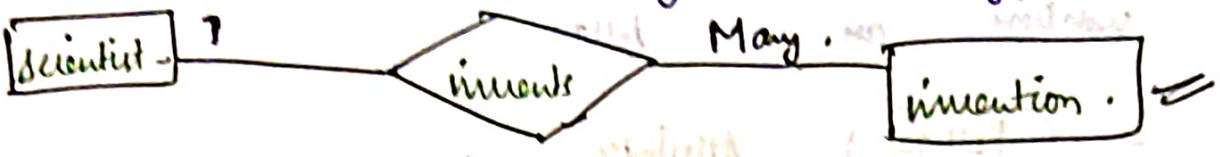
known as a one to many Relationship.

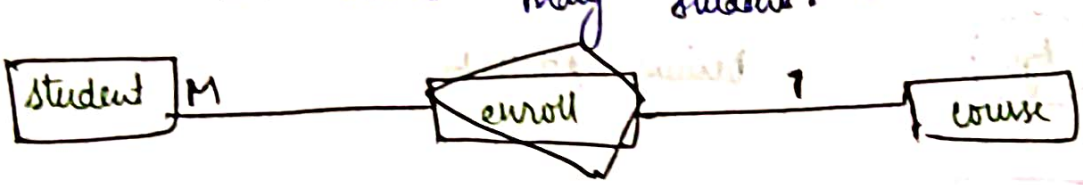
For example :- Scientist can invent many inventions, but the many inventions is done by the only specific scientist.



(iii) Many to one Relationship

Is when more than one instance of the entity on the left and only one instance of an entity on the right ~~Accessin~~ Associates with the Relationship it is known as Many to one Relation.

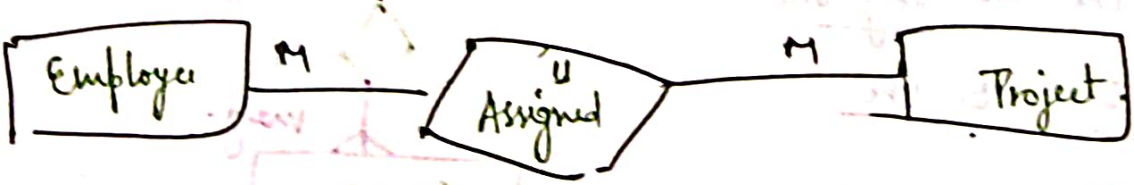
For example :- student enroll for only one course, but a course can have many students.



(iv) Many to Many Relationship

Is when more than one instance of the entity on the left and more than one instance of an entity on the right associates with the relationship then it is known as a many to many relationship.

For example :- Employe can Assign by many projects and Project can have many employes.



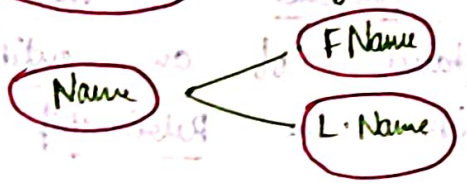
Notation of ER Diagram

↳ DB can be represented using the relations in ER diagram, many notations are used to express the cardinality.

These notations are as follows

Attributes

Key Attributes

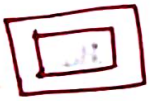


Composite Attributes

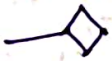
Multivalue attributes

Derived Attributes

strong entity set



weak entity set



Relationship

Links (one to one)

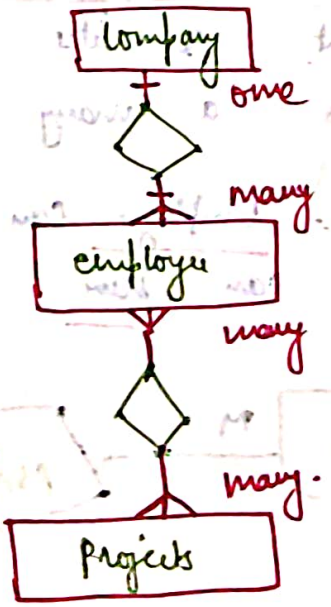
one to many

many to one

one and only one

Zero or one

Zero or many.



Keys in DBMS.

