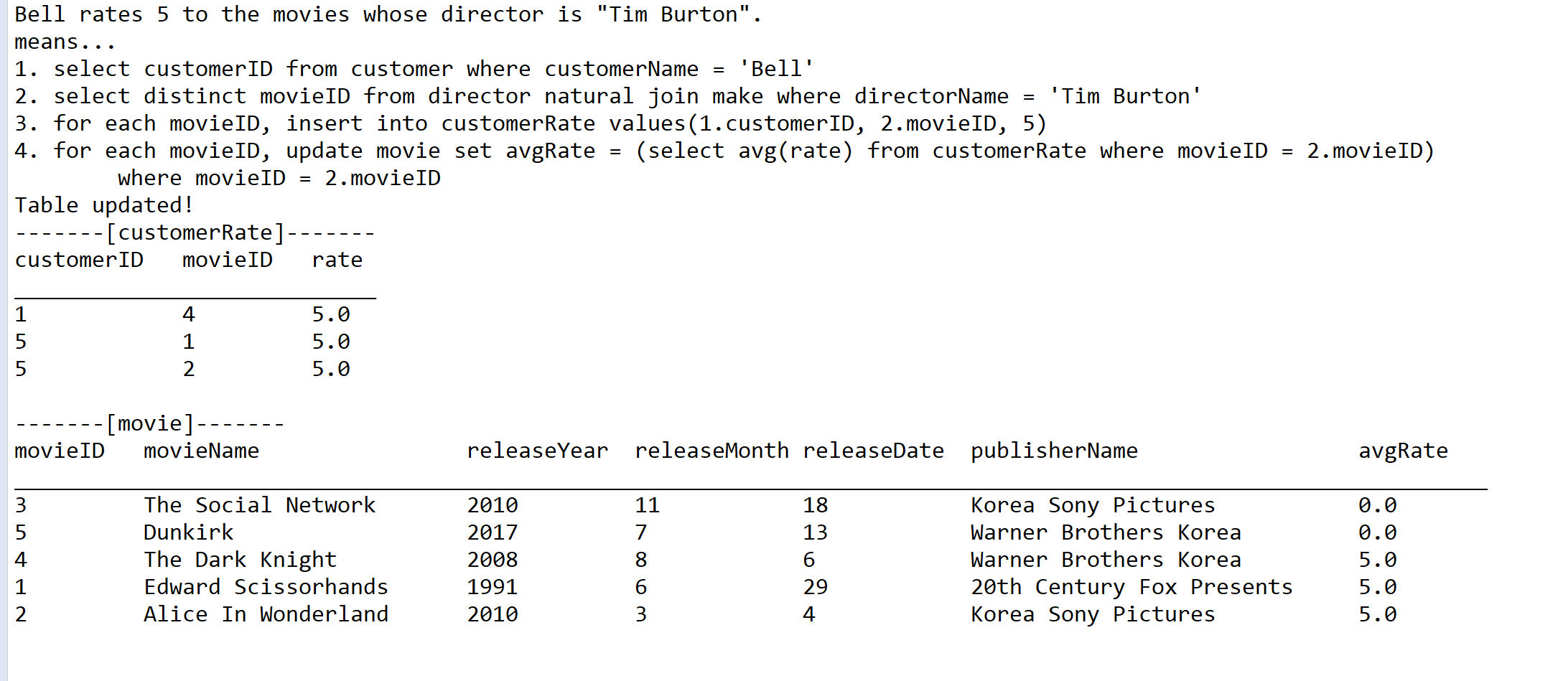


3. Insert data. 여기서는 문제마다 customer에서 해당 소비자의 ID를 찾고, movie에서 해당조건에 부합하는 movieID를 찾아(문제마다 다름) customerRate에 새로운 데이터를 insert한 뒤, movie의 각 movieID마다 avgRate값을 update하는 과정을 반복했다.

