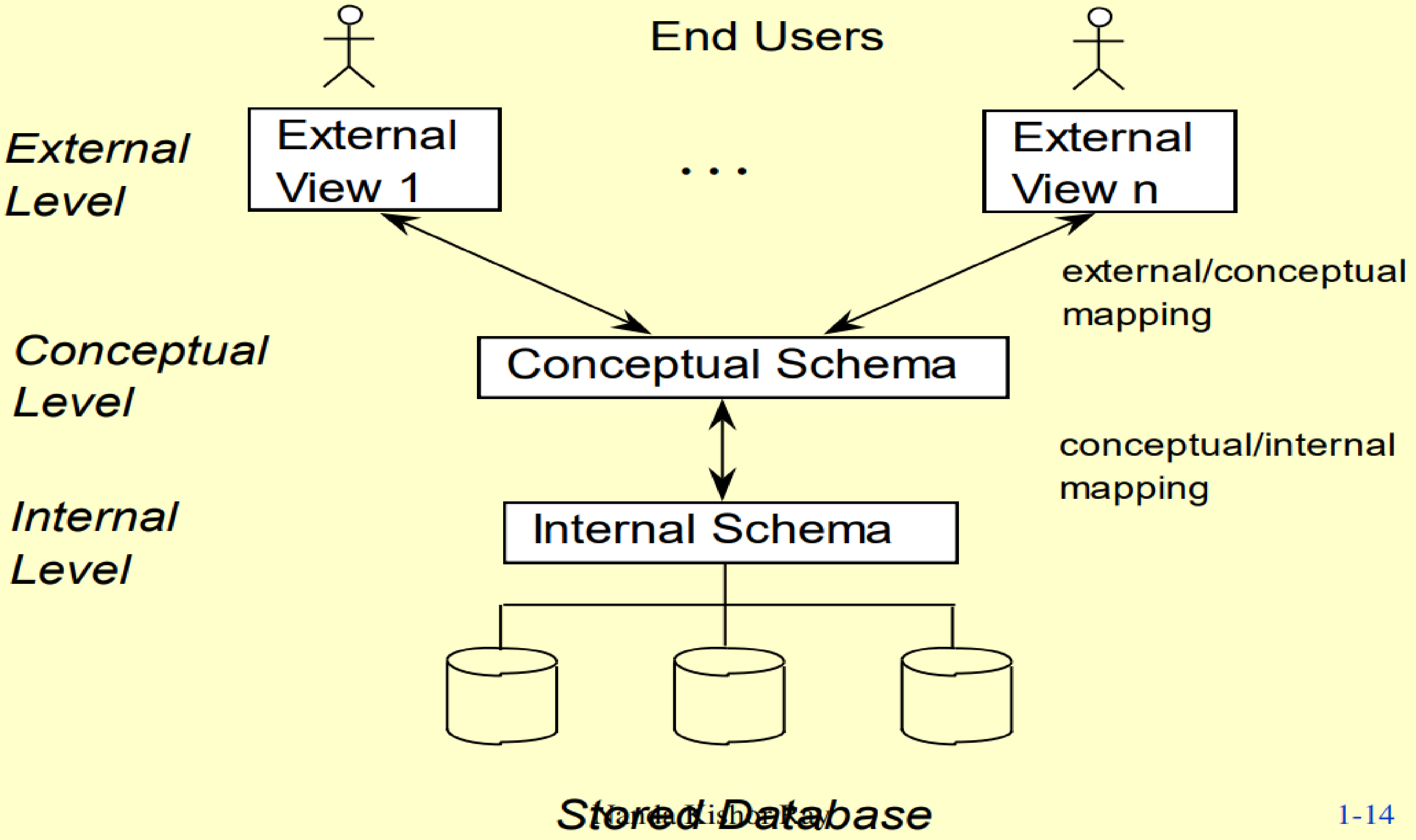
Unit 2

Database System –Concept and Architecture

DBMS Architecture

- The DBMS architecture proposed by ANSI/SPARC (*American National Standards Institute, Standards Planning And Requirements Committee)* (**ANSI/SPARC architecture**) is defined at three levels. This architecture is also called **three-schema architecture**.
- This architecture provides three levels of abstraction to simplify users' interaction with the system.
- It provides users with an abstract view of data. The system hides certain details of how data are stored and maintained.
- The goal of this architecture is to separate the user applications from physical database.
- It divides the system into three levels of abstraction: the *internal* or *physical level*, the *logical* or *conceptual level*, and the *external* or *view level.*

DBMS Architecture



DBMS Architecture

- **Physical Level or Internal Level:**
  - It is the lowest level of abstraction and describes ***how*** the data in the database are actually stored.
  - This level describes complex low-level data structures in detail and is concerned with the way the data is physically stored.
  - Data only exists at physical level.
- **Logical Level or Conceptual Level:**
  - This is the next higher level of abstraction and describes ***what*** data are stored in the database, and what relationships exist among those data.
  - It describes the structure of whole database and hides details of physical storage structure.
  - It concentrates on describing entities, data types, relationships, attributes and constraints.
  - All of the views must be derivable from this conceptual schema.

DBMS Architecture

- **View Level or External Level:**
  - It is the highest level of abstraction and is concerned with the way the data is seen by individual users.
  - This level simplifies the users’ interaction with the system.
  - It includes a number of user views and hence is guided by the end user requirement.
  - It describes only those part of the database in which the users are interested and hides rest of all from those users. Each user group refers to its own external schema.
- The DBMS must transform a request specified on an **external schema** into a request against the **conceptual schema**, and then into a request on the **internal schema** for processing over the database. The process of transforming requests and results between levels is called **mapping**.

Instances and Schemas

- **Instances:**
  - Databases change over time as information is inserted and deleted. The collection of information stored in the database at a particular moment is called an **instance** of the database.
  - It is also known as **database state**.
- **Schema:**
  - The overall design of the database which is not expected to change frequently is called the database **schema**.
  - There are three schemas, partitioned according to the levels of abstraction. The **physical schema** describes the database design at physical level. The logical **schema describes** the database design at the logical level. The schema at the view level is sometimes called **subschema** and describes the view of the database. A database may have several subschema.

Data Independence

- The three schema architecture further explains the concept of data independence, the capacity to change the schema at one level without having to change the schema at the next higher level.
  - **Logical Data Independence**
  - **Physical Data Independence**
- **Logical Data Independence:**
  - The capacity to change the conceptual schema without having to change the external schemas and their associated application programs.
- **Physical Data Independence**:
  - The capacity to change the internal schema without having to change the conceptual schema.

Data Independence

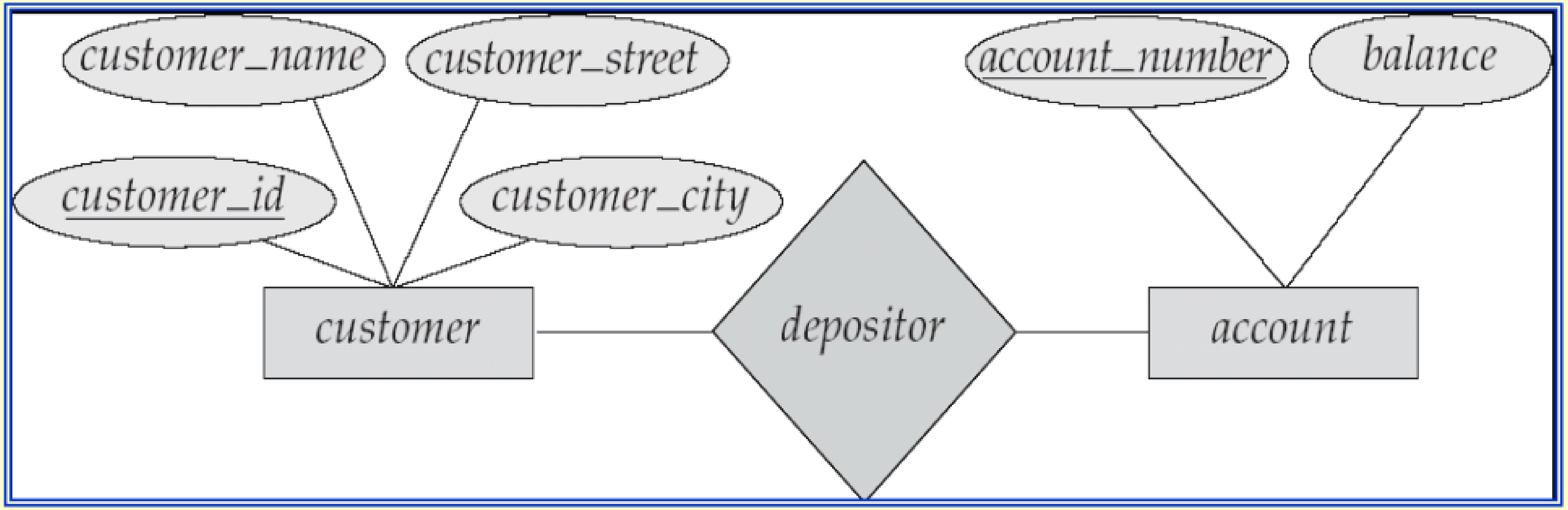
- For example, the internal schema may be changed when certain file structures are reorganized or new indexes are created to improve database performance
- When a schema at a lower level is changed, only the **mappings** between this schema and higher-level schemas need to be changed in a DBMS that fully supports data independence. The higher-level schemas themselves are unchanged.
- Hence, the application programs need not be changed since they refer to the external schemas.

Data Models

- The basic structure or design of the database is the **data model**. A data model is a collection of conceptual tools for describing data, data relationships, data semantics, and consistency constraints. Some data models are given below:
- **Entity-Relationship Model:**
    - **Entity-relationship (E-R) model** is a *high level* data model based on a perception of a real world that consists of collection of basic objects, called **entities**, and of **relationships** among these entities.
    - An **entity** is a thing or object in the real world that is distinguishable from other objects.
    - Entities are described in a database by a set of **attributes**.
    - A **relationship** is an association among several entities.
    - The set of all entities of the same type is called an **entity set** and the set of all relationships of the same type is called a **relationship** ***set***.

Data Models

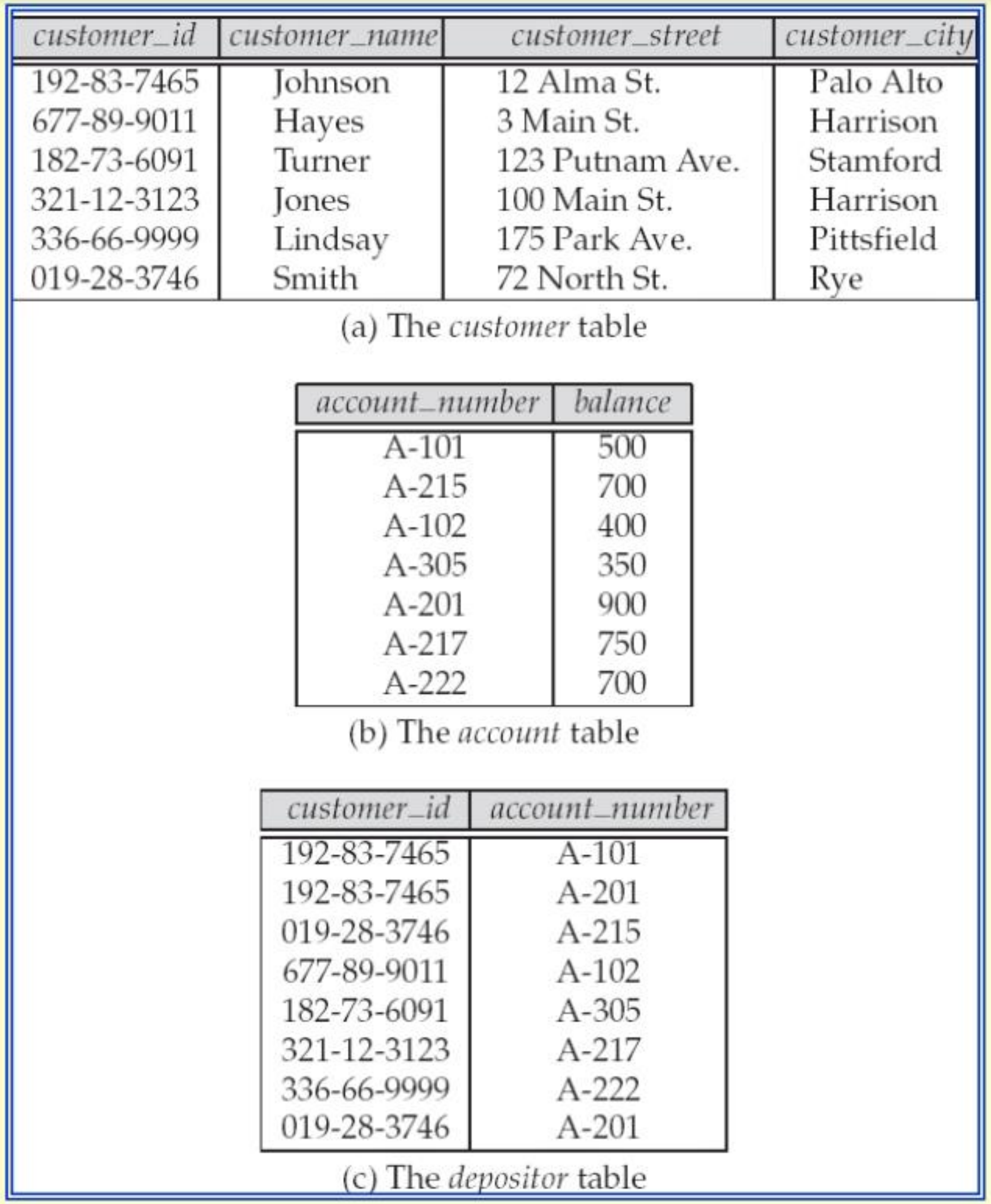
- Overall logical structure of a database can be expressed graphically by E-R diagram. The basic components of this diagram are:
  - **Rectangles** (represent entity sets)
  - **Ellipses** (represent attributes)
  - **Diamonds** (represent relationship sets among entity sets)
  - **Lines** (link attributes to entity sets and entity sets to relationship sets)
- The figure below shows an example of E-R diagram.



Data Models

  - In addition, the E-R model also represents certain constraints to which the contents of the database must conform. The constraints are **mapping cardinalities** and **participation constraints**. (discussed later)

- **Relational Model:**
  - It is the current pervasive model. The relational model is a *lower level* model that uses a collection of tables to represent both data and relationships among those data. Each table has multiple columns, and each column has a unique name. Each table corresponds to an entity set or relationship set, and each row represents an instance of that entity set or relationship set.
  - Relationships link rows from two tables by embedding row identifiers (keys) from one table as attribute values in the other table.
  - Structured query language (SQL) is used to manipulate data stored in tables.



customer_id	*customer_name*	*customer_street*	*customer_city*
192-83-7465	Johnson	12 Alma St.	Palo Alto
677-89-9011	Hayes	3 Main St.	Harrison
182-73-6091	Turner	123 Putnam Ave.	Stamford
321-12-3123	Jones	100 Main St.	Harrison
336-66-9999	Lindsay	175 Park Ave.	Pittsfield
019-28-3746	Smith	72 North St.	Rye

(a) The *customer* table

account_number	*balance*
A-101	500
A-215	700
A-102	400
A-305	350
A-201	900
A-217	750
A-222	700

(b) The *account* table

customer_id	*account_number*
192-83-7465	A-101
192-83-7465	A-201
019-28-3746	A-215
677-89-9011	A-102
182-73-6091	A-305
321-12-3123	A-217
336-66-9999	A-222
019-28-3746	A-201

(c) The *depositor* table

Fig: A sample relational database

Data Models

  - The relational data model is the most widely used data model, and a vast majority of current database systems are based on the relational model. The relational model is at a lower level of abstraction than the E-R model. Database designs are often carried out in the E-R model, and then translated to the relational model.

- **Object-oriented Model:**
  - This model represents an entity set as a class. A class represents both object attributes as well as the behavior of the entity.
  - Instances of class are objects. Within an object, the class attributes takes specific values. However the behavior patterns of the class is shared by all the objects belonging to the class.
  - Attribute values can be primitive data types usually associated with databases and programming languages or other objects. The object-oriented model maintains relationships through 'logical-containment'.

Data Models

- **Object Relational Model:**
  - This model combines the features of the object-oriented data model and relational data model.
- **Semistructured Model:**
  - This model permits the specification of data where individual data items of the same type may have different set of attributes. The ***extensible markup language*** (XML) is widely used to represent semistructured data.
- **Hierarchical Model:**
  - This model assumes that a tree structure is the most frequently occurring relationship.
- **Network Model:**
  - The network model replaces the hierarchical tree with a graph thus allowing more general connections among the nodes. This model was evolved to specially handle non-hierarchical relationships.

* Types of database systems

1) Centralized Database System

→ In centralized system, all programs run on the main host computer, including the DBMS, the application that access the database and the communication facilities that send or receive data from the users terminals.

→ The users access the database through either locally connected or dial-up(remote) terminals

→ The terminals are generally dumb, having little or no processing power of their own & consists of only a screen, keyboard & hardware to communicate with the host

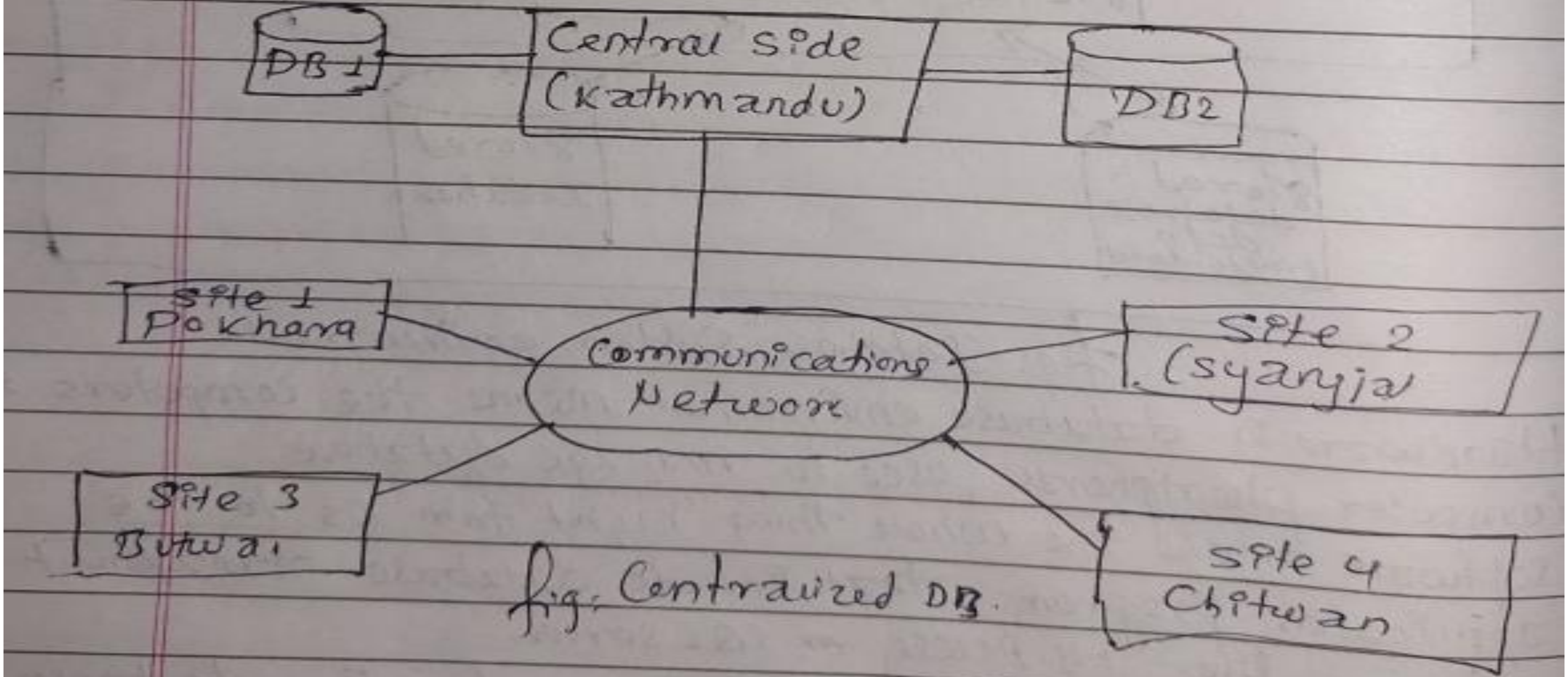


fig. Centralized DB.

Advantages:

- Since all data is stored at a single location only thus it is easier to access and coordinate data.
- The centralized database has very minimal data redundancy since all data is stored in a single place.
- It is cheaper in comparison to all other databases available.

Disadvantages:

- The data traffic in the case of a centralized database is more.
- If any kind of system failure occurs in the centralized system then the entire data will be destroyed.

2) Client-Server Database System.

→ In a generalized concept, client PC is the computer from where the user requests for data and information and the server provides the requested information.

→ The database application on the client pc referred to as the 'front end system' that handles all the screen and user input/output processing.

→ The 'back end system' on the database server handles data processing and disk access.

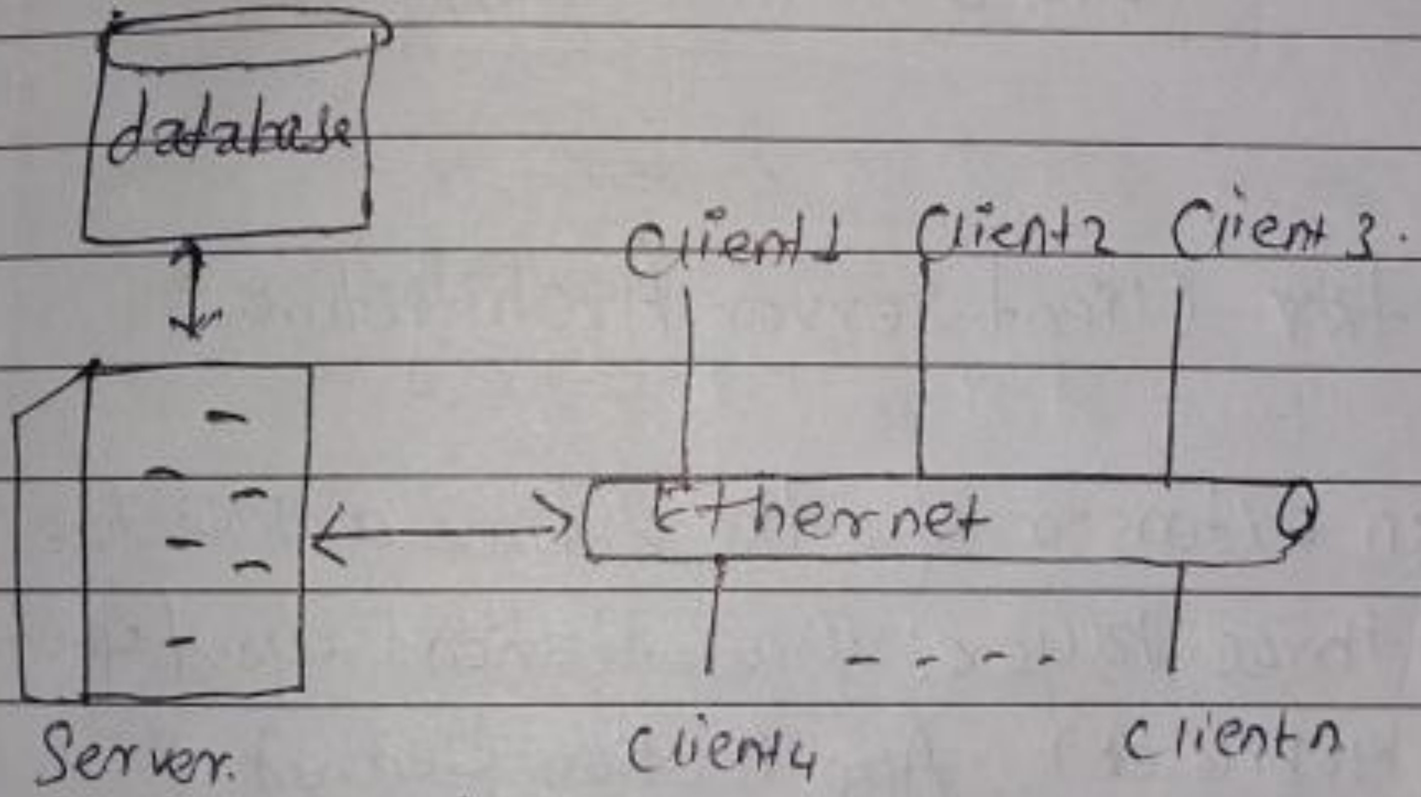


fig: Client server database system

→ Client server approach is implemented by two approachs two-tier architecture & three-tier architecture.

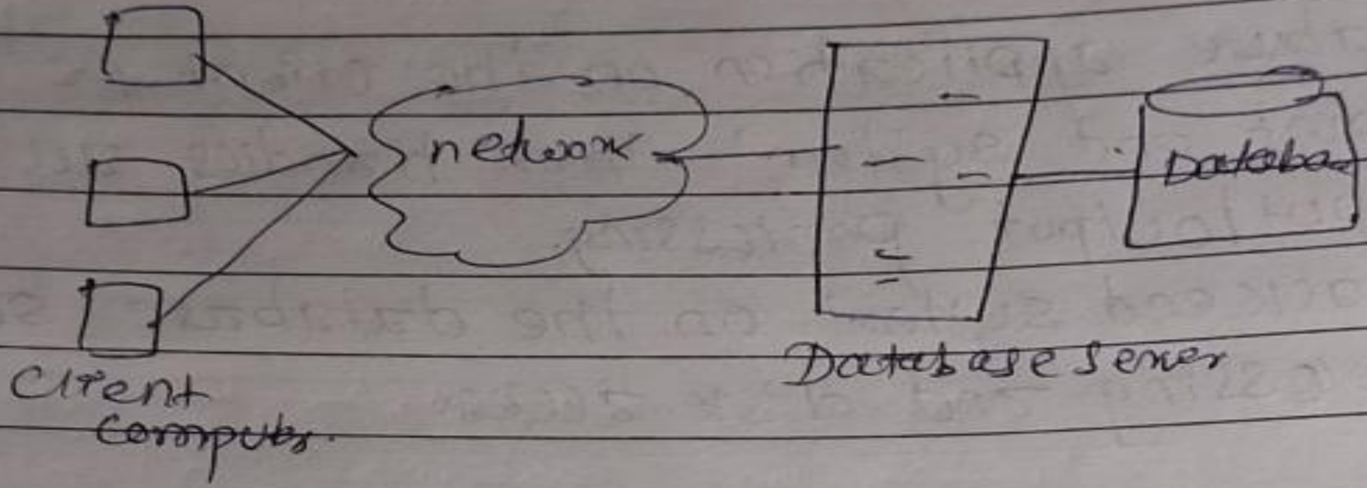
- Two-tier architecture:

  → The user interface & application programs are placed on the client side & database system on server side.

  → The application programs that reside at client side ~~revoke~~ invoke the DBMS at server side.

→ Open Database connectivity (OBDC) and JAVA ~~application~~ Database connectivity (JDBC) are used for interaction between client & server.

→ Because of tight coupling 2-tier application will run faster.

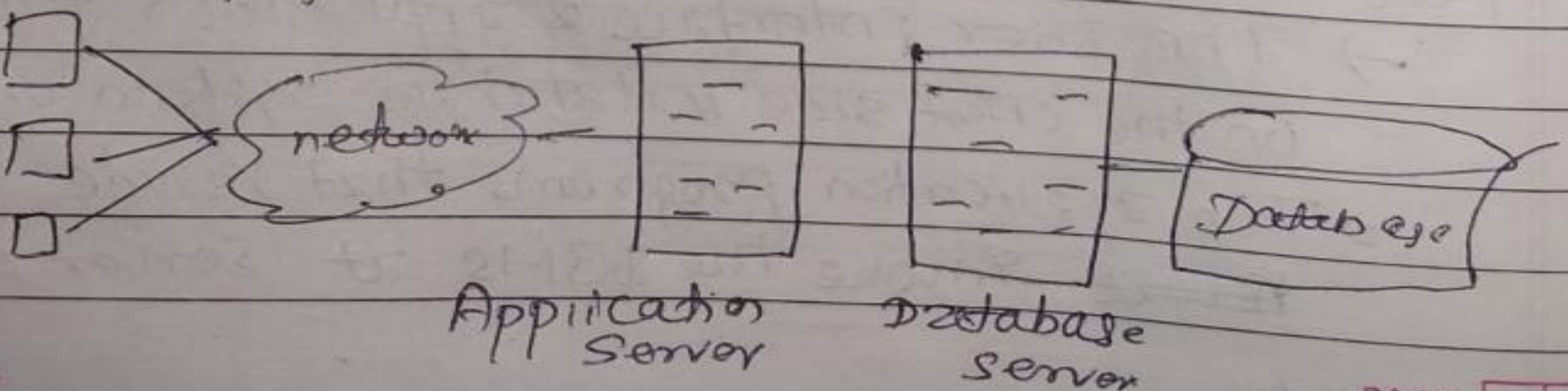


→ Advantages: Easy to understand, modify & maintain

→ Disadvantage: Time consuming when large users and little cost effective.

Three tier Client-Server Architecture :

- → It is an extension of the 2-tier architecture.
- → It has three layers: Presentation layer (your pc, Tablet, Mobile, etc), Application (server) layer & Database server.
- → The application layer is responsible for communicating the user's request to DBMS system & send response from DBMS system to user.
- → It is more popular DBMS architecture.



→ Advantages: Easy to maintain & modify, improved Security and good performance.

→ Disadvantage: It is little more complex & little more effort is required in terms of hitting the db.

* Database Languages

-> Database languages are the classification of programming languages that programmers or developers use in order to define and access the databases.

-) It can be used to read, store and update the data in the database.

-) They are used basically to communicate with the database.

* Types of database language

i) DDL (Data Definition Language)

→ It is used to define database structure & patterns

→ It is used to create schema, tables and indexes in the database

→ DDL is used to store the information of metadata like the number of tables & schemas, their indexes, columns in the table, etc.

Some commands of DDL are:

- CREATE : To create database instance
- RENAME : Use to rename database instance
- DROP : Used to delete database instance
- ALTER : To alter the database instance.

