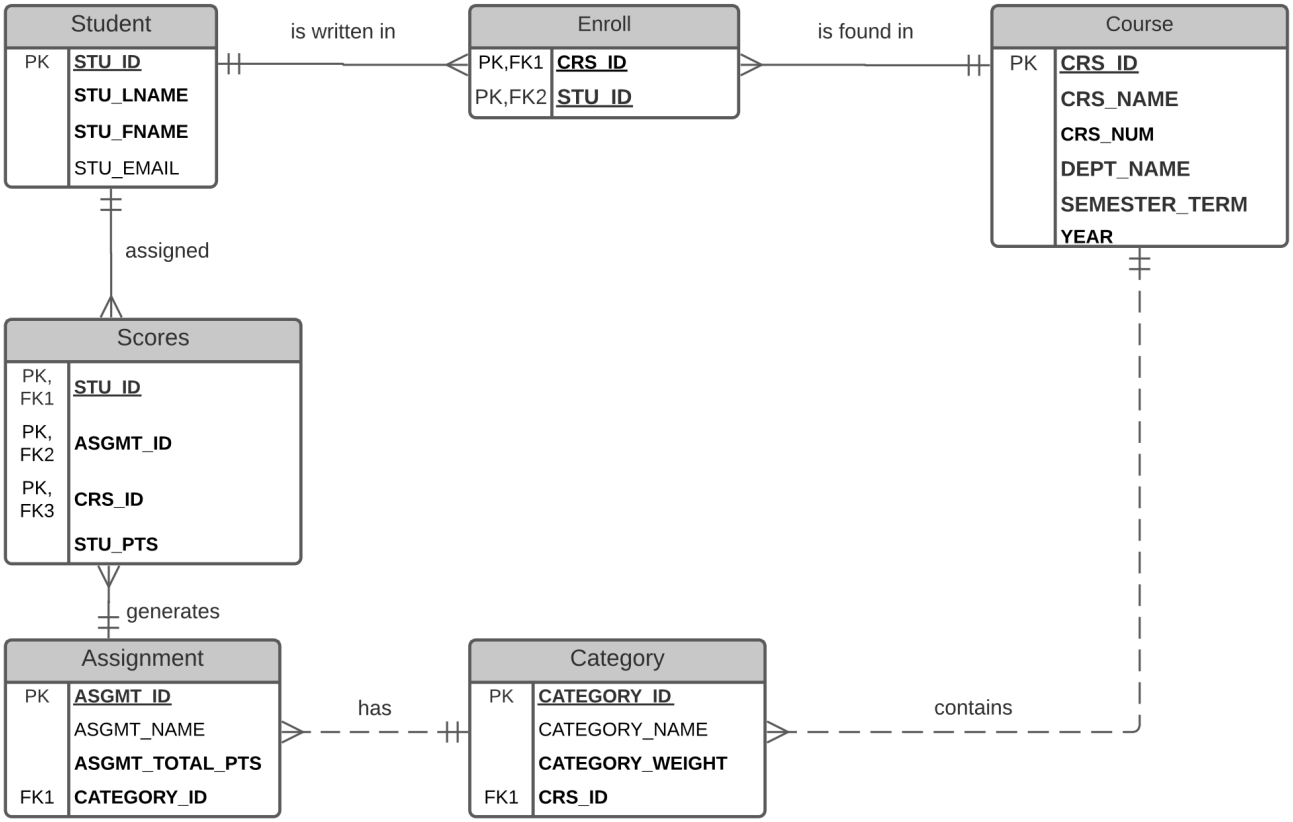
Asis Johnson
Tolulope Oluyadi
Liliann Ulysse

