

Data:

Data is a collection of facts and figures that can be processed to produce information.

Database:

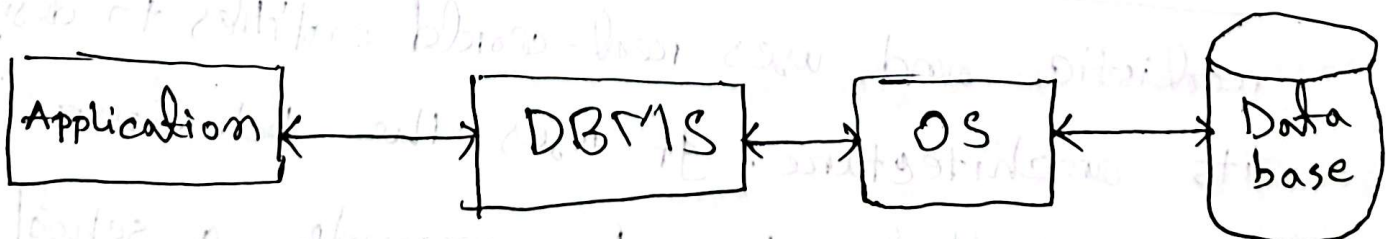
Database is a collection of related data.

Database Management System:-

A database management system stores data in such a way that it becomes easier to retrieve, manipulate and produce information.

A ~~DBMS~~ DBMS is a software that allows creation, definition and manipulation of database. DBMS is actually a tool used to perform any kind of operation on data in database. DBMS also provides protection and security to database.

Example: MySQL, Oracle.



DBMS can also be define as an interface between the application program and the operating system to access and manipulate that database.

* 1. List and explain four reason why a DBMS is used instead of file processing system.

= Traditionally data was organized in file formats.

DBMS was a new concept then, and all the research was done to make it overcome

the deficiencies in traditional style of data management.

The reason why DBMS is used is given below:

▣ Real world entity: A modern DBMS is more

realistic and uses real-world entities to design

architecture: it uses the behaviours

and attributes too. For example, a school

database may have used students as an entity and their ages as an attribute.

☐ Relation-based tables: DBMS allows entities and relations among them to form tables. A user can understand the architecture of a database just by looking table names.

☐ Less Redundancy: DBMS follows the rules of normalization, which splits a relation when any of its attributes is having redundancy in values. Normalization is a mathematically rich and scientific process that reduces data redundancy.

☐ ACID Properties: DBMS follows the concepts of Atomicity, consistency, Isolation, and Durability. These concepts are applied on transactions, which manipulate data in a database.

ACID properties help the database stay healthy in multi-transactional environments in case of failure.

* TDB = Traditional Database

* MDB = Multimedia Database

* GIS = Geography Information System.

2.

Database

DBMS

1. A database is a collection of related data about people, location or things.	1. A database management system is a collection of programs that allow one to create, manage and operate database
2. Besides computers DB can even be maintained in physical ledgers, books or papers.	2. In a DBMS all the records are maintained only on a computer
3. The retrieval of information from the databases can be done manually, through queries or by using programs. (C, C++, Java)	3. We can retrieve the data from the DBMS through queries written in SQL
4. The retrieval of information is very slow.	4. The retrieval of information is very quick
5. Data stored is stored in DB.	5. A DBMS manages and manipulates data.
6. In DB very less information can be modified at a time.	6. In DBMS a lot of info can be modified at a time.

3. What is SQL?

= SQL (Structured Query Language) is a standard language used to communicate and manipulate databases. It is used for querying, updating and managing data in relational database management systems (RDBMS).

4. Different types of SQL commands →

* DML (Data Manipulation Language): Commands that manipulate data in tables.

* DDL (Data Definition Language): Commands that define the structure of database objects.

* DCL (Data Control Language): Commands that control access to data.

* TCL (Transaction Control Language): Commands that manage transactions.

* DQL (Data Query Language): Primarily the SELECT command, used to query data from the database.

4. Diff.

specialization

Generalization

1. This process operates in a top-down manner.

1. This process operates in a bottom up manner.

2. A user can easily apply a generalization to a group of different entities.

2. We can only apply specialization to a single entity.

3. It leads to reduction in the size of a schema.

3. It leads to an increase in the size of schema.

4. This process leads to formation of a single entity out of multiple entities.

4. This process leads to formation of multiple entities from one single entity.

5. There is inheritance in specialization.

5. There is no inheritance in generalization.

Example

Example

5. Why should 'Null' in a relation be avoided as possible? explain →

= Avoiding the use of NULL values in a relation is a good practice in database design for several reasons.

i. Semantic Ambiguity: Null values can introduce semantic ambiguity, as they represent different things in different contexts. This lack of clarity can lead to misunderstanding and errors in data interpretation.

ii. Complexity in Querying: Handling Null values in queries adds additional logic, such as using the 'IS NULL' or 'IS NOT NULL' conditions. This complexity makes queries more difficult to read, write and maintain.

iii. Aggregation Challenges: NULL values can cause challenges when aggregation data operators like SUM, AVG, COUNT etc. may not provide accurate results.

iv. Data integrity: Null values can make it challenging to enforce data integrity constraints.

v. Indexing & searching: Indexing on columns with NULL values may not be as efficient as indexing on columns with non-NULL values.

6. What is DBA? What are five main functions of a database administrator?

= A database administrator (DBA) is the information technician responsible for directing or performing all activities related to maintaining a successful database environment. A DBA makes sure an organization's database and its related applications operate functionally and efficiently.

Five main functions: →

- a) Recoverability: Creating and testing Backups
- b) Integrity: Verifying or helping to verify data integrity.
- c) Security: Defining or implementing access controls to the data.

d) Availability: Ensuring maximum uptime.

e) Performance: Ensuring maximum performance.

7. What is database trigger? Discuss the strengths and weakness of the trigger mechanism.

= A database trigger is procedural code that is automatically executed in response to certain events on particular table or view in database. The trigger is mostly used for maintaining the integrity of the information on the database.

Strengths:

- Used for enforcing referential integrity.
- Easy to implement business rules through triggers.
- Event logging is easier.
- It can control transactions hence lead to safer transactions.
- consistency is maintained.
- Efficiency in performing transactions.

Weakness :

- Triggers should be written with extra care.

Trigger error may lead to failure of insert, delete or update operations at runtime.

- Excessive use of triggers can result in complex interdependencies.

Increased complexity.

- May decreased the performance of the database.

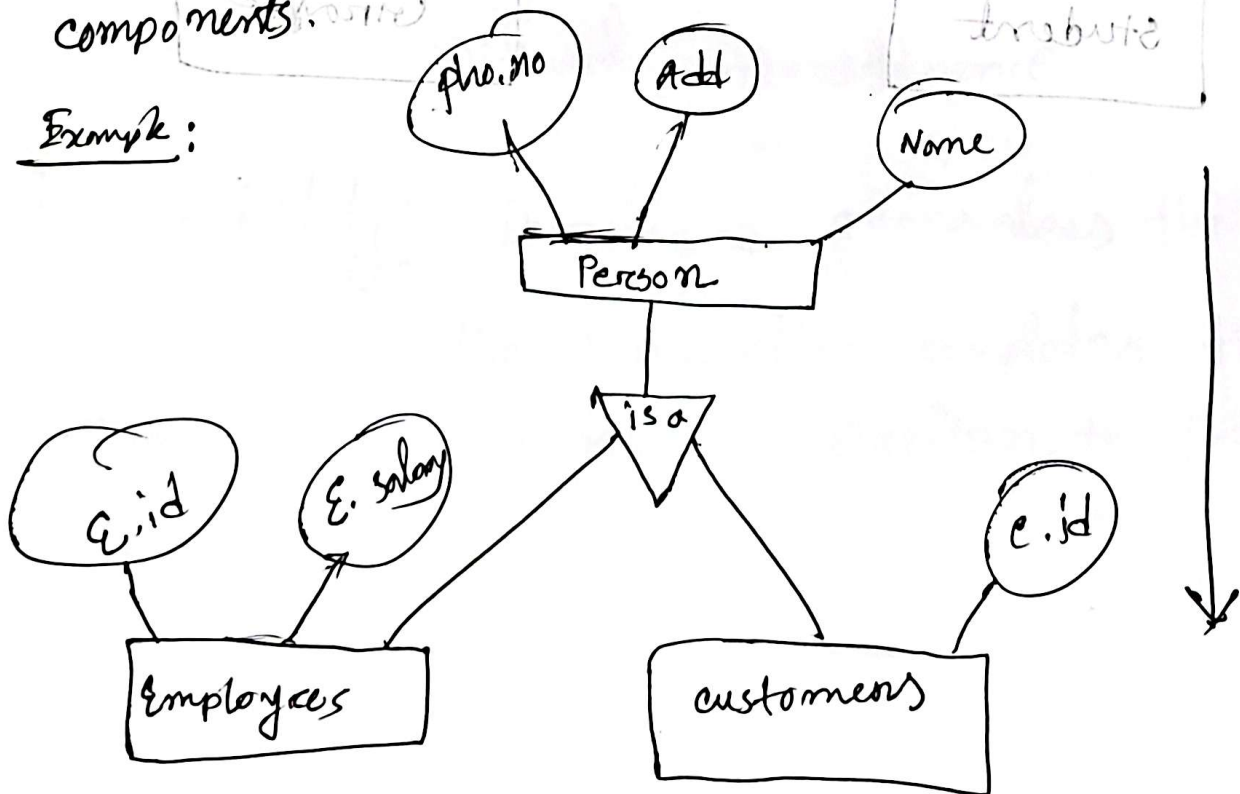
- Everytime trigger is fired, its code must be recompiled.

Specialization:

specialization is the process of dividing a high level entity into lower level entities based on uncommon attributes.

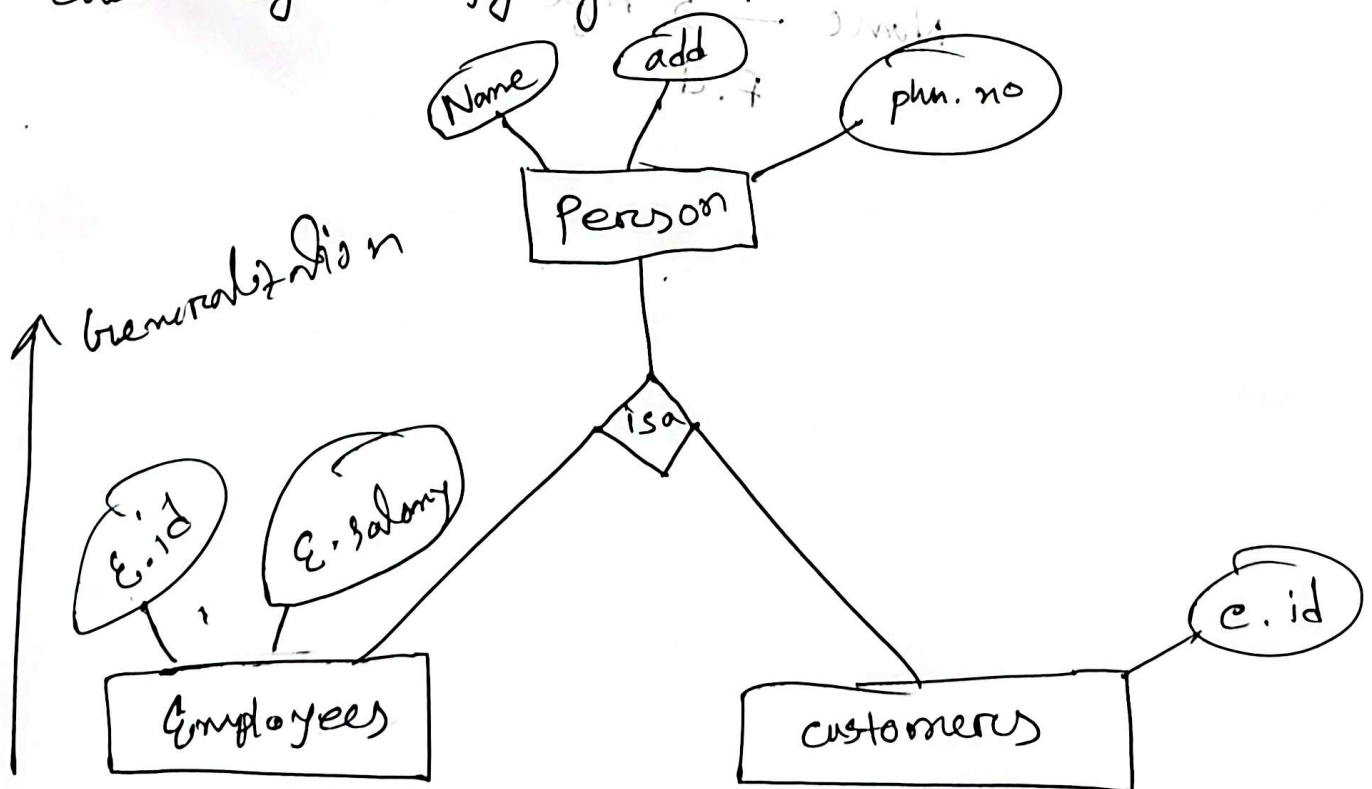
- It is a top-down approach.
- Special attributes form the basis of specialization.
- It expands the schema by multiplying the components.

Example:



Generalization:

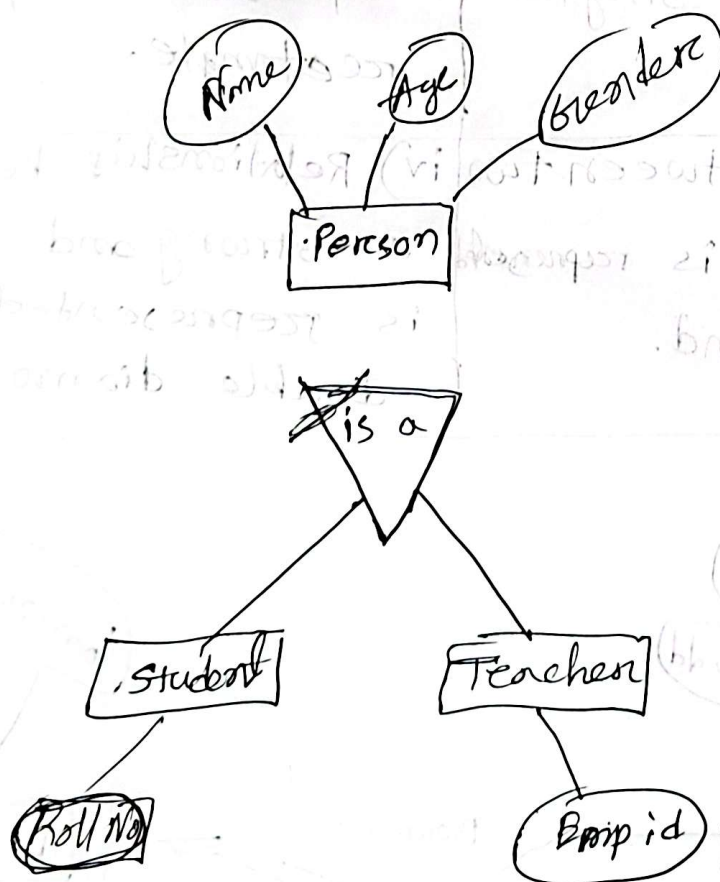
- It is the process of grouping lower level entities into broader categories (high level entity) based on common attribute.
- It is a bottom-up approach.
 - Common attributes between different entities ~~form~~ form the basis for generalization.
 - Generalization reduces the schema of the data by unifying components.



* Explain in detail about inheritance, specialization, generalization using ER diagram.

= Inheritance:

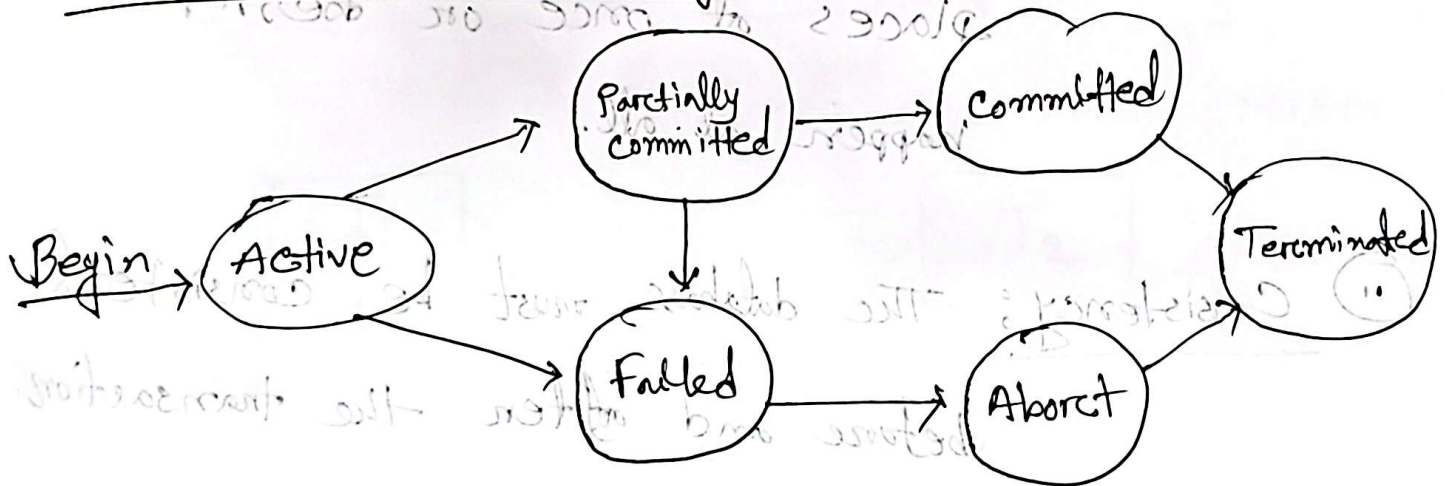
Inheritance is an important feature of generalization and specialization. It allows lower-level entities to inherit the attributes of higher-level entities. For example, the attributes of a person class such as name, age and gender can be inherited by lower-level entities such as student or Teachers.



است 27 first a

Transaction

- It is a set of operations used to perform a logical unit of work.
- Transaction state diagram:



Normalization: It is the process of organizing the data in the database.

- It is used to remove or reduce the redundancy from a relation.

ACID Properties:

① Atomicity: The entire transaction takes place at once or doesn't happen at all.

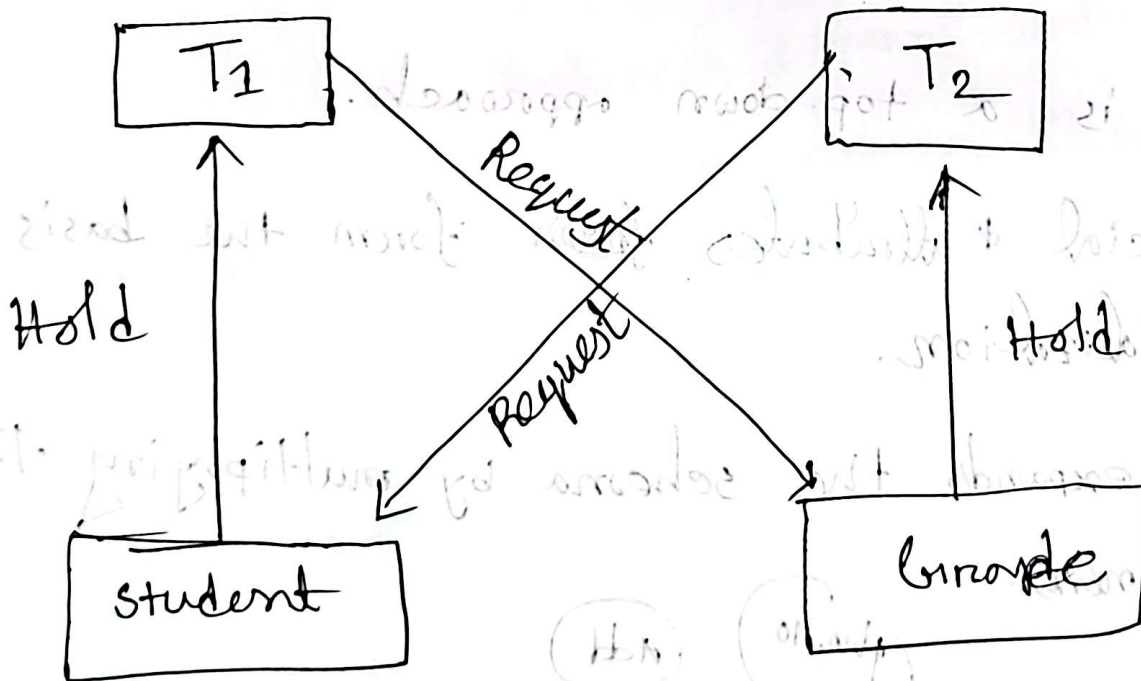
② Consistency: The database must be consistent before and after the transaction.

③ Isolation: Multiple transactions occur independently without interference.

④ Durability: Durability guarantees that once the transaction completes then changes are written to the database.

Deadlock

- A deadlock is a condition where two or more transactions are waiting indefinitely for one another to give up locks.



Key

- Candidate key: The minimal set of attributes that can uniquely identify a tuple is known as a candidate key.
- Super Key: Super key is a single key or a group of multiple keys that can uniquely identify tuples in a table. Super key can contain redundant attributes that might not be important for identifying tuples. Candidate keys are a subset of a super key.
- Primary key: (Not null and unique) A primary key is one of the candidate key chosen by the database designer to uniquely identify the tuple in the relation.
- Key: A key is a attribute or set of attributes that uniquely identifies any record from the table.

Alternative keys: Out of all candidate keys, only one gets selected as primary key, remaining keys are known as alternative keys.

Foreign key: A foreign key is a key used to link two tables together. Foreign key references the primary key of the table.

- Take null value.

There is no restriction on a foreign key to be unique.

Composite key: A key that has more than one attributes is known as composite key. It is also known as compound key.

Advantages of Normalization:

- Normalization helps to minimize data redundancy.
- Data consistency within the database.
- Much more flexible database design.
- Greater overall database organization.
- Maintain database task faster.
- A better handle on database security.
- Increase storage efficiency.
- Speed up data access.

Disadvantages of Normalization:

- you cannot start building the database before knowing what the user needs.
- Requires much CPU, memory and input-output process.
- Requires more joins to get the desired result.
- It is very time-consuming and difficult to normalize relations at a higher degree.
- Careless decomposition may lead to a bad database design, leading to a serious problem.

• Difference between DBMS and RDBMS:

DBMS	RDBMS
1. In DBMS, relationship between two tables or files are maintained programmatically.	1. In RDBMS, relationship between two tables and files can be specified at the time of creation.
2. DBMS applications store data as files.	2. RDBMS applications store data in tabular form.
3. Normalization is not present in DBMS.	3. Normalization is present in RDBMS.
4. DBMS does not support client/server architecture.	4. Most of the RDBMS support client/server architecture.
5. DBMS uses file system to store data, so there will be no relation between the tables.	5. RDBMS uses table to store data, so there will be a relation between the tables.
6. DBMS does not support distributed database.	6. RDBMS support distributed database.
7. DBMS is designed to handle small amount of data.	7. RDBMS is designed to handle large amount of data.
8. It support single user.	8. It support multiple users.
9. Examples: File system, XML etc.	9. Examples: MySQL, Oracle

DDL → Data Definition Language.

DML → Data Manipulation Language

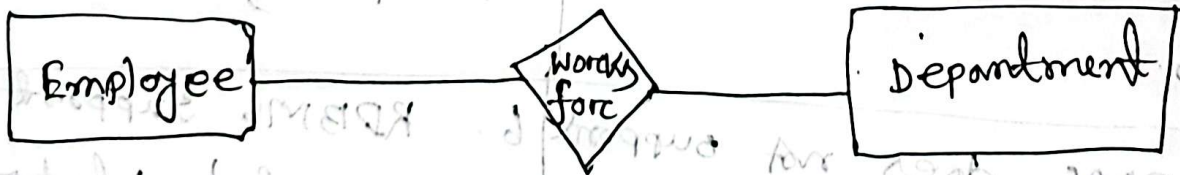
DCL → Data Control Language

TCL → Transaction control language.

Entity: Anything about which we store information is called an Entity.

Entity set: It is a set of entities of the same type that share the same properties or attributes.

An Entity set can be represented as rectangles.



Attributes:

- The attribute is used to describe the ~~property~~ property of an entity.
- Attributes are represented in an elliptical shape.

Key attribute

student - ID

multivalued attribute

phone-no

Derived attribute

Age

for primary school



for higher school



Notation of ER diagram

Student ID

Roll no Attributes.

Roll no Key attributes.

Name
First N.
Last N.
Composite attributes

phon: no multiplicative attributes.

Age derived attributes

strong entity set.

Weak entity set.

Relationship

Links (one to one)

(one to many)

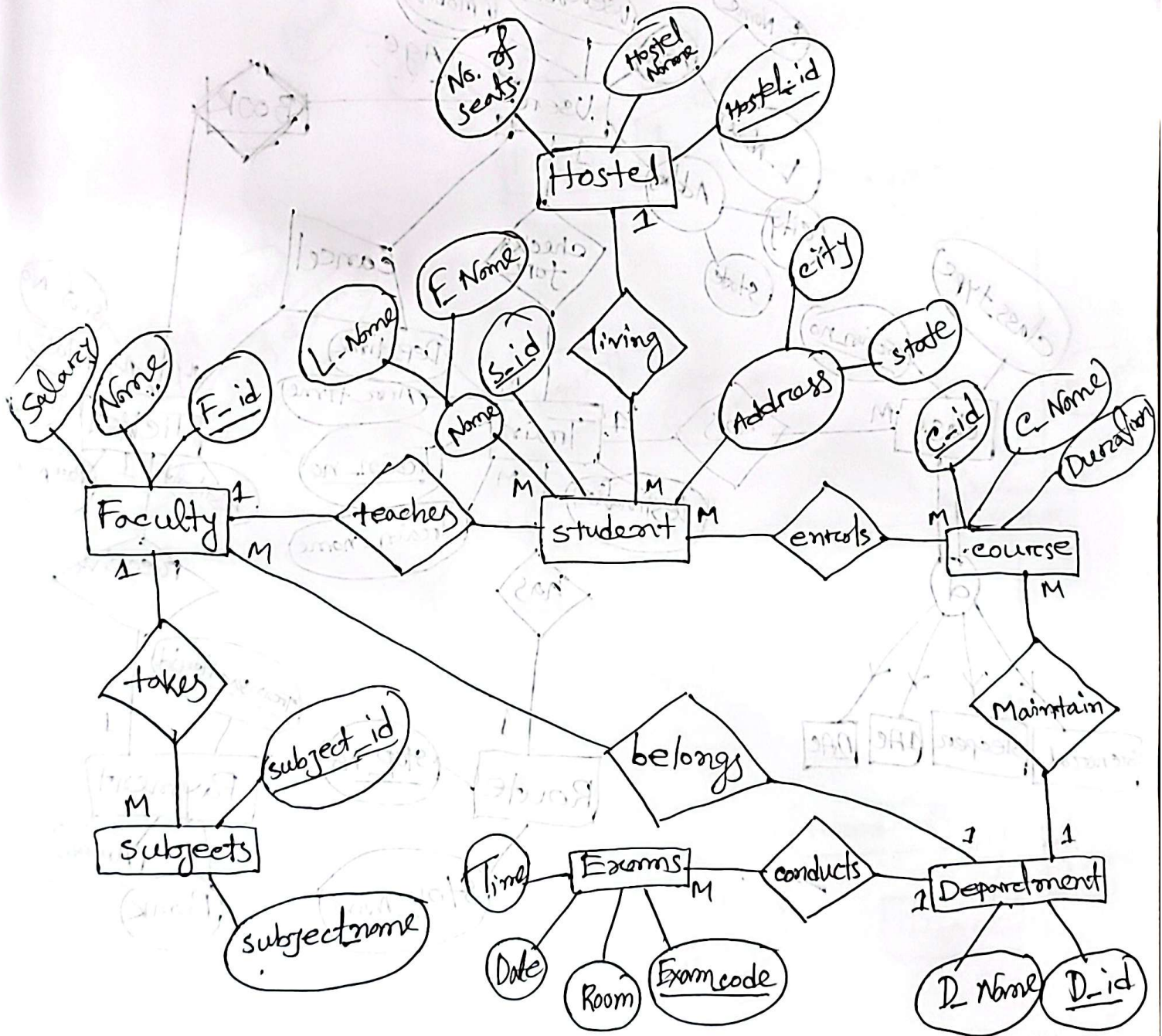
(many to one)

