Note :This is a bonus assignment worth 5% of the total grade. Your submission should be a pdf fil with all the solutions
Due Date: 3rd December 11:59 PM

# Transactions

**Q1:** Specify which of the following schedules could be generated by

a) Two phase locking scheduler
b) Rigorous two-phase locking scheduler

If the schedule could be generated by 2PL, specify when transactions would acquire locks and release locks.
If not, explain why the lock request will fail and hence the schedule would not be generated.

1. w_1 (x) w_2(x) w_1(y) w_2(y) c_1 c_2

| | | | | |
|---|---|---|---|---|
| T1 | w(x) | w(y) | c1 | |
| T2 | w(x) | w(y) | c2 | |

| | | | | | |
|---|---|---|---|---|---|
| T1 | $x(x), x(y)$ w(x)$R(x)$ | w(y) $R(y)$ | c1 | | |
| T2 | | $x(x)$ w(x) | $x(y)$w(y) | c2 $R(x)$ $R(y)$ | |

2PL can be done if we acquire shared lock on x and y in the beginning of the transaction itself
Rigorous 2PL isn't applicable as we need to release locks in between transactions

2. w_1(x) w_2(x) w_2(y) w_1(y) c_1 c_2

| | | | | |
|---|---|---|---|---|
| T1 | w(x) | w(y) | c1 | |
| T2 | w(x) | w(y) | c2 | |

| | | | | | |
|---|---|---|---|---|---|
| T1 | $x(x)$ w(x) $R(x)$ | | | $x(y)$w(y) | c1 $R(y)$ |
| T2 | | $x(x)$ w(x) | $x(y)$w(y) $R(y)$ | | c2$R(x)$ $R(y)$ |

2 PL is not possible because T1 is acquiring and then releasing lock on x, then it again acquires and releases lock on y.
Even if we acquire lock on x and y at start, we have to release lock on x and y for T2 and again acquire it for T1. Similarly Rigorous 2PL is also not possible

3. w_1(x) r_2(x) w_3(y) w_2(y) c_1 c_2 c_3

| | | | | |
|---|---|---|---|---|
| T1 | w(x) | | | c1 |
| T2 | | r(x) | w(y) | c2 |
| T3 | | | w(y) | c3 |

| | | | | |
|---|---|---|---|---|
| T1 | x(x)w(x) R(x) | | | c1 |
| T2 | | s(x) r(x) | X(y)w(y) | c2 R(x)R(y) |
| T3 | | | x(y)w(y) R(y) | c3 |

2PL is possible because we are acquiring locks first and then releasing them. No transaction acquires lock without releasing.
Also rigorous 2PL is not possible because T1 has to release the lock on X before committing

# Indexing

Q2: Consider a table employee(E_id, name, manager, department, Salary). Suppose it has an index on Salary.
Given a query “select * from employee where Salary < 100”, find out the times taken to do a linear scan and an index scan and satte which is better.
There are 1 billion records on the disk and each record is of size 512B. The block size is 8KB. Rotational latency is 10 ms and disk transfer rate is 50MB/s. The query selectivity is 10%.Time to do a sequential scan of the index is 60 seconds. Assume that to do a linear scan, you need to read entire file.

Linear scan -
1 billion records = 10^9 records
each record size = 512B
Total size = 512 * 10^9 B
disk transfer rate 50MB/s = 50 * 10^6 B
Total time taken for linear scan = (512 * 10^9) / 50 * 10^6 = 10240 s + rotational latency (10 ms) = approx equal to 10240 seconds

Index Scan -
size of 1 block = 8kb = 8000B
disk transfer rate 50MB/s = 50 * 10^6 B

Time to scan one block = 10 ms + (8000 B) / (50 * 10^6 B/s)
= 10.16 ms
query selectivity = 10%,
no. of matching records = 10% of 10^9 = 10^8

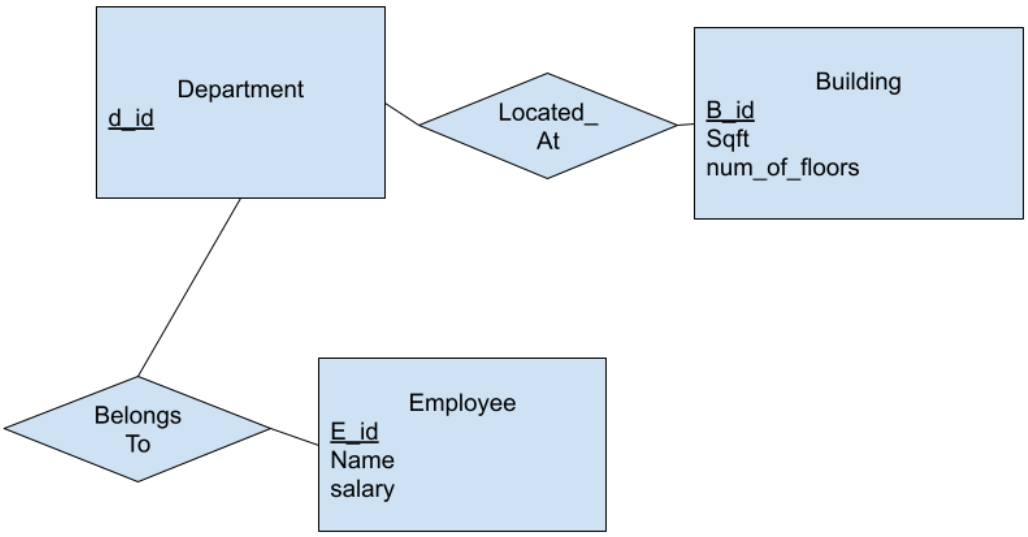
Time for index scan =  time to do sequential scan of the index + cost of dereferencing pointers for each matched record.

Time for index scan = 60 s + 10.16 ms * 10^8 = 1016060 seconds

**Hence sequential scan is better here**

# RDF

Q3: Consider an ER diagram




Instances of the entities are as follows:-
Department -- CS, EECS, Mechanical
Employee -- Alice (CS),  Bob (EECS), Carol (Mech), Dennis (CS)
Buildings - DBH (CS), ETower (EECS), EHall (MEch)

1. Represent the above using RDF triples. Manufacture any information that is not provided.

   cs instance-of Department
   cs d_id 1001
   EECS instance-of Department
   EECS d_id 1002
   Mechanical instance-of Department
   Mechanical d_id 1003

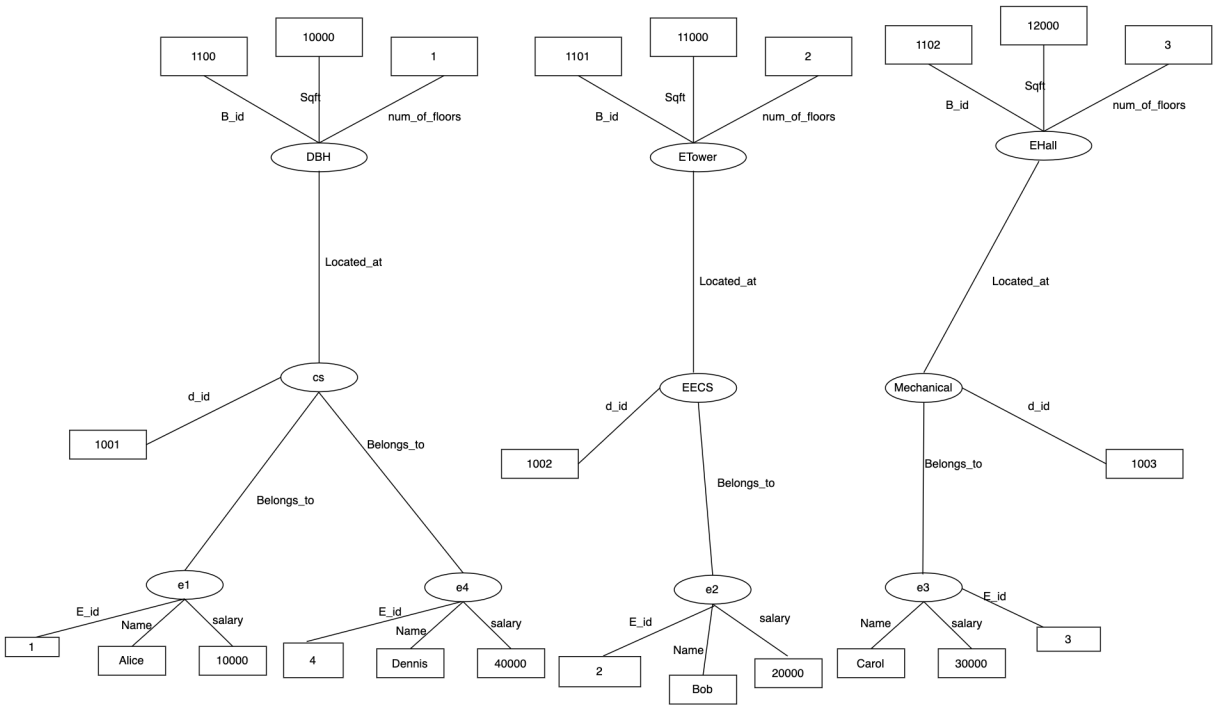
   e1 instance-of Employee
   e1 E_id 1
   e1 Name Alice
   e1 salary 10000
   e2 instance-of Employee
   e2 E_id 2
   e2 Name Bob
   e2 salary 20000
   e3 instance-of Employee
   e3 E_id 3
   e3 Name Carol
   e3 salary 30000
   e4 instance-of Employee
   e4 E_id 4
   e4 Name Dennis
   e4 salary 40000

   DBH instance-of Building
   DBH B_id 1100
   DBH Sqft 10000
   DBH num_of_floors 1
   ETower instance-of Building
   ETower B_id 1101
   ETower Sqft 11000
   ETower num_of_floors 2
   EHall instance-of Building
   EHall B_id 1102
   EHall Sqft 12000
   EHall num_of_floors 3

   e1 Belongs_to CS
   e2 Belongs_to EECS
   e3 Belongs_to Mechanical
   e4 Belongs_to CS

CS Located_at DBH
EECS Located_at ETower
Mechanical Located_at EHall

2. Draw the graph view of the RDF data.




3. Find the SparQL query for the following: Retrieve the number of floors of the building Alice works in.

```
select ?num_of_floors
where{
?student  Name  Alice
?dept  Belogs_to  ?student
?building  Located_at  ?dept
?building  num_of_floors  ?num_of_floors
}
```