Raman Shukla

(DS) (2020201098)

(11) The issue with conventional indexes :-

→ It is difficult to add keys (update) once they are made.
→ To use the index whole index file has to be loaded in the memory.
→ The indexes has to be sorted after every updation.

Btree Solves all the above problem :-

→ We can add keys, the keys will be sorted and the memory need to be filled with the portion of Btree which is needed to be accessed.

(11) Object Based DBS and Object Oriented Protocol :-

Object Based DBS is popular as it follows the real world architecture and thus is scalable and robust and modular

(v) NoSQL and SQL :-

→ NoSQL provides system for storage and retrieval non-tabular data

→ NoSQL databases are non-relational
SQL databases are relational.

→ NoSQL for unstructured data have dynamic schemas

→ SQL DBS is table based while NoSQL DBS is document, key-value, graph or wide column stores.

(iv) Concurrency Protocols :-

→ Lock Based Protocols :-

→ 2 Phase Locking Protocol
Strict 2PL
Centralized 2PL

→ Timestamp Based Protocols.

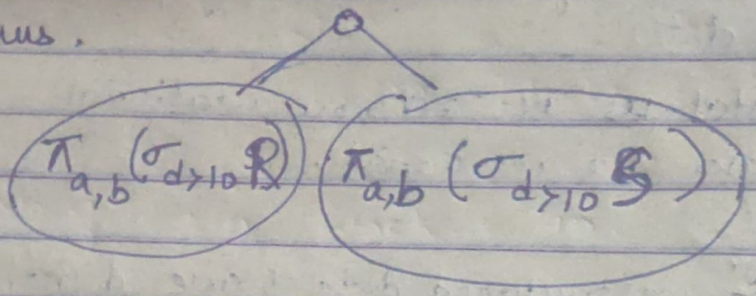
→ Validⁿ Based Protocols.

⇒ The faster concurrency control method would be 2phase Locking Protocol.

vii) $\pi_{a,b}(\sigma_{d>10}(R \cap S))$

Since both have a, b, d columns.

Thus,



$$\pi_{a,b}(\sigma_{d>10}R) \cap \pi_{a,b}(\sigma_{d>10}S)$$ //

(9) 10×10^6 tuples, 100 bytes, disk I/O 15msec

- 50MB Memory, B = 4096 bytes.

tuples/block = $\frac{4096}{100}$ = (40) //

total Blocks = $\frac{10^7}{40}$ = 2.5×10^5 Blocks

total Memory Blocks = $\frac{50 \times 10^6}{4096} \approx$ 1.25×10^4 Blocks

(3)

T_1	T_2	T_3
$r_1(A)$		
	$r_2(B)$	
		$r_3(C)$
$r_1(B)$		
	$r_2(C)$	
		$r_3(A)$
$w_1(A)$		
	$w_2(B)$	
		$w_3(C)$

(i)

T_1	T_2	T_3
S(A)		
$r_1(A)$		
Unlock(A)		
S(B)	S(B)	
$r_1(B)$	$r_2(B)$	
Unlock(B)	Unlock(B)	
	S(C)	S(C)
	$r_2(C)$	$r_3(C)$
	Unlock(C)	Unlock(C)
		S(A)
		$r_3(A)$
		Unlock(A)
X(A)	X(B)	X(C)
$w_1(A)$	$w_2(B)$	$w_3(C)$
Unlock(A)	Unlock(B)	Unlock(C)
Commit	Commit	Commit

(2) → 10^7 tuples

50MB = $\frac{50 \times 2^{20}}{2^{12}}$ Blocks

= 12800 Blocks

No. of tuples / Block = 40

Total Blocks = 2.5×10^5 Blocks

(i) Accessing in Random order :-

$2.5 \times 10^5 \times t = 10^6$ disk I/O.

(ii) cylinder Blocks = 512 Blocks

$\Rightarrow \frac{12800}{512}$ = 25 cylinders for 50MB.

Time for 25 transfer = 25×15 = 375 ms.

∴ phase 1 time = $\frac{2.5 \times 10^6}{512} \times 375$ = 183.4 s

thus phase 2 time = 183.4 s

Total time = $2.5 \times 10^6 \times 2 \times 15 \times 10^{-3} \times 2 \times 183.4$

= 7500 + 366.8

= 7866 s