[2019-02학기 데이터베이스과제보고서]

제목 : 영화 목록 관리 시스템

INDEX

과목: 데이터베이스
분반: 01분반
담당교수: 정현숙
제출자: 구명회
학번: 20154215
제출일자: 2019.12.22.
학과: 컴퓨터공학과

1, 2. 개발동기, 개발 설계 목적

개발하게 된 동기는 영화관에서 상영하는 영화 목록을 관리하는 것을 편하게 하기 위함이고,
설계목적은 개발동기에 대한 프로그램을 구현하는것입니다.

3. 개발환경, 개발일정

- 시스템 개발 환경은 다음과 같습니다.

  Window10 home edition
  Oracle 11g expression edition
  java 13.0.1

- 시스템 개발 일정은 다음과 같습니다.

  12.05 개발 동기 및 목적 기술, 일정 설계, 요구사항 분석, 논리적 설계를 통한 데이터 모델링
  12.06 물리적 설계를 통한 데이터베이스 상태기술 및 정규화수행 후 데이터베이스 테이블 작성과 데이터 삽입
  12.19 물리적 설계를 통한 데이터베이스 상태기술 및 정규화수행 후 데이터베이스 테이블 작성과 데이터 삽입
  12.20 데이터베이스에 저장된 영화관리 시스템을 자바 언어로 구현
  12.21 미흡한점 보완

4. 요구사항 분석

ERwin과 Oracle 11g expression edition을 사용해서 영화관 데이터베이스를 구현하고
Eclipse를 사용해서 자바언어로 어플레이케이션을 만드는것입니다.

5. 논리적 설계

- 데이터모델링 7단계

  업무분석 > 개체도출 > 속성도출 > 속성중 식별자 도출 > 관계 설정 > 관계 차수 설정 > 관계 식별 파악

1) 업무분석

이번 과제에서 영화의 기본정보, 영화를 제작한 감독, 영화를 찍은 배우들, 영화의 장르들과 타입, 등급등을 알 수 있는 정보들로 구성되는 영화관 데이터베이스를 만들어야합니다.
영화 기본정보에서는 각 영화들을 구분할 수 있는 고유코드와 영화 이름, 영화가 언제 개봉했는지, 영화를 시청한 관객의 수, 영화의 예매율과 평점, 영화를 홍보하기 위한 홈페이지 등을 알 수 있습니다.
영화 감독정보에서는 감독을 구분할 수 있는 고유 코드와 감독의 이름을 알 수 있습니다.
영화 배우정보에서는 배우를 구분할 수 있는 고유 코드와 배우의 이름을 알 수 있습니다.
영화 장르정보에서는 장르를 구분할 수 있는 고유 코드와 장르의 이름을 알 수 있습니다.
영화 등급정보에서는 등급을 구분할 수 있는 고유 코드와 등급의 이름을 알 수 있습니다.

2) 개체도출

영화 기본정보, 감독 기본정보, 배우 기본정보, 장르 기본정보, 타입 기본정보, 등급 기본정보

3) 속성도출

영화 기본정보 - 영화 고유코드, 영화 이름, 영화 개봉일, 관객수, 예매율, 평점, 영화 홈페이지
감독 기본정보 - 감독 고유코드, 감독 이름
배우 기본정보 - 배우 고유코드, 배우 이름
장르 기본정보 - 장르 고유코드, 장르 이름
타입 기본정보 - 타입 고유코드, 타입 이름
등급 기본정보 - 등급 고유코드, 등급 이름

4) 속성중 식별자 도출

영화 기본정보 - 영화 고유코드
감독 기본정보 - 감독 고유코드
배우 기본정보 - 배우 고유코드
장르 기본정보 - 장르 고유코드
타입 기본정보 - 타입 고유코드
등급 기본정보 - 등급 고유코드

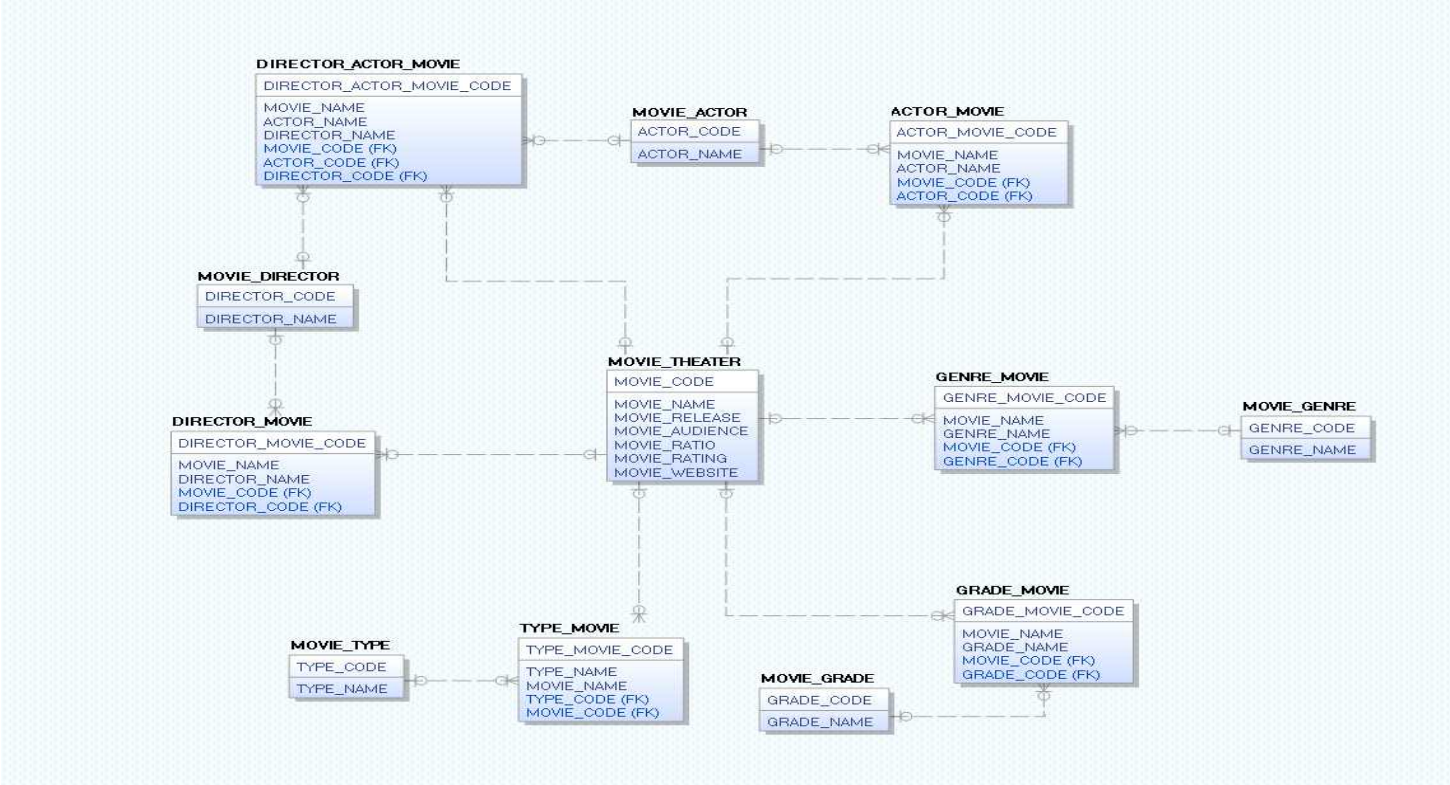
5, 6) 관계설정, 관계차수 설정

영화는 여러명의 감독에게 만들어질 수 있고, 감독은 여러개의 영화를 찍을 수 있습니다. N:M
영화는 여러명의 배우가 참여할 수 있고, 배우는 여러개의 영화를 찍을 수 있습니다. N:M
영화는 하나의 장르를 가지고, 장르는 여러개의 영화에게 선택될 수 있습니다. 1:N
영화는 하나의 타입을 가지고, 타입은 여러개의 영화에게 선택될 수 있습니다. 1:N
영화는 하나의 등급을 가지고, 등급은 여러개의 영화에게 선택될 수 있습니다. 1:N

7) 관계 식별 파악

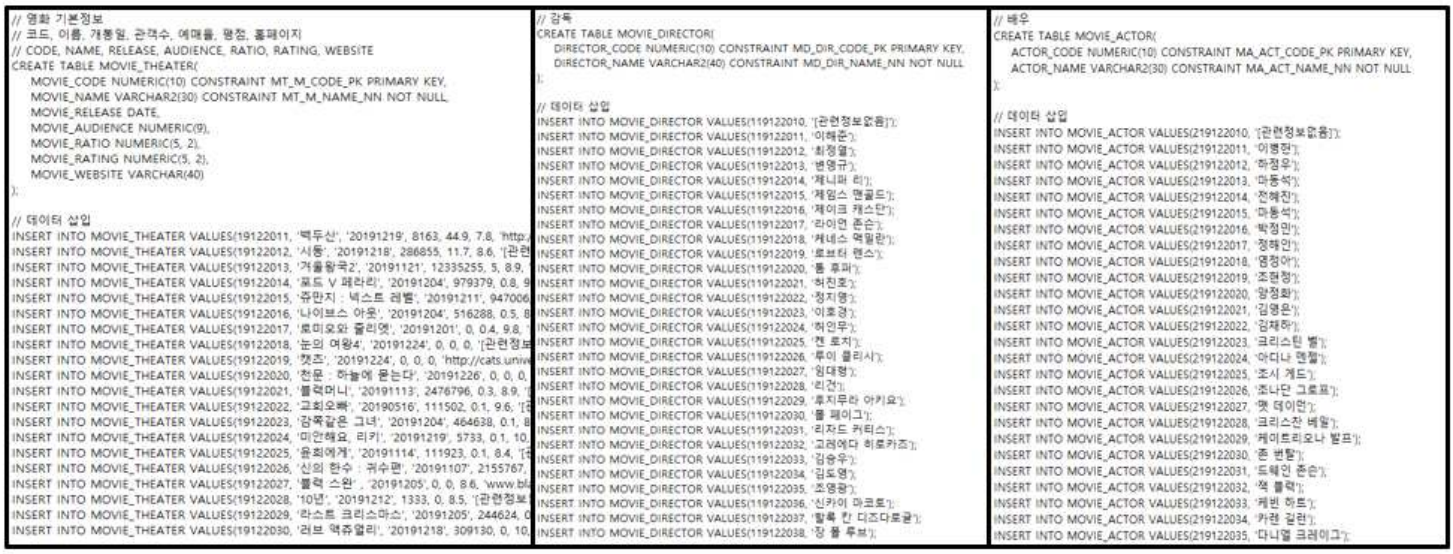
ERWin에서 표현할 모든 관계는 비식별관계입니다.

