10/9/21

PAGE NO: 1
DATE: / /

Unit → I

☆ Transaction :- ACID Property.

A → Atomacity
C → Consistency
I → Isolation
D → Durability

- A transaction is the unit of Program execution that access and Possibly updates various data base.

☆ Properties of ACID :-

I) Atomacity :- The data base System keeps track of the values of any data on which a transaction Performe a write operation and it the transaction doesn't complete its execution the data base System keeps hold the whole values to make it appear the transaction never executed

II) Consistency :- Execution of transaction in isolat -ion. that is with no other trans -action executing concurrently preserves the consi -ency of the data base.

III) Isolation :- Even though multiple transaction may execute concurrently the System guarantees that for every pair of transaction - Ti & Tj, it appears to Ti that either Tj

finish execution before Ti started or Tj that execution after Ti finish.

IV) Durability :- One the transaction complete Successful. all the updates that is carried out data base Persist, even if there is a system failure after the transaction complete execution.

☆ DBMS :-

1) Database :- A database is collection of relative data which represent some except of the real word.

- A database System is design to be build and Populated with data for the certain class.

2) DBMS :- DBMS is a Software for storing and retrieving the information data while considering appropriate security measures.

- It consists of a group of Program which manipulate the data base.

- DBMS accept the user request for the data of an application instructs the operating System to provide the specific data.

- It provides an interface b/w data & s/w application.
- Help to create own database as per their requirement.

☆ Characteristics of DBMS :-

- Provide Security.
- Remove Redundancy → duplicate
- Support Multiple View of data
- Sharing of data
- Multi user transaction Processing
- Allows entities and relation among them to forms tables.
- Follows ACID concept (Atomacity, Consistency, Isolation, Durability).
- Support Multi-user environment that allows users to access & manipulate data in Parallel.

☆ DBMS VS Flat file :-

DBMS	Flate file Management System.
Multi user access.	It Does not Support multi user access.
Design to fulfill the need for small & large Business	It is only limited to small -ler DBMS System.

3)	Remove redundancy and intagraty	Redundancy & integraty issuess.
4)	Expensive (but in long term total cost of ownership is cheap.)	It is a cheaper.
5)	Easy to implement Complicated Transacti -on.	No Support for complic -ted transaction.

★ Advantages of DBMS :-

- DBMS offer a varity of techniques to sto and retrieve data.
- DBMS Serves as an efficient handler to balance the need of multiple applicatio using the same data.
- Uniform administration Procedure for the data.
- Application Programer never exposed to retain of data representation and storage.
- A DBMS users various Powerful function store and retrieve data.

- DBMS offer data Security and integrity.
- A DBMS Schedules concurrent access to the data in such a manner that only one users can accen the same data at a time.
- Reduce application development time.

★ Disadvantages of DBMS:-

1) Cost of Hardware and Software is quite high which increases the bujet of your organization.

2) Most database Management System are often complex System, so the training for the user to use the DBMS is required.

3) In some organizations all the data is intigrat -ed into a single database which can be damaged b/c of electric failure or database is correpted on the storage media.

4) Use of the same program at a time by many users some times lead to the loss of some data.

★ Database Architecture :- It is a representati ron of DBMS design.

- It helps to design, develope, implement & maintain the DBMS.
- A DBMS architecture allows dividing the database system into individual components that can be independently modified, changed, Replaced & altered.
- It also helps to understand the components of a database.
- A database stores critical information & helps access data quickly & securely. Therefore, Selecting the ~~cost~~ correct architecture of DBMS helps in easy & efficient data Management.

☆ Types of DBMS Architecture :-

1) 1 - Tier Architecture
2) 2 - Tier Architecture.
3) 3 - Tier Architecture.

I) 1- Tier Architecture :-

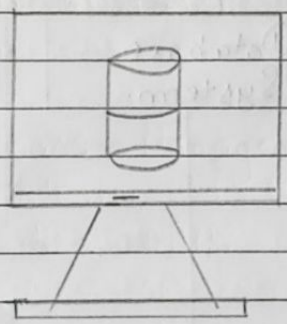
It is the simplest architecture of database in which the Client, Server and database all reside on the same machine.

→ In this architecture the database is direct

available to the user. It means the user can directly sit on the DBMS and uses it.

→ Any changes here will directly be done on the database itself. It doesn't provide a handy tool of an users.

→ The one tier architecture is use for development of a local application., where Programers can directly communicate with a database for the quick response.



Single - Tier Architecture.

II) 2- Tier Architecture :- It is similar to basic client - Server.

→ Application on client end can directly communicate both the DB at the server side for this interaction, API's like ODBC, JDBC are used.

ODBC → Object Database Connectivity
JDBC → Java Database Connectivity.

→ The user interface & application Programs are Sum on the client - side.

→ The server side is responsible to provide the functionalities like → Query Processing
→ Transaction Management

→ To Communicate with DBMS, client side applications established a connection with the Server side.

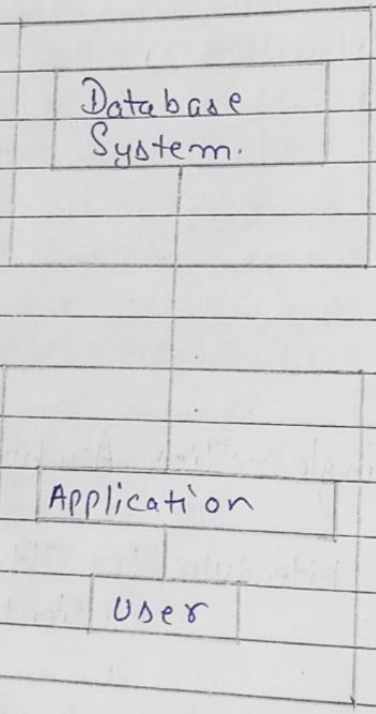


Figure 1 Second Tier Architcture

iii) 3- Tier Architecture :- It contains another layer b/w the client & Server.

-> Client cann't directly communicate with the server.

-> The application on the client-end interacts with an application server which further communicate with the database System.

-> End user has no idea about the existence of the DB beyond the application server.

-> The DB also has no idea about any other user beyond the application.

-> It is used in case of large web application.

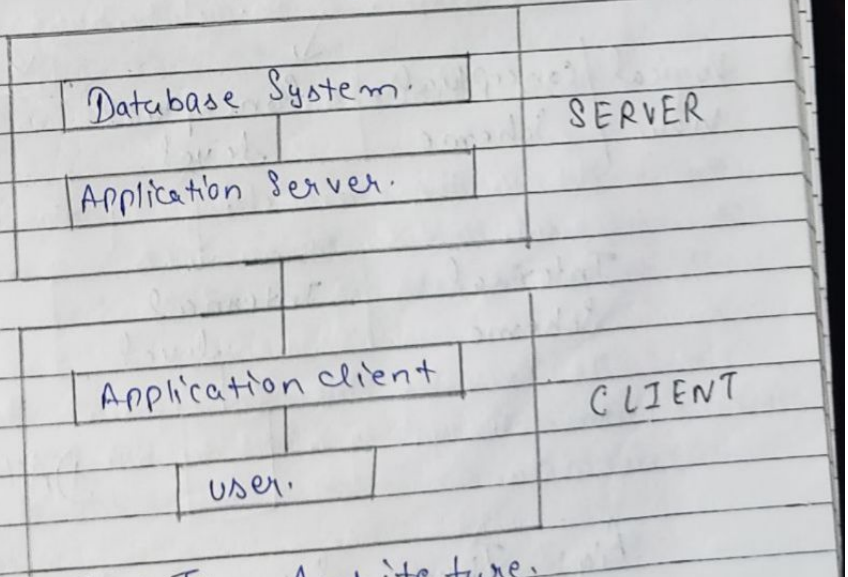


Figure : Third Tier Architecture.

