CST 204 : Database Management Systems

CST 204	Database Management Systems	CATEGORY	L	T	P	CREDIT	YEAR OF INTRODUCTION
		PCC	3	1	0	4	2019

Module 1: Introduction & Entity Relationship (ER) Model

Concept & Overview of Database Management Systems (DBMS) - Characteristics of Database system, Database Users, structured, semi-structured and unstructured data. Data Models and Schema - Three Schema architecture. Database Languages, Database architectures and classification.

ER model - Basic concepts, entity set & attributes, notations, Relationships and constraints, cardinality, participation, notations, weak entities, relationships of degree 3.

Module 2: Relational Model

Structure of Relational Databases - Integrity Constraints, Synthesizing ER diagram to relational schema

Introduction to Relational Algebra - select, project, cartesian product operations, join - Equi-join, natural join. query examples, introduction to Structured Query Language (SQL), Data Definition Language (DDL), Table definitions and operations – CREATE, DROP, ALTER, INSERT, DELETE, UPDATE.

Module 3: SQL DML (Data Manipulation Language), Physical Data Organization

SQL DML (Data Manipulation Language) - SQL queries on single and multiple tables, Nested queries (correlated and non-correlated), Aggregation and grouping, Views, assertions, Triggers, SQL data types.

Physical Data Organization - Review of terms: physical and logical records, blocking factor, pinned and unpinned organization. Heap files, Indexing, Singe level indices, numerical examples, Multi-level-indices, numerical examples, B-Trees & B+-Trees (structure only, algorithms not required), Extendible Hashing, Indexing on multiple keys – grid files.

Module 4: Normalization

Different anomalies in designing a database, The idea of normalization, Functional dependency, Armstrong's Axioms (proofs not required), Closures and their computation, Equivalence of Functional Dependencies (FD), Minimal Cover (proofs not required). First Normal Form (1NF), Second Normal Form (2NF), Third Normal Form (3NF), Boyce Codd Normal Form (BCNF), Lossless join and dependency preserving decomposition, Algorithms for checking Lossless Join (LJ) and Dependency Preserving (DP) properties.

Module 5: Transactions, Concurrency and Recovery, Recent Topics

Transaction Processing Concepts - overview of concurrency control, Transaction Model, Significance of concurrency Control & Recovery, Transaction States, System Log, Desirable Properties of transactions.

Serial schedules, Concurrent and Serializable Schedules, Conflict equivalence and conflict serializability, Recoverable and cascade-less schedules, Locking, Two-phase locking and its variations. Log-based recovery, Deferred database modification, check-pointing.

Introduction to NoSQL Databases, Main characteristics of Key-value DB (examples from: Redis), Document DB (examples from: MongoDB)

Main characteristics of Column - Family DB (examples from: Cassandra) and Graph DB (examples from : ArangoDB)

Text Books

1. Elmasri R. and S. Navathe, Database Systems: Models, Languages, Design and Application Programming, Pearson Education, 2013.
2. Sliberschatz A., H. F. Korth and S. Sudarshan, Database System Concepts, 6/e, McGraw Hill, 2011.

University Question Pattern

	No of Qstns	**Each Carries**	**Total**
PART A	10 (Answer All)	3	30
PART B	10 (2 from each module, Need to answer any 1)	14	70

Database Management System
Module 1

Lect 1 : Introduction to Database

Introduction to database

- Data : known facts that can be recorded and that have implicit meaning.
  - Eg : Text- names, telephone nos, address,
  - Image, Video, audio etc
  - Flipkart, Amazon, Uber etc contains data

- **Database** : Collection of related data
  - Eg : Student database which contains

Roll No	Name	Dept	Marks

  - Eg 2 :

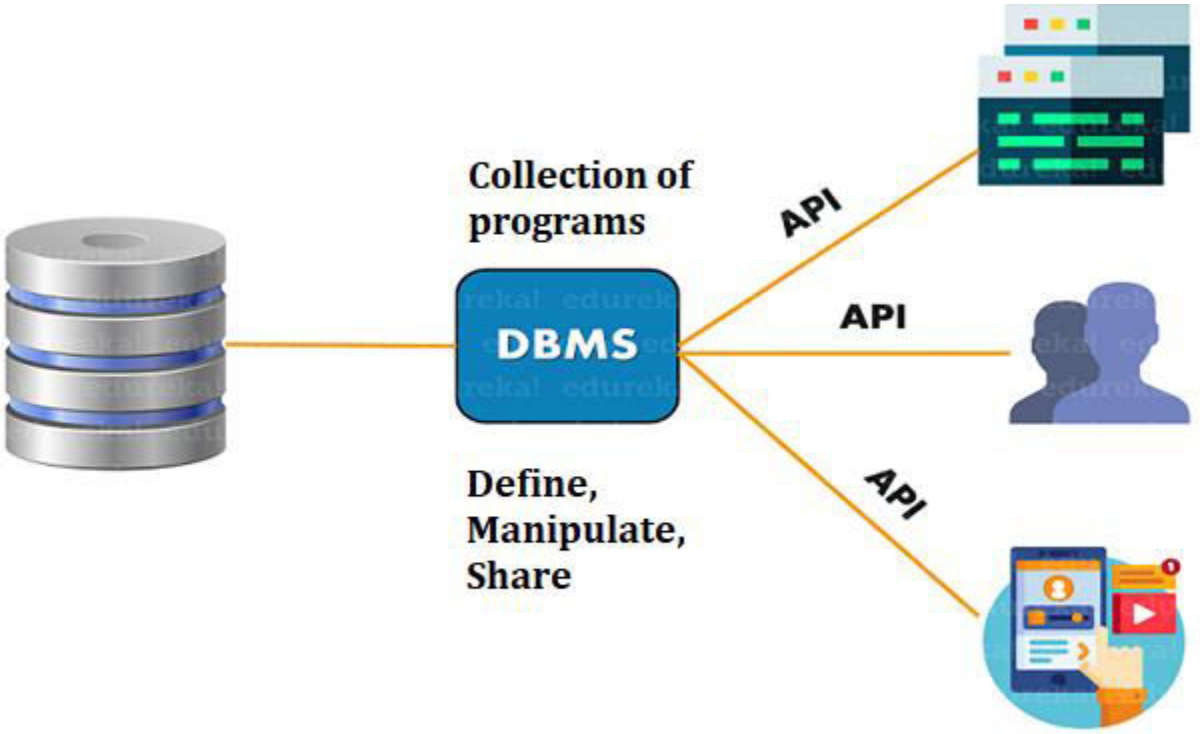
Roll No	Grocery Price	Car Model

These are data but not database since they are not related data

Type of Database

- **Traditional Database** : information is stored and accessed is either textual or numeric.
- **Multimedia databases** : store images, audio clips, and video streams digitally
- **Geographic information systems (GIS)** : store and analyze maps, weather data, and satellite images.

- **Database management system (DBMS)** is a collection of programs that enables users to create and maintain a database



- DBMS is a general-purpose software system that facilitates the processes of defining, constructing, manipulating, and sharing databases among various users and applications

  - **Defining a database** involves specifying the data types, structures, and constraints of the data to be stored in the database

Eg: Student database having foll attributes

Roll no – String
Name – String
Marks - Float

Roll no	Name	Marks

The database definition or descriptive information is also stored by the DBMS in the form of a database catalog or dictionary; it is called **meta-data**

- **Constructing** the database is the process of storing the data on some storage medium

eg

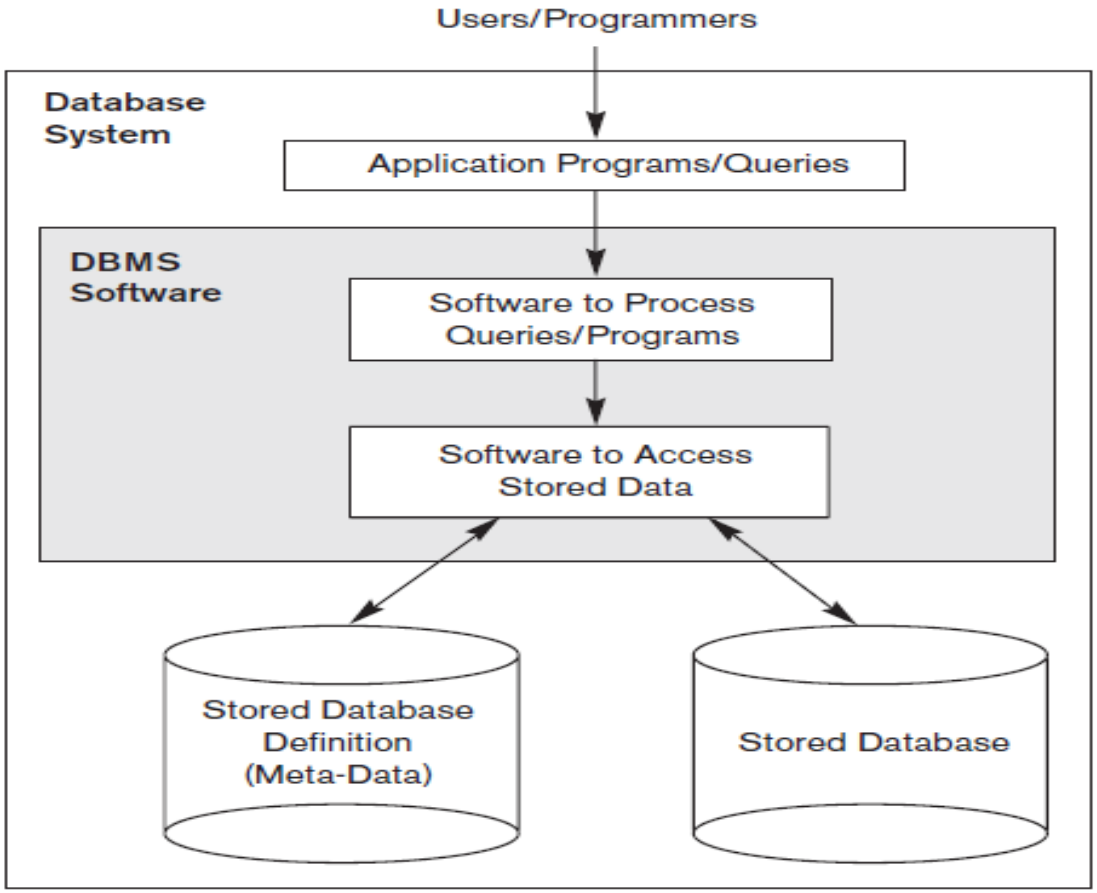
Roll no	Name	Marks
1	Akhil	34
2	Sajan	42
3	Vivek	21

- **Manipulating a database** includes functions such as querying the database to retrieve specific data, updating the database
- **Sharing a database allows multiple** users and programs to access the database simultaneously

- An **application program** accesses the database by sending queries or requests for data to the DBMS

- DBMS include protecting the database and maintaining it over a long period of time

Database System Environment



Characteristics of the Database Approach

Characteristics of the Database Approach

- 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 multiuser transaction processing

1. Self-describing nature of a database system

- database system contains not only the database itself but also a complete definition or description of the database structure and constraints.
- This definition is stored in the DBMS catalog
- information stored in the catalog is called **meta-data** and it describes the structure of the primary database

RELATIONS

Relation_name	No_of_columns
STUDENT	4
COURSE	4
SECTION	5
GRADE_REPORT	3
PREREQUISITE	2

COLUMNS

Column_name	Data_type	Belongs_to_relation
Name	Character (30)	STUDENT
Student_number	Character (4)	STUDENT
Class	Integer (1)	STUDENT
Major	Major_type	STUDENT
Course_name	Character (10)	COURSE
Course_number	XXXXNNNN	COURSE
….	….	…..
….	….	…..
….	….	…..
Prerequisite_number	XXXXNNNN	PREREQUISITE

2. Insulation between programs and data, and data abstraction

- The structure of database is stored in the DBMS catalog separately from the access programs.
- This property is called program-data independence

Eg: in Student db

Roll no	Name	Marks
1	Akhil	34
2	Sajan	42
3	Vivek	21

If we wish to add a new attribute DOB we can do without affecting the data

Roll no	Name	Marks	DOB
1	Akhil	34	
2	Sajan	42	
3	Vivek	21	

This is known an program-data independence

- Users can define operations on data
- Operations (also called a function or method) is specified in two parts
  - The interface (or signature) of an operation includes the operation name and the data types of its arguments
  - The implementation (or method) of the operation is specified separately and can be changed without affecting the interface
- User application programs can operate on the data by invoking these operations through their names and arguments, regardless of how the operations are implemented.
- This is termed **program-operation independence**
- The characteristic that allows program-data independence and program-operation independence is called **data abstraction**

3. Support of multiple views of the data

- A database typically has many users, each of whom may require a different perspective or **view of the database.**
- A view may be a subset of the database or it may contain virtual data that is derived from the database files but is not explicitly stored

Roll no	Name	Marks
1	Akhil	34
2	Sajan	42
3	Vivek	21

Student database

Name
Akhil
Sajan
Vivek

View 1: Name of the students

Roll no	Name	Marks
2	Sajan	42

View 2: Details of the students with highest mark

4. Sharing of Data and Multiuser Transaction Processing

- The DBMS must include **concurrency control software**
  - to ensure that several users trying to update the same data do so in a controlled manner so that the result of the updates is correct

- For example, when several reservation agents try to assign a seat on an airline flight, the DBMS should ensure that each seat can be accessed by only one agent at a time for assignment to a passenger

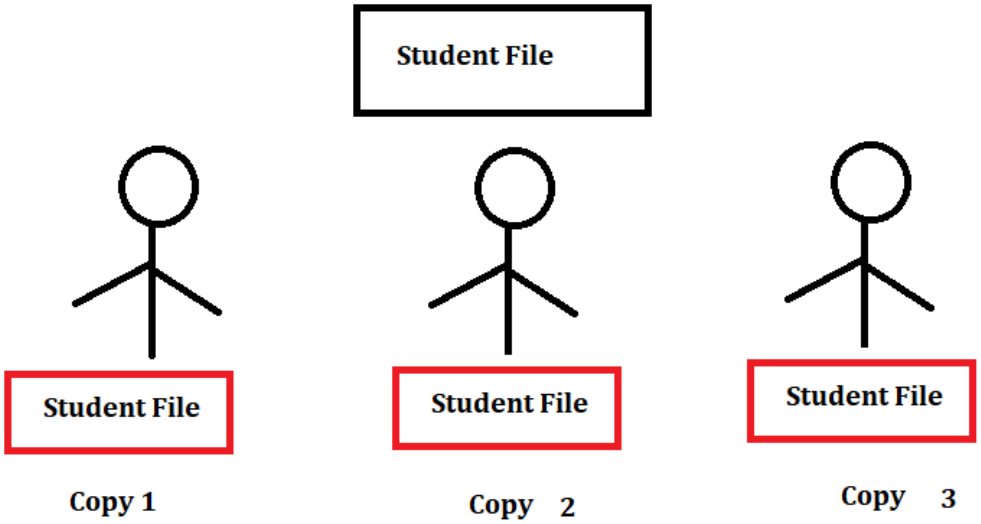
Module 1
Lect 3 – Advantages & Disadvantages of DBMS

Advantages of DBMS

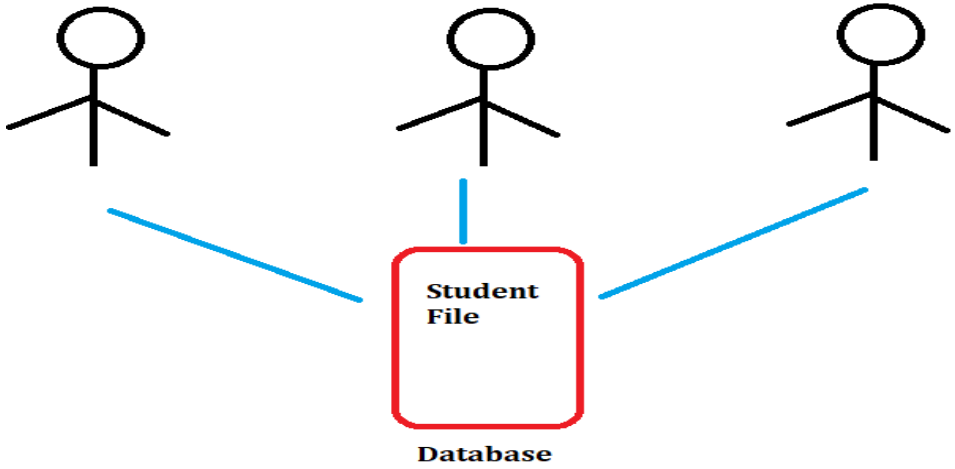
1.Controlling Redundancy

- In traditional file processing every user maintains the copy of same file
- But in DBMS same copy can be shared among many users, thus redundancy can be avoided

TRADITIONAL FILE PROCESSING



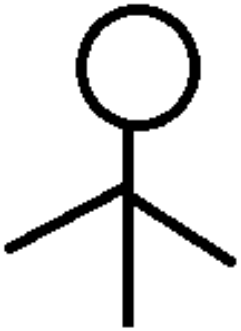
DATABASE MANAGEMENT SYSTEM



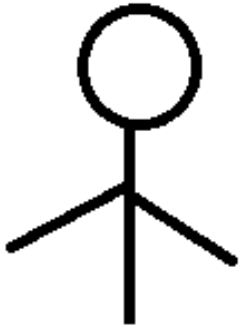
2. Restricting Unauthorized Access

- When multiple users share a large database, it is likely that most users will not be authorized to access all information in the database
- Eg: Financial data is confidential, and only authorized persons are allowed to access such data.

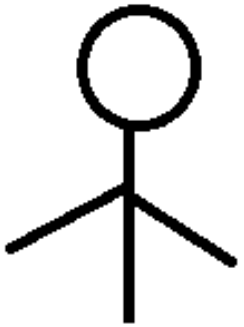
Student



Teacher



Parent

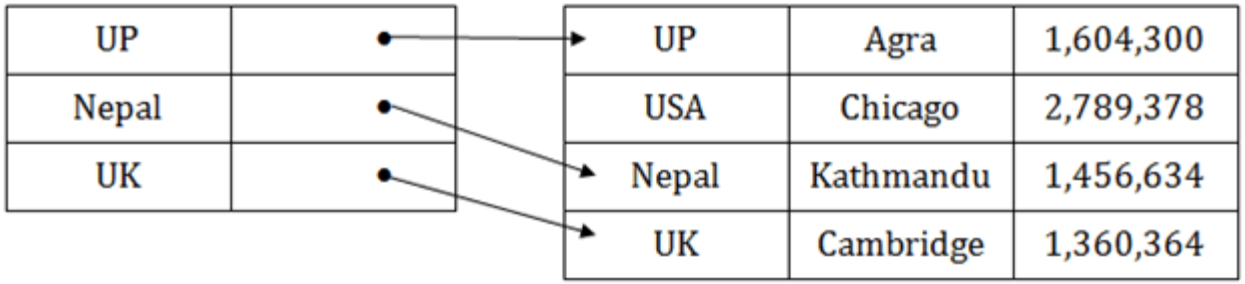


Roll No	Name	Dept	CGPA	Ph no	Fee details
1	John	CSE	7.7	787656445	55,000
2	Abin	ECE	6.8	765454343	45,000
3	Shehin	EEE	7.2	985645231	43,000
Student DB					

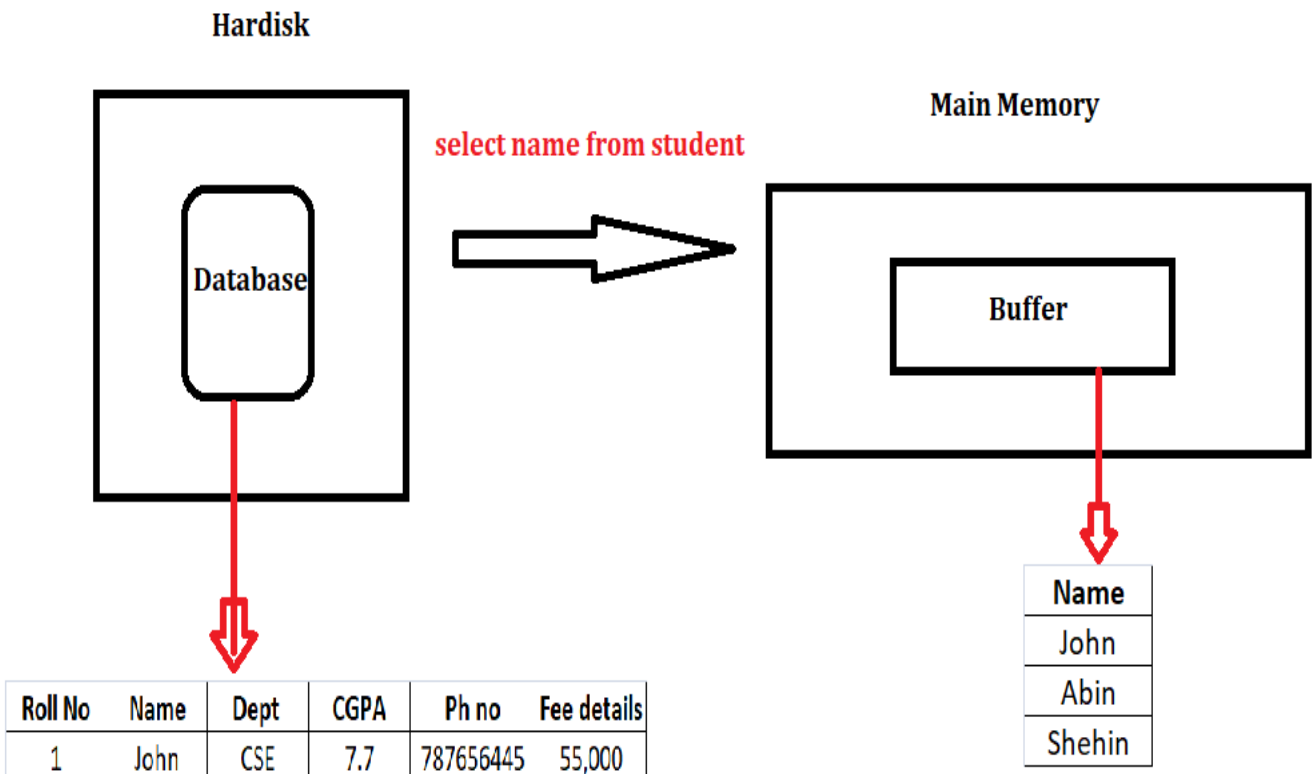
3. Providing Storage Structures for Efficient Query Processing

- DBMS provide specialized data structures and search techniques to speed up disk search for the desired records.
- indexes are used for this purpose
- In order to process the database records needed by a particular query, those records must be copied from disk to main memory. DBMS often has a **buffering or caching module** that maintains parts of the database