↳ A key is a value which can always be used to uniquely identify an object instance.

↳ Key is used to identify identify any record or row of data from the table.

↳ It is also used to establish and identify relationship b/w tables.

For example :- Id is used as a key in the student table because it is unique for each student. In the person table, passport number license number are keys since they are unique for each person.

Types of keys

super key | candidate key | Primary key | Compaik key | Foreign key.

1. super key

↳ A super key is a set of one or more attributes that taken collectively, allow us to identify uniquely an entity in the entity set.

$${}^nC_r = \frac{n!}{r!(n-r)!}$$

$${}^5C_1 + {}^5C_2 + {}^5C_3 + {}^5C_4 + {}^5C_5$$

For example :- In student table with Attribute (S_Roll no, S-Name, S-Branch, S-Year)

Super key ⇒ S_1 → S_Roll no, S-Name
S_2 → S-Roll no, S-Branch
S_3 → S-Roll no, S-Year.

Max super key $\rightarrow 2^n - 1$

(ii) Candidate key

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

↳ Candidate key can be defined as the min. no. of super key that identifies the record uniquely

↳ It must contain unique values.

↳ Every table must have at least a single candidate key.

For ex:- In student table with attribute (S_Rollno, S_Name, S_Branch, - S_Year)

Candidate key :- $C_1 \Rightarrow$ S_Roll no.

$C_2 \Rightarrow$ S_Roll no, Sname.

{super key whose proper subset is not a super key.... Jeevis lecture}

(iii) Primary key

↳ It is defined as the min. no. of candidate key that is chosen by the dB designer as the principle means of identifying entities within an entity set.

↳ It is a unique key.

↳ It can identify only one (row) tuple (~~are cord~~) at a time

↳ It has no duplicate values, it has only unique values.

↳ It cannot be null

↳ Primary keys are not necessarily to be a single column, more than one the column can also be a Primary key for a table.

For ex:- In student table with attribute (S_Roll_no, S_Name, S_Branch, S_Year)

Primary key (PK) ⇒ PK ⇒ S_Roll_No., S_Id

(iv) Composite key

↳ Whenever a PK consists of more than one attribute, it is known as a composite key.

For ex: In students table with attribute (S_Roll_no, S_Name, S_Branch, S_Year)

Composite key ⇒ S_Roll_no, S_ID. (Both in same table are primary key)

(v) Foreign key

↳ A Foreign key is a column whose value are the same as the primary key of another table.

↳ It combines two or more relations (table) at a time.

↳ They act as a cross reference b/w the tables.

↳ Foreign key are the column of the table used to point to PK of another table.

Primary key

Foreign key

Roll no	Name	ID	(F.K)	ID	Branch	address
1	Piyush	143		234	IT	Kalroy
2	Avinash	184	Foreign key	416	CSE	Baylor
3	Santosh	156		768	EC	Titeland

DBMS Generalization → It is a relationship

↳ Generalization is like a bottom up approach in which two or more entities of lower level combine to form a higher level entity if they have some Attributes in common.

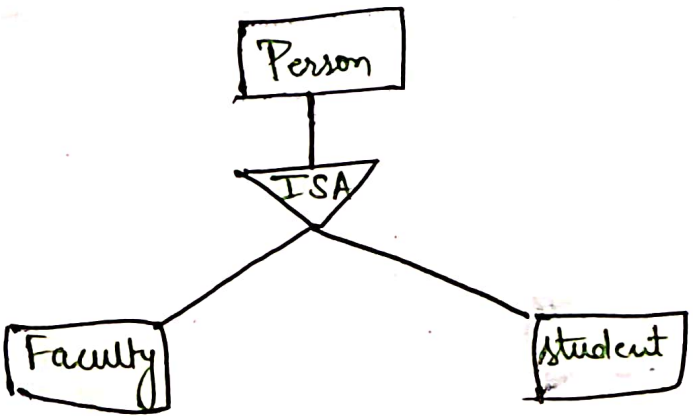
↳ Generalization is a relationship that exists b/w a high level entity set and one or more lower-level entity set.

↳ Generalization is more like subclass and superclass system, but the only difference is the approach. Generalization uses the bottom up Approach.