☆ Three - Scheme Architecture :-

- It is also called Three - level Architecture
- This frame work is used to describe the structure of a specific database system.
- It is also used to separate the user application & Physical database.
- It contains three - levels it breaks the database down into three different Categories.

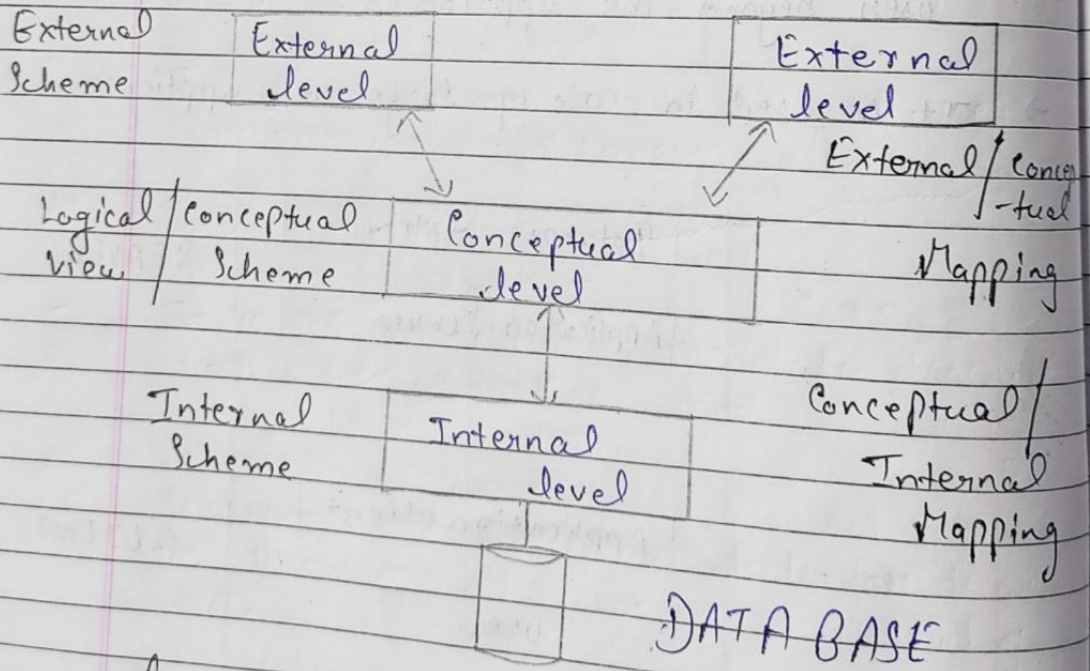


Fig: Three - Scheme Architecture.

PAGE NO.: 11
DATE: / /

I) Internal level :- Internal level has an internal speed which describe the physical storage structure of the database.

- It is also k/s as physical scheme.

- It uses of physical data model. It is used to define that how the data will be store in a block.

- The Physical level is use to describe complex low level data structure in detail.

II) Conceptual level :- The Conceptual level describe the design of a database at the Conceptual level. It is also k/s as logical level.

- It is describe the structure of the whole database.

- The Conceptual level describe what data are to be store in the database and also describe what relationship exists among those data.

- In the Conceptual level, internal detail such as an implementation of the data structure are hidden.

- Programers and database administrators work at this level together.

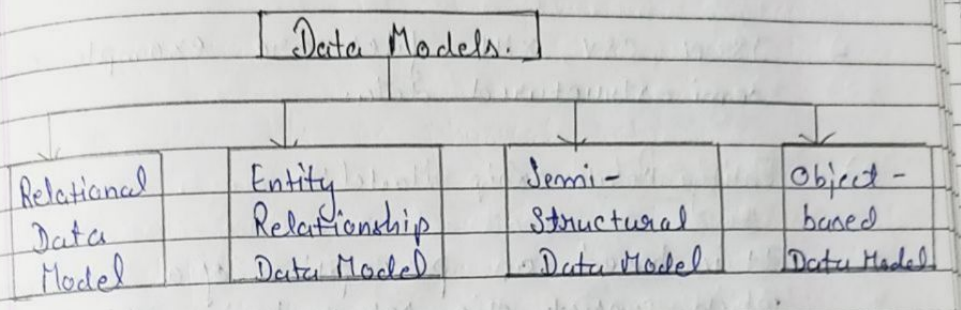
III) External level :- At the External level, a database contain scheme that some time it's sub scheme. The Sub scheme use to describe the different views of the database.

- It is also called view scheme.
- Each view scheme describe the data base part that a particular user interested on hide the remaning database from that the user.
- The view scheme describe the end user interaction with database system.

☆ Data Models :-

- It is the modelling of the data description, data semantics, and consistency constraints of the data.
- It Provides the conceptual tools for describing the design of a database at each level of data.
- There are 4 models used for understanding

the structure of database.



1) Relational Data Model :-

- It designs the data in the form of rows and column within a table.
- It uses tables for representing data and in b/w relationship.
- Tables are also called Relations.
- This Model was initially described by Edgar F. Codd in 1969.
- It is widely used by Commercial data processing applications.

2) Semi - Structured Data Model. :-

- It allows the data specification at places where the individual data items of the same type may have different attributes sets.

- XML is used for representing it.
- JSON, CSV, XML files are example of semi-structured data.

3) Entity Relationship Model :-

- It is based on perception of a real world which consist of a set of basic objects.
- Introduced by P.P. chen.
- Components.
  - Entity
  - Attributes
  - Relationship
    - one-to-one Relationship
    - one-to-many Relationship
    - many-to-many Relationship
  - Key attributes
- It is represented by symbols.
  - Rectangle
  - Oval
  - Diamond
  - Line.

4) Object based Model :-

- It is an extension of E.R. Model with notions of (Representation)
  - Functions
  - Encapsulation
  - Object identity.
- It support structured & collection types.

- Objects are data carrying its properties.

* Data Independence :-

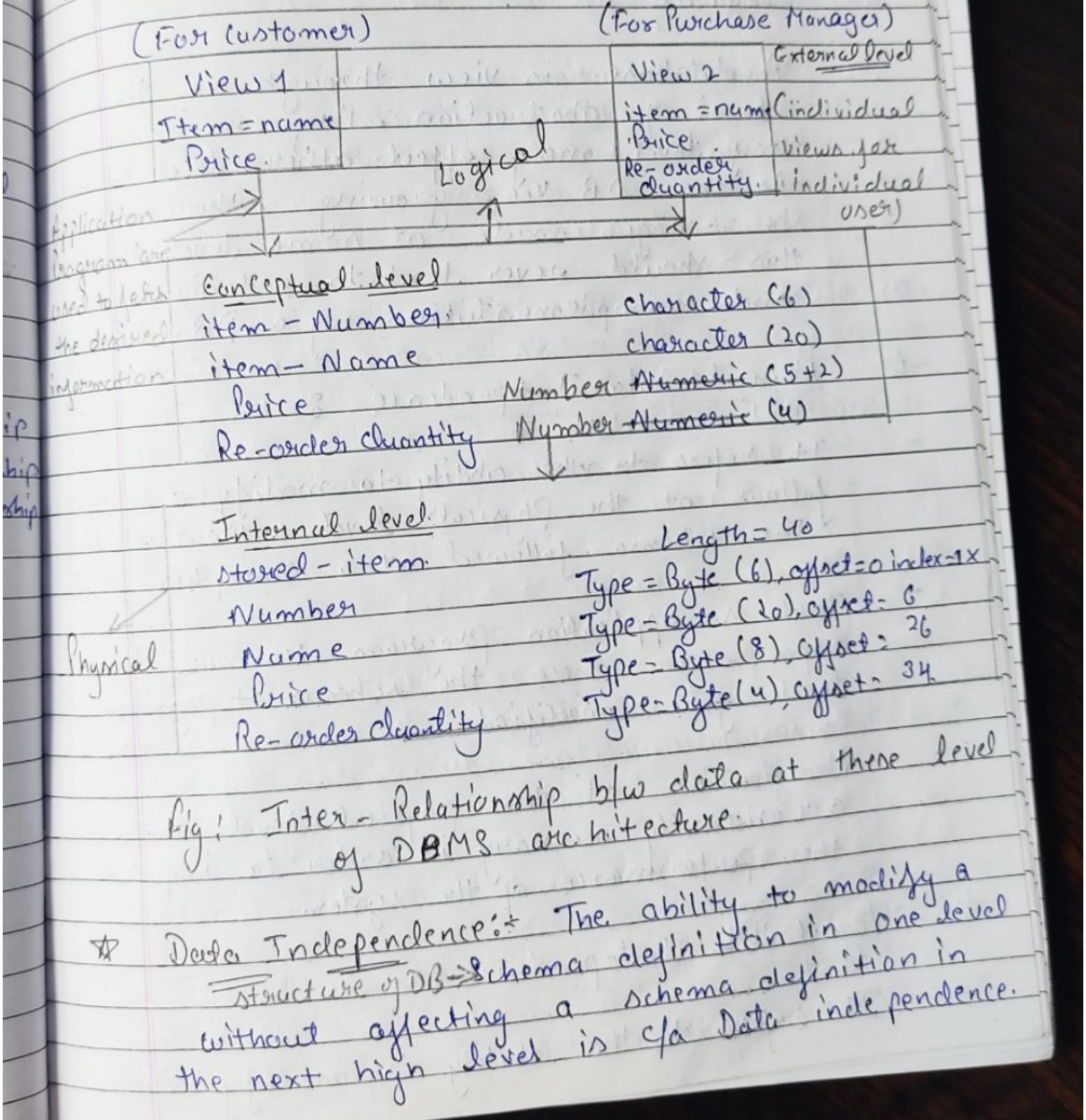


Fig: Inter-Relationship b/w data at three level of DBMS architecture.

* Data Independence :- The ability to modify a schema definition in one level without affecting a schema definition in the next high level is c/a Data independence.
Structure of DB → Schema

- There are two level of D.I.

1) Physical
2) Logical

- A database is view through any three level of abstraction. Any change at any level may effect other level scheme. As the DB keep growing, there may be changes made at some level. However, this should never lead to redesigning and reimplementation of a DB.

★ Physical Data Independence :-

- It refers to the cibility to modify a schema follow at the physical level without affecting the schema followed at a conceptual level.
- That is application program remains the same even those the schema at physical level gets modified.
- Modification at the physical level are occasionally necessary in order to improve the performance of the system.

★ Logical Data Independence :-

- It refers to the ability to modify the

conceptual schema without any change in the schema follow at view level.

- The LDI ensures that the application program remains the same.

- Modification at the conceptual level are necessary whenever logical structure of database get altered b/c of some unavoidable reasons.

Note:- It is more difficult to achive logical data independence than the physical data independence.

- The reason being that the application program are heavily dependent on the logical structure of D.B.

★ Database Administrator (DBA) :-

- Database Administration is implemented by a person or group of person under the supervision of a knowledgeable person c/a Administrator.

- This person, k/s as DBA.

- He is responsible for supervising → Creation, → Modification, → Maintenance } of DB

- DBA Control the DB structure & set-up the

definition for Physical as well as logical implementation of the DB.

☆ Function of DBA OR Role of DBA :-

1. Schema Definition
2. Storage structure & access method definition
3. Schema & Physical-organisation modification.
4. Granting of authorisation for data access.
5. Routing Maintenance.

I) Schema Definition :- The original DB Schema is created by writing a set of definition which are translated by DDL compiler to a set of table that are permanently store in the data dictionary.

II) Storage structure & access method definition :-

- Appropriate storage structure and access method are created by writing a set of definition which are translated by Data structure and definition language compiler.

III) Schema & Physical organization modification :-

- The DBA carries out changes to the schema and physical organisation to reflect the changing need of the organisation or to

altered the physical organization to improve the performance.

IV) Granting of authorization of Data access :-

- DBA also maintain that the DB is not accessable to unauthorized users.
- The DBA is responsible for granting permission to use the data base and store the profile of each user of a data base.
- The profile describe the permissible activity of a user on that Portion data base which is authorized to him.

V) Routine Maintenance :-

- The DBA responsible for defining processes to recover data some failure due to humans, natural or hardware malfunctioning with minim -um loss.
- This recovery processes would enable an organization to continue working with the available intact portion of the Data base.
- He also in short that enough free disk space is available for normal operation and upgrading this space as required.

PAGE NO.: 20.
DATE: / /

* Database Language :- DL are used to read, update and store data in a database.

Types → DDL
→ DML
TCL DCL

DB Language

1) DDL (Data definition language)
2) DML (Data Manipulation language)
3) DCL (Data control language)
4) TCL (Transaction control language)

1) DDL :- Is used to specifying the database schema.

- It is used for creating tables, schema, indexes, constraints in DB.
- The operations it can perform are :-

a) To create the DB instance - CREATE
b) To alter the structure of DB - ALTER
c) To Rename DB instance - RENAME

d) To drop objects from DB such as tables - DROP

e) To Comment - COMMENT.

2) DML :- It is used for accessing & manipulating data in a DB.

- Operation are :-

a) To Read records from table - SELECT

b) To insert records into the table - INSERT

c) Update the data in table - UPDATE

d) Delete all the records from the table - DELETE

3) DCL :- Used for granting and revoking users access on a DB.

a) To grant access to user - GRANT

b) To Revoke access from user - REVOKE

4) TCL :- The changes in the DB that we made using DML are either performed or rollback using TCL.

a) To persist the changes made by DML Commands in DB - COMMIT → (Successfully completion).

PAGE NO.: 22.
DATE: / /

b) To Rollback the change made to the DB – ROLL BACK.