one and only one

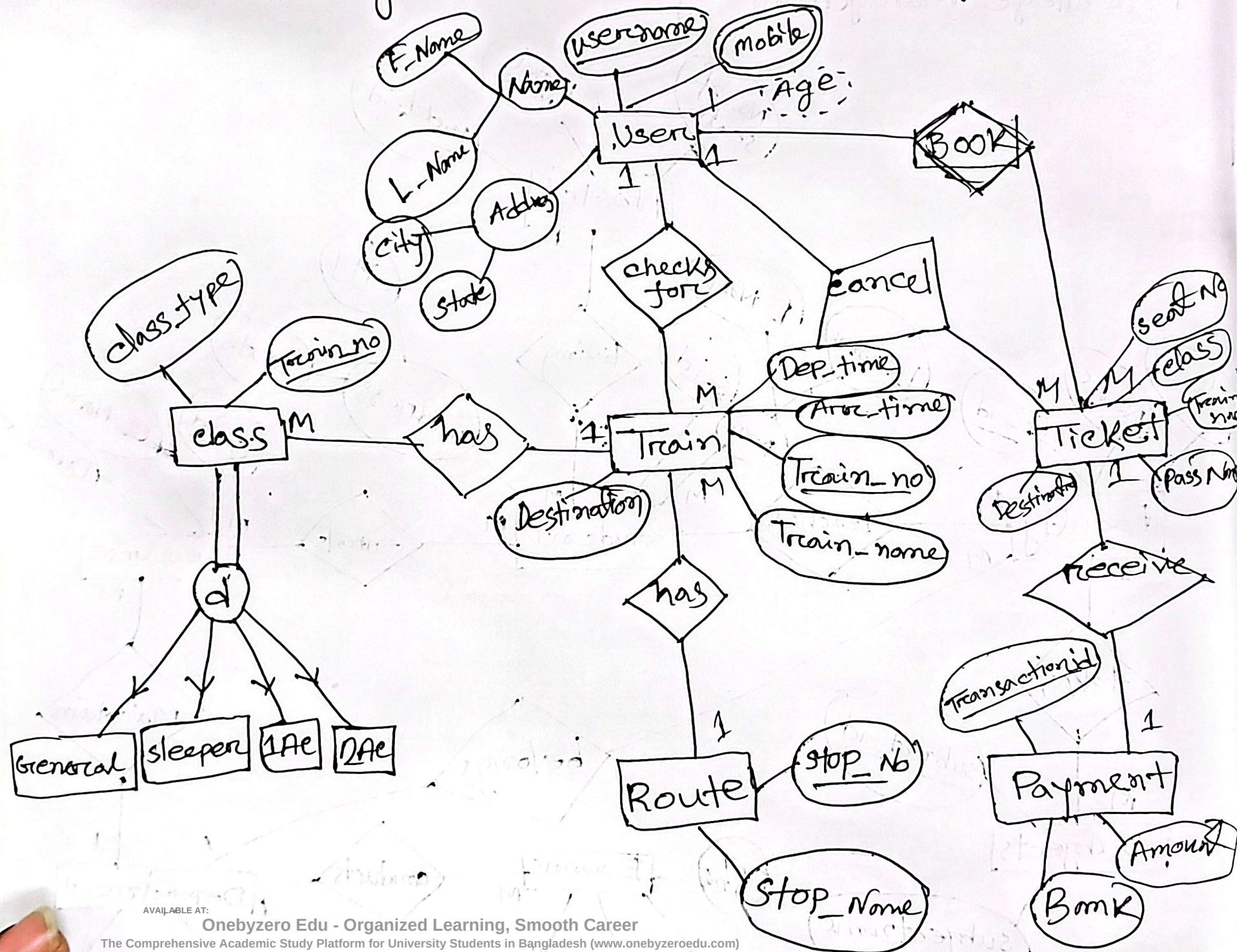
zero or one

zero or many

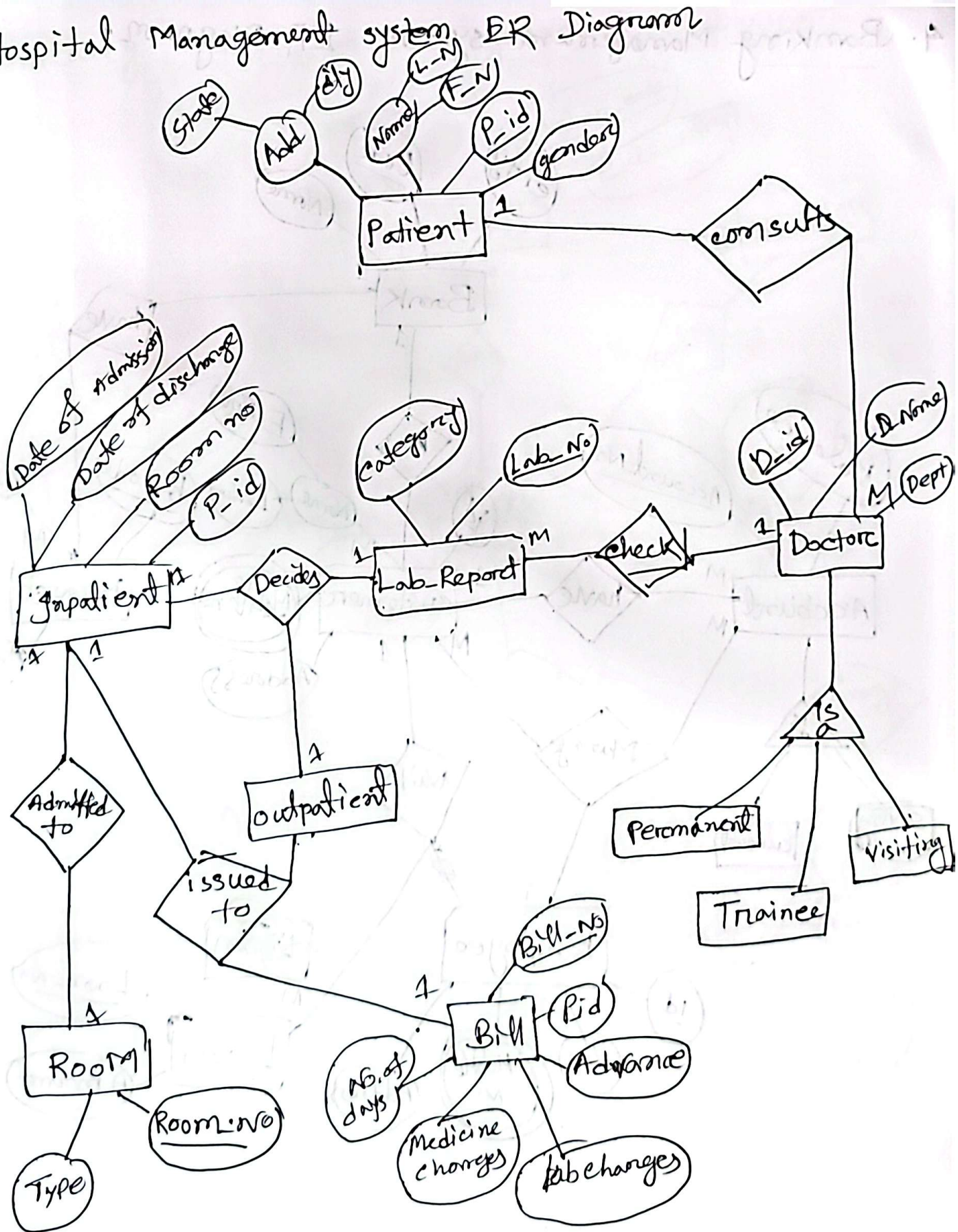
1. college Management system ER-Diagram.



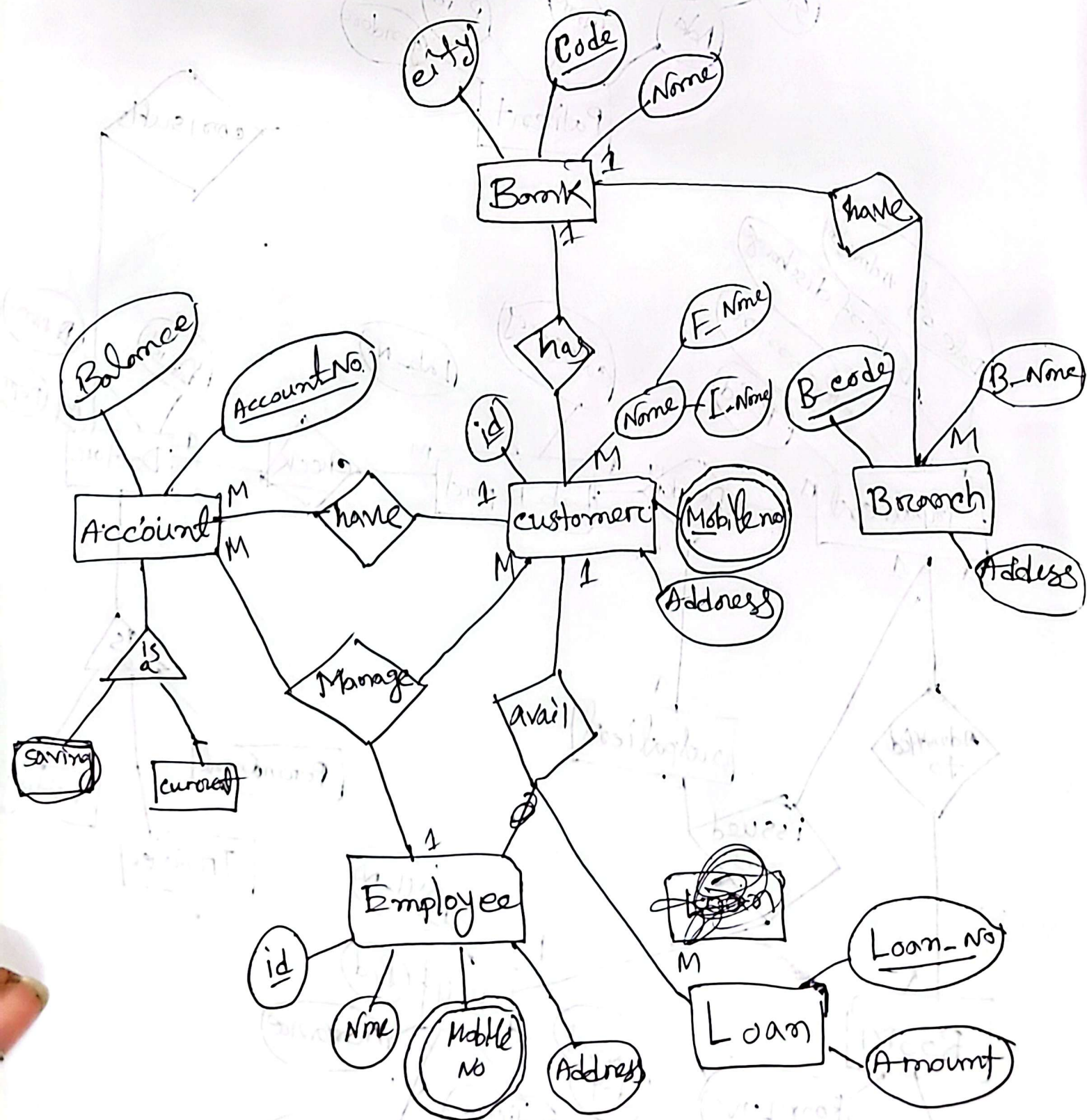
2. Online Railway Reservation system ER Diagram.



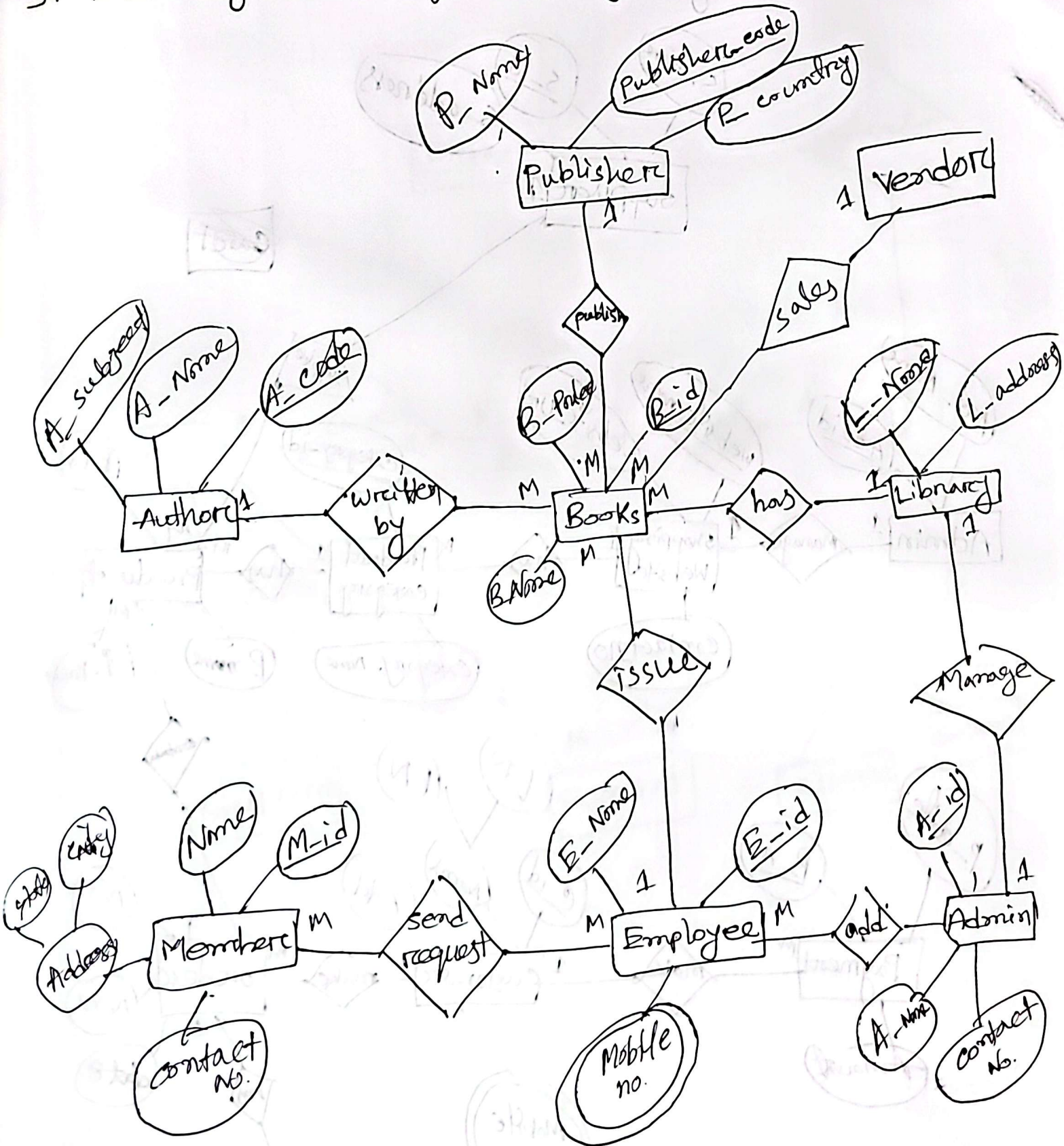
3. Hospital Management system ER Diagram



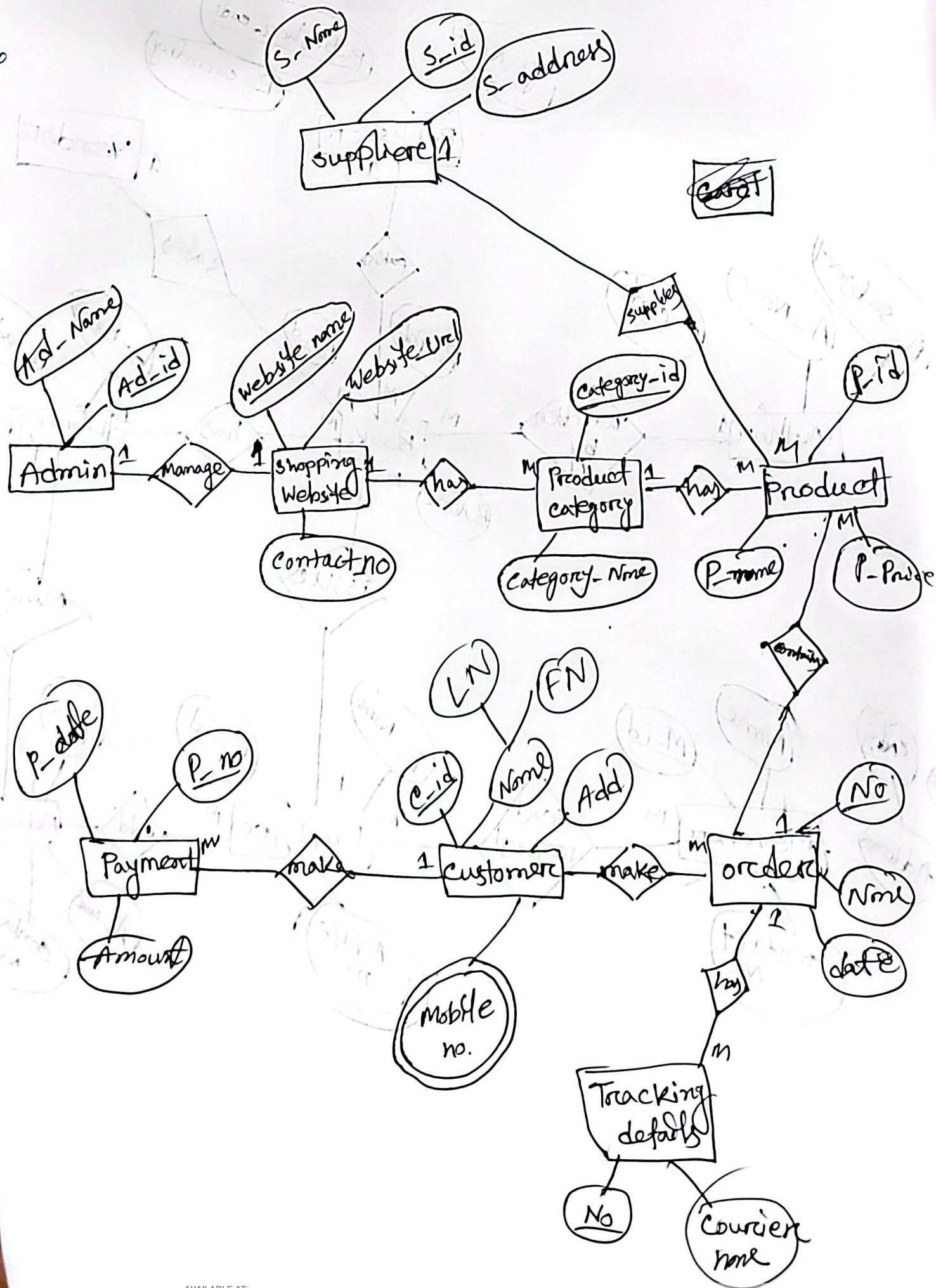
4. Banking Management system ER Diagram:



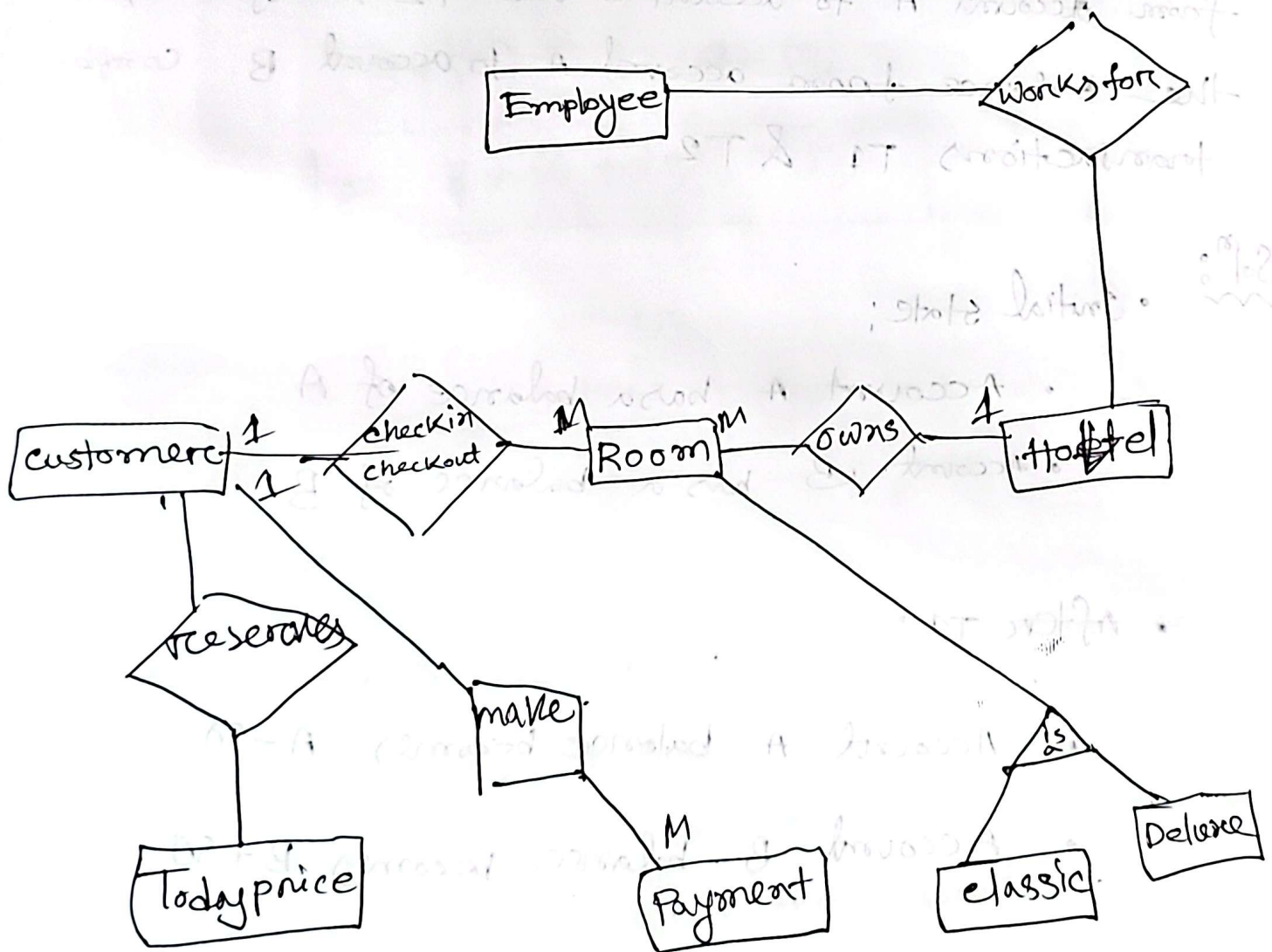
5. Library Management system ER Diagram.



6. Online shopping Management system etc



7. Hotel Management system ER Diagram.



* Consider two transactions T1 and T2. T1 transfers 50\$ from account A to account B and T2 transfers 10% of the balance from account A to account B. Complete transactions T1 & T2.

Solⁿ:

• Initial state;

• Account A has a balance of A.

• Account B has a balance of B.

• After T1:

• Account A balance becomes $A - 50$.

• Account B balance becomes $B + 50$.

Again,

• Account A has new balance of $A - 50$.

• Account B has new balance of $B + 50$.

Calculation for T2:

• 10% of the balance in Account A is ~~0.10~~ $0.10 \times (A - 50)$.

• After T2:

Account A balance becomes
 $(A - 50) - 0.10 \times (A - 50)$

$$= 0.90 (A - 50)$$

• Account B balance becomes

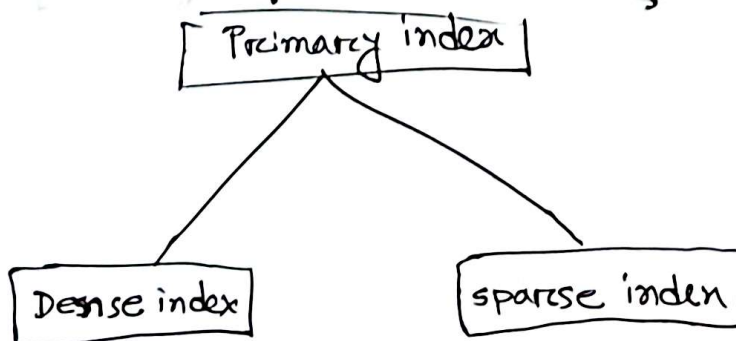
$$(B+50) + 10 \times (A-50)$$

$$= B + 50 + 10 \times (A - 50)$$

Index

Primary index :

- If the index is created on the basis of the primary of the table, then it is known as primary indexing. These primary keys are unique to each record and contain 1:1 relation between the records.
- As primary keys are sorted in sorted order, the performance of the searching operation is quite efficient.



Dense index:

- The dense index contains an index record for every search key value in the data file. It makes searching faster.
- In this, the number of records in the index table is same as the number of records in the main table.
- It needs more space to store index record itself.

UP	•	→	UP	Agroa	1,604,300
USA	•	→	USA	chicago	2,789,378
Nepal	•	→	Nepal	Katthimunder	1,456,34
UK	•	→	UK	Embridge	163045

Sparse index:

- In the data file, index record appears only for a few items. Each item points to a block.
- In this, instead of pointing to each record in the main table, the index points to the records in the main table in a gap.

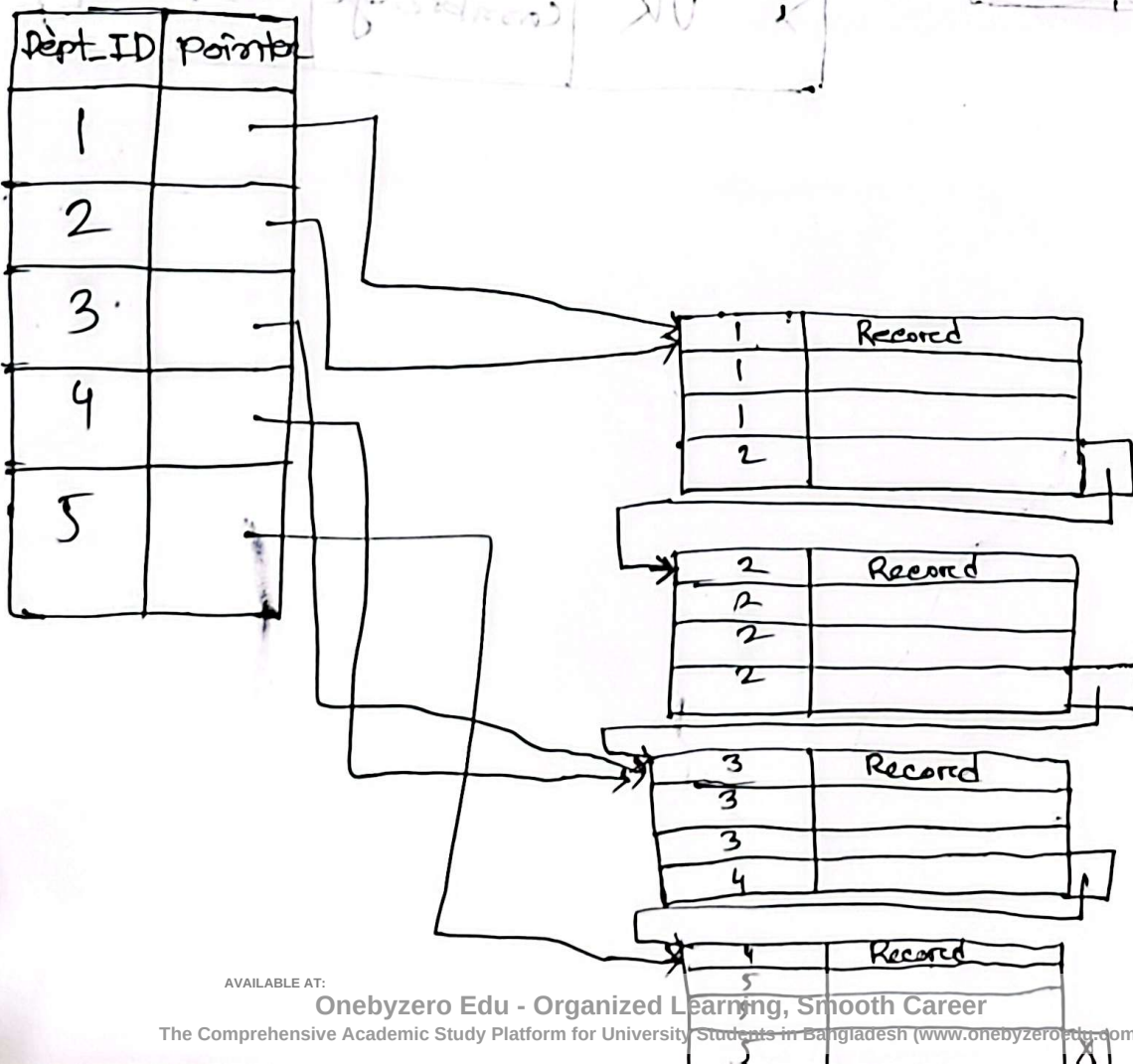
UP		→	UP	Agra	1604300
Nepal		→	USA	Chicago	2789378
UK		→	Nepal	Kathmandu	1456634
 	 	→	UK	Cambridge	1360364

Clustering index:

A clustered index can be defined as an ordered data file. Sometimes the index is created on non-primary key columns which may not be unique for each record. In this case, to identify the record faster, we will group two or more columns to get the unique value and create index out of them. This method is called a clustering index.

Example:

A company have →



Dept id	Pointer
1	
2	
3	
4	
5	

1	Record
1	
1	
	X

2	Record
2	
2	
2	

2	Record
	X

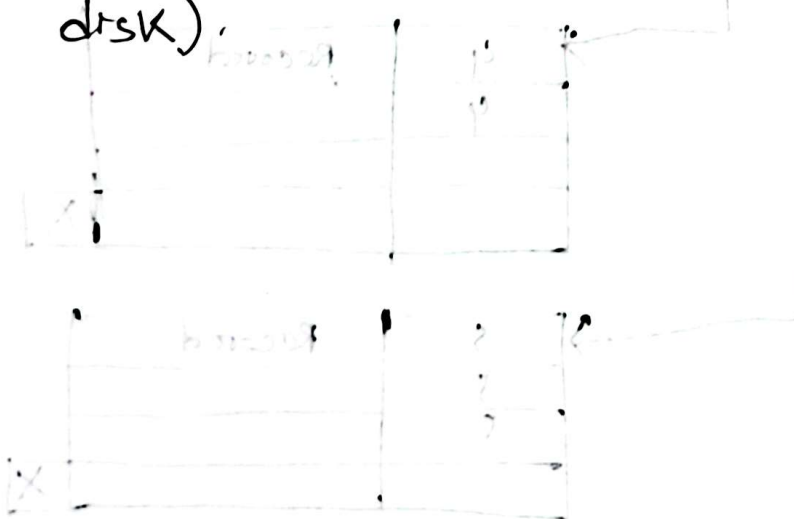
3	Record
3	
3	
	X

4	Record
4	
	X

5	Record
5	
5	
	X

Secondary index:

In secondary indexing, to reduce the size of mapping, another level of indexing is introduced. In this method, the huge range for the columns is selected initially, so that the mapping size of the first level becomes small. Then each range is further divided into smaller ranges. The mapping of the first level is stored in the primary memory, so that address fetch is faster. The mapping of the second level and actual data are stored in the secondary memory (Hard disk).



* What are the differences between a primary index and a secondary index?

⇒ Primary index	Secondary index
i) Only based upon the primary key.	i) It maybe relies on candidate key or secondary key.
ii) Primary index is a kind of nondense index.	ii) Secondary index is a type of dense index.
iii) It does not contain duplicates.	iii) It can contain dupl. duplicates
iv) Requires less storage space.	iv) Requires less more storage space.

* If an index structure occupies 1000 disk blocks, how many blocks reads will be required in the best case and worst case.

= Best case:

The best case occurs when the desired index entry is located in the first block read. Only 1 block read is required.

Worst case:

In the worst case, the search must scan all 1000 blocks. This requires 1000 block reads.

* Explain dirty read problem?

= A dirty read problem occurs in a database when a transaction reads uncommitted data from another transaction. This can lead to inconsistencies, making the data read incorrect or invalid.

T ₁	T ₂
R(A)	
W(A)	R(A) // Dirty
⋮	W(A) Read.
Failure	commit

* To Prevent dirty reads →

- Use an isolation level of Read committed or higher in your transactions.
- Use rules for the execution of concurrent transactions, known as concurrency control protocols.
- Use concurrency control mechanisms like locking, timestamp ordering and multiversion concurrency control.
- Have the transaction wait until rows write-locked by other transactions are unlocked.

* When is it preferable to use a dense index rather than a sparse index?

= In an indexing structure, a dense index contains an entry for every record in the data file, while a sparse index contains only for some records.

✓ A dense index is preferable when —

- The dataset is small or manageable in size.
- Frequent point queries are expected.
- The indexed column has unique values.
- Data updates are frequent.
- sorted data cannot be guaranteed.

On the other hand, a sparse index is typically used for large, ~~A sparse~~ sorted datasets. ~~where~~ range queries dominate

* Multilevel indexing improves the efficiency of searching an index file in the following ways:

1. Reduces Search space :

Instead of scanning a single large index, the search begins at the topmost index level, which is much smaller.

2. Hierarchical structure :

The index is divided into multiple levels. Each level serves as a summary or pointer to the next, reducing the amount of data to be scanned at each stage.

3. Memory Optimization:

since higher levels of the index are smaller, they can fit into faster memory.

By organizing the index into multiple levels, the search process becomes faster and more efficient, specially for large datasets. Where a single level index would be too large and slow to process.

Lost update problem

The lost update problem occurs in multi-user environment when two or more transactions update the same data concurrently and one transaction's update overwrites the other without proper synchronization. This results in one update being "lost" because the database doesn't retain both changes.

T_1	T_2
$R(A)$	
$W(A)$	
	$W(A)$
	Commit
Commit	

question: Let $R(A, B, C, D)$. If AB, CD are superkeys, then in a relation find $\boxed{CK, SK, PK}$

* Determine candidate keys

The minimal set of attributes that can uniquely identify a tuple is known as a candidate key.

Problem statement:

Relation $R(A, B, C, D)$

- i. AB can uniquely identify a tuple. So, AB is candidate key.
 - ii. BD can uniquely identify a tuple. So, BD is candidate key.
- candidate keys are: AB, BD .

* Determine superkeys

• super key is a key or a group of multiple keys that can uniquely identify tuples in a table.

superkeys derived from AB :

• $AB, ABC, ABD, ABCD$.

superkeys derived from BD :

• $BD, ABD, BCD, ABCD$.

Thus, the unique superkeys are:

$AB, BD, ABC, ABD, BCD, ABCD$.

Determine Primary Key

A primary key is one of the candidate key chosen by the database designer to uniquely identify the tuple in the relation.

- Either AB or BD can be selected as the primary key.

Summary:

- Superkeys : 6
 $\{AB, BD, ABC, ABD, BCD, ABCD\}$
- Candidate keys : 2
 $\{AB, BD\}$
- Primary keys : 1 (either AB or BD)

* How many attributes you can use in a table? Is there any limitation? Why do you need to split the attributes into multiple tables?

= The number of attributes (columns) in a table depends on the database management system being used. For example:

- MySQL supports up to 4096 columns.
- PostgreSQL supports an unlimited columns, limited only by system memory.
- other ~~database~~ DBMSs have their own limitations.

Limitations:

- DBMS constrains the maximum row size or system limits on storage.
- Performance issues as too many attributes can slow down queries and make the database harder to maintain.

Reasons to split attributes into multiple tables:

- To normalize data and avoid redundancy.
- To ensure data integrity and consistency.
- To improve query performance by reducing the size of individual tables.
- To organize data logically based on relationship using primary & foreign keys.

* What are the functions of DDL & DML in database languages? How they differ from each other?

= i. Data Definition Language (DDL)

DDL is used to define the structure and schema of a database. It deals with the creation, alteration, and deletion of database objects like tables, views, indexes etc.

key function of DDL

• CREATE:

Creates new database objects.

Example: `CREATE TABLE students (id INT, Name VARCHAR (50));`

• ALTER:

Modifies the structure of an existing database object.

Example: `ALTER TABLE students ADD column age INT;`

• DROP: Deletes database objects permanently.

Example: `DROP TABLE students;`

ii. Data Manipulation Language (DML):

DML is used to

manipulate and retrieve data within a database. It deals with inserting, updating, deleting etc.

• Key functions of DML

~~Insert~~
• INSERT: Adds new records to a table.

• Example: INSERT INTO students (id, name, age) VALUES (1, 'John', 20);

• UPDATE: Modifies existing records in a table.

Example: UPDATE students SET
age = 21 WHERE id = 1;

• DELETE: Removes records from a table

Example: DELETE FROM students
WHERE id = 1;

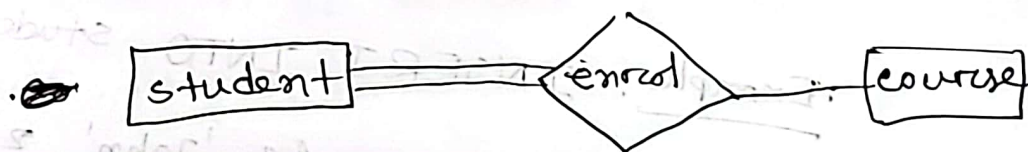
* DDL focuses on defining and managing database structure, while DML is used for manipulate data within that structure. Both play complementary roles in DBM, with DDL handling the "design" of the database and DML handling the "usage" of the data.

* Distinct: between total & partial participation

constraints.

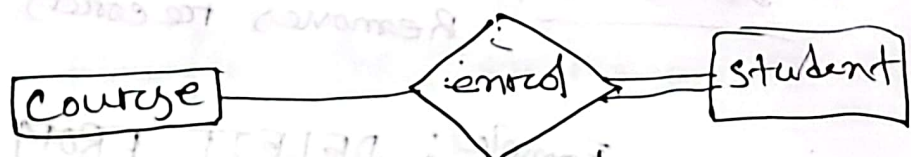
• Total participation:

Every entity in the entity set must participate in relationship. It denoted by double line between the entity set and the relationship in an E-R diagram.



• Partial participation:

Some entities in the entity set may or may not participate in the relationship. It is represented by single line between the entity set and the relationship in E-R diagram.



Total participation	Partial participation
definition	definition
2. Represented by double line in the E-R diagram.	2. Single line
3. Mandatory participation.	3. optional participation
4. The relationship is necessary for the entity set.	4. The relationship is not necessary for the entity set.

* What is Entity relationship Diagram? Explain the notational conventions used in ER diagram.

= An Entity Relationship Diagram (ERD) is a visual representation of data and the relationship between different entities in a system. It is primarily used in database design to understand and structure how data will be stored, managed and accessed. ER diagrams provide a clear framework for developers and analysts to visualize the logical structure of a database.

• Components of ER Diagrams:

- i. Entities.
- ii. Attributes.
- iii. Relationships
- iv. Participation

Notational Convention:

1. Entities.

• Strong entity: Represented by single rectangle.

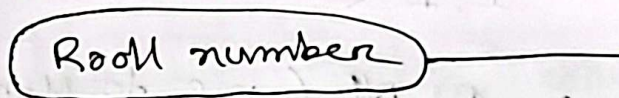
• Weak entity: Represented by

double rectangle.

2. Attributes

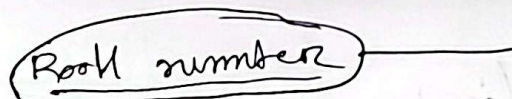
- Simple attribute:

Represented by an oval.



- Key attribute:

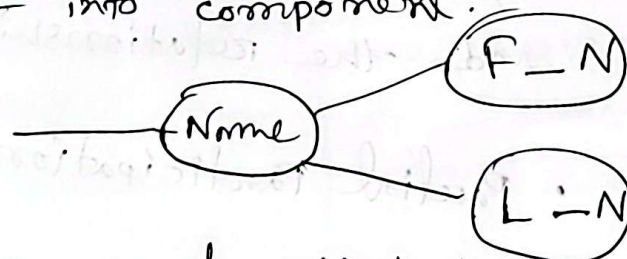
Represented by underlined in the oval.



- Composite attribute:

Represented by a main

oval with smaller sub-ovals breaking it into component.



- Derived attribute:

Represented by

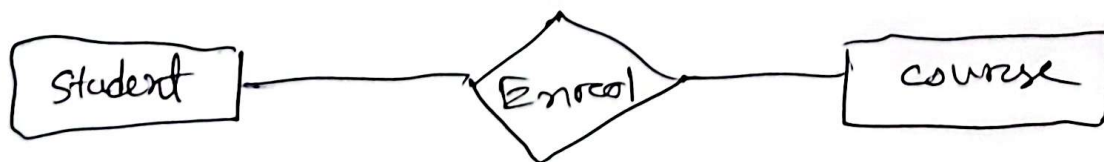
a dashed oval.



3. Relationship:

It represented by a diamond shape

with the relationship name.



4. Cardinality

- 1:1 (one to one)
- 1:M (one to Many)
- M:N (Many to Many)
-

5. Participation

- Total participation; Represented
by double line between the entity
and the relationship.
- Partial participation: Represented
by single line.

* Diff: between ER Diagram & Schema Diagram

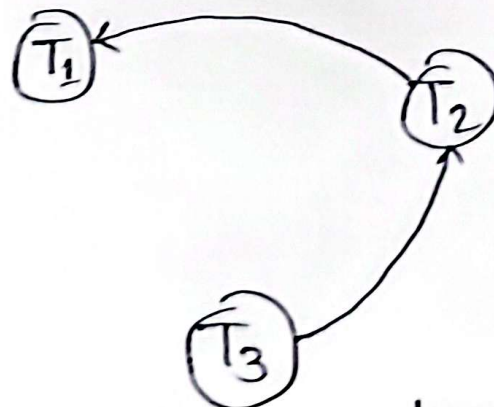
Aspect	ER Diagram	Schema Diagram
Purpose	Conceptual design	Implementation design
Representation	Entities, attributes, relations.	Tables, columns, keys
Abstraction	High level, abstract	Detailed, implementation-focused.
Phase	Early Design stage	Later Design stage
Definition	ER is a graphical representation of entities and their relationships in a database.	A Schema Diagram is a visual representation of the logical structure of a database.

* Check whether the schedule is conflict serializable or not?

S: $R_1(A); R_2(A); R_3(B); W_1(A); R_2(C); R_2(B); W_2(B); W_1(C)$.

⇒

T_1	T_2	T_3
*R(A)	—	—
—	*R(A)	—
—	—	*R(B)
*W(A)	—	—
—	*R(C)	—
—	*R(B)	—
—	*W(B)	—
*W(C)	—	—



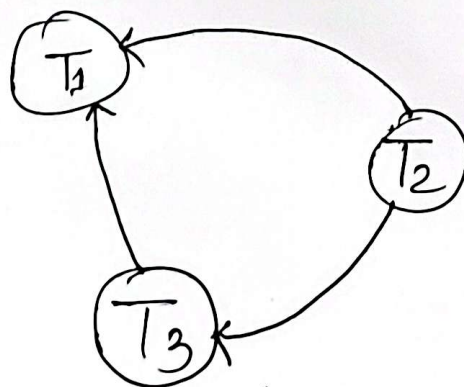
So, there is no cycle or loop. It's a conflict serializable

*

S : R2(A), W2(A), R3(c), W2(B), W3(A), W3(c)
 R1(A), R1(B), W1(A), W1(B)

Solⁿ:

T ₁	T ₂	T ₃
—	* R(A)	—
—	* W(A)	—
—	—	* R(c)
—	* W(B)	—
—	—	* W(A)
—	—	* W(c)
R(A)	—	—
R(B)	—	—
W(A)	—	—
W(B)	—	—



So, there is no cycle or loop, jts a
 serialize conflict.

* consider a relational schema R with attributes A, B, C, D, E, F and the FDs are

$$A \rightarrow BC$$

$$B \rightarrow E$$

$$CD \rightarrow EF$$

(i) find out candidate keys?

(ii) prove that ~~FD~~ functional dependency $AD \rightarrow F$ holds in R.

Solⁿ Given,

R (A, B, C, D, E, F).

known

$$A \rightarrow BC$$

$$B \rightarrow E$$

$$CD \rightarrow EF$$

$$AD^+ = ABCDEF \checkmark$$

$$AB^+ = ABCE \times$$

$$AC^+ = ABCE$$

{ e.g., AD candidate key }

ii) To prove $AD \rightarrow F$, find the closure of AD , and check if it includes F .

$$\therefore AD^+ = AD$$

Find $(AD)^+$: $AD^+ = AD$

using $A \rightarrow BC$; $(AD)^+ = ABCD$

using $B \rightarrow E$; $(AD)^+ = ABCDE$

using $CD \rightarrow EF$; $(AD)^+ = ABCDEF$

so, $(AD)^+$ includes F . so $AD \rightarrow F$ hold.

* $R(A, B, C, D, E)$ shown below, which of the following functional dependencies (FD's) hold in the relation R ? Briefly justify answer.

- i) $AB \rightarrow C$
 ii) $B \rightarrow D$
 iii) $DE \rightarrow A$

A	B	C	D	E
1	2	3	4	5
1	4	3	4	5
1	2	4	4	1

Sol:

Functional dependencies:

i) $AB \rightarrow C$

$(1, 2) \rightarrow 3$
 $(1, 4) \rightarrow 3$
 $(1, 2) \rightarrow 4$

$\{AB \rightarrow C \text{ does not hold.}\}$

Because same input is giving different output.

ii) $B \rightarrow D$

$2 \rightarrow 3$

$4 \rightarrow 3$

$2 \rightarrow 4$

$\{B \rightarrow D \text{ holds}\}$

iii) $DE \rightarrow A$

$(4, 5) \rightarrow 1$

$(4, 5) \rightarrow 1$

$(4, 1) \rightarrow 1$

$\{DE \rightarrow A \text{ holds}\}$

* List all non-trivial functional dependencies satisfied by following relation.

A	B	C	D
a ₁	b ₁	c ₁	d ₁
a ₁	b ₂	c ₁	d ₂
a ₂	b ₂	c ₂	d ₂
a ₂	b ₂	c ₂	d ₃
a ₃	b ₃	c ₂	d ₄

1. check single attributes as determinants

• $A \rightarrow B$:

$A = a_1$, B values are b_1, b_2 (not unique)
so, doesn't hold.

• $A \rightarrow C$:

$A = a_1$, C values are c_1 (unique).
so it's hold.

• $A \rightarrow D$

$A = a_1$, D values are d_1, d_2 (not unique)
so, doesn't hold.

• $B \rightarrow A$

$B = b_2$, A values a_1, a_2 (not unique)
so, doesn't hold.

$$B \rightarrow C$$

$B = b_2$, C values are c_1, c_2 (not unique)

Doesn't hold.

$$B \rightarrow D$$

$B = b_2$; D values are d_2, d_3 . (not unique)

Doesn't hold.

$$C \rightarrow A$$

$C = c_1$, A values are a_2 (unique)

So hold.

$$C \rightarrow B \quad \times$$

$$C \rightarrow D \quad \times$$

$$D \rightarrow A \quad \times$$

2. checks for combine attributes

$$AB \rightarrow C \quad \checkmark$$

$$AB \rightarrow D \quad \checkmark$$

$$AC \rightarrow D \quad \times$$

$$BC \rightarrow A \quad \checkmark$$

$$CD \rightarrow A \quad \checkmark$$

Final list of Non-Trivial

1. $A \rightarrow C$

2. $C \rightarrow A$

3. $AB \rightarrow C$

4. $AB \rightarrow D$

5. $BC \rightarrow A$

6. $CD \rightarrow A$