PAGE NO.
DATE / /

DBMS Assignment-1

1) Define the following terms:

i) Data - Any fact i.e unprocessed information that can be recorded and stored. It can be text, numbers, images, vedios etc.

ii) Database - It is a collection of related data, it may be collection of text, num or images. A database can be of any size and complexity. An eg for this large commercial database is Amazon.com. The database occupies 2 terabytes & is stored in 200 different companies. The database is copied updated as new items are added & stock quantities are updated as purchases are made.

iii) DBMS (Database management system) - It is a collection of programs that enables the user to define, construct and manipulate and sharing database among various user and application

Defining the database involves specifying the datatypes & constraints of the data to be stored in the database.

Constructing the database is the process of storing the data on some storage medium that is controlled by DBMS.

* Manipulating a database includes retrieves data or updates the data.
* Sharing the database allows multiple users and programs to access the database simultaneously.

2) Discuss the main characteristics of the database approach and how does it different from traditional file system.

In traditional file processing, each user definition defines and implements their own files needed for a specific s/w application. For eg the grade reporting officer may keep a file on students and their grades. An accounting office may keep track of students fees & their payments. Although the students information is needed by both the users, they maintain seperate files, each user requires some data not available from the other user files. This causes redundancy.

In the database single repository of data is maintained that is define once and then accessed by various users.

Characteristics of Database approach v/s the files processing approach are:

1) Self describing nature of the Database system:

In DBMS, the database structure is stored in a catalog and it also contains the storage detail with constraints. The information stored in the catalog is called meta-data & it describes the structure of primary Database.

A general purpose DBMS s/w package, must refer to the catalog to known the structure of the file in a specific Database. DBMS s/w must work equally with any number of Database application. But in traditional file processing data definition is the part of the application program, these program are contrained to work with only one specific

database, where structure is declared in the application program, it can access only specific Database.

2) Insulation between program and data and data abstraction:

In file processing, if changes are done in the structure of the file, then we may require changing all program that access the file.

In DBMS, the access program are written independent of any specific files. i.e data is stored in the system catalog not in the program [i.e one change is all that's needed] we call this property program-data independent.

The characteristics that allows program-data independent is called data abstraction. A DBMS provides the user, with a conceptual representation of data, that does not include many of the detail of how the data is stored or how the operation are implemented.

3) Support of multiple views of data:

A database may have many users & each one may be interested on a particular view of the DB. A view is conceptually a table which contain virtual data that is derived from the database files but is not explicitly stored.

For eg: One user of the database of fig may be interested only in student grade in various courses to obtain this information the course & grade report table are to be joined and created as a view.

4) Sharing of data and multiuser Transaction process:

A DBMS must allow multiple user to access the database at the same time. This is essential if

data for multiple app is to be integrated and maintained in a single DB. This is achieved by concurrency control that ensures several user access to the same data item at the same time.
for eg, when several reservation clerks try to assign a seat on an airline. DBMS should ensure that each seat can be accessed by only one clerk at a time for assignment to the passenger.

3) Briefly discuss the advantages of using the DBMS.
1) Controlling Redundancy:
In traditional files system, each files stores the same data in different files which leads to data redundancy problem encountered due to redundancy.
→ Duplication effort : We need to perform single update such as entering data on new student - multiple times One for each file where student data is recorded
→ Storage space : space is wasted when the same data is stored seperately.
→ Inconsistancy in data : This may happen an update is applied to some of the files but not to others.

2) Restricting Unauthorized access:
When multiple users share a large database the type of access operation must be controlled. It is likely that most user will not be authorized to access all the information in the DB. For eg any confidential data of a company like finance will be accessable only by a few authorized users. Some users will be allowed only to retrieve data & some users will be allowed to retrieve & update data in the database.

DBMS provide the security and authorization subsystem. This is done by a DBA while creating the user account.

3) Providing persistant storage for program objects:
As we have Object oriented programming concept, based on the same concepts, we have object oriented Database systems. Since DB provides the storage of program objects for long period of time we have this concept of OODB.

In traditional file system the values of the program variable will be discarded or deleted once the program terminates. But in Database management the values of the program variable will not be discarded when the program terminates. They allow the program object to store permanent & such an object is said to be persistant.

4) Providing storage structures for efficient query processing:
Database systems must provide capabilities for efficiently executing queries and updates. Since DB is stored on disk must provide specialized DS (indexes) to speed up disk search. To process the query the DB records will be copied from disk to the main memory, there is something called DB buffer that allows temporary storage of data in the main memory.

5) Providing backup and recovery:
The backup and recovery subsystem of DBMS is responsible for recovery in case of H/W or S/W

failures. The failure may occur at any time and therefore the DB should be in a position to recover the system to the point where the crash occured & may undo all the transaction.

5) Providing Multiple user interfaces:
As number of users use a DB with different levels of technical knowledge, a DBMS should provide a variety of user interfaces, like providing Query language for casual user, programming interfaces for application programmer forms of parametric user.

4) Representing complex relationship among data:
A Database include variety of data that are interelated in many ways so a DBMS must be capable of
→ Representing complex relationship among data
→ so definition a new new relationship as they arise &
→ to retrieve & update related data easily & efficiently.

8) Enforcing integrity constraints:
Most DB application have certain integrity constraint that must hold for the data. A DBMS should provide capabilities for define and enforcing these constraints.
The simplest type of integrity constraint involves specifying a datatype for each data item. For eg we can specific that the value of the 'name' data item must be string of not

more than 30 char.

g) Permitting inferencing and actions using rules.

Deductive DB system provide capabilities for defining deduction rules for inferencing new information from the stored DB facts. We can have rules in the miniworld application for determining whether the student can write his exam or not. We can keep rule saying that only student > 70% of attendance can take up the exam. There are the rules that can be applied and based on there rules some actions can be taken.

4) Explain actors on the scene and workers behind the scene.

