Oracle Class IV

Joins

Subquery

Join

A join query extracts information from two or more tables or views. A join query differs from a regular query in at least the following two ways:

✓The FROM clause of a join query refers to two or more tables or views.

✓A condition is specified in the join query (known as join condition) that relates the rows of one table to the rows of another table.

```
SELECT
departments.location_id, departments.department_name,
locations.state_province FROM
departments JOIN locations
ON departments.location_id = locations.location_id;
```

Type of Joins

- Inner Join : Inner joins are the regular joins. An inner join returns the rows that satisfy the join condition.

- Outer Join : Outer joins are an extension to inner joins. An outer join returns the rows that satisfy the join condition and also the rows from one table for which no corresponding rows (i.e., that satisfy the join condition) exist in the other table.

  FROM *table1* { LEFT | RIGHT | FULL } [OUTER] JOIN *table2*

  - Left Outer join :
  - Right Outer Join
  - Full outer Join
- Cartesian join or cross join

  when you don't specify a join condition when joining two tables
- *Self joins*

  A self join is a join of a table to itself.
- *Equi- and non-equi-joins*

  An equi-join is a join where the join condition uses the equal to (=) operator to relate the rows of two tables

Other points of join

- ✓ Oracle proprietary syntax is different , inner join is written in table 1, table 2 where table 1.columnname= table2.columnname.
- ✓ A + sign is used to indicate an outer join

```
SELECT FIRSTNAME, LASTNAME, PROJECTNAME
FROM EMPLOYEE E, PROJECT P
WHERE E.PROJECT_ID(+)=P.PROJECT_ID
```

Subquery

- A subquery is simply a query that is nested inside another query or a nested query
- The nested query, sometimes referred to as an inner query, is evaluated first, and its resulting data set is passed back to the outer query and evaluated to completion.
- Subqueries can be effectively used in situations where the combined data has no direct relationship as compared to joins
- scalar or single-row subqueries :The subquery returns a single value to the outer query
- Scaler subqueries can be used for both where and having clause.
- Subqueries are most often found in the WHERE clause of a SELECT, UPDATE, or DELETE statement

- Ex : Select name of the employees other than Kevin Feeney who gets the same salary as Kevin

Non correlated subquery

- Noncorrelated subqueries allow each row from the containing SQL statement to be compared to a set of values.
- The nested query, sometimes referred to as an inner query, is evaluated first, and its resulting data set is passed back to the outer query and evaluated to completion.
- Subqueries can be effectively used in situations where the combined data has no direct relationship as compared to joins
- scalar or single-row subqueries :The subquery returns a single value to the outer query
- Scaler subqueries can be used for both where and having clause.

Correlated subquery

- A correlated subquery is a subquery that uses values from the outer query, requiring the inner query to execute once for each outer query

- With a correlated subquery, the database must run the subquery for each evaluation because it is based on the outer query's data.

```
select book_key, store_key, quantityfrom sales s
Where quantity <
    (select max(quantity) from sales where book_key = s.book_key);
```

```
SELECT S.Number, S.Name FROM Salesman S
WHERE S.Number IN (SELECT C.Salesman FROM Customer C
  WHERE C.Name = S.Name)
```

Non correlated and correlated subquery – contd.

- Noncorrelated subqueries allow each row from the containing SQL statement to be compared to a set of values.
  - Single-row, single-column subqueries
  - Multiple-row, single-column subqueries
  - Multiple-column subqueries

SELECT lname FROM employee WHERE salary > (SELECT AVG(salary) FROM employee);

SELECT fname, lname FROM employee WHERE dept_id = 30 AND salary >= ALL (SELECT salary FROM employee WHERE dept_id = 30);

SELECT fname, lname FROM employee WHERE dept_id = 30 AND NOT salary < ANY (SELECT salary FROM employee WHERE dept_id = 30);

Assignments

- Display name of employees , department name and job name for each employee

- Display the department name along with no of employees and average salary of that department

- For each department , find out no. of jobs the employees are assigned to.

- Check for correctness of the above queries in terms of count, if you want to bring in all entries , how would you achieve the same?

- Group by the employees based on the first character of employee first name. Display the results in alphabetic order (descending) of first character.

- Display name of those employees who get a salary more than the average salary

- Display name of the all the employees who are 'stock manager' , except the one who gets the minimum salary .

Assignments contd.

- Display firstname,lastname,salary of those sales representatives who earns a higher salary than the minimum salary a sales manager receives.

- Display the name of the employees/employee who gets the second highest salary. (sub query)

- Come up with the query for previous question using set operators

- Display the name of the employee (manager) who has the maximum no. of employees reporting to him.

- Display the name of those employees , who are in the same department as Timothy Gates and gets an salary more than the average salary of all the employees

Assignments contd.

- If an employee have spent less than 5 years then he is considered entry level id 5 – 10 then midlevel else a senior employee. Write a query, which will label the employees in either of the above categories

- Write query to find out any departments that are present in department table but does not have employees

- Write a query which will display job id , which are present in both job and employee columns

- Increase salary of each employee of all the department who draws the minimum salary by 100$.