Made by : Raushan kumar , IIEST shibpur(2019-2023)

Sql queries

Create the following tables:-

Table: cust

Attributes	Data Type	Size	Condition
Cust_id	Varchar2	3	
Lname	Varchar2	15	
Fname	Varchar2	15	
Area	Varchar2	2	
Phone_no	number	8	

Table: movie

Attributes	Data Type	Size	Condition
Mv_no	number	2	
Title	Varchar2	25	
Type	Varchar2	10	
Star	Varchar2	25	
Price	number	8,2	

Table: invoice

Attributes	Data Type	Size	Condition
Inv_no	Varchar2	3	
Mv_no	number	2	
Cust_id	Varchar2	3	
Issue_date	date		
Return_date	date		

→ FIRST YOU NEED TO ENTER (6+10+10) DATA IN YOUR LOCAL MACHINE BY USING BELOW TABLE;

```
mysql> select * from cust;
+---------+------------+---------+------+----------+
| cust_id | Lname      | Fname   | area | phone_no |
+---------+------------+---------+------+----------+
| a01     | Bayross    | Ivan    | sa   |      234 |
| a02     | Saitwal    | Vandana | mu   |  5560379 |
| a03     | Jaguste    | Pramada | da   |   466389 |
| a04     | Navindgi   | Basu    | ba   |  6125401 |
| a05     | Sreedharan | Ravi    | va   |     NULL |
| a06     |            | Rukmini | gh   |  5125274 |
+---------+------------+---------+------+----------+
6 rows in set (0.03 sec)
mysql> select * from movie;
+-------+--------------------+----------+----------------+--------+
| Mv_no | title              | type     | star           | price  |
+-------+--------------------+----------+----------------+--------+
|     1 | bloody vengeance   | action   | jackie chan    | 180.95 |
|     2 | the firm           | thriller | tom cruise     | 200.00 |
|     3 | pretty woman       | romance  | richard gere   | 200.00 |
|     4 | home alone         | comedy   | macaulay culkin | 150.00 |
|     5 | the fugitive       | thriller | harrison ford  | 200.00 |
|     6 | coma               | suspense | michael douglas | 100.00 |
|     7 | dracula            | horror   | gary oldman    | 150.25 |
|     8 | quick change       | comedy   | bill murray    | 100.00 |
|     9 | gone with the wind | drama    | clarke gable   | 200.00 |
|    10 | carry on doctor    | comedy   | leslie phillips | 100.00 |
+-------+--------------------+----------+----------------+--------+
10 rows in set (0.01 sec)
mysql> select  * from invoice;
+--------+-------+---------+------------+-------------+
| inv_no | mv_no | cust_id | Issue_date | Return_date |
+--------+-------+---------+------------+-------------+
| i01    |     4 | a01     | 0000-00-00 | 2003-07-25  |
| i02    |     3 | a02     | 2003-08-12 | 2003-08-16  |
| i03    |     1 | a02     | 2003-08-15 | 2003-08-18  |
| i04    |     6 | a03     | 2003-07-10 | 2003-07-12  |
| i05    |     7 | a04     | 2003-08-05 | 2003-08-08  |
| i06    |     2 | a06     | 2003-09-18 | 2003-09-21  |
| i07    |     9 | a05     | 2003-07-07 | 2003-07-10  |
| i08    |     9 | a01     | 0000-00-00 | 2003-08-14  |
| i09    |     5 | a03     | 2003-07-06 | 2003-07-07  |
```

| i10 | 8 | a06 | 2003-09-03 | 2003-09-06 |
+--------+-------+---------+------------+-------------+
10 rows in set (0.02 sec)

Single Table Retrieval

1. Find out the names of all the customers.
   $select\ concat(Fname, ' ', Lname)\ as\ Name\ from\ cust;$

2. List the various movie types available from the movie table.
   $select\ distinct\ (type)\ as\ Movie\_type\ from\ movie;$

3. Print the list of all employees whose phone numbers are greater than the value 466398.
   $select\ concat(Fname, ' ', Lname)\ as\ Nmae\_Emp\ from\ cust\ where\ phone\_no\ >\ 466398;$

