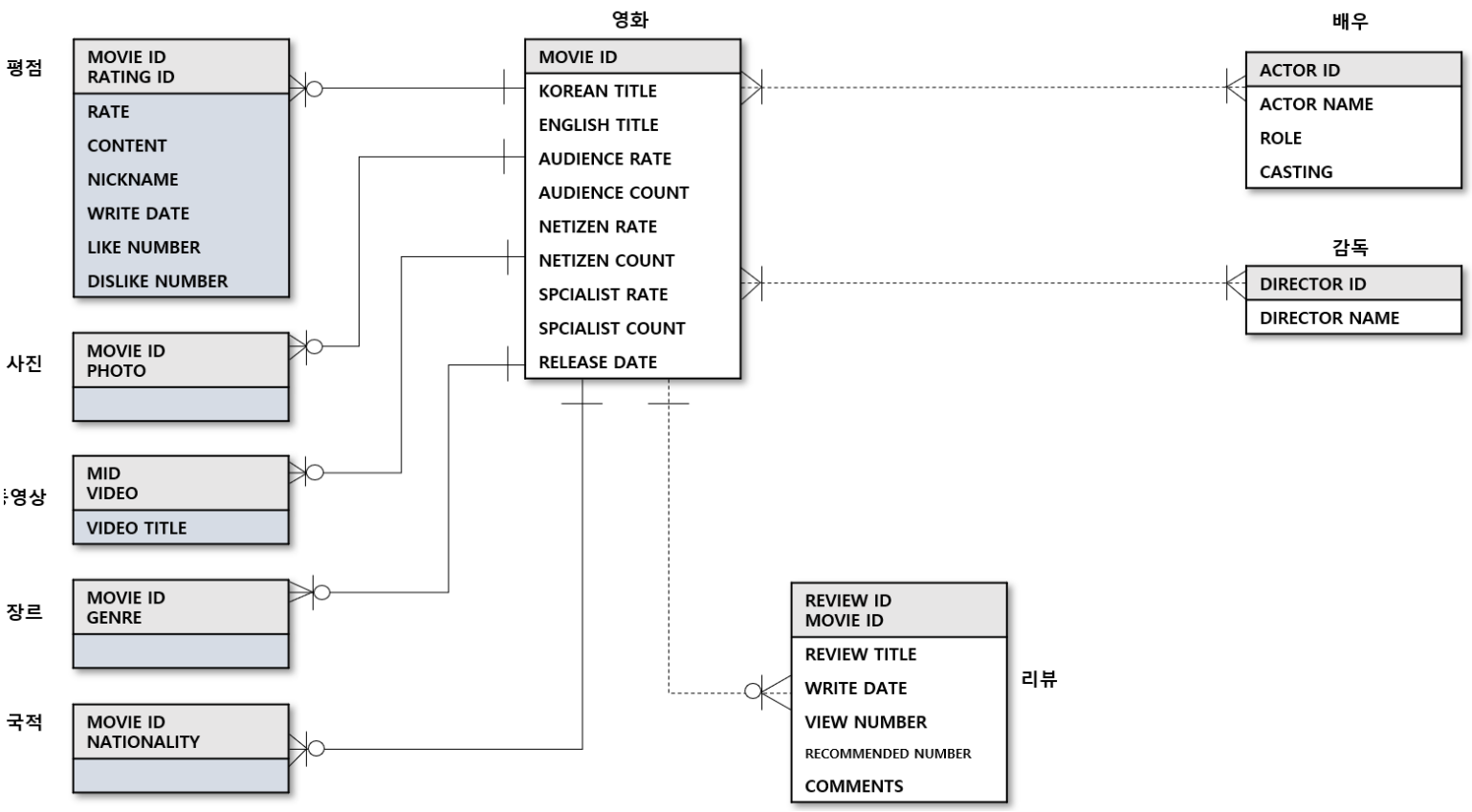
데이터베이스 기말 프로젝트 보고서

컴퓨터공학부 3학년 202011376 조현서(분반3183)