- ERWin 작성



6. 물리적 설계

1, 2) 스키마 / 데이터베이스 상태 > 스키마/데이터베이스 상태.zip 첨부했습니다.



```
// 영화 기본정보
// 코드, 이름, 개봉일, 관객수, 예매율, 평점, 홈페이지
// CODE, NAME, RELEASE, AUDIENCE, RATIO, RATING, WEBSITE
CREATE TABLE MOVIE_THEATER(
    MOVIE_CODE NUMERIC(10) CONSTRAINT MT_M_CODE_PK PRIMARY KEY,
    MOVIE_NAME VARCHAR2(30) CONSTRAINT MT_M_NAME_NN NOT NULL,
    MOVIE_RELEASE DATE,
    MOVIE_AUDIENCE NUMERIC(9),
    MOVIE_RATIO NUMERIC(5, 2),
    MOVIE_RATING NUMERIC(5, 2),
    MOVIE_WEBSITE VARCHAR(40)
);

// 데이터 삽입
INSERT INTO MOVIE_THEATER VALUES(19122011, '백두산', '20191219', 8163, 44.9, 7.8, 'http:/
INSERT INTO MOVIE_THEATER VALUES(19122012, '시동', '20191218', 286855, 11.7, 8.6, '[관련
INSERT INTO MOVIE_THEATER VALUES(19122013, '겨울왕국2', '20191121', 12335255, 5, 8.9,
INSERT INTO MOVIE_THEATER VALUES(19122014, '포드 V 패라리', '20191204', 979379, 0.8, 9
INSERT INTO MOVIE_THEATER VALUES(19122015, '쥬만지 : 넥스트 레벨', '20191211', 947006
INSERT INTO MOVIE_THEATER VALUES(19122016, '나이브스 아웃', '20191204', 516288, 0.5, 8
INSERT INTO MOVIE_THEATER VALUES(19122017, '로미오와 줄리엣', '20191201', 0, 0.4, 9.8, '
INSERT INTO MOVIE_THEATER VALUES(19122018, '눈의 여왕4', '20191224', 0, 0, 0, '[관련정보
INSERT INTO MOVIE_THEATER VALUES(19122019, '캣츠', '20191224', 0, 0, 0, 'http://cats.univ
INSERT INTO MOVIE_THEATER VALUES(19122020, '천문 : 하늘에 묻는다', '20191226', 0, 0, 0,
INSERT INTO MOVIE_THEATER VALUES(19122021, '블랙머니', '20191113', 2476796, 0.3, 8.9, '
INSERT INTO MOVIE_THEATER VALUES(19122022, '교회오빠', '20190516', 111502, 0.1, 9.6, '[
INSERT INTO MOVIE_THEATER VALUES(19122023, '강쪽같은 그녀', '20191204', 464638, 0.1, 8
INSERT INTO MOVIE_THEATER VALUES(19122024, '미안해요, 리키', '20191219', 5733, 0.1, 10,
INSERT INTO MOVIE_THEATER VALUES(19122025, '윤희에게', '20191114', 111923, 0.1, 8.4, '[
INSERT INTO MOVIE_THEATER VALUES(19122026, '신의 한수 : 귀수편', '20191107', 2155767,
INSERT INTO MOVIE_THEATER VALUES(19122027, '블랙 스완', '20191205', 0, 0, 8.6, 'www.bl
INSERT INTO MOVIE_THEATER VALUES(19122028, '10년', '20191212', 1333, 0, 8.5, '[관련정보
INSERT INTO MOVIE_THEATER VALUES(19122029, '라스트 크리스마스', '20191205', 244624, 0
INSERT INTO MOVIE_THEATER VALUES(19122030, '러브 액츄얼리', '20191218', 309130, 0, 10,
```

```
// 감독
CREATE TABLE MOVIE_DIRECTOR(
    DIRECTOR_CODE NUMERIC(10) CONSTRAINT MD_DIR_CODE_PK PRIMARY KEY,
    DIRECTOR_NAME VARCHAR2(40) CONSTRAINT MD_DIR_NAME_NN NOT NULL
);

// 데이터 삽입
INSERT INTO MOVIE_DIRECTOR VALUES(119122010, '[관련정보없음]');
INSERT INTO MOVIE_DIRECTOR VALUES(119122011, '이해준');
INSERT INTO MOVIE_DIRECTOR VALUES(119122012, '최정열');
INSERT INTO MOVIE_DIRECTOR VALUES(119122013, '변영규');
INSERT INTO MOVIE_DIRECTOR VALUES(119122014, '제니퍼 리');
INSERT INTO MOVIE_DIRECTOR VALUES(119122015, '제임스 맨골드');
INSERT INTO MOVIE_DIRECTOR VALUES(119122016, '제이크 캐스단');
INSERT INTO MOVIE_DIRECTOR VALUES(119122017, '라이언 존슨');
INSERT INTO MOVIE_DIRECTOR VALUES(119122018, '케네스 맥밀란');
INSERT INTO MOVIE_DIRECTOR VALUES(119122019, '로브터 랜스');
INSERT INTO MOVIE_DIRECTOR VALUES(119122020, '톰 후퍼');
INSERT INTO MOVIE_DIRECTOR VALUES(119122021, '허진호');
INSERT INTO MOVIE_DIRECTOR VALUES(119122022, '정지영');
INSERT INTO MOVIE_DIRECTOR VALUES(119122023, '이호경');
INSERT INTO MOVIE_DIRECTOR VALUES(119122024, '허인무');
INSERT INTO MOVIE_DIRECTOR VALUES(119122025, '첸 로지');
INSERT INTO MOVIE_DIRECTOR VALUES(119122026, '루이 클리사');
INSERT INTO MOVIE_DIRECTOR VALUES(119122027, '임대형');
INSERT INTO MOVIE_DIRECTOR VALUES(119122028, '리건');
INSERT INTO MOVIE_DIRECTOR VALUES(119122029, '후지무라 아키요');
INSERT INTO MOVIE_DIRECTOR VALUES(119122030, '톨 페이그');
INSERT INTO MOVIE_DIRECTOR VALUES(119122031, '리자드 커티스');
INSERT INTO MOVIE_DIRECTOR VALUES(119122032, '고레에다 히로카즈');
INSERT INTO MOVIE_DIRECTOR VALUES(119122033, '김승우');
INSERT INTO MOVIE_DIRECTOR VALUES(119122034, '김도영');
INSERT INTO MOVIE_DIRECTOR VALUES(119122035, '조영광');
INSERT INTO MOVIE_DIRECTOR VALUES(119122036, '신카이 마코토');
INSERT INTO MOVIE_DIRECTOR VALUES(119122037, '할록 칸 디즈다로글');
INSERT INTO MOVIE_DIRECTOR VALUES(119122038, '장 폴 루브');
```

```
// 배우
CREATE TABLE MOVIE_ACTOR(
    ACTOR_CODE NUMERIC(10) CONSTRAINT MA_ACT_CODE_PK PRIMARY KEY,
    ACTOR_NAME VARCHAR2(30) CONSTRAINT MA_ACT_NAME_NN NOT NULL
);

// 데이터 삽입
INSERT INTO MOVIE_ACTOR VALUES(219122010, '[관련정보없음]');
INSERT INTO MOVIE_ACTOR VALUES(219122011, '이병헌');
INSERT INTO MOVIE_ACTOR VALUES(219122012, '하정우');
INSERT INTO MOVIE_ACTOR VALUES(219122013, '마동석');
INSERT INTO MOVIE_ACTOR VALUES(219122014, '전혜진');
INSERT INTO MOVIE_ACTOR VALUES(219122015, '마동석');
INSERT INTO MOVIE_ACTOR VALUES(219122016, '박정민');
INSERT INTO MOVIE_ACTOR VALUES(219122017, '정해인');
INSERT INTO MOVIE_ACTOR VALUES(219122018, '염정아');
INSERT INTO MOVIE_ACTOR VALUES(219122019, '조한철');
INSERT INTO MOVIE_ACTOR VALUES(219122020, '양정화');
INSERT INTO MOVIE_ACTOR VALUES(219122021, '김영은');
INSERT INTO MOVIE_ACTOR VALUES(219122022, '김재하');
INSERT INTO MOVIE_ACTOR VALUES(219122023, '크리스틴 벨');
INSERT INTO MOVIE_ACTOR VALUES(219122024, '아디나 멘젤');
INSERT INTO MOVIE_ACTOR VALUES(219122025, '조시 게드');
INSERT INTO MOVIE_ACTOR VALUES(219122026, '조나단 그로프');
INSERT INTO MOVIE_ACTOR VALUES(219122027, '앳 데이먼');
INSERT INTO MOVIE_ACTOR VALUES(219122028, '크리스찬 베일');
INSERT INTO MOVIE_ACTOR VALUES(219122029, '케이트리오나 발프');
INSERT INTO MOVIE_ACTOR VALUES(219122030, '존 번할');
INSERT INTO MOVIE_ACTOR VALUES(219122031, '드웨인 존슨');
INSERT INTO MOVIE_ACTOR VALUES(219122032, '잭 블랙');
INSERT INTO MOVIE_ACTOR VALUES(219122033, '케빈 하트');
INSERT INTO MOVIE_ACTOR VALUES(219122034, '카렌 길란');
INSERT INTO MOVIE_ACTOR VALUES(219122035, '다니엘 크레이그');
```

```
// 감독 - 영화
CREATE TABLE DIRECTOR_MOVIE(
    DIRECTOR_MOVIE_CODE NUMERIC(10),
    MOVIE_CODE NUMERIC(10),
    MOVIE_NAME VARCHAR2(30) CONSTRAINT DM_M_NAME_NN NOT NULL,
    DIRECTOR_CODE NUMERIC(10),
    DIRECTOR_NAME VARCHAR2(40) CONSTRAINT DM_DIR_NAME_NN NOT NULL,
    CONSTRAINT DM_DIR_MOVIE_CODE_PK PRIMARY KEY(DIRECTOR_MOVIE_CODE),
    CONSTRAINT DM_M_CODE_FK FOREIGN KEY(MOVIE_CODE) REFERENCES MOVIE_THEATER(MOVIE_CODE),
    CONSTRAINT DM_DIR_CODE_FK FOREIGN KEY(DIRECTOR_CODE) REFERENCES MOVIE_DIRECTOR(DIRECTOR_CODE)
);

// 데이터 삽입
INSERT INTO DIRECTOR_MOVIE VALUES(619122011, 19122011, '백두산', 119122011, '이해준');
INSERT INTO DIRECTOR_MOVIE VALUES(619122012, 19122012, '시동', 119122012, '최정열');
INSERT INTO DIRECTOR_MOVIE VALUES(619122013, 19122013, '겨울왕국2', 119122014, '제니퍼 리');
INSERT INTO DIRECTOR_MOVIE VALUES(619122014, 19122014, '포드 V 패라리', 119122015, '제임스 맨골드');
INSERT INTO DIRECTOR_MOVIE VALUES(619122015, 19122015, '쥬만지 : 넥스트 레벨', 119122016, '제이크 캐스단');
INSERT INTO DIRECTOR_MOVIE VALUES(619122016, 19122016, '나이브스 아웃', 119122017, '라이언 존슨');
INSERT INTO DIRECTOR_MOVIE VALUES(619122017, 19122017, '로미오와 줄리엣', 119122018, '케네스 맥밀란');
INSERT INTO DIRECTOR_MOVIE VALUES(619122018, 19122018, '눈의 여왕4', 119122019, '로브터 랜스');
INSERT INTO DIRECTOR_MOVIE VALUES(619122019, 19122019, '캣츠', 119122020, '톰 후퍼');
INSERT INTO DIRECTOR_MOVIE VALUES(619122020, 19122020, '천문 : 하늘에 묻는다', 119122021, '허진호');
INSERT INTO DIRECTOR_MOVIE VALUES(619122021, 19122021, '블랙머니', 119122022, '정지영');
INSERT INTO DIRECTOR_MOVIE VALUES(619122022, 19122022, '교회오빠', 119122023, '이호경');
INSERT INTO DIRECTOR_MOVIE VALUES(619122023, 19122023, '강쪽같은 그녀', 119122024, '허인무');
INSERT INTO DIRECTOR_MOVIE VALUES(619122024, 19122024, '미안해요, 리키', 119122025, '첸 로지');
INSERT INTO DIRECTOR_MOVIE VALUES(619122025, 19122025, '윤희에게', 119122027, '임대형');
INSERT INTO DIRECTOR_MOVIE VALUES(619122026, 19122026, '신의 한수 : 귀수편', 119122028, '리건');
INSERT INTO DIRECTOR_MOVIE VALUES(619122027, 19122027, '블랙 스완', 119122010, '[관련정보없음]');
INSERT INTO DIRECTOR_MOVIE VALUES(619122028, 19122028, '10년', 119122029, '후지무라 아키요');
INSERT INTO DIRECTOR_MOVIE VALUES(619122029, 19122029, '라스트 크리스마스', 119122030, '톨 페이그');
INSERT INTO DIRECTOR_MOVIE VALUES(619122030, 19122030, '러브 액츄얼리', 119122031, '리자드 커티스');
```

```
// 배우 - 영화
CREATE TABLE ACTOR_MOVIE(
    ACTOR_MOVIE_CODE NUMERIC(10),
    MOVIE_CODE NUMERIC(10),
    MOVIE_NAME VARCHAR2(30) CONSTRAINT AM_M_NAME_NN NOT NULL,
    ACTOR_CODE NUMERIC(10),
    ACTOR_NAME VARCHAR2(30) CONSTRAINT AM_ACT_NAME_NN NOT NULL,
    CONSTRAINT AM_ACT_M_CODE_PK PRIMARY KEY(ACTOR_MOVIE_CODE),
    CONSTRAINT AM_M_CODE_FK FOREIGN KEY(MOVIE_CODE) REFERENCES MOVIE_THEATER(MOVIE_CODE),
    CONSTRAINT AM_ACT_CODE_FK FOREIGN KEY(ACTOR_CODE) REFERENCES MOVIE_ACTOR(ACTOR_CODE)
);

// 데이터 삽입
INSERT INTO ACTOR_MOVIE VALUES(719122011, 19122011, '백두산', 219122011, '이병헌');
INSERT INTO ACTOR_MOVIE VALUES(719122012, 19122011, '백두산', 219122012, '하정우');
INSERT INTO ACTOR_MOVIE VALUES(719122013, 19122011, '백두산', 219122013, '마동석');
INSERT INTO ACTOR_MOVIE VALUES(719122014, 19122011, '백두산', 219122014, '전혜진');
INSERT INTO ACTOR_MOVIE VALUES(719122015, 19122012, '시동', 219122013, '마동석');
INSERT INTO ACTOR_MOVIE VALUES(719122016, 19122012, '시동', 219122016, '박정민');
INSERT INTO ACTOR_MOVIE VALUES(719122017, 19122012, '시동', 219122017, '정해인');
INSERT INTO ACTOR_MOVIE VALUES(719122018, 19122012, '시동', 219122018, '염정아');
INSERT INTO ACTOR_MOVIE VALUES(719122019, 19122013, '겨울왕국2', 219122023, '크리스틴 벨');
INSERT INTO ACTOR_MOVIE VALUES(719122020, 19122013, '겨울왕국2', 219122024, '아디나 멘젤');
INSERT INTO ACTOR_MOVIE VALUES(719122021, 19122013, '겨울왕국2', 219122025, '조시 게드');
INSERT INTO ACTOR_MOVIE VALUES(719122022, 19122013, '겨울왕국2', 219122026, '조나단 그로프');
INSERT INTO ACTOR_MOVIE VALUES(719122023, 19122014, '포드 V 패라리', 219122027, '앳 데이먼');
INSERT INTO ACTOR_MOVIE VALUES(719122024, 19122014, '포드 V 패라리', 219122028, '크리스찬 베일');
INSERT INTO ACTOR_MOVIE VALUES(719122025, 19122014, '포드 V 패라리', 219122029, '케이트리오나 발프');
INSERT INTO ACTOR_MOVIE VALUES(719122026, 19122014, '포드 V 패라리', 219122030, '존 번할');
INSERT INTO ACTOR_MOVIE VALUES(719122027, 19122015, '쥬만지 : 넥스트 레벨', 219122031, '드웨인 존슨');
INSERT INTO ACTOR_MOVIE VALUES(719122028, 19122015, '쥬만지 : 넥스트 레벨', 219122032, '잭 블랙');
INSERT INTO ACTOR_MOVIE VALUES(719122029, 19122015, '쥬만지 : 넥스트 레벨', 219122033, '케빈 하트');
INSERT INTO ACTOR_MOVIE VALUES(719122030, 19122015, '쥬만지 : 넥스트 레벨', 219122034, '카렌 길란');
INSERT INTO ACTOR_MOVIE VALUES(719122031, 19122016, '나이브스 아웃', 219122035, '다니엘 크레이그');
INSERT INTO ACTOR_MOVIE VALUES(719122032, 19122016, '나이브스 아웃', 219122036, '크리스 에반스');
INSERT INTO ACTOR_MOVIE VALUES(719122033, 19122016, '나이브스 아웃', 219122037, '아나 디 아르마스');
INSERT INTO ACTOR_MOVIE VALUES(719122034, 19122016, '나이브스 아웃', 219122038, '제이미 리 커티스');
INSERT INTO ACTOR_MOVIE VALUES(719122035, 19122017, '로미오와 줄리엣', 219122010, '[관련정보없음]');
INSERT INTO ACTOR_MOVIE VALUES(719122036, 19122018, '눈의 여왕4', 219122039, '양정화');
INSERT INTO ACTOR_MOVIE VALUES(719122037, 19122018, '눈의 여왕4', 219122040, '박지윤');
```

```
// 감독 - 배우 - 영화
CREATE TABLE DIRECTOR_ACTOR_MOVIE(
    DIRECTOR_ACTOR_MOVIE_CODE NUMERIC(10),
    MOVIE_CODE NUMERIC(10),
    MOVIE_NAME VARCHAR2(30) CONSTRAINT DAM_M_NAME_NN NOT NULL,
    DIRECTOR_CODE NUMERIC(10),
    DIRECTOR_NAME VARCHAR2(40) CONSTRAINT DAM_DIR_NAME_NN NOT NULL,
    ACTOR_CODE NUMERIC(10),
    ACTOR_NAME VARCHAR2(30) CONSTRAINT DAM_ACT_NAME_NN NOT NULL,
    CONSTRAINT DAM_DIR_ACT_M_CODE_PK PRIMARY KEY(DIRECTOR_ACTOR_MOVIE_CODE),
    CONSTRAINT DAM_M_CODE_FK FOREIGN KEY(MOVIE_CODE) REFERENCES MOVIE_THEATER(MOVIE_CODE),
    CONSTRAINT DAM_ACT_CODE_FK FOREIGN KEY(ACTOR_CODE) REFERENCES MOVIE_ACTOR(ACTOR_CODE),
    CONSTRAINT DAM_DIR_CODE_FK FOREIGN KEY(DIRECTOR_CODE) REFERENCES MOVIE_DIRECTOR(DIRECTOR_CODE)
);

// 데이터 삽입
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122011, 19122011, '백두산', 119122011, '이해준', 219122011, '이병헌');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122012, 19122011, '백두산', 119122011, '이해준', 219122012, '하정우');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122013, 19122011, '백두산', 119122011, '이해준', 219122013, '마동석');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122014, 19122011, '백두산', 119122011, '이해준', 219122014, '전혜진');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122015, 19122012, '시동', 119122012, '최정열', 219122013, '마동석');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122016, 19122012, '시동', 119122012, '최정열', 219122016, '박정민');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122017, 19122012, '시동', 119122012, '최정열', 219122017, '정해인');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122018, 19122012, '시동', 119122012, '최정열', 219122018, '염정아');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122019, 19122013, '겨울왕국2', 119122014, '제니퍼 리', 219122023, '크리스틴 벨');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122020, 19122013, '겨울왕국2', 119122014, '제니퍼 리', 219122024, '아디나 멘젤');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122021, 19122013, '겨울왕국2', 119122014, '제니퍼 리', 219122025, '조시 게드');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122022, 19122013, '겨울왕국2', 119122014, '제니퍼 리', 219122026, '조나단 그로프');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122023, 19122014, '포드 V 페라리', 119122015, '제임스 맨골드', 219122027, '맷 데이먼');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122024, 19122014, '포드 V 페라리', 119122015, '제임스 맨골드', 219122028, '크리스찬 베일');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122025, 19122014, '포드 V 페라리', 119122015, '제임스 맨골드', 219122029, '케이트리오나 발프');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122026, 19122014, '포드 V 페라리', 119122015, '제임스 맨골드', 219122030, '존 반탈');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122027, 19122015, '쥬만지 : 넥스트 레벨', 119122016, '제이크 캐스단', 219122031, '드웨인 존슨');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122028, 19122015, '쥬만지 : 넥스트 레벨', 119122016, '제이크 캐스단', 219122032, '잭 블랙');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122029, 19122015, '쥬만지 : 넥스트 레벨', 119122016, '제이크 캐스단', 219122033, '케빈 하트');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122030, 19122015, '쥬만지 : 넥스트 레벨', 119122016, '제이크 캐스단', 219122034, '카렌 길런');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122031, 19122016, '나이브스 아웃', 119122017, '라이언 존슨', 219122035, '다니엘 크레이그');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122032, 19122016, '나이브스 아웃', 119122017, '라이언 존슨', 219122036, '크리스 에반스');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122033, 19122016, '나이브스 아웃', 119122017, '라이언 존슨', 219122037, '아나 디 아르마스');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122034, 19122016, '나이브스 아웃', 119122017, '라이언 존슨', 219122038, '제이미 리 커티스');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122035, 19122017, '로미오와 줄리엣', 119122018, '케네스 맥밀란', 219122010, '[관련정보없음]');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122036, 19122018, '눈의 여왕4', 119122019, '로브터 렌스', 219122039, '양정화');
INSERT INTO DIRECTOR_ACTOR_MOVIE VALUES(819122037, 19122018, '눈의 여왕4', 119122019, '로브터 렌스', 219122040, '박지윤');
```

```
// 장르
CREATE TABLE MOVIE_GENRE(
    GENRE_CODE NUMERIC(10) CONSTRAINT MG_GEN_CODE_PK PRIMARY KEY,
    GENRE_NAME VARCHAR2(25) CONSTRAINT MG_GEN_NAME_NN NOT NULL
);

// 데이터 삽입
INSERT INTO MOVIE_GENRE VALUES(319122011, '드라마(한국)');
INSERT INTO MOVIE_GENRE VALUES(319122012, '애니메이션(한국)');
INSERT INTO MOVIE_GENRE VALUES(319122013, '사극(한국)');
INSERT INTO MOVIE_GENRE VALUES(319122014, '다큐멘터리(한국)');
INSERT INTO MOVIE_GENRE VALUES(319122015, '멜로/로맨스(한국)');
INSERT INTO MOVIE_GENRE VALUES(319122016, '스릴러(한국)');
INSERT INTO MOVIE_GENRE VALUES(319122017, '드라마(미국)');
INSERT INTO MOVIE_GENRE VALUES(319122018, '애니메이션(미국)');
INSERT INTO MOVIE_GENRE VALUES(319122019, '액션(미국)');
INSERT INTO MOVIE_GENRE VALUES(319122020, '스릴러(미국)');
INSERT INTO MOVIE_GENRE VALUES(319122021, '멜로/로맨스(미국)');
INSERT INTO MOVIE_GENRE VALUES(319122022, '드라마(영국)');
INSERT INTO MOVIE_GENRE VALUES(319122023, '드라마(일본)');
INSERT INTO MOVIE_GENRE VALUES(319122024, '애니메이션(일본)');
INSERT INTO MOVIE_GENRE VALUES(319122025, '드라마(프랑스)');
INSERT INTO MOVIE_GENRE VALUES(319122026, '애니메이션(기타)');
```

```
// 타입
CREATE TABLE MOVIE_TYPE(
    TYPE_CODE NUMERIC(10) CONSTRAINT MT_T_CODE_PK PRIMARY KEY,
    TYPE_NAME VARCHAR2(15) CONSTRAINT MT_T_NAME_NN NOT NULL
);

// 데이터 삽입
INSERT INTO MOVIE_TYPE VALUES(419122011, '없음');
INSERT INTO MOVIE_TYPE VALUES(419122012, '더빙');
INSERT INTO MOVIE_TYPE VALUES(419122013, '자막');
INSERT INTO MOVIE_TYPE VALUES(419122014, '영어자막');
INSERT INTO MOVIE_TYPE VALUES(419122015, '더빙,자막');
```

```
// 등급
CREATE TABLE MOVIE_GRADE(
    GRADE_CODE NUMERIC(10) CONSTRAINT MG_GRA_CODE_PK PRIMARY KEY,
    GRADE_NAME VARCHAR2(25) CONSTRAINT MG_GRA_NAME_NN NOT NULL
);

// 데이터 삽입
INSERT INTO MOVIE_GRADE VALUES(519122011, '전체관람가');
INSERT INTO MOVIE_GRADE VALUES(519122012, '12세이상관람가');
INSERT INTO MOVIE_GRADE VALUES(519122013, '15세이상관람가');
INSERT INTO MOVIE_GRADE VALUES(519122014, '청소년관람불가');
```

```
// 장르 - 영화
CREATE TABLE GENRE_MOVIE(
    GENRE_MOVIE_CODE NUMERIC(10),
    MOVIE_CODE NUMERIC(10),
    MOVIE_NAME VARCHAR2(30) CONSTRAINT GENM_M_NAME_NN NOT NULL,
    GENRE_CODE NUMERIC(10),
    GENRE_NAME VARCHAR2(30) CONSTRAINT GENM_GENN_NAME_NN NOT NULL,
    CONSTRAINT GENM_GEN_M_CODE_PK PRIMARY KEY(GENRE_MOVIE_CODE),
    CONSTRAINT GENM_GENM_CODE_FK FOREIGN KEY(MOVIE_CODE) REFERENCES MOVIE_THEATER(MOVIE_CODE),
    CONSTRAINT GENM_GENG_CODE_FK FOREIGN KEY(GENRE_CODE) REFERENCES MOVIE_GENRE(GENRE_CODE)
);

// 데이터 삽입
INSERT INTO GENRE_MOVIE VALUES(919122011, 19122011, '백두산', 319122011, '드라마(한국)');
INSERT INTO GENRE_MOVIE VALUES(919122012, 19122012, '시동', 319122011, '드라마(한국)');
INSERT INTO GENRE_MOVIE VALUES(919122013, 19122013, '겨울왕국2', 319122018, '애니메이션(미국)');
INSERT INTO GENRE_MOVIE VALUES(919122014, 19122014, '포드 V 페라리', 319122017, '드라마(미국)');
INSERT INTO GENRE_MOVIE VALUES(919122015, 19122015, '쥬만지 : 넥스트 레벨', 319122019, '액션(미국)');
INSERT INTO GENRE_MOVIE VALUES(919122016, 19122016, '나이브스 아웃', 319122016, '스릴러(미국)');
INSERT INTO GENRE_MOVIE VALUES(919122017, 19122017, '로미오와 줄리엣', 319122022, '드라마(영국)');
INSERT INTO GENRE_MOVIE VALUES(919122018, 19122018, '눈의 여왕4', 319122026, '애니메이션(기타)');
INSERT INTO GENRE_MOVIE VALUES(919122019, 19122019, '캣츠', 319122017, '드라마(미국)');
INSERT INTO GENRE_MOVIE VALUES(919122020, 19122020, '천문 : 하늘에 묻는다', 319122013, '사극(한국)');
INSERT INTO GENRE_MOVIE VALUES(919122021, 19122021, '블랙머니', 319122011, '드라마(한국)');
INSERT INTO GENRE_MOVIE VALUES(919122022, 19122022, '교회오빠', 319122014, '다큐멘터리(한국)');
INSERT INTO GENRE_MOVIE VALUES(919122023, 19122023, '감쪽같은 그녀', 319122011, '드라마(한국)');
INSERT INTO GENRE_MOVIE VALUES(919122024, 19122024, '미안해요, 리키', 319122025, '드라마(프랑스)');
INSERT INTO GENRE_MOVIE VALUES(919122025, 19122025, '윤희에게', 319122015, '멜로/로맨스(한국)');
INSERT INTO GENRE_MOVIE VALUES(919122026, 19122026, '신의 한 수 : 귀수편', 319122016, '스릴러(한국)');
INSERT INTO GENRE_MOVIE VALUES(919122027, 19122027, '블랙 스완', 319122020, '스릴러(미국)');
INSERT INTO GENRE_MOVIE VALUES(919122028, 19122028, '10년', 319122023, '드라마(일본)');
INSERT INTO GENRE_MOVIE VALUES(919122029, 19122029, '라스트 크리스마스', 319122021, '멜로/로맨스(미국)');
INSERT INTO GENRE_MOVIE VALUES(919122030, 19122030, '러브 액츄얼리', 319122021, '멜로/로맨스(미국)');
```