↳ In Generalization entities are combined to form a more generalized entity such as subclass are combined to make a superclass.

↳ Generalization is represented by a triangle component labeled ISA. The label ISA stands for "is a" and represents.

For example :- Faculty and Student entities can be generalized and crate a higher level entity person.

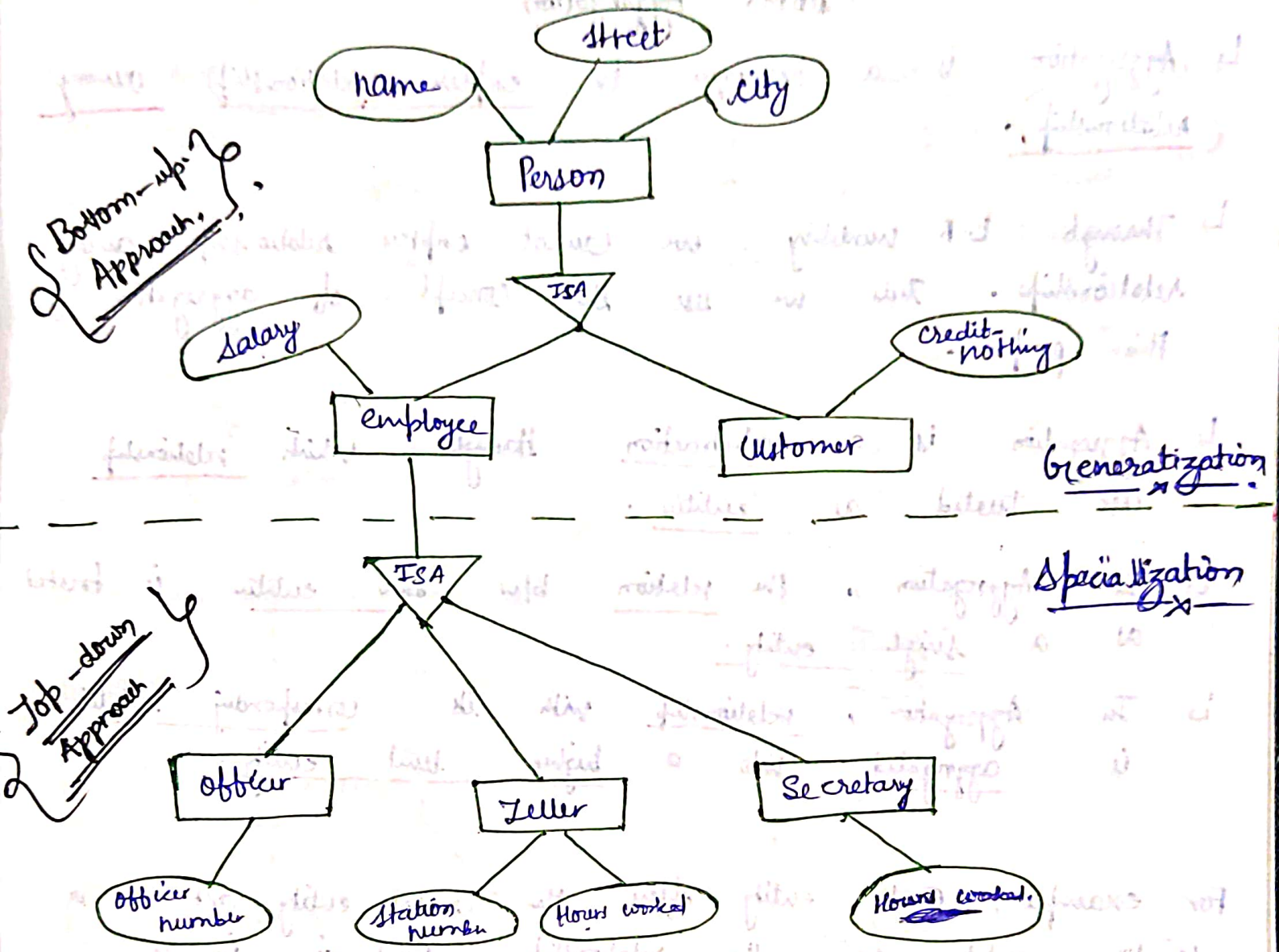


Generalize → 

DBMS specialization

- Specialization is a top down approach, and it is opposite to generalization. In specialization one higher level entity can be broken down into two lower level entities.
- Specialization is used to identify the subset of an entity set that share some distinguishing characteristics.
- The specialization Normally, the super class is defined first, the subclass and its related attributes are defined next and relationship set are then added.
- Specialization is represented by a triangle component labeled ISA. The label ISA stands for "is a" and represents-

