AIM: Nested Queries,Joins,Sorting

Software Used

- **Server:** Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production With the Partitioning, OLAP, Data Mining and Real Application Testing options
- **Client :** SQL*Plus: Release 9.0.1.3.0

Note: Write the appropriate results for all the **examples** as well as **exercises**, on the left hand side of the record

JOINS

SQL joins are used to query data from two or more tables, based on a relationship between certain columns in these tables.

Syntax:

```
SELECT column_list
 FROM table_name
 [INNER|{LEFT|RIGHT|FULL OUTER}] JOIN table_name
     ON qualification_list
WHERE <qualifier>;
```

INNER JOINS:

Only rows that match the search conditions are returned.

Example:

```
SELECT s.sid,s.sname,r.bid
 FROM sailors s INNER JOIN reserves r
 ON s.sid=r.sid;
```

(OR)

```
SELECT s.sid, s.sname ,r.bid
FROM sailors s, reserves r
WHERE s.sid=r.sid;
```

NATURAL JOIN

NATURAL means equi-join for each pair of attributes with the same name.

Example:

```
SELECT sid, sname, bid
 FROM sailors NATURAL JOIN reserves;
```

LEFT OUTER JOIN

Returns all matched rows, plus all unmatched rows from the table on the left of the join clause.
Displays nulls in fields of non-matching tuples.

Example:
```
SELECT s.sid, s.sname ,r.bid
FROM sailors s LEFT OUTER JOIN reserves r
ON s.sid=r.sid;
```

(OR)

```
SELECT s.sid, s.sname ,r.bid
FROM sailors s, reserves r
WHERE s.sid=r.sid(+);
```

Also returns details of sailors who didn't reserve a boat.

RIGHT OUTER JOIN

Returns all matched rows, plus all unmatched rows from the table on the right of the join clause.
Displays nulls in fields of non-matching tuples.

Example:

```
SELECT r.sid,b.bid,b.bname
        FROM reserves r RIGHT OUTER JOIN boats b
        ON r.bid=b.bid;

        (OR)

        SELECT r.sid,b.bid,b.bname
        FROM reserves r,boats b
        WHERE  r.bid(+)=b.bid;
```

Also returns details of boats which don't have any reservations

FULL OUTER JOIN

Returns all matched or unmatched rows from the tables on both sides of the join clause

Example:

```
SELECT r.sid,b.bid,b.bname
FROM reserves r FULL OUTER JOIN boats b
ON r.bid=b.bid;
```

Note: In this case it is the same as the right outer join because bid is a foreign key in reserves, so all
reservations must have a corresponding tuple in boats.

Exercise

1. Find the names of Sailors who have reserved boat number 103
2. Find the names of the Sailors who have reserved a red boat
3. Find the names and colors of the boats reserved by the sailor named Lubber
4. Find the Sailors who have reserved at least one boat
5. Compute Increments for the ratings of sailors who have sailed two different boats on the same day.
6. Find the sailors who have reserved both a red and a green boat.

SORTING

ORDER BY clause:

Syntax: SELECT column_name(s)
FROM table_name
ORDER BY column_name(s) ASC|DESC

Order of sorting

DATA TYPE	ASCENDING	DESCENDING
Number	least to highest	highest to least
Char/Varchar2	A to Z, a to z	z to a, Z to A
Date	latest last	latest first

- Examples:

- SELECT sid, sname, rating, age FROM sailors ORDER BY age;
- SELECT sid,sname,rating,age FROM sailors ORDER BY 4;
- SELECT sid,sname,rating,age FROM sailors ORDER BY rating desc;
- SELECT sid, sname, rating, age FROM sailors ORDER BY 3 desc ;
- SELECT sid, sname, rating R, age FROM sailors ORDER BY R;
- SELECT * FROM sailors ORDER BY 2; //orders by 2nd column in table structure (sname)

- Note: column alias is valid in order by clause (this is the only clause where column alias is valid)

THE GROUP BY and HAVING Clauses:

Often, there are situations when we need to apply aggregate operations to each of a number of *groups* of rows in a relation.

Syntax:

SELECT [DISTINCT] select-list
FROM from-list
WHERE qualification
GROUP BY grouping-list
HAVING group-qualification;

Note 1: All the columns appearing in select-list, except the aggregation-operation must appear in the grouping-list.

2: Expressions appearing in the group-qualification in the HAVING clause must have a single value per group.

Examples:

1.
```
SELECT   S.rating, AVG(S.age) as avg_age
FROM Sailors S
WHERE S.age >=40
GROUP BY S.rating;
```

2.
```
SELECT   S.rating, AVG(S.age) as avg_age
FROM Sailors S
GROUP BY S.rating
HAVING AVG(S.age) >= 40;
```

Exercise:

1. **Find the age of the youngest sailor for each rating level.**
2. **Find the age of the youngest sailor who is eligible to vote, for each rating level with at least two such sailors.**
3. **For each red boat, find the number of reservations for the boat.**
4. **Write a query to give bid and the number of reservations for each boat in the Boats table.**
5. **Find the average age of sailors for each rating level that has at least two sailors.**

You need not include the contents of this slide to your records, just run these on your db before you execute queries 6 and 7

> CREATE TABLE sail AS SELECT * FROM sailors; //will not include the constraints
> INSERT INTO sail VALUES (22,'JOHN',7,35);
>INSERT INTO sail VALUES (22,'JACK',8,40);
>COMMIT;

6. Display duplicate rows

7. Delete duplicate rows

8. a) Find the details of the sailors with - Top 3 ratings
b) Find the details of the Top 3 sailors (Top-N analysis)

9. Display the details of the sailors with the Top 3rd rating

10. Find those rating(s) for which the average age of sailors is the minimum overall ratings.

Query:

```
SELECT sailors1.rating, sailors1.avgage
FROM (SELECT rating, avg(age) as avgage
        FROM Sailors
        GROUP BY rating) sailors1
WHERE sailors1.avgage = (SELECT MIN(avgage)
                         FROM (SELECT rating, avg(age) as avgage
                               FROM Sailors
                               GROUP BY rating) sailors1);
```

Note: The explanation given below need not be written in your lab records.

The query

```
SELECT S.rating
FROM Sailors S
WHERE AVG(S.age) = (SELECT MIN(AVG(S2.age))
FROM Sailors S2
GROUP BY S2.rating);
```

Is wrong because MIN(AVG( )) is not allowed.

Sailors is partitioned into groups by rating, and the average age is computed for each rating value. For each group, applying MIN to this average age value for the group will return the same value.

Viva Questions

1. What is a JOIN Clause?
2. What are the different types of SQL Join clauses and how are they used?
3. Differentiate inner join and outer join
4. Differentiate left join and right join
5. What are different types of outer join?
6. What is difference between joins and union?
7. Can you join table by itself? If Yes how? If no Why?
8. What is Cartesian join?
9. What is a tuple? Are tuple and record related to eachother? If yes how?
10. Define a relation?

Text books

1. Abraham Silberschatz, Henry F Korth, S. Sudarshan, Database System Concepts, 6th Edition, McGraw-Hill International Edition, 2010.
2. Raghu Ramakrishnan, Johannes Gehrke, Database Management Systems, Third Edition, McGraw-Hill International Edition, 2003.
3. Elmasri, Navathe, Somayajulu and Gupta, Fundamentals of Database System, 6 th Edition, Pearson Education, 2011.
4. Patric O'Neil, Elizabeth O'Neil, Database-principles, programming, andperformance,Morgan Kaufmann Publishers, 2001.

References

- **Text References:**

1. https://docs.oracle.com/cd/A58617_01/server.804/a58225/ch3all.htm
2. https://www.tutorialspoint.com/sql/sql-sub-queries.htm
3. https://www.essentialsql.com/what-is-the-difference-between-a-join-and-subquery/
4. https://www.geeksforgeeks.org/nested-queries-sql/
5. https://www.w3resource.com/sql/joins/joining-with-group-by-and-order-by.php

- **Video Links:**

1. https://www.youtube.com/watch?v=lBpSMeQjNqQ
2. https://www.youtube.com/watch?v=G3vz92XZvGA