DBMS LAB ASSIGNMENT

Assignment SQL – I

1. Create the following Tables -

1.1 Customer master table: **Cust**

Column Name	Format	Remarks
cust_id	varchar2(3)	Primary key, not-null
lname	varchar2(15)	
fname	varchar2(15)	
area	varchar2(2)	
phone_no	number(8)	

1.2. Movies master table : **Movie**

Column Name	Format	Remarks
mv_no	varchar(2)	Primary key not null
title	varchar2(25)	
type	varchar2(10)	
star	varchar2(25)	

1.3. Invoice transaction table : **Invoice**

Column Name	Format	Remarks
Inv_no	varchar2(3)	primary key not null
Mv_no	varchar(2)	
Cust_id	varchar2(3)	
Issue_date	date	
Return_date	date	

1.4. Add the following constraints

- INVOICE(Cust_id) references CUST(Cust_id)
- INVOICE(Mv_no) references MOVIE(Mv_no)
- Declare NOT NULL : lname, fname, title, type

1.5. Add a new column PRICE in Movie table with data type number(8,2)

Assignment SQL – II

2. Insert the following data into the respective tables :

2.1. **Data for Cust table**

Cust_id	Lname	fname	Area	Phone_no
a01	Bayross	Ivan	sa	6125467
a02	Saitwal	Vandana	mu	5560379
a03	Jaguste	Pramada	da	4563891
a04	Navindgi	Basu	ba	6125401
a05	Sreedharan	Ravi	va	-
a06	-	Rukmini	gh	5125274

2.2. **Data for Movie table**

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	150.55
4	home alone	comedy	macaulay culkin	150.00
5	the fugitive	thriller	harisson ford	200.00
6	coma	suspense	michael douglas	100.00
7	dracula	horror	gary oldman	150.25
8	quick change	comedy	bill muray	100.00
9	gone with the wind	drama	clarke gable	200.00
10	carry on doctor	comedy	leslie phillips	100.00

2.3. **Data for Invoice table**

inv_no	mv_no	cust_id	issue_date	return_date
i01	4	a01	23-july-93	25-jul-93
i02	3	a02	12-aug-93	15-aug-93
i03	1	a02	15-aug-93	18-aug-93
i04	6	a03	10-sep-93	12-sep-93
i05	7	a04	05-aug-93	08-aug-93
i06	2	a06	18-sep-93	21-sep-93
i07	9	a05	07-jul-93	10-jul-93
i08	9	a01	11-aug-93	14-aug-93
i09	5	a03	06-jul-93	07-jul-93
i10	8	a06	03-sep-93	06-sep-93

Assignment SQL – III

3. Write SQL statements to retrieve these query :

3.1. Find out the names of all the customers.
3.2. Print the entire customer table.
3.3. Retrieve the list of fname and the area of all the customers.
3.4. List the various movie types available from the movie table.
3.5. Print the information of 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).

3.6. Change the telephone number of prarnada to 466389.
3.7. Change the issue - date of cust- id 'A01' to 24/07/93.
3.8. Change the price of 'gone with the wind' to Rs. 250. 00.
3.9. Delete the record with invoice number 'l 08' from the invoice table.
3.10. Delete all the records having return date before 10th July'93
3.11. Change the area of cust – id 'A05' to 'vs'.
3.12. Change the return date of invoice number 'I08' to 16-08-93.
3.13. Find the names of all customers having 'a' as the second letter in their fnames.
3.14. Find the lnames of all customers that begin with 's ' or 'j' .
3.15. Find out the customers who stay in an area whose second letter is 'a'.
3.16. Find the list of all customers who stay in area 'da' or area 'mu' or area 'gh'.
3.17. Print the list of employees whose phone numbers are greater than the value 5550000.
3.18. Print the information from invoice table of customers who have been issued movies in the month of September.
3.19. Display the invoice table information for cust – id 'a01' and 'a02'.
3.20. Find the movies of type 'action' and 'comedy'.
3.21. Find the movies whose price is greater than 150 and less than or equal to 200.
3.22. Find the movies that cost more than 150 and also find the new cost as original cost * 15.
3.23. Rename the new column in the above query as new-price.
3.24. List the movies in sorted order of their titles.
3.25. Print the names and types of all the movie except horror movies.
3.26. Divide the cost of movie 'home alone' by difference between its price and 100.
3.27. List the names, areas and cust - id of customers without phone numbers.
3.28. List the names of customers without lname.
3.29. List the mv - no, title, type of movies whose stars begin with letter 'm'.
3.30. List the mv-no and inv-no of customers having inv-no less than 'i05'from the Invoice Transaction Table.