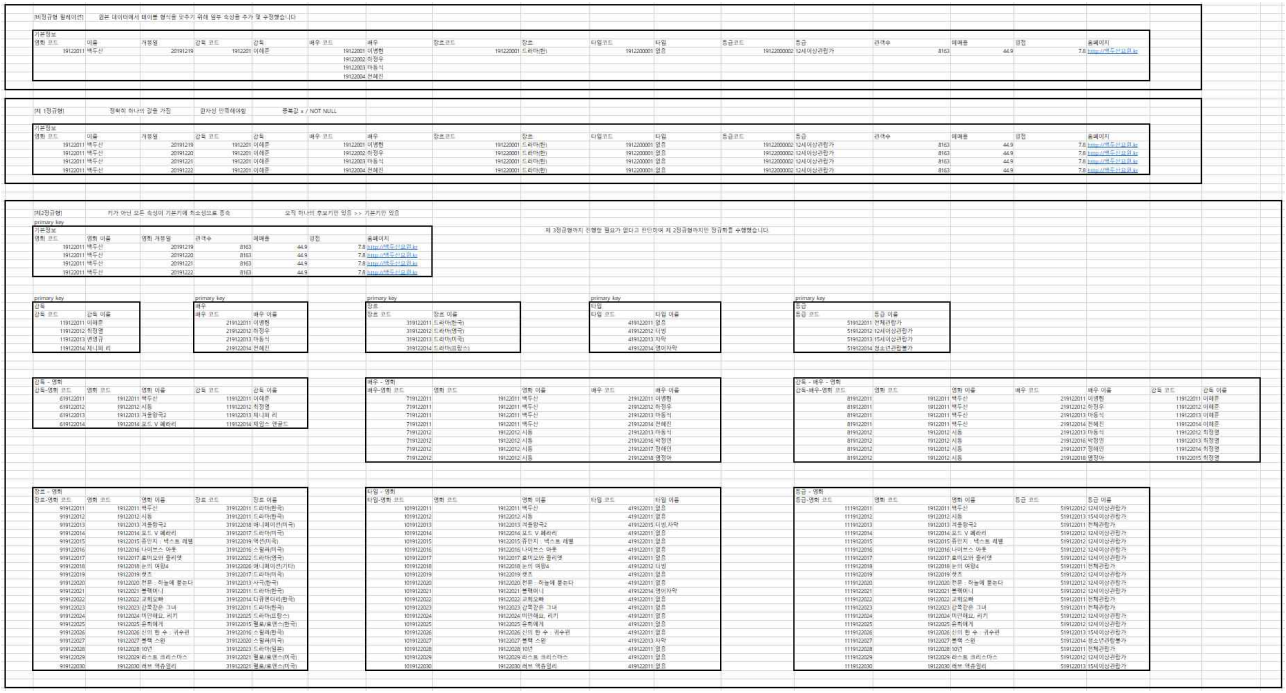
```
// 타입 - 영화
CREATE TABLE TYPE_MOVIE(
    TYPE_MOVIE_CODE NUMERIC(10),
    MOVIE_CODE NUMERIC(10),
    MOVIE_NAME VARCHAR2(30) CONSTRAINT TM_M_NAME_NN NOT NULL,
    TYPE_CODE NUMERIC(10),
    TYPE_NAME VARCHAR2(15) CONSTRAINT TM_T_NAME_NN NOT NULL,
    CONSTRAINT TM_T_M_CODE_PK PRIMARY KEY(TYPE_MOVIE_CODE),
    CONSTRAINT TM_M_CODE_FK FOREIGN KEY(MOVIE_CODE) REFERENCES MOVIE_THEATER(MOVIE_CODE),
    CONSTRAINT TM_T_CODE_FK FOREIGN KEY(TYPE_CODE) REFERENCES MOVIE_TYPE(TYPE_CODE)
);

// 데이터 삽입
INSERT INTO TYPE_MOVIE VALUES(1019122011, 19122011, '백두산', 419122011, '없음');
INSERT INTO TYPE_MOVIE VALUES(1019122012, 19122012, '시동', 419122011, '없음');
INSERT INTO TYPE_MOVIE VALUES(1019122013, 19122013, '겨울왕국2', 419122015, '더빙,자막');
INSERT INTO TYPE_MOVIE VALUES(1019122014, 19122014, '포드 V 페라리', 419122011, '없음');
INSERT INTO TYPE_MOVIE VALUES(1019122015, 19122015, '쥬만지 : 넥스트 레벨', 419122011, '없음');
INSERT INTO TYPE_MOVIE VALUES(1019122016, 19122016, '나이브스 아웃', 419122011, '없음');
INSERT INTO TYPE_MOVIE VALUES(1019122017, 19122017, '로미오와 줄리엣', 419122011, '없음');
INSERT INTO TYPE_MOVIE VALUES(1019122018, 19122018, '눈의 여왕4', 419122012, '더빙');
INSERT INTO TYPE_MOVIE VALUES(1019122019, 19122019, '캣츠', 419122011, '없음');
INSERT INTO TYPE_MOVIE VALUES(1019122020, 19122020, '천문 : 하늘에 묻는다', 419122011, '없음');
INSERT INTO TYPE_MOVIE VALUES(1019122021, 19122021, '블랙머니', 419122014, '영어자막');
INSERT INTO TYPE_MOVIE VALUES(1019122022, 19122022, '교회오빠', 419122011, '없음');
INSERT INTO TYPE_MOVIE VALUES(1019122023, 19122023, '감쪽같은 그녀', 419122011, '없음');
INSERT INTO TYPE_MOVIE VALUES(1019122024, 19122024, '미안해요, 리키', 419122011, '없음');
INSERT INTO TYPE_MOVIE VALUES(1019122025, 19122025, '윤희에게', 419122011, '없음');
INSERT INTO TYPE_MOVIE VALUES(1019122026, 19122026, '신의 한 수 : 귀수편', 419122011, '없음');
INSERT INTO TYPE_MOVIE VALUES(1019122027, 19122027, '블랙 스완', 419122013, '자막');
INSERT INTO TYPE_MOVIE VALUES(1019122028, 19122028, '10년', 419122011, '없음');
INSERT INTO TYPE_MOVIE VALUES(1019122029, 19122029, '라스트 크리스마스', 419122011, '없음');
INSERT INTO TYPE_MOVIE VALUES(1019122030, 19122030, '러브 액츄얼리', 419122011, '없음');
```

```
// 등급 - 영화
CREATE TABLE GRADE_MOVIE(
    GRADE_MOVIE_CODE NUMERIC(10),
    MOVIE_CODE NUMERIC(10),
    MOVIE_NAME VARCHAR2(30) CONSTRAINT GM_M_NAME_NN NOT NULL,
    GRADE_CODE NUMERIC(10),
    GRADE_NAME VARCHAR2(25) CONSTRAINT GM_GARM_NAME_NN NOT NULL,
    CONSTRAINT GM_G_M_CODE_PK PRIMARY KEY(GRADE_MOVIE_CODE),
    CONSTRAINT GM_GARM_CODE_FK FOREIGN KEY(MOVIE_CODE) REFERENCES MOVIE_THEATER(MOVIE_CODE),
    CONSTRAINT GM_GARG_CODE_FK FOREIGN KEY(GRADE_CODE) REFERENCES MOVIE_GRADE(GRADE_CODE)
);

// 데이터 삽입
INSERT INTO GRADE_MOVIE VALUES(1119122011, 19122011, '백두산', 519122012, '12세이상관람가');
INSERT INTO GRADE_MOVIE VALUES(1119122012, 19122012, '시동', 519122013, '15세이상관람가');
INSERT INTO GRADE_MOVIE VALUES(1119122013, 19122013, '겨울왕국2', 519122011, '전체관람가');
INSERT INTO GRADE_MOVIE VALUES(1119122014, 19122014, '포드 V 페라리', 519122012, '12세이상관람가');
INSERT INTO GRADE_MOVIE VALUES(1119122015, 19122015, '쥬만지 : 넥스트 레벨', 519122012, '12세이상관람가');
INSERT INTO GRADE_MOVIE VALUES(1119122016, 19122016, '나이브스 아웃', 519122012, '12세이상관람가');
INSERT INTO GRADE_MOVIE VALUES(1119122017, 19122017, '로미오와 줄리엣', 519122012, '12세이상관람가');
INSERT INTO GRADE_MOVIE VALUES(1119122018, 19122018, '눈의 여왕4', 519122011, '전체관람가');
INSERT INTO GRADE_MOVIE VALUES(1119122019, 19122019, '캣츠', 519122012, '12세이상관람가');
INSERT INTO GRADE_MOVIE VALUES(1119122020, 19122020, '천문 : 하늘에 묻는다', 519122012, '12세이상관람가');
INSERT INTO GRADE_MOVIE VALUES(1119122021, 19122021, '블랙머니', 519122012, '12세이상관람가');
INSERT INTO GRADE_MOVIE VALUES(1119122022, 19122022, '교회오빠', 519122011, '전체관람가');
INSERT INTO GRADE_MOVIE VALUES(1119122023, 19122023, '감쪽같은 그녀', 519122011, '전체관람가');
INSERT INTO GRADE_MOVIE VALUES(1119122024, 19122024, '미안해요, 리키', 519122012, '12세이상관람가');
INSERT INTO GRADE_MOVIE VALUES(1119122025, 19122025, '윤희에게', 519122012, '12세이상관람가');
INSERT INTO GRADE_MOVIE VALUES(1119122026, 19122026, '신의 한 수 : 귀수편', 519122013, '15세이상관람가');
INSERT INTO GRADE_MOVIE VALUES(1119122027, 19122027, '블랙 스완', 519122014, '청소년관람불가');
INSERT INTO GRADE_MOVIE VALUES(1119122028, 19122028, '10년', 519122011, '전체관람가');
INSERT INTO GRADE_MOVIE VALUES(1119122029, 19122029, '라스트 크리스마스', 519122012, '12세이상관람가');
INSERT INTO GRADE_MOVIE VALUES(1119122030, 19122030, '러브 액츄얼리', 519122013, '15세이상관람가');
```

3) 정규화과정 > xlsx 첨부했습니다.



