

# Oppgave2\_3\_Yahye Abdi Ahmed

## Table of Contents

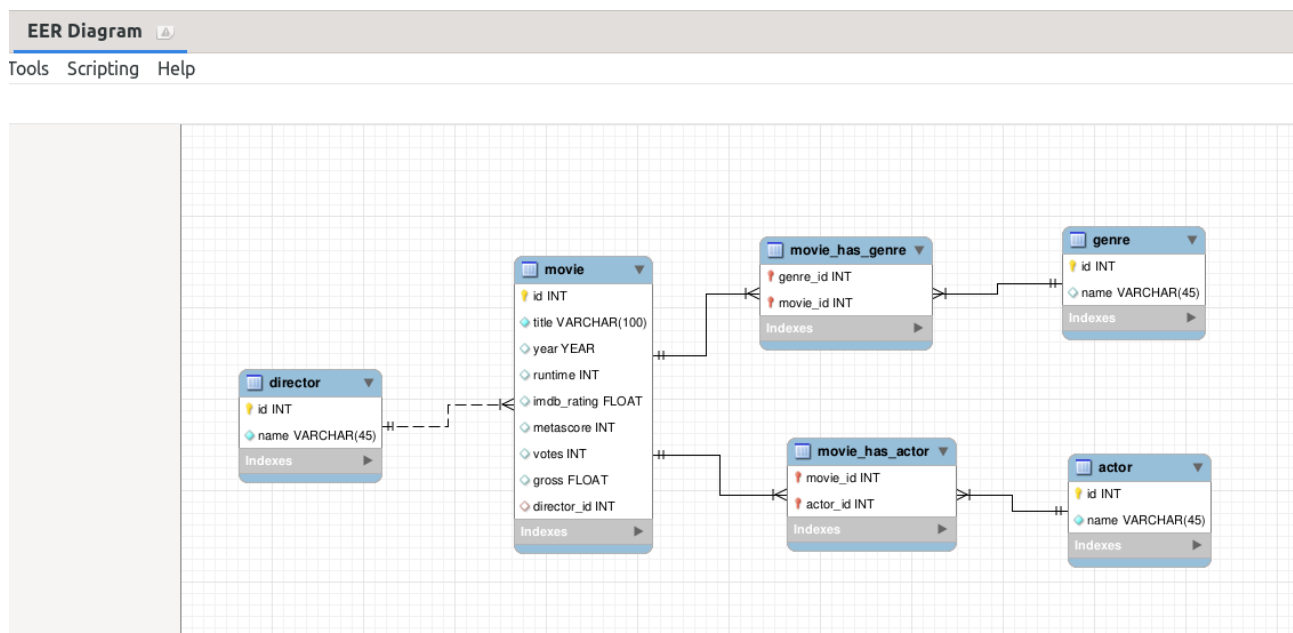
Lag tabeller.....	2
Innhold av de 6 tabellene.....	3
actor tabell.....	3
director tabell.....	3
genre tabell.....	4
movie tabell.....	4
movie_has_actor tabell.....	5
movie_has_genre tabell.....	5
Datamodell.....	7
Tabeller:.....	7
Primærnøkler:.....	7
Fremmednøkler:.....	7
Hvor tabellene er koblet sammen.....	7
Rekkefølge tabellene blir fylt med informasjon.....	8

## Lag tabeller

Etter å kjørt alle sql-spørringene i Movie script filen, har vi fått 6 tabeller. Databasen heter movies\_imdb

```
6
7
8  -- Schema movies_imdb
9
10
11
12  -- Schema movies_imdb
13
14 • CREATE SCHEMA IF NOT EXISTS `movies_imdb` DEFAULT CHARACTER SET utf8 ;
15 • USE `movies_imdb` ;
16
17
18  -- Table `movies_imdb`.`director`
```

Sånn ser tabell forholdene ut i datamodell.



