DBMS UNIT V

Locking,
Deadlocks
Other Concurrency Control Techniques

Concurrency Solution:Locks

- Lock: A (bookkeeping) object associated with a database object(Table/Row/Column).
- Locking Protocol: A set of rules to be followed by transaction (enforced by DBMS) to ensure consistency in concurrent transactions.
- Strict 2PL Protocol: Strict Two Phase Locking Protocol:

  1)Shared lock for Read / eXclusive lock for Write

  2)Released when transaction is completed.

SQL Commands

- All transactions are ended by either Commit or Rollback.
- PostgreSQL uses BEGIN; to denote a beginning of transaction and END; or Commit and Rollback to terminate a transaction.
- Setting Autocommit to on makes every command to be committed. (this is default is postgresql – psql)
- In SQL, we can lock rows using the ”FOR UPDATE OF table_name”.

Examples

- Update employee table.
  - BEGIN;
  - SELECT * FROM emp FOR UPDATE OF emp;
  - UPDATE EMP SET salary = salary*1.05;
  - END;
- A Bank transaction:
  - BEGIN;
  - SELECT balance FROM acct_balance WHERE acct_no = 12345 FOR UPDATE OF acct_balance;
  - /* Some intermediate checks enough balance, transaction allowed, etc.?*/
  - INSERT INTO acct_transactions values (12345, 'atm no 987', 'Cash withdrawal', 10000.00);
  - UPDATE balance SET acct_balance = balance – 100000.00 WHERE acct_no = 12345;
  - END;

Notation Summary

- $R_T(A)$ Read object A for transaction T
- $W_T(A)$ Write object A for transaction T
- $S_T(A)$ Request Shared Lock on object A for transaction T
- $X_T(A)$ Request Exclusive Lock on object A for transaction T
- Commit Commit the transaction
- Abort Abort or Rollback the transaction.

Strict 2PL Protocol

- Strict 2PL Protocol: Strict Two Phase Locking Protocol:
- Two types of locks:
  - Shared lock for Read / eXclusive lock for Write
  - Released when transaction is completed.
- Two phases:
  - There is a **growing or expanding phase** – locks are obtained (the number of locks are increasing) and
  - There is a **shrinking phase or contracting** (as locks are released).
- **Strict 2PL**-**Strict** refers to the fact that locks are not released till the transaction is committed or aborted.
- **2PL** locking protocol may release the locks before commit or abort, but once a lock is released, no more locks may be obtained.

Transaction Schedules with Locks

T1	T2
X(A)	
R(A)	
W(A)	
X(B)	
R(B)	
W(B)	
Commit	
	X(A)
	R(A)
	W(A)
	X(B)
	R(B)
	W(B)
	Commit

T3	T4
S(A)	
R(A)	
	S(A)
	R(A)
	X(B)
	R(B)
	W(B)
	Commit
X(C)	
R(C)	
W(C)	
Commit	

Locking

- Advantages
  - Always produces serializable schedules.
- Disadvantages
  - Additional data structures (lock tables) and processing.
  - Deadlocks
    - Deadlock identification
    - Deadlock prevention
    - Deadlock resolution

Performance of Locking

- Locking affects the performance of DBMS
- Higher the active transactions, lower is the rate of increase of throughput (transactions/sec)
- Trashing is a point where any increase in the number of active transactions reduces the throughput (rate of execution of transactions per second).
- Throughput can be increased by
  - By locking the smallest unit (row)
  - By reducing the time that a transaction can hold on to a lock.
  - By reducing hot spots (frequently accessed and updated data)

Deadlocks

- Deadlock is a condition when Transaction T1 is waiting for a transaction T2 to release a lock and T2 is waiting for T1 to release a lock.
- Deadlocks may involve more than 2 transactions and objects.
- DBMS have deadlock identification, prevention and resolution mechanisms.

