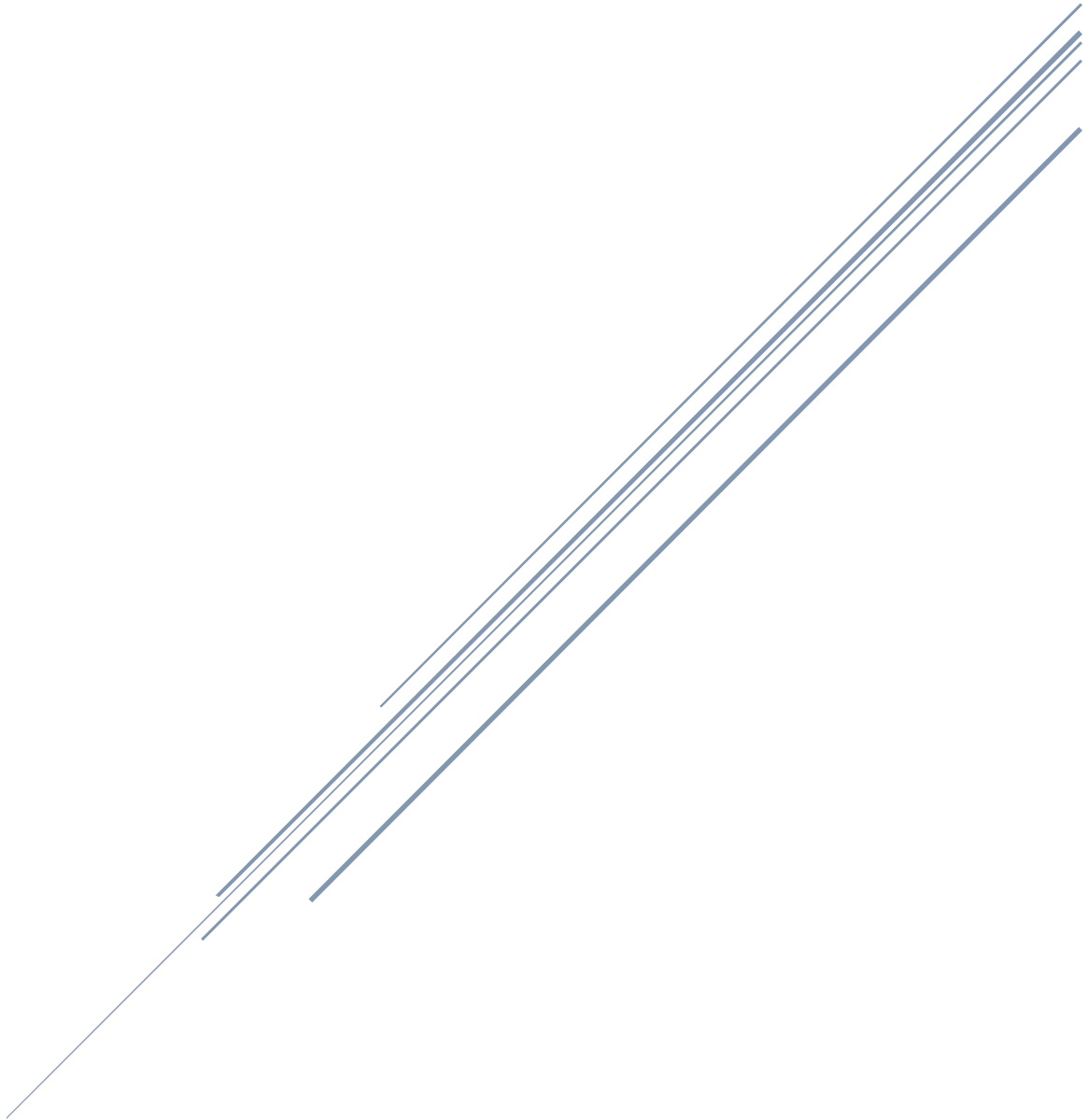# ASSIGNMENT 5

WRITE UP

Vuochlang Chang

WASHINGTON STATE UNIVERSITY OF VANCOUVER
CS 351 – INTRO TO DATABASE

Part 1: Data

List of libraries included:

- csv
- mysql.connector
- json

Description of each relation in the schema:

Relation	Attributes	Description
Movie	Id	*Primary key(s):* Id
	Budget	
	Homepage	*Foreign key(s):* None
	Original language	
	Title	This relation contains all movies attribute that is single value.
	Tagline	
	Vote average	
	Vote count	
	Status	
	Release date	Note: All single value attributes are listed to the left side.
	Run time	
	Revenue	
	Popularity	
	Overview	
	Original title	

(Table 1: Movie relation)

Relation	Attributes	Description
Genre	Id	*Primary key(s):* Id
		Foreign key(s): None
	name	This relation contains all unique genres id and their names.

(Table 2: Genre relation)

Relation	Attributes	Description
SpokenLanguage	iso_639_1	*Primary key(s):* iso_639_1
		Foreign key(s): None
	name	This relation contains all unique spoken languages id and their names.

(Table 3: SpokenLanguage relation)

Relation	Attributes	Description
ProductionCountries	iso_3166_1	*Primary key(s):* iso_3166_1
		Foreign key(s): None
	name	This relation contains all unique country id and their names.

(Table 4: ProductionCountries relation)

Relation	Attributes	Description
ProductionCompanies	id	*Primary key(s):* id
		Foreign key(s): None
	name	This relation contains all unique company id and their names.

(Table 5: ProductionCompanies relation)

Relation	Attributes	Description
Keywords	id	*Primary key(s):* id
		Foreign key(s): None
	name	This relation contains all unique keywords id and their names.

(Table 6: Keywords relation)

Relation	Attributes	Description
FromCountry	mId	*Primary key(s):* None
		Foreign key(s): mId, cId
	cId	This is a join table contains foreign keys: • mId reference to Movie.id • cId reference to ProductionCountries.id

(Table 7: FromCountry relation)

Relation	Attributes	Description
FromCompany	mId	*Primary key(s):* None
		Foreign key(s): mId, cId
	cId	This is a join table contains foreign keys: • mId reference to Movie.id • cId reference to ProductionCompanies.id

(Table 8: FromCompany relation)

Relation	Attributes	Description
HasKeyword	mId	*Primary key(s):* None
		Foreign key(s): mId, kId
	kId	This is a join table contains foreign keys: • mId reference to Movie.id • kId reference to Keywords.id

(Table 9: HasKeyword relation)

Relation	Attributes	Description
HasGenre	mId	*Primary key(s):* None
		Foreign key(s): mId, gId
	gId	This is a join table contains foreign keys: • mId reference to Movie.id • gId reference to Genre.id

(Table 10: HasGenre relation)

Relation	Attributes	Description
HasLanguages	mId	*Primary key(s):* None
		Foreign key(s): mId, lId
	lId	This is a join table contains foreign keys: • mId reference to Movie.id • lId reference to SpokenLanguage.id

(Table 11: HasLanguages relation)
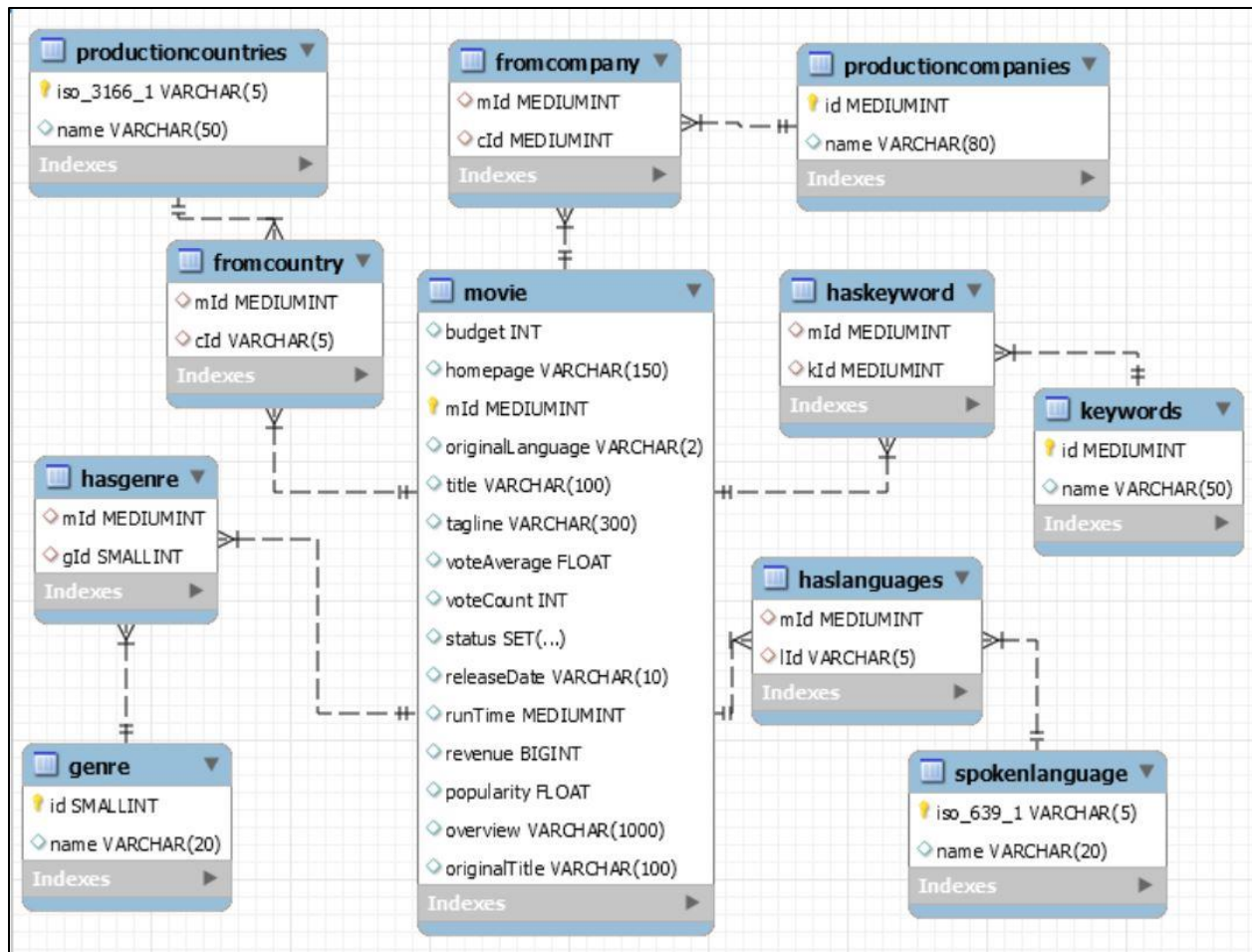
(Figure 1: Database schema)

Analysis of the schema:

To recap:

- 1NF: each attribute of a table must have single value
- 2NF: Non non-prime attribute is dependent on the proper subset of any candidate key of the table
- 3NF: each functional dependency X -> Y, X is a super key of the table, Y is a prime attribute of the table

As shown in table (1-11) and figure 1, all relations are in 1NF, 2NF and 3NF since each relations has a primary key that is also a super key of the table.

PART 2: Queries

1. 10% What is the average budget of all movies? Your output should include just the average budget value.

```
SELECT voteAverage AS 'vote average'
FROM Movie
LIMIT 5;
```

```
+------------------+
| vote average |
+------------------+
|        6.5       |
|        8.1       |
|        7.6       |
|        8.2       |
|        7.9       |
+------------------+
```

2. 10% Show the movies that were produced in the United States. Your output must include the movie title and the production company name.

```
SELECT Movie.title AS 'movie title' , ProductionCompanies.name AS 'production company name'

FROM Movie

INNER JOIN FromCompany

ON Movie.mId = FromCompany.mId

INNER JOIN ProductionCompanies

ON FromCompany.cId = ProductionCompanies.id

INNER JOIN FromCountry

ON Movie.mId = FromCountry.mId

INNER JOIN ProductionCountries

ON FromCountry.cId = ProductionCountries.iso_3166_1

WHERE iso_3166_1 = "US"

LIMIT 5;
```

```
+--------------------------+------------------------------------------------------+
| movie 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                              |
+--------------------------+------------------------------------------------------+
```

3. 10% Show the top 5 movies that made the most revenue. Your output must include the movie title and how much revenue it brought in.

```
SELECT title AS 'movie title', revenue

FROM Movie

ORDER BY revenue DESC

LIMIT 5;
```

```
+--------------------+------------------+
| movie title        | revenue          |
+--------------------+------------------+
| Avatar             | 2787965087 |
| Titanic            | 1845034188 |
| The Avengers       | 1519557910 |
| Jurassic World     | 1513528810 |
| Furious 7          | 1506249360 |
+--------------------+------------------+
```

4. 10% What movies have both the genre Science Fiction and Mystery. Your output must include the movie title and all genres associated with that movie.

```
SELECT Movie.title, GROUP_CONCAT(Genre.name)

FROM Movie

INNER JOIN HasGenre

ON Movie.mId = HasGenre.mId

INNER JOIN Genre
```

```
ON HasGenre.gId = Genre.id

WHERE Movie.mId in (

        SELECT Movie.mId

        FROM Movie

        INNER JOIN HasGenre

        ON Movie.mId = HasGenre.mId

        INNER JOIN Genre

        ON HasGenre.gId = Genre.id

        WHERE Genre.name = 'Science Fiction' or Genre.name = 'Mystery'

        GROUP BY Movie.title

        HAVING COUNT(*) = 2)

GROUP BY Movie.title

LIMIT 5;
```

```
+----------------------------------------------+--------------------------------------------------------+
| title                                        | GROUP_CONCAT(Genre.name)                               |
+----------------------------------------------+--------------------------------------------------------+
| 2001: A Space Odyssey                        | Science Fiction,Mystery,Adventure                      |
| Atlas Shrugged Part II                       | Drama,Science Fiction,Mystery                          |
| Atlas Shrugged Part III: Who is John Galt?   | Drama,Science Fiction,Mystery                          |
| Beneath the Planet of the Apes               | Adventure,Science Fiction,Mystery                      |
| Blindness                                    | Drama,Mystery,Science Fiction,Thriller                 |
+----------------------------------------------+--------------------------------------------------------+
```

5. 10% Find the movies that have a popularity greater than the average popularity. Your output must include the movie title and their popularity.

```
SELECT title, popularity

FROM Movie

WHERE popularity > (SELECT AVG(popularity) FROM Movie)

LIMIT 5;
```

```
+--------------------------+---------------+
| title                    | popularity |
+--------------------------+---------------+
| Four Rooms               |    22.8762 |
| Star Wars                |    126.394 |
| Finding Nemo             |    85.6888 |
| Forrest Gump             |    138.133 |
| American Beauty          |    80.8786 |
+--------------------------+---------------+
```