12/10/21

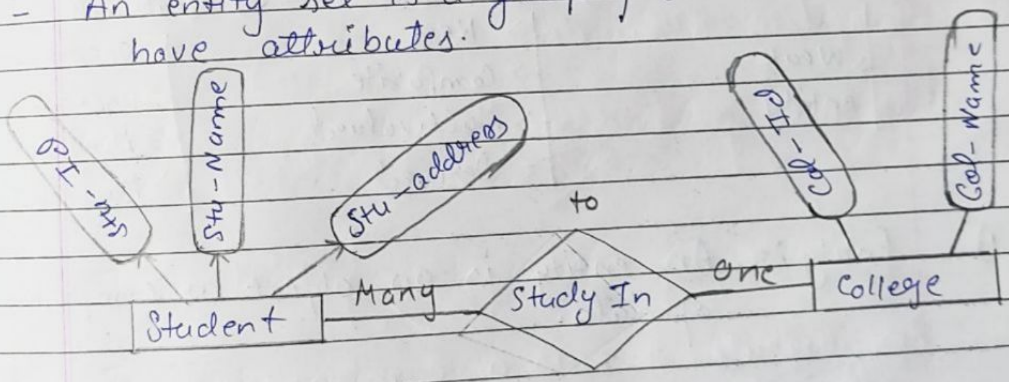
PAGE NO.: 1
DATE: / /

Unit → II

☆ ER Diagram :-

⇒ Entity Relationship Model

- It describes the structure of a DB with the help of a diagram, which is c/a ER-Diagram.
- ER Model is a design or blue-print of a DB that can be later implemented as DB.
- Main components of ER Model → 1) Entity Set
  → 2) Relationship Set
- An entity set is a group of similar entities & have attributes.



ER - Diagram

- We have 2 entities → Student
  → College

- Relationship b/w student & college is many to one
- Student entity have attributes → Stu-Id, Stu-Name, Stu-Address
- College entity have → col-Id, col-Name.

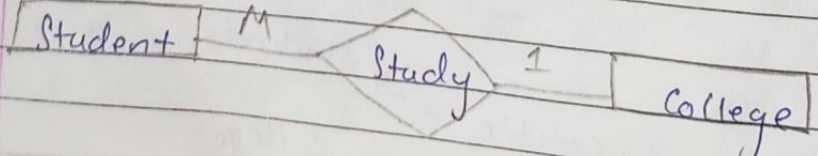
★ Components of ER-Model :-

ER-Model
- 1) Entity
  - Weak entity.
- 2) Attribute
  - Key
  - Composite
  - Multivalued
  - Derived
- 3) Relationship
  - One-to-one
  - One-to-Many
  - Many-to-one
  - Many-to-Many

1) Entity :- An entity is an object or component of data.

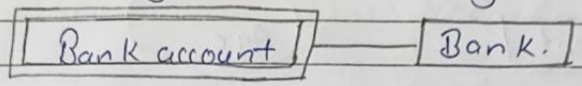
- Represented by a Rectangle.

→ Ex: Two entities → Student, College



a) Weak Entity :- An entity that can't be uniquely identified by its own attributes & relies on the relationship with other entity is c/a weak entity.

- Represent by Double Rectangle.



2) Attribute :- It describe the property of an entity.

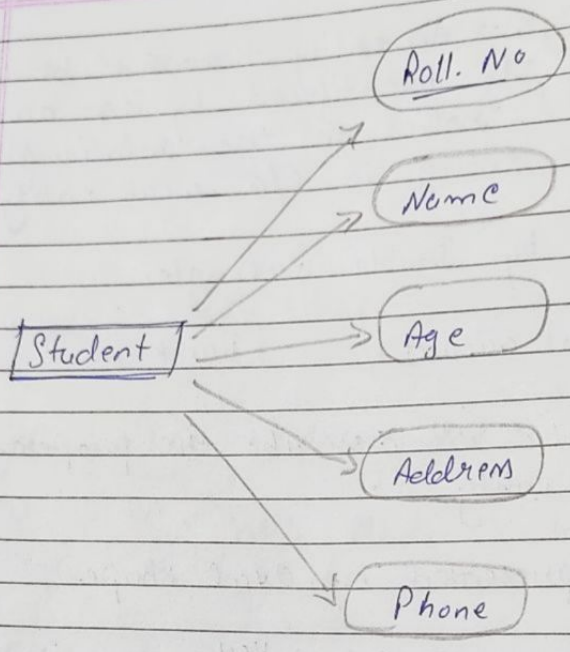
- It is represented by Oval shape.

- 4 types of attributes → Key
  → Composite
  → Multivalued
  → Derived.

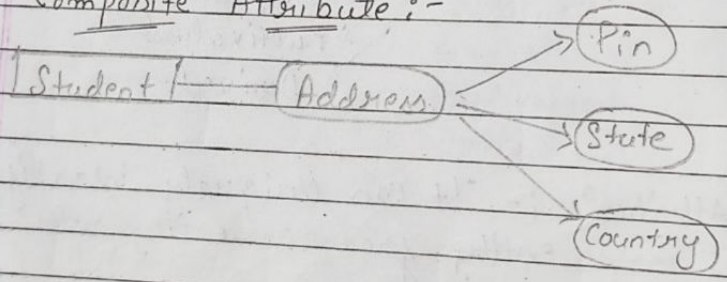
a) Key Attribute :- It can uniquely identify an entity from an entity set.

- Student Roll number can uniquely identify a student from a set of students.

- Test of Key attribute is underlined.



b) Composite Attribute :-



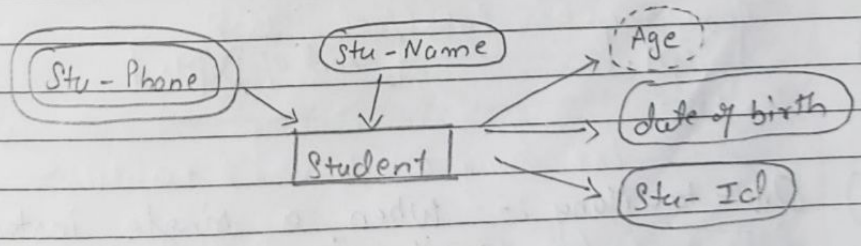
- An attribute that is a combination of other attribute is k/s as composite attribute.
- For ex:- In Student entity, the student address is a composite attribute as an address is composed of other attribute like Pin code, state, country.

c) Multivalued Attribute :-

- An attribute that can hold multiple values is k/s as Multivalued attribute.
- Represented with double ovals in an ER diagram.
- For ex:- A Person can have more than one Phone number therefore the Phone no. attribute is multivalue.

d) Derived Attribute :-

- A derived attribute is one whose value is dynamic and derived from another attribute
- Represented by dashed oval in an ER diagram
- For ex:- Person age is a derived attribute b/c it changes over time and can be derived from another attribute (date of birth).



3) Relationship :- A relationship is represented

by diamond shape in ER-diagram, its shows the relationship b/w diff. entities. There are basically 4 type of relationship.

1) one-to-one
2) one-to-many
3) many-to-one
4) many-to-many.

a) One-to-One -> When a single instances of an entity is associated with a single instance of another entity then it is of a one-to-one relationship.

- For ex:- A person has only one pamport and a single pamport is given to one person only.

a) [Person] 1 — has — 1 [Passport]

b) [Automobile] → [Engine]

c) [Dept.] → [Head of Dept.]

