2023-1 데이터베이스 기말과제

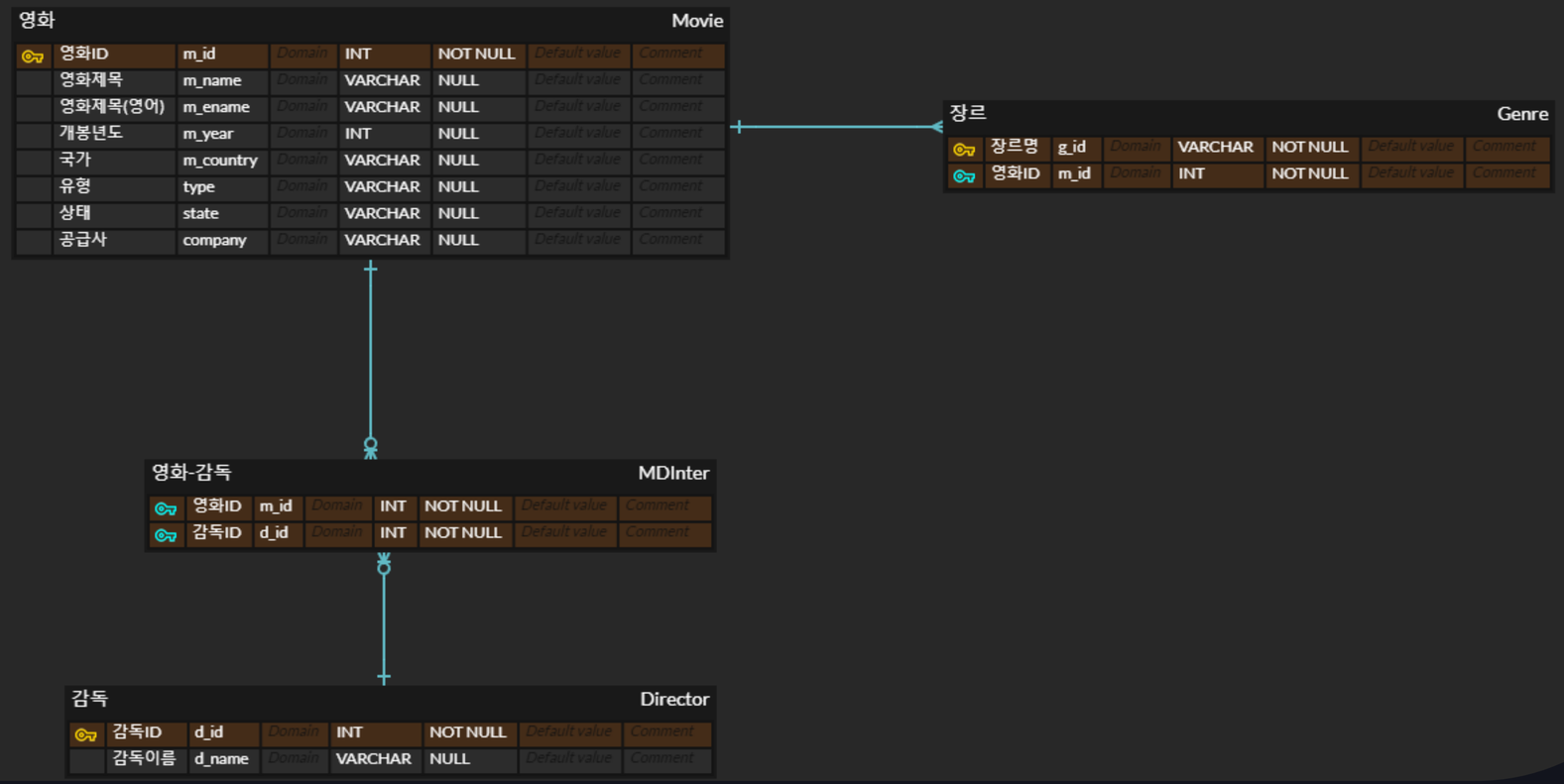
영화 정보 DB 구축 및 검색

소프트웨어학과

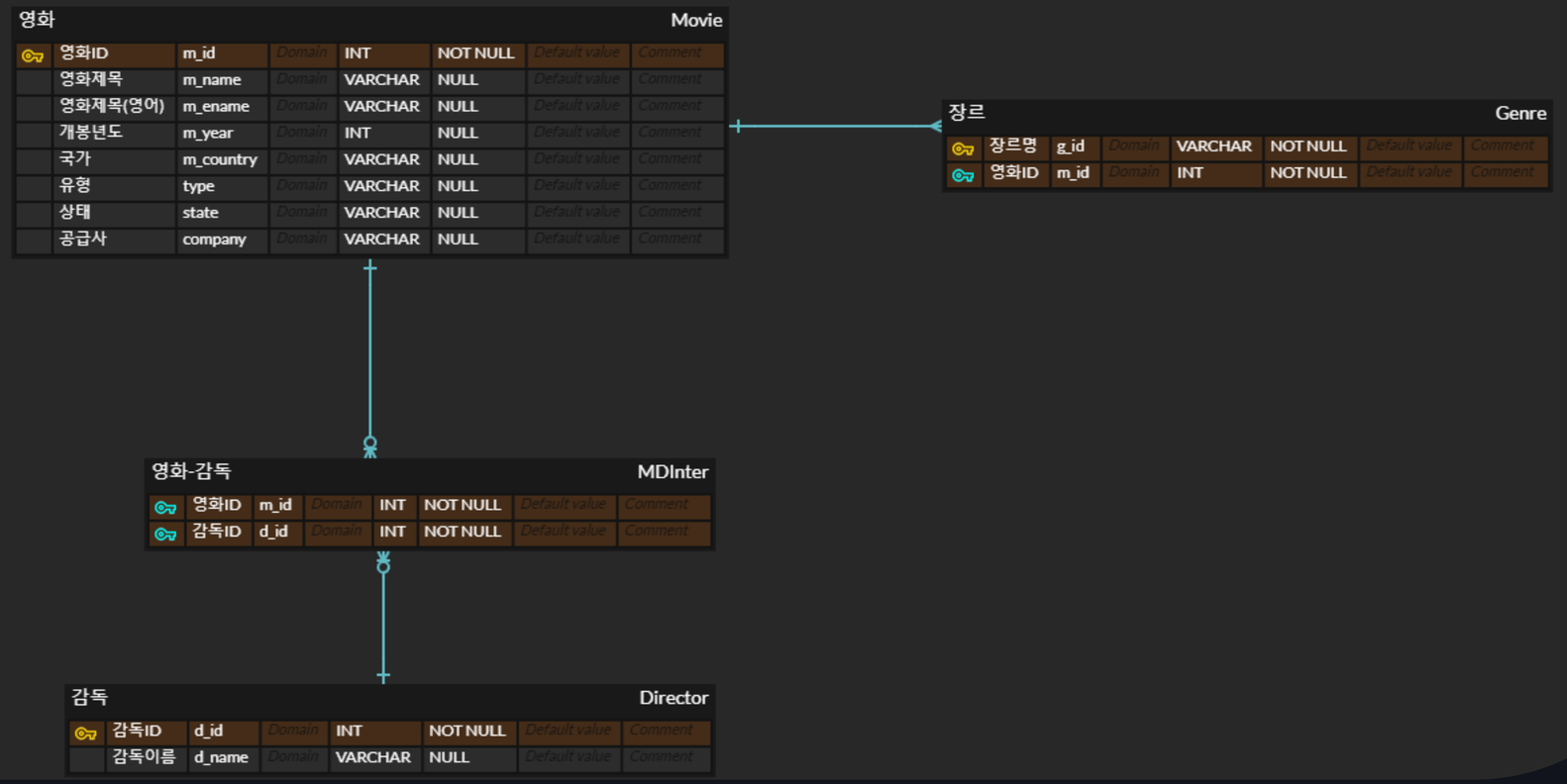
201711357 최병찬

목차

ERD


영화
Movie
영화ID m_id Domain INT NOT NULL Default value Comment
영화제목 m_name Domain VARCHAR NULL Default value Comment
영화제목(영어) m_ename Domain VARCHAR NULL Default value Comment
개봉년도 m_year Domain INT NULL Default value Comment
국가 m_country Domain VARCHAR NULL Default value Comment
유형 type Domain VARCHAR NULL Default value Comment
상태 state Domain VARCHAR NULL Default value Comment
공급사 company Domain VARCHAR NULL Default value Comment
장르
Genre
장르명 g_id Domain VARCHAR NOT NULL Default value Comment
영화ID m_id Domain INT NOT NULL Default value Comment
영화-감독
MDInter
영화ID m_id Domain INT NOT NULL Default value Comment
감독ID d_id Domain INT NOT NULL Default value Comment
감독
Director
감독ID d_id Domain INT NOT NULL Default value Comment
감독이름 d_name Domain VARCHAR NULL Default value Comment

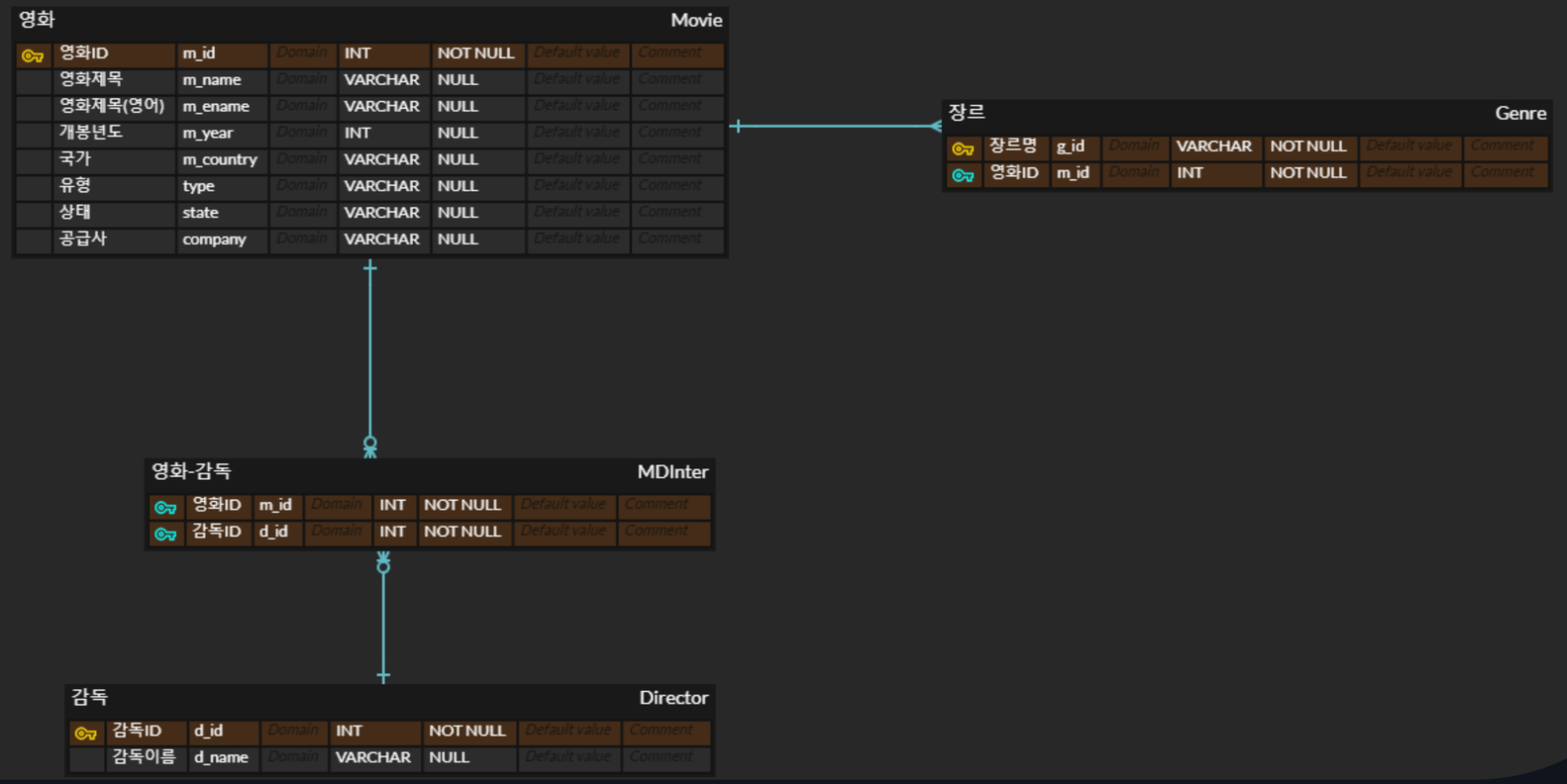
ERD



Movie - Direction ; M:N Strong Relationship

→ Intersection Table 생성

ERD



Movie - Genre ; multi-valued attribute (1:N)

→ **Genre Table 생성 (ID-Dependent)**

PYTHON CODE : 테이블 생성

def create_table()

총 4개의 테이블 생성

Movie , Director , Genre , mdInter

Movie , Director : PK로 ID 생성
Genre : 식별자로 MovieID와 GenreName
mdInter : 식별자로 MovieID와 DirectorID

