The assignment is to be turned in before Midnight (by 11:59pm) on January 19th. You should turn in the solutions to this assignment as a PDF file through Canvas. The solutions should be produced using editing software programs, such as LaTeX or Word, otherwise they will not be graded. The assignment should be done in groups of two students. Each group must submit only one file that contains the full name, OSU email, and ONID of every member of the group.

1: Relational Query Languages and SQL (8 points)

Consider the following relational schema:
emp(*eid*:integer, *ename*:string, *age*:integer, *salary*:real)
works(*eid*:integer, *did*:integer, *pc_time*:integer)
dept(*did*:integer, *dname*:string, *budget*:real, *managerid*:integer)

The underlined attributes are keys for their relations. Note that a manager is an employee as well and their manager id and employee id are the same. An employee can work in more than one department. The pct_time field of the works relation shows the percentage of time that a given employee works in a given department and is always greater than zero.

A sample database (sample_db.sql) is provided with this assignment and the output of the correct queries over this sample database is given in each part of this question. You may use this sample database to test or debug your queries. We have created an account for each student on the MySQL server of our department. The access guide to the MySQL server (database_access_guide.txt) is also posted with this assignment. You can import the sample database to your account on the MySQL server and use it to ensure that your queries are correct. Notice that your SQL queries must return the correct result over every possible database instance of the aforementioned schema for the questions and not only over the sample database.

You should *not* submit any .sql file in your assignment submission and must write your final SQL queries in a single PDF file that you submit for all questions in this assignment.

Write the queries in parts (a) and (b) using relational algebra, Datalog, and SQL. Write the rest of queries only in SQL.

(a) Return names of every employee who works in the "Hardware", "Software", and "Research" departments. (1 point)

The answer on the sample database is:

ename
Shirish Ossenbruggen

(solution)
Relational Algebra:

$$\Pi_{ename}(\sigma_{dname=\text{"}Hardware\text{"}}(Emp \bowtie_{Emp.eid=Works.eid} Works \bowtie_{Works.did=Dept.did} Dept))\cap$$
$$\Pi_{ename}(\sigma_{dname=\text{"}Software\text{"}}(Emp \bowtie_{Emp.eid=Works.eid} Works \bowtie_{Works.did=Dept.did} Dept))\cap$$
$$\Pi_{ename}(\sigma_{dname=\text{"}Research\text{"}}(Emp \bowtie_{Emp.eid=Works.eid} Works \bowtie_{Works.did=Dept.did} Dept))$$

Datalog:

$$Q1(y)\text{:- } Emp(x,y,w,z), Works(x,p,q), Dept(p,\text{"}Hardware\text{"},s,t)$$

$$Q2(y) \text{:- } Emp(x,y,w,z), Works(x,p,q), Dept(p, "Software", s, t)$$
$$Q3(y) \text{:- } Emp(x,y,w,z), Works(x,p,q), Dept(p, "Research", s, t)$$
$$Q(y) \text{:- } Q1(y), Q2(y), Q3(y)$$

SQL:

```
SELECT ename
FROM emp E, works W1, works W2, works W3, dept D1, dept D2, dept D3
WHERE E.eid = W1.eid AND W1.did = D1.did AND D1.dname = "Hardware"
AND E.eid = W2.eid AND W2.did = D2.did AND D2.dname = "Software"
AND E.eid = W3.eid AND W3.did = D3.did AND D3.dname = "Research";
```

(b) Return the names of every department without any employee. (1.5 point)

The answer on the sample database is:

dname
Business Development

(solution)

Relational Algebra:

$$\Pi_{dname}(Dept) - \Pi_{dname}(Dept \bowtie_{Dept.did=Works.did} Works)$$

Datalog:

$$Q1(x,y,w,z) \text{:- } Dept(x,y,w,z), Works(s,x,t)$$
$$Q(y) \text{:- } Dept(x,y,w,z), \text{ not } Q1(x,y,w,z)$$

SQL:

```
SELECT D1.dname
FROM dept D1
WHERE D1.did NOT IN (
    SELECT D.did
    FROM emp E, works W, dept D
    WHERE E.eid = W.eid AND W.did = D.did );
```

(c) Print the *managerids* of managers who manage only departments with budgets greater than \$1.5 million. (1 point)

The answer on the sample database is:

managerid
110511

(solution)

```
SELECT DISTINCT D.managerid FROM dept D
WHERE 1500000 <
ALL (SELECT D2.budget FROM dept D2
WHERE D2.managerid = D.managerid)
```

