DataBase Management System

Indira Gandhi

Unit – I

Shruti Jain

Database Management System :- DBMS is a collection of interrelated Data and a set of progs. to access those data. The collection of data, usually reffered to as the Database, contains info^n relevant to an enterprise.

Database
- relational (data stored in tabular form)
- hierarchical (data organised in tree structure)
- network (data stored in graph representing relationship bet^n object)

Database System Applications

① **Banking :-** For customer Info^n, accounts, loans and banking transactions.

② **Airlines :-** For reservation and schedule info^n

③ **Universities :-** For student Info^n, course registration, and grades.

④ **Credit Card Transactions :-** For Purchase on credit cards and generation of monthly statements.

⑤ **Telecommunication :-** For keeping records of calls made generating bills, storing info^n

(6) Finance :- For storing infor^n about holdings, sales and purchases of financial instruments such as stocks and bonds.

(7) Sales :- For customer, product and purchase infor^n

(8) On-line Retailers :- For sales data noted above plus on-line order tracking, generation of recommendation lists.

(9) Manfacturing :- For management of the supply chain and for tracking production of item in factories.

(10) Human Resources :- For infor^n about employees, salaries payroll taxes, benefits and generation of paychecks.

Purpose of Database System (File Sys. Vs DBMS)

→ Database sys. arose in response to early methods of computerized management of commercial Data.

→ The Typical "File Processing System" is supported by a conventional OS. The sys. stores permanent records in various files, and it needs diff Appl^n progs to extract records ~~from~~, and add records to the appro-priate files. Before DBMS came along, organization

actually stored info in such sys.

→ File sys. has a no. of major Disadvantages

① Data Redundancy and inconsistency :- (Redundancy leads to higher storage and access cost. In addition, it may lead to data inconsistency) that is the various copies of the same data may no longer agree.

② Difficulty in Accessing Data :- The point here is that conventional file processing environment do not allow needed data to be retrieved in a convenient and efficient manner. More Responsive data retrieval sys. are required for general use.

③ Data Isolation :- (becoz data are scattered in various files, and files may be in diff format writting new appl^n progs to retrieve the appropriate data is difficult)

④ Integrity Problem :- The data values stored in the database must satisfy certain types of consistency constraints. The Problem arise when constraints involve several Data items from Diff. files.

Eg:- Minimum balance for an Account.

5) Atomicity Problems :- A computer sys., is subject to failure. In many appln, it is crucial that, If a failure occurs, the data be restored to the consistent state that existed prior to the failure. eg: It is essential to database consistency that either both the credit and debit occur, or that neither occure during amount transfer from one acc^nt to another Account i.e. the funds transfer must be atomic it must happen in its entirety or not at all.

6) Concurrent - access anomalies :- Interaction of concurrent updates is possible and may result in inconsistent data.) eg:- Banking Transaction for two concurrent withdrawal for amount $50 and $100 is fired from amount $500 so the amount reflects $450 or $400 when both process read amount $500 and performed separatly and shows inconsistent results.

7) security Problem :- Not every user of the database sys: should be able to access all the data.)

→ These Difficulties of file Database system will be over-come in Database Management Sys. (DBMS).

View of Data

→ A Database sys is a collection of interrelated data and a set of progs that allow users to access and modify these data. A major purpose of a database sys. is to provide users with an abstract view of the Data. i.e. the sys hides certain details of how the Data are stored and maintained.

* Data Abstraction :- Since many database sys. users are not computer trained, developers hide the complexity from users though several levels of abstraction, to simplify user's interaction with the system:

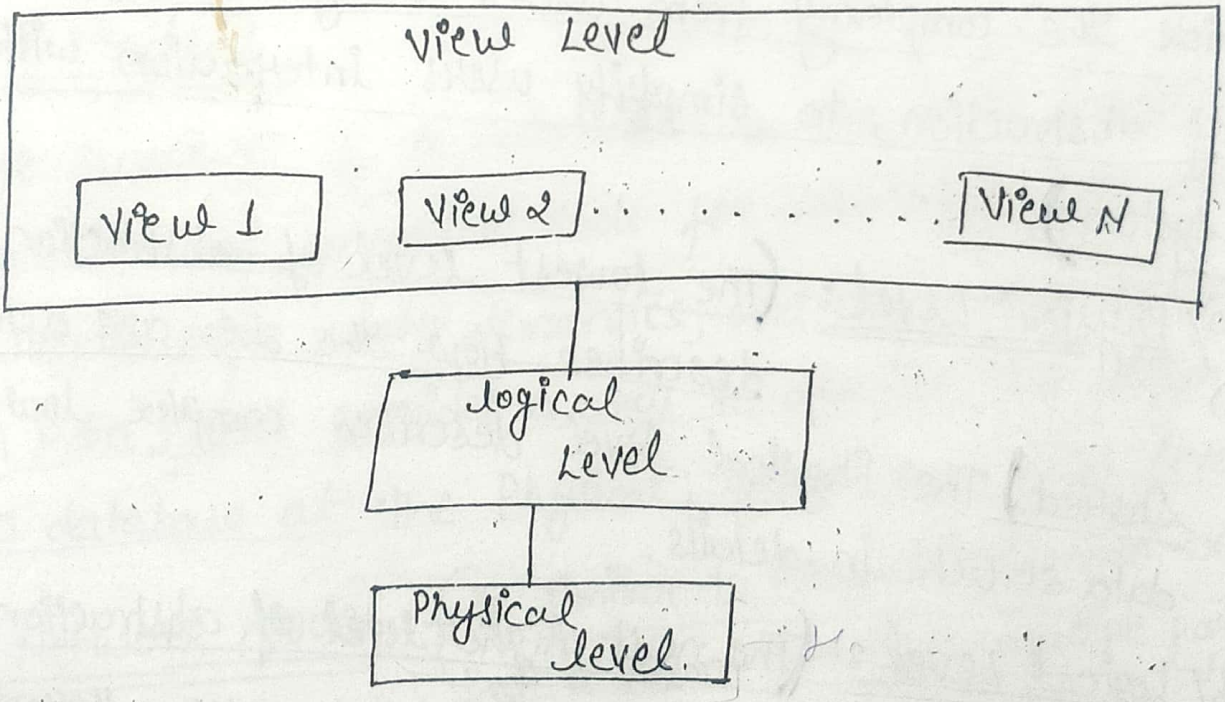
a) Physical Level :- The lowest level of abstraction describes How the data are actually stored. The Physical level describes complex low level data struct^{r} in details. (How)

b) Logical Level :- The next higher level of abstraction describes what data are stored in the database and what relationships exist among those data. The logical level thus describes the entire database in terms of a small no. of relatively simple structures. Although implementation of simple struct^{r} (What)

at logical level may involve complex physical level struct^r ; the user of the logical level does not need to be aware of this complexity.

c) View level :- The highest level of abstraction involves only part of the entire database.