Assignment SQL – IV

4.1. Calculate the square root of the price of each movie.
4.2. Count the total number of customers.
4.3. Calculate the total price of all the movies.
4.4 Calculate the average price of all the movies.
4.5. Determine the maximum and minimum movie prices. Rename the title as max-price and min_price respectively.
4.6. Count the number of movies having price greater than or equal to 150.
4.7. Print the type and average price of each movie.
4.8. Find the number of movies in each type.
4.9. Count separately the number of movies in the 'comedy' and 'thriller' types.
4.10. Calculate the average price for each type that has a maximum price of 150.00.
4.11. Calculate the average price of all movies where type is 'comedy' or 'thriller' and price is greater than or equal to 150.00.
4.12. Display the invoice number and day on which customers were issued movies.

4.13. Display the month (in alphabets) in which customers are supposed to return the movies.
4.14. Display the 15 days after the issue-date in the format 'dd-month-yy'.
For eg. 12-february-93.
4.15. Find the number of days elapsed between the current date and the return date of the movie for all customers.

Assignment SQL – V

5.1. Find out the movie number which has been issued to 'ivan'.
5.2. Find the names and movie numbers of all the customers who have been issued a movie.
5.3. Select the title, cust - id, mv - no for all the movies that are issued.
5.4. Find out the title and types of the movies that have been issued to 'Vandana'.
5.5. Find the names of customers who have been issued movie of type 'drama'.
5.6. Display the title, lname, fname for customers having movie number greater than or equal to three, in the following format:
The movie taken by (fname} {lname} is {title}.
5.7. Find out which customers have been issued movie number 9.
5.8. Find the customer name and area with invoice number 'i10'.
5.9. Find the customer names and phone numbers who have been issued movies before the month of August.
5.10. Find the name of the movie issued to 'vandana' and 'ivan'.
5.11. List the movie number, movie names issued to all customers.
5.12. Find the type and movie number of movie issued to cust – id 'a01' and 'a02'.
5.13. Find out if the movie starring' tom cruise' is issued to any customer and print the custid to whom it is issued.
5.14. Find the lname, fname who have been issued movies.
5.15. Find the name of customer whose has not issued any movie

Assignment SQL – VI

6.1. Create a view with custno, fname, lname of cust table.
6.2. Retrieve all customers name from cust table.
6.3. Create another view for those customers who have been issued movies.
6.4. Create a table with two columns odd number, even number to store odd and even numbers between 1 to 100.
6.5. Generate odd and even numbers using sequences.
6.6. Create a sequence for the column mv_no of Movie table to generate movie no with initial value 10, maximum value 100 and incremented by 1.
6.7. Generate mv_no with sequence above in Movie table.
6.8. Create an index on price column of movie table and retrieve the indexed data.

Assignment SQL – VII

7.1. Create a user with your name

7.2. Give select, insert, update permission on cust, movie table of user 'scott' to your user account

7.3. Assign only insert permission on invoice table of user 'scott' to your current account.

7.4. Check the permission has properly granted using system tables.

7.5. Write a PL/SQL block to print the message "Welcome to PL/SQL" on the screen.

7.6. Write a PL/SQL block to find the summation of given two numeric values.

7.7 Write a PL/SQL to find out the customer id and name who has issued a movie and their elapsed date between issue date and return date is less than two days

7.8. Write a PL/SQL block to find the summation of two numeric values using function where two numeric values will be passed into function as parameters.

7.9. Create an insert trigger to check the price of the each movie must not be greater than 500.

7.10. Add a new column: total_issued in the Movie table to count how many copies has issued to each movie. Create a insert trigger on invoice to increment the values of Total_issued whenever a new rows inserted into the invoice table.