Ex: CREATE DATABASE Test; – name

DROP TABLE Student; – table name

(ii) DML (Data Manipulation Language)

→ It is used for accessing and manipulating the data in the database

→ It handles the user requests (insert, update, delete)

→ Express database queries and updates

Some commands of DML are :-

- SELECT : To read records from the table
- INSERT : To insert records in the table.
- UPDATE : To update data in the table
- DELETE : To delete all the records from the table

Ex: INSERT INTO table_name VALUES(data1, data2, ...)

ex: consider a table student with following fields.

S_id | name | age.

INSERT INTO student VALUES (101, 'Adam', 15);

iii) DCL (Data Control Language)

→ It is used for granting and revoking access in the db.

→ To perform any operations in the database like Creating tables, we need privileges: Such privileges are controlled by DCL.

Some Commands:

- GRANT : To grant access to the user
- REVOKE : To revoke(take back) the access from the user.

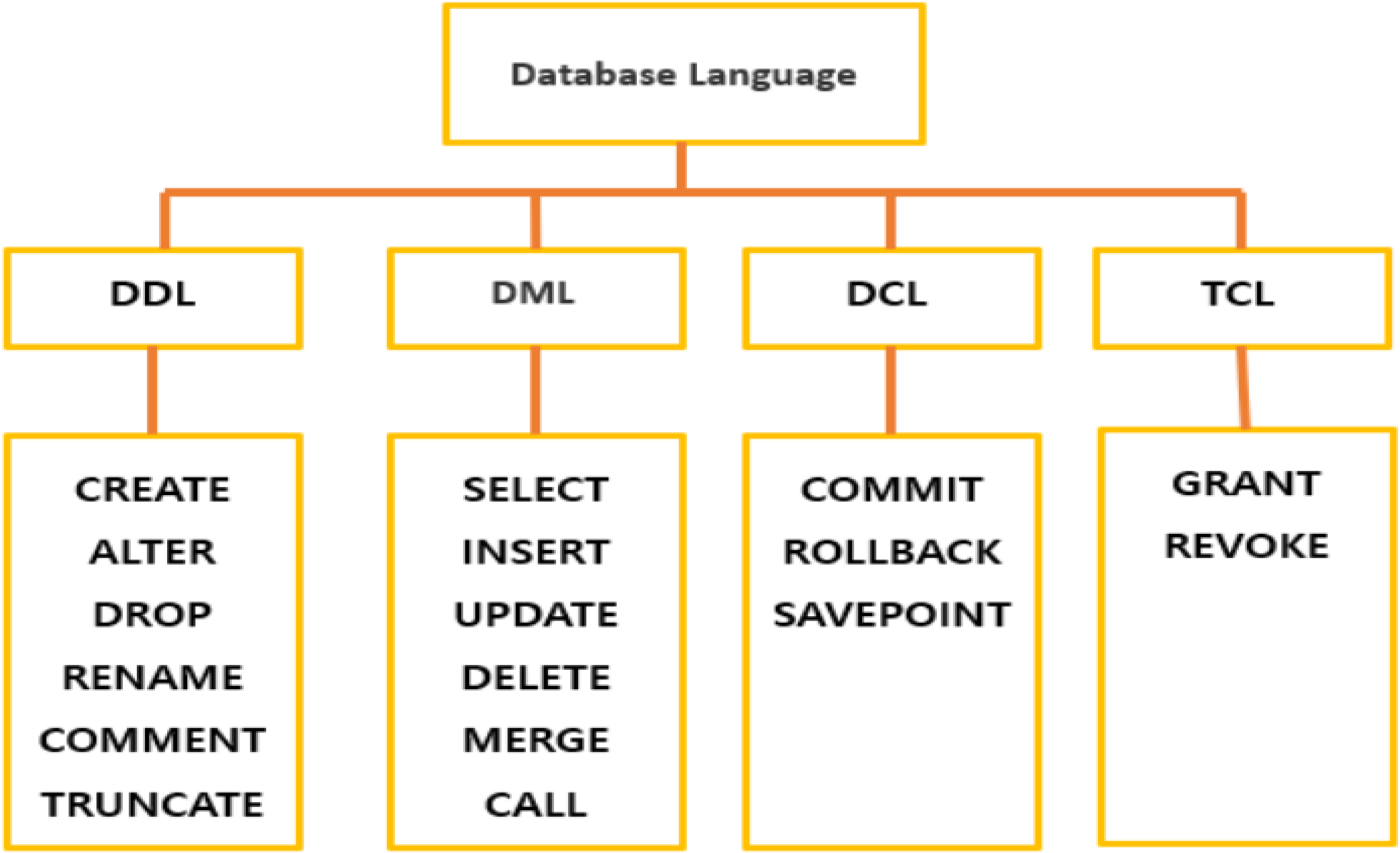
```
Ex: GRANT CREATE TABLE TO username;
    REVOKE CREATE TABLE FROM username;
```

iv) TCL (Transaction Control Language)

→ It provides commands that are used to manage transactions in the database

Some important commands:

- COMMIT: To save transactions in the database
- ROLLBACK: To restore the database to the original since the last commit
- SAVEPOINT: Used to temporarily save the transactions so you can rollback to previous point



Database Language

DDL

CREATE
ALTER
DROP
RENAME
COMMENT
TRUNCATE

DML

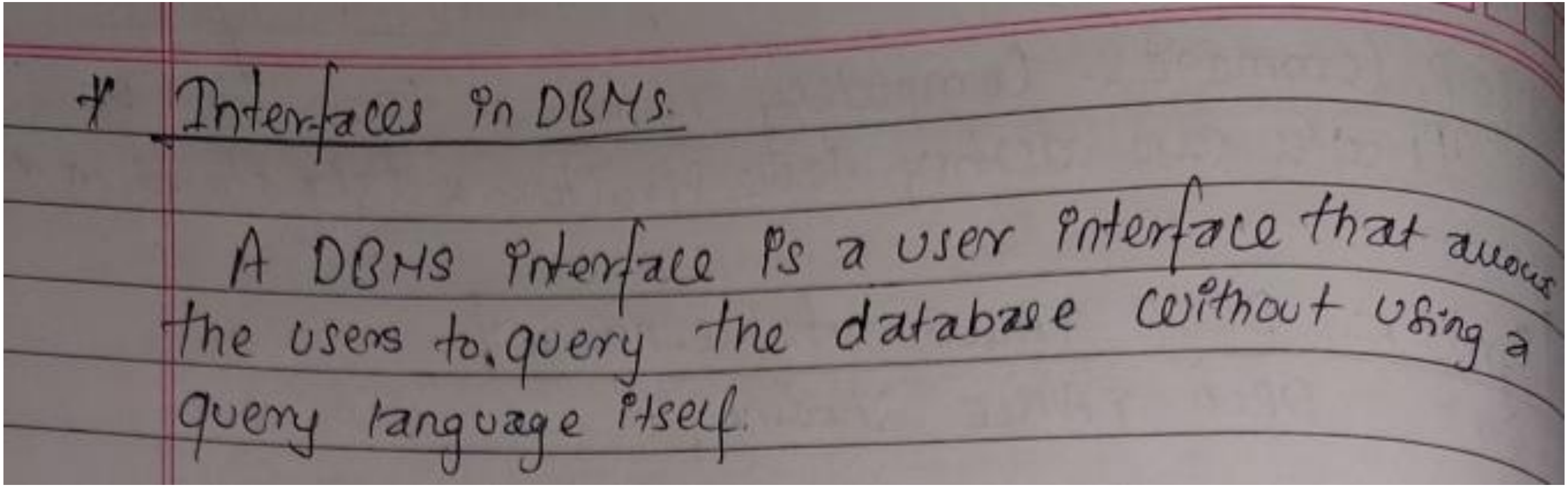
SELECT
INSERT
UPDATE
DELETE
MERGE
CALL

DCL

COMMIT
ROLLBACK
SAVEPOINT

TCL

GRANT
REVOKE



* Interfaces in DBMS.

A DBMS interface is a user interface that allows the users to query the database without using a query language itself.

UI provided by DBMS are

1. Menu-Based Interface.

-) These interfaces present the users with a list of options (menus)

-) Here, users don't need to memorize specific commands or syntax for query language

-> Pull-down menus is most popular

-) other Example is browsing on online store.

2. Forms- Based Interface.

- → Displays a form to each user where the users can fill the necessary data.
- → Designed & programmers for naive users
- → Ex: when taking online attendance, or checking result online by filling online forms.

online forms

3. Graphical user Interfaces

→ It is a type of interface through which users interact with schema in diagrammatic form.

users can specify the query by manipulating the diagram

→ In most of the cases, it uses both menu & form based interface (like searching youtube videos and choosing accordingly).

4. Natural Language Interface.

→ These are the simple interfaces where the system and user communicate via natural language ie. human language.

→ It has its own schema and dictionary.

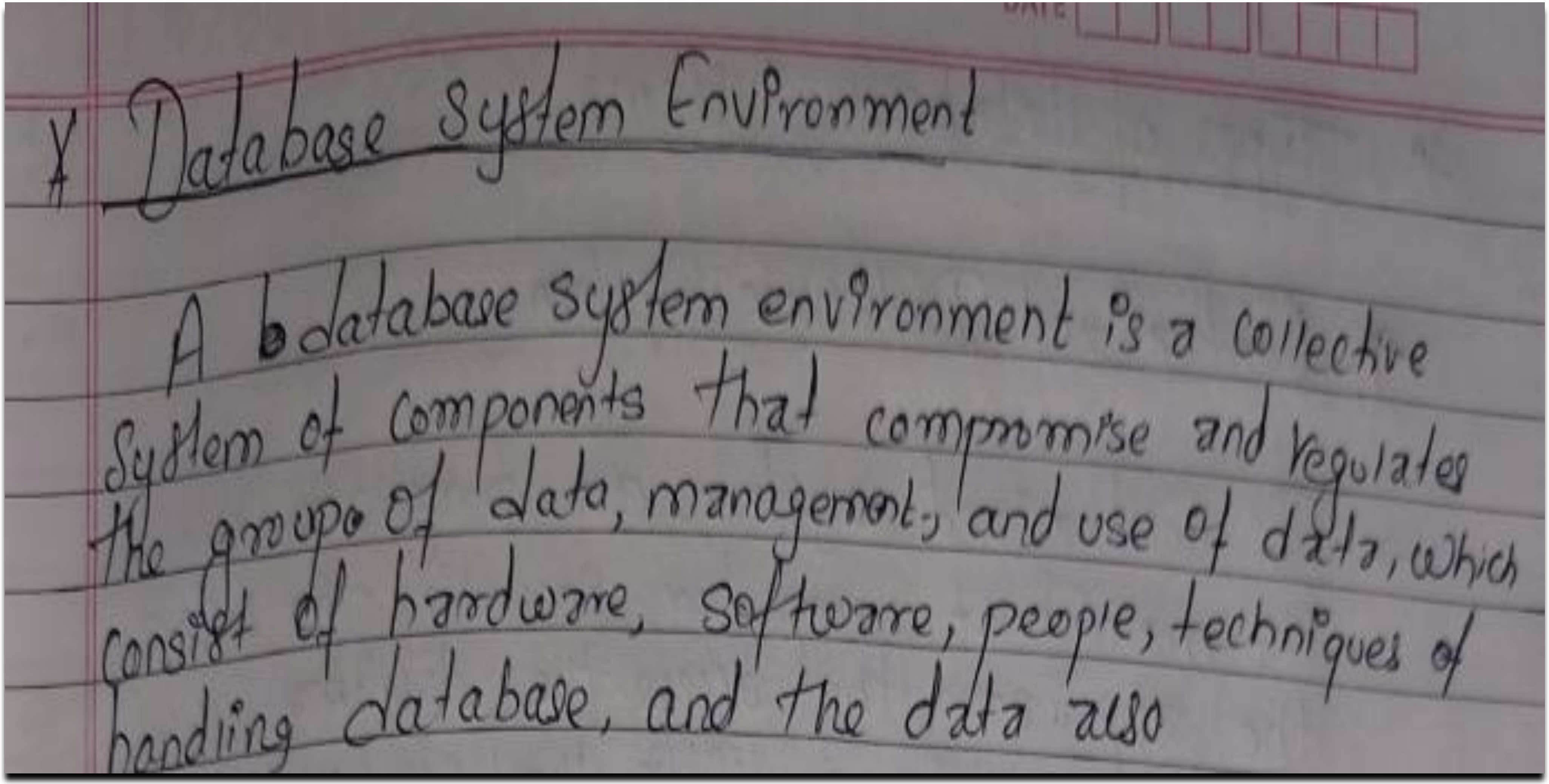
- Ex: Google search.

5. Speech Input and output

→ It is the most common type of interface used in today's world where user queries the interface in the form of speech & gets response the same way

→ Here also the natural language is used by the interpreter & finds related keyword so that it can retrive data from database.

→ Ex: Google Assistant, Alexa, siri, etc.



* Database System Environment

A ~~b~~database system environment is a collective system of components that compromise and regulates the groupo of data, management, and use of data, which consist of hardware, software, people, techniques of handling database, and the data also

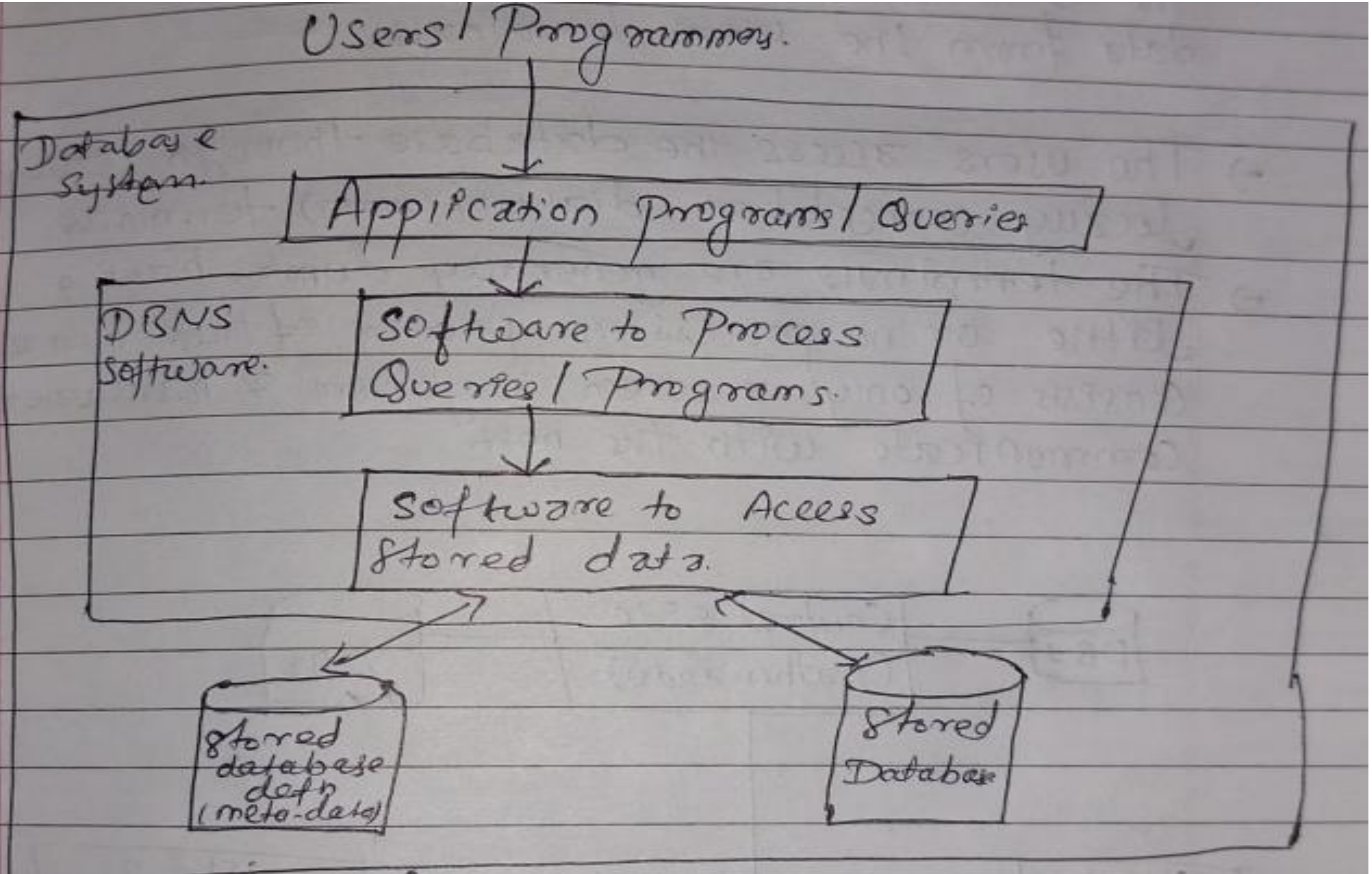

Users/ Programmers.
Database System.
Application Programs/ Queries
DBNS Software.
Software to Process Queries/ Programs.
Software to Access Stored data.
Stored database defn (meta-data)
Stored Database

fig: Database System environment.

[illegible]

-) Hardware in database environment means the computers & Computer pherripherals used to manage database

-> Software means the whole thing right from OS to the application programs that include database management Software like. MS. Access or SQL server.

-> People in database environment include inuude those people Who administrate & use the System

-> Techniques are the rules, Concepts & instructions to manage db.

* Classification of Database Management system.

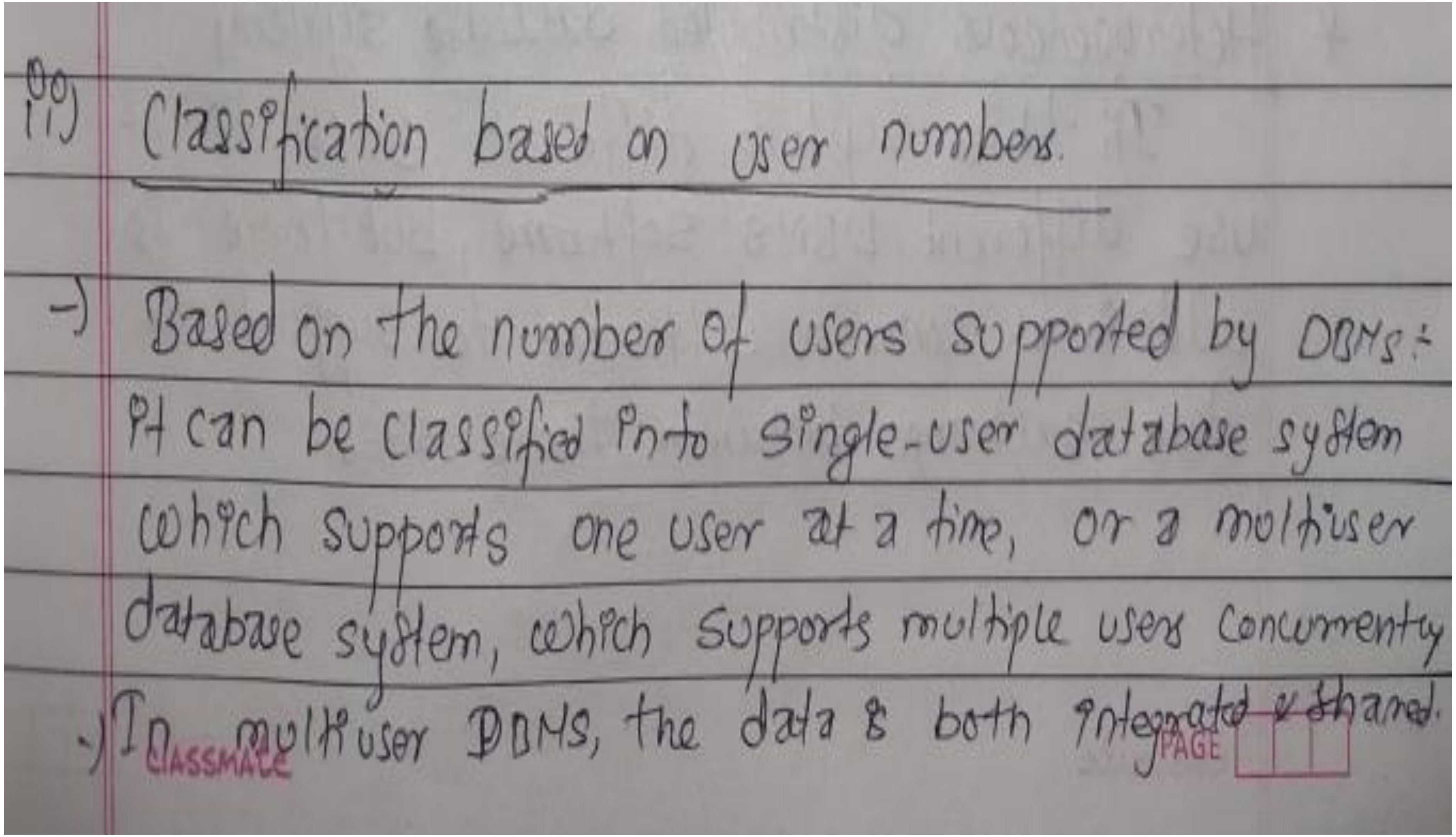
i) Classification based on data model.

→ The most popular data model used today is the relational data model.

→ Well known DBMS like Oracle, MS SQL Server, support this model.

→ Other traditional models like hierarchical, network data models are still used in industry mainly on mainframe platforms. However, they are not commonly used due to their complexity.

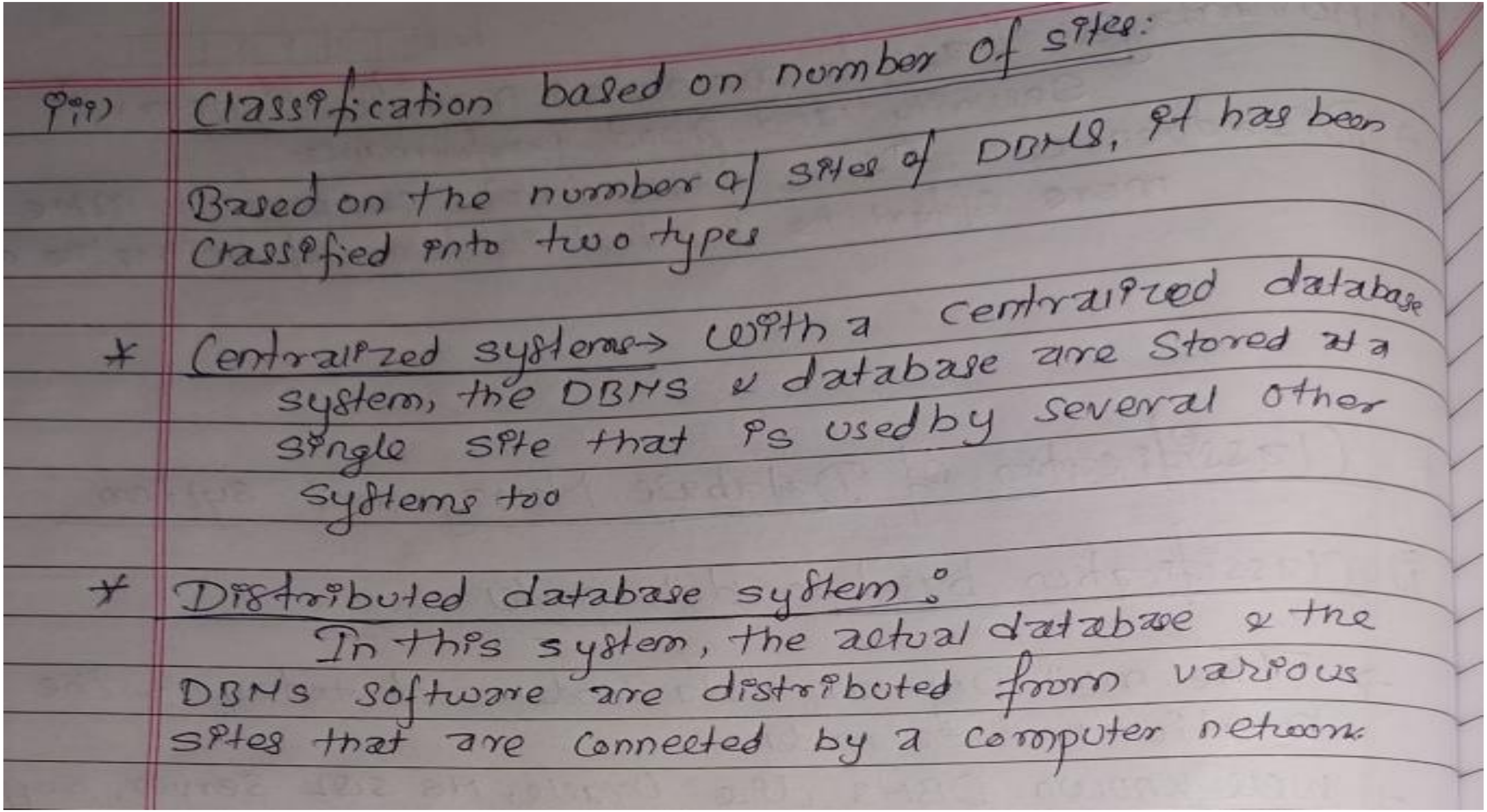
→ In recent years, newer object-oriented were introduced where information is represented in the form of objects as used in object-oriented programming.



iii) Classification based on user numbers.

-) Based on the number of users supported by DBMS:- it can be classified into single-user database system which supports one user at a time, or a multiuser database system, which supports multiple users concurrenty

-) In multiuser DBMS, the data is both integrated & shared.



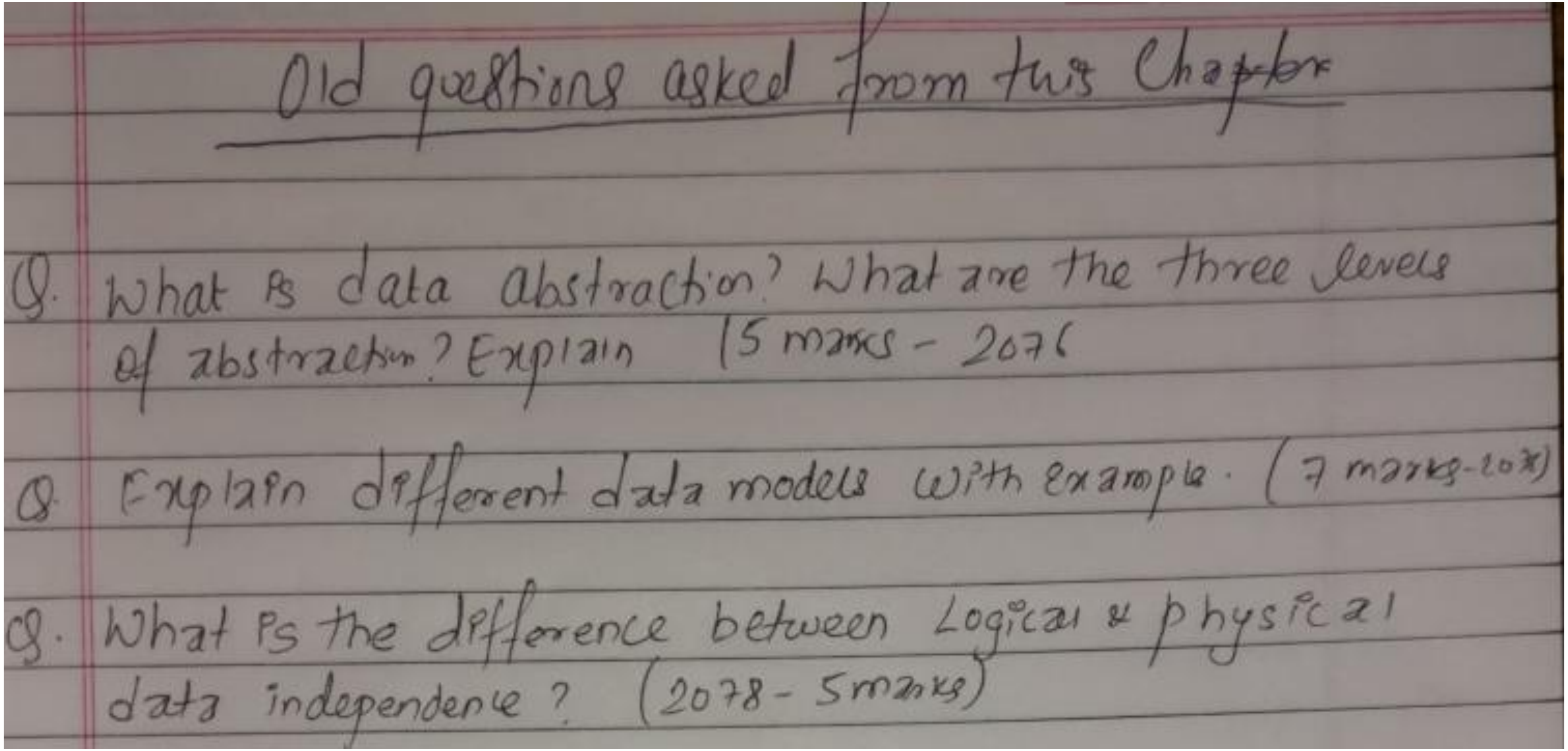
iii) Classification based on number of sites:

Based on the number of sites of DBMS, it has been Classified into two types

* Centralized systems→ With a centralized database system, the DBMS & database are Stored at a single site that is used by several other systems too

* Distributed database system:

In this system, the actual database & the DBMS software are distributed from various sites that are connected by a computer network.



Old questions asked from this Chapter

Q. What is data abstraction? What are the three levels of abstraction? Explain (5 marks - 2076

Q. Explain different data models with example. (7 marks-2078)

Q. What is the difference between Logical & physical data independence? (2078 - 5marks)

Q Explain: a) Three-schema architechre

b) Data abstraction.

c) Two-tier & three-tier client/server arch.

Q. Difference between distributed & client-server DBMS.
(2073- 5 marks)

Q. What is data definition language? How is it different from data manipulation language? (2075 - 5 marks)

Q. What is database system architecture? Describe three levees & benefits of this architecture. (5 marks)

Questions

- Define data abstraction, data model, schemas, instances and database state?
- What do you mean by Schema and Instance in DBMS? Explain both with examples.
- Explain the DBMS architecture with a diagram. What is data independence?
- What is difference between logical data independence and physical data independence?
- Explain the Data independence with example.
- Explain different data models with example.
- What is data abstraction? What are three levels of data abstraction? Explain.