(d) Print the name of employees whose salary is less than or equal to the salary of every employee. (0.5 point)

The answer on the sample database is:

ename
Antonio Lavante

(solution)

```
SELECT E1.ename
FROM emp E1
WHERE E1.salary <= ALL (SELECT E2.salary
FROM emp E2);
```

(e) Print the *enames* of managers who manage the departments with the largest budget. (1 point)

The answer on the sample database is:

ename
Tonny Butterworth

(solution)

```
SELECT E.ename FROM emp E WHERE E.eid IN
(SELECT D.managerid FROM dept D WHERE D.budget =
(SELECT MAX(D2.budget) FROM dept D2))
```

(f) Print the name of every department and the average salary of the employees of that department. The department must have a budget more than or equal to $50. (0.5 point)

The answer on the sample database is:

dname	average employee salary
Software	48291
Human Resources	717092.5
Research	490439.6666666667
Hardware	61842.125
Customer Service	40000

(solution)

```
SELECT D1.dname, Avg(E1.salary)
FROM dept D1, emp E1, works W1
WHERE E1.eid = W1.eid AND W1.did = D1.did AND D1.budget >= 50
Group By D1.did
```

(g) Print the *managerids* of managers who control the largest amount of total budget. As an example, if a manager manages two departments, the amount of total budget for him/her will be the sum of the budgets of the two departments. We want to find managers that have max total budget. (1 point)

The answer on the sample database is:

managerid
111692

(solution)

```
SELECT D.managerid from dept as D
Group By D.managerid HAVING SUM(D.budget) >= ALL
(SELECT SUM(budget) FROM dept GROUP BY managerid);
```

(h) Print the name of every employee whose salary is less than or equal to the average salary of all employees in his/her departments. (0.5 point)

The answer on the sample database is:

ename
Alex Dalas
Antonio Lavante
Tonny Conner
Shirish Ossenbruggen
DeForest Hagimont
Tonny Butterworth
Shigehito Kropatsch

(solution)

```
SELECT E1.ename
FROM emp E1
WHERE E1.salary <=
    (SELECT Avg(E2.salary)
    FROM emp E2, works W1, works W2
    WHERE E1.eid = W1.eid AND W1.did = W2.did AND E2.eid = W2.eid);
```

(i) Print the name of every employee who works only in the "Hardware" department. (1 point)

The answer on the sample database is:

ename
Alex Dalas
Sergio Ravarez
Antonio Lavante
Tonny Conner
Gladys Cooper
Rodney Ferreri
Arie Staelin

(solution)

```
SELECT E.ename
FROM emp E, works W, dept D
WHERE E.eid = W.eid AND W.did = D.did AND D.dname = 'Hardware'
AND NOT EXISTS ( SELECT *
    FROM works W2, dept D2
    WHERE E.eid = W2.eid AND W2.did = D2.did AND D2.dname <> 'Hardware');
```

2: Query Containment (2 points)

(a) Explain whether the homomorphism theorem holds for conjunctive queries with equality, e.g., $(T(s) :- R(x, y), x = y)$. If your answer is no, then modify the theorem to hold for these queries. (1 point)

Solution: The theorem does not hold. For example, consider the following queries: $Q_1(x) :- R(x, y, z), x = y, y = z$ and $Q_2(x) :- R(x, y, z), x = y, x = z$. They are equivalent but there is not homomorphism mapping between them. To fix this problem, one has to add new literals that are implied based on the symmetry and transitivity of equality. For example, we may modify the aforementioned clauses to:

$Q_1(x) :- R(x, y, z), x = y, y = x, y = z, z = y, x = z, z = x$ and $Q_2(x) :- R(x, y, z), x = y, y = x, x = z, z = x, y = z, z = y$.

(b) Explain whether the homomorphism theorem holds for conjunctive queries with $\leq$, e.g., $(T(s) :- R(x, y), x \leq y)$. (1 point)

Solution: It does not hold. For example, consider the queries: $Q_1(z) :- R(u, v, z), u \leq v$ and $Q_2(z) :- R(x, y, z), R(y, x, z)$. Since there is an order between y and x, e.g., they are real numbers, we have either $x \leq y$ or $y \leq x$. Thus, Q_2 is a subset of Q_1. However, there is not any homomorphism mapping between them.