Last Name: First Name: Student ID:

1. [10pts] For all students majoring in ‘ART’ and having taken at least 4 classes offered by ‘CS’ dept, list their user_id, first_name, and last_name. Expected result row(s): 1

a) [7pts] SQL Query:

b) [3pts] Result:

2. [10pts] For each assignment for class CS122A (cno '122A' dept 'CS') in year 2017 and term Fall, list their assign_id, name, and the maximum score completed by students. Expected result row(s): 4

a) [7pts] SQL Query:

b) [3pts] Result:

3. [10pts] For all assignments which belong to class CS223 (cno '223' and dept 'CS') that have been completed by *at least 90% of* the students taking that class, list their assign_id, name, and how many students have completed it. Expected result row(s): 4

a) [7pts] SQL Query:

b) [3pts] Result:

4. **Views** [20 pts]

Congratulations! The CTO has formed a data science team to analyze the study activities of each student who is a **major in 'CS'**, and she has made you the head of that team. As the team leader, you have been asked to create a SQL view so that the rest of the team can look at the data and draw meaningful conclusions without having to deal with all of its underlying complexity.

The view should provide access to a combination of the following information:

- ❏ Student info (user_id, first_name, last_name)
- ❏ Take info (class_no, dept, cno, grade)
- ❏ Course info (title, level)

a) [15 pts] Create the desired view (CSStudentView) by writing an appropriate CREATE VIEW statement.

CREATE VIEW CSStudentView …;

b) [5 pts] Show the usefulness of your view by writing a SELECT query against the view that prints the user_id, first_name, last_name of the students who have the highest grade in the course titled 'Transaction Processing'. Expected result set: 1 row

5. **Stored Procedures** [20 pts]

a) [15 pts] Create and exercise a SQL stored procedure called RegisterStudent(…) that the site's application can use to add a new student with a mobile phone to the database.

```
DELIMITER //
CREATE PROCEDURE RegisterStudent(
        user_id varchar(20),
        email varchar(50),
        first_name varchar(30),
        last_name varchar(30),
        major varchar(50),
        mobile_number varchar(20))
    BEGIN
        ...
    END;  //
DELIMITER ;
```

b) [5pts] Verify that your new stored procedure works properly by calling it as follows to add a new student and then running a SELECT query to show the stored procedure's after-effects:

```
call RegisterStudent ('12345', 'johndoe@email.com', 'John','Doe', 'CS', '123-456-789');

select User.user_id, User.email, Student.major, Phone.number, Phone.type
from User, Student, Phone
where User.user_id = Student.user_id
AND User.user_id = Phone.user_id AND User.user_id = '12345';
```

Expected row is: 1 (Ignore null values of last row if it appears)

Result:

6. **Alter Table** [10 pts]

a) [5 pts] Write and execute the ALTER TABLE statement(s) needed to modify the Phone table so that whenever the user associated with the phone information is deleted then the phone data is deleted too. (Note: the name of the existing foreign key constraint for user_id is phone_ibfk_1).

b) [5 pts] Execute the following DELETE and SELECT statements to show the effect of your change. Write down the result returned by the SELECT statement.

DELETE from User where user_id = '12345';

SELECT * from Phone where Phone.user_id = '12345';

Result:

7. **Triggers** [20 pts]

a) [15 pt] Use the CREATE TRIGGER statement in MySQL to define a trigger that will do the desired job.

```
DELIMITER //

CREATE TRIGGER update_popularity       ...

END;//

DELIMITER ;
```

b) [5 pts] Execute the following INSERT and SELECT statements to show the effect of your trigger. Write down the popularity returned by each SELECT statement.

(Note: the first two selected popularity values may not differ by just 1 since the previous value is stale.)

```
SELECT popularity FROM Post where Post.post_id = '1';
```

Result:

```
INSERT INTO Likes VALUES('1', '1');

SELECT popularity FROM Post where Post.post_id = '1';
```

Result:

```
INSERT INTO Likes VALUES('5', '1');

SELECT popularity FROM Post where Post.post_id = '1';
```

Result: