Ex.No.: 8	
Date: 3/9/24	**WORKING WITH MULTIPLE TABLES**

Objective

After the completion of this exercise, the students will be able to do the following:
• Write SELECT statements to access data from more than one table using equality and nonequality joins
• View data that generally does not meet a join condition by using outer joins
• Join a table to itself by using a self join
Sometimes you need to use data from more than one table.

Cartesian Products
• A Cartesian product is formed when:
− A join condition is omitted
− A join condition is invalid
− All rows in the first table are joined to all rows in the second table
• To avoid a Cartesian product, always include a valid join condition in a WHERE clause.
A Cartesian product tends to generate a large number of rows, and the result is rarely useful. You should always include a valid join condition in a WHERE clause, unless you have a specific need to combine all rows from all tables.
Cartesian products are useful for some tests when you need to generate a large number of rows to simulate a reasonable amount of data.

Example:
To displays employee last name and department name from the EMPLOYEES and DEPARTMENTS tables.

SELECT last_name, department_name dept_name
FROM   employees, departments;

Types of Joins

• Equijoin
• Non-equijoin
• Outer join
• Self join
• Cross joins
• Natural joins
• Using clause
• Full or two sided outer joins
• Arbitrary join conditions for outer joins

Joining Tables Using Oracle Syntax

SELECT table1.column, table2.column

This query was completed in earlier releases as follows:

SELECT e.last_name, e.department_id, d.department_name
FROM   employees e, departments d
WHERE d.department_id = e.department_id (+);

FULL OUTER JOIN
Example:

SELECT e.last_name, e.department_id, d.department_name
FROM   employees e
FULL OUTER JOIN departments d
ON   (e.department_id = d.department_id) ;

This query retrieves all rows in the EMPLOYEES table, even if there is no match in the DEPARTMENTS table. It alslso retrieves all rows in the DEPARTMENTS table, even if there is no match in the EMPLOYEES table.

Find the Solution for the following:

1. Write a query to display the last name, department number, and department name for all employees.

> Select e.last-name , e. department_id , d. department_name
> from employees e , departments d while e.departmed_id
> = d.department_id ;

2. Create a unique listing of all jobs that are in department 80. Include the location of the department in the output.

> select distinct job-id, location_id from employes
> departments where employees . department_id =
> department . department_id and employees .
> department_id = 80 ;

3. Write a query to display the employee last name, department name, location ID, and city of all employees who earn a commission

> Select e. last-name , d. department_name , d. location.
> e. city from employees e , department d , location e
> where e. department_id : d . department_id and
> d. location_id . d. location-id and
> e. commission-pct is not null;

2. Display the employee last name and department name for all employees who have an a(lowercase) in their last names. P

Select last-name, department-name from employee, departments where employees.department-id = departments.department-id and last-name like '%a';

5. Write a query to display the last name, job, department number, and department name for all employees who work in Toronto.

Select e.last-name, e.job-id, e.department-id, d.department name from employees e JOIN departments d ON (e.department-id = d.department-id) JOIN locations l on (d.location-id = d.location-id) where lower (l.cities) = 'toronto';

6. Display the employee last name and employee number along with their manager's last name and manager number. Label the columns Employee, Emp#, Manager, and Mgr#, Respectively

Select w.last-name "Employee", W.employee-id "EMP#", M.last-name "Manager", m.employee-id "Mgr#" from employees W JOIN employees m on (w.manager-id = m.employee-id);

7. Modify lab4_6.sql to display all employees including King, who has no manager. Order the results by the employee number.

Select w.last-name "Employee", w.employee-id "EMP#", M.last-name "Manager", M.employee-id "Mgr#", from employees W left outer join employees m on (w.Manager-id = m.employee-id);

8. Create a query that displays employee last names, department numbers, and all the employees who work in the same department as a given employee. Give each column an appropriate label

Select e.department-id department, e.last-name employee, c.last-name colleague from employees e JOIN employees c ON (e.department-id = c department-id) where e.employee-id <> c.employee-id order by e.department-id, e.last-name, c.last-name;

9. Show the structure of the JOB_GRADES table. Create a query that displays the name, job, department name, salary, and grade for all employees

Desec Job-Grades
Select e.last-name, e.job-id, d.department-name, e.salary, grade-level from employees e, departments d, job-grades j where e.department-id = d.department-if AND e.salary b/w j.lowest-sal and j.highest-sal;

10. Create a query to display the name and hire date of any employee hired after employee Davies.

Select e.last_name, e.hire_date from employees e,
employees davies whose devies, last_name = 'Davies'
AND davies.hire_date < e.hire_date.

11. Display the names and hire dates for all employees who were hired before their managers, along with their manager's names and hire dates. Label the columns Employee, Emp Hired, Manager, and Mgr Hired, respectively.

Select w.last_name, w.hire_data, M.last_name,
M.hire_date from employees W, employees M where
W.manager_id = M.employee_id AND W.hire_date
< M.hire_date;

Evaluation Procedure	Marks awarded
Query(5)	5
Execution (5)	5
Viva(5)	5
Total (15)	15
Faculty Signature	