DBMS

a) Defining
b) Constructing
c) Manipulating
d) Sharing

Q. What do you mean by data?
Q. What is information?
Q. Define database.
Q. What do you mean by database management system?
Q. What are the advantages of DBMS?
Q. Differentiate traditional file system and database management system.

Answers

Ans (1). Data refers to different pieces of information, that has been translated into a form that is efficient for movement or processing.

Ans (2). Information is processed, organized and structured data.

Ans (3). A database is an organized collection of related data stored and accessed electronically.

Ans (4). Database Management Systems are software systems used to store, retrieve and run queries on data.

Ans (5). The advantages of DBMS are:

i) A DBMS may store and retrieve data in a series of manners.

ii) A DBMS is a reliable manager for handling numerous applications requiring identical data.

iii) Data Integrity and Security are guaranteed by DBMS.

iv) Data can also be shared to other systems through a DBMS.

Ans (6):-

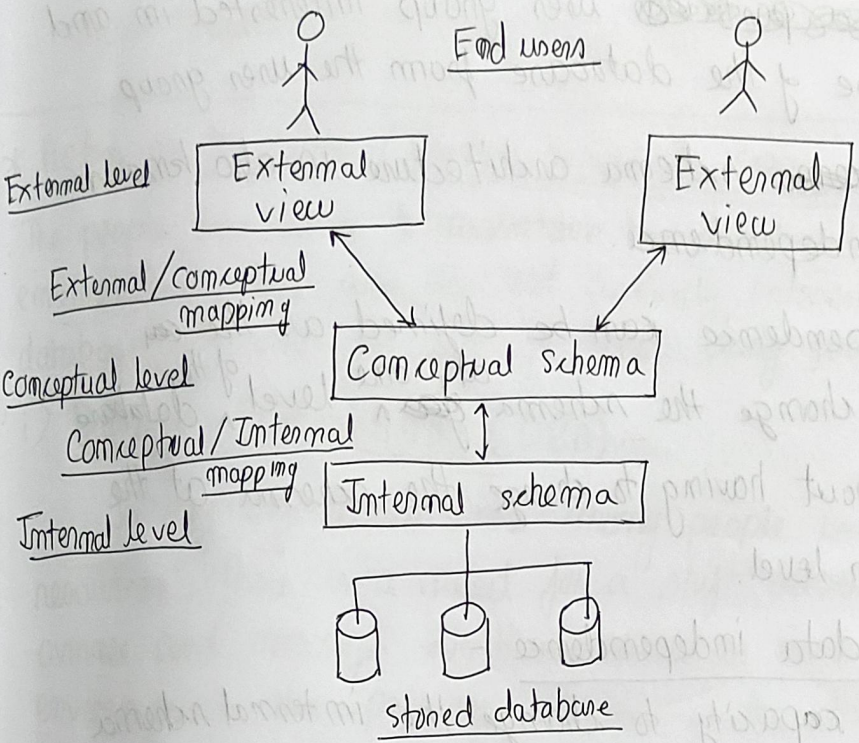
File system	DBMS
i) Software that manages the data files in a computer system.	i) Software to create and manage databases.
ii) Helps to store a collection of raw data files into the hard disk.	ii) Help to easily store, retrieve and manipulate data in a database.
iii) Has data inconsistency.	iii) Provides higher data consistency using normalization.
iv) Handling is easy.	iv) Handling is complex.
v) Eg:- NTFS and Ext.	v) Eg:- My SQL, Oracle, etc.

---

* Database Schemas & Instances / snapshot :-

- The description of a database is called the database schema which is specified during database design and is not expected to change frequently.
- The data in the database at a particular moment in time is called database state or snapshot. It is also called the current state of occurences or instances in the database.

