Devendra Shah @02744231, Vijay Chaudhary, Samman Bikram Thapa

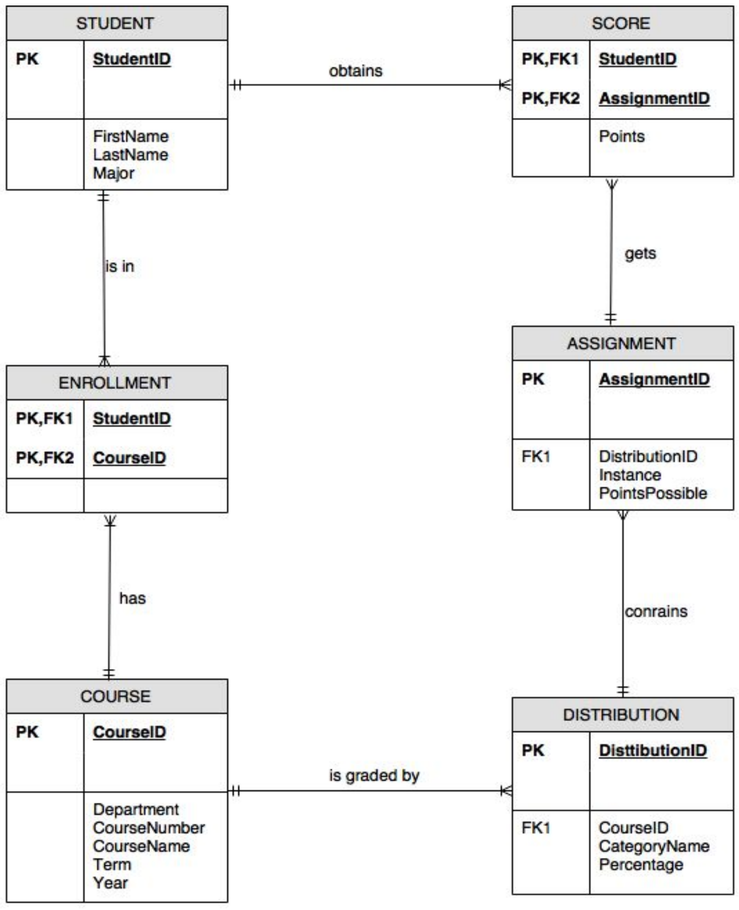
Database Systems

Professor Chunmei Liu

April 19, 2017

Project 1

1. The ER diagram (with the attributes and foreign keys/primary keys indicated);



ER Diagram for Grade Book Database

DATABASE SYSTEMS FINAL PROJECT

-Devendra Shah, Samman Bikram Thapa, Vijay Chaudhary

Legend:

PK = Primary Key

And FK = Foreign Key

2. The commands for creating tables and inserting values;

a) Commands for creating tables

```
drop table if exists STUDENT;

CREATE TABLE STUDENT (
    FirstName VARCHAR(50) NOT NULL,
    LastName VARCHAR(50) NOT NULL,
    Major VARCHAR(30),
    StudentID INT NOT NULL UNIQUE,
    PRIMARY KEY (StudentID)
);

drop table if exists COURSE;

CREATE TABLE COURSE (
    Department VARCHAR(30) NOT NULL,
    CourseNumber INT NOT NULL,
    CourseName VARCHAR(50) NOT NULL,
    Term VARCHAR(15) NOT NULL,
    Year INT NOT NULL,
    CourseID INT NOT NULL UNIQUE,
    PRIMARY KEY (CourseID)
);

drop table if exists ENROLLMENT;

CREATE TABLE ENROLLMENT (
    StudentID INT NOT NULL,
    CourseID INT NOT NULL,
    PRIMARY KEY(StudentID, CourseID)
);

drop table if exists DISTRIBUTION;

CREATE TABLE DISTRIBUTION (
    DistributionID INT NOT NULL UNIQUE,
    CourseID INT NOT NULL,
    CategoryName VARCHAR(30) NOT NULL,
    PERCENTAGE INT NOT NULL,
    PRIMARY KEY(DistributionID)
);

drop table if exists ASSIGNMENT;

CREATE TABLE ASSIGNMENT (
    AssignmentID INT NOT NULL UNIQUE,
    DistributionID INT NOT NULL,
    Instance INT NOT NULL,
    PointsPossible INT DEFAULT 0 NOT NULL,
    PRIMARY KEY(AssignmentID)
);

drop table if exists SCORE;

CREATE TABLE SCORE (
    StudentID INT NOT NULL,
    AssignmentID INT NOT NULL,
    Points INT DEFAULT 0 NOT NULL,
    PRIMARY KEY(StudentID, AssignmentID)
);
```

