Assignment 5 Write-up

The original movies data had several attributes which contained a list of items. For example, the ‘genre’ attribute might contain a list with id and name values for multiple genres. This violates first normal form rules, meaning that the data wasn’t even in 1NF. To solve this problem, two new relations had to be created. One would contain information about genre ids and names, and the other would contain information that linked together specific movies with the genres that originally described it. This same process had to be repeated for several other attributes of the data, including ‘keywords’ and ‘spoken_languages’. The good news is that once these adjustments are made, the schema is not only in 1NF, it is also in 3NF since there aren’t that many functional dependencies.

Relations

movies

- budget
- homepage
- id (Primary key)
- original_lang
- original_title
- overview
- popularity
- release_date
- revenue
- runtime
- status
- tagline
- title
- vote_average
- vote_count

genres

- genre_id (Primary key)
- genre_name

genre_links

- genre_id (Primary ley) (Foreign key from genres)
- movie_id (Primary key) (Foreign key from movies)

keywords

- keyword_id (Primary key)
- keyword_name

keywords_links

- keyword_id (Primary ley) (Foreign key from keywords)
- movie_id (Primary key) (Foreign key from movies)

production_companies

- production_company_id (Primary key)
- production_company_name

production_company_links

- production_company_id (Primary ley) (Foreign key from production_companies)
- movie_id (Primary key) (Foreign key from movies)

production_countries

- production_country_iso (Primary key)
- production_country_name

production_country_links

- production_country_iso (Primary ley) (Foreign key from production_countries)
- movie_id (Primary key) (Foreign key from movies)

languages

- language_iso (Primary key)
- language_name

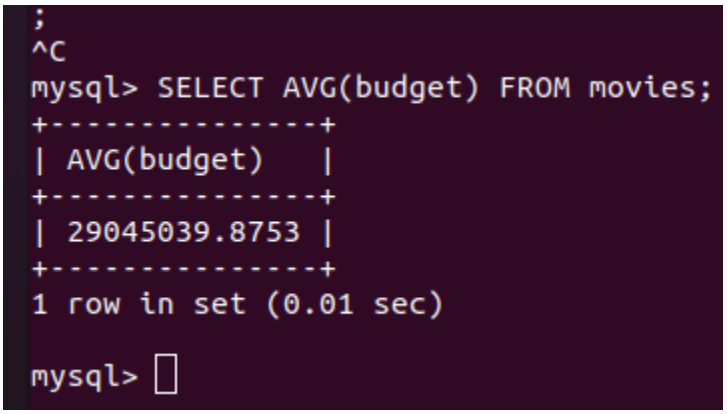
language_links

- language_id (Primary ley) (Foreign key from languages)
- movie_id (Primary key) (Foreign key from movies)

Queries

Query 1 : average budget of all movies

"SELECT AVG(budget) FROM movies"



Query 2: movies produced in US

```
SELECT title, production_company_name FROM production_companies INNER JOIN
    (SELECT title, production_company_id FROM production_company_links INNER JOIN
        (SELECT title, id FROM movies INNER JOIN
            (SELECT movie_id FROM production_country_links
            WHERE production_country_iso = 'US') AS x
        ON movies.id = x.movie_id) AS y
    ON production_company_links.movie_id = y.id) AS z
ON production_companies.production_company_id = z.production_company_id
```

```
Database changed
mysql> SELECT title, production_company_name FROM production
_links INNER JOIN (SELECT title, id FROM movies INNER JOIN (
 AS x ON movies.id = x.movie_id) AS y ON production_company_
uction_company_id LIMIT 5;
+--------------+----------------------------------------+
| title        | production_company_name                |
+--------------+----------------------------------------+
| Four Rooms   | Miramax Films                          |
| Four Rooms   | A Band Apart                           |
| Star Wars    | Lucasfilm                              |
| Star Wars    | Twentieth Century Fox Film Corporation |
| Finding Nemo | Pixar Animation Studios                |
+--------------+----------------------------------------+
5 rows in set (0.00 sec)

mysql>
```

Query 3: 5 movies that made most revenue

SELECT title, revenue FROM movies ORDER BY revenue DESC LIMIT 5

```
+----------------+------------+
| title          | revenue    |
+----------------+------------+
| Avatar         | 2787965087 |
| Titanic        | 1845034188 |
| The Avengers   | 1519557910 |
| Jurassic World | 1513528810 |
| Furious 7      | 1506249360 |
+----------------+------------+
5 rows in set (0.02 sec)

mysql>
```

Query 4: movies with both science fiction and mystery genres

```
SELECT id, title, genre_name FROM genres INNER JOIN
        (SELECT id, title, genre_id FROM genre_links INNER JOIN
                (SELECT id, title FROM movies INNER JOIN
                        (SELECT DISTINCT movie_id
                        FROM genre_links WHERE movie_id in
                                (SELECT movie_id FROM genre_links JOIN
                                genres WHERE genre_name='Science Fiction')
                                AND movie_id in
                                (SELECT movie_id FROM genre_links JOIN
                                genres WHERE genre_name='Mystery')
                                ) AS bothGen
                ON movies.id = bothGen.movie_id) AS withIDs
        ON genre_links.movie_id = withIDs.id) AS withGenreID
ON genres.genre_id = withGenreID.genre_id ORDER BY id
```

This returned a list of movie_ids titles and genres, then some python code was done to combine the results into one entry for each movie that contains all of the genres for that movie.

```
koga@koga-HP-15-TS-Notebo
koga@koga-HP-15-TS-Notebook-PC:~/Documents/SP21/351/A5$ python3 ./assi
Connection to MySQL DB successful

Query 4: movies with both science fiction and myster genres
(title, list of genres)
(Tomorrowland, (Adventure, Family, Mystery, Science Fiction))
(Inception, (Action, Thriller,  Science Fiction, Mystery, Adventure))
(Watchmen, (Action, Mystery, Science Fiction))
(Prometheus, (Science Fiction, Adventure, Mystery))
(Oblivion, (Action, Science Fiction, Adventure, Mystery))
koga@koga-HP-15-TS-Notebook-PC:~/Documents/SP21/351/A5$
```

Query 5: movies with popularity greater than the average

SELECT title, popularity FROM movies

WHERE popularity > (SELECT avg(popularity) FROM movies)

```
Database changed
mysql> SELECT title, popularity FROM movies WH
+-----------------+------------+
| title           | popularity |
+-----------------+------------+
| Four Rooms      |   22.87623 |
| Star Wars       | 126.393695 |
| Finding Nemo    |  85.688789 |
| Forrest Gump    | 138.133331 |
| American Beauty |  80.878605 |
+-----------------+------------+
5 rows in set (0.01 sec)

mysql>
```