Unit - I

What is data?

Data in DBMS are raw facts from which the required information is produced. So, data are distinct piece of information.

eg. Text, image, audio, video. These are the form of data.

What is information?

Information in DBMS is processed, organized, or summarized data. It may be defined as a collection of related data that, when put together, becomes a useful messages to a receipient.

eg. Let say a mobile number belongs to someone.

Diff. b/w Data & Information

Data	Information
• Data is the raw facts	Processed form of data
• Not significant.	Significant to a business.
• Atomic level piece of information	collection of data
• does not help in decision making.	The information helps in decision making.
eg. 23 is a data	eg. Age 23 is an example of information

DataBase

A database is a collection of inter-related data which helps in the efficient retrieval, insertion, and deletion of data from the database and organizes the data in the form of tables, views, schemas, reports, etc.

Types

- TDB - Traditional DataBase - Text, Numbers
- MDB - Multimedia DataBase - Video, Audio
- GIS - Geographical Image System - Images
- RDB - Real time DataBase -
- Data Warehouse - collection of large data

Data Base Management System

The Software which is used to manage databases is called DataBase Management System (DBMS).

For eg - Mysql, Oracle etc.

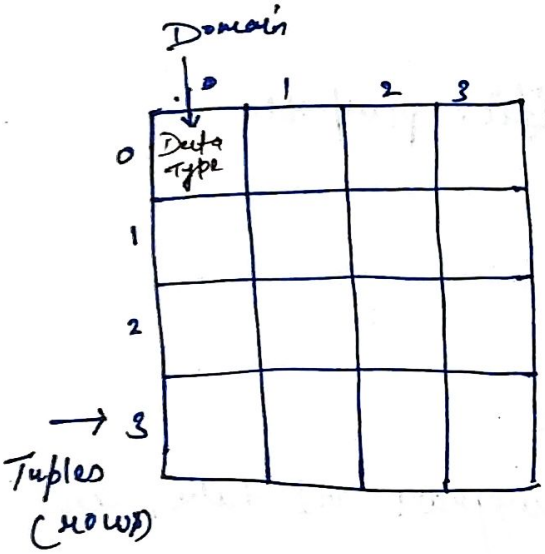
DBMS allows users the following tasks

- Data definition
- Data Updation
- Data Retrival
- User Administration.

Function of DBMS

1. Define
2. Construct
3. Manipulation
4. Sharing
5. Database protection

1. Define



- Datatype must be define before storing the data in database.

Domains: Varchar
Varchar2
Number
Data

and two rows have same data.

- Imp. things that must be define within every DB
  - Datatypes,
  - Structure
  - Constraints

2. Construct:

Process of storing data into some storage device

Manipulation

The operations that are performs on database

- Insert / update / delete ....

Properties of database

- Always database represents some real world (mini world)
- Database is a logical coherent (relation) of real-world data.
- Data Base can be work for specific purpose

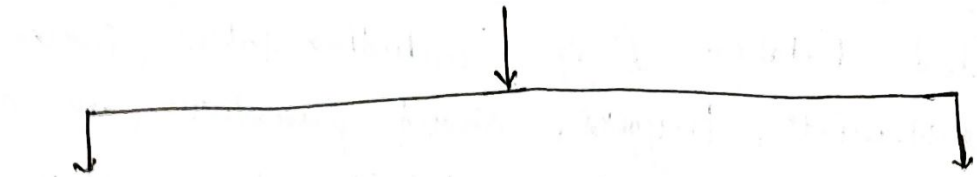
Characteristics of DataBase

- self-described nature of database
- Insulation / encapsulation between program and data, and data abstraction (program data independent)
- multiple views of data
- Sharing of data to multiple users and support multiple transaction
  concurrency control

Advantages of DataBase

- Reduce data redundancy
- Integrity : Accurate and consistent
- Security
- Speed

DataBase Uses



Actor in Scene
- DB Administrator
- DB designer
- End User
- System analyst

Actor in Backend
- Tools developer
- System designer
- Operation and maintenance person

- **DB Administrator**

DA (Data Administrator) + DBA (Database Administrator)

- making strategies and policies
- Implement the policies and strategies

Work of DBA

- Define the conceptual schemas
- Defining internal schemas
- Talk with users
- Defining security & integrity
- Monitoring the performance (Time & Space) & (indexing)

DB designer

The database designer is responsible for defining the detailed database design, including tables, indexes, views, constraints, triggers, stored procedures, and other database-specific constructs needed to store, retrieve, and delete persistent objects.

* End Users

Those people whose jobs require access to the database for querying, updating, and generating reports.

- Casual End users

  These are the users who occasionally access the database but they require different information each time.

- Naive or parametric end users

  These are the users who basically make up a sizeable portion of database end-users. The main job function revolves basically around constantly querying and updating the database for this we basically use a standard type of query known as the canned transaction that has been programmed and tested.

- Sophisticated end users

  These users basically include engineers, scientists, business analytics, and others who thoroughly familiarize themselves with the facilities of the DBMS in order to implement their application to meet their complex requirements

- Standalone users -

These are those users whose job is basically to maintain personal databases by using a ready-made program package that provides easy-to-use menu-based or graphics-based interfaces.

System Analyst

System analysts determine the requirements of the database users (especially naive users) to create a solution for their business need and focus on non-technical and technical aspects.

Bugs identification.

Tool developer

design and implement tools — the software packages that facilitate database modeling and design, database system design, and improved performance.

System designers

design and implement the DBMS modules and interfaces as a software package.

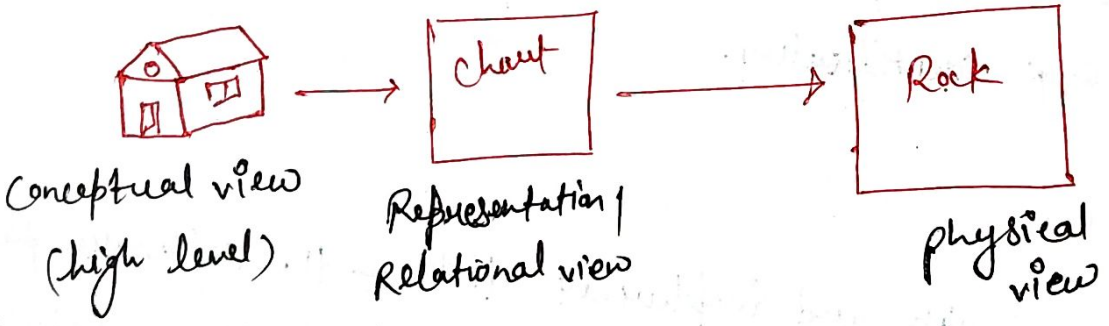
Operators and maintenance personnel (System administration personnel)

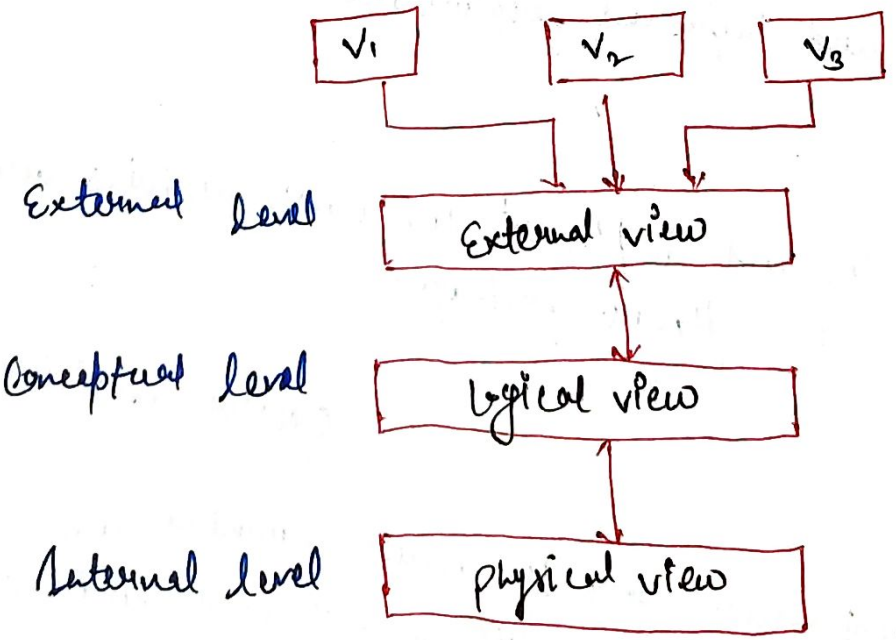
responsible for the actual running and maintenance of the hardware and software environment for the database system.

Data Base Model

The primary goal of using data model are:

- Ensures that all data objects required by the database are accurately represented. Omission of data will lead to creation of faulty reports and produce incorrect results.
- Data model structure helps to define the relational tables, primary and foreign keys and stored procedures.





- **Conceptual Data model**

This data model defines WHAT the system contains. This model is typically created by business stakeholders and Data Architects. The purpose is to organize, scope and define business concepts and rules.

- **Logical Data Model:**

Defines HOW the system should be implemented regardless of the DBMS. This model is typically created by Data Architects and business analysts. The purpose is to developed technical map of rules and data structures

- **physical Data Model**

This Data model describes HOW the system will be implemented using a specific DBMS system. This model is typically created by DBA and developers. The purpose is actual implementation of the database.

Data Base Languages

Read, Update, manipulate, and store data in a database using Database languages.

- Data definition Language
- Data Manipulation Language
- Data Control Language
- Transaction Control Language

- **Data Definition Language**

  The language is used to create database, tables, alter them etc. With this, we can also rename the database, or drop them. It specifies the database schemas.

  CREATE | ALTER | DROP | RENAME

- **Data Manipulation Language**

  The language used to manipulate the database like inserting data, updating data (Table), retrieving record from a table etc.

  SELECT | INSERT | UPDATE | DELETE

- **Data Control Language**

  Grant privilege to a user using the GRANT statement. In the same way, revoke the privilege using the REVOKE statement.

  GRANT | REVOKE

- **Transaction Control Language**

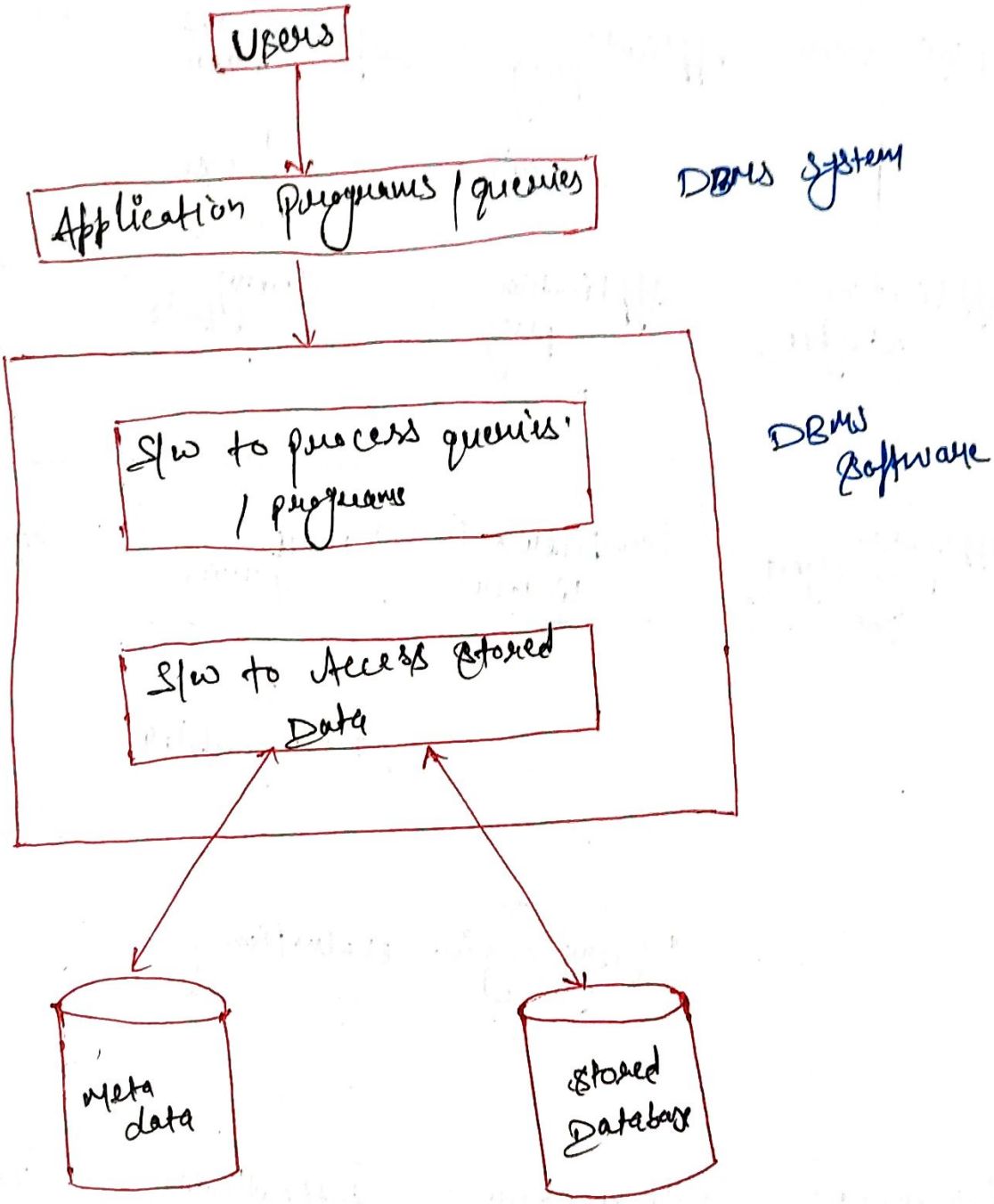
  Manage transactions in the Database

  COMMIT : Save the work

  SAVEPOINT : set a point in transaction to rollback later

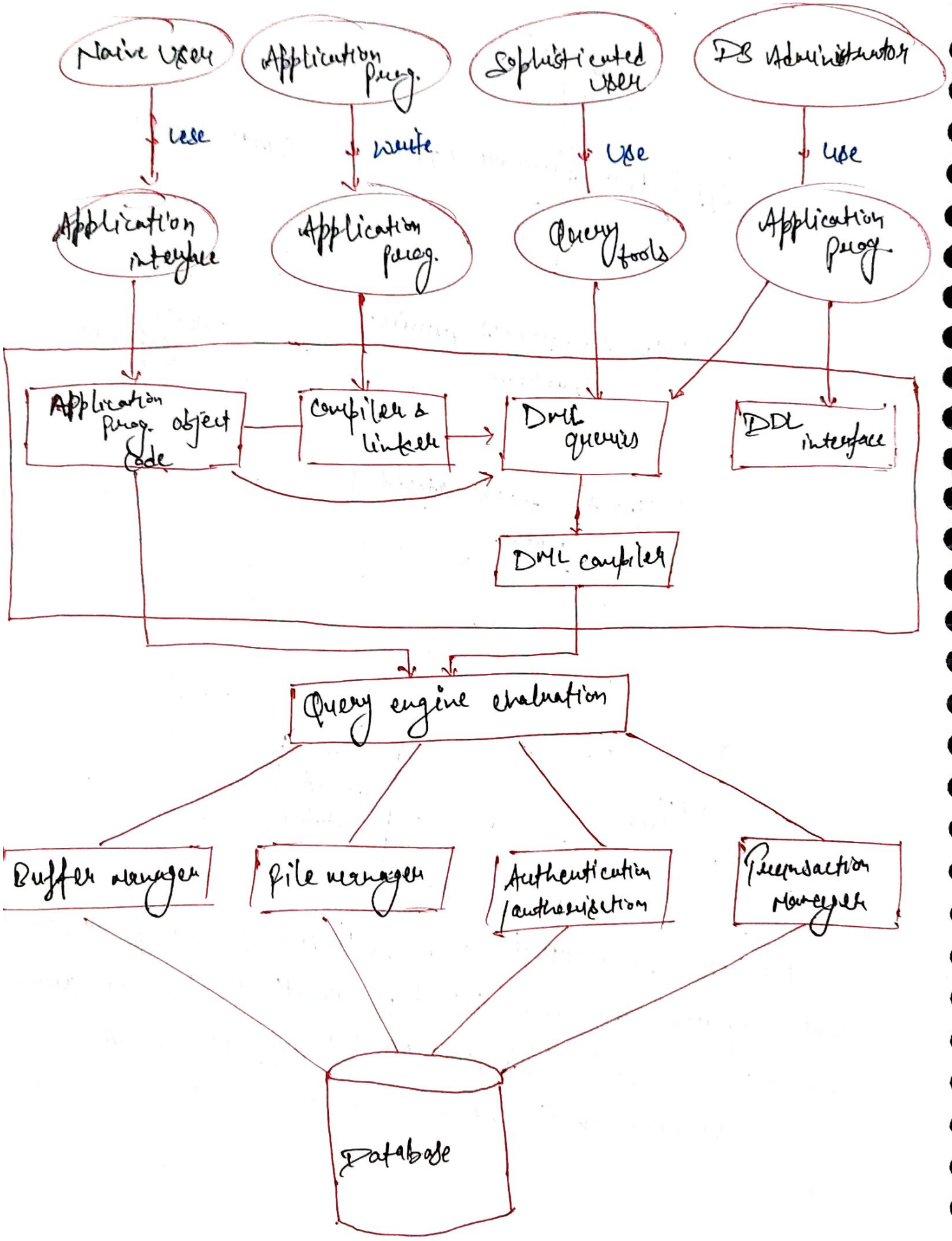
  ROLLBACK : Restores since last commit

DataBase Environment



A database environment is a collective system of components that comprise and regulates the group of data, management, and use of data, which consists of software, hardware, people, techniques of handling database, and the data also.

Overall Structure of DB



E-R diagram

ER diagram stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In Other words, ER diagrams help to explain the logical structure of databases. In other words ER diagrams are created based on three basic concepts: entities, attributes and relationships

E-R Model :- ER model is high-level conceptual data model diagram. ER models helps to systematically analyze data requirements to produce a well-designed database. The ER model represents real-world entities and the relationships between them.

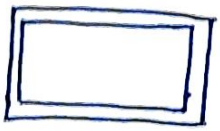
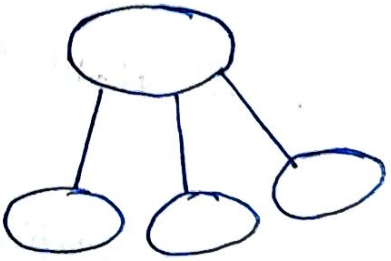
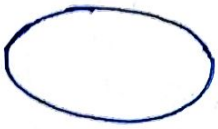
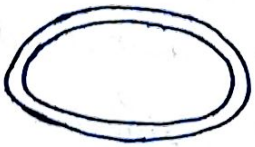
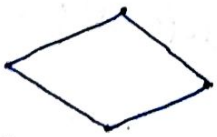
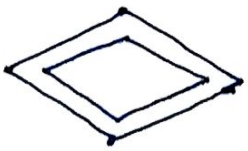
Why ER ?

Here, are prime reasons for using the ER Diagram

- Helps to define terms related to entity relationship modeling.
- Provides a preview of how all your tables should connect, what fields are going to be on each table
- Helps to describe entities, attributes, relationships
- ER diagrams are translatable into relational tables which allows you to build database quickly

- ER diagrams can be used by database designers as a blueprint for implementing data in specific software applications.

ER Diagrams Symbols & Notations

- entity or strong entity 
- Weak entity 
- Composite attribute 
- Attribute 
- Multivalued attributes 
- Relationship 
- Weak Relationship 
- Relationship b/w (entity and relations) 
- Total participation 
- Derived attributes 
- key 

Components of ER Diagram

This model is based on three basic concepts

- Entities
- Attributes
- Relationships

Entity: A real-world thing either living or non-living that is easily recognizable and non-recognizable. It is anything in the enterprise that is to be represented in our database.

Examples: Person, Place, object, event, concept.

Entity set: An entity set is a group of similar kind of entities. It may contain entities with attribute sharing similar values.

There are two type of entity set

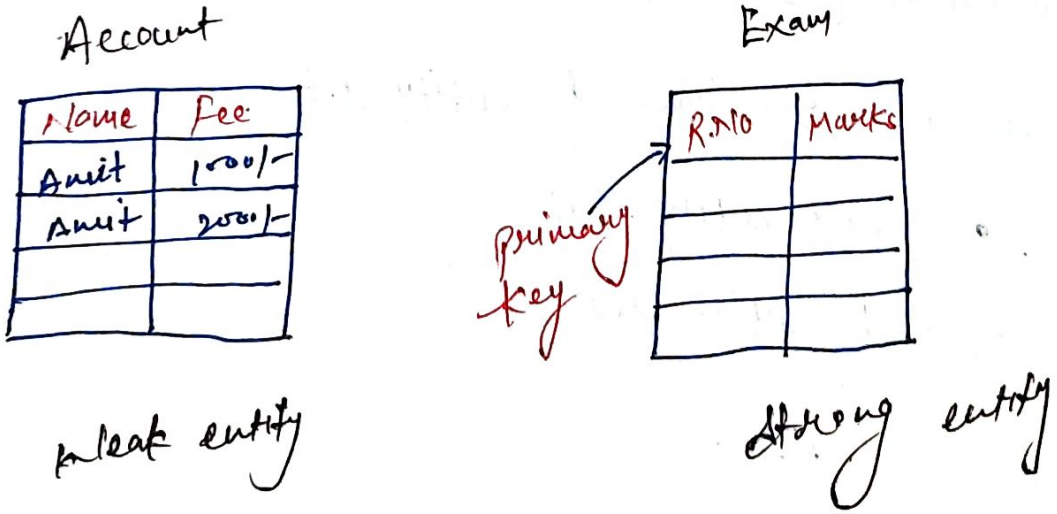
- Strong entity set
- Weak entity set

- **Strong entity:**

1. It has primary key
2. It doesn't depend on any other entity

3. It can be represented using a single rectangle
4. The relationship between two strong entities can be represented using a single diamond.
5. It can either have total participation or no participation.

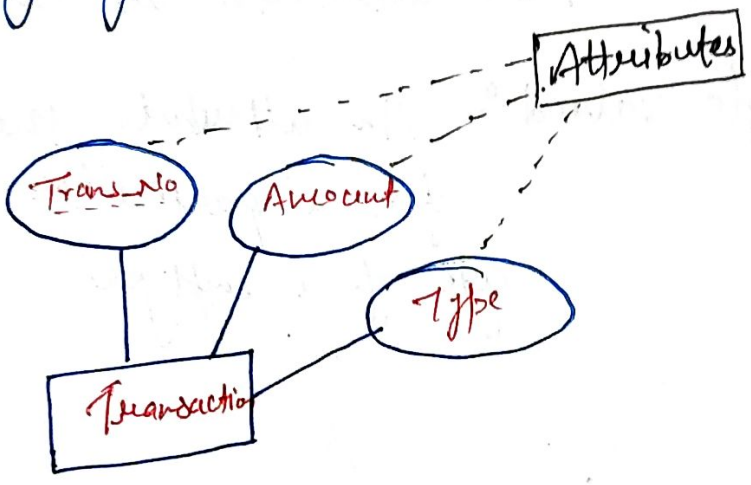
- Weak Entity:
  1. A weak entity has a partial discriminator key.
  2. It depends on the strong entity.
  3. It can be represented using a double rectangle.
  4. The relationship between a strong and a weak entity can be represented using a double diamond.
  5. They always have total participation.



- **Relationship:** nothing but an association among two or more entities.

- **Attributes:** It is a single-valued property of either an entity-type are relationship-type.

example:



Types:

1. Simple attribute: Simple attributes can't be divide any further. For eg, a student's contact No. It is also called an atomic value.

2. Composite attribute: It is possible to break down composite attribute. For example, a student's full name may be further divided into first name, second name, and last name.

3. Derived attribute: This type of attribute does not include in the physical database. However, their values are derid from other attributes present in the database. For eg, age should not be stored directly. Instead, it should be derived from the DOB.

4. Multivalued attribute: Multivalued attributes can have more than one values. For example, a student can have more than one mobile number, email address, etc.

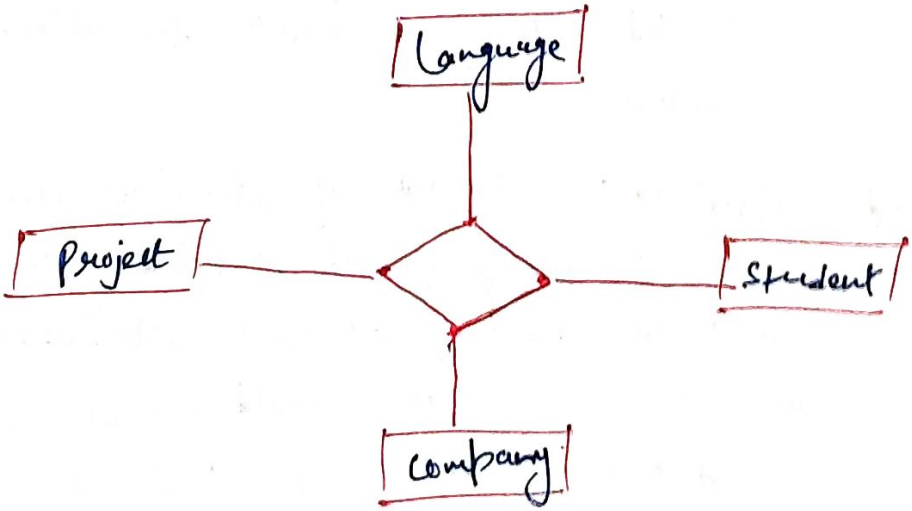
5. Complex attributes: If an attribute of an entity, is built using composite and multivalued attributes, then these attributes are called complex attributes.

6. Single Valued: An attribute that has a single value for a particular entity. For example, age of a employee.

- Relationships (Types)

Binary: Two entities.

Ternary: more than 2



Cardinality

Defines the numerical attributes of the relationship between two entities or entity sets.
(maximum participation)

Four types of cardinal relationships are:

1. One-to-one:

one entity from entity set X can be associated with at most one entity set Y and vice versa.

eg:- cardinality ratio = 1:1

2. One-to-many:

one entity from entity set X can be associated with multiple entities of entity set Y, but an entity from entity set Y can be associated with at least one entity.

eg:- cardinality ratio = 1:n

3. Many-to-one:

More than one entity from entity set X can be associated with at most one entity of entity set Y. However, an entity from entity set Y may or may not be associated with more than one entity from entity set X.

eg:- Cardinality ratio = m:1

4. Many-to-many:

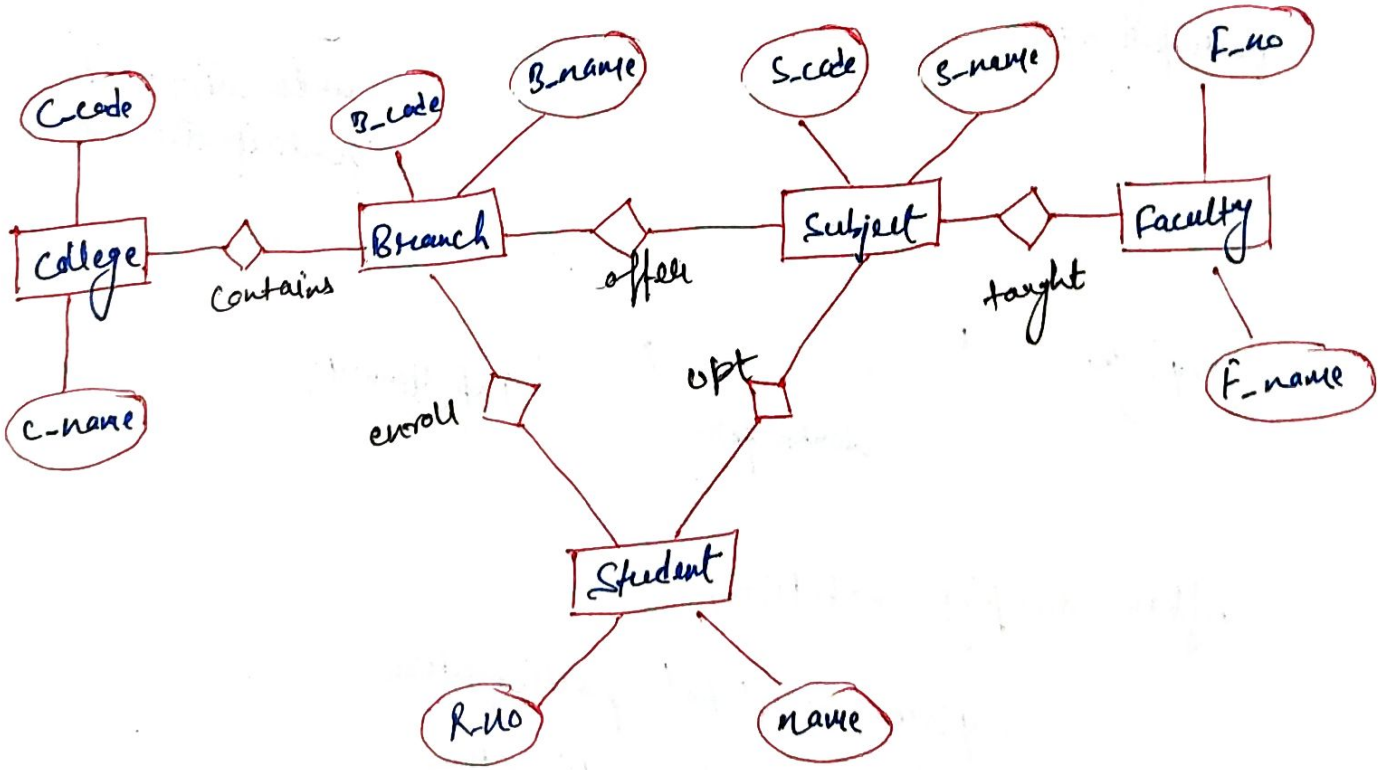
One entity from X can be associated with more than one entity from Y and vice versa

eg:- Cardinality ratio = m:n

Ques: Construct ER diagram from a college with many branches, offer many subjects, also enroll many students in every branch. Every student opt various subjects and every subject taught by a faculty.

Sol^n : Entities with their attributes

College : C_code, C_name
Student : R_no., name, ~~branch~~
Branch : B_code, B_name
subject : S_code, S_name
Faculty : F_no., F_name

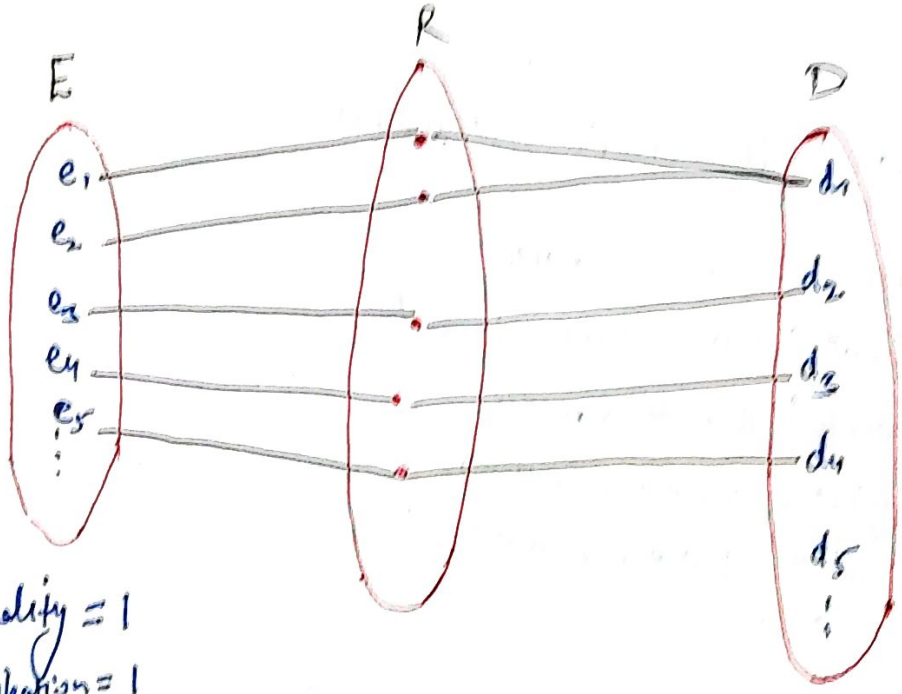


Ques: 1. every employee works for a department

2. A department have many employee

3. New department does not have any employee

Soln



degree = 2

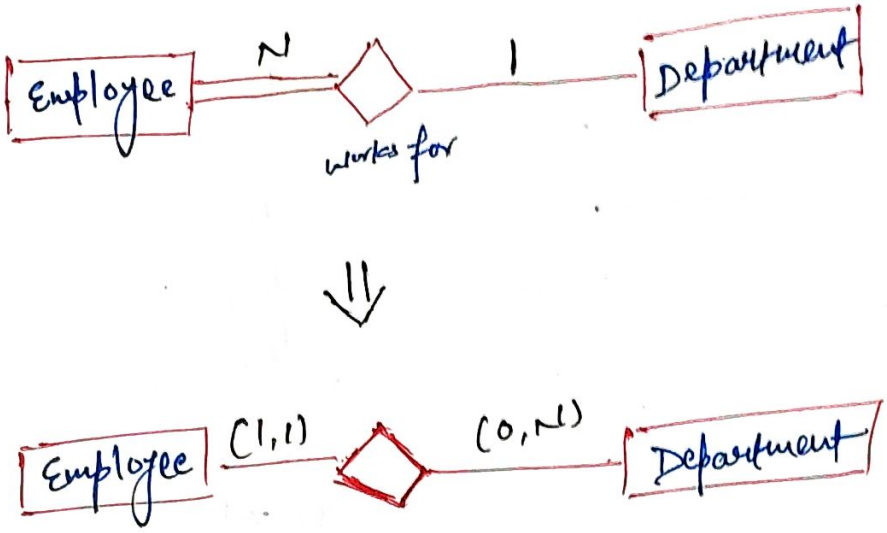
cardinality = 1
participation = 1

cardinality = N
participation = 0

After swaping notations,

Employee = Total participation

Dept = Partial participation

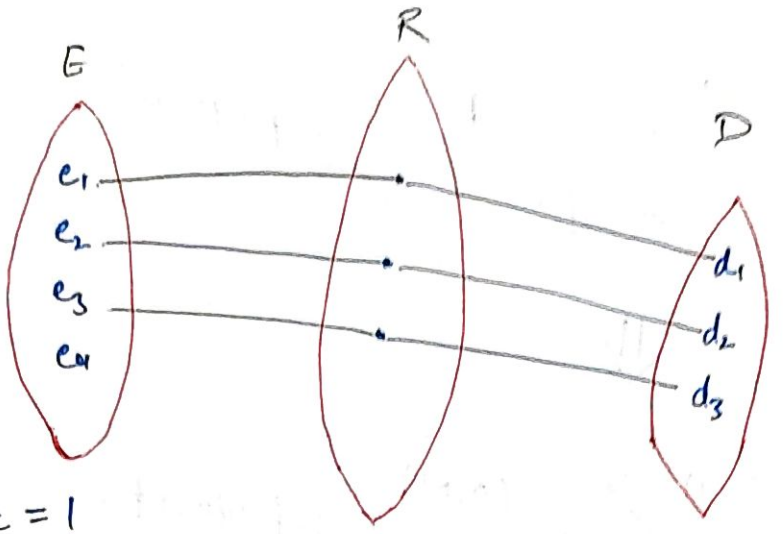


Some important terms

- Degree : No. of entities participated in relation.
- Cardinality : How many relationships, a particular entity involved. (Maximum participation)
- Participation : Minimum participation
- All entities have independent relationship.

