# **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))