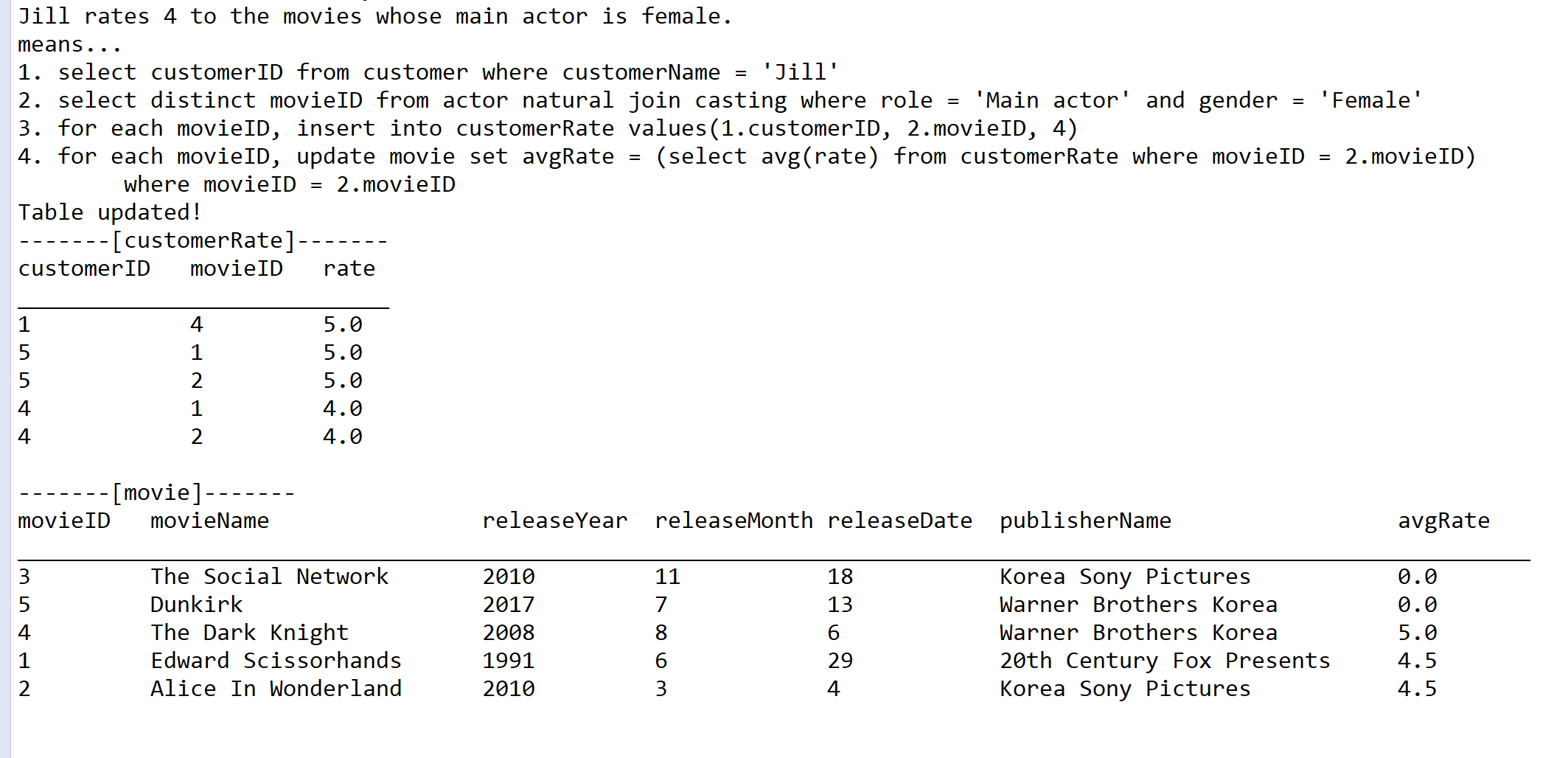
3.1실행결과출력

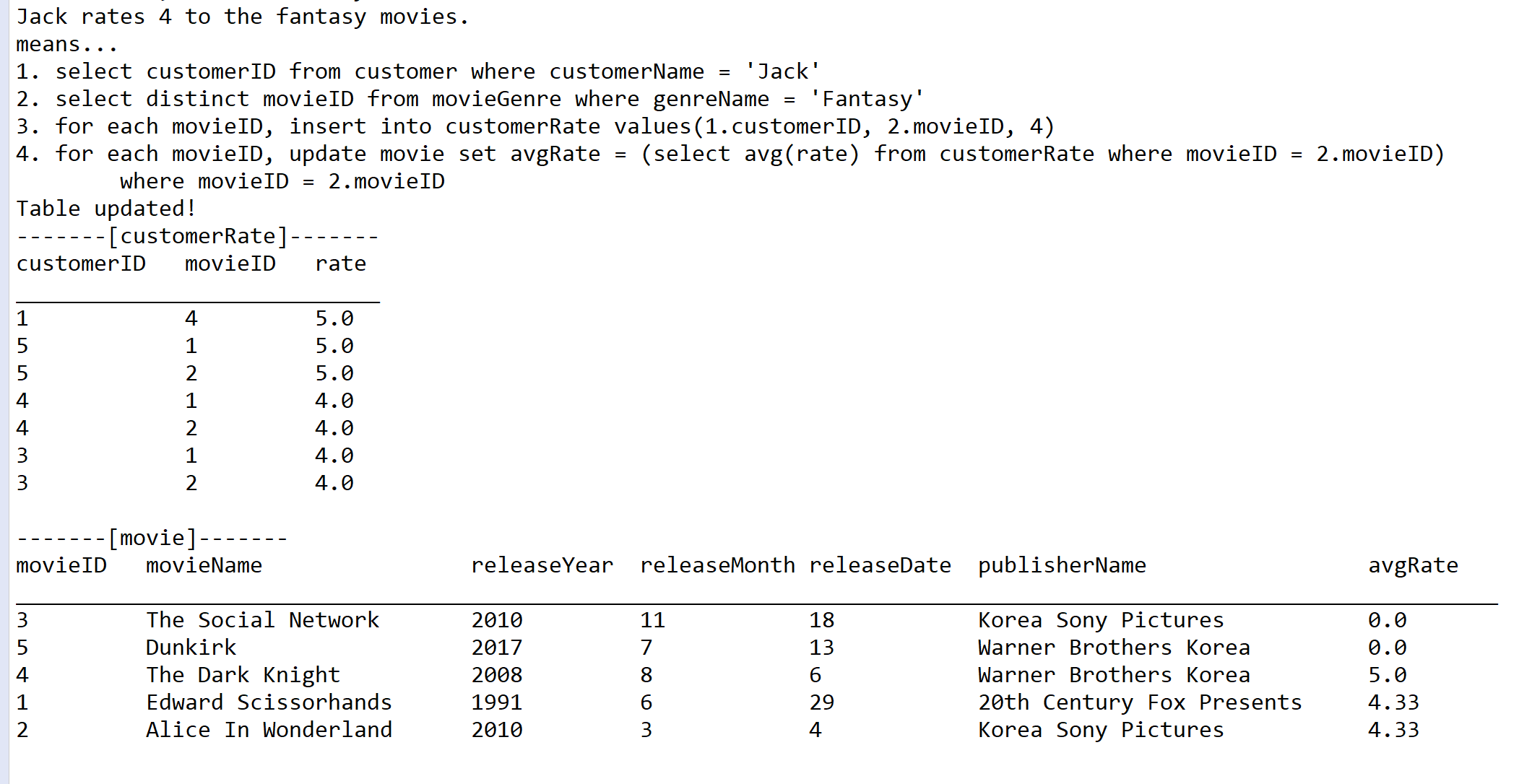


3.2실행결과