* Three schema Architecture / Database Architecture :-



i) Internal level :- The Internal level has an internal schema which describes physical storage data.

ii) Conceptual level :- Conceptual level has a conceptual schema which describes the structure of a whole database for a community of users. The conceptual schema hides the physical storage structure and concentrate on description entities, data types, relationships, user ope separation and constraints.

iii) External level :- The External / view level inputs a no. of external schemas or user views, each external schema describes the part of the database that a particular

~~interested in and hides the part of the~~ database that particular user group interested in and hides the base of the database from the user group.

Three ~~schema~~ schema architecture in also known as data independence.

- Data independence can be defined as the capacity to change the schema at one level of the database system without having to change the schema at the next higher level.

i) Physical data independence :-

It is the capacity to change the internal schema without having to change the conceptual schema.

ii) Logical data independence :-

It is the capacity to change the conceptual schema without having to change the external schemas / application programs.

Syntax :-

- CREATING DATABASE

```
CREATE DATABASE database_name;
USE database_name;
CREATE TABLE table_name (column_name1 datatype (size),
column_name2 datatype (size), ... );
```

- To see every database available, syntax :- SHOW DATABASE;

- To see a particular table within a database, syntax:-

  SHOW TABLES

---

* Actors on the scene / Workers behind the scene :-

The people who work to maintain the database system environment but who are not actively interested in the database contents as part of their daily job.

i) Database Administrator (DBA):-

In any organization where many people use the same resources, there is a need for a skilled administrator to oversee and manage these resources in a database environment the primary resource is the database itself and the secondary resource is the DBMS and related software. Administering is the responsibility of the Database Administrator. The DBA is responsible for authorizing access to the database, co-ordinating and monitoring its use and acquiring software and hardware resources as needed.

ii) Database Designer :-

So " " are responsible for identifying the data to be stored in the database and for choosing appropriate data structures to represent and store these data.

iii) End Users :-

" " are the people whose jobs require access to the database for querying, updating and generating reports. The " primarily exist for their use.

a) Casual End Users :-

" " " occasionly access the database, but they may need different information each time.

b) Naive or parametric end Users :-

" " " " " make up a sizeable portion of database end Users, their main job functions revolve around constantly querying and update the database, using standard types of queries and updates called canned transaction, that have been carefully programmed and tested.

c) Sophisticated end Users :-

" " " include engineers, scientist, business analist and others who thoroughly familiarized themselves with the facilities of the DBMS in order to implement their own applications to meet their complex requirements.

d) Standalone Users :-

" " maintain personal databases by using readymade program packages that provides easy to use menu base or graphics based interfaces.

iv) System analysts & Application programmers :-

" " determine the requirements of end user, especially naive & parametric end Users and develop specification for standard canned tr~~ansa~~ctions

transaction that need this requirement.

Application Programmer

---

* Entity :-

An entity is a thing or object in the real world with an independent existence. An entity maybe an object with a physical existence. (For eg:- A particular person, car, house or employee) or it may be an object with an conceptual existence (For eg - A company, A job or a university course).

---

* Attributes :-

" means properties of an Entity.

---

* The Attributes of a student entity are Roll no., phone number, age, gender, etc.

---

* Types of Attributes :

i) Composite vs Atomic (Simple) Attribute

Composite Attributes can be divided into smaller sub parts which represent more basic attributes with more independent meanings. For eg: Name of a person, Name can be divided into first name, middle name, last name.

Attributes that are not divisible are called simple or Atomic attributes. For eg: Phone number.

(ii) Single valued vs multi-valued attribute :

Most attributes have a single value for a particular entity such " are called single value. For eg. Age of a entity (student, employee, etc).

The attributes which have more than one value for a particular entity is called multi-valued attribute. For eg - Degree of a person, etc.

(iii) Stored vs derived attribute :-

Some attribute values can be derived from another another attributes. The attribute values which is derived is called derived " and from where it is derived is called stored attribute. For eg - Date of birth of a person is a stored attribute and the age of the person is a derived attribute.