4. Find movies of the type 'action' or 'comedy'.
   $select\ title\ as\ Move\_Name\ from\ movie\ where\ type\ in\ ('action'\ ,\ 'comedy');$

5. Find the movies whose price is greater than 150 and less than or equal to 200.
   $select\ title\ as\ Movie\_Name\ from\ movie\ where\ price\ >\ 150\ and\ price\ <=\ 200;$

6. Find the names of all customers having 'a' as the second letter in their names.
   $select\ concat(Fname, ' ', Lname)\ from\ cust\ where\ Fname\ LIKE\ '\_a\%';$

7. Find the Lname of all customers that begins with 's' or 'j'.
   $select\ lname\ from\ cust\ WHERE\ lname\ LIKE\ 's\%'\ or\ lname\ like\ 'j\%';$

8. Find out the customers who stay in an area whose second letter is 'a'.
   $SELECT\ CONCAT(Fname, ' ', Lname)\ AS\ Name\_Emp\ FROM\ cust\ WHERE\ area\ like\ '\_a\%';$

9. Find the list of all customers who stay in area 'da' or area 'mu' or area 'gh'.
   $SELECT\ Fname\ FROM\ cust\ WHERE\ area\ IN\ ('da'\ ,\ 'mu',\ 'gh');$

10. List the mv_no , title, type of movies whose stars begin with the letter 'm'.
    $SELECT\ mv\_no\ ,\ title, type\ FROM\ movie\ WHERE\ star\ LIKE\ ('m\%');$

11. Find the movies that cost more than 150 and also find the new cost as original cost * 15.
    $SELECT\ title\ FROM\ movie\ WHERE\ price\ >\ 150;$
    And
    $SELECT\ title\ ,\ (price\ *\ 15)\ AS\ New\_Price\ from\ movie\ ;$

12. List the movies in the stored order of their titles.

$SELECT\ title\ FROM\ movie\ ORDER\ BY\ title$;

13. Print the names and types of all the movies except horror movies.
    $SELECT\ title\ AS\ Name\ ,\ type\ FROM\ movie\ WHERE\ type\ <>\ 'horror'$;

14. List the names, areas, and cust_id of customers without phone numbers.
    $SELECT\ CONCAT(Fname,' ',Lname)\ ,\ area\ FROM\ cust\ WHERE\ phone\_no\ is\ NULL$;

15. List the names of customers without lname.
    $SELECT\ Fname\ FROM\ cust\ WHERE\ Lname\ IS\ NULL$;

16. Print the information from the invoice table of customers who have been issued movies in the month of September.
    $SELECT\ *\ FROM\ invoice\ WHERE\ MONTH(issue\_date)\ =\ 9$;

Set function and Concatenation

17. Count the total no. of customers.
    $SELECT\ COUNT(cust\_id)\ AS\ Total\_Cust\ from\ cust$;

18. Calculate the total price of all movies.
    $SELECT\ SUM(price)\ AS\ Total\_Price\ FROM\ movie$;

19. Calculate the average price of all movies.
    $SELECT\ AVG(price)\ AS\ Total\_Avg\ FROM\ movie$;

20. Calculate the maximum and minimum movie prices. Rename the title as max_price and min_price respectively .
    $SELECT\ MAX(price)\ AS\ max\_price\ ,\ MIN(price)\ AS\ min\_price\ FROM\ movie$;

21. Count the number of movies having price greater than or equal to 150.
    $SELECT\ COUNT(Mv\_no)\ AS\ Total\_Movie\ FROM\ movie\ WHERE\ price\ >=\ 150$;

22. Print the information of the invoice table in the following format for all records.
    A) The Invoice no. of Customer Id {cust_id} is {inv_no} and Movie no. is {mv_no}.
    B) {cust_id} has taken Movie no. {mv_no} on {issue_date} and will return on {return_date}

→ *SELECT CONCAT('The Invoice no of the customer id ',cust_id,' is ',inv_no, ' and Movie no ',mv_no) AS detail from invoice ;*