Ques : Every department have a manager and only one employee manages a department.

Soln:

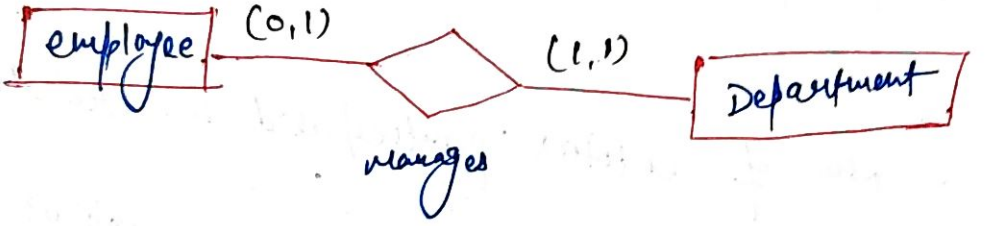


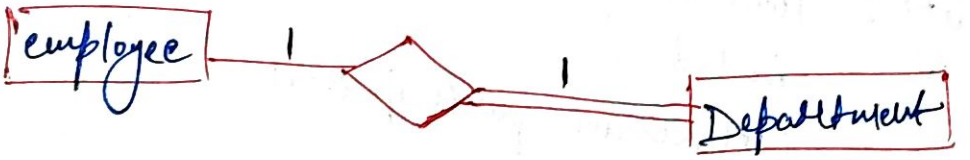
$E_C = 1$

$E_P = 0$

$D_C = 1$

$D_P = 1$





Question:

Every employee is suppose to work atleast one project and he/she can work on many projects. A project can have many employees and every project must have atleast one employee.

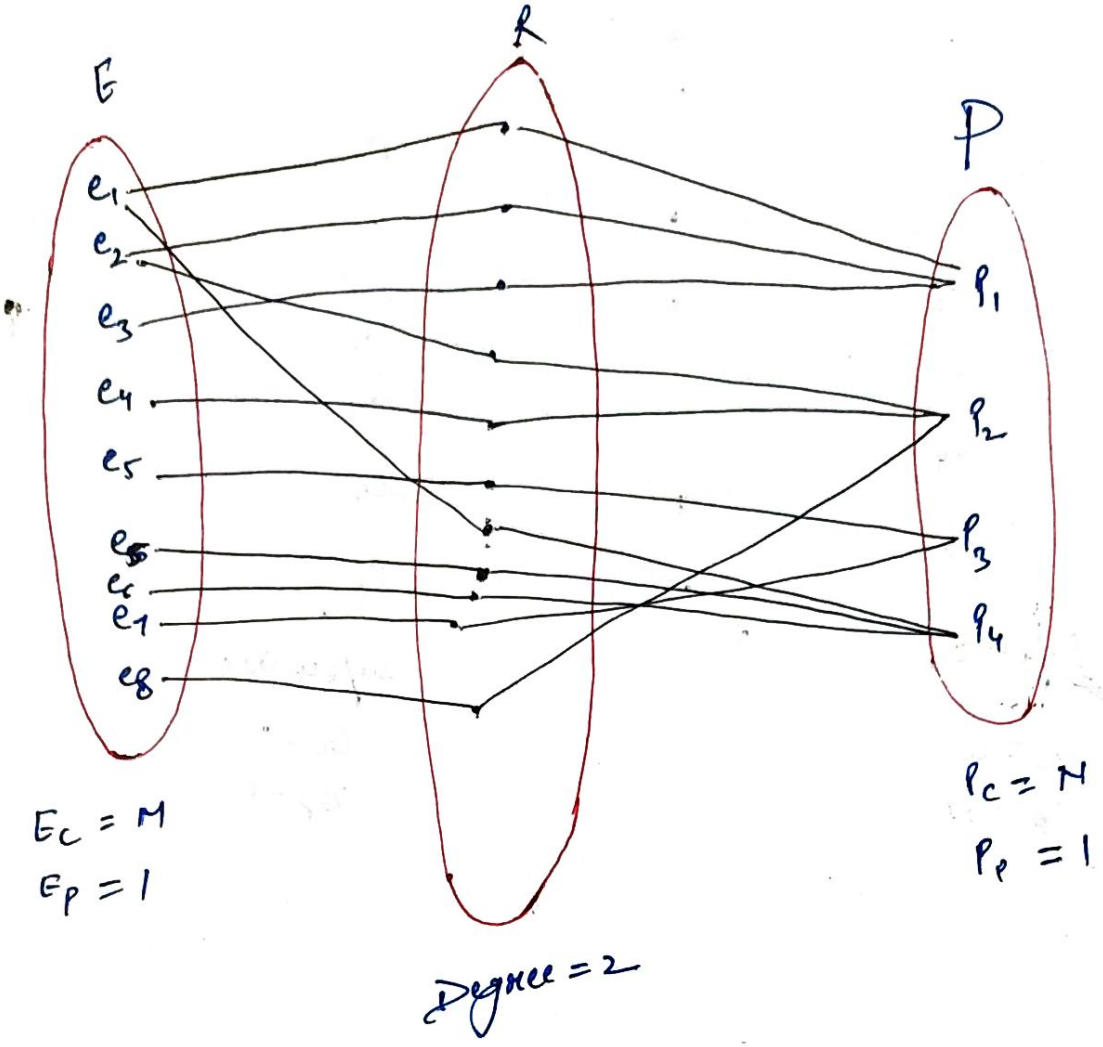
Soln

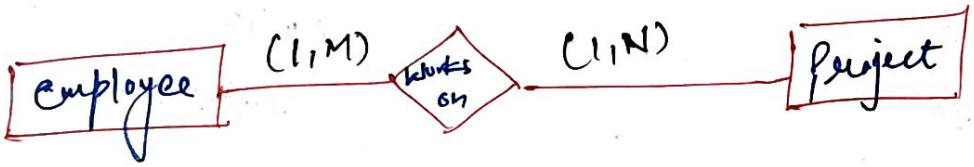
P_1 : e_1, e_2, e_3

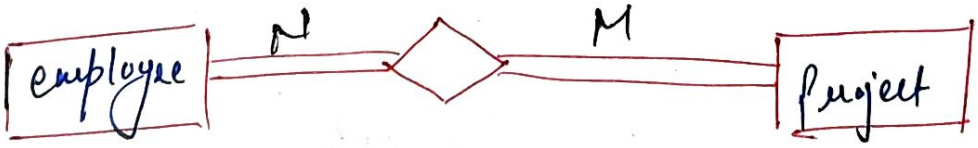
P_2 : e_2, e_4, e_8

P_3 : e_5, e_7

P_4 : e_1, e_5, e_6



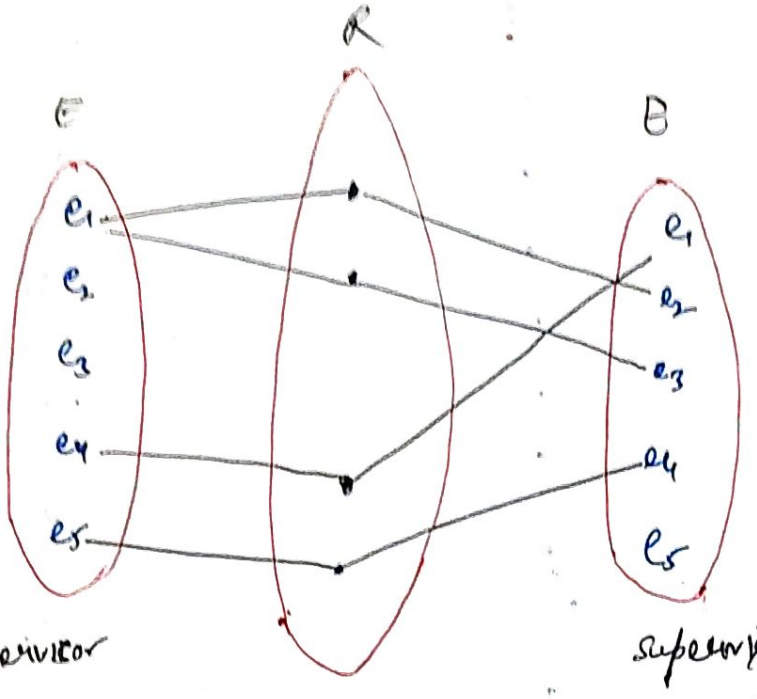




Ques: Recursive ER Diagram

1. some supervisor or some supervisee
2. some not supervisor or some ~~not~~ supervisee

Ans)