For ex:- In the employee management system, employee entity can be specialized as officer, Teller, Tester, Developer. Secretary and developer based on what role they play in the company.



ER diagram with generalisation and Specialization

Kujush ...

DBMS Aggregation

↳ Aggregation is a technique to express relationship among relationship.

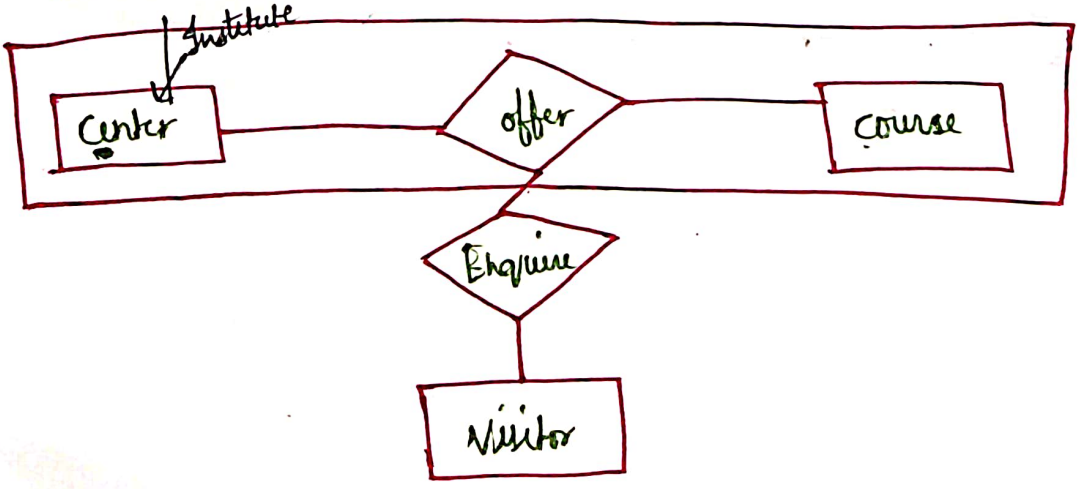
↳ Through E-R modeling we cannot express relationship among relationships. Thus we use the concept of aggregation for this purpose.

↳ Aggregation is an abstraction through which relationship are treated as entities.

↳ In Aggregation, the relation b/w two entities is treated as a single entity.

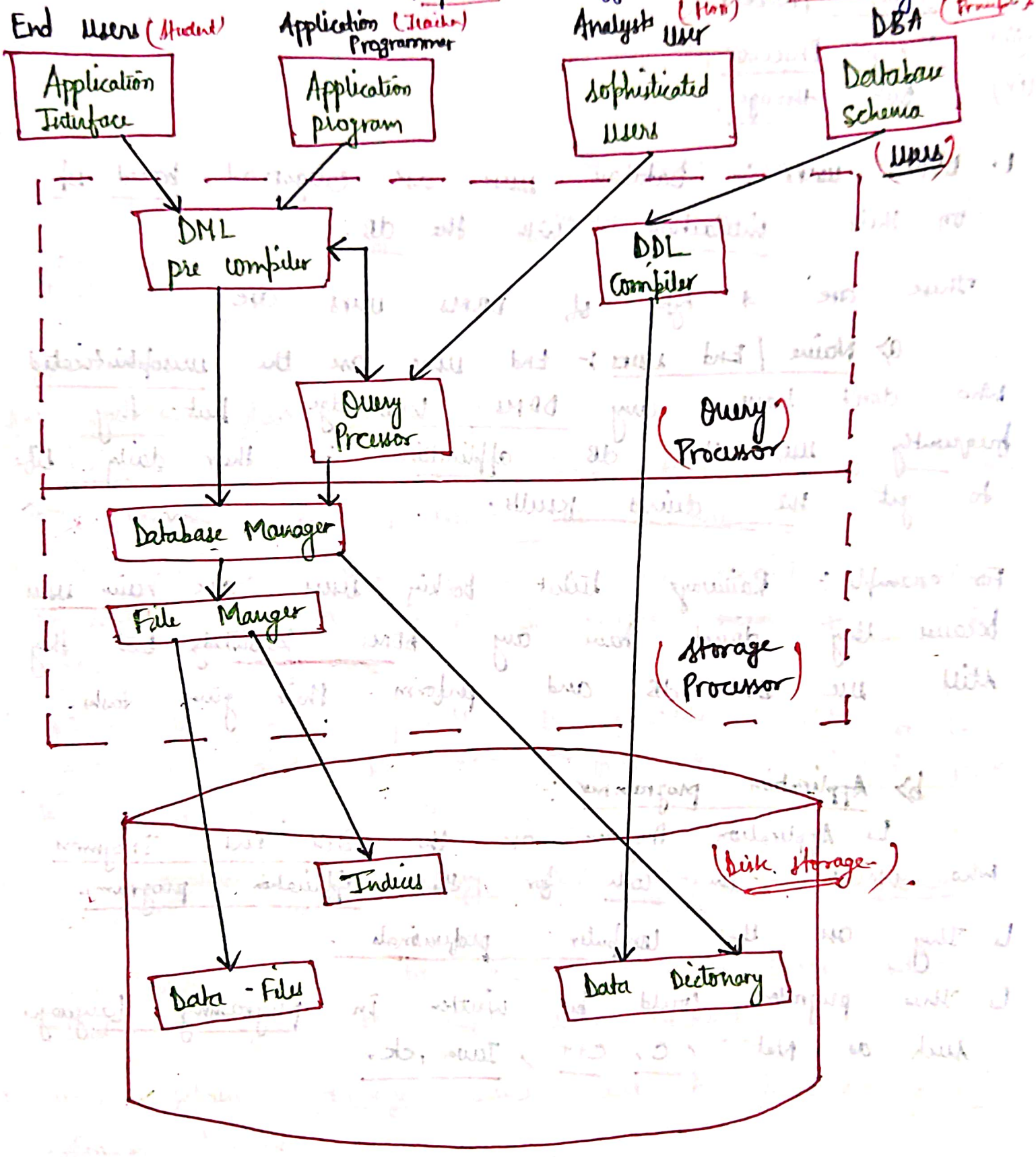
↳ In Aggregation, relationship with its corresponding entities is aggregated into a higher level entity.

For example: Center entity offers the Course entity act as a single entity in the relationship which is in a relationship with another entity Visitor. In the real world, if a visitor visits a coaching Center then he will never enquiry about the Course only or just about the Center instead he will ask the enquiry about both.



DBMS Architecture / structure / component

↳ A database system is partitioned into modules that deal with each of the responsibilities of the overall system



DBMS Architecture divided into four parts

(i) DBMS users
(ii) Query Processor
(iii) Storage Processor
(iv) Disk storage

1. DBMS users :- Database users are categorized based up on their interaction with the DB.

There are 4 types of DBMS users are

a) Naive / End users :- End users are the unsophisticated who don't have any DBMS knowledge but they frequently use the DB application in their daily life to get the desired results.

For example :- Railways ticket booking users are naive users because they do not have any DBMS knowledge but they still use the DB and perform their given tasks.

b) Application programmer :-

↳ Application Program are the back end Progmers who write the code for the application program.

↳ They are the computer professionals.

↳ This program could be written in programming language such as Net, C, C++, Java, etc.

(c) Sophisticated users

↳ Sophisticated users can be engineers, scientists, business Analyst who are familiar with the DB.

↳ They can develop their own DB application according to their requirement.

↳ They don't write the Program code but they interact the DB by writing SQL Queries directly through the Query processor.

d) Database Administrator (DBA)

↳ DBA is a person/team who defines the schema and also controls the 3 levels of databases.

