Practice 6 -1 Cross Joins and Natural Joins

CROSS JOIN	Returns the Cartesian product from two tables
NATURAL JOIN	Joins two tables based on the same column name

1. Create a cross-join that displays the last name and department name from the employees and department tables

   SELECT last_name, first_name, department_name
   FROM employees CROSS JOIN departments;

2. Create a query that uses a natural join to join the departments tables and the locations table. Display the department id, department name, location id, and city.

   160 rows

3. Create a query that uses a natural join to join the departments tables and the locations table. Restrict the output to only department IDs of 20 and 50. Display the department id, department name, location id, and city.

   SELECT department_id,department_name, location_id, city
   FROM departments NATURAL JOIN locations;

Practice 6-2 Join Clauses

ON clause	Allows a natural join based on an arbitrary condition or two columns with different names
USING clause	Performs an equijoin based on one specified column name

1. Join the Oracle database locations and departments table using the location_id column. Limit the results to location 1400 only.

   SELECT department_id,department_name, location_id, city
   FROM departments JOIN locations USING (location_id)
   WHERE location_id = 1400;

2. Join DJs on Demand d_play_list_items, d_track_listings, and d_cdss tables with the JOIN USING syntax. Include the song ID, CD number, title, and comments in the output.

   SELECT song_id, cd_number, title, comments

FROM d_cds JOIN d_track_listings USING (cd_number) JOIN d_play_list_items USING (song_id);

3. Display the city, department name, location ID, and department ID for departments 10, 20, and 30 for the city of Seattle.

   SELECT city, department_name, location_id, department_id
   FROM departments JOIN locations USING (location_id)
   WHERE department_id in (10, 20 , 30) AND city = 'Seattle';

4. Display country name, region ID, and region name for Americas.

   SELECT country_name, region_id, region_name
   FROM countries JOIN regions USING(region_id)
   WHERE region_name = 'Americas';

5. Write a statement joining the employees and jobs tables. Display the first and last names, hire date, job id, job title, and maximum salary. Limit the query to those employees who are in jobs that can earn more than $12,000.

   SELECT first_name, last_name, hire_date, job_id, job_title, max_salary
   FROM employees JOIN jobs USING (job_id)
   WHERE max_salary > 12000;

6. Display job title, employee first name, last name, and email for all employees who are stock clerks.

   SELECT job_title, first_name, last_name, LOWER(email) || 'somecomname.sometld' as email
   FROM employees JOIN jobs USING(job_id)
   WHERE job_title = 'Stock Clerk' ;

7. Write a statement that displays the employee ID, first name, last name, manager ID, manager first name, and manager last name for every employee in the employees table. (this is a self-join)

SELECT emp.employee_id AS "employee ID", emp.first_name AS "first name", emp.last_name "last name", emp.manager_id "manager ID", mgr.first_name "manager first name", mgr.last_name "manager last name"
FROM employees emp LEFT JOIN employees mgr ON emp.manager_id = mgr.employee_id;

8. Use JOIN ON syntax to query and display the location ID< city, and department name for all Canadian locations.

```
SELECT dp.location_id, loc.city, dp.department_name
FROM departments dp INNER JOIN locations  loc ON dp.location_id = loc.location_id
INNER JOIN  countries ct ON loc.country_id = ct.country_id
WHERE ct.country_name = 'Canada';
```

9. Query and display manager ID, department ID, department name, first name, and last name for all employees in department 80, 90, 110, and 190.

```
        SELECT emp.manager_id "Employee's Manager ID", emp.department_id
    "Department ID", dpt.department_name  "Department Name", emp.first_name
    "First Name", emp.last_name "Last Name"
    FROM  employees emp INNER JOIN departments dpt ON emp.department_id =
dpt.department_id
WHERE emp.department_id in (80, 90, 110, 190);
```

10. Display employee ID, last name, department ID, department name, and hire date for those employees whose hire date was July 7, 1994.

```
SELECT emp.Employee_id "Employee ID", emp.last_name "Last Name" ,
emp.department_id "Department ID", dpt.department_name  "Department Name",
emp.hire_date "Hire Date"
FROM  employees emp LEFT JOIN departments dpt ON emp.department_id =
dpt.department_id

WHERE emp.hire_date = TO_DATE('June 7, 1994', 'fmMonth DD, YYYY');
```

Practice 6-3 Inner versus Outer Joins

FULL OUTER JOIN	Performs a join on two tables, retrieves all the rows in the Left table, even if there is no match in the Right table. IT also retrieves all the rows int hr Right table, even if there is no match in the Left table.
OUTER JOIN	A join that returns the unmatched rows as well as matched rows.

LEFT OUTER JOIN	Performs a join on two tables, retrieves all the rows in the Left table even if there is no match in the Right table.
RIGHT OUTER JOIN	Performs a join on two tables, retrieves all the rows in the Right table even if there is no match in the Left table.
INNER JOIN	A join of two or more tables that returns only matched rows

1. Return the first name, last name, and department name for all employees including those employees not assigned to a department.