Even through the logical level uses simpler struct^r, complexity remains becoz of the variety of infor^n stored in a large database.



→ Instance and schemas :-

→ Databases change overtime as infor^n is inserted and deleted. The collection of infor^n stored in the database at a particular movement is called "instance" of the database.

→ The overall design of the database is called the database schema. schema are changed infrequently

→ Database sys. have several schemas, partitioned acc to the level of abstraction. The Physical schema describes the database design at the Physical level, while the logical schema describes the database design at the logical level. A database may also have several schemas at the view level: sometimes called subschemas, that describes diff views of the database.

Data Models

→ The structure of a database is the data model: a collection of conceptual tools for describing data, data relationships, data semantics, and consistency constraints.

→ A Data Model provides a way to describe the design of a database at the Physical, logical and view level.

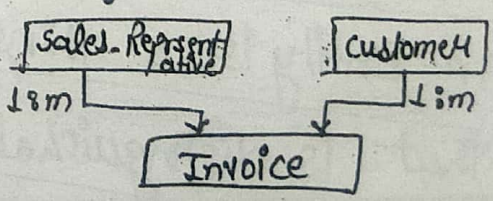
① **Network Model:-** is similar to hierarchical model except that a record can have multiple parents

The network data model has three basic: components such as record type, data items and links.

Relationship is called set which consist at least two record type.

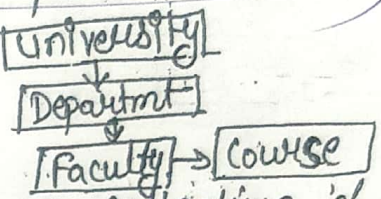
① owner record / Parent Record

② member record / child record



(2) Hierarchical Model :- The hierarchical Data model is represented by an upside-Down tree.

The hierarchical database is a collection of records that is perceived as organised to conform to the upside Down tree struct. A hierarchical model can represent a one-to many relationship b/w two entities where the two are respectively parent and child. The nodes of the tree represent record type.



(3) Relational Model :- This model uses a collection of tables to represent both data and the relationship among those data. Each Table has multiple columns, and each columns has a unique name. The relationship Model is also called record Based Model becoz the database is structured in fixed format records of several types. The column of the table correspond to the attributes of the record type.

(4) The Entity Relationship Model :- is based on a perception of a real world that consists of a collection of basic objects called entities and of relationship among those object. An Entity is a thing or object in the real world that is distinguishable from other object.

(5) Object Based Data Model :- is that has seen increasing attention. This model can be seen as extending E-R Model with notions of encapsulation, methods, and object identity. The Object-relational Data Model combines features of the object oriented Data Model and relational Data Model.

(6) Semistructured Data Model :- permits the specification of data where individual data items of the same type may have diff set of attributes. This is in contrast to the data models mentioned earlier, where every data item of a particular type must have the same set of attributes. The Extensible Markup lang. (XML) is widely used to represent Semistructured Data Model.

Database Languages

→ A database sys. provides (1) a data defination lang. to specify the database schema and (2) a data manipulation lang. to express database queries and updates.

(1) Data Manipulation Lang:- (DML) is a lang. that enables users to access or manipulate data as organized by the appropriate Data Model. The type of access are:-

* Retrieval of infon
* Insertion of new infon
* Deletion of infon from the Database.
* Modification of infon stored in Database.

→ There are basically Two types:-

a) Procedural DMLs require a user to specify what data are needed and how to get those data.

b) Declarative DMLs (Nonprocedural) require a user to specify what data are needed without specifying how to get those data.

A Query is a statmt requesting the retrieval of infon. The portion of a DML that involves infon retrieval is called Query lang. Eg:- SQL.

→ The level of abstraction apply also to manipulation of data. At the Physical level, we must define algo that allow efficient access to data. At higher

level of abstraction, we emphasice ease of use. The goal is to allow humans to interact efficiently with the sys.

② Data Defination Lang:-(DDL)

→ We specify a database schema by a set of definations expressed by a special lang called a data defination lang. It also specify additional properties of the data.

→ We specify the storage struct' and access method used by the database sys. by a set of stmts in a special type of DDL called Data storage and Defination lang. These stmts define the implementn details of the database schemas, which are usually hidden from the user.

→ Database system concentrate on integrity constraints that can be tested.

a) Domain Constraints:- A domain of possible values must be associated with every attribute (eg. integer type, character type etc.) Domain constraints are the most elementary form of

integrity constraints. They are tested easily by the sys: whenever new data item is entered into the database.

(b) Referential Integrity :- There are cases where we wish to ensure that a value that appears in one relation for a given set of attributes also appears for a certain set of attributes in another relation. Database modification cause ~~voit~~ violations of referential integrity. When a referential integrity constraint is violated the normal procedure is to reject the action that cause the violation.

(c) Assertions :- An assertion is an cond^n that the database must always satisfy. Domain constraints and referential integrity contraints are special form of assertion. When an assertion is created the sys. test it for validity, if this is valid~~ity~~ then any future modification to the database is allowed only if it doesn't cause that assertion to be ~~violated.~~ violated.

(d) Authorization :- we may want to differntiate among the users as for as the

Type of access they are permitted on various data values in the database.

This differentiation are expressed in terms of authentication. The most common being read authorization which allows the reading but not modification of data.