↳ The DBA will then create a new account id and password for the user if he/she need to access the DB.

↳ DBA is also responsible for providing security to the database and he allows only the authorized User to access/modify the DB.

↳ DBA monitors the recovery and backup and provide technical support.

↳ The DBA has a DBA account in the DBMS which called a system or super user account.

↳ DBA repairs damage caused due to hardware and/or software failures.

2. Query Processor

↳ It interprets the request (queries) received from end user via an application program into instructions.

↳ It also executes the user request which is received from the DML compiler.

Query Processor contains the following components

a) DDL compiler

↳ The DDL statements are sent to DDL compiler, which converts these statements to a set of tables.

↳ These tables contains the metadata concerning the DB and are in the form that can be used by other components of the DBMS.

b. DML Pre-compiler and Query Processor

↳ The DML pre-compiler converts the DML statements embedded in an application program to normal procedure calls in the host language.

The Query processor component include

(i) DDL interpreter

↳ It processes the DDL statements into a set of tables containing meta data (data about data)

(ii) DML compiler

↳ It processes the DML statements into low level instruction (Machine language), so that they can be executed.

(iii) Query Evaluation Engine

↳ which executes low-level instructions generated by the DML compiler.

3* Storage Manager / Processor

↳ Storage Manager is a Program that Provides an interface b/w the data stored in the dB and the Queries received.

↳ It is also known as dB control system

↳ It maintains the consistency and integrity of the dB by applying the constraints and executes the DCL statements.

↳ It is responsible for updating, storing, deleting and retrieving data in the dB.

It contains the following components

a. Authorization Manager

↳ It tests for the satisfaction of integrity constraints and checks the authority of users to access data.

b. Transaction Manager

↳ which ensures that the dB remains in a consistent (correct) state despite system failures, and that concurrent transaction execution proceed without conflicting.

c. File Manager

↳ which Manages the Allocation of space on disk storage and the data structures used to represent information stored on disk.

d. Buffer Manager

↳ It is responsible for cache memory and the transfer of data b/w the secondary storage and main memory.

4. Disk storage

↳ It contains the following components

a. Data file ↳ It stores the data.

b. Data dictionary

↳ It contains the information about the structure of any DB object.

↳ It is the repository of information that governs the Metadata.

c. Indices.