## Innhold av de 6 tabellene

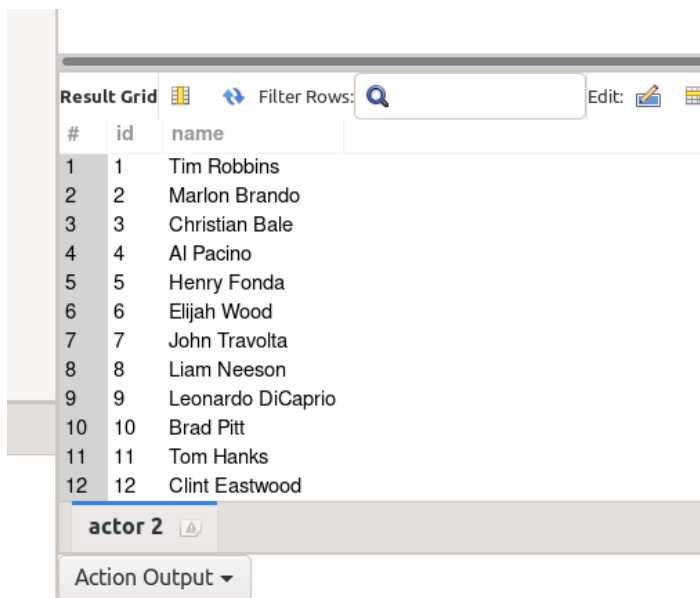
Her noen skjermbilder av innholdene i tabellene med SELECT \* spørring.

### actor tabell

```
#actor tabell
```

```
SELECT *  
FROM actor;
```

actor tabellen har 2 kolonner. Id er primary key og name kolonnen er navnet på skuespiller.



#	id	name
1	1	Tim Robbins
2	2	Marlon Brando
3	3	Christian Bale
4	4	Al Pacino
5	5	Henry Fonda
6	6	Elijah Wood
7	7	John Travolta
8	8	Liam Neeson
9	9	Leonardo DiCaprio
10	10	Brad Pitt
11	11	Tom Hanks
12	12	Clint Eastwood

actor 2

Action Output ▾

### director tabell

```
#director tabell
```

```
SELECT *  
FROM director;
```

director tabell inneholder også 2 kolonner der id er primary key og name kolonne er navnet på film direktor.

13 FROM movie\_has\_actor;

#	id	name
1	1	Frank Darabont
2	2	Francis Ford Coppola
3	3	Christopher Nolan
4	4	Sidney Lumet
5	5	Peter Jackson
6	6	Quentin Tarantino
7	7	Steven Spielberg
8	8	David Fincher
9	9	Robert Zemeckis
10	10	Sergio Leone
11	11	Lilly Wachowski
12	12	Martin Scorsese

director 3

Action Output

#	Time	Action	Message
2	17:54:49	USE movies_imdb	0 row(s)
3	17:54:51	SELECT * FROM movie_has_actor LIMIT 0, 10000	4000 row(s)
4	17:55:32	SELECT * FROM actor LIMIT 0, 10000	2961 row(s)
5	18:01:15	SELECT * FROM director LIMIT 0, 10000	559 row(s)

genre tabell

```
SELECT *  
FROM genre;
```

genre tabell inneholder også 2 kolonner. Id er primary key.

#	id	name
1	1	Drama
2	2	Crime
3	3	Action
4	4	Biography
5	5	Drama
6	6	Western
7	7	Comedy
8	8	Adventure
9	9	Animation
10	10	Horror
11	11	Mystery
12	12	Comedy

genre 4

Action Output

#	Time	Action	Message
3	17:54:51	SELECT * FROM movie_has_actor LIMIT 0, 10000	4000 row(s)
4	17:55:32	SELECT * FROM actor LIMIT 0, 10000	2961 row(s)
5	18:01:15	SELECT * FROM director LIMIT 0, 10000	559 row(s)
6	18:04:59	SELECT * FROM genre LIMIT 0, 10000	54 row(s)

## movie tabell

```
# movie tabell  
SELECT *  
FROM movie;
```

movie tabell er hoved tabellen i movie\_imdb databasen. Movie tabellen har 10 kolonner. Id er primary key. director\_id er foreign key som referer eller kobler til director tabellen sin primary key. Resten av kolonnene beskriver filmen mer med title, year, runtime, imdb, arting, metacore, votes and gross.

