Chapter - 9, 10 Crash Recovery, Transaction Processing and concurrency control.

6a) What is remote backup system? Describe paging shadow page recovery. Why is this recovery technique called no undo/no redo technique.

Ans:-

Remote backup system

- Remote backup system is a mechanism where every bit of real-time data is baked up simultaneously at two distant places. one of them is directly connected to the system and the other one is kept at remote place at there as backup.

- Remote backup system provides high degree of availability allowing transaction processing to continue even if the primary site is destroyed by fire, flood or earthquake.

- Data and log records from a primary sites are continously backup to remote backup site. If the primary site fails, the remote backup sites takes over transaction processing, after executing certain recovery actions

- Remote backup provides a sense of security in case the primary location where the database is located gets destroyed

- Remote backup can be offline or real-time or online. In case it is offline it is maintained manually.

Working principle of Remote backup system.

- The remote backup site is stored at different state or places to prevent the damage on the remote backup site when the primary site is damaged.
- The primary site or local server is connected to the backup site through internet connection (WAN/LAN)
- The remote site is kept synchronized with primary site, as updates are performed at primary site. The synchronization is achieved by sending all log records from primary site to the remote backup site.
- When the primary site fails, the remote backup site takes over processing.

Archicture of Remote backup system

- Primary site :- The primary site are used in traditional-transactions processing system such as centralized or client-server systems. These system are those that run on a single computer system and do not interact with other systems.
- Network :- Remote backup software primarily bridges a connection between local network/IT environments with a remote backup location. The backup data is copied on a scheduled or specific routine by the backup software and is transferred and copied to the remote location over WAN, Internet or VPN connection.

- Backup :- A backup, or data backup is a copy of computer data taken and stored elsewhere so that it may be used to restore the original data after a data loss event. A backup system contains at least one copy of all data after its loss from data deflection or corruption, or to recover data from an earlier time. Backups provide a simple form of disaster recovery.

- Log Records: Log records contains old values and new values for all update data items. The new values are used in case the updates need to be redone after system crash. The old values are used to roll back the updates of the transaction if the transactions aborts during normal operations, as well as to roll back the updates of the transaction in case the system crashed before the transaction committed.

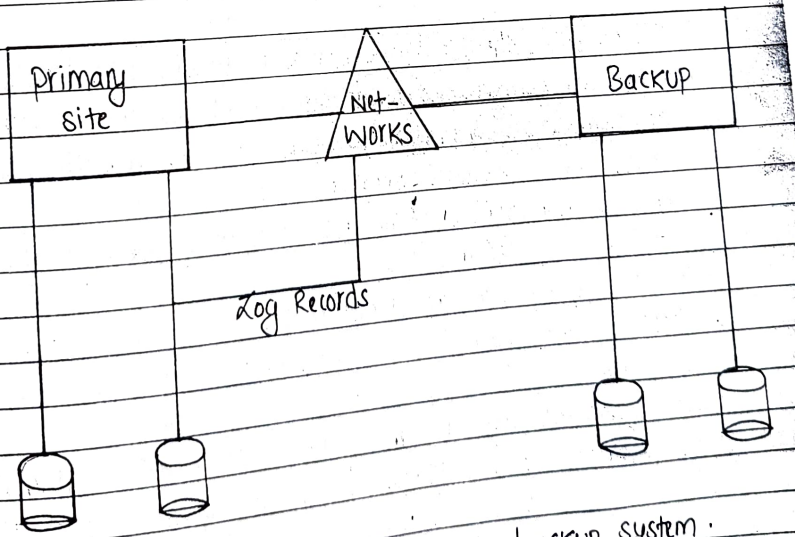


Figure :- Architecture of Remote backup system.

Example of Remote backup system.

- Subscription backup services provided by commercial data centers.
- Backup to an offsite FTP server over the Internet.

Advantages of Remote backup system.

- It provides a sense of security incase the primary location where database is located gets damaged.
- Manage and secure digital data information.
- Remote access to backed up data from any location.
- It can recover even if all data at the primary site are lost.
- It can be offline or real-time or online. In case it is offline it is maintained manually.

Disadvantages of Remote backup system.

- Depending on the available network bandwidth, the restoration of data can be slow. Because data is stored offsite, the data must be recovered either via the internet or via a disk shipped from the online backup service provider.
- Some backup service providers have no guarantee that stored data will be kept private.
- If the encryption password is lost, data recovery will be impossible, However, with managed services this should not be a problem.
- It is possible that a remote backup service provider could go out of business or be purchased, which may affect the accessibility of one's data or the cost to continue using the service.

Shadow Paging and it's technique

- It is an alternative to log based recovery.
- It is a recovery technique that is used to recover database. In this technique, database is considered as made up of fixed size of logical units of storage which are referred as pages.
- Pages are mapped into physical blocks of storage, with help of the page table which allow one entry for each logical page of database.
- This method uses two page tables named current page table and shadow page table.
- To start with, both the page tables are identical
  - only current page table is used for data item accesses during execution of the transaction.
- whenever any page is about to be written for first time.
  - A copy of this page is made onto an unused page.
  - The current page table is than made to point to the copy.
  - The update is performed on the copy.
- To commit a transaction
  - Flush all modified pages in main Memory to disk.
  - Output current page table to disk.
  - Make the current page table the new shadow page table as follows:
    - Keep a pointer to shadow page table at a fixed known location on disk.
    - Simply update the pointer to point to current page table on disk.

- Once pointer to shadow page table has been written, transaction is committed.
- Non recovery is needed after a crash, new transactions can start right away using shadow page table.
- Pages not pointed to from current/shadow page table should be free.

- Advantages of shadow paging over log based are:-
  - No overhead of writing log records.
  - Recovery is trivial. It makes undoing the effect of the executing transaction very simple.
- Disadvantages of shadow paging are:-
  - Coping the entire page table is very expensive.
    - Can be reduced by using a page table structured like B+ tree. Here, no need to copy entire tree, only need to copy paths in the tree that lead to update leaf nodes.
  - Commit overhead is high even with above extension. Need to flush every updated page and page table.
  - Data gets fragmented. (related pages gets seperated on disk)
  - After every transaction completion old version of modified data need to be garbage collected.
  - Hard to extend algorithm to allow transactions to run concurrently.

Shadow paging

Shadow page

⇒ Since, the recov
mistake, such
it also does
that are pr
so, we can s
no undo te

(b) What is ser
with it's
with concu

Ans:-

- Shadow paging

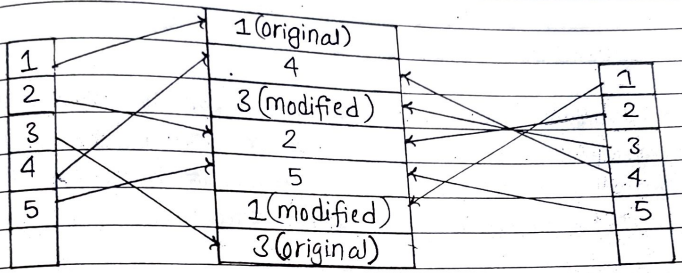


Fig:- Shadow Paging

⇒ Since, the recovery do not involves the undo function to reverse the mistake, such as deleting the wrong word in a sentence and it also does not involves redo function to restore any action that are previously undo using the function called undo. so, we can say that shadow paging is called as no Redo/ no undo techniques.

6b) What is serialzability? What are concurrency control protocols with it's benifits? Also Explain the problems associated with concurrency?

Ans:-

18/5/96 29

6a) Serializability and serializable schedule

- Serializability is a concept that helps us to check which schedules are serializability.
- When multiple transactions are running concurrently then there is a possibility that the database may be left in an inconsistent state. A serializable schedule is the one that always leaves the database in consistent state.
- Serializability in DBMS decides if an interleaved non-serial schedule is serializable or not.
- A serial schedule is always a serializable schedule because in serial schedule, a transaction only starts when the other transaction finished execution. However a non-serial schedule needs to be checked for serializability.
- A non-serial schedule of 'n' number of transactions is said to be serializable schedule, if it is equivalent to the serial schedule of those 'n' transactions executes at a time and the other starts when the already running transactions finished
- eg of serializability :- consider 2 schedules ; schedule 1 and 2.

Schedule 1

Transaction 1	Transaction 2
• A1	
• A2	• B1
	• B2

Schedule 2

Transaction 1	Transaction 2
• $A1_{①}$	• $B1_{②}$
• $A2_{③}$	• $B2_{④}$

18/5/96 30

Here, in schedule-2, 1.2.3.4- are order of transactions for the schedule 2 that portrays an interleaved execution.

In the given above examples;

- Schedule 1 is a serial schedule consisting of Transaction1 and Transaction2 where the operation on data items A (A1 and A2) are performed first and later the operations on data item B (B1 and B2) are carried out serially.

- schedule 2 is a non-serial schedule consisting of Transaction1 and Transaction2 where the operations are interleaved.

- In the given senario, schedule2 is serializable if the output obtained from both schedule2 and schedule1 are equivalent to one another in a nutshell, a transaction within a given non-serial schedule is serializable if its outcome is equivalent to the outcome of the same transaction when executed serially.

Rules of serializability:-

- ordering of read/write is important.
- If two transactions only read data item they do not conflict and order is not important.
- If two transactions either read or write completely separate data items, they do not conflict and order is not important.
- If one transaction write a data item and another reads or writes same data item, order of execution is important.

Types of serializability:-

18/5/96 31

i) Conflict Serializability :-

- Instructions I_i and I_j of transactions T_i and T_j respectively conflicts.
  - If and only if there exists same item Q accessed by both I_i and I_j and at least one of these instructions wrote Q.

- If the schedule S can be transformed into a schedule S' by a series of swapping of non-conflicting instructions, then S and S' are conflict equivalent.

- A schedule S is conflict serializable if it is conflict equivalent to a serial schedule.

Instructions I_i	Instructions I_j	Result
Read(Q)	Read(Q)	No conflict
Read(Q)	Write(Q)	conflict
Write(Q)	Read(Q)	conflict
Write(Q)	Write(Q)	conflict

- eg 1 :- Let we have schedule A and schedule B follows:-

Fig Schedule A

T1	T2
Read(A)	
write(A)	
	Read(A)
	write(A)
Read(B)	
write(B)	
	Read(B)
	write(B)

Fig Schedule B

T_1	T_2
Read(A)	
write(A)	
Read(B)	
write(B)	
	Read(A)
	write(A)
	Read(B)
	write(B)

18/5/46 82

Here, schedule A can be transformed into serial schedule B by series of swaps of non conflict instructions.

Hence, schedule A is called conflict serializable.

- eg2 :- Let we have schedule x and schedule y as follows

T1	T2
Read(a)	
	Write(a)
Write(a)	

Fig :- schedule x

T1	T2
Read(a)	
Write(a)	
	Write(a)

Fig :- schedule y

Here, we can't transform schedule x into schedule y by swapping of non conflict instructions.

Hence, schedule x is non conflict serializable.

ii) View serializability.

- Two schedule are said to be view equivalent if following three conditions hold :-
  - For each data item a, if Transaction T_i reads the initial value of a in schedule s, then transaction T_i in schedule s' must also read initial value of a.

- For each data item Q, if T_i executes read(Q) in S and if value was produced by write(Q) of T_i then read(Q) of T_i in S' must also read values of Q produced by some write(Q) if T_i.
- For each data item Q, the transaction that performs the final write(Q) operation in S must perform the final write operation in S'.

- A schedule 'S' is view serializable if it is view equivalent to a serial schedule.

- E.g.1 Let we have schedule A and schedule B as follows:-

T1	T2	T3
Read(Q)		
	Write(Q)	
Write(Q)		Write(Q)

Fig:- Schedule A

T1	T2	T3
Read (Q)		
Write (Q)		
	Write(Q)	
		Write(Q)

Fig:- Schedule B

18154C

34

- Here, schedule A is view serializable schedule, because it view equivalent to serial schedule B, as transaction T1 perform Read(Q) i.e. reads the initial value of Q in both schedules. and also,

- Here, write(Q) of T2 and T3 are called blind writes because it is performed without having performed Read(Q).

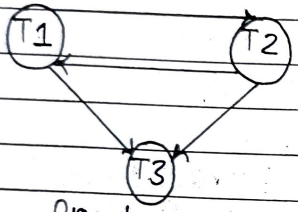
- Every conflit serializable schedule is also view serializable but not vice-versa.

Testing for serializability :-

- Construct a directed graph called precedence graph from S.
- Directed Graph consists of a pair $G=(V,E)$ where,
  - V is set of vertices ; it consist of all transaction in the schedule
  - E is a set of edges. Set of edges consists of all edges $T_i \rightarrow T_j$ for which one of three conditions holds
    * T_i executes write(Q) before T_j executes read(Q).
    * T_i executes read(Q) before T_j executes write(Q).
    * T_i executes write(Q) before T_j executes write(Q).
- If an edge $T_i \rightarrow T_j$ exists in precedence graph, then in any serial schedule S' equivalent to S, T_i must appear before T_j.
- If the precedence graph has a cycle then the schedule is not conflict serializable.
- If the precedence graph has not any cycle then schedule is conflict serializable.

eg 1

T1	T2	T3
Read(A)		
	write(A)	
write(A)		
		write(A)

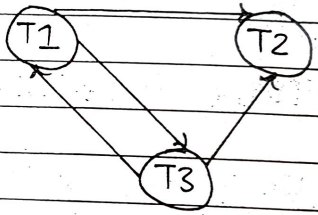


Precedence graph

Here, as graph contains cycle, above schedule is non conflict serializable.

eg :- 2

T1	T2	T3
write(A)		
	Read(A)	
		write(B)
write(B)		
		write(B)
	write(A)	
		Read(B)
	Read(B)	



precedence graph.

Here, as graph contains cycle, above schedule is not conflict serilizable.

Some advantages of serializability are :-

- It helps to preserve the consistency and concurrency of a database.
- It helps to obtain an equivalent output as of a serial schedule for the same 'n' number of transactions.
- It is a classical concurrency schema which assumes that all accesses to the database are done using read and write operations also it helps that the schedules are serializable or not.

concurrency control protocols with it importance in transaction

- concurrency control is a procedure of managing simultaneous operations without conflicting with each other.
- It ensures that database transactions are performed concurrently and accurately to produce correct results without violating data integrity of the respectively.
- It is used to address such conflicts, which mostly occur with a multi-user system. Therefore it is the most important element for proper functioning of a database where two or more database transactions are executed simultaneously, which require access to the same data.
- The concurrency control protocols ensures the atomicity, consistency, isolation, durability and serializability of the concurrent execution of the database transactions.
- Therefore, these protocols are categorize densly as :-

i) Lock Based concurrency control protocol.

ii) Time stamp concurrency control protocol.

iii) Validation Based concurrency control protocol.

i) Lock Based concurrency control protocol.

- In this type of protocol, any transaction cannot read or write data until it acquires as appropriate lock on it. There are two types of lock:-

i) Shared lock:-

- It is also known as a Read-only lock. In a shared lock data item can only read by the transaction.
- It can be shared between the transactions because when the transaction holds a lock, then it can't update the data on the data item.

ii) Exclusive lock.

- In the exclusive lock, the data item can be both reads as well as written by the transactions.
- This lock is exclusive, and in this lock, multiple transactions do not modify the same data simultaneously.

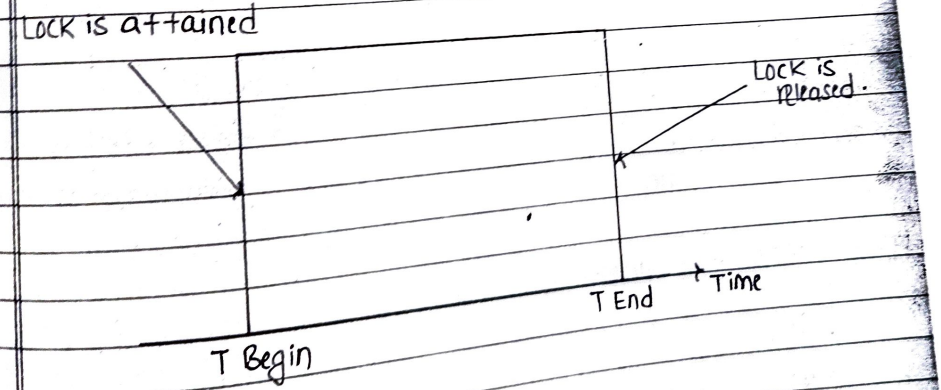
There are four types of lock protocols available:-

1) Simplistic lock protocol

- It is the simplest way of locking the data while transaction. Simplistic lock-based protocols allow all the transactions to get the lock on the data before insert or delete or update on it.
- It will unlock the data item after completing the transaction.

2) Pre-claiming lock protocol

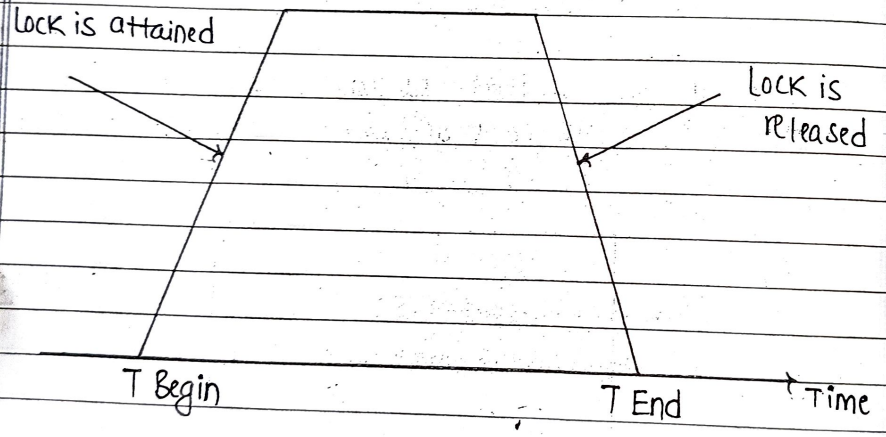
- Pre-claiming lock protocols evaluate the transaction to list all the data items on which they need locks.
- Before initiating an execution of the transaction, it requests DBMS for all the lock on all those data items.
- If all the locks are granted then this protocol allows the transaction to begin. When the transaction is completed then it releases all the lock.
- If all the locks are not granted then this protocol allows the transaction to rolls back and waits until all the locks are granted.



3) Two-phase locking (2PL)

- The two-phase locking protocol divides the execution phase of the transaction into three parts.

- In the first part, when the execution of the transaction starts, it seeks permission for the lock it requires.
- In the second part, the transaction acquires all the locks. The third phase is started as soon as the transaction releases its first lock.
- In the third phase, the transaction cannot demand any new locks. It only releases the acquired locks.



There are two phases of 2PL:-

i) Growing phase:- In the growing phase, a new lock on the data item may be acquired by the transaction, but none can be released.

ii) Shrinking Phase:- In the shrinking phase, existing lock held by the transaction may be released, but no new locks can be acquired.

10/12/2018

In the below example, if lock conversion is allowed then the following phase can happen:

a) upgrading of lock (from s(a) to x(a)) is allowed in growing phase

b) Downgrading of lock (from x(a) to s(a)) must be done in shrinking phase.

eg:-

	T1	T2
0	Lock-S(A)	
1		Lock-S(A)
2	Lock-X(B)	
3	——	——
4	Unlock(A)	
5		Lock-X(C)
6	Unlock(B)	
7		Unlock(A)
8		Unlock(C)
9	——	——

The following way shows how unlocking and locking work with 2PL.

Transaction (T1):-

- Growing phase: From step 1-3
- Shrinking phase: From step 5-7
- Lock point: at 3

Transaction (T2):

- Growing phase : from step 2-6
- shrinking phase : from step 8-9
- Lock point : at 6

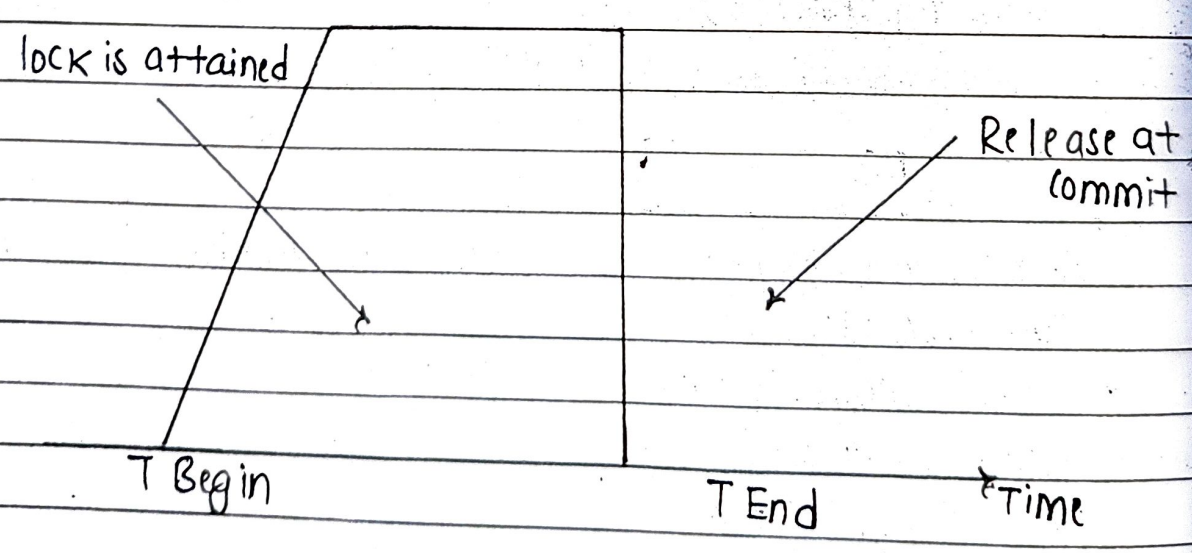
4) Strict Two-phase locking (strict-2PL)

- The first phase of strict-2PL is similar to 2PL. In the first phase, after acquiring all the locks, the transaction continues to execute normally.

- The only difference between 2PL and strict 2PL is that strict 2PL does not release a lock after using it.

- strict-2PL waits until the whole transaction to commit, and then it releases all the locks at a time.

- strict-2PL protocol does not have shrinking phase of lock release.



- It does not have cascading abort as 2PL does.

10/12/21

Page : 31

ii) Timestamp ordering protocol.

- The Timestamp ordering protocol is used to order the transactions based on their Timestamps. The order of transaction is nothing but the ascending order of the transaction creation.

- The priority of the older transaction is higher that's why it executes first. To determine the Timestamp of the transaction, this protocol uses system time or logical counter.

- The lock-based protocol is used to manage the order between conflicting pairs among transactions at the execution time but Timestamp based protocols start working as soon as a transaction is created.

- Let's assume there are two transactions T1 and T2. Suppose the transaction T1 has entered the system at 007 times and transaction T2 has entered the system at 009 times. T1 has the higher priority, so it executes first as it is entered the system first.

- The timestap ordering protocol also maintains the timestamp of last 'read' and 'write' operation on a data.

Basic Timestamp ordering protocol works as follows:-

a) Check the following condition whenever a transaction (T_i) issues a Read(x) operation:-

- If W_TS(x) > TS(T_i) then operation is rejected.
- If W_TS(x) <= TS(T_i) then operation is executed.
- Timestamps of all the data items are updated.

b) check the following condition whenever a transaction (T_i) issues a write(x) operations :-

- If TS(T_i) < R_TS(x) then operation is rejected.
- If TS(T_i) < W_TS(x) then the operation is rejected and T_i is rolled back otherwise the operation is executed.

where,

- TS(T_i) denotes the Time stamp of the transaction T_i.
- R_TS(x) denotes the Read time-stamp of data-item x.
- W_TS(x) denotes the write time_stamp of data-item x.

Advantages and disadvantages of TO protocols.

- TO protocol ensures serializability since the precedence graph is as follows:

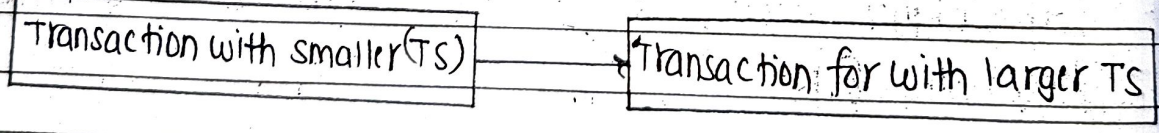


Image:- Precedence Graph for TS ordering.

- TS protocol ensures freedom from dead lock that means no transaction ever waits.
- But the schedule may not be recoverable and may not even be cascade-free.

iii) Validation Based concurrency protocol.

- Validation Phase is also known as optimistic concurrency control technique. In the validation based protocol, the transaction is executed in the following three phases.

a) Read Phase :-

- In this phase, the transaction (T) is read and executed.
- It is used to read the values of various data items and stores them in temporary local variables.
- It can perform all the write operations on temporary variables without an update to the actual database.

b) Validation phase :-

- In this phase, the temporary variable value will be validated against the actual data to see if it violates the serializability.

c) Write phase :-

- If the validation of the transaction is validated, then the temporary results are written to the database or system otherwise the transaction is rolled back.

Here, each Phase has the following different timestamps :-

Start (T_i) :- It contains the time when T_i started its execution.
Validation (T_i) : It contains the time when (T_i) finishes its read phase and starts its validation phase.
Finish (T_i) : It contains the time when T_i finishes its write phase.

- This protocol is used to determine the time stamp for the transaction for serialization using the time stamp of the validation phase, as it is the actual phase which determines if the transaction will commit or rollback.

- Hence TS(T) = validation(T).

- The serializability is determined during the validation process. It can't be decided inadvance.

- While executing the transaction, it ensures a greater degree of concurrency and also less number of conflicts.

- Thus it contains transactions which have less number of rollbacks.

Benifits/advantages of concurrncy. Importance of it :-

- Reduced waiting time response time or turn around time.
- Increased throughput or resource utilization.
- If we run only one transaction at a time than the acid property is sufficient but it is possible that when multiple transactions are

executed concurrently than database may become inconsistent
- overlapping with the input-output activity with cpu also increases the response time better.
- But interleaving of instruction between transaction may also lead to many problems due to which concurrency control is required.

Drawbacks of concurrency
- It is required to protect multiple applications from one another.
- Additional performance overheads and complexities in OS are required for switching among applications.
- It is required to coordinate multiple applications through additional mechanisms.
- sometimes running too many applications concurrently leads to severely degraded performance.

Problems due to concurrency.
There are many which may occur due to concurrency.

1) Dirty read problems

- If a transaction reads an uncommitted temporary value written by some other transaction than it is called dirty read problem.
- In this one transaction read item updated by another uncommited transaction that may be future be aborted or failed.
- In such cases, the read value disappears from the database upon abort this is turned on dirty read the reading transaction end with incorrect results.

eg:-

T1	T2
	R(A)
	W(A)
R(A)	

The values of item X which is read by T2 is called dirty read because this data can be created by a transactions that has not been committed yet.

2) Loss update problem / Write-Write problem.

- This problem occur when two transactions access the same data item and have their operations interleaved in a way that makes the value of some database items incorrect.

- If there are two write operations of the different transaction on some data values and in between them there are no read operations then the second write over the first.

Consider a schedule below:-

eg:-

T1	T2
R(A)	
W(A)	
	W(A)

- Here, is a blind write that means write without a read. Here the changes made by transaction T1 are lost which is updated by a transaction T2.

3) unrepeatable and phantom read problem

- When a transaction cannot repeat the read instructions because the variable is deleted the variable by some other transactions then this problem is called phantom read problem. In this problem at different instances of time a transaction read gives different values it is because data item might have been updated by another transaction.

- This causes a problem while execution of some aggregate by a transaction and due to changes in the values of the data item by another transaction it leads to incorrect results. when a transactions read values of data twice and another transaction's updates data item in between then the results of two read operations will differ.

eg:-

T1	T2
R(A)	
	R(A)
Delete(A)	
	R(A)

4) Incorrect summary problem.

- When one of the transactions is checking on aggregate summary function while other transactions are updating then this problem is called incorrect summary problem.

- The aggeregate functions may calculate some values before they updated and others after they are updated.

5b) What is data dictionery storage? Mention the information's which are kept in it by defining file categories.

Data dictionery storage

- Storing the relational schemas and other metadata about the relations in a structure is known as Data dictonery or system catalog.

- A data dictionery is like the A-Z dictionery of the relational database system holding all information of each relation in the database.

→ The types of information a system must store are:-

- Name of the relations.
- Name of the attributes of each relation.
- Lengths and domains of attributes.
- Name and definations of the views defined on the database.
- Various integrity constraints.

→ With this, the system also keeps the following data based on users of the system:-

- Name of authorized users.
- Accounting and authorization information about users.
- The authentication information for users, such as passwords or other related information.

→ In addition to this, the system may also store some statistical and descriptive data about the relations, such as:-

- Number of tuples in each relation.
- Method of storage for each relation, such as clustered or non-clustered.

→ A system may also store the storage organization, whether sequential, hash, or heap. It also notes the location where each relation is stored:-

- If relations are stored in the files of the OS, the data dictionery note, and stores the names of file.
- If the database stores all the relations in a single file, the data dictonery notes and store the blocks of containing records of each relation in a data structure similar to a linked list.

→ At last, it also stores the information regarding each index of all the relations:-

- Name of the index.
- Name of the relation being indexed.
- Attributes on which the index is defined.
- The type of index formed.

→ All the above information or metadata is stored in a data dictionery. The data dictionery also maintains updated information whenever the

occur in the relations. such metadata constitutes a miniature database. some systems store the metadata in the form of a relation in the database itself. The system designers design the way of representation of the data dictioneny. Also, a data dictioney stores the data in a non-formalized manner. It does not use any normal form so as to fastly access the data stored in the dictioney.

- For eg:- In the data dictioney, it uses underline below the value to represent that the following field contains a primary key.

So, whenever the database system requires fetching records from a relation of data dictioney about the location and storage organization of the relation. After confirming the details, it finally retriens the required record from the database.

Importance advantages and needs of data dictioney

- Avoid duplication.
- Make maintenance straightforward.
- To locate the error in the system.
- Easy to search data in huge database. (Searchable)
- provides quick report on the data and hence making the data management easy. (Report)
- (Authorization) Record what data belongs to whom.
- (Catalogue) A central catalogue for metadata.
- (Easy) DBA can easily able to track any chaos in the database.

Disadvantage of data dictionery.

- A Data Dictionery storage is a useful management tool, but at a price.
- The Data Dictionery storage 'project' may itself take two or three years.
- It needs careful planning, defining the enact requirements.
- Designing its contents, testing, implementation and evaluation.
- The cost of a Data dictionery storage includes not only the initial price of its installation and any hardware requirements but also the cost of collecting the information entering it into the Data dictionery storage.
- Keeping it up-to-date and enforcing standards.
- The use of a Data Dictionery storage requires management commitment which is not easy to achieve, particularly where the benefits are intangible and long term.

6a) What is crash recovery?

- Crash recovery is the scheme in a database system that can restore the database system to the consistent state that existed before crash.
- The time for recovery must be minimal.

- This is done by rolling back incomplete transactions and completing committed transactions that were still in memory when the crash occurred in the below figure.

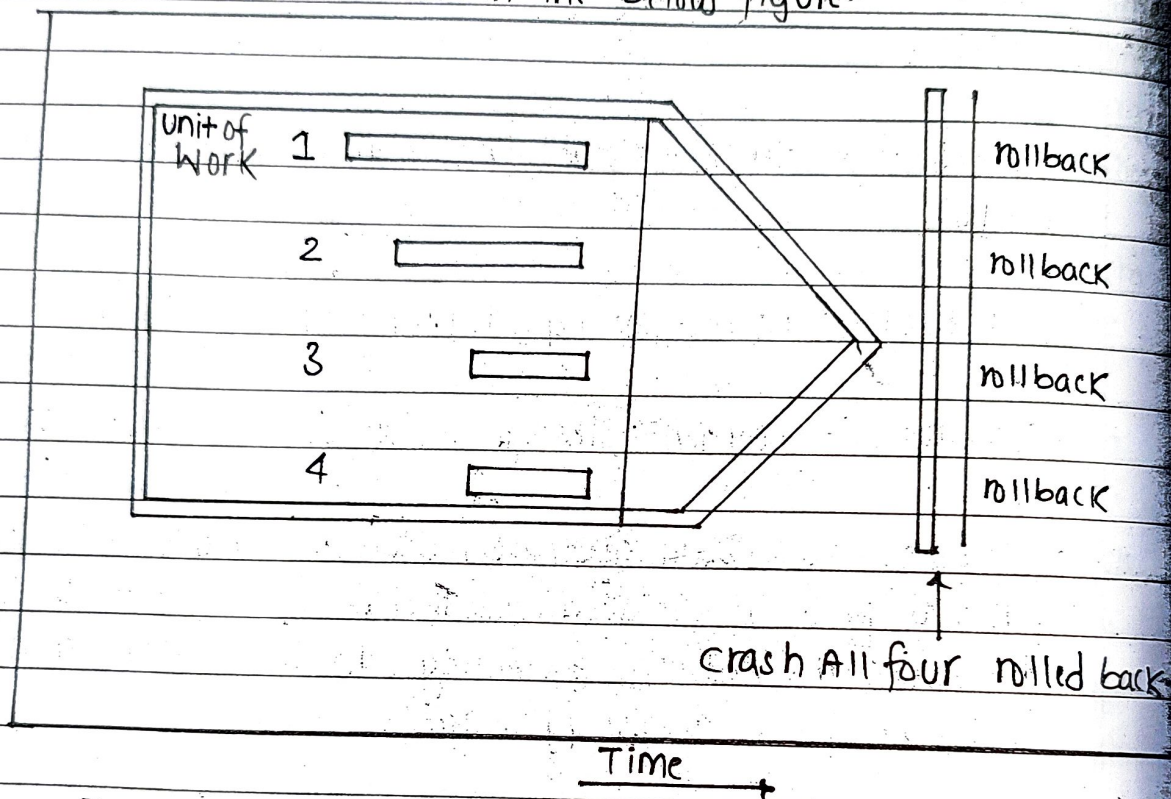


Fig :- Rolling back units of work (crash recovery)

conditions that can necessitate a crash recovery include :-

- A power failure on the machine, causing the database manager and the database partitions on it to go down.
- A hardware failure such as memory disk, cpu, or network failure.
- A serious operating system error that causes the Db2® instance to end abnormally.

⇒ Importance of crash recovery

- Needed for achieving atomicity and durability.
  - need to abort transactions or restart them.
  - Need to recover from crashes.
- crash recovery algorithms had major impact beyond databases
  - Algorithms are interesting in their own right.
- Logging for crash recovery has significant impact on DBMS performance.

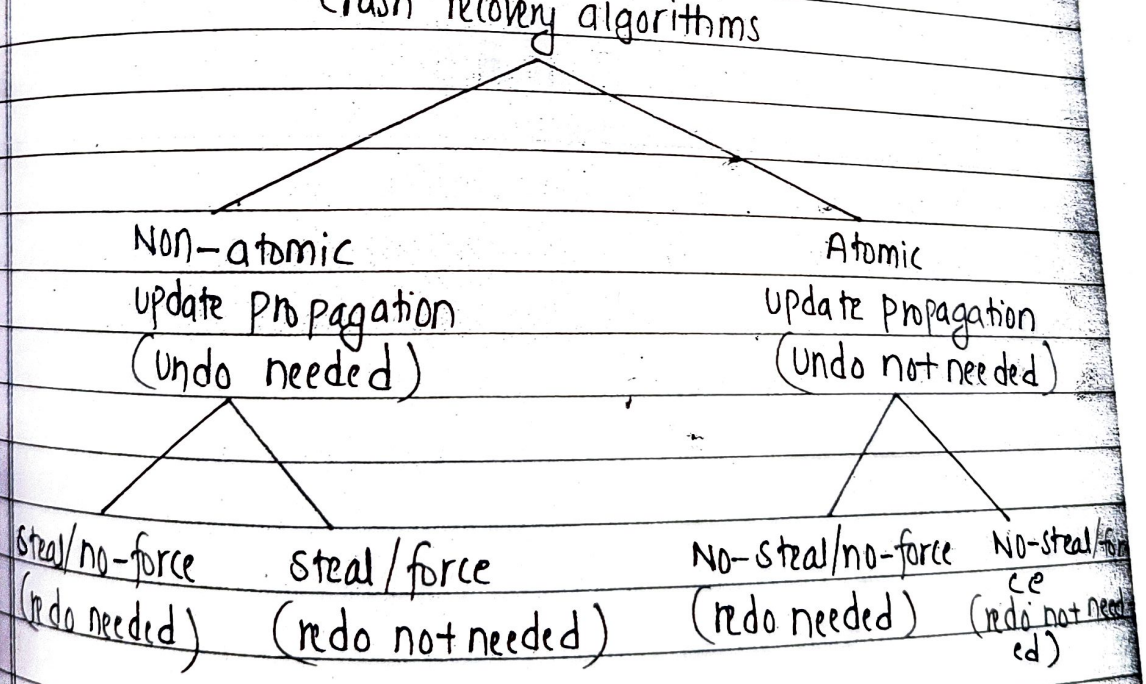


Fig:- Taxonomy of crash recovery algorithms

⇒ Importance of crash recovery

- Needed for achieving atomicity and durability.
  - need to abort transactions or restart them.
  - Need to recover from crashes.
- crash recovery algorithms had major impact beyond databases
  - Algorithms are interesting in their own right.
- Logging for crash recovery has significant impact on DBMS performance.

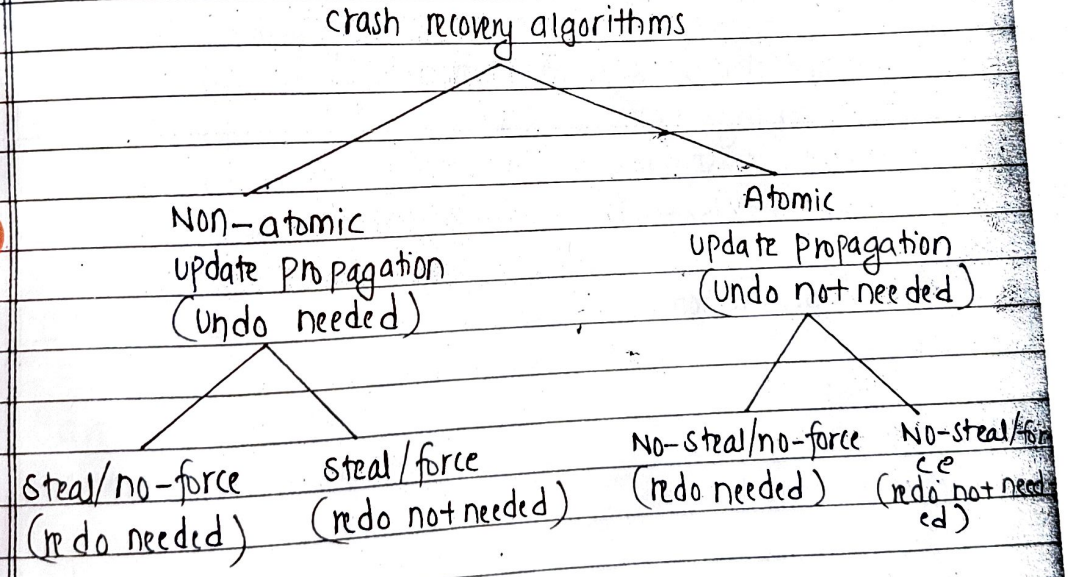


Fig:- Taxonomy of crash recovery algorithms

6b) What is transaction?

- A transaction is a single logical unit of work which accesses and possibly modifies the contents of a database.
- Transaction access data using read and write operations.
- In order to maintain consistency in a database, before and after the transaction, certain properties are followed.

→ Types of Transactions.

* Based on Application areas

- Non-distributed vs. distributed.
- Compensating transactions.
- Transactions Timing.
- on-line vs. batch.

* Based on Actions

- Two-step
- Restricted
- Action model.

* Based on structure

- Flat or simple transactions: It consists of a sequence of primitive operations executed between a begin and end operations.
- Nested transactions: A transaction that contains other transacti

Workflow

Q NO 5 10120181 Page 22

[redacted] 18546

Q NO 5

⇒ In order to maintain consistency in database, before and after transactions the acid properies are followed:-
ACID properties.

- ACID is a set of properties of database transactions intended to guarantee data validity despite errors, power failures and other mishaps. In the context of databases, a sequence of database operations that satisfies the ACID properties is called a transaction.

i) Atomicity:-

- By this, we mean that either the entire transaction takes place at once or doesn't happen at all. There is no midway i.e. transactions do not occur partially. Each transaction is considered as one unit and either runs to completion or is not executed at all.

- It involves the following two operations.
  - Abort: If a transaction aborts, changes made to database are not visible.
  - commit: If a transaction commits, changes made are visible.

- Atomicity is also known as the "All or Nothing rule".

Consider the following transaction 'T' consisting of T1 and T2:
Transfer of 100 from account 'X' to account 'Y'.

Before: X: 500	Y: 200
Transaction T	
T1	T2
Read(X)	Read(Y)
X := X-100	Y := Y+100
Write(X)	Write(Y)
After: X: 400	Y: 300

- If the transaction fails after completion of T1 but before completion of T2. (say, after write(X) but before write(Y)), then amount has beendeducted from X but not added to Y. This results in an incosistent database state. Therefore, the transaction must be executed in entirety in order to ensure correctness of database state.

ii) Consistency :-

- This means that integrity constraints must be maintained so that the database is consistent before and after the transaction.
- It refers to the correctness of a database.
- Referring to the example above, The total amount before and after the transaction must be maintened.
Total before T occours = 500+200 = 700. Total after T occours = 400+300 = 700, Therefore, database is consisten
Inconsistency occurs in case T1 completes but T2 fails.
As a result T is incomplete.

iii) Isolation :-

- This property ensures that multiple transactions can occur concurrently without leading to the inconsistency of database state.

- This property ensures that the execution of transactions concurrently will result in a state that is equivalent to a state achieved these were executed serially in some order.

Let $X = 500$, $Y = 500$

Consider two transactions T and T''.

T	T''
Read(x)	Read(x)
x := x * 100	Read(y)
Write(x)	z := x+y
Read(y)	Write(z)
y := y-50	
Write(y)	

Suppose T has been executed till Read(y) and the T'' starts. As a result, interleaving of operations takes place due to which T'' reads correct value of 'x' but incorrect value of y and sum computed by T'': (x+y = 50,000+500 = 50,500) is thus not consistent with the sum at end of transaction T: (x+y = 50,000+450 = 50,450). This results in database inconsistency, due to a loss of 50 units. Hence, transactions must take place in isolation and changes should be visible only after they have been made to the main memory.

iv) Durability :-

- This property ensures that once the transaction has completed execution, the updates and modifications to the database are stored in and written to disk and they persist even if a system failure occurs.

- These updates now become permanent and are stored in non-volatile memory. The effects of the transactions, thus are never lost.

- The ACID properties, in totality, provide a mechanism to ensure correctness and consistency of a database in a way such that each transaction is a group of operations that acts a single unit, produces consistent results, acts in isolation from other operations and updates that it makes are durably stored.

2012 Fall

5b) Describe about failure classification? Write down Differences between Deferred Database modification and Immediate Database modification.

To find that where the problem has occurred, we generalize a failure into the following categories.

i) Transaction failure :-

- The transaction failure occurs when it fails to execute or when it reaches a point from where it can't go any further.

- If a few transaction or process is hurt, then this is called as transaction failure.

- Reasons for transaction failure could be :-

a) Logical errors :- If a transaction cannot complete due to some code error or an internal error condition, then the logical error occurs.

b) syntax error :- It occurs where the DBMS itself terminates an active transaction because the database system is not able to execute it. For eg :- The system aborts an active transaction in case of deadlock of resource unavailability.

ii) System crash.

- System failure can occur due to power failure or other hardware or software failure. Example : operating system error.

- Fail-stop assumption : In the system crash, non-volatile storage is assumed not to be corrupted.

iii) Disk Failure

- It occurs where hard-disk drives or storage drives used to fail frequently. It was a common problem in the early days of technology evolution.

- Disk failure occurs due to the formation of bad sectors, disk head crash, and unreachability to the disk or any other failure, which destroy all or part of disk storage.

Deferred Modification	Immediate Modification
• In deferred modification, the changes are not applied immediately to the database.	• In immediate modification, the changes are applied directly to the database.
• The log files contains all the changes that are to be applied to the database.	• The log file contains both old as well as new values.
• In this method once rollback is done all the records of log file are discarded and no changes are applied to the database.	• In this method once rollback is done the old values are restored into the database using the records of the log file
• Concepts of buffering and caching are used in deferred modification method.	• Concept of shadow paging is used in immediate modification method.
• It is called NO-UNDO/REDO technique.	• It is also called UNDO/REDO technique.
• The major disadvantage of this method is that it requires a lot of time for recovery in case of system failure.	• The major disadvantages of this method is that there are frequent I/O operations while the transaction is active.
• The deferred modification technique occurs if the transaction does not modify the database until it has	• The immediate modification technique occurs if database modification occurs while the transaction is st

6a) What do you mean by a schedule? When schedule is called serializable?

- A series of operation from one transaction to another transaction is known as schedule.
- It is used to preserve the order of the operation in each of the individual transaction.

Types of schedule in DBMS

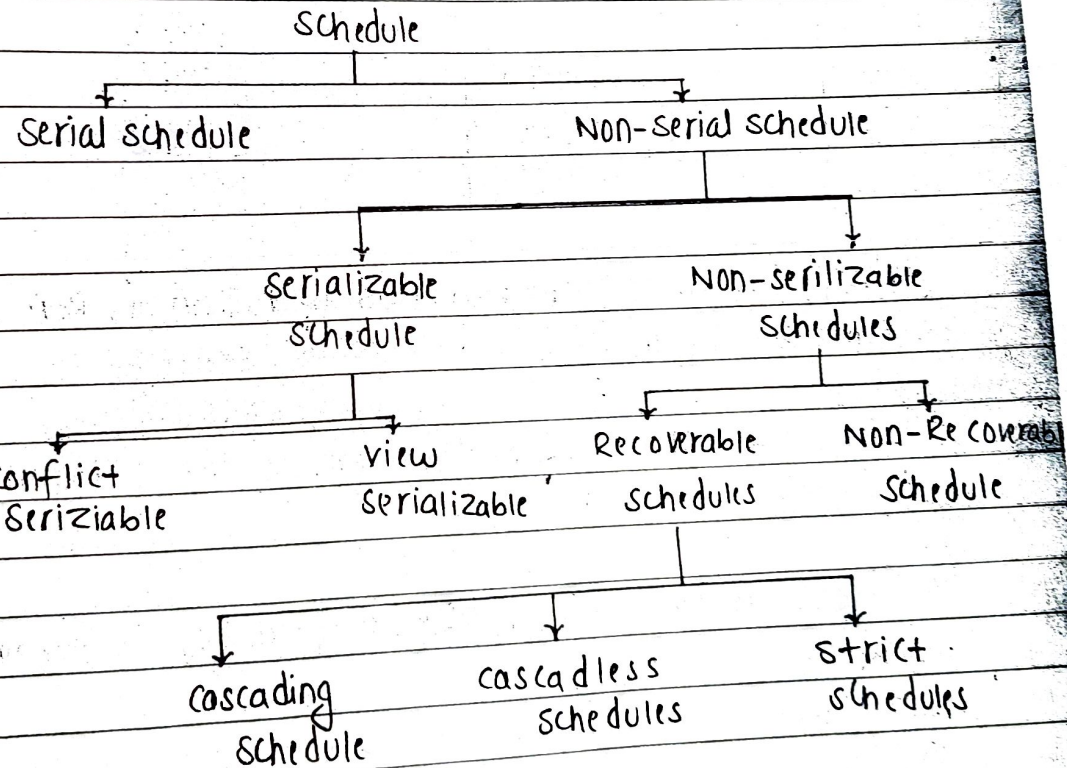


Fig:- Hierarchy of schedules.

i) serial schedules :-

- schedules in which the transactions are executed non-interleaved, i.e. a serial schedule is one in which no transaction starts until a running transaction has ended are called serial schedules.

eg :- consider the following schedule involving two Transactions T1 and T2.

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

where, R(A) denotes that a read operation is performed on some data item 'A'.

This is serial schedule since the transactions perform serially in the order T1 → T2

and R(B) denotes that a Read operation is performed on some data item A

similiarly,

W(A) denotes that a write operation is performed on some data A. and,

W(B) denotes that a write operation is performed on some data B.

ii) Non - Serial schedule.

- If interleaving of operations is allowed then there will be non-serial schedule.
- It contains many possible orders in which the system can execute the individual operations of the transactions.

eg:- Consider the following transaction T1 and T2 in schedule.

T1	T2
R(A)	
W(B)	
	R(A)
R(B)	
W(B)	
	commit
commit	

In this schedule,

- There are two transaction T1 and T2 executing concurrently.
- The operation of T1 and T2 are interleaved.
- So, the schedule is called non-serial schedule.
- Here, non-serial schedule is divided into many other hierarchy shown in fig above.

⇒ • A schedule is called serializable whenever executing the transactions sequentially, in order of some, could have left the database in the same state as the actual schedule. Serializability is the commonly accepted criterion for correctness.
• If the two transactions do not have operations on the same data item, so the schedule is serializable.

2012-spring

6a) How does 2PL gurantee serialization? (2 Marks)

• In case of two phase locking the main idea is to set lock before the transaction start and before releasing lock it will take all the lock at the end of the transaction and it helps transactions to be carried out in serial form that is how it guarantee serializability. In 2PL technique it has two states or situtation in which it perform locking mechanism, in order to perform concurrency control of the transactions and it will also keep the data histories,

i) Expanding phase :- In this phase only one process is adoptable means locking mechanism can be used but unlocking mechanism cannot be done if operation is in progress

ii) shrinking phase :- In this phase unlocking unlocking can be processed but locking cannot be processed by users when operation is under process or running.

66) What is log-based recovery? how it is different from shadow paging.

181546 35

66). The log is a sequence of records. Log of each transaction is maintained in some stable storage so that if any failure occurs, then it can be recovered from there.

- If any operation is performed on the database, then it will be recorded in the log.

- But the process of storing the logs should be done before the actual transaction is applied is applied in the database.

Let's assume, there is transaction to modify the city of a student. The following logs are written for this transaction.

- When the transaction is initiated, then it writes 'start' log.
  <Tn, start>

- When the transaction modifies the city from 'Noida' to 'Bangalore', then another log is written to the file.
  <Tn, city, 'Noida', 'Bangalore'>

- When the transaction is finished, then it writes another log to indicate the end of the transaction.
  <Tn, commit>

There are two approaches to modify the database:-
i) Deffered database modification ii) Immediate database modification

18/5/16 36

=> Recovery using log records

When the system is crashed, then the system consults the log to find which transactions need to be undone and which need to be redone.

i) If the log contains the record <Ti, start> and <Ti, commit> or <Ti, commit>, then the transaction 'Ti' needs to be redone.

ii) If log contains record <Tn, start> but does not contain the record either <Ti, commit> or <Ti, abort>, then the Transaction Ti needs to be undone.

Log Based recovery [redacted]

Log Based recovery

- one of the most widely used structure for database modifications recording is the log.
- Log can be defined as a sequence of log records, recording all the update activites in the database.
- Log record contains various fields such as transaction identifier, data item identifier, old value and new value.
- Whenever a transaction performs a write, it is essential that the log records for that write be created before the database is modified.

- Once a log record exists, we can output the modification to the database if that is desirable.
- Also, we have the ability to undo a modification that has already been output to the database.
- We undo it by using the old-value field in log records.

Shadow Paging

- Shadow paging considers the database to be made up of a number of fixed size disk pages.
- A directory with n entries is constructed, where the ith entry points to the ith database page on disk.
- The directory is kept in main memory if it is not too large, and all read and write references, to the database pages on disk go through it.
- When a transaction begins executing, the current directory, whose entries point to the most recent or current database pages on disk, is copied into a shadow directory. The shadow directory is then saved on disk while the current directory is used by the transaction.
- During transaction execution, the shadow directory is never modified. When a write_item operation is performed, a new copy of the modified database page is created, but the old copy of that page is not overwritten.

- Instead, the new page is written elsewhere, on some previously unused disk block. The current directory entry is modified to the new disk block, whereas the shadow directory is not modified and continues to point to the old unmodified disk block.

2013 Fall

5b) What is stable storage? (2 Marks)

- A stable storage is a storage in which information is never lost.

- Stable storage devices are the theoretically imposible to obtain. But, we must use some technique to design a stable storage system in which the chances of data loss are extremely low.

- Stable storage is a classification of computer data storage technology that guarantees atomicity for any given write operation and allows software to be written that is robust against some hardware and power failures.

- During the recovery from a failure each of the physical block is examined. The data present in the stable storage is safe unless a failure destroys all the copies.

- The data that is present in the stable storage is guaranteed to be safe unless a failure destroys all the copies.

6a) Describe the dead lock handling mechanism.

- Dead lock Handling
- system is deadlocked if there is a set of transactions such that every transaction in the set is waiting for another transaction in the set.
- Let T1 : Write(x) Write(y) T2: Write(y) Write(x)
- Now, schedule with deadlock

T1	T2
Lock-X(x)	
Write(x)	
	Lock-X(y)
	Write(y)
	{wait for Lock-X on x}
	Write(x)
{wait for Lock-X on y}	
Write(y)	

- To deal with deadlock we can use 1. Deadlock prevention protocol (which ensures that system will never enter a deadlock state) 2. Deadlock Detection and Recovery system once scheme (which try to recover system once it entered dead lock state).
- Both methods may result in transaction rollback.

- If probability of system entering deadlock state is relatively high, prevention is used. otherwise detection and recovery are more efficient.

- Deadlock prevention

- Deadlock prevention protocol ensure that the system will never enter into deadlock state.

- Some prevention strategies:

  - Requires that each transaction looks all data item before it begins execution.
  - Impose partial ordering of all data items. Requires that a transaction can lock data items only in order specified by partial order (e.g. graph based protocol).

  - Timeout based schemes: A transaction waits for a lock only for specified amount of time. After the wait time is out then transaction is roll back.
    - Simple to implement but starvation is possible.
    - Also difficult to determine good value of time out interval.

- Following schemes use transaction time stamps for the sake of deadlock prevention:

* Wait-Die scheme
  - Non preemptive
  - older transaction may wait for yonger one to release data item

- younger transactions never wait for older ones; they are rolled back instead.
- A transaction may die several times before acquiring needed data item.

* Wound-wait scheme
  - preemptive
  - older transaction may wait for younger one to release data may wounds (forces rollback) younger transactions instead of waiting for it.
  - younger transactions may wait for older ones.
  - May be fewer rollbacks than wait die scheme.

- In both schemes, a rolled back transaction is restarted with its original time stamp.

- Older transaction thus have precedence over newer ones in these schemes and starvation is hence avoided.

- Deadlock Detection and Recovery:
- It is used if no protocol is used to ensure deadlock freedom.

- Here, to determine whether deadlock occurs or not, some algorithms to check must be implemented.

- If deadlock occurs, then must recover from deadlock

Deadlock detection

Deadlocks can be described as a wait for graph which consists of a pair $G = (V, E)$ where 'V' is the set of vertices (transactions) and E is a set of edges; each edge is ordered pair $T_i \rightarrow T_j$

If $T_i \rightarrow T_j$, then there is a directed edge from T_i to T_j. Here, T_i is waiting for T_j to release data item.

When transaction T_i request a data item held by T_j then $T_i \rightarrow T_j$ is inserted in wait for graph. This edge is removed only when T_i no longer holding data item needed by T_j.

The system is in deadlock state if and only if wait for graph has a cycle.

The system invokes a deadlock detection algorithm periodically to look for a cycle.

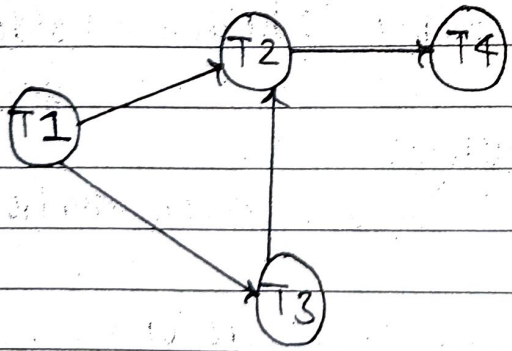


Fig:- wait for graph without cycle

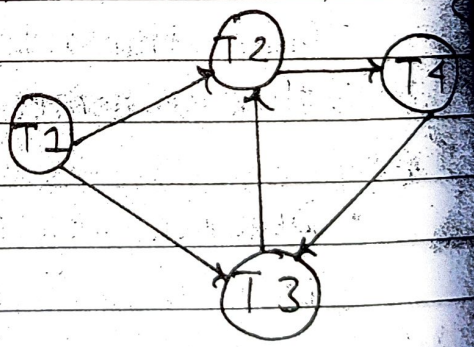


Fig:- wait for graph with cycle.

Deadlock Recovery.

When deadlock is detected:-

Deadlock detection

- Deadlocks can be described as a wait for graph which consists of a pair $G = (V, E)$ where 'V' is the set of vertices (transactions) and E is a set of edges; each edge is ordered pair $T_i \rightarrow T_j$
- If $T_i \rightarrow T_j$, then there is a directed edge from T_i to T_j. Here, T_i is waiting for T_j to release data item.
- When transaction T_i request a data item held by T_j then $T_i \rightarrow T_j$ is inserted in wait for graph. This edge is removed only when T_i no longer holding data item needed by T_j.
- The system is in deadlock state if and only if wait for graph has a cycle.
- The system invokes a deadlock detection algorithm periodically to look for a cycle.

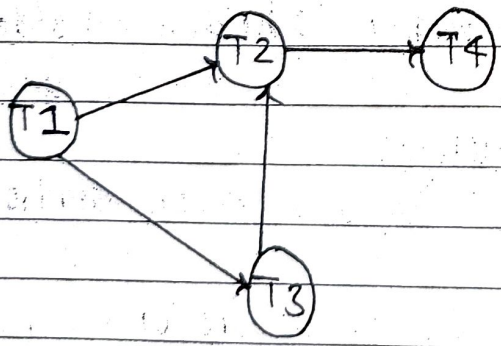


Fig:- wait for graph without cycle

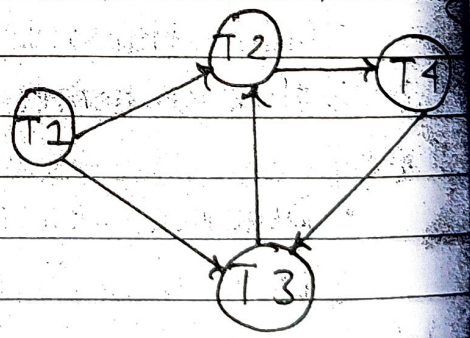


Fig:- wait for graph with cycle.

Deadlock Recovery.

- When deadlock is detected:-

- some transactions will have to rollback to break deadlock. (select a victim)
- Rollback – determine how far to rollback transaction.
  - Total Rollback : abort transaction and then restart it.
  - partial rollback : rollback transaction only as far as necessary to break the deadlock, and is more effective.
- Starvation happens if same transactions is always chosen as victim.
  - must ensure that transaction can be picked as a victim only a small no. of times.

2013 – spring

5a) Why database recovery is needed? (3)

- Data recovery can retrieve or recover the user's personal photos, files etc which are precious and important to them.
- This provides ease off any information or file getting lost. This will provide companies being run by computers to securely keep confidential files well backed up. No issues of information lost in any event after crashing or any virus or any event of device problems.
- It is important in sense that we are secured that even after crashing or any virus or malware attacks, our data could be still salvaged.

- It recovers lost or corrupted data in no time. The process is quick, efficient and restores data in a matter of minutes.
- This could serve as an additional storage device to companies.

6b) Advantages and disadvantages of 2PL over Single phase locking.

Advantages.

- Two transactions cannot have conflicting locks.
- No unlock operation can precede a lock operation in the same transaction.
- No data are affected until all locks are obtained - that is, until the transaction is in its locked point.
- 2PL gurantees serializability.

Disadvantage of 2PL

- It guarantees serializability but it does not prevent deadlocks.
- This protocol sometimes becomes overhead when a transaction traps in a deadline causing rolling back again and again.
- cascading rollback occurs.
- It leads to a lower degree of concurrency among transaction.

2014- Fall

6a) under which situations will be benificials to have replication or fragmentation of data? Explain with suitable example.

- Data replication is the process of storing data in more than one site or node.
- It is usefull in improving the availability of data. In this situation the data replication is important.
- It is simply copying data from a database from one server to another server so that all the users can share the same data without any inconsistency.
- Data replication encompasses duplication of transactions on an ongoing basis, so that the replicate is in a consistently updated state and synchronized with the source.
- However in data replication data is available at different locations, but a particular relation has to reside at only one location.

It is beneficial for

- Improved reliability and availability.
- Improved network performance.
- Increased data analytics support.
- Improved test system performance.

some commonly used replication technique are:-

- snapshot replication.
- Near-real-time replication.
- pull replication.

eg:- data can be copied between two on-premises hosts, between hosts in different locations, to multiple storage devices on the same host, or to or from a cloud-based host.

⇒ advantages of data replication.

- To provide a consistent copy of data across all the database nodes.
- To increase the availability of data.
- The reliability of data is increased through data replication
- Data replication supports multiple users and gives high performance
- To remove any data redundancy, the databases are updated with outdated or incomplete data.
- To perform faster execution of queries.
- since replicas are created there are chances that the data is found itself where the transaction is executing which reduces the data movement.

⇒ Disadvantages of data Replication.

- More storage space is needed as storing the replicas of same data at different sites consumes more space.
- Data replication becomes expensive when the replicas at all different sites need to be updated.
- maintaining Data consistency at all different sites involves complex measures.

Deferred Database Modification

- The deferred database modification scheme records all modifications to the log, but defers all the writes to after partial commit.
- Assume that the transaction execute serially
- Transaction starts by writing $\langle T_i \text{ start} \rangle$ record to log.
- A write(x) operations results in a log record $\langle T_i, x, V \rangle$ and old value is not needed in this scheme.
- The write is not performed on x at this time, but is deferred.
- When T_i partially commits, $\langle T_i \text{ commit} \rangle$ is written to the log
- Finally, the log records are read and used to actually execute the previously deferred writes.
- During recovery after a crash, a transaction needs to be redone if and only if both $\langle T_i \text{ start} \rangle$ and $\langle T_i \text{ commit} \rangle$ are there in the log.
- Redoing a Transaction T_i ($\text{redo } T_i$) sets the value of all data items updated by the transaction to the new values.
- Crashes can occur while
  - The transaction is executing the original updates or.
  - while recovery action is being taken.

- eg- Transaction T_0 and T_1 (T_0 executes before T_1) :-

T_0	T_1
read(A)	T1 = read (C);
A := A - 50;	C := C - 100;
write(A);	write(C);
read(B);	
B := B + 50;	
write(B);	

Immediate database modification.

- The Immediate database Modification technique allows database modifications to be output to the database while the transaction is still in the active state.

- update log record must be written before database item is written.
  - we assume that the log record is output directly to stable storage.
  - can be extended to postpone log records output, so long as prior to execution of an output(B) operations for a data Block B, all log records corresponding to items B must be flushed to stable storage.

- output of updated blocks can take place at any time before or after transaction commit.

- order in which blocks are output can be different from the order in which they are written.

181546 39

eg:- Immediate DB modification Recovery

Below we show the log as it appears at three instance of time.

(a)	(b)	(c)
<T0 start>	<T0 start>	<T0 start>
<T0, A, 1000, 950>	<T0, A, 1000, 950>	<T0, A, 1000, 950>
<T0, B, 2000, 2050>	<T0, B, 2000, 2050>	<T0, B, 2000, 2050>
	<T0 commit>	<T0 commit>
	<T1 start>	<T1 start>
	<T1, C, 700, 600>	<T1, C, 700, 600>
		<T1 commit

Recovery actions in each case above are:-

a) undo(T_0): B is restored to 2000 and A to 1000.

b) undo(T_1) and redo(T_0): C is restored to 700, and then A and B are set to 950 and 2050 respectively.

c) redo(T_0) and redo(T_1): A and B are set to 950 and 2050 respectively. Then C is set of to 600.

6b) Differences between Sharded lock and Exclusive lock.

Shared lock	Exclusive Lock
• Lock mode is read only operation.	• Lock mode is read as well as write operation.
• Shared lock can be placed on objects that do not have an exclusive lock already placed on them.	• Exclusive lock can only be placed on objects that do not have any other kind of lock.
• Prevents others from updating the data.	• Prevents others from reading or updating the data.
• Issued when transaction wants to read item that do not have an exclusive lock.	• Issued when transaction wants to update the unlocked item.
• Any number of transaction can hold shared lock on an item.	• Exclusive lock can be hold by only one transaction.
• Shared-lock is requested using lock-S instruction.	• Exclusive lock is requested using lock-X instruction.
• It is denoted as Lock-S.	• It is denoted as Lock-X.

2017 -spring

6a) Discuss the several issues that must be addressed while designing the Remote backup system.

- The several issues that must be addressed while designing Remote backup system are :-

i) Detection of failure :-

- As in failure - handling protocols for distributed system, it is important for the remote backup system to detect when the primary has failed.

- Failure of communication lines can fool remote backup system into believing that the primary has failed.

- To avoid this problem, we maintain several communication links with independent modes of failure between the primary and remote backup.

- For example, in addition to the network connection, there may be a separate modem connection, there may be a separatemodem connection over a telephone line, with services provided by different telecommunication companies. These connections may be the telephone system.

ii) Transfer of control :-

- When the primary fails, the backup site takes over Processing and becomes the new primary.

- When the original primary site recovers, it can either play the role of remote backup, or take over the role of primary site again.

- In either case, the old primary must receive a log of updates carried out by the backup site while the old primary was down.

- The simplest way of transferring control is for the old primary to receive redo logs from the old backup site, and to catch up with the updates by applying them locally.

- The old primary can then act as a remote backup site. If control must be transferred back, the old backup site can pretend to have failed, resulting in the old primary taking over.

iii) Time to recover

- If the log at the remote backup grows large, recovery will take a long time. The remote backup site can periodically process the redo log records that it has received and can perform a checkpoint, so that earlier parts of the log can be deleted. The delay before the remote backup takes over can be significantly reduced as a result.

- A hot-spare configuration can make takeover by the backup site almost instantaneous. In this configuration, the remote backup site continually processes redo log

records as they arrive, applying the updates locally.

- As soon as the failure of the primary is detected, the backup site completes recovery by rolling back incomplete transactions; it is then ready to process new transactions.

iv) Time to commit

- To ensure that the updates of a committed transaction are durable, a transaction must not be declared committed until its log records have reached the backup site.

- The delay can result in a longer wait to commit a transaction, and some systems therefore permit lower degrees of durability.

The degrees of durability can be classified as follows:-

a) one-safe:-

- A transaction commits as soon as its commit log record is written as its commit log stable storage at the primary site.

- The problem with this scheme is that the updates of a committed transaction may not have made it to the backup site, when the backup site takes over processing.

- Thus, the updates may appear to be lost. when the primary site recovers, the lost updates cannot be merged in directly, since the updates may conflict with later updates

performed at the backup site. Thus, human intervention may be required to bring the database to a consistent state.

b) Two-very-safe :-

- A transaction commits as soon as its commit log record is written to stable storage at the primary and the backup site.

- The problem with this scheme is that transaction processing cannot proceed if either the primary or the backup site is down.

- Thus, availability is actually less than in the single-site case, although the probability of data loss is much less.

c) Two-safe :-

- This scheme is the same as two-very-safe if both primary and backup sites are active. If only the primary is active, the transaction is allowed to commit as soon as its commit log record is written to stable storage at the primary site.

- This scheme provides better availability than does two-very-safe, while avoiding the problem of lost transactions faced by the one-safe scheme.

- It results in a slower commit than the one-safe scheme, but the benefits generally outweigh the cost.

2018 - Spring

6a) Explain the different states in a transaction (5 marks)

In a database, the transaction can be in one of the following states.

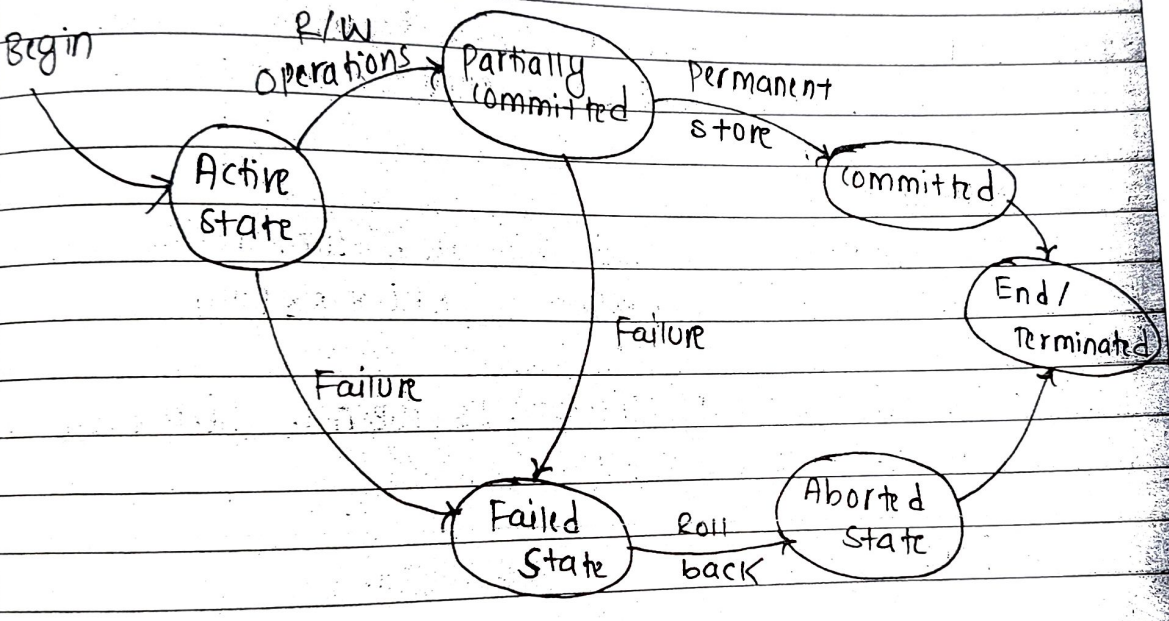


Fig:- Transaction States in DBMS

→ Active State

- The active state is the first state of every transaction. In this state, the transaction is being executed.
- Foreg :- Insertion or deletion or updating a record is done here. But all the records are still not saved to the database.

→ partially committed

- In the partially committed state, a transaction executes its final operation, but the data is still not saved to the database.

- In the total mark calculation example, a final display of the total marks step is executed in this state.

→ committed.

- A transaction is said to be in a committed state if it executes all its operations successfully.

- In this state, all the effects are now permanently saved on the database system.

→ Failed state

- If any of the checks made by the database recovery system fails, then the transaction is said to be in the failed state.

- In the example of total mark calculation, if the database is not able to fire a query to fetch the marks, then the transaction will fail to execute.

→ Aborted State

- If any of the checks fail and the transaction has reached a failed state then the database recovery system will make sure that the database is in its previous consistent state. If not then it will abort or roll back the transaction to bring the database into a consistent state.
- If the transaction fails in the middle of the transaction then before executing the transaction, all the executed transactions are rolled back to its consistent state.
- After aborting the transaction, the database recovery module will select one of the two operations:
  i) Re-start the transaction
  ii) Kill the transaction.

2016 - Fall

6a) When does deadlock occurs? [1 Marks]

- A deadlock occurs when 2 processes are completing for enclusive access to a resource but is unable to obtain enclusive access to it because the other process is preventing it.
- Deadlock can occur in a situation when a thread is waiting for an object lock, that is acquired by another thread and second thread is waiting for an object lock that is acquired by

6 b) What are data fragmentations ? State the various fragmentation with examples.

Data fragmentations

- Fragmentation is the task of dividing a table into a set of smaller tables.
- The subsets of the tables are called fragments.
- Fragmentation should be done in a way so that the original table can be reconstructed from the fragments.
- This is needed so that the original table can be reconstructed from the fragments whenever required. The requirement is called "reconstructiveness."

The types of data fragmentations are :-

i) Vertical Fragmentation :-

- In vertical fragmentation, the fields or columns of a table are grouped into fragments.
- In order to maintain reconstructiveness, each fragment should contain the primary key field(s) of the table.
- Vertical fragmentation can be used to enforce privacy of data.

- For eg:-

Student (Regd_NO, Name, course, Address, semester, Fees, Marks)

Now, fees are maintained in the accounts section. In this case, the designer will fragment the database as follows -

```
Create table std_Fees As select Regd_NO, Fees
  From Student;
```

ii) Horizontal Fragmentation groups the tuples of a table in accordance to values of one or more fields.

- It should also confirm to the rule of reconstructiveness. Each horizontal fragment must have all columns of the original base table.

- Horizontal fragmentation can further be divided into two techniques: primary Horizontal fragmentation and derived horizontal fragmentation.

- For example, the student scheme, if the details of all students of computer science course needs to be maintained at the school of computer science, then the designer will horizontally fragment the db as follows.

```
Create comp_std as
Select * From student
Where course = "computer science";
```

iii) Hybrid fragmentation

- In hybrid fragmentation, a combination of horizontal and vertical fragmentation technique are used.
- This is the most flexible fragmentation technique since it generates fragments with minimal extraneous information.
- However, reconstruction of the original table is often an expensive task.
- Hybrid fragmentation can be done in two alternative ways-
  - At first, generate a set of horizontal fragments; then generate vertical fragments from one or more of the horizontal fragments.
  - At first, generate a set of vertical fragments; then generate horizontal fragments from one or more of the vertical fragments.

⇒ Advantages of Fragmentation.

- Since data is stored close to the site of usage, efficiency of the database system is increased.
- Local query optimization techniques are sufficient for most

queries since d

since, irrelevan
security an
maintained

⇒ Disadvantage

- When data
access spe
- In case of
will need
- Lack of
may rend
failure o

2019-SP

6b) When th
State?

- When e
that
and
the u
which
denote
are

queries since data is locally available.

- since, irrelevant data is not available at the sites, security and privacy of the database system can be maintained.

⇒ Disadvantages of Fragmentation.

- When data from different fragments are required, the access speeds may be very high.
- In case of recursive fragmentations, the job of reconstruction will need expensive techniques.
- Lack of back-up copies of data in different sites may render the database ineffective in case of failure of a site.

2019-spring

When the two transaction are said to be in deadlock state?

- When each transaction is waiting for a data item that is being locked by some other transaction and a deadlock can be indicated by a cycle in the wait-for-graph where it is a directed graph in which the vertices denote transactions and the edges denote waits for data items then two transaction are said to be in deadlock state.

A deadlock occurs when the first process locks the first resource at the same time as the second process locks the second resource. The deadlock can be resolved by cancelling and restarting the first process.

2020 Fall 2020 Fall

6b) What are the various crash recovery algorithm?

- Crash recovery includes three phases.

- Analysis :- Determines which transactions committed since checkpoint and which ones failed.
- REDO all actions.
  - (repeat history)
- UNDO effects of uncommitted transactions (the active transactions at the time of the crash.

Crash recovery phases.