Assignment No:- 4

Q(1) What are the ACID properties of transaction

Ans:-

- The ACID properties are set of four key properties that ensure reliability, consistency, and correctness of database transactions.
- The ACID properties ensures that the database remains accurate even in cases of errors, power failures or crashes.

① Atomicity :- All or Nothing

- A transaction is atomic meaning it must be fully completed or not done at all.
- If any part of the transaction fails, the whole transaction is rolled back.
- eg:- Transferring ₹1000 from Acc A to Acc B
  step 1: Deduct ₹1000 from A
  step 2: Add ₹1000 to B
  if step 2 fails, step 1 is undone so no money is lost.

② Consistency :- Valid State

- A transaction should bring the database from one valid state to another valid state, following all rules & constraints
- The database should never be left in a invalid or corrupted state.
- eg:- If a rule says that the account balance can't be -ve, a transaction that

tries to withdraw more than the available balance will be rejected.

③ Isolation :- Transaction Don't Interfere
- If multiple transaction occur at the same time they should not affect each others outcome
- Each transaction behaves as if its the only one running until its completed
- eg:- Two customers withdrawing ₹500 each from the same account at same time
  - without isolation : both may see same balance ₹1000 & withdraw, causing an incorrect balance.
  - with isolation : one withdrawal completes first before the other starts.

④ Durability - Permanent changes
- once a transaction is committed, the changes are permenant even if there is a power failure or system crash afterward.
- eg:-
  If payment transaction is completed & confirmed the updated balance remains in the database even after a power outage.

Q(2) Explain the concept of recoverable and non-recoverable schedules with suitable example

Ans:-

- A schedule defines the order in which operations (like read and write) from different transaction are executed.
- When multiple transaction run together the way their operations are inter-leaved affected consistency and recovery after a failure.

① Recoverable schedule :-

- A schedule is recoverable if a transaction commits only after all the transactions whose data it has read have commited.
- eg:- Suppose we have 2 transaction T1 and T2 operating on data item x.

Step	Operation
1	T1 : write(x)
2	T2 : read(x)
3	T1 : commit
4	T2 : commit

- T2 reads the value of x written by T1
- But T2 commits after T1 commits → no problem.

- If T1 fails before commit, T2 can also be rolled back since it hasn't commited yet.
- This schedule is recoverable

② Non-recoverable Schedule :-

- If schedule is non-recoverable if a transaction commits before the transaction whose data it read commits.
- If the first transaction later fails, we can't roll back the already commited transaction - leading to inconsistency.

- eg :-

step	Operation
1	T1 : write(x)
2	T2 : read(x)
3	T2 : commit
4	T1 : abort

- T2 reads the value of x written by T1
- T2 commits before T1 comits → violation
- If T1 aborts (roll back), T2 has already commited with an invalid value of x.
- The database becomes inconsistent

- This schedule is non-recoverable

Q (3) Define the following terms

① Concurrency :-

- Concurrency in DBMS refers to the simultaneous execution of multiple transactions in multi-user system without interfering with each other.
- Its goal is to maximize resource utilization and improve system performance while ensuring data consistency.
- eg:- Two bank transactions ruming at the same time
  - > T1 deposits ₹500 into an account
  - > T2 checks the account balance.

② Timestamp :-

- A timestamp is a unique identifier assigned to each transaction or operation to indicate the order of their arrival or execution.
- It helps DBMS to maintain the correct seqn of execution for concurrent transaction
- eg:- T1 starts at time 5 → timestamp = 5
  T2 starts at time 8 → timestamp = 8
  T1 is considered older than T2

(3) Timestamp ordering :-

- Timestamp ordering is a concurrency control protocol that uses timestamps to decide the order in which conflicting operation of concurrent transactions should be executed.
- It ensures that the final results of concurrent execution is the same as if the transaction were executed in timestamp order.

(4) Schedule :-

- A schedule is the sequence of execution of operations from one or more transaction
- Types :- 1) Serial schedule
  2) Concurrent schedule
- eg :-

(5) Transaction :-

- A transaction is a single logical unit of work that consists of one or more operations performed on the database.
- Transactions must satisfy the ACID properties.
- eg :- In bank system.

Q.(4) In database transaction management. what is role of schedule? Explain serial schedule in details

Ans:-

* Role of Schedule in Transaction Management

- In a database system, multiple transaction may execute concurrently to improve system performance & resource utilization. However, concurrency can cause conflicts (e.g:- lost updates, dirty reads)

- To manage this, the DBMS defines the order of execution of operations (like read/write) of different transaction.

- This order seq^n is called as schedule Schedule tells us when and in what order the operations (read, write, commit, abort) of multiple transactions are executed. It is used to analyze whether transaction maintain consistency and isolation. The DBMS checks if the schedule is serializable, meaning it ensures the same results as some serial execution of transactions.

- Serial schedule :-

- A serial schedule is the simplest form of scheduling.
- A schedule is serial if all operations of one transaction are executed completely before starting another transaction.
- No interleaving of instructions is allowed.
- Transactions executes one after another like a queue.

- eg :- Let's say we have two transaction T1 and T2 :
  - T1 : Read(A), A = A-50, Write(A)
  - T2 : Read(A), A = A x 2, Write(A)

Possible serial schedules :

1. T1 followed by T2

Schedule S1 :

T1 : Read(A), A = A - 50, Write(A)
T2 : Read(A), A = A x 2, Write(A)

2. T2 followed by T1 :-

Schedule S2 :

T2 : Read(A), A = A x 2, Write(A)
T1 : Read(A), A = A - 50, Write(A)

- Both cases, transactions execute without overlapping.

Q. (5) What is need of concurrency control in database management? Explain locking method and deadlock handling.

Ans:-

* Need of Concurrency Control in DBMS :-
In a database, many transactions may run concurrently (at same time).
If concurrency is not controlled properly, it can cause inconsistencies in database.

Reasons :-
1) Avoid Problems of concurrent Execution
2) Maintain ACID properties.
3) Ensure correctness of Transactions.

* Locking Methods :-

Locking is the most common concurrency control technique.
It restricts access to a data item so that only one transaction can access it in a particular mode at a time.

Types :-
1) Shared Lock (S-Lock) :-
Used when a transaction wants to read a data item.
2) Exclusive Lock (X-Lock) :-
Used when a transaction wants to write/update a data item.

* ~~Locking~~ Protocols:- ~~Deadlock in Locking~~:-

* When Deadlock Handling

1. Deadlock :- When two or more transaction wait for each other forever due to locked resources, it creates a Deadlock.

2. Deadlock Prevention :-
- Design system so that deadlock never occurs.

- Methods:
  - Ordering of Resources: Always acquire locks in a predefined order.
  - Wait - Die scheme : older transaction wait younger are rolled back.

3. Deadlock Avoidance
- The system makes decisions at runtime to avoid unsafe states
- Banker's Algorithm can be used to check if granting a lock keeps the ~~syt~~ system safe.

Q.6) Explain the transaction states with state diagram.

Ans:-

* Transaction states in DBMS :-

A transaction is a seq of op performed as a slg logical unit of work. To ensure ACID properties the DBMS monitors the state of a transaction during its lifetime.

States are :-

1) Active :-
- The transaction starts execution.
- Operations like READ, WRITE are performed
- If everything goes fine, it moves to Partially Commited.
- If an error occurs it moves to Failed

2) Partially Committed
- After the final statement of the transaction is executed but before all changes are made permanent
- If commit succeeds → move to Commited
- If crash occurs here → may move to Failed

3) Commited :-
- When all operations are succesful and the database changes are permanently stored.