Deadlock Detection, prevention

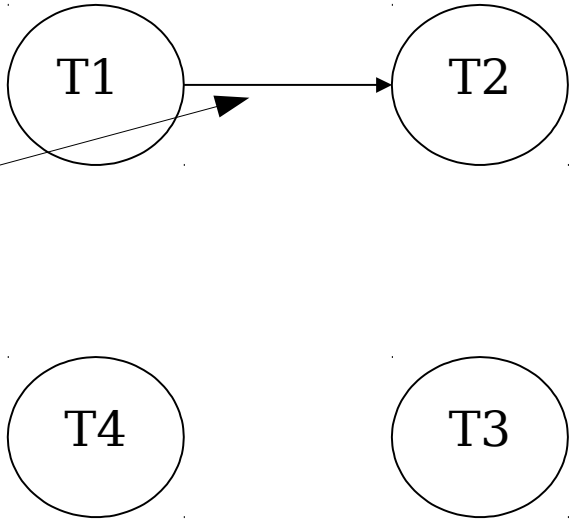
- A data structure called Waits-for graph
  - If there is a cycle in a waits-for graph, there is a deadlock
- Aborting transactions with long wait(time-outs)
- Prioritizing transactions
  - Earlier transactions are given priority over locks.
  - Later transactions are made to wait or Aborted.
- Conservative 2PL
  - Wait till all locks for a transaction are obtained.

Waits For Graph

- Nodes correspond to active transactions.
- There is an arc from T_i to T_j if T_i is waiting for T_j. That is T_i is waiting for T_j to release the lock.
- The lock manager adds an arc when a request for lock is made and removes the arc when it grants the lock.
- The waits for graph is periodically checked for cycles, if there is a cycle, a waiting transaction is aborted. This may allow some transactions to proceed
- Alternative is to time-out (abort) waiting transactions.

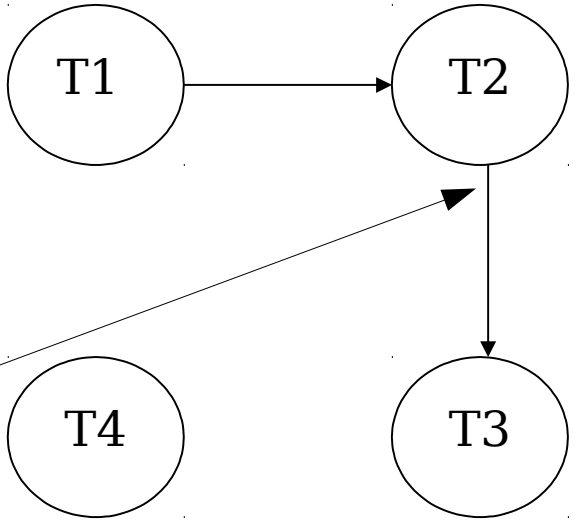
Waits for graph

T1	T2	T3	T4
S(A)			
R(A)			
	X(B)		
	R(B)		
S(B)			
		S(C)	
		R(C)	
	X(C)		
			X(B)
		X(A)	



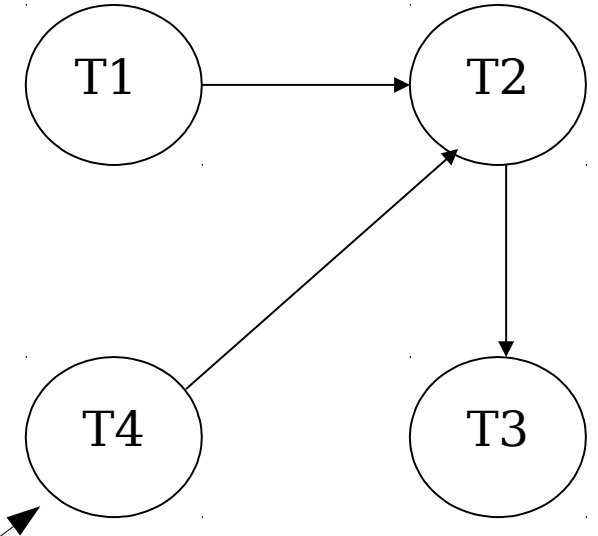
Waits for graph

T1	T2	T3	T4
S(A)			
R(A)			
	X(B)		
	R(B)		
S(B)			
		S(C)	
		R(C)	
	X(C)		
			X(B)
		X(A)	



