**A:12**

1. Insert a new instructor: Mr. John Locke with an ID of 815. His address is 2342 Oceanic Way, Bayonne, NJ, 07002. He doesn't have a phone number.

**INSERT INTO instructor (INSTRUCTOR\_ID, SALUTATION, FIRST\_NAME, LAST\_NAME,**

**STREET\_ADDRESS, ZIP, CREATED\_BY, CREATED\_DATE,**

**MODIFIED\_BY, MODIFIED\_DATE)**

**VALUES (815, 'Mr.', 'John', 'Locke', '2342 Oceanic Way, Bayonne, NJ',**

**07002, 'Ryan Buchanan', SYSDATE, 'Ryan Buchanan', SYSDATE);**

1. Create a new section ID of 48 for Mr. Locke. He'll be teaching section 4 of the Project Management Course (142) beginning on September 22, 2011 at 8:15am in Room L211. The capacity is 15 students.

**INSERT INTO section (SECTION\_ID, COURSE\_NO, SECTION\_NO, START\_DATE\_TIME,**

**LOCATION, INSTRUCTOR\_ID, CAPACITY, CREATED\_BY, CREATED\_DATE, MODIFIED\_BY, MODIFIED\_DATE)**

**VALUES (48, 142, 4, TO\_DATE('2011/09/22 09:15:00', 'yyyy/mm/dd hh24:mi:ss'), 'L211',**

**815, 15, 'Ryan Buchanan', SYSDATE, 'Ryan Buchanan', SYSDATE);**

1. Enroll students 375, 137, 266 and 382 in the course. Their enrollment date should be the current date. (Use ONE INSERT statement).

**INSERT ALL**

**INTO ENROLLMENT (STUDENT\_ID, SECTION\_ID, ENROLL\_DATE,**

**CREATED\_BY, CREATED\_DATE, MODIFIED\_BY, MODIFIED\_DATE)**

**VALUES (375, 48, SYSDATE, 'Ryan Buchanan', SYSDATE, 'Ryan Buchanan', SYSDATE)**

**INTO ENROLLMENT (STUDENT\_ID, SECTION\_ID, ENROLL\_DATE,**

**CREATED\_BY, CREATED\_DATE, MODIFIED\_BY, MODIFIED\_DATE)**

**VALUES (137, 48, SYSDATE, 'Ryan Buchanan', SYSDATE, 'Ryan Buchanan', SYSDATE)**

**INTO ENROLLMENT (STUDENT\_ID, SECTION\_ID, ENROLL\_DATE,**

**CREATED\_BY, CREATED\_DATE, MODIFIED\_BY, MODIFIED\_DATE)**

**VALUES (266, 48, SYSDATE, 'Ryan Buchanan', SYSDATE, 'Ryan Buchanan', SYSDATE)**

**INTO ENROLLMENT (STUDENT\_ID, SECTION\_ID, ENROLL\_DATE,**

**CREATED\_BY, CREATED\_DATE, MODIFIED\_BY, MODIFIED\_DATE)**

**VALUES (382, 48, SYSDATE, 'Ryan Buchanan', SYSDATE, 'Ryan Buchanan', SYSDATE)**

**SELECT \* FROM DUAL;**

1. Remove the enrollment for student 147 from section 120.  
   Note the error. You cannot delete the enrollment record because there are foreign keys in the Grade table that depend on the Enrollment table records, thus creating an integrity constraint violation.  
   First remove all grades for student 147, section 120 and then remove remove the enrollment for student 147, section 120.  (Use two DELETE statements).