   SELECT emp.first_name "First Name", emp.last_name "Last Name" ,
   dpt.department_name "Department Name"
   FROM employees emp LEFT OUTER JOIN departments dpt ON emp.department_id =
   dpt.department_id;

2. Return the first name, last name, and department name for all employees including those departments that do not have an employee assigned to them.

   SELECT emp.first_name "First Name", emp.last_name "Last Name" ,
   dpt.department_name "Department Name"
   FROM employees emp RIGHT OUTER JOIN departments dpt ON emp.department_id =
   dpt.department_id;

3. Return the first name, last name, and department name for all employees including those departments that do not have an employee assigned to them.

   SELECT emp.first_name "First Name", emp.last_name "Last Name" ,
   dpt.department_name "Department Name"
   FROM employees emp FULL OUTER JOIN departments dpt ON emp.department_id =
   dpt.department_id;

4. Create a query of the DJs on Demand database to return the first name, last name, event date, and description of the event the client held. Include all the clients even if they have not has an event scheduled.

   SELECT ct.first_name, ct.last_name, ev.event_date, ev.description
   FROM d_clients ct LEFT OUTER JOIN d_events ev ON ct.client_number =
   ev.client_number;

5. Using the Global Fast Foods database, show the shift description and shift assignment date even of there is no date assigned for each shift description.

```
SELECT f_shifts.description "shift description", f_shift_assignments.shift_assign_date AS
"shift assignment date"
FROM  f_shifts LEFT OUTER JOIN f_shift_assignments ON f_shifts.code =
f_shift_assignments.code;
```

Practice 6-3

Self join	Joins a table to itself
Hierarchical query	Retrieves data based on a natural hierarchical relationship between rows in a table
level	Determines the number of steps down from the beginning row that should be returned by a hierarchical query
Start with	Identifies the beginning row for a hierarchical query
Connect by	Specifies the relationship between parent rows and child rows of a hierarchical query

1. Display the employe's last number and employee number along with the manager's last name and manager number. Label the columns. Employee, Emp#, Manager, and Mgr#, respectively.

```
SELECT emp.last_name "Employee", emp.employee_id "Emp#", mgr.last_name
"Manager", mgr.employee_Id  "Mgr#"
FROM  employees emp INNER JOIN  employees  mgr  ON emp.manager_id  =
mgr.employee_Id;
```

2. Modify question 1 to display all employees and their managers, even if the employee does not have a manager. Order the list alphabetically by the last name of the employee.

```
SELECT emp.last_name "Employee", emp.employee_id "Emp#", mgr.last_name
"Manager", mgr.employee_Id  "Mgr#"
FROM  employees emp LEFT OUTER JOIN  employees  mgr  ON emp.manager_id  =
mgr.employee_Id
ORDER BY  "Employee";
```

3. 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 emp.last_name "Employee",  emp.hire_date "Emp Hired",   mgr.last_name
"Manager", mgr.hire_date "Mgr Hired"
FROM  employees emp LEFT OUTER JOIN  employees  mgr  ON emp.manager_id  =
mgr.employee_Id
WHERE emp.hire_date < mgr.hire_date
ORDER BY  "Employee";
```

4. Write a report that shows the hierarchy for Lex De Haans department. Include last name, salary, and department id in the report.

```
SELECT last_name, salary, department_id
FROM employees
START WITH first_name = 'Lex'  AND last_name = 'De Haan'
CONNECT BY PRIOR employee_id = manager_id;
```

5. What is wrong in the following statement?

```
SELECT last_name, department_id, salary
FROM employees
START WITH last_name = ‘King’
CONNECT BY PRIOR manager_id = employee_id;
```

```
SELECT LEVEL, last_name, department_id, salary
FROM employees
START WITH first_name = 'Lex'  AND last_name = 'De Haan'
CONNECT BY  employee_id = PRIOR manager_id;
```

6. Create a report that shows the organization chart for the entire employee table. Write the report so that each level will indent each employee 2 spaces. Since Oracle Application Express cannot display the spaces in front of the column, use - (minus) instead.

```
SELECT LPAD(last_name, LENGTH(last_name) + (LEVEL-1)*2, '-') "organization chart "
FROM employees
START WITH last_name = ( SELECT last_name from employees WHERE manager_id IS NULL)
CONNECT BY PRIOR employee_id = manager_id;
```

7. Re-write the report from 6 exclude De Haan and all the people working for him.

```
SELECT LPAD(last_name, LENGTH(last_name) + (LEVEL-1)*2, '-') "organization chart "
FROM employees
START WITH last_name = ( SELECT last_name from employees WHERE manager_id IS NULL)
```

```
CONNECT BY PRIOR employee_id = manager_id AND last_name != 'De Haan';
```