#	id	title	year	runtime	imdb_rating	metascore	votes	gross	director_id
1	1	The Shawshank Redemption	1994	142	9.3	80	2377480	28.34	1
2	2	The Godfather	1972	175	9.2	100	1646818	134.97	2
3	3	The Dark Knight	2008	152	9	84	2341261	534.86	3
4	4	The Godfather: Part II	1974	202	9	90	1146666	57.3	2
5	5	12 Angry Men	1957	96	9	96	701882	4.36	4
6	6	The Lord of the Rings: The Return ...	2003	201	8.9	94	1663902	377.85	5
7	7	Pulp Fiction	1994	154	8.9	94	1852195	107.93	6
8	8	Schindler's List	1993	195	8.9	94	1229549	96.9	7
9	9	Inception	2010	148	8.8	74	2099854	292.58	3
10	10	Fight Club	1999	139	8.8	66	1881336	37.03	8
11	11	The Lord of the Rings: The Fellows...	2001	178	8.8	92	1683989	315.54	5
12	12	Forrest Gump	1994	142	8.8	82	1838908	330.25	9

#	Time	Action	Message	Duration
4	17:55:32	SELECT * FROM actor LIMIT 0, 10000	2961 row(s) returned	0,00
5	18:01:15	SELECT * FROM director LIMIT 0, 10000	559 row(s) returned	0,00
6	18:04:59	SELECT * FROM genre LIMIT 0, 10000	54 row(s) returned	0,00
7	18:07:48	SELECT * FROM movie LIMIT 0, 10000	1000 row(s) returned	0,00

movie\_has

## \_actor tabell

```
# movie_has_actor tabell
SELECT *
FROM movie_has_actor;
```

Denne tabellen er spesiell. Den inneholder bare 2 kolonner og begge er foreign keys til 2 andre tabeller. Foreign til **movie** tabellen og foreign key til **actor** tabellen. Denne tabellen er mellom tabell, en tabell som kobler sammen 2 andre tabeller. Det er sånn "many to many" forhold mellom tabeller er dannet.

Her kan vi actor med id 1 er i filmene med id: 1, 510, 824 og 884. Altså kan 1 actor være assosiert med flere forskjellige filmer. En film kan også være assosiert med flere forskjellige actor

#	movie_id	actor_id
1	1	1
2	510	1
3	824	1
4	884	1
5	2	2
6	76	2
7	312	2
8	453	2
9	3	3
10	37	3
11	65	3
12	157	3

## movie\_has\_genre tabell

```
# movie_has_genre tabell
SELECT *
FROM movie_has_genre;
```

Denne er også mellom tabell. Den kobler sammen movie og actor tabell.