Indexing technique



Buffered storage



Roll No	Name	Dept	CGPA	Ph no	Fee details
1	John	CSE	7.7	787656445	55,000
2	Abin	ECE	6.8	765454343	45,000
3	Shehin	EEE	7.2	985645231	43,000
Student DB					

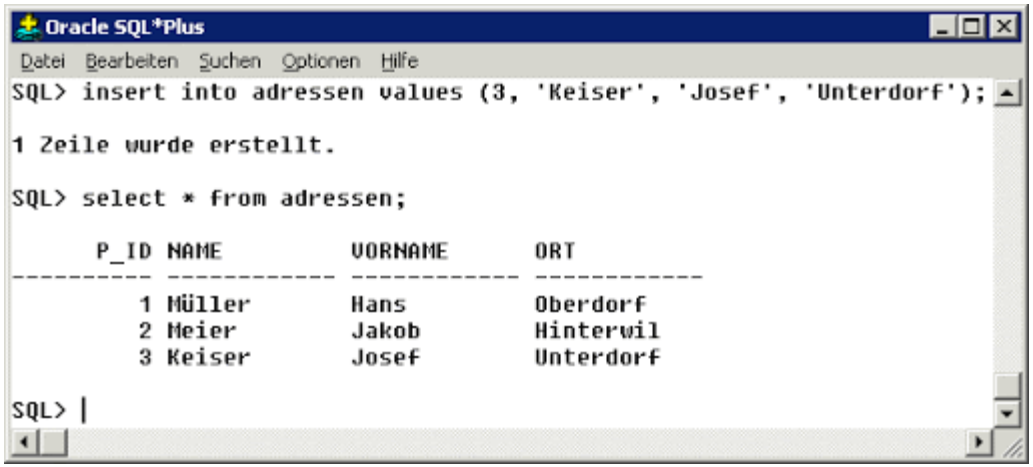
4. Providing Backup and Recovery

- DBMS provide facilities for recovering from hardware or software failures.
- **backup and recovery subsystem** of the DBMS is responsible for recovery

- For example, if the computer system fails in the middle of a complex update transaction, the recovery subsystem is responsible for making sure that the database is restored to the state it was in before the transaction started executing

5. Providing Multiple User Interfaces

- DBMS provide a variety of user interfaces. These include
  - query languages for casual users,
  - programming language interfaces for programmers,
  - menu-driven interfaces for standalone users known as **graphical user interfaces (GUIs).**
  - **Form based interfaces**



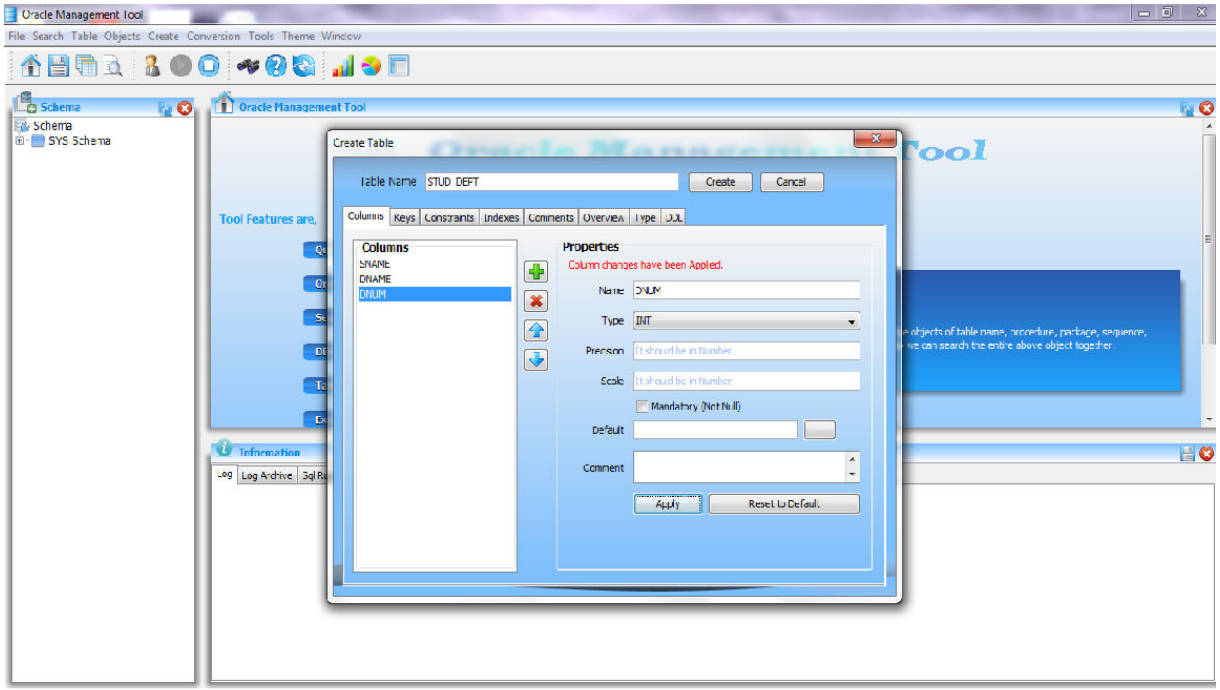


Fig.2 Table Creation

User Complaint Form

User Name	:-	
Department	:-	CSE
Floor	:-	1
Block	:-	A-Block
Room No.	:-	--
Problem Reported on	:-	
Telephone No	:-	--
Complaint	:-	

Submit Reset

6. Enforcing Integrity Constraints

- Most database applications have certain **integrity constraints** that must hold for the data
- specifying a data type for each data item is an integrity constraint
- ***referential integrity constraint :*** A more complex type of constraint that frequently occurs involves specifying that a record in one file must be related to records in other files.
- ***key or* uniqueness constraint :**
- This constraint specifies uniqueness on data item values,

RELATIONS

Relation_name	No_of_columns
STUDENT	4
COURSE	4
SECTION	5
GRADE_REPORT	3
PREREQUISITE	2

COLUMNS

Column_name	Data_type	Belongs_to_relation
Name	Character (30)	STUDENT
Student_number	Character (4)	STUDENT
Class	Integer (1)	STUDENT
Major	Major_type	STUDENT
Course_name	Character (10)	COURSE
Course_number	XXXXNNNN	COURSE
….	….	…..
….	….	…..
….	….	…..
Prerequisite_number	XXXXNNNN	PREREQUISITE

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Disadvantages of DBMS

- Cost of Hardware & Software
- Cost of Data Conversion
- Cost of Staff Training
- Appointing Technical Staff
- Database Damage

Database Management System

Module 1

Lect 4 : Actors on the Scene

Actors on the Scene

- Database Administrators
- Database Designers
- End Users
  - Casual end users
  - Naive or parametric end users
  - Sophisticated end users
  - Standalone users
- System Analysts and Application Programmers

I . Database Administrators

- A person who has such central control over the system is called a database administrator (DBA).
- **Role of DBA**
  - Schema definition.
    - The DBA creates the original database schema by executing a set of data definition statements in the DDL
  - Storage structure and access-method definition.
    - The DBA may specify some parameters pertaining to the physical organization of the data and the indices to be created.

- Schema and physical-organization modification
  - The DBA carries out changes to the schema and physical organization to reflect the changing needs of the organization
- Granting of authorization for data access.
  - By granting different types of authorization, DBA can regulate which parts of the database various users can access

Routine Maintenance:

- Routine maintenance activities of DBA are
- Periodically backing up the database
- Ensuring that enough free disk space is available for normal operation
- Monitoring jobs running on the database and ensuring that performance is not degraded by expensive tasks submitted by some users

II Database Designers

- Responsible for
  - identifying the data to be stored in the database
  - choosing appropriate structures to represent and store this data
  - communicate with all database users in order to understand their requirements and to create a design that meets these requirements

III End Users

- They are the people, whose jobs are to access the database for querying, updating, and generating reports
- Categories of end users
  - **1. Casual end users** occasionally access the database, but they may need different information each time.
  - They use a sophisticated database query language to specify their requests

2. **Naive** or **parametric end users**

- constantly query and update the database, using standard types of queries and updates

- The tasks that such users perform are
  - Bank tellers check account balances and post withdrawals and deposits
  - Reservation agents for airlines, hotels, and car rental companies check availability for a given request and make reservations

3. **Sophisticated end users**

- They include engineers, scientists, business analysts, and others
- They familiarize themselves with the facilities of the DBMS in order to implement their own applications

4. **Standalone users**

- maintain personal databases by using ready-made program packages that provide easy-to-use menu-based or graphics-based interfaces
- An example is the user of a tax package that stores a variety of personal financial data for tax purposes

IV System Analysts and Application Programmers (Software Engineers)

- **System analysts** determine the requirements of end users and develop specifications that meet these requirements.
- **Application programmers** implement these specifications as programs; then they test, debug, document, and maintain these system.
- Such analysts and programmers—commonly referred to as **software developers or software engineers**

Workers behind the Scene

- These are people who work to maintain the database system environment, and they are not interested in the database content

1. **DBMS system designers and implementers**
2. **Tool developers**
3. **Operators and maintenance personnel**

1. DBMS system designers and implementers

- They design and implement the DBMS modules and interfaces as a software package

- DBMS modules include
  - Module for implementing the catalog
  - query language processing
  - Interface processing
  - accessing and buffering data
  - controlling concurrency
  - handling data recovery and security

2. **Tool developers**

- Tools are optional packages that are often purchased separately.
- They include
  - packages for database design
  - performance monitoring
  - natural language or graphical interfaces
  - Prototyping
  - Simulation, and test data generation.
- In many cases, independent software vendors develop and market these tools.

3. **Operators and maintenance personnel**

- responsible for the actual running and maintenance of the hardware and software environment for the database system

Database Management System

Module 1

Lect 5 : Structured, Semi Structured & Unstructured data and DATA MODELS

- Data can be classified into three types:
  - Structured Data
  - Semi-structured Data
  - Unstructured Data

Structured Data

- The data is well organized either in the form of tables or some other way
- Searching & accessing information from such data is very easy
- Eg Relational Database (data stored in tables), spreadsheets

Semi - structured data

- Semi-structured data is information that doesn't reside in a relational database but have some organizational properties that make it easier to analyze.
- It is difficult to store and retrieve
- These data can be stored in a database after processing it
- For example, CSV, XML and web data such as JSON

Unstructured data

- Data which is not organized in a predefined manner
- It is not fit for relational database
- It includes text and multimedia content. For example, e-mail messages, word processing documents, videos, photos, audio files, presentations, web pages

Unstructured data

The university has 5600 students.
John's ID is number 1, he is 18 years old and already holds a B.Sc. degree.
David's ID is number 2, he is 31 years old and holds a Ph.D. degree. Robert's ID is number 3, he is 51 years old and also holds the same degree as David, a Ph.D. degree.

Semi-structured data

```
<University>
  <Student ID="1">
    <Name>John</Name>
    <Age>18</Age>
    <Degree>B.Sc.</Degree>
  </Student>
  <Student ID="2">
    <Name>David</Name>
    <Age>31</Age>
    <Degree>Ph.D. </Degree>
 </Student>
 ....
</University>
```

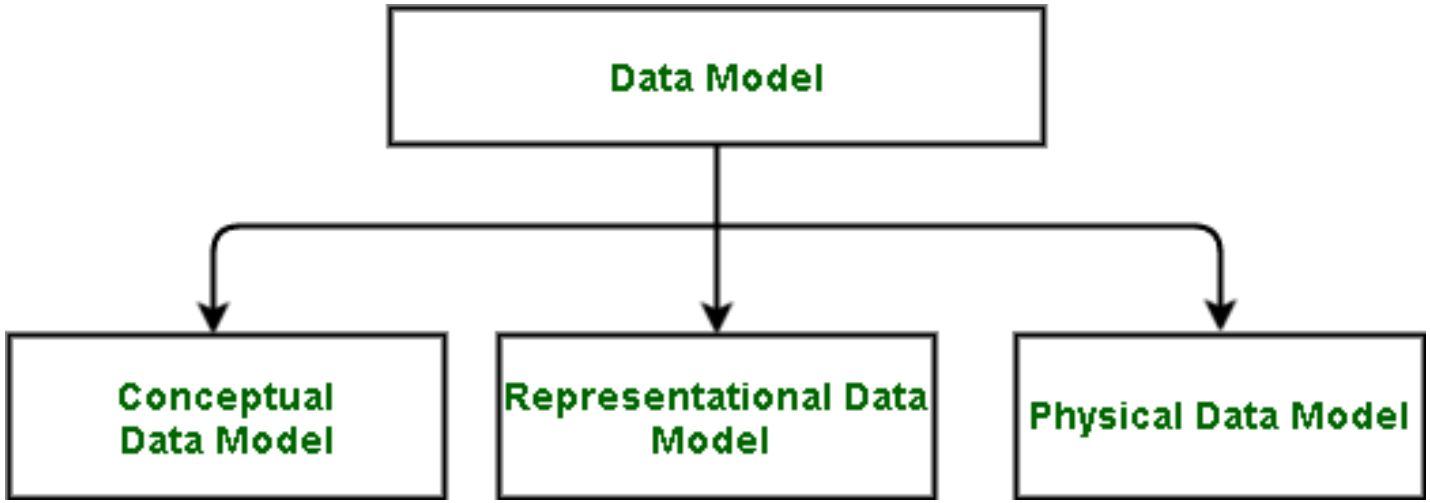
Structured data

ID	Name	Age	Degree
1	John	18	B.Sc.
2	David	31	Ph.D.
3	Robert	51	Ph.D.
4	Rick	26	M.Sc.
5	Michael	19	B.Sc.

DATA MODELS

DATA MODELS

- Collection of concepts that can be used to describe the structure of a database
- Structure of a database we mean the data types, relationships, and constraints that should hold on the data
- Most data models also include a set of basic operations for specifying retrievals and updates on the database


Data Model
Conceptual Data Model
Representational Data Model
Physical Data Model

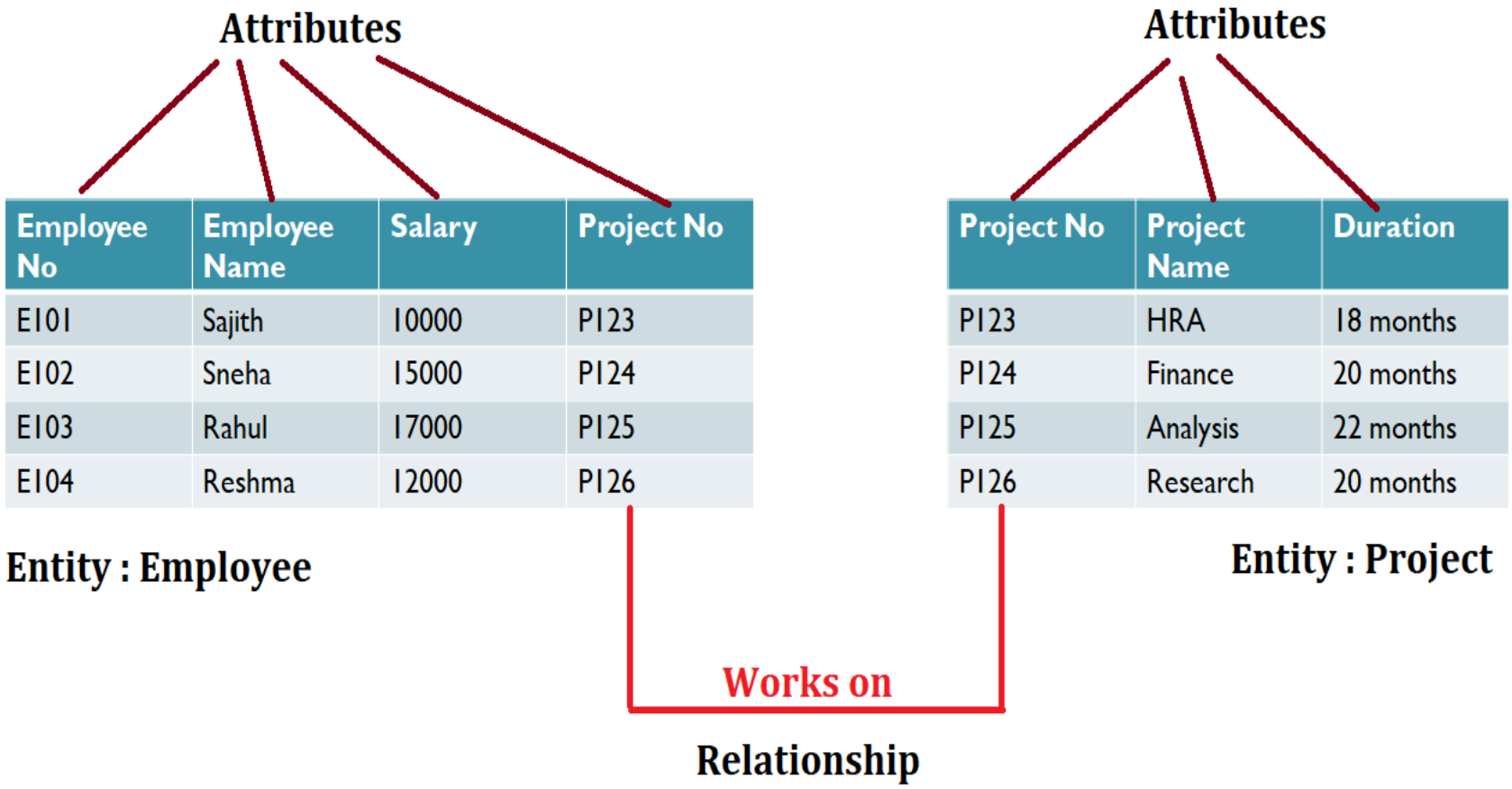
Categories of Data Model

- Every database and DBMS is based on a particular data model.
- There are mainly three categories of data model
  - **1. High-level or conceptual data models**
  - **2. Low-level or physical data models**
  - **3. Representational (or implementation) data models**
    - Hierarchical data model
    - Relational data model
    - Network Data model

1. High-level or conceptual data models

- It provides concepts that are close to the way many users perceive data.
- Conceptual data models use concepts such as
  - Entities
  - Attributes, and
  - Relationships

- An **entity** represents a real-world object or concept, such as an employee or a project from the miniworld that is described in the database.
- An **attribute** represents some property of interest that describes an entity, such as the employee's name or salary.
- A **relationship** among two or more entities represents an association among the entities, for example, a works-on relationship between an employee and a project


Attributes
Attributes
Employee No
Employee Name
Salary
Project No
E101
Sajith
10000
P123
E102
Sneha
15000
P124
E103
Rahul
17000
P125
E104
Reshma
12000
P126
Project No
Project Name
Duration
P123
HRA
18 months
P124
Finance
20 months
P125
Analysis
22 months
P126
Research
20 months
Entity : Employee
Entity : Project
Works on
Relationship

2. Representational (or implementation) data models

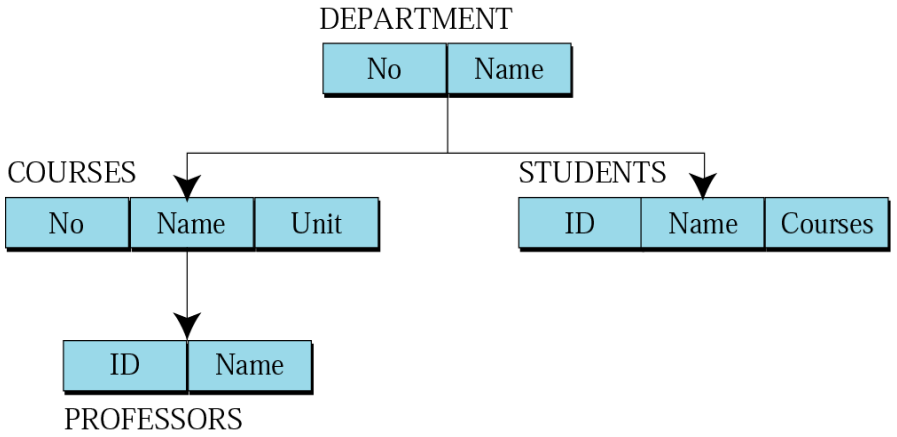
- It provide concepts that may be understood by end users but hides the way data is organized within the computer
- It hides some details of data storage but can be implemented on a computer system
- These are again classified into three types.
  - Hierarchical data model
  - Relational data model
  - Network Data model

1. Hierarchical Model

- Here data is organized in a tree-like structure, where nodes represent the records and relationship is provided using edges.

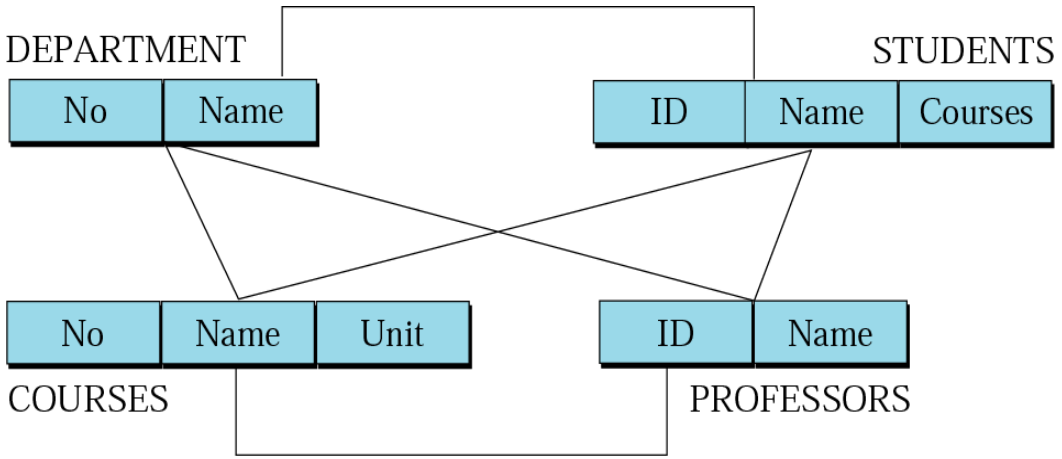
- One of the rules of a hierarchical database is that a parent can have multiple children, but a child can only have one parent.

