GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

PRESIDENCY COLLEGE
(Autonomous)

DATABASE MANAGEMENT SYSTEM

Reaccredited by NAAC with A+

By
Mr. N Kartik / Mr. Jeelan

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

PRESIDENCY COLLEGE
(AUTONOMOUS)

AFFILIATED TO BENGALURU CITY UNIVERSITY, APPROVED BY AICTE, DELHI & RECOGNISED BY THE GOVT. OF KARNATAKA
RE-ACCREDITED BY NAAC WITH 'A+' GRADE

Presidency Group
OVER 40 YEARS OF ACADEMIC WISDOM

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

Presidency College
(Autonomous)

Reaccredited by NAAC with A+

Presidency Group
OVER 40 YEARS OF ACADEMIC WISDOM

Syllabus

- Transaction and system concepts
- Desirable properties of transactions
- Transaction support in SQL.
- Concurrency control techniques: two-phase locking techniques, concurrency control based on timestamp ordering.
- Recovery techniques: recovery concepts, recovery in multi-database systems, database backup and recovery.

UNIT 04
Transaction Processing Concept

What is Transaction?

- A transaction is an atomic unit comprised of one or more SQL statements.
- A transaction begins with the first executable statements and ends when it is committed or rollback.

Transaction and system concepts

- A transaction is a logical unit of database processing that includes one or more database access operation.
- **Example of transaction processing system:**

1. Reservation systems.
2. Credit card processing system
3. Stock market processing system.
4. Supermarket processing system
5. Insurance processing system etc..

Singile user vs Multiuser Systems

- **Single-User** : at most one user at a time can use the system
- **Multiuser** : many users can use the system concurrently.

Desirable Properties of transactions or ACID properties of transactions

ACID should be enforced by the concurrency control and recovery methods of the DBMS.

ACID properties of transactions :

1.Atomicity : A transaction is an atomic unit of processing; it is either performed entirely or not performed at all.

2.Consistency : Transaction must preserve database consistency.

A transaction transform the database from one consistent state to another consistent state.

Desirable Properties of transactions (continued)

3.Isolation :

- The execution of the transaction should be isolated from other transactions (Locking).
- No inference with other transactions.

4.Durability :

Once a transation completes, the changes made to database permanent and are available to all the transactions that follow it.

What is concurrency control?

- In a multiprogramming environment where multiple transactions can be executed simultaneously, it is highly important to control the concurrency of transactions.
- We have concurrency control protocols to ensure atomicity, isolation, and serializability of concurrent transactions.

- Concurrency control protocols can be broadly divided into two categories –

i. Lock based protocols

ii. Timestamp based protocols

Advantages of concurrent execution:

1.Increased performance

2.Resource utilization.

3.Decreased waiting time

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

Presidency College
(Autonomous)

Reaccredited by NAAC with A+

Presidency Group

OVER 40 YEARS OF ACADEMIC WISDOM

Why is concurrency control needed?

In a multiuser database, transaction submitted by the various user may execute concurrently and many update the same data concurrently executing transactions must be guaranteed to produce the same effect as serial execution of transaction (one by one).

****Problems with concurrent execution

The problems that occur two transactions run concurrently are.

1. **The Lost update problem.**
2. **Temporary update (or Dirty read) problem.**
3. **The incorrect summary problem.**

***The Lost update problem:

Suppose transaction T1 and T2 are submitted at the same time, when these two transactions are executed concurrently then the final value of X is incorrect because T2 reads the value of x before T1 changes it in the database and hence the updated value resulting from T1 is lost.

Example: x=100 at the start (100 reservation at the beginning), n=10 (T1 updated with 10 seats reservations from flight Y to X) and m=20 (T2 transfer 20 seats on y), the final result should be x =90 but due to interleaving of operations x = 80 because T1 updating that added 10 seats from Y was Lost.

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

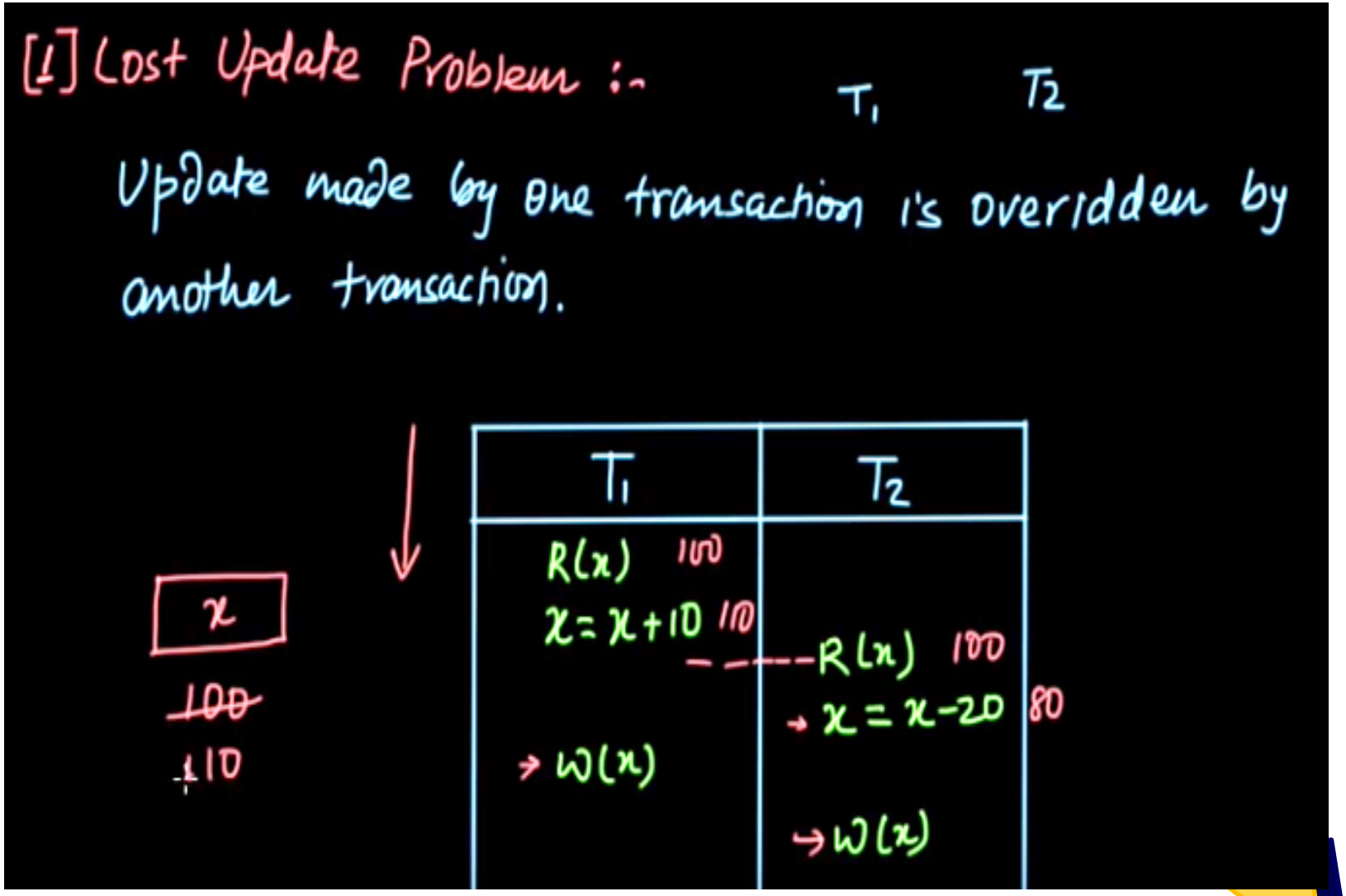
Presidency College
(Autonomous)

Reaccredited by NAAC with A+

Presidency Group

OVER 40 YEARS OF ACADEMIC WISDOM

Example

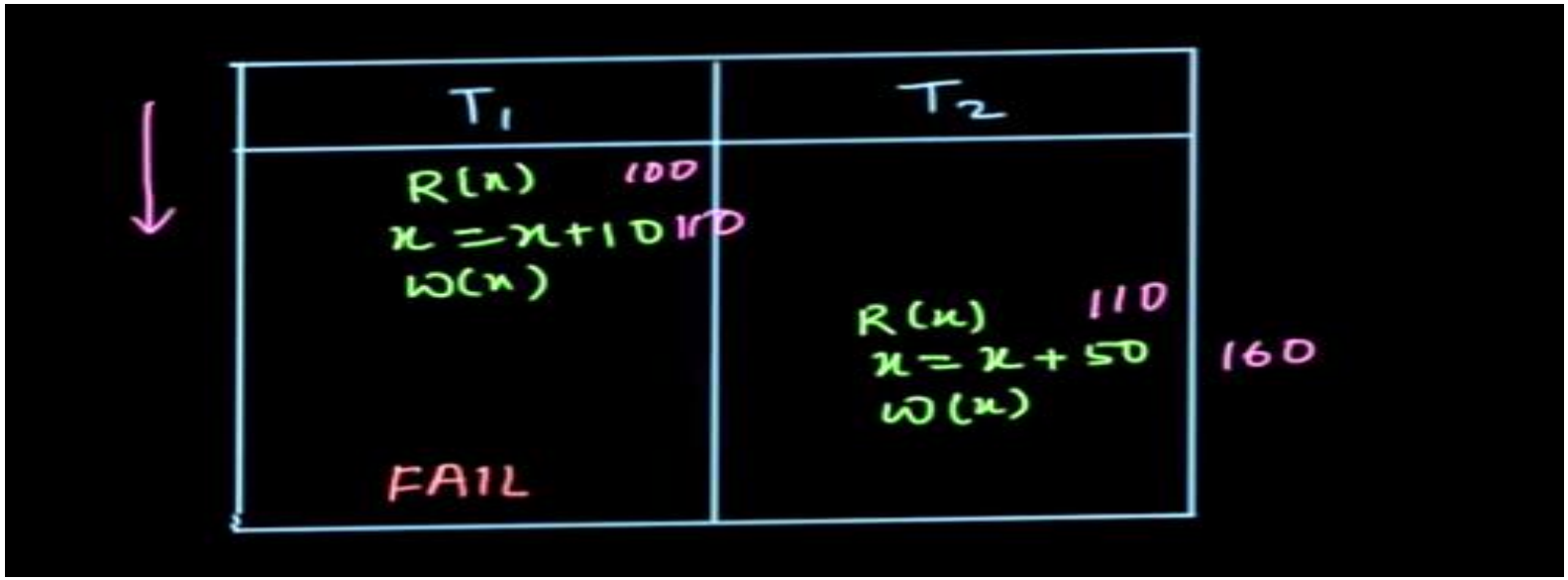


***2,Dirty read Problem:

This operation occurs when one transaction updates a database item and then the transaction fails for some reason, the update item is accessed by another transaction before it is changed back to its original value.

- **Example:** T1 updates item x and then fails before completion, so the system must change x back to original value, before it can be do so, however, transaction T2 reads the temporary value of x, which will be no record permanently in the database because of the failure of T1. The value of item x that is read by T2 is called **Dirty data.** Because it has been created by a transaction that has not completed and committed yet, hence this problem is also known as the temporary update problem.

Example for dirty read problem



GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

Presidency College
(Autonomous)

Reaccredited by NAAC with A+

Presidency Group

OVER 40 YEARS OF ACADEMIC WISDOM

***3.Incorrect summary problem:

- If one transaction is calculating an aggregate summary function on several records while other transaction are updating some of the records, the aggregate functions may calculate some values before they are update and others after they are updated.

Example: Transaction T3 is calculating the total number of reservations on all the flights, mean while transaction T1 is executing. The T3 reads the value of x after n seats have been subtracted from it but reads the value of y before those n seats have been added to it.

T1	T3	
	Sum:=0 read_item(a) sum := sum+a;	

read_item(x);
x:=x-N;
write_item(x);		

	read_iterm(x); sum:=sum+x;	T3 reads X after N is subtracted and reads y before N is added.
	read_item(y); sum:= sum+y;	

read_item(y);
y:=y+N;
write_item(y);		

	-a wrong summary is the result.	
fig(a): Transaction [t1]	fig(b) Transaction[T2].	

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

Presidency College
(Autonomous)

Reaccredited by NAAC with A+

Presidency Group

OVER 40 YEARS OF ACADEMIC WISDOM

****Concurrency control techniques

Some of the main techniques used to control concurrent execution of transaction are based on the concept of locking data items.

- A **lock** is a restriction on access to the data in a multiuser environment.
- It prevents multiple users from changing the same data simultaneously.

Types of LOCKs can be used, they are,

1. Binary LOCK.
2. Shared LOCK.
3. Exclusive lock

1.Binary LOCK: A binary lock can have two states or values:

i)Locked (1)

ii)Unlocked(0)

A distinct lock is associated with each database item *A.*

If the value of the lock on *A* is 1, item *A* cannot be accessed by a database operation that requests the item. If the value of the lock on *A* is 0 then item can be accessed when requested.

Locking Technique

2.Shared lock (s):

It is used for read only operations. i.e., Used for operations that does not change or update the data.

Once a transaction puts the **s lock** on a particular resource, only read can be performed by the other transaction ,modification or updates not possible to perform.

3.Exclusive LOCK: Exclusive locks are used for data **modification operations** such as update, delete and insert.

Once a transaction puts the X lock on a particular resource, no other transaction can put any kind of lock on this resource.

This resource is exclusively reserved for the first transaction and no other transaction can use it for read or write operation.

Hence X lock allows least concurrency

Locking Technique

– The effect of a lock is to lock other transaction out of the object.

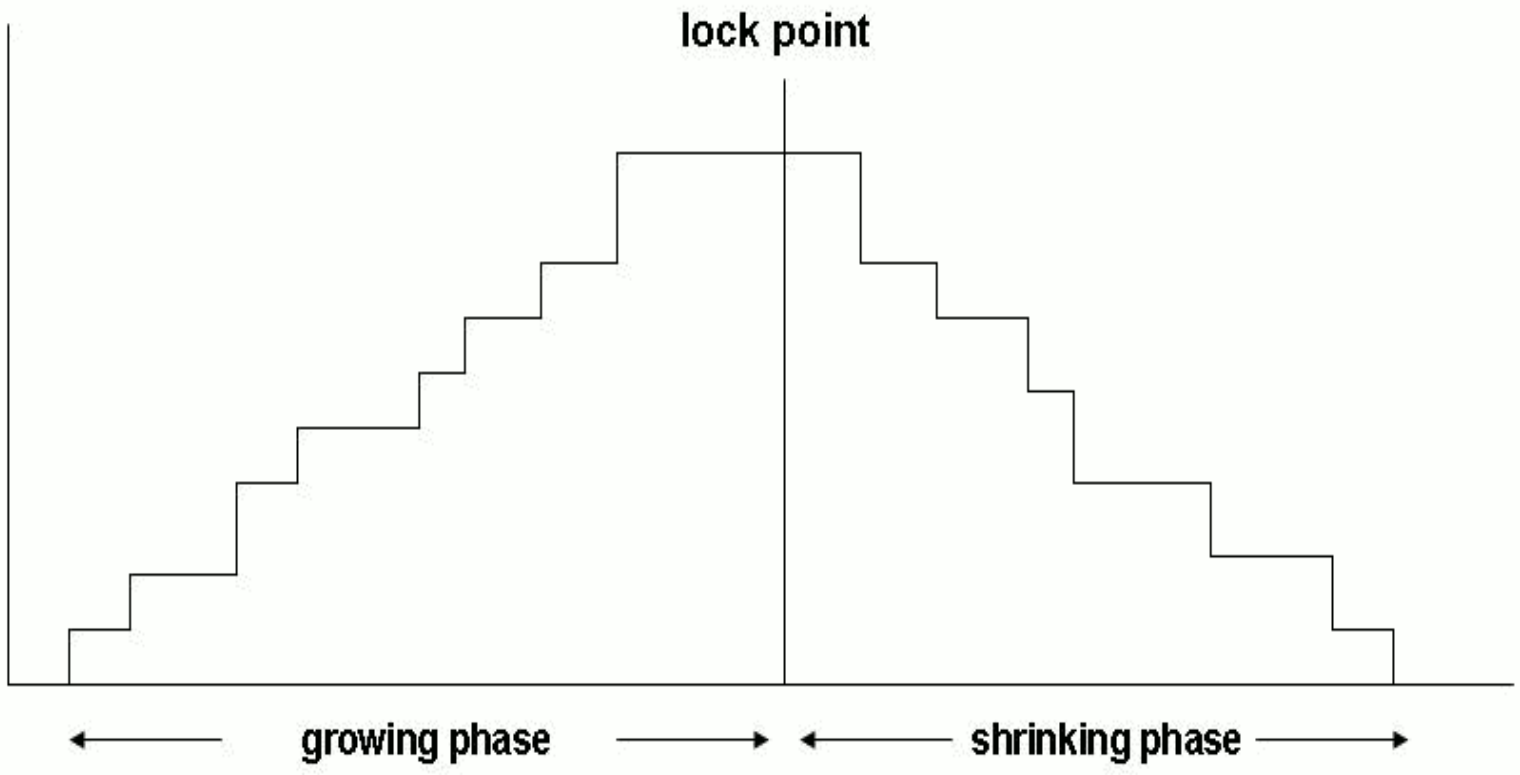
A \ B	X	S	—
X	N	N	Y
S	N	Y	Y
—	Y	Y	Y

- Compatibility matrix

— : no lock
N : request not compatible
Y : request compatible

The two-Phase locking(2PL) protocol

- A transaction is said to follow the Two-Phase Locking protocol if Locking and Unlocking can be done in two phases.
- **Growing Phase** In this phase we put read or write lock based on need on the data. In this phase we do not release any lock.
- New locks on data items may be acquired but none can be released.
- **Shrinking Phase** This phase is just reverse of growing phase. In this phase we release read and write lock but doesn't put any lock on data.
- Existing locks may be released but no new locks can be acquired.



EXAMPLE

	T_1	T_2
1	lock-S(A)	
2		lock-S(A)
3	lock-X(B)	
4	.....	.....
5	Unlock(A)	
6		Lock-X(C)
7	Unlock(B)	
8		Unlock(A)
9		Unlock(C)
10	.....	.....

Transaction T_1:
The growing Phase is from steps 1-3.
The shrinking Phase is from steps 5-7.
Lock Point at 3

Transaction T_2:
The growing Phase is from steps 2-6.
The shrinking Phase is from steps 8-9.
Lock Point at 6

What is **LOCK POINT?** The Point at which the growing phase ends, i.e., when a transaction takes the final lock, it needs to carry on its work.

Timestamp:

Timestamp is a unique identifier created by the DBMS to identify a transaction. Timestamp values are assigned in the order in which the transactions are submitted to the system. A timestamp of transaction T can be referred by TS(T).

The idea for this scheme is to order the transactions based on their timestamps.

TIME	10:00	10:02	10:05	10:20
TRANSACTION	T1	T2	T3	T4
TIMESTAMP	100	200	300	400
	OLDER			YOUNGER

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

Presidency College
(Autonomous)

Reaccredited by NAAC with A+

Presidency Group

OVER 40 YEARS OF ACADEMIC WISDOM

RTS= 30 WTS= 20

10	20	30
T1	T2	T3
R(A)		
	R(A)	
		R(A)

10	20	30
T1	T2	T3
		W(A)
W(A)		
	W(A)	

Time Stamp Ordering Algorithm

- **The Algorithm must ensure that for each item accessed by competing operati**ons in the schedule, the order in which the item is accessed does not violate the serializability order. To do this, the algorithm associates with each database item X two timestamp (TS) VALUES:
- **1. Read –RTS (X):** The read timestamp of item X; this is the largest time stamp among all the timestamps of transactions that have successfully read item – X
- i.e. Read – RTS (X) = TS (T). Where T is the youngest transaction that has read X successfully.
- **2. Write – WTS (X) :** The write timestamp of item X; this is the largest of all the timestamps of transactions that have successfully written item X
- – i.e., write –WTS (X) = TS(T), where T is the youngest transaction that has written X successfully.

Basic Time Stamp Ordering

- Wherever some transaction T tries to issue a read- item (X) or write-item (X) operation, the basic TO (Time Ordering) compares the timestamp of T with read – TS (X) and write – TS (X) to ensure that the timestamp order of transaction is not violate.
- If this order is violated, then transaction T is aborted, resubmitted to the system as a new transaction with a new timestamp.
- If T is aborted and rolled back , any transaction T1 that may have used a value written by T must also be rolled back similarly, any transaction T2 that may have used a value written by T1 must also be rolled back, and so on.
- **This effect is known as cascading rollback and** is one of the problems associated with basic TO, since the schedules produced are not recoverable, codeless or stick.

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

PRESIDENCY COLLEGE
(Autonomous)

Recovery techniques: recovery concepts, recovery in multi-database systems, database backup and recovery.

Reaccredited by NAAC with A+

Presidency Group

OVER 40 YEARS OF ACADEMIC WISDOM

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

PRESIDENCY COLLEGE
(AUTONOMOUS)

AFFILIATED TO BENGALURU CITY UNIVERSITY, APPROVED BY AICTE, DELHI & RECOGNISED BY THE GOVT. OF KARNATAKA
RE-ACCREDITED BY NAAC WITH 'A+' GRADE

DataBase Failures(Transaction Failures)

Database Failures are generally classified into three types:

1 .Transaction failure

a)Transaction or system error

b)Local errors or exception conditions deleted by the transaction

c)Concurrency control enforcement

2.System failure (system crash)

a)System crash(Computer Failure)

b)Physical problems

3.Media failure

Disk Failure: Some disk block may lose their data because of a read or write malfunctions.

Database Failures(Transaction Failures)

Transaction Failures

- **Transaction or system error:** Some operation in the transaction may cause it to fail, such as integer overflow or division by ‘zero’ etc.
- **Local errors or exception conditions deleted by the transaction:** During transaction execution, certain conditions may occur that perform cancelation of the transaction. For example: data for the transaction may not be found.
- **Concurrency control enforcement:** The concurrency control method may decide to abort the transaction to be restarted later, because several transactions are in a state of dead lock.

System failure

- **A computer failure (system crash):** Hardware, software and network error occurs in the computer system during transaction execution.
- **Physical problems :** This refers to list of problem that includes power or air conditioning failure, fire, theft, overwriting disks and etc.

Why is recovery needed?

A major responsibility of database administrator is to prepare for the possibilities of hardware, software, network and system failure. It is usually desirable to recover the databases and return to normal operation as quickly as possible, recovery should process in such a manner to protect the database and users from unnecessary problem.

Whenever a transaction is submitted to a DBMS for execution the system is responsible for making sure that either,

- All the operations in the transaction are completed successfully and their effects is recorded permanently in the database or,

- The transaction has no effect on the database; this may happen if a transaction fails after executing some of its operations but before executing all of them.

Recovery Technique: Recovery in Multidatabase Systems

- In some cases, a single transaction, called a **multidatabase transaction**, may require access to multiple databases.
- These databases may even be stored on different types of DBMSs; for example, some DBMSs may be relational, whereas others are object-oriented, hierarchical, or network DBMSs.
- To maintain the atomicity of a multidatabase transaction, it is necessary to have a two-level recovery mechanism.
- A **global recovery manager**, or **coordinator**, is needed to maintain information needed for recovery, in addition to the local recovery managers and the information they maintain (log, tables).
- The coordinator usually follows a protocol called the **two-phase commit protocol**, whose two phases can be stated as follows:

Phase 1. Voting Phase

- When all participating databases signal the coordinator that the part of the multidatabase transaction involving each has concluded
- The coordinator sends a message *prepare for commit* to each participant to get ready for committing the transaction.
- Each participating database receiving that message will force-write all log records and needed information for local recovery to disk and then send a *ready to* ***commit*** or ***OK*** **signal** to the coordinator.
- **FAILURE:** If the force-writing to disk fails or the local transaction cannot commit for some reason, the participating database sends a *cannot commit* or *not OK* signal to the coordinator.
- **Response timeout:** If the coordinator does not receive a reply from the database within a certain time out interval, it assumes a *not OK* response.

Phase 2. COMMIT PHASE

- If *all* participating databases reply *OK*, and the coordinator's vote is also *OK*, the transaction is successful, and the coordinator sends a *commit* signal for the transaction to the participating databases.
- Because all the local effects of the transaction and information needed for local recovery have been recorded in the logs of the participating databases, recovery from failure is now possible.
- Each participating database completes transaction commit by writing a [commit] entry for the transaction in the log and permanently updating the database if needed.
- On the other hand, if one or more of the participating databases or the coordinator have a *not OK* response, the transaction has failed, and the coordinator sends a message to *roll back* or UNDO the local effect of the transaction to each participating database. This is done by undoing the transaction operations, using the log.

The net effect of the two-phase commit protocol

1. participating data-bases commit the effect of the transaction or

2. none of them do.

- In case any of the participants—or the coordinator—fails, it is always possible to recover to a state where either the transaction is committed, or it is rolled back.

1. A failure during or before Phase 1 usually requires the transaction to be rolled back
2. A failure during Phase 2 means that a successful transaction can recover and commit.

Database backup and recovery.

Reasons of Failure in a Database

There can be multiple reasons of failure in a database because of which a database backup and recovery plan is required. Some of these reasons are:

- **User Error -** Normally, user error is the biggest reason of data destruction or corruption in a database. To rectify the error, the database needs to be restored to the point in time before the error occurred.
- **Hardware Failure -** This can also lead to loss of data in a database. The database is stored on multiple hard drives across various locations. These hard drives may sometimes malfunction leading to database corruption. So, it is important to periodically change them.
- **Catastrophic Event -** A catastrophic event can be a natural calamity like a flood or earthquake or deliberate sabotage such as hacking of the database. Either way, the database data may be corrupted, and backup may be required.

Methods of Backup

The different methods of backup in a database are:

- **Full Backup -** This method takes a lot of time as the full copy of the database is made including the data and the transaction records.
- **Transaction Log -** Only the transaction logs are saved as the backup in this method. To keep the backup file as small as possible, the previous transaction log details are deleted once a new backup record is made.
- **Differential Backup -** This is like full backup in that it stores both the data and the transaction records. However only that information is saved in the backup that has changed since the last full backup. Because of this, differential backup leads to smaller files.

Database Recovery

There are two methods that are primarily used for database recovery.

- **Log based recovery -** In log-based recovery, logs of all database transactions are stored in a secure area so that in case of a system failure, the database can recover the data. All log information, such as the time of the transaction, its data etc. should be stored before the transaction is executed.
- **Shadow paging -** In shadow paging, after the transaction is completed, its data is automatically stored for safekeeping. So, if the system crashes in the middle of a transaction, changes made by it will not be reflected in the database.

LOG EXAMPLE

In ATM withdrawal, we had four steps:

- Check the Account for balance in his account : T1
- If enough exists, give the money to user : T2
- Calculate balance = balance – 500 : T3
- Update account to new balance: T4

T1			T2			T3			T4		
Before	During	After	Before	During	After	Before	During	After	Before	During	After
balance = 2000	balance = 2000	balance = 2000	balance = 2000	balance = 2000	balance = 2000	balance = 2000	balance = 2000 - 500	balance = 2000 - 500	balance = 2000	balance = 1500	balance = 1500
user = 0	user = 0	user = 0	user = 0	user = 0	user = 500	user = 500	user = 500	user = 500	user = 500	user = 500	user = 500
Begin	Exec	End	Begin	Exec	End	Begin	Fail				