Unit 1 - Database System Concepts and Conceptual Modelling

* **Data** : Known facts that can be recorded and that have implicit meaning

* **Database** : a collection of related data (OR)
a very large, integrated collection of related data.

* **Implicit Properties of a Database** :

→ represents some aspects of the real world called a mini world and changes in the mini world are reflected in the db.

→ logically coherent collection of data with some inherent meaning but not a random collection.

→ It is designed, built and populated with data for a specific purpose.

→ Has an intended group of users and some preconceived applications in which users are interested.

* **Examples of Databases**

→ a directory of names and addresses
→ computerized catalog of a large library
→ Facebook
→ Amazon

* Database Management System

→ a collection of programs that enables users to create and maintain the database.

→ A general purpose software system the processes of defining, constructing, manipulating and sharing databases for various applications

A. Defining : Involves specifying the data types, structures and constraints for the data to be stored in the database.

The database definition is also stored by the DBMS in the form of a database catalogue or directory - called meta data.

B. Constructing a Database : Process of storing the data itself on a storage medium that is controlled by the DBMS.

C. Manipulating a Database : Includes functions such as querying for retrieval, updation, generating reports.

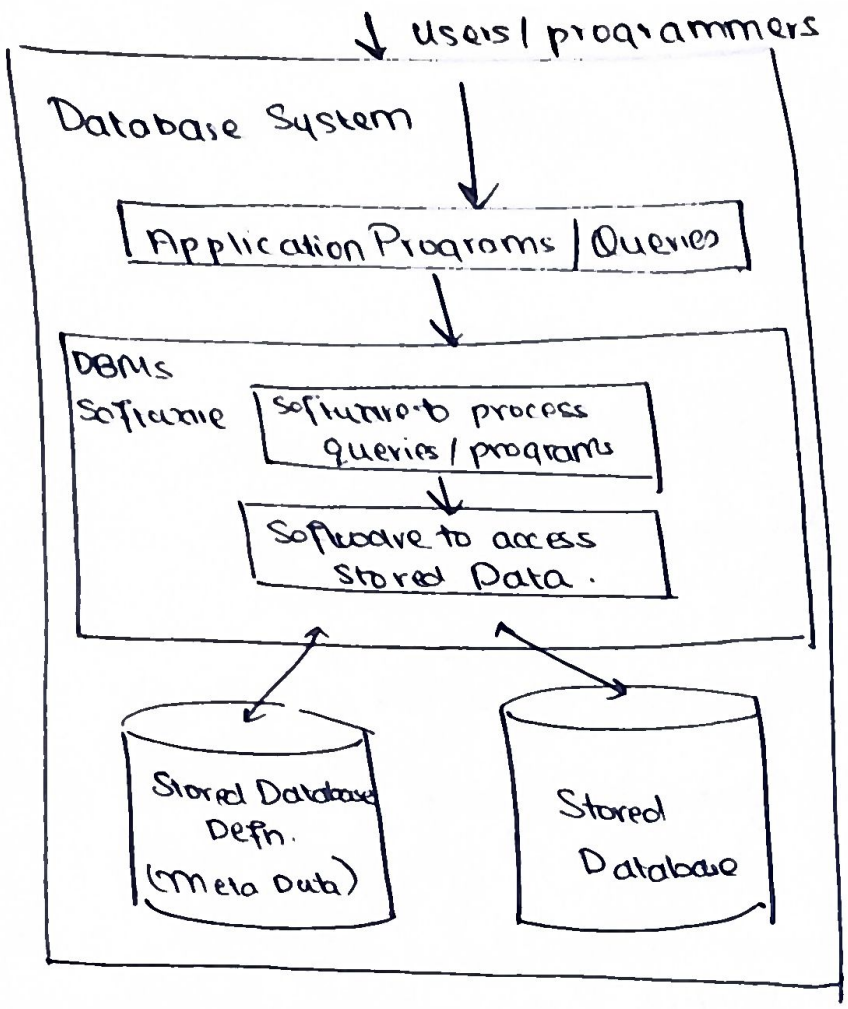
D. Sharing : Allows multiple users and programs to access the database simultaneously

* Database Environment

Meta Data - database definition or descriptive information is stored in the form of a database catalog or dictionary.

