COMP23111 2018-2019
EX03-10136960
Tejas Chandrasekar

```
SQL> -- sends everything to <spoolfilename>
SQL> ------------------------------------Part 1a-----------------------------------
SQL> -- i) All unique names of students who have taken >= 1 CS course
SQL> SELECT DISTINCT name FROM student
  2  WHERE student.ID IN (SELECT DISTINCT ID FROM takes
  3                                  WHERE takes.course_id LIKE 'CS%');

NAME
--------------------
Zhang
Brown
Bourikas
Shankar
Levy
Williams

6 rows selected.

SQL>
SQL> -- ii) all names and IDs of students who haven't taken course before < 2009
SQL> SELECT ID, name FROM student
  2  WHERE student.ID IN (SELECT DISTINCT ID FROM takes
  3                                  WHERE takes.year > 2009);

ID        NAME
----- --------------------
12345 Shankar
19991 Brandt
23121 Chavez
45678 Levy
55739 Sanchez
76543 Brown
98765 Bourikas
98988 Tanaka

8 rows selected.

SQL>
SQL> -- iii) Maximum salaries of instructors in each department
SQL> SELECT MAX(salary), instructor.dept_name
```

```
  2  FROM instructor
  3  GROUP BY dept_name;

MAX(SALARY) DEPT_NAME
----------- --------------------
          80000 Elec. Eng.
          95000 Physics
          92000 Comp. Sci.
          90000 Finance
          72000 Biology
          40000 Music
          62000 History

7 rows selected.

SQL>
SQL> -- iv) Smallest of part iii)
SQL> SELECT MIN(salary), instructor.dept_name
  2  FROM instructor
  3  GROUP BY dept_name
  4  HAVING MIN(salary) = (SELECT MIN(MAX(salary))
  5                        FROM instructor
  6                        GROUP BY dept_name);

MIN(SALARY) DEPT_NAME
----------- --------------------
          40000 Music

SQL> ------------------------------------Part 1b-----------------------------------
SQL> -- i) new CS-001 course in compsci, Weekly Seminar, 10 credits
SQL> INSERT INTO course VALUES ('CS-001', 'Weekly Seminar', 'Comp. Sci.', 10);

1 row created.

SQL> -- ii) new CS-002 course in compsci, monthly seminar, 0 credits
SQL> -- INSERT INTO course VALUES ('CS-002', 'Weekly Seminar', 'Comp. Sci.', 0);
SQL> -- ERROR at line 1:
SQL> -- ORA-02290: check constraint (MBAXATC4.SYS_C001299534) violated
SQL> -- iii) not possible to have a course with 0 credits since 'check (credits > 0)'
SQL> -- exists in the create table definition
SQL> -- iv) not enough columns
SQL> INSERT INTO section(course_id, sec_id, semester, YEAR)
  2  VALUES ('CS-001', '1', 'Fall', 2009);

1 row created.
```

```
SQL> -- v) missing surname of student of course, student id, and grade
SQL> -- must parameterize it by selecting specific columns
SQL> -- vi) enrol every student in CS department in above new section
SQL> --following attributes of takes are part of primary key
SQL> INSERT INTO takes(ID, course_id, sec_id, semester, YEAR)
  2  SELECT ID, 'CS-001', '1', 'Fall', 2009 FROM student
  3  WHERE dept_name = 'Comp. Sci.';

4 rows created.

SQL> -- vii) Deleting everything from above course where student's name is Zhang
SQL> DELETE FROM takes
  2  WHERE course_id = 'CS-001' AND sec_id = '1' AND semester = 'Fall' AND YEAR =
2009
  3  AND ID IN (SELECT ID FROM student WHERE student.name = 'Zhang');

1 row deleted.

SQL> -- viii) Delete all takes tuples which are course with database as a title
SQL> DELETE FROM takes
  2  WHERE course_id IN (SELECT course_id
  3                      FROM course
  4                      WHERE lower(course.title) LIKE '%database%');

2 rows deleted.

SQL> -- ix) Delete course CS-001
SQL> DELETE FROM takes
  2  WHERE course_id = 'CS-001';

3 rows deleted.

SQL> -- x) course_id is a valid attribute therefore the above statement does as intended
SQL> ----------------------------------Part 2a----------------------------------
SQL> -- i) Find number of acciedents in which Jane Rowling's cars were involved
SQL> SELECT COUNT(driver_id)
  2  FROM participated
  3  WHERE license IN (SELECT license
  4                    FROM owns
  5                    WHERE driver_id IN (SELECT driver_id
  6                                        FROM person
  7                                        WHERE person.name = 'Jane Rowling'));

COUNT(DRIVER_ID)
----------------
               3
```

```
SQL> -- ii) Update the amount of damage for car with license number KUY 629
SQL> UPDATE participated
  2  SET damage_amount = 2500
  3  WHERE report_number = 7897423 AND license = 'KUY 629';

1 row updated.

SQL> -- iii) list name of people who participated in accidents along with total
SQL> -- damage caused, but only include those whose total damage > 3000
SQL> SELECT name, SUM(damage_amount)
  2  FROM person NATURAL JOIN participated
  3  GROUP BY person.name
  4  HAVING SUM(damage_amount) > 3000
  5  ORDER BY SUM(damage_amount) DESC;

NAME            SUM(DAMAGE_AMOUNT)
--------------------- ------------------
William Hardy                   4500
Jane Rowling                    4500
Kelly Woolf                     4000

SQL> -- iv) Create view returning locations where accidents happened along with avg
damage
SQL> CREATE VIEW average_damage_per_location AS
  2  SELECT location, AVG(damage_amount) AS avgdmg
  3  FROM accident NATURAL JOIN participated
  4  GROUP BY location;

View created.

SQL> -- accessible with
SQL> -- v) use average_damage_per_location to find location with highest avg damage
SQL> SELECT location
  2  FROM average_damage_per_location
  3  WHERE avgdmg = (SELECT MAX(avgdmg)
  4                  FROM average_damage_per_location);

LOCATION
--------------------
Stockport

SQL>
SQL> SPOOL OFF
```