BRIGHT MINDS PROJECT DOCUMENTATION

I/ Introduction:

We created a light-weighed database for Bright Minds Mathematics Tutoring Center. The database will store information regarding the courses, tutors, and students. The database will also tracks the amount of time students spend studying and tutors work at the center.

II/ System Requirements:

- The system shall provide an interface to enter student's and tutor's ID and duration (in hours) to enter into the database.
- The system also shall provide an interface to search for student's and tutor's information by entering student ID, course ID, and tutor ID.
- The system shall retrieve all the required information regarding the student and tutor that matches the ID the user entered.

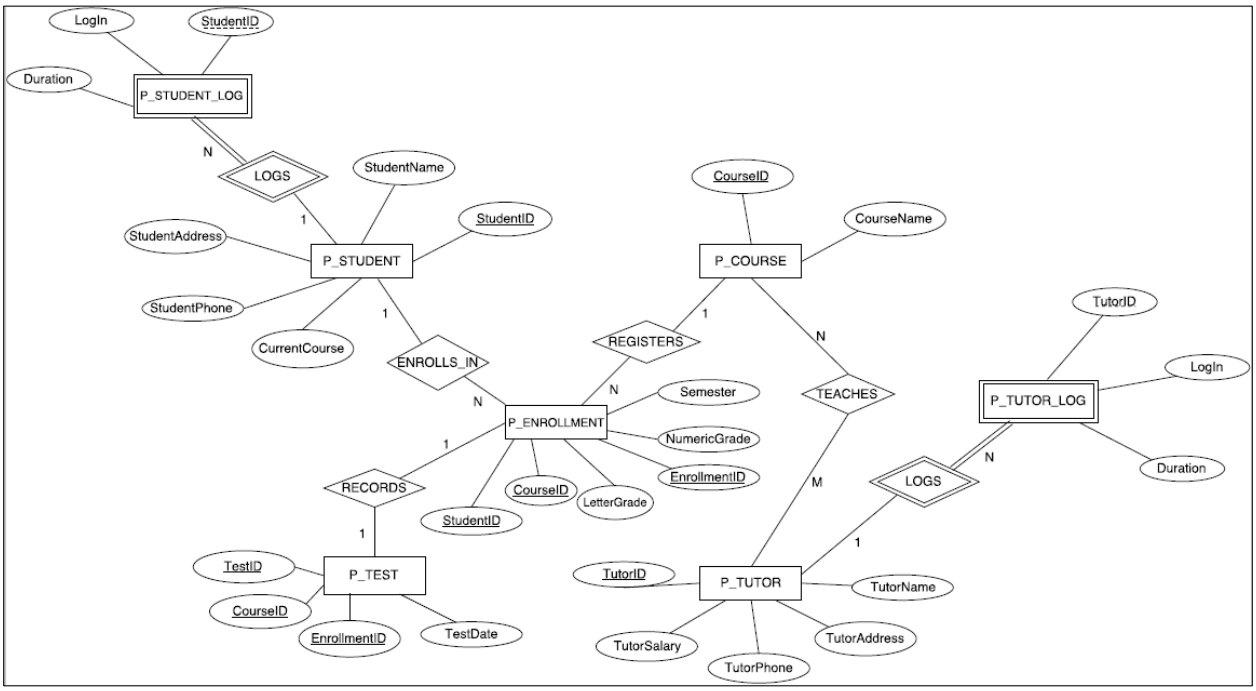
III/ Data Requirements:

- The center offers to tutor 4 courses (P_COURSE). Each course has its own unique course ID (CourseID), unique course name (CourseName), and the tutors (P_TUTOR) teaching (TEACHES) that course.
- The center's database also keeps track of the list of students (P_STUDENT) enrolled and their information. A student's information includes name (StudentName), unique student ID (StudentID), address (StudentAddress), phone number (StudentPhone), the course (CurrentCourse) he or she is enrolled in.
- In order to pass the course, students need to take a test (P_TEST) at the end of each course. The database keeps track of each test's unique ID (TestID), the course ID of the course which the test is on (CourseID), the enrollment ID (EnrollmentID) of the student taking the test, and the date (TestDate) on which the student takes the test.
- The tutors' (P_TUTOR) information includes the tutor's name (TutorName), unique id (TutorID), address (TutorAddress), phone number (TutorPhone), salary (TutorSalary).
- The database also keeps track of each student's enrollment (P_ENROLLMENT). Each student is given a unique enrollment ID (EnrollmentID), which is recorded in the database. The enrollment information of a student also includes the student's ID (StudentID), the semester (Semester) and course ID (CourseID) of the course the student is enrolled in. Also, when a student takes a test at the end of each course, the numeric grade (NumericGrade) and letter grade (LetterGrade) is also recorded in the enrollment information.
- In addition, every time a student comes and studies at the center, the student is required to enter into the system his or her student ID (StudentID) at the time he or she leaves the center (LogIn), and the number of hours spent studying (Duration) at the center. All of this information

will be stored under the student's log (P_STUDENT_LOG), which only exists if the student is given a student's ID and logs in at the center.

- The same information is also required from the tutors (TutorID, LogIn, and Duration) every time a tutor comes to work at the center and will be stored under the tutor's log (P_TUTOR_LOG).

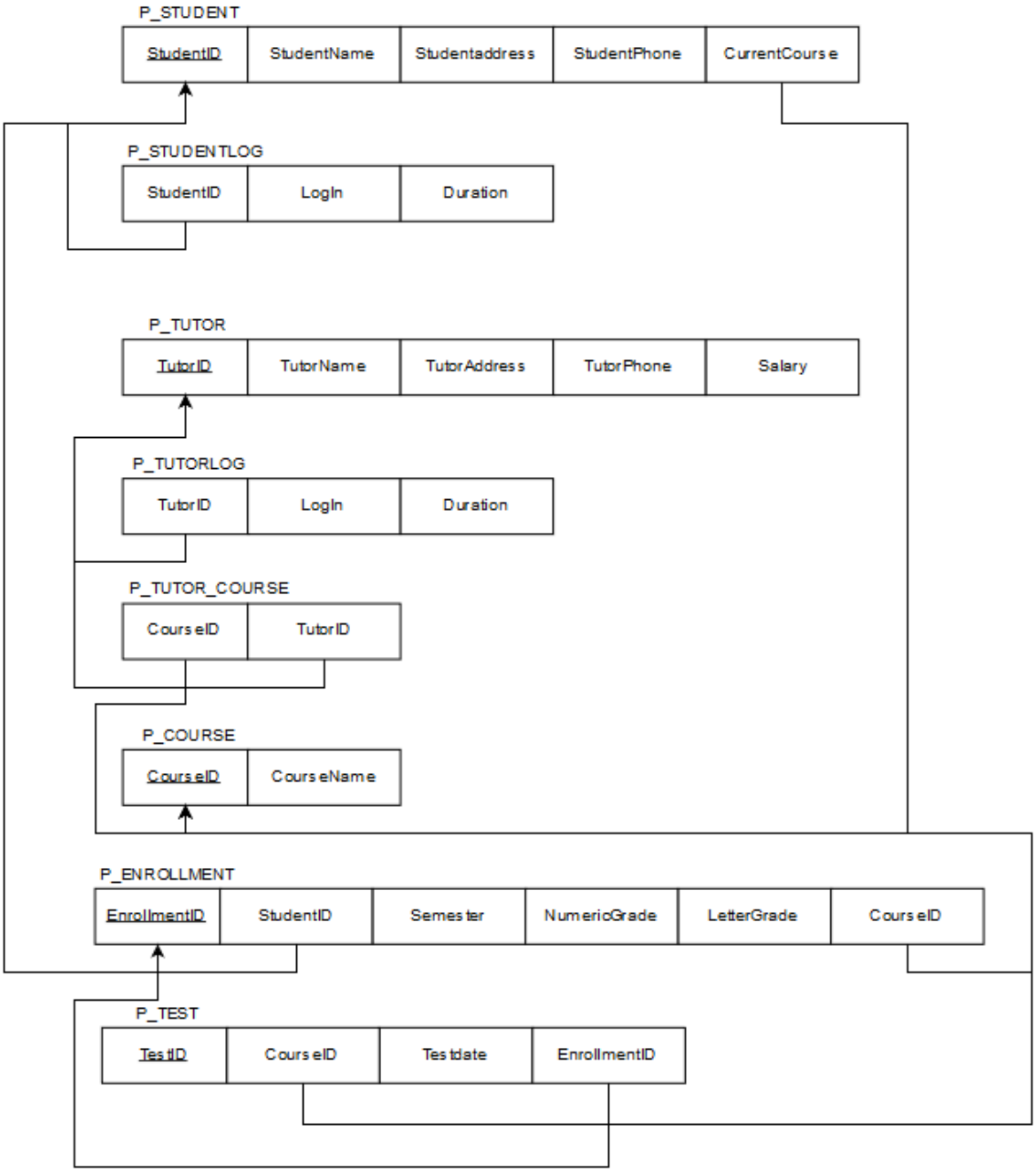
IV/ <u>ER Diagram</u>:



<u>Assumptions</u>:

- P_STUDENT_LOG does not exist without a Student ID from P_STUDENT.
- One student can log in his or her study hours many times.
- P_TUTOR_LOG does not exist without a Tutor ID from P_TUTOR.
- One tutor can log in his or her work hours many times.
- One student can be enrolled in many courses per semester.
- One student can only takes one test per course.
- One tutor can teach many courses.

V/ Relational Database Design:



VI/ <u>DDL schema creation in MySQL</u>:

(This group project's database was implemented using William Herrera's account: cs332u12).

```
mysql> CREATE TABLE P_COURSE(
    -> CourseID integer not null,
    -> PRIMARY KEY(CourseID),
    -> CourseName varchar(30) not null);

mysql> DESC P_COURSE;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| CourseID   | int(11)     | NO   | PRI |         |       |
| CourseName | varchar(30) | NO   |     |         |       |
+------------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)

-----------------------------------------------------------------------

mysql> CREATE TABLE P_STUDENT(
    -> StudentID integer not null,
    -> StudentName varchar(30) not null,
    -> StudentAddress varchar(255),
    -> StudentPhone int(10),
    -> CurrentCourse int,
    -> PRIMARY KEY(StudentID),
    -> FOREIGN KEY(CurrentCourse) REFERENCES P_COURSE(CourseID));
Query OK, 0 rows affected (0.03 sec)

mysql> DESC P_STUDENT;
+----------------+--------------+------+-----+---------+-------+
| Field          | Type         | Null | Key | Default | Extra |
+----------------+--------------+------+-----+---------+-------+
| StudentID      | int(11)      | NO   | PRI |         |       |
| StudentName    | varchar(30)  | NO   |     |         |       |
| StudentAddress | varchar(255) | YES  |     | NULL    |       |
| StudentPhone   | int(10)      | YES  |     | NULL    |       |
| CurrentCourse  | int(11)      | YES  | MUL | NULL    |       |
+----------------+--------------+------+-----+---------+-------+
5 rows in set (0.00 sec)

-----------------------------------------------------------------------

mysql> CREATE TABLE P_TUTOR(
    -> TutorID int not null,
    -> TutorName varchar(30) not null,
    -> TutorAddress varchar(255),
```

```
    -> TutorPhone int(10),
    -> TutorSalary int,
    -> PRIMARY KEY(TutorID));
Query OK, 0 rows affected (0.04 sec)

mysql> DESC P_TUTOR;
+--------------+--------------+------+-----+---------+-------+
| Field        | Type         | Null | Key | Default | Extra |
+--------------+--------------+------+-----+---------+-------+
| TutorID      | int(11)      | NO   | PRI |         |       |
| TutorName    | varchar(30)  | NO   |     |         |       |
| TutorAddress | varchar(255) | YES  |     | NULL    |       |
| TutorPhone   | int(10)      | YES  |     | NULL    |       |
| TutorSalary  | int(11)      | YES  |     | NULL    |       |
+--------------+--------------+------+-----+---------+-------+
5 rows in set (0.00 sec)

-----------------------------------------------------------------------

mysql> CREATE TABLE P_TUTOR_COURSE(
    -> CourseID int not null,
    -> TutorID int not null,
    -> FOREIGN KEY(CourseID) REFERENCES P_COURSE(CourseID),
    -> FOREIGN KEY(TutorID) REFERENCES P_TUTOR(TutorID));

mysql> DESC P_TUTOR_COURSE;
+----------+---------+------+-----+---------+-------+
| Field    | Type    | Null | Key | Default | Extra |
+----------+---------+------+-----+---------+-------+
| CourseID | int(11) | NO   | MUL |         |       |
| TutorID  | int(11) | NO   | MUL |         |       |
+----------+---------+------+-----+---------+-------+
2 rows in set (0.00 sec)

--------------------------------------------------------------------------

mysql> CREATE TABLE P_ENROLLMENT(
    -> EnrollmentID int not null,
    -> PRIMARY KEY(EnrollmentID),
    -> StudentID int not null,
    -> FOREIGN KEY(StudentID) REFERENCES P_STUDENT(StudentID),
    -> CourseID varchar(30) not null,
    -> FOREIGN KEY(CourseID) REFERENCES P_COURSE(CourseID),
    -> Semester date,
    -> NumericGrade int,
    -> LetterGrade char);
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> DESC P_ENROLLMENT;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| EnrollmentID | int(11)     | NO   | PRI |         |       |
| StudentID    | int(11)     | NO   | MUL |         |       |
| CourseID     | varchar(30) | NO   | MUL |         |       |
| Semester     | date        | YES  |     | NULL    |       |
| NumericGrade | int(11)     | YES  |     | NULL    |       |
| LetterGrade  | char(1)     | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