- For example consider the database which represents an institutional department. A department has courses and students, each course has professors



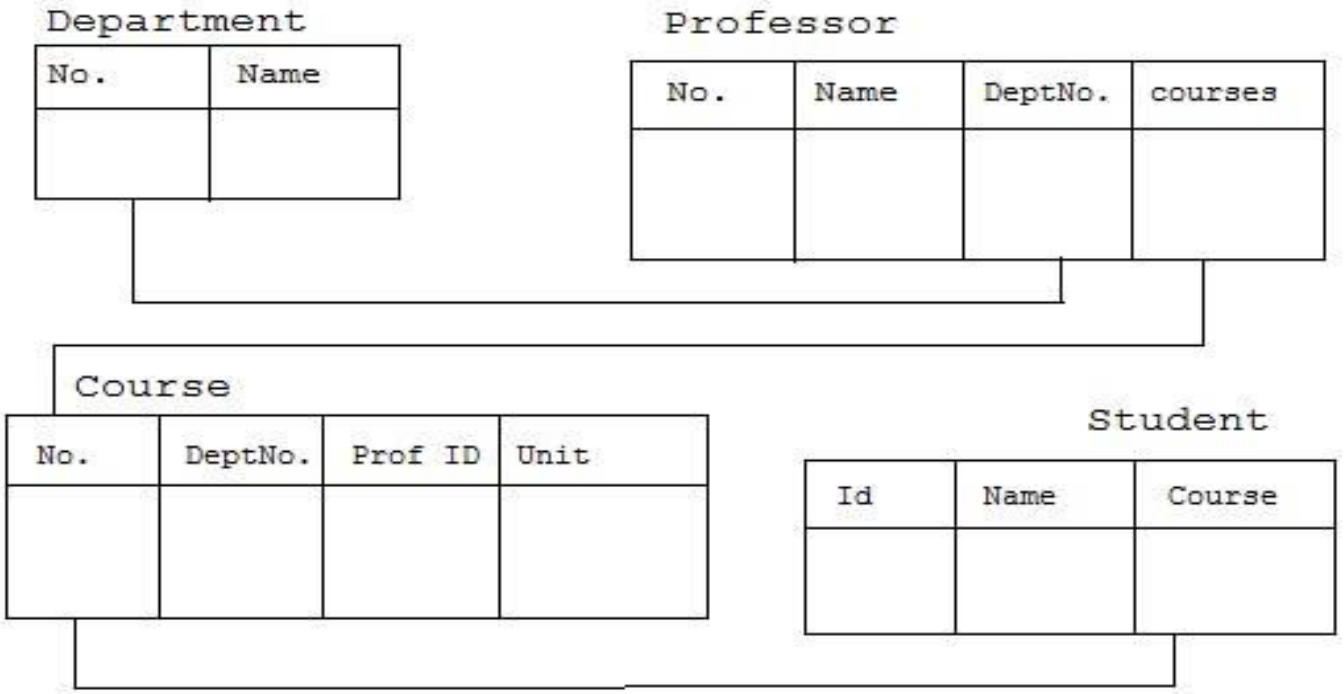
2. **Network model**

- Every data is represented as collection of records.
- The database structure is like a graph.
- This is similar to the hierarchical model and also provides a tree-like structure. However, a child is allowed to have more than one parent.



3. **Relational Model**

- Data is organized in two-dimensional tables called **relations**.
- The tables or relation are related to each other.
- Each row represents a record, also referred to as an entity.
- Each column represents a field, also referred to as an attribute.
- A relational DBMS uses multiple tables to organize the data.
- Relationships are used to link the various tables together.
- Relationships are created using a field that uniquely identifies each record.


Department
No.
Name
Professor
No.
Name
DeptNo.
courses
Course
No.
DeptNo.
Prof ID
Unit
Student
Id
Name
Course

3. Low-level or physical data models

- It provides concepts that describe the details of how data is stored in the computer
- Data models are generally meant for computer specialists, not for typical end users
- It **describe how data is stored** as files in the computer by representing information such as
  - record formats
  - record orderings and
  - access paths
- An **access path** is a structure that makes the search for particular database records efficient

Database Management System

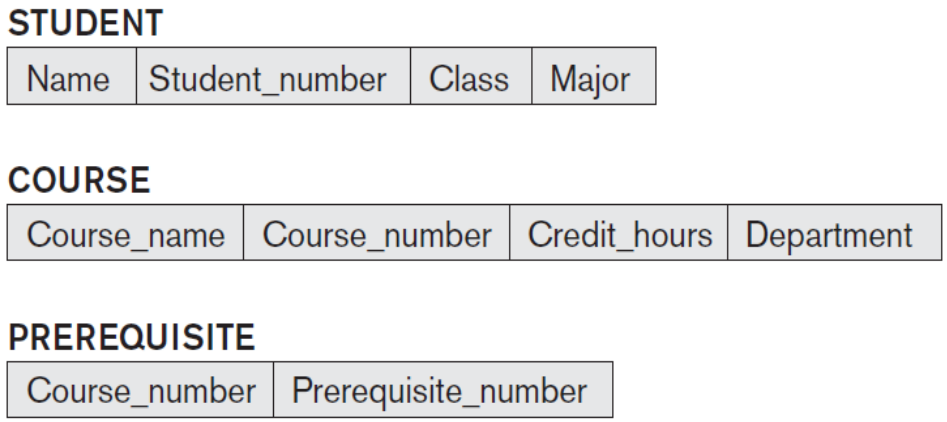
Module 1

Lect 6 : THREE-SCHEMA ARCHITECTURE

Schema, Instances and Database State

- The description of a database is called the **Database schema**, which is specified during database design and is not expected to change frequently
- A Schema is represented in a Schema Diagram

Figure 2.1
Schema diagram for the database in Figure 1.2.

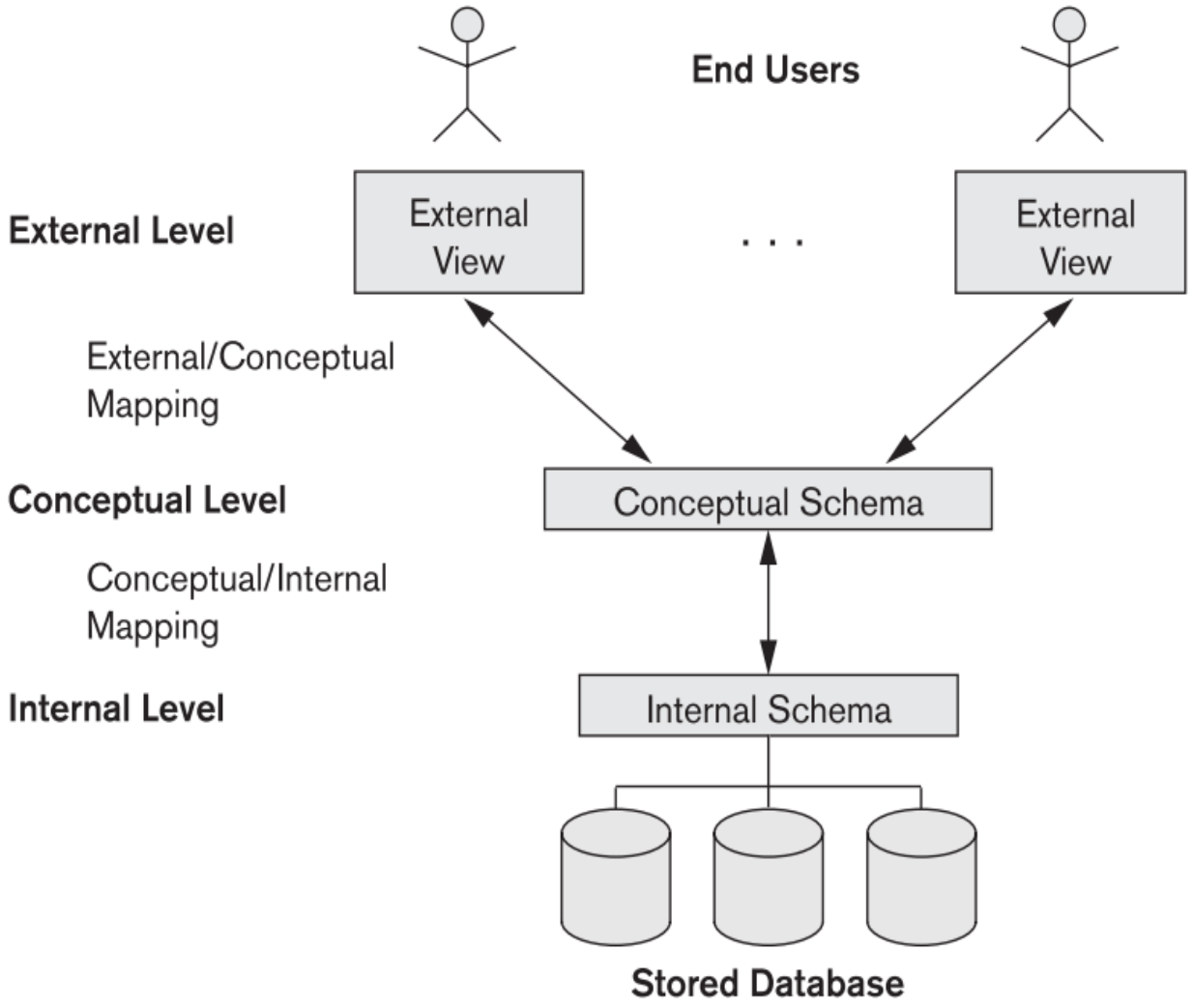


- The data in the database at a particular moment in time is called a **database state or snapshot**
- Instance is the snapshot of a database at a particular moment.

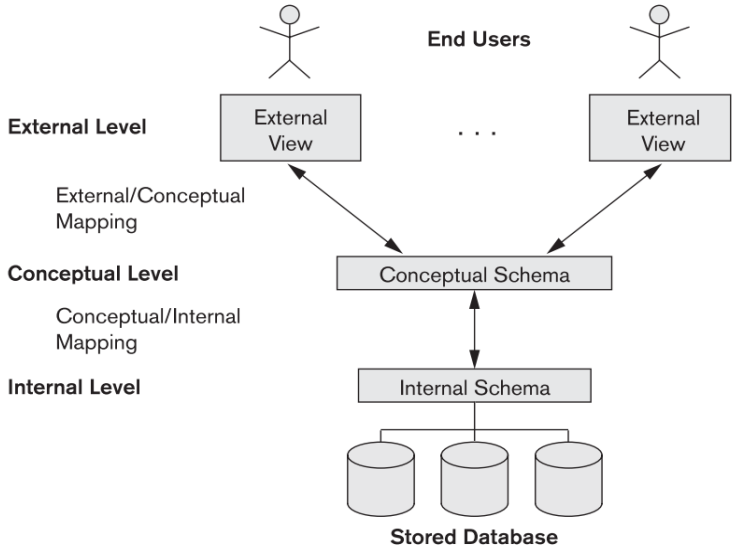
Name	Student_number	Class	Major
Ram	CS001	R4	CSE
Shyam	CS002	R4	CSE

Database state

THREE SCHEMA ARCHITECTURE

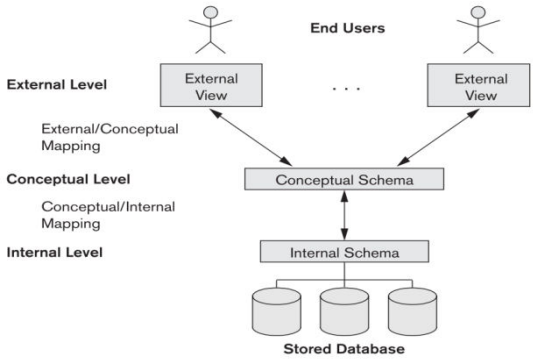


- This architecture is proposed to achieve the characteristics of database approach
- The goal of the three-schema architecture is to **separate the user applications and the physical database**
- In this architecture, schemas can be defined at the following three levels
  - External Level
  - Conceptual Level
  - Internal Level



INTERNAL LEVEL

- It uses low level /physical data model
- It has an **internal schema**, which **describes the physical storage structure of the database**
- This level describes **how data is stored in the database.**
- It deals with the data structures and the file organizations used to store data on storage device
- This level is concerned with things such as
  - Storage space allocation for data and indexes
  - Record description for storage
  - Record Placement
  - Data encryption and data decryption



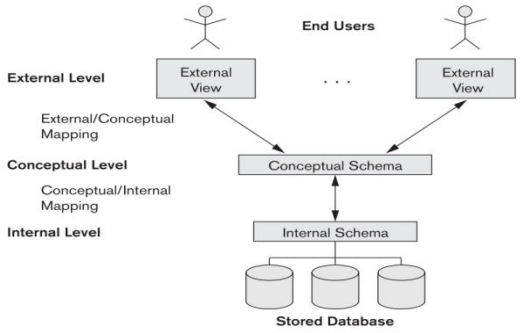
CONCEPTUAL LEVEL

- It uses representational/Implementational data model
- The conceptual level has a **conceptual schema**, which **describes the structure of the whole database for users**

- hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints

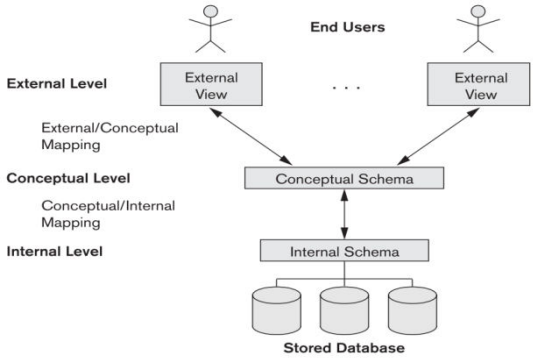
- describes **what data is stored in the database and the relationships among the data.**

- The conceptual level represents
  - All entities, their attributes and their relationships
  - The constraints on the data
  - Semantic information about the data
  - Security and integrity information



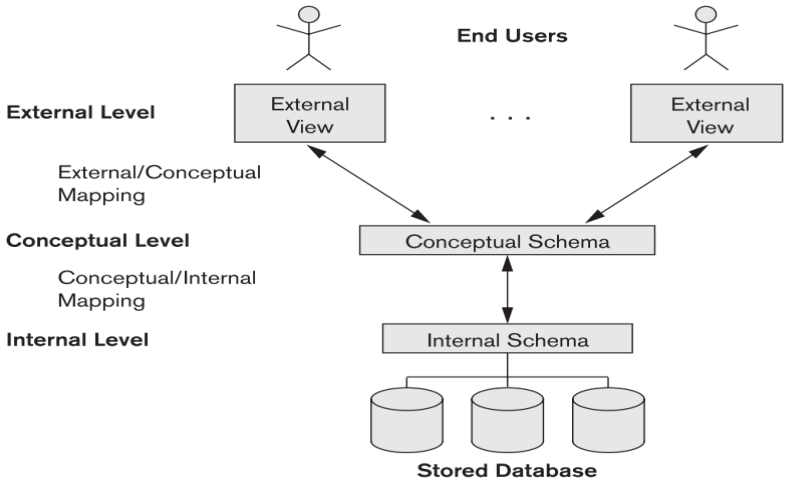
EXTERNAL OR VIEW LEVEL

- It uses HighLevel/Conceptual data Model
- Describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group
- DBMS must transform a request specified on an external schema into a request against the conceptual schema, and then into a request on the internal schema for processing over the stored database
- If the request is a database retrieval, the data extracted from the stored database must be reformatted to match the user's external view.



Mappings

- The processes of transforming requests and results between levels are called **mappings**
- These mappings may be time-consuming



Univ Qstns

1. List any 3 categories of database users, highlighting any one important characteristics of each category
2. 2. List any 3 salient features of database System
3. What are the responsibilities of DBA
4. Define the following terms

(a) Data Model

(b) Database Schema

(c)Meta-data

5. With neat diagram explain the 3 schema architecture
6. Illustrate with an example, the difference between the conceptual data models and the physical data models

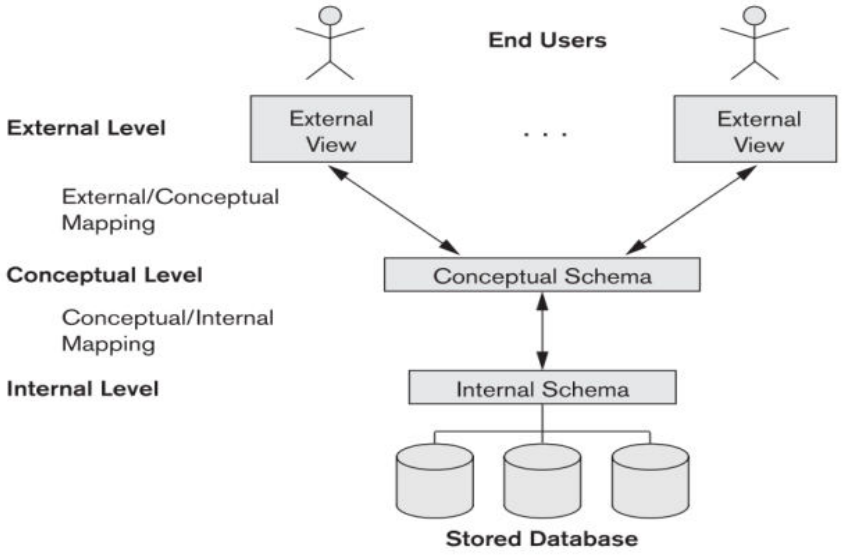
Database Management System

Module 1

Lect 7 :
->Data Independence
->Database Architectures

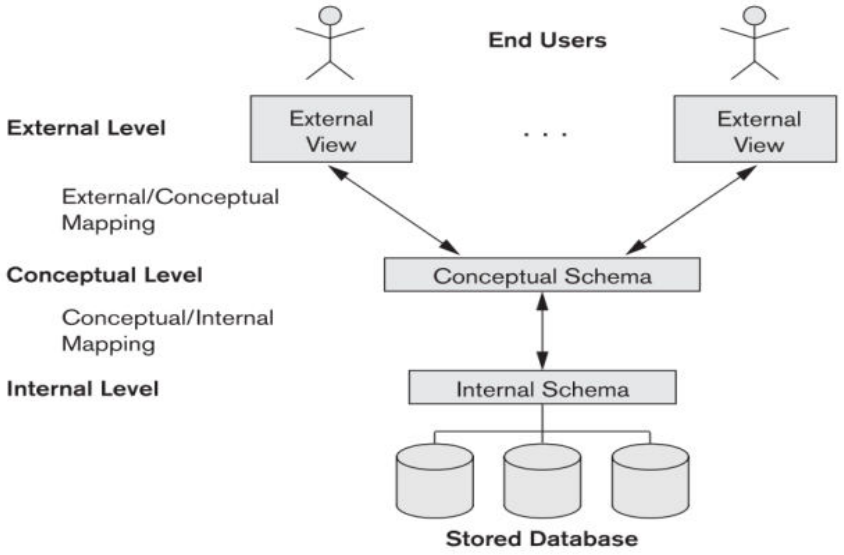
DATA INDEPENDENCE

- Data independence is the capacity to change the schema at one level of a database system without having to change the schema at the next higher level.
- There are two types of data independence.
  - **Logical data independence**
  - **Physical data independence**

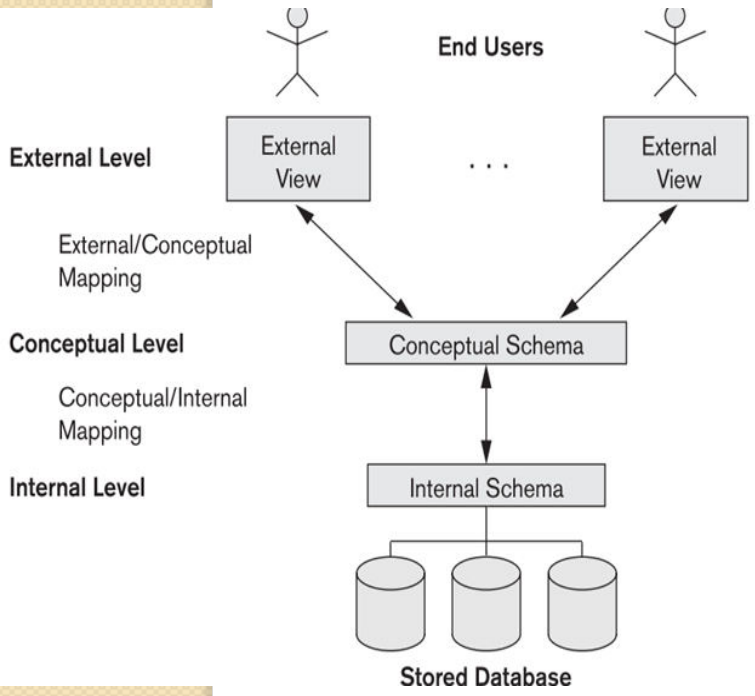


1. Logical Data Independence

- The capacity to change the conceptual schema without having to change external schemas or application programs
- Refers to the immunity of external schemas to changes made in the conceptual schema.