Supervisor

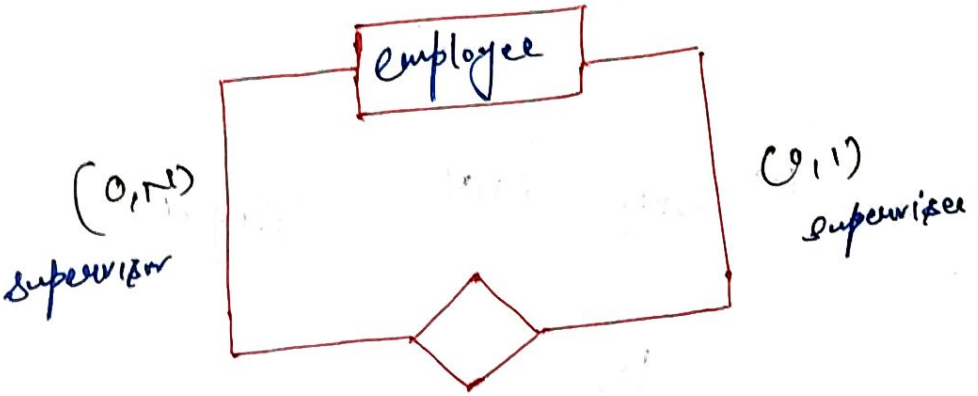
$E_C = 0$

$E_P = N$

Supervisee

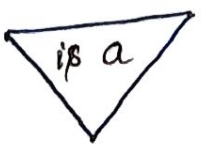
$E_C = 0$

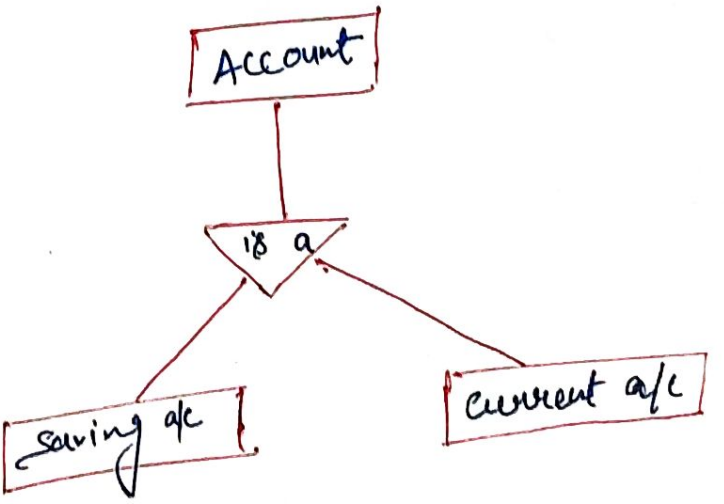
$E_P = 1$



Generalisation relationship

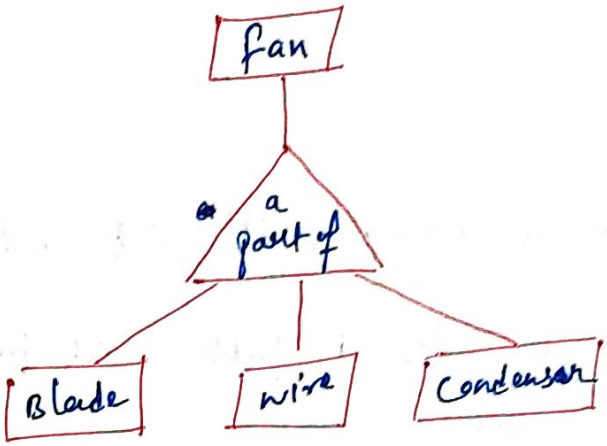
Symbol :-



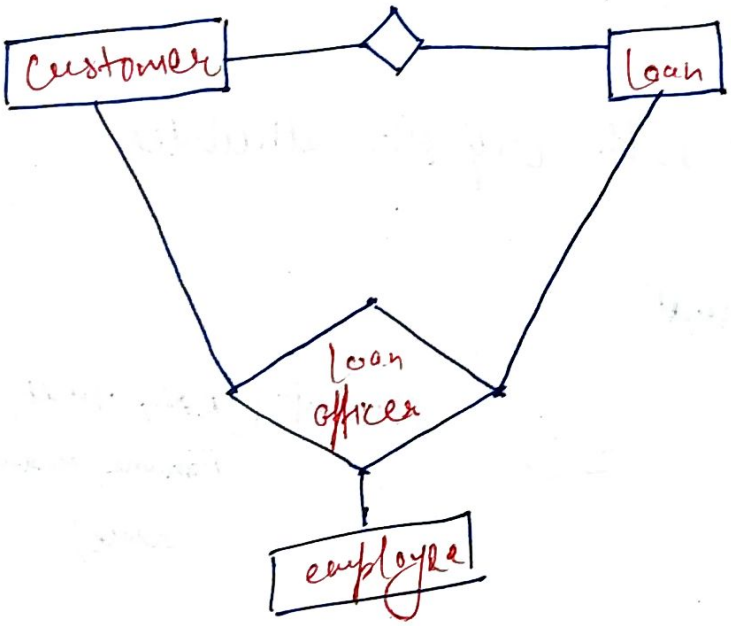


Aggregation relationship

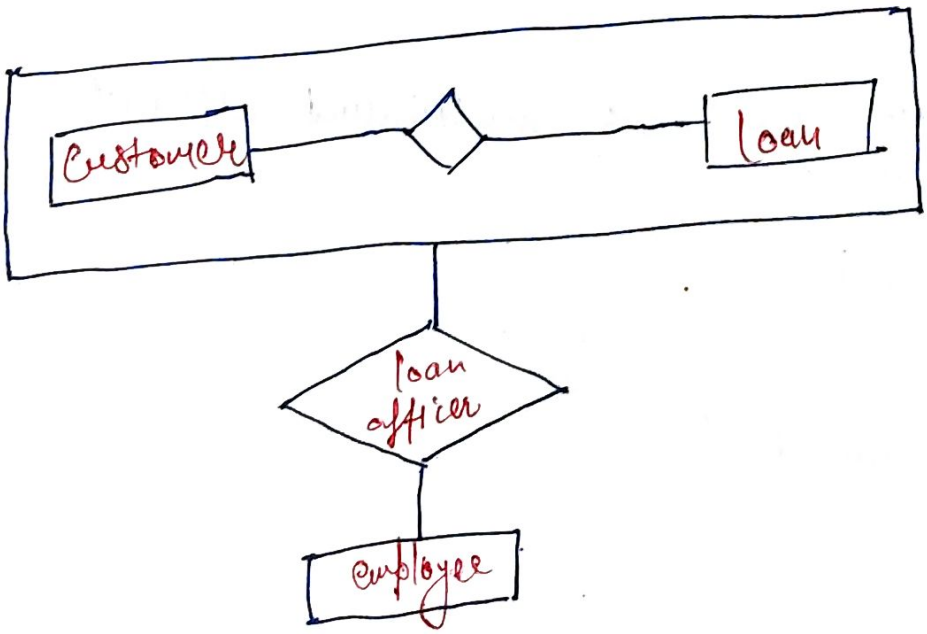
Symbol :



⇒ If there is independent relationship



Normal



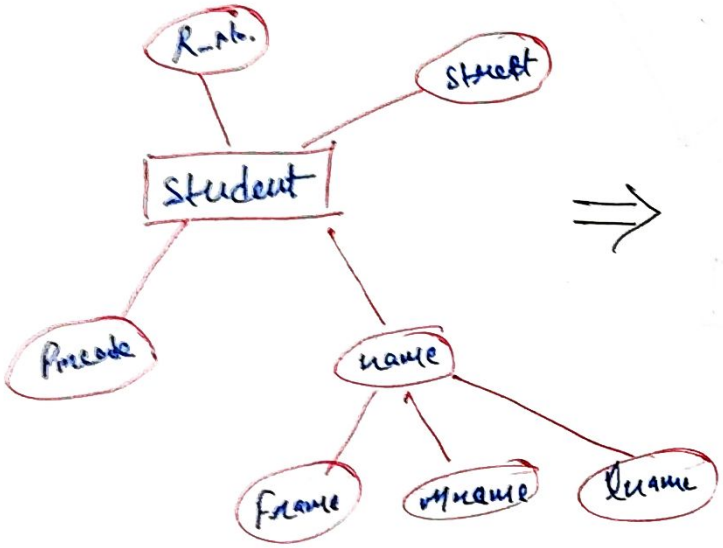
Aggregation

ER Diagram into relational Table

Rules:

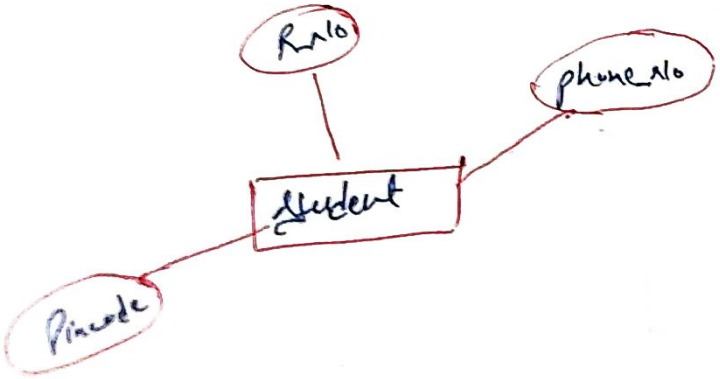
1. For a strong entity set with simple attribute.
   a. Attributes of table will be the attributes of entity set.
   b. Primary key of the attribute of entity set is treated as key of table.

2. Strong entity set with composite attributes



⇒ Student (R_no, street, Pincode, Fname, mname, lname).

3. Strong entity set with multivalued attribute



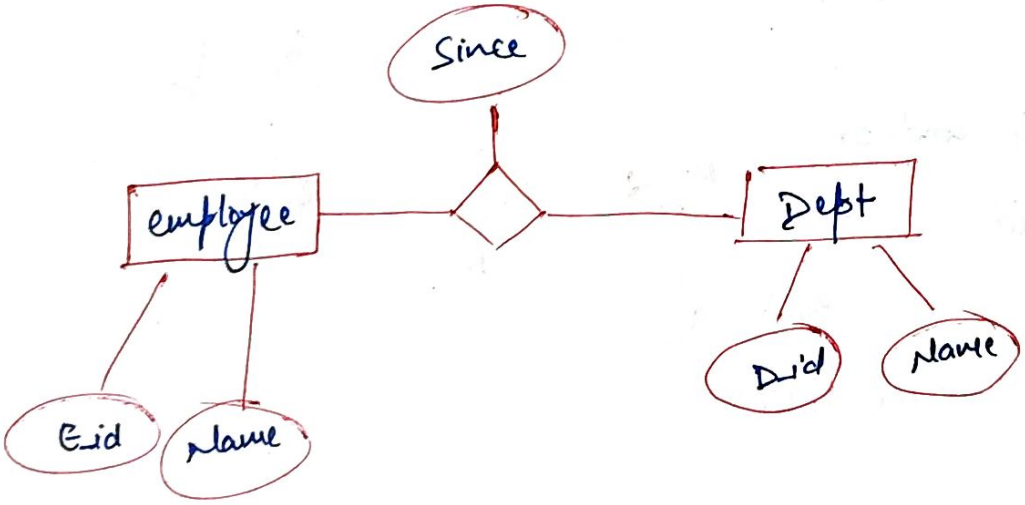
It contains two table

- One table will contain all simple attributes with key value.
- One table will be key value with multi-valued attributes

↓
Primary key

Need of two table is because of data redudancy.

4. Translating relationship into table

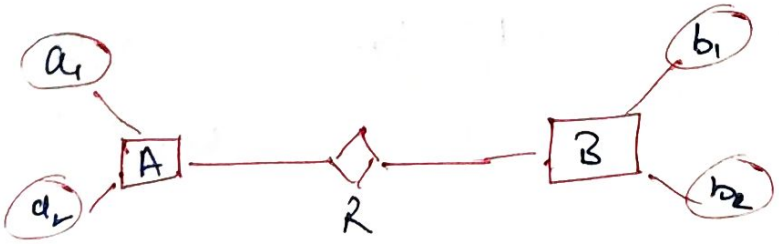


The "Since" attribute will be in Employee table.

5. Binary relationship with cardinality ratio

Case I: ratio : m:n

$A(a_1, a_2)$
$B(b_1, b_2)$
$R(a_1, b_1)$



Case II Ratio = 1 : n

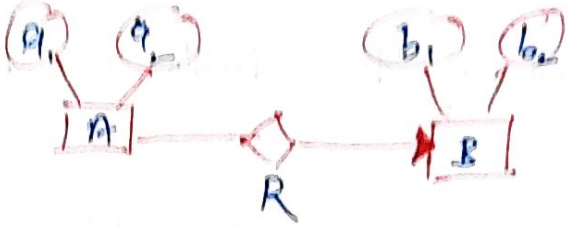
A(a1, a2)

BR(b1, b2, a1)

Case III : Ratio = m : n

AR(a1, a2, b1)

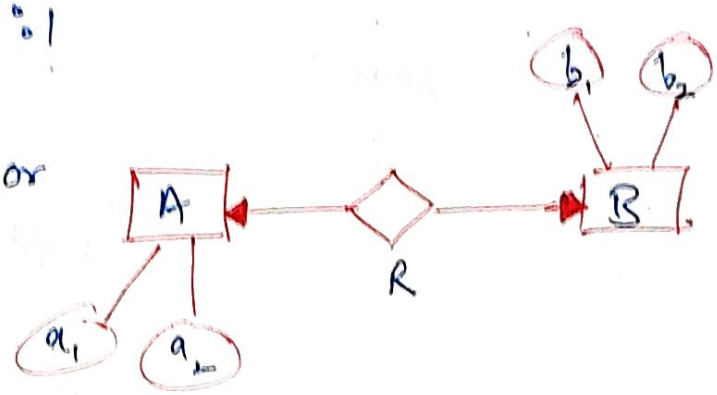
B(b1, b2)



Case IV Ratio = 1 : 1

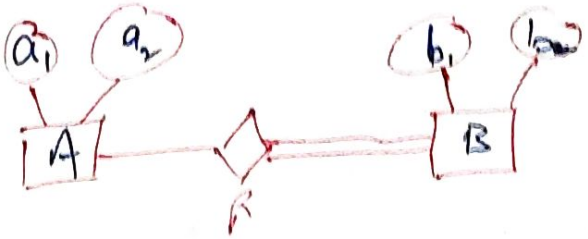
Both case II and or case III valid.