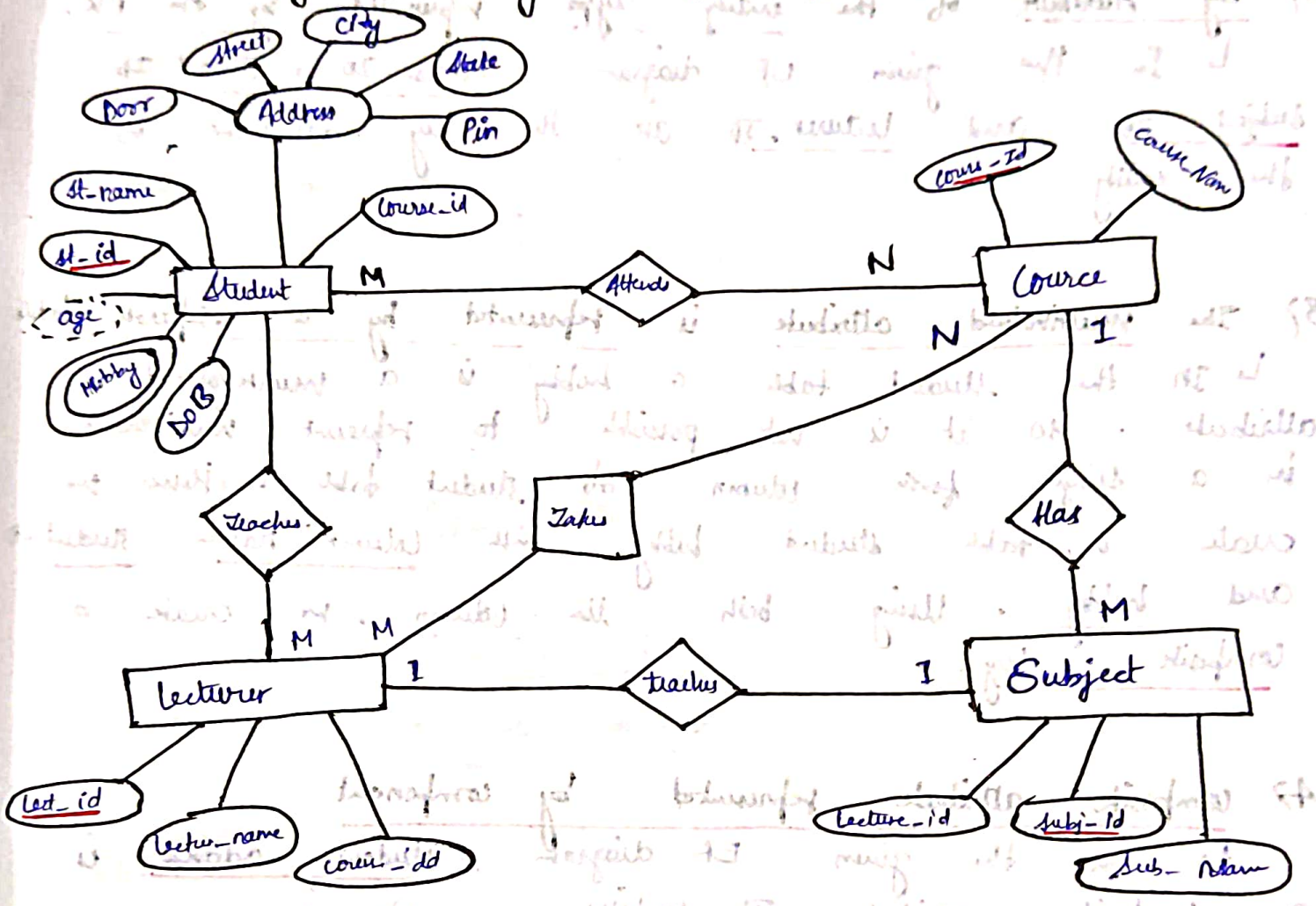
↳ It provide fast access to data items that hold particular values.

Reduction of E.R diagram to Table

↳ The dB can be represented using the notations and these notation can be reduced to a collection of tables.

↳ In the dB every entity set or relationship set can be represented in tabular form.

The E-R diagram is given below to convert in table



These are some points for converting the ER diagram to the table

1) Entity type become a table

↳ In the given ER diagram, lecturer, student, subject and course from individual tables.

2) Key Attribute of the entity type represented by the PK.

↳ In the given ER diagram course ID, student ID subject ID and lecturer_ID are the key attribute of the entity

3) The multivalued attribute is represented by a separate table

↳ In the student table a hobby is a multivalued attribute. so it is not possible to represent multivalues in a single form column of student table. Hence we create a table student hobby with column_name student_ID and hobby. Using both the column, we create a composite key.

4) composite attribute represented by component

↳ In the given ER diagram, student address is a composite attribute. It contains pin, and state. In the student table, these attribute can merge as an individual column.

5. Derived Attribute are not considered in the table

↳ In the student table, age is the derived attribute It can be calculated at any point of time by Calculating the difference b/w current date and DOB.