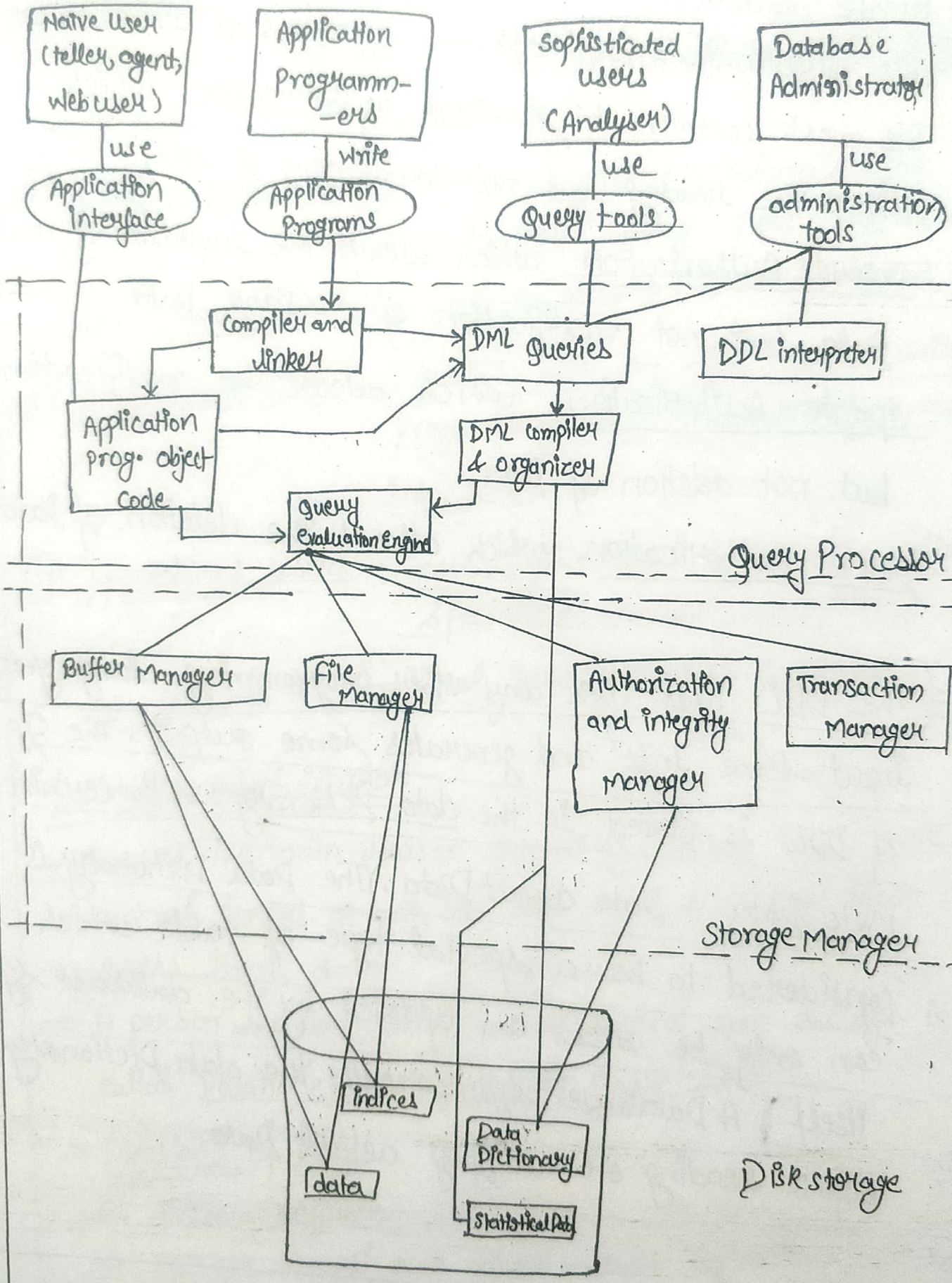
Insert Authorization which allows the insertion of new Data but not modification of existing Data.

Update Authorization which allows the modification but not deletion of Data.

Delete Authorization which allows the deletion of Data.

→ The DDL just like any other programming lang. gets input some Instr and generates some output. The Op of DDL is placed in the data Dictionary. With certain meta Data - Data about Data. The Data Dictionary is considered to be a special type of table which can only be access an updated by the database sys. itself. A Database sys. consults the data Dictionary before reading or modifying actual Data.

DataBase Architecture



Database users and user interfaces

① Naive Users :- are unsophisticated users who interact with the sys. by invoking one of the appln prog. that has been written previously.

② Appln Programmers :- are computer professionals who write appln prog. Appln programmers can choose from many tools to develope user interfaces.

③ Sophisticated users :- interact with the sys. without writing progs. They use query processor as interfaces.

④ Specialized users :- are sophisticated users who write specialized database appln that do not fit into the traditional Data processing framework.

Database Administrator

→ One of the main reason for using DBMSs is to have central control of both the data and the progs that access those data.

→ A person who has such central control over the sys. is called Database Administrator. ★ Ques Paper

→ Functions of DBA

a) Schema defination

b) storage struct and access method defination.

c) Schema and Physical organisation modification.

d) Routine maintenance :- * backing up database
* Free disk space availibility
* Monitoring jobs Runing on DB.

Storage Manager

→ A storage Manager is a prog. module that provides the interface blw the low-level data stored in the Database and the appln progs and queries submitted to the sys. The storage manager is responsible for the interaction with the file manager.

→ Major components

① Authorization and integrity Manager :- test Authority of users to access data, also ensure integrity constraints

② Transaction Manager :- Consistency checking and executing concurrent Transactions without conflict.

③ File Manager :- manages allocation of space on Disk.

④ Buffer Manager :- responsible for fetching Data from Disk storage to main memory.

→ Storage Manager implements several Data struct as part of the physical sys. implementation.

① Data Files. (Database itself)

② Data Dictionary (for Meta Data)

③ Indices :- Fast access to Data items.

Query Processor

→ Components

① DDL Interpreter :- interpretes DDL stmt and records the defination in Data Dictionary.

② DML Compiler :- DML stmts converted into low level instr so that the query evaluation engine understand that. also perform query optimization

③ Query Evaluation Engine :- executes low level instr

Transaction Management

→ A Transaction is a collection of opern that performs a single logical Funn in a database applⁿ. Each transaction is a unit of both atomicity and consistency