Actors on the scene:

It is not possible to maintain the Database by a single person in a large application. A set of people are responsible for the design, development & maintaining the data. There people whose job involve day-to-day use of a large Database are called actors on scene.

Types:

1) Database Administrator (DBA): DBA is a person responsible for over all control of database system. All the activities in the database system are controlled by DBA. Hence he is responsible for.

* Authorization access to the Database
* Co-ordinating & monitoring its use.
* Managing the resources.
* Providing security & authorization.

more than 30 char.

g) Permitting inferencing and actions using rules.

Deductive DB system provide capabilities for defining deduction rules for inferencing new information from the stored DB facts. We can have rules in the miniworld application for determining whether the student can write his exam or not. We can keep rule saying that only student > 70% of attendance can take up the exam. There are the rules that can be applied and based on there rules some actions can be taken.

4) Explain actors on the scene and workers behind the scene.

Actors on the scene:

It is not possible to maintain the Database by a single person in a large application. A set of people are responsible for the design, development & maintaining the data. There people whose job involve day-to-day use of a large Database are called actors on scene.

Types:

1) Database Administrator (DBA): DBA is a person responsible for over all control of database system. All the activities in the database system are controlled by DBA. Hence he is responsible for.

* Authorization access to the Database
* Co-ordinating & monitoring its use.
* Managing the resources.
* Providing security & authorization.

* Tuning the DB depending upon the needs of its user.

2) Database designer: DB designer task is undertaken before the database is actually implemented. He is responsible for:
* Interacting with each group of user & develop view of DB. Then this view is analyzed & integer with the view of other user group. Then he identify the data to be stored in the database and chose the appropriate structure to represent & store this data. Finally he plans the DB design that meets the requirement of all user.

3) End user: The jobs of end user is required to access the DB for quering, updating and generating reports. The categories of end user are:
i) Casual end user: There are the people who use the DB occasionally, they need different each time. Eg: occasional browsers, middle or high level manager.
ii) Naive or Parametric end user: There are the user who constantly querying and updating the DB, without having much knowledge about the DB, for eg reserving clerks for railways, airline & hotel.
iii) Sophisticated end user: Engineers, scientists, business analyst etc who would use the DB for their complex requirement.
iv) Stand alone end user: who maintain DB for personal use: for eg accounting package

4) System analysts and application programmer.

s/w analyst determine the requirement of endus develop specifications to meet their requirement. App programmer implement these specifications as programs, then they test, debug, document & maintain their transaction.

WORKERS BEHIND THE SCENE:

These are the people who work to maintain the DB system environment but who are not actively interested in the DB itself. They include the follow -ing categories:

1) DBMS system designers and implementers: The DBMS system designer knows the h/w requirement the system and produces an efficient system design for the DBMS. The DBMS system implem -nter are those who help for h/w implementation of the entire system. They do not play any role in the s/w implementation.

2) Tool developers: These work is to develop s/w tools to enhance the performance of DBMS package. Eg to debu SQL program they may develop a good debugging tool.

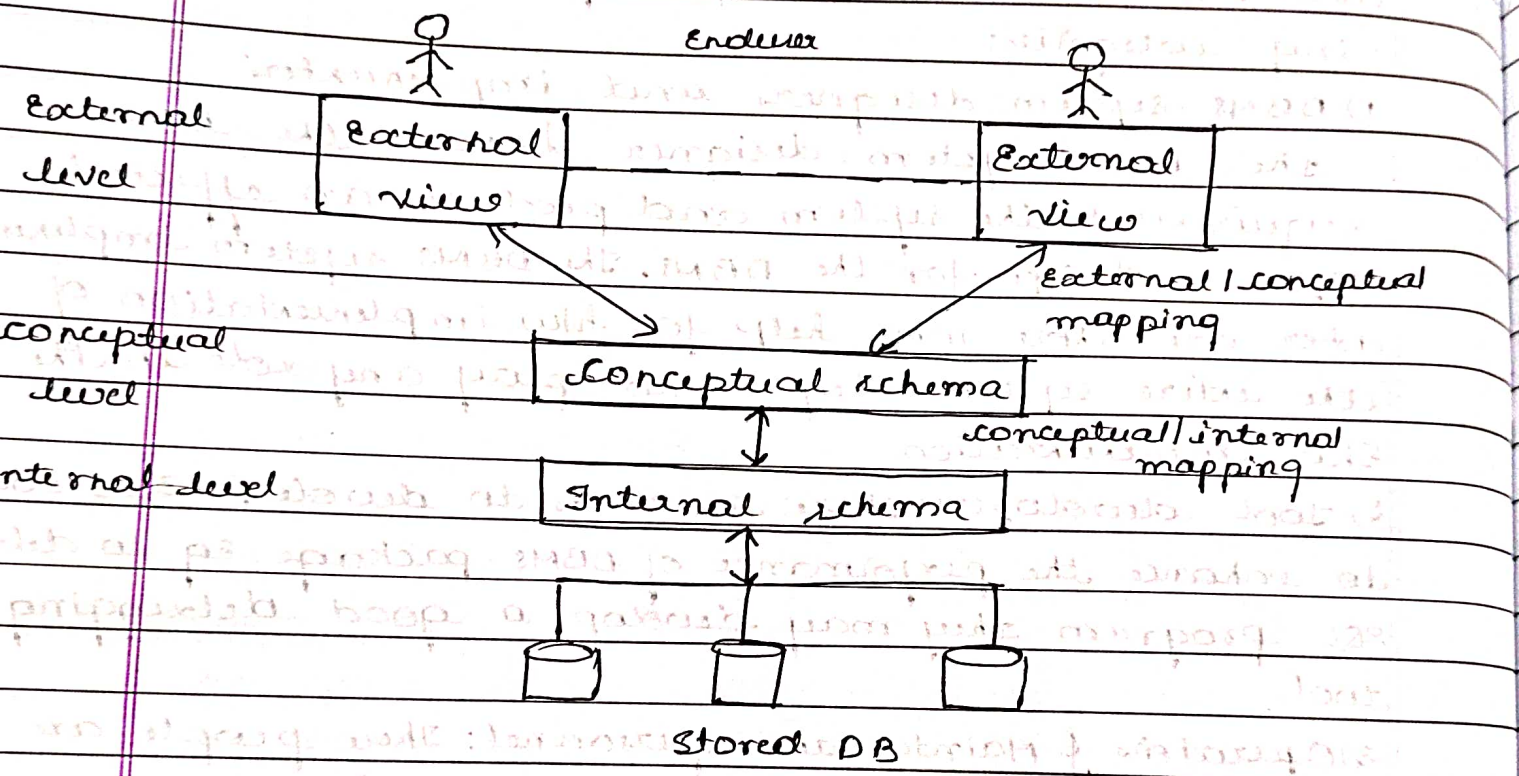
3) Operators & Maintainance personnel: These people are responsible for the actual running & maintaining of the h/w & s/w environment of the database. For eg a careful decision must be made as which DB package need to be bought for the organisation.

