# Discussion #4

Diwanshu, Elena

July 1, 2020

# Logistics

- Take Midterm by EOD.
- Lab - 1 due July 5, 11:59 PM.

# Outline

- Practice SQL Queries

# Query 1

**BANK PROBLEM**

***Find the names, street address, and cities of residence for all employees who work for 'UWCU Bank' and earn more than $8,000.***

**Relational Schema**

employee(employee-name, street, city)
works(employee-name, company-name, salary)
company(company-name, city)
manages(employee-name, manager-name)

# Query 1

**BANK PROBLEM**

***Find the names, street address, and cities of residence for all employees who work for 'UWCU Bank' and earn more than $8,000.***

```
SELECT employee.employee-name,
employee.street, employee.city
FROM employee, works
WHERE
employee.employee-name=works.empl
oyee-name and company-name =
'UWCU' and salary > 8000)
```

**Relational Schema**

employee(employee-name, street, city)
works(employee-name, company-name, salary)
company(company-name, city)
manages(employee-name, manager-name)

# Query 2

**BANK PROBLEM**

***Find the names of all employees in the database who live in the same cities as the companies for which they work.***

**Relational Schema**

employee(employee-name, street, city)
works(employee-name, company-name, salary)
company(company-name, city)
manages(employee-name, manager-name)

# Query 2

**BANK PROBLEM**

***Find the names of all employees in the database who live in the same cities as the companies for which they work.***

```
SELECT e.employee-name
FROM employee e, works w, company c
WHERE e.employee-name =
w.employee-name and e.city = c.city
 and w.company-name =
c.company-name
```

**Relational Schema**

employee(employee-name, street, city)
works(employee-name, company-name, salary)
company(company-name, city)
manages(employee-name, manager-name)

# Query 3

**BANK PROBLEM**

***Find the names of all employees in the database who do not work for 'UWCU'. Assume that all people work for exactly one company.***

**Relational Schema**

employee(employee-name, street, city)
works(employee-name, company-name, salary)
company(company-name, city)
manages(employee-name, manager-name)

# Query 3

**BANK PROBLEM**

***Find the names of all employees in the database who do not work for 'UWCU'. Assume that all people work for exactly one company.***

```
SELECT employee-name
FROM works
WHERE company-name <> 'UWCU'
```

**Relational Schema**

employee(employee-name, street, city)
works(employee-name, company-name, salary)
company(company-name, city)
manages(employee-name, manager-name)

# Query 4

**BANK PROBLEM**

***Find the names of all employees in the database who earn more than every employee of 'UWCU'. Assume that all people work for at most one company.***

**Relational Schema**

employee(employee-name, street, city)
works(employee-name, company-name, salary)
company(company-name, city)
manages(employee-name, manager-name)

# Query 4

**BANK PROBLEM**

***Find the names of all employees in the database who earn more than every employee of 'UWCU'. Assume that all people work for at most one company.***

```
SELECT employee-name
FROM works
WHERE salary > all
      (SELECT salary
      FROM works
      WHERE company-name = 'UWCU')
```

**Relational Schema**

employee(employee-name, street, city)
works(employee-name, company-name, salary)
company(company-name, city)
manages(employee-name, manager-name)

# Query 5

**BANK PROBLEM**

***Assume that the companies may be located in several cities. Find all companies located in every city in which 'UWCU' is located.***

**Relational Schema**

employee(employee-name, street, city)
works(employee-name, company-name, salary)
company(company-name, city)
manages(employee-name, manager-name)

# Query 5

**BANK PROBLEM**

***Assume that the companies may be located in several cities. Find all companies located in every city in which 'UWCU' is located.***

```
SELECT s.company-name
FROM company s
WHERE NOT EXISTS (
(SELECT city
FROM company
WHERE company-name = 'UWCU')
EXCEPT
(SELECT city
FROM company t
WHERE s.company-name =
t.company-name)
)
```

**Relational Schema**

employee(employee-name, street, city)
works(employee-name, company-name, salary)
company(company-name, city)
manages(employee-name, manager-name)

# Query 6

**BANK PROBLEM**

***Find the names of all employees who earn more than the average salary of all employees of their company. Assume that all people work for at most one company.***

**Relational Schema**

employee(employee-name, street, city)
works(employee-name, company-name, salary)
company(company-name, city)
manages(employee-name, manager-name)

# Query 6

**BANK PROBLEM**

***Find the names of all employees who earn more than the average salary of all employees of their company. Assume that all people work for at most one company.***

```
SELECT employee-name
FROM works t
WHERE salary >
(SELECT avg(salary)
FROM works s
WHERE t.company-name =
s.company-name)
```