Database Project Documentation
Group 15

Tasks

1. Design the ER diagram;



2. Write the commands for creating tables and inserting values;

```
CREATE DATABASE IF NOT EXISTS `gradebook` /*!40100 DEFAULT
CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci */
/*!80016 DEFAULT ENCRYPTION='N' */;
USE `gradebook`;

CREATE TABLE `Course` (
  `CRS_ID` int NOT NULL,
  `CRS_NAME` varchar(45) NOT NULL,
```

```
  `CRS_NUM` int NOT NULL,
  `DEPT_NAME` varchar(45) NOT NULL,
  `SEMESTER_TERM` varchar(45) NOT NULL,
  `YEAR` int NOT NULL,
  PRIMARY KEY (`CRS_ID`),
  UNIQUE KEY `CRS_ID_UNIQUE` (`CRS_ID`),
  UNIQUE KEY `CRS_NUM_UNIQUE` (`CRS_NUM`)
);

CREATE TABLE `Student` (
  `STU_ID` int NOT NULL AUTO_INCREMENT,
  `STU_LNAME` varchar(45) NOT NULL,
  `STU_FNAME` varchar(45) NOT NULL,
  `STU_EMAIL` varchar(45) NOT NULL,
  PRIMARY KEY (`STU_ID`),
  UNIQUE KEY `STU_ID_UNIQUE` (`STU_ID`),
  UNIQUE KEY `STU_EMAIL_UNIQUE` (`STU_EMAIL`)
);

CREATE TABLE `Enroll` (
  `CRS_ID` int NOT NULL,
  `STU_ID` int NOT NULL,
  PRIMARY KEY (`CRS_ID`, `STU_ID`),
  CONSTRAINT `FK_Enroll_Course` FOREIGN KEY (`CRS_ID`)
REFERENCES `Course` (`CRS_ID`),
  CONSTRAINT `FK_Enroll_Student` FOREIGN KEY (`STU_ID`)
REFERENCES `Student` (`STU_ID`)
);

CREATE TABLE `Category` (
  `CATEGORY_ID` int NOT NULL AUTO_INCREMENT,
  `CATEGORY_NAME` varchar(45) NOT NULL,
  `CATEGORY_WEIGHT` int NOT NULL,
  `CRS_ID` int NOT NULL,
  PRIMARY KEY (`CATEGORY_ID`),
  UNIQUE KEY `CATEGORY_ID_UNIQUE` (`CATEGORY_ID`),
  CONSTRAINT `FK_Category_Course` FOREIGN KEY (`CRS_ID`)
REFERENCES `Course` (`CRS_ID`)
);

CREATE TABLE `Assignment` (
```

```
  `ASGMT_ID` int NOT NULL AUTO_INCREMENT,
  `ASGMT_NAME` varchar(45) DEFAULT NULL,
  `ASGMT_TOTAL_PTS` int NOT NULL,
  `CATEGORY_ID` int NOT NULL,
  PRIMARY KEY (`ASGMT_ID`),
  UNIQUE KEY `ASGMT_ID_UNIQUE` (`ASGMT_ID`),
  CONSTRAINT `FK_Assignment_Category` FOREIGN KEY
(`CATEGORY_ID`) REFERENCES `Category` (`CATEGORY_ID`)
);

CREATE TABLE `Scores` (
  `CRS_ID` int NOT NULL,
  `ASGMT_ID` int NOT NULL,
  `STU_ID` int NOT NULL,
  `STU_PTS` int NOT NULL,
  PRIMARY KEY (`CRS_ID`, `ASGMT_ID`, `STU_ID`),
  CONSTRAINT `FK_Scores_Course` FOREIGN KEY (`CRS_ID`)
REFERENCES `Course` (`CRS_ID`)
);

-- Insert Statements
INSERT INTO `Course` VALUES (123, 'Database Systems', 401,
'Computer Science', 'Spring', 2021);
INSERT INTO `Course` VALUES (456, 'Machine Learning', 402,
'Computer Science', 'Fall', 2021);

-- Changing student names and emails
INSERT INTO `Student` (`STU_LNAME`, `STU_FNAME`,
