# *Database Management I (420-D10-HR)*

# *Lab 13 - Multiple Table Queries I*

Date assigned: Monday, November 7, 2016

Date due: **Monday, November 7, 2016, 4:50pm**

**Learning Objectives**

Upon successful completion of this lab exercise, the student will be able to:

* 1. explain the difference between a Cartesian product, an equijoin, a non-equijoin, an outer join and a self join
  2. write SQL queries joining two or more tables using an equijoin
  3. use table aliases in a SELECT query
  4. write SQL queries joining two or more tables using an non-equijoin
  5. write SQL queries joining two or more tables using an outer join
  6. write SQL queries joining a table to itself using an self join

Review functions, grouping, sorting.

***References:*** w3schools.com

SQL Cheat Sheet (Moodle)

[Equi join](http://www.orafaq.com/wiki/Equi_join)

[Nonequi join](http://www.orafaq.com/wiki/Nonequi_join) (see the See Also, portion of this page for other types of joins)

**To uploaded to Moodle:**

1. The ***username\_*D10*\_*L13\_Multiple\_Tables.docx** file containing the SELECT statements you wrote for this lab.

**To Start:**

Rename this document to ***username\_*D10**\_**L13\_Multiple\_Table\_Queries.docx**.

You will be using the ShahDB . The script to create the DB is in Moodle. This is the same database from last week’s lab, so you won’t need to run the installation script again unless you think you’ve changed the database and would like to recover.

The model is show in an appendix to this document.

**To Do:**

Copy the **SELECT** statements that you code for the questions in this lab into the appropriate locations in this file.

I also want to see a sample of your output. Include the column headers and the first 5-10 lines of output.

## **Part A – N2 Corporation**

You will be using the N2 Corporation Database in these exercises. These are the tables in the ShahDB that start with NN\_.

The key table to understand is the NN\_EMPLOYEE table. In Oracle SQL Developer, double click on this table and select the Model tab to see a datamodel of the tables you need to know for this lab.

## **Note the following**:

### All names should be displayed as *lastname, firstname*.

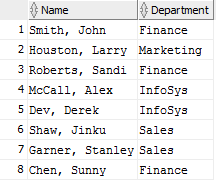
### Salaries should be output in currency format.

## Display all employee names and their department names.

SELECT e.lname||', '||e.fname AS "Name", d.deptname AS "Department"

FROM nn\_employee e, nn\_dept d

WHERE e.deptid = d.deptid;



## Find name of the supervisor for employee number 433.

SELECT e1.lname||', '||e1.fname AS "Employee Name"

, e2.lname||', '||e2.fname AS "Supervisor Name"

FROM nn\_employee e1, nn\_employee e2

WHERE e1.supervisor = e2.employeeid AND e1.employeeid = 433;



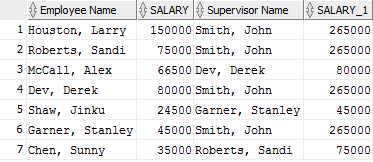
## Find all employee’s full names (lastname, firstname format) with salary and their supervisor’s name with salary.

SELECT e1.lname||', '||e1.fname AS "Employee Name", e1.salary

, e2.lname||', '||e2.fname AS "Supervisor Name", e2.salary

FROM nn\_employee e1, nn\_employee e2

WHERE e1.supervisor = e2.employeeid;

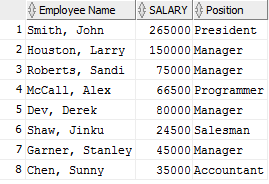


## Find each employee’s salary information and level based on the salary.

SELECT e1.lname||', '||e1.fname AS "Employee Name", e1.salary, p.posdesc AS "Position"

FROM nn\_employee e1, nn\_position p

WHERE e1.positionid = p.positionid;



## Display each employee’s name, department name, position description, and qualification description. Which employee is missins? Why?

SELECT e.lname||', '||e.fname AS "Employee Name"

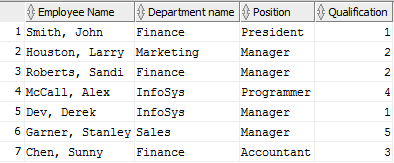
, d.deptname AS "Department name"

, p.posdesc AS "Position"

, e.qualid AS "Qualification"

FROM nn\_employee e, nn\_position p, nn\_dept d

WHERE e.positionid = p.positionid AND e.deptid = d.deptid and e.qualid IS NOT NULL;



## Find all employees in the sales department.

SELECT e.lname||', '||e.fname AS "Employee Name"

, d.deptname AS "Department name"

FROM nn\_employee e, nn\_dept d

WHERE e.deptid = d.deptid AND d.deptname='Sales';



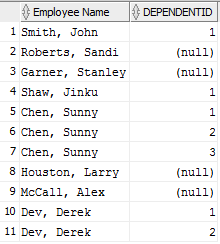
## Display employee names and dependent information using an outer join.

SELECT e.lname||', '||e.fname AS "Employee Name", d.dependentid

FROM nn\_employee e

LEFT JOIN nn\_dependent d

ON e.employeeid = d.employeeid;



## Find the names of employees and number of yers worked along with their department names in descending order by number of years worked.

SELECT e.lname||', '||e.fname AS "Employee Name",

EXTRACT(year FROM current\_timestamp) - EXTRACT(year FROM e.hiredate) AS "Years with dept"

, d.deptname AS "Department name"

FROM nn\_employee e, nn\_dept d, nn\_job\_history j

WHERE e.deptid = j.deptid AND e.deptid = d.deptid

ORDER BY 'Years with dept' desc;



## Who works in the same department in which John Smith works?

SELECT e1.lname||', '||e1.fname AS "Employee Name",

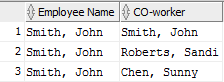
e2.lname||', '||e2.fname AS "CO-worker"

FROM nn\_employee e1, nn\_employee e2

WHERE e2.deptid = e1.deptid

AND e1.fname='John'

AND e1.lname='Smith';



## **Part B – IU College Database**

You will be using the IU College Database in these exercises. These are the tables in the ShahDB that start with IU\_.

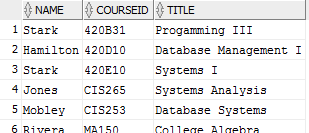
## Get Spring 2003 Course sections with the faculty member assigned to teach the class. Include course sections without any faculty assigned to them.

SELECT f.name, c1.courseid, c2.title

FROM iu\_crssection c1, iu\_course c2, iu\_faculty f

WHERE c1.facultyid = f.facultyid

AND c1.courseid = c2.courseid;



## Display course titles along with their prerequisite names. Display courses without prerequisites as well.

SELECT c1.title, c2.title AS "Prerequisite"

FROM iu\_course c1, iu\_course c2

WHERE c1.prereq = c2.courseid

union

SELECT c1.title "course title", 'none' "Prerequisite"

FROM iu\_course c1

WHERE c1.prereq IS NULL;

## 

## Create a sorted student phone list of all students **currently** being taught by a teacher whose name is input at run time. ***Note***: A student is currently taking a course if his/her final grade in the course is null.

SELECT s.last||', '||s.first AS "Name",

'('||substr(s.phone,0,3)||') '||substr(s.phone,4,3)||'-'||substr(s.phone, 7) AS "Phone number"

FROM iu\_student s, iu\_faculty f

WHERE s.facultyid = f.facultyid AND f.name = '&name';

A sample run is shown here. Jones was entered as the faculty name.

Student Name Phone

------------------------------- ----------------------

Khan, Amir (201) 555-6666

Lee, Brian (212) 555-5555

Patel, Rajesh (732) 555-3333

Rickles, Deborah (732) 555-4444

## For all courses, list the termid, the courseid, the title, the number of sections offered and the total number of students who have registered in them. Courses that have never been offered and course sections with no students registered should be included. Sort the output by termid and courseid.

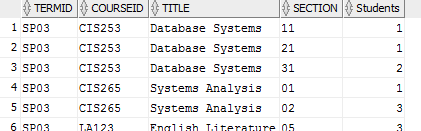
SELECT c1.termid, c1.courseid, c2.title, c1.section, COUNT(r.studentid) AS "Students"

FROM iu\_crssection c1, iu\_course c2, iu\_registration r

WHERE c1.courseid = c2.courseid AND r.csid = c1.csid

GROUP BY c1.termid, c1.courseid, c2.title, c1.section

ORDER BY c1.termid, c1.courseid;



A sample run is shown here. (Your data may be different for the Heritage courses.)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TERMID** | **COURSEID** | **TITLE** | **Sections** | **Students** |
| FL14 | 420B31 | Programming III | 1 | 0 |
| FL14 | 420C30 | Web Programming III | 1 | 0 |
| FL14 | 420D10 | Database Management I | 1 | 0 |
| FL14 | 420E11 | Systems I | 1 | 0 |
| SP03 | CIS253 | Database Systems | 3 | 4 |
| SP03 | CIS265 | Systems Analysis | 2 | 4 |
| SP03 | LA123 | English Literature | 1 | 3 |
| WN03 | AC101 | Accounting | 1 | 2 |
| WN03 | CIS253 | Database Systems | 1 | 2 |
| WN03 | CIS265 | Systems Analysis | 1 | 2 |
| WN03 | MA150 | College Algebra | 1 | 1 |
|  | EN100 | Basic English | 0 | 0 |
|  |  |  |  |  |

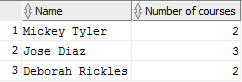
## List the names of all students who have completed more than 1 course. A student is a considered to have completed a course if the final mark is 'A', 'B', 'C', 'D' or 'F'.

SELECT s.first||' '||s.last AS "Name", COUNT(r.studentid) AS "Number of courses"

FROM iu\_student s, iu\_registration r

WHERE r.final IS NOT NULL AND s.studentid = r.studentid

GROUP BY s.first||' '||s.last;



A sample run is shown here.

Student Number of Courses

------------------------------ ----------------------

Jose Diaz 3

Mickey Tyler 2

2 rows selected