5) Describe the three-schema architecture. Why do we need mappings between schema levels? Explain about data independence.

In order to achieve data and program independence support of multiple user view, a three-layer architecture is proposed.

The main goal of the 3-schema architecture is the seperate the user applications of the phepical DB. In the architecture, schema can definition at the following 3 levels:

1) Internal level or physical level
2) Conceptual level or logical level
3) External level or view level.



1) External level or view level: This level includes no of user views. Each external schema describes the part of the DB that a particular user group is interested in and hides the rest of the DB from that user group.

2) Conceptual level or logical level: This level has conceptual schema, it describes what data are stored the DB and what relationship exists among those data.

This level of abstraction is used by DB admin

must decide what information is to be kept in the DB.

3) Internal level or physical level: This level has an internal schema, it specifies additional storage details and describe how the data is actually stored on the physical device i.e secondary storage device.

Data Independence:

The ability to modify a schema definition in one level without affecting a schema definition in the next higher level is called data independence. There are 2 levels of data independence.

1) Logical data independence: It is the capacity to change the conceptual schema without having to change external schema.

2) Physical data independents: It is the capacity to change the internal schema without having to change the conceptual schema changes to the internal schema may be needed : some physical files were reorganize to improve the performance of retrieved or update.

6) What is data model and explain its categories

Data Models is a collection of concepts that can be used to describe structure of the DB, which provided the necessary means to achieve abstraction, structure of DB means datatypes, relationships, constraints that should hold for the data.

categories of Data Models:

Based upon the structure of the DB, data models are categorized into:

1) High-level or conceptual data Model:
This model provides the concepts that are close to the way many users perceive data. eg: ER Model
This uses concepts like Entity, relation, attributes to describe the data model.

a) Entity: An entity is one that represents a real world object. eg: Employee, project.

b) attribute: An attribute is the property that further describes an entity. eg: for employee: Empid, Ename, age, salary.

c) Relationship: The relation between two or more entities called as a relationship that describes the interaction between the entities. eg: borrows relationship. This relationship exists between the customer & book entities.

2) Low level or physical data Model: This model provides the concepts that describe the details of how data is stored in the computer (physical storage). This model is used for DBMS developers not for end users. Eg: record formats, record ordering access path, etc.

3) Representational or Implementational Data Model: This model represents in between high & low data model, which means to provide concepts that may be understood by enduser and developers. This model hides some details of data storage, but can be implemented on a computer system directly.

4) Define the following terms:
i) database instance:
ii) Database schema
iii) Database state.

The description of a DB is called the database schema, which is specified during DB design and is not expected to change frequently. A displayed schema is called a schema diagram.

student

name	RollNo	class	Major

section

SID	ENUM	SEM	Year	Instructor

course

cname	cnum	creditHr	Dept

grade_report

RegNo	SID	grade

The diagram displays the structure of each record type but not the actual instances of records.

The actual data in a DB changes everytime we add a student. The data in the DB at a particular moment time is called the instance of the DB or DB state.

For eg: student table contains 2 records, if a record is added to student table, the state of the table from 2 to 3 records. This replies that the instances of the DB may change time to time if it is called as DB state.

Name	RollNo	Class	Major
Scott	MCA001	6	MCA
Tiger	CS003	4	CS

When new DB is created the DB state is Empty state as there is no data.

2) Explain the component modules of DBMS and their interaction with the help of a diagram OR explain DB system environment with a diagram OR explain the interaction of DBMS component modules with database system environment.

The fig illustrates the typical DBMS component module. The fig is divided into 2 halves. The top half of the fig refers to the various users of DB environment & their interfaces. The lower half shows the internal structure of the DBMS responsible for storage of data & processing of transactions.

The DB and the DBMS catalog are stored on the disk. Access to the disk is controlled by the OS. A higher level stored data manager module of the DBMS controls access to DBMS information i.e, stored on disk.