either case II or case III



6. Binary relationship with cardinality constraints and participation.

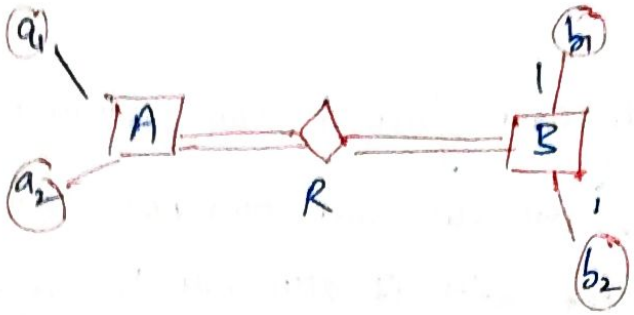
Case I: Binary relationship with cardinality constraint and total participation in one-side



A(a1, a2)

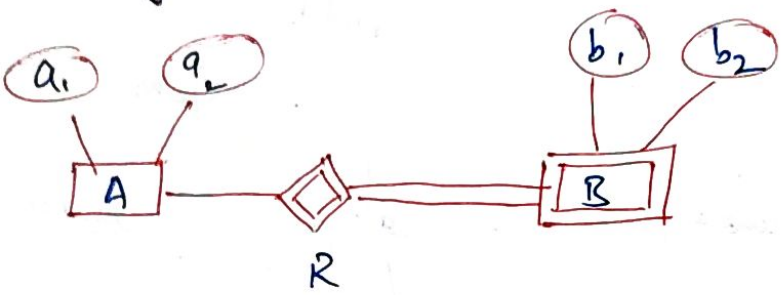
BR(a, b1, b2)

Case II Both side total participation



ABR (a_1, a_2, b_1, b_2)

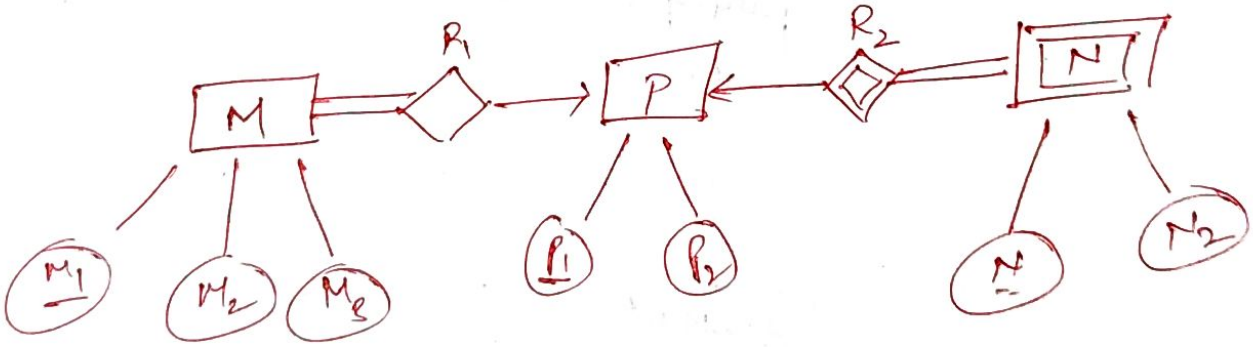
★ 4. Binary relationship with weak entity set



A (a_1, a_2)

BR (b_1, b_2, a_1)

Ques: How many tables are required (minimum) ?



Sol^n Minimum 3 tables are required

P (P_1, P_2)

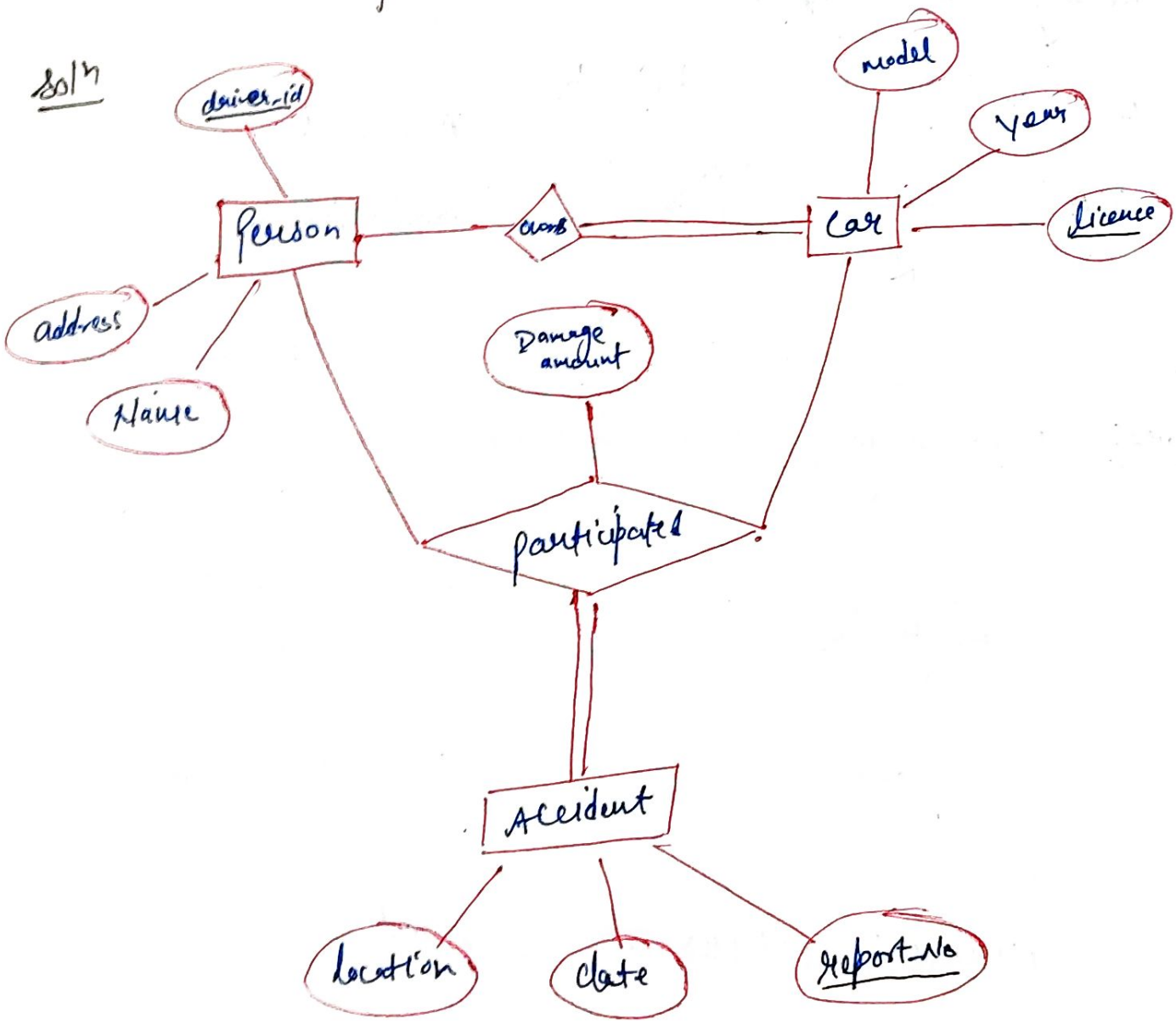
MR_1 (M_1, M_2, M_3, P_1)

NR_2 (N_1, N_2, P_1)

Problems based on ER Diagram

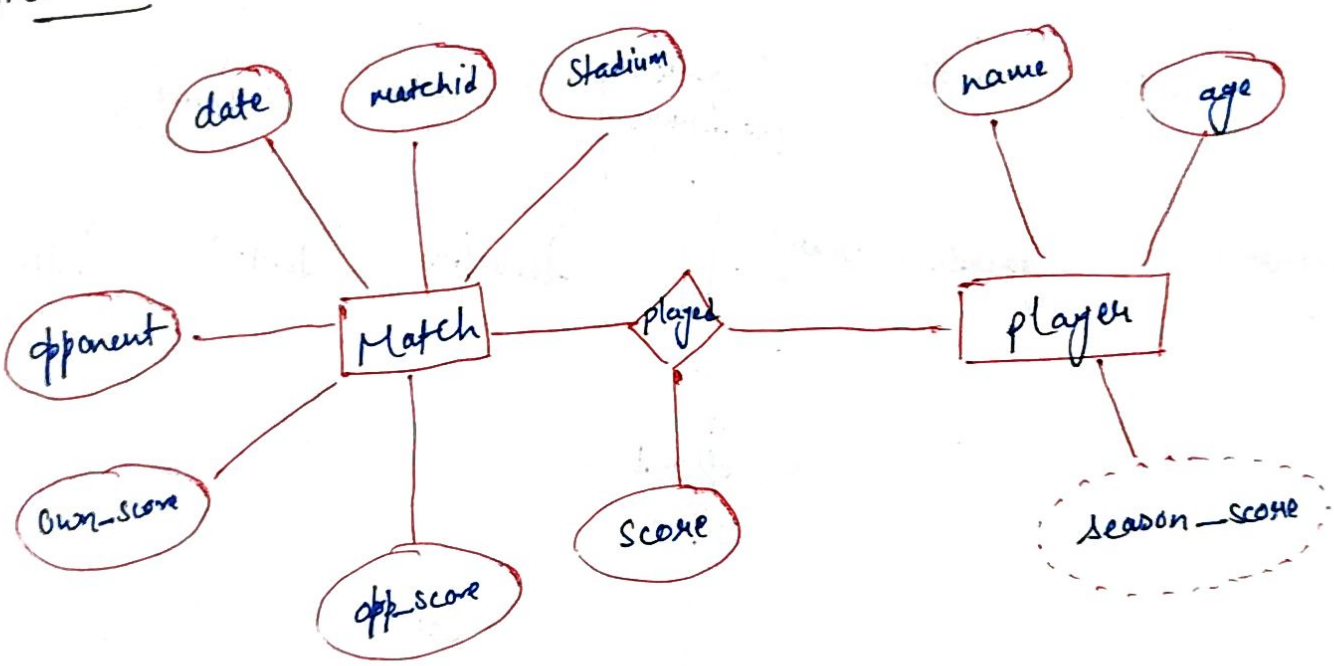
Ques 1: Construct an ER diagram for a car-insurance company whose customers own one or more car each. Each car has associated with it zero ~~or~~ to any number of recorded accidents.

Soln



Ques 2 : Design an E-R diagram for keeping track of the exploits of your favorite sports team. You should store the matches played, the scores in each match, the players in each match and individual player statistics for each match. Summary statistics should be modeled as derived attributes.