PYTHON CODE : 데이터 삽입

def xls_to_table()

pandas module : excel 데이터 추출

예외처리 : 중간테이블 중복 값

동일한 이름의 감독은 동일한 ID로 설정

PYTHON CODE : 데이터 삽입

예외처리 : 중간테이블 중복 값

```
sql = f"Select * from mdinter where m_id = {m_id} and d_id ={d_id}"
cursor.execute(sql)

if cursor.fetchone() is None :
    sql = "INSERT INTO mdinter (m_id,d_id) values (%s,%s)"
    val =(m_id,d_id)
    cursor.execute(sql,val)
else :
    print(m_name , d_name , m_id , d_id)
```

PYTHON CODE : 데이터 삽입

동일한 이름의 감독은 동일한 ID로 설정

```
sql = "SELECT d_id from director where d_name = %s"
val = (d_name,)
cursor.execute(sql,val)
res = cursor.fetchone()

#감독 아이디가 없는 경우
if res is None :
# 감독 아이디가 이미 존재하는 경우
else :
    sql = f"Select * from mdinter where m_id = {m_id} and d_id ={res['d_id']}"
    cursor.execute(sql)
```

PYTHON CODE : 인덱스 설정

def create_index()

검색에 자주 쓰이는 어트리뷰트에 대한 인덱스 생성

영화 , 감독 , 장르명

PYTHON CODE : 인덱스 설정

검색에 자주 쓰이는 어트리뷰트에 대한 인덱스 생성

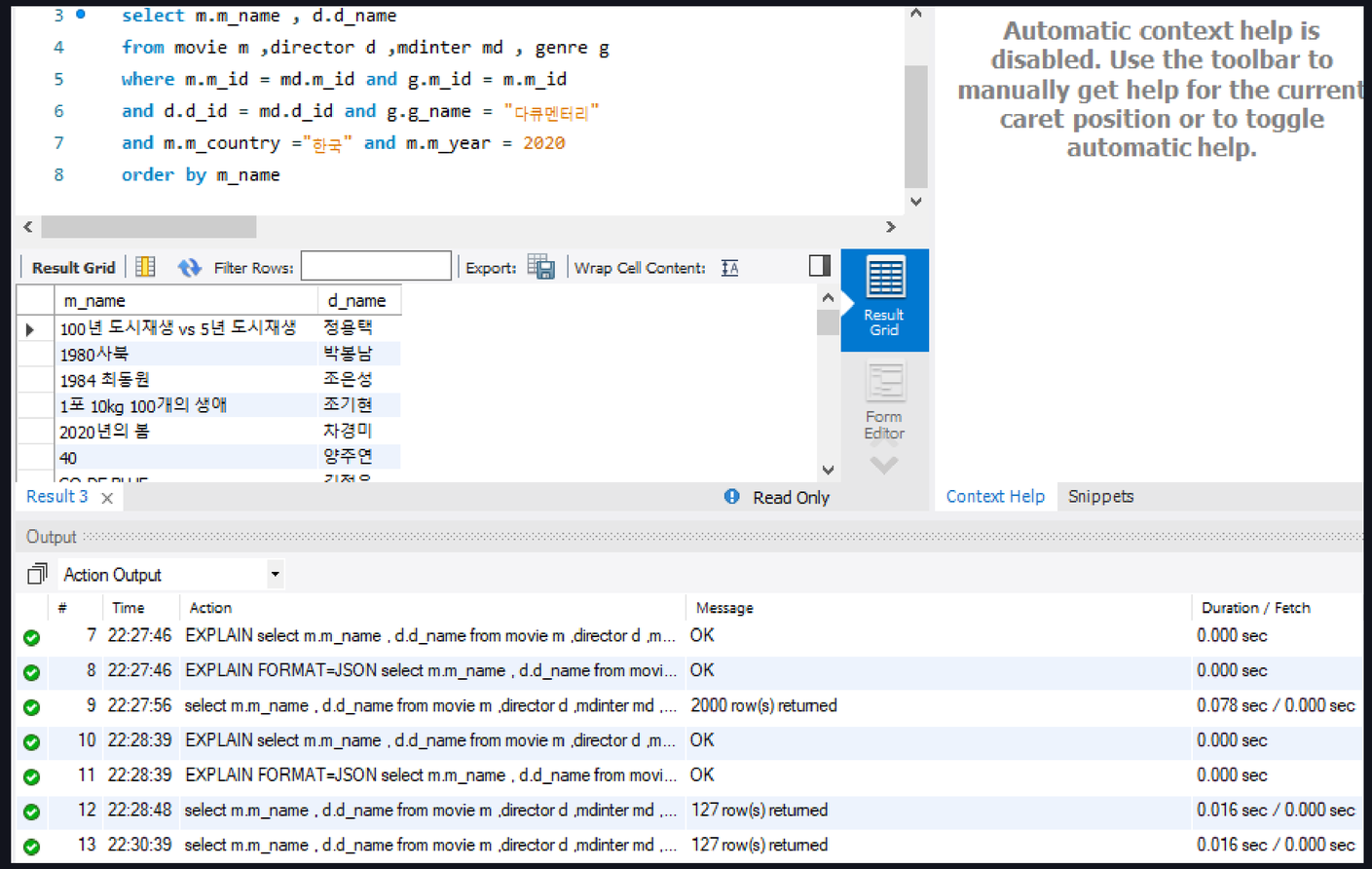
```
def create_idx() :
    conn ,cursor = sql_connect()

    cursor.execute("CREATE INDEX idx_m_name on movie (m_name)")
    cursor.execute("CREATE INDEX idx_d_name on director (d_name)")
    cursor.execute("CREATE INDEX idx_g_name on genre (g_name)")

    commit_n_close(conn)
```

Workbench 결과

1. 2020년에 제작된 다큐멘터리 한국 영화의 영화명과 감독을 영화명 순으로 검색하라



```
3  select m.m_name , d.d_name
4  from movie m ,director d ,mdinter md , genre g
5  where m.m_id = md.m_id and g.m_id = m.m_id
6  and d.d_id = md.d_id and g.g_name = "다큐멘터리"
7  and m.m_country ="한국" and m.m_year = 2020
8  order by m_name
```

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

m_name	d_name
100년 도시재생 vs 5년 도시재생	정용택
1980사북	박봉남
1984 최동원	조은성
1포 10kg 100개의 생애	조기현
2020년의 봄	자경미
40	양주연

Result Grid | Form Editor

Result 3 | Read Only | Context Help | Snippets

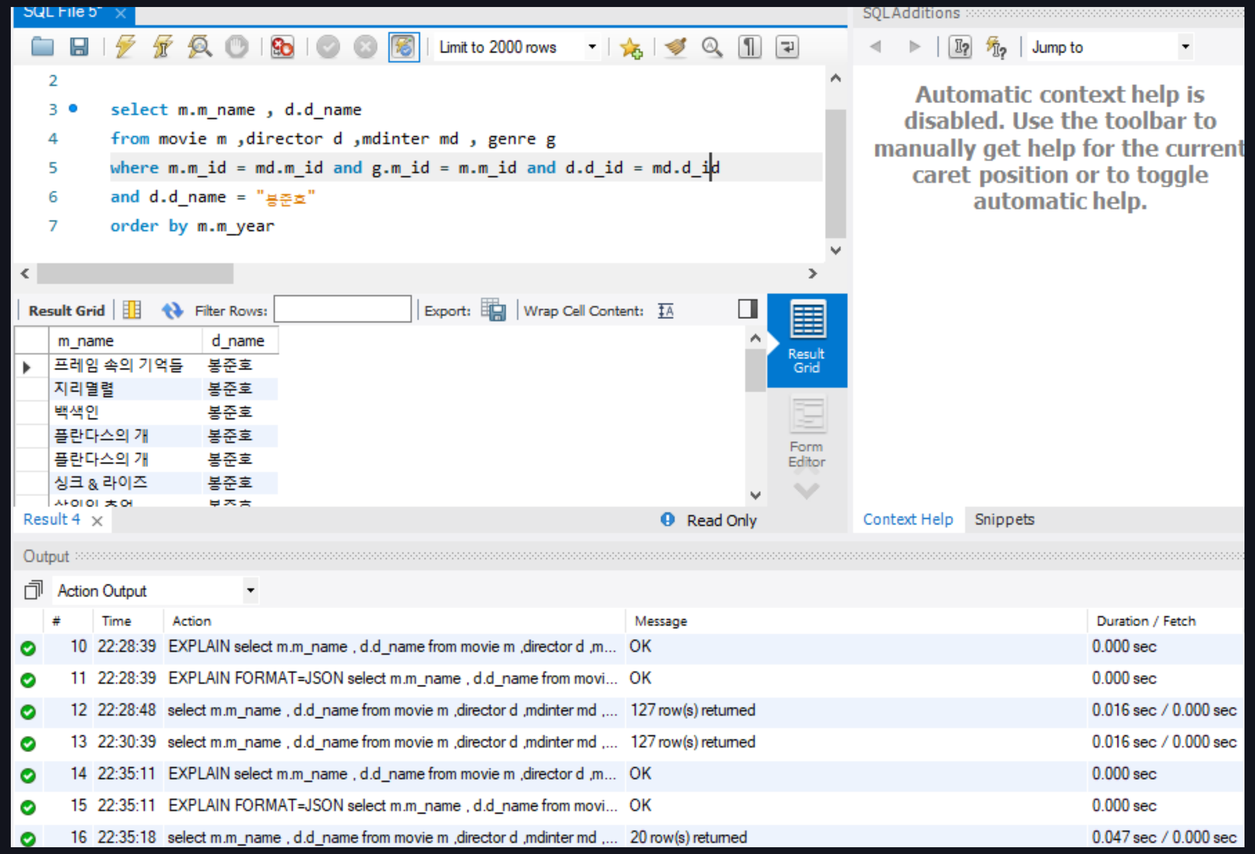
Output

Action Output

#	Time	Action	Message	Duration / Fetch
7	22:27:46	EXPLAIN select m.m_name , d.d_name from movie m ,director d ,m...	OK	0.000 sec
8	22:27:46	EXPLAIN FORMAT=JSON select m.m_name , d.d_name from movi...	OK	0.000 sec
9	22:27:56	select m.m_name , d.d_name from movie m ,director d ,mdinter md ,...	2000 row(s) returned	0.078 sec / 0.000 sec
10	22:28:39	EXPLAIN select m.m_name , d.d_name from movie m ,director d ,m...	OK	0.000 sec
11	22:28:39	EXPLAIN FORMAT=JSON select m.m_name , d.d_name from movi...	OK	0.000 sec
12	22:28:48	select m.m_name , d.d_name from movie m ,director d ,mdinter md ,...	127 row(s) returned	0.016 sec / 0.000 sec
13	22:30:39	select m.m_name , d.d_name from movie m ,director d ,mdinter md ,...	127 row(s) returned	0.016 sec / 0.000 sec

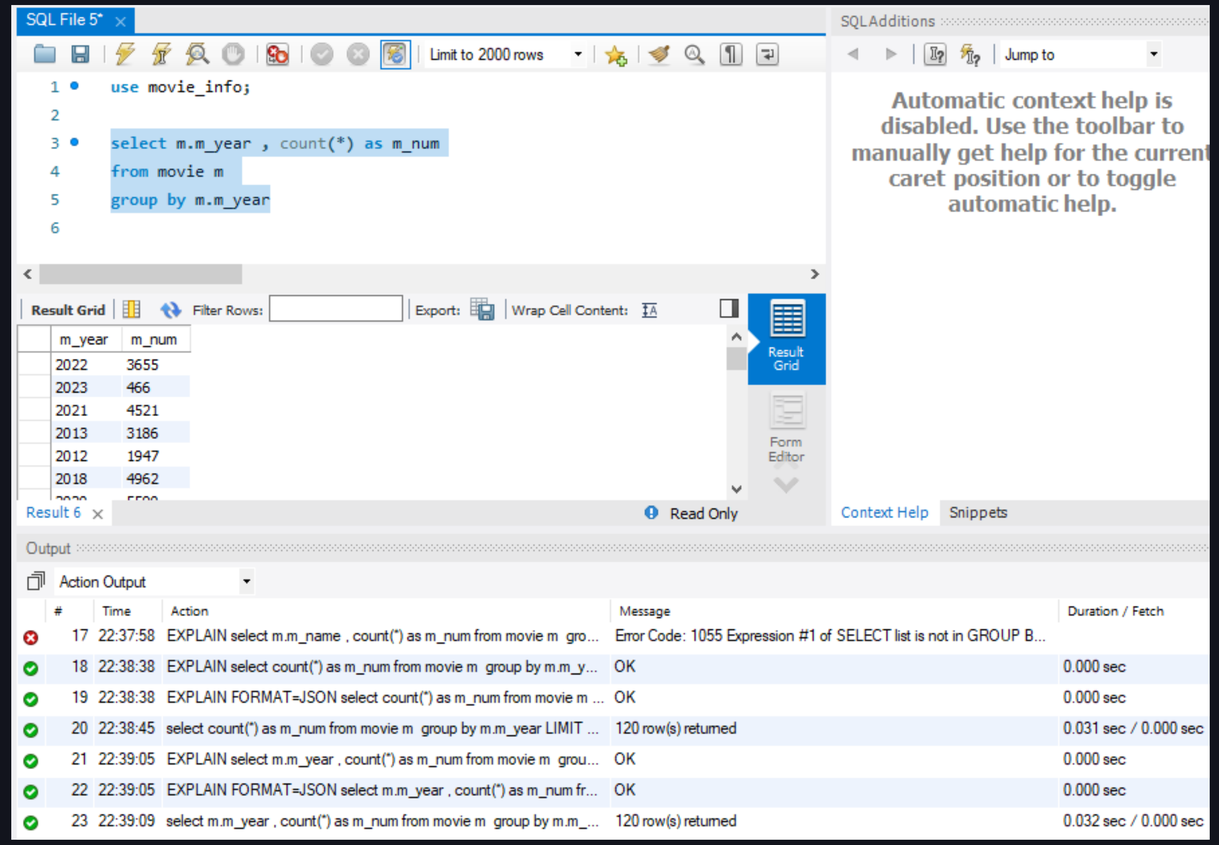
Workbench 결과

2. '봉준호' 감독의 영화를 제작년도 순으로 검색하라



Workbench 결과

3. 각 연도별 제작된 영화의 편수를 검색하되, 연도별로 출력하라



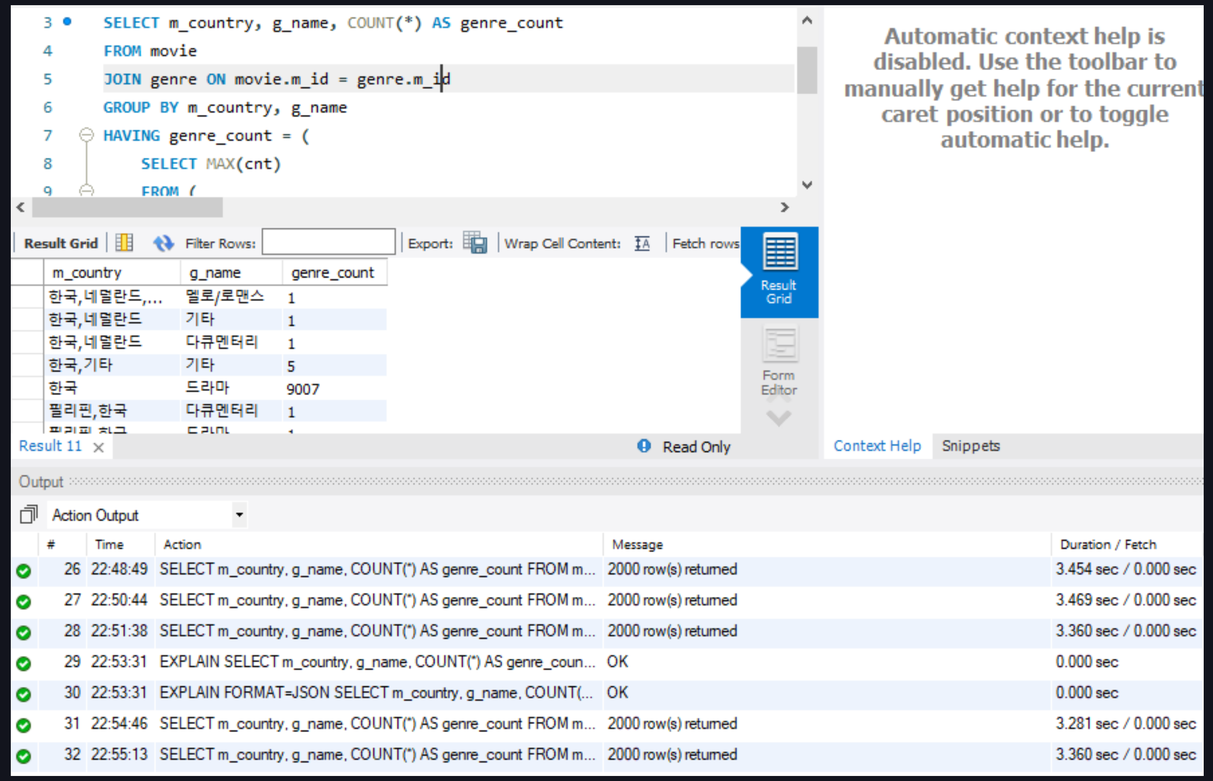
Workbench 결과

4. 각 나라별 가장 많이 제작된 영화 장르를 검색하라

```
SELECT m_country, g_name, COUNT(*) AS genre_count
FROM movie
JOIN genre ON movie.m_id = genre.m_id
GROUP BY m_country, g_name
HAVING genre_count = (
    SELECT MAX(cnt)
    FROM (
        SELECT m_country, COUNT(*) AS cnt
        FROM movie
        JOIN genre ON movie.m_id = genre.m_id
        GROUP BY m_country, g_name
    ) AS counts
    WHERE counts.m_country = movie.m_country
)
order by m_country desc
```

Workbench 결과

4. 각 나라별 가장 많이 제작된 영화 장르를 검색하라



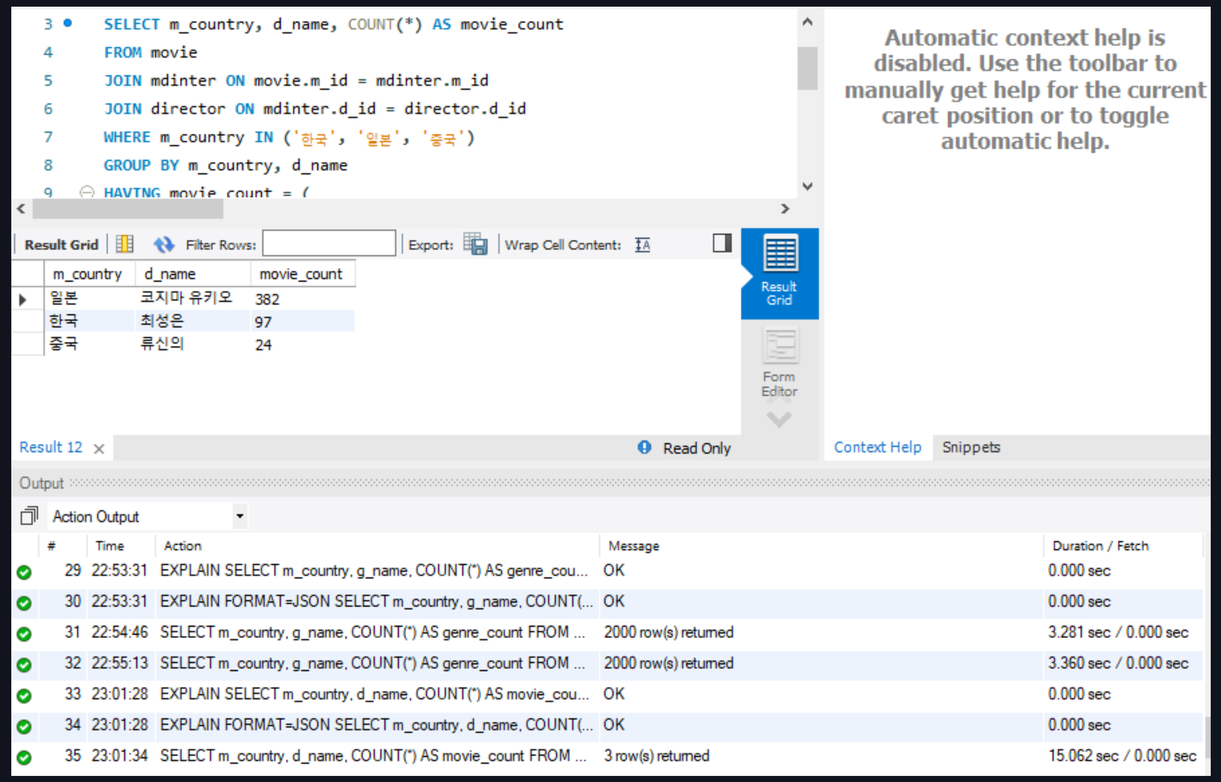
Workbench 결과

5. 한국 일본 중국 세나라에 대하여 각 나라별 영화를 가장 많이 감독한 감독의 이름을 검색하라

```
SELECT m_country, d_name, COUNT(*) AS movie_count
FROM movie
JOIN mdinter ON movie.m_id = mdinter.m_id
JOIN director ON mdinter.d_id = director.d_id
WHERE m_country IN ('한국', '일본', '중국')
GROUP BY m_country, d_name
HAVING movie_count = (
    SELECT MAX(cnt)
    FROM (
        SELECT m_country, COUNT(*) AS cnt
        FROM movie
        JOIN mdinter ON movie.m_id = mdinter.m_id
        JOIN director ON mdinter.d_id = director.d_id
        WHERE m_country IN ('한국', '일본', '중국')
        GROUP BY m_country, d_name
    ) AS counts
    WHERE counts.m_country = movie.m_country
)
```

Workbench 결과

5. 한국 일본 중국 세나라에 대하여 각 나라별 영화를 가장 많이 감독한 감독의 이름을 검색하라



프로그램 시연

사용자로부터 어트리뷰트 입력받고
해당 어트리뷰트에 대해 AND 연산 수행 후
검색결과 출력