Unit-1

Data:- It means known facts that can be recorded and that have implicit meaning.

Information:- Transform the data into a meaningful form is called information.

Database:- Database is a collection of related data.

DBMS:- DBMS is a collection of programs that enables user to create and maintain a database. DBMS is a general purpose software that facilitates the processing of defining, constructing, manipulating, sharing databases among various user and application.

Defining:- It means datatype, structure or constraints of the data to be stored in the database.

Constructing:- Constructing the database is the process of storing the database on some storage Media.

Manipulating:- Manipulating include functions such as query the database to retrieve

specific data.

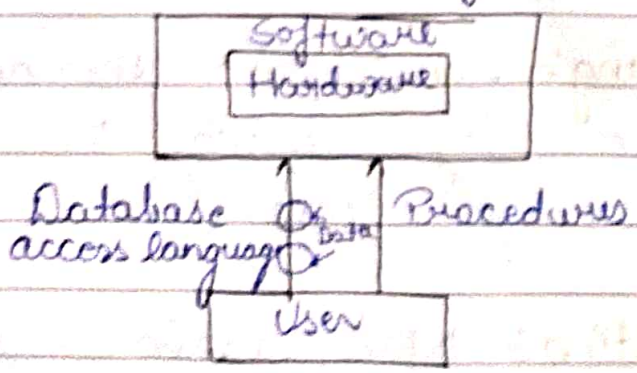
Sharing database:- It allows multiple user and programs to access the database parallelly or simultaneously.

- characteristics of DBMS:-

1. Data stored in tables.
2. Reduced Redundancy.
3. Data consistency.
4. Support multiple User and concurrent access.
5. Query language
6. Security.
7. Support transactions

Components of DBMS:-

1) Hardware
2) Software
3) Data
4) Procedures
5) Database Access language



End User:-

1) Casual end User
2) Naive or parametric end User
3) Sophisticated end User
4) Standalone User

1) Casual end User:- Occasionally, access the database.

2) Naive or parametric end User:- There main job function constantly query and update the database using standard type of queries.
Ex:- Reservation clerk, bank employees

3) Sophisticated end User:- It includes scientists, engineers, business analysts

4) Standalone Users:- They maintain personal databas
Ex:- User of tax package

Database schema and Instance:-
It is the overall description of the database is called schema. Basic structure of how the data will be stored in the database. It is of 3 types:-

1) Logical schema
2) Physical "
3) View "

Instance is the stw collection of information stored at particular movement. Instance can be changed by certain operations like updation, deletion of the database.

Difference b/w schema and Instance:-

Schema	Instance
1. It is the overall description of the database.	Collection of information stored in a database at a particular movement.
2. It is same form whole database.	Data in instance can be changed.
3. Does not change frequently.	Changes frequently.
4. Defines how the data will be stored in database.	Set of information stored at a particular time.

Actors:- Database Administrator
Database Designers.

Advantages & Disadvantages of DBMS:-

Disadvantages:-
1. Costly
2. Complex
3. Larger in size.

4. Performance.

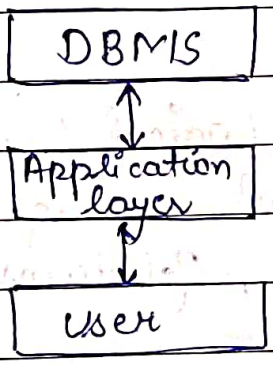
Advantages:-

1. Redundancy problem can be solved.
2. Security.
3. Data integrity.
4. Support Multiple users.
5. provide back up of data.

DBMS Architecture:-

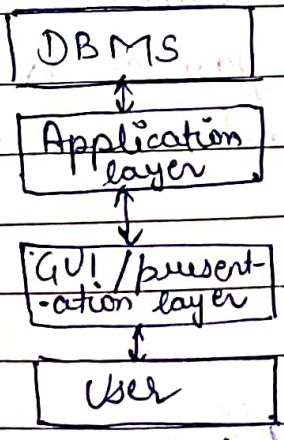
1. One-tier architecture.
2. two "
3. three "

2.

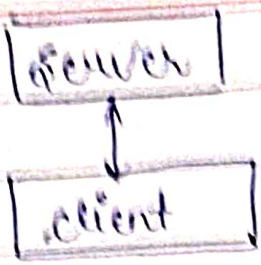


2-tier.

3.



3-tier.



1-tier.

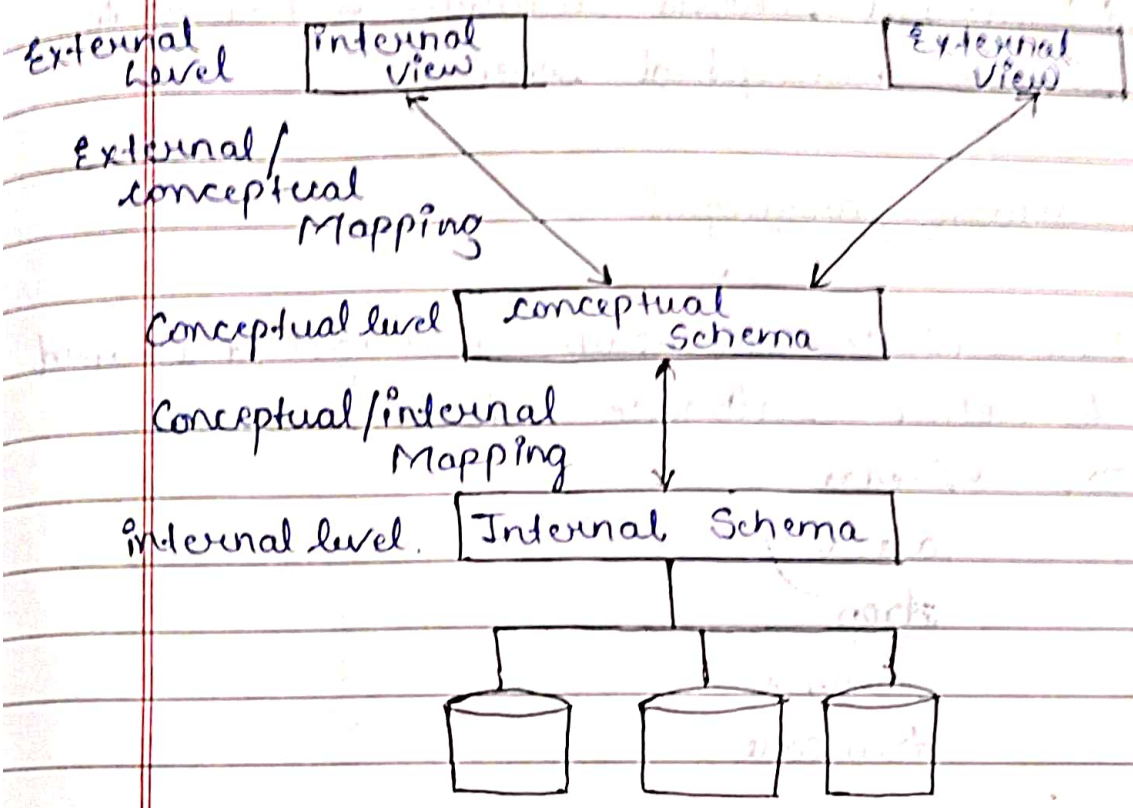
→ Database Schema is sometimes called the "intension" and database state is called "Extension" of the schema.

Three Schema Architecture:- It can be defined at the following three levels:-

1) Internal Level or Physical Level:- which describe physical storage structure of the database. It also describes complete details of data storage and access path of the database.

2) Conceptual Level or Logical Level:- It hides the detail of physical storage and describes (Entities, Datatype, Relationship, Constraints).

3) External Level or View Level:- It includes user view, describes the part of the database that particular user group is interested in and hides the rest of the database.



Data Independance:- It can be defined as capacity to change at one level of a database system without having to change next higher level.

We can define 2 types of data independence

1) Logical Data independence
2) Physical " "

1) Logical DI:- It is the capacity to change the conceptual schema without having to change external schema or application program.

5) Physical DI:- It is the capacity to change the internal schema without having to change conceptual schema.

Database languages

1. DDL:- (Data definition language) It is used to define database structures.
   Eg:- create
   alter
   drop
   rename
   truncate

2. DML:- (Data Manipulation language) It is used for storing and manipulating data in a database. It handles user request.
   Eg:- Insert, Update, delete

3. DCL:- (Data control language) It is used for to retrieve stored or saved data
   Eg:- Grant, Revoke

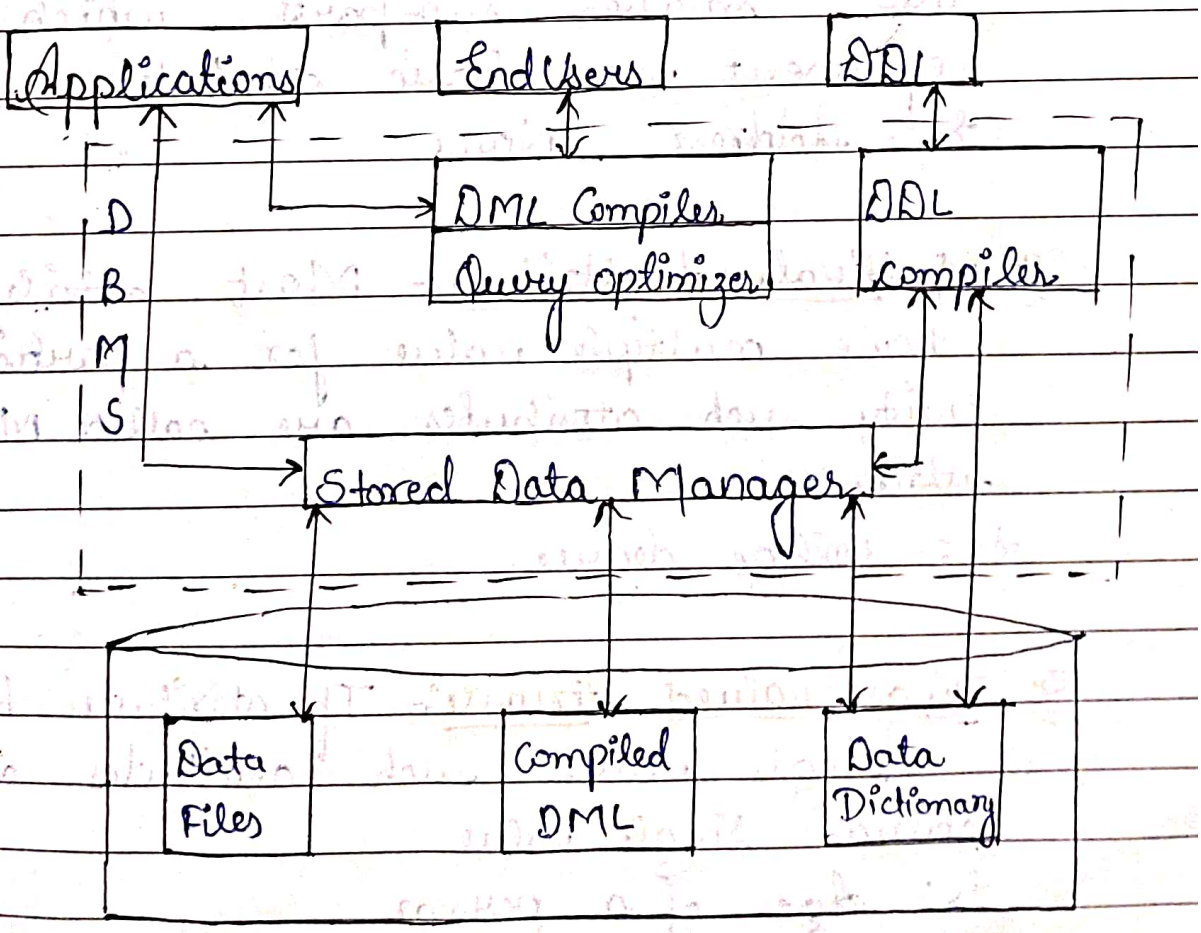
4. TCL:- (Transactional control language) It is used to run the changes made by the DML statement. TCL can be grouped into logical transactions.
   Eg:- Commit, Rollback

5. DQL:- (Data Query language) It is used to retrieve data from database.

Dbms Interfaces:-

1. Menu Based Interface
2. Forms " "
3. Graphical User Interface
4. Natural language "
5. Speech I/O
6. Interface for DBA
7. Interface for parametric user.

Overall database structure:-



E-R Model Concept:-

1. Entities:- Entity is a thing in the real world with independent existance. Entity may be object with physical existance or logical existance.

2. Attribute:- Each entity has a attributes, the particular properties that describe it.
Ex:- Employee entity may be described by employee_id, emp_name, salary.

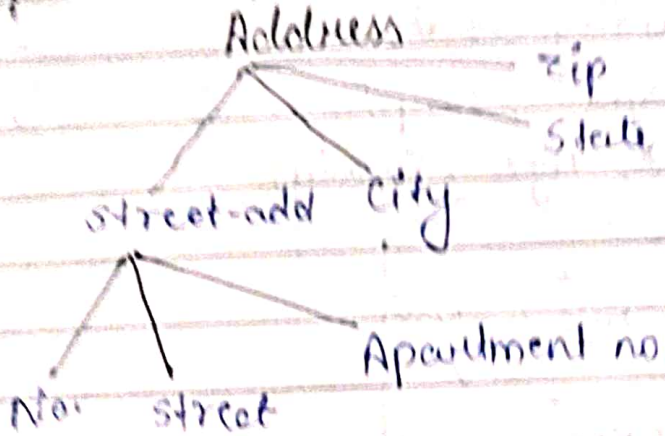
3. Type of Attribute:-
1. Composite attribute:- It can be divided into smaller sub-parts, which represent more basic attribute.
Ex:- Address, Name.

2. Multivalued attribute:- Most attribute have multiple value for a particular entity. such attributes are called Multivalue attribute.
Ex:- College degree

3. Single valued Attribute:- If attribute have a single value such attributes are called single value.
Ex:- Age of a person.

Stored vs derived attribute:- In some cases, two attributes are related.
Ex:- Age & birth date.
Age is called derived attribute & birth date is called stored attribute.

④ Composite:-



⑤ Null values:- In some cases, particular entity may not have applicable values for an attribute.
Ex:- Appartment no. may be null for a particular entity.

⑥ Complex Attribute:- It is the combination of composite & Multivalued attribute.
Ex:- {Address_phone({phone(Area_code, phone_number)}, Address(street_add, No., Apart_no)}

Entity Type:- Entity type defines collection of entities that have same attributes. Each (E.T) in the database is described by its name & attribute.

Entity Set:- Collection of all entities in the database at any point in time is called entity set.

Ex: ET: Employee, employee_id
Name {Name, Age, Salary}.

Entity set (Extension)

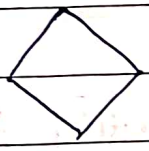
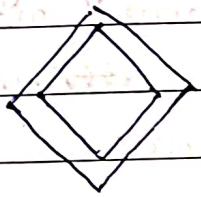
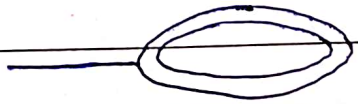
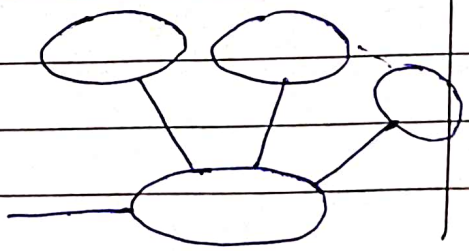
e_1 (ABC, 20, 40K)
e_2 (XYZ, 30, 50K)
e_3 (EFG, 40, 60K)

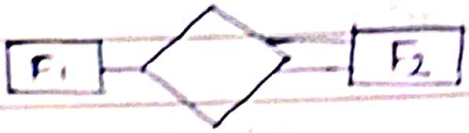
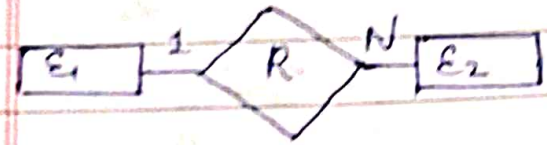
Key attributes of an Entity type:- E.T. has an attribute whose value are different for each individual entity. In the entity set, such an attribute is called key attribute and its value can be used to identify each entity uniquely.

Value set of attribute (Domain):- Simple attribute of an entity type is associated with value set, which specifies set of values that may be assigned to that attribute.

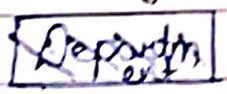
Value set are specified using basic data type available in programming language

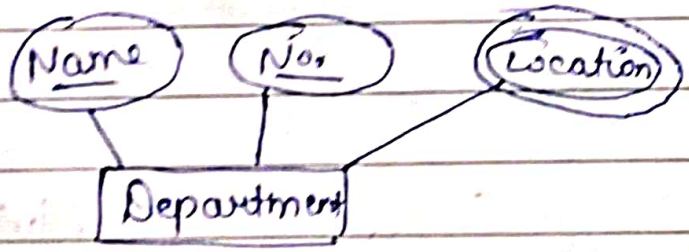
E-R diagram Notation:-

Symbol	Meaning
	Entity
	Weak Entity
	Relationship
	Identifying Relationship
	Attribute
	Key Attribute
	Multi-valued attribute
	composite Attribute

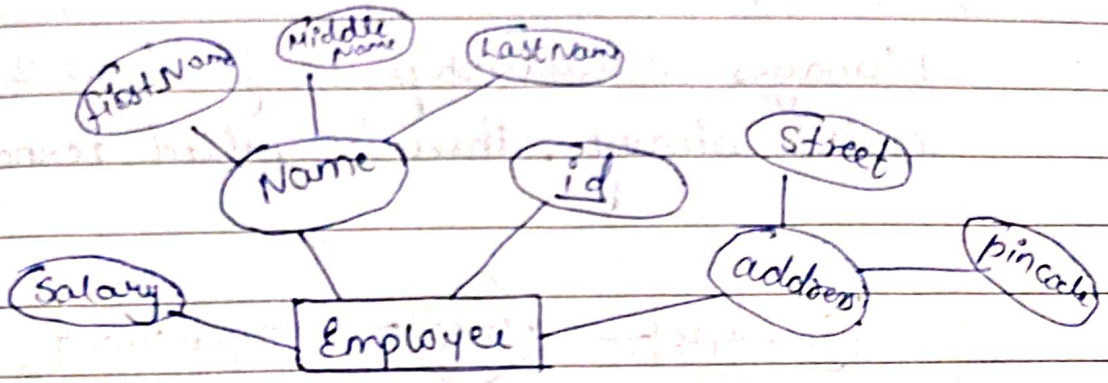
	Derived attribute
	Total participation of E_2 in R
	Cardinality Ratio 1:N for $E_1 : E_2$ in R

Que Draw an ER-Model, entity type department with attribute name, no., location. Location is multi-valued attribute. Name and no. are key attribute.



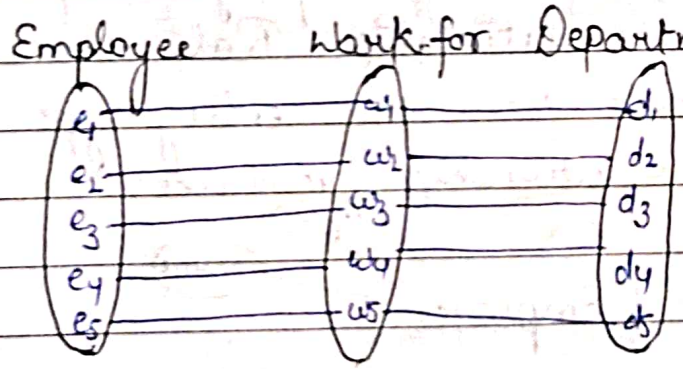


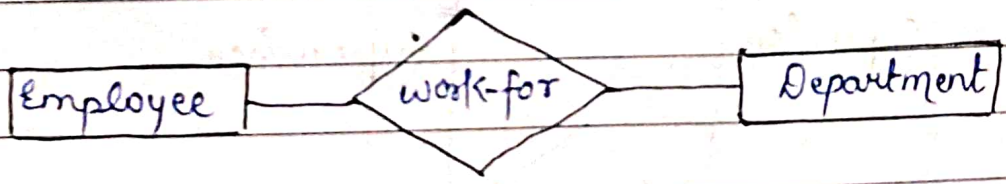
Entity employee with attribute name, id, address, salary. Both Name & address may be composite attribute & Id will be the key attribute



Relationship types and sets and instance :-
Relationship type R among n entity type $E_1 E_2 \cdots E_n$ defines a set of association or relationship set among entities from these entity type

Ex :-

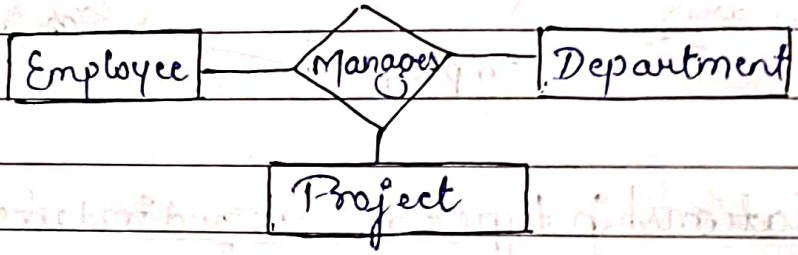




Relationship degree:- Degree of relationship type is number of participating entity type.

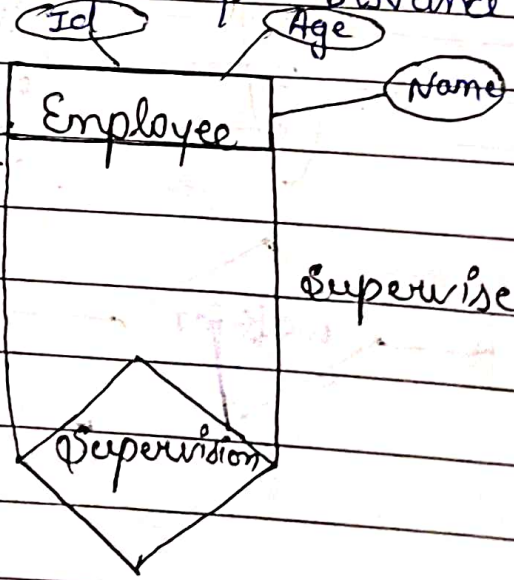
Ex:-

Manages relationship is of degree 2 is called binary, three is called ternary



Imp Role name and recursive relationship:-
Each entity type, that participate in a relationship type plays a particular role in the relationship. Role name signifies the role from the entity type plays in each relationship instance.

Ex:-



Constraints on Relationship type:- Two Type of relationship constraints.

1) Cardinality Ratio.
2) Participation Constraints.

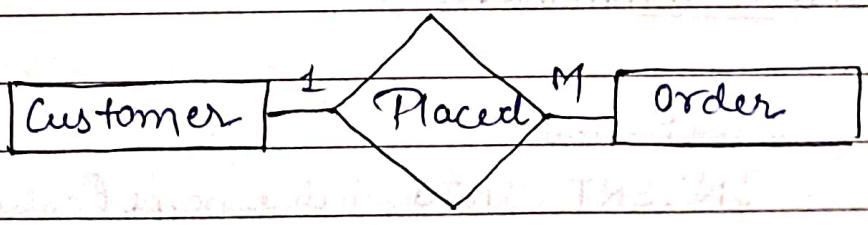
1. Cardinality Ratio:- It will be defined by 4 types:

a) One to One
b) One to Many
c) Many to One.
d) Many to Many.

Cardinality ratio for binary relationship specifies maximum no. of relationship instance that entity can participate in.

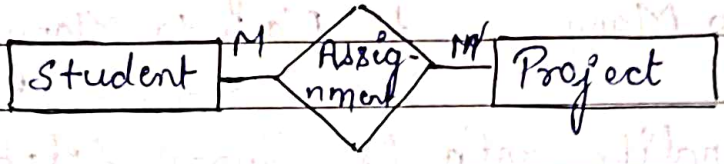
a) One to One:-

b) One to Many:-



c) Many to One :-

d) Many to Many :-



Create Command :-

~~CREATE TABLE TABLE_NAME~~

CREATE TABLE table_name(attribute_name datatype, attribute, datatype)

→ CREATE TABLE student (Rollno. int, Name varchar(20), class varchar(20));

Insert Command :-

~~Insert into~~

INSERT INTO table_name (column1, column2, ...)

VALUES (value1, value2 ...);

INSERT INTO student (Rollno, Name, class) values (10, 'ABC', 'MCA');

3. SELECT * FROM student;

when we want to display roll no.
SELECT Rollno. From student;

4. SELECT * FROM student where class = 'MCA'.
for specifing only MCA.

5. DELETE FROM table_name;
DELETE FROM student where Name = 'ABC';

6. UPDATE table_name set Name = 'Gaurav' where Rollno = 10;

UPDATE student set class = 'BTECH' where class = 'MCA';

1. Create a table Employee with emp_id, name, department, salary.
2. Insert atleast 5 records
3. Display all the attributes
4. Display Name & salary for those employee who exist in department MCA
5. Change the value of department whose empid is 10.

6. Delete at least 2 rows from the table.

1. CREATE TABLE EMPLOYEE (emp_id int, emp_name varchar[20], department var_char[20], salary int);

2. INSERT INTO EMPLOYEE (emp_id, emp_nam department, salary) values((1, 'Parul', 'MCA', 20000), (2, 'Shikha', 'MCA', 10000), (3, 'Suresh', 'MBA', 20000), (4, 'Harish 'MTECH', 50000), (5, 'Heena', 'MBA', 5000);

3. SELECT * FROM EMPLOYEE;

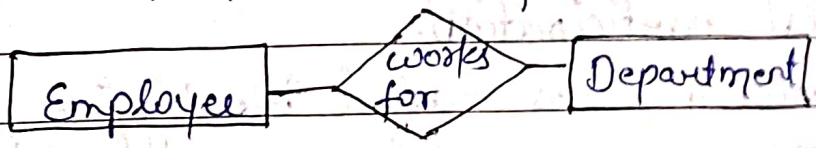
4. SELECT Name, Salary FROM EMPLOYEE where department = 'MCA';

5. UPDATE EMPLOYEE set department = 'MBA' where emp_id=10;

6. DELETE FROM EMPLOYEE where department = 'MBA';

2) Participation :- Participation constraints specify whether existance of an entity depends on its being related to another entity by the relationship type.

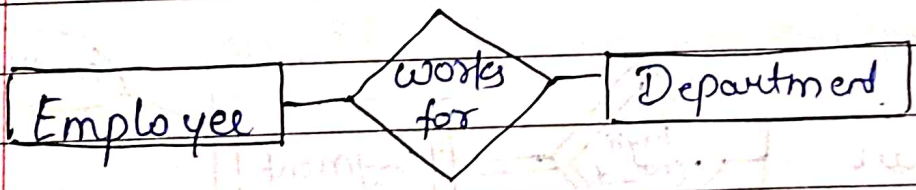
Ex:- Employee works for department



There are 2 type of participation :-

1) Total
2) Partial

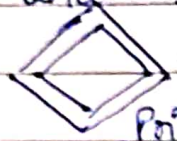
Ex:- Employee entity can exist only if it participate in atleast one works for relationship, the participation of employee in works for is called total participation.

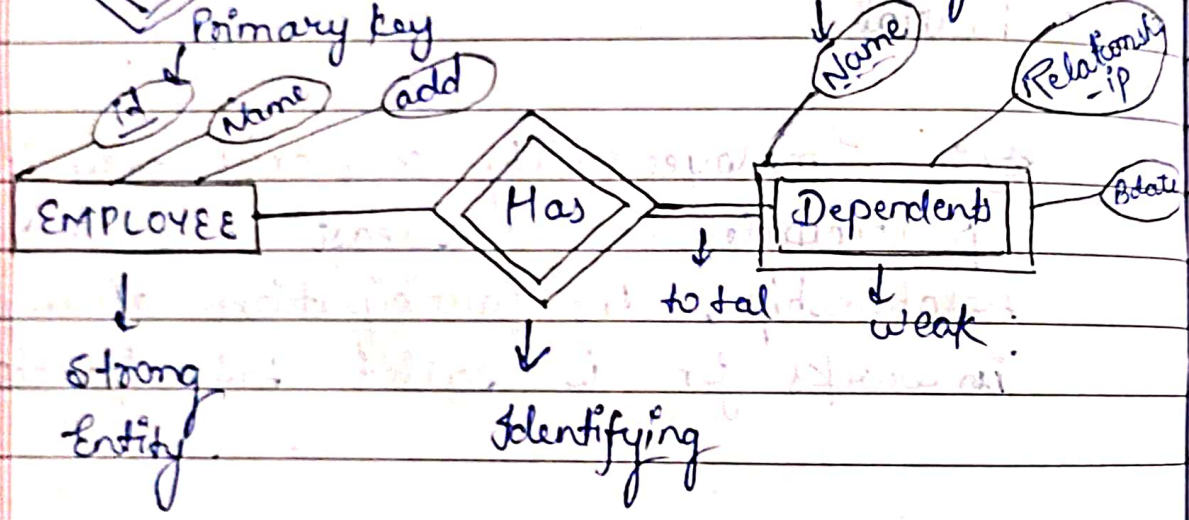


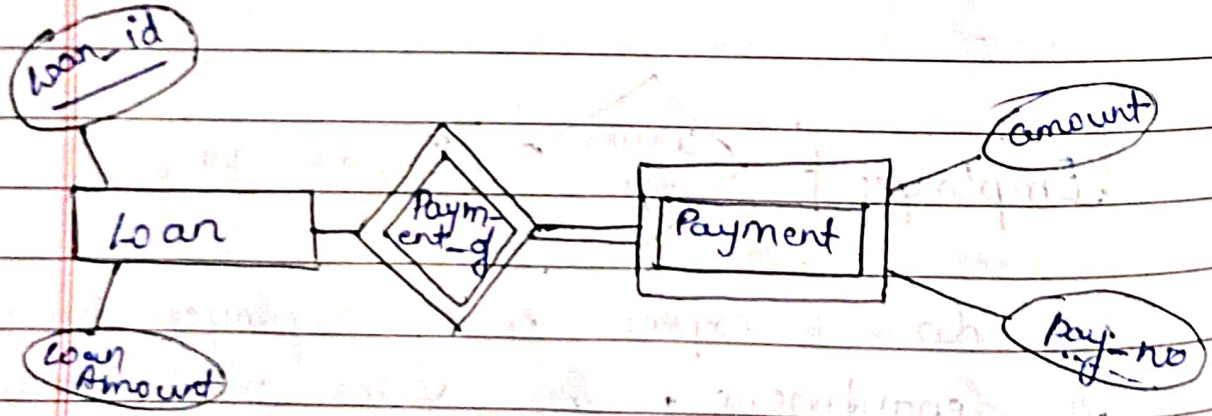
We do not expect every employee to manage a department, so the participation of employee in the manages relationship is partial.

Weak Entity type :- Weak Entity is a type of entity which does not have its key attribute. It can be identified uniquely by considering the primary key of another entity. Weak entity need to have total participation.

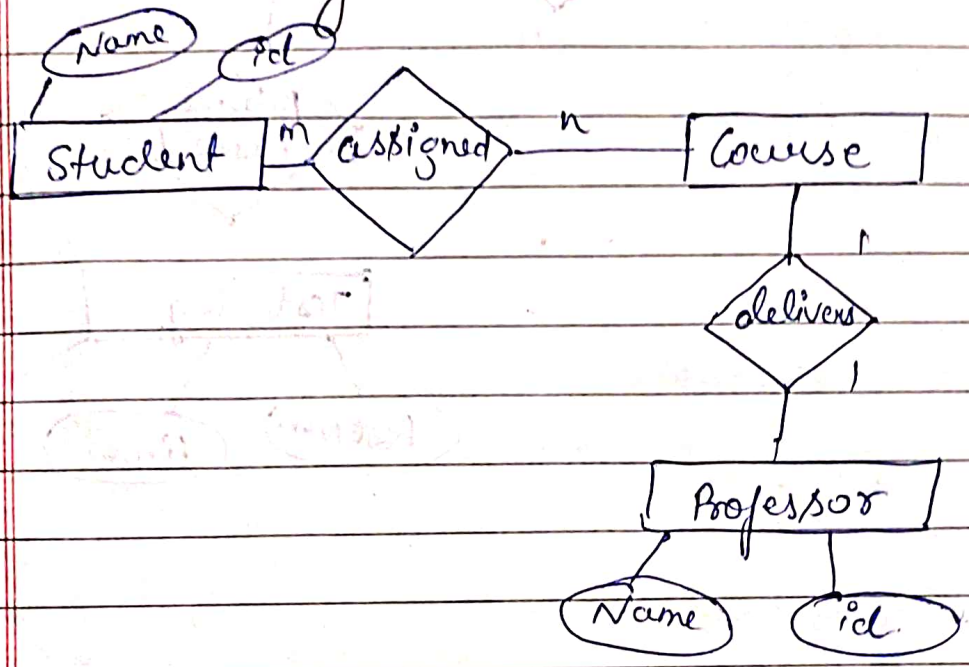
Relationship is called identifying relationship.

Ex:- It is represented by  and relationship is represented by 





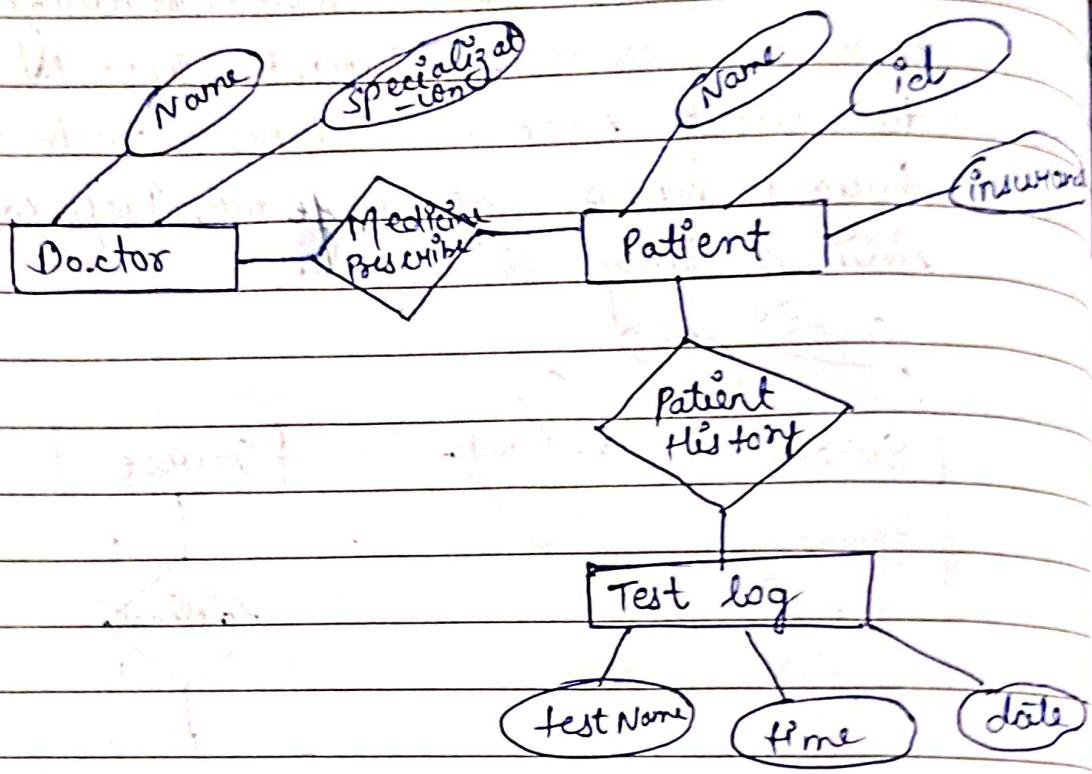
Q: Draw an ERDiagram in a University student enrolls in course. Student Must be assigned to atleast 1 or more courses. Each courses is taught By a single professor. Professor can delivers only one course.



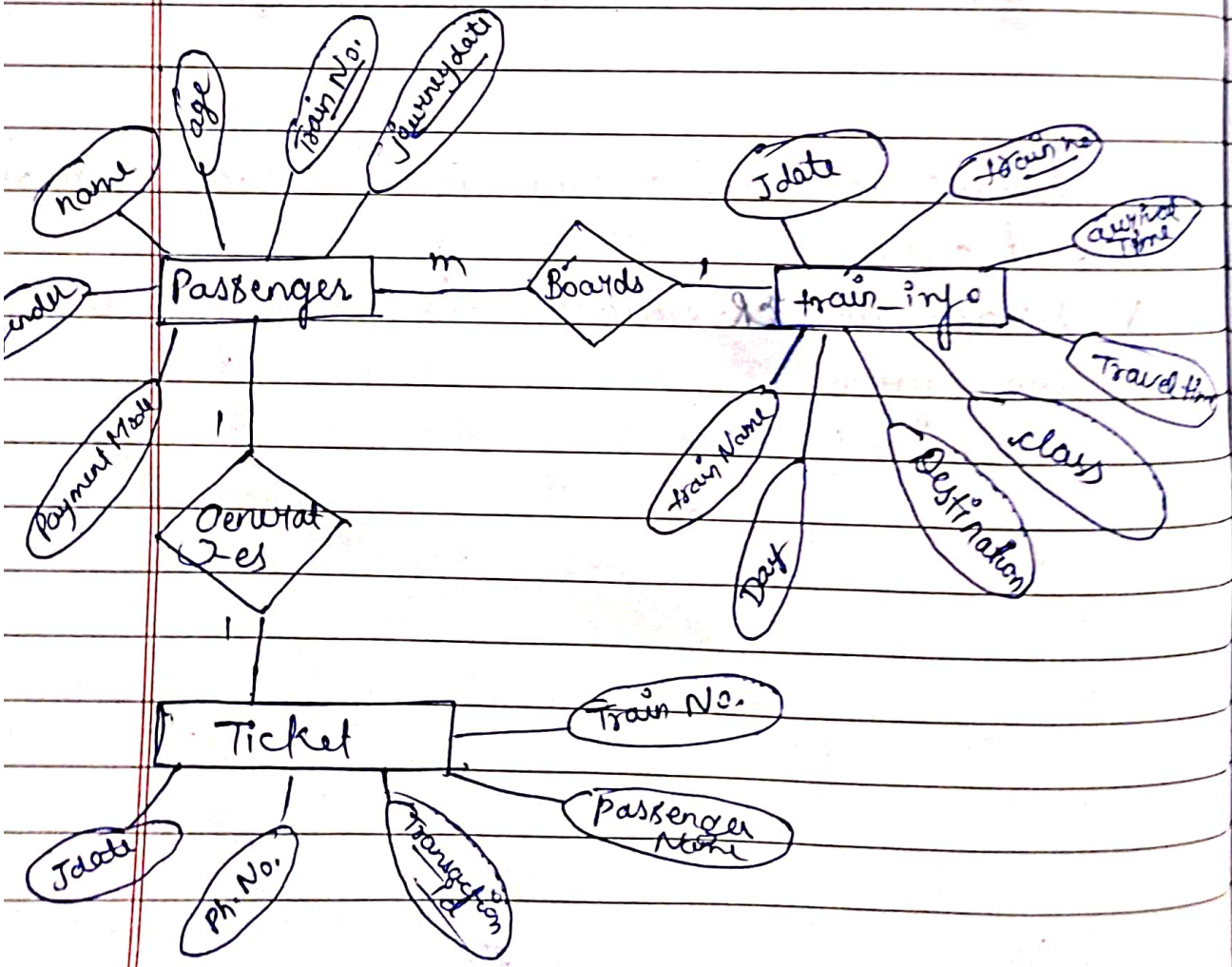
Q. Draw an ER-Diagram for hospital Management system.

Q. Draw an ER-Diagram for Railway Reservation system.

①


Name
Specialization
Doctor
Medical Prescribe
Name
id
Insurance
Patient
Patient History
Test log
test Name
time
date

②


name
age
Train No.
Journeydate
Passengers
m
Boards
1
Jdate
Train No
arrival Time
train_info
Travel time
class
Destination
Dept
train Name
Payment Mode
1
Operated Yes
1
Ticket
Train No.
Passenger Name
Jdate
Ph. No.
Transaction Id

Concept of keys:- There are different types of key.

1. Superkey
2. Candidate key
3. Primary key
4. Unique key
5. Composite key
6. Foreign key
7. Alternate key

Keys in dbms is an attribute or set of attributes which help to identify a row in a table.

Emp-id	Emp-name	Emp_adhar	Emp_sal	Email
1	ABC	1111	40K	a@gmail.com
2	XYZ	2222	50K	NULL
3	EFGh	3333	60K	c@gmail.com

1. Super key:- Combination of one or more attribute which taken collectively to identify a row (record) uniquely.

Ex:-

S.K = {Emp_id}, {Emp_adhar}, {Empid, name}, {Empid, sal}, {Empid, Empemail} {Empadhar, name}, {Empadhar, sal}, {Empadhar, email}

2. Candidate key:- Minimal set of superkey is called candidate key.
Ex:- CK = {Emp-id}, {Emp_adhar}

3. Primary key:- Any key from candidate key is called primary key.
The value of that attribute can never be null. The value can never be duplicate.

4. Unique key:- It can be NULL

Ex:- {Emp_email} will be unique.

5. Alternate key:- All candidate key except primary key is called primary key.

Ex:- {Emp_adhar}

6. Foreign key:- Foreign key is a column that creates a relationship between two tables.
The purpose of foreign key is to maintain data integrity and allows navigation between two different instance of an entity. It acts as a cross reference between 2 tables as it reference primary key of another table.

Ex:-

⊙ It can be null or repeated. but not in primary key.

① Employee

Emp_id	Empname	Dept_id

Emp_id ↑ P.K

Dept_id ↑ Foreign key

② Department

Dept ID	Dname	Project

↑ Primary key

Ex:-

Emp

Empid	EmpName	Deptid
01	XYZ	1
02	ABC	1
03	Efg	NULL

Department

Dept_id	D-name	Project
1	MCA	DB
2	MBA	c++
3	Btech	Java.

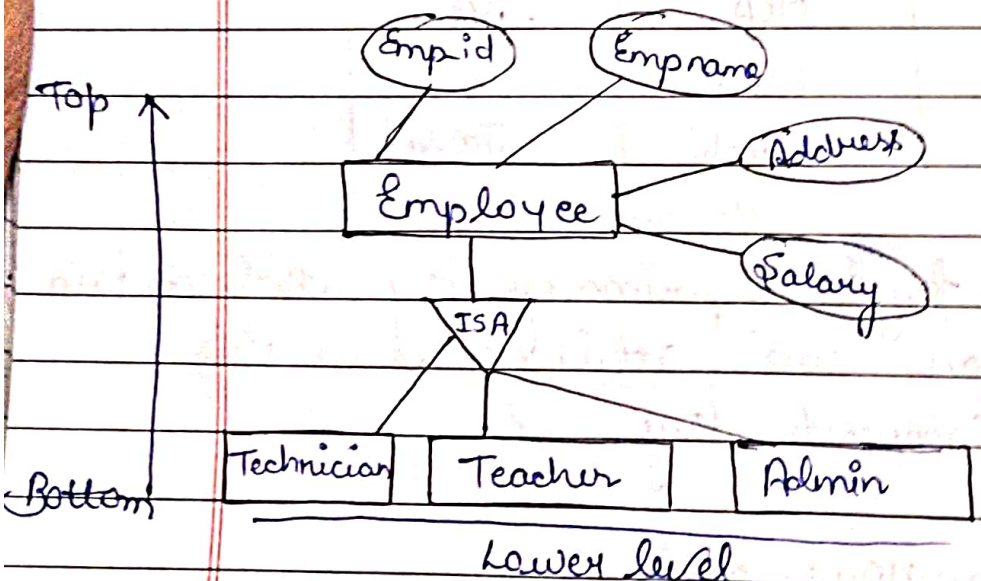
Composite key:- A primary key which contain more than one attribute so this called Composite key.

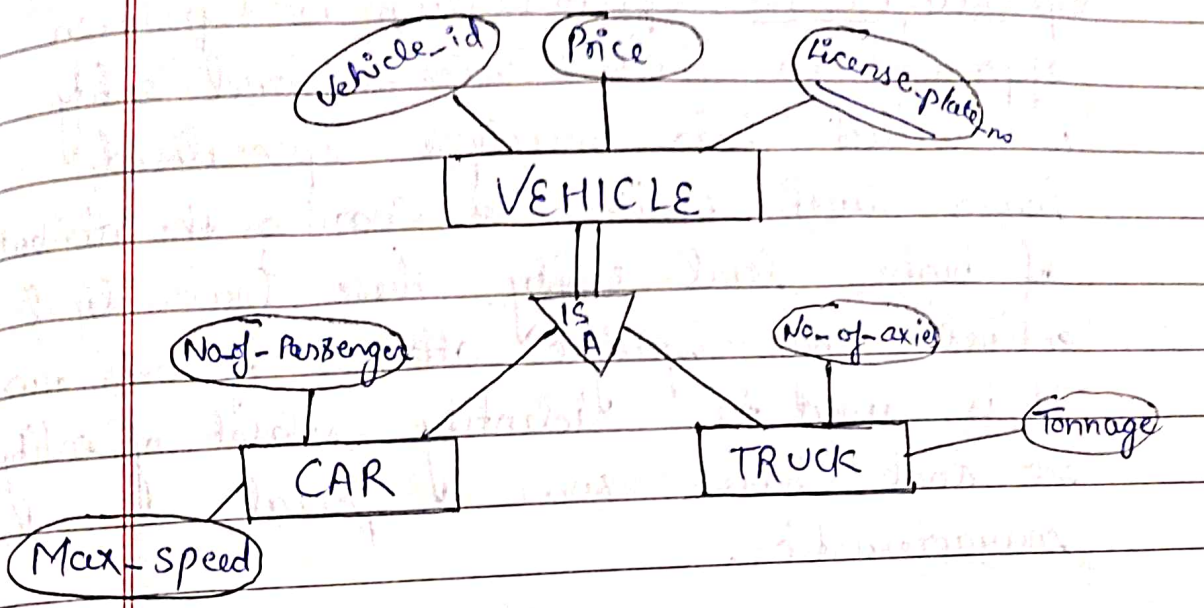
Ex:- { Empname, add }

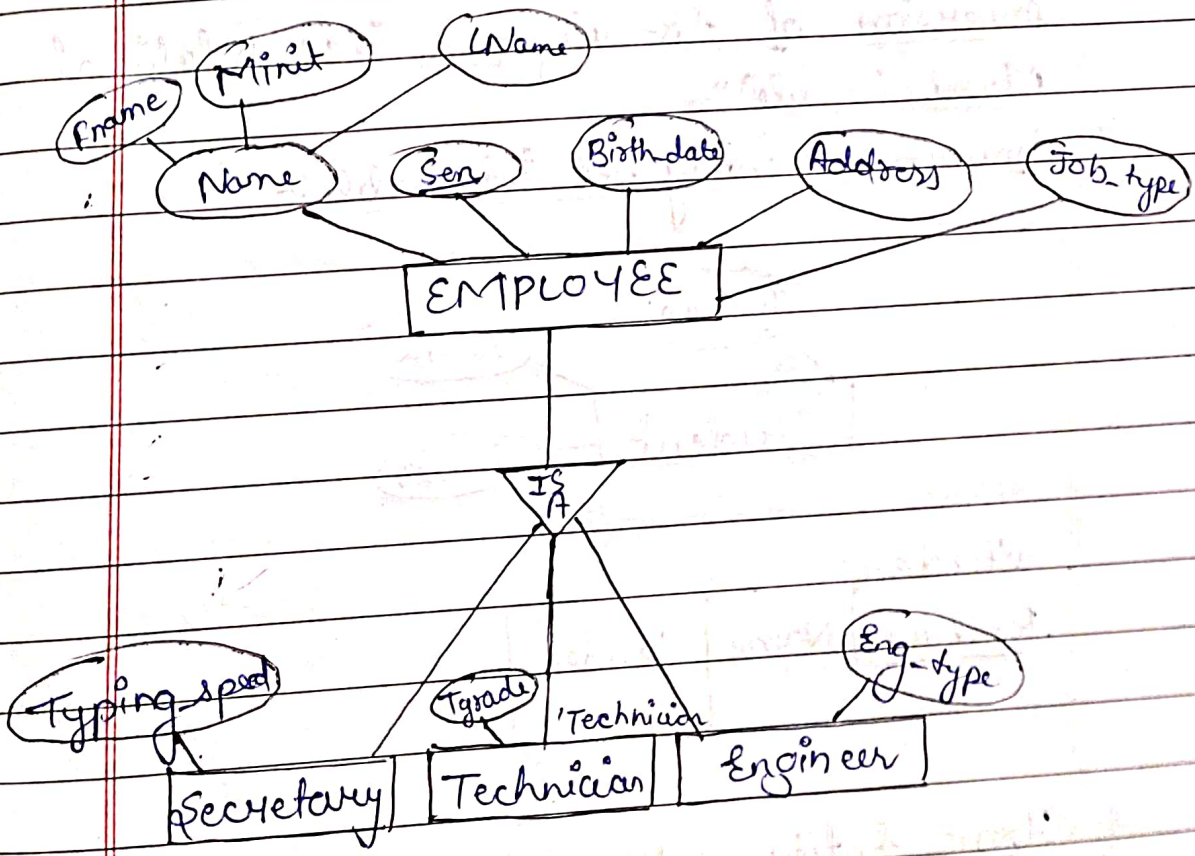
Extended / Enhanced E-R Model:- (EER-Model)

1. Generalization
2. Specialization
3. Aggregation.

1. **Generalization:-** Generalization is bottom-up approach in which two or more entities of lower level combine to form higher level entity if they have some attribute in common. Generalization is more like subclass, & superclass system.
Entities are combined to form more generalized entity i.e., subclasses are combined to make a superclass.



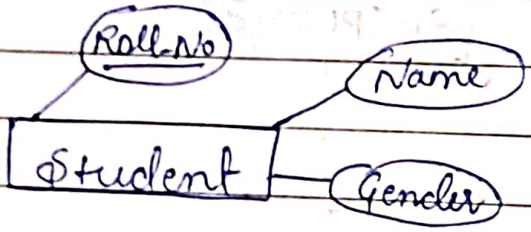

Vehicle_id
Price
License_plate_no
VEHICLE
IS A
No_of_pass
No_of_axles
CAR
TRUCK
Tonnage
Max_speed


Fname
Minit
Lname
Name
Sex
Birthdate
Address
Job_type
EMPLOYEE
IS A
Typing_speed
Tgrade
Technician
Eng_type
Secretary
Technician
Engineer

Specialization:- Specialization is top down approach in which higher level entity is divided into multiple specialised lower level entities. To sharing the attribute of higher level entity, these lower level entities have specific attribute of their own It is used to identify subset of entity set that shares some special characteristics.

Conversion of ER-diagram into Relational Model (table):-

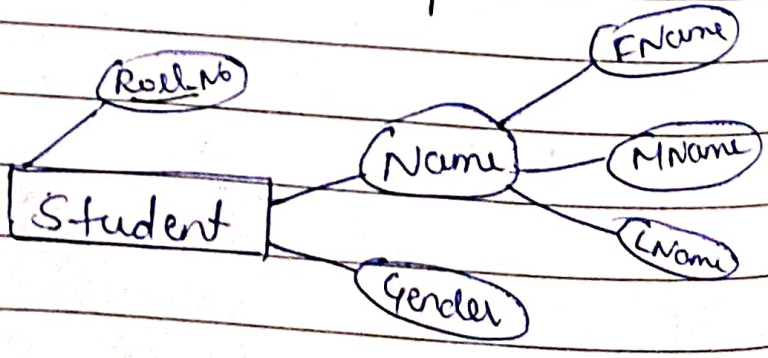
1. Converting Entity set into table:-



Student

Roll No.	Name	Gender

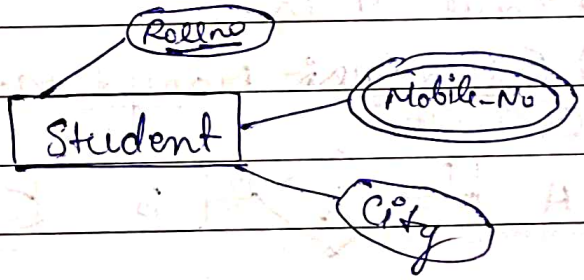
2. Strong Entity set with composite attribute:-



Student

RollNo	fName	MName	LName	Gender

3. Strong Entity with Multivalued Attribute:-
It will require two table in Relational Model
① One table will contain simple attribute with primary key.
② Other table will contain primary key & multivalued attribute.

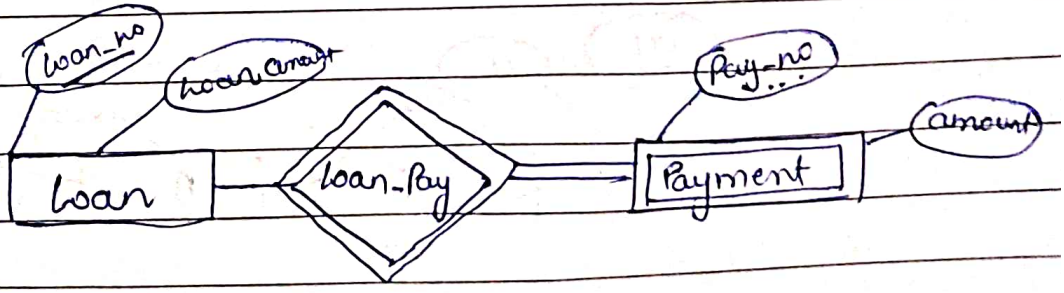


Student

Rollno	city

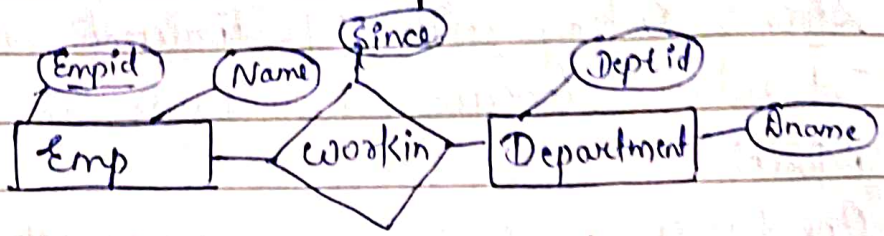
Rollno.	Mobile No

4. Convert weak entity set into table :-



Loan No.	payno.	amount

Convert Relationship set into Table:-

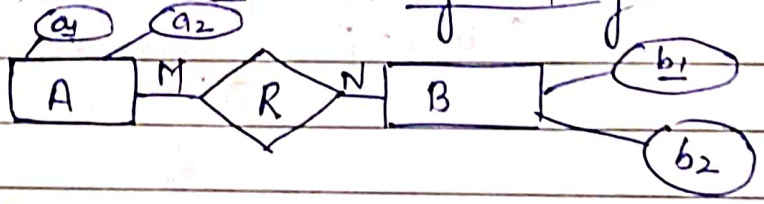


Empid	Deptid	Since

Binary Relationship with Cardinality Ratio:-

There are 4 cases:-

① (M:N) :- Many to Many:-



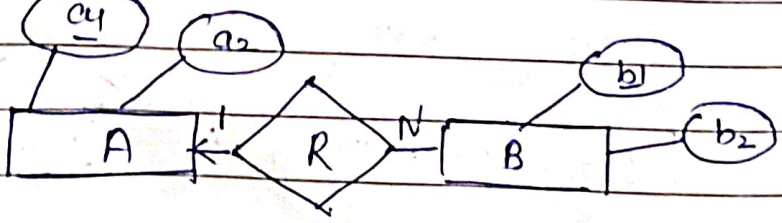
Three tables will be required.

i) A (a_1, a_2)

ii) R (a_1, b_1)

iii) B (b_1, b_2)

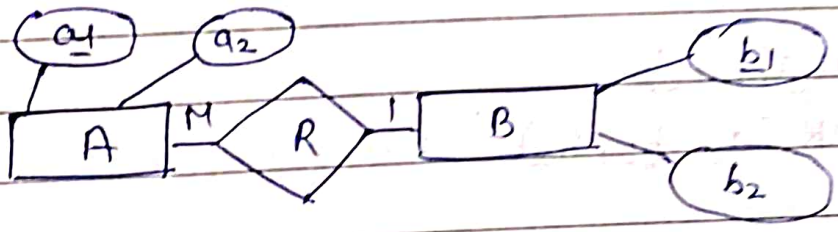
② (1:M) One to Many:-



Two tables are required.

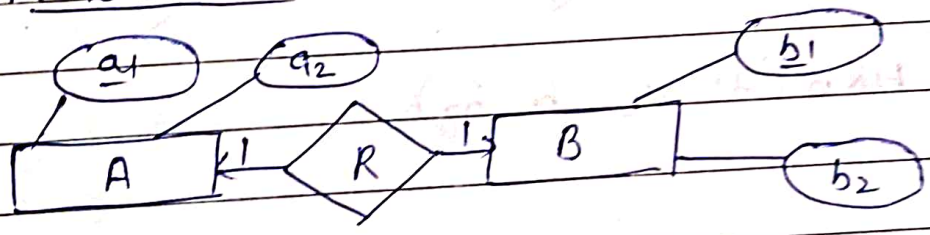
1. $A(\underline{a_1}, a_2)$
2. $BR(a_1, \underline{b_1}, b_2)$

③ Many to One (M:1):-



1. $AR(\underline{a_1}, a_2, b_1)$
2. $B(\underline{b_1}, b_2)$

④ One to One (1:1):-

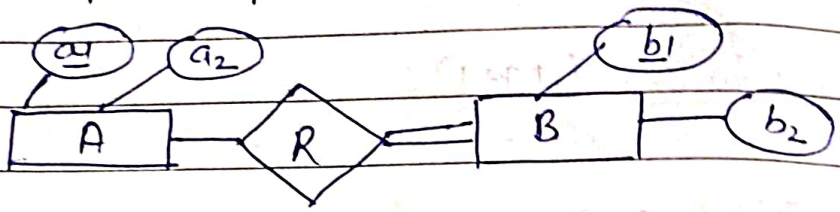


① Two tables will be required either combine R with A or B.

1. way one:- $AR(\underline{a_1}, a_2, b_1)$
   $B(b_1, b_2)$
2. way two:- $A(\underline{a_1}, a_2)$
   $BR(\underline{a_1}, \underline{b_1}, b_2)$

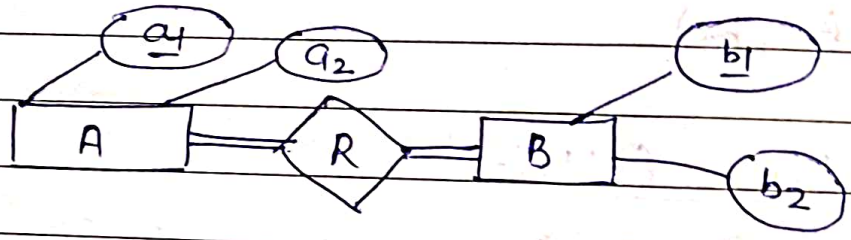
Binary relationship with both cardinality & participation constraints:-

Partial & Total participation:- case 1:-



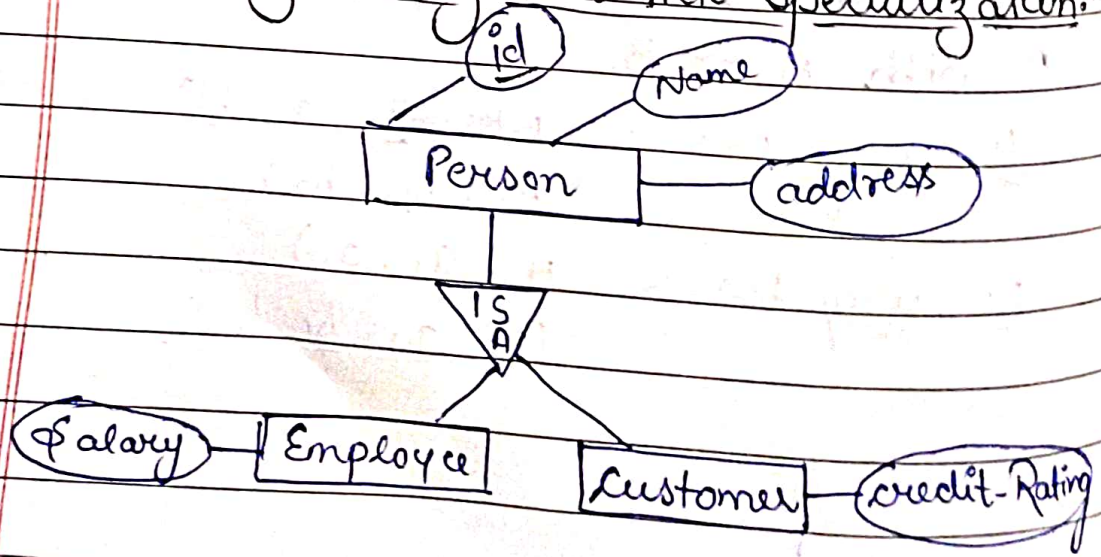
1. A($\underline{a_1}$, a_2)
2. BR (a_1, $\underline{b_1}$, b_2)

case 2:-



ARB ($\underline{a_1}$, a_2, $\underline{b_1}$, b_2)

Convert generalization into Specialization:-



Employee(Id, name, add, salary)
Customer(Id, name, add, credit_rating)

Employee:-

Empid	name	add	salary

Customer:-

cust_id	name	add	cred_rating

Generalization.

person (id, name, add)
Employee (id, salary)
Customer (id, credit_rating)

Person

id	Name	add

Employee

id	Salary

Customer

id	Credit_Rating

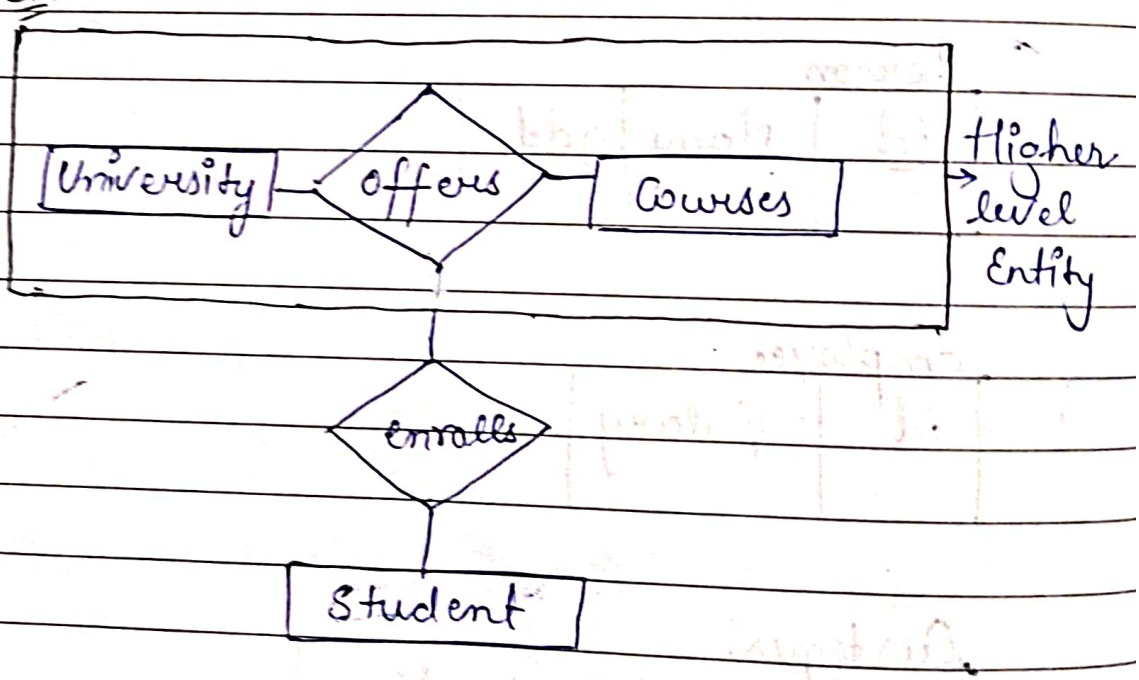
Specialization.

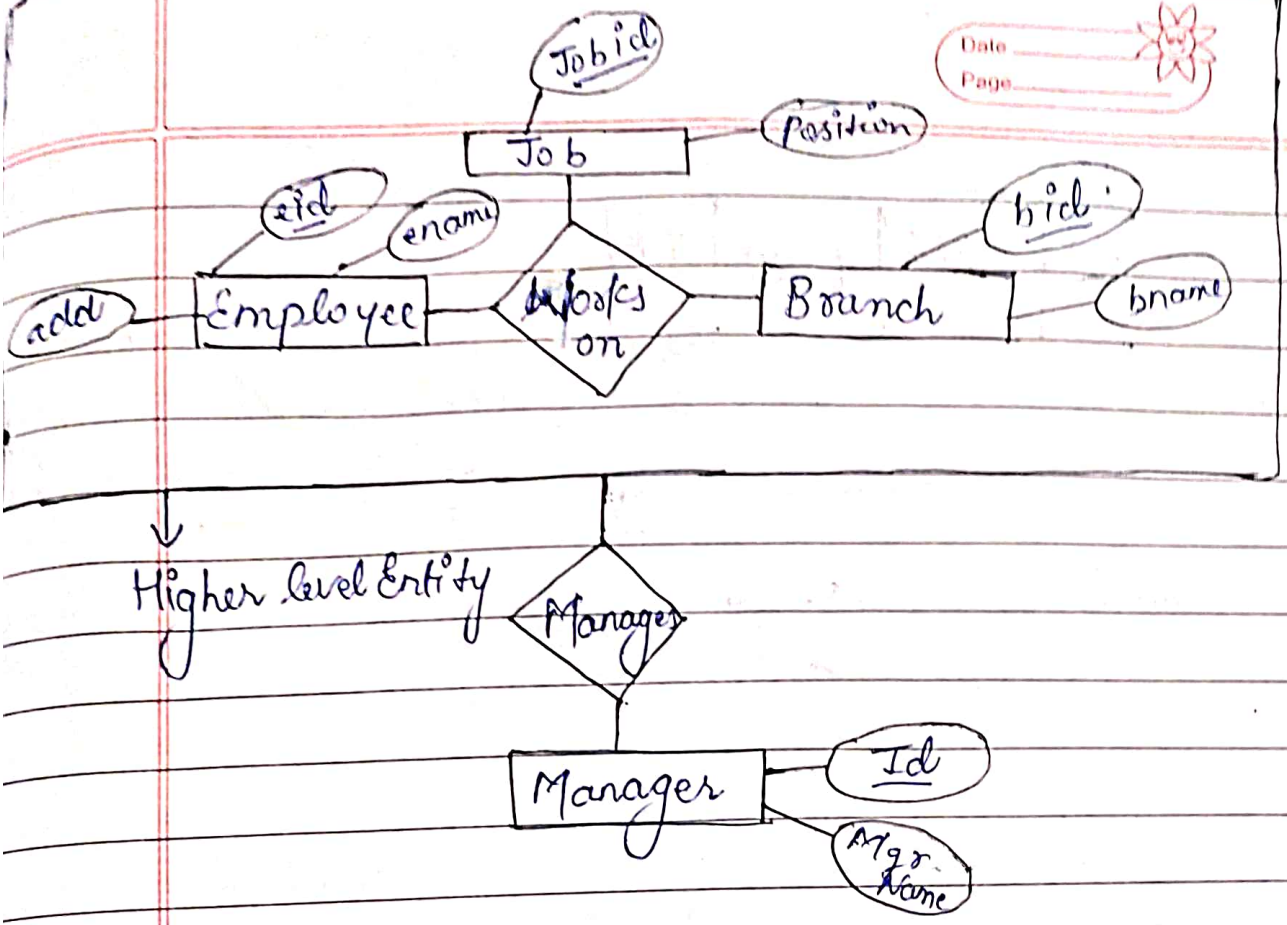
Aggregation:- Aggregation refers to the process by which entities are combine to form single meaningful entities to establish single entity, Aggregation creates a relationship that combines these entities.

It is the design strategy in which relationship is modelled b/w a collection of entities & another relationship.

It is used when we need Express a relationships among other relationship.

Ex:-





↓ Higher level Entity

Relationship to higher degree:-

Cardinality Ratio

1. One to One
2. One to Many
3. Many to One
4. Many to Many.

Employee

eid	ename	add

Branch

bid	bname

Manages

Id	Mgr Name

WorksOn

eid	Job id	bid

Date
Page

Manages

eid	Bid	jobid	Id

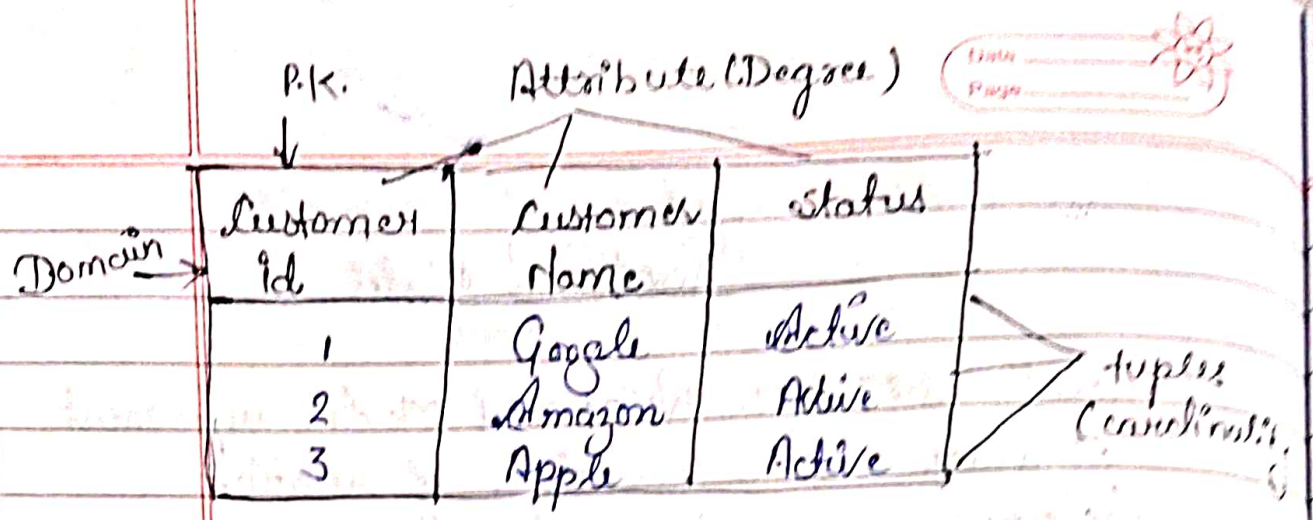
Unit-2

Relational Model

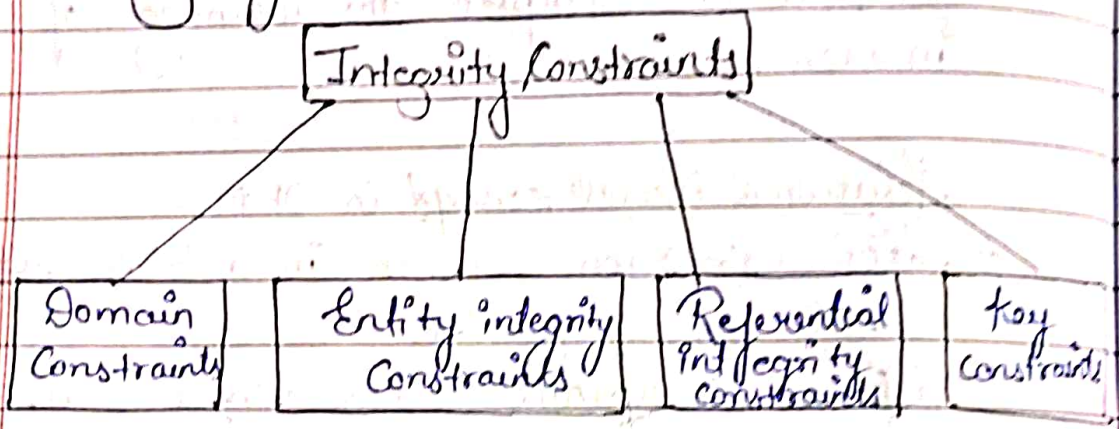
Relational Model:- Relational Model represent database as a collection of relation. Relation is nothing but a table of values. Table name and column name are helpful to interpret the meaning of values in each row.

Relational Model concept in DBMS:-

1. Attribute:- Each column in a table. Attribute are the properties.
2. Tuples:- It nothing but a single row, which contain single record.
3. Relation Schema:- It represent the name of the relation with attribute.
4. Degree:- Total no. of attribute in the relation is called the degree of the relation.
5. Cardinality:- Total no. of rows present in the table.
6. Relation Instance:- It is the finite set of tuples in RDMS system. They can never be same/duplicate.
7. Relation key:- Every Row has one or multiple attribute which is used to define the record uniquely. It is called relation key.
8. Attribute Domain:- Every attribute has some predefined value & scope which is known as domain.



Integrity Constraints:-



It is the set of rules. It is used to maintain quality of information.
It ensure that data insertion, updation & other processes have to be performed in such a way that data integrity is not effected.

Domain Constraints:- * It can be defined as valid set of values for an attribute.
* Datatype of domain include integer, character, date, time etc. Value of the attribute must be available in the corresponding domain.

Id	Name	Age
101	John	18
102	ABC	'A'
103	EFG	19

integer

→ Age is integer value

Entity integrity Constraints:- It states that (primary key value can not be NULL). Table can contain NULL value other than primary key field.

Id	Name	Age
101	John	18
102	ABC	16
103	EFG	19
NULL	John	20

↙ Not allowed as primary key can't contain NULL value.

Referential Integrity Constraints:- It specifies b/w 2 tables.

* If foreign key 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 available in table 2.

Table 1

Emp_No.	Name	D-No.
1	John	11
2	ABC	24
3	EFG	18

Table 2

D-No.	D-Location
11	Noida
24	Noida
35	Delhi

Not allowed ← F.K

P.K

18 not allowed as D.No. 18 is not defined as P.K.

Key Constraints:- * Keys are the entities set that is used to identify an entity within its entity set uniquely.

* Entity set can have multiple keys, out of which one key will be a primary key.