b) Commands for inserting values

```
/* Inserting values in STUDENT table */

INSERT INTO STUDENT VALUES('Richard', 'Hendricks', 'Computer
Science', 1234);
INSERT INTO STUDENT VALUES('Jared', 'Dunn', 'Management Science',
5678);
INSERT INTO STUDENT VALUES('Erlich', 'Bachman', 'Aviato', 3456);
INSERT INTO STUDENT VALUES('Jimmy', 'Quoyang', 'Marine Biology',
4590);
INSERT INTO STUDENT VALUES('Dinesh', 'Gilfoyle', 'Computer
Engineering', 5337);
INSERT INTO STUDENT VALUES('John', 'Doe', 'English', 5555);

/* Inserting values in COURSE table */

INSERT INTO COURSE VALUES('Math', 157, 'Calculus-2', 'Fall',
2017, 85675);
```

```
INSERT INTO COURSE VALUES('Computer Science', 350, 'Programming
Languages', 'Spring', 2017, 89994);
INSERT INTO COURSE VALUES('English', 109, 'Technical Writing',
'Fall', 2016, 56738);
INSERT INTO COURSE VALUES('Computer Science', 533, 'Senior
Project', 'Spring', 2017, 90573);
INSERT INTO COURSE VALUES('Physics', 100, 'Mechanics', 'Fall',
2016, 48387);


/* Inserting values in ENROLLMENT table */


INSERT INTO ENROLLMENT VALUES(1234, 85675);
INSERT INTO ENROLLMENT VALUES(5678, 85675);
INSERT INTO ENROLLMENT VALUES(3456, 85675);
INSERT INTO ENROLLMENT VALUES(4590, 85675);
INSERT INTO ENROLLMENT VALUES(5337, 85675);
INSERT INTO ENROLLMENT VALUES(5555, 85675);

INSERT INTO ENROLLMENT VALUES(1234, 56738);
INSERT INTO ENROLLMENT VALUES(5678, 56738);
INSERT INTO ENROLLMENT VALUES(3456, 56738);
INSERT INTO ENROLLMENT VALUES(4590, 56738);
INSERT INTO ENROLLMENT VALUES(5337, 56738);
INSERT INTO ENROLLMENT VALUES(5555, 56738);

INSERT INTO ENROLLMENT VALUES(1234, 89994);
INSERT INTO ENROLLMENT VALUES(1234, 90573);
INSERT INTO ENROLLMENT VALUES(1234, 48387);
INSERT INTO ENROLLMENT VALUES(5337, 90573);

/* Inserting values in DISTRIBUTION table */

INSERT INTO DISTRIBUTION VALUES(1, 85675, 'Quiz', 50);
INSERT INTO DISTRIBUTION VALUES(2, 85675, 'HW', 10);
INSERT INTO DISTRIBUTION VALUES(3, 85675, 'MidTerm', 20);
INSERT INTO DISTRIBUTION VALUES(4, 85675, 'Final', 20);

INSERT INTO DISTRIBUTION VALUES(5, 89994, 'Participation', 40);
INSERT INTO DISTRIBUTION VALUES(6, 89994, 'HW', 10);
INSERT INTO DISTRIBUTION VALUES(7, 89994, 'MidTerm', 25);
INSERT INTO DISTRIBUTION VALUES(8, 89994, 'Final', 25);

INSERT INTO DISTRIBUTION VALUES(9, 56738, 'Quiz', 40);
```

```sql
INSERT INTO DISTRIBUTION VALUES(10, 56738, 'HW', 15);
INSERT INTO DISTRIBUTION VALUES(11, 56738, 'MidTerm', 20);
INSERT INTO DISTRIBUTION VALUES(12, 56738, 'Final', 25);

INSERT INTO DISTRIBUTION VALUES(13, 90573, 'Quiz', 20);
INSERT INTO DISTRIBUTION VALUES(14, 90573, 'HW', 25);
INSERT INTO DISTRIBUTION VALUES(15, 90573, 'Project', 30);
INSERT INTO DISTRIBUTION VALUES(16, 90573, 'Final', 25);

INSERT INTO DISTRIBUTION VALUES(17, 48387, 'Quiz', 30);
INSERT INTO DISTRIBUTION VALUES(18, 48387, 'HW', 25);
INSERT INTO DISTRIBUTION VALUES(19, 48387, 'Project', 20);
INSERT INTO DISTRIBUTION VALUES(20, 48387, 'Final', 25);


/* Inserting values in ASSIGNMENT table */

INSERT INTO ASSIGNMENT VALUES(1, 1, 1, 100);
INSERT INTO ASSIGNMENT VALUES(2, 1, 2, 100);

INSERT INTO ASSIGNMENT VALUES(3, 2, 1, 100);
INSERT INTO ASSIGNMENT VALUES(4, 2, 2, 100);

INSERT INTO ASSIGNMENT VALUES(5, 3, 1, 100);
INSERT INTO ASSIGNMENT VALUES(6, 3, 2, 100);

INSERT INTO ASSIGNMENT VALUES(7, 4, 1, 100);
INSERT INTO ASSIGNMENT VALUES(8, 4, 2, 100);

INSERT INTO ASSIGNMENT VALUES(9, 5, 1, 100);
INSERT INTO ASSIGNMENT VALUES(10, 5, 2, 100);

INSERT INTO ASSIGNMENT VALUES(11, 6, 1, 100);
INSERT INTO ASSIGNMENT VALUES(12, 6, 2, 100);

INSERT INTO ASSIGNMENT VALUES(13, 7, 1, 100);
INSERT INTO ASSIGNMENT VALUES(14, 7, 2, 100);

INSERT INTO ASSIGNMENT VALUES(15, 8, 1, 100);
INSERT INTO ASSIGNMENT VALUES(16, 8, 2, 100);

INSERT INTO ASSIGNMENT VALUES(17, 9, 1, 100);
INSERT INTO ASSIGNMENT VALUES(18, 9, 2, 100);

INSERT INTO ASSIGNMENT VALUES(19, 10, 1, 100);
INSERT INTO ASSIGNMENT VALUES(20, 10, 2, 100);

INSERT INTO ASSIGNMENT VALUES(21, 11, 1, 100);
INSERT INTO ASSIGNMENT VALUES(22, 11, 2, 100);

INSERT INTO ASSIGNMENT VALUES(23, 12, 1, 100);
INSERT INTO ASSIGNMENT VALUES(24, 12, 2, 100);

INSERT INTO ASSIGNMENT VALUES(25, 13, 1, 100);
INSERT INTO ASSIGNMENT VALUES(26, 13, 2, 100);

INSERT INTO ASSIGNMENT VALUES(27, 14, 1, 100);
INSERT INTO ASSIGNMENT VALUES(28, 14, 2, 100);

INSERT INTO ASSIGNMENT VALUES(29, 15, 1, 100);
INSERT INTO ASSIGNMENT VALUES(30, 15, 2, 100);

INSERT INTO ASSIGNMENT VALUES(31, 16, 1, 100);
INSERT INTO ASSIGNMENT VALUES(32, 16, 2, 100);

INSERT INTO ASSIGNMENT VALUES(33, 17, 1, 100);
INSERT INTO ASSIGNMENT VALUES(34, 17, 2, 100);

INSERT INTO ASSIGNMENT VALUES(35, 18, 1, 100);
INSERT INTO ASSIGNMENT VALUES(36, 18, 2, 100);

INSERT INTO ASSIGNMENT VALUES(37, 19, 1, 100);
INSERT INTO ASSIGNMENT VALUES(38, 19, 2, 100);

INSERT INTO ASSIGNMENT VALUES(39, 20, 1, 100);
INSERT INTO ASSIGNMENT VALUES(40, 20, 2, 100);


INSERT INTO ASSIGNMENT VALUES(42, 1, 3, 100);

/* Inserting values in SCORE table */

INSERT INTO SCORE VALUES(1234, 1, 85);
INSERT INTO SCORE VALUES(5678, 1, 80);
INSERT INTO SCORE VALUES(3456, 1, 95);
INSERT INTO SCORE VALUES(4590, 1, 65);
INSERT INTO SCORE VALUES(5337, 1, 100);
INSERT INTO SCORE VALUES(5555, 1, 93);

INSERT INTO SCORE VALUES(1234, 2, 81);
INSERT INTO SCORE VALUES(5678, 2, 84);
INSERT INTO SCORE VALUES(3456, 2, 95);
INSERT INTO SCORE VALUES(4590, 2, 62);
INSERT INTO SCORE VALUES(5337, 2, 90);
INSERT INTO SCORE VALUES(5555, 2, 93);


INSERT INTO SCORE VALUES(1234, 18, 87);
INSERT INTO SCORE VALUES(5678, 18, 94);
INSERT INTO SCORE VALUES(3456, 18, 55);
INSERT INTO SCORE VALUES(4590, 18, 72);
INSERT INTO SCORE VALUES(5337, 18, 0);
INSERT INTO SCORE VALUES(5555, 18, 99);

INSERT INTO SCORE VALUES(1234, 16, 80);
INSERT INTO SCORE VALUES(1234, 30, 85);
INSERT INTO SCORE VALUES(1234, 40, 95);

INSERT INTO SCORE VALUES(5337, 26, 78);

INSERT INTO SCORE VALUES(1234, 8, 84);

INSERT INTO SCORE VALUES(1234, 42, 80);
INSERT INTO SCORE VALUES(5678, 42, 81);
INSERT INTO SCORE VALUES(3456, 42, 94);
INSERT INTO SCORE VALUES(4590, 42, 46);
INSERT INTO SCORE VALUES(5337, 42, 91);
INSERT INTO SCORE VALUES(5555, 42, 97);
```