**DELETE FROM GRADE**

**WHERE STUDENT\_ID = 147**

**AND SECTION\_ID = 120;**

**DELETE FROM ENROLLMENT**

**WHERE STUDENT\_ID = 147**

**AND SECTION\_ID = 120;**

1. Use the same procedure to remove the enrollment record for student 180 in section 119.

**DELETE FROM GRADE**

**WHERE STUDENT\_ID = 180**

**AND SECTION\_ID = 119;**

**DELETE FROM ENROLLMENT**

**WHERE STUDENT\_ID = 180**

**AND SECTION\_ID = 119;**

1. Change Mr. Locke's (ID 815) phone number to be 4815162342.

**UPDATE INSTRUCTOR**

**SET PHONE = '4815162342'**

**WHERE INSTRUCTOR\_ID = 815;**

1. Change the grade on the first homework (HM) assignment for each student in section 119 to 100.

**UPDATE GRADE**

**SET NUMERIC\_GRADE = 100**

**WHERE GRADE\_TYPE\_CODE = 'HM'**

**AND GRADE\_CODE\_OCCURRENCE = 1**

**AND SECTION\_ID = 119;**

1. Increase the final exam (FI) score for each student in section 119 by 10 percent.

**UPDATE GRADE**

**SET NUMERIC\_GRADE = NUMERIC\_GRADE \* 1.1**

**WHERE GRADE\_TYPE\_CODE = 'FI'**

**AND SECTION\_ID = 119;**

1. For each section of the Project Management course, list the section ID, location and number of students enrolled. Sort by section ID.

**SELECT s.section\_id, s.location, COUNT(e.student\_id)**

**FROM SECTION s**

**JOIN enrollment e**

**ON s.section\_id = e.section\_id**

**WHERE s.course\_no IN**

**(SELECT course\_no**

**FROM course**

**WHERE description LIKE '%Project Management%')**

**GROUP BY s.section\_id, s.location**

**ORDER BY s.section\_id;**

1. Provide an alphabetic list of instructor names and phone numbers for instructors that teach the Project Management course.

**SELECT i.last\_name, i.first\_name, i.phone**

**FROM instructor i**

**JOIN section s**

**ON i.instructor\_id = s.instructor\_id**

**WHERE s.section\_id IN**

**(SELECT s.section\_id**

**FROM SECTION s**

**JOIN course c**

**ON s.course\_no = c.course\_no**

**WHERE c.description LIKE '%Project Management%')**

**ORDER BY i.last\_name;**

1. Provide a list of students in section 119 along with their average numeric grade. Include the student id, student name and average grade. Format the average to two places.  Sort by student ID.

**SELECT s.student\_id, s.last\_name, s.first\_name,**

**ROUND(AVG(g.numeric\_grade), 2) AS "GPA"**

**FROM student s, enrollment e, grade g**

**WHERE s.student\_id = e.student\_id**

**AND e.section\_id = g.section\_id**

**AND e.section\_id = 119**

**GROUP BY s.student\_id, s.last\_name, s.first\_name**

**ORDER BY s.student\_id;**

1. Create a query to determine the number of instructors who have taught more than 3 students in a course section taught in Room L211.

**SELECT s.instructor\_id**

**FROM SECTION s**

**JOIN enrollment e**

**ON s.section\_id = e.section\_id**

**WHERE s.location = '211'**

**HAVING COUNT(e.student\_id) >= 1**

**GROUP BY s.instructor\_id, e.section\_id**

**ORDER BY s.instructor\_id;**

1. Create a list of instructors that have taught only the Project Management course (142). List the instructor ID, instructor name (salutation, first name, last name) and phone number.

**SELECT instructor\_id,**

**CONCAT(CONCAT(CONCAT(CONCAT(salutation, ' '), first\_name), ' '), last\_name)**

**AS "Instructor", phone**

**FROM instructor**

**WHERE instructor\_id IN**

**(SELECT s.instructor\_id**

**FROM section s**

**JOIN course c**

**ON s.course\_no = c.course\_no**

**WHERE c.description LIKE '%Project Management%'**

**MINUS**

**SELECT s.instructor\_id**

**FROM section s**

**JOIN course c**

**ON s.course\_no = c.course\_no**

**WHERE c.description NOT LIKE '%Project Management%')**

**ORDER BY last\_name;**

1. List student names, section id and course number for students that have not yet received any grades in any section.

**SELECT s.last\_name, s.first\_name, c.section\_id, c.course\_no**

**FROM student s, section c**

**WHERE s.student\_id = c.section\_id**

**AND NOT EXISTS**

**(SELECT '\*'**

**FROM grade g**

**WHERE g.student\_id = s.student\_id)**

**GROUP BY s.last\_name, s.first\_name, c.section\_id, c.course\_no;**

1. List each unique section start time and the number of courses (not sections) that have been taught at that time.  Sort by start time.

**SELECT section\_no, TO\_CHAR(start\_date\_time, 'MON/DD/YYYY HH:MI') AS "Start Time",**

**COUNT(course\_no)**

**FROM section**

**GROUP BY section\_no, start\_date\_time**

**ORDER BY section\_no, start\_date\_time;**