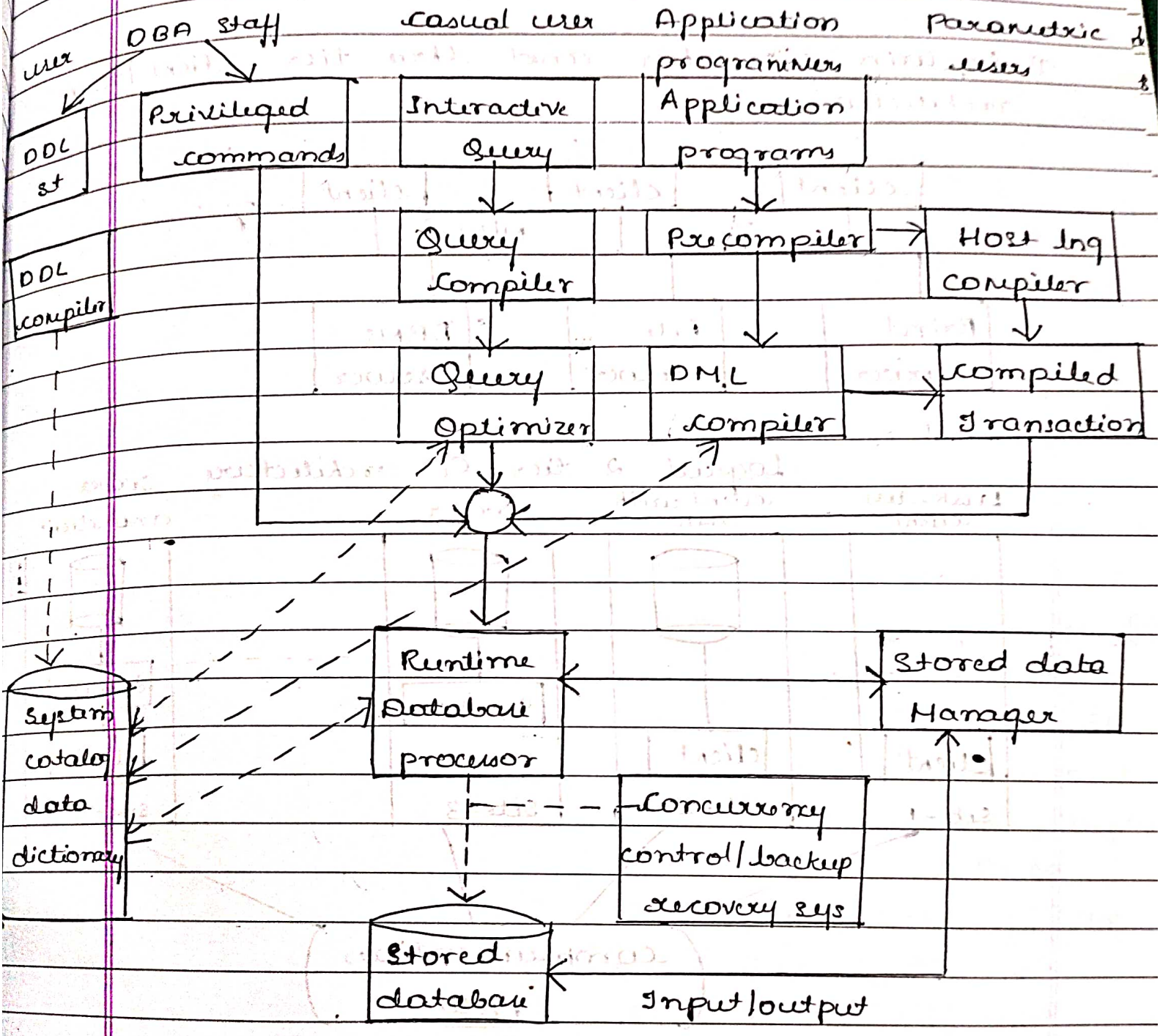
Let us consider the top half of the fig first. It the interfaces for DBA staff, casual user, APP programmers & parametric users.

The DBA staff works on defining the DB and tuning it by making changes to its defination using the DDL & other privileged commands. The DDL compiler processes schema defination & stores describe of the schema in the DBMS catalog. DBMS SW modules then look up the catalog information as needed.

A casual user interact with DB using interactive Query interface. Their Queries are analyzed for correction of the operations for the model by a query compile and the query optimizer is concerned with elimination of redundancies & use of correct algorithm during execution.

Application programmer write program in host languages such as Java, C or COBOL. The pre compiler extracts DM commands from an application program written in a host program language. Ther commands are sent to the DML compiler for compilation into

object code for DB access. The rest of the program is sent to the host lng compiler. The object codes for the DML commands & the rest of the program are linked to the runtime DB process

