# CS 540- Database Management Systems
# Assignment 1

Submitted by :

Name: Chiayu Tu , Chunghao Tuan
Email: tuchi@oregonstate.edu , tuanc@oregonstate.edu
ONID: tuchi tuanc

---

**Question**

Consider the following relational schema:

emp (<u>eid</u>: integer (a), ename: string (b), age: integer (c), salary: real (d))
works (<u>eid</u>: integer (a), <u>did</u>: integer (e), pc_time: integer (f))
dept (<u>did</u>: integer (e), dname: string (g), budge: real (h), managerid: integer (i))

---

**Q1. Return names of every employee who works in the "Hardware", "Software", and "Research" departments.**

**a) Datalog:**

Q1(b) : emp(a,b,c,d),works(a,e,f),dept(e,“Hardware”,h,i)

Q2(b) : emp(a,b,c,d),works(a,e,f),dept(e,“Software”,h,i)

Q3(b) : emp(a,b,c,d),works(a,e,f),dept(e,“Research”,h,i)

Q(b) : Q1(b),Q2(b),Q3(b)

**b) Relational Algebra:**

$$\pi_b\left(\sigma_{g=\text{"Hardware"}}\left(emp \bowtie_{emp.a=works.a} works \bowtie_{works.e=dept.e} dept\right)\right) \cap$$
$$\pi_b\left(\sigma_{g=\text{"Software"}}\left(emp \bowtie_{emp.a=works.a} works \bowtie_{works.e=dept.e} dept\right)\right) \cap$$
$$\pi_b\left(\sigma_{g=\text{"Research"}}\left(emp \bowtie_{emp.a=works.a} works \bowtie_{works.e=dept.e} dept\right)\right)$$

**c) Relational Calculus:**

Q(b) = ∃a,∃e,∃g (dept(e,g,h,i) ∧ (g = ”Hardware” ∨ g = ”Software” ∨ g = ”Research”) ∧ works(a,e,f) ∧ emp(a,b,c,d))

e) SQL

```
SELECT DISTINCT e.name

FROM emp e

JOIN works w ON e.eid = w.eid

JOIN dept d ON w.did = d.did

WHERE d.dname IN (“Hardware”, “Software”, “Research”)
```

---

**Q2. Return the names of every department without any employee.**

**a) Datalog:**

Q1(e, g, h, i) :- dept(e, g, h, i), works(a, e, f)
Q(b) :- dept(e, g, h, i), not Q1(e, g, h, i)

**b) Relational Algebra:**

$\pi_g(\text{dept}) - \pi_g(\text{dept} \bowtie_{\text{dept.e=works.e}} \text{works})$

**c) Relational calculus :**

Q(g) = ∃e,∃h,∃i (dept(e,g,h,i) ∧ ¬∃a,∃f (works(a,e,f)))

d) SQL

```
SELECT d.dname

FROM dept d

LEFT JOIN works w On d.did = w.did

WHERE w.eid IS NULL;
```

---

**Q.3 Print the managerid of managers who manage only departments with budgets greater than $1.5 million.**

**a) SQL**

```
SELECT d.managerid

FROM dept d

WHERE d.budget > 1500000;
```

---

**Q.4 Print the name of employees whose salary is less than or equal to the salary of every employee.**

**a) SQL**

```
SELECT e.ename

FROM emp e

WHERE e.salary <= ALL(SELECT salary FROM emp);
```

---

**Q.5 Print the enames of managers who manage the departments with the largest budget.**

**a) SQL**

```
SELECT e.ename

FROM emp e, dept d

WHERE e.eid = d.managerid

AND d.budget = (SELECT MAX(budget) FROM dept);
```

---

**Q6 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.**

**a) SQL**

```
SELECT d.dname, AVG(e.salary) AS avg_salary

FROM emp e

JOIN dept d ON e.eid = d.managerid

WHERE d.budget >= 50

GROUP BY d.dname;
```

---

**Q7 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.**

**a) SQL**

```
SELECT d.managerid

FROM dept d

GROUP BY d.managerid

HAVING SUM(d.budget) = (

    SELECT MAX(total_budget)

    FROM (

        SELECT SUM(budget) AS total_budget

          FROM dept

          GROUP BY managerid

    ) AS subquery

);
```

---

**Q8 Print the name of every employee who works only in the "Hardware" department.**

**a) SQL**

```sql
SELECT emp.ename

FROM emp

JOIN works ON emp.eid = works.eid

JOIN dept ON works.did = dept.did

WHERE dept.dname = 'Hardware'

GROUP BY emp.ename

HAVING COUNT(distinct works.did) = 1;
```

---

**Q9** Prove that non-recursive Datalog without negation and relational algebra with selection, projection, and Cartesian product operators express the same set of queries. In this question, we consider only the non-recursive Datalog without negation queries with a single rule. We also consider only the relational algebra queries that produce non-empty answers over at least one database instance. Theorem 4.4.8 in Alice Book provides a summary of this proof. You should complete this summary and submit your proof.

a) Prove that every non-recursive Datalog without negation query can be expressed in relational algebra with selection, projection, and Cartesian product operators.
   a. A non-recursive Datalog query without negation that has just one rule can be turned into relational algebra. The rule's head matches the output table. The body is like doing a series of steps: first, combine tables (Cartesian product), then choose rows (selection) based on matching values in repeated variables. Finally, pick the columns (projection) that are in the rule's head.

b) Prove that every non-recursive Datalog without negation query can be expressed in relational algebra with selection, projection, and Cartesian product operators.
   a. A query in relational algebra can be converted to a non-recursive Datalog query without negation. The selection step is like a rule with one condition in the body, and the variables in the selection match those in the rule. The projection step is also like a rule with one condition, where the chosen columns match the rule's variables. The Cartesian product is shown as a rule with two conditions in the body.

c) Combine the results of steps a and b.

    a. From combining steps 1 and 2, it's clear that non-recursive Datalog without negation and relational algebra using selection, projection, and Cartesian product can represent the same queries.

d) Non-recursive Datalog without negation and relational algebra with selection, projection, and Cartesian product operators express the same set of queries.