`STU_EMAIL`) VALUES ('Alvarado', 'Stephanie',
'sa@uni.edu');
INSERT INTO `Student` (`STU_LNAME`, `STU_FNAME`,
`STU_EMAIL`) VALUES ('Miller', 'Luther', 'lm@uni.edu');
INSERT INTO `Student` (`STU_LNAME`, `STU_FNAME`,
`STU_EMAIL`) VALUES ('Bell', 'Dana', 'db@uni.edu');
INSERT INTO `Student` (`STU_LNAME`, `STU_FNAME`,
`STU_EMAIL`) VALUES ('Queen', 'Stephen', 'sq@uni.edu');
INSERT INTO `Student` (`STU_LNAME`, `STU_FNAME`,
`STU_EMAIL`) VALUES ('Lambert', 'Nina', 'nl@uni.edu');
INSERT INTO `Student` (`STU_LNAME`, `STU_FNAME`,
`STU_EMAIL`) VALUES ('Washington', 'Dixie', 'dw@uni.edu');
```

```
INSERT INTO `Student` (`STU_LNAME`, `STU_FNAME`,
`STU_EMAIL`) VALUES ('Sapinoso', 'Eric', 'es@uni.edu');
INSERT INTO `Student` (`STU_LNAME`, `STU_FNAME`,
`STU_EMAIL`) VALUES ('Alvarado', 'Stephanie',
'sa@bison.howard.edu');
INSERT INTO `Student` (`STU_LNAME`, `STU_FNAME`,
`STU_EMAIL`) VALUES ('Miller', 'Luther',
'lm@bison.howard.edu');
INSERT INTO `Student` (`STU_LNAME`, `STU_FNAME`,
`STU_EMAIL`) VALUES ('Bell', 'Dana',
'db@bison.howard.edu');
INSERT INTO `Student` (`STU_LNAME`, `STU_FNAME`,
`STU_EMAIL`) VALUES ('Queen', 'Stephen',
'sq@bison.howard.edu');
INSERT INTO `Student` (`STU_LNAME`, `STU_FNAME`,
`STU_EMAIL`) VALUES ('Lambert', 'Nina',
'nl@bison.howard.edu');
INSERT INTO `Student` (`STU_LNAME`, `STU_FNAME`,
`STU_EMAIL`) VALUES ('Washington', 'Dixie',
'dw@bison.howard.edu');
INSERT INTO `Student` (`STU_LNAME`, `STU_FNAME`,
`STU_EMAIL`) VALUES ('Sapinoso', 'Eric',
'es@bison.howard.edu');

-- Enrollment for CRS 456
INSERT INTO `Enroll` VALUES (456, 1);
INSERT INTO `Enroll` VALUES (456, 2);
INSERT INTO `Enroll` VALUES (456, 3);
INSERT INTO `Enroll` VALUES (456, 4);
INSERT INTO `Enroll` VALUES (456, 5);
INSERT INTO `Enroll` VALUES (456, 6);

-- Enrollment for CRS 123
INSERT INTO `Enroll` VALUES (123, 1);
INSERT INTO `Enroll` VALUES (123, 7);

-- Categories for CRS 456
INSERT INTO `Category` (`CATEGORY_NAME`, `CATEGORY_WEIGHT`,
`CRS_ID`) VALUES ('Participation', 10, 456);
INSERT INTO `Category` (`CATEGORY_NAME`, `CATEGORY_WEIGHT`,
`CRS_ID`) VALUES ('Homework', 20, 456);
```

```
INSERT INTO `Category` (`CATEGORY_NAME`, `CATEGORY_WEIGHT`,
`CRS_ID`) VALUES ('Tests', 50, 456);
INSERT INTO `Category` (`CATEGORY_NAME`, `CATEGORY_WEIGHT`,
`CRS_ID`) VALUES ('Projects', 20, 456);

-- Categories for CRS 123
INSERT INTO `Category` (`CATEGORY_NAME`, `CATEGORY_WEIGHT`,
`CRS_ID`) VALUES ('Project', 100, 123);

-- Assignments for CRS 456
INSERT INTO `Assignment` (`ASGMT_NAME`, `ASGMT_TOTAL_PTS`,
`CATEGORY_ID`) VALUES ('HW1', 20, 2);
INSERT INTO `Assignment` (`ASGMT_NAME`, `ASGMT_TOTAL_PTS`,
`CATEGORY_ID`) VALUES ('HW2', 20, 2);
INSERT INTO `Assignment` (`ASGMT_NAME`, `ASGMT_TOTAL_PTS`,
`CATEGORY_ID`) VALUES ('Midterm', 100, 3);
INSERT INTO `Assignment` (`ASGMT_NAME`, `ASGMT_TOTAL_PTS`,
`CATEGORY_ID`) VALUES ('Project 1', 50, 4);
INSERT INTO `Assignment` (`ASGMT_NAME`, `ASGMT_TOTAL_PTS`,
`CATEGORY_ID`) VALUES ('Semester Participation', 50, 1);
INSERT INTO `Assignment` (`ASGMT_NAME`, `ASGMT_TOTAL_PTS`,
`CATEGORY_ID`) VALUES ('Final', 100, 3);

-- Assignments for CRS 123
INSERT INTO `Assignment` (`ASGMT_NAME`, `ASGMT_TOTAL_PTS`,
`CATEGORY_ID`) VALUES ('Final Project', 300, 5);

-- Scores for CRS 456
INSERT INTO `Scores` VALUES (456, 1, 1, 15);
INSERT INTO `Scores` VALUES (456, 2, 1, 20);
INSERT INTO `Scores` VALUES (456, 3, 1, 90);
INSERT INTO `Scores` VALUES (456, 4, 1, 45);
INSERT INTO `Scores` VALUES (456, 5, 1, 50);
INSERT INTO `Scores` VALUES (456, 6, 1, 98);
INSERT INTO `Scores` VALUES (456, 1, 2, 20);
INSERT INTO `Scores` VALUES (456, 2, 2, 17);
INSERT INTO `Scores` VALUES (456, 3, 2, 85);
INSERT INTO `Scores` VALUES (456, 4, 2, 50);
INSERT INTO `Scores` VALUES (456, 5, 2, 45);
INSERT INTO `Scores` VALUES (456, 6, 2, 90);
INSERT INTO `Scores` VALUES (456, 1, 3, 10);
```

```
INSERT INTO `Scores` VALUES (456, 2, 3, 12);
INSERT INTO `Scores` VALUES (456, 3, 3, 70);
INSERT INTO `Scores` VALUES (456, 4, 3, 30);
INSERT INTO `Scores` VALUES (456, 5, 3, 40);
INSERT INTO `Scores` VALUES (456, 6, 3, 75);
INSERT INTO `Scores` VALUES (456, 1, 4, 18);
INSERT INTO `Scores` VALUES (456, 2, 4, 19);
INSERT INTO `Scores` VALUES (456, 3, 4, 100);
INSERT INTO `Scores` VALUES (456, 4, 4, 48);
INSERT INTO `Scores` VALUES (456, 5, 4, 50);
INSERT INTO `Scores` VALUES (456, 6, 4, 95);
INSERT INTO `Scores` VALUES (456, 1, 5, 12);
INSERT INTO `Scores` VALUES (456, 2, 5, 18);
INSERT INTO `Scores` VALUES (456, 3, 5, 85);
INSERT INTO `Scores` VALUES (456, 4, 5, 35);
INSERT INTO `Scores` VALUES (456, 5, 5, 50);
INSERT INTO `Scores` VALUES (456, 6, 5, 60);
INSERT INTO `Scores` VALUES (456, 1, 6, 20);
INSERT INTO `Scores` VALUES (456, 2, 6, 15);
INSERT INTO `Scores` VALUES (456, 3, 6, 80);
INSERT INTO `Scores` VALUES (456, 4, 6, 45);
INSERT INTO `Scores` VALUES (456, 5, 6, 50);
INSERT INTO `Scores` VALUES (456, 6, 6, 100);

-- Scores for CRS 123
INSERT INTO `Scores` VALUES (123, 7, 1, 290);
INSERT INTO `Scores` VALUES(123, 7, 7, 280);
```