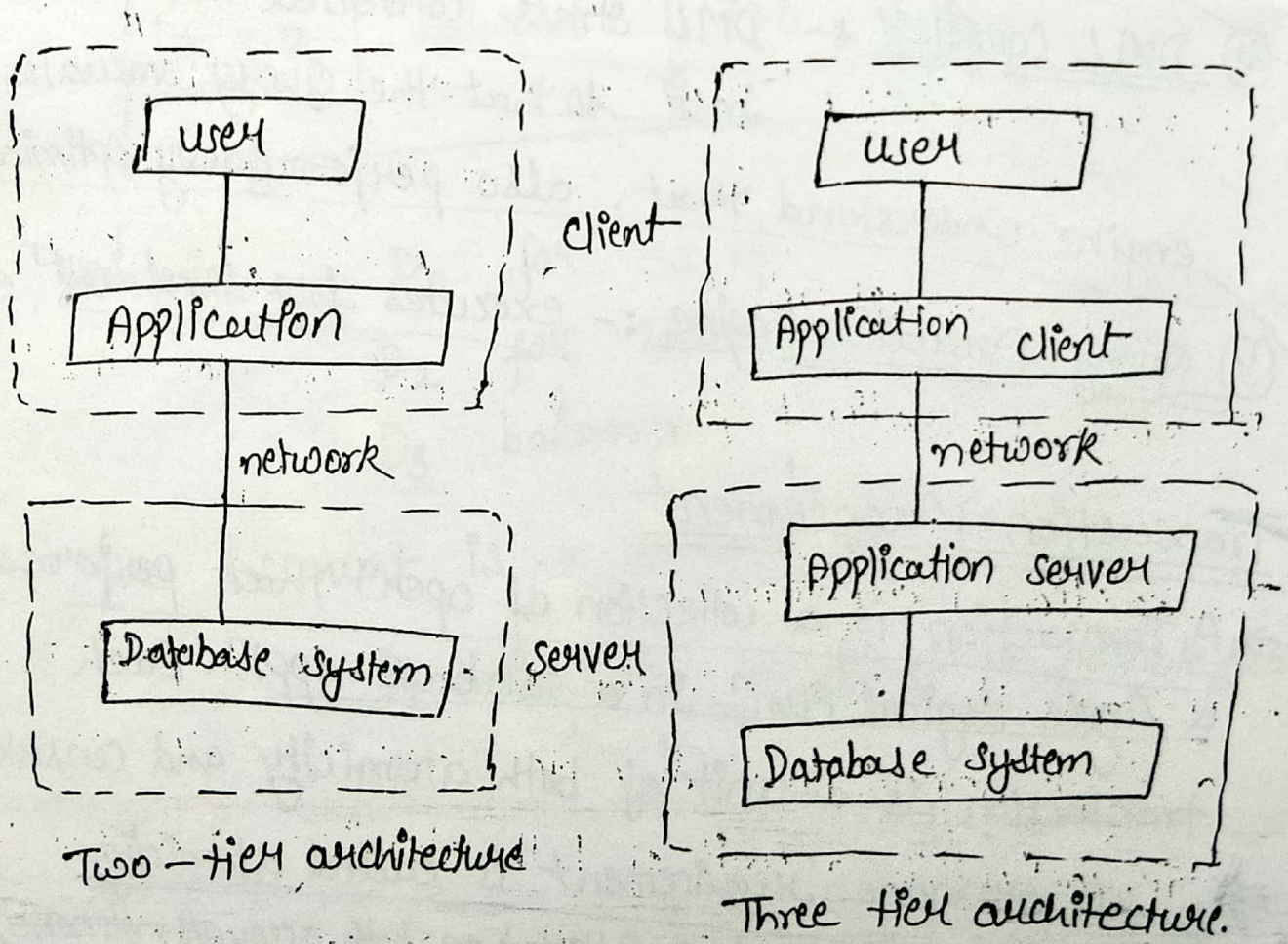
all or none requirement is called atomicity.
(Transfer of amount Reflected on both acounts or none)

* The correctness requirement is called consistency. (Amount of both of the data must be appropriate)

* ~~It's~~ The persistence requirement is called Durability. (Amount must persist in both Accounts despite sys failure).

→ If any of that fails there must be a recovery for the failure and concurrency manager for consistency and atomicity.

Two-tier and Three Tier Architecture



Structure of Relational Databases (RDBMS)

→ Consists of a collection of tables, each of which is assigned a unique name. A row in a table represents a relationship among a set of values.

→ the table is called entity set.

eg:- account

acc_no.	branch_name	balance
A_101	Downtown	500
A_102	Perryridge	400
A_217	Redwood	700

← attributes

} tuples/Rows

→ Domain → D_1 for acc_no
D_2 for brach_name
D_3 balance

the account is a subset of

$D_1 \times D_2 \times D_3$ Set of all possible Rows.

Database schema

→ It is logical Design of Database.

→ Database instance, is a snapshot of the data in the database at a given instant of time.

→ The concept of a relation schema corresponds to the programming lang. notion of type defination

eg:-

Account_schema = (acc_no, brach_name, balance)

account (Account_schema)

eg:- Branch_schema = (branch_name, branch_city, assets)

Entity sets :- is a set of entities of the same type that share the same properties or attributes.

→ Each entity has a value for each of its attributes.

→ An entity is represented by a set of attributes.

Eg:- customer ← entity set

customer_id	customer_name	customer_city ← attributes
321-12-3123	John	Harrison
244-63-8100	Jackson	Rye. ← value

Relationship set :- is a set of relationship of the same type.

⇒ If $E_1, E_2, E_3 \ldots E_n$ are entity sets. then relationship set R is a subset of :-

$$\{(e_1, e_2, \ldots e_n) \mid e_1 \in E_1, e_2 \in E_2 \ldots e_n \in E_n\}$$

where $(e_1, e_2, e_3 \ldots e_n)$ is a relationship.

Eg:-

customer

customer_id	customer_name	customer_city
321-12-3123	John	Harrison
244-63-8100	Jackson	Rye
963-96-3963	Smith	Woodside

loan

loan_no.	Amount
L-17	1000
L-11	5000
L-23	1300

→ The association b/w entity sets is referred to as participation.

→ sets $E_1, E_2, \ldots E_n$ participate in relationship A.

Attributes :- For each attribute, there is a set of permitted values, called the domain or value set of that attribute.

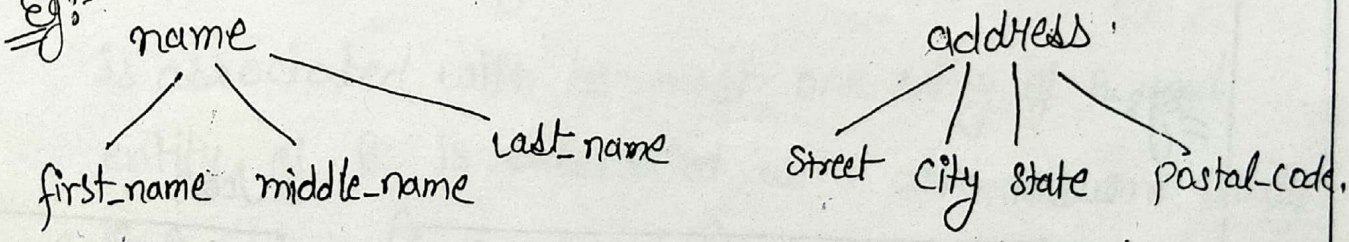
Types of attributes

① Simple and composite attributes :- ① Simple attributes have not been divided into subparts.

Eg:- customer_id.

② Composite attributes can be divided into ~~subparts~~.

Eg:-



② single_valued and multivalued :- single valued have single value for a particular entity. loan no.

Eg:- loan no.

→ A attribute have more than one value that attribute is known as multivalued Attribute.

Eg:- Phone_no.