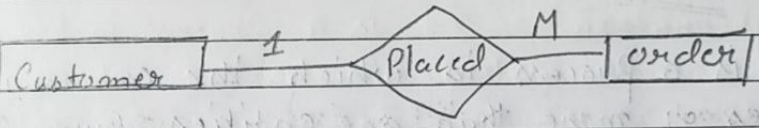
b) One-to-Many :- When a single instance of an entity is associated with more than one instance of another entity, then

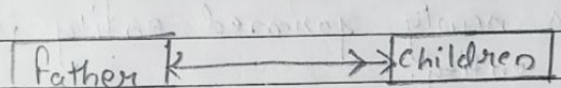
it is c/a one-to-many Relationship.

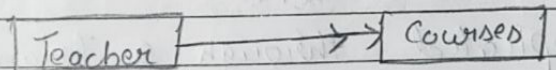
- For ex→ A customer can place many order but a order can't be placed by many customers.

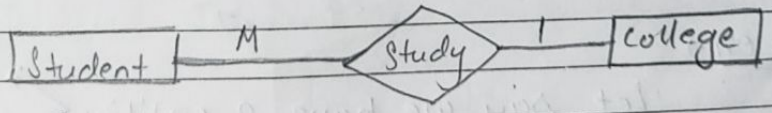
c) Many-to-One Relationship :- When more than one instance of an entity is associated with a single instance of another entity then it is c/a many-to-one Relationship.

- For ex→ Many students are study in a single college but a student can't study in many colleges at the same time.

b) 



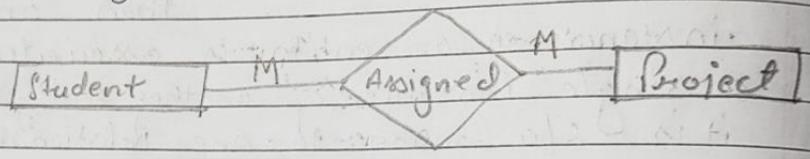


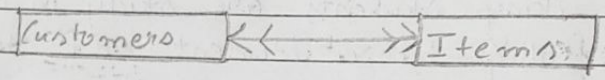
e) 

d) Many-to-Many Relationship :- When more than one instance of an

entity is associated with more than one instance of another entity, then it is c/a Many-to-Many Relationship.

- Foreg → A student can be assigned to many projects and a project can be assigned to many students.



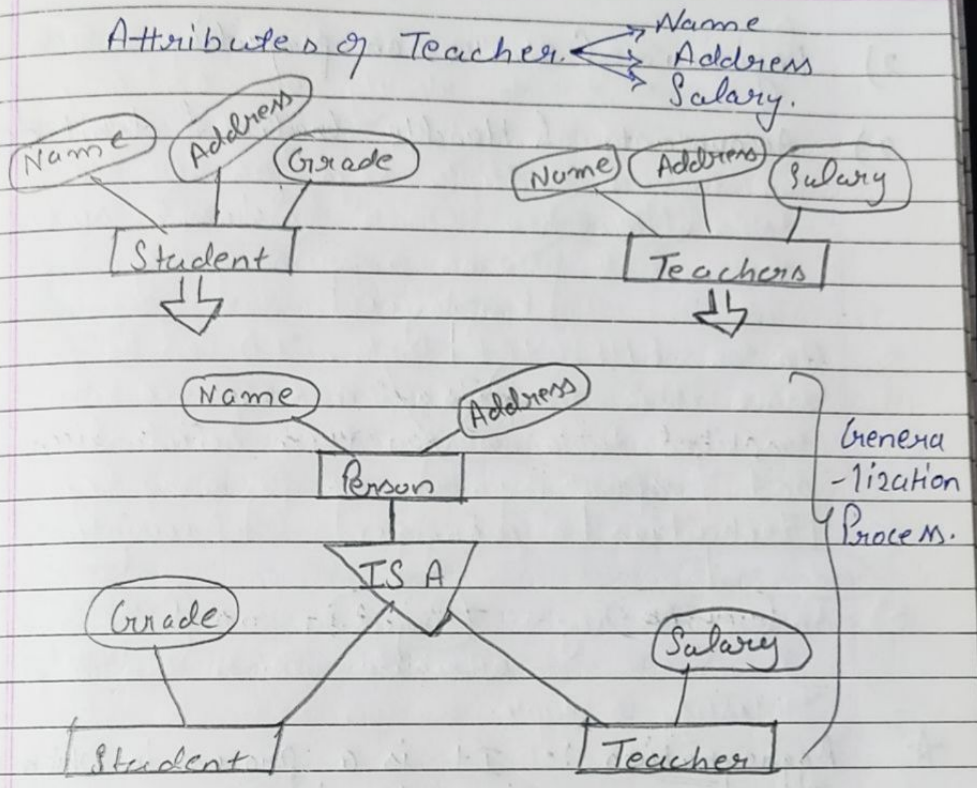


★ Generalization :-

- It is a process in which the common attributes of more than one entities form a new entity. This newly formed entity is c/a generalized entity.
- It is depicated through a Triangle component labelled "IS A".
- Foreg →

Let say we have 2 entities → Student, Teacher.

Attributes of Student → Name, Address, Grade.



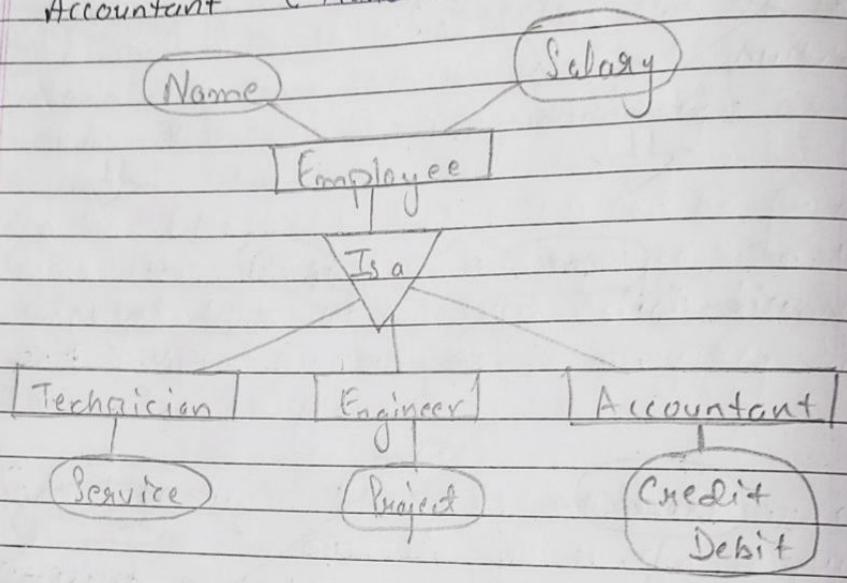
☆ Specialization :- It is a process in which an entity is divided into sub-entities.

- It is a reverse from of generalization.
- It find the subsets of entities that have few distinguish attributes.
- Parent Entity - Employee can be further classified as.

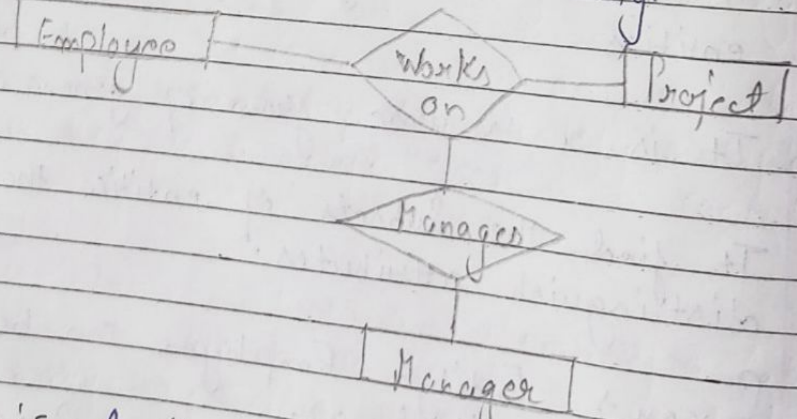
1) Technician (Handle Service Request)

2) Engineer (Works on project)

3) Accountant (Handle Credit & debit details)



☆ Aggregration :- It is a process in which a single entity alone is not able to make sense in a relationship so the relationship of two entities acts as one entity.



☆ Keys :- A Key is that data item that

exclusively identifier a record.

- It is of 4 types:
  - Primary Key
  - Candidate Key
  - Super Key
  - Foreign Key.

1) Primary Key :-

STUDENT	PERSON.
ID	Name (Primary key)
Name.	DOB.
Address	Passport - Number
Course	License - Number
	SSN.

(Passport - Number, License - Number, SSN. : candidate key.)

Employee	Department
Employee - ID (Primary Key)	Department - ID
Employee - Name	Department - Name
Passport - Number (candidate Key)	
License - Number (candidate Key)	
SSN (candidate Key)	
Department - ID	

Department - ID → Primary Key

Department - ID → Foreign Key

1) Primary Key → It is a first key which is used to identify one and only one instance of an entity uniquely. If an entity can contain multiple keys as we see in person table. The key which is most suitable from those list become a primary key.

- In the EMPLOYEE table, ID can be primary key since it is unique each employee. In the employee table we can even select license number and passport number as primary key b/c they are also unique.

2) Candidate Key → It is an attribute or set of a attribute which can uniquely identify a tuple.

- The remaning attribute except for the primary key are considerd as a candidate key. The candidate key is are as strong as primary key.

3) Super Key → It is a set of an attribute which can uniquely identify a tuple. Super key is a super set of a candidate key.

4) Foreign Key → It is a column of a table which is used to point to the primary key of another table.

PAGE NO.:
DATE: / /

☆ Relational Algebra (RA):-

- Relational Algebra uses a procedural query language which is a collection of operations to manipulate relations.
- It consist of a set of such operation that take one or more relation as input & produce a new relation as their result.
- Relation Algebra Operations
  - Select (σ)
  - Project (π)
  - Join
  - Union.
  - Difference.
  - Intersection, etc.

1) Select (σ):-

- The select operation extracts specific tuples from a relation.
- Sigma (σ) Greek letter.
- Deposite Relation.

P.T.O

Branch Name	Account Name	Customer Name	Balance
Triveni Nagar	000123	Gopal	5000/-
Malviya Nagar	001445	Divya	10,000/-
Aadarsh Nagar	001566	Isha	2000/-
Malviya Nagar	00773	Akshita	4000/-
Vaishali Nagar	001129	Yashraj	8000/-

- forex→ We want to find tuples in which the balance amount is more than 7000/-

$$\sigma_{Balance > 7000}(Deposit)$$

2) Project (Π) :-

- It is a unary operation extracts attributes from a relation.
- Denoted by Greek pi (Π)
- forex→ Suppose we want to know the customer names and their bank balance amounts from deposite relation.

$$\Pi_{customer\text{-}name,\ Balance}(Deposite)$$

3) Set-Union :-

- The result of union is denoted by the symbol (∪).

- For ex → X ∪ Y is a relation that includes all tuples that are either in X or Y or in both X & Y.

- Duplicate tuples will be eliminated by using set union.

- X & Y are two Relations.

- X holds details of employee working on Project J_1.

- Y holds employee working on J_2.

X

Emp-No	Emp-Name
101	Rita
103	Ritu
104	Setu
106	Leela
107	Neena

Y

Emp-No.	Emp-Name
101	Rita
105	Isha
107	Neena
112	Seema

- For ex → To select all those employee who are working on either in Project J_1 or J_2 or in both Project.

- The result will by X ∪ Y.

X ∪ Y :

Emp-No.	Emp-Name
101	Rita
103	Ritu
104	Sehu
105	Isha
106	Leela
107	Neena
112	Seema

4) Set Intersection :- (common Records)

- It is denoted by $X \cap Y$
- It includes all ~~tuples~~ tuples that are in both X and Y relation.
- For ex → To select those employees who are exclusively working on both projects J_1 and J_2.
- Output $X \cap Y$

Emp-No	Emp-Name
101	Rita
107	Neena

5) Set Difference :-

- Denoted by X - Y

- It allows us to find tuples that are in one relation but are not in another relation.

- For ex→ To select all those employee who are exclusively working on Project J1 given by X-Y.

- Output X-Y

Emp-No.	Emp-Name
103	Ritu
104	Seta
106	Leela

6) Cartesian Product (Cross Product) :-

- Denoted by X x Y and returns a relation on tuples whose schema contains all fields of x (in the same order they appear in x) followed by all fields of y (in the same order as they appear in y).

- The result of X x Y contains are tuples (r, s) (the concentenation of tuples r and s) for each pair of tuples where r belongs to S (r ∈ S) and S belongs to S (S ∈ S).

y : Relation

Project
AX10
AX11

X; Relation

Emp-No.	Emp-Name
101	Rita
103	Ritu
104	Setu
106	Leela
107	Neena

- Eg:- The operation X x Y give the resulting combination as follows.

Emp-No	Emp-Name	Project
101	Rita	A x 10
101	Rita	A x 11
103	Ritu	A x 10
103	Ritu	A x 11
104	Setu	A x 10
104	Setu	A x 11
106	Lella	A x 10
106	Lella	A x 11
107	Neena	A x 10
107	Neena	A x 11

★ Join Operation (⋈) :-

- It allows to combine two relation to form a single new relation.
- The tuples from the operand relations that participate in the operation and contribute

to the result are related.

- The join operation allows the processing of relationship existing b/w the operand relations.
- For ex→ Suppose we want to retrieve the names of the employee who work in department 10, then we will use the following query.

Employee

Emp-No	Emp-Name	Project
1	Rita	AX10
2	Reetu	AX10
3	Seema	AX11
4	Leela	AX10

Department

Emp-No	Dept-No
1	10
2	20
3	10
4	30

Π (Employee.Emp-Name) (Employee ⋈ Department Dept-No =10)

1) Equi Join :-

- A Join where the only comparision operation used in the join condition "=" is called Equi-Join.
- The result of equi-join is always has one or more pairs of attributes that have identical values in every tuple

Employee

Name	Emp ID	Dept Name
Harry	3415	Sales
Sally	3241	Sales
George	3401	Finance
Harriot	2202	Production

Department

Dept Name	Manager
Sales	Harriot
Production	Charles.

Employee ⋈ Dept OR employee.DeptName = Dept.Dept Name.

