Lab Activity 7

Creating table with Check Constraint :-
Suppose we want to add rule for rating of the sailors - ―Rating should be between 1 to 10‖ while creating table then we can use following command.

```
CREATE TABLE SAILORS
(
 SID NUMBER(3) PRIMARY KEY, SNAME VARCHAR2(30), RATING NUMBER(3),
AGE NUMBER(4,2), CHECK ( RATING >=1AND RATING <=10)
);


CREATE TABLE BOATS
(
 BID NUMBER(3) PRIMARY KEY, BNAME VARCHAR2(20), BCOLOR
VARCHAR2(20)
);

CREATE TABLE RESERVES
 (
SID NUMBER(3), BID NUMBER(3), DAY DATE,PRIMARY KEY(SID,BID,DAY),
FOREIGN KEY(SID) REFERENCES SAILORS,FOREIGN KEY(BID)
REFERENCES BOATS
);
```

Show schema of Sailors
DESC SAILORS;

Show schema of Boats
DESC BOATS;

Show schema of Reserves
DESC RESERVES;

Add records or data to Sailors, Boats as well as Reserves Table.
Insert data in SAILORS Table

```
INSERT INTO SAILORS VALUES(1,'VIJAY', 9, 36);
INSERT INTO SAILORS VALUES(2,'RAJESH', 10, 25);
INSERT INTO SAILORS VALUES(3,'MOHAN', 8, 23);
INSERT INTO SAILORS VALUES(4,'KUMAR', 7, 28);
INSERT INTO SAILORS VALUES(5,'SAGAR', 9, 21);
INSERT INTO SAILORS VALUES(6,'MAHESH', 9, 36)
```

Insert data in BOATS Table

```
INSERT INTO BOATS VALUES(1,'GANGA', 'RED');
INSERT INTO BOATS VALUES(2,'JAMUNA', 'GREEN');
INSERT INTO BOATS VALUES(3,'KAVERI', 'PINK');
INSERT INTO BOATS VALUES(4,'GODAVARI', 'RED');
INSERT INTO BOATS VALUES(5,'KRISHNA', 'BLUE' );
```

Insert data in RESERVES Table

```
INSERT INTO RESERVES VALUES(1,1,'12-FEB-2017');
INSERT INTO RESERVES VALUES(1,2, '12-FEB-2017');
INSERT INTO RESERVES VALUES(2,1,'13-FEB-2017‘);
INSERT INTO RESERVES VALUES(3,2, '14-FEB-2017');
INSERT INTO RESERVES VALUES(3,3,'14-FEB-2017');
```

Experiment 1: Working with Queries USING Aggregate Operators & views

Queries using Aggregate Functions (COUNT, SUM, AVG, MAX and MIN), GROUP BY, HAVING and Creation and Dropping of Views

1. **COUNT (A)** :- The number of values in the A column.Or COUNT (DISTINCT A): The number of unique values in the A column.
   Ex:- 1) To count number SIDs of sailors in Sailors table

   SELECT COUNT (SID) FROM SAILORS;

2) To count numbers of boats booked in Reserves table.
SELECT COUNT (DISTINCT BID) FROM RESERVES;

3) To count number of Boats in Boats table.
SELECT COUNT (*) FROM BOATS;

2. **SUM (A)** :- The sum of all values in the A column.Or SUM (DISTINCT A): The sum of all unique values in the A column.
   Ex:- 1) To find sum of rating from Sailors
   SELECT SUM (RATING) FROM SAILORS;

2) To find sum of distinct age of Sailors (Duplicate ages are eliminated).
SELECT SUM (DISTINCT AGE) FROM SAILORS;

3. **AVG (A)** :- The average of all values in the A column.Or AVG (DISTINCT A): The average of all unique values in the A column.
   Ex:- 1) To display average age of Sailors.
   SELECT AVG (AGE) FROM SAILORS;

2) To find average of distinct age of Sailors (Duplicate ages are eliminated).
SELECT AVG (DISTINCT AGE) FROM SAILORS;

4. **MAX (A)** :- The maximum value in the A column.
   Ex:- To find age of Oldest Sailor.
   SELECT MAX (AGE) FROM SAILORS;

5. **MIN (A)** :- The minimum value in the A column.
   Ex:- To find age of Youngest Sailor.**SELECT MIN (AGE) FROM SAILORS;**

Lab Activity 8

Experiment 2: Working with Queries USING Order By, Having Clause, Group By

1.ORDER BY Clause :- The ORDER BY keyword is used to sort the result-set by a specified column. The ORDER BY keyword sorts the records in ascending order by default (we can even use ASC keyword). If we want to sort the records in a descending order, we can use the DESC keyword.
The general syntax is
SELECT ATT_LIST FROM TABLE_LIST ORDER BY ATT_NAMES [ASC | DESC];
Ex:- 1) Display all the sailors according to their ages.
SELECT * FROM SAILORS ORDER BY AGE;

2) Display all the sailors according to their ratings (topper first).
SELECT * FROM SAILORS ORDER BY RATING DESC;

3) Displays all the sailors according to rating, if rating is same then sort according to age.
SELECT * FROM SAILORS ORDER BY RATING, AGE;

2. GROUP BY:- Group by is used to make each a number of groups of rows in a relation, where the number of groups depends on the relation instances. The general syntax is
SELECT [DISTINCT] ATT_LIST FROM TABLE_LIST WHERE CONDITION GROUP BY GROUPING_LIST;

Ex:- Find the age of the youngest sailor for each rating level.
SELECT S.RATING, MIN (S.AGE) FROM SAILORS S GROUP BY S.RATING;

3. HAVING :- The extension of GROUP BY is HAVING clause which can be used to specify the qualification over group.
The general syntax is
SELECT [DISTINCT] ATT_LIST FROM TABLE_LIST WHERE CONDITION GROUP BY GROUPING_LIST HAVING GROUP_CONDITIION;

Ex :- Find the age of youngest sailor with age >= 18 for each rating with at least 2 such sailors.
SELECT S.RATING, MIN (S.AGE) AS MINAGE FROM SAILORS S WHERE S.AGE >= 18GROUP BY S.RATING HAVING COUNT (*) > 1;

Lab Activity 9

Experiment 3 : Working with views.

VIEWS :- A view is a table whose rows are not explicitly stored in the database but are computed as needed from a view definition.
The views are created using **CREATE VIEW** command.
Ex :- Create a view for Expert Sailors (A sailor is an Expert Sailor if his rating is more than 8).
CREATE VIEW EXPERTSAILOR AS SELECT SID, AGE, RATING FROM SAILORS WHERE RATING > 9;

Now on this view we can use normal SQL statements as we are using on Base tables.
Eg:- Find average age of Expert sailors.
SELECT AVG (AGE) FROM EXPERTSAILOR;

If we decide that we no longer need a view and want to destroy it (i.e. removing the definition of view) we can drop the view. A view can be dropped using the **DROP VIEW** command.
To drop the ExpertSailor view.

DROP VIEW EXPERTSAILOR;