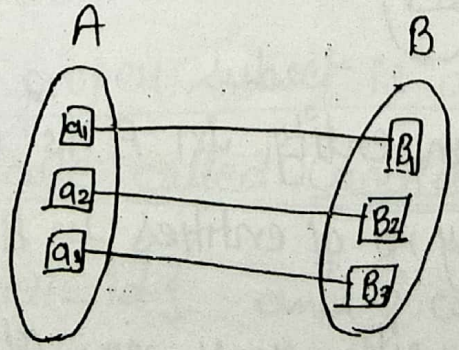
③ Derived attribute :- The value of this type of attribute can be derived from the value of other related attributes or entities.

Eg:- age ← dervided from DOB

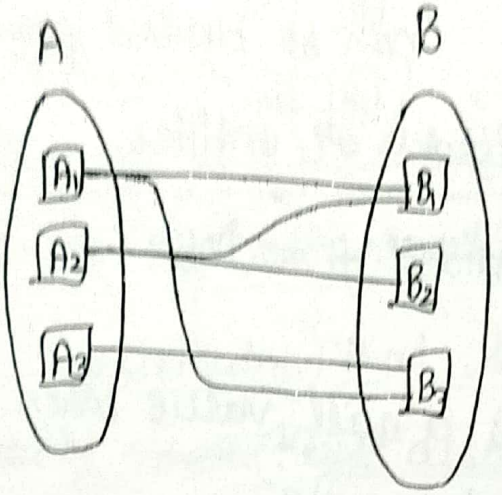
→ An attribute takes a null value when an entity does not have a value for it.

Mapping cardinalities :- or cardinality ratios, express the no. of entities to which another entity can be associated via a relation ship set.

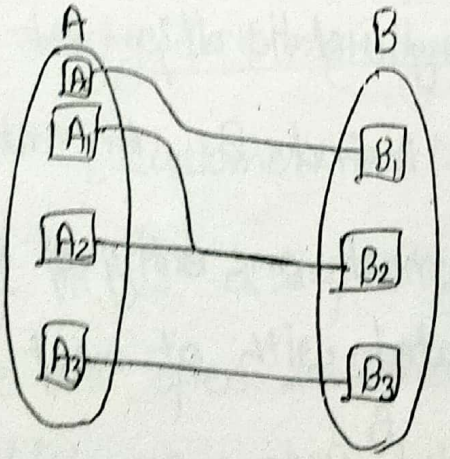
① one to one :- In a Binary relationship set R b/w entity sets A and B, An entity in A is associated with at most one entity of B and entity of B is associated with at most one entity in A.



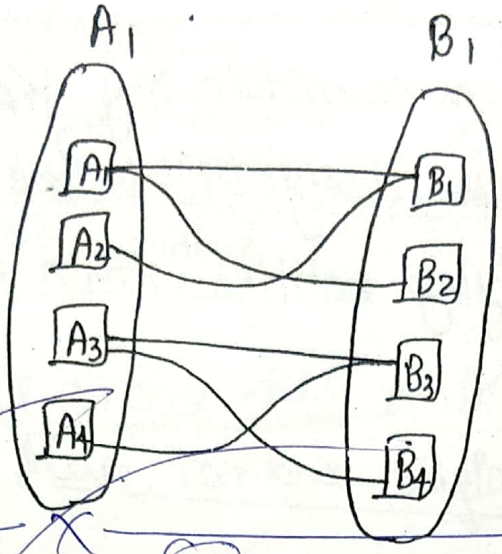
(2) One to many :- An entity of A is associated with any no. of entities in B, An entity of B can be associated with at most one entity in A



(3) Many to one :- An entity of A is associated with at most one entity in B, An entity of B can be associated with any no. of entities in A.



(4) Many to Many :- An entity in A is associated with any no. of entities in B and an entity of B is associated with any no. of entities in A.



Keys :- A Key allows us to identify a set of attributes that suffice to distinguish entities from each other. Keys also help uniquely identify relationships and thus distinguish relation ship from each other.

★ → A superKey is a set of one or more attributes that, taken collectively allow us to identify uniquely an entity in the entity set. → table

Eg:- { customer_id, customer_name }

→ If K is a superKey, then so is any superset of K we are often ~~instead~~ interested in superKeys for which no proper subset is a superKey. Such minimal subset of superKeys are called Candidate Keys.

Eg:- { cutomer_id } and { customer_name, customer_street }

(1) → A primary key to denote a candidate key that is choosen by the database designer as the principle means of identifying entities within an entity set.

(2) → primary key must be unique and not null.

Eg:- social-security no.

→ If a relationship set R has attributes $a_1, a_2 \cdots a_n$ associated with it then

$$R = \text{primary-key}(E_1) \cup \text{primary-key}(E_2) \cup \cdots \text{primary-key}(E_n) \cup \{a_1, a_2 \cdots a_n\}$$

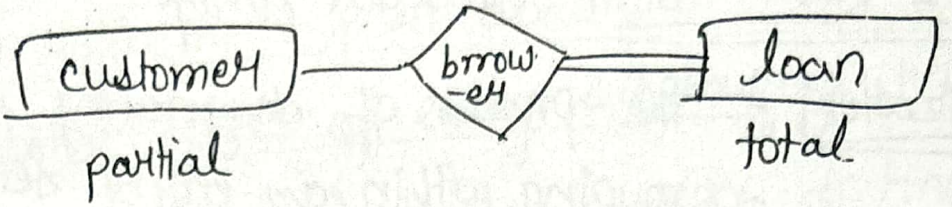
Underlined is superkey

Participation Constraints

→ The participation of an entity set E in a relationship set R is said to be total if every entity in E participates in at least one relationship R.

→ If only some entities in E participate in relationship in R, the participation of entity set E in relationship R is said to be partial.

Eg:- borrower Relationship



Weak Entity sets :- An entity set may not have sufficient attributes to form a primary key. Such an entity set is termed as weak entity set. An entity set that has a primary key is termed as strong entity set

(1) (2)

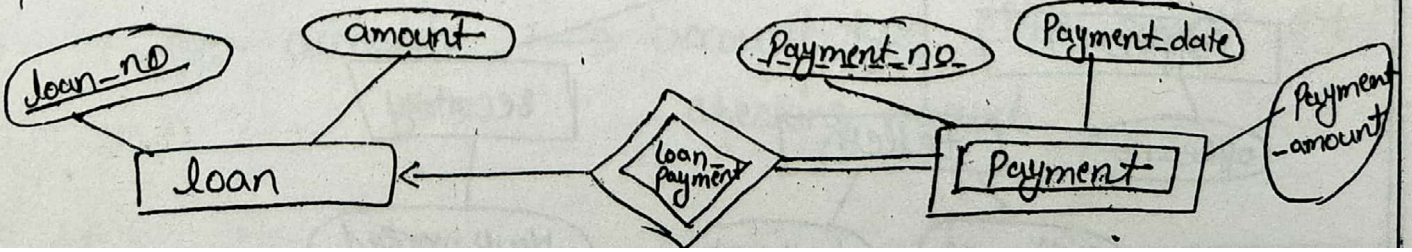
Eg:- payment = { payment_no , payment_date , payment_amount }

(3) → For a weak entity set to be meaningful, it must be associated with another entity set called the identifying or owner entity set

→ Weak entity set — existence dependent → Identifying entity set
Identifying relationship

~~⇒ Heat~~

→ The Discriminator of a weak entity set is a set of attributes that allows this distinction to be made.



Extended E-R Features

Bottom to Top

(1) **Specialization** :- The process of designating sub-grouping within an entity set is called specialization.

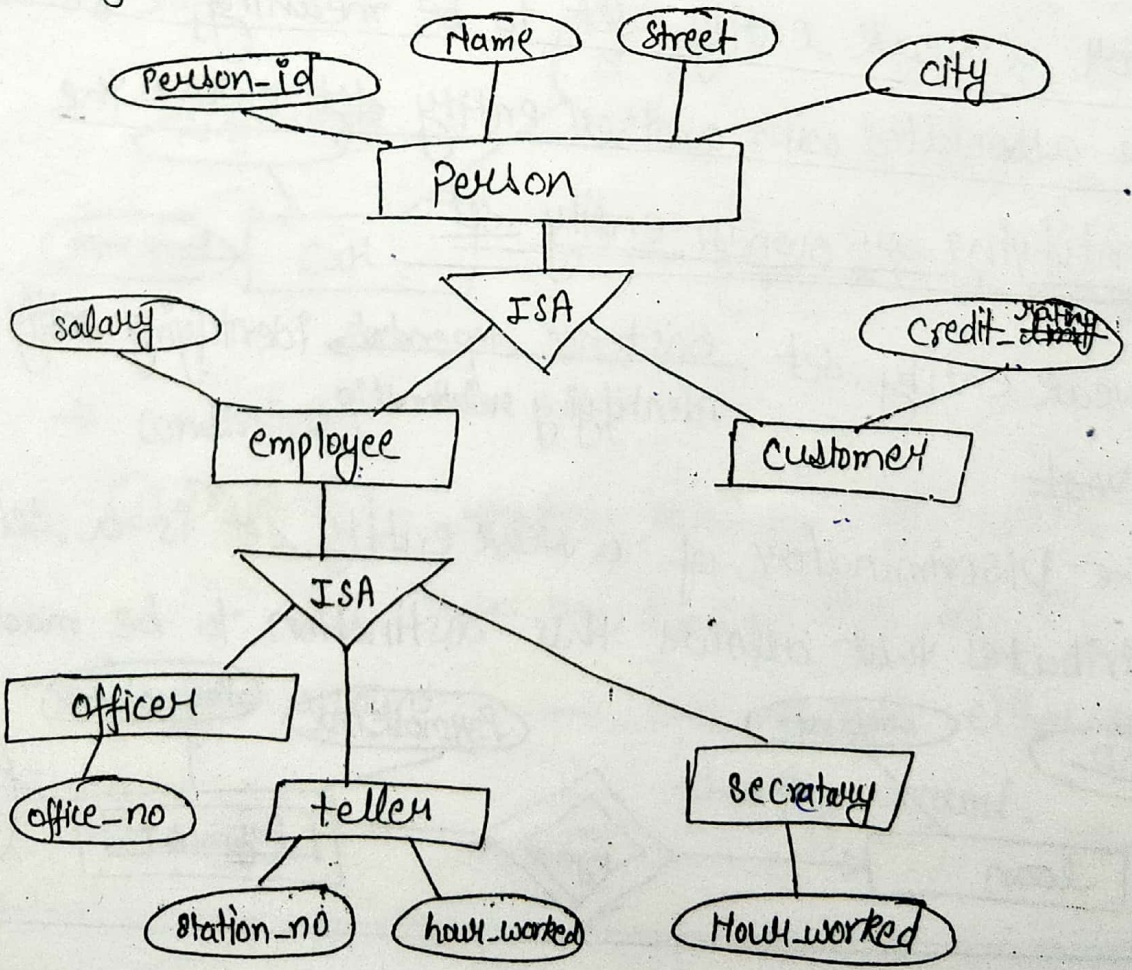
Eg:- an entity set person {person_id, name, street, city}

classified as

(1) Customer → {credit_rating} along with person

(2) employee → {salary} along with person.

→ Always have a "is a" relation denoted as "ISA"

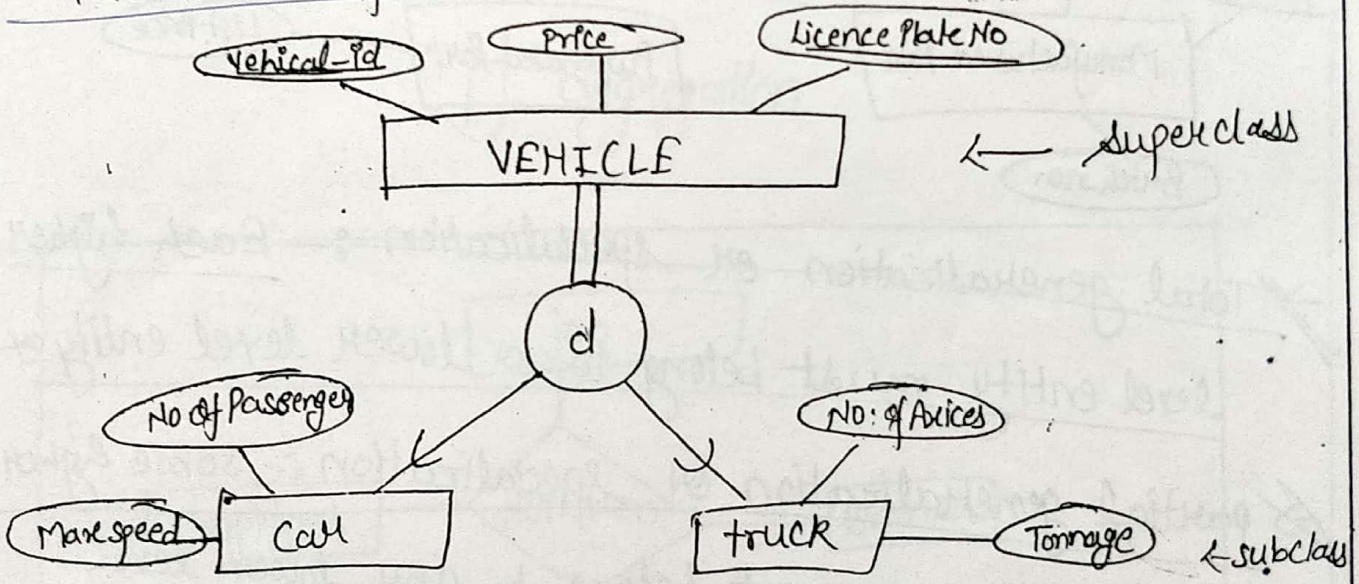


② Generalization

Top to Bottom

→ The refinement from an initial entity set into successive levels of entity subgrouping represents a top-down design process in which distinctions are made explicit.