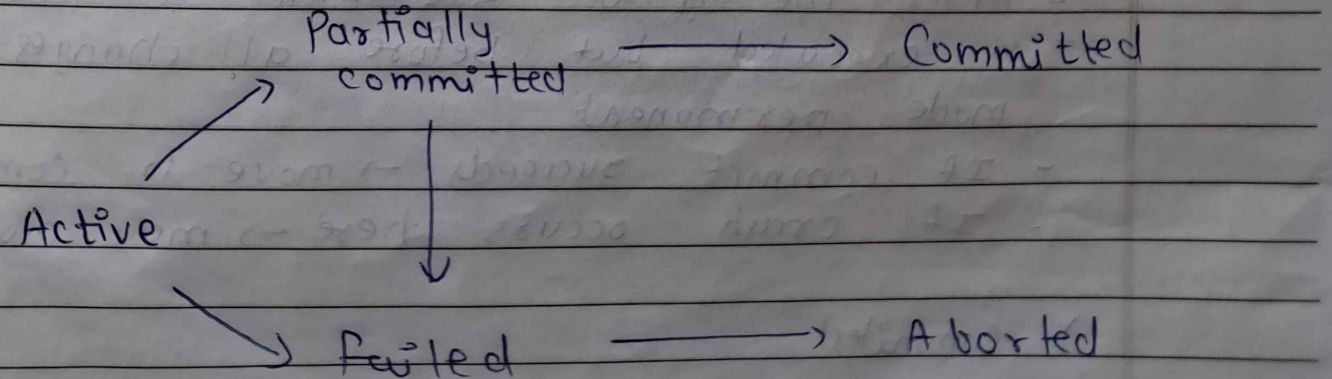
- The transaction can not be rolled back after this pt.

4) Failed state
- If some error occurs (system crash, deadlock, or invalid op.)
- The transaction can no longer proceed.

5) Aborted state :-
- When a transaction is rolled back due to failure.
- Changes made so far are undone

6) Terminated state :-
- Final state after a transaction is successfully committed or rolled back.
- Transaction ends and all resources are released.

* Transaction state Diagram :-



Q. (7) Identify the following schedule is view serializable or not. Justify answer.

T1	T2	T3
R(x)		
	W(x)	
W(y)		W(z)
W(x)		
		W(x)

Ans :-

The schedule is view-serializable, equivalent to serial order T1 → T2 → T3 but not conflict-serializable.

Justification :-

1) Same 'read-from' relationships:
R1(x) happens before any write on x in the schedule, so it reads the initial value of x (call it x0).
In any serial schedule view-equivalent to this, T1 must appear before every transaction that writes x so that R1(x) still reads x0.
~~So~~ So, T1 before T2 & T3

2) Same final-writer for each item.
Final write on x here W3(x) → T3 must be

the last x-writer.
Final write on x is W1(x) → T1 must be the last x-writer.
Final write on z is W3(z) → T3 must be the last z-writer.

Q(e) Explain two phase locking protocol with example.

Ans:-

- Two phase Locking (2PL) is a concurrency control protocol used in DBMS to ensure serializability of transactions (i.e., the concurrent schedule is equivalent to some serial schedule).

- Phases of 2PL :-

1. Growing Phase :-
   A transaction may acquire locks (read-lock/ shared or write-lock/ exclusive).
   It can't release any lock in this phase.

2. Shrinking Phase :-
   A transaction may release locks
   It can't acquire any new lock in this phase.

* Types of Locks
. Shared Lock (S-Lock): for read operation, multiple transactions can hold S-lock on

same item.
- Exclusive lock (X-lock) : For write operation; only one transaction can hold X-lock, no other lock is allowed simultaneously

* eg:- Consider two transactions:-
T1 : Read(X), write(X), Read(Y)
T2 : Read(X), write(X)

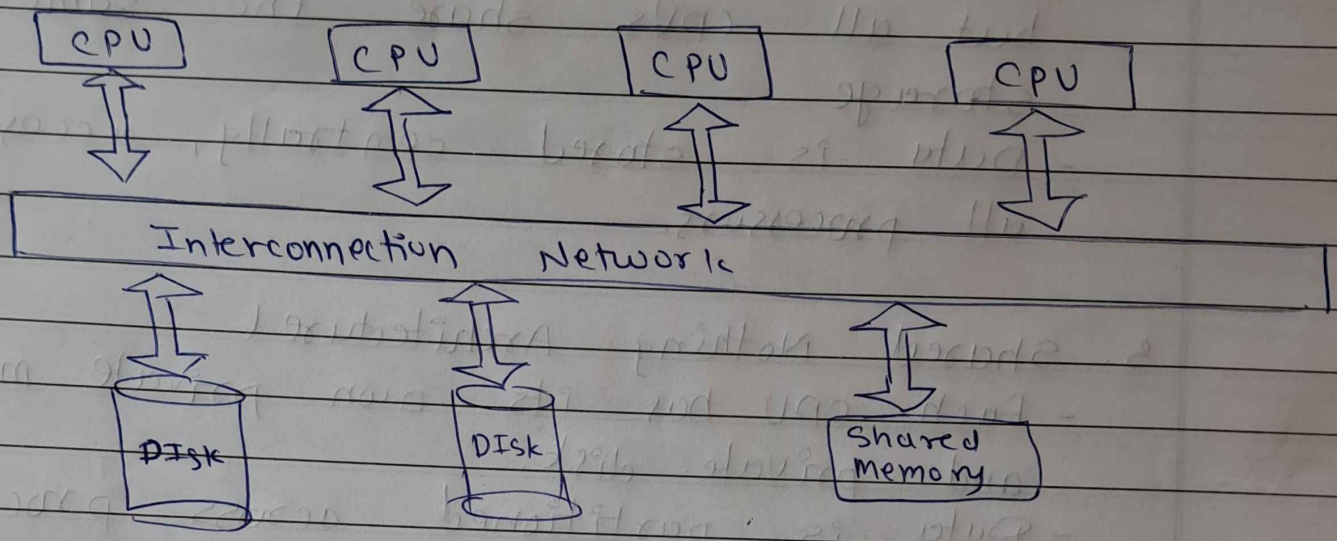
without 2PL :-
If T1 reads X, then T2 writes X, then T1 writes X → schedule may cause inconsistency

with 2PL :-
T1 execution -
- Growing : Acquire X-lock(X), read(X), write(X), aquire S-lock(Y), read(Y).
- Shrinking : release locks on X & Y.

T2 execution -
- Growing : Requests X-lock(X). Must wait until T1 releases it
- Then reads / writes X
- shrinking releases lock.

Assignment No :- 5

Q (1) Explain in detail an architecture of a 11er database



- A parallel Database system is designed to improve performance by executing database operations (query processing, transaction management, etc) in 11er using multiple processors and storage units.

- Parallel Database Architecture's goal is High performance, efficient resource utilization, Reliability & availability.

- Parallel DBMS can be classified based on hardware organization.

1. Shared Memory architecture:
   - Multiple CPUs share a sly common memory & shared disk.
   - All processors access the same database in memory.

2. Shared Disk Architecture :-
- Each CPU has its own private memory but all CPUs share the same disk storage.
- Data is stored centrally, accessible by all processors.

3. Shared Nothing Architecture
- Each CPU has its own private memory and private disk.
- Data is partitioned across processors
- CPUs comm^n over a N/W.

4. Hybrid Architecture :-
- Combination of the above models to balance scalability & availability

Q.(2) Explain multiuser DBMS architecture

Ans :-

A multiuser DBMS allows multiple users to access and manipulate the database simultaneously.

It ensures that
- The integrity of data is preserved.
- Concurrency control prevents conflicts.
- Each user gets correct & consistent results even when many users are active at the same time.

- Key requirement :-
1. Concurrency control
2. Transaction management
3. Security & Authorization
4. Recovery Management
5. Comm^n management.

- Multiuser DBMS Architecture Layers

1. External Level (user views)
   - Each user/ appl^n interacts with DB th^r their own external shema.
   - eg:- A cashier sees customer transaction, while a manager sees summary reports.

2. Conceptual Level (Global Schema)
   - Provides a community view of the entire database.
   - Describes what data is stored & relationships among data.
   - Independent of physical storage details.

3. Internal level (Physical Schema)
   - Defines how data is physically stored
   - Deals with efficiency, space utilization

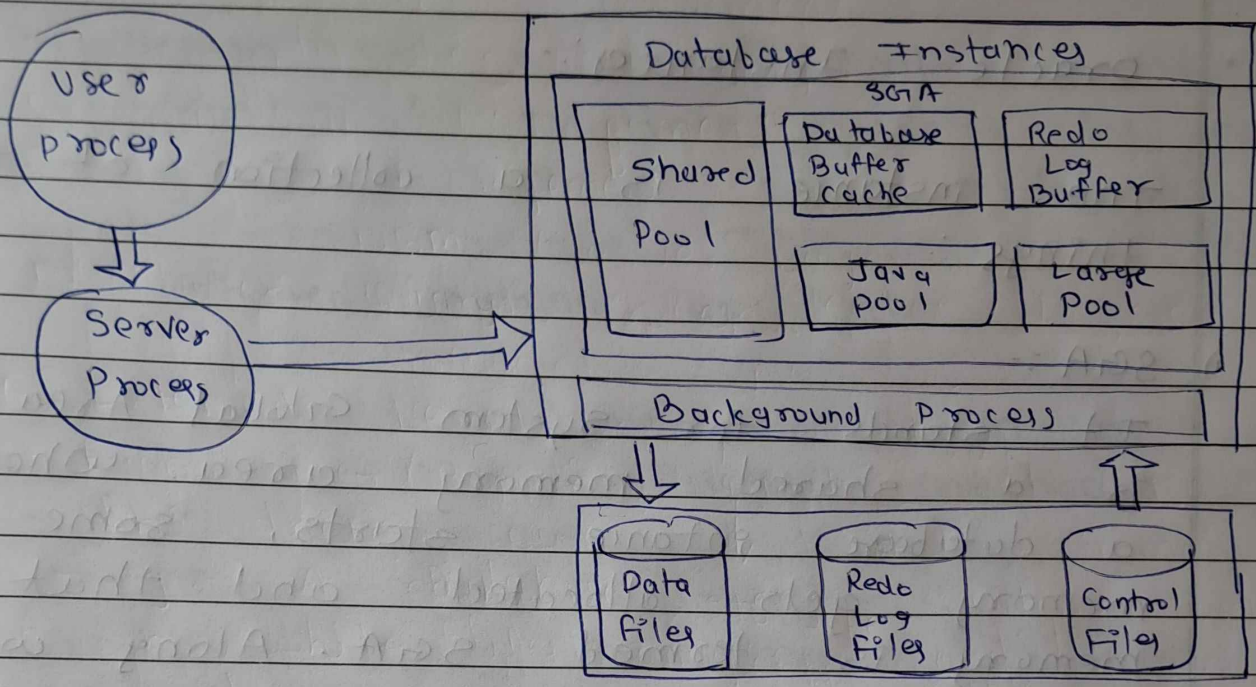
Multiple users
↓ queries
User interfaces
↓
DBMS (Server Layer
- Query Processor
- Transaction Manager
- Concurrency control
- Recovery Manager
↓
Database storage system
(files, Indexes, Buffers)

- There are typically three system architectures used

1) Centralized DBMS Architecture
2) Client - Server DBMS Architecture
3) Distributed DBMS Architecture.

Q 6) With schematic diagram, explain the architecture of oracle Database.

Ans:-



- Oracle Database is relational database management system (RDBMS) that Oracle corporation created & marketed. It is one of the most popular RDMS on the market & is used to store & retrieve data for a wide range of appl^n.
- It is well known for its dependability, scalability, & performance & it is compatible with a wide range of programming language & development frameworks.
- The oracle database architecture consists of:

1) Memory structure (Instances)

2) Database system
3) Process.

- Oracle Instances :-

The instance is a collection of two things :-

1) SGA :-

It stands for System Global Area. It is a shared memory area. Whenever a database intance starts, some memory gets allocated and that memory is termed SGA. Along with memory allocation one or more background processes will. SGA is used to store data as well as control info about one database instance thr its various subcomponents where each comp. is dedicated to a specific purpose.

2) Background Processes :-

Oracle has a collection of processes that are called background processes. These processes are responsible for managing memory, performing I/O operations, and other maintenance activities.

Q (4) Explain the intra query parallelism query evaluation technique.

Ans:-

- Parallelism in a query allows us to user execution of multiple queries by decomposing them into the parts by that work in parallel. This can be achieved by shared nothing architecture.
- We can achieve parallelism in a query by the following methods:
  1. I/O parallelism
  2. Intra-query parallelism
  3. Inter-query parallelism
  4. Intra-operation parallelism
  5. Inter-operation parallelism
- Intra-query parallelism :-
  - Intra-query parallelism refers to the execution of a single query in a user process on different CPUs using a shared nothing parelleling architecture technique.
  - This uses two types of approaches:
    - first approach-
      In this approach, each CPU can execute the duplicate task against some data portion.
    - second approach:
      In this approach the task can be

divided into different sectors with each CPU executing a distinct subtask.

Q(5) What are different parallel database architecture Explain any two with their adv. & disadv?

Ans:-

Parallel Database Architecture :-

1. Shared Memory Architecture
2. Shared Disk Architecture
3. Shared Nothing Architecture.
4. Hybrid Architecture.

1. Shared Memory Architecture :-

In a shared memory architecture, multiple CPUs share a common pool of main memory and can access all disks directly. Comm^n bet^n processors is efficient due to direct memory access

Advantages :-

1) High-speed data access :- Processors can access data in shared memory quickly
2) Efficient comm^n :- Inter-processor comm^n is fast as it occurs th^r shared memory
3) Simplified programming :- Managing data consistency and concurrency can be less complex than in distributed systems.

Disadvantages:
1) Limited scalability:- Performance degraded significantly beyond a certain number of CPUs
2) Single point of failure:- Failure of the shared memory or interconnection N/W can bring down the ~~entire~~ entire system.

2. Shared Disk Architecture:-

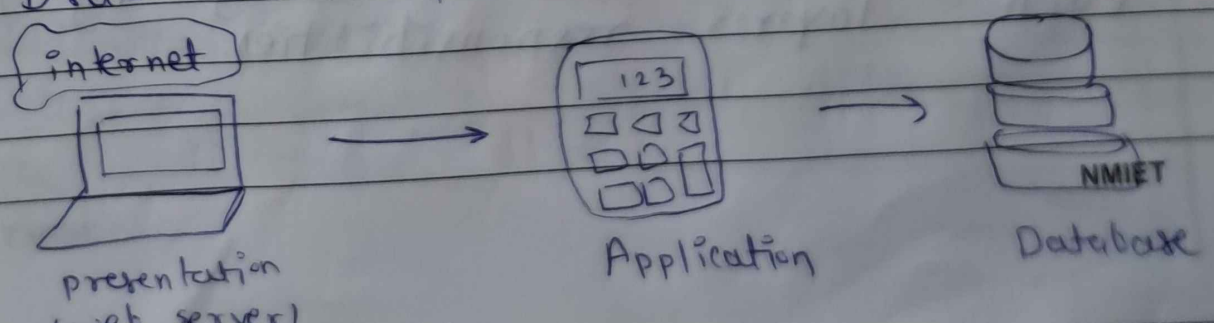
In a shared disk architecture, each CPU has its own private memory, but all CPUs can access all disks directly thr an interconnection N/W. This is common in clustered environment.

Advantages:-
1) Improved load balancing
2) Enhanced fault tolerance
3) Easier data recovery.

Disadvantages:-
1) Cache coherence issues
2) Interference and contention

Q.(6) Draw and explain 3-tier web ~~st~~ architectur

Ans:-



- The 3-tier web architecture divides systems into presentation, appl^n and data layers, increasing scalability, maintainability, this model optimizes resource management & allows for independent scaling & updates making it a popular choice for complex distributed systems.

- Presentation Tier :-
The user interface layer, where interactions occur, It handles data display and user input.

- Application Tier :- The buissness logic layer, which processes user requests, performs computations and makes decisions. It acts as a mediator betn the presentation and data tiers.

- Data Tier :- The storage layer, is responsible for managing and storing data. It handles database operations and data retrieval.

- The separation helps improves scalability, manageability and flexibility by isolating each layer's responsibilities.

Q. (7) Explain data fragmentation types in distributed database design (any two)

Ans:-

Types of Data Fragmentation in distributed
1) Horizontal Fragmentation
2) Vertical Fragmentation
3) Mixed or Hybrid Fragmentation

① Horizontal Fragmentation :-
- Horizontal fragmentation splits a table by rows based on conditions on one or more attributes. Each fragment contains a subset of rows, which are then stored at different sites. This helps in localizing data access.

- In relational algebra, it's represented using the SELECT (σ) operation on table

$\sigma_p(T)$

- σ is relational algebra operator for selection
- p is the condn satisfied by a horizontal fragment.

② Vertical Fragmentation :-
- Vertical fragmentation splits a table by columns (attributes), storing different colums on different sites.

- since each site may not need all colums this improves efficiency. To reconstruct the orgiinal table, each fragement must include a common key so that rows can be joined correctly using a natural JOIN.

$\Pi_{a_1, a_2, \ldots, a_n}(T)$

- Π is relational algebra operator
- a1 ...., an are the attributes of T
- T is the table (relation)