→ *SELECT CONCAT(cust_id ,' has taken Movie no. ' ,mv_no, ' no ' , issue_date, ' and will return on ',return_date) AS detail FROM invoice;*

Having and Group By:-

24. Print the type and average price of each movie.
$SELECT\ Type\ , title,\ COUNT(type), SUM(price),\ AVG(price)\ FROM\ movie\ GROUP\ BY\ type$;

25. Find the number of movies in each type.
$SELECT\ type, COUNT(type)\ FROM\ movie\ GROUP\ BY\ type$;

26. Count separately the number of movies in the ‘comedy’ and ‘thriller’ types.

$SELECT\ type\ , COUNT(type)\ FROM\ movie\ GROUP\ BY\ type\ HAVING\ type\ =\ 'comedy'\ or\ type\ =\ 'thriller'$;

Or

$SELECT\ type\ , COUNT(type)\ FROM\ movie\ GROUP\ BY\ type\ HAVING\ type\ IN\ ('thriller', 'comedy')$;

27. Calculate the average price for each type that has a maximum price of Rs. 150.

SELECT type , COUNT(type) AS Total_Count, AVG(price) AS Avg_Price FROM movie GROUP BY type HAVING max(price)=150;

28. Calculate the average price of all movies where type is ‘comedy’ or ‘thriller’ and price is greater than or equal to Rs. 150.

→ SELECT type ,AVG(price) AS Average_Price from movie WHERE price>=150 GROUP BY type HAVING type IN ('comedy','thriller');

Joins and Correlations:-

29. Find out the movie number which has been issued to ‘Ivan’.
    1. Nested Queries (bottom up)

$SELECT\ mv\_no\ FROM\ invoice\ WHERE\ cust\_id\ =\ (SELECT\ cust\_id\ FROM\ cust\ WHERE\ Fname\ =\ 'Ivan')$;

Or

$SELECT\ mv\_no\ FROM\ invoice\ WHERE\ cust\_id\ IN\ (SELECT\ cust\_id\ FROM\ cust\ WHERE\ Fname\ =\ 'Ivan')$;

Or

    2. Correlated Subquery (top down)

SELECT mv_no FROM invoice WHERE EXISTS (SELECT cust_id FROM cust WHERE cust.Fname='Ivan' AND invoice.cust_id=cust.cust_id);

3. Joins (cross product + some conditions)

$SELECT\ mv\_no\ \ FROM\ invoice\ AS\ inv\ ,\ cust\ AS\ cus\ WHERE\ cus.Fname\ =\ 'Ivan'\ and\ inv.cust\_id\ =\ cus.cust\_id;$

30. Find the names and movie numbers of all the customers who have been issued a movie .

$SELECT\ \ Fname\ ,\ mv\_no\ FROM\ invoice\ JOIN\ cust\ USING(cust\_id)\ WHERE\ mv\_no\ IS\ NOT\ NULL;$

31. Select the title ,cust_id, mv_no for all the movies that are issued.

$SELECT\ title, cust\_id,\ mv\_no\ FROM\ invoice\ JOIN\ cust\ USING(cust\_id)\ JOIN\ movie\ \ USING(mv\_no);$

32. Find out the title and types of the movies that have been issued to ‘Vandana’ .

SELECT title , type FROM invoice JOIN cust USING(cust_id) JOIN movie USING(mv_no) WHERE fname='Vandana';

Or

SELECT title , type FROM invoice JOIN movie USING(Mv_no) JOIN cust USING(cust_id) WHERE fname='Vandana';

33. Find the names of customers who have been issued movies of type ‘drama’.

SELECT CONCAT(Fname ,' ' ,Lname) AS Name FROM invoice JOIN movie USING(Mv_no) JOIN cust USING(cust_id) WHERE type='drama';

34. Display the title, lname, fname for customers having a movie number greater than or equal to three , in the following format:
‘The movie taken by {fname} {lname} is {title}.

