CSC 343H1S 2013 Test 2
Duration — 50 minutes
Aids allowed: none

Student Number: |_|_|_|_|_|_|_|_|_|_|

Last Name: _____ **First Name:** _____

Lecture Section: Day **Instructor: Horton**

Please fill out the identification section above as well as the one on the back page, and read the instructions below.

Good Luck!

There is an API on the back page that will help you with the JDBC question.

You are welcome to use views.

Comments are not necessary in your SQL or Java code, but may help us mark your answer.

If you use any space for rough work or need to scratch out an answer, circle the part that you want us to mark.

1: _____/ 9

2: _____/12

3: _____/ 6

TOTAL: _____/27

Question 1. [9 MARKS]

Consider the following schema:

```
create table Department(
   dID integer primary key,
   name text not null,
   division integer);



create table Employee(
   eID integer primary key,
   name text not null,
   salary float,
   department integer references Department(dID));



-- Employee "manager" manages employee "junior".
create table Manages(
   manager integer references Employee(eID),
   junior integer references Employee(eID),
   primary key(manager, junior));



-- This employee sold this amount on this day.
create table Sales(
   eID integer references Employee(eID),
   day date,
   amount integer);
```

Part (a) [4 MARKS]

Imagine what would happen if someone were removed from the Employee table. Suppose that we want the following to occur automatically:

- Any tuple(s) that in the Manages table that record who manages this person should remain in the Manages table, but their `junior` column should become null.
- Any tuple(s) that in the Manages table that record who this person manages should be removed from the Manages table.

Revise the DDL above to implement this reaction policy.

Part (b) [3 MARKS]

Consider this query:

```
SELECT ________________________
FROM Employee natural join Sales
GROUP BY day, eID;
```

Which of the following can go in the blank space shown? Circle either "Can" or "Can't" for each.

`eID`	Can	Can't
`count(day)`	Can	Can't
`average(amount)`	Can	Can't
`name`	Can	Can't
`max(salary)`	Can	Can't
`department`	Can	Can't

Part (c) [1 MARK]

Write a SQL statement to increase the salary of every employee in the department with `dID` 212 by 10,000.

Part (d) [1 MARK]

According to the schema, must every employee have a manager? (No mark without an explanation.)

Circle one: YES NO

Explain your answer briefly

Question 2. [12 MARKS]

This question also uses the schema from question 1. It is summarized here in a very abbreviated fashion. Underscores indicate a primary key.

```
Department(_dID_, name, division)

Manages(_manager, junior_)
   manager references Employee(eID)
   junior references Employee(eID)

Employee(_eID_, name, salary, department)
   department references Department(dID)

Sales(eID, day, amount)
   eID references Employee(eID)
```

1. Write a query to produce output structured like this:

```
 managername | managersalary | juniorname | juniorsalary
-------------+---------------+------------+--------------
```

   Include in it a row for every pair of employees A and B such that A manages B, A's salary is more than twice as high as B's salary, B's total sales are more than 100,000, and B doesn't manage anyone.

2. Let's say that employee x is a "level 1 report" to employee y if y manages x, employee x is a "level 2 report" to employee y if y manages someone who manages x, and so on. Write a query to produce output structured like this:

```
 level | totalsales
-------+------------
```

It should have three rows: one for the total sales of all employees who are level 1 reports to "Marissa Mayer", one for the total sales of her level 2 reports, and one for the total sales of her level 3 reports. If she has no reports at a given level, `totalsales` for that level should be 0.

Question 3. [6 MARKS]

The following Java program connects to a database conforming to the schema from question 1, and then reports the employee IDs of everyone who reports to Marissa Mayer, everyone who reports to Bill Gates, and everyone who reports to Sheryl Sandberg. Complete the program. Assume all appropriate imports have been done. IMPORTANT: There is an API at the back of the test. Marking of Java syntax will not be strict.

```
class Company {
    public static void main(String args[]) throws IOException {
        String[] names = {"Marissa Mayer", "Bill Gates", "Sheryl Sandberg"};
        Connection conn;









        try {
            Class.forName("org.postgresql.Driver");
            // Assume a Connection to the database has been made and conn stores it.









            for (String who : names) {
                // Print the eID of everyone directly managed by who.
```

```
                }
                conn.close();
        } catch (Exception e) { System.err.println("uh oh!" + e.getMessage()); }
    }
}
```

[Use the space below for rough work. This page will not be marked unless you clearly indicate the part of your work that you want us to mark.]

Last Name: _____ **First Name:** _____

Abbreviated Java API:

```
Connection:
  void close()
    Releases this Connection object's database and JDBC resources immediately
  Statement createStatement()
    Creates a Statement object for sending SQL statements to the database.
  PreparedStatement prepareStatement(String sql)
    Creates a PreparedStatement object for sending parameterized SQL statements to the
    database.

PreparedStatement (extends Statement):
  boolean execute()
    Executes the SQL statement in this PreparedStatement object, which may be any kind of
    SQL statement.
  ResultSet executeQuery()
    Executes the SQL query in this PreparedStatement object and returns the ResultSet
    object generated by the query.
  void setInt(int parameterIndex, int x)
    Sets the designated parameter to the given Java int value.
  void setString(int parameterIndex, String x)
    Sets the designated parameter to the given Java String value.

ResultSet:
  void close()
    Releases this ResultSet object's database and JDBC resources immediately.
  int getInt(int columnIndex)
    Retrieves the int value of the designated column in the current row of this ResultSet.
  int getInt(String columnLabel)
    Retrieves the int value of the designated column in the current row of this ResultSet.
  String getString(int columnIndex)
    Retrieves the String value of the designated column in the current row of this
    ResultSet.
  String getString(String columnLabel)
    Retrieves the String value of the designated column in the current row of this
    ResultSet.
  boolean next()
    Moves the cursor froward one row from its current position.

Statement:
  boolean execute(String sql)
    Executes the given SQL statement, which may return multiple results.
  ResultSet executeQuery(String sql)
    Executes the given SQL statement, which returns a single ResultSet object.
```