DBMS Software - includes software to proces queries / programs software to access stored data.

Database System - Has application programs to access the database by sending queries or requesting info



database + DBMS software

= database system.

* Other DBMS Functionalities - Protection and Maintenance

Protection - includes system protection against hardware or software malfunction, and security protection against unauthorized or malicious access.

Maintain - A typical large database may have a long life cycle, the DBMS must be able to maintain the database system by allowing the system to evolve as requirements change over time.

* Steps in Database Construction

(i) Specification and Analysis Phase

(ii) Conceptual Design - Database Design using data model

(iii) Logical Design - Designing the relationship using DBMS software

(iv) Physical Design - How the logical structure is physically stored in the target dbms.

@characteristics of a database approach - databases vs. files

→ Self-describing nature of a database system

→ Insulation between programs and data and data abstraction

→ Support of multiple views of the data

→ Sharing of data and multi-user transaction processing

A. Self Describing Nature of a Database System

→ DB System has not only the database, but definition of the db structure as well.

→ Structure of each file, type & storage format stored in meta data

→ a dbms software is not written for one specific database applications - should work on any no. of applications

→ Traditional file processing systems could only w/ one specific database -

B. Insulation between programs and data abstraction

Allows for program-data independence

= change structure of catalog file w/o changing the access programs

Program Operation Independence : app. programs can operate on data by invoking the operations, regardless of how the operations are implemented.

Data abstraction => which allows for program-data independence & program operation independence

C. Supports multiple views

→ each user may have a different perspective or view of the database

→ Views = virtual data derived from one or more relations & not explicitly stored.

→ a multiuser DBMS, has a variety of applications, it must provide facilities for defining multiple views.

D. Sharing of data and multi-use transaction processing

→ allows multiple users to access database at the same time

→ concurrency control - each transaction is correctly executed or aborted

→ Transaction - process like reading or updating database records

→ each transaction should not interfere w/ other transactions

→ transactions should enforce ACID properties
↓
atomicity, consistency, isolation, durability

* Actors on the scene

A. Database Administrators

→ oversee and manage primary & secondary resource
→ authorize access
→ coordinate & monitor
→ acquire hardware & software resources
→ accountable for security breaches & poor system response time

B. Database Designers

→ communicates with potential users to understand requires

→ create design to understand requirements

→ identify and choose appropriate structures

→ develop ~~user~~ views

Actors on the scene: end users

(1) Casual End Users

→ occasionally access db, need diff. info each time

→ use a sophisticated database query interface

(2) Naive or Parametric End users

→ constantly update & query from the database

→ uses standard types of queries and updates called canned transactions

→ Learn very little about DBMs facilities

(3) Sophisticated End Users

→ include engineers, scientists, business analysts

→ are familiar w/ the facilities of the DBMS

→ implement their own applications to meet their complex requirements

(4) Standalone Users

→ maintain personal databases

→ proficient in using specific software packages

→ eg. a financial software package that stores a variety of personal financial data

* System Analysts and Application Programmers

A. System Analysts

→ develop requirements of end users

→ develop specifications for standard canned transactions

B. Application Programmers

→ implement specifications as programs

→ Test, debug, document canned transactions

* Workers behind the scene

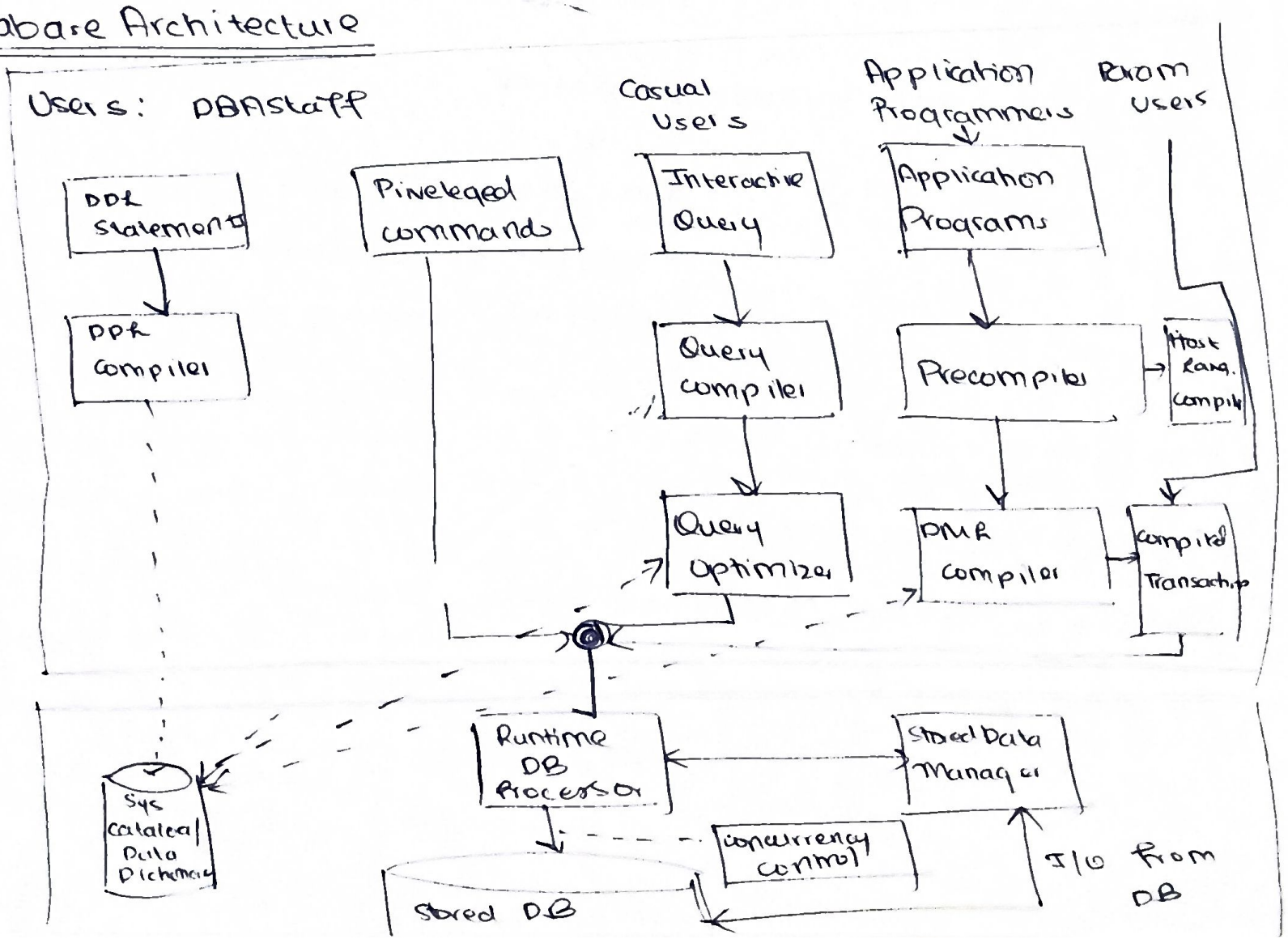
System designers & implementers

Tool developers design & implement tools

Operators & maintenance personnel

(run & maintain software & hardware environment)

* Database Architecture



DB Architecture has 2 parts :

(i) top part = diff. users & interfaces

(ii) lower part = internal modules of DBMS responsible for data storage & processing transactions.

* Database Users – Database Administrator

→coordinates all activities of DB

responsible for:

(i) schema definition

(ii) storage structure & access method defn.

(iii) Granting user authority

(iv) Periodical backup to prevent loss of data

(v) monitoring performance & responding to changes in requirements.

DDL Compiler

→processes schema defns specified in the DDL

→ stores metadata in system catalog

↓

name & size of files

datatypes of data items

mapping info among schemas & constraints

* Database Users - Casual Users

→ persons w/ occasional need for info

→interact using interactive query interface

→do data entry for predefined transactions (canned)

Query Compiler

→ compiles queries into an internal form after they are parsed & validated for correctness.

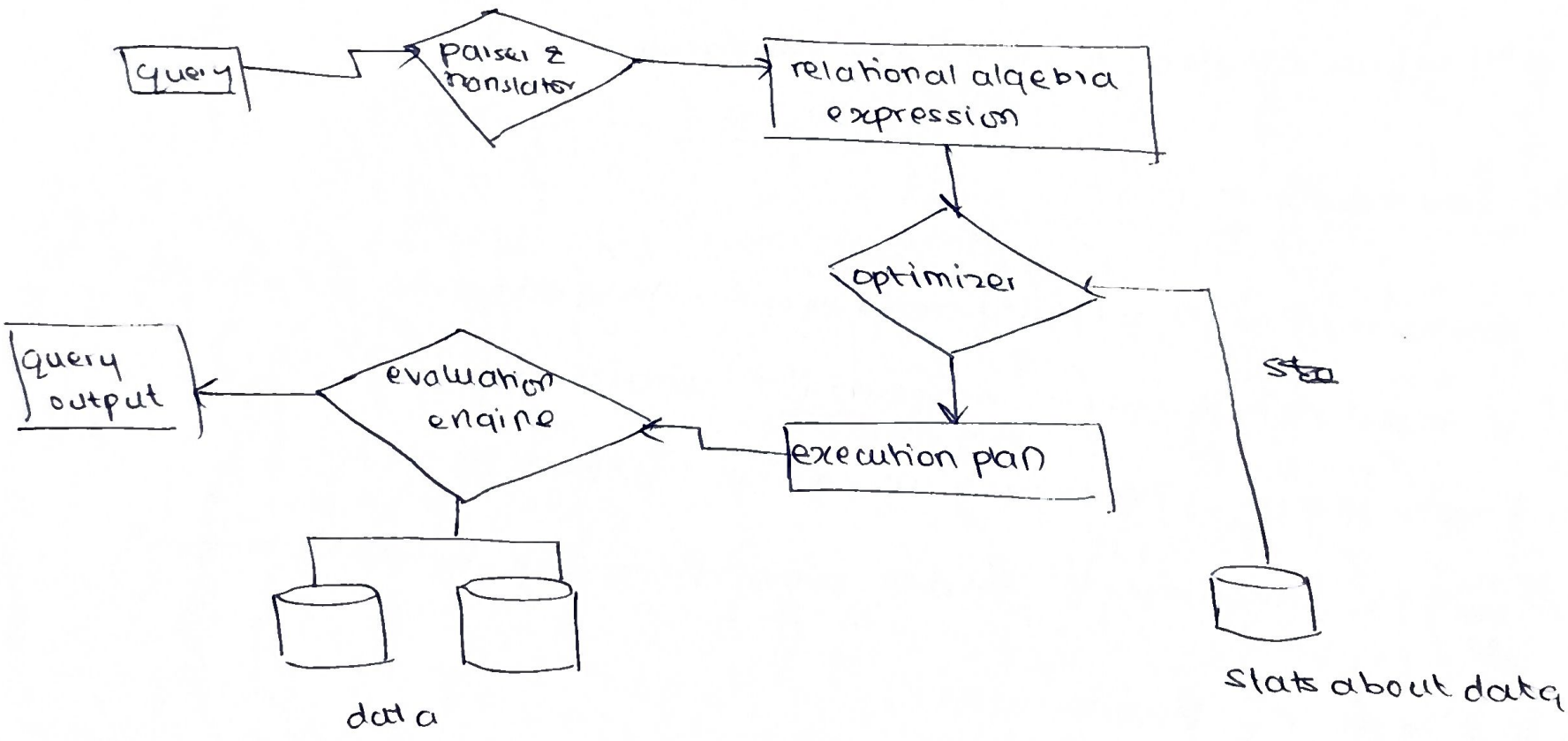
Query Optimizer

→ Rearrangement & reordering queries

→ eliminate redundancies

→ consults system catalog for statistical & physical info

→ Makes call or runtime process



Database Users: Application Programmers

→ Interact with system through application programs

→ Application programs written in Java, C etc.

→ Progs sent to pre compiler

→ pre compiled extracts DML

→ DML commands sent to DML compiler

Lower level of the Architecture

System Catalog → contains metadata

→ The DB & catalog are stored on the disk

→ Access to disk controlled by operating system.

Runtime DB Processor

executes :

privileged commands

executable query plans

canned transactions w/ runtime parameters

works w/ sys catalog & stored db manager

Storage manager

has:

authorization & integrity manager — tests integrity constraints

transaction manager - manages concurrency

File manager - manages allocation of space on disk

buffer manager - fetches data from disk to main memory

* Two Types of Architecture

Two Tier Architecture

→ Client - Server Architecture

→ User interface programs & applications run on client side

→ establish connection using ODBC - API

→ client sends request to server

→ server processes requests related to query & transaction processing.

Three Tier Architecture

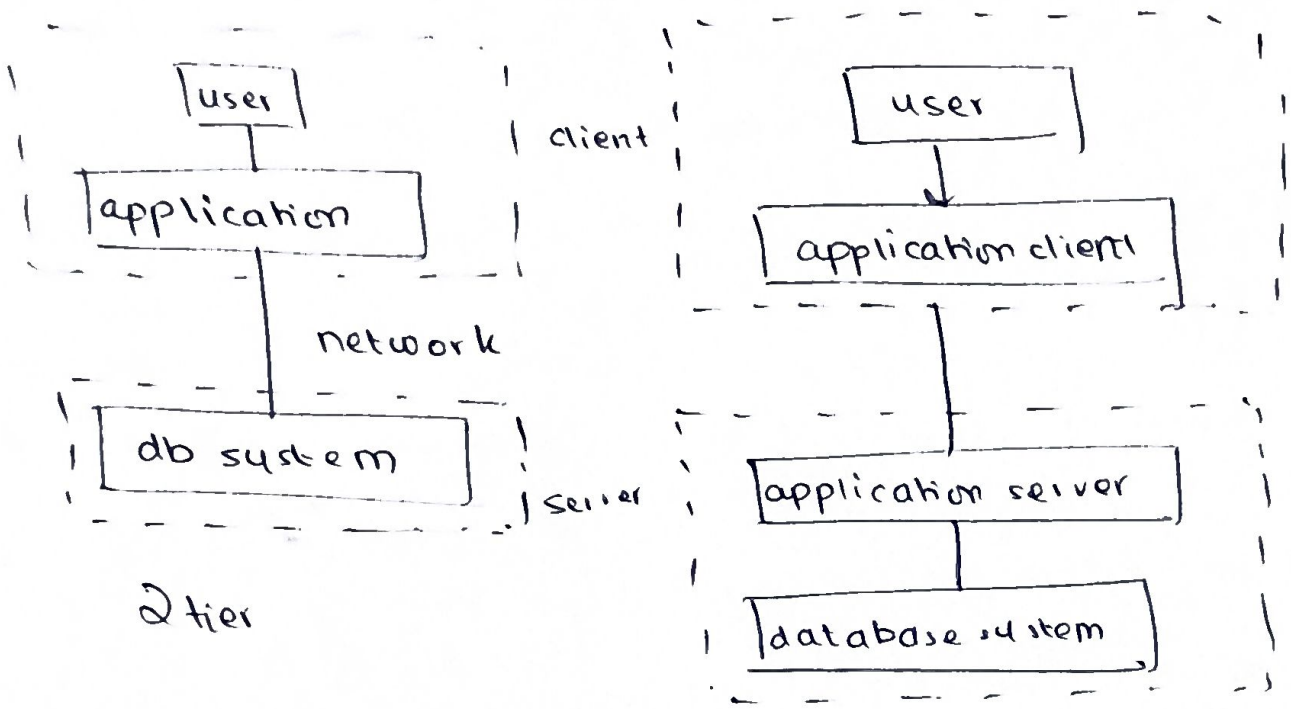
→ has a middle tier -

has the role of:

→ running application programs

→ checking client credentials

→ acts as an intermediary between client & server



2 tier

3-tier architecture

Entity Relation Diagram

* Entity Relational Model : a high level conceptual data model used for conceptual design of database applications

has 3 basic notions :

→ entity sets
→ relationship sets
→ attributes

* entity - an object in the miniworld that are represented in the database

* attributes - properties used to describe the entity

Attribute types

(i) simple / atomic — cannot be divided further

(ii) composite - attributed can be divided into smaller subparts eg. address

(iii) single valued - can have only a single value eg. age

(iv) multi-valued - can have multiple values (say, an upper or lower bound)

(v) Stored & derived attributes

↳ values derived from other attributes

eg. age is determined from current date & value of a person's DOB.

* Entity Type and Entity Set

Entity Type - collection of entities that share common properties
- described by its name & attributes
- denoted by a rectangular box

Entity Set : collection of all entities of a particular entity type

* Key - an attribute that uniquely identifies individual instances

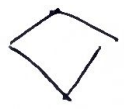
→ underlined in ER diagram

→ weak entity => no key attributes

* ER Diagram Notation


→ entity type


→ weak entity type

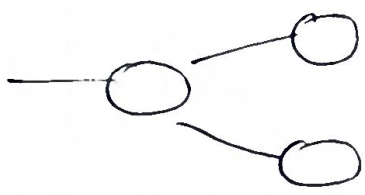

→ relationship type


→ identifying relation type


attribute


key attribute


multivalued attr.


composite attr.


derived attr.

* Relationship Degree

→ no of participating entity types

* Role Name: role of participating entity

* Recursive relationship: same entity type participates in relationship more than once in different roles.

* Relationship Constraints

(i) cardinality ratio

(ii) participation

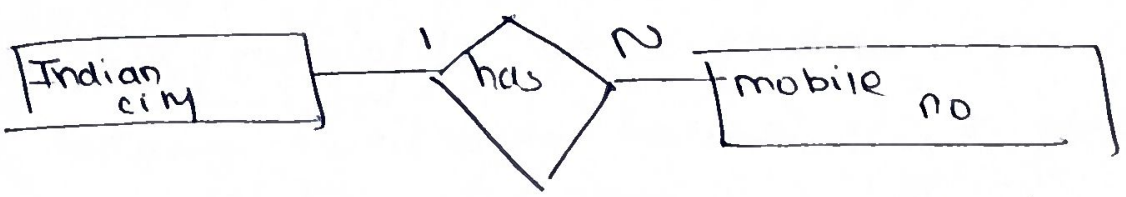
Cardinality Ratio - specifies the no. of relationship instances an entity can participate in. Can be:

1 : 1 → Indian citizen — has — aadhar number (at most one, even no relation = 1:1)

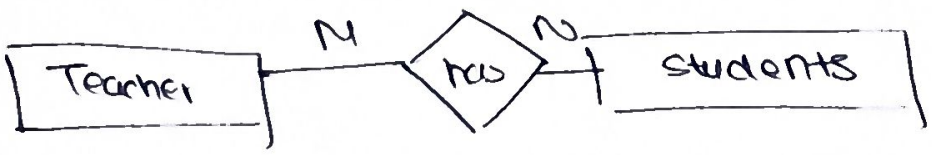
1 : N

N : 1 → (reversed)

M : N



(means that every entity can be related to at max N other entities



Participation Constraint

→ whether all or only some entity occurences participate in a relationship

can be

(i) Total Participation – every entity needs to be related
eg. every student belongs to a dept.

(ii) Partial Participation – some entities are involved

eg. only some employees manage a dept.

Total participation = ═══

Partial participation = ———

ER Diagram Questions

Q1:

Company Database

→ The company is organized into departments. Each dept. has a name, no, location, an employee who manages the dept. Locations is the only multivalued attribute. Name & Number are separate key attributes, because each was specified to be unique.

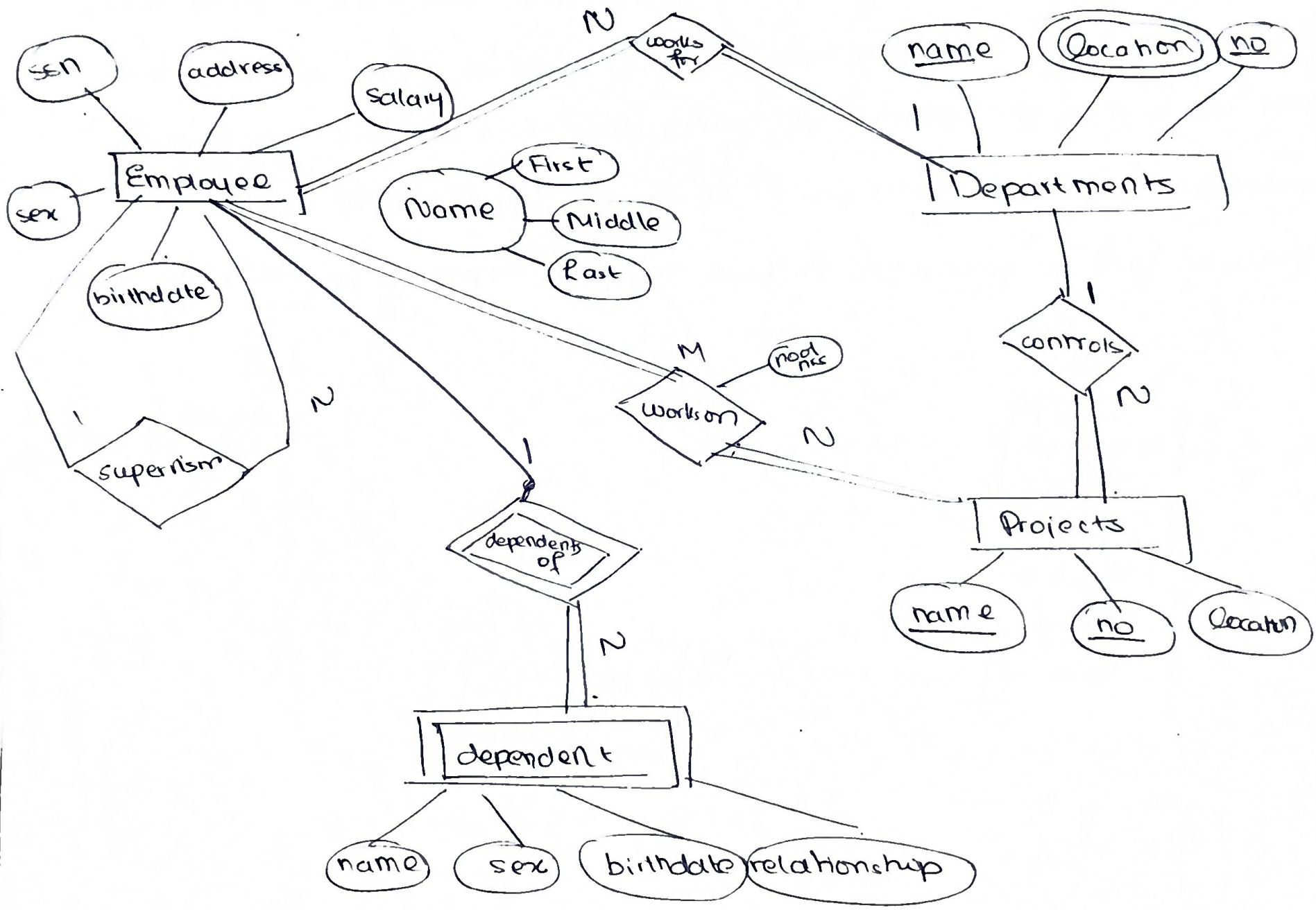
→ Each dept. controls a no. of projects. Each project has a name, number & is located at a single location & controlling dept. Both name & no. are separate key attributes because each was specified to be unique.

→ Store each employee's social security no, address, salary, sex and birthdate. Name may be a composite attribute. Each employee works for 1 dept. but may work on several projects. We keep track of the no. of hrs per week that an

employee works on a project. We also keep track of the direct supervisor of each employee.

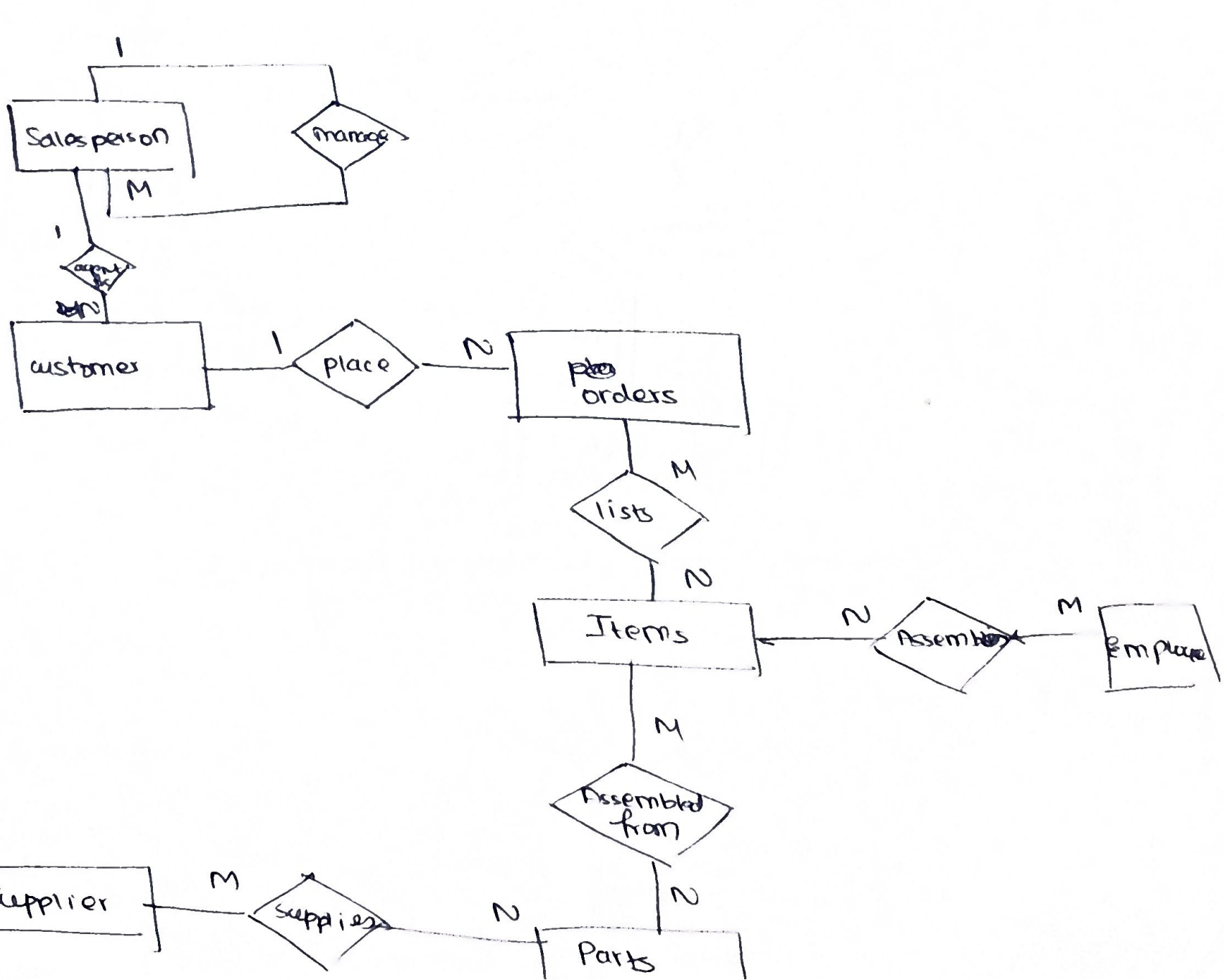
→ Each employee may have a no. of dependents. For each dependent, we keep track of their name, sex, birthdate & relationship to employee.

Solution



Q2 A company has the following scenario

There are a set of salespersons. Some of them manage other salespersons. However, a salesperson cannot have more than 1 manager. A salesperson can be an agent for many customers. A customer is managed by exactly one salesperson. A customer can place any no of orders. An order can be placed by exactly one customer. Each order lists one or more item. An item may be listed in many orders. An item can be assembled from different parts and parts can be common for many items. One or more employees assemble an item from parts. A supplier can supply diff. parts in certain quantities. A part may be supplied by diff. suppliers.



Q3 A university registrar's office maintains data about (7)
the following entities:

(a) courses: including no, title, credits, syllabus & prerequisites

(b) course offerings: including course no, year, sem, section no, instructor(s), timing & classroom (consider as weak entity)

(c) students: w/ student-id, name & program

(d) instructors: w/ id no, name, dept. title.

Further, the enrollment of students in courses & grades awarded in each course must be appropriately modeled.