Name	Emp ID	Dept-Name	Dept-Name	Manager
Harry	3415	Sales	Sales	Harriot
Sully	2241	Sales	Sales	Harriot
Harriot	2202	Production	Production	Charles.

2) Theta Join :- (Θ)

- The Θ- join is a binary operator that is written as $R \bowtie S$ where a and b are attribute manner, $a \Theta b$.
- Θ is a binary relation in the set $(<, <=, >, >=, =)$
- The result of this operation consists of all combination of tuples in R and S that satisfy the relation Θ.
- For ex → Consider tables Car and boat which list models of cars and boats and their respective prices. Suppose a customer wants to buy a car and a boat, but doesn't want to spend more money for the boat than for the car.

Car

Car Model	Car Price
Car A	200,000
Car B	300,000
Car C	500,000

Boat

Boat Model	Boat Price
Boat 1	100,000
Boat 2	400,000
Boat 3	600,000

PAGE NO.:
DATE: / /

Car ⋈ Boat (carPrice ≥ Boat Price)

Car Model	Car Price	Boat Model	Boat Price
Car A	200,000	Boat 1	100,000
Car B	300,000	Boat 1	100,000
Car C	500,000	Boat 1	100,000
Car C	500,000	Boat 2	400,000

3) Natural Join :-

- Natural Join is same as equi join but. meths the value in columns of the same name and do not output those columns twice.
- The natural join is a binary operation that allows us to combine certain selection and a cartesian product into one operation.
- This operation forms a cartesian product of its two argument, performs a selection forcing equality attribute that appear in both relation and finally remove duplicate attributes.

Unit → III

☆ Normalization :-

- It is a name given to the process of simplifying the relationship amount data elements in a record.

- It is a replaces are collection of data in a record structure by another record design which is simpler, more predict ruble and therefor more managable.

- The gole of relation database design is to generate the set of relation key that allows us to store information without any repeated data or redunded data.

- If allows us to retervive information easily and more efficiently.

- The first step towards normalization to convert ER Model into Table or relation. The next step is to examine the table for redundancy and if necessary, change them to non-redundant form this non-redundant model is then converted to a database definition, which achive the objective of the database design phase.

☆ Need for Normalization :-

- Normalization reduces redundancy. Redundancy is the unnecessary repetation of a field it can cost problem with storage, retrieval and updation of data.

- Redundancy can lead to
  - (a) Inconsistencies → Errors are likely to occur when facts are repeated.
  - (b) Update Anamolies → Inserting, modifying and deleting data may cause inconsistence.

- Inconsistency occurs when we perform updation or deletion of data in one relation while forgetting to make corresponding changes in other relations.

- A Fully normalization Record Consist of :-

(a) A Primary key that identifiers that entity.
(b) A set of attributes that describe that entity.

★ First Normal Form (1NF) :-

- A table is in the first Normal form when it contains no repeating groups.

Salesperson				Sales		
Employee No.	Employee Name	Store branch	Dept.	Item No.	Item Discription	Salse Price
21130680	Anand K	Dountown	Hardware	TR10	Router	35.00
				SA1	Saw	19.00
				PT6	Drill	21.00
				AB16	Lawnmover	245.00
30142101	Zadoos	Dadeland	Home Appliances	TT1	Humidifier	114.00
				DS10	Dishwasher	262.00
41984620	Balwant	Cutter Point	Autopaste	MC16	Sinow tire	85.00
				AC146	Alternator	65.00
				BB100	Battery	49.00
61204721	Bhagwan	Faishon Spot	Men's clothing	MS10	Suit	215.00

Fig 1 :- Unnormalized file for Sales.

(A) Salesman Data file

Employee No.	Employee Name	Store Branch	Dept.
21130680	Anand K	Dountown	Hardware
30142101	Zadoos	Dadeland	Home Appliances
41984620	Balwant	Cutter Point	Autopaste
61204721	Bhagwan	Faishon Spot	Men's clothing

(B) Salesperson Item file

* Employee No.	* Item No.	Item Discription	Sales Price
21130680	TR10	Router	35.00
21130680	SA1	Saw	19.00
21130680	PT6	Drill	21.00
21130680	AB16	Lawnmover	245.00
30142101	TT1	Humidifier	114.00
30142101	DS10	Dishwasher	262.00
41984620	MC16	Snowtire	85.00
41984620	AC146	Alternator	65.00
41984620	BB100	Battery	49.00
61204721	HS10	Suit	215.00

Fig 2:- First Normalised file for Sales.

☆ First Normal Form (1NF):- A table is in the first normal form when it contains no repeating groups. The repeating columns or field present in an unnormalized table are remove from the table an put into seperate table or tables. This table are depending on the parent table from which it is derived. The key to these tables must also be a part of the pairing table, so that the parent table and the derived table can be related to each other. isolate repeating group from

an entity b/c they are easier to process seperatly, from rest of the entity.

Imp * Def :- When a table has no repeating group, it is said to First Normal Form (1NF). That is, for each in a table (1 row & 1 column), there can be only one value. This value should be atomic in the sence that it can't be decom-pose into smaller piceses.

- As in Fig 1, first four attribute (employee no., employee Name, store branch, Dept.) are virtualy constant.
- The remaning three attribute (item no., item discription, sales price) contain data that change and are repetate with diff. salse person. Therefore the repeating group should be seperated from the entity "Sales person".
- The normalized file show in fig 2. its consists of two file :-

(A) The Sales Person data file with employee no. as the primary key.

(B) The Salesperson item file with employee no. and item no. as new attribute.

This attribute are added to relate the records in this file to the salesperson data file. The

two attribute are used together for accessing data. Therefore, two keys are used together and such key is called concantenated key.

✱ Functional Dependencies :-

- When in a given relation R, attribute Y of R is functionally dependent on attribute X of R if and only if, each X-value of R is associated with it precisely on a Y-value in R.
- A functional dependency is denoted by $X \rightarrow Y$, b/w two sets of attributes X and Y.

Suppliers (S).

S.No.	Name	Status	City.
S_1	Shyam	20	Bombay
S_2	Ram	10	Calcutta
S_3	Amit	30	Calcutta
S_4	Chirag	20	Delhi
S_5	Ramesh	30	Calcutta.

S. S.No → S. Name
S. S.No → S. Status
S. S.No → S. City.

- In Suppliers database, attributes Name, Status and city. of relation S are each functionally

dependent on attribute SNo, b/c given a particular value of SNo, there exists precisely one corresponding value for each of Name, status and city.

* Functional Dependencies → It plays an imp. role it differentiating good database design from not show good data base.

- Functional Dependencies are the consequences of the inter relationship among attribute of an entity represented in a relation.

- The database containing information concerning suppliers (S) and parts (P).

- The Suppliers and parts are uniquely identified by suppliers numbers (S.No) and Part No (P.No).

