Q4. (15 pts.) Consider the execution schedule H below. The symbol ri(x) stands for a read by transaction Ti to item x and wi(x) stands for a write by Ti to item x. Suppose Basic Timestamp Ordering (TSO) is used as the concurrency control protocol. Suppose furthermore that transaction T1 has timestamp 10, T2 has timestamp 11 and T3 has timestamp 12. Describe what happens during each operation below, justifying whether the operation is accepted or rejected, and showing how the RTS and WTS timestamps of the data items are updated in each step.

You can assume that the initial values for RTS(X) and WTS(X) are 0. Similarly, you can assume that the initial values for RTS(Y) and WTS(Y) are 0.

H: $r_1(x)\ r_3(x)\ w_2(x)\ r_3(y)\ r_1(y)\ w_3(y)\ w_1(y)\ w_2(y)$

Note: *If an access is rejected, its parent transaction is aborted; so you can ignore (remove from the schedule) all the subsequent accesses by that transaction)*

T1 : 10
T2 : 11
T3 : 12

Operation	X		Y	
	RTS	WTS	RTS	WTS
$r_1(x)$	10	0	0	0
$r_3(x)$	12	0	0	0
$w_2(x)$ rejected	12	0	0	0
$r_3(y)$	12	0	12	0
$r_1(y)$	12	0	12	0
$w_3(y)$	12	0	12	12
$w_1(y)$ rejected	12	0	12	12

T2 aborts

Q5. (25 pts.) Consider two transactions T1 and T2 accessing data items A, B, C in a multiuser database system. Assume that the system uses **Strict Two Phase Locking** with deadlock detection policy in conjunction with the crash recovery system which uses NO FORCE and STEAL policies. Assume there are only two buffer pages available. Show the contents of disk, database buffer and log as well as pages moving back and forth between memory and disk for the following schedule of the transactions:

T1	T2
R1(A)	
	R2(A)
W1(A)	
	W2(B)
	R2(C)
	W2(C)
	Commit T2
R1(C)	
W1(C)	
CRASH	

Write your answer in the following table:

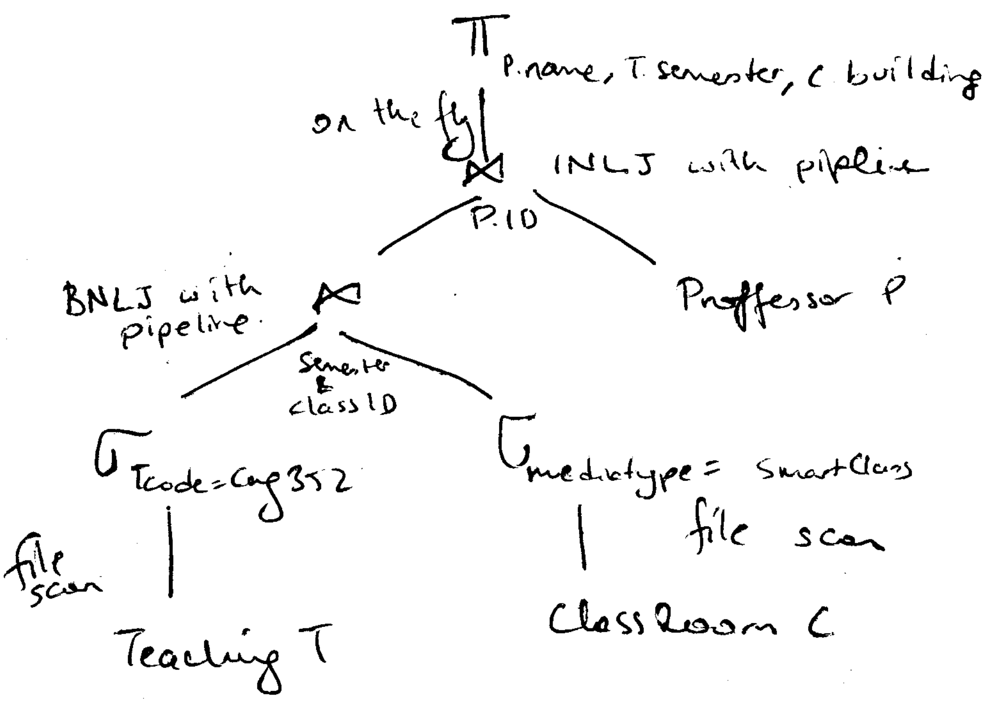
Operation	Locks A	Locks B	Locks C	Buffer	Log	Disk A	Disk B	Disk C
R1(A)	S_1	—	—	A^0		A^0	B^0	C^0
R2(A)	S_1, S_2	—	—	A^0				
W1(A)	S_1, S_2 T_1 waits for T2	—	—	A^0				
W2(B)	S_1, S_2	X_2	-	A^0 B^1	10 T2 B^0 B^1	A^0	B^0	C^0
R2(C)	S_1, S_2	X_2	S_2	C^0 B^1		A^0	B^0	C^0
W2(C)	S_1, S_2	X_2	X_2	C^1 B^1	20 T2 C^0 C^1 25 commit T2			
commit T2 W1(A)	S_1 X_1	— —	— —	C^1 A^1	flushed LSN 20 30 end T2 40 T1 A^0 A^1	A^0	B^1	C^0
R1(C)	X_1	—	S_1	C^1 A^1				
W1(C)	X_1	—	X_1	C^2 A^1	50 T1 C^1 C^2	A^0	B^1	C^0
CRASH	!							

Q2) (20 pts.) Consider the following relations Professor, Teaching and ClassRoom:

Professor (P_ID, Name, Dept_ID)
Teaching(P_ID, Code, Semester, ClassID)
ClassRoom(ClassID, Semester, Building, MediaType)

a) (*8 pts.*) Given the following SQL query, draw the query plan based on a leftdeep tree where all selections are done as early as possible, a block-nested loop join (BNLJ) for the result of the selections and an index nested loops join strategy afterwards, knowing that the relation Professor is indexed on P_ID. In order to decide which relation is the outer for the BNLJ, consider that there are about 30 semesters in the database and ceng352 is at most taught twice a semester. However, there are several hundreds of smart classrooms (they wouldn't fit in main memory).

```
SELECT   P.Name, T.Semester, C. Building
FROM     Professor P, Teaching T, ClassRoom C
WHERE    P.P_ID = T.P_ID AND C.Semester=T.Semester AND
         T.ClassID=C.ClassID AND C.MediaType = "SmartClass" AND
         T.Code = "CENG352"
```



(5 points for the query tree, 3 points for the plan)

b) (*12 pts.*) Suppose we have 10 buffer pages (blocks) in main memory, and assuming a uniform distribution for all the values in the database (except for classroom mediatype, where 60% of classrooms are smart classrooms), estimate the query execution I/O cost for the above plan.

Relations	# of tuples in the relation	# of tuples per page	
Professor	200	8	→ 25 page
Teaching	25,000	10	2500 pages
ClassRoom	1,000	10	100 pages

There is no index on attribute Code for the relation Teaching and the relation is neither clustered nor sorted on this attribute. Professor is indexed on P_ID using a clustered hash index. All CENG352 teachings are in smart-classrooms.

$\sigma_{code=352}$(Teaching) : File Scan 2500 blocks.

352 is taught at most 60 times

60 records ⇒ 6 pages (fit in memory)

$\sigma_{mediatype = Smart\,Class}$(ClassRoom) ⇒ File Scan 100 blocks

60% satisfy selection ⇒ 60 blocks (can write them in a temp file ∴ additional 60 blocks)

BNLJ : scan temp , 60 blocks.

Result = 60 records (assume they fit in memory)

INLJ : read Professor using index 60 times

$60 \times 1.2 = 72$ I/Os.

Total cost = 2500 + 100 + 60 + 60 + 72 = 2792 I/Os.
(with materialized selection)

Total cost = 2500 + 100 + 72 = 2672 I/Os
(pipelining)

. **(20 pts.) Query Processing**

Consider a relation **s** over the attributes A and B with the following characteristics:

- 7,000 tuples with 70 tuples per page
- A hash index on attribute A
- The values that the attribute A takes in relation s are integers that are uniformly distributed in the range 1 – 200.

a) Assuming that the index on *A* is *unclustered,* estimate the number of disk accesses needed to compute the query $\sigma_{A=18}(s)$.

$\frac{7000}{200} = 35$ records are selected for $A=18$

1.2 I/o to find the right bucket $\Rightarrow$ cost $= \lceil 1.2 + 35 \rceil = 37$ I/Os.

b) What would be the cost estimate if the index were clustered? Explain your reasoning.

$\lceil 1.2 + 2 \rceil = 4$ (upper bound)

ii) Compute the cost of **r** $\bowtie_{A=B}$ **s** using the following methods:

(a) Block-nested loops join

(b) Index-nested loops join with a hash index on *B* in **s**. (Do the computation for both clustered and unclustered index.)

where **r** occupies 2,000 pages, 20 tuples per page, **s** occupies 5,000 pages, 5 tuples per page, and the amount of main memory available for block-nested loops join is 402 pages. Assume that at most 5 tuples in **s** match each tuple in **r**.

a) Cost of Block-nested loops join

r: 2000 pages   s. 5000   B = 402

$2000 + \frac{2000}{400} \times 5000 = 27000$ pages.

b) Cost of Index-nested loops join

i. With clustered index

r: $2000 * 20 = 40,000$ records

s: $5000 * 5 = 25000$ records.

$2000 + (1 * 1 + 1.2) * 40000 = 90000$ I/Os.

ii. With unclustered index

$2000 + (1 * 5 + 1.2) \times 40000 = 250000$ I/Os.