(iv) Complex attribute :-

" " is the combination of two or more other attributes. For eg - {Address_phone({phone(Area-code, Phone number)}, Address(Street_address(Number, Street, Apartment_number), city, state, zip))}

(v) Null valued attribute :-

In some cases a particular entity may not have an applicable value for an ~~attribute~~ entity such attribute is called Null value attribute. For eg - PAN number of a person.

(vi) Key attributes :-

An important constraints on the entities of an entity type is the key or uniqueness constraints on an attribute. An entity type usually has one or more attributes whose values are distinct for each individual entity in the entity state, such an attribute is called a key attribute and its values can be used to identify each entity uniquely.

---

* Relationship type, set & Instances :-

A " " 'R' among 'n' entity types $E_1, E_2, \ldots, E_n$ defines a set of association or a relationship set among entities from this entity types.

Similar to the concept of entity types and entity sets a relationship type and its corresponding relationship set are customarily called by the same name 'R'.

---

* Command (my SQL) :-

Select * from Student where Rollno = 2;

[For showing specific row]

delete from tablename;

[For deleting the details inside a table]

delete from tablename where Rollno = "xx";

Employee

eid	name	address	Salary
101	Anup	Ghy	20000
102	Ajay	Tezpur	10000
103	Binay	Nagaon	25000
104	Gautam	Ghy	20000

1Q. Create the above given table

2Q. Display all the details of the employees.

3Q. " " " " " " who lives in ghy.

4Q. Display eid, address and Salary for all the employee.

5Q. Delete the details of the employee whose eid is 103

6Q. Display the name address of the emp whose name is Anup.

7Q.

Employee 1

Eno.	name	Age	Salary
1	aa	30	20000
2	ab	35	25000
3	bb	32	23000
4	cc	40	40000

select * from Employee1 where Eno. in (1, 3, 4);

select * from Employee1 where Eno. between 20 and 30;

Select * from Employee1 where Eno = 3 and Eno = 2;

" " " " " Eno. = 3 on Age = 40;

" " " " " Eno = 2 on 3 on 4;

" " " " " Salary > 10000;

Modification

Ud update Employee1 set name = "ac" where Age = 30;

Agregate function:

sum, min, max, count, avg

i) Select sum (salary) from Employee1;

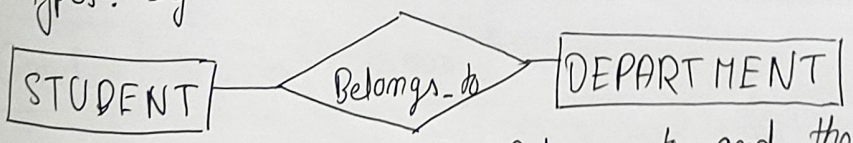
sum
↓
min
↓
max
↓
avg

i) Select count (*) from Employee1;

---

* Degree of a Relationship type :-

The " " " " is the no. of participating entity types. Eg:



Here, the relationship type is Belongs_to and the degree of the relationship type 'Belongs_to' is 2, since 2 entities are participating

- A relationship type of degree 2 is called binary, and one of degree 3 is called ternary