- Eg: We may change the conceptual schema to expand the database (by adding a record type or data item), or to reduce the database (by removing a record type or data item).
- After the conceptual schema undergoes a logical reorganization, application programs that reference the external schema constructs must work as before



Roll	Name	Marks
1	John	42
2	Micheal	35
3	Ebin	38

Roll	Name	Marks

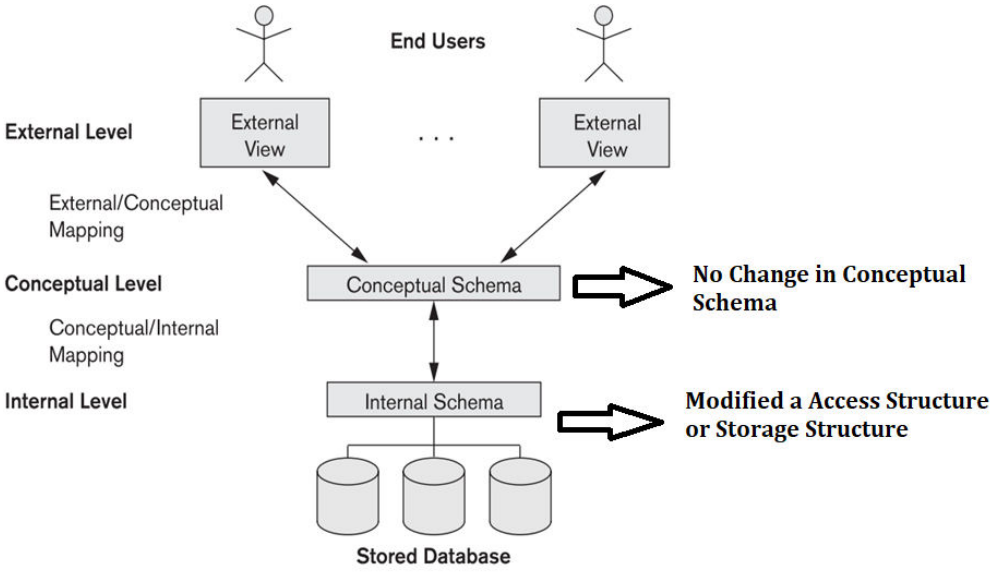
Roll	Name	Marks	DOB
1	John	42	19-02-1990
2	Micheal	35	14-05-1990
3	Ebin	38	12-12-1989

Roll	Name	Marks	DOB

2. Physical Data Independence

- It is the capacity to change the internal schema without having to change the conceptual schema or
- it refers to the immunity of conceptual schemas to the changes of internal schema

- Eg : for example, by creating additional access structures—to improve the performance of retrieval or update.
- If the same data as before remains in the database, we should not have to change the conceptual schema

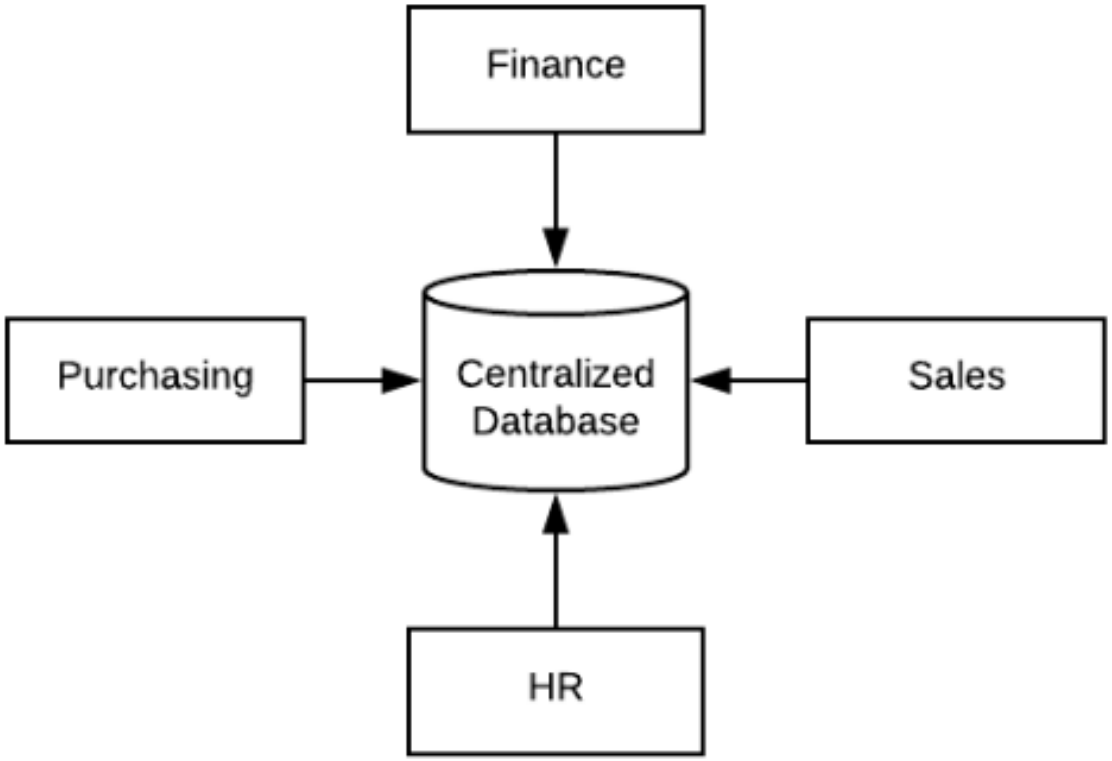


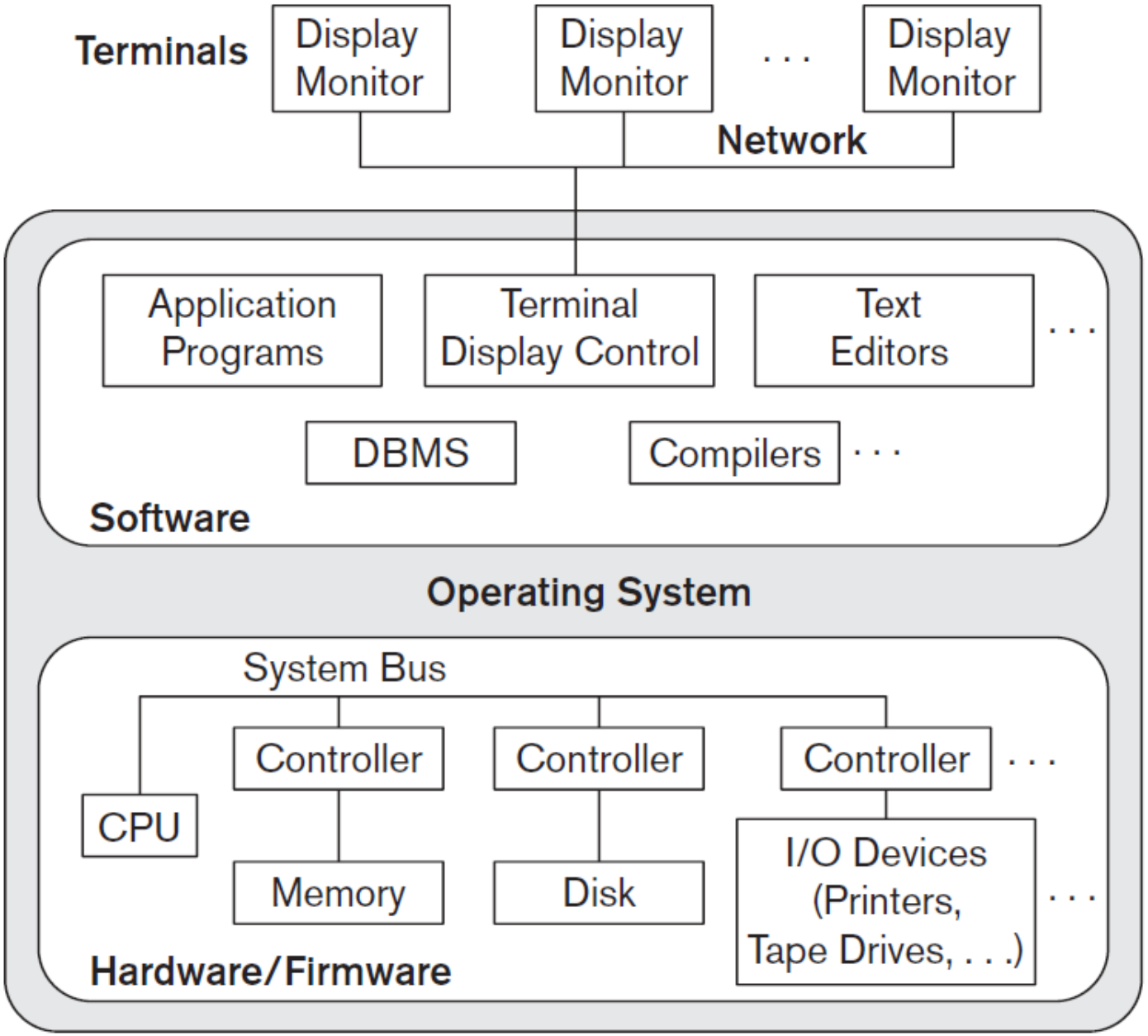
Database Architectures

- DBMS architecture are of different types
  - **Centralized DBMSs Architecture**
  - **Basic Client/Server Architectures**
  - **Two-Tier Client/Server Architectures**
  - **Three-Tier and n-Tier Architectures**

1. Centralized DBMSs Architecture

- All the DBMS functionality, application program execution, and user interface processing were carried out on one machine
- A centralized database is stored at a single location such as a mainframe computer.
- It is maintained and modified from that location only and usually accessed using an internet connection such as a LAN or WAN.
- The centralized database is used by organizations such as colleges, companies, banks etc.


Finance
Purchasing
Centralized Database
Sales
HR


Terminals
Display Monitor
Display Monitor
Display Monitor
Network
Application Programs
Terminal Display Control
Text Editors
DBMS
Compilers
Software
Operating System
System Bus
Controller
Controller
Controller
CPU
Memory
Disk
I/O Devices (Printers, Tape Drives, . . .)
Hardware/Firmware

2. Basic Client/Server Architectures

- The **client/server architecture was developed to deal with computing environments** in which a large number of PCs, workstations, file servers, printers, database servers, Web servers, e-mail servers, and other software and equipment are connected via a network
- The idea is to define specialized servers with specific functionalities
- The resources provided by specialized servers can be accessed by many client machines.
- Eg
  - **file server** maintains the files of the client machines**.**
  - **printer server** is connected to various printers

- The **client machines** provide the user with the appropriate interfaces to utilize these servers

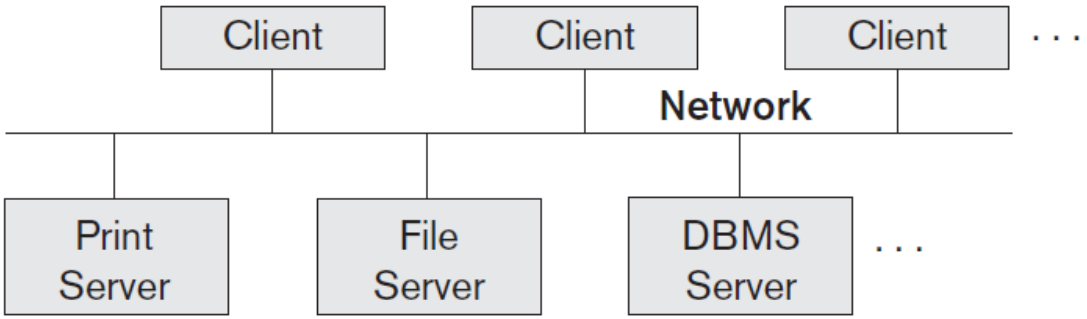
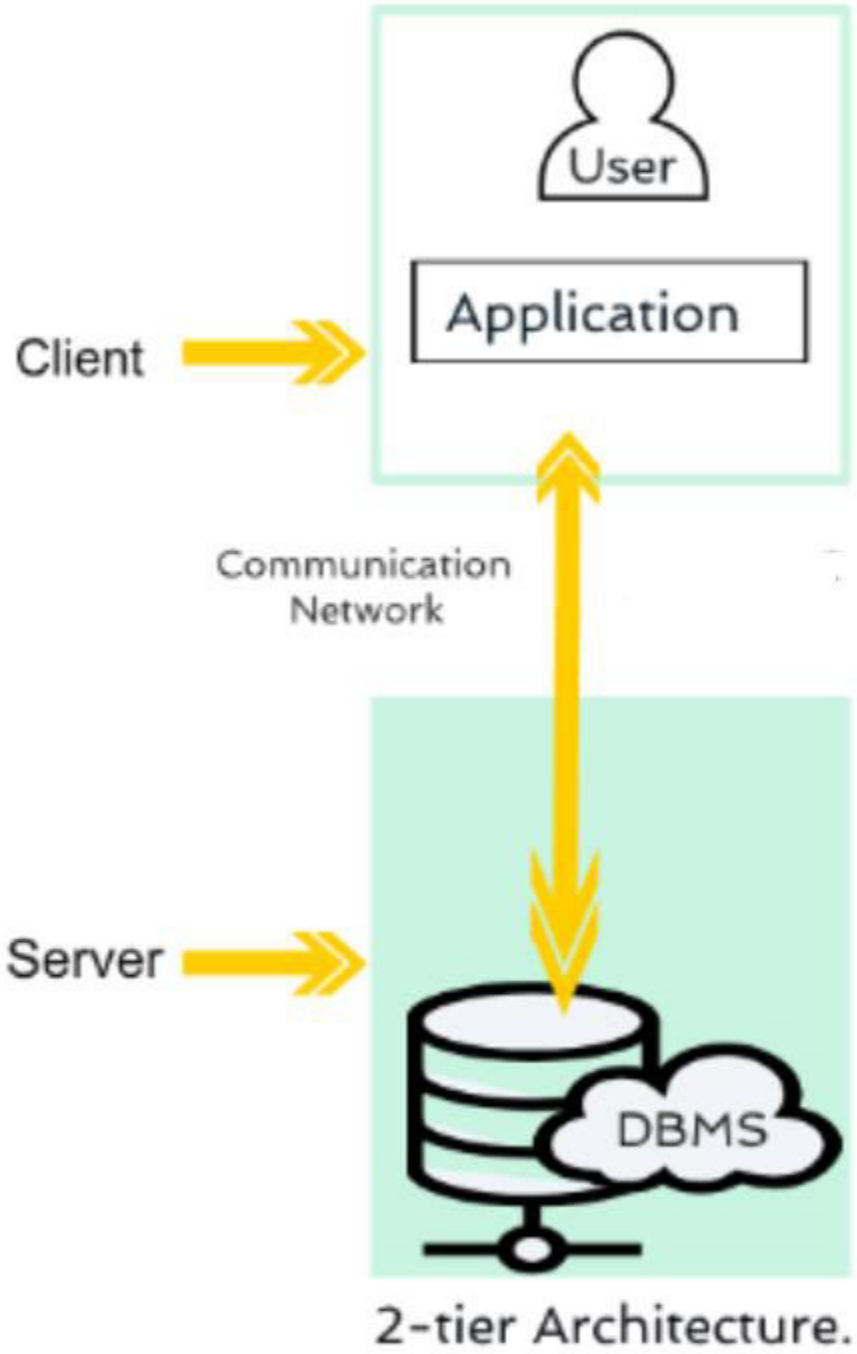


Figure 2.5
Logical two-tier client/server architecture.

- A client is typically a user machine that provides user interface capabilities and local processing.
- When a client requires access to additional functionality— such as database access— that does not exist at that machine, it connects to a server that provides the needed functionality.
- A server is a system containing both hardware and software that can provide services to the client machines

Two-Tier Client/Server Architectures

- Multiple clients are connected to a database server
- The user interface programs and application programs can run on the client side
- Query provided in the client side is processed by the database server
- When DBMS access is required, the program establishes a connection to the DBMS

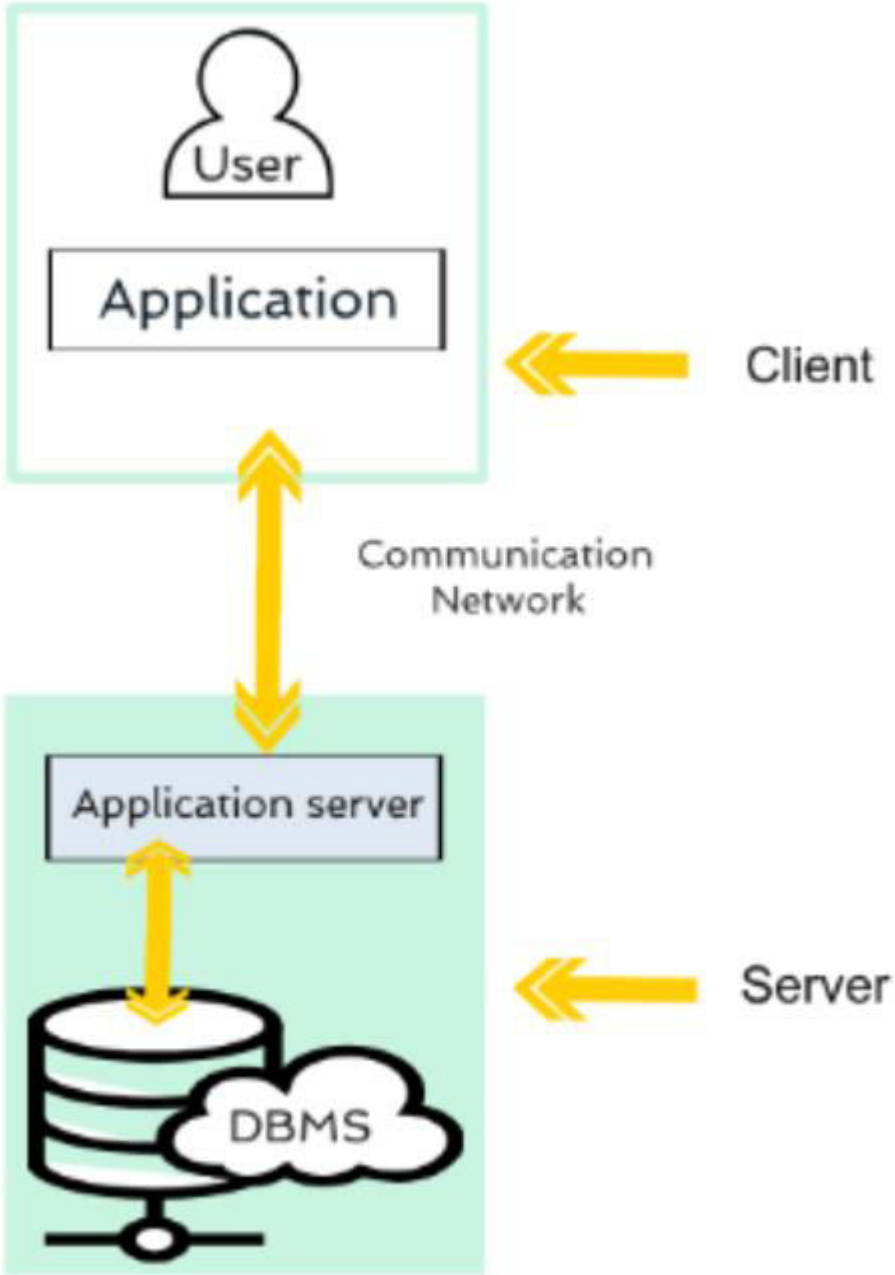


2-tier Architecture.

- A standard called Open Database Connectivity (ODBC) allows client-side programs to communicate with DBMS,.
- Most DBMS vendors provide ODBC drivers for their systems
- This architecture is called Two tier because software components are distributed over two systems: Client & Server

Three-Tier and n-Tier Architectures

- It adds an intermediate layer between the client and the database server.
- This intermediate layer or middle tier is called the application server or the Web server, depending on the application.
- Web server plays an intermediary role by running application programs and storing business rules (procedures or constraints) that are used to access data from the database server.
- It can also improve database security by checking a client's credentials before forwarding a request to the database server



3-tier Architecture.

- **Clients** contain GUI interfaces and some additional application-specific business rules.
- The **Application server** accepts requests from the client, processes the request and sends database queries and commands to the **database server**,
- Response from the database is partially processed by database server which is passed to the clients, where it may be processed further and filtered to be presented to users in GUI format

Database Management System

Module 1

Lect 8 :
->Classifications of DBMS
->Database Languages

Classification of DBMS

CLASSIFICATION OF DBMS

- DBMS can be classified based on several criteria.
  - **1. Based on Data Model**
  - **2. Based on number of users**
  - **3. Based on Database distribution**
  - **4. Based on cost**
  - **5. Based on access path**
  - **6. Based on purpose**

1. Based on Data Model

- Relational DBMS: It is most commonly used commercial DBMS.
- Object DBMS: The object data model has been implemented in some commercial systems using object- oriented concepts.
- Hierarchical DBMS: Many legacy applications still run on database systems based on the hierarchical data models. Examples of hierarchical DBMSs include IMS (IBM)
- Network DBMS: The network data model was used by many vendors and by the products like IDMS, DMS 1100

2.Based on Number of Users

- Single-user systems support only one user at a time and are mostly used with PCs.
- Multiuser systems, which include the majority of DBMSs, support concurrent multiple users

3.Based on Database distribution

- Centralized DBMS: Centralized DBMS can support multiple users, but the DBMS and the database reside totally at a single computer site.

- Distributed DBMS: A distributed DBMS (DDBMS) can have the actual database and DBMS software distributed over many sites, connected by a computer network.

- Homogeneous DDBMSs use the same DBMS software at all the sites, whereas heterogeneous DDBMSs can use different DBMS software at each site

4. Based on cost

- Open Source (Free) DBMS products like MySQL and PostgreSQL that are supported by third-party vendors with additional services.

- RDBMS products are available as free examination 30-day copy versions as well as personal versions, which may cost under $100 and allow a fair amount of functionality.

5. Based on access path

- DBMS can be classified based on the types of access path options for storing files.
- Access paths are the alternative ways for retrieving specific record from a relation/table
- One well-known family of DBMSs is based on inverted file structures.

6. Based on purpose

- DBMS can be general purpose or special purpose.
- When performance is a primary consideration, a special-purpose DBMS can be designed and built for a specific application
- such a system cannot be used for other applications without major changes.
- Many airline reservations and telephone directory systems are special-purpose DBMSs. These fall into the category of online transaction processing (OLTP) systems,

Database Languages

DATABASE LANGUAGES

- Database languages can be classified into three categories.
  - 1. Data Definition Language (DDL)
  - 2. Data Manipulation Language (DML)
  - 3. Data Control Language (DCL)

1. Data definition language (DDL)

- A database schema can be specified by a by a special language called a data definition language
- **DDL is a set of commands to create, modify and delete database structure but not data**
- When data values are stored in a database, it must satisfy certain consistency constraints. DDL provides facilities to specify such constraints
- Domain constraints, integrity constraints, authorization etc. are different constraints.
- It is used by the DBA and by database designers to define both conceptual and internal schemas

2. Data Manipulation Language (DML)

- A DML is a language that **enables users to access or manipulate data**
- Through DML we can
  - Insert new data in the database
  - Modify the data already stored in the database
  - Delete data
  - Retrieve the information
- There are two main types of DMLs.
  - Procedural DML
  - Non procedural DML

- **Low level or Procedural DML**:
  - Require a user to specify what data are needed and how to get those data.
  - It requires writing procedures or methods to specify what data is needed.
  - A lowlevel or procedural DML must be embedded in a general-purpose programming language

- **High level or Non Procedural DML**:
  - Requires a user to specify what data are needed without specifying how to get those data.
  - A query is a statement requesting the retrieval of information.
  - Highlevel DMLs, such as SQL, can specify and retrieve many records in a single DML statement;
  - therefore, they are called **set-at-a-time** or **set-oriented DMLs**

3. Data Control language (DCL)

- It is the language that controls access to the data and to the database.
- DCL statements are grouped with the DML statements.
- Through DCL we can
  - Commit: Save work done
  - Rollback: restore database to original since the last commit

Database Management System

Module 1

Lect 9 :

ENTITY-RELATIONSHIP DIAGRAM

ENTITY-RELATIONSHIP MODEL

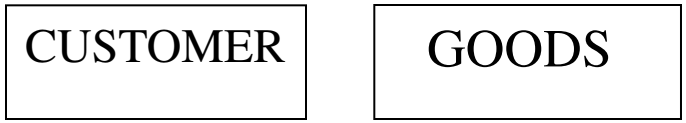
- Entity-Relationship (ER) model is a popular high-level conceptual data model
- Diagrammatic notation associated with an E-R model is known as **E-R Diagram**
- The E-R model describes data as Entities, Attributes and Relationships.

Entity Relationship Diagram (E-R Diagram)

- It is not a technical method
- It is used for designing a database
- Diagrammatic representation and easy to understand for non technical users
- Collection of **entities and their properties called attributes and relationship between them**
- Diagrammatic representation and easy to understand for non technical users

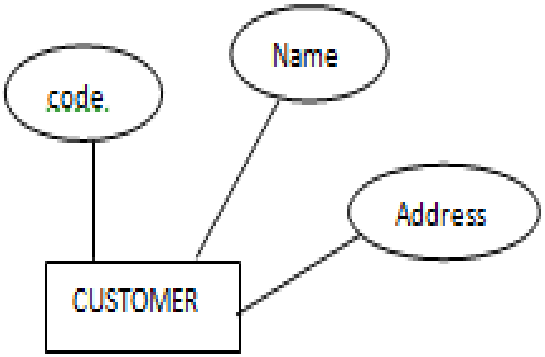
Entity

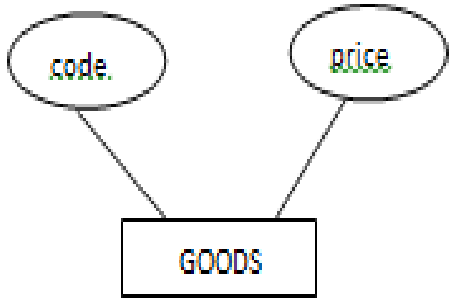
- Entity is a thing in the real world with an independent existence.
- for example, a particular person, car, house, or employee
- For example, if we say that a customer buys goods, it means customers and goods are entities.
- In E-R diagrams, entities are represented using rectangles



Attributes

- Each entity has attributes—the particular properties that describe it.
- For example, an EMPLOYEE entity may be described by the employee's name, age, address, salary, and job.
- In E-R diagrams attributes are drawn in elliptical shapes along with the entity rectangles



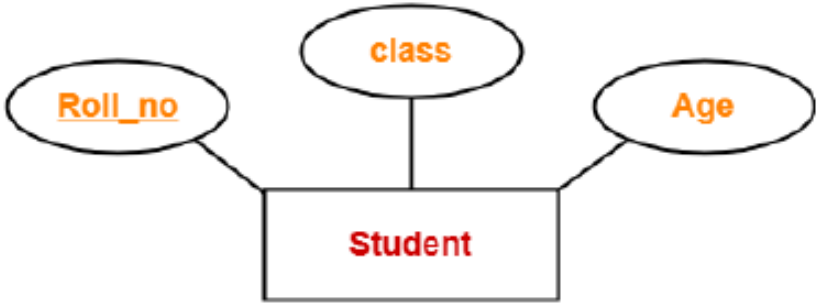


Types of attribute

- Simple attribute vs Composite attribute
- Single valued vs Multivalued attributes
- Stored vs Derived attributes

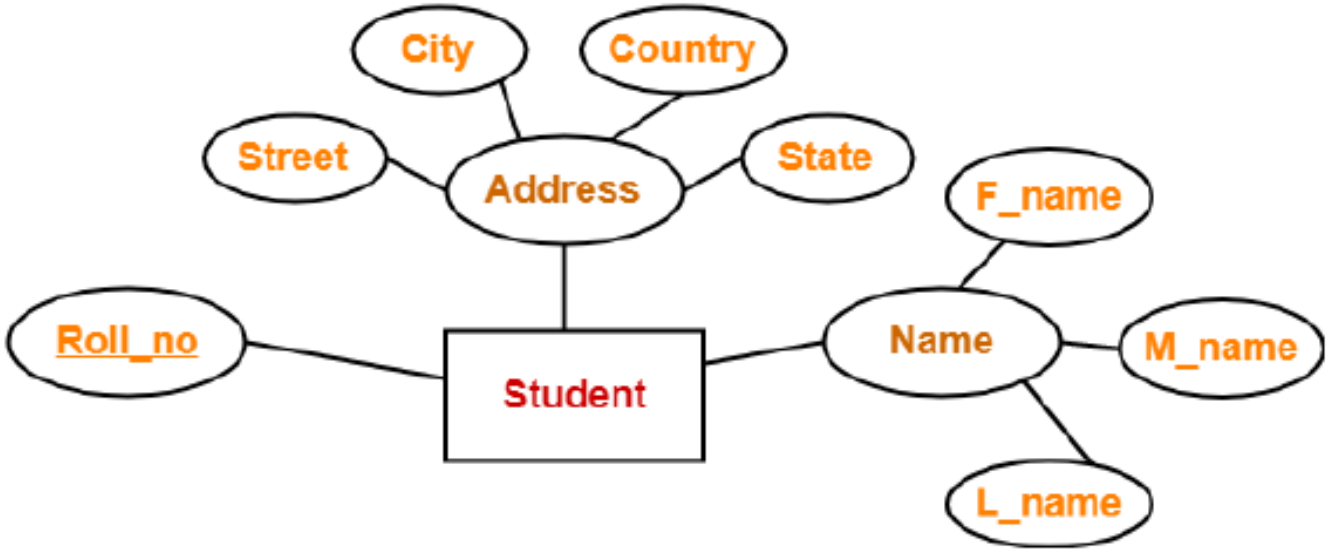
Simple attribute vs Composite attribute

- Simple attributes
  - Attributes which are not divisible that is they cannot be divided
  - Eg: City, State, etc,.



Here, all the attributes are simple attributes as they can not be divided further.

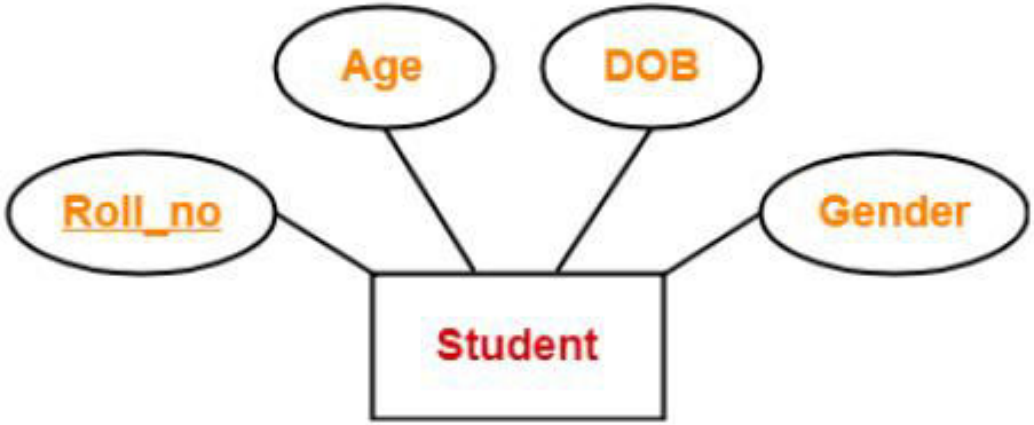
- Composite Attribute
  - Attributes that can be divided into smaller sub parts
  - Example: Name attribute can be divided into FirstName, MiddleName, LastName



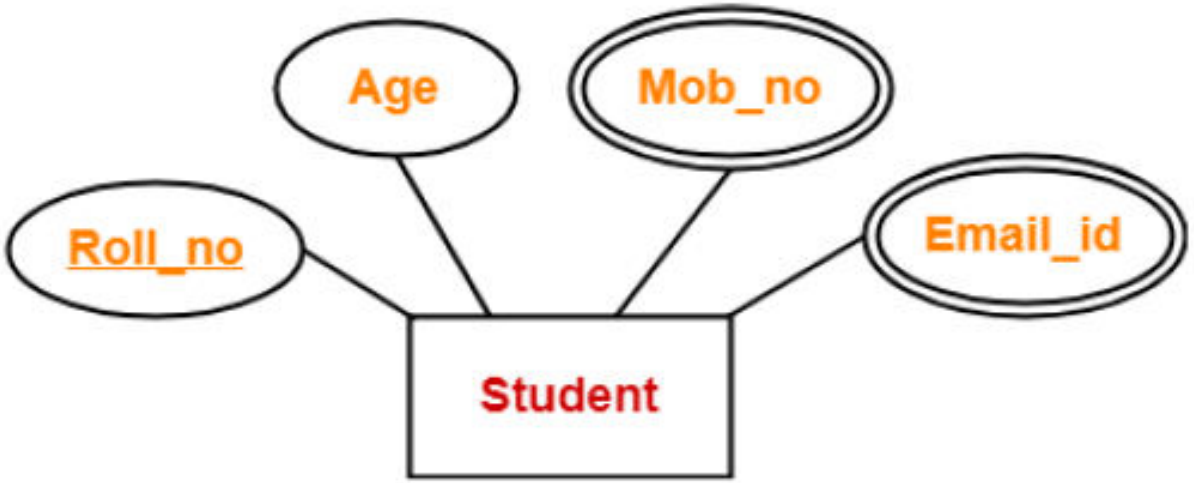
Here, the attributes "Name" and "Address" are composite attributes as they are composed of many other

Single valued vs Multivalued Attribute

- Single Valued
  - Attributes which are having single values
  - Example: Age



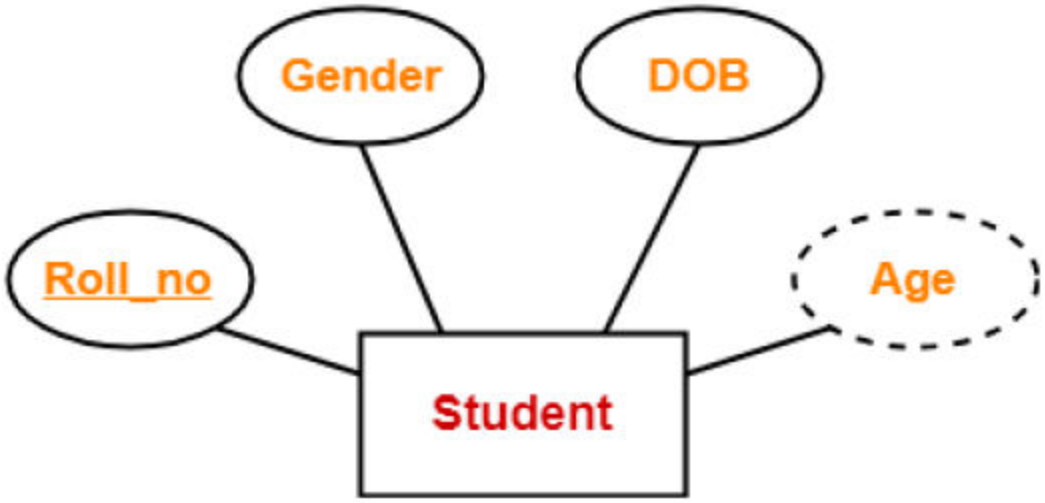
- Multi Valued Attributes
  - Multi valued attributes are those attributes which can take more than one value for a given entity from an entity set.
  - Represented by double oval



Stored Vs Derived Attribute

- In some cases, two (or more) attribute values are related ,for example, the Age and Birthdate attributes of a person.
- For a particular person entity, the value of Age can be determined from the current (today's) date and the value of that person's birthdate.
- The Age attribute is hence called a derived attribute and is said to be derivable from the birthdate attribute, which is called a stored attribute .

- Derived attributes are represented by dotted attributes



Here, the attribute “Age” is a derived attribute as it can be derived from the attribute “DOB”.

Complex Attributes

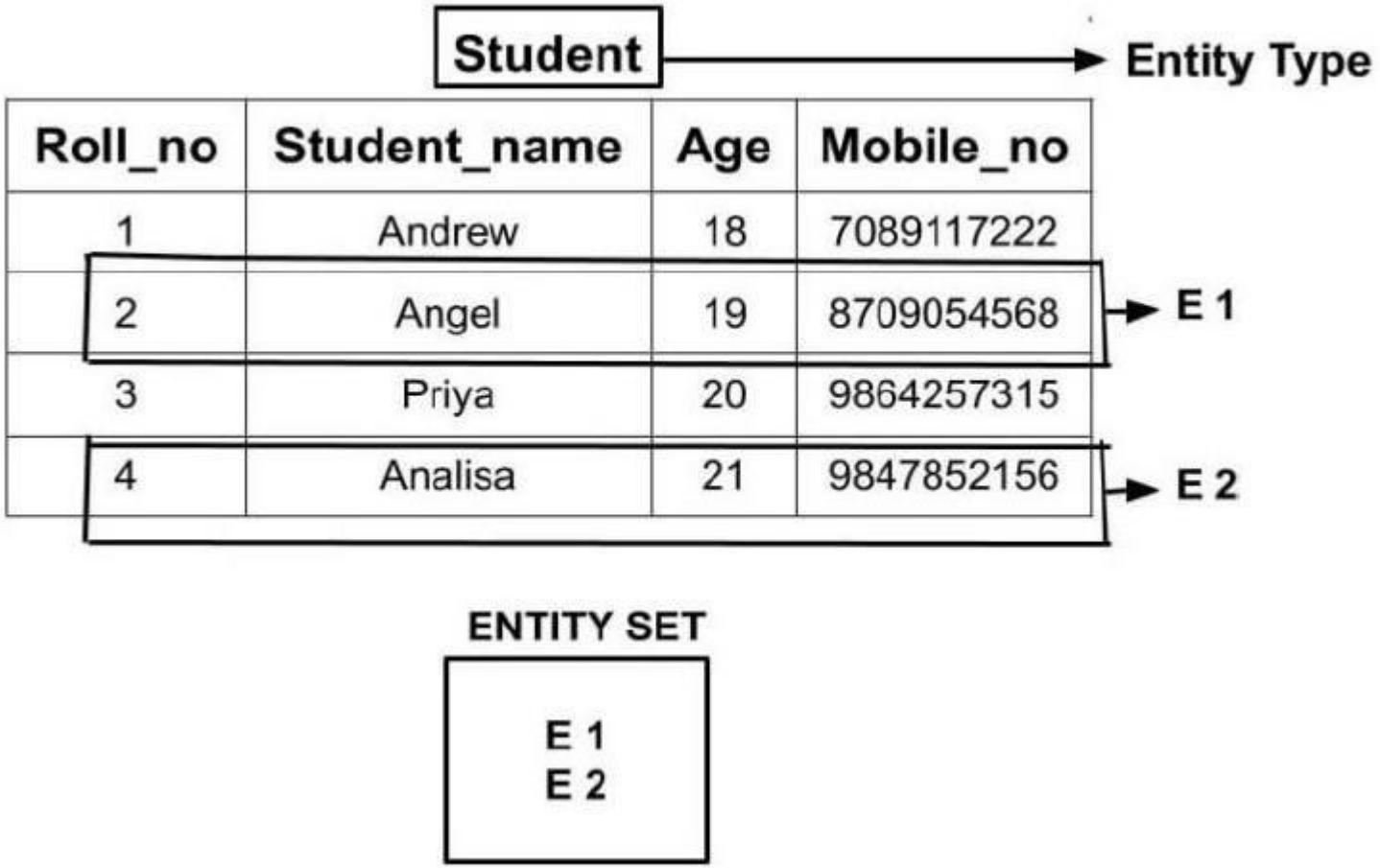
- It is a combination of composite and multi-valued attribute.
- Group components of a composite attribute between parentheses () and separating the components with commas, and by displaying multivalued attributes between braces { }.
- Eg : Address_phone({phone_no},{Address(Flat no,city,state)})

Null Valued Attributes

- Null value is a value which is not inserted but it does not hold zero value.
- The attributes which can have a null value called null valued attributes.
- Example: Mobile_no attributes of a person may not be having mobile phones.

Entity Types and Entity Sets

- An **entity type** defines a collection of entities that have the same attributes or properties.
- Each entity type in the database is described by its name and attributes.
- **Entity Set** is the collection of same type of entities, i.e their attributes are same is called entity set

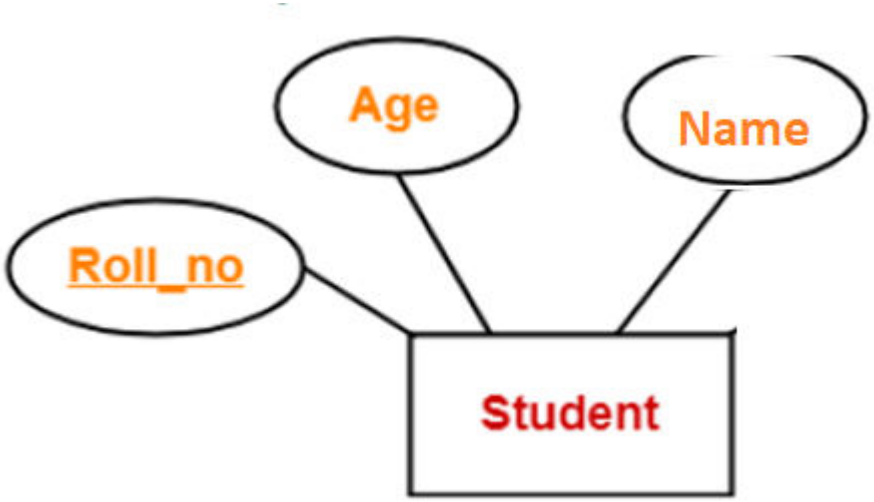

Student
Entity Type
Roll_no
Student_name
Age
Mobile_no
1
Andrew
18
7089117222
2
Angel
19
8709054568
E 1
3
Priya
20
9864257315
4
Analisa
21
9847852156
E 2
ENTITY SET
E 1
E 2

Key Attributes of an Entity

- An entity type usually has one or more attributes whose values are distinct for each individual entity in the entity set.
- Such an attribute is called a key attribute, and its values can be used to identify each entity uniquely

Roll no	Name	Age
1	Akhil S	21
2	Akhil	20
3	Anand	21

In ER diagrammatic notation, each key attribute has its name underlined inside the oval



Value Sets (Domains) of Attributes

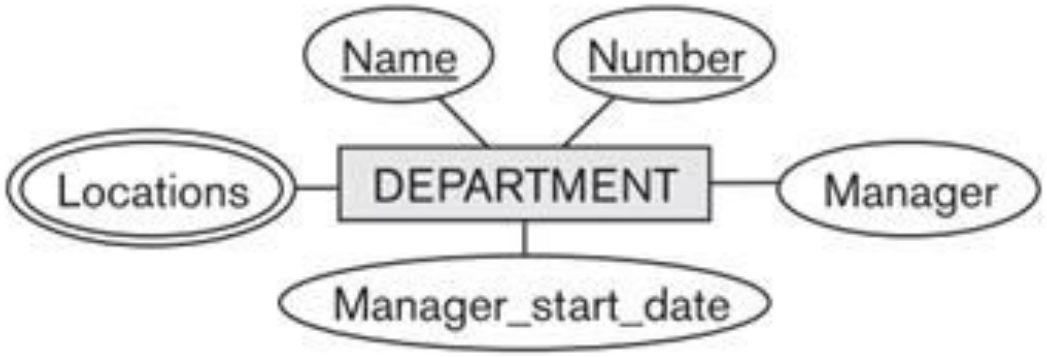
- Each simple attribute of an entity type is associated with a value set (or domain of values), which specifies the set of values that may be assigned to that attribute for each individual entity.

- For example, if the range of ages allowed for employees is between 16 and 70, we can specify the value set of the Age attribute of EMPLOYEE to be the set of integer numbers between 16 and 70.

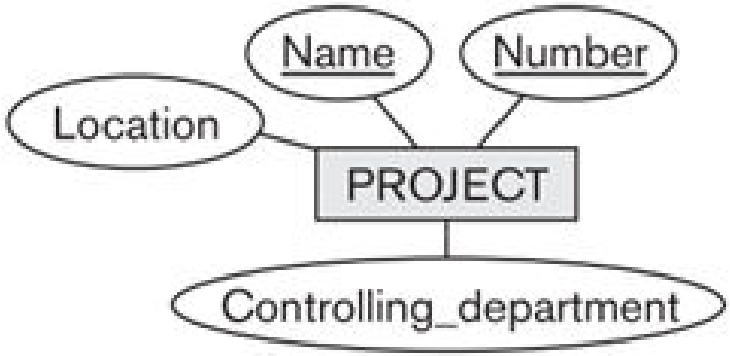
Emp No	Ename	Age
E101	Rahul	17
E102	Ramesh	23
E103	Samuel	**14 (Error)**

- Consider a database application COMPANY which keeps track of a company's employees, departments, and projects. Suppose that after the requirements collection and analysis phase, the database designers provide the following description.

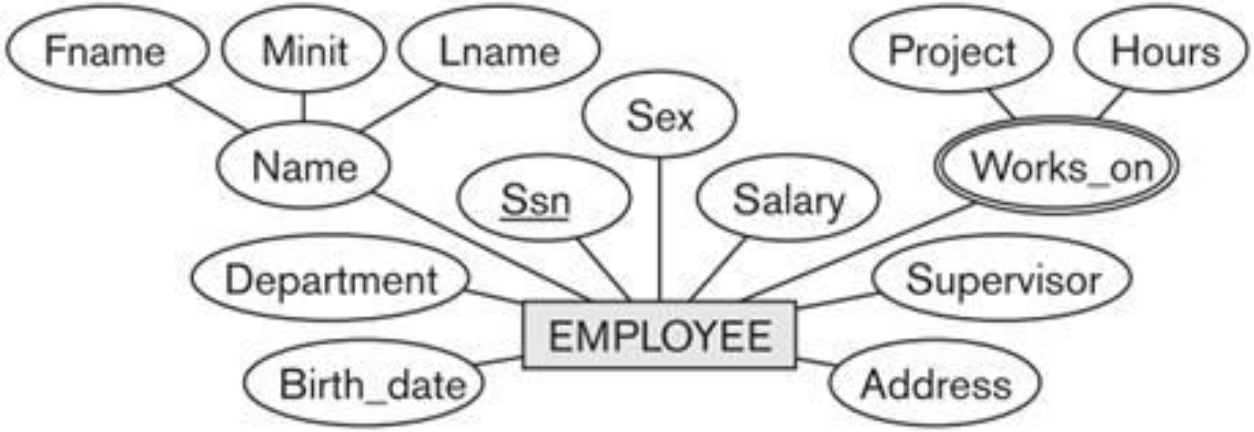
1. The company is organized into departments. Each department has a unique name, a unique number, and a particular employee who manages the department. We keep track of the start date when that employee began managing the department. A department may have several locations.



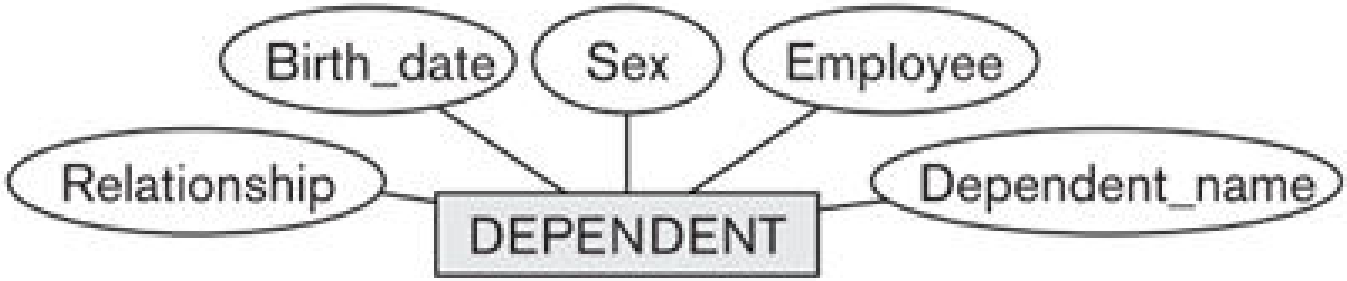
- 2. A department controls a number of projects, each of which has a unique name, a unique number, and a single location.



- We store each employee's name, Social Security number, address, salary, sex (gender), and birth date. An employee is assigned to one department, but may work on several projects, which are not necessarily controlled by the same department. We keep track of the current number of hours per week that an employee works on each project. We also keep track of the direct supervisor of each employee (who is another employee).

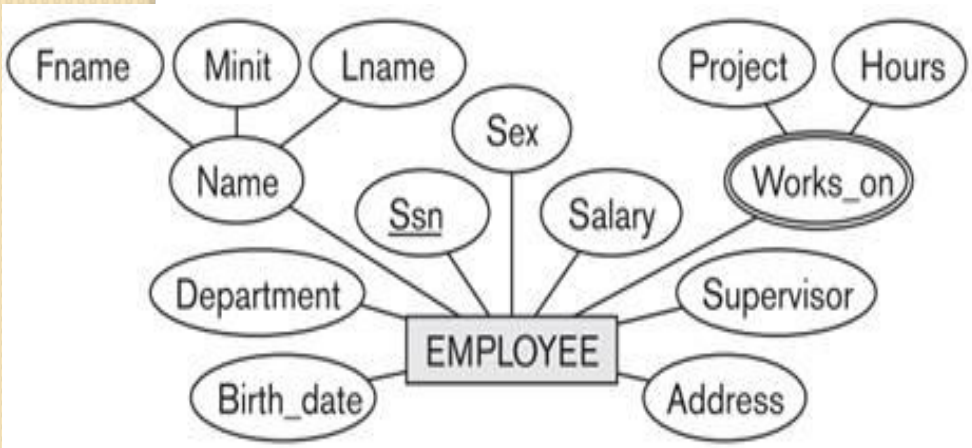


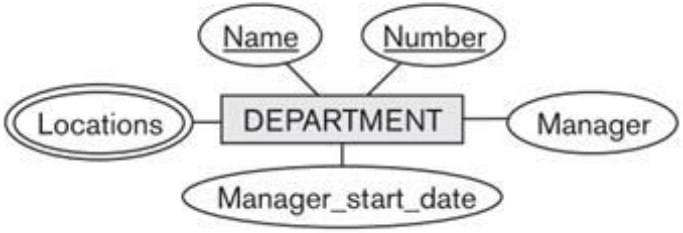
We want to keep track of the dependents of each employee for insurance purposes. We keep each dependent's first name, sex, birth date, and relationship to the employee

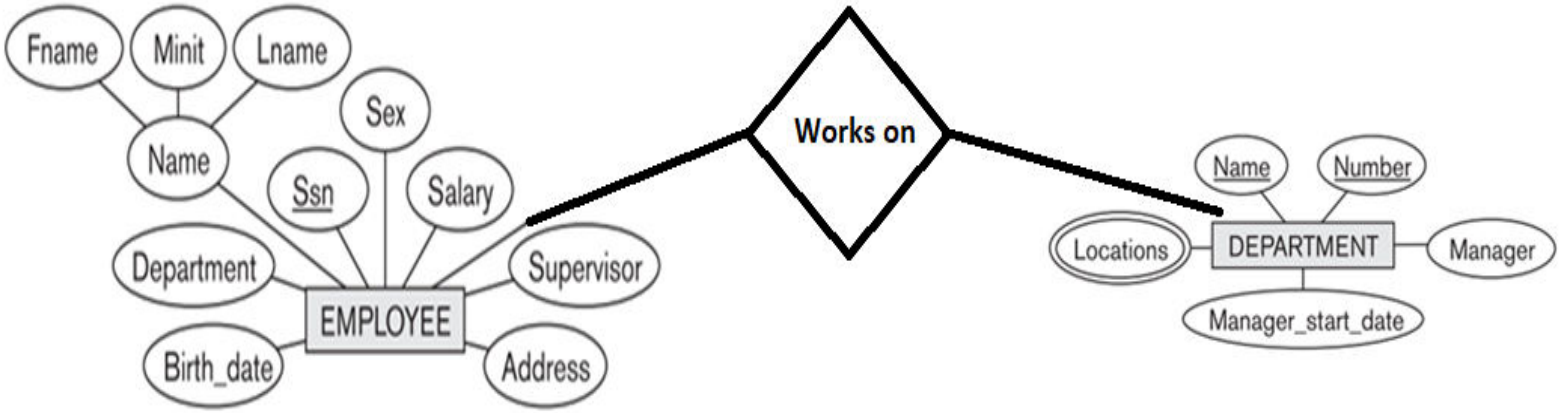


Relationship in DBMS

- Whenever an attribute of one entity type refers to another entity type, some relationship exists.
- the attribute Department of EMPLOYEE refers to the department for which the employee works






Fname
Minit
Lname
Name
Sex
Ssn
Salary
Department
Supervisor
EMPLOYEE
Birth_date
Address
Works on
Name
Number
Locations
DEPARTMENT
Manager
Manager_start_date

Database Management System

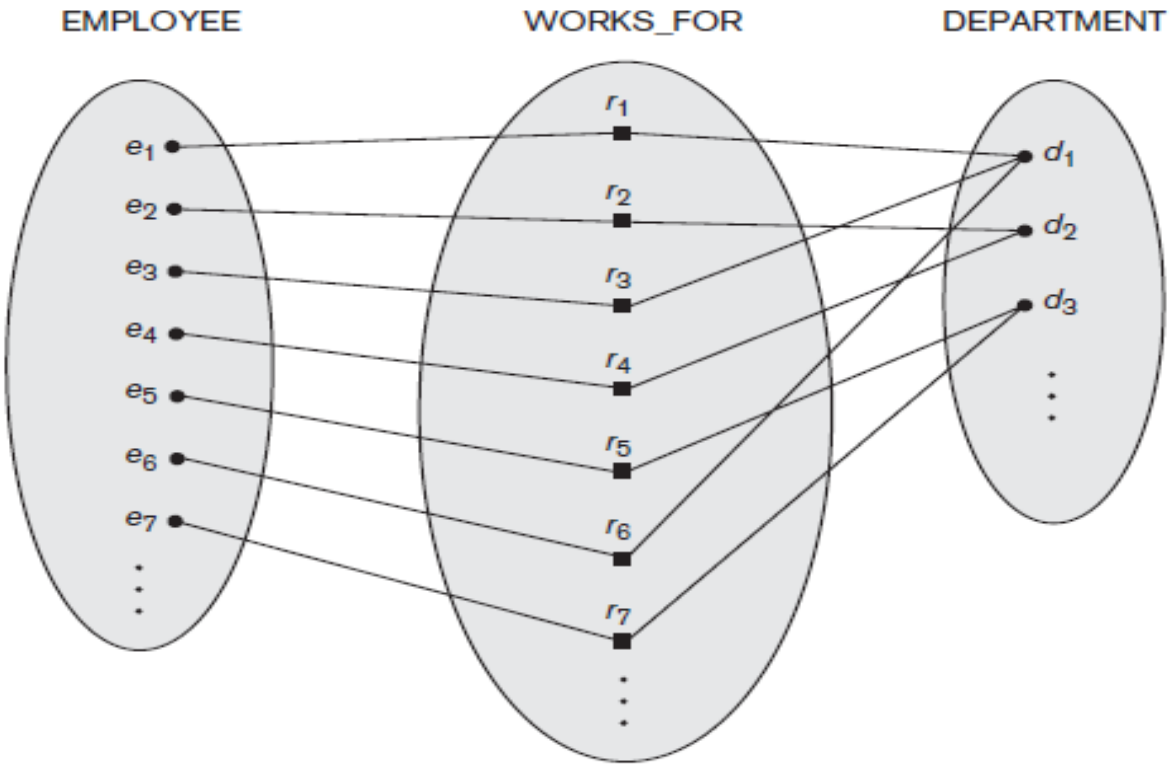
Module 1

Lect 10 :

E-R Diagram Contn.. - RELATIONSHIP

Relationship

- Relationship is the association among several entities.



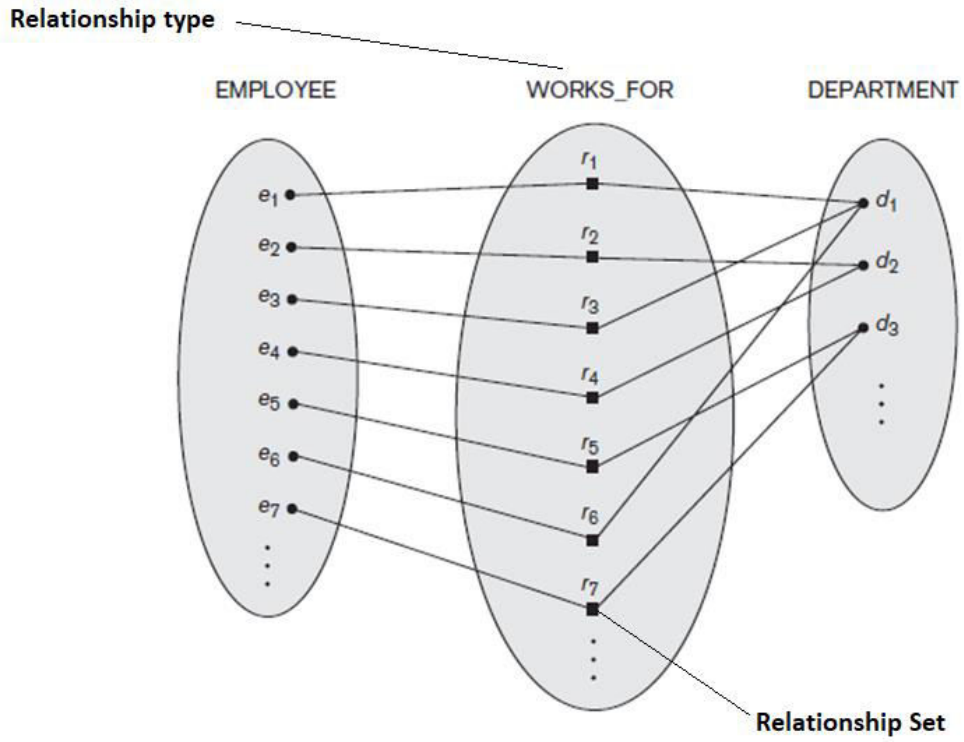
In ER diagram Relationship is represented in diamond

Eg1

Eg2

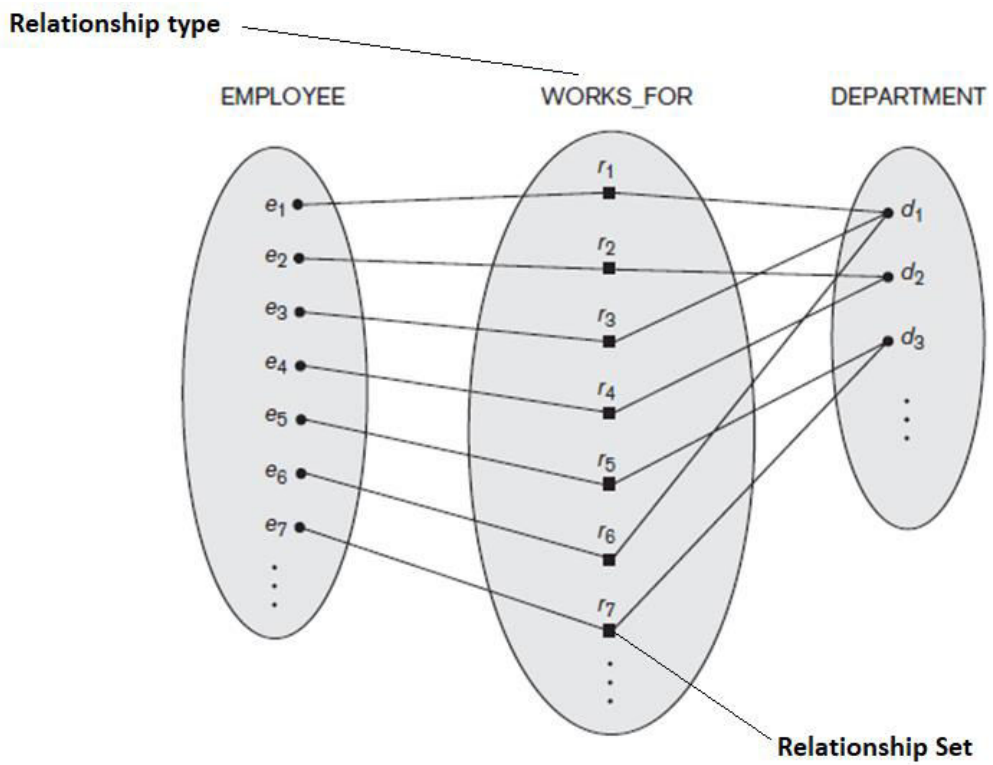
Relationship Type

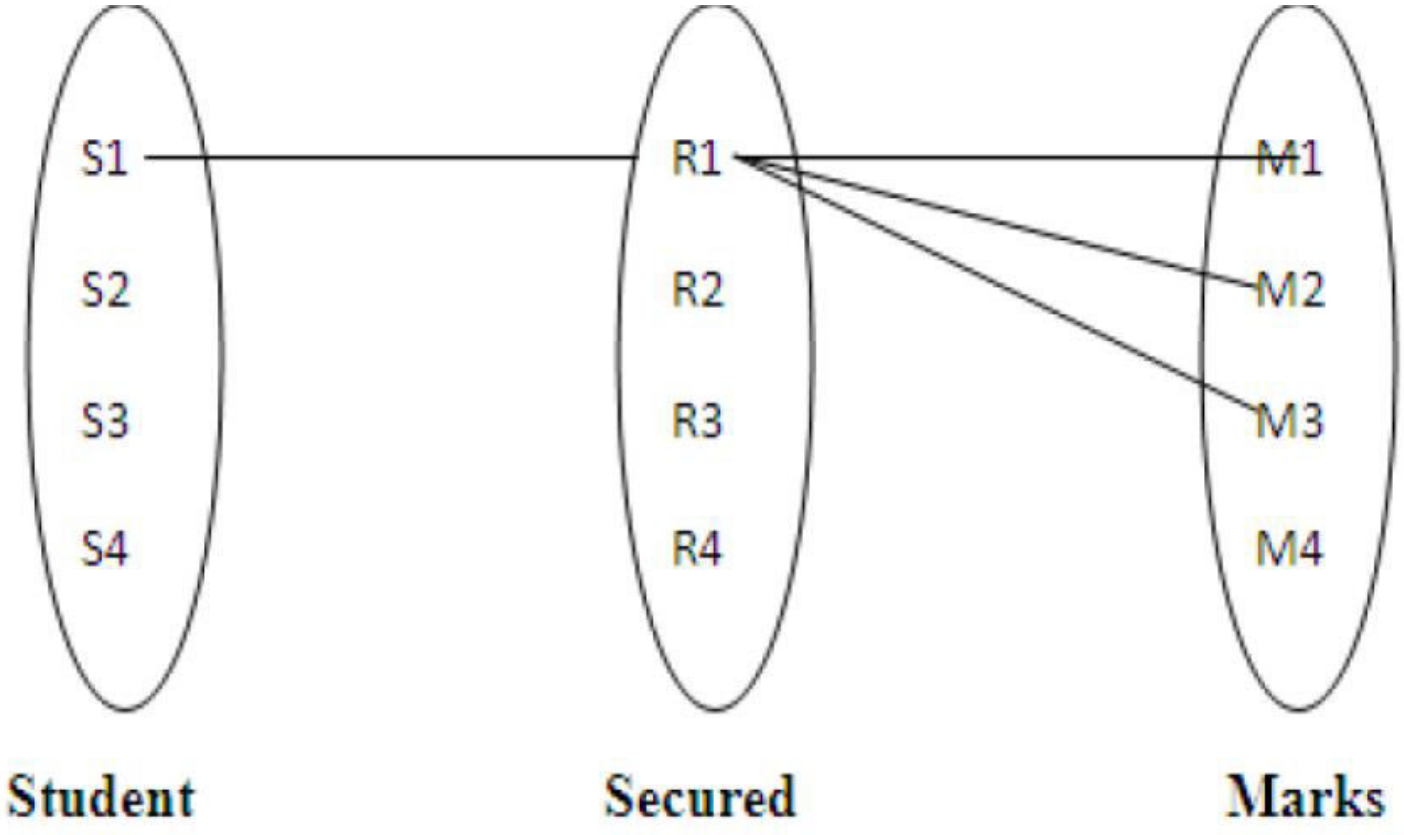
- The relationship which is associating with different entities



Relationship Set

- A relationship set is a set of relationships of the same type.





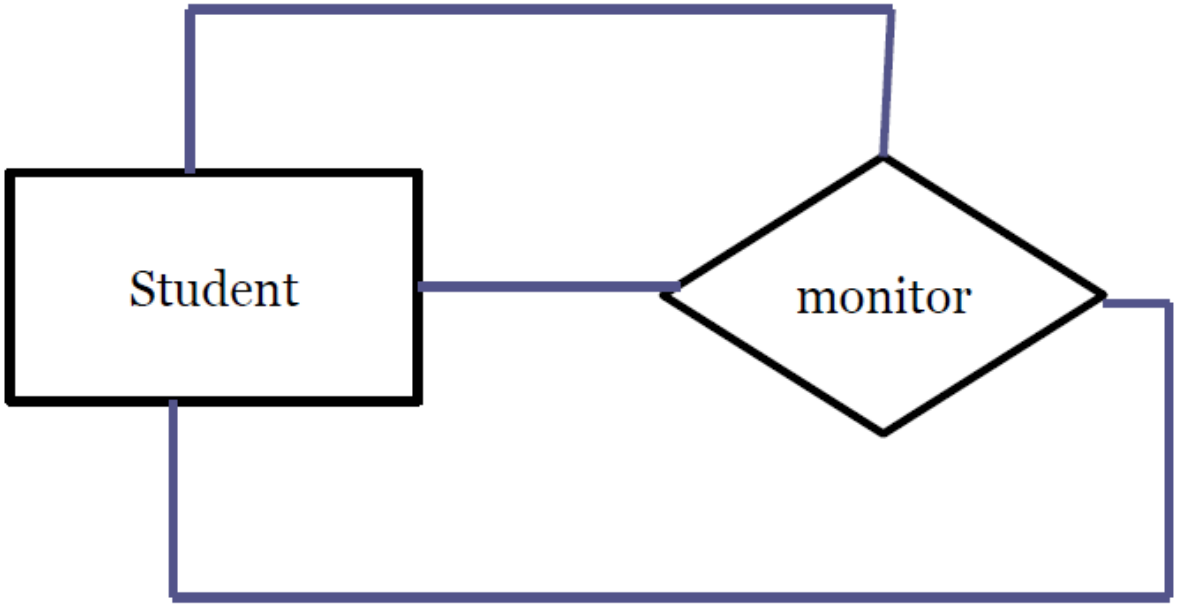
Relationship type: secured
Relationship set: {R1, R2, R3, R4}
Relationship instances: R1

Degree of a Relationship Type

- The degree of a relationship type is the number of participating entity types.
- 3 Types
  - Unary relationship
  - Binary relationship
  - Ternary Relationship

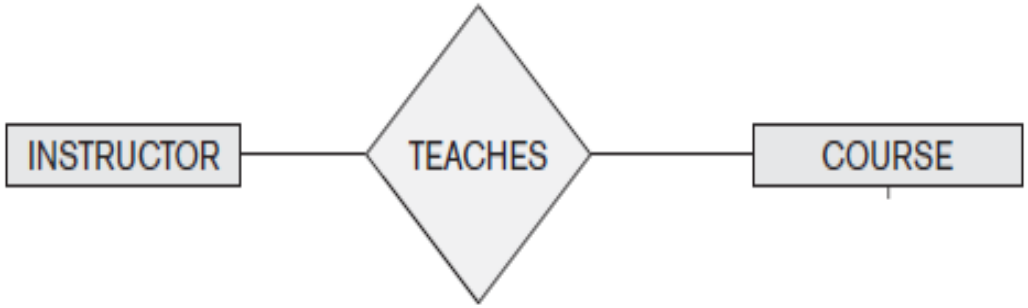
Unary relationship

- Association is maintained within a single entity.



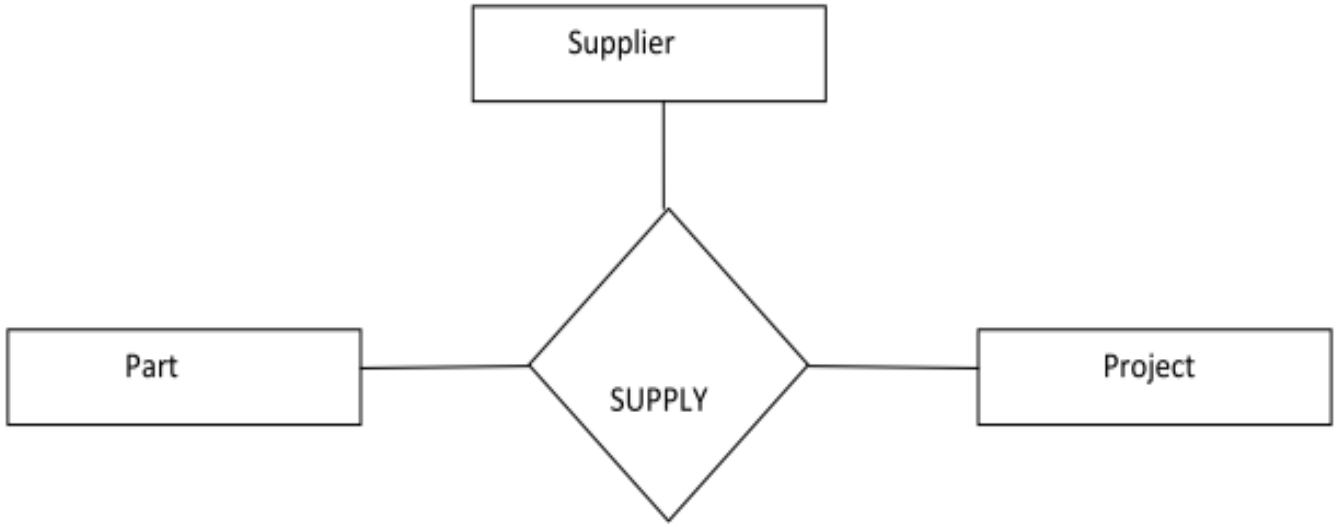
Binary relationship

- Association is maintained within two entities.



