

PROJECT : MOVIE DATA ANALYSIS

1. Project Goals

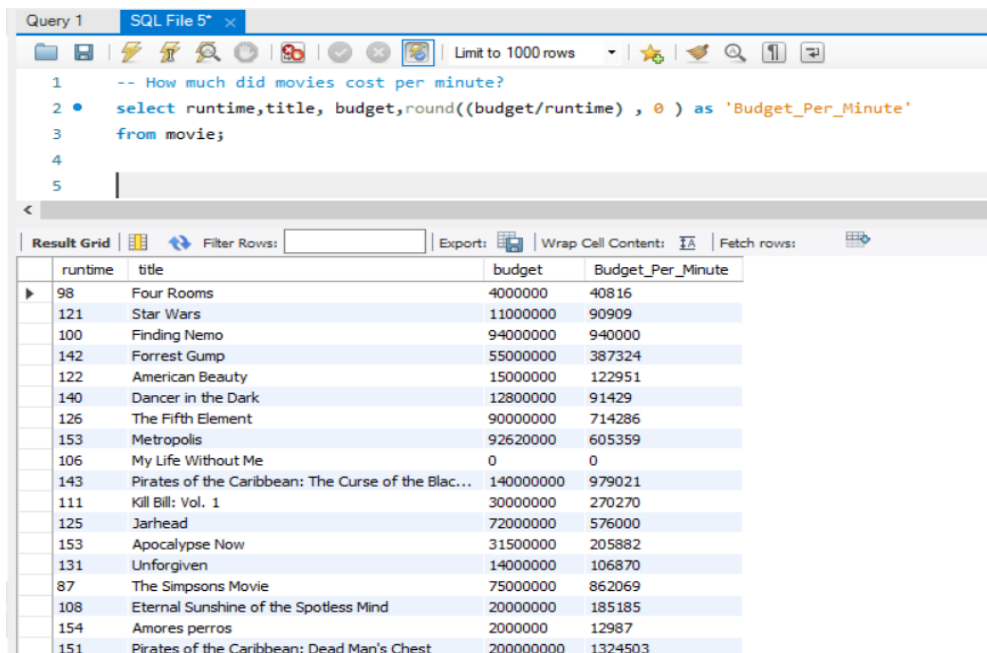
In this project, I utilized MySQL to conduct a thorough analysis of a movie database. Through SQL queries and functions, I addressed a series of pertinent business questions. This involved working with multiple tables and applying various SQL operations to extract meaningful insights from the dataset.

2. Data

In this project, I worked with a database containing 17 tables. Below, you'll find details on their attribute types and the Entity-Relationship (ER) model.

QUESTIONS

Q1: How much did movies cost per minute?



The screenshot shows a MySQL query editor window titled 'Query 1' and 'SQL File 5*'. The query is as follows:

```
1 -- How much did movies cost per minute?
2 • select runtime,title, budget,round((budget/runtime) , 0 ) as 'Budget_Per_Minute'
3   from movie;
4
5
```

Below the query editor is the 'Result Grid' showing the results of the query. The grid has four columns: 'runtime', 'title', 'budget', and 'Budget_Per_Minute'. The results are sorted by budget in descending order.

	runtime	title	budget	Budget_Per_Minute
▶	98	Four Rooms	4000000	40816
	121	Star Wars	11000000	90909
	100	Finding Nemo	94000000	940000
	142	Forrest Gump	55000000	387324
	122	American Beauty	15000000	122951
	140	Dancer in the Dark	12800000	91429
	126	The Fifth Element	90000000	714286
	153	Metropolis	92620000	605359
	106	My Life Without Me	0	0
	143	Pirates of the Caribbean: The Curse of the Blac...	140000000	979021
	111	Kill Bill: Vol. 1	30000000	270270
	125	Jarhead	72000000	576000
	153	Apocalypse Now	31500000	205882
	131	Unforgiven	14000000	106870
	87	The Simpsons Movie	75000000	862069
	108	Eternal Sunshine of the Spotless Mind	20000000	185185
	154	Amores perros	2000000	12987
	151	Pirates of the Caribbean: Dead Man's Chest	200000000	1324503

Q2: What is the top-5 movies in terms of budget?

Result Grid		
	Filter Rows:	Export: Wrap Cell
	title	budget
▶	Pirates of the Caribbean: On Stranger Tides	380000000
	Pirates of the Caribbean: At World's End	300000000
	Avengers: Age of Ultron	280000000
	Superman Returns	270000000
	Tangled	260000000





Q3: In terms of release date, how old is every single movie? show top ten youngest movies.

```

12 • select release_date, title, (year(curdate()) - year(release_date)) as Age
13 from movie
14 order by age asc
15 limit 10;

```

<

Result Grid   Filter Rows: | Export:  | Wrap Cell Content: ☒ | Fetch rows: 

	release_date	title	Age
▶	2017-02-03	Growing Up Smith	6
	2016-02-04	Pride and Prejudice and Zombies	7
	2016-07-14	Ghostbusters	7
	2016-02-19	Triple 9	7
	2016-06-22	Independence Day: Resurgence	7
	2016-08-26	Hands of Stone	7
	2016-05-25	Warcraft	7
	2016-07-07	Star Trek Beyond	7

Result 9 x

Q4: In which years did the producer make a movie?

```

17 -- In which years did the producer make a movie?
18 • select distinct year(release_date) as release_date
19 from movies;
20
21
22
23
24
25

```

Result Grid	
	Filter Rows: Export: Wrap Cell Content:
	release_date
	2008
	1986
	1954
	1991
	2014
	1973
	1940
	1972

Result 15

Q5: What is the bottom-10 popular movies, and which company made them?

```
-- What is the bottom-10 popular movies, and which company made them?

• SELECT movie.title, SUM(movie.popularity) as sum_popularity, production_company.company_name
FROM movie
JOIN movie_company ON movie.movie_id = movie_company.movie_id
JOIN production_company ON movie_company.company_id = production_company.company_id
GROUP BY movie.title, production_company.company_name
ORDER BY sum_popularity asc
LIMIT 10;
```

Q6: Which movies cost less than 50,000 and what were their genres? Just top-10

```
34 -- Which movies cost less than 50,000 and what were their genres?
35 • SELECT movie.title, movie.budget, genre.genre_name
36 FROM movie
37 JOIN movie_genres ON movie.movie_id = movie_genres.movie_id
38 JOIN genre ON movie_genres.genre_id = genre.genre_id
39 WHERE movie.budget < 50000
40 ORDER BY movie.budget DESC
41 limit 10;
42
43
44
```

Result Grid			
Filter Rows: <input type="text"/>			
Export: <input type="text"/>			
Wrap Cell Contents: <input type="text"/>			
Fetch: <input type="text"/>			
	title	budget	genre_name
▶	Osama	46000	Drama
	Down Terrace	31192	Drama
	Down Terrace	31192	Action
	Down Terrace	31192	Comedy
	Clerks	27000	Comedy
	Pink Narcissus	27000	Drama
	Pink Narcissus	27000	Romance
	Dry Spell	22000	Comedy

Q7: Which movie companies invested the budget between 150K to 200K and for which movie?

```
43
44 -- Which movie companies invested the budget between 150K to 200K and for which movie?
45
46 • SELECT movie.title, movie.budget, production_company.company_name
47 FROM movie
48 JOIN movie_company ON movie.movie_id = movie_company.movie_id
49 JOIN production_company ON movie_company.company_id = production_company.company_id
50 WHERE movie.budget BETWEEN 150000 AND 200000
51 ORDER BY budget DESC;
52
```

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

	title	budget	company_name
▶	Per un pugno di dollari	200000	Jolly Film
	Per un pugno di dollari	200000	Ocean Films
	Per un pugno di dollari	200000	Constantin Film Produktion
	Per un pugno di dollari	200000	United Artists
	Wendy and Lucy	200000	Glass Eye Pix
	Wendy and Lucy	200000	Washington Square Films
	Wendy and Lucy	200000	Film Science
	Wendy and Lucy	200000	Field Guide Films

Result 25 x

Output

Q8: Which director did make the most popular movie in English and French language?

```
54 -- which director did make the most popular movie in English and French language?
55
56 • SELECT language.language_name, person.person_name, COUNT(*) as movie_count, MAX(movie.popularity) as max_popularity
57 FROM movie
58 join movie_crew on movie.movie_id = movie_crew.movie_id
59 join person on movie_crew.person_id = person.person_id
60 join movie_languages on movie.movie_id = movie_languages.movie_id
61 join language on movie_languages.language_id = language.language_id
62 GROUP BY language.language_name, person.person_name
63 having language.language_name IN ('English','Latin')
64 limit 5;
65
```

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

	language_name	person_name	movie_count	max_popularity
▶	English	Ivan Voru00edu010dek	2	20.745052
	English	'Chema' Hernandez	4	14.970654
	English	'Cowboy' James Hollan	2	109.684788
	English	50 Cent	2	28.060472
	English	A L Katz	2	7.710797

Q9: Make a list of actors(actress) who played after the year 2010.

```
>>
56 -- Make a list of actors(actress) who played after the year 2010.
57
58 • select person.person_name, movie.title, movie_crew.job
59 from movie
60 join movie_crew on movie.movie_id = movie_crew.movie_id
61 join person on movie_crew.person_id = person.person_id
62 where movie_crew.job = "Characters" and release_date > "2010-01-01"
63 GROUP BY person.person_id, movie.title, movie_crew.job;
```

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

person_name	title	job
Stan Lee	The Amazing Spider-Man	Characters
Steve Ditko	The Amazing Spider-Man	Characters
Marlon Wayans	Scary Movie 5	Characters
Shawn Wayans	Scary Movie 5	Characters
Buddy Johnson	Scary Movie 5	Characters
Phil Beaman	Scary Movie 5	Characters
Jason Friedberg	Scary Movie 5	Characters
Aaron Seltzer	Scary Movie 5	Characters