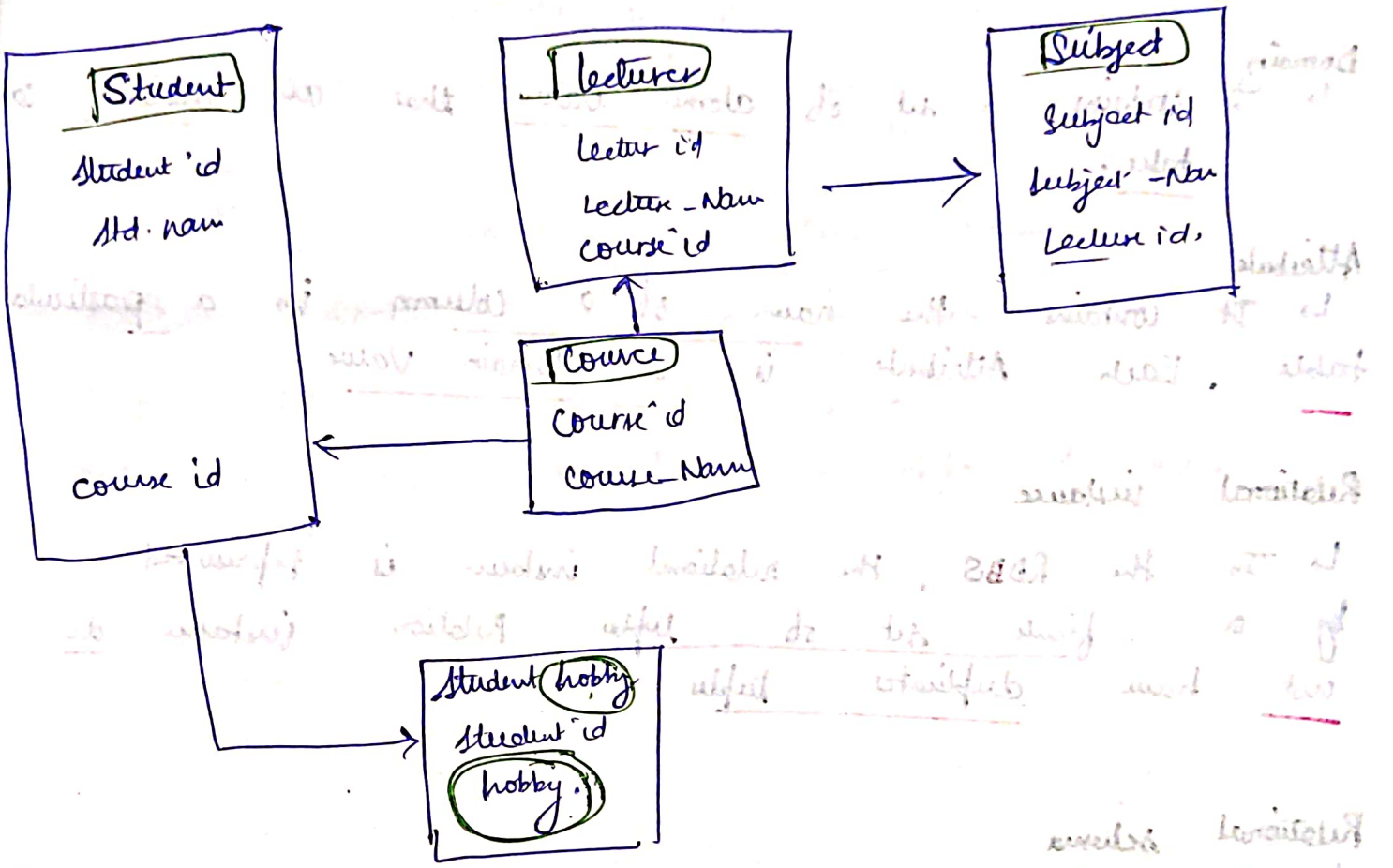


Table Structure

Relational Model concept

- Relational Model can represent as a table with column and rows.
- Each row is known as tuple.
- Each table of the column has a name or attribute.

Domain
- It contains a set of atomic value that an attribute can take.

Attribute
- It contains the name of a column in a particular table. Each Attribute is a domain value.

Relational instance
- In the RDBS, the relational instance is represented by a finite set of tuples Relation instance do not have duplicate tuples.

Relational schema
- A relational schema contains the name of the relation and name of all columns or Attributes.

Relational key
- In the relational key, Each row has one or more attributes. It can identify the row in the relation uniquely

ex:- student Relation

Name	Roll_no.	Phone no.	Address
Ram	123	34567	Delhi
Kamal	234	25678	dehradun
Kailash	345	13245	Haridwar.

↳ In the given table, Name, Roll no, Phone no are the attributes

↳ The instance of schema student has 3 tuples.

Properties of Relations

↳ Name of the relation is distinct from all other relations

↳ Each relation cell contains exactly one atomic (single) value.

↳ Each Attribute contains a distinct Name.

↳ Attribute domain has no significance

↳ Tuple has no duplicate value.

↳ order of tuple can have a diff. sequence.