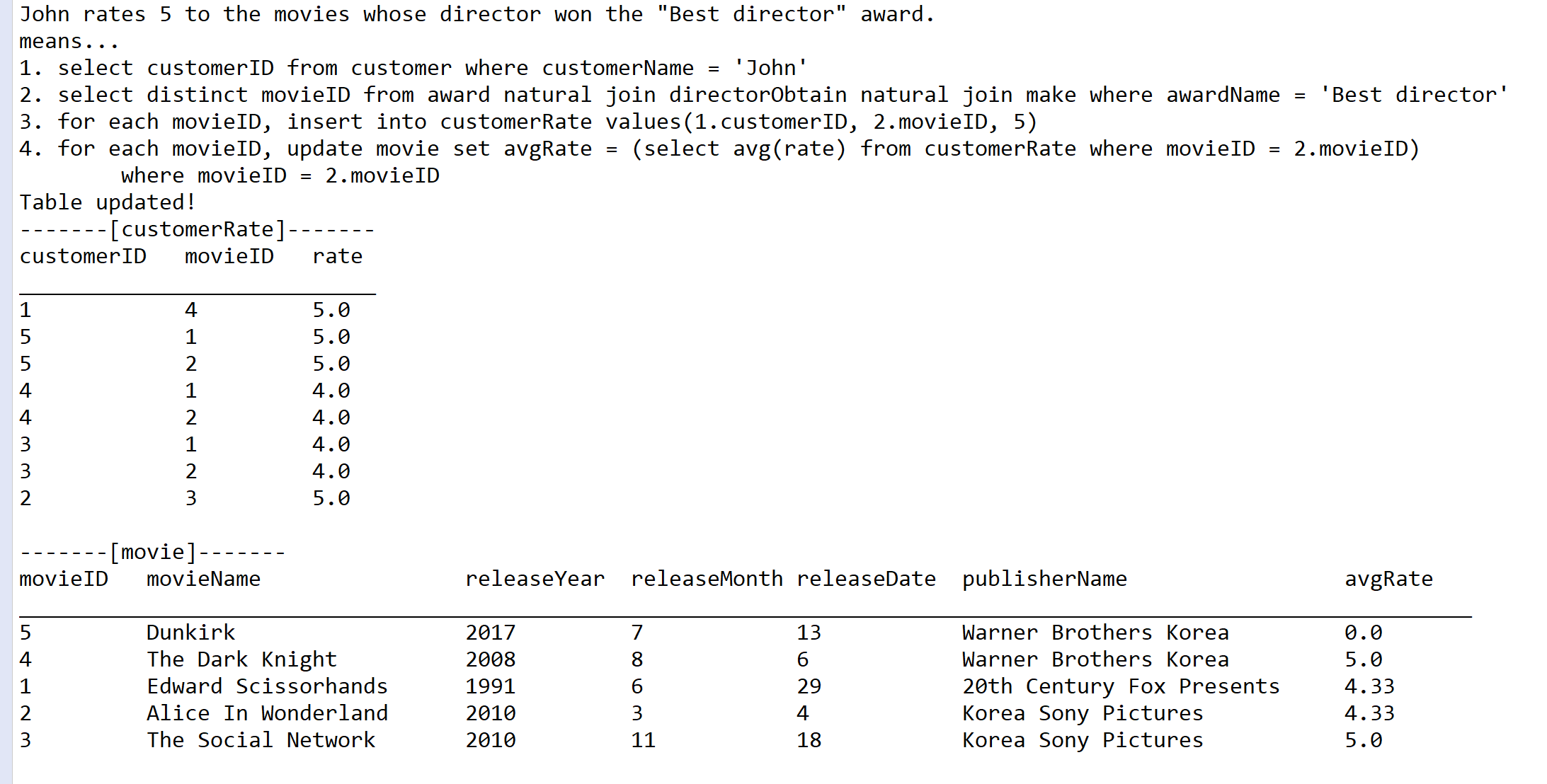


3.3실행결과



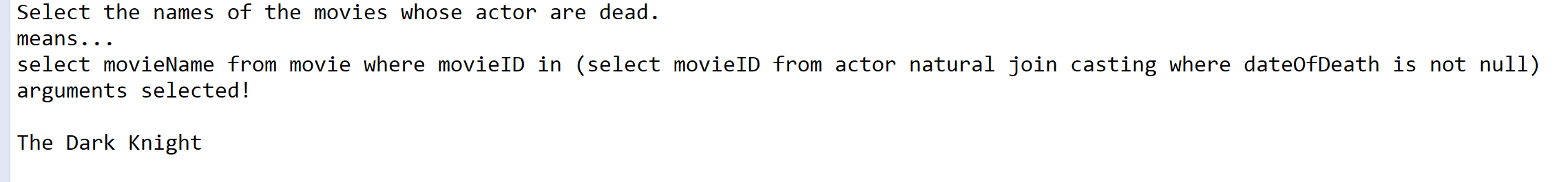
3.4실행결과

3.5실행결과