1. 문제 정의

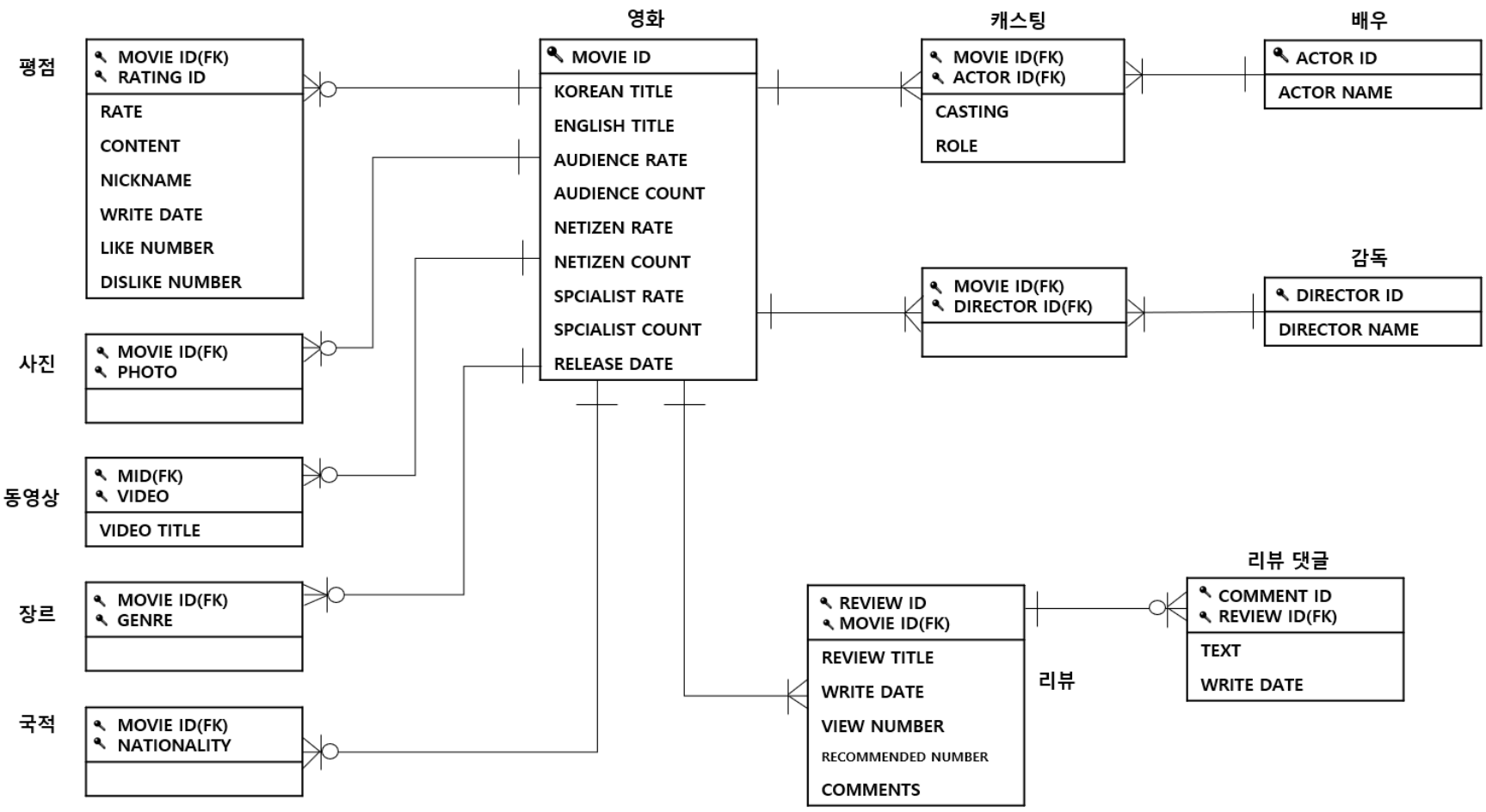
- 우리가 일상에서 마주치는 다양한 문제들을 해결하는데 있어 데이터 베이스의 역할은 매우 중요합니다.
- 이러한 데이터 베이스를 ER 모델링과 관계형 다이아그램, 관계형 데이터 베이스로 효율적인 구조로 구축하여 성능이 좋은 검색 시스템을 만들 수 있습니다.
- 효율적인 구조와 더불어 자주 검색하는 속성들을 고려하여 적절한 인덱싱을 이용한다면 보다 좋은 성능을 보일 수 있습니다.
- 이러한 과정의 전처리 과정으로 원하는 정보를 크롤링하여 데이터 베이스화 할 수 있습니다.
- 위 점들을 종합적으로 고려하여 네이버 영화 페이지로부터 데이터를 수집하여 데이터 베이스화 한 후 효과적인 검색 엔진을 설계하는 것이 이번 과제의 목표입니다.
- 본 과제를 수행함에 있어 Window 10 Education 운영체제, 12th Gen Intel(R) Core(TM) i5-12400 CPU와 아나콘다로 구축한 파이썬 3.6.13 가상 환경을 이용하여 수행하였습니다.

