Task 2: Write the commands for creating tables and inserting values

```
CREATE TABLE courses (
 course_id int NOT NULL,
 department varchar(30) NOT NULL,
 course_num int NOT NULL,
 course_name varchar(35) NOT NULL,
 semester varchar(30) NOT NULL,
 year int NOT NULL,
 PRIMARY KEY (course_id)
);

CREATE TABLE students (
 student_id int NOT NULL,
 lname varchar(30) NOT NULL,
 fname varchar(30) NOT NULL,
 classification varchar(30) NOT NULL,
 major varchar(30) NOT NULL,
 PRIMARY KEY (student_id)
);

CREATE TABLE enroll (
 student_id int NOT NULL,
 course_id int NOT NULL,
 PRIMARY KEY (student_id, course_id),
 FOREIGN KEY (student_id) REFERENCES students(student_id),
 FOREIGN KEY (course_id) REFERENCES courses(course_id)
);

CREATE TABLE distribution (
 distribution_id int NOT NULL,
 course_id int NOT NULL ,
 category varchar(30) NOT NULL,
 percentage int NOT NULL,
 PRIMARY KEY (distribution_id),
 FOREIGN KEY (course_id) REFERENCES courses(course_id)
);
```

```
CREATE TABLE assignments (
 assignment_id int NOT NULL,
 distribution_id int NOT NULL,
 max_points int NOT NULL,
 number int not NULL,
 PRIMARY KEY (assignment_id),
 FOREIGN KEY (distribution_id) REFERENCES distribution(distribution_id)
);

CREATE TABLE scores (
 student_id int NOT NULL,
 assignment_id int NOT NULL,
 points int NOT NULL,
 PRIMARY KEY (student_id, assignment_id),
 FOREIGN KEY (student_id) REFERENCES students(student_id),
 FOREIGN KEY (assignment_id) REFERENCES assignments(assignment_id)
);

/*commands for inserting values*/
INSERT INTO courses VALUES
 (01, 'Computer Science', 432, 'Database Systems', 'Spring', 2021),
 (02, 'Computer Science', 383, 'Affective Computing', 'Spring', 2021),
 (03, 'Computer Science', 375, 'Software Engineering', 'Spring', 2021);

INSERT INTO students VALUES
 (4321, 'Belcher', 'Camryn', 'junior', 'Computer Science'),
 (5678, 'Ivey', 'Aaron', 'senior', 'Rocket Science'),
 (1234, 'Barnes', 'Kora', 'sophomore', 'Creative Writing'),
 (8765, 'Qierrot', 'Handlie', 'freshman', 'Crimonology');

INSERT INTO enroll VALUES
 (4321, 01),
 (5678, 02),
 (1234, 01),
 (8765, 03);
```

```
INSERT INTO distribution VALUES
 (01, 01, 'participation',10),
 (02, 01, 'homework', 20),
 (03, 01, 'tests', 50),
 (04, 01, 'projects', 20),
 (05, 02, 'participation',20),
 (06, 02, 'homework', 20),
 (07, 02, 'tests', 40),
 (08, 02, 'projects', 20),
 (09, 03, 'participation',20),
 (10, 03, 'homework', 10),
 (11, 03, 'tests', 40),
 (12, 03, 'projects', 30);

INSERT INTO assignments VALUES
(01, 01, 100, 1),
(02, 02, 50, 1),
(03, 02, 50, 2),
(04, 03, 100, 1),
(05, 04, 100, 1),
(06, 05, 100, 1),
(07, 06, 50, 1),
(08, 06, 50, 2),
(09, 07, 100, 1),
(10, 08, 100, 1),
(11, 09, 100, 1),
(12, 10, 50, 1),
(13, 10, 50, 2),
(14, 11, 100, 1),
(15, 12, 100, 1);

INSERT INTO scores VALUES
 (4321, 01, 100),
 (4321, 02, 95),
 (4321, 03, 67),
```

```
(4321, 04, 90),
(4321, 05, 98),
(4321, 06, 95),
(4321, 07, 91),
(4321, 08, 96),
(4321, 09, 95),
(4321, 10, 85),
(4321, 11, 74),
(4321, 12, 97),
(4321, 13, 68),
(4321, 14, 71),
(4321, 15, 83),
(5678, 01, 100),
(5678, 02, 50),
(5678, 03, 74),
(5678, 04, 62),
(5678, 05, 86),
(5678, 06, 99),
(5678, 07, 100),
(5678, 08, 43),
(5678, 09, 78),
(5678, 10, 87),
(5678, 11, 33),
(5678, 12, 92),
(5678, 13, 84),
(5678, 14, 12),
(5678, 15, 21),
(1234, 01, 28),
(1234, 02, 33),
(1234, 03, 43),
(1234, 04, 76),
(1234, 05, 100),
(1234, 06, 91),
(1234, 07, 94),
(1234, 08, 82),
(1234, 09, 88),
```