4) 데이터베이스 테이블 작성 > 데이터베이스 테이블 작성.zip 첨부했습니다.

```
SQL> SELECT
  2      GRADE_MOVIE_CODE "등급-영화 코드", MOVIE_CODE 영화코드, MOVIE_NAME 영화이름,
  3      GRADE_CODE 등급코드, GRADE_NAME 등급이름
  4  FROM GRADE_MOVIE;

등급-영화 코드   영화코드 영화이름                 등급코드 등급이름
-------------- -------- ---------------------- -------- ------------
    1119122011 19122011 백두산                 519122012 12세이상관람가
    1119122012 19122012 시동                   519122013 15세이상관람가
    1119122013 19122013 겨울왕국2              519122011 전체관람가
    1119122014 19122014 포드 V 페라리          519122012 12세이상관람가
    1119122015 19122015 쥬만지 : 넥스트 레벨   519122012 12세이상관람가
    1119122016 19122016 나이브스 아웃          519122012 12세이상관람가
    1119122017 19122017 로미오와 줄리엣        519122012 12세이상관람가
    1119122018 19122018 눈의 여왕4             519122011 전체관람가
    1119122019 19122019 캣츠                   519122012 12세이상관람가
    1119122020 19122020 천문 : 하늘에 묻는다   519122012 12세이상관람가
    1119122021 19122021 블랙머니               519122012 12세이상관람가
    1119122022 19122022 교회오빠               519122011 전체관람가
    1119122023 19122023 감쪽같은 그녀          519122011 전체관람가
    1119122024 19122024 미안해요, 리키         519122012 12세이상관람가
    1119122025 19122025 윤희에게               519122012 12세이상관람가
    1119122026 19122026 신의 한 수 : 귀수편    519122013 15세이상관람가
    1119122027 19122027 블랙 스완              519122014 청소년관람불가
    1119122028 19122028 10년                   519122011 전체관람가
    1119122029 19122029 라스트 크리스마스      519122012 12세이상관람가
    1119122030 19122030 러브 액츄얼리          519122013 15세이상관람가

20 rows selected.
```

```
SQL> SELECT
  2      DIRECTOR_CODE 감독코드, DIRECTOR_NAME 감독이름
  3  FROM MOVIE_DIRECTOR;

 감독코드 감독이름
--------- --------------------
 119122010 [관련정보없음]
 119122011 이해준
 119122012 최정열
 119122013 변영규
 119122014 제니퍼 리
 119122015 제임스 맨골드
 119122016 제이크 캐스단
 119122017 라이언 존슨
 119122018 케네스 맥밀란



 119122029 후지무라 아키요
 119122030 폴 페이그
 119122031 리차드 커티스
 119122032 고레에다 히로카즈
 119122033 김승우
 119122034 김도영
 119122035 조영광
 119122036 신카이 마코토
 119122037 할룩 칸 디즈다로글
 119122038 장 폴 루브

29 rows selected.
```

```
SQL> SELECT
  2      ACTOR_CODE 배우코드, ACTOR_NAME 배우이름
  3  FROM MOVIE_ACTOR;

 배우코드 배우이름
--------- --------------------
 219122010 [관련정보없음]
 219122011 이병헌
 219122012 하정우
 219122013 마동석
 219122014 전혜진
 219122015 마동석
 219122016 박정민
 219122017 정해인



 219122102 박상우
 219122103 서반석
 219122104 석승훈
 219122105 이명호
 219122106 루디빈 사니에
 219122107 호세 가르시아
 219122108 장 폴 루브
 219122109 람지 베디아

100 rows selected.
```

```
SQL> SELECT
  2      DIRECTOR_MOVIE_CODE "감독-영화코드", MOVIE_CODE 영화코드, MOVIE_NAME 영화이름,
  3      DIRECTOR_CODE 감독코드, DIRECTOR_NAME 감독이름
  4  FROM DIRECTOR_MOVIE;

감독-영화코드   영화코드 영화이름                 감독코드 감독이름
------------- -------- ---------------------- --------- --------------------
    619122011 19122011 백두산                 119122011 이해준
    619122012 19122012 시동                   119122012 최정열
    619122013 19122013 겨울왕국2              119122014 제니퍼 리
    619122014 19122014 포드 V 페라리          119122015 제임스 맨골드
    619122015 19122015 쥬만지 : 넥스트 레벨   119122016 제이크 캐스단
    619122016 19122016 나이브스 아웃          119122017 라이언 존슨
    619122017 19122017 로미오와 줄리엣        119122018 케네스 맥밀란
    619122018 19122018 눈의 여왕4             119122019 로브터 렌스
    619122019 19122019 캣츠                   119122020 톰 후퍼
    619122020 19122020 천문 : 하늘에 묻는다   119122021 허진호
    619122021 19122021 블랙머니               119122022 정지영
    619122022 19122022 교회오빠               119122023 이호경
    619122023 19122023 감쪽같은 그녀          119122024 허인무
    619122024 19122024 미안해요, 리키         119122025 켄 로치
    619122025 19122025 윤희에게               119122027 임대형
    619122026 19122026 신의 한수 : 귀수편     119122028 리건
    619122027 19122027 블랙 스완              119122010 [관련정보없음]
    619122028 19122028 10년                   119122029 후지무라 아키요
    619122029 19122029 라스트 크리스마스      119122030 폴 페이그
    619122030 19122030 러브 액츄얼리          119122031 리차드 커티스

20 rows selected.
```

```
SQL>  SELECT
  2      ACTOR_MOVIE_CODE "배우-영화코드", MOVIE_CODE 영화코드, MOVIE_NAME 영화이름,
  3      ACTOR_CODE 배우코드, ACTOR_NAME 배우이름
  4  FROM ACTOR_MOVIE;

배우-영화코드  영화코드 영화이름                 배우코드 배우이름
------------ -------- -------------------- -------- --------------------
   719122011 19122011 백두산                219122011 이병헌
   719122012 19122011 백두산                219122012 하정우
   719122013 19122011 백두산                219122013 마동석
   719122014 19122011 백두산                219122014 전혜진
   719122015 19122012 시동                  219122013 마동석
   719122016 19122012 시동                  219122016 박정민

   719122069 19122028 10년                  219122071 스기사키 하나
   719122070 19122028 10년                  219122072 이케와키 치즈루
   719122071 19122028 10년                  219122073 타이가
   719122072 19122029 라스트 크리스마스     219122074 에밀리아 클라크
   719122073 19122029 라스트 크리스마스     219122075 헨리 골딩
   719122074 19122029 라스트 크리스마스     219122076 양자경
   719122075 19122029 라스트 크리스마스     219122077 엠마 톰슨
   719122076 19122030 러브 액츄얼리         219122078 휴 그랜트
   719122077 19122030 러브 액츄얼리         219122079 콜린 퍼스
   719122078 19122030 러브 액츄얼리         219122077 엠마 톰슨
   719122079 19122030 러브 액츄얼리         219122081 키이라 나이틀리

69 rows selected.
```

```
SQL> SELECT
  2      DIRECTOR_ACTOR_MOVIE_CODE "감독-배우-영화코드", MOVIE_CODE 영화코드, MOVIE_NAME 영화이름,
  3      DIRECTOR_CODE 감독코드, DIRECTOR_NAME 감독이름, ACTOR_CODE 배우코드, ACTOR_NAME 배우이름
  4  FROM DIRECTOR_ACTOR_MOVIE;

감독-배우-영화코드 영화코드 영화이름           감독코드 감독이름         배우코드 배우이름
------------------ -------- -------------------- -------- ---------- -------- --------------------
         819122011 19122011 백두산              119122011 이해준      219122011 이병헌
         819122012 19122011 백두산              119122011 이해준      219122012 하정우
         819122013 19122011 백두산              119122011 이해준      219122013 마동석
         819122014 19122011 백두산              119122011 이해준      219122014 전혜진
         819122015 19122012 시동                119122012 최정열      219122013 마동석
         819122016 19122012 시동                119122012 최정열      219122016 박정민
         819122017 19122012 시동                119122012 최정열      219122017 정해인
         819122018 19122012 시동                119122012 최정열      219122018 염정아
         819122019 19122013 겨울왕국2           119122014 제니퍼 리   219122023 크리스틴 벨
         819122020 19122013 겨울왕국2           119122014 제니퍼 리   219122024 아디나 멘젤

         819122073 19122029 라스트 크리스마스   119122030 폴 페이그   219122075 헨리 골딩
         819122074 19122029 라스트 크리스마스   119122030 폴 페이그   219122076 양자경
         819122075 19122029 라스트 크리스마스   119122030 폴 페이그   219122077 엠마 톰슨
         819122076 19122030 러브 액츄얼리       119122031 리차드 커티스 219122078 휴 그랜트
         819122077 19122030 러브 액츄얼리       119122031 리차드 커티스 219122079 콜린 퍼스
         819122078 19122030 러브 액츄얼리       119122031 리차드 커티스 219122077 엠마 톰슨
         819122079 19122030 러브 액츄얼리       119122031 리차드 커티스 219122081 키이라 나이틀리

69 rows selected.
```

```
SQL>  SELECT
  2      GENRE_CODE 장르코드, GENRE_NAME 장르이름
  3  FROM MOVIE_GENRE;

  장르코드 장르이름
---------- --------------------
 319122011 드라마(한국)
 319122012 애니메이션(한국)
 319122013 사극(한국)
 319122014 다큐멘터리(한국)
 319122015 멜로/로맨스(한국)
 319122016 스릴러(한국)
 319122017 드라마(미국)
 319122018 애니메이션(미국)
 319122019 액션(미국)
 319122020 스릴러(미국)
 319122021 멜로/로맨스(미국)
 319122022 드라마(영국)
 319122023 드라마(일본)
 319122024 애니메이션(일본)
 319122025 드라마(프랑스)
 319122026 애니메이션(기타)

16 rows selected.
```

```
SQL> SELECT
  2      GENRE_MOVIE_CODE "장르-영화 코드", MOVIE_CODE 영화코드, MOVIE_NAME 영화이름,
  3      GENRE_CODE 장르코드, GENRE_NAME 장르이름
  4  FROM GENRE_MOVIE;

장르-영화 코드 영화코드 영화이름           장르코드 장르이름
-------------- -------- -------------------- -------- --------------------
     919122011 19122011 백두산              319122011 드라마(한국)
     919122012 19122012 시동                319122011 드라마(한국)
     919122013 19122013 겨울왕국2           319122018 애니메이션(미국)
     919122014 19122014 포드 V 페라리       319122017 드라마(미국)
     919122015 19122015 쥬만지 : 넥스트 레벨 319122019 액션(미국)
     919122016 19122016 나이브스 아웃       319122016 스릴러(미국)
     919122017 19122017 로미오와 줄리엣     319122022 드라마(영국)
     919122018 19122018 눈의 여왕4          319122026 애니메이션(기타)
     919122019 19122019 캣츠                319122017 드라마(미국)
     919122020 19122020 천문 : 하늘에 묻는다 319122013 사극(한국)
     919122021 19122021 블랙머니            319122011 드라마(한국)
     919122022 19122022 교회오빠            319122014 다큐멘터리(한국)
     919122023 19122023 감쪽같은 그녀       319122011 드라마(한국)
     919122024 19122024 미안해요, 리키      319122025 드라마(프랑스)
     919122025 19122025 윤희에게            319122015 멜로/로맨스(한국)
     919122026 19122026 신의 한 수 : 귀수편 319122016 스릴러(한국)
     919122027 19122027 블랙 스완           319122020 스릴러(미국)
     919122028 19122028 10년                319122023 드라마(일본)
     919122029 19122029 라스트 크리스마스   319122021 멜로/로맨스(미국)
     919122030 19122030 러브 액츄얼리       319122021 멜로/로맨스(미국)

20 rows selected.
```

```
SQL> SELECT
  2      TYPE_CODE 타입코드, TYPE_NAME 타입이름
  3  FROM MOVIE_TYPE;

  타입코드 타입이름
---------- ----------
 419122011 없음
 419122012 더빙
 419122013 자막
 419122014 영어자막
 419122015 더빙,자막
```

```
SQL> SELECT
  2      TYPE_MOVIE_CODE "장르-타입 코드", MOVIE_CODE 영화코드, MOVIE_NAME 영화이름,
  3      TYPE_CODE 타입코드, TYPE_NAME 타입이름
  4  FROM TYPE_MOVIE;

장르-타입 코드 영화코드 영화이름           타입코드 타입이름
-------------- -------- -------------------- -------- ----------
    1019122011 19122011 백두산              419122011 없음
    1019122012 19122012 시동                419122011 없음
    1019122013 19122013 겨울왕국2           419122015 더빙,자막
    1019122014 19122014 포드 V 페라리       419122011 없음
    1019122015 19122015 쥬만지 : 넥스트 레벨 419122011 없음
    1019122016 19122016 나이브스 아웃       419122011 없음
    1019122017 19122017 로미오와 줄리엣     419122011 없음
    1019122018 19122018 눈의 여왕4          419122012 더빙
    1019122019 19122019 캣츠                419122011 없음
    1019122020 19122020 천문 : 하늘에 묻는다 419122011 없음
    1019122021 19122021 블랙머니            419122014 영어자막
    1019122022 19122022 교회오빠            419122011 없음
    1019122023 19122023 감쪽같은 그녀       419122011 없음
    1019122024 19122024 미안해요, 리키      419122011 없음
    1019122025 19122025 윤희에게            419122011 없음
    1019122026 19122026 신의 한 수 : 귀수편 419122011 없음
    1019122027 19122027 블랙 스완           419122013 자막
    1019122028 19122028 10년                419122011 없음
    1019122029 19122029 라스트 크리스마스   419122011 없음
    1019122030 19122030 러브 액츄얼리       419122011 없음

20 rows selected.
```

```
SQL> SELECT
  2      GRADE_CODE 등급코드, GRADE_NAME 등급이름
  3  FROM MOVIE_GRADE;

  등급코드 등급이름
---------- ---------------
 519122011 전체관람가
 519122012 12세이상관람가
 519122013 15세이상관람가
 519122014 청소년관람불가
```

```
SQL> SELECT
  2      GRADE_MOVIE_CODE "등급-영화 코드", MOVIE_CODE 영화코드, MOVIE_NAME 영화이름,
  3      GRADE_CODE 등급코드, GRADE_NAME 등급이름
  4  FROM GRADE_MOVIE;

등급-영화 코드   영화코드 영화이름                 등급코드 등급이름
-------------- --------- -------------------- --------- ---------------
    1119122011  19122011 백두산                519122012 12세이상관람가
    1119122012  19122012 시동                  519122013 15세이상관람가
    1119122013  19122013 겨울왕국2             519122011 전체관람가
    1119122014  19122014 포드 V 페라리         519122012 12세이상관람가
    1119122015  19122015 쥬만지 : 넥스트 레벨  519122012 12세이상관람가
    1119122016  19122016 나이브스 아웃         519122012 12세이상관람가
    1119122017  19122017 로미오와 줄리엣       519122012 12세이상관람가
    1119122018  19122018 눈의 여왕4           519122011 전체관람가
    1119122019  19122019 캣츠                 519122012 12세이상관람가
    1119122020  19122020 천문 : 하늘에 묻는다 519122012 12세이상관람가
    1119122021  19122021 블랙머니             519122012 12세이상관람가
    1119122022  19122022 교회오빠             519122011 전체관람가
    1119122023  19122023 감쪽같은 그녀        519122011 전체관람가
    1119122024  19122024 미안해요, 리키       519122012 12세이상관람가
    1119122025  19122025 윤희에게             519122012 12세이상관람가
    1119122026  19122026 신의 한 수 : 귀수편  519122013 15세이상관람가
    1119122027  19122027 블랙 스완           519122014 청소년관람불가
    1119122028  19122028 10년               519122011 전체관람가
    1119122029  19122029 라스트 크리스마스   519122012 12세이상관람가
    1119122030  19122030 러브 액츄얼리       519122013 15세이상관람가

20 rows selected.
```

7. 구현 및 데모

```
/**********************************************************************
 * import 설명 *
 * import java.awt.BorderLayout;
 * >> BorderLayout은 컨테이너를 North, South, East, West, Center 모두 5개의 영역으로 나누고
 * >> 각 영역에 하나의 컴포넌트만을 배치할 수 있도록 합니다.
 *
 * import java.awt.Image;
 * >> 이미지를 만들고 수정하기 위한 클래스
 *
 * import java.awt.event.ActionEvent;
 * >> 버튼이 클릭되거나 리스트, 메뉴 등이 선택되었을 때 발생하는 이벤트 클래스
 * >> ActionListener 인터페이스의 actionPerformed() 메서드를 이용해서 처리합니다.
 *
 * import java.awt.event.ActionListener;
 * >> 이벤트 인터페이스
 *
 * import java.sql.Connection;
 * >> Connection 객체를 자동완성으로 import 하기 위한 java 표준 java.sql.Connection 클래스
 *
 * import java.sql.DriverManager;
 * >> java.sql의 인터페이스들을 상속해서 메소드의 몸체를 구현한 JDBC 드라이버 클래스
 *
 * import java.sql.ResultSet;
 * >> select 쿼리 실행 시 executeQuery() 메서드를 사용하며
 * >> 실행 결과를 java.sql.ResultSet형으로 리턴하는데, 이 결과 집합을 사용하기 위한 클래스
 *
 * import java.sql.SQLException;
 * >> DB 액세스 에러 또는 그 외의 에러에 관한 정보를 제공합니다.
 *
 * import java.sql.Statement;
 * >> 정적인 쿼리문을 처리할 수 있는 Statement를 사용하기 위한 클래스
 *
 * import javax.swing.ImageIcon;
 * >> 이미지에서 아이콘을 그리는 Icon 인터페이스를 구현한 클래스
 *
 * import javax.swing.JButton;
 * >> 버튼을 구성하기위해 사용하는 클래스
 *
 * import javax.swing.JFrame;
 * >> 프레임을 생성시키기 위해 사용하는 클래스
 *
 * import javax.swing.JLabel;
 * >> 컴포넌트에 텍스트와 이미지를 넣을 때 사용하는 클래스
 * >> getText()와 setText(String str)을 사용할 수 있습니다.
 *
 * import javax.swing.JPanel;
 * >> 컴포넌트들을 그룹별로 묶어서 처리할 때 사용하는 컨테이너 클래스
 * >> 일반적으로 Frame에 컴포넌트들을 직접 붙이지 않고 Panel을 사용합니다.
 *
 * import javax.swing.JScrollPane;
 * >> 스크롤을 이용해서 컴포넌트들을 보여주는 컴포넌트 클래스
 * >> 스크롤을 이용해서 보여주는 화면을 상하좌우로 이동하여 포함된 컴포넌트의 원래 크기를 유지합니다.
 *
```

```
 * import javax.swing.JTable;
 * >> 데이터를 행과 열로 구성되어 있는 테이블 형식으로 보여주는 컴포넌트 클래스
 * >> JTable을 사용하기 위해서는 먼저 데이터를 저장할 모델을 만들고 JTable에 연결합니다.
 *
 * import javax.swing.SwingConstants;
 * >> 일반적으로 화면에서 구성 요소를 배치하고 방향을 지정할 때 사용하는 클래스
 *
 * import javax.swing.table.DefaultTableModel;
 * >> 데이터를 행과 열로 구성되어 있는 테이블 형식으로 보여주는 컴포넌트 클래스
 * >> 데이터를 Vector로 생성하기 때문에 JTable에서 데이터의 추가, 삭제를 할 수 있습니다.
 *
 ************************************************************************/

import java.awt.BorderLayout;
import java.awt.Container;
import java.awt.FlowLayout;
import java.awt.Image;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.table.DefaultTableModel;


/*****************************************************************************
* Title : 영화관 영화 상영관리에 대한 Oracle DB 연동 및 java언어 구현
* Date : 2019.12.21
* Purpose : Oracle DB 연동 및 JFrame 사용
*
* 과목 : 데이터베이스
* 이름 : 구명회
* 학번 : 20154215
* 학과 : 컴퓨터공학과
*****************************************************************************/

public class GUI extends JFrame {
	private Connection conn = null;	// DB와 연결된 상태를 담은 객체 conn을 선언했습니다.
	private JLabel state;

	// 초기 화면에서 보여질 테이블의 필드값 설정
	private String colName[] = { "영화코드", "영화이름", "개봉일", "관객수(명)", "예매율(%)", "평점", "영화홈페이지" };
	private DefaultTableModel model = new DefaultTableModel(colName, 0);
	private JTable table = new JTable(model);
	private String row[] = new String[7];

	public GUI() {
		/*****************************************************************************
		 * >> setTitle() 메소드를 사용해서 프레임의 타이틀을 설정합니다. DataBase Test 20154215 구명회
		 * >> setDefaultCloseOperation(EXIT_ON_CLOSE) : 프레임 윈도우를 닫으면 프로그램이 종료됩니다.
		 *****************************************************************************/
		setTitle("DataBase Test 20154215 구명회");
		setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

		/*****************************************************************************
		* 사진 부착하기  *
		* >> ImageIcon() 메소드를 사용해서 사진을 읽어왔습니다.
		* >> 사진의 경로는 현재 프로젝트 폴더에서 생성한 image디렉터리입니다.
		* >> 그런데 읽어온 사진의 크기가 커서 크기를 조절했습니다.
		*****************************************************************************/
		ImageIcon image = new ImageIcon("image/Create_homework.jpg");
		Image resizeImage = image.getImage();
		Image FinresizeImage = resizeImage.getScaledInstance(900, 400, java.awt.Image.SCALE_SMOOTH);
		ImageIcon reimage = new ImageIcon(FinresizeImage);
		JLabel imageLabel = new JLabel(reimage);

		/*****************************************************************************
		 * 컴포넌트옹 패널 *
		 * >> 하단부 연결 및 출력 버튼 컴포넌트용 패널
		 * >> 이미지용 패널
		*****************************************************************************/
		JPanel Btn_panel = new JPanel();
```

```
JPanel image_panel = new JPanel();

/*****************************************************************************
 * 컴포넌트 설정 *
 * >> 버튼의 이름을 각각 Connect와 Show로 설정합니다.
 * >> add() 메소드를 사용해서 패널에 버튼과 이미지를 부착합니다.
*****************************************************************************/
// 삽입 삭제 추가 조회 버튼
JButton con = new JButton("Connect");          JButton show = new JButton("Show");
JButton add = new JButton("add");              JButton del = new JButton("delete");
JButton upd = new JButton("update");
JTextField tf = new JTextField(10);            add(tf);
image_panel.add(imageLabel);
Btn_panel.add(con);                            Btn_panel.add(show);
Btn_panel.add(add);                            Btn_panel.add(del);
Btn_panel.add(upd);

/*****************************************************************************
 * Connect 버튼 이벤트 리스너 (액션 리스너) *
 * >> Connect 버튼을 클릭했을 때 DB와 연결하는 Connect() 메소드를 호출하는 부분입니다.
 *****************************************************************************/
con.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
                connect();
        }
});

/*****************************************************************************
 * Show 버튼 이벤트 리스너 (액션 리스너) *
 * >> Show 버튼을 클릭했을 때 DB를 읽어오는 show() 메소드를 호출하는 부분입니다.
 * >> 버튼을 클릭하면 쿼리문을 수행한 결과를 보여줍니다.
 * >> model.setRowCount(0)
 * >> 데이터베이스에서 영화 리스트를 가져와서 출력할 때 레코드가 중복되어 나타나므로
 * >> 이 메소드를 사용해서 JTable을 초기화시켜줍니다.
 * ***************************************************************************/
show.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
                model.setRowCount(0);
                show_db();
        }
});

/*****************************************************************************
 * add 버튼 이벤트 리스너 (액션 리스너) *
 * >> add 버튼을 클릭하면 데이터베이스에 레코드를 삽입할 수 있습니다.
 * ***************************************************************************/

add.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
                new add_db();
        }
});

/*****************************************************************************
 * del 버튼 이벤트 리스너 (액션 리스너) *
 * >> del 버튼을 클릭하면 데이터베이스에 저장된 레코드를 삭제할 수 있습니다.
 * ***************************************************************************/
del.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
                new del_db();
        }
});

/*****************************************************************************
 * upd 버튼 이벤트 리스너 (액션 리스너) *
 * >> upd 버튼을 클릭하면 데이터베이스에 저장된 레코드를 수정할 수 있습니다.
 * ***************************************************************************/
upd.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
                new upd_db();
        }
});

/*****************************************************************************
 * 배치 *
 * >> DB 상태 출력용 라벨에 "Oracle DB 연동 테스트"를 출력합니다.
 * >> DB 상태 출력용 라벨의 위치를 BorderLayout을 사용해서 배치합니다.
 * >> JScrollPane() : 화면에서 넘어갈 경우 스크롤바가 생깁니다.
 * >> Btn_panel과 image_panel의 위치를 폼에 맞춰 조절했습니다.
 * >> setSize() 메소드를 사용해서 화면에 900x500 크기의 프레임을 보여줍니다.
 * >> setResizable(false) : 프레임의 크기는 수정할 수 없습니다.
 *****************************************************************************/
```

```
		state = new JLabel();
		state.setText("Oracle DB 연동 테스트");
		add(state, BorderLayout.NORTH);

		add(new JScrollPane(table), BorderLayout.CENTER);
		add(Btn_panel, BorderLayout.SOUTH);
		add(image_panel, BorderLayout.WEST);

		setSize(900, 500); setVisible(true);	setResizable(false);
}

/*****************************************************************************
 * add class *
 * >> 이 클래스에서 데이터 삽입에 관한 명령을 처리합니다
 * >> 초기화면에서 add버튼을 누르면 생성되는 자식창에서 MOVIE_THEATER 테이블에 대한 필드를 입력할 수 있습니다.
 * >> 필드를 입력하고 insert를 누르면 각각의 값들이 테이블 필드 형식에 맞게 변환됩니다. (ex) string > int)
 * >> 변환된 변수값들을 insert_db() 메소드의 인자값으로 넘겨줍니다.
  *****************************************************************************/
class add_db extends JFrame {
		// insert
		Container text;	JButton btn;

		add_db(){
			setTitle("add");
			// 컨텐트팬의 주소를 알아냄
			text = getContentPane();
			text.add(new insertData(), BorderLayout.CENTER);

			setSize(330,300);	setResizable(false);		setVisible(true);
		}

		class insertData extends JPanel {

			insertData(){

			/*************************************************************************
				 * 필드값 입력 *
				 * >> 필드를 입력한 값들이 각 변수들에 string형태로 저장됩니다.
			  *************************************************************************/
				JTextField movie_code	= new JTextField(20);
				JTextField movie_name	= new JTextField(20);
				JTextField movie_release	= new JTextField(20);
				JTextField movie_aud	= new JTextField(20);
				JTextField movie_ratio	= new JTextField(20);
				JTextField movie_rating	= new JTextField(20);
				JTextField movie_website	= new JTextField(20);

				add(new JLabel("영화코드"));		add(movie_code);
				add(new JLabel("영화이름"));		add(movie_name);
				add(new JLabel("개봉일"));		add(movie_release);
				add(new JLabel("관객수(명)"));		add(movie_aud);
				add(new JLabel("예매율(%)"));		add(movie_ratio);
				add(new JLabel("평점"));		add(movie_rating);
				add(new JLabel("영화홈페이지"));	add(movie_website);

				btn = new JButton("insert");		add(btn);


			/*************************************************************************
				 * 버튼 이벤트 *
				 * >> 여기서 각 변수값을 변환시킨 것을 메소드의 인자값으로 넘겨줍니다.
			  *************************************************************************/
				btn.addActionListener(new ActionListener(){
					public void actionPerformed(ActionEvent e) {

						int m_code		= Integer.parseInt(movie_code.getText());
						String m_name	= movie_name.getText();
						String m_rel		= movie_release.getText();
						int m_aud		= Integer.parseInt(movie_aud.getText());
						Double m_ratio	= Double.parseDouble(movie_ratio.getText());
						Double m_rating	= Double.parseDouble(movie_rating.getText());
						String m_website	= movie_website.getText();

						insert_db(m_code, m_name, m_rel, m_aud, m_ratio,
									m_rating, m_website);
					}
				});
			}
		}
}

/*****************************************************************************
 * del class *
 * >> 이 클래스에서 데이터 삭제에 관한 명령을 처리합니다.
 * >> 초기화면에서 delete버튼을 누르면 생성되는 자식창에서 MOVIE_THEATER 테이블에 저장되어 있는 영화들 중에
 * >> 삭제하고싶은 영화의 영화코드를 입력할 수 있습니다.
 * >> 영화코드를 입력하고 delete를 누르면 입력한 영화코드값을 delete_db() 메소드에게 인자값으로 넘겨주면
```

```
 * >> 영화코드에 해당하는 레코드가 삭제됩니다.
 ***************************************************************************/
class del_db extends JFrame {
	// delete
	Container text;	JButton btn;

	del_db(){
		setTitle("delete");
		text = getContentPane(); // 컨텐트팬의 주소를 알아냄
		text.add(new deleteData(), BorderLayout.CENTER);

		setSize(300,130);	setResizable(false);		setVisible(true);
	}

	class deleteData extends JPanel {

		deleteData(){

		/*************************************************************************
		 * 필드값 입력 *
		 * >> 필드를 입력한 값이 변수에 string형태로 저장됩니다.
		 *************************************************************************/
			JTextField movie_code = new JTextField(20);

			btn = new JButton("delete");
			add(new JLabel("영화코드"));		add(movie_code);
								add(btn);

		/*************************************************************************
		 * 버튼 이벤트 *
		 * >> 여기서 변수값을 변환시킨 것을 메소드의 인자값으로 넘겨줍니다.
		*************************************************************************/

			btn.addActionListener(new ActionListener(){
				public void actionPerformed(ActionEvent e) {
					int m_code = Integer.parseInt(movie_code.getText());
					delete_db(m_code);
				}
			});
		}
	}
}

/***************************************************************************
 * upd class *
 * >> 이 클래스에서 데이터 삭제에 관한 명령을 처리합니다.
 * >> 초기화면에서 update버튼을 누르면 자식창에서 MOVIE_THEATER 테이블에 저장되어 있는 영화들 중에
 * >> 변경하고 싶은 영화의 영화코드를 입력하고, 변경할 영화의 정보들을 입력할 수 있습니다.
 * >> 해당 값들을 전부 입력하고 update를 누르면 insert와 동일하게 입력한 값들의 형식을 바꿔서
 * >> update_db() 메소드의 인자로 전달하면 해당 레코드의 정보가 변경됩니다.
 ***************************************************************************/
class upd_db extends JFrame {
	// update
	Container text;	JButton btn;

	upd_db(){
		setTitle("update");
		text = getContentPane(); // 컨텐트팬의 주소를 알아냄
		text.add(new updateData(), BorderLayout.CENTER);

		setSize(330,350);	setResizable(false);		setVisible(true);
	}

	class updateData extends JPanel {

		updateData(){

		/*************************************************************************
		 * 필드값 입력 *
		 * >> 필드를 입력한 값들이 각 변수들에 string형태로 저장됩니다.
		*************************************************************************/
			JTextField b_movie_code	= new JTextField(20);
			JTextField movie_code	= new JTextField(20);
			JTextField movie_name	= new JTextField(20);
			JTextField movie_release	= new JTextField(20);
			JTextField movie_aud	= new JTextField(20);
			JTextField movie_ratio	= new JTextField(20);
			JTextField movie_rating	= new JTextField(20);
			JTextField movie_website	= new JTextField(20);

			btn = new JButton("update");

			add(new JLabel("바꾸고 싶은 영화의 코드를 입력하세요"));
			add(b_movie_code);
			add(new JLabel("변경할 영화의 정보를 입력하세요 (전부 입력해야 함)"));

			add(new JLabel("영화코드"));		add(movie_code);
```

```
                                add(new JLabel("영화이름"));            add(movie_name);
                                add(new JLabel("개봉일"));              add(movie_release);
                                add(new JLabel("관객수(명)"));                  add(movie_aud);
                                add(new JLabel("예매율(%)"));                   add(movie_ratio);
                                add(new JLabel("평점"));                        add(movie_rating);
                                add(new JLabel("영화홈페이지"));                add(movie_website);
                                                                                add(btn);


/*************************************************************************
                                 * 버튼 이벤트 *
                                 * >> 여기서 변수값을 변환시킨 것을 메소드의 인자값으로 넘겨줍니다.

*************************************************************************/
                                btn.addActionListener(new ActionListener(){
                                        public void actionPerformed(ActionEvent e) {
                                                int m_bcode       = Integer.parseInt(b_movie_code.getText());
                                                int m_code                = Integer.parseInt(movie_code.getText());
                                                String m_name     = movie_name.getText();
                                                String m_rel      = movie_release.getText();
                                                int m_aud                 = Integer.parseInt(movie_aud.getText());
                                                Double m_ratio    = Double.parseDouble(movie_ratio.getText());
                                                Double m_rating = Double.parseDouble(movie_rating.getText());
                                                String m_website= movie_website.getText();

                                                upd_db(m_bcode, m_code, m_name, m_rel, m_aud, m_ratio,
                                                                                                m_rating, m_website);
                                        }
                                });
                        }
                }
        }


        /*************************************************************************************************
         * DB 상태 출력용 라벨을 변경합니다.
         * 초기 상태는 "Oracle DB 연동 테스트"를 출력하며
         * Connect 버튼을 눌렀을 때 DB와 연결이 성공하면 "성공적 연결"문장으로,
         * 실패하면 "DB 연결 실패" 문장으로 바꾸어 출력합니다.
         * Show 버튼을 눌렀을 때 쿼리문 수행에 성공하면 "DB 읽기 성공"문장으로,
         * 실패하면 "DB 읽기 실패" 문장으로 바꾸어 출력합니다.
         *************************************************************************************************/
        public void connect() {
                try {
        /*************************************************************************************************
         * DriverManager 클래스에 등록 *
         * >> 드라이버 로딩
         * >> 드라이버 인터페이스를 구현한 클래스를 로딩합니다.
         *
         * DriverManager.getConnection() : DB에 연결(접속)하기 위한 메소드 *
         * >> 드라이버 매니저에게 Connection 객체를 요청합니다.
         * >> Connection을 얻기위해 getConnection메소드를 사용하며 인자는 ("접속url", "사용자 계정", '계정 비밀번호")입니다.
         * >> 이 객체를 사용해서 statement로 쿼리문을 수행할 수 있습니다.
        *************************************************************************************************/
                        Class.forName("oracle.jdbc.driver.OracleDriver");
                        conn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "MHHW4", "1234");
                        System.out.println("성공적 로딩");
                        state.setText("성공적 연결");

                } catch (ClassNotFoundException e) {
                        e.printStackTrace();
                        state.setText("DB 연결 실패 " + e.toString());

                } catch (SQLException e) {
                        e.printStackTrace();
                        state.setText("DB 연결 실패 " + e.toString());
                }
        }

        /*************************************************************************************************
         * 쿼리문을 수행하는 메소드입니다.
         * >> Show 버튼의 이벤트 리스너에 의해 호출되며, 버튼이 클릭되면 try ~ catch문을 수행해서 결과를 보여줍니다.
         *************************************************************************************************/
        public void show_db() {
                try {
                        /*****************************************************************
                         * conn.createStatement() : Statement 생성 *
                         * >> select 쿼리 수행을 위한 Statement 객체를 생성합니다.
                         * 쿼리문 수행 *
                         * >> JDBC에서 쿼리를 작성할 때 세미콜론 (;)을 빼고 작성합니다.
                         * >> *로 모든 칼럼을 읽어오는 것보다 가져와야 할 칼럼을 직접 명시해주는게 좋다고 합니다.
                         * >> 쿼리를 한 줄로 쓰기 어려운 경우 들여쓰기를 사용해도 상관 없습니다.
                         * >> 레코드들은 ResultSet 객체에 추가됩니다.
                         *****************************************************************/
                        Statement stmt = conn.createStatement();
                        ResultSet rset = stmt.executeQuery("select * from MOVIE_THEATER");
                        System.out.println("영화코드\t영화이름\t개봉일\t관객수(명)\t예매율(%)\t평점\t영화홈페이지");
```

```
                /*********************************************
                 * while(rset.next()) : 쿼리문 결과 출력 *
                 * >> test table
                 * >> num                name    address          phone
                 * >>  1                 용길정   광주 서구                010-1234-5678
                 * >>  2                 최정성   광주 북구                010-1111-7148
                 * >>  3                 김소민   광주 남구                010-2222-1248
                 * >>  .... 이런식으로 저장되어 있고
                 * >>  next() 메소드는 현재 행에서 한 행 다음으로 이동하므로
                 * >>  test의 레코드 첫행부터 출력하게 됩니다.
                 * JTable에 출력 *
                 * addRow() 메소드를 사용해서 DB에서 읽어온 레코드들을 보여줍니다.
                 *********************************************/
                while (rset.next()) {
                        for (int i = 1; i <8; i++) {
                                System.out.print(rset.getString(i) + "\t");
                                row[i - 1] = rset.getString(i);
                        }

                        System.out.println();
                        model.addRow(row); // JTable에 출력
                }

                state.setText("DB 읽기 성공");
        } catch (SQLException e) {
                e.printStackTrace();
                state.setText("DB 읽기 실패 " + e.toString());
        }
}

public void insert_db(int code, String name, String release, int audience,
                                                        Double ratio, Double rating, String website) {
        try {
                /*********************************************************************
                 * conn.prepareStatement() : Statement 생성 *
                 * >> insert 쿼리 수행을 위한 Statement 객체를 생성합니다.
                 * 쿼리문 수행 *
                 * >> JDBC에서 쿼리를 작성할 때 세미콜론 (;)을 빼고 작성합니다.
                 * >> *로 모든 칼럼을 읽어오는 것보다 가져와야 할 칼럼을 직접 명시해주는게 좋다고 합니다.
                 * >> 쿼리를 한 줄로 쓰기 어려운 경우 들여쓰기를 사용해도 상관 없습니다.
                 * >> pstmt.executeUpdate() : 값이 할당되면  이 메서드를 사용해서 insert 쿼리를 실행합니다.
                 * >> count : 반환 값은 영향을 미친 row 개수입니다.
                 * >>                 값이 입력되지 않았으면 count값이 0이므로 입력 실패를 출력합니다.
                 *********************************************************************/
                String sql = "INSERT INTO MOVIE_THEATER VALUES(?, ?, ?, ?, ?, ?, ?)";
                PreparedStatement pstmt = conn.prepareStatement(sql);

                pstmt.setInt(1, code);                                  pstmt.setString(2, name);
                pstmt.setString(3, release);            pstmt.setInt(4, audience);
                pstmt.setDouble(5, ratio);                      pstmt.setDouble(6, rating);
                pstmt.setString(7, website);

                int count = pstmt.executeUpdate();
                if (count == 0)     System.out.println("데이터 입력 실패");
                else                                    System.out.println("데이터 입력 성공");

        }
        catch (SQLException e) {
                e.printStackTrace();
                state.setText("DB 읽기 실패 " + e.toString());
        }
}

public void delete_db(int code)       {
        try {
                /*********************************************************************
                 * conn.prepareStatement() : Statement 생성 *
                 * >> delete 쿼리 수행을 위한 Statement 객체를 생성합니다.
                 * 쿼리문 수행 *
                 * >> JDBC에서 쿼리를 작성할 때 세미콜론 (;)을 빼고 작성합니다.
                 * >> *로 모든 칼럼을 읽어오는 것보다 가져와야 할 칼럼을 직접 명시해주는게 좋다고 합니다.
                 * >> 쿼리를 한 줄로 쓰기 어려운 경우 들여쓰기를 사용해도 상관 없습니다.
                 * >> pstmt.executeUpdate() : 값이 할당되면  이 메서드를 사용해서 delete 쿼리를 실행합니다.
                 * >> count : 반환 값은 영향을 미친 row 개수입니다.
                 * >>                 값이 입력되지 않았으면 count값이 0이므로 삭제 실패를 출력합니다.
                 *********************************************************************/
                String sql = "DELETE FROM MOVIE_THEATER WHERE MOVIE_CODE = ?";
                PreparedStatement pstmt = conn.prepareStatement(sql);

                pstmt.setInt(1, code);

                int count = pstmt.executeUpdate();
                if (count == 0)     System.out.println("데이터 삭제 실패");
                else                                    System.out.println("데이터 삭제 성공");
```

```
		}
		catch (SQLException e) {
			e.printStackTrace();
			state.setText("DB 읽기 실패 " + e.toString());
		}
	}

	public void upd_db(int b_code, int code, String name, String release, int audience,
                                                    Double ratio, Double rating, String website) {
		try {
			/*****************************************************************
			 * conn.prepareStatement() : Statement 생성 *
			 * >> update 쿼리 수행을 위한 Statement 객체를 생성합니다.
			 * 쿼리문 수행 *
			 * >> JDBC에서 쿼리를 작성할 때 세미콜론 (;)을 빼고 작성합니다.
			 * >> *로 모든 칼럼을 읽어오는 것보다 가져와야 할 칼럼을 직접 명시해주는게 좋다고 합니다.
			 * >> 쿼리를 한 줄로 쓰기 어려운 경우 들여쓰기를 사용해도 상관 없습니다.
			 * >> pstmt.executeUpdate() : 값이 할당되면  이 메서드를 사용해서 update 쿼리를 실행합니다.
			 * >> count : 반환 값은 영향을 미친 row 개수입니다.
			 * >>                값이 입력되지 않았으면 count값이 0이므로 변경 실패를 출력합니다.
			 *****************************************************************/
			String sql = "UPDATE MOVIE_THEATER "
				   + "SET MOVIE_CODE = ?, MOVIE_NAME = ?, MOVIE_RELEASE = ?,
				      MOVIE_AUDIENCE = ?, " + "MOVIE_RATIO = ?, MOVIE_RATING = ?,
				      MOVIE_WEBSITE = ?" + "WHERE MOVIE_CODE = ?";

			PreparedStatement pstmt = conn.prepareStatement(sql);

			pstmt.setInt(1, code);                          pstmt.setString(2, name);
			pstmt.setString(3, release); pstmt.setInt(4, audience);
			pstmt.setDouble(5, ratio);              pstmt.setDouble(6, rating);
			pstmt.setString(7, website); pstmt.setInt(8, b_code);

			int count = pstmt.executeUpdate();
			if (count == 0)                 System.out.println("데이터 변경 실패");
			else                                    System.out.println("데이터 변경 성공");

		}
		catch (SQLException e) {
			e.printStackTrace();
			state.setText("DB 읽기 실패 " + e.toString());
		}
	}

	public static void main(String[] args)
	{
		new GUI();
	}
}
```

