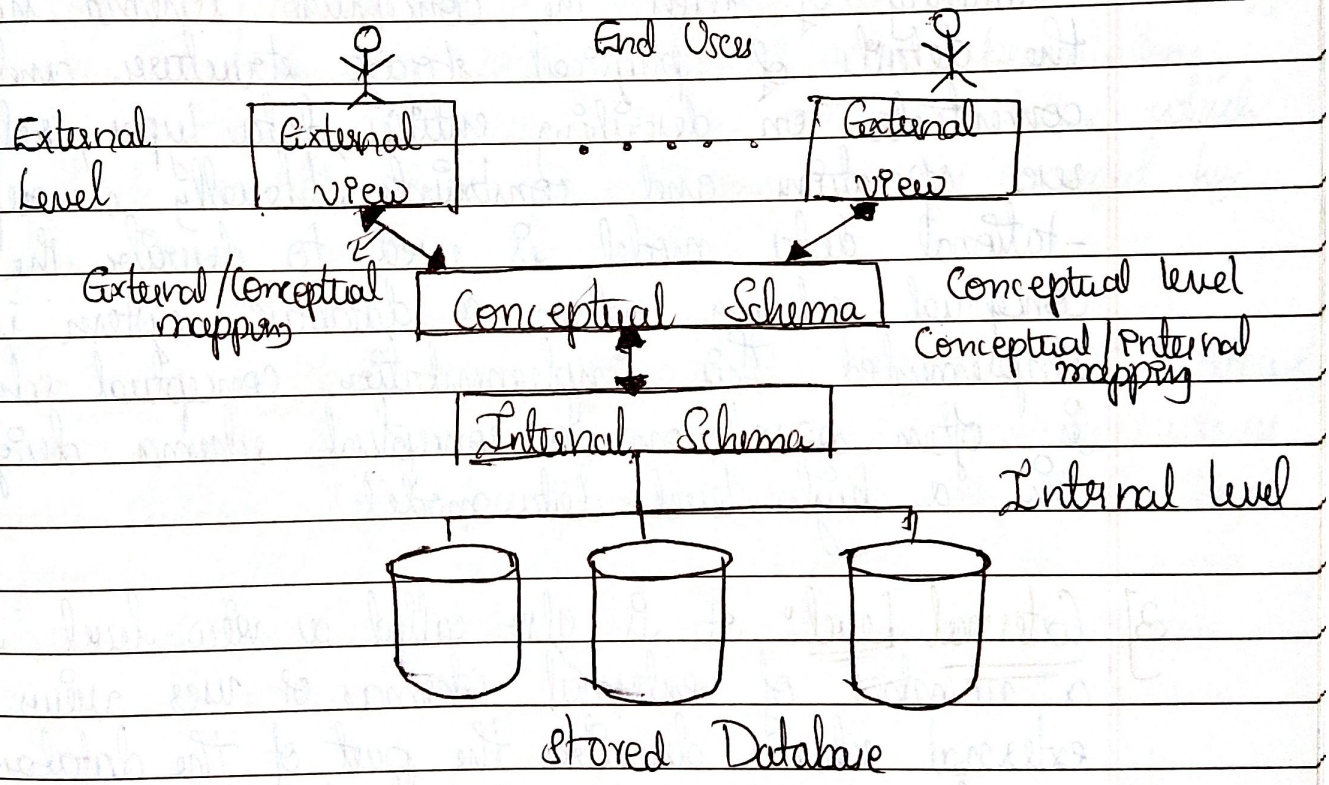
D D M M Y Y Y Y

MODULE - 1

1] What is a Database? Explain the three schema architecture with neat diagram.

→ A database is an organized collection of data that can be easily accessed, managed and updated. It is used to store information in a structured format, allowing efficient retrieval, insertion, deletion and modification of data.
A database management system (DBMS) is software that interacts with end users, applications and the database to capture and analyze data.



~: The Three-schema architecture :~

The goal of the three-schema architecture, illustrated is to separate the user applications from the

physical database. In this architecture the schemas can be defined at the following three levels:

1] Internal Level 2] Conceptual Level 3] External level.

1] Internal Level: It has an internal schema, which describes the physical storage structure of the database. The internal schema uses a physical model & describes the complete details of data storage & access paths to the database.

2] Conceptual Level: It has a conceptual schema which describes the structure of the whole database for a community of users. The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations and constraints. Usually a representational data model is used to describe the conceptual schema when a database system is implemented. This implementation conceptual schema is often based on a conceptual schema design in a high level data model.

3] External Level: It is also called as view level includes a number of external schemas or user views. Each external schema describes the part of the database that a particular user group is interested in & hides the rest of the database from that user group. As in the previous level, each external schema is typically implemented using a representational data model, possibly based on an external schema design in a high level conceptual data model.

D D M M Y Y Y Y

Q] What are the advantages of using DBMS approach? Explain.

→ 1] Controlling Redundancy: In traditional file systems, the same data might be stored in multiple places, leading to wastage of space and chances of inconsistency. DBMS reduce data redundancy. DBMS reduces data redundancy by storing data in a central database and managing access through multiple applications ensuring data is stored only once and updated in one place.

2] Restricting unauthorized access: A DBMS provides security feature to control who can access or modify data. Through user authentication & access permissions, only authorized users can perform certain actions which protects sensitive data from being seen or changed by the wrong people.

3] Providing persistent storage for program objects: DBMS allows storage of complex program-related objects such as user defined data types & record in a way that they persist beyond the execution of a program. This ensures data is available even after the program ends or the system is restarted.

4] Providing storage structure and search techniques for efficient query processing: To make data access faster, DBMS uses various structures like indexes & search algorithms. These help in quickly locating & retrieving the required data, especially when dealing with large amount of information

5] Providing backup and recovery: DBMS include tools automatically backing up data & recovering it in case of hardware failure, power loss or system crash. This ensures that data is not permanently lost and systems can be restored to previous working state.

6] Providing multiple user interfaces: A DBMS supports different types of interfaces like graphical forms, reports, & command line tools. This allows different users - such as developers, admins & end-users to interact with the system in a way that suits their need & skills.

7] Representing Complex Relationships Among Data: Real world data is often interconnected. A DBMS can represent complex relationships (like one to many or many to many) using relational models, foreign keys & join operations which allows for more accurate & meaningful data representation.

8] Enforcing Integrity Constraints: DBMS helps maintain the accuracy & consistency of data through rules called integrity constraints. These ensures that data entered into the system follows certain rules such as unique ID, a valid age ranges, helping prevent errors.

9] Permitting Inferencing & Action Using Rules & Triggers: DBMS support the use of rules and triggers to automatically perform actions when certain conditions are met. For example, a trigger can send a notification when stock levels drop below a threshold making systems more responsive & automated.

3] Explain the following terms
i] Data Dictionary ii] Weak Entity

i] Data Dictionary: In DBMS a data dictionary is a centralized collection of information about the data in the database. It stores metadata, which is essentially data about the database itself. This includes details such as the name of tables, columns, data types, size, constraints like primary keys & foreign keys, relationships b/w tables, default values & access permissions. The data dictionary plays a crucial role in helping database administrators & developers understand the structure, constraints & rules of the database making it easier to manage & maintain the system efficiently.

ii] Weak Entity: It is a type of entity that cannot be uniquely identified by its own attributes alone. It depend on a related strong entity also known as owner entity to ensure its uniqueness. A weak entity typically has a partial key, which is an attributes that can only identify the entity when combined with the primary key of the strong entity through a total participation constraint meaning every instance of the weak entity is related to one instance of the strong entity. A common example is a "Dependent" entity in an employee database, where each dependent must be associated with an "Employee" entity & cannot exist in database without the relationship. Ex: a DRIVER_LICENSE entity cannot exist unless it is related to a PERSON entity, even though it has its own key (License_number) and hence is not a weak entity

4) Explain the categories of data models.

→ A data model is a collection of concepts that can be used to describe the structure of a database which provides the necessary means to achieve this abstraction.

There are three categories of data models:
1) High Level (Conceptual)
2) Representational (Implemen
3) Low Level (Physical)

1) High Level (Conceptual) Data Models:
These models are closest to how humans perceive data. They use concepts such as entities, attributes & relationships to describe data & its structure. These models are independent of how the data is stored physically. A popular example of a high-level data model is the Entity Relationship (E-R) model, which helps in designing databases at the conceptual level. Object oriented data models are also considered high level models and are used for more complex applications

2) Representational (Implementation) Data Models:
These data models are closer to physical storage but still hide some details of data storage. They are widely used in modern DBMS because they can be implemented directly on a computer system. Common examples include: Relational model (data in tables), Network model (data in records connected by links), Hierarchical model (data in tree structure). These models are

more concrete & efficient for database operation compared to high-level models.

3] Low-Level (Physical) Data Models:

These models describes how the data is actually stored in the computer system, including details about file formats, indexing, record placements, & storage paths. They provide details that are necessary for fine-tuning performance & storage efficiency making them useful mostly for database administrators & system designers.

These are object data models, like the ODMG object model, which support features like object identity & encapsulation bringing object oriented concepts into database design. These models can be seen as a bridge between conceptual & implementation models.

5] Explain the component modules of DBMS & their interaction with diagram.

→ The component modules of DBMS includes:

1] Users types and their input:

There are four types of users interacting with the DBMS.