→ The design process may also proceed in a bottom-up manner, in which multiple entity sets are synthesized into a higher-level entity set on the basis of common features.



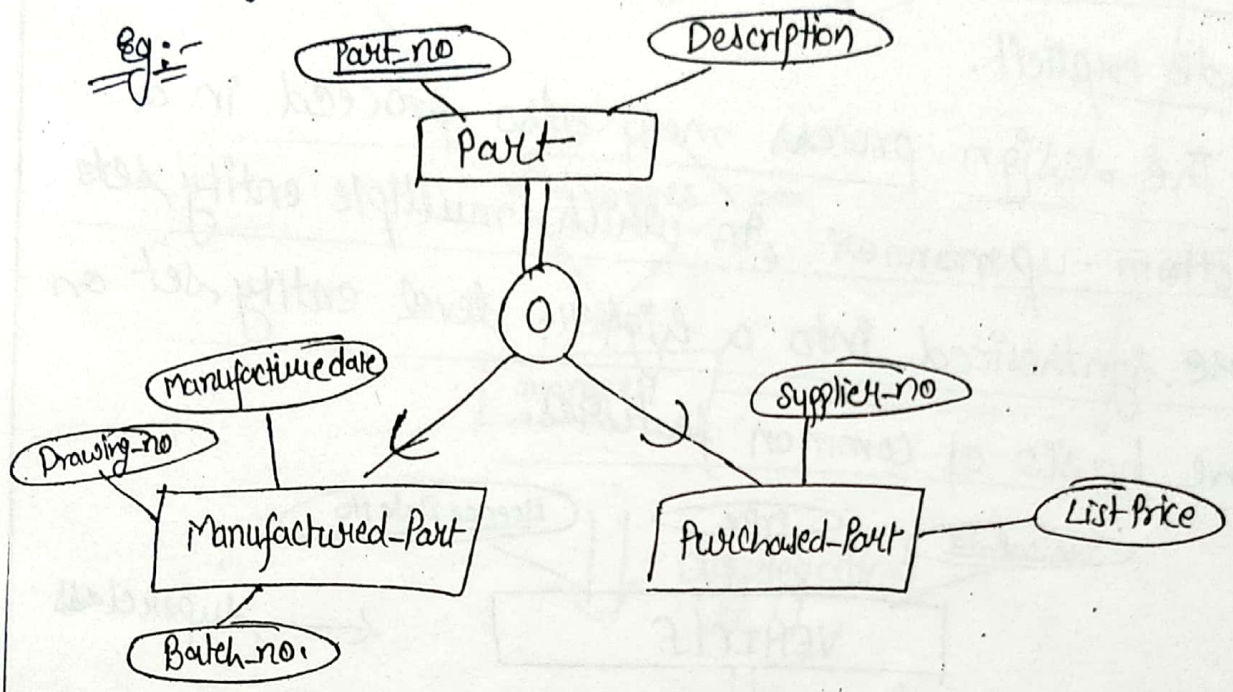
→ constraints.

① Disjoint :- requires that an entity belong to no more than one lower entity set.

Eg:- account → account-type either savings or checking account.

(2) overlapping :- the same entity may belong to more than one lower-level entity set within a single generalization.

Eg:-

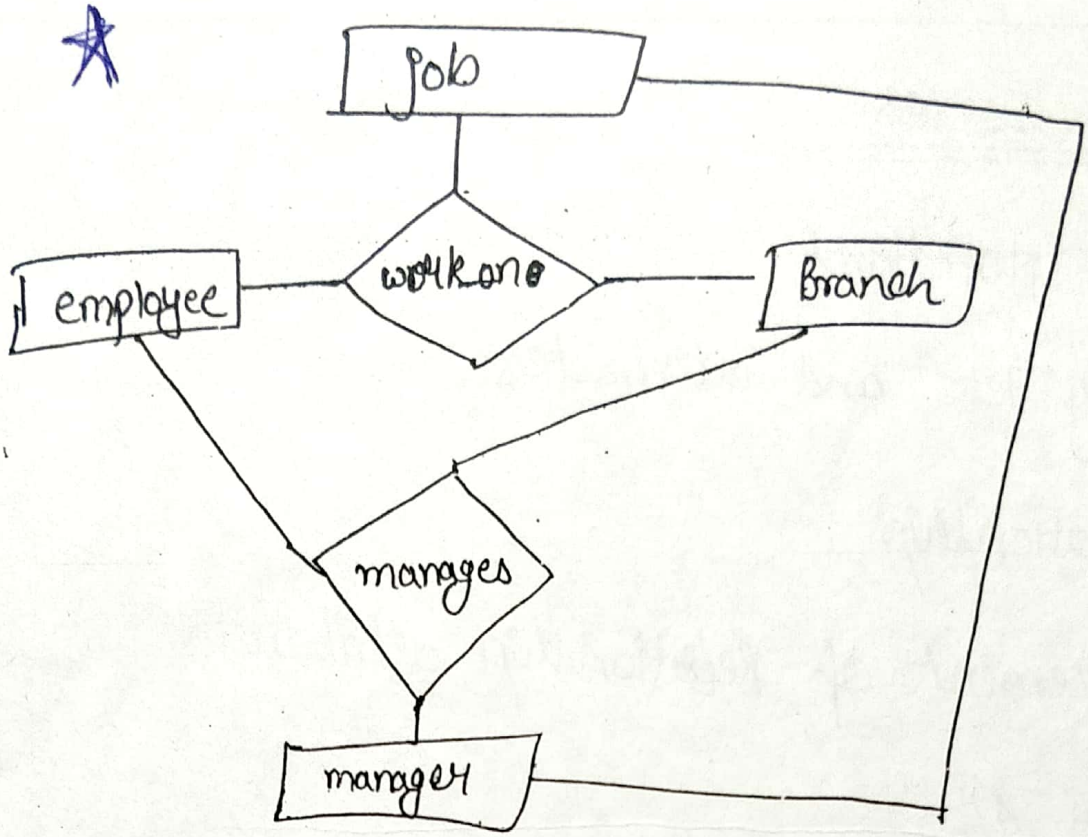


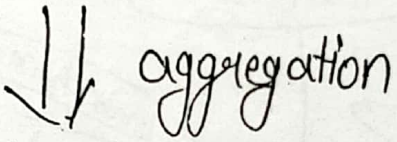
→ Total generalization or specialization :- Each higher level entity must belong to a lower level entity set