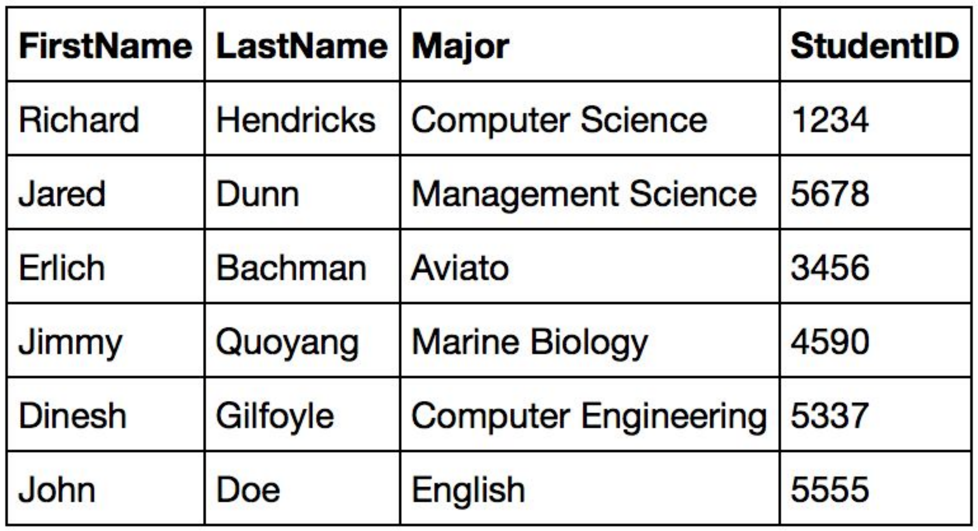
3) The tables with the contents that you have inserted;

Command to show the contents of the tables: `SELECT * from [table_name];`

1) STUDENT

Command Used: `SELECT * from STUDENT;`

Table:

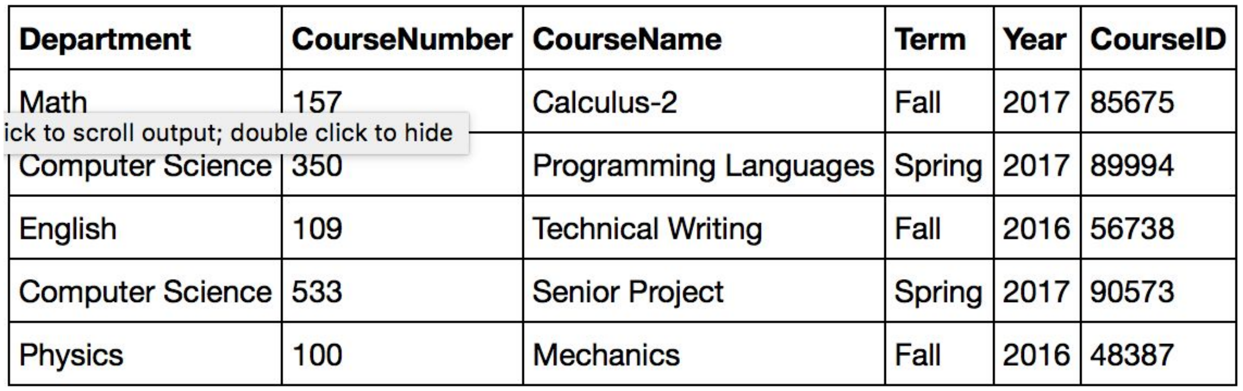


FirstName	LastName	Major	StudentID
Richard	Hendricks	Computer Science	1234
Jared	Dunn	Management Science	5678
Erlich	Bachman	Aviato	3456
Jimmy	Quoyang	Marine Biology	4590
Dinesh	Gilfoyle	Computer Engineering	5337
John	Doe	English	5555

2) COURSE

Command Used: `SELECT * from COURSE;`

Table:

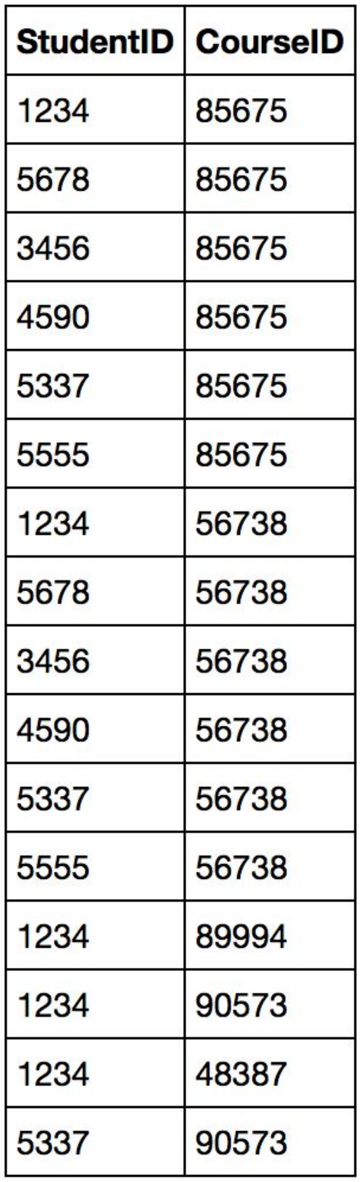

ick to scroll output; double click to hide

Department	CourseNumber	CourseName	Term	Year	CourseID
Math	157	Calculus-2	Fall	2017	85675
Computer Science	350	Programming Languages	Spring	2017	89994
English	109	Technical Writing	Fall	2016	56738
Computer Science	533	Senior Project	Spring	2017	90573
Physics	100	Mechanics	Fall	2016	48387

3) ENROLLMENT

Command Used: `SELECT * from ENROLLMENT;`

Table:

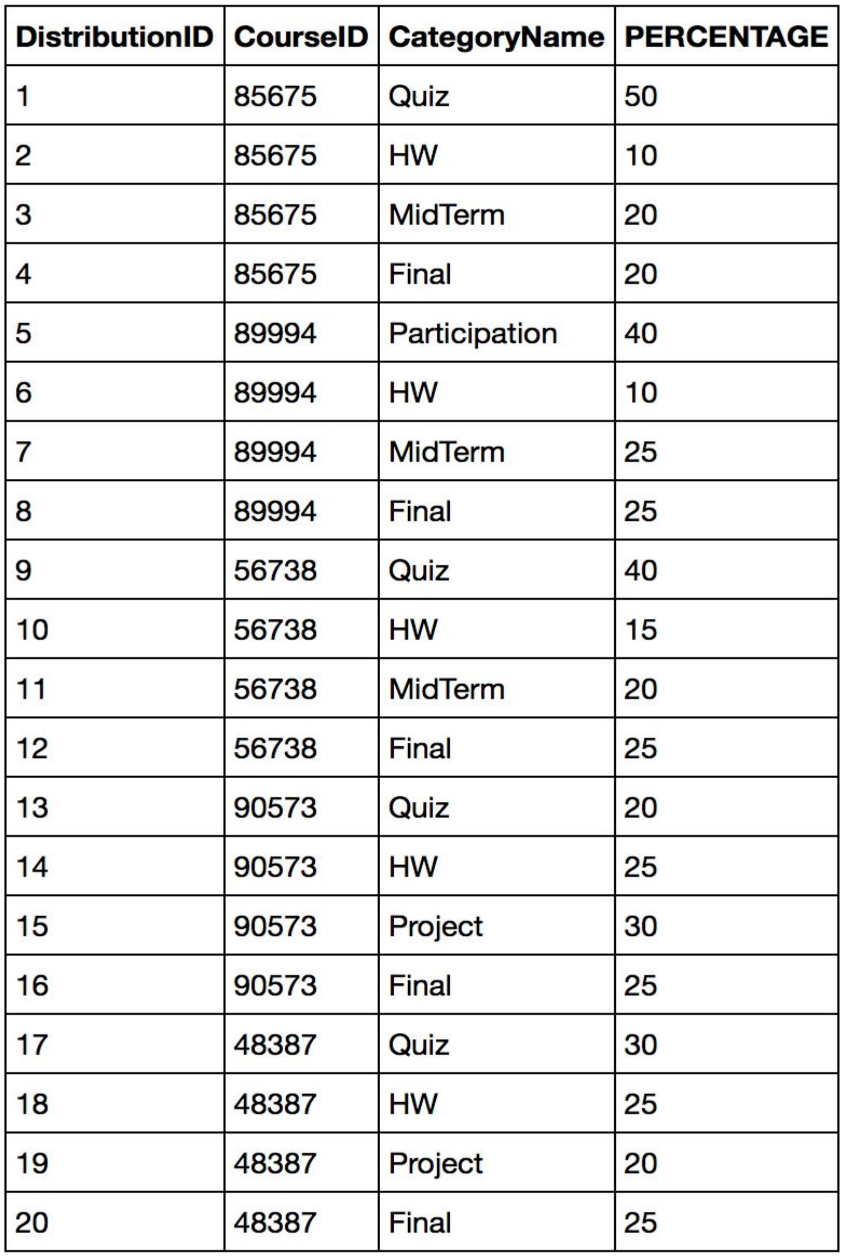


StudentID	CourseID
1234	85675
5678	85675
3456	85675
4590	85675
5337	85675
5555	85675
1234	56738
5678	56738
3456	56738
4590	56738
5337	56738
5555	56738
1234	89994
1234	90573
1234	48387
5337	90573

4) DISTRIBUTION

Command Used: `SELECT * from DISTRIBUTION;`

Table:

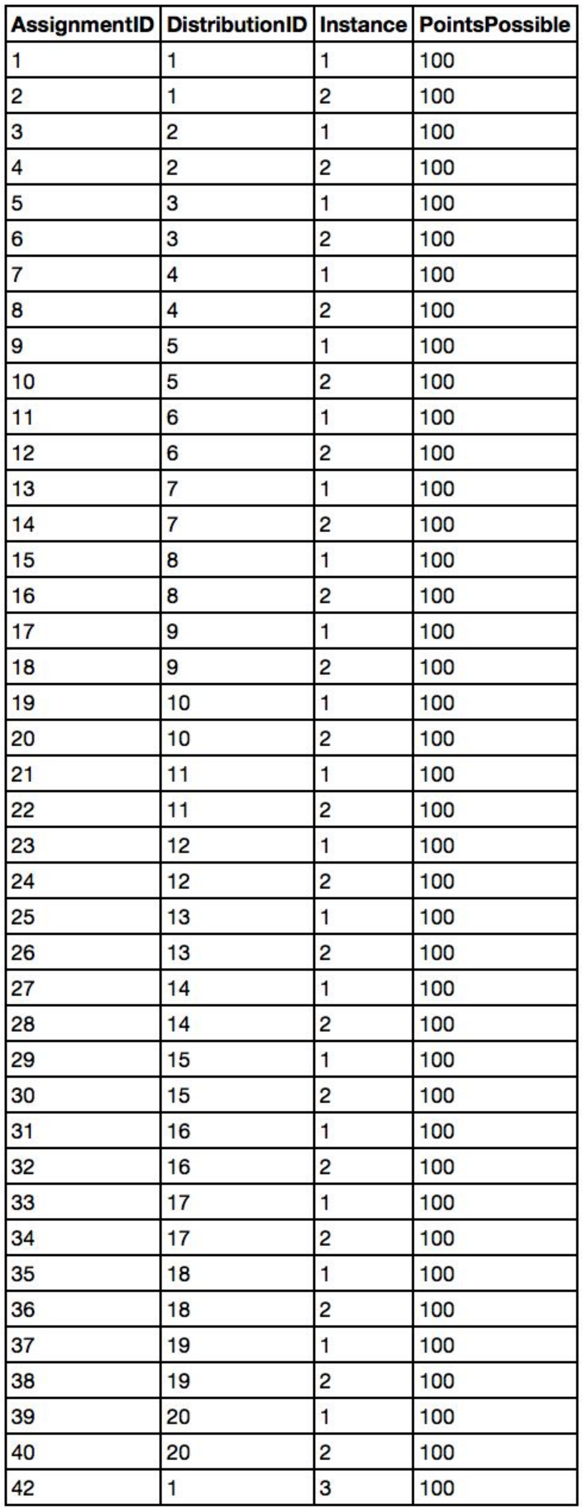


DistributionID	CourseID	CategoryName	PERCENTAGE
1	85675	Quiz	50
2	85675	HW	10
3	85675	MidTerm	20
4	85675	Final	20
5	89994	Participation	40
6	89994	HW	10
7	89994	MidTerm	25
8	89994	Final	25
9	56738	Quiz	40
10	56738	HW	15
11	56738	MidTerm	20
12	56738	Final	25
13	90573	Quiz	20
14	90573	HW	25
15	90573	Project	30
16	90573	Final	25
17	48387	Quiz	30
18	48387	HW	25
19	48387	Project	20
20	48387	Final	25

5) ASSIGNMENT

Command Used: `SELECT * from ASSIGNMENT;`

Table:

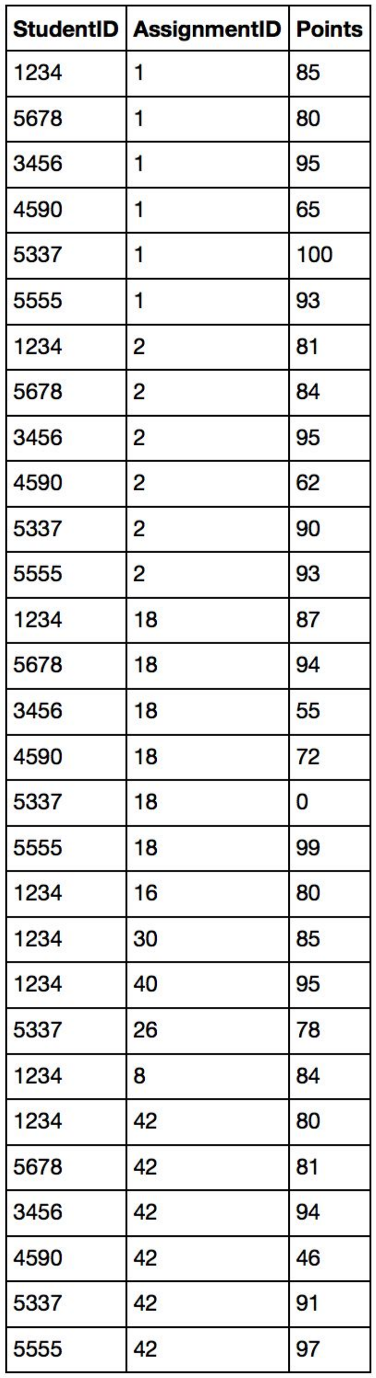


AssignmentID	DistributionID	Instance	PointsPossible
1	1	1	100
2	1	2	100
3	2	1	100
4	2	2	100
5	3	1	100
6	3	2	100
7	4	1	100
8	4	2	100
9	5	1	100
10	5	2	100
11	6	1	100
12	6	2	100
13	7	1	100
14	7	2	100
15	8	1	100
16	8	2	100
17	9	1	100
18	9	2	100
19	10	1	100
20	10	2	100
21	11	1	100
22	11	2	100
23	12	1	100
24	12	2	100
25	13	1	100
26	13	2	100
27	14	1	100
28	14	2	100
29	15	1	100
30	15	2	100
31	16	1	100
32	16	2	100
33	17	1	100
34	17	2	100
35	18	1	100
36	18	2	100
37	19	1	100
38	19	2	100
39	20	1	100
40	20	2	100
42	1	3	100