26 FROM movie\_has\_genre;

#	genre_id	movie_id
1	1	1
2	1	2
3	1	3
4	1	4
5	1	5
6	1	6
7	1	7
8	1	8
9	1	10
10	1	11
11	1	12
12	1	14

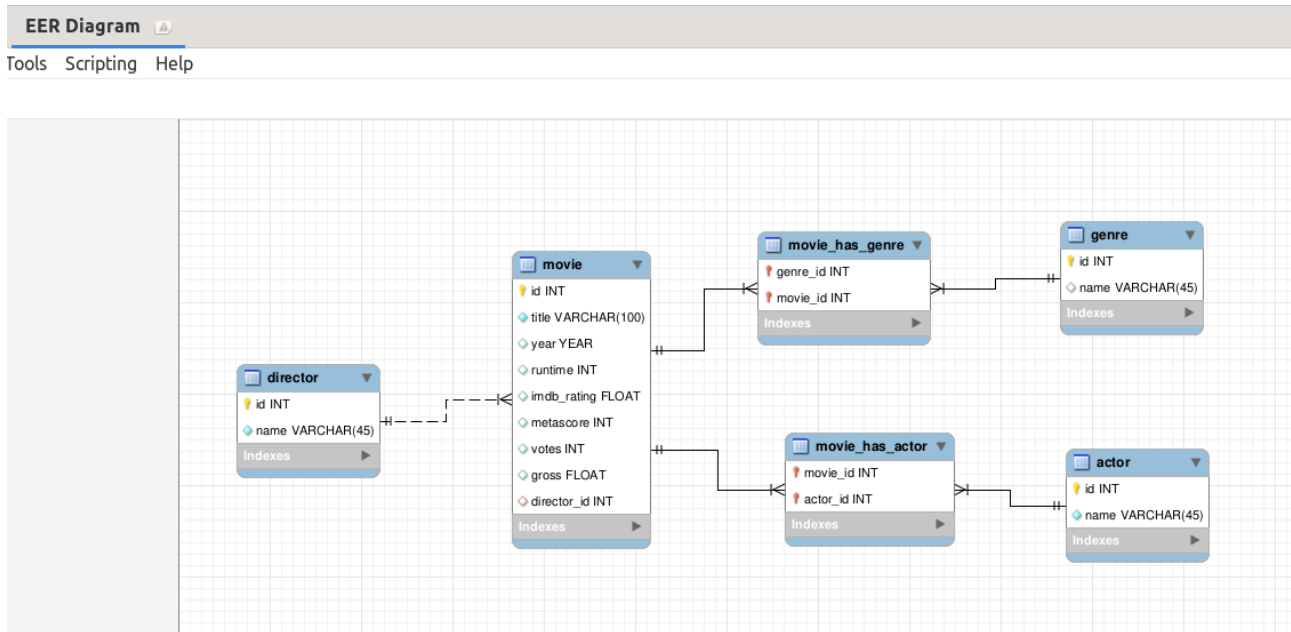
movie\_has\_genre 7

Action Output

#	Time	Action
6	18:04:59	SELECT * FROM genre LIMIT 0, 10000
7	18:07:48	SELECT * FROM movie LIMIT 0, 10000
8	18:13:28	SELECT * FROM movie_has_actor LIMIT 0, 10
9	18:36:10	SELECT * FROM movie_has_genre LIMIT 0, 10

Vi kan se at id 1 er assosiert med alle første 14 filmer fra movie tabellen. Det er mer filmer som er assosiert med genre id 1, men her ser vi bare 14 filmer.

# Datamodell



## Tabeller:

Vi har 6 tabeller der 2 av dem er mellom tabeller(**movie\_has\_actor** og **movie\_has\_genre**).

**movie** er hovedtabellen. **genre** og **actor** tabellene er begge koblet til **movie** tabellen via de 2 mellom tabellene.

## Primærnøkler:

unntatt for mellom tabellene(**movie\_has\_actor** og **movie\_has\_genre**) er primærnøkklene resten 4 tabeller id kolonne med int data type.

## Fremmednøkler:

**movie** har foreign key director\_id som referer til primary key i **director** tabellen: director.id.

Mellom tabellene inneholder kun foreign key kolonner. **movie\_has\_genre** har foreign key til movie.id og genre.id. **movie\_has\_actor** for movie.id og actor.id.

## Hvor tabellene er koblet sammen

**director** tabellen og **movie** har 1 til mange forhold. Hver director som individ er unik menneske. Det finnes bare en av denne personen. Med to strek ||, betyr det i ER diagramet at 1 og kun bare 1 av hver instans i director tabellen. Med en director kan være direktor for flere filmer.

Forhold mellom **movie** og **actor** er "many to many", flere til flere. En skuespiller kan være med i flere filmer og en film kan ha flere skuespillere. En mellom tabell må for at forholdet skal være fungere godt. Mellom tabellen inneholder referanse til fremmednøkkelen til begge tabeller. Mellom tabellen heter **movie\_has\_actor**

Forhold mellom **movie** og **genre** er også ”many to many”. Mellom tabellen heter **movie\_has\_genre**.

## **Rekkefølge tabellene blir fylt med informasjon**

Jeg fikk hjelp fra studassistenter til dette spørsmålet. Sql filen var for stor, så jeg forsto ikke rekkefølge informasjon ble fylt inn i tabellene utfra den.

Rekkefølgen blir director -> movie -> actor/genre og tilslutt de 2 mellom tabellene som ikke kan eksistere uten tabellene den er mellom.