Ternary Relationship

Association is maintained within three entities



Constraints on Binary Relationship Types

- Relationship types have certain constraints that limit the possible combinations of entities that may participate in the corresponding relationship set.
- We can distinguish two main types of binary relationship constraints:
  - **cardinality ratio**
  - **Participation constraint**

Cardinality ratio

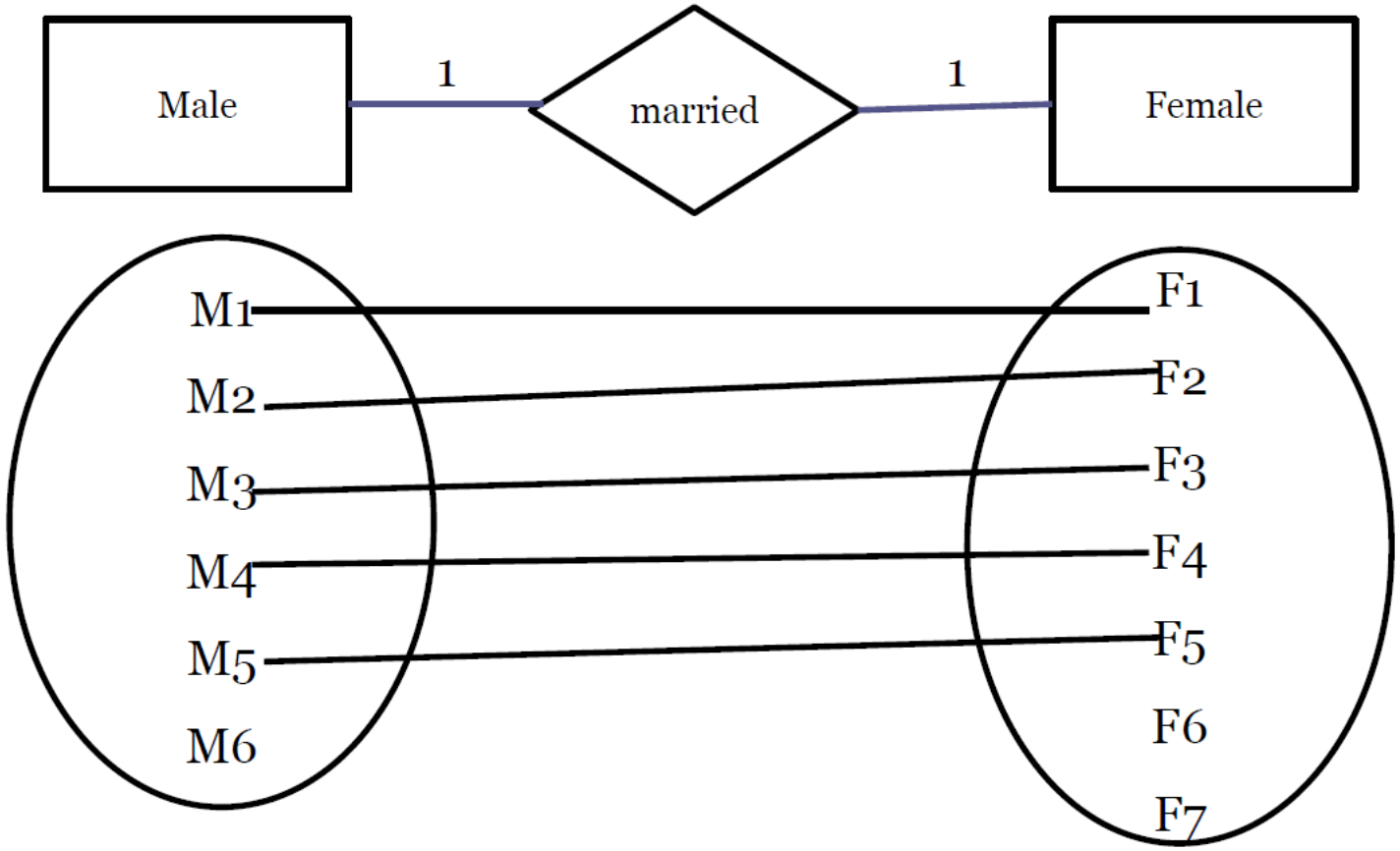
- The cardinality ratio for a binary relationship specifies the maximum number of relationship instances to which an entity can take part in it.

- The no of entities to which another entity can be associated via a relationship set
- The possible cardinality ratios for a binary relationship types are
  - One to one (1:1)
  - One to many (1:N)
  - Many to one (N:1)
  - Many to many (M:N)

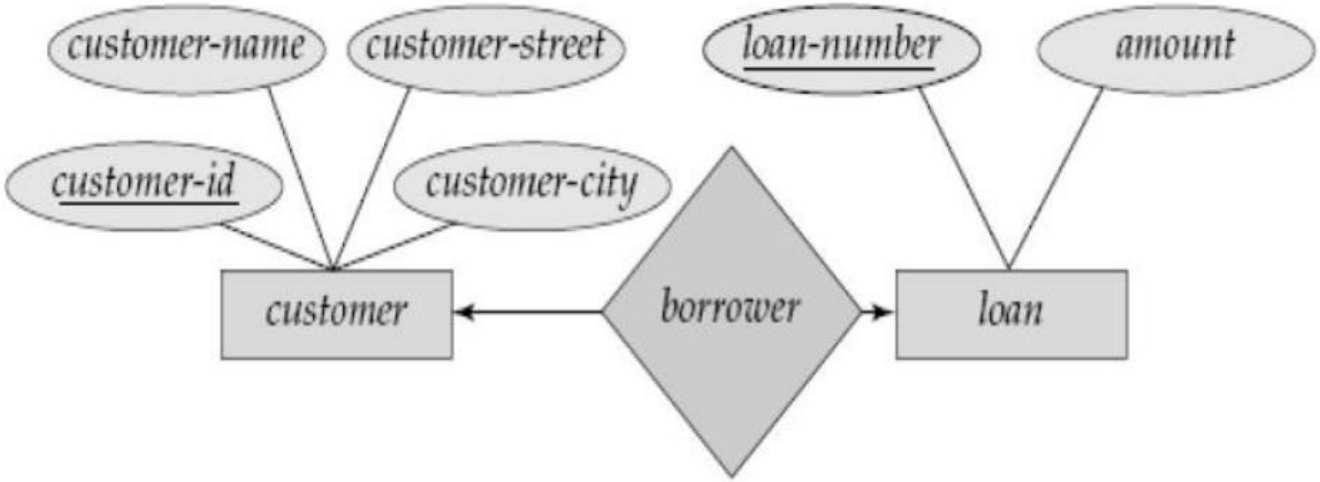
- We express cardinality ratio by drawing
- directed line (→), signifying “one,” or an
- undirected line (—), signifying “many,”

1. One to one:

- An entity A is associated with atmost one entity in B and an entity in B is associated with atmost one entity in A.

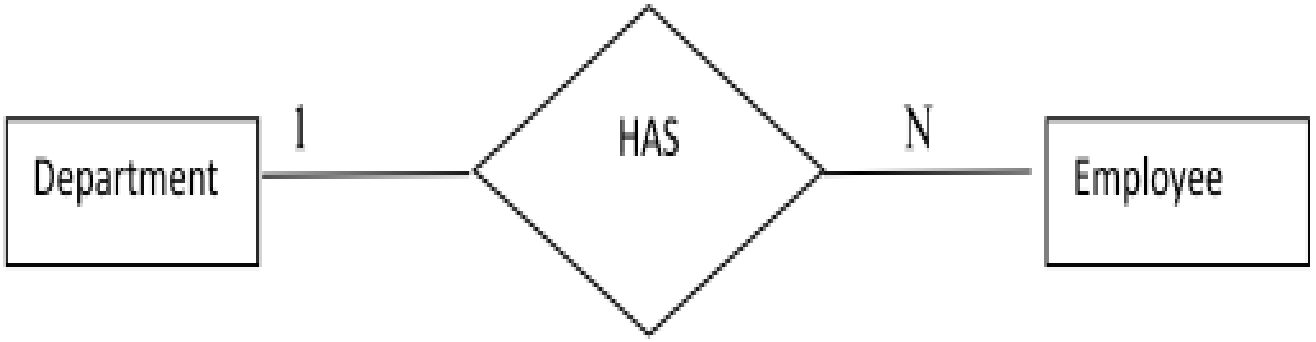

Male
1
married
1
Female
M1
M2
M3
M4
M5
M6
F1
F2
F3
F4
F5
F6
F7

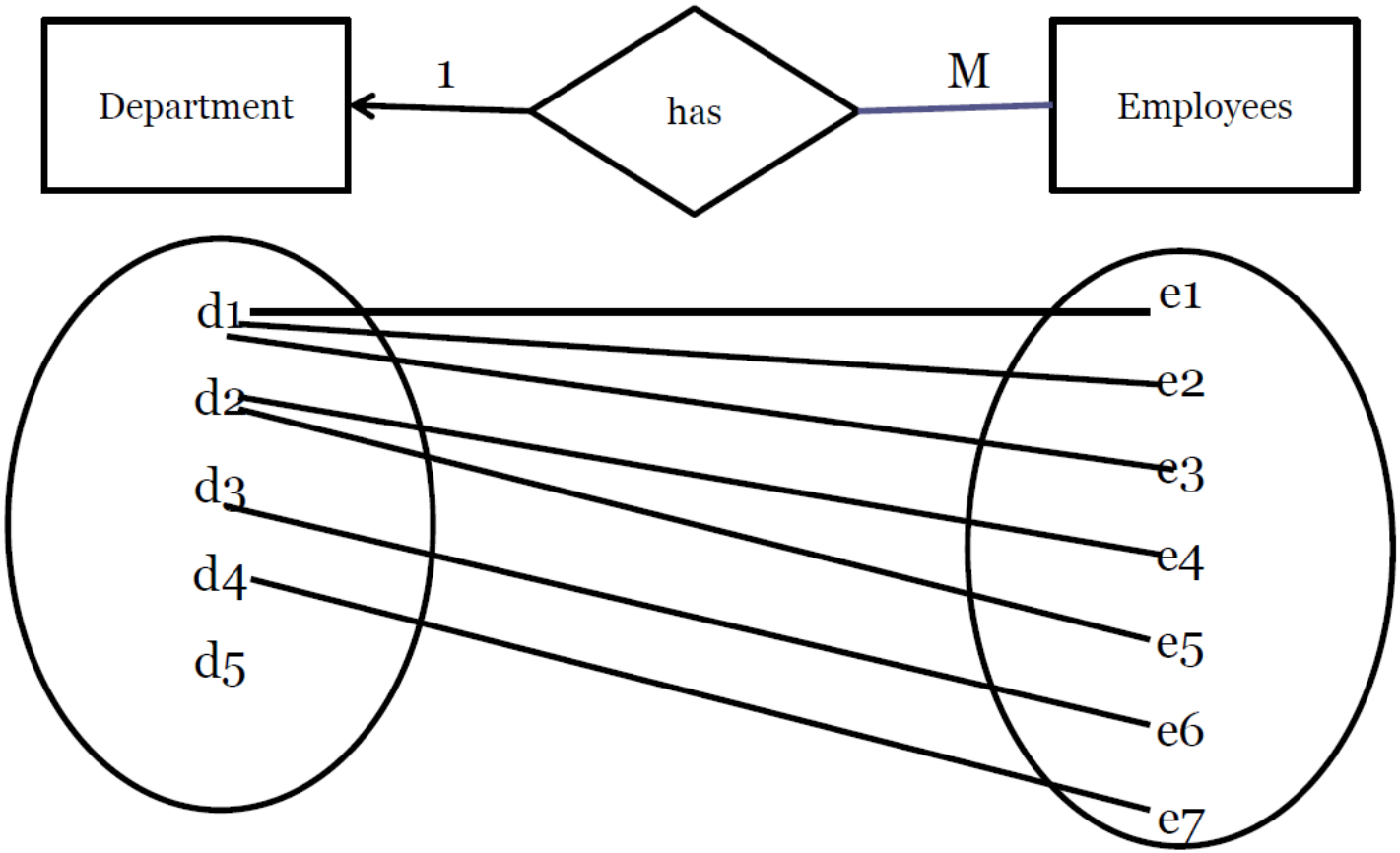
- A customer can borrow a single loan,
- A single loan can be borrowed by a customer



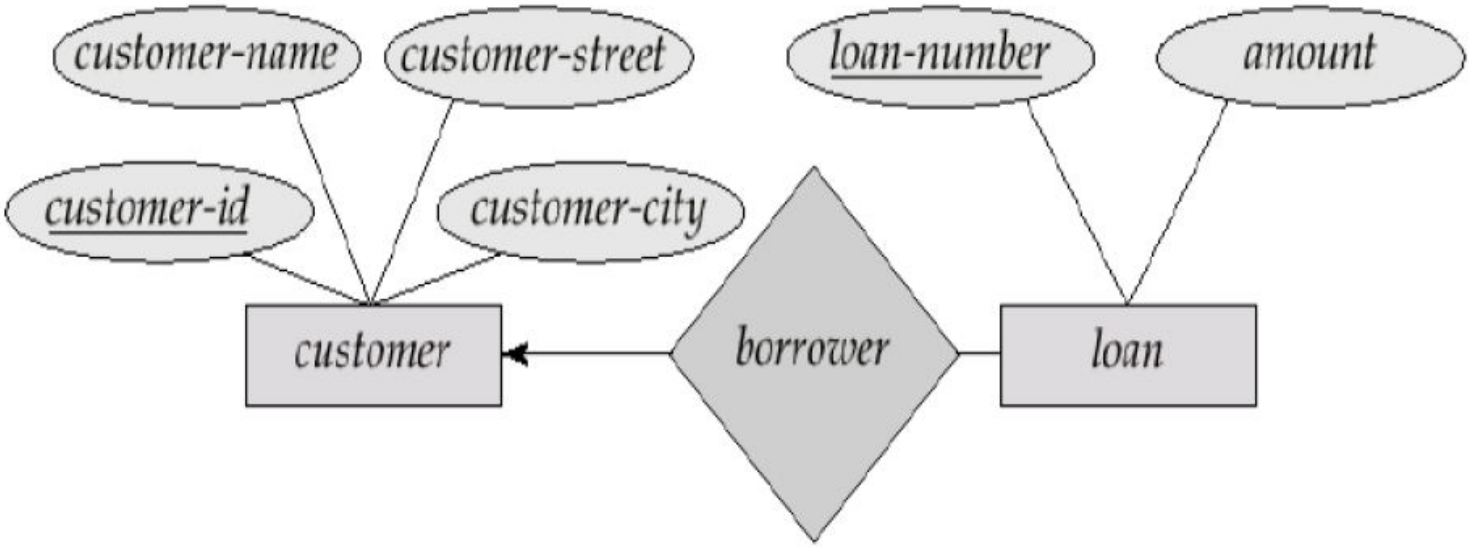
One to Many:

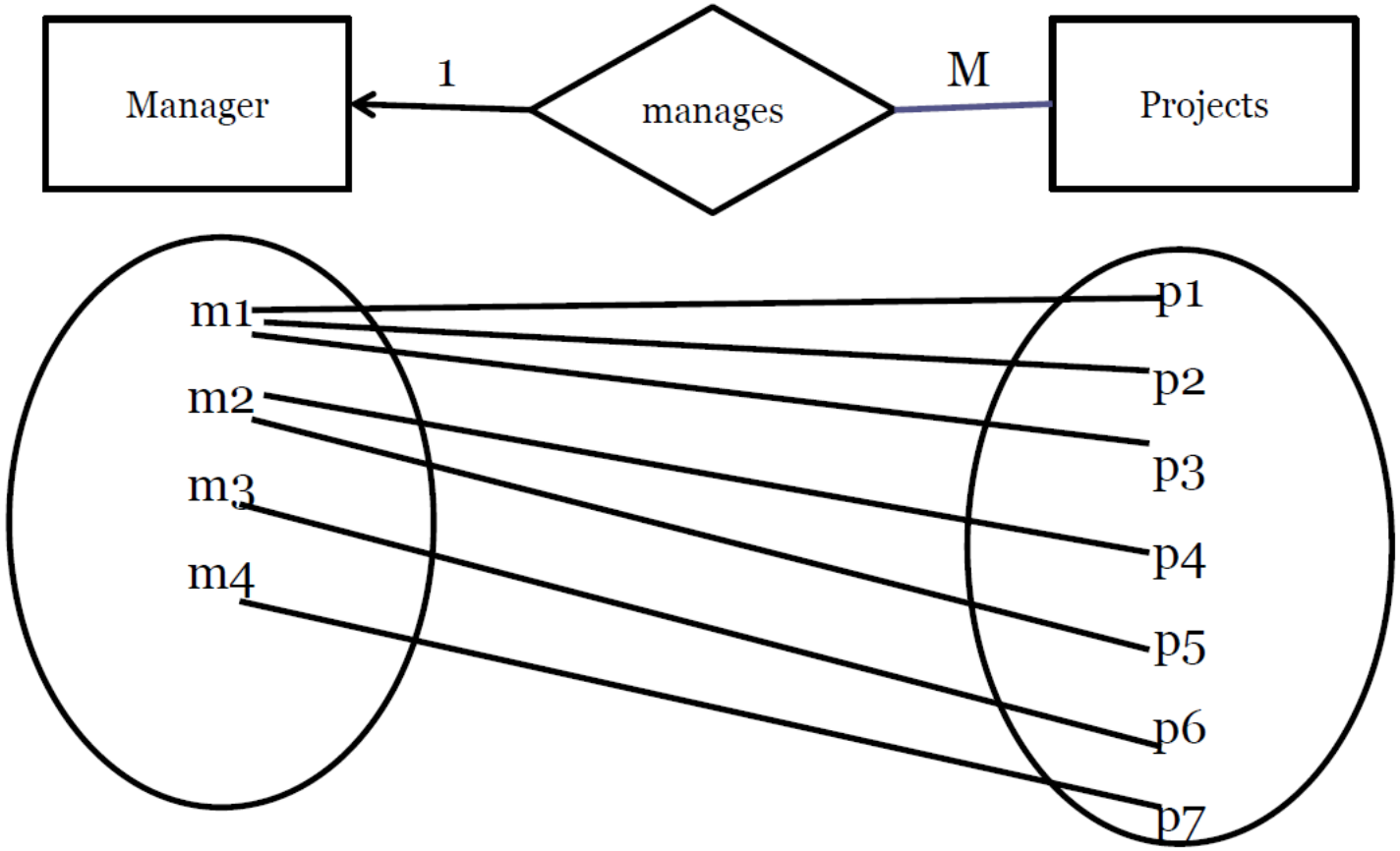
- An entity A is associated with any no of entities in B and an entity in B can be associated with atmost one entity in A.




Department
1
has
M
Employees
d1
d2
d3
d4
d5
e1
e2
e3
e4
e5
e6
e7

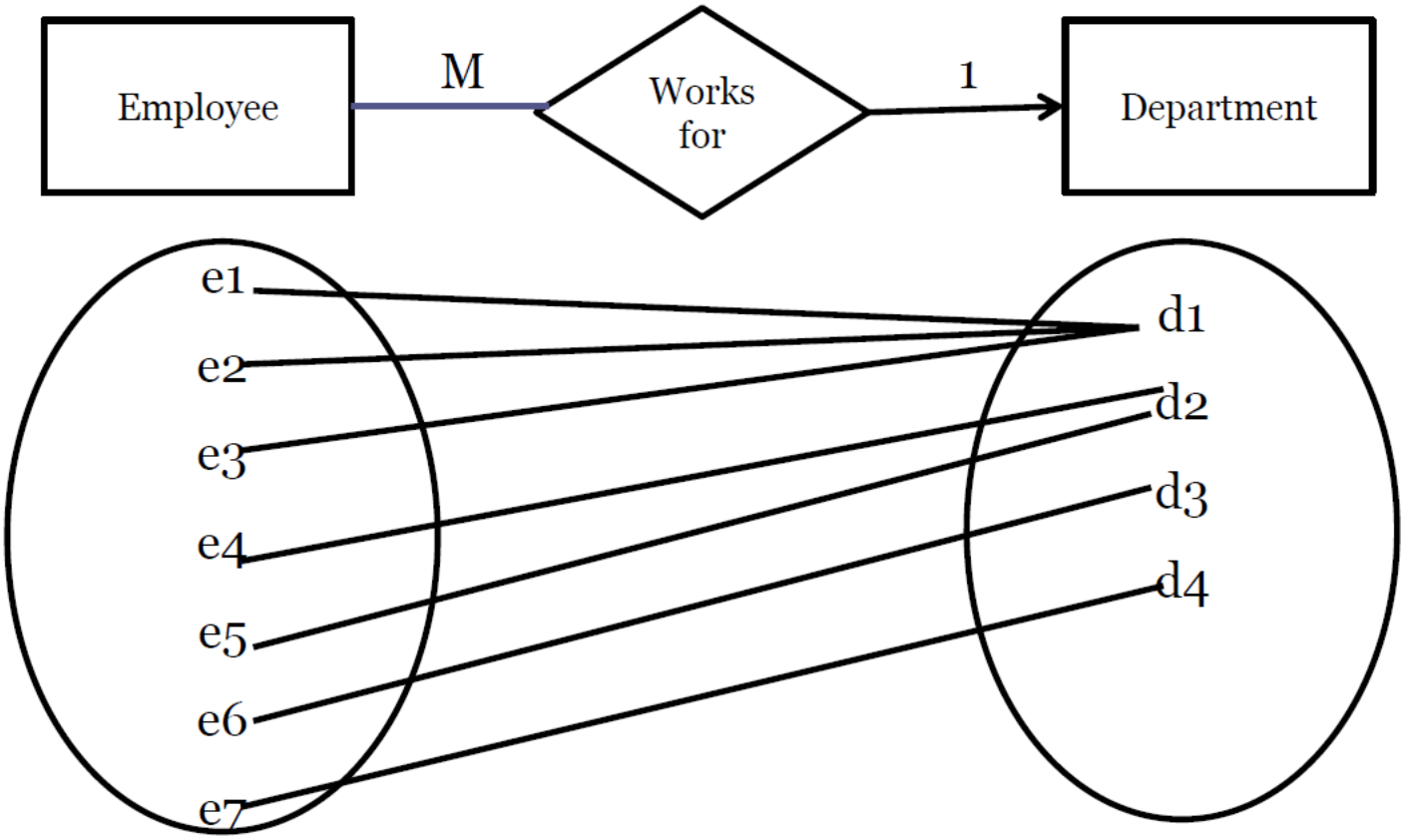
- In the one-to-many relationship a customer is associated with several loans via *borrower*



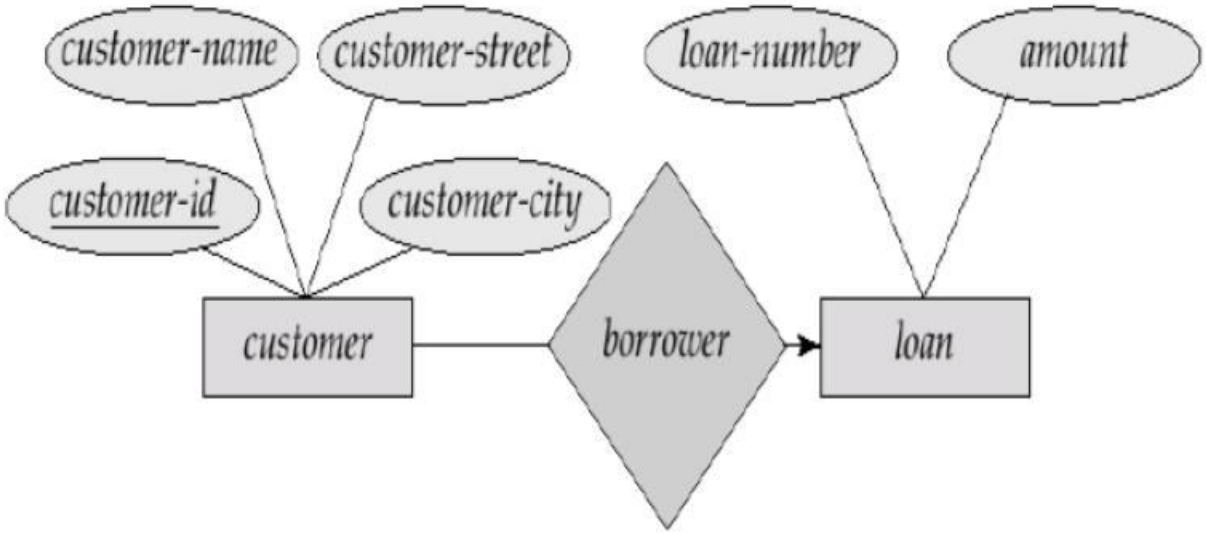

Manager
1
manages
M
Projects
m1
m2
m3
m4
p1
p2
p3
p4
p5
p6
p7

Many to one:

- An entity in A is associated with atmost one entity in B. An entity in B can be associated with any no of entities in A.