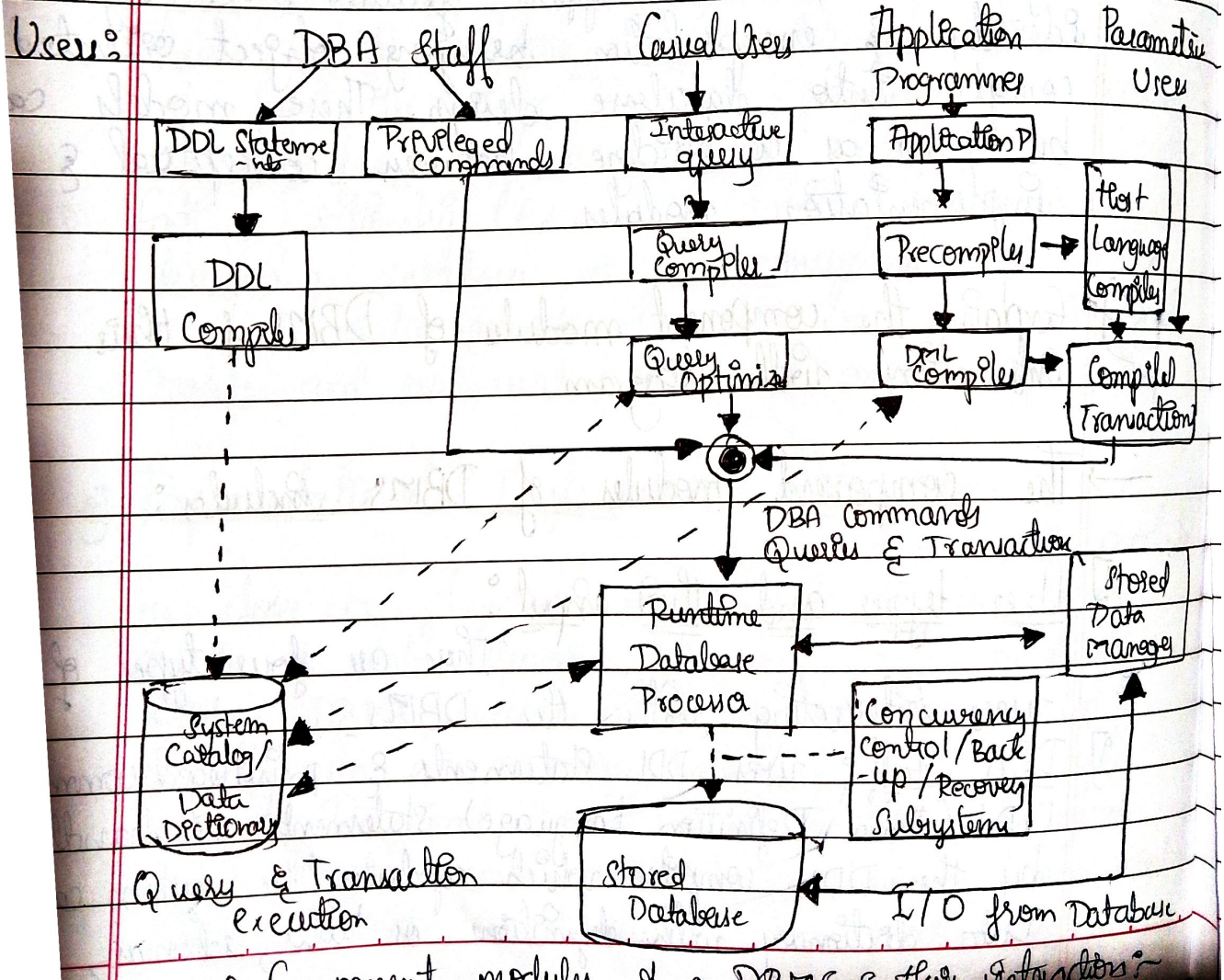
1] DBA Staff: uses DDL statements & privileged commands DDL (Data Definition Language) statements are handled by the DDL compiler which updates the system catalog/ Data dictionary with definitions of tables, schemas,

constraints etc.

ii] Casual Users: Use interactive queries, typically SQL. These go through the query compiler & then the query optimizer which generates efficient execution plans.

iii] Application Programmers: Write programs that interact with the database. These programs go through a precompiler, which extracts DML (Data Manipulation Language) statements.
DML code is passed to the DML compiler & non-DML parts are compiled by the Host language compiler into compiled transactions.

iv] Parametric Users: Use predefined compiled transactions directly (eg in ATMs or POS systems).



~: Component modules of a DBMS & their interactions:-

2] Compilation Modules: These modules translate user commands into a form that the DBMS can execute.
i] DDL Compiler: Processes DDL statements to define or modify database structure.
ii] Query Compiler: Translates interactive queries into low-level instructions.
iii] Query Optimizer: Optimizes queries for efficient execution (eg choosing best join methods & access paths).
iv] Precompiler: Extracts DML statements from application programs before they're compiled.
v] DML Compiler: Translates data manipulation language (DML) commands (like SELECT, INSERT) into executable code.
vi] Host Language compiler: Compiles program that contain embedded DML into machine code.
vii] Compiled Transactions: The final output of application programs, ready to be executed by the DBMS.

3] Runtime DBMS Components (Execution Phase): These components work together during the actual execution of queries & transactions.
i] Runtime Database Processor: Executes compiled queries & interacts with the stored data.
ii] Stored Data Manager: Manages the actual storage & retrieval of data on disk.
iii] Concurrency Control / Backup / Recovery Subsystem: Ensures safe concurrent access, manages transaction consistency & handles failures.

4] System Catalog / Data Dictionary: Stores metadata about the database such as schema, tables, attributes, constraints, views & access rights.

Accessed by many modules (like the compiler, optimiser & runtime processor) to ensure operations are valid & secure.

5] Stored Database: This is the actual database stored on disk, consisting of files containing data & metadata.

Interaction:

1] Users provide input (queries or commands).
2] Input is compiled & optimized by respective compilers.
3] Runtime processor executes this instructions.
4] It communicates with: Stored data manager (to access the database), Concurrency control & recovery (for safe execution), System Catalog (for schema information).
5] Results are returned to users.

6] What are the responsibilities of DBA & database designers?

→ 1] Database Administrator: [DBA] The DBA is the chief administrator responsible for managing database resources in any organization. Their key responsibilities include:

* Authorizing access to the database (security control).
* Co-ordinating & monitoring database use.
* Acquiring software & hardware as needed.
* Ensuring database security and response time performance.
* Managing the database system software & database itself.

In large organisations, the DBA's duties are often supported by a dedicated team.

2] Database Designer: The database designer is responsible for planning the structure & layout of the database before it's built. Their main tasks include:

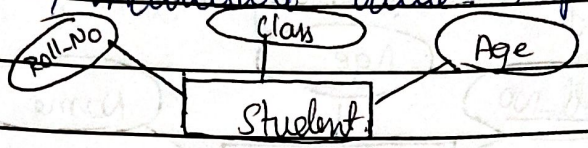
* Identifying the data to be stored in the database.
* Choosing appropriate structures to represent & store the data.
* Understanding the requirements of all user groups by interacting with them.
* Designing views of the database to meet specific group needs.
* Integrating different views into a final database design that supports all user requirements.

Often work closely with the DBA & may also take on other roles post implementation.

2] Explain types of attributes

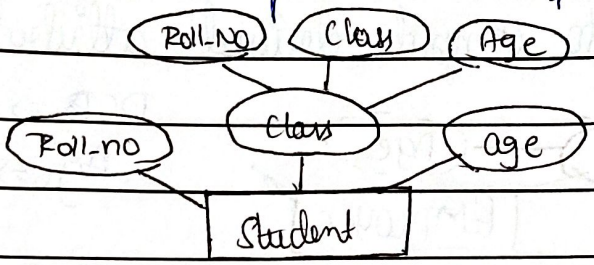
→ Attributes are the property that describes an entity. In table attributes are the columns that store data.

1] Simple (Atomic) Attribute : An attribute that cannot be divided into smaller parts.
* Stores single, indivisible value. Represented by oval shape.
Ex:



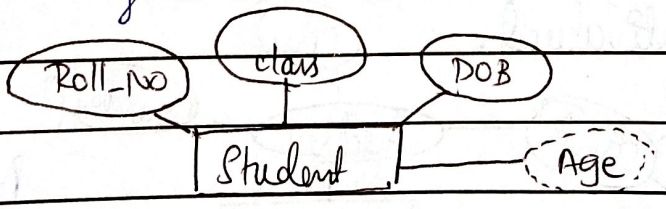
Roll_NO, Class } simple attributes

2] Composite Attribute : An attribute that can be broken down into smaller sub-parts (each sub part also hold meaning)
Ex:



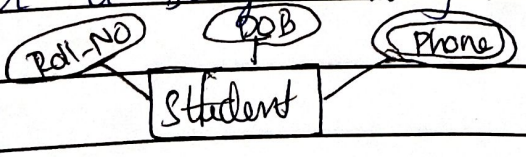
Class ⇒ composite attribute

3] Derived Attributes : An attribute whose value is calculated or derived from other stored attributes. Not stored directly in database generated when needed. Represented by
Ex:



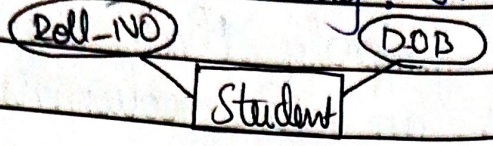
dashed oval
Age ⇒ derived attribute

4] Multi-valued attribute : An attribute that can hold multiple values for a single entity. Represented by double oval.
Ex:



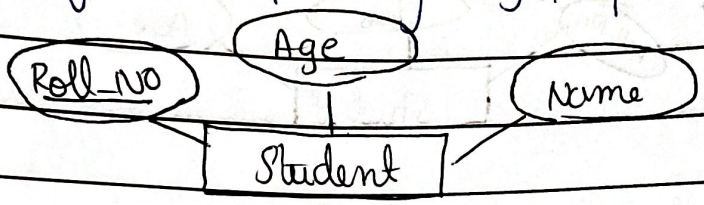
phone ⇒ multivalued attribute

5] Single valued attribute: An attribute that holds only one value for each entity. It is represented by single oval.
Ex:



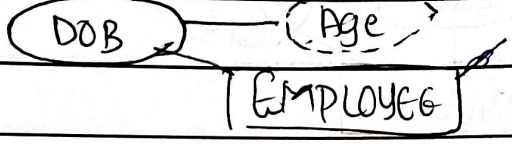
Roll_NO } single valued
DOB

6] Key Attribute: An attribute used to uniquely identify an entity in a table.
* Forms part of the primary key, represented by underlined oval. Ex:



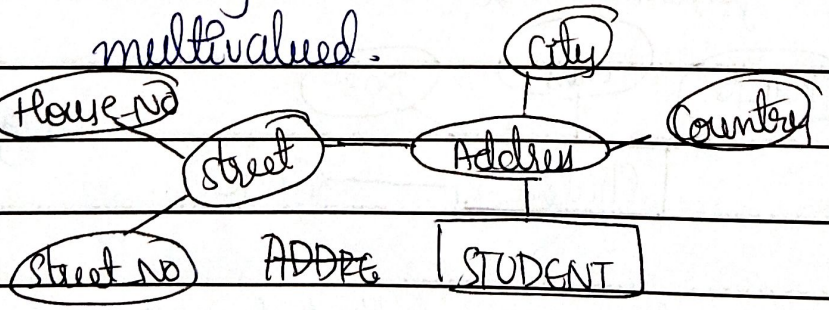
Roll_NO ⇒ key attribute

7] Stored Attribute: The actual value that is stored in database. Used to compute derived attribute. Represented by regular oval.
Ex:



DOB ⇒ is stored attribute
Age ⇒ is derived attribute

8] Complex Attribute: An attribute that combines composite & multivalued characteristics. used in advance DBMS & ER model. Represented by oval connected to both composite & multivalued.
Ex:



Address ⇒ complex attribute

9] Explain the university database, drawing the tables required along with defining, constructing & manipulation of university database

→ 1] Description of university database: A university database keeps track of:

* Students

* Courses * Instructors * Departments * Class enrollment
It allows for operations like student registration, instructor assignments, course offering and result management

2] Table Required : Here are the main tables typically used in a university database :

Table Name	Purpose
Student	Stores student details
Instructor	Stores instructor details
Department	Stores department information
Course	Stores course details
Section	Represents a particular offering of a course
Teaches	Links instructors to course sections
Takes	Links students to course sections
Classroom	Stores classroom info.
Time_slot	Stores timings for classes

Example :

1] Student : Student (student_id, name, dept_name, total_credits)

2] Instructor : Instructor (instructor_id, name, dept_name, salary)

3] Department : Department (dept_name, building, budget)

4] Course : Course (course_id, title, dept_name, credits)

5] Section : Section (course_id, sec_id, semester, year, building, room_no, time_slot_id)

6] Teaches : Teaches (instructor_id, course_id, sec_id, semester, year)

3] Takes: Takes(student_id, course_id, sec_id, semester, year, grade)

3] Database Constructor: [DLL - Data Definition Language].

Creating the student table:

```
CREATE TABLE student (
    student_id VARCHAR(10) PRIMARY KEY,
    name VARCHAR(50),
    dept_name VARCHAR(30),
    total_credits INT
);
```

4] Data Manipulation (DML - Data manipulation Language):

Insert Data:

```
INSERT INTO student Value('S01', 'Anita', 'CS', 120);
```

Update Data:

```
UPDATE student SET total_credits = 130 WHERE student_id = 'S01';
```

Delete Data:

```
DELETE FROM student WHERE student_id = 'S01';
```

Select Data:

```
SELECT name FROM student WHERE dept_name = 'CS';
```

10] Develop an ER diagram for keeping track of information about a company database taking into account at least five entities

1] Explain the DBMS Languages.

→ DBMS Languages are categorized based on their purpose in database operations. They ~~are~~ These languages are

1] ~~Data~~ Definition is used to define, manage & interact with a database. These languages are designed to allow users to create the structure of the database, store and retrieve dat and manage access and storage mechanisms.

DBMS languages can be classified into four major types:

1] Data Definition Language (DDL): It is used to define the structure and schema of the database. It includes command that allow you to create, alter & delete database objects such as tables, indexes & views.

Ex:
```
CREATE TABLE Students (
    StudentID INT,
    Name VARCHAR(100),
);
```

2] Storage Definition Language (SDL): It is concerned with the internal schema of the database. i.e how the data is stored physically on the storage medium. It defines low-level storage structures like file formats, indexes & data compression techniques.

In modern DBMS systems, SDL is not often exposed directly to users. Instead, it is handled automatically or configured through DBMS parameter and options.

3] View Definition Language (VDL): It is used to define external schemas or views of the database. A view is a virtual table that shows specific parts of the database depending on the user's needs or security access.

Example: CREATE VIEW StudentView AS
SELECT Name, Age FROM Students;

4] Data Manipulation Language (DML): It is used to perform data operation such as inserting, modifying, deleting or querying data in the database.

There are two types of DML:

a] High-level (Non-Procedural) DML: It operates on sets of data. It has simple syntax, often used in SQL and does not require knowledge of how the data is stored or retrieved

Example: SELECT * FROM Students WHERE Age>18;

b] Low-level (Procedural) DML: It operates on one record at a time. It requires logic and looping. Used in embedded programming with SQL, like PL/SQL, T-SQL etc

Example:

SELECT Name INTO v_name FROM
Students WHERE StudentID = 101;

DBMS Interfaces:

It provides multiple interfaces for different types of users:

1] Menu-based Interfaces: Common in web & client apps. Users interact by selecting option from menus.

2] Forms-based Interfaces: Users input data into forms, which are processed by the DBMS. Common in applications like oracle forms.

3] Graphical User Interfaces (GUI): Visual representation of the database schema. Users interact through drag & drop of visual elements.

4] Natural Language Interfaces: Users type commands in natural language (eg: English). System interprets & transfers

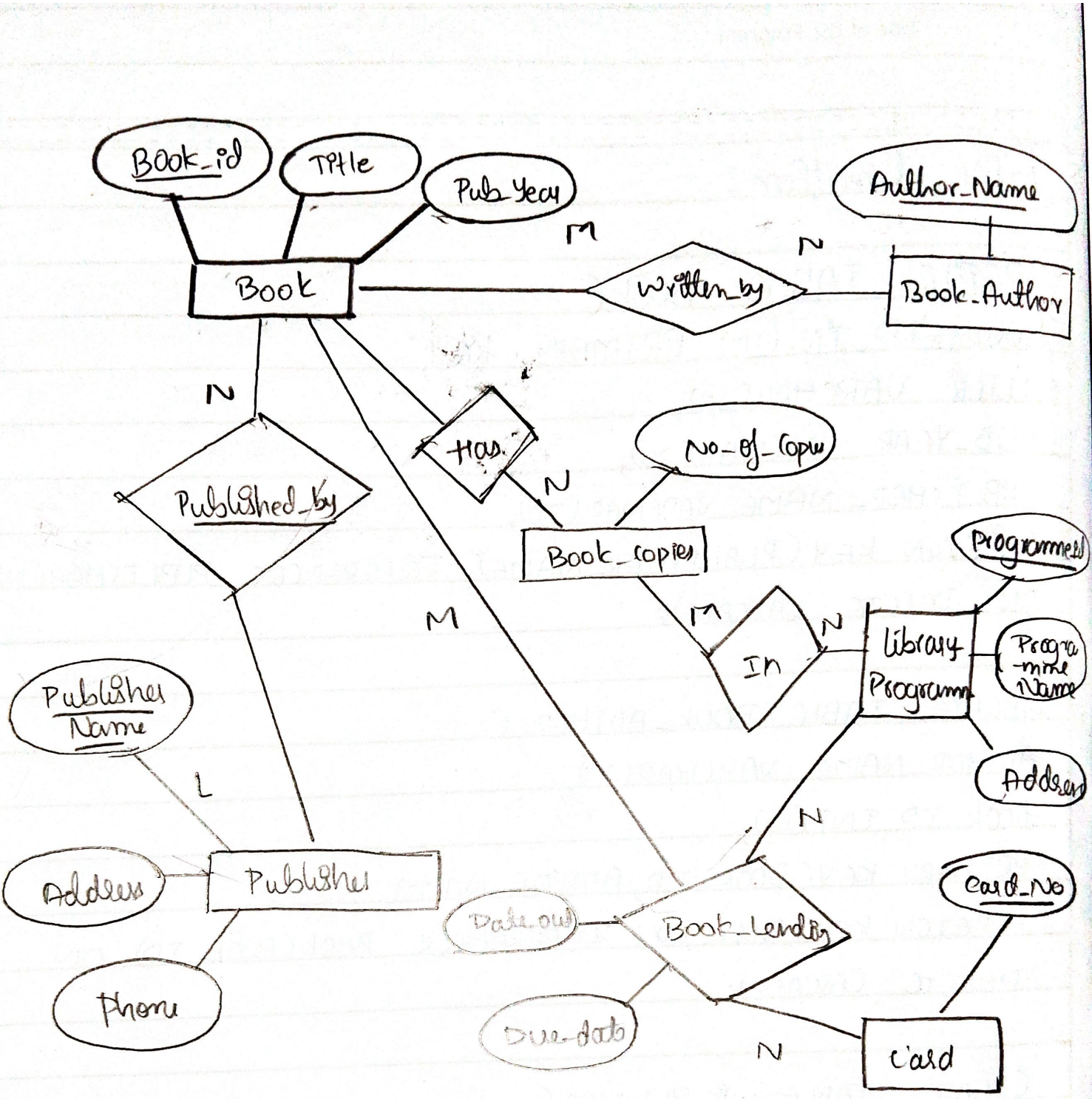
translates them into DBMS queries.

