9.6 Homework: SQL and Databases

H. Diana McSpadden (hdm5s)

```
Reference Databases
classroom(building, room_number, capacity)
department(dept_name, building, budget)
course(course_id, title, dept_name, credits)
instructor(ID, name, dept_name, salary)
section(course_id, sec_id, semester, year, building, room_number, time_slot_id)
teaches(ID, course_id, sec_id, semester, year)
student(ID, name, dept_name, tot_cred)
takes(ID, course_id, sec_id, semester, year, grade)
advisor(s_ID, i_ID) time slot(time_slot_id, day, start_time, end_time) prereq(course_id,
prereq_id)
```

Question 11

Part a: Find the ID and name of each student who has taken at least one Comp. Sci course; make sure there are no duplicate names in the result.

SELECT DISTINCT(student.name)

FROM student

JOIN takes ON student.ID = takes.ID

JOIN course on takes.course_id = course.course_id

WHERE course.dept_name = 'Comp. Sci.'

ORDER BY student.name DESC

Part b: Find the ID and name of each student who has not taken any course offered before 2017.

SELECT student.ID, student.name

FROM student

WHERE NOT EXISTS(

SELECT 1

FROM

takes

WHERE takes.year < 2018 AND takes.ID = student.ID)

Part c: For each department, find the max salary of the instructors in that department. You may assume that every department has at least one instructor.

SELECT dept_name, MAX(salary)

FROM instructor

GROUP BY dept_name

Part d: Find the lowest, across all departments, of the per-department maximum salary computed by the preceding query.

```
SELECT MIN(salary), dept_name FROM

(SELECT dept_name, MAX(salary) AS salary

FROM instructor

GROUP BY dept_name)
```

Question 12

Part a: Create a new course “CS-001” titled “Weekly Seminar” with 0 credits.

```
INSERT INTO course (course_id, title, dept_name, credits)

VALUES ('CS-001', 'Weekly Seminar', 'Comp. Sci.',  0)
```

Part b: Create a section of this course in Fall 2017, with sec_id = 1, and with the location of the section not yet specified.

```
INSERT INTO section (course_id, sec_id, semester, year)

VALUES ('CS-001', 1, 'Fall', 2017)
```

Part c: Enroll every student in the Comp. Sci department in the above section.

```
INSERT INTO takes (ID, course_id, sec_id, semester, year, grade)

SELECT student.ID, 'CS-001', 1, 'Fall', 2017, ''

FROM student

WHERE student.dept_name = 'Comp. Sci.'
```

Part d: Delete enrollments in the above section where the student’s ID is 12345.

```
DELETE FROM takes

WHERE ID = 12345 AND course_id = 'CS-001' AND sec_id = 1 AND semester = 'Fall' AND year = 2017
```

Part f: Delete all *takes* tuples corresponding to any section of any course with the word “advanced” as a part of the title; ignore case when matching the word with the title.

```
DELETE FROM takes

WHERE EXISTS

 ( SELECT *

   FROM takes

   JOIN course ON takes.course_id = course.course_id

   WHERE course.title LIKE '%advanced%' )
```

Reference: https://www.techonthenet.com/sqlite/delete.php

Question 13

Write SQL DDL corresponding to the schema in Figure 3.17. Make any reasonable assumptions about data types, and be sure to declare primary and foreign keys.

```
# first DROP existing tables to recreate

# warning, this will delete any data in these tables

DROP TABLE IF EXISTS person

DROP TABLE IF EXISTS car

DROP TABLE IF EXISTS accident

DROP TABLE IF EXISTS owns

DROP TABLE IF EXISTS participated

# create person table

CREATE TABLE IF NOT EXISTS person

(driver_id integer NOT NULL PRIMARY KEY AUTOINCREMENT NOT NULL,

name text NOT NULL, address text)

# create car table

CREATE TABLE IF NOT EXISTS car

(license_plate text NOT NULL PRIMARY KEY

CHECK(

    typeof("license_plate") = "text" AND

    length("license_plate") > 0 AND

    length("license_plate") <= 20

  ),

model text NOT NULL,

year integer NOT NULL

CHECK (year > 1930 AND))

# create accident table

CREATE TABLE IF NOT EXISTS accident

(report_number integer NOT NULL PRIMARY KEY AUTOINCREMENT NOT NULL,
year integer NOT NULL
CHECK (year > 2000),
location text NOT NULL DEFAULT 'Unknown')
# create owns table
CREATE TABLE IF NOT EXISTS owns
(driver_id integer NOT NULL,
license_plate text NOT NULL,
PRIMARY KEY(driver_id,license_plate))
# create participated table
CREATE TABLE IF NOT EXISTS participated
(report_number integer NOT NULL,
license_plate text NOT NULL,
driver_id integer NOT NULL,
damage_amount real NOT NULL DEFAULT 0,
FOREIGN KEY(driver_id) REFERENCES person(driver_id),
PRIMARY KEY(report_number,license_plate))
```