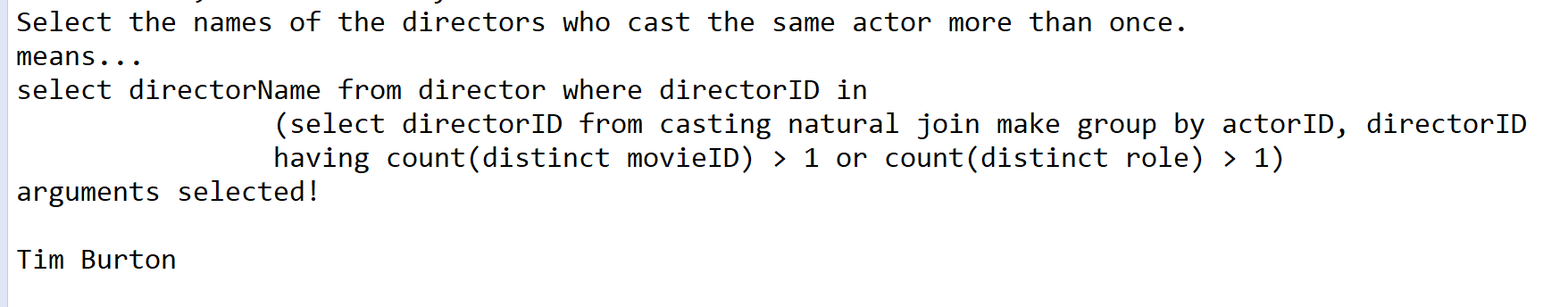
4. selection

movieName이 필요하므로 movie가 필요하고, dateOfDeath가 null인 actor의 movieID를 찾아야하므로 actor natural join casting을 하였다. 이렇게 찾아진 movieID에 movie의 movieID가 있는지 찾아야 하므로 where절에 in을 사용하였다.



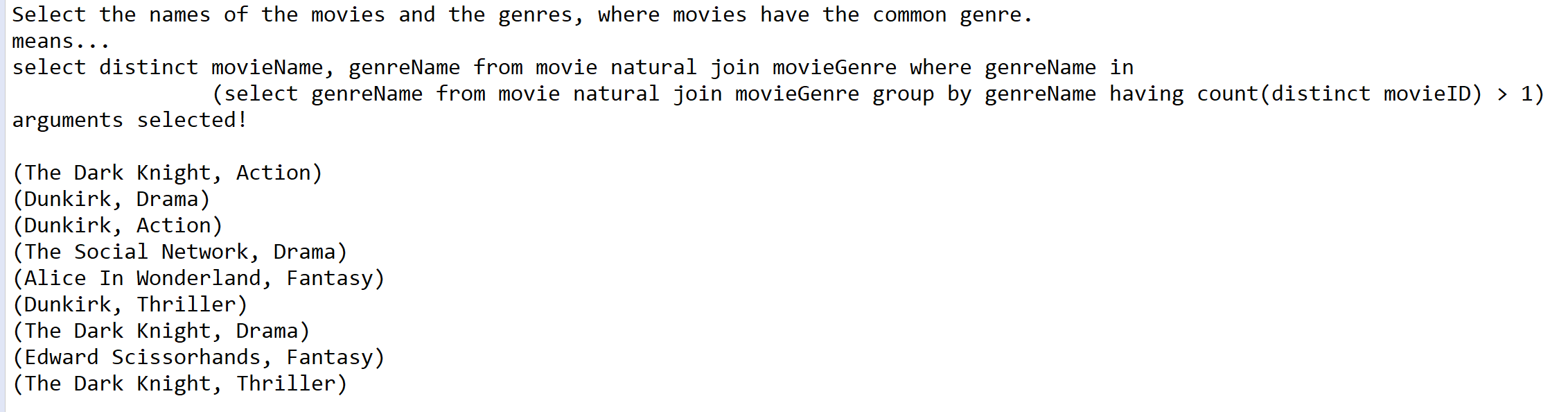
5. selection.

actorID, directorID로 group by했을 때 서로 다른 movie나 서로 다른 role이 존재하면 directors가 같은 actor을 2번이상 캐스팅 한 것이므로 그 때의 directorID로 director에서 directorName을 찾았다.



6. selection

