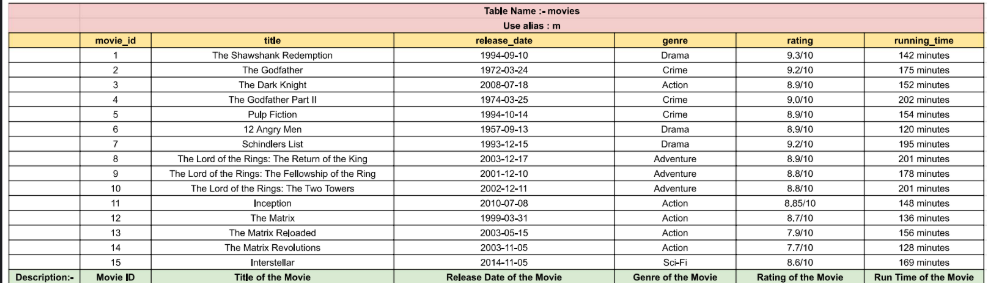
**Case Study:**  
This case study scenario involves user bookings associated with a movie theatre.  
You have been given a dataset containing multiple tables related to the case study.

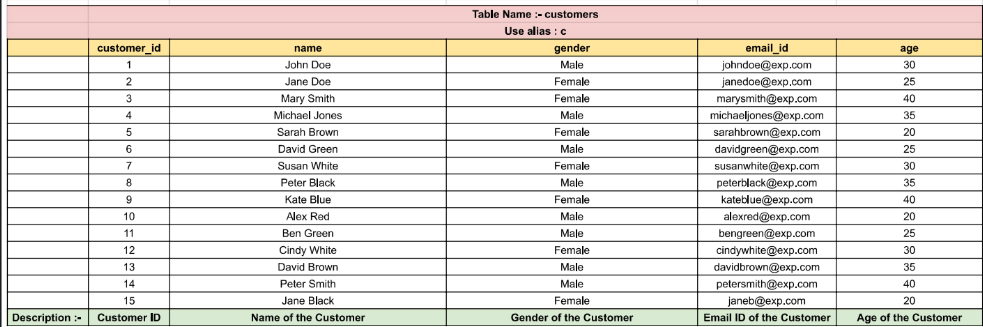
Now, let's explore some objective questions related to SQL and the given case study.

**Dataset tables**

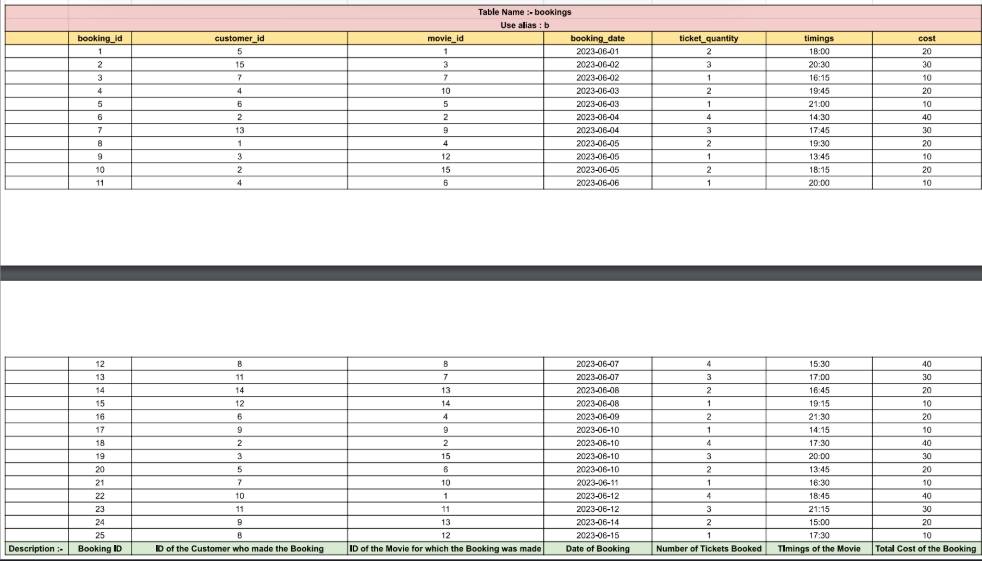
1. Movies:



1. Customers:



1. Bookings



**Level - Basic**

1. How can you calculate the average age of male customers in the "customers" table?

- SELECT AVG(age) FROM customers WHERE gender = ‘Male’;

1. How can you retrieve the top 4 movies with the highest ratings from the "movies" table?

- SELECT \* FROM movies ORDER BY rating desc LIMIT 4;

1. How can you retrieve the movie titles and ratings of movies with a rating greater than 8.5?

- SELECT title, ratings FROM movies WHERE rating > 8.5 ;

1. How can you retrieve the highest rating for each genre?

- SELECT genre, max(rating) FROM movies GROUP BY genre

1. Which SQL statement is used to calculate the total number of bookings for each movie genre?

- SELECT movies.genre, bookings.count(\*)

FROM movies JOIN bookings ON movies.movie\_id = bookings.booking.id

GROUP BY movies.genre

1. Which SQL statement is used to retrieve the customers who made more than 2 bookings?

- SELECT c.name

FROM customers c

JOIN bookings b ON c.customer\_id = b.customer\_id

GROUP BY c.customer\_id

HAVING COUNT(\*) > 2;

1. How can you calculate the percentage of male customers in the "customers" table?

- WITH (SELECT count(\*) FROM customers) as total

SELECT count(SELECT (count(\*) FROM customers WHERE Gender = “Male”) / total) FROM customers

**SELECT AVG(CASE WHEN gender = 'Male' THEN 1 ELSE 0 END) \* 100 FROM customers;**

1. How can you retrieve the customer who has made the most bookings?

- SELECT c.name, COUNT(\*) AS booking\_count FROM customers c JOIN bookings b ON c.customer\_id = b.customer\_id GROUP BY c.customer\_id ORDER BY booking\_count DESC LIMIT 1;

1. Which SQL query can be used to list the names and ages of customers who have booked tickets for the movie with movie\_id = 5?

- SELECT c.name, c.age FROM customers c JOIN bookings b ON c.customer\_id = b.customer\_id WHERE b.movie\_id = 5

1. **Rearrange code-1**

Given the Customers, Movies and Bookings tables, write an SQL query to retrieve the count of bookings for each genre.

***Think critically and rearrange the subqueries to unveil the SQL solution!***

SELECT count(\*) FROM bookings b JOIN movies m ON b.movie\_id = m.movie\_id

GROUP BY m.genre

**SELECT m.genre, COUNT(b.booking\_id) AS booking\_count**

**FROM movies m**

**LEFT JOIN bookings b**

**ON m.movie\_id = b.movie\_id**

**GROUP BY m.genre;**

1. List the movie titles and their corresponding average ticket quantities.

SELECT m.title, (SELECT avg(ticket\_quantity) FROM bookings GROUP BY movie\_id) as avg

FROM movies JOIN bookings b ON m.movie\_id = b.movie\_id

**SELECT m.title, AVG(ticket\_quantity) AS average\_ticket\_quantity**

**FROM movies m**

**JOIN bookings b**

**ON m.movie\_id = b.movie\_id**

**GROUP BY m.title;**

1. Retrieve the customer names and their email IDs for bookings made on the date '5th June 2023'.

SELECT c.name, c.email\_id FROM customer c JOIN bookings b ON c.customer\_id = b.customer\_id WHERE b.booking\_date = ‘2023-06-05’

1. Get the details of the movies with the highest rating.

SELECT \* FROM movies WHERE rating = (SELECT max(rating) FROM movies);

1. Retrieve the customer ID, name, and assign an age group based on the customer's age using the CASE statement.

SELECT customer\_id, name

CASE WHEN age >= 20 AND age <= 30 then ‘small’

WHEN age >= 30 AND age <= 40 then ‘medium’

WHEN age >= 40 AND age <= 50 then ‘large’

END

AS age\_group

FROM customers;

**Level - Intermediate**

**Query-1**

As a data analyst at a movie theater, you have been tasked with analyzing the booking trends and identifying listings that have been successfully booked.  
Your analysis will help the company understand the popularity of listings and make informed decisions for improving user experiences and maximizing revenue.