2. ER Modeling



- ER 모델링을 시작으로 구조적인 데이터 베이스 구축을 시작합니다.
- 문제 해결을 위한 영화를 중심으로 한 ER 모델링은 위와 같이 나타낼 수 있습니다.
- 평점, 사진, 동영상, 장르 및 국적은 영화의 특성이지만 여러 개의 값을 가질 수 있어 새로운 개체로 정의를 하였습니다.
- 이 과정에서 이들의 정의가 영화에 의존적일 수 밖에 없으므로 ID-Dependent하며 weak한 개체로 정의하였습니다.
- ID dependent하므로 원래는 둥근 모서리를 가져야 하는 이러한 개체들을 보다 직관적으로 구분되기 쉽게하기 위해 다른 색으로 표현해 봤습니다.
- 그 외의 배우, 감독, 영화 및 리뷰는 Strong entity로 표현하였습니다.
- 하지만 영화에 출연해야 배우라고 할 수 있고, 영화를 만들어야 감독이라고 할 수 있으며 영화 또한 배우와 감독이 모두 한 명 이상 씩은 있어야 만들어 질 수 있다고 생각하여 최소는 1이라고 정의하였으며, 감독과 배우는 여러 영화들을 만들 수도, 연기를 할 수도 있다고 생각하여 최대 n개의 관계를 가질 수 있다고 정의하였습니다.
- 리뷰의 경우에는 Strong entity로 정의하였습니다.

3. Relational Diagram



- 앞에서 정의한 ER 모델링을 이용하여 관계형 다이어그램으로 구체화하여 나타낸 것입니다.
- 이전과는 달리 리뷰의 속성 중 리뷰에 달린 댓글과 리뷰와의 관계는 1:n 관계를 가지기 때문에 이 둘을 리뷰 댓글 table에 외래 키를 이용하여 정의합니다.
- 이 때 리뷰의 id는 직접 crawl한 정보로 그 자체로 unique하지만, mid와 함께 primary key로 정의하여 mysql에서 clustered index로 mid 정보를 같이 사용할 수 있게 되어 검색 속도를 향상시키고자 하였습니다.
- 또한 앞에서의 ER Diagram에서보다 배우 - 영화 관계, 감독 - 영화 관계를 더 구체적으로 표현하였습니다.
- 두 관계 모두 m:n 관계를 띄지만, 배우 - 영화 사이에는 배우의 역할 및 주연 혹은 조연 정보 등의 추가적인 정보가 필요하기 때문에 association table로 관계를 나타내며 감독 - 영화 사이의 관계는 추가적인 정보가 필요하지 않기 때문에 중간에 intersection table로 정의하였습니다.
- 나머지 weak한 개체들은 영화의 id를 외래키로 이용하여 id-dependent한 속성들을 구체화할 수 있습니다.