Ex:

Id	Name	Age
101	John	18
102	ABC	16
103	EFG	19
102	John	20

↓

Not allowed, Because all must be unique.

Relational Algebra:- It is procedural query language. It consists of set of operations that take one or two relation as input & produce new relation as their result.
There are different type of operation:
Unary Relational operations:-

1. Select (σ)
2. Project (π)
3. Rname (ρ)
4. ~~Division~~ ($\div$)

Set theory :-

1. Union (U)
2. Intersection (∩)
3. Difference (−)
4. Cartesian product (X)

Binary Relational operation :-

1. Join
2. Division

1. Select :- It is used for selecting subset of the tuples according to given selection condition (σ) Sigma symbol is used to denote it.

Syntax :- $\sigma_{(p)}(R)$

R means input relation
p means predicate that is to be evaluated.

Select those account which belongs to SBI.

$$\sigma_{branchname = SBI}(Account)$$

2. Project (π) :- It defines a relation that contains vertical subset of relation. This helps to extract the value of specified attribute.

Ex :-

Syntax :- $\pi_{<Attribute>}(Relation)$

Ex:- find the bank name from the account.

$\Pi_{BankName}$ (Account)

Q. Find the name of all customers living in the new delhi.

Customers

Customer Name	City
ABC	Delhi
XYZ	New Delhi
EFG	Noida

$\Pi_{City = NewDelhi}$

$\Pi_{CustomerName}$ ($\Pi_{City = NewDelhi}$)

$\Pi_{CustomerName}$ ($\Pi_{City = NewDelhi}$ (Customer))

Union (U) operation:- Union of two sets combines all data i.e. appearing either 1 or both relations

table A

Customer 1	Customer 2
1	1
1	2

B

Column 1	Column 2
1	1
1	3

A ∪ B

Col1	Col2
1	1
1	2
1	3

Intersection (∩):-

table A ∩ B

Col1	Col2
1	1
1	2

table B

Col1	Col2
1	1
1	3

A ∩ B

Col1	Col2
1	1

Difference:-

A - B

Col1	Col2
1	2

B - A

Col1	Col2
1	3

Cartesian product:- This operation (A×B) allows us to combine information from any 2 relation.

table R_1

Rollno.	Name
1	Vijay
2	Gopal
3	Santosh

R_2

Subject
English
Maths

$R_1 \times R_2$

Rollno.	Name	Subject
1	Vijay	Eng.
1	Vijay	Maths
2	Gopal	Eng.
2	Gopal	Maths
3	Santosh	Eng.
3	Santosh	Maths

Rename (ρ):- Rename is unary operator used for renaming attribute of a relation.

$\rho_{a/b}(R)$

a & b are attribute Name

R → Relation Name

$\rho_{a/b}$ → New Name
Old Name

Employee →

Name	Emp-id
Vijay	11
Gopal	12

$\rho_{Emp\text{-}Name/Name}$ (Employee)

Emp-Name	Emp-id
Vijay	11
Gopal	12

Relation Name. (To change Relation ~~attribute~~ Name)

- $\rho_S^1(R)$
- $\rho_{S(a_1, a_2 \dots a_n)}(R)$

- $\rho_{a/b}(R)$ (To change Attribute Name)

In Query:-

Alter table employee Rename Names to Emp_name;

Rename Employee to Employee1 (To change old New table Name)

Division operator:- The symbol of this operator is used for select "for all"

A =

SNo.	PNo.
S1	P_1
S1	P_2
S1	P_3
S1	P_4
S_2	P_1
S_2	P_2
S_3	P_2
S_4	P_2

B =

Phno.
P_2

C =

Phno.
P_2
P_4

D =

PNo.
P_1
P_2
P_4

A ÷ D

SNo.
S1

A ÷ B

SNo.
S_1
S_2
S_3
S_4

A ÷ C

SNo.
S1

Join Operator:- Join Operator statement is used to combine data from two or more tables based on common field blw them.

Different type of joins are:-

1. Inner join
2. Outer join ← Left join, Right join, Full "
3. Natural.

1. Inner join:-

Student

Roll-No	Name	Address	Phone	Age
1	HARSH	DELHI	xxxxxxxxxx	18
2	PRATIK	BIHAR	xxxxxxxxxx	19
3	RIYANKA	SILIGURI	xxxxxxxxxx	20
4	DEEP	RAMNAGAR	xxxxxxxxxx	18
5	SAPTARHI	KOLKATA	xxxxxxxxxx	19
6	DHANRAJ	BARABAJAR	xxxxxxxxxx	20
7	ROHIT	BALURGHAT	xxxxxxxxxx	18
8	NIRAJ	ALIPUR	xxxxxxxxxx	19

Student Course

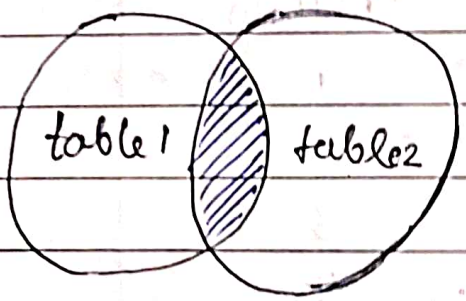
Course_ID	RollNo
1	1
2	2
2	3
3	4
1	5
4	9
5	10
4	11

Inner join:- Inner join keyword select all rows from both the table as long as condition is satisfied. This keyword will create the result set by combining all rows from both the table. where the condition satisfied i.e. value of the common field will be the same.

Syntax:-

select table1.column1, table1.column2, table2.column... from table1 Inner Join table2 (Inner Join: Keyword)

on table1.matching = table2.matching (matching: joining condition)



Display the name of the student who are enrolled in course.

Select Studentcourse.courseID, Student Name, from Student InnerJoin Studentcourse on student.Rollno = Studentcourse.Rollno

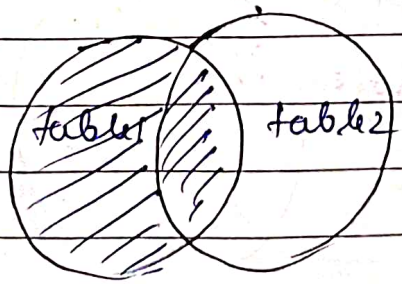
Course ID	Name
1	HARSH
2	PRATIK
2	RIYANKA
3	DEEP
1	SAPTARHI

Outer Join:- In Outer Join, along with tuples that satisfies the matching criteria, we also include some or all tuples that do not match the criteria.

It has 3 types:-

1. Left join:- This join returns all the rows of the table on the left side of the join. & matches rows for the table on the right side of the join. For the rows for which there is no matching on the right side result set will contain null value.

Syntax:- Select table1.column1, table1.column2, table2.column1, ... from table1 left join table2 on table1.matching = table2.matching

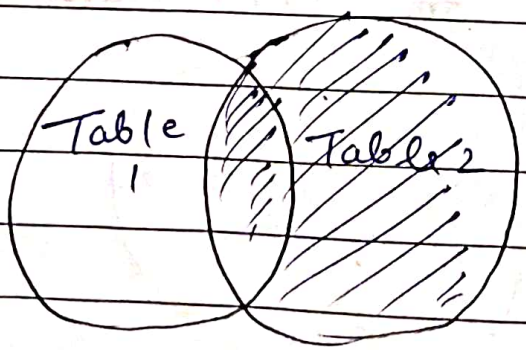


Select student.name, studentcourse.courseid from student Left join studentcourse on student.Rollno = studentcourse.Rollno.

Name	CourseID
Harsh	1
prodip	2
Riyanka	2
Deep	3
Saptarshi	1
Dhanraj	NULL
Rahit	NULL
Niraj	NULL

Right join:- It is same as left join. This joins returns all the rows on the right side table & matching rows for the table on the left side

Syntax:- Select table1.column1, table2.column2, table2.column1 - - - from table1 right join table2 on table1.Matching = table2.Matching.



Select student course. courseID, student. name, from student right join studentcourse on student. Rollno = studentcourse. Rollno

Name	CourseID
Harsh	1
pratik	2
Riya	2
Deep	3
Saptarshi	1
NULL	4
NULL	5
NULL	4

FULL Join:-

Syntax:- Select table1. column1, table1. column2, table2. column1 ... from table1 FULL Join table2 on table1. Matching = table2. Matching.

Select studentcourse. courseID, student. name, from student FULL Join student course on Student. Rollno = studentcourse. Rollno

Name	Course ID
Harsh	1
Prolik	2
Riya	2
Deep	3
Saptarshi	1
Dhanraj	NULL
Rohit	"
Niraj	"
NULL	4
"	5
"	4

Natural join:- It can join tables based on common values in the tables. It returns all rows by matching values in common column having same name & datatype of column

Ex:- Employee

Emp-id	Emp-Name	Dept-id
1	ABC	10
2	EFG	30
3	XYZ	50

Depart

Depart-id	Depart-Name
10	IT
30	HR
40	CS

Select * from Employee natural join Department

Emp-id	Emp-name	Dept-id	Dept-name
1	ABC	10	IT
2	EFG	30	HR

Joins in Relational Algebra:-

Theta join:- $R_1 \bowtie_{<\text{join condition}>} R_2$

Inner join has three types:-

1. Theta join
2. Equi Join
3. Natural Join

- It combines tuples from different relation provided, they satisfies θ (condition).
- It is a general case of join & it is used when we want to join two or more relation based on some condition.
- Join condition is denoted by θ.
- It uses all kind of comparison operator like >, <, >=, <=, =, ≠.

Syntax:- $A \bowtie_\theta B$

$$A \bowtie_\theta B = \sigma_\theta (A \times B)$$

S1

Sid	Name	Rating	age
22	ABC	7	45
31	EFG	8	55
58	XYZ	10	35

R1

Sid	bid	day
22	101	10/10
58	103	9/12

$S1 \bowtie_{(S1.sid < Ri.sid)} R1$

↓

θ

sid	Name	Rating	age	R.sid	bid	day
22	ABC	7	45	58	103	9/12
31	ABC	8	55	58	103	9/12

2. Equi Join:- when θ (theta) Join uses only equality operator (=), it becomes equijoin. Special case of theta Join where condition contain equals to operator.

sid	Name	Rating	Age	R.sid	bid	day
22	ABC	7	45	22	101	10/10
58	XYZ	10	35	58	103	9/12

$S1 \bowtie_{S1.sid = Ri.sid} R1$

3. Natural Join:- It is same as equi join which occurs implicitly by comparing all the common attribute, bcz difference is that in Natural Join common attribute appears only once.

$S_1 \bowtie R_1$

Course

Cid	course	Dept
CS01	Database	CS
ME01	Mechanics	ME
EE01	Elect	EE

HOD

Dept	Head
CS	Rohan
ME	Rasa
EE	Jiya

Course ⋈ HOD

CID	course	Dept	Head
CS	Data	CS	Ro
ME	Mech	ME	S
EE	Elec	EE	J