Answer :



Ques 3 : A University registrar's office maintains data about the following entities:

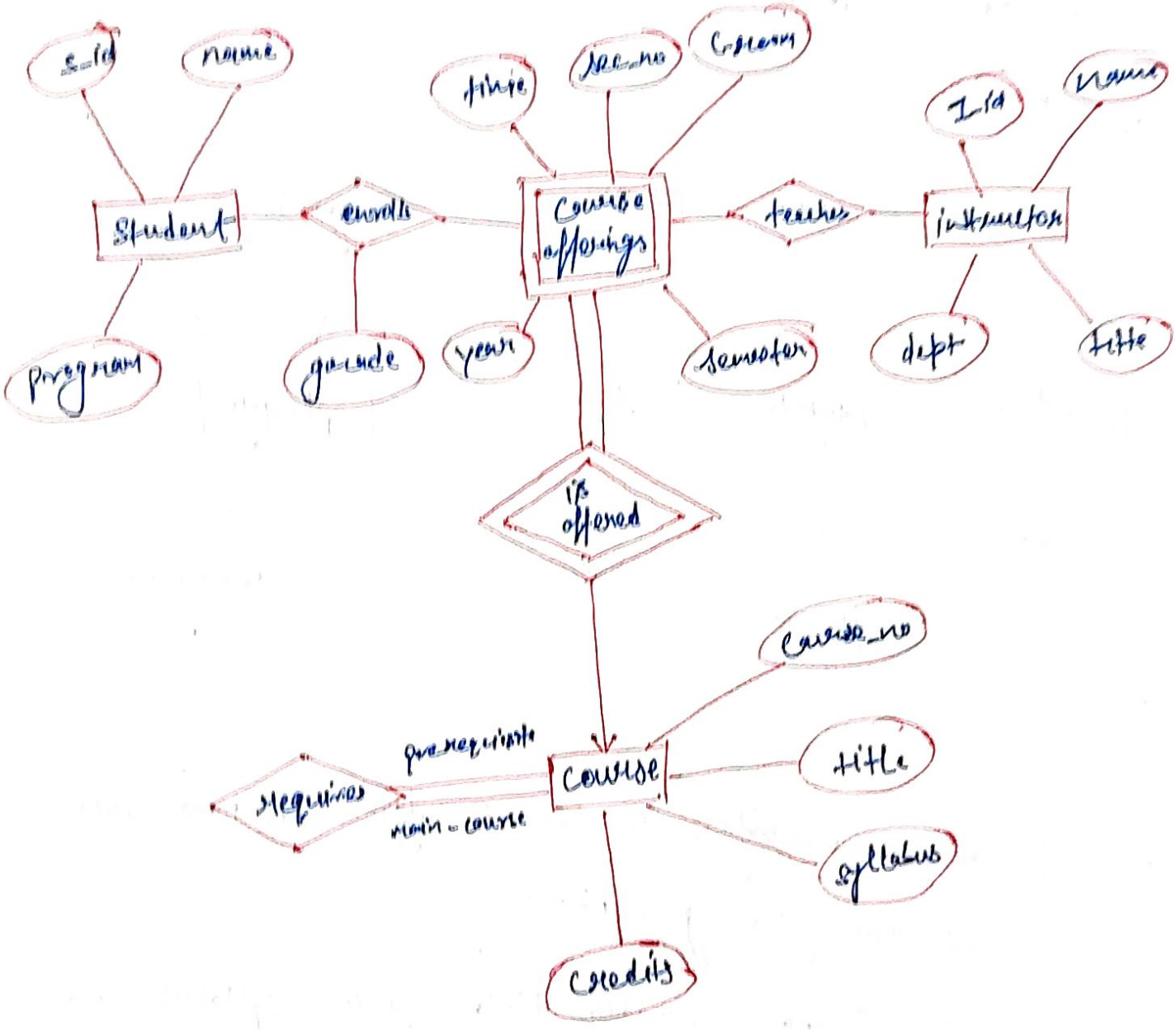
1. courses, including number, title, credits, syllabus, and prerequisites.
2. Course offerings, including course number, year, semester, section number, instructor(s), timings, and classrooms.

2. students, including student_id, name, program,

4. instructors, including Id_no, name, department, and title.

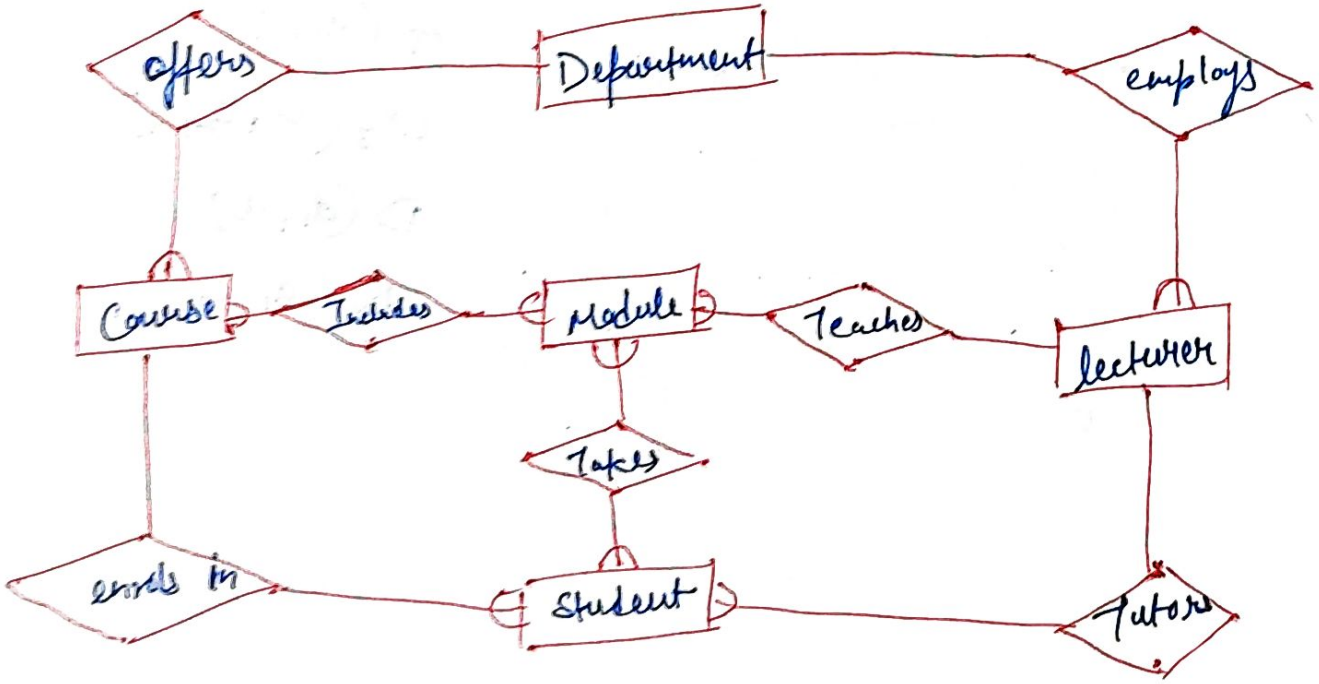
Construct ER diagram for registrars office. Document all assumptions that you make about the mapping constraints.

Answer:-

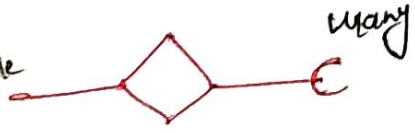


Ques 4: A University consists of number of department. Each dept offers several courses. A no. of modules make up each course. Students enrol in a particular course and take modules towards the completion of that course. Each module is taught by a lecturer from the appropriate department, and each lecturer tutors a group of students. Construct an E-R diagram for mentioned scenario, take necessary assumption such that cardinality, participation and mapping constraints.

Ans:

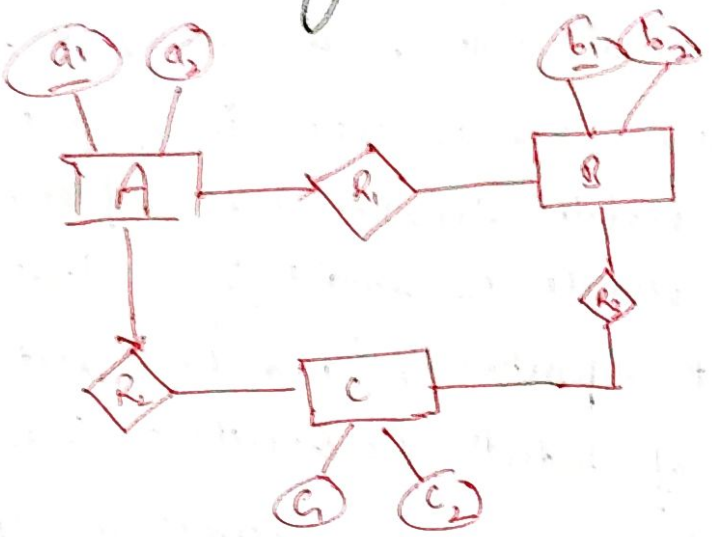


Here,



Ques: How many possible Tables are required?

(1)



Minimum possible Tables are

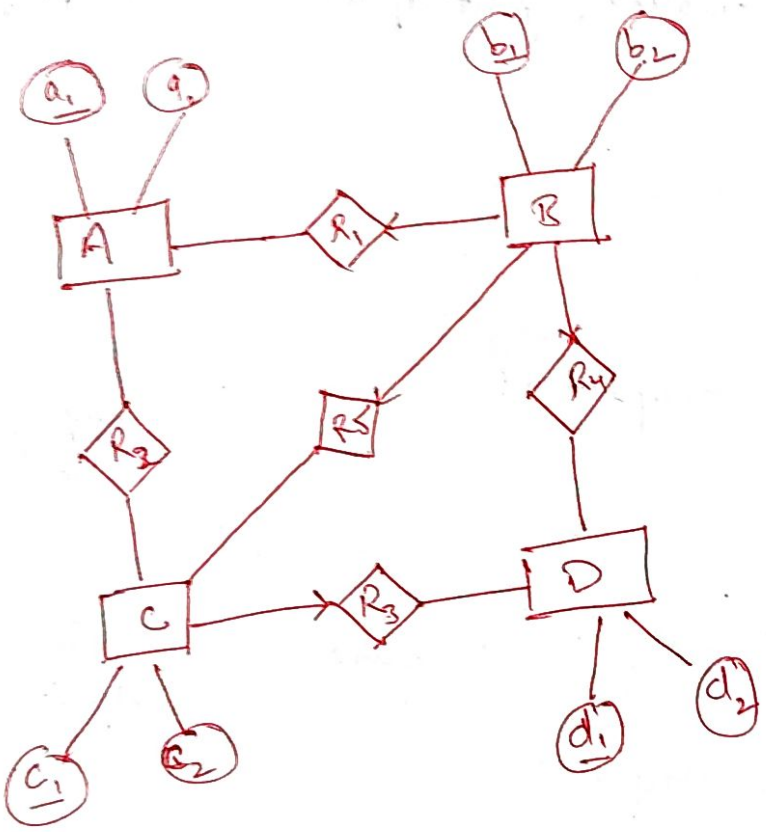
$AR_1R_2(a_1, a_2, b_1, c_1)$

$B(b_1, b_2)$

$C(c_1, c_2)$

$R_3(b_1, c_1)$

(2)



$BR_1R_4R_5(b_1, b_2, a_1, c_1, d_1)$

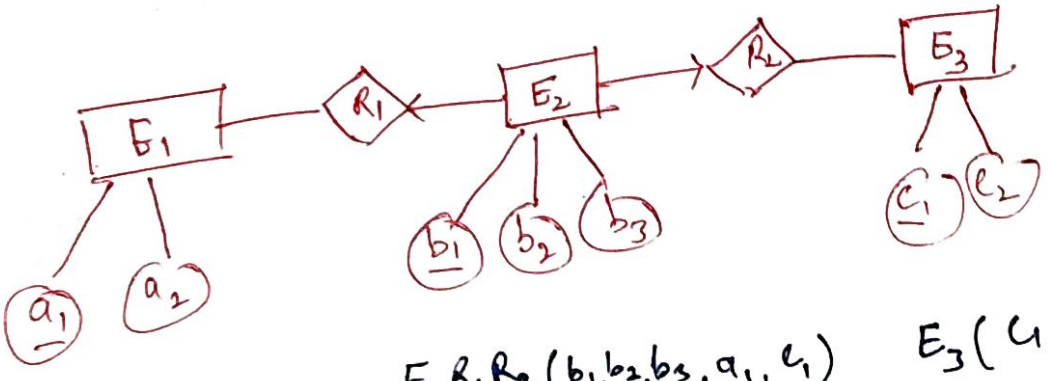
$A(a_1, a_2)$

$CR_3(c_1, c_2, d_1)$

$D(d_1, d_2)$

$R_2(a_1, c_1)$

(3)