Waits for graph

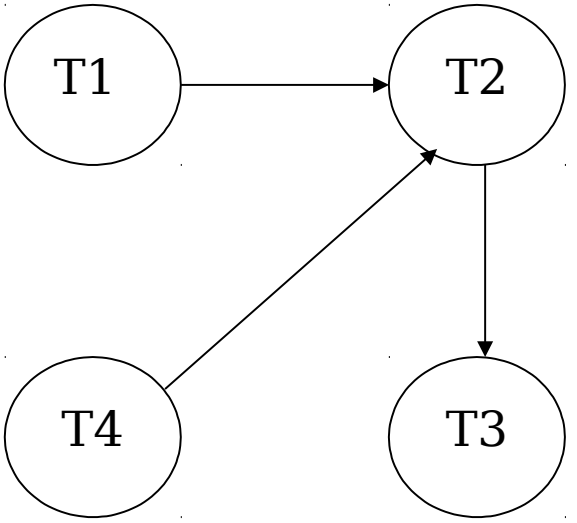
T1	T2	T3	T4
S(A)			
R(A)			
	X(B)		
	R(B)		
S(B)			
		S(C)	
		R(C)	
	X(C)		
			X(B)
		X(A)	



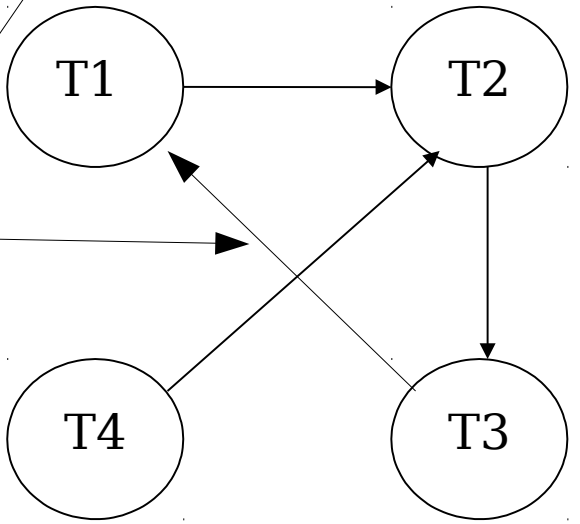
Before Deadlock

Waits for graph

T1	T2	T3	T4
S(A)			
R(A)			
	X(B)		
	R(B)		
S(B)			
		S(C)	
		R(C)	
	X(C)		
			X(B)
		X(A)	



Before Deadlock



After Deadlock

Timestamp Based Methods

- Every database object (O) is given a read timestamp RTS(O) and a write timestamp WTS(O).
- Every transaction is given a timestamp TS(T_i)
- If a transaction wants to read an object O
  - If TS(T) < WTS(O)
    - Abort T and restart with a new timestamp.
  - If TS(T) > WTS(O)
    - T reads object O and sets the RTS(O) to TS(T)
    - The change is written to the disk and log and is a significant overhead.
  - Else continue with read.
- If a transaction wants to write and object O
  - If TS(T) < RTS(O)
    - Abort T and restart with a new timestamp.
  - If TS(T) < WTS(O)
    - Abort T and restart with a new timestamp is one approach.
    - Ignore the updates write and continue (This is called the Thomas Write rule)
  - Else Else continue with write.

Multi-Version Concurrency Control

- Also called MVCC
- Here multiple versions of the object are maintained with timestamps
- If the timestamp of the transaction TS(T) is less than the read timestamp of the object RTS(O) then abort and start the transaction with a new timestamp,
- Else (TS(T) > RTS(O)) then creates a new version with read and write timestamps with the timestamp of transaction T.

Questions