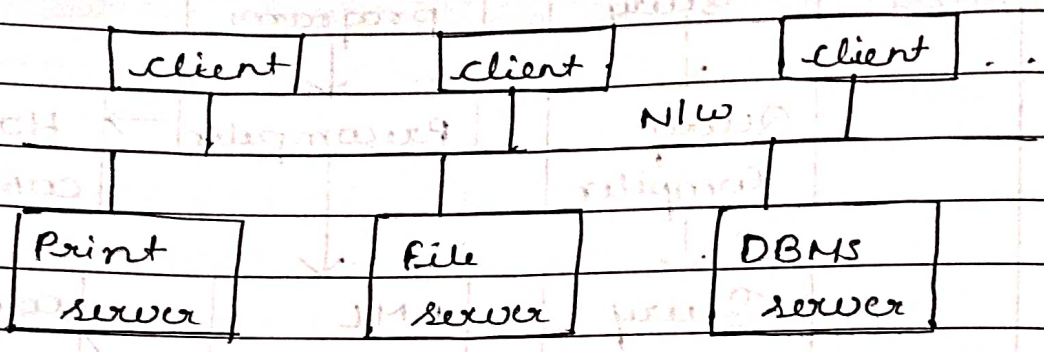


Component modules of DBMS & their interaction

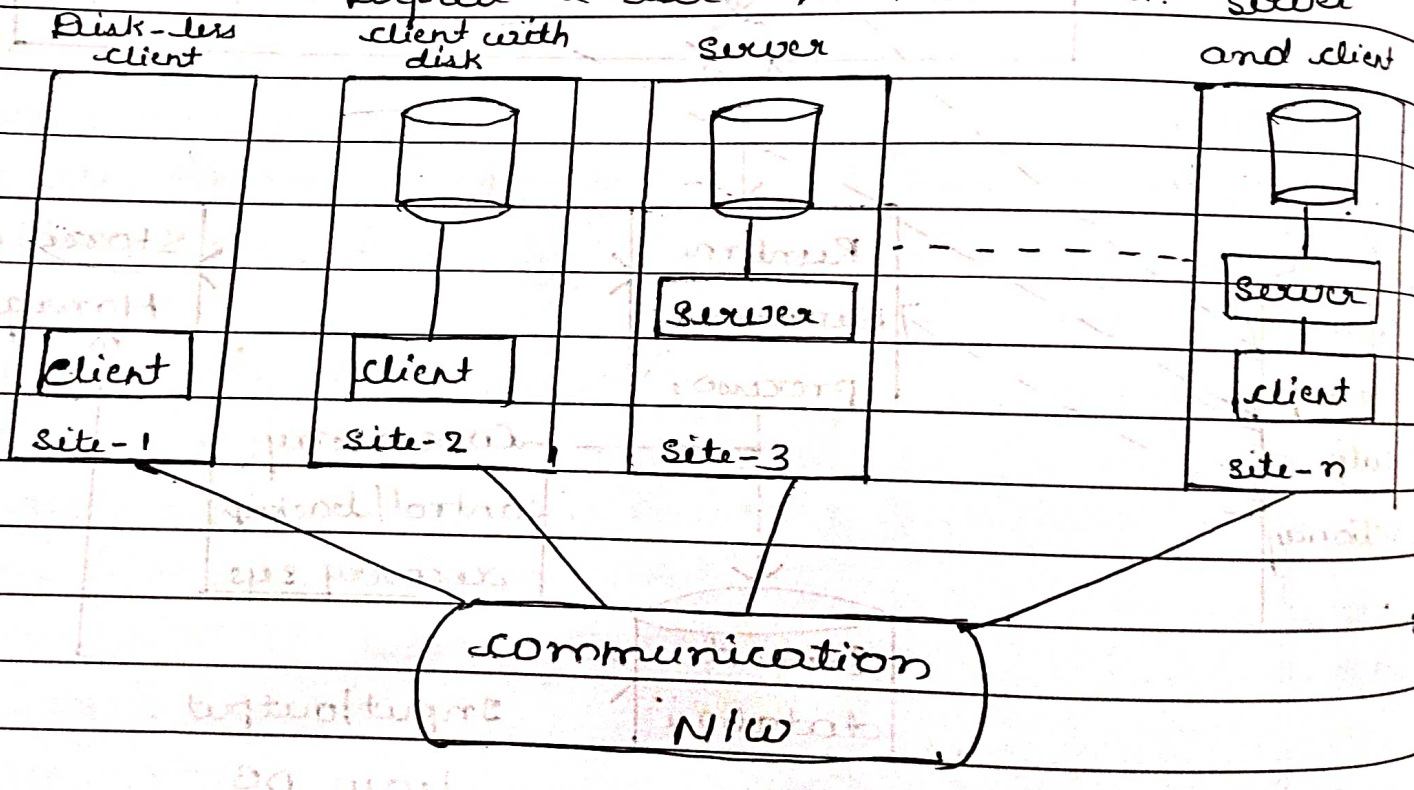
In the lower half of the fig the dotted lines are controlled by stored data manager. The stored data manager use OS services for carrying out

low-level data transfer b/w disk & main memory. once the data is in main memory buffer, it can be processed by other DBMS modules. some DBMS have their own buffer manager module. while other depend on the OS for buffer management.

9) Explain two-tier and three-tier client/server architecture.



Logical 2 tier C/S architecture.



A client in this framework is typically a user m/c that provides user interface capabilities & local processing. When a client request access to additional function that provides the needed functionality

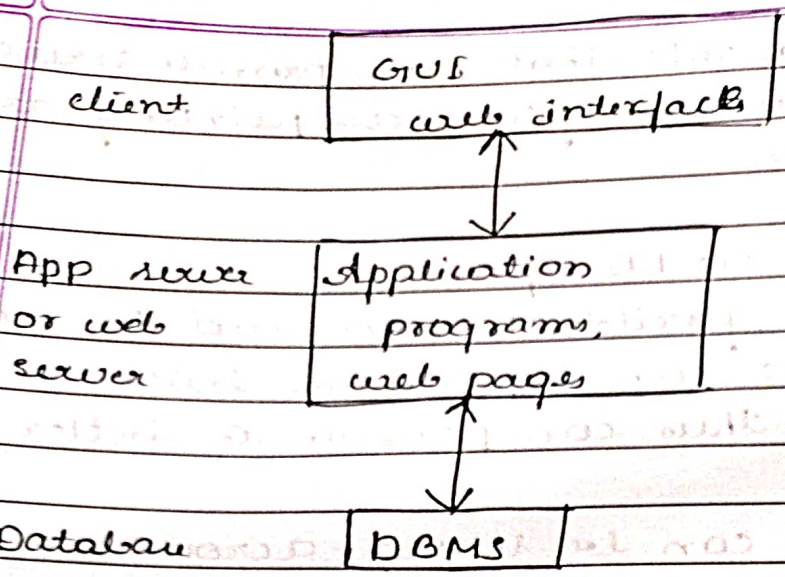
A server is a m/c that can provide services to the client m/c such as file access, printing, archiving or DB access.

* Advantages of C/S DB system:
→ This is more flexible as compared to centralized
→ The server m/c can be custom-built to the DBMS function and thus can provide a better DBMS performance.
→ A single DB can be shared across several distinct client system.

Two main types of basis DBMS arch were created on this underlying C/S framework:
2 tier &
3 tier.

1) Two-tier C/S Architecture for DBMS:
In this C/S architecture the query & transaction functionality remained on the server side, & the user interface program & app program can run on the client side. When DBMS access is required, the prgram establishes connection to the DBMS using open DB connectivity (ODBC) Once the connection is created, the client program to communicate with the DBMS, most DBMS vendors provides ODBS drivers for their systems.
This arch is called 2-tier. the S/w components are distributed over 2 systems i.e, client and server.