P.No	Name	Color	Weight	City
P_1	Nut	Red	12	Bombay
P_2	Bolt	Green	17	Calcutta
P_3	Screw	Blue	17	Goa
P_4	Screw	Red	14	Bombay
P_5	Handle	Blue	12	Bombay
P_6	Wire	Red	19	Delhi

Parts (P)

P. PNo → P. Name
P. PNo → P. Color
P. PNo → P. Weight
P. PNo → P. City.

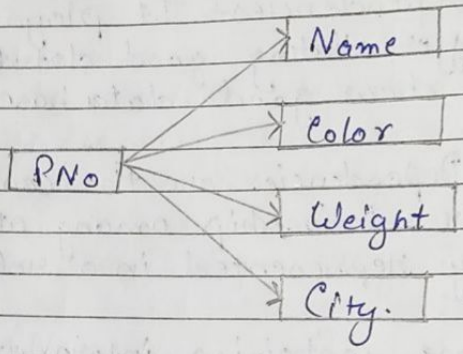


Fig:- Functional Dependencies in Relation (P)

Student Relation (Name, Address, Subject, Grade)

Name → Address
Name Subject → Grade.

Functional dependencies of Student Relation.

- The name of a student is unique
- In each subject a student gets a unique grade

Employee (Emp-Code, Emp-Name, Dept, Grade,

salary, Age, Address).

- f1: Emp-code → Emp-Name;

→ Each emp. has a unique emp. code.

- f2: Emp-code → Dept;

→ An emp. can work in one dept. only.

- f3: Emp-code Grade Age → Salary;

→ Emp. Salary depends on his age & grade.

- f4: Emp-code → Age;

→ Each emp. has a unique age

- f5: Emp-code → Address;

→ Each emp. has unique address.

* In this relation employee code is not functional dependencies on his salary or age, b/c more than one emp. can have the same salary and can be on the same age.

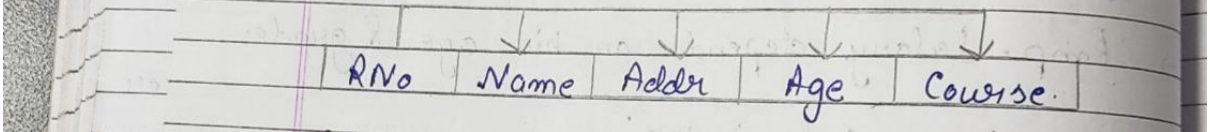
☆ Types of Functional Dependencies :-

1) Full functional dependencies.

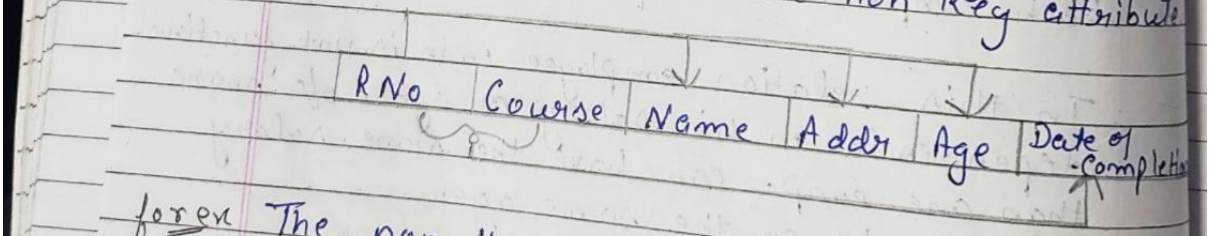
2) Partial Dependencies
3) Transitive Dependencies
4) Multivalue Dependencies.

1) Full functional dependencies :- When all non Key attribute are depended on the key attribute, it is called full functional dependencies.

for ex In student relation the non key attribute (Name, age, address & course) are depended on key attribute roll no.



2) Partial Dependencies :- Partial dependencies in a record type occurs when some non key attribute depends on the key attribute and the remaning non key attribute depend on key attribute and on one or more non key attribute

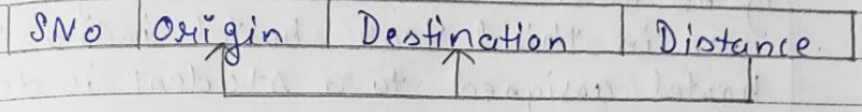


for ex The non key attributes Name, Addr and age are dependant as RNo and date of completion i.e. non-key attribute depends on RNo (key attribute) as well as course

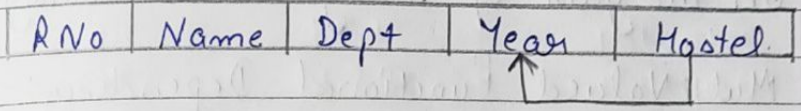
(non-Key attribute).

3) Transitive Dependencies :-

(a)



(b)



- If A, B, C are 3 Columns in a table.
  If C is related to B
  If B is related to A
  Then C is indirectly related to A.

- A F.D: $X \to Y$ in a relation Scheme R, is a transitive dependency if there is a set of attributes X that is a neither a candidates Keys nor a Subset Key of R and both $X \to Z$ & $Z \to Y$ holds.

* When one non Key attribute depend on other non Key attribute it is c/a a transitive dependency.

- To Calculate distance in fig (a) which is a non Key attribute we must know the origin & destination which are also non

Teacher's Signature.....

Key attributes:

- In fig (b) if a student is in first year than he will assign hostel gandhi if he is a second year hostel jawahar .On in third year hostel indira. thereby hostel assigned to a student is dependent on the year of study in the college.

4) Multivalued Functional Dependency:- In this entities of the dependent set are not dependent on each other.
i.e. If $a \rightarrow \{b, c\}$ and there exists no functional dependency b/w b and c, then it is called a Multivalued functional dependency.

Car-Model	Manufacture-Year	Color
H001	2017	Metallic
H002	2018	Green
H005	2017	Blue
H0012	2012	Metallic
H0033	2010	Grey

- Manufacturing year and color are independent of each other but dependent on

Car - Model.

Car - Model →→ Manufacture - Year.
Car - Model →→ Color

☆ Second Normal form (2NF) :- A relation that is in first normal form and every non primary key attribute is fully functionally dependent on the primary key, then the relation is in second normal form (2NF).

* A table in the 2NF if all its non key fields are fully dependent on the whole key this means that each field in the table, must depend on the entire key. Those that do not depend on the combination key, are move to another table on whose key they depend on.

- The Second normalization make sure that each non key attribute depend on a key attribute. for ex in given fig each attribute in the Sales person data file depends on the primary key employee no. in the Sales person item file, the attribute sales price depends on a composite key employee no. and item no.

⇒ Referred to Sales person data file

Teacher's Signature.....

(B) Sales Person Item file

Employee No.	Item No.	Sales Price
21130680	TR10	35.00
21130680	SA1	19.00
21130680	PT6	21.00
21130680	AB16	245.00
21130680	TT1	114.00
30142101	DS10	262.00
30142101	MC16	85.00
41984620	AC146	65.00
41984620	BB100	49.00
41984620	MS10	215.00
61204721		

(C) Item file

Item	Item discription
TR10	Router
SA1	Saw
PT6	Drill
AB16	Lown mover
TT1	Humidfier
DS10	Dish washer
MC16	Snowtire
AC146	Alternator
BB100	Battery
MS10	Suit

DATE : / /
PAGE NO. :

☆ Third Normal form (3NF) → A table is said to be 3NF if all the non key fields of the table are independent of all other non key fields of the table.

- In 3NF no non primary attribute is functionally dependent on other non prime attribute.

- This normalization simplify the relationship and provide logical links b/w file without losing the information.

Teacher's Signature.....