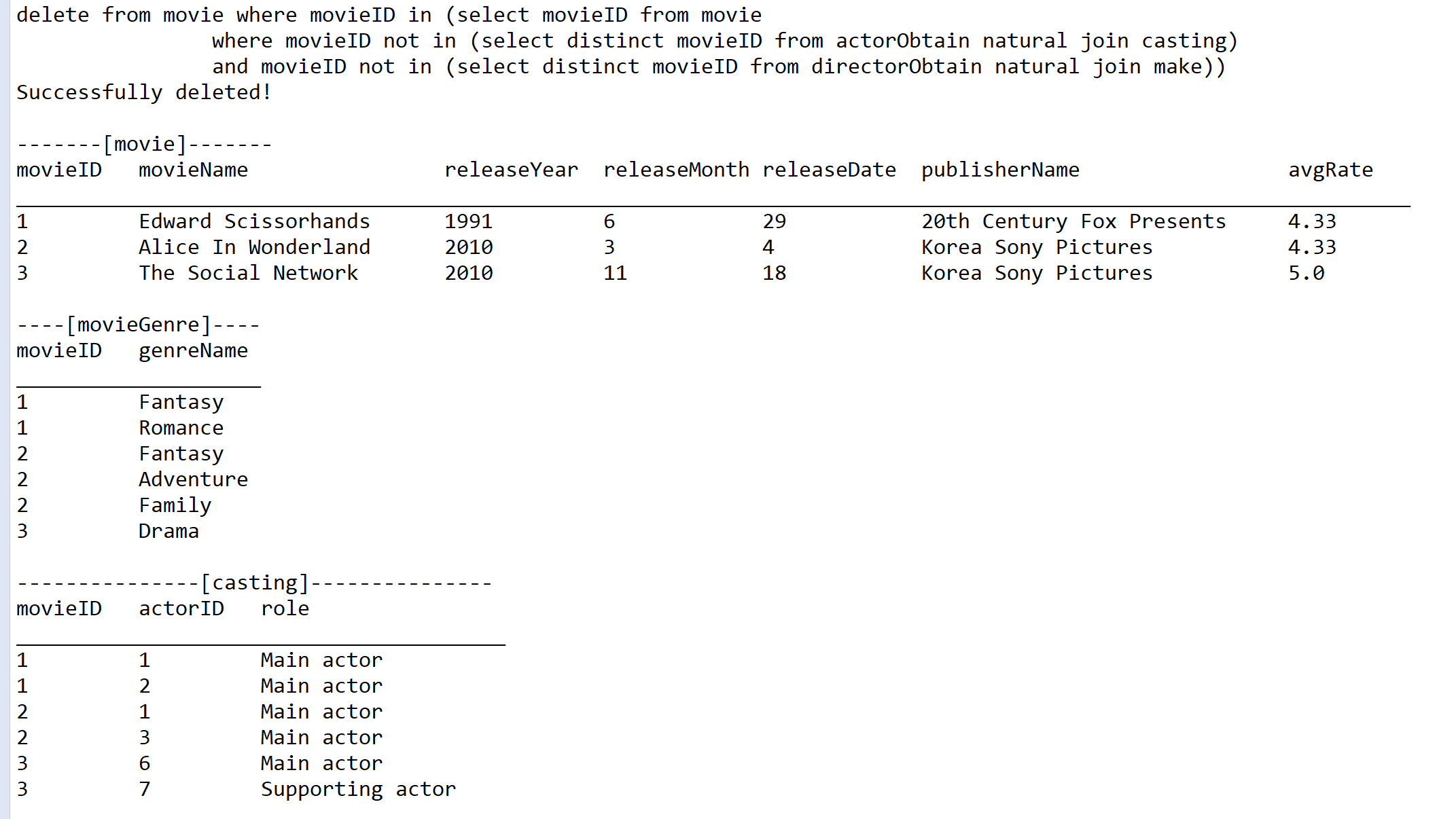
genreName으로 group by를 해서 각 genreName마다 서로다른 movieID가 2개이상이면 movies have the common genre이므로 이때의 genrename으로 movie natural join movieGenre에서 movieName, genreName을 찾으면 된다.



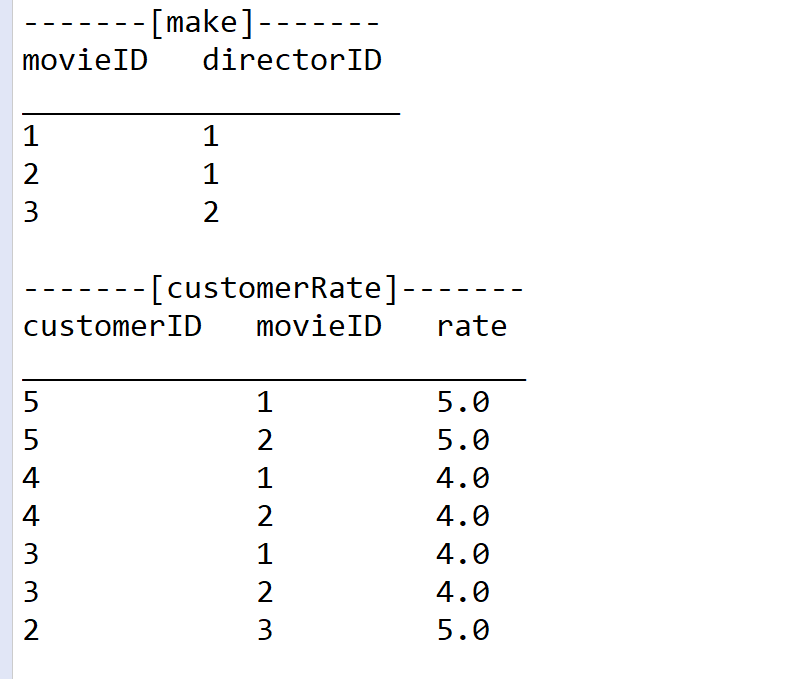
7. deletion

(Movie의 movieID) – ((상받은 actorID의 movieID) and (상받은 directorID의 movieID))이므로 이를

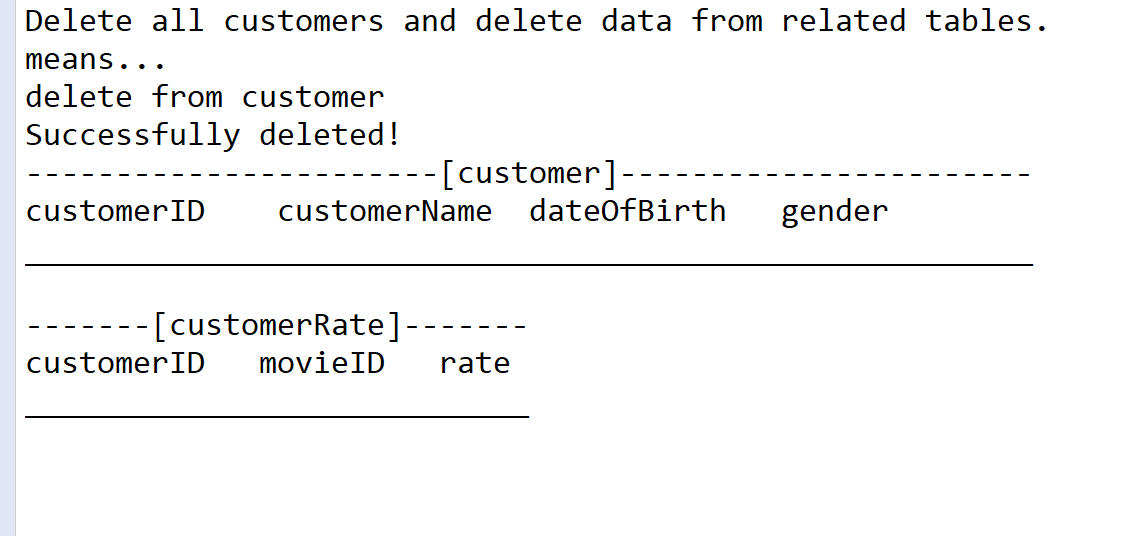
-를 not in으로 표현하면 아래와 같이 된다.



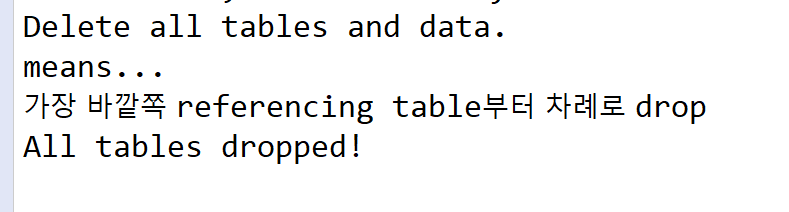
아래에 나머지 표 계속됨



8. deletion



9. drop table



[5] 1,2,3 번은 비교적 쉽게 수행 가능하였으나 4,5,6,7번에서는 지시문이 sql로 해석하기에 굉장히 함축적이라고 느껴 어려움이 있었다. 한 지시문을 수행하기위해 수업시간과 강의자료에서 다루었던 단순한 예시들 여러 개를 복합적으로 고려하여 만들어내야했기 때문에(subquery를 이중으로 쓴다던지, subquery와 group by와 having을 모두 사용한다던지) 꽤나 시간이 걸렸다. 그렇지만 이 과정 때문에 정말 의미있는 프로젝트가 되었고 앞으로 이런식으로 연습해 볼 수 있겠다는 자신감을 얻었다. 한가지 걸리는 것은 반복적인 과정들이 많아(같은table 뒤에도 계속출력 등) 이를 매서드로 만드는 것을 시도해보았지만 실패하였다는 것이다. 이 점이 아쉽기는 하지만 계속 오류를 해결하는 과정에서 알게된 게 많았다.

[6] [hyunjin2098@gmail.com](mailto:hyunjin2098@gmail.com)이나 hyunjin2098@naver.com.