2) Three-tier & N-tier C/S architecture for web applications.
This arch adds an intermediate layers btw the client & the server as shown below.

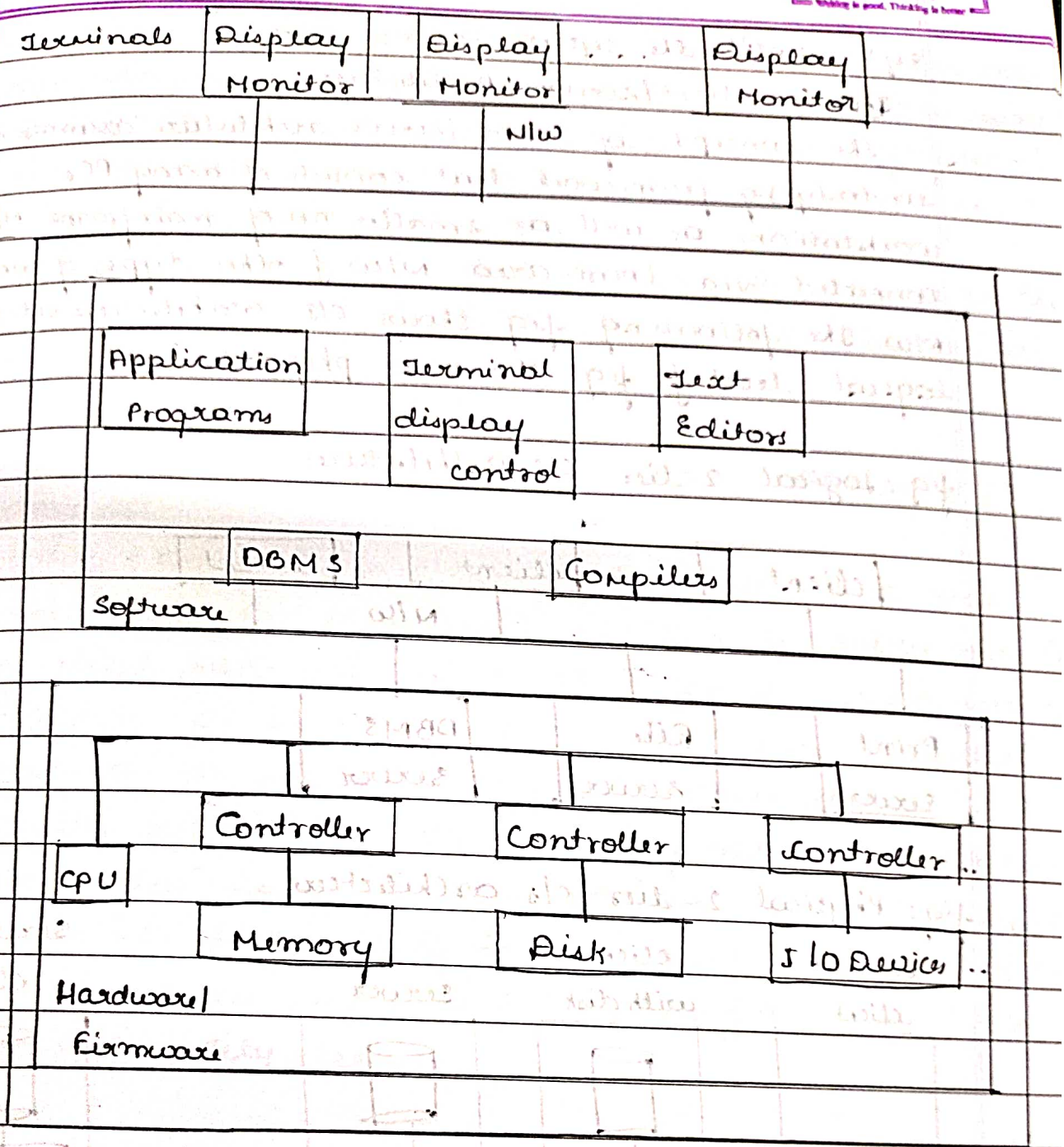


This intermediate layer is called app server web server depending on the application. client contains GUI interfaces. The intermediate server accept requests from the client, process the request & sends DB commands to the DB server, & then acts as a channel for passing processed data from the DB server to the client. Thus the user interface, app rules & data access act as 3 tier.

10) Explain centralized and client server architecture with a diagram.

Centralized DBMS architecture:

It consists of a single processor together with its associated data storage devices & other peripherals. It is physically confined to a single location. In centralised DBMS the DBMS functionality, application program execution & user interface processing were carried out on one machine. The following fig illustrates the physical components in a centralized architecture



Physical centralized architecture.

The main advantage of centralized DB system.

* Most of the function such as update, backup, query, control access are easier to accomplish in CDB system
* The size of the DB of the computer on which it resides need not have any bearing on whether DB is centrally located.

The disadvantage is that when the central site computer goes down, then every user is blocked from using

sys until the system comes back.

Basic client/server Architecture:

The concept of client/server architecture assumes an underlying framework that consists of many PCs workstations as well as smaller no of mainframe HLL connected via local area N/w & other types of computer N/w. The following fig shows C/S architecture at the logical level & fig shows at physical level.

fig: logical 2-tier C/S architecture.