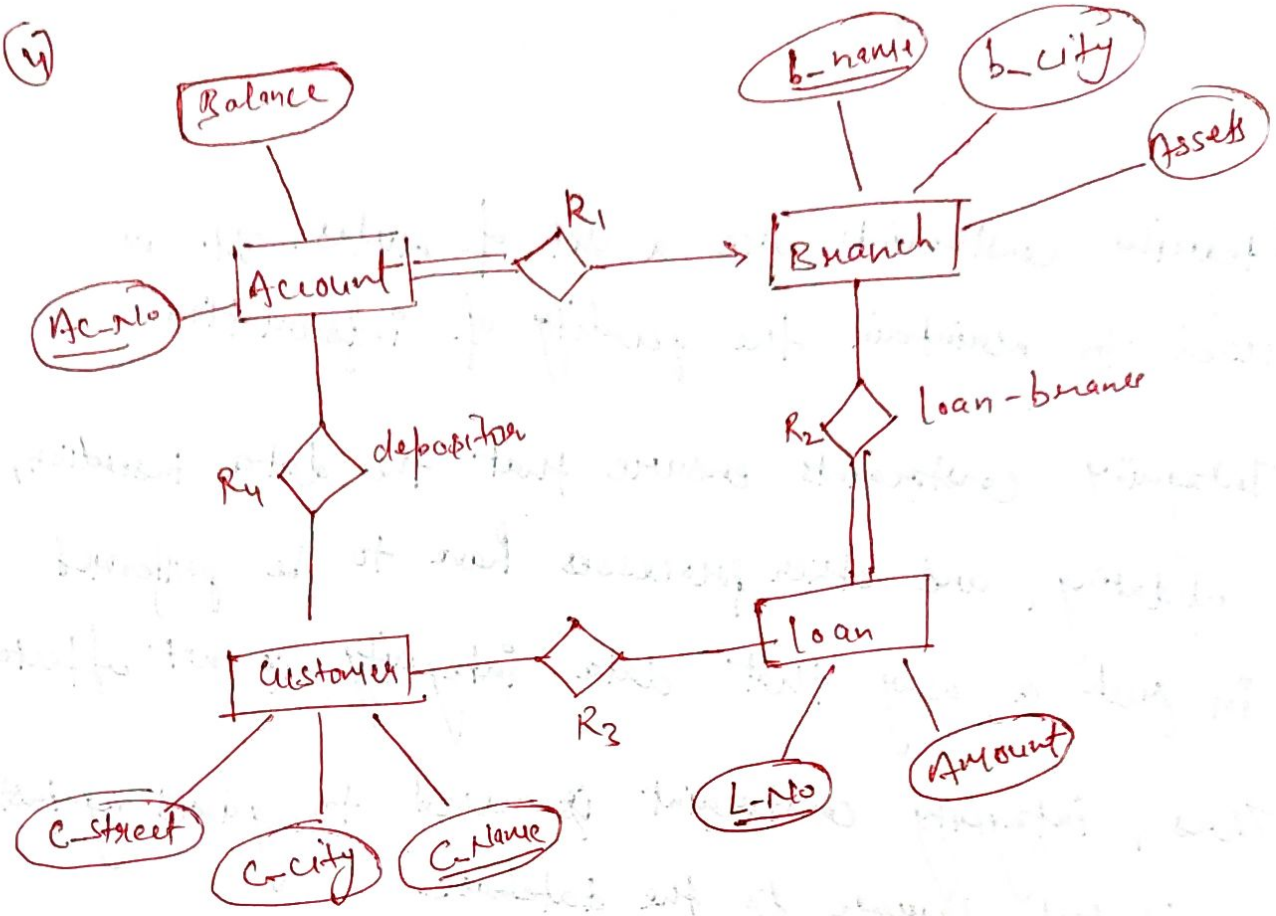


$E_1(a_1, a_2)$

$E_2R_1R_2(b_1, b_2, b_3, a_1, c_1)$

$E_3(c_1, c_2)$

(4)



Account R_1 (Ac_No, Balance, b_name)

Branch (b_name, b_city, Assets)

loan R_2 (L_No, Amount, b_name)

R_3 (C_name, L_No.)

Customer (cname, c_city, c_street)

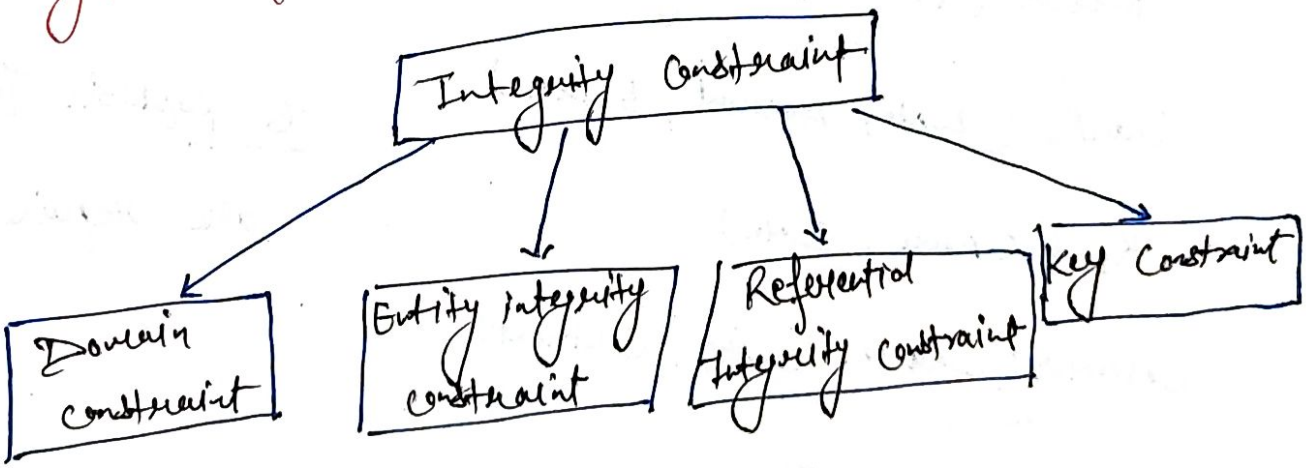
R_4 (Ac_no, cname)

6 possible Tables are required.

DataBase Integrity Constraints

- Integrity constraints are a set of rules. It is used to maintain the quality of information.
- Integrity constraints ensure that the data insertion, updating, and other processes have to be performed in such a way that data integrity is not affected.
- Thus, integrity constraint is used to guard against accidental damage to the database.

Types of constraints



1. Domain Constraints :

- All the value should be atomic (single values)
- All the composite attribute should be converted into the single attributes.

- Any multivalued are not allowed to store in the form of domain constraint.
- We can change the values of domain but not the structure if it is finalized.

ID	Sem	Age
1000	1st	17
1001	2nd	18
1002	3rd	19
1003	4th	20
1004	5th	A

← Not allowed. Because Age is integer.

2. Entity Constraints :-

- The entity constraint states that primary key value can't be NULL.
- This is because the primary key value is used to identify individual rows in relation and if the primary key has a null value, then we can't identify those rows.
- A table can contain a null value other than the primary key field.

ID	Sem	Age
1000	1st	17
1001	2nd	18
1002	3rd	19
	4th	20

Not allowed in PK

3. Referential Constraints :-

- A referential integrity constraint is specified between two tables.
- In the Referential integrity constraints, if a foreign in Table 1 refers to the primary key of Table 2, then every value of the foreign key in Table 1 must be null or be available in Table 2.

D_No	D_locatio
11	Mumbai
24	Delhi
13	Noida

primary key

Ename	Name	Age	D_No
1	Jack	20	11
2	Henry	40	24
3	John	27	18
4	David	38	13

Not allowed.

foreign key

4. Key Constraints :

- Keys are the entity set that is used to identify an entity within its entity set uniquely
- An entity set can have multiple keys, but out of which one key will be the primary key. A primary key can contain a unique and null value in the relational table.

ID	Name	Sem	Age
1000	Tom	1st	17
1001	Johnson	2nd	24
1002	Leonardo	5th	21
1003	Kate	3rd	19
1002	Morgan	8th	22

Not allowed.

Action of constraints:

Insert :- violates all integrity constraints

Delete :- violates referential constraints

update :-

- Insert operation:

on inserting the tuples in the relation, it may cause violation of the constraints in the following way:

1. Domain constraint:

Domain constraint gets violated only when a given value to the attribute does not appear in the corresponding domain or in case it is not of the appropriate datatype.

2. Entity Integrity Constraint:

On inserting NULL values to any part of the primary key of a new tuple in the relation can cause violation of the entity integrity constraint.

3. Key constraints:

On inserting a value in the new tuple of a relation which is already existing in another tuple of the same relation, can cause violation of key Constraints

4. Referential integrity:

On inserting a value in the foreign key of relation 1, for which there is no corresponding value in the primary key which is referred to in relation 2, in such case Referential integrity is violated

- Delete Operation:

It may cause only violation of referential integrity constraints.

Referential Integrity Constraints:

It causes violation only if the tuple in relation 1 is deleted which is referenced by foreign key from other tuples of table 2 in the database, if such deletion takes place then the values in the

tuple of the foreign key in the table 2 will become empty, which will eventually violate referential Integrity constraint.

Solutions that are possible to correct the violation to the referential integrity due to deletion are listed below:

- Restrict:- Here we reject the deletion
- Cascade:- Here if a record in the present parent table (referencing relation) is deleted, then the corresponding records in the child table (referenced relation) will automatically be deleted.
- Set null or set default: Here we modify the referencing attribute values that cause violation and we either set NULL or change to another valid value.

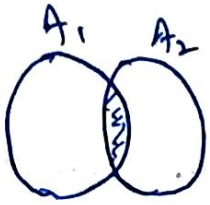
- Updation operation:
  - It is not allowed to update a row of the referenced relation if the referencing attribute uses the value of the referenced attribute of that row.
  - Such an updation violates the referential integrity constraint.

Calculate possible super keys

- $A_1, A_2, A_3, A_4 \ldots A_n$

① c.k $\{A_1\}$ $= 2^{n-1}$

② c.k $= \{A_1, A_2\}$



$= 2^{n-1} + 2^{n-1} - 2^{n-2}$

③ c.k $= \{A_1, A_2 A_3\}$

$= 2^{n-1} + 2^{n-2} - 2^{n-3}$

④ c.k $= \{A_1 A_2, A_3 A_4\}$

$= 2^{n-2} + 2^{n-2} - 2^{n-4}$

⑤ ck $= \{A_1 A_2, A_3 A_1\}$

$= 2^{n-2} + 2^{n-2} - 2^{n-3}$

Relation Algebra

1. Select (σ)
2. Project (π)
3. Union ($\cup$) - $A \cup B$
4. Set Difference - $(A - B)$
5. Cartesian product ($\times$) - $A \times B$

R :-

A	B	C	D
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10

① Select: $\sigma_{A = B \wedge D > 5}(R)$

$A = B$: Q_1 ; $D > 5$: Q_2

$Q = Q_1 + Q_2$

Q_1

A	B	C	D
α	α	1	7
β	β	12	3
β	β	23	10

Q_2

A	B	C	D
α	α	1	7
β	β	23	10

② Project :

$\Pi_{AC}(R) =$

A	C
α	1
α	1
β	12
β	12

⇒

A	C
α	1
β	12

③ Union :

No. of attributes should be same

R_1 :-

A	B
α	1
α	2
β	1

R_2 :-

C	D
α	3
α	2
β	3

$R = R_1 \cup R_2$

A	B
α	1
α	2
α	3
β	1
β	3

④ Set Difference

R :-

A	B
α	1
α	2
β	1

S :-

A	B
α	2
β	3

R − S :

A	B
α	1
β	1

⑤ Cartesian Product:

R :-

A	B
α	1
β	2

S :-

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

Row = R × S

= 2 × 4

= 8

(R × S)A=C

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b