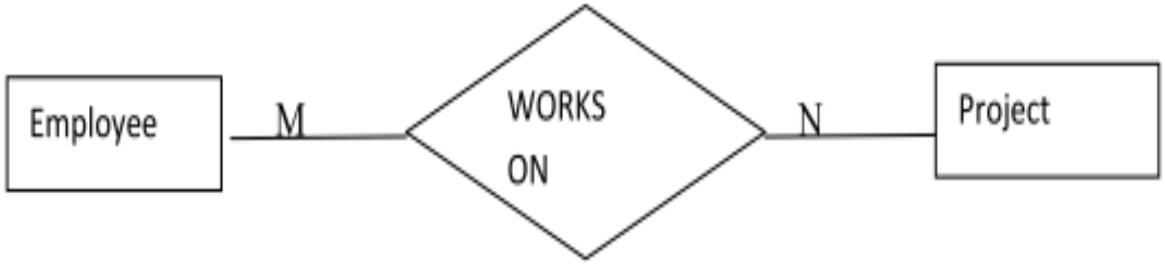

Employee
M
Works for
1
Department
e1
e2
e3
e4
e5
e6
e7
d1
d2
d3
d4

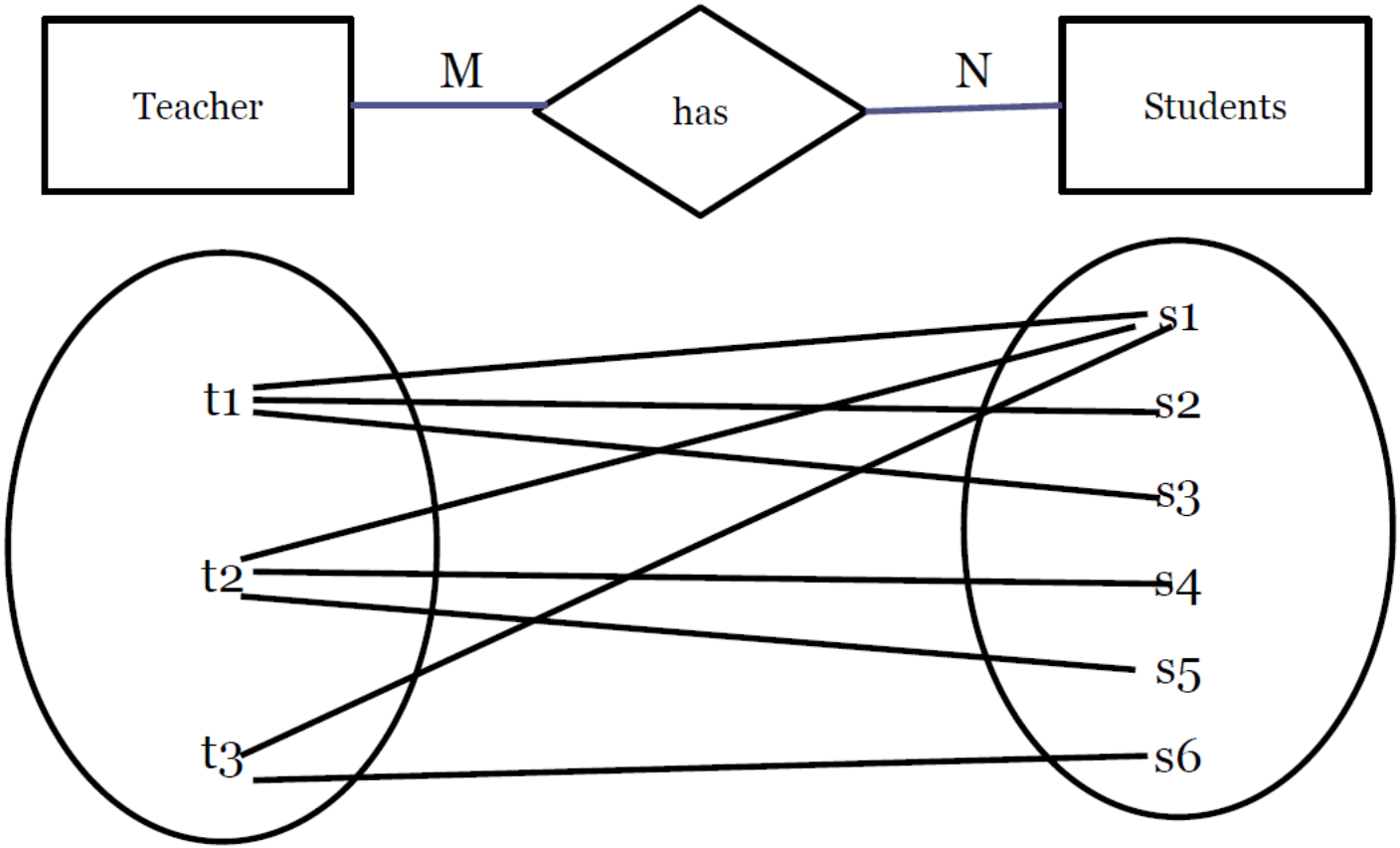
- In a many-to-one relationship a loan is associated with several customers via *borrower*.

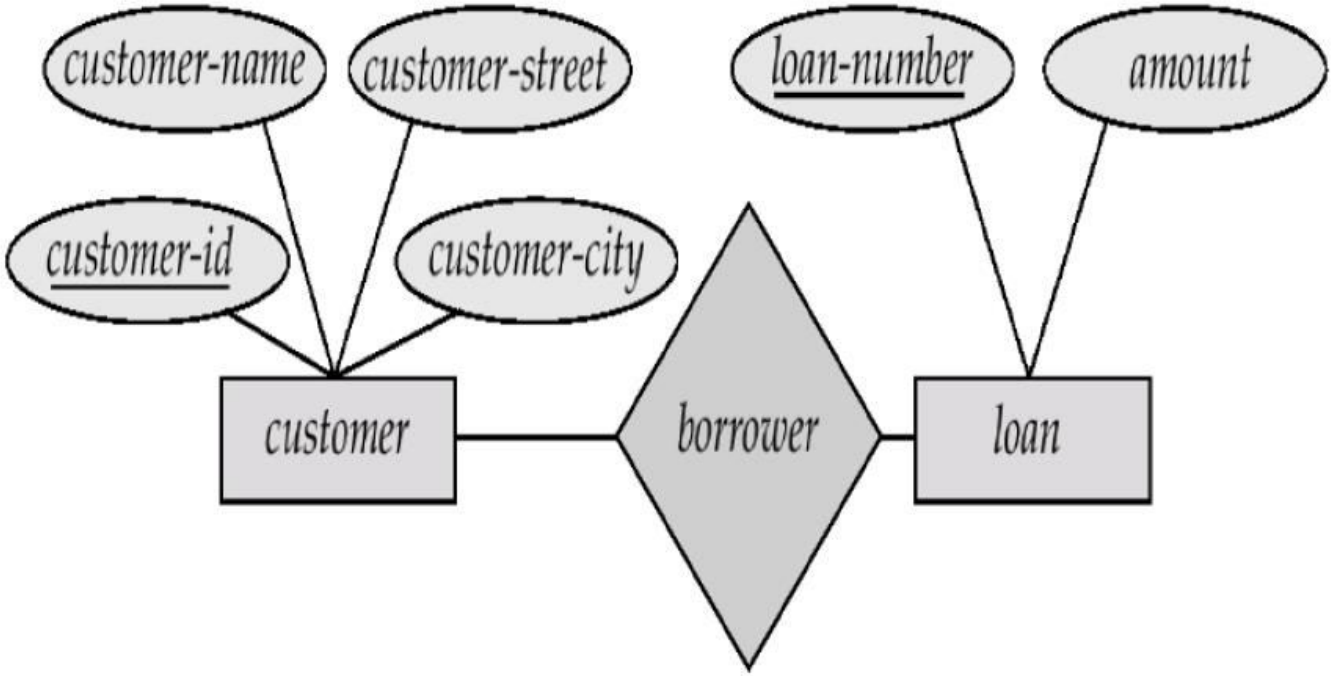


Many to Many:

- An entity in A is associated with any no of entities in B and an entity in B is associated with any no of entities in A.




Teacher
M
has
N
Students
t1
t2
t3
s1
s2
s3
s4
s5
s6


customer-name
customer-street
loan-number
amount
customer-id
customer-city
customer
borrower
loan

- Database Management System

Module 1

Lect 11 :

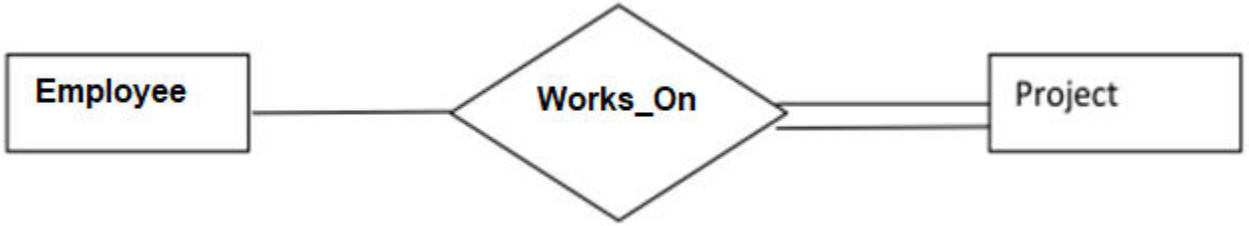
E-R Diagram Contn.. -

-> Participation Constraint

-> Weak Entity Type

PARTICIPATION CONSTRAINTS

- This constraint specifies the minimum number of relationship instances that each entity can participate in,
- Also called the minimum cardinality constraint.
- There are two types of participation constraints—total and partial.

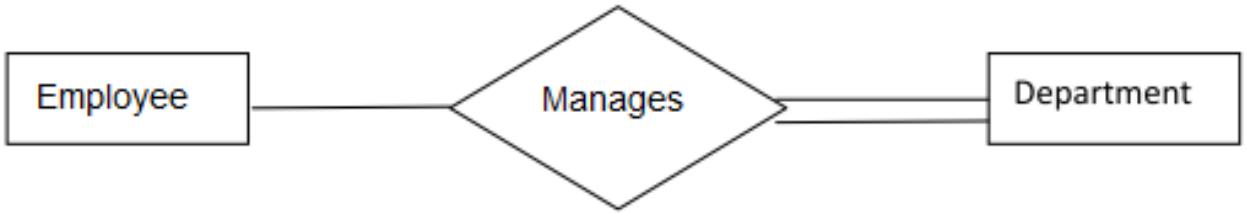


- **Total Participation** :
  - If every entity in the entity type participates in at least one relationship in the relationship type
  - Represented by double lines in ER Diagram
- **Partial Participation :**
  - Some entities may not participate in any relationship in the relationship type
  - Represented by single line in ER Diagram

- Eg1

- Entity Project should have atlease one department, So Project has **Total Participation**
- There can be no Department which is assigned any project , So Department has **Partial Participation**
- In ER diagrams, total participation is displayed as a double line connecting the participating entity type to the relationship, whereas partial participation is represented by a single line.

- Eg2



  - Entity Department should have atlease one Employee as Manager, So Department has **Total Participation**
  - Not all Employee Manages a department. So Employee has **Partial Participation**

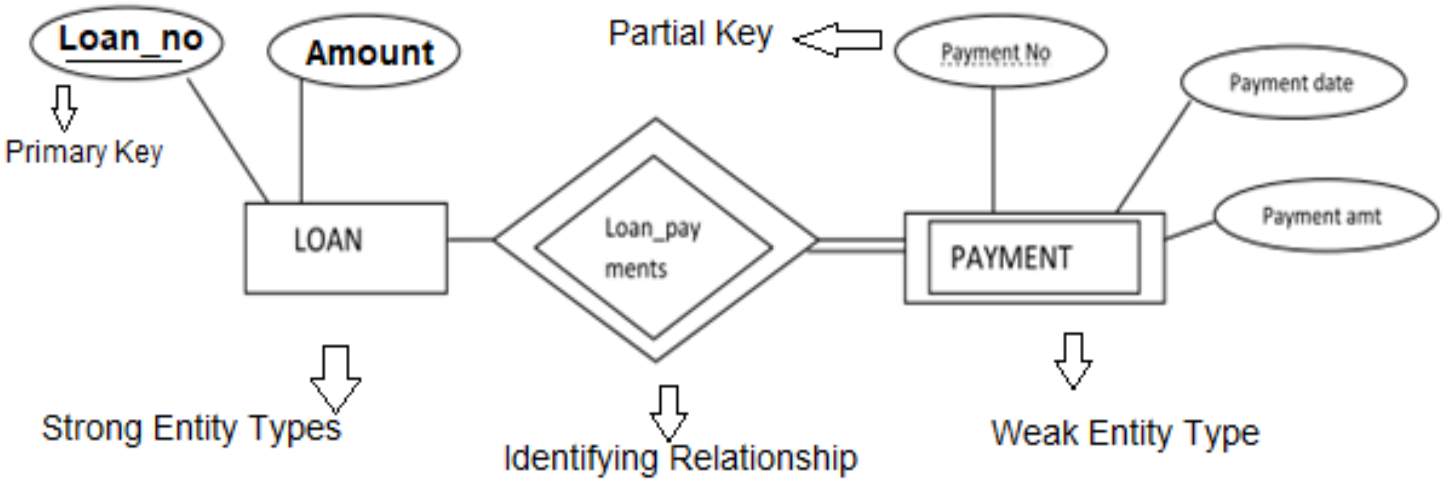
TYPES OF ENTITY TYPES

- **Strong entity type**
  - Entity types that have at least one key attribute.
  - A strong entity is not dependent of any other entity in the schema.
  - A strong entity will always have a primary key.
  - Strong entities are represented by a single rectangle.
  - The relationship of two strong entities is represented by a single diamond.
  - Various strong entities, when combined together, create a strong entity set.

- **Weak entity type**
  - Entity type that does not have any key attribute.
  - A weak entity is dependent on a strong entity to ensure the its existence.
  - Unlike a strong entity, a weak entity does not have any primary key.
  - It instead has a partial key.
  - A weak entity is represented by a double rectangle. The relation between one strong and one weak entity is represented by a **double diamond.**

Identifying Relationship

- It links the strong and weak entity and is represented by a double diamond sign.

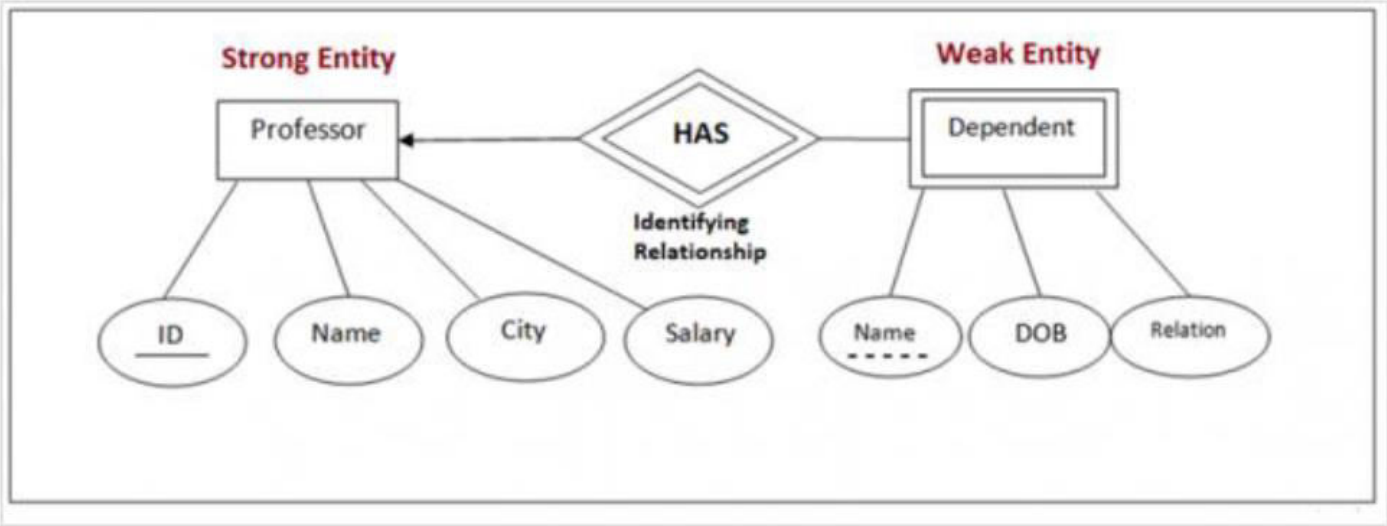


Loan_no	Amount
L1	1,00,000
L2	2,00,000
L3	1,50,000

Payment_no	Payment_date	Payment Amt
P101	26-May	10,000
P101	26-May	10,000
P101	26-May	10,000

Loan_no	Payment_no	Payment_date	Payment Amt
L1	P101	26-May	10,000
L2	P101	26-May	10,000
L3	P101	26-May	10,000

Payment Entity type can exist only by combining the Primary key of Loan Entity Type & Partial key of Payment Entity



Above we saw that, Dependent Name could not exist on its own but in relationship to a Professor.

Professor	Strong Entity
Dependent	Weak Entity
Partial Key (Weak Entity)	Name
Primary Key (Strong Entity)	ID

- Database Management System

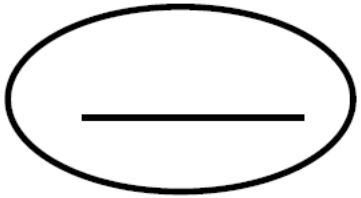
Module 1

Lect 12 :

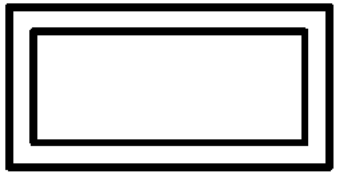
E-R Diagram Construction

NOTATIONS USED IN E-R DIAGRAM

Entity

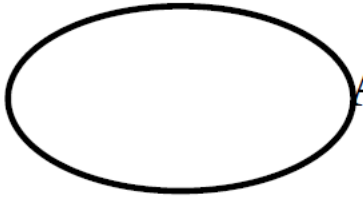


Key attribute

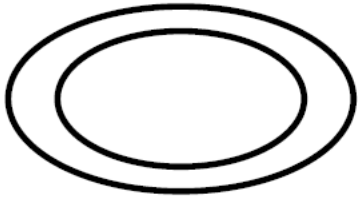


Weak Entity

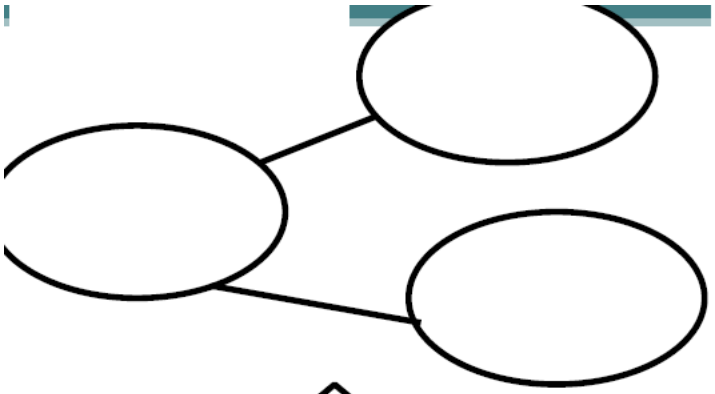
Derived attribute



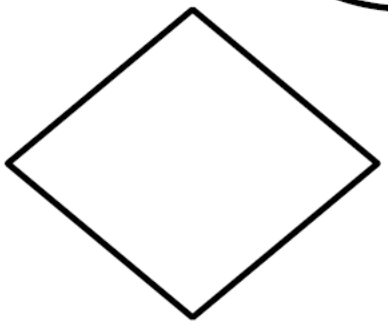
Attribute



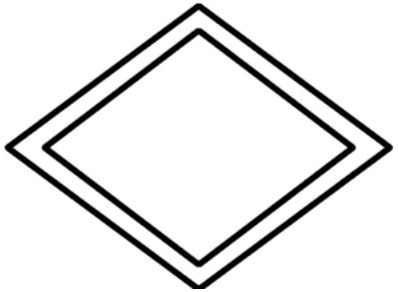
Multivalued Attribute



Composite Attribute



Relