DatDat Øving 5

Adrian Langseth, Tore Fossland, Eldar Sandanger, Mathias Chunnoo

April 2019

1 Querying different indexing data structures

SELECT * FROM ExamReg

We have to go through every element at least once to select all elements. Therefore a Heap file should be sufficient. Using a B+-tree we get 3 other levels we have to access first needlessly.

SELECT * FROM ExamReg WHERE Examno = 963432

If we are to find all exam-registrations where Examno = 963432, it would be most efficient to use a static hash structure with Examno as the hashed key. There are several complications using this, as not all of the hashed values in the "bucket" you access are guaranteed to be the correct Examno. However, you can guarantee that all of the posts you are looking for are in the referenced area. The main question is whether or not a hashed solution is more efficient than a B+-tree-solution. As the B+-tree has 4 levels and the hashed structure accesses on 1.25 blocks on average per search, the B+-tree is almost guaranteed to be less efficient. Therefore the hashed solution would be better for this query.

SELECT * FROM ExamReg WHERE Examno = 963432 ORDER BY Studentno DESC

Using a clustered B+-tree with (Examno, StudentNo) as search key, you can easily insert all the entries in the correct order with with Examno sorted by StudentNo in the tree-nodes. Then it's easy to select them in a descending order afterwards, as they're already sorted.

INSERT INTO ExamReg (14556589, 963439, '26.11.2018', "Exercises passed")

Using a heap file with you can easily insert all the entries. Heap file is great at insertion as it is raw and unsorted, so there is no need for extra work when inserting other than purely the insertion itself.

2 Nested loop join

$$b_S + \left\lceil \frac{b_S}{n_b - 2} \right\rceil * b_E = 800 + \left\lceil \frac{800}{34 - 2} \right\rceil * 12800 = 320800 \quad (1)$$

We use the one with the least amount of blocks as the outer loop, as the benefit of the buffer is greatest this way.

3 Transactions

a) Give two reasons why we would want transactions in our DBMS in the first place

Concurrency control: Transactions help us secure that actions are done separately and in order. We want to be able to share and access the the same data between several users simultaneously. To do this, we have to set some rules to control what happens whenever someone wants to access the data to avoid errors and conflicts. We call this concurrency-control.
Atomic access to large amounts of data

b) Explain briefly the ACID properties for transactions

Atomicity: A transaction should either be performed in its entirity or not at all.
Consistency Preservation: If a transaction is completed uninterrupted from beginning to end, it should take the database from one consistent state to another.
Isolation: A transaction should appear to be executed isolated from any other transaction.
Durability: Changes applied to the database by the transaction should persist in the database.

c) Determine the recovery property (strict, ACA, recoverable, unrecoverable)

H_1**: w1(X);r2(X);w1(X);w3(Y);w2(Y);c2;c1;c3**

Unrecoverable as transaction 2 commits before transaction 1 which it reads from.

H_2**: w1(X);w3(Y);w2(Y);c1;r2(X);r3(X);c2;c3**

ACA recovery property as there are no reads of uncommitted values, however there are writes of uncommitted values.

H_3: **r3(Y);w3(Y);r1(Y);w2(X);c2;w1(X);c3;c1**

Recoverable as transaction 1 commits after transaction 3 from which it reads. However, there is a read of Y, which is written to by transaction 3, before transaction 3 is committed.

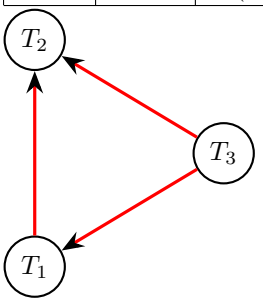
d) When are two operations in a schedule in conflict?

Two operations are in conflict if they belong to different transactions, they operate on the same element, and at least one of the operations is a write.

e) Determine if the following schedule, H_4, conflict serializable by drawing the precedence graph.

H_4 : w3(X);r1(X);r2(x)w2(X);r3(Y);c1;c2;c3

T_1	T_2	T_3
R(X)		W(X)
	R(X)	
	W(X)	
		R(Y)



As there are no cycles, it is conflict serializable.

f) What does it mean to have a deadlock between two or more transactions?

A deadlock occurs when at least two transactions are simultaneously waiting for each others locks.

g) Show how the schedule will look after applying rigorous two-phase locking.

See the table.

T_1	T_2	T_3
		wl(X)
		Write(X)
		rl(Y)
		R(Y)
		commit
		unlock(X,Y)
rl(X)		
R(X)		
commit		
ul(X)		
	wl(X)	
	R(X)	
	W(X)	
	commit	
	ul(X)	

4 Crash recovery with Aries

a) Which transactions are later given the UNDO operation and which are given the REDO operation?

Transaction T1 and T3 are given the UNDO operation, because they did not reach their commit points. T2 are given the REDO operation, because its commit point is after the last system checkpoint

b) How do the transaction table and DPT look after the analyzing phase is over?

Upper table is the Transaction table and below is the dirty page table.

TransactionID	LastLSN	Status
T_1	151	In Progress
T_2	150	Commit
T_3	152	In Progress

PageID	LSN
A	148
B	149