동영상은 용량이 너무 커서 유튜브 영상을 업로드한 것을 제출하겠습니다.

8. 소감 및 느낀점

- 과제를 수행하는데 소요된시간
  약 30시간 [12/05(2), 12/06(2), 12/19(2), 12/20(10), 12/21(15)]

- 개념 공부가 필요하다고 느낀 부분
  1:1, 1:n, n:m 관계 식별할줄 알기
  ERwin에서 식별/비식별 구분하는 방법
  테이블 설계(스키마 작성, 정규화 등)

과제를 수행하면서 힘들었던점은 자바는 어느정도 사용해봤는데 스윙을 사용해본적은 과제3에서 사용해보긴 했지만 진짜 너무 오랜만이라서 버튼이랑 패널 넣는것부터 책이랑 인터넷을 찾아보느라 시간을 소모했던 것 같습니다. 그리고 자바로 데이터베이스 연동하는것도 문제가 있었습니다. 쿼리 수행을 위한 객체와 메소드 사용법을 찾아보는 것은 확실히 시간이 많이 걸렸습니다. 그래도 과제를 진행하면서 문제가 있지는 않았는데 힘들었던점은 시험끝나고 주어진? 과제 기간이 짧아서 밤새 컴퓨터 모니터를 보고 있어야 했는데, ERwin 그리는것과 정규화는 머리가 아팠고 테이블을 생성하고 삽입할 데이터가 많아서 육체적/정신적으로 두배로 힘든 과제였던 것 같습니다. 그리고 마지막 과제를 마치면서 아무래도 과제1을 끝낼때랑은 다르게 ERwin 그리는것도 익숙하게 그릴수있었고 SQL문 사용하는것도 빨라졌고 정규화 그리는 것도 슥슥 그릴 수 있어서 신기했습니다.

항상 느끼는거지만 특히나 이번 과제는 제출하기전 검토할 시간이 부족했습니다. 원래 12/08까지 과제를 끝내고 시험공부를 하려고 했는데 다른 과목의 과제랑 시험이 겹쳐버려서 부득이하게 과제를 시험 끝나고 부랴부랴 급하게 마치게 되었습니다. 과제를 제출하기전에 몇 번씩 보고 제출했었는데 이번엔 그렇지 못해서 아쉽습니다. 그런데 과제를 하면서 교수님이 과제를 한주만 빨리 내줬어도 과제가 좀 더 수월했을거라는 생각도 해봤지만 어차피 과제가 빨리나왔으면 빨리나온만큼 느긋느긋하느라 비슷한 결과가 나왔을 것 같기도 해서 의도치않게 지금보다 더 열심히 해야겠다는 자기반성을 할 수 있는 기회도 가지게 되었습니다.

마지막으로 교수님! 한학기 수업하느라 고생하셨습니다. 다음학기에 뵙겠습니다!

9. 참고문헌 및 참고사이트

- 과제1, 과제2, 과제3

- 명품 JAVA Programming, 황기태. 김효수 지음

- 영화사이트
http://www.lottecinema.co.kr/LCHS/Contents/Movie/Movie-List.aspx

- 기타 사이트
https://m.blog.naver.com/PostView.nhn?blogId=jihoon8912&logNo=220238842336&proxyReferer=https%3A%2F%2Fwww.google.com%2F
https://beautifulkim.tistory.com/353
http://mwultong.blogspot.com/2006/10/java-string-to-number-int-float-double.html
https://okky.kr/article/49654
https://opentutorials.org/module/3569/21222
https://victorydntmd.tistory.com/145
https://m.blog.naver.com/heoguni/130170492705

+ 추가로 데이터베이스 테이블 작성 코드를 txt파일로 제출하겠습니다