```
mysql> CREATE TABLE P_TEST(
    -> TestID int not null,
    -> PRIMARY KEY(TestID),
    -> CourseID int not null,
    -> EnrollmentID int not null,
    -> FOREIGN KEY(CourseID) REFERENCES P_COURSE(CourseID),
    -> FOREIGN KEY(EnrollmentID) REFERENCES P_ENROLLMENT(EnrollmentID),
    -> TestDate date);
Query OK, 0 rows affected (0.04 sec)

mysql> DESC P_TEST;
+--------------+---------+------+-----+---------+-------+
| Field        | Type    | Null | Key | Default | Extra |
+--------------+---------+------+-----+---------+-------+
| TestID       | int(11) | NO   | PRI |         |       |
| CourseID     | int(11) | NO   | MUL |         |       |
| EnrollmentID | int(11) | NO   | MUL |         |       |
| TestDate     | date    | YES  |     | NULL    |       |
+--------------+---------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

```
mysql> CREATE P_STUDENT_LOG(
    -> StudentID int not null,
    -> FOREIGN KEY(StudentID) REFERENCES P_STUDENT(StudentID),
    -> Login timestamp not null,
    -> Duration int);
Query OK, 3 rows affected (0.03 sec)
```

```
mysql> DESC P_STUDENT_LOG;
+-----------+-----------+------+-----+-------------------+-------+
| Field     | Type      | Null | Key | Default           | Extra |
+-----------+-----------+------+-----+-------------------+-------+
| StudentID | int(11)   | NO   | MUL |                   |       |
| Login     | timestamp | NO   |     | CURRENT_TIMESTAMP |       |
| Duration  | int(11)   | YES  |     | NULL              |       |
+-----------+-----------+------+-----+-------------------+-------+
3 rows in set (0.00 sec)
```

```
mysql> CREATE P_TUTOR_LOG(
    -> TutorID int not null,
    -> FOREIGN KEY(TutorID) REFERENCES P_TUTOR(TutorID),
    -> Login timestamp not null,
    -> Duration int);
Query OK, 3 rows affected (0.03 sec)

mysql> DESC P_TUTOR_LOG;
+----------+-----------+------+-----+-------------------+-------+
| Field    | Type      | Null | Key | Default           | Extra |
+----------+-----------+------+-----+-------------------+-------+
| TutorID  | int(11)   | NO   | MUL |                   |       |
| Login    | timestamp | NO   |     | CURRENT_TIMESTAMP |       |
| Duration | int(11)   | YES  |     | NULL              |       |
+----------+-----------+------+-----+-------------------+-------+
3 rows in set (0.00 sec)
```

VII/ <u>DML for Data Manipulation</u>:

1/ studentLogIn - Allows student to log in his or her student's ID and duration of studying time at the center, then adds the entered info into the database.

```
mysql> insert into P_STUDENT_LOG values (@studentID, now(), @duration);
```

2/ tutorLogIn - Allows tutor to log in his or her tutor's ID and duration of working time at the center, then adds the entered info into the database.

```
mysql> insert into P_TUTOR_LOG values (@tutorID, now(), @duration);
```

3/ getEnrolledStudents - Returns a list of all students currently enrolled in a specific course. The user will pass in a course ID, and the result will be a list of student names and their IDs.

```
mysql> select P_STUDENT.StudentName, P_STUDENT.StudentID
    -> from P_STUDENT, P_ENROLLMENT
    -> where P_STUDENT.StudentID=P_ENROLLMENT.StudentID
    -> and P_ENROLLMENT.CourseID = @userCourseID;
```

4/ getTutorInformation - Returns all the details of a tutor: ID, name, address, phone, salary, and teaching course. The tutor will enter a tutor ID, and the tutor's information will be returned.

```
mysql> select T.TutorID, T.TutorName, T.TutorAddress, T.TutorPhone,
    -> T.TutorSalary, C.CourseID, sum(L.Duration) as 'Hours'
    -> from P_TUTOR as T, P_TUTOR_COURSE as C, P_TUTOR_LOG as L
    -> where T.TutorID=11111111  and T.TutorID=C.TutorID and
    -> T.TutorID=L.TutorID
    -> group by T.TutorID, T.TutorName, T.TutorAddress, T.TutorPhone,
    -> T.TutorSalary, C.CourseID;
```