**Relational Schema**

employee(employee-name, street, city)
works(employee-name, company-name, salary)
company(company-name, city)
manages(employee-name, manager-name)

# Query 7

**BANK PROBLEM**

***Find the name of the company that has the smallest payroll.***

**Relational Schema**

employee(employee-name, street, city)
works(employee-name, company-name, salary)
company(company-name, city)
manages(employee-name, manager-name)

# Query 7

**BANK PROBLEM**

***Find the name of the company that has the smallest payroll.***

```
SELECT company-name
FROM works
GROUP BY company-name
HAVING sum(salary) <= ALL
(SELECT sum(salary)
 FROM works
 GROUP BY company-name)
```

**Relational Schema**

employee(employee-name, street, city)
works(employee-name, company-name, salary)
company(company-name, city)
manages(employee-name, manager-name)

## Relational Schema

## AIRLINE PROBLEM

Flights(flno: integer, from: string, to: string, distance: integer, departs: time, arrives: time, price: integer)
Aircraft(aid: integer, aname: string, cruisingrange: integer)
Certified(eid: integer, aid: integer)
Employees(eid: integer, ename: string, salary: integer)

***Find the names of aircraft such that all pilots certified to operate them earn more than $80,000.***

```
SELECT  DISTINCT  A.aname
FROM    Aircraft A
WHERE   A.Aid IN  (SELECT C.aid
                   FROM Certified C, Employees E
                   WHERE C.eid = E.eid AND
                   NOT EXISTS  ( SELECT *
                                 FROM Employees E1
                                 WHERE E1.eid = E.eid AND E1.salary < 80000 ))
```

## Relational Schema

## AIRLINE PROBLEM

Flights(flno: integer, from: string, to: string, distance: integer, departs: time, arrives: time, price: integer)
Aircraft(aid: integer, aname: string, cruisingrange: integer)
Certified(eid: integer, aid: integer)
Employees(eid: integer, ename: string, salary: integer)

***For each pilot who is certified for more than three aircraft, find the eid and the maximum cruisingrange of the aircraft for which she or he is certified.***

```
SELECT      C.eid, MAX (A.cruisingrange)
FROM        Certified C, Aircraft A
WHERE       C.aid = A.aid
GROUP BY    C.eid
HAVING      COUNT (*) > 3
```

## Relational Schema

## AIRLINE PROBLEM

Flights(flno: integer, from: string, to: string, distance: integer, departs: time, arrives: time, price: integer)
Aircraft(aid: integer, aname: string, cruisingrange: integer)
Certified(eid: integer, aid: integer)
Employees(eid: integer, ename: string, salary: integer)

***Find the names of pilots whose salary is less than the price of the cheapest route from Los Angeles to Honolulu***

```
SELECT   DISTINCT   E.ename
FROM     Employees E
WHERE    E.salary <     ( SELECT MIN (F.price)
                        FROM Flights F
                        WHERE F.from =   'Los Angeles'   AND F.to =   'Honolulu'   )
```

## Relational Schema

## AIRLINE PROBLEM

Flights(flno: integer, from: string, to: string, distance: integer, departs: time, arrives: time, price: integer)
Aircraft(aid: integer, aname: string, cruisingrange: integer)
Certified(eid: integer, aid: integer)
Employees(eid: integer, ename: string, salary: integer)

***For all aircraft with cruisingrange over 1000 miles, find the name of the aircraft and the average salary of all pilots certified for this aircraft.***

Observe that aid is the key for Aircraft, but the question asks for aircraft names; we deal with this complication by using an intermediate relation Temp:

```
SELECT   Temp.name, Temp.AvgSalary
FROM     ( SELECT   A.aid, A.aname AS name, AVG (E.salary) AS AvgSalary
           FROM     Aircraft A, Certified C, Employees E
           WHERE    A.aid = C.aid AND C.eid = E.eid AND A.cruisingrange > 1000
           GROUP BY A.aid, A.aname ) AS Temp
```

## Relational Schema

## AIRLINE PROBLEM

Flights(flno: integer, from: string, to: string, distance: integer, departs: time, arrives: time, price: integer)
Aircraft(aid: integer, aname: string, cruisingrange: integer)
Certified(eid: integer, aid: integer)
Employees(eid: integer, ename: string, salary: integer)

***Find the names of pilots certified for some Boeing aircraft***

```
SELECT   DISTINCT  E.ename
FROM     Employees E, Certified C, Aircraft A
WHERE    E.eid = C.eid AND C.aid = A.aid AND A.aname LIKE  'Boeing%'
```