3. Show the tables with the contents that you have inserted;

COURSE

CRS_ID	CRS_NAME	CRS_NUM	DEPT_NAME	SEMESTER_TERM	YEAR
123	Database Systems	401	Computer Science	Spring	2021
456	Machine Learning	402	Computer Science	Fall	2021

STUDENT

STU_ID	STU_LNAME	STU_FNAME	STU_EMAIL
1	Alvarado	Stephanie	sa@uni.edu
2	Miller	Luther	lm@uni.edu
3	Bell	Dana	db@uni.edu
4	Queen	Stephen	sq@uni.edu
5	Lambert	Nina	nl@uni.edu
6	Washington	Dixie	dw@uni.edu
7	Sapinoso	Eric	es@uni.edu
8	Alvardo	Stephanie	sa@bison.howard.edu
9	Miller	Luther	lm@bison.howard.edu
10	Bell	Dana	db@bison.howard.edu
11	Queen	Stephen	sq@bison.howard.edu
12	Lambert	Nina	nl@bison.howard.edu
13	Washington	Dixie	dw@bison.howard.edu
14	Sapinoso	Eric	es@bison.howard.edu

ENROLL

CRS_ID	STU_ID
123	1
456	1
456	2
456	3
456	4
456	5
456	6
123	7

CATEGORY

CATEGORY_ID	CATEGORY_NAME	CATEGORY_WEIGHT	CRS_ID
1	Participation	5	456
2	Homework	25	456
3	Tests	45	456
4	Projects	25	456
5	Project	100	123
6	Participation	10	456
7	Homework	20	456
8	Tests	50	456
9	Projects	20	456
10	Project	100	123
11	Participation	10	456
12	Homework	20	456
13	Tests	50	456
14	Projects	20	456
15	Project	100	123

ASSIGNMENT

ASGMT_ID	ASGMT_NAME	ASGMT_TOTAL_PTS	CATEGORY_ID
1	HW1	20	2
2	HW2	20	2
3	Midterm	100	3
4	Project 1	50	4
5	Semester Participation	50	1
6	Final	100	3
7	Final Project	300	5
8	HW3	20	2
9	HW1	20	2
10	HW2	20	2
11	Midterm	100	3
12	Project 1	50	4
13	Semester Participation	50	1
14	Final	100	3
15	Final Project	300	5
16	HW3	20	2
17	HW1	20	2
18	HW2	20	2
19	Midterm	100	3
20	Project 1	50	4
21	Semester Participation	50	1
22	Final	100	3
23	Final Project	300	5

23 rows in set (0.00 sec)

SCORES

CRS_ID	ASGMT_ID	STU_ID	STU_PTS
123	7	1	290
123	7	7	280
456	1	1	15
456	1	2	20
456	1	3	10
456	1	4	22
456	1	5	12
456	1	6	20
456	2	1	20
456	2	2	17
456	2	3	12
456	2	4	23
456	2	5	18
456	2	6	15
456	3	1	90
456	3	2	85
456	3	3	70
456	3	4	104
456	3	5	85
456	3	6	80
456	4	1	45
456	4	2	50
456	4	3	30
456	4	4	52
456	4	5	35
456	4	6	45
456	5	1	50
456	5	2	45
456	5	3	40
456	5	4	54
456	5	5	50
456	5	6	50
456	6	1	102
456	6	2	94
456	6	3	79
456	6	4	103
456	6	5	64
456	6	6	104

38 rows in set (0.00 sec)

4. Compute the average/highest/lowest score of an assignment;

```
SELECT a.ASGMT_ID, AVG(s.STU_PTS) AS AverageScore,
MAX(s.STU_PTS) AS HighestScore, MIN(s.STU_PTS) AS
LowestScore
```

```
FROM Assignment a
JOIN Scores s ON a.ASGMT_ID = s.ASGMT_ID
WHERE s.CRS_ID = 456 AND a.ASGMT_ID = 3
GROUP BY a.ASGMT_ID;
```

Output:

```
+----------+----------------+----------------+----------------+
| ASGMT_ID | AVG(s.STU_PTS) | MAX(s.STU_PTS) | MIN(s.STU_PTS) |
+----------+----------------+----------------+----------------+
|        3 |        85.6667 |            104 |             70 |
+----------+----------------+----------------+----------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

5. List all of the students in a given course;

```
SELECT s.STU_ID, s.STU_FNAME, s.STU_LNAME, s.STU_EMAIL
FROM Student s JOIN Enroll e ON s.STU_ID = e.STU_ID
WHERE e.CRS_ID = 456;
```

Output:

```
+--------+-----------+------------+------------+
| STU_ID | STU_FNAME | STU_LNAME  | STU_EMAIL  |
+--------+-----------+------------+------------+
|      1 | Stephanie | Alvarado   | sa@uni.edu |
|      2 | Luther    | Miller     | lm@uni.edu |
|      3 | Dana      | Bell       | db@uni.edu |
|      4 | Stephen   | Queen      | sq@uni.edu |
|      5 | Nina      | Lambert    | nl@uni.edu |
|      6 | Dixie     | Washington | dw@uni.edu |
+--------+-----------+------------+------------+
6 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

6. List all of the students in a course and all of their scores on every assignment;

```
SELECT s.STU_ID, s.STU_FNAME, s.STU_LNAME, s.STU_EMAIL,
sc.ASGMT_ID, sc.STU_PTS
FROM Student s INNER JOIN Enroll e ON s.STU_ID = e.STU_ID
INNER JOIN Scores sc ON sc.STU_ID = e.STU_ID
WHERE e.CRS_ID = 456;
```

Output:

ASGMT_ID	ASGMT_NAME	ASGMT_TOTAL_PTS	CATEGORY_ID
1	HW1	20	2
2	HW2	20	2
3	Midterm	100	3
4	Project 1	50	4
5	Semester Participation	50	1
6	Final	100	3
7	Final Project	300	5
8	HW3	20	2
9	HW1	20	2
10	HW2	20	2
11	Midterm	100	3
12	Project 1	50	4
13	Semester Participation	50	1
14	Final	100	3
15	Final Project	300	5
16	HW3	20	2
17	HW1	20	2
18	HW2	20	2
19	Midterm	100	3
20	Project 1	50	4
21	Semester Participation	50	1
22	Final	100	3
23	Final Project	300	5
24	HW3	20	2

24 rows in set (0.00 sec)

7. Add an assignment to a course;

```
INSERT INTO `Assignment` (ASGMT_NAME, ASGMT_TOTAL_PTS,
CATEGORY_ID) VALUES ('HW3', 20, 2);

-- Show new Assignment table
SELECT * FROM Assignment;
```

Output:

```
+-------------+---------------+-----------------+--------+
| CATEGORY_ID | CATEGORY_NAME | CATEGORY_WEIGHT | CRS_ID |
+-------------+---------------+-----------------+--------+
|           1 | Participation |               5 |    456 |
|           2 | Homework      |              25 |    456 |
|           3 | Tests         |              45 |    456 |
|           4 | Projects      |              25 |    456 |
|           6 | Participation |              10 |    456 |
|           7 | Homework      |              20 |    456 |
|           8 | Tests         |              50 |    456 |
|           9 | Projects      |              20 |    456 |
|          11 | Participation |              10 |    456 |
|          12 | Homework      |              20 |    456 |
|          13 | Tests         |              50 |    456 |
|          14 | Projects      |              20 |    456 |
+-------------+---------------+-----------------+--------+
12 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

8. Change the percentages of the categories for a course;

```
UPDATE Category
SET
  CATEGORY_WEIGHT = CASE CATEGORY_ID
    WHEN 1 THEN 5
    WHEN 2 THEN 25
    WHEN 3 THEN 45
    WHEN 4 THEN 25
    ELSE CATEGORY_WEIGHT
  END
WHERE
  CRS_ID = 456 AND CATEGORY_ID IN (1, 2, 3, 4);

-- Show new Category table
SELECT * FROM Category c
```

```
WHERE c.CRS_ID = 456;
```

Output:

CATEGORY_ID	CATEGORY_NAME	CATEGORY_WEIGHT	CRS_ID
1	Participation	5	456
2	Homework	25	456
3	Tests	45	456
4	Projects	25	456
6	Participation	10	456
7	Homework	20	456
8	Tests	50	456
9	Projects	20	456
11	Participation	10	456
12	Homework	20	456
13	Tests	50	456
14	Projects	20	456

12 rows in set (0.00 sec)

9. Add 2 points to the score of each student on an assignment;

```
UPDATE Scores sc
INNER JOIN
Student st ON sc.STU_ID = st.STU_ID
SET
sc.STU_PTS = sc.STU_PTS + 2
WHERE
CRS_ID = 456 AND st.STU_LNAME LIKE '%q%';

-- Show new Scores Table
SELECT
CRS_ID, ASGMT_ID, STU_ID, STU_PTS
FROM
Scores
WHERE
CRS_ID = 456 AND ASGMT_ID = 6;
```

Output:

CRS_ID	ASGMT_ID	STU_ID	STU_PTS
456	1	4	24
456	2	4	25
456	3	4	106
456	4	4	54
456	5	4	56
456	6	4	107

6 rows in set (0.00 sec)

10. Add 2 points just to those students whose last name contains a ‘Q’.

```
UPDATE Scores sc
          INNER JOIN
     Student st ON sc.STU_ID = st.STU_ID
SET
     sc.STU_PTS = sc.STU_PTS + 2
WHERE
CRS_ID = 456 AND st.STU_LNAME LIKE '%q%';

-- Show new Scores Table
SELECT
     CRS_ID, ASGMT_ID, STU_ID, STU_PTS
FROM
     Scores
WHERE
     CRS_ID = 456 AND ASGMT_ID = 6;
```

Output:

CRS_ID	ASGMT_ID	STU_ID	STU_PTS
456	6	1	104
456	6	2	96
456	6	3	81
456	6	4	105
456	6	5	66
456	6	6	106

6 rows in set (0.00 sec)

11. Compute the grade for a student;

```
SELECT
     s.CRS_ID,
     s.STU_ID,
     SUM(((s.STU_PTS / a.ASGMT_TOTAL_PTS) * 100) *
     (ca.CATEGORY_WEIGHT)) / SUM(ca.CATEGORY_WEIGHT) AS
     Final_Grade
FROM
     Scores s
          LEFT JOIN
     Assignment a ON s.ASGMT_ID = a.ASGMT_ID
          JOIN
     Category ca ON ca.CATEGORY_ID = a.CATEGORY_ID
WHERE
     s.CRS_ID = 456 AND STU_ID = 1;
```

Output:

```
+--------+--------+--------------+
| CRS_ID | STU_ID | Final_Grade |
+--------+--------+--------------+
|    456 |      1 | 93.26470588 |
+--------+--------+--------------+
1 row in set (0.00 sec)
```

12. Compute the grade for a student, where the lowest score for a given category is dropped.

```
SELECT
     s.CRS_ID,
     s.STU_ID,
     SUM(((s.STU_PTS / a.ASGMT_TOTAL_PTS) * 100) *
(ca.CATEGORY_WEIGHT)) / SUM(ca.CATEGORY_WEIGHT) AS
Dropped_Score_Grade
FROM
     Scores s
          LEFT JOIN
     Assignment a ON s.ASGMT_ID = a.ASGMT_ID
          JOIN
     Category ca ON ca.CATEGORY_ID = a.CATEGORY_ID
WHERE
     s.CRS_ID = 456 AND STU_ID = 1
          AND STU_PTS NOT IN (SELECT
               MIN(s.STU_PTS)
          FROM
               Scores s
                    LEFT JOIN
               Assignment a ON s.ASGMT_ID = a.ASGMT_ID
                    JOIN
               Category ca ON ca.CATEGORY_ID =
          a.CATEGORY_ID
          WHERE
               a.CATEGORY_ID = 3 AND s.CRS_ID = 456
                    AND STU_ID = 1);
```

Output:

```
+--------+--------+---------------------+
| CRS_ID | STU_ID | Dropped_Score_Grade |
+--------+--------+---------------------+
|    456 |      1 |         94.44000000 |
+--------+--------+---------------------+
1 row in set (0.00 sec)
```

Source Code
https://github.com/lil-uly/DBProject

README File
Instructions to Compile and Execute

1. Download and install the MySQL Community Server found https://dev.mysql.com/downloads/mysql/
2. When you installed MySQL, you had the option to create a password for the root user. To connect to MySQL, type the following command into your terminal:
   `/usr/local/mysql/bin/mysql -u root -p`
   and type in the password.

   If that does not work, try connecting from localhost. Type the following command into your terminal:
   `/usr/local/mysql/bin/mysql -u root`
3. Once you are connected to the MySQL Community server, type the source command followed by the pathway to db_project.sql. It should look something like this:
   `source /Users/admin/Documents/DB_project/db_project.sql`

   The tables of my database and the status of the query's should appear in the terminal.

Test Cases

1. Compute the grade for Student 1 in CRS 456
   Expected: 92.21
   Result: 92.0588235
   Status: Passed
2. Compute the grade for Student 1 in CRS 456, where the lowest test score is dropped.
   Expected: 93.00
   Result: 93.0000000
   Status: Passed

3. Add 2 points just to those students whose last name contains a ‘Q’. One of the student’s last name is ‘Queen’
   Expected: Increase Stephen Queen’s (STU_ID = 4) points by 2
   Result: Increased Stephen Queen’s points by 2
   Status: Passed
4. Add 2 points to the score of each student on their CRS 456 Final Exam
   Expected: Increase Final Exam points by 2
   Result: Increased Final Exam points by 2
   Status: Passed
5. Add a new assignment (HW3) to CRS 456
   Expected: Add HW3 to CRS 456 assignment list
   Result: Added HW3 to CRS 456 assignment list
   Status: Passed
6. Change the percentages of the categories for a CRS 456
   Expected: Participation = 5, HW = 25, Tests = 45, Projects = 25
   Result: Participation = 5, HW = 25, Tests = 45, Projects = 25
   Status: Passed
7. Compute the average/highest/lowest score of ASSIGNMENT_ID = 3 for CRS 456
   Expected: AVG = 85, MAX = 100, MIN = 70
   Result: AVG = 85.0000, MAX = 100, MIN = 70
   Status: Passed
8. Insert a non-unique field as a unique key into tables. Tested on Course, Student and Category tables.
   Expected: Throws an error
   Result: Throws an error
   Status: Passed
9. Query a table that does not exist.
   Expected: Throws an error
   Result: Throws an error
   Status: Passed
10. Query a key that does not exist. Tested on Course, Student, Assignment, Scores, Category
    Expected: Throws an error
    Result: Throws an error
    Status: Passed
11. Select column from a table that does not exist. Tested on Student, Assignment, and Category
    Expected: Throws an error
    Result: Throws an error
    Status: Passed

12. Select row from a table that does not exist. Tested on Student, Assignment, and Category
    Expected: Throws an error
    Result: Throws an error
    Status: Passed