---

* Cardinality Ratio :-

- " " for binary relationship :-

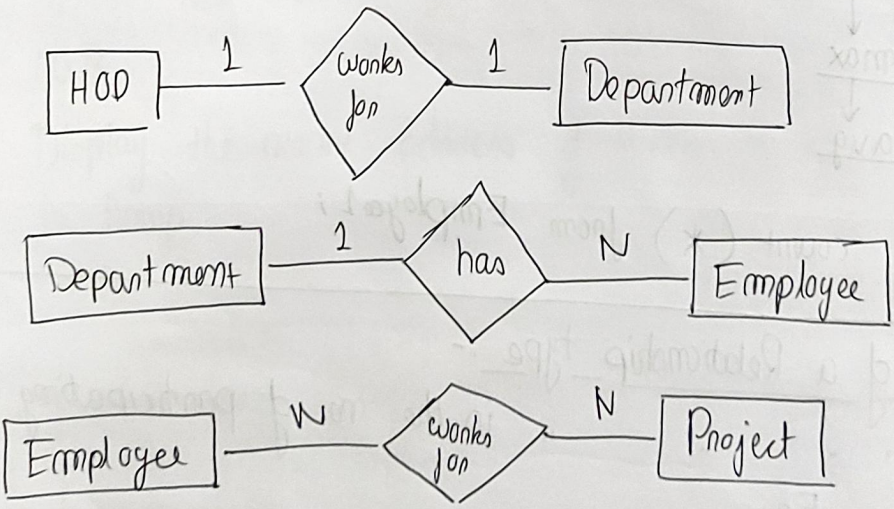
The " " " a " specifies the maximum no. of relationship instances that an entity can participate in.

- Types of Cardinality Ratio :

i) 1 : 1 (One to One)

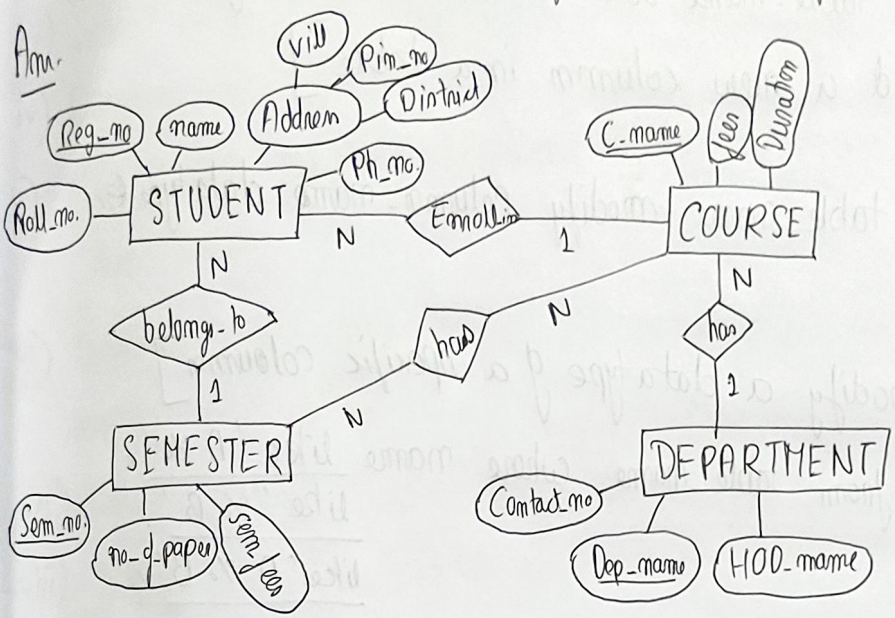
ii) 2 : N (One to many)

iii) N : N (many to many)



Q. Draw the E-R diagram for student information system.

Ans.



STUDENT

Roll_no	Reg_no	name	Addrem	vill	pin	by_lame	Dintrict	ph_no.	sem_no.	C_name

SEMESTER

Sem_no	no_of_paper	sem_fees	

COURSE

C_name	Total_fees	Duration	Dep_name

DEPARTMENT

Contact_no	Dep_name	HOD_name

- Alter table table_name add column_name datatype (size);

  [To add a new column in a table]

- Alter table table_name modify column_name datatype (size);

  [To modify a datatype of a specific column]

- Select * from table_name where name like 'A%';
  like '%B';
  like 'A%B';
  like 'A__';
  like '___Y';

STUDENT

- Select distinct column_name from table_name;
- Select * from table_name order by column_name;
- Select * from table_name " " column_name desc;

EMPLOYEE

emp_no	name	address	ph_no

i) Create the above employee table.

ii) Add a new column salary to the table.
iii) Insert salaries for all employee
iv) Display max salary
v) " the details of employee in the ascending order of their salary
vi) Display name and salary of all employee whose name starting with R.
vii) Delete employee details whose name ending with y.
viii) Display name, emp_no and ph_no whose name starting with 'A' and ending with 'Y'

@

* Keys :-

i) Superkey :

A set of one or more attributes when taken together helps in uniquely identifying each ~~id~~ entity is called a superkey.

ii) Candidate key :

A minimal superkey that does not contain any super-fluous (extra) attribute in it is called a candidate key. A candidate key contains a minimized set of attribute that can be used for uniquely identify a single entity instances. All candidate key are superkey but not all superkey is a candidate key.

iii) Primary key:

A candidate key that is chosen to uniquely identify a tuple in a relation is known as primary key.

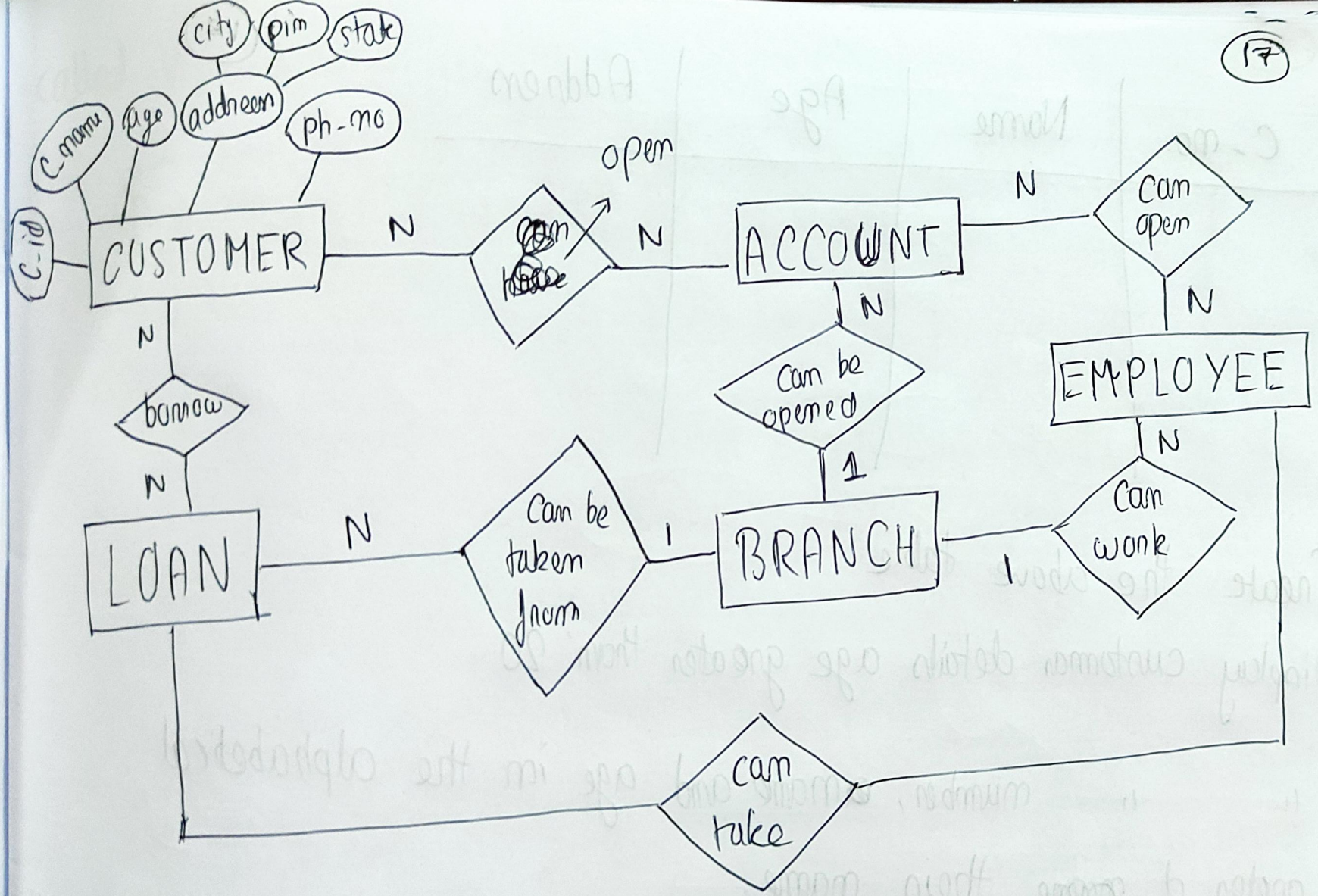
Other candidate keys that are not chosen as primary key are known as alternate key.

A primary key is a candidate key also superkey but all superkeys or candidate keys are not primary keys.

iv) Foreign key:

Attribute of one relation can be accessed in another relation by enforcing a link bet^n the attributes of two relations, this can be done by defining the attribute of one relation as the foreign key refers to the primary key to the another relation.

Q. Consider a bank database having customer, loan, account, employ and branch as entity types each branch of bank allows customers to open accounts and borrow loans. A customer can open more than 1 account and 1 account can belong to more than 1 customer. Similarly, a customer may take out more than 1 loans and 1 loan can be held by more than 1 customer. The Bank has a number of employee working in different branches. Add appropriate attributes for each entity type. Represent the key attribute and show the cardinality ratio. Design an ER diagram for bank details.


city
pin
state
address
C_name
age
ph-no
C_id
CUSTOMER
N
open
N
ACCOUNT
N
Can open
N
EMPLOYEE
N
borrow
N
N
Can be opened
1
N
Can work
1
LOAN
N
Can be taken from
1
BRANCH
can take

C_no.	Name	Age	Addres

i) Create the above table

ii) Display customar details age greater than 20

iii) " " number, name and age in the alphabetical order of their name.

iv) Display C_number, name, age and address in the descending order of their age

v) Remove Customar details whose name start with 'B' followed by exactly 3 characters

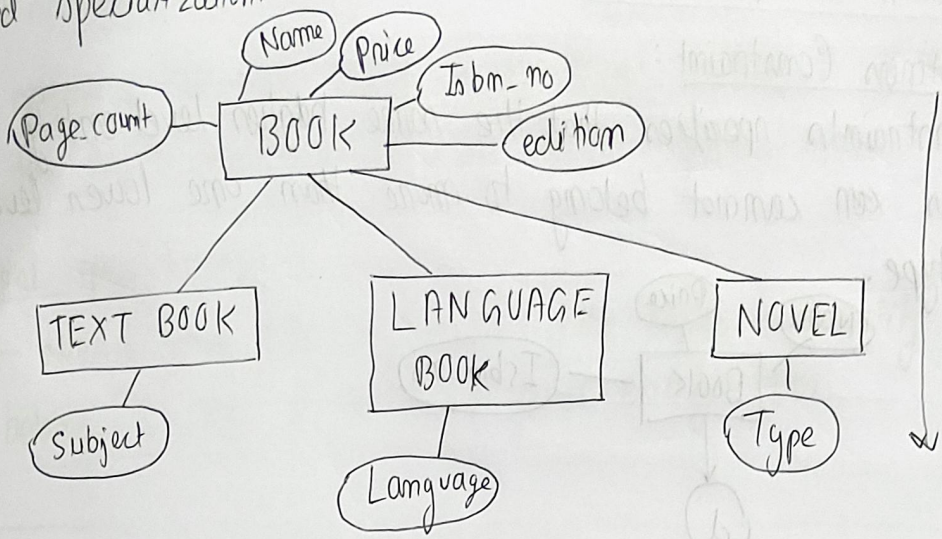
vi) Display all the addresses from the given table and remove the duplicate address.