SELECT CONCAT('THE movie taken by ',Fname,' ' , Lname,' is ', title,' . ') as Required_Format from invoice JOIN cust USING(cust_id) JOIN movie USING(mv_no) WHERE mv_no>=3;

Nested Queries

35. Find out which customers have been issued movie number 9.

$SELECT\ Fname\ FROM\ cust\ WHERE\ cust\_id\ IN\ (\ SELECT\ cust\_id\ FROM\ invoice\ WHERE\ mv\_no\ =\ 9);$

36. Find the customer name and area with invoice number i10’.

$SELECT\ Fname\ FROM\ cust\ WHERE\ cust\_id\ in\ (SELECT\ cust\_id\ FROM\ invoice\ WHERE\ inv\_no\ =\ 'i10');$

37. Find the name of the movie issued to ‘Vandana’ or ‘Ivan’

SELECT title FROM movie WHERE Mv_no IN (SELECT mv_no FROM invoice WHERE cust_id in (SELECT cust_id FROM cust WHERE Fname in ('vandana','Ivan')));

38.Find the type and movie number of the movie issued to cust_id ‘a01’ or ‘a02’.

SELECT type, mv_no FROM movie WHERE mv_no in (SELECT mv_no FROM invoice WHERE cust_id in ('a01','a02'));

39. Find out if the movie starring ‘tom cruise’ is issued to any customer and print the cust_id to whom it is issued.

$SELECT\ cust\_id\ FROM\ invoice\ WHERE\ mv\_no\ IN\ (SELECT\ mv\_no\ FROM\ movie\ WHERE\ star\ =\ 'tom\ cruise'$);

40. Find the customer names and phone numbers who have been issued movies before the month of August.

SELECT Fname ,phone_no FROM cust WHERE Phone_no is NOT NULL and cust_id IN (SELECT cust_id FROM invoice WHERE MONTH(issue_date)<8);

41. List the movie number, movie issued to all customers.

$SELECT\ mv\_no\ ,\ title\ FROM\ movie\ WHERE\ mv\_no\ in\ (\ SELECT\ mv\_no\ FROM\ invoice)$;

Queries Using Date:-

42. Display the invoice number and day on which customers were issued movies

$SELECT\ inv\_no\ AS\ invoice\_number\ ,DAY(issue\_date)\ AS\ Issue\_Day\ FROM\ invoice$;

Or

$SELECT\ inv\_no\ AS\ invoice\_number\ ,DAYNAME(issue\_date)\ AS\ Issue\_Day\ FROM\ invoice$;

43. Display the month (in alphabets) in which customers are supposed to return the movies.

$SELECT\ MONTHNAME(return\_date)\ AS\ Month\_Name\ FROM\ invoice$;

44. Display the issue_date in the format “dd-month-yy” e.g. 12-February- 93

$SELECT\ date\_format(issue\_date,"\%d\ -\ \%M\ -\ \%Y")\ as\ new\_format\ from\ invoice$;

Yaha pe d to D or M to m karne se difference aa jayega

45. Find the date, 15 days after the current date.

$SELECT\ DATE\_ADD(CURDATE(),INTERVAL\ 15\ DAY)\ AS\ "DATE\ +\ 15\ days"$;

46. Find the number of days elapsed between the current date and the return date of the movie for all customers.

$SELECT\ DATEDIFF(CURDATE(),\ return\_date)\ AS\ Date\_Diff\ from\ invoice;$

Table Updations:-

47. Change the telephone number of ‘Pramada’ to 466389.
$UPDATE\ cust\ SET\ Phone\_no\ =\ 466389\ WHERE\ Fname\ =\ 'Pramada';$

48. Change the issue_date of cust_id ‘a01’ to 24/07/93
$UPDATE\ invoice\ set\ Issue\_date\ =\ 24/07/93\ WHERE\ cust\_id\ =\ 'a01';$

49.Delete the record with invoice number ‘i08’ from the invoice table.
$DELETE\ FROM\ invoice\ WHERE\ Inv\_no\ =\ 'i08';$

50. Delete all the records having a return date before 10th July 93.
$DELETE\ FROM\ invoice\ WHERE\ return\_date\ <\ '1993\ -\ 07\ -\ 10';$