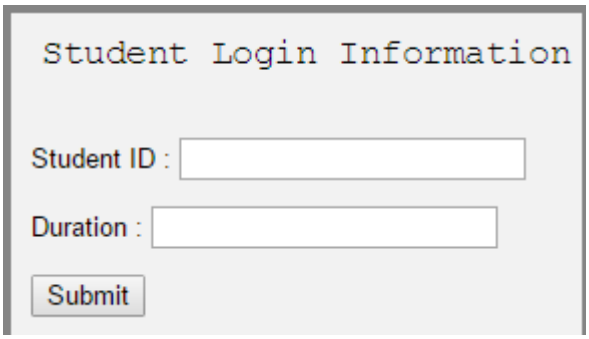
5/ getStudentHistory - Return a student's history at the tutoring center. The student ID will be entered, and information regarding the student's information, past enrollments, and study hours will be returned.

```
mysql> select S.StudentName, S.StudentID, S.StudentAddress, S.StudentPhone,
    -> S.CurrentCourse, E.EnrollmentID, E.Semester, E.NumericGrade,
    -> E.LetterGrade, sum(L.Duration) as 'Hours'
    -> from P_STUDENT as S, P_ENROLLMENT as E, P_STUDENT_LOG as L
    -> where S.StudentID=1 and S.StudentID=E.StudentID and
    -> S.StudentID=L.StudentID
    -> group by S.StudentName, S.StudentID, S.StudentAddress, S.StudentPhone,
    -> S.CurrentCourse, E.EnrollmentID, E.Semester, E.NumericGrade,
    -> E.LetterGrade;
```

VIII/ PHP Implementation:

Group Project's website: http://ecs.fullerton.edu/~cs332u12/index.php

1/ studentLogIn

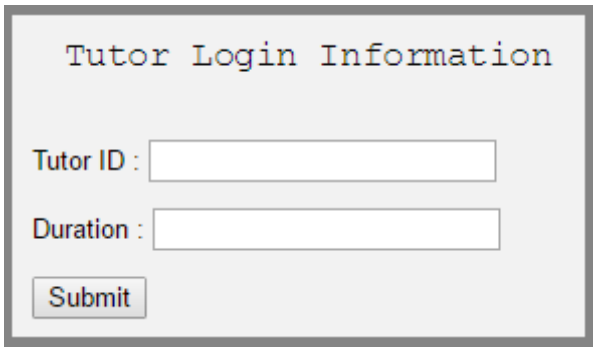


Student ID: 1

Duration: 9

Expected Result: New record created successfully

2/ tutorLogIn

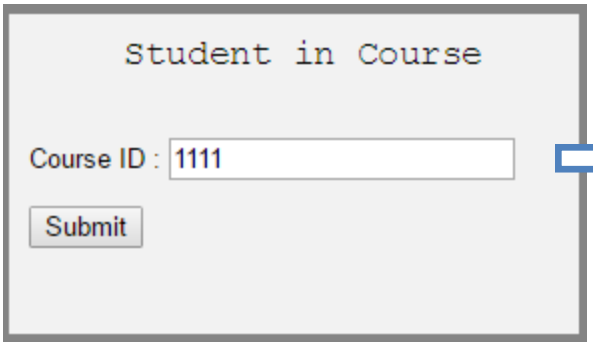


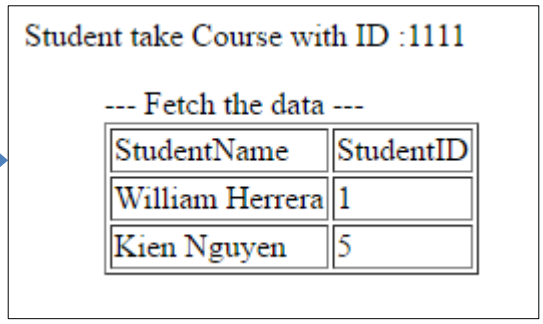
Tutor ID: 11111111

Duration: 2

Expected Result: New record created successfully

3/ getEnrolledStudents





4/ getTutorInformation

Tutor Information

Tutor ID : 11111111

Submit

Information of Tutor with ID :

--- Fetch the data ---

Tutor ID	Tutor Name	Tutor Address	Tutor Phone No.	Tutor Salary	Course ID
11111111	Barry Allen	731 Fonden, Houston, TX	858757656	30000	1111

Total Hours of Tutor: 8

5/ getStudentHistory

Get History of a Student

Student ID: 1

Submit

Get Student history with studentID : 1

--- Fetch the data ---

Enrollment ID	Student Name	Student ID	Current Course ID	Semester	Address	Numeric Grade	Letter Grade	Phone	Total Hours
111111	William Herrera	1	1111	2012-01-01	234 Stret Dr, Orange CA, 92345	1	D	1123452200	47