---

* Specialization & Generalization:

An entity type may include sub groups of its entities in such a way that entities of one sub group are distinct in some way from the entities of other sub groups. For eg:- the entity type 'book' classified into three types. namely text book, language book and Novel. There entity types are described by a set of attributes that include all the attributes of the entity type 'book' and some additional. There process of defining a sub-group of a given entity type is

called specialization.



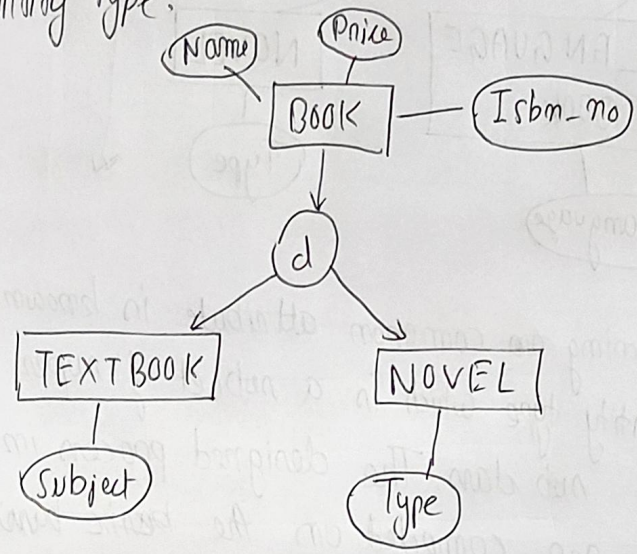
The entity type containing a common attribute is known as the super class and the entity type which is a subset of super class is known as its sub class. The designed process in which multiple lower level are compiled on the basic basis of common features to form higher level entity type is known as generalization. For eg:- The database designer may first identify the entity types textbook, language book & novel and then combine the common attributes of the entity types to form a higher level entity types books.

Specialization starts with single higher level entity type and ends with a set of lower level

* Constraints on Specialization :-

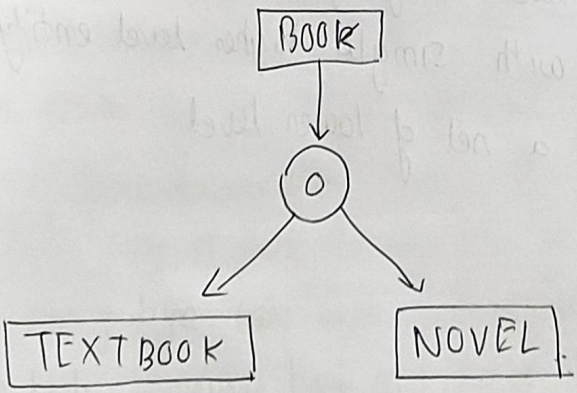
i) Disjointness Constraint :

This constraints specifies that the same highen level entity instances can cannot belong to more than one lower level entity type.



ii) Overlapping Constraint :

This " specifies that the same highen to level entity instances can belong to more than one lower level entity type.



iii) Completeness :

a) Total specialization.

b) Partial "

- Total specialization:

It specifies that each higher entity must belong to at least to one of the lower level entity type in the specialization.

- Partial specialization:

It allows some of the instances of the higher level entity type not to belong to any of the lower level entity types.

---

* Relational algebra :-

" algebra in associated with relational model that are used to specify the basic set of request. Relational algebra consists of a basic set of operation which can be used for carrying out basic.

i) Unary operations
ii) Binary operations

* Unary operations :-

The " operating on a single relation are known as unary operations. The various unary operations in relational algebra are :- select, project and Rename.

- Select :-

The " operation retrieves all those tuples from a relation which satisfy a specific conditional. The greek letter (σ) is used as a select operator.