```
 (1234, 10, 79),
 (1234, 11, 65),
 (1234, 12, 57),
 (1234, 13, 80),
 (1234, 14, 100),
 (1234, 15, 100),
 (8765, 01, 10),
 (8765, 02, 78),
 (8765, 03, 100),
 (8765, 04, 93),
 (8765, 05, 48),
 (8765, 06, 55),
 (8765, 07, 67),
 (8765, 08, 88),
 (8765, 09, 91),
 (8765, 10, 100),
 (8765, 11, 100),
 (8765, 12, 27),
 (8765, 13, 39),
 (8765, 14, 89),
 (8765, 15, 98);
```

Task 3: Show the tables with the contents you have inserted

```
SELECT * FROM courses;
SELECT * FROM students;
SELECT * FROM enroll;
SELECT * FROM distribution;
SELECT * FROM assignments;
SELECT * FROM scores;
```

'Task 4: Compute the average/highest/lowest score of an assignment

```
SELECT avg(points)
FROM scores
WHERE assignment_id = 10;

SELECT max(points)
```

```
FROM scores
WHERE assignment_id = 10;

SELECT min(points)
FROM scores
WHERE assignment_id = 10;

'Task 5: List all the students in a given course
SELECT s.student_id, s.fname, s.lname
FROM students s
WHERE s.student_id in (SELECT e.student_id FROM enroll e WHERE e.course_id
= (SELECT c.course_id FROM courses c WHERE c.course_id = 01)
);

/*'Task 6: List all of the students in a course and all of their scores on
every assignment'*/
SELECT s.student_id, s.fname, s.lname, e.student_id, p.assignment_id,
p.points
FROM students s, enroll e, scores p
WHERE s.student_id = p.student_id AND p.student_id = e.student_id AND
e.course_id = 02;

/*'Task 7: Add an assignment to a course'*/
INSERT INTO assignments VALUES
 (16, 03, 100, 2);

SELECT * FROM assignments;

/*'Task 8: Change the percentage of the categories for a course'*/
UPDATE distribution
SET percentage = 25
WHERE course_id = 03;

SELECT *
FROM distribution
WHERE course_id = 03;
```

```
/*'Task 9: Add 2 points to the score of each student on an assignment'*/
UPDATE scores
SET points = points + 2
WHERE assignment_id = 12;

SELECT *
FROM scores
WHERE assignment_id = 12;

/*'Task 10: Compute the grade for a student*/
UPDATE scores
SET points = points + 2
WHERE scores.student_id = (SELECT student_id FROM students WHERE
scores.student_id = students.student_id AND students.lname LIKE '%Q%');

/*TASK 11 - Compute the grade for a student*/
SELECT SUM(s.points * (d.percentage/c.COUNTER)/a.max_points) AS Grade
FROM scores s
JOIN assignments a on s.assignment_id = a.assignment_id
JOIN distribution d on d.distribution_id = a.distribution_id
JOIN (SELECT d.distribution_id, COUNT(*) AS COUNTER FROM scores s
JOIN assignments a on s.assignment_id = a.assignment_id
JOIN distribution d on d.distribution_id = a.distribution_id
WHERE student_id = 5678 AND course_id = 02 GROUP BY d.distribution_id
) c ON c.distribution_id = d.distribution_id
WHERE student_id = 5678 AND course_id = 02;


/*TASK 12 - Compute the grade for a student, where the lowest score for a
given category is dropped*/
DELETE FROM scores
WHERE scores.student_id = 5678 AND scores.assignment_id = 14;

SELECT SUM(s.points * (d.percentage/c.COUNTER)/a.max_points) AS Grade2
FROM scores s
```

```
JOIN assignments a on s.assignment_id = a.assignment_id
JOIN distribution d on d.distribution_id = a.distribution_id
JOIN (SELECT d.distribution_id, COUNT(*) AS COUNTER FROM scores s
JOIN assignments a on s.assignment_id = a.assignment_id
JOIN distribution d on d.distribution_id = a.distribution_id
WHERE student_id = 5678 AND course_id = 02 GROUP BY d.distribution_id
) c ON c.distribution_id = d.distribution_id
WHERE student_id = 5678 AND course_id = 02;
```