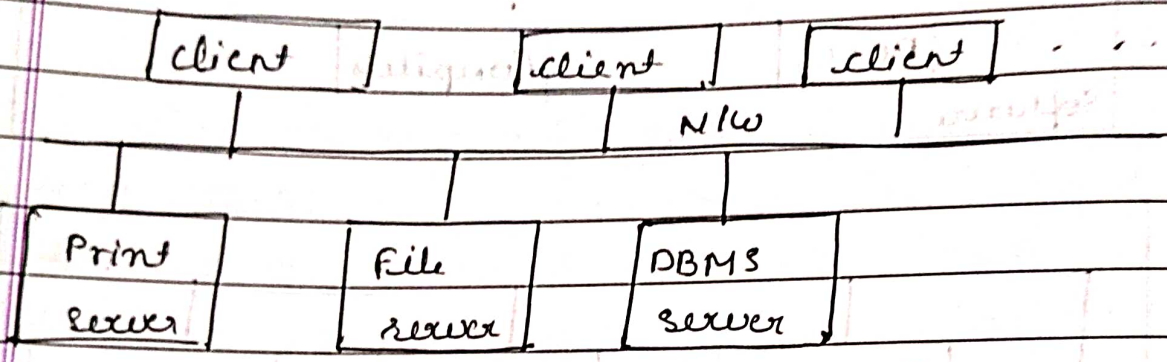
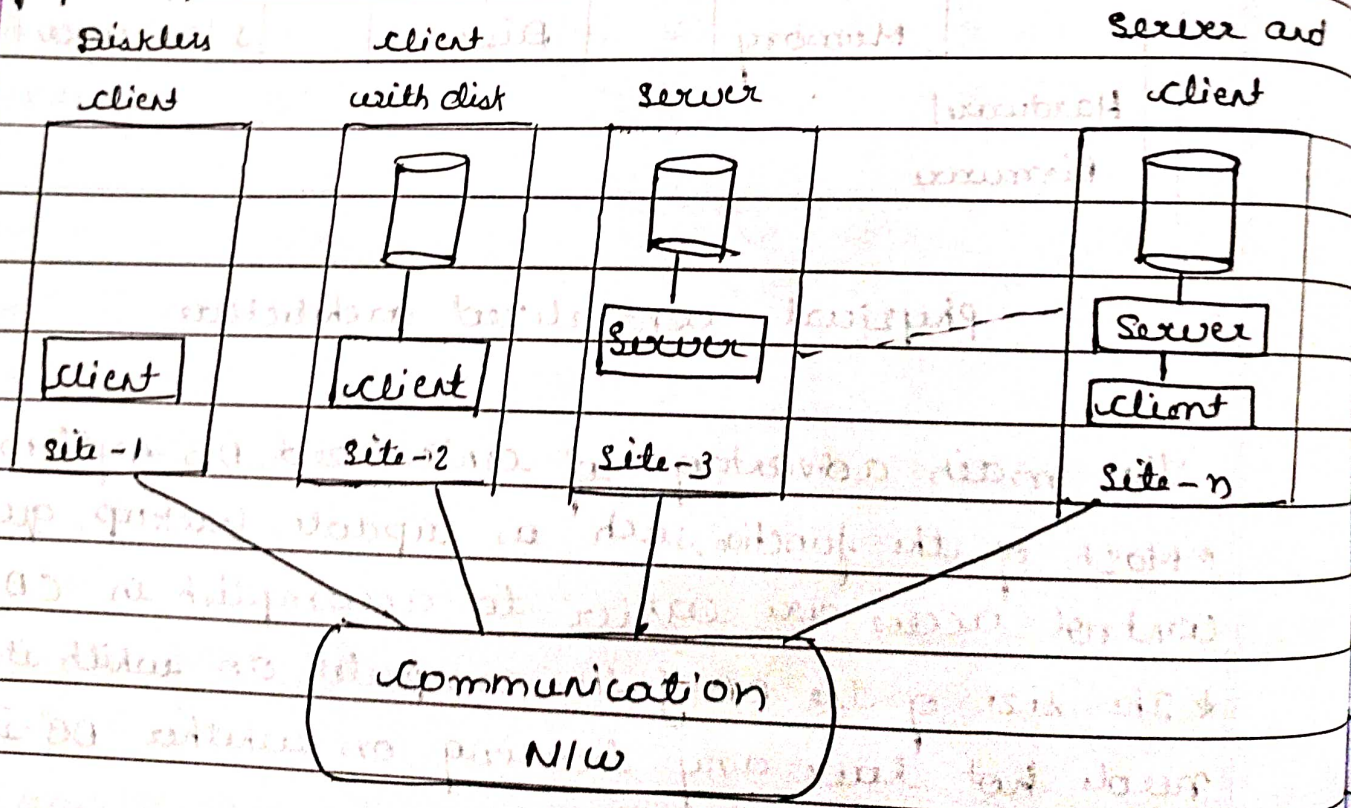


fig: Physical 2-tier C/S architecture.



A client in this framework is typically a user m/c that provides user interface capabilities & loc processing When a client request access to additional function that does not exists at that m/c, it connects to a sequence that provides the needed functionality.

A server is a m/c that can provide services to th client m/c such as file access, printing, archiving or DB access.

11) Define the following terms Entity, Entity type & entity set.

1) Entity: An entity is anything that exists in a real world with an independent existance. An entity may b an object with physical existance for eg person, car o employee; or it may be an object with conceptual existance for eg: company, job, course etc.

2) Entity types: An entity type defination a collection entity that have the same attribute. Each entity type in the DB is describing by its name & attributes. Fig sh 2 entity types employee & company & a list of attribute for each entity set.

Entity type 1) Employee 2) Company

Collection of entities Name age salary Name Head q, presiden wing common attr

Entity set

e_1	c_1
John, 55, 80K	Sun, UK, John
e_2	c_2
Tred, 40, 30K	Fast cmp, canada,
e_3	Bob
Cleark, 25, 20K	c_3

c) Entity set:

The collection of all entities of a particular entity type at any point in a time is called entity set.

eg: e1

John, 55, 80k.

