Objectives

By the end of the lab you will be able to:

- Join multiple tables
- Create a view
- Use triggers

Lab Assignment

1. (14 points, 2 points each)
   Given the structure and contents of the database that you built for courses, students, and registration of courses in lab2, write SQL code to answer questions below.

   You do not have to use the JOIN keyword in all of them for it to still count as a join, but you are welcome to use it. LEFT JOIN will make some of them easier.

(1) Which courses (course name, course credits) have been taken by the student with student ID 861103-2438?

(2) For students who have registered courses, how many credits has each student taken? In the query result, list student ID, student first name, student last name, and his/her credits.

(3) Define a view that gives the student ID, student last name, student first name, and the grade average for each student who has registered courses.

(4) Use the view in question (3) to find which students have the highest grade average.

(5) Which students (student first name, student last name) have taken the course "Database Systems"?

(6) List all the male students (student first name, student last name) who have taken both "Database Systems" and "C++".

(7) List all students (student ID, student first Name, student last Name) and the courses (course name, course grade) that they have taken. If there are students who have not taken any course yet, the query result should reflect that (i.e. null for courseName, null for grade).

2. (6 points, 2 points each)
A database for inventory and transaction of Apple store has been developed. There are three tables in this database – Inventory, Transaction, Inventory_history. You can use the following script to create the tables.

```
create table Inventory
(
 itemid varchar(20) primary key,
 name varchar(30),
 price decimal(6,2),
 quantity int
)ENGINE = innoDB;

create table Transaction
(
  transid int auto_increment primary key,
  itemid varchar(20),
  quantity int,
  time datetime,
  foreign key (itemid) references Inventory(itemid)
)ENGINE = innoDB;

create table Inventory_history
(
  id int auto_increment primary key,
  itemid varchar(20),
   action varchar(20),
   oldprice decimal(6,2),
   time datetime,
   foreign key (itemid) references Inventory(itemid)
)ENGINE = innoDB;
```

(1) Create a trigger “insert_inventory” on table “Inventory”. The trigger is fired after a row is inserted in table “Inventory”. After a row is inserted in table “inventory”, the “itemid”, the insertion time, and the action is inserted in table “Inventory_history”. The action is set to ‘add an item’. The oldprice is set to null.

(2) Create a trigger “change_quantity” on table “Transaction”. The trigger is fired after a row is inserted in table “Transaction”. After a row is inserted in table “Transaction”, update the

"quantity" in table "Inventory". For example, if 3 iWatch are sold, then the quantity of iWatch in table "Inventory" is decreased by 3.

(3) Create a trigger "change_price" on table "Inventory". The trigger is fired before a change is made to the "Inventory" table. Before the price of an item is changed, the "itemid", the item's old price, the action, and the time of change are inserted into the table "Inventory_history". The action is set to "price change".

<u>Submission</u>

Use Canvas to submit your SQL statements (in a Word file or .sql file or plain text file). Make sure you test your SQL statements on the database server "cssql.seattleu.edu" before submission. I will grade your SQL statements by running them on the server.