Database Programming

Review of Joins

ORACLE ACADEMY

Objectives

This lesson covers the following objectives:

- Determine the correct join syntax to use given a scenario requiring the join of data from two or more tables.

Purpose

Knowing when to use the correct join syntax to meet the needs stated in a business scenario requiring the join of data is very important to your success. This lesson will afford you the opportunity to review the join syntax.

Classroom Activity

Try the examples listed on the following slides. Confirm that your results match the expected result. If you need help, ask another student or your teacher. All example code is based on the Oracle database.

Cross Join

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

All rows will show.

Classroom Activity (cont.)

Natural Join

```
SELECT employee_id, last_name, department_name
FROM employees NATURAL JOIN departments;
```

Joins by column names and data types that are identical in each table. Both the employees and departments tables have the columns department_id and manager_id. Therefore, the query will return the rows where the values in both columns match.

Classroom Activity (cont.)

Join .. Using

```
SELECT employee_id, last_name, department_name
FROM employees JOIN departments
USING (department_id);
```

Joins by column names and data types that are identical in each table but the USING statement limits to one column.

Classroom Activity (cont.)

Join .. On

```
SELECT e.employee_id, e.last_name, d.department_id, d.location_id
FROM employees e JOIN departments d
ON (e.department_id = d.department_id);
```

All employees and their work locations.

Classroom Activity (cont.)

Join .. On

```
SELECT e.employee_id, e.last_name, e.salary, j.grade_level
FROM employees e JOIN job_grades j
ON (e.salary BETWEEN j.lowest_sal AND j.highest_sal);
```

This displays the grade level for each employee based on salary.

Classroom Activity (cont.)

Right Outer Join

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

Retrieves all data in the right table (DEPARTMENTS). Returns departments with employees as well as departments without employees.

Classroom Activity (cont.)

Left Outer Join

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

Retrieves all data in the left table (EMPLOYEES). Returns employees who are assigned to a department as well as employees who are not assigned to a department.

Classroom Activity (cont.)

Full Outer Join

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

Retrieves all data in the left table and all data in the right table. This includes departments with employees as well as departments without employees. It also includes employees who are assigned to a department as well as employees who are not assigned to a department.

Summary

In this lesson, you should have learned how to:

- Determine the correct join syntax to use given a scenario requiring the join of data from two or more tables