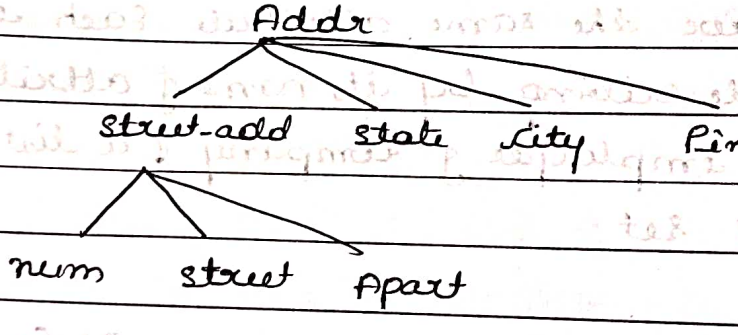
10) Explain the different types of attributes that occur in an ER model with an eg:

1) Composite versus simple attributes:

Simple attribute cant be feb divided, i.e each entity has a single atomic value for the attribute for eg SSN

Composite attribute can be divided into smaller subpart for eg. Address attribute of the employee entity can be subdivided into Street-address, state, city, pincode.

It can form a heirarchy for eg street address can be ulb divided into neum, street, apart neum.



2) Single valued versus Multivalued attributes:

Single valued attribute have only one value for 1 perticular entity set.

eg: age of a person.

Multivalued attribute have more than one value for the same attribute.

eg: color of a car = {red, white, black}

college degree = {BSc, MCA}

3) Stored versus derived attribute:

The values of the stored attribute cannot be derived from some other attribute.

eg: SSN, DOB, Id.

The values of the derived attribute cannot be derived from some other attribute in an entity set

eg: Age can be derived from DOB, by calculating the difference between sysdate and DOB.

eg: Birth date - stored
Age - derived

Joining date - stored
Yr of Exp - derived

4) NULL Values:

A particular entity may not have the value for particular attribute i.e the value may not be applicable for value is unknown, in such a cases special value NULL is written.

eg: Every employee may not have fax number.

5) Complex attributes:

It is the combination of composite and multi-valued attribute composite can be nested arbitrary components of composite attribute can be shown in () & the multivalued attribute can be shown in {}

eg: If a person have more than one residence & each residence can have multiple phones, an attribute addr. phone for a person is.

{ Addr_Phone ({ Phone (area_code, Phone_num)}, Address (Street_addr (num, street, Apart_num), city, state))}

6) Key attributes:

An attribute of an entity type for which each entity must have a unique value is called a key

attribute of the entity type i.e its values can be used to identify each entity uniquely
eg: SSN of Emp, sid of student.
A key attribute may be composite i.e several attributes together form a key called composite key; meaning is that the combination of the attribute values must be distinct for each entity set of entity type.
eg: (Vehicle-id, registration) of car entity type.
All key attribute should be underlined in ER diagram
Selected key will work as primary key, other potential key will be alternate keys.
An entity type may also have no key, in which case it is called weak entity type.

1) Value sets (domains) of attribute:
Domain specifies the set of permitted values for each attribute for each individual entity. For eg
* Birthdate : Range of all valid dates.
* Ename : 20 Chr long, should have only alphabets [A-Z] [a-z] & blank space.
* Dcode : set of all department codes
These value sets are specified using the basic data types, size & other constraints. Value sets provides all possible values.

13) Define Cardinality ratio, participation and weak entity type.
Cardinality ratio for a binary relationship specifies the num of relationship instances that an entity can participate in the possible cardinality ratio for binary relationship types are
a) 1:1

b) 1:N

c) N:1

d) M:N

Participations constraints:

Apart from the key constraints and cardinality mapping it is necessary to incorporate whether participation of an entity to a relationship is total or partial.

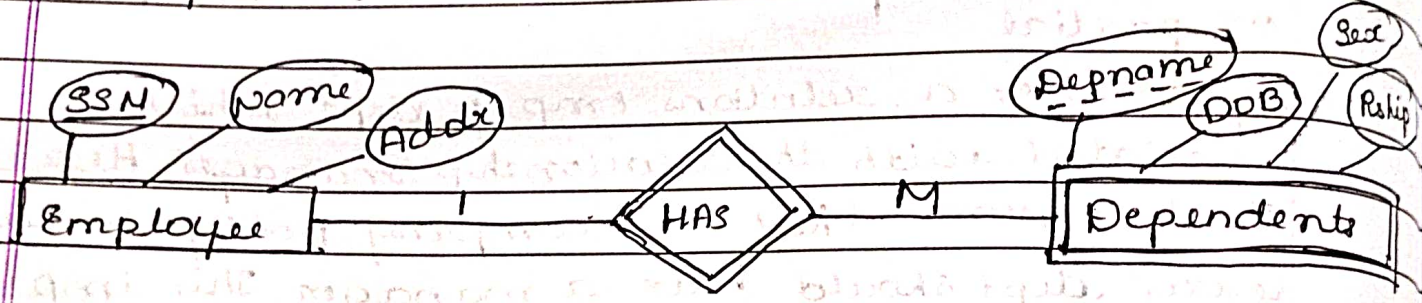
eg: Consider a relations emp & dept which are associated with the relationship 'manages'. Here we shall assume that the company policy is that every dept should have a manager. This implies that there will be full/total participation in dept side and this type of participation is called as 'total participation'. However the participation in the empside is not total: not all Emp are manage The double line in the fig indicates total participa

When an entity set E doesnot participate fully with a relationship R in turn to the participating entity such a constraint is called as partial participation. In the above eg on the emp side the participation is partial: not all the Emp are managers.

Weak entity type:

Entity type that do not have key attribute of their own are called weak entity types. entities belonging to a weak entity types are identified

by being related to specific entities from another entity type in combination with one of their attr values. We call this other entity type relationship type the identifying relationship of the weak entity type. A weak entity set will always have a total participation with identifying over owner entity set, the owner entity set & weak entity set must participate in 1:M relationship.



eg: Consider that the company wishes to store the Emp's dependents in an entity set called 'Dependents' with the attribute Depname, sex, DOB, relationship. Since the dependents entity set cannot have its own primary key, it has to be considered as WES.