STUDENT

Roll No.	Name	Address	Age
1	aa	Ghy	20
2	ab	Tezpur	21
3	bb	Nalbari	20
4	ba	Ghy	21

i) Display student details whose rollno is 1

Soln

$$\sigma_{rollno = 1}(STUDENT)$$

Rollno	Name	Address	Age
1	aa	Ghy	20

- Project operation :-

The " " is used to select the required attributes from a relation while discarding the other attributes. The greek letter (π) is used as project operation.

ii) Display name and address for all employees.

Soln.

$$\pi_{name, address}(STUDENT)$$

Name	address
aa	Ghy
ab	Tezpur
bb	Nalbari
ba	Ghy

iii) Display student details whose address is malbari

iv) " " " " name is aa and address is ghy.

v) " " " who lives in either ghy or tezpur.

vi) " Rollno, address and age for all students.

vii) " " and " for the student whose age is 20

iii)

Sol^n -

$\sigma_{address = "malbari"}(STUDENT)$

['V' is or symbol
'^' is and symbol

Rollno	Name	Address	Age
3	bb	Malbari	20

iv)

Sol^n.

$\sigma_{name = "aa" \wedge address = "ghy"}(STUDENT)$

Rollno	Name	Address	Age
1	aa	ghy	20

v)

Sol^n.

$\sigma_{address = "ghy" \vee address = "Tezpur"}(STUDENT)$

Rollno	Name	Address	Age
1	aa	ghy	20
2	ab	Tezpur	21
4	ba	ghy	21

vi)

Sol^n :-

$\Pi_{rollno, address, age}$ (STUDENT)

rollno	address	age
1	ghy	20
2	tezpur	21
3	nalbari	20
4	ghy	21

v)

Sol^n -

$\Pi_{rollno, address}$ ($\sigma_{age = 20}$ (STUDENT))

- Rename operation :-

" " is used to provide a name to the relation obtained after applying any relational algebra operation. The greek letter 'ρ' is used as a rename operator.

ρ(R, E) where 'ρ' is the rename operator, 'E' is the expression representing relational algebra and 'R' is the name given to relation obtain by applying relational algebra operation specified in expression 'E'.

---

ii) Binary operation :-

The " operating on two relation are known as binary

operation, the various binary operation in relational algebra are:- set, join and division operation.

Various set operation in relational operation are union, intersection, difference and cartesian product.

- The union operation denoted by 'E' returns a third relation that contains tuples from both or either of the operand tables. It is denoted by $E = R_1 \cup R_2$

STUDENT2

Rollno	Name	Age
1	aa	20
3	ab	21
4	bb	20

STUDENT1

Rollno	Name	Age
1	aa	20
2	ab	21

STUDENT 1 ∪ STUDENT 2

Rollno	Name	Age
1	aa	20
2	ab	21
3	ab	21
4	bb	20

- The intersection operation returns a third relation that contains tuples common to both the operand relation. The intersection of relation R_1 and R_2 is denoted by $R_1 \cap R_2$

STUDENT 1 ∩ STUDENT2

Rollno	Name	Age
1	aa	20

- The difference operation denoted by '−' returns a third relation that contains all tuples present in one 1st relation which

not present in 2nd relation. The difference of Relation R_1 and R_2 is denoted by $R_1 - R_2$

STUDENT 1 - STUDENT 2

Roll no	Name	Age
2	ab	21

- The cartesian product denoted by the symbol '×' and returns a third relation that contains all possible combinations of the tuples from the two operand relation. The cartesian product of relation R_1 and R_2 is denoted by $R_1 \times R_2$

DEPARTMENT

Dno.	Dname
101	CS
201	Mgmt

STUDENT 1 × DEPARTMENT

Roll no.	Name	Age	Dno	Dname
1	aa	20	101	CS
1	aa	20	201	Mgmt
2	ab	21	101	CS
2	ab	21	201	Mgmt

Create table student (Roll no int primary key, Name varchar(20), Address varchar(20), Dno. int, foreign key (Dno) references Department (Dno.));