**Task**

* What is the total number of customers in the database as "total\_customers"?

**Expected Output Format**

**total\_customers**

SELECT count(\*) total\_customers FROM customers

**Task**

* Find the average rating of movies having running\_time greater than 178 as average\_rating.

**Expected Output Format**

**average\_rating**

SELECT avg(rating) as average\_rating FROM movies

WHERE running\_time > 178 ;

**Task**

* Retrieve the movie with the longest running time.

**Expected Output Format**

**movie\_id**  
**title**  
**release\_date**  
**genre**  
**rating**  
**running\_time**

SELECT \* FROM movies WHERE running\_time = (SELECT max(running\_time) FROM movies);

**Query-4**

**Task**

* Calculate the total revenue generated by the cinema hall based on the cost of bookings between 4th June 2023 to 12th June 2023 as total\_revenue.

**Expected Output Format**

**total\_revenue**

SELECT sum(cost) as total\_revenue FROM bookings

WHERE booking\_date >= '2023-06-04' AND booking\_date <= '2023-06-12';

**Query-5**

**Task**

* Calculate the total duration (in minutes) of all movies combined that are of the genre Drama as total\_duration.

**Expected Output Format**

**total\_duration**

SELECT sum(running\_time) as total\_duration FROM movies WHERE genre = "Drama";

**Query-6**

**Task**

* How many bookings were made on each booking date?

**Expected Output Format**

**booking\_date** **COUNT(\*)**

SELECT booking\_date, count(\*) FROM bookings GROUP BY booking\_date;

**Query-7**

**Task**

* Calculate the total cost of all bookings made by each customer as total\_cost and display the customer's name along with the total cost. Group the results by customer\_id.

**Expected Output Format**

**name**  
**total\_cost**

SELECT c.name, sum(cost) as total\_cost FROM customers c JOIN bookings b

ON c.customer\_id = b.customer\_id GROUP BY c.customer\_id;

**Query-8**

**Task**

* Calculate the total number of bookings as total\_bookings made for each movie.

**Expected Output Format**

**title**  
**total\_bookings**

SELECT m.title, (SELECT count(\*) FROM bookings WHERE movie\_id = m.movie\_id) as total\_bookings

FROM movies m GROUP BY m.movie\_id;

**Query-9**

**Task**

* Find the customer id and name of the customer who made the first booking in the database.

**Expected Output Format**

**customer\_id**  
**name**

SELECT c.customer\_id, c.name FROM customers c JOIN bookings b

ON c.customer\_id = b.customer\_id

WHERE b.booking\_date = (SELECT min(booking\_date) FROM bookings) ;

**Query-10**

**Task**

* Retrieve the customer names and the number of bookings made by each customer as booking\_count, sorted in descending order of the booking count.

**Expected Output Format**

**name**  
**booking\_count**

SELECT c.name, (SELECT count(\*) as booking\_count FROM bookings WHERE customer\_id = c.customer\_id) booking\_count

FROM customers c

ORDER BY booking\_count desc;

**Query-11**

**Task**

* Retrieve the total no of seats booked for a time later than 20:00:00 as total\_seats\_booked.

**Expected Output Format**

**total\_seats\_booked**

SELECT sum(ticket\_quantity) as total\_seats\_booked

FROM bookings WHERE timings > '20:00:00';

**Query-12**

**Task**

* Retrieve the details of the customer who has bought the most number of tickets.

**Expected Output Format**

**customer\_id**  
**name**  
**gender**  
**email\_id**  
**age**

SELECT \* FROM customers c WHERE c.customer\_id IN

(SELECT customer\_id FROM bookings

GROUP BY customer\_id ORDER BY count(ticket\_quantity) desc LIMIT 1) ;