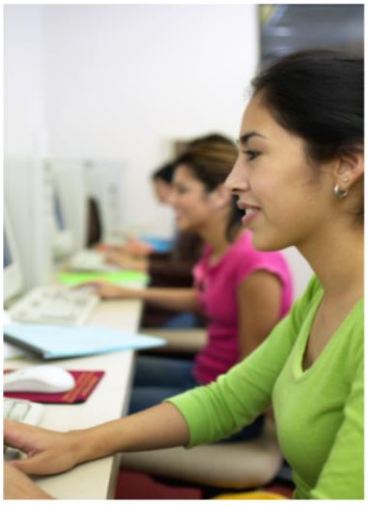

ORACLE
Academy

Database Programming with SQL

7-1

Oracle Equijoin and Cartesian Product



ORACLE
Academy

Objectives

- This lesson covers the following objectives:
  - Name the Oracle proprietary joins and their ANSI/ISO SQL: 99 counterparts
  - Construct and execute a SELECT statement that results in a Cartesian product
  - Construct and execute SELECT statements to access data from more than one table using an equijoin
  - Construct and execute SELECT statements that add search conditions using the AND operator
  - Apply the rule for using table aliases in a join statement

Purpose

- The previous section looked at querying and returning data from more than one table in a relational database using ANSI/ISO SQL: 99 syntax
- Legacy versions of Oracle databases required joins to use Oracle Proprietary join syntax, and many of these older databases are still in use
- This lesson introduces Oracle Proprietary join syntax for Equijoins and Cartesian Product, and their ANSI/ISO SQL: 99 counterparts

Versions prior to Oracle 9i Database required the use of Oracle Proprietary join syntax.

Join Commands

- The two sets of commands or syntax which can be used to make connections between tables in a database:
  - Oracle proprietary joins
  - ANSI/ISO SQL: 99 compliant standard joins

Join Comparison

- Comparing Oracle Proprietary Joins with ANSI/ISO SQL: 1999 Joins

Oracle Proprietary Join	ANSI/ISO SQL: 1999 Equivalent
Cartesian Product	Cross Join
Equijoin	NATURAL JOIN JOIN USING clause JOIN ON clause (if the equality operator is used)
Non-equijoin	ON clause

ORACLE Proprietary Joins

- To query data from more than one table using the Oracle proprietary syntax, use a join condition in the WHERE clause
- The basic format of a join statement is:

```
SELECT table1.column, table2.column
FROM table1, table2
WHERE table1.column1 = table2.column2;
```

ORACLE Proprietary Joins

- Imagine the problem arising from having two students in the same class with the same last name
- When needing to speak to "Jackson," the teacher clarifies which "Jackson" by prefacing the last name with the first name
- To make it easier to read a Join statement and to speed up database access, it is good practice to preface the column name with the table name

```
SELECT table1.column, table2.column
FROM table1, table2
WHERE table1.column1 = table2.column2;
```

ORACLE Proprietary Joins

- This is called "qualifying your columns"
- The combination of table name and column name helps eliminate ambiguous names when two tables contain a column with the same column name
- When the same column name appears in both tables, the column name must be prefaced with the name of the table

Join Syntax Example

- To qualify the columns, you use the syntax tablename.columnname as shown in the example below

```
SELECT table1.column, table2.column
FROM table1, table2
WHERE table1.column1 = table2.column2;
```

EQUIJOIN

- Sometimes called a "simple" or "inner" join, an equijoin is a table join that combines rows that have the same values for the specified columns
- An equijion is equavalent to ANSI:
  - NATURAL JOIN
  - JOIN USING
  - JOIN ON (when the join condition uses "=")
- The next slide demonstrates the what, where and how required to join the tables

EQUIJOIN

- What? The SELECT clause specifies the column names to display
- Where? The FROM clause specifies the tables that the database must access, separated by commas
- How? The WHERE clause specifies how the tables are to be joined
- An Equijoin uses the equals operator to specify the join condition

EQUIJOIN

```
SELECT employees.last_name, employees.job_id, jobs.job_title
FROM employees, jobs
WHERE employees.job_id = jobs.job_id;
```

What?

Where?

How?

LAST_NAME	JOB_ID	JOB_TITLE
King	AD_PRES	Presiden
Kochhar	AD_VP	Administration Vice President
De Haan	AD_VP	Administration Vice President
Whalen	AD_ASST	Administration Assistant
Higgins	AC_MGR	Accounting Manager
Gietz	AC_ACCOUNT	Public Accountant
Zlotkey	SA_MAN	Sales Manager
Abel	SA_REP	Sales Representative
…	…	…

EQUIJOIN

- Another example:

```
SELECT employees.last_name, departments.department_name
FROM employees, departments
WHERE employees.department_id = departments.department_id;
```

LAST_NAME	DEPARTMENT_NAME
Whalen	Administration
Hartstein	Marketing
Fay	Marketing
Mourgos	Shipping
Rajs	Shipping
Davies	Shipping
Matos	Shipping
…	…

Note: the column used to join both tables does not need to be in the SELECT column list.

Aliases

- Working with lengthy column and table names can be cumbersome
- Fortunately, there is a way to shorten the syntax using aliases
- To distinguish columns that have identical names but reside in different tables, use table aliases
- A table alias is similar to a column alias; it renames an object within a statement
- It is created by entering the new name for the table just after the table name in the from-clause

Table Aliases

- Table aliases are used in the query below.

```
SELECT last_name, e.job_id, job_title
FROM employees e, jobs j
WHERE e.job_id = j.job_id
  AND department_id = 80;
```

LAST_NAME	JOB_ID	JOB_TITLE
Zlotkey	SA_MAN	Sales Manager
Abel	SA_REP	Sales Representative
Taylor	SA_REP	Sales Representative

- When column names are not duplicated between two tables, you do not need to add the table name or alias to the column name

Table Aliases

- If a table alias is used in the FROM clause, then that table alias must be substituted for the table name throughout the SELECT statement
- Using the name of a table in the SELECT clause that has been given an alias in the FROM clause will result in an error

```
SELECT last_name, employees.job_id, job_title
FROM employees e, jobs j
WHERE e.job_id = j.job_id
  AND department_id = 80;
```

Cartesian Product Join

- If two tables in a join query have no join condition specified in the WHERE clause or the join condition is invalid, the Oracle Server returns the Cartesian product of the two tables
- This is a combination of each row of one table with each row of the other
- A Cartesian product is equivalent to an ANSI CROSS JOIN
- To avoid a Cartesian product, always include a valid join condition in a WHERE clause

Cartesian Product Join

- In this query, the join condition has been omitted:

```
SELECT employees.last_name, departments.department_name
FROM employees, departments;
```

LAST_NAME	DEPARTMENT_NAME
Abel	Administration
Davies	Administration
De Haan	Administration
Ernst	Administration
Fay	Administration
Gietz	Administration
Grant	Administration
…	…

160 rows returned in 0.01 seconds

ORACLE Academy

Restricting Rows In a Join

- As with single-table queries, the WHERE clause can be used to restrict the rows considered in one or more tables of the join
- The query shown uses the AND operator to restrict the rows returned

```
SELECT employees.last_name, employees.job_id, jobs.job_title
FROM employees, jobs
WHERE employees.job_id = jobs.job_id
  AND employees.department_id = 80;
```

LAST_NAME	JOB_ID	JOB_TITLE
Zlotkey	SA_MAN	Sales Manager
Abel	SA_REP	Perwakilan Penjualan
Taylor	SA_REP	Perwakilan Penjualan

Join Syntax Example

- If you wanted to join three tables together, how many joins would it take?
- How many bridges are needed to join three islands?
- To join three tables, you need to add another join condition to the WHERE clause using the AND operator

Join Syntax Example

- Suppose we need a report of our employees and the city where their department is located?
- We need to join three tables: employees, departments and locations

```
SELECT last_name, city
FROM employees e, departments d, locations l
WHERE e.department_id = d.department_id
   AND d.location_id = l.location_id;
```

LAST_NAME	CITY
Hartstein	Toronto
Fay	Toronto
Zlotkey	Oxford
Abel	Oxford
…	…

ORACLE Academy

Terminology

- Key terms used in this lesson included:
  - Alias
  - Cartesian Product
  - Equijoin
  - Join Conditions
  - Proprietary Join

Summary

- In this lesson, you should have learned how to:
  - Name the Oracle proprietary joins and their ANSI/ISO SQL: 99 counterparts
  - Construct and execute a SELECT statement that results in a Cartesian product
  - Construct and execute SELECT statements to access data from more than one table using an equijoin
  - Construct and execute SELECT statements that add search conditions using the AND operator
  - Apply the rule for using table aliases in a join statement


ORACLE
Academy