6) SCORE

Command Used: `SELECT * from SCORE;`

Table:



StudentID	AssignmentID	Points
1234	1	85
5678	1	80
3456	1	95
4590	1	65
5337	1	100
5555	1	93
1234	2	81
5678	2	84
3456	2	95
4590	2	62
5337	2	90
5555	2	93
1234	18	87
5678	18	94
3456	18	55
4590	18	72
5337	18	0
5555	18	99
1234	16	80
1234	30	85
1234	40	95
5337	26	78
1234	8	84
1234	42	80
5678	42	81
3456	42	94
4590	42	46
5337	42	91
5555	42	97

Part 4) Commands for tasks 4 to 12

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

Average score

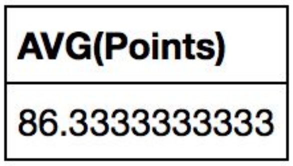
Command:

```
SELECT AVG(Points) FROM SCORE WHERE AssignmentID =
[Assignment_ID];
```

Example Command:

```
/*4. Calculating average score for assignment 1.*/
SELECT AVG(Points) FROM SCORE WHERE AssignmentID = 1;
```

Result:



AVG(Points)
86.3333333333

Highest score

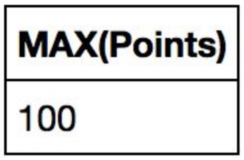
Command:

```
      SELECT MAX(Points) FROM SCORE WHERE AssignmentID =
[Assignment_ID];
```

Example Command:

```
/*4. Calculating highest score for assignment 1. */
SELECT MAX(Points) FROM SCORE WHERE AssignmentID = 1;
```

Result:



MAX(Points)
100

Lowest Score

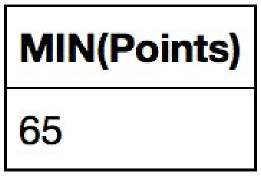
Command:

```
        SELECT MIN(Points) FROM SCORE WHERE AssignmentID =
[Assignment_ID];
```

Example Command:

```
        /*4. Calculating lowest score for assignment 1. */

        SELECT MIN(Points) FROM SCORE WHERE AssignmentID = 1;
```

Result:



MIN(Points)
65

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

Command:

```
SELECT STUDENT.StudentID, FirstName, LastName, Major, CourseID
FROM STUDENT JOIN ENROLLMENT
WHERE CourseID = [CourseID]
AND STUDENT.StudentID = ENROLLMENT.StudentID;
```

Example Command:

```
/* 5. List all of the students in a given course;
Listing all students in Calculus 2 class */
```

```
SELECT STUDENT.StudentID, FirstName, LastName, Major, CourseID
FROM STUDENT JOIN ENROLLMENT
WHERE CourseID = 85675
AND STUDENT.StudentID = ENROLLMENT.StudentID;
```

Result:

StudentID	FirstName	LastName	Major	CourseID
1234	Richard	Hendricks	Computer Science	85675
5678	Jared	Dunn	Management Science	85675
3456	Erlich	Bachman	Aviato	85675
4590	Jimmy	Quoyang	Marine Biology	85675
5337	Dinesh	Gilfoyle	Computer Engineering	85675
5555	John	Doe	English	85675

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

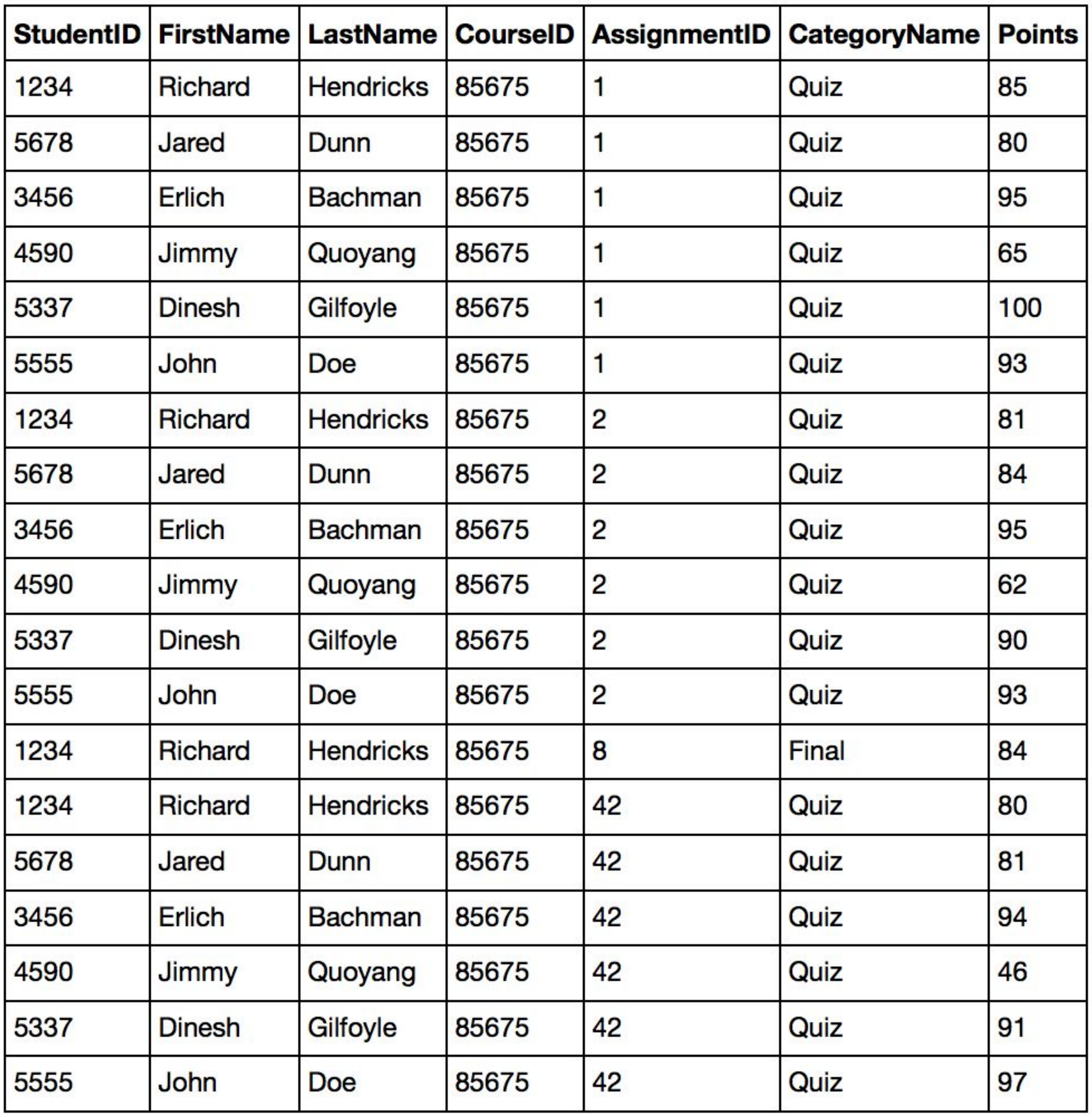
Command:

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

Doing this for Calculus 2 class
*/

SELECT pt.StudentID as StudentID, st.FirstName as FirstName,
st.LastName as LastName, pt.CourseID as CourseID,
pt.AssignmentID as AssignmentID, pt.CategoryName as CategoryName,
pt.Points as Points
FROM (SELECT STUDENT.StudentID, AssignmentID, FirstName, LastName,
      CourseID, Points
    FROM STUDENT JOIN ENROLLMENT JOIN SCORE
    WHERE CourseID = [CourseID]
    AND STUDENT.StudentID = ENROLLMENT.StudentID
    AND STUDENT.StudentID = SCORE.StudentID) st
JOIN
    (SELECT StudentID, CourseID, CategoryName,
ASSIGNMENT.AssignmentID,Points
    FROM DISTRIBUTION JOIN ASSIGNMENT JOIN SCORE
    WHERE DISTRIBUTION.CourseID = [CourseID]
    AND DISTRIBUTION.DistributionID = ASSIGNMENT.DistributionID
    AND ASSIGNMENT.AssignmentID = SCORE.AssignmentID) pt
WHERE st.AssignmentID = pt.AssignmentID
AND st.Points = pt.Points;
```

Example Command:

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

Doing this for Calculus 2 class
*/

SELECT pt.StudentID as StudentID, st.FirstName as FirstName,
st.LastName as LastName, pt.CourseID as CourseID,
pt.AssignmentID as AssignmentID, pt.CategoryName as CategoryName,
pt.Points as Points
FROM (SELECT STUDENT.StudentID, AssignmentID, FirstName, LastName,
      CourseID, Points
    FROM STUDENT JOIN ENROLLMENT JOIN SCORE
    WHERE CourseID = 85675
    AND STUDENT.StudentID = ENROLLMENT.StudentID
    AND STUDENT.StudentID = SCORE.StudentID) st
JOIN
    (SELECT StudentID, CourseID, CategoryName,
ASSIGNMENT.AssignmentID,Points
    FROM DISTRIBUTION JOIN ASSIGNMENT JOIN SCORE
    WHERE DISTRIBUTION.CourseID = 85675
    AND DISTRIBUTION.DistributionID = ASSIGNMENT.DistributionID
    AND ASSIGNMENT.AssignmentID = SCORE.AssignmentID) pt
WHERE st.AssignmentID = pt.AssignmentID
AND st.Points = pt.Points;
```

Result:



StudentID	FirstName	LastName	CourseID	AssignmentID	CategoryName	Points
1234	Richard	Hendricks	85675	1	Quiz	85
5678	Jared	Dunn	85675	1	Quiz	80
3456	Erlich	Bachman	85675	1	Quiz	95
4590	Jimmy	Quoyang	85675	1	Quiz	65
5337	Dinesh	Gilfoyle	85675	1	Quiz	100
5555	John	Doe	85675	1	Quiz	93
1234	Richard	Hendricks	85675	2	Quiz	81
5678	Jared	Dunn	85675	2	Quiz	84
3456	Erlich	Bachman	85675	2	Quiz	95
4590	Jimmy	Quoyang	85675	2	Quiz	62
5337	Dinesh	Gilfoyle	85675	2	Quiz	90
5555	John	Doe	85675	2	Quiz	93
1234	Richard	Hendricks	85675	8	Final	84
1234	Richard	Hendricks	85675	42	Quiz	80
5678	Jared	Dunn	85675	42	Quiz	81
3456	Erlich	Bachman	85675	42	Quiz	94
4590	Jimmy	Quoyang	85675	42	Quiz	46
5337	Dinesh	Gilfoyle	85675	42	Quiz	91
5555	John	Doe	85675	42	Quiz	97

7.Add an assignment to a course;

Command:

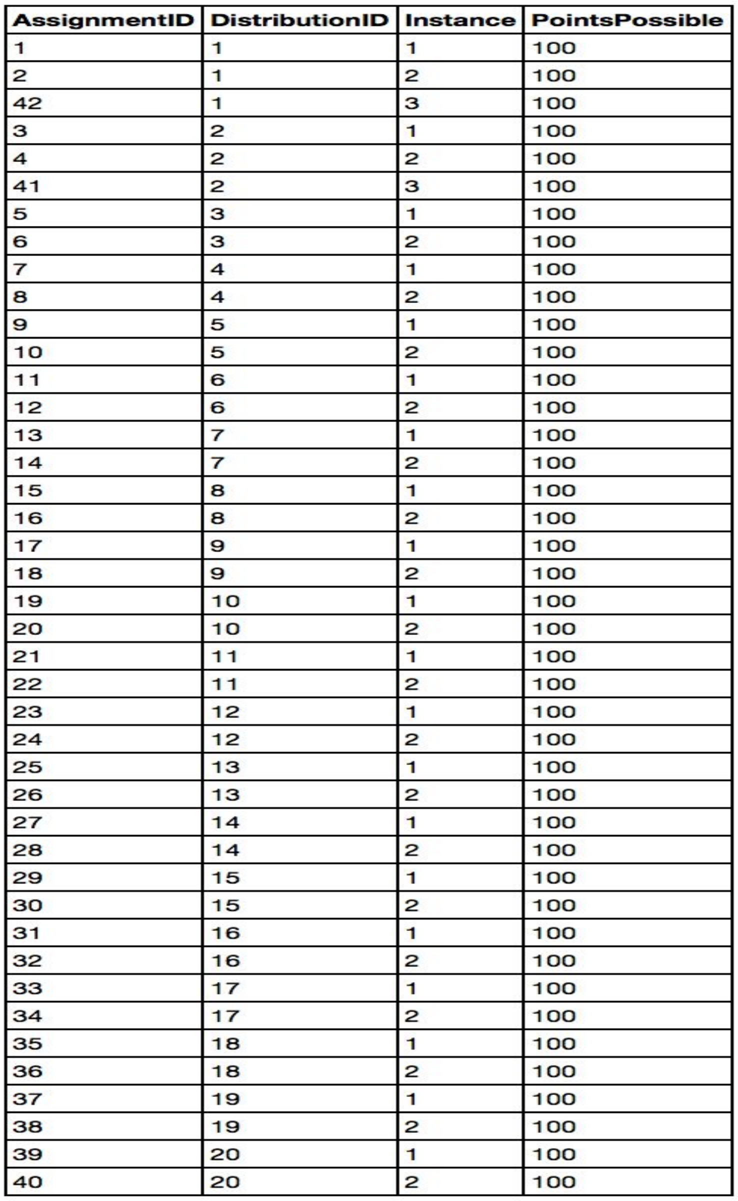
```
INSERT INTO ASSIGNMENT VALUES ([Assignment_ID], [Distribution_ID],
[Instance], [PointsPossible]);
```

Example Command:

```
/*7. Add an assignment to a course

Adding a Homework Assignment for Calculus-2 class
HW ID for Calculus-2 class is 2, and
there are already 2 HWs for this class in the database;*/

INSERT INTO ASSIGNMENT VALUES (41, 2, 3, 100);
```

Result:

```
SELECT * FROM ASSIGNMENT ORDER BY DistributionID;
```



AssignmentID	DistributionID	Instance	PointsPossible
1	1	1	100
2	1	2	100
42	1	3	100
3	2	1	100
4	2	2	100
41	2	3	100
5	3	1	100
6	3	2	100
7	4	1	100
8	4	2	100
9	5	1	100
10	5	2	100
11	6	1	100
12	6	2	100
13	7	1	100
14	7	2	100
15	8	1	100
16	8	2	100
17	9	1	100
18	9	2	100
19	10	1	100
20	10	2	100
21	11	1	100
22	11	2	100
23	12	1	100
24	12	2	100
25	13	1	100
26	13	2	100
27	14	1	100
28	14	2	100
29	15	1	100
30	15	2	100
31	16	1	100
32	16	2	100
33	17	1	100
34	17	2	100
35	18	1	100
36	18	2	100
37	19	1	100
38	19	2	100
39	20	1	100
40	20	2	100