4. 테이블 생성 코드

```
create table movie(
    mid int primary key,
    kor_title varchar(50),
    eng_title varchar(50),
    aud_rate float,
    aud_count int,
    net_rate float,
    net_count int,
    spc_rate float,
    spc_count int,
    rel_date datetime
);
```

```
113 •  select count(*) as mvcount from movie;
```

Result Grid | Filter Rows: | Export:

mvcount
24636

```
create table director(
    did int primary key,
    dname varchar(50)
);
```

```
115 •  select count(*) as drcount from director;
```

Result Grid | Filter Rows: | Export: | Wra

drcount
11221

```
create table actor(
    aid int primary key,
    aname varchar(50)
);
```

```
114 •  select count(*) as atcount from actor;
```

Result Grid | Filter Rows: | Export:

atcount
41342

- 앞에서 정의한 대로 primary key, foreign key, attribute들을 반영하여 생성한 코드입니다.

- 각각의 attribute들에 해당하는 적절한 형태로 선언을 해 주며 각각의 table 수는 영화가 24363개, 배우가 41342개, 감독이 11221개 인 것을 확인할 수 있습니다.

```
create table genre(
    mid int,
    genre varchar(10),
    primary key(mid, genre),
    foreign key(mid) references movie(mid)
);
```

gcount
32175

```
create table photo (
    mid int,
    photo  varchar(200),
    primary key(mid, photo),
    foreign key(mid) references movie(mid)
);
```

pcount
80075

```
create table video(
    mid int,
    video  varchar(200),
    title varchar(200),
    primary key(mid, video),
    foreign key(mid) references movie(mid)
);
```

vicount
23605

```
create table review(
    mid int,
    rid int,
    primary key(rid),
    foreign key(mid) references movie(mid) ,

    title varchar(1000),
    wr_date datetime,
    view_num int,
    rec_num int
);
```

rvcount
43082

```
create table rating(
    mid int,
    rid int,
    primary key(mid, rid),
    foreign key(mid) references movie(mid) ,

    rate int,
    content varchar(1000),
    nickname varchar(50),
    wr_date datetime,
    like_num int,
    dis_num int
);
```

rtcount
66317

```
create table actor_movie(
    mid int,
    aid int,
    primary key(mid, aid),
    foreign key(mid) references movie(mid) ,
    foreign key(aid) references actor(aid),
    rol varchar(50),
    casting varchar(50)
);
```

amcount
112086

```
create table director_movie(
    mid int,
    did int,
    primary key(mid, did),
    foreign key(mid) references movie(mid) ,
    foreign key(did) references director(did)
);
```

dmcount
22828

5. 데이터 베이스의 생성 - Crawling and Inserting

- 데이터 베이스를 생성하기 위해서 크롤링이 필요하였습니다.
- 크롤링을 한 데이터는 평점순으로 정렬한 랭킹 차트와 한국 영화들에서 수집하였습니다.
- 이때, 랭킹 차트에 있는 영화들에서는 네이버에서 추천하는 관련 영화들 또한 수집하여 총 약 2만 5천 개 정도의 데이터를 수집할 수 있었습니다.

```
insert_sql1 = """insert ignore into movie(mid, kor_title, eng_title, aud_rate, aud_count, net_rate, net_count, spc_rate, spc_count, rel_date)
                values(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)"""

insert_sql2 = """insert ignore into genre(mid, genre) values(%s, %s)"""

insert_sql3 = """insert ignore into nationality(mid, nationality) values(%s, %s)"""
insert_sql4 = """insert ignore into photo(mid, photo) values(%s, %s)"""
insert_sql5 = """insert ignore into video(mid,video,title) values(%s, %s, %s)"""
insert_sql6 = """insert ignore into rating(mid, rid,rate, content, nickname, wr_date, like_num,dis_num)
 values(%s, %s,%s, %s,%s, %s,%s, %s)"""

insert_sql7 = """insert ignore into review(mid, rid, title,  wr_date, view_num,rec_num ) values(%s, %s,%s, %s, %s,%s)"""
insert_sql8 = """insert into comments(rid, cid,txt,wr_date) values(%s, %s, %s,%s)"""

insert_sql9 = """insert ignore into director(did, dname) values(%s, %s)"""
insert_sql10 = """insert ignore into actor(aid, aname) values(%s, %s)"""
insert_sql11 = """insert ignore into director_movie(mid, did) values(%s, %s)"""
insert_sql12 = """insert ignore into actor_movie(mid, aid, rol,casting) values(%s, %s, %s,%s)"""
cur.executemany(insert_sql1, movie_info)
cur.executemany(insert_sql2, genre_info)
cur.executemany(insert_sql3, nationality_info)
cur.executemany(insert_sql4, photo_info)
cur.executemany(insert_sql5, video_info)
cur.executemany(insert_sql6, rating_info)
cur.executemany(insert_sql7, review_info)
cur.executemany(insert_sql8, comments_info)
cur.executemany(insert_sql9, director_info)
cur.executemany(insert_sql10, actor_info)
cur.executemany(insert_sql11, director_movie_info)
cur.executemany(insert_sql12, actor_movie_info)
conn.commit()
```

- Inserting을 수행할 때에는 한 페이지에 대한 crawling을 수행할 때 마다 execute many를 이용하여 list에 저장해 뒀던 데이터들을 batch 처리하여 하나씩 처리한 경우보다 더 속도를 개선 하였습니다.

- Insert를 수행하다가 기본키가 중복되는 배우, 영화, 감독 등의 정보가 있기 때문에 이 경우 ignore로 처리하여 에러가 발생하지 않도록 하였습니다. 영화가 많이 겹친 이유는 추천 관련 영화가 매우 많이 겹쳤습니다.
- 저는 이 과정에서 파이썬 라이브러리인 pymysql, requests, BeautifulSoup와 webdriver 프레임 워크들을 사용하였습니다.

6. 인덱스를 이용한 속도 개선

- 검색 효율에 있어 index를 생성하면 검색을 할 때에 각각의 조건에 맞는 투플에 대하여 테이블의 페이지 read 비용 발생하여 full table search보다 시간이 단축됩니다.

```
sql = """
    select kor_title, eng_title, rel_date, net_rate, m.mid
    from movie m, genre g
    where m.mid = g.mid and genre like %s
    order by kor_title
"""
```

- 특히 저는 본 과제를 수행함에 있어 영화 명, 배우 명 등의 검색어가 완전한 형태가 아니더라도 검색이 가능하도록 위와 같은 형태의 like문을 사용하였는데, 이처럼 시작하는 단어를 알 경우 index가 있으면 index가 없는 경우보다 훨씬 더 성능이 좋아집니다.
- 검색을 할 때에는 검색된 데이터가 너무 많을 수 있으므로 fetchall을 사용하지 않고 fetchone을 null이 될 때 까지 수행하여 이를 반복하여 컴퓨터가 사용하는 메모리가 지나치게 커지지 않도록 할 수 있습니다.

```
row = cur.fetchone()
while row:
    search_result.append(
        [row['kor_title'], row['eng_title'], row['rel_date'], row['net_rate'], row['mid']])
    row = cur.fetchone()
```

7. 시행 예

시작 화면 입니다.

배우명 검색에서 장동 으로 검색을 하면

정렬방식 페이지로 이동하며

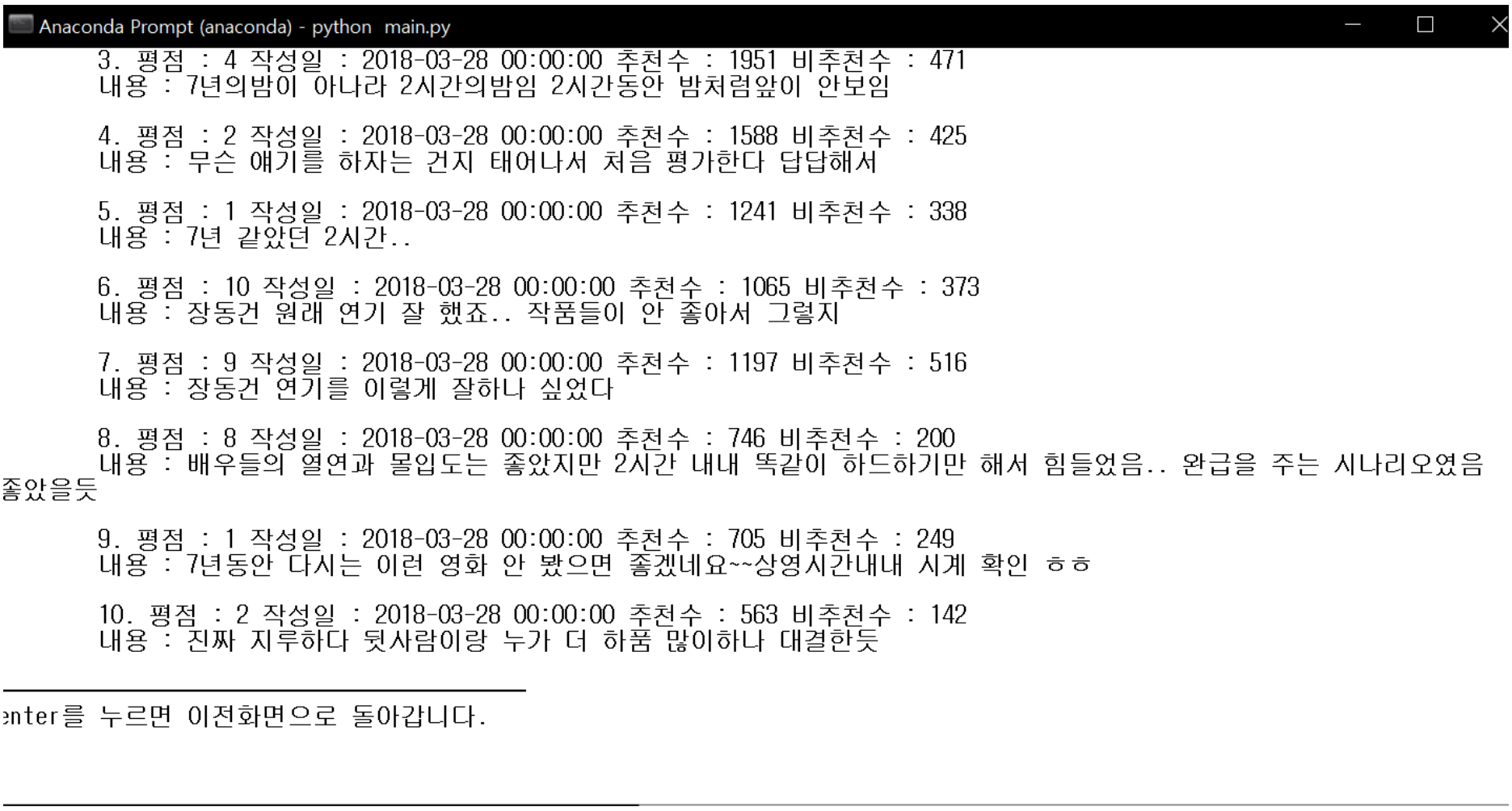
```
Anaconda Prompt (anaconda) - python  main.py
        165.    영구와 땡칠이   Yong-Gu And Daeng Chiri 1989    9.04
        166.    영구와 땡칠이 2 - 소림사 가다   영구와 땡칠이 소림사 가다: Yong-Gu And Taeng-Chiri Go To Sori   1989   9
.06
        167.    속사정 쌀롱                             0000    9.15
        168.    첨밀밀          甛蜜蜜, Comrades: Almost A Love Story   1997    9.15
        169.    태극기 휘날리며 TaeGukGi: Brotherhood Of War    2021    9.21
        170.    만무방          Manmubang       1994    9.29
        171.    성난 능금                               0000    9.5
        172.    심장이 뛴다                             0000    9.59
        173.    내 친구의 집은 어디인가                                0000    9.6
        174.    렛츠고 시간탐험대 2                                    0000    9.69
        175.    그 여자                         0000    9.91
        176.    무작정 패밀리 시즌3                                    0000    10.0
        177.    노래로 응답하라 - 응급남녀편                                   0000    10.0
        178.    나홀로 연애중                           0000    10.0
        179.    빅프렌드                                0000    10.0
        180.    사랑하기 좋은 날                                       0000    10.0
        181.    갑돌이와 갑순이 Gap-Dol and Gap-Sun     1972    10.0
        182.    동경 사자와 명동 호랑이                                0000    10.0
        183.    마지막 황태자 영친왕                                   0000    10.0
        184.    선배            The Last Heist  1979    10.0
        185.    수색대          수색대  0000    10.0
        186.    5인의 사형수    오인의 사형수   0000    10.0
        187.    1950년 4시                              0000    10.0

        정렬방식 : 평점 순
--------------------------------------------------------------------------------
자세히 보고싶은 영화의 번호를 입력하세요 :
```

이전 페이지에서 평점 순 정렬을 입력하였기 때문에 평점 순으로 정렬 된 것을 확인할 수 있습니다(맨 오른쪽 소수)

```
Anaconda Prompt (anaconda) - python  main.py
--------------------------------------------------------------------------------
    데이터베이스 과제 - 202011376 조현서
--------------------------------------------------------------------------------
영화 정보

        한국 제목 : 7년의 밤
        영어 제목 : Seven Years of Night
        감독 : 추창민|
        출연
        장동건 :(주연) 주연역
        최광일 :(조연) 현수부역
        정석용 :(조연) 박소장역
        이상희 :(조연) 무녀역
        문창길 :(조연) 임영감역
        김정팔 :(조연) 장반장역
        정인겸 :(조연) 박수무당역
        성병숙 :(조연) 하영모역
        류승룡 :(주연) 최현수역
        송새벽 :(주연) 안승환역
        전배수 :(조연) 현태역
        여무영 :(조연) 하영부역
        유순웅 :(조연) 무녀부역
        서현우 :(조연) 이형사역
        고경표 :(주연) 최서원역
        정준원 :(조연) 어린현수역
        이레 :(조연) 오세령역
        탕준상 :(조연) 어린 서원역
        우미화 :(조연) 현수모역

        관람객 수 :436  관람객 평점 :6.01
        네티즌 수 :5054  네티즌 평점 :4.77
        전문가 수 :8  전문가 평점 :5.5
        장르 : 드라마|스릴러|
--------------------------------------------------------------------------------

        다음 번호를 누르면 상세 페이지로 이동합니다.
        1. 사진
        2. 동영상
        3. 평점
        4. 리뷰
        5. 처음으로
--------------------------------------------------------------------------------
상세 페이지 번호 :
```

여기서 원하는 페이지를 누르면 영화 정보 페이지로 이동하게 됩니다

정보 페이지에서 사진, 동영상, 평점 및 리뷰 등의 상세 페이지로 이동할 수 있습니다.



3번을 눌러 평점을 확인한 예입니다. 평점 페이지에서는 평점, 작성일 및 추천 수와 비추천 수를 확인할 수 있습니다.

다른 상세 페이지도 이와 유사하게 작동합니다.

붙임. naver_movie.zip(crawl.py, main.py, naver_movie.sql). 끝.