5) Keyword-based Search Interfaces: Similar to search engines users enter keywords & get results, emerging in relational database.

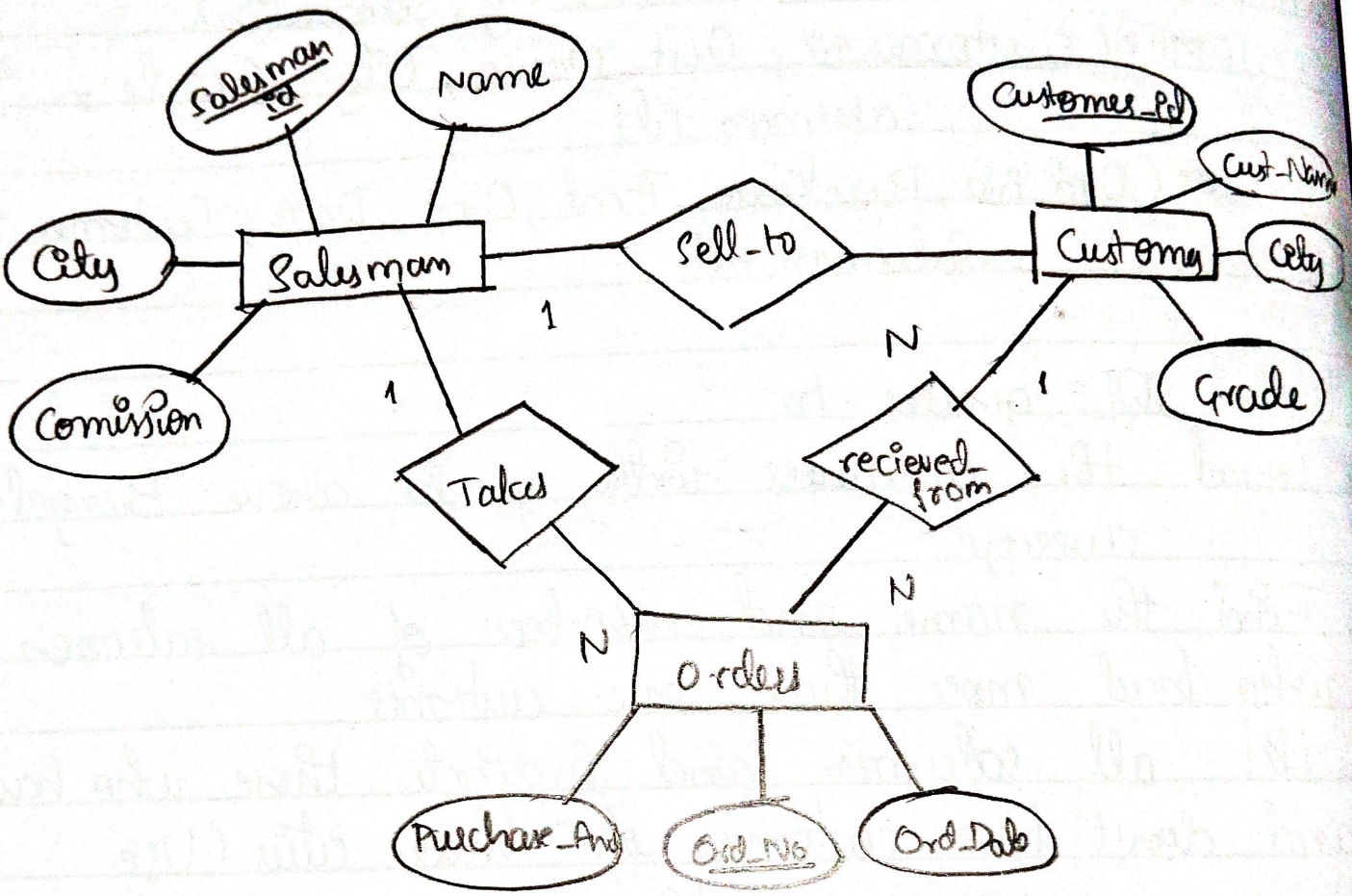
6) Speech Input/Output Interfaces: Allow spoken queries & responses. Used in specific applications like customer services

7) Interfaces for parametric users: Predefined functions for users like bank clerks (eg. balance enquiry). Simple, repetitive operations.

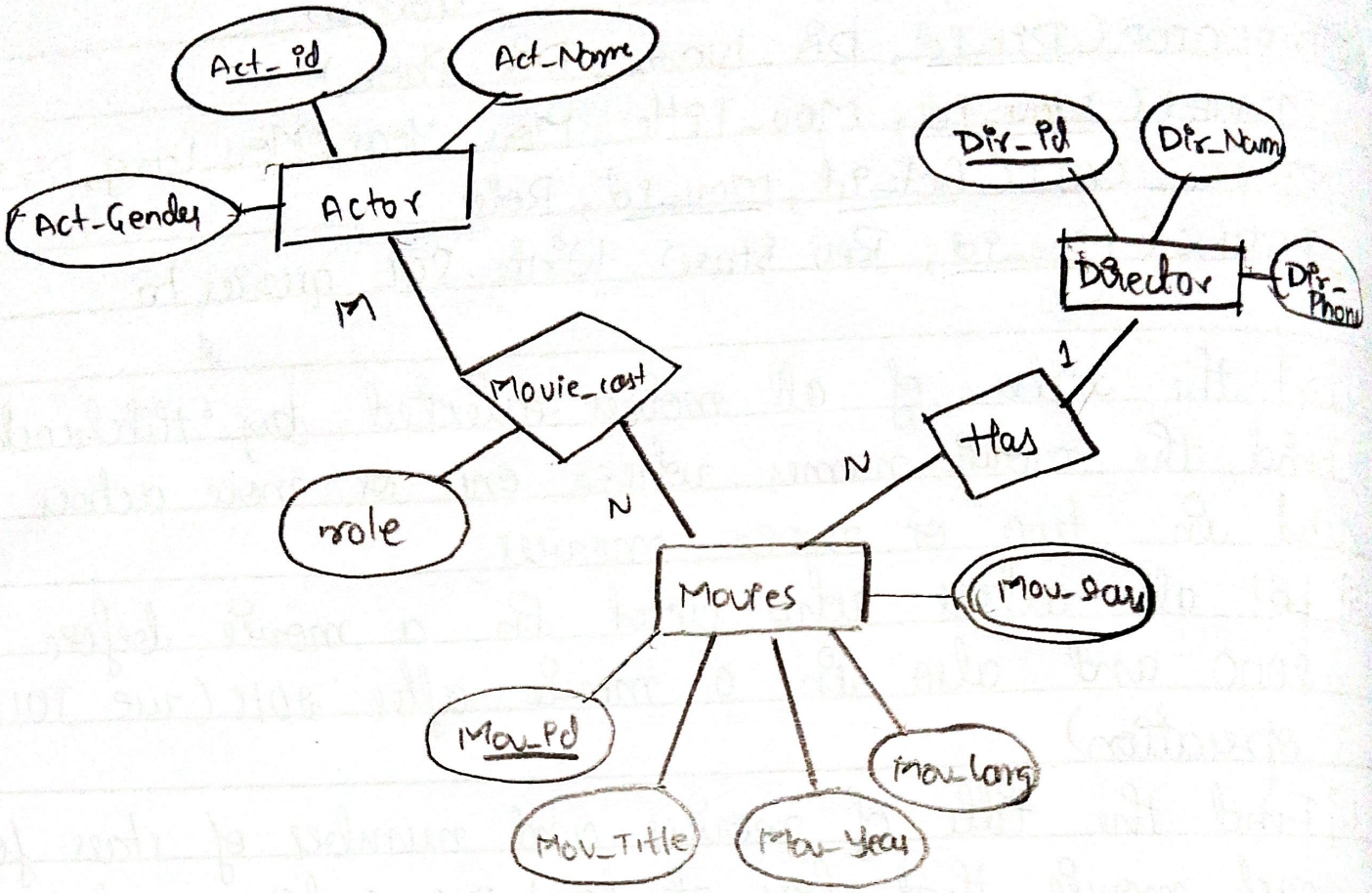
8) Interfaces for DBAs: Special commands to manage database security, structure & performance.



ER DIAGRAM FOR LIBRARY DATABASE



ER DIAGRAM FOR ORDER DATABASE



ER DIAGRAM FOR MOVIE DATABASE