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

Command Used:

```
UPDATE DISTRIBUTION
SET PERCENTAGE = [Percentage]
WHERE CourseID = [CourseID]
AND CategoryName = [CategoryName]
AND DistributionID = [DistributionID];
```

Example Command:

```
/*8. Change the percentages of the categories for a course;

Changing the percentages for all the categories of Calculus-2 class*/

UPDATE DISTRIBUTION
SET PERCENTAGE = 80
WHERE CourseID = 85675
AND CategoryName = 'Quiz'
AND DistributionID = 1;

UPDATE DISTRIBUTION
SET PERCENTAGE = 5
WHERE CourseID = 85675
AND CategoryName = 'HW'
AND DistributionID = 2;

UPDATE DISTRIBUTION
SET PERCENTAGE = 5
WHERE CourseID = 85675
AND CategoryName = 'MidTerm'
AND DistributionID = 3;

UPDATE DISTRIBUTION
SET PERCENTAGE = 10
WHERE CourseID = 85675
AND CategoryName = 'Final'
AND DistributionID = 4;
```

Result:

```
SELECT * FROM DISTRIBUTION;
```

DistributionID	CourseID	CategoryName	PERCENTAGE
1	85675	Quiz	80
2	85675	HW	5
3	85675	MidTerm	5
4	85675	Final	10
5	89994	Participation	40
6	89994	HW	10
7	89994	MidTerm	25
8	89994	Final	25
9	56738	Quiz	40
10	56738	HW	15
11	56738	MidTerm	20
12	56738	Final	25
13	90573	Quiz	20
14	90573	HW	25
15	90573	Project	30
16	90573	Final	25
17	48387	Quiz	30
18	48387	HW	25
19	48387	Project	20
20	48387	Final	25

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

Command Used:

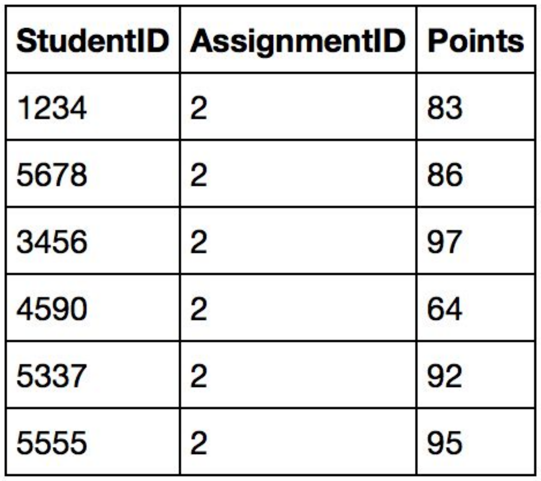
```
UPDATE SCORE
SET Points = Points + 2
WHERE AssignmentID = [AssignmentID];
```

Example Command:

```
/*9. Add 2 points to the score of each student on an assignment;
Adding 2 points to the score of all the students on Quiz-2
in Calculus 2 class*/

UPDATE SCORE
SET Points = Points + 2
WHERE AssignmentID = 2;
```

Result:

```
SELECT *
FROM SCORE
WHERE AssignmentID = 2;
```



StudentID	AssignmentID	Points
1234	2	83
5678	2	86
3456	2	97
4590	2	64
5337	2	92
5555	2	95

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

Command Used:

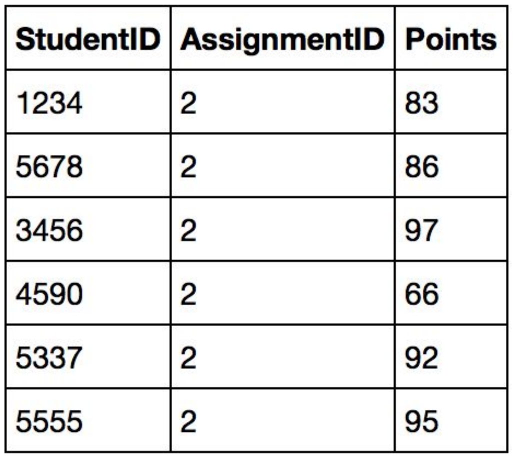
```
UPDATE SCORE
SET Points = Points + 2
WHERE StudentID IN (SELECT StudentID
                    FROM STUDENT
                    WHERE LastName LIKE '%Q%')
AND AssignmentID = [AssignmentID];
```

Example Command:

```
/*10. Add 2 points just to those students whose
last name contains a 'Q'.
Adding 2 points to the score of the students whose
last name contains a 'Q' on Quiz-2 in Calculus 2 class
*/
```

```
UPDATE SCORE
SET Points = Points + 2
WHERE StudentID IN (SELECT StudentID
                    FROM STUDENT
                    WHERE LastName LIKE '%Q%')
AND AssignmentID = 2;
```

Result:



StudentID	AssignmentID	Points
1234	2	83
5678	2	86
3456	2	97
4590	2	66
5337	2	92
5555	2	95

11. Compute the grade for a student;

Command:

```
drop table if exists currgrades;

CREATE TABLE currgrades AS
SELECT pt.StudentID as StudentID, st.FirstName as FName,
st.LastName as LName, pt.CourseID as CourseID,
pt.AssignmentID as asn_id, pt.DistributionID as distrID,
CategoryName as category, PERCENTAGE as Weight,
PointsPossible as maxP,
pt.Points as points ,
(1.0*pt.Points)/ PointsPossible * PERCENTAGE AS grade
FROM (SELECT STUDENT.StudentID, AssignmentID, FirstName, LastName,
    CourseID, Points
    FROM STUDENT JOIN ENROLLMENT JOIN SCORE
    WHERE CourseID = [CourseID]
    AND STUDENT.StudentID = ENROLLMENT.StudentID
    AND STUDENT.StudentID = SCORE.StudentID) st
JOIN
    (SELECT StudentID, CourseID, CategoryName,
DISTRIBUTION.DistributionID,
    PERCENTAGE, ASSIGNMENT.AssignmentID, ASSIGNMENT.PointsPossible,
Points
    FROM DISTRIBUTION JOIN ASSIGNMENT JOIN SCORE
    WHERE DISTRIBUTION.CourseID = [CourseID]
    AND DISTRIBUTION.DistributionID = ASSIGNMENT.DistributionID
    AND ASSIGNMENT.AssignmentID = SCORE.AssignmentID) pt
WHERE st.AssignmentID = pt.AssignmentID
AND st.Points = pt.Points
AND pt.StudentID = [StudentID]
ORDER BY DistributionID;


/*Calculating weighted grades*/

UPDATE currgrades
SET grade = (1.0 * points * Weight) / maxP;

/*Calculating grade on a 100*/

drop table if exists finalgrades;
CREATE table finalgrades AS
SELECT *, sg.instancecount as instances,
grade/instancecount as finalgrade
FROM currgrades cg
INNER JOIN (SELECT distrID, count(distrID) as instancecount
           FROM currgrades
           GROUP BY distrID) sg ON cg.distrID = sg.distrID;
SELECT SUM(finalgrade) as FINALGRADE from finalgrades;
```

Example Command:

```
/*11. Compute the grade for a student;

Computing the grade for Richard Hendricks
StudentID 1234 in Calculus 2 class
CourseID 85765*/

drop table if exists currgrades;
```

```sql
CREATE TABLE currgrades AS
SELECT pt.StudentID as StudentID, st.FirstName as FName,
st.LastName as LName, pt.CourseID as CourseID,
pt.AssignmentID as asn_id, pt.DistributionID as distrID,
CategoryName as category, PERCENTAGE as Weight,
PointsPossible as maxP,
pt.Points as points ,
(1.0*pt.Points)/ PointsPossible * PERCENTAGE AS grade
FROM (SELECT STUDENT.StudentID, AssignmentID, FirstName, LastName,
    CourseID, Points
    FROM STUDENT JOIN ENROLLMENT JOIN SCORE
    WHERE CourseID = 85675
    AND STUDENT.StudentID = ENROLLMENT.StudentID
    AND STUDENT.StudentID = SCORE.StudentID) st
JOIN
    (SELECT StudentID, CourseID, CategoryName,
DISTRIBUTION.DistributionID,
    PERCENTAGE, ASSIGNMENT.AssignmentID, ASSIGNMENT.PointsPossible,
Points
    FROM DISTRIBUTION JOIN ASSIGNMENT JOIN SCORE
    WHERE DISTRIBUTION.CourseID = 85675
    AND DISTRIBUTION.DistributionID = ASSIGNMENT.DistributionID
    AND ASSIGNMENT.AssignmentID = SCORE.AssignmentID) pt
WHERE st.AssignmentID = pt.AssignmentID
AND st.Points = pt.Points
AND pt.StudentID = 1234
ORDER BY DistributionID;


/*Calclating weighted grades*/

UPDATE currgrades
SET grade = (1.0 * points * Weight) / maxP;

/*Calculating grade on a 100*/

drop table if exists finalgrades;
CREATE table finalgrades AS
SELECT *, sg.instancecount as instances,
grade/instancecount as finalgrade
FROM currgrades cg
INNER JOIN (SELECT distrID, count(distrID) as instancecount
           FROM currgrades
           GROUP BY distrID) sg ON cg.distrID = sg.distrID;
```

```
SELECT SUM(finalgrade) as FINALGRADE from finalgrades;
```

Result:

FINALGRADE
74.0

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

Command Used:

```
drop table if exists currgrades;


CREATE TABLE currgrades AS
SELECT pt.StudentID as StudentID, st.FirstName as FName,
st.LastName as LName, pt.CourseID as CourseID,
pt.AssignmentID as asn_id, pt.DistributionID as distrID,
CategoryName as category, PERCENTAGE as Weight,
PointsPossible as maxP,
pt.Points as points ,
(1.0 * pt.Points) / PointsPossible * PERCENTAGE AS grade
FROM (SELECT STUDENT.StudentID, AssignmentID, FirstName, LastName,
    CourseID, Points
    FROM STUDENT JOIN ENROLLMENT JOIN SCORE
    WHERE CourseID = [CourseID]
    AND STUDENT.StudentID = ENROLLMENT.StudentID
    AND STUDENT.StudentID = SCORE.StudentID) st
JOIN
    (SELECT StudentID, CourseID, CategoryName, DISTRIBUTION.DistributionID,
    PERCENTAGE, ASSIGNMENT.AssignmentID, ASSIGNMENT.PointsPossible, Points
    FROM DISTRIBUTION JOIN ASSIGNMENT JOIN SCORE
    WHERE DISTRIBUTION.CourseID = [CourseID]
    AND DISTRIBUTION.DistributionID = ASSIGNMENT.DistributionID
    AND ASSIGNMENT.AssignmentID = SCORE.AssignmentID) pt
WHERE st.AssignmentID = pt.AssignmentID
AND st.Points = pt.Points
AND pt.StudentID = [StudentID]
ORDER BY DistributionID;

/*Calculating weighted grade */
UPDATE currgrades
SET grade = points * 100.0 / maxP;

/*Dropping lowest score for an assignment*/
DELETE from currgrades
WHERE category = [Assignment_Category]
and grade IN (SELECT MIN(grade) as grade from currgrades);

/*Calculating weighted grade */
UPDATE currgrades
SET grade = 1.0 * points * Weight / maxP;
```

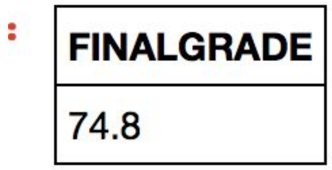
```
/*Calculating grade on a 100 */
drop table if exists finalgrades;

CREATE table finalgrades AS
SELECT *, sg.instancecount as instances,
1.0 * grade/instancecount as finalgrade
FROM currgrades cg
INNER JOIN (SELECT distrID, count(distrID) as instancecount
           FROM currgrades
           GROUP BY distrID) sg ON cg.distrID = sg.distrID;

/*Final Answer for task 12 */

SELECT SUM(finalgrade) as FINALGRADE from finalgrades;
```

Example Command:

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

Computing the grade for Richard Hendricks
StudentID 1234 in Calculus 2 class
CourseID 85765
Dropping the lowest Quiz is dropped*/

drop table if exists currgrades;


CREATE TABLE currgrades AS
SELECT pt.StudentID as StudentID, st.FirstName as FName,
st.LastName as LName, pt.CourseID as CourseID,
pt.AssignmentID as asn_id, pt.DistributionID as distrID,
CategoryName as category, PERCENTAGE as Weight,
PointsPossible as maxP,
pt.Points as points ,
(1.0 * pt.Points) / PointsPossible * PERCENTAGE AS grade
FROM (SELECT STUDENT.StudentID, AssignmentID, FirstName, LastName,
    CourseID, Points
    FROM STUDENT JOIN ENROLLMENT JOIN SCORE
    WHERE CourseID = 85675
    AND STUDENT.StudentID = ENROLLMENT.StudentID
    AND STUDENT.StudentID = SCORE.StudentID) st
JOIN
    (SELECT StudentID, CourseID, CategoryName, DISTRIBUTION.DistributionID,
    PERCENTAGE, ASSIGNMENT.AssignmentID, ASSIGNMENT.PointsPossible, Points
    FROM DISTRIBUTION JOIN ASSIGNMENT JOIN SCORE
    WHERE DISTRIBUTION.CourseID = 85675
    AND DISTRIBUTION.DistributionID = ASSIGNMENT.DistributionID
    AND ASSIGNMENT.AssignmentID = SCORE.AssignmentID) pt
WHERE st.AssignmentID = pt.AssignmentID
AND st.Points = pt.Points
AND pt.StudentID = 1234
ORDER BY DistributionID;

/*Calculating weighted grade */
UPDATE currgrades
SET grade = points * 100.0 / maxP;
```

```
/*Dropping lowest score */
DELETE from currgrades
WHERE category = 'Quiz'
and grade IN (SELECT MIN(grade) as grade from currgrades);

/*Calculating weighted grade */
UPDATE currgrades
SET grade = 1.0 * points * Weight / maxP;


/*Calculating grade on a 100 */
drop table if exists finalgrades;

CREATE table finalgrades AS
SELECT *, sg.instancecount as instances,
1.0 * grade/instancecount as finalgrade
FROM currgrades cg
INNER JOIN (SELECT distrID, count(distrID) as instancecount
           FROM currgrades
           GROUP BY distrID) sg ON cg.distrID = sg.distrID;

/*Final Answer for task 12 */

SELECT SUM(finalgrade) as FINALGRADE from finalgrades;
```

Result:



FINALGRADE
74.8

5. Source Code

Please turn over for the source code

6. ReadME

Please turn over for the ReadMe

7. Test Cases

Please turn over for the test cases