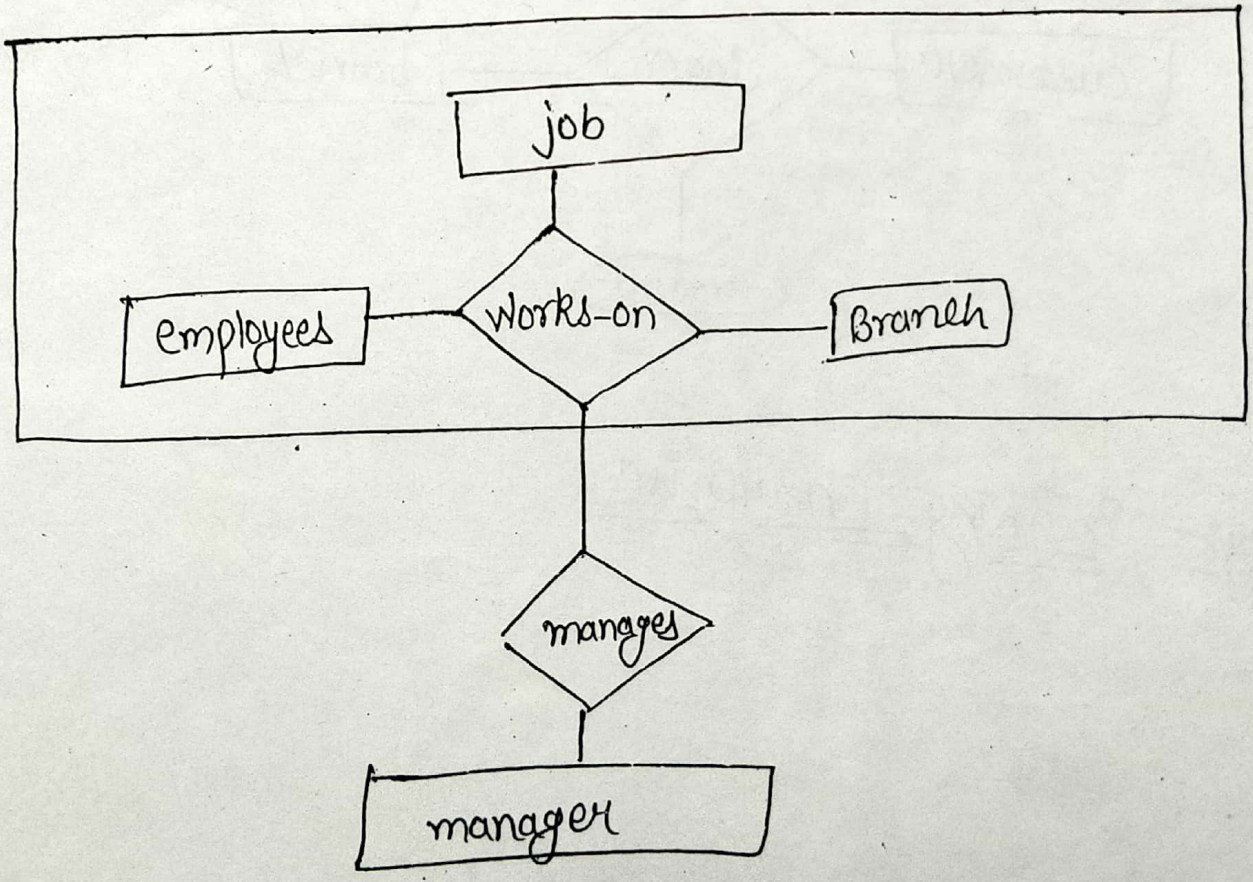
→ partial generalization or Specialization :- some higher level entities may not belong to any lower level entity set.

(3) Aggregation :- Aggregation is an abstraction through with relationship are treated as higher level entities.

It represents Relationship among Relationship set.



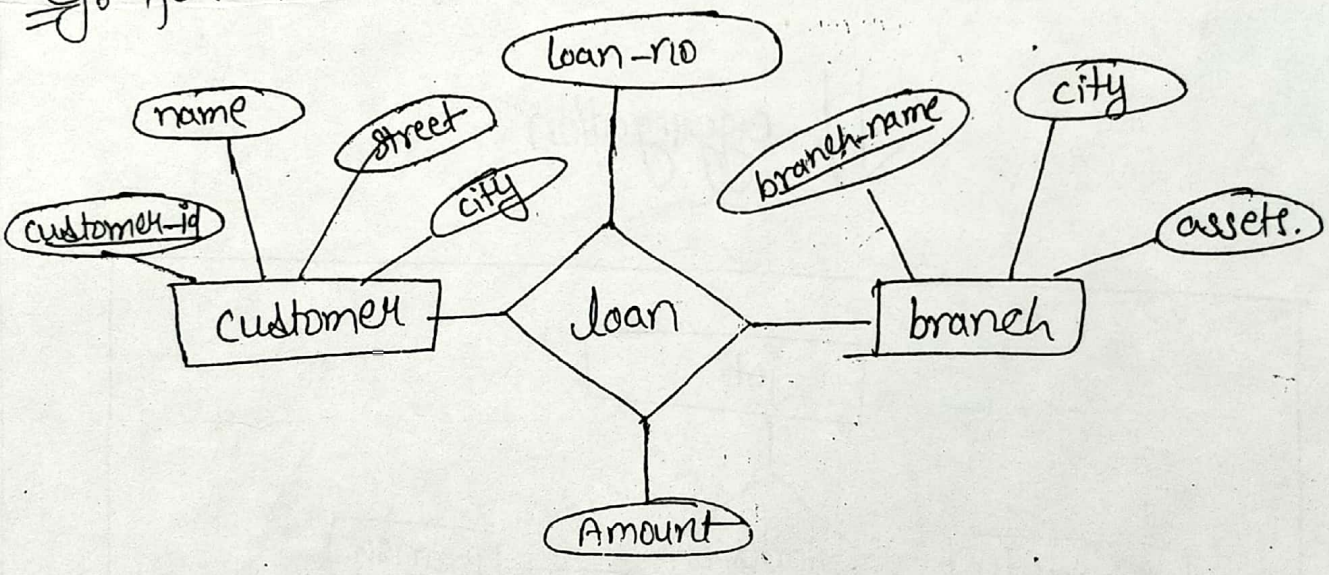




Design Issues

1. E-R Notations
2. Entity set and attributes
3. Relationship
4. Placement of Relationship attributes.

eg:- for 4th.



eg:- Banking Enterprise.