## Relational Schema

## AIRLINE PROBLEM

Flights(flno: integer, from: string, to: string, distance: integer, departs: time, arrives: time, price: integer)
Aircraft(aid: integer, aname: string, cruisingrange: integer)
Certified(eid: integer, aid: integer)
Employees(eid: integer, ename: string, salary: integer)

***Find the aids of all aircraft that can be used on routes from Los Angeles to Chicago.***

```
SELECT  A.aid
FROM    Aircraft A
WHERE   A.cruisingrange > ( SELECT  MIN (F.distance)
                            FROM    Flights F
                            WHERE   F.from = 'Los Angeles'  AND F.to = 'Chicago' )
```

## Relational Schema

## AIRLINE PROBLEM

Flights(flno: integer, from: string, to: string, distance: integer, departs: time, arrives: time, price: integer)
Aircraft(aid: integer, aname: string, cruisingrange: integer)
Certified(eid: integer, aid: integer)
Employees(eid: integer, ename: string, salary: integer)

***Identify the routes that can be piloted by every pilot who makes more than $100,000.***

```
SELECT   DISTINCT  F.from, F.to
FROM     Flights F
WHERE NOT EXISTS  ( SELECT  *
                    FROM     Employees E
                    WHERE    E.salary > 100000
                    AND
                    NOT EXISTS  (SELECT  *
                                 FROM     Aircraft A, Certified C
                                 WHERE    A.cruisingrange > F.distance AND
                                          E.eid = C.eid AND A.aid = C.aid) )
```

## Relational Schema

# AIRLINE PROBLEM

Flights(flno: integer, from: string, to: string, distance: integer, departs: time, arrives: time, price: integer)
Aircraft(aid: integer, aname: string, cruisingrange: integer)
Certified(eid: integer, aid: integer)
Employees(eid: integer, ename: string, salary: integer)

***Print the enames of pilots who can operate planes with cruisingrange greater than 3000 miles but are not certified on any Boeing aircraft.***

```
SELECT  DISTINCT  E.ename
FROM    Employees E
WHERE   E.eid IN  ( ( SELECT C.eid
                      FROM Certified C
                      WHERE EXISTS  ( SELECT A.aid
                                      FROM Aircraft A
                                      WHERE A.aid = C.aid
                                      AND A.cruisingrange > 3000 )
                                      AND
                                      NOT EXISTS ( SELECT A1.aid
                                                   FROM Aircraft A1
                                                   WHERE A1.aid = C.aid
                                                   AND A1.aname LIKE  'Boeing%'  ))
```

## Relational Schema

## AIRLINE PROBLEM

Flights(flno: integer, from: string, to: string, distance: integer, departs: time, arrives: time, price: integer)
Aircraft(aid: integer, aname: string, cruisingrange: integer)
Certified(eid: integer, aid: integer)
Employees(eid: integer, ename: string, salary: integer)

***Compute the difference between the average salary of a pilot and the average salary of all employees (including pilots).***

```
SELECT  Temp1.avg - Temp2.avg
FROM    (SELECT   AVG (E.salary) AS avg
         FROM     Employees E
         WHERE    E.eid IN (SELECT   DISTINCT C.eid
                            FROM     Certified C )) AS Temp1,
         (SELECT  AVG (E1.salary) AS avg
         FROM     Employees E1 ) AS Temp2
```

**Relational Schema**

**AIRLINE PROBLEM**

Flights(flno: integer, from: string, to: string, distance: integer, departs: time, arrives: time, price: integer)
Aircraft(aid: integer, aname: string, cruisingrange: integer)
Certified(eid: integer, aid: integer)
Employees(eid: integer, ename: string, salary: integer)

***Print the name and salary of every nonpilot whose salary is more than the average salary for pilots.***

```
SELECT  E.ename, E.salary
FROM    Employees E
WHERE   E.eid NOT IN ( SELECT  DISTINCT  C.eid
                       FROM    Certified C )
AND E.salary > ( SELECT AVG (E1.salary)
                 FROM    Employees E1
                 WHERE   E1.eid IN
                         ( SELECT  DISTINCT  C1.eid
                           FROM    Certified C1 ) )
```

## Relational Schema

# AIRLINE PROBLEM

Flights(flno: integer, from: string, to: string, distance: integer, departs: time, arrives: time, price: integer)
Aircraft(aid: integer, aname: string, cruisingrange: integer)
Certified(eid: integer, aid: integer)
Employees(eid: integer, ename: string, salary: integer)

***Challenge Question: A customer wants to travel from Madison to New York with no more than two changes of flight. List the choice of departure times from Madison if the customer wants to arrive in New York by 6 p.m.***

Thanks!