The Shirpur Education Society's

R. C. Patel College of Engineering & Polytechnic, Shirpur

Department of Computer Engineering and Computer Science & Engineering

Course Title - DATABASE MANAGEMENT SYSTEM (DMS)

Course Code - 313302

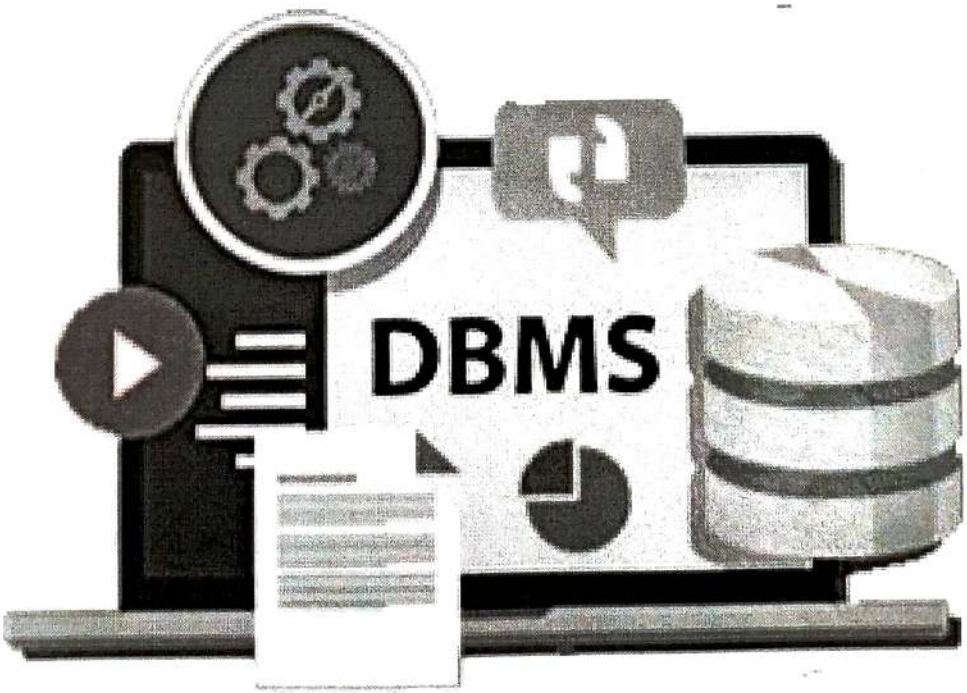
Programme Name - Computer Engineering and Computer Sci. & Engineering

Semester - Third

Unit - V Database Administration

Total Marks: 10

Prepared By: Ms. Namrata R. Bhamare



Unit - V Database Administration

5.1 ## Introduction to database administration

Types of database users

A database user is defined as a person who interacts with data daily for updating, reading, and modifying the given data.

- Database users can access and retrieve data from the database through the database management system (DBMS) applications and interfaces.
- Database users can be classified based on their interaction with the database.

1. **Naive / parametric end users** –

Naive users are users who interact with the system using application programs that have been developed previously.

e.g Railway's ticket booking users.

2. **Application programmers** –

Application programmers responsible for writing application programs that use the database.

- Application Programmers, also known as System Analysts or Software Engineers, writes backend code for application programs.
- They use languages like C, Visual Basic, COBOL etc to design, test, and maintain programs.

3. Sophisticated Users –

sophisticated users like engineers or analysts interact directly with the database using SQL queries.

- They don't write full application code but use the query processor to access and manipulate data.

4. Specialized users –

creates actual database and implements technical controls needed to enforce various policy decisions.

- specialized users are sophisticated users who develope database applications.
- Among these applications are computer aided design systems, knowledge base and expert systems etc.

5. Database Administrator (DBA)

The database administrator is responsible for the overall planning of the company's data resources, for the design of data and for the day-to-day operational aspects of data management.

- A database administrator is a person responsible for the installation, configuration, up graduation maintainance & monitoring databases in an organization.

1. Schema Definition →

The database administrator creates the database schema by executing DDL statements. Schema includes the logical structure of database table (Relation) like data types of attributes, length of attributes, integrity constraits etc.

2. Storage structure and access method definition →

The DBA creates appropriate storage struct-res and access methods by writing a set of definitions which is translated by data

Storage and DDL compiler.

3. Schema and physical organization modification → DBA writes a set of definitions to modify the database schema or description of physical storage organization.

4. Ganting authorization for data access →

The DBA provides different access rights to the users according to their level. ordinary users might have highly restricted access to data, while you go up in the hierarchy to the administrator, you will get more access rights. Integrity constraits specification. Integrity constraits are written by DBA and they are stored in a special file which is accessed by database manager while updating data.

5. Routine Maintenance some of the routine maintanance activities of DBA is given below.

i) Taking backup of database periodically.

ii) Ensuring enough disk space is available all time

iii) Monitoring jobs running on the database.

iv) Ensure the performance is not degraded by some expensive task submitted by some users.

6. Integrity - Constrait specification -

Integrity constraits are written by DBA and they are stored in a special file, which is accessed by database manager, while updating the data.

Create and delete users

In real time database applications having multiple users with same access rights and all needs to have same grant priviledges

e.g There are many users who are type of managers and having same access rights on system in a company.

Instead of giving grant to each and every manager ~~and~~ user can create a generalized set of grant priviledges with name manager and this can be assigned to all employees belongs to category manager

syntax –

Create User <User_name>

Identified By <password>

password Expire Never;

e.g. CREATE USER 'Mahesh@localhost'

IDENTIFIED By '123'

PASSWORD EXPIRE INTERVAL 180 DAY

Droping a User and Role →

- A role can be deleted from created roles in databases.
- We can also remove role from selected user entered in database system as per authorization required.

syntax –

DROP USER <User_name>

e.g DROP USER 'sahil1'@'host';

* Priviledges :→

The set of actions that a user can perform on a database object are called the priviledges.

- Types of priviledges -

1) Ownership priviledges :→

- When a user creates a database object (like table, view, or procedure), they automatically become the owner.
- Owners have all priviledges on their objects (SELECT, INSERT, UPDATE, DELETE, ALTER & DROP).
- These priviledges can't be revoked from the owner.
- e.g. If user A creates a table Employees, user A can perform any operation on it without needing explicit grants.

2) Object priviledges :→

- These are permissions to perform specific actions on a particular object.
- Granted by the owner of the object.
- common object priviledges.

  SELECT - Read data from a table / view.

  INSERT - Add new rows.

  UPDATE - Modify existing rows.

DELETE - Remove rows.

EXECUTE - Run a stored procedure or function.

e.g GRANT SELECT, UPDATE ON Employees TO UserB;

3) System priviledges :→

- These apply to entire database system, not just one object.
- Granted by database administrator.
- Common system priviledges :

CREATE TABLE - Create new tables.

ALTER ANY TABLE - Modify any table in the database.

DROP ANY TABLE - Delete any table.

CREATE USER - Add new database users

GRANT ANY PRIVILEGE - Allow granting priviledge to others.

e.g GRANT CREATE TABLE TO UserC;

5.2 Transaction

A transaction refers to a sequence of one or more operations (such as read, write, update or delete) performed on the database as a

as a single logical unit of work.

- Transactions ensures that either all the operations are successfully executed (committed) or none of them take effect (rolled back).
- Transactions are designed to maintain, the integrity, consistancy and reliability of the database, even in the case of system failures or concurrent access.

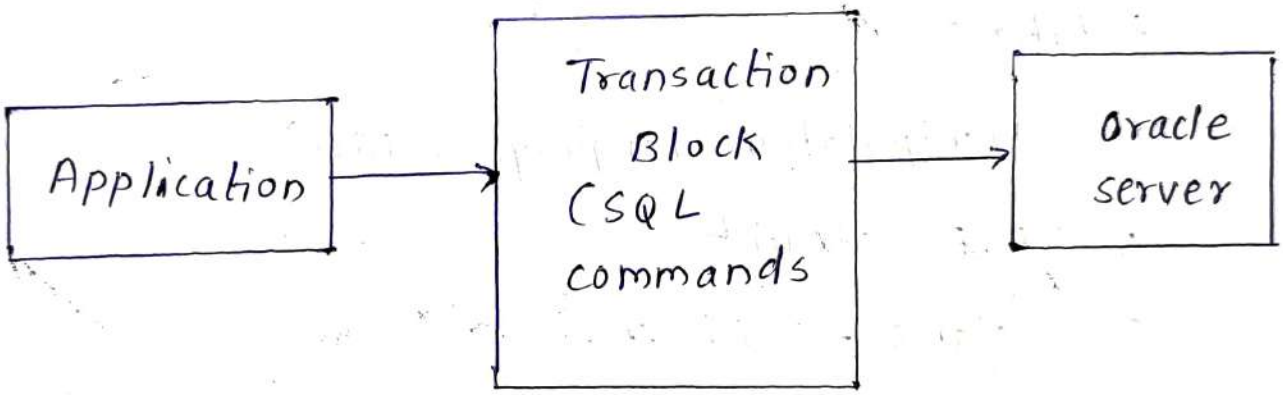


Fig. Executing transactions in DBMS

- Types of operations -

① Read
② Write

① Read operation -

Read operation transfers data item from (e.g Table) the database memory to a local buffer of the transaction that executed

the read operation.

e.g
```
SELECT *
FROM students.
```

② Write operation

Write operation trasfers data from local memory to database.

- write operation transfers one data item from the local buffer of the transaction that executed the write back to the database.

e.g. Data Manipulation language (DML)

```
UPDATE student
SET Name = 'Bhavna'
Where sid = 1186
```

- Transaction Boundaries →

ⓐ commit transcation -

Makes all changes performed by the transaction permanent in the database.

- Once committed, the changes can't be undone.
- Ensure durability in ACID properties.

e.g
```
BEGIN TRANSACTION;
UPDATE Accounts SET Balance = Balance-500 WHERE
AccountID = 101;
```

```
UPDATE Accounts SET Balance = Balance + 500
AccountID = 202;
COMMIT;
```

(b) Rollback transaction

- Undoes all changes made by the transaction, restoring the database to its previous consistent state.
- Used when an error occurs or when the transaction cannot be completed successfully.
- Ensures atomicity in ACID properties.

e.g

```
BEGIN TRANSACTION;
UPDATE Accounts SET Balance = Balance - 500
WHERE AccountID = 101;
UPDATE Accounts SET Balance = Balance + 500
WHERE AccountID = 202;
ROLLBACK.
```

* properties & states of transaction :→

A transaction can be defined as a group of tasks. A single task is the minimum processing unit which can not be divided further

ACID Properties -

Atomicity - This property states that a transaction must be treated as an atomic unit, that is, either all of its operations are executed or none. There must be no state in a database where a transaction is left partially completed. states should be defined either before the execution of the transactions or after the execution / abortion / failure of the transaction.

Consistency - The database must remain in a consistent state after any transaction. No transaction should have any adverse effect on data residing in the database. If the database was in a consistent state before the execution of a transaction, it must remain consistent after the execution of the transaction as well.

Isolation - In a database system where more than one transaction are being executed simultaneously and in parallel, the property of isolation states that all the transactions will be carried out and executed as if it is

the only transaction in the system. No transaction will affect the existence of any other transaction.

Durability - The database should be durable enough to hold all its latest updates even if the system stops or transaction. IF a transaction updates a chunk of data in the database and commits, then the database will hold the modified data. IF a transaction commits but the system fails before the data could be written on the disk, then that data will be updated once the system springs back into action

- States of Transaction :→

A transaction moves through several states during its execution. These states help the system track whether the transaction should be completed, rolled back, or terminated.

1) Active

- The transactions is currently executing operations

e.g Deducting money from one account during a transfer.

2) Partially committed

- All operations have been executed, but the changes are not yet permanent.

e.g Both debit and credit steps are done, waiting for commit

3) Committed

- The transactions have successfully completed, and all the changes are permanently saved.

e.g Transfer finalized and balances updated in the database.

4) Failed

- An error occurs during execution, preventing completion.

e.g Debit succeeds but credit fails due to insufficient Funds

5) Aborted

- The transaction is rolled back to maintain consistency.

e.g Transfer cancelled, balances restored to their original state.

5.3 # Database Backup

- **Types and causes of failure -**

There are various types of failure that may occur in a system, each of these needs to be dealt with a different manner.

1. **Hardware Failure / System crash -**

There is a hardware malfunction that causes the loss of content of volatile storage, and brings the transaction processing to a half.

- The content of non-volatile storage remains intact, and is not corrupted or changed.

2. **Software Failure -**

The database software or the operating system may be corrupted or failed to work correctly, that may causes the loss of the content of volatile storage, and brings about database failure.

3. **Media failure -**

A disk block loses its content as a result of either a head crash or failure during a data transfer operation.

Copies of the data on other disks such as tapes, CD's are used to recover from the failure

4. Network Failure –

The problem with network interface card can cause network failure.
There may be problem with network connection.

5. Transaction Failure –

There are two types of error that may cause a transaction to fail.

1) Logical Error
2) system Error

6. Application Software Error –

The problem with software accessing the data from database. This may cause database failure as data cannot be updated using such application to it.

7. physical Disasters –

The problem caused due to Flood, Fire and the earthquake etc.

8. Human Error –

Mistakes like accidental deletion or incorrect updates.

• Database Backup Introduction :⇒

It refers to creating a copy of the database's data and structure to safeguard against unexpected data loss.

- This backup ensures that the database can be restored to its original state in case of failures, corruption, or accidental deletions.

* Database backup techniques –

① Full backup - physical backup

- Full database backup is maintained at one recovery site as backup copy of that site.
- A Full and incremental database backup option targets to make it more feasible to store several copies of the source data.
- At First step a full backup (of all files) is made. After that, any number of updates can be applied to that an incremental backup.

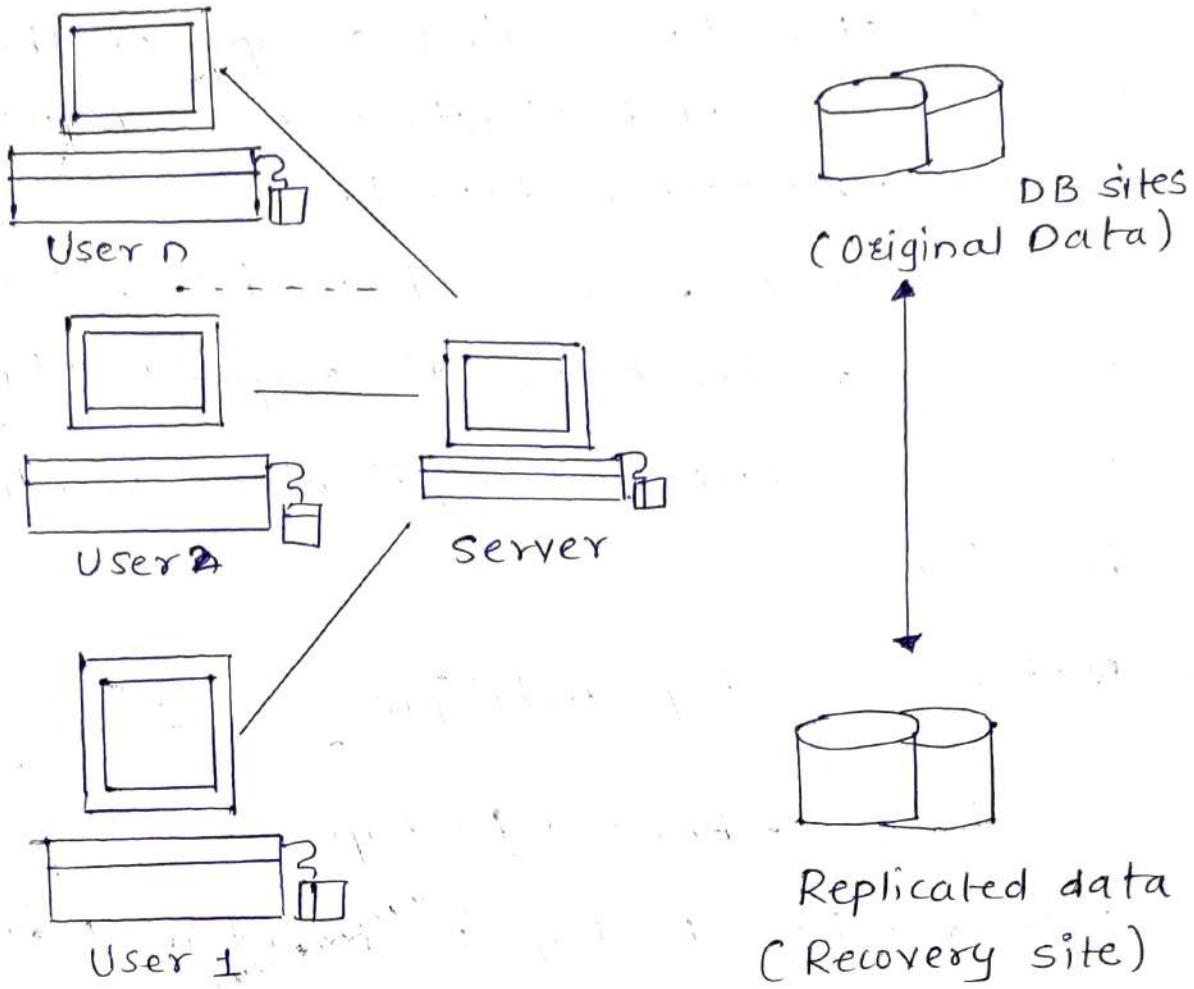


Fig. physical backup.

② Partial Backup or Logical Backup -

- It contains logical data which is retrived from the database.
- It contains a view, procedure or function, and table.
- Its like saving log files for all transactions done on databases as a backup.
- Logical backups are not secure as physical backups in preventing data loss.

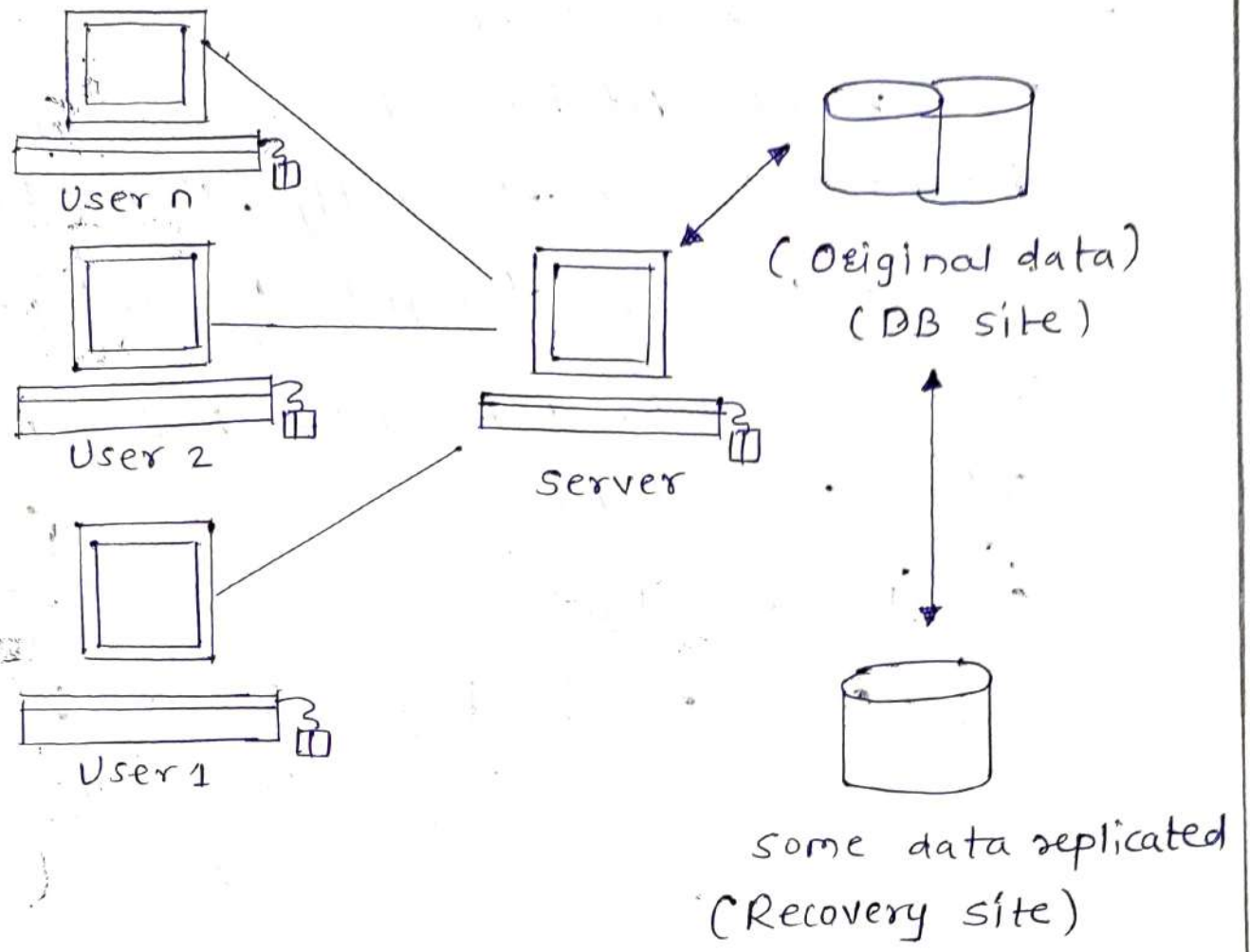


Fig. Logical Backup

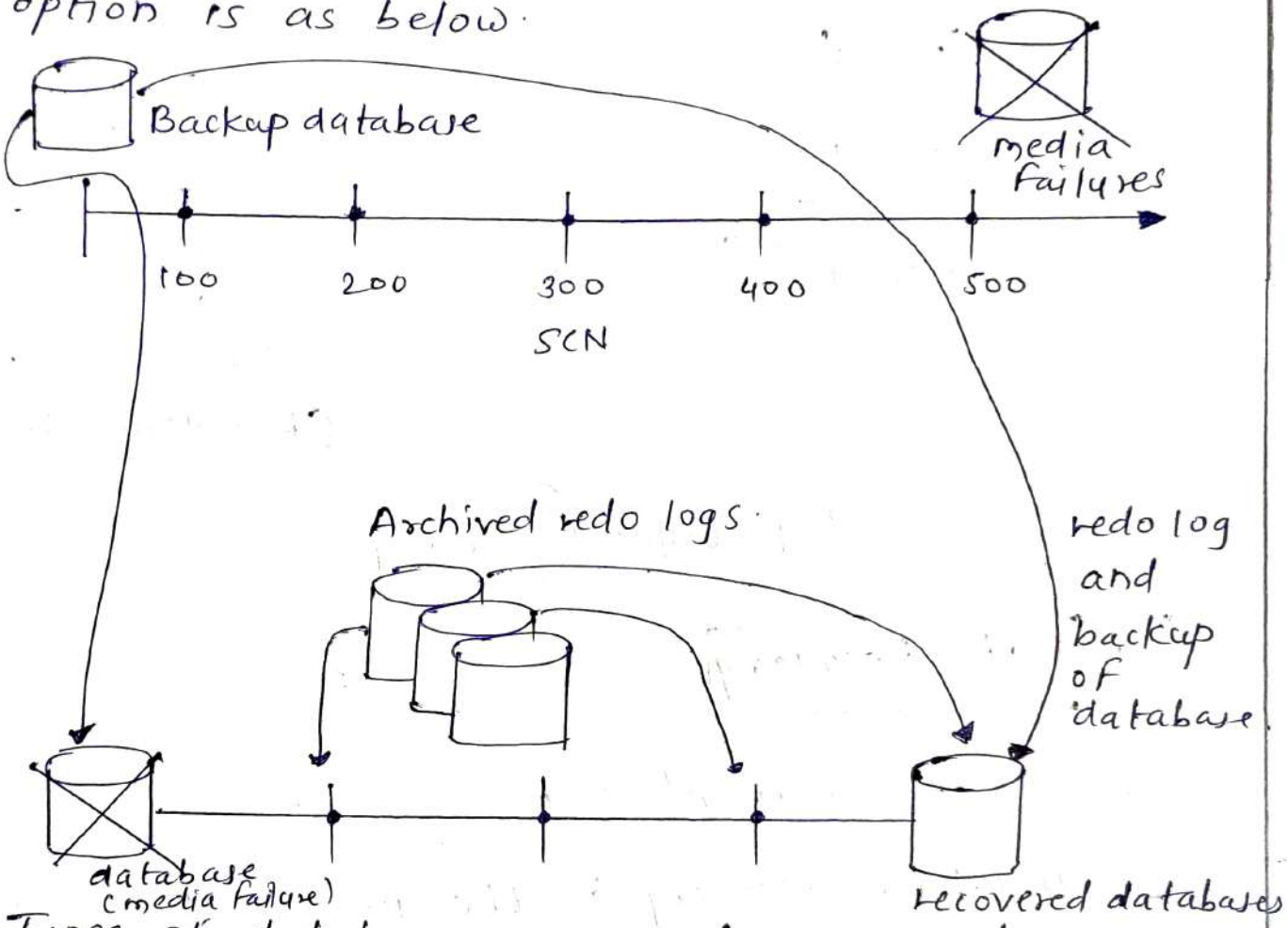
5.4 Data Recovery

- Recovery concepts :→

- Database recovery is the process of restoring the database to original (correct) state as it was before database failure occurs.
- The process of solving any type of database failures, quickly and without data loss and keep database available is called database recovery.

- The main element of database recovery is the most recent database backup. If you maintain database backup efficiently, then database recovery is very straight forward process.

e.g To recover data from system having full backup option is as below.



- Types of database recovery / recovery phases.

(a) Forward database recovery / Roll Forward (REDO phase)

(b) Backward database recovery / Roll Backward (UNDO phase)

- Database recovery techniques –

(a) Log based recovery

(b) shadow paging recovery

(c) checkpoints.

(a) Log based recovery –

- Uses a log file to record all changes before applying them.
- Supports Undo (rollback uncommitted transaction) and Redo (reapply committed transactions)
- Two approaches –
  - Immediate Update (Undo / Redo)
  - Deffered Update (Redo only)

(b) Shadow Paging –

- Maintains two page tables – current & shadow.
- If the transactions commits, the current table replaces the shadow.
- If it fails, revert to the shadow table. (no undo/redo needed)
- simple but can cause fragmentation.

(c) Checkpointing –

- creates a bookmark in the log file.
- speeds up recovery by avoiding scanning the entire log.
- often used with log-based recovery.

• Recovery techniques –

1) Roll forward –

- Reapplies changes from the transaction log to the database.
- Used to recover committed transactions that were not written to disk before a crash.
- Ensures durability by moving the database forward to the latest consistent state.
- Relies on redo log entries.

2) Rollback –

- Undoes changes made by uncommitted transactions
- Used when a transaction fails or is aborted.
- Ensures atomicity by removing partial or incomplete changes.
- Relies on undo log entries.

5.5 # Overview of Advanced Database Concepts

- **Data warehouse** –

- Central repository for structured, historical data
- Optimized for querying, reporting, and analytics
- Uses ETL (Extract, Transform, Load) processes.
- characteristics of data warehouse –

(1) subject Oriented.
(2) Integrated.
(3) Non-volatile.
(4) Time varient.

eg Amazon Redshift, SnowFlake.

- **Data Lake** –

- store raw, unstructured, semi-structured, and structured data.
- Flexible schema-on-read approach.
- Suitable for big data and machine learning workloads.

eg Hadoop, Azure Data Lake.

- **Data Mining –**
  - Process of discovering patterns, correlations, and insights from large datasets.
  - Techniques include clustering, classification, regression, and association rules.
  - Used in fraud detection; marketing and recommendations system.

- **Big Data –**
  - Refers to massive datasets with high volume, velocity, and variety.
  - Requires distributed storage and parallel processing.
  - Tools - Hadoop, Spark, Kafka.
  - Applications – Social media analytics, IOT, real-time monitoring.

- **MongoDB –**
  - NoSQL database using document-oriented storage.
  - stores data in JSON-like BSON Format.
  - Flexible schema, scalable horizontally.
  - Best for applications needing agility and rapid development.

- **DynamoDB** -

- AWS - managed NoSQL database service.
- key-value and document-based storage.
- Fully serverless, auto-scaling, high availability.
- Ideal For web apps, gaming, IOT, and serverless architectures.

29/06/26

27/6/26

Unit-V Database Administration Questions

- 2 Marks Questions

1) Define Database Administration. (Winter 2019)(I)
2) List two responsibilities of a DBA (Winter 2020)(I)
3) What is database security? (Winter 2021)(I)
4) State two functions of database maintenance. (Winter 2022)(K)
5) Differentiate between database users and DBA. (Winter 2018)(I) (Winter 2020)

- 4 Marks Questions

1) Explain database backup and recovery. (Winter 2022)(K)
2) Describe database performance monitoring. (Winter 2019)(I)
3) Explain database security measures. (Winter 2020)(I)
4) Discuss database tuning with examples. (Winter 2020)(I)
5) Explain the role of DBA in user management (Winter 2018)(I)

- 6 Marks Questions

1) Explain roles and responsibilities of DBA in detail. (Winter 2019)(I)
2) Discuss database backup strategies. (Winter 2021)(K)
3) Explain recovery techniques. (Winter 2018)(I)
4) Describe database security threats and solutions. (Winter 2022)(K)
5) Explain transaction management and its role in database administration. (Winter 2019)(I)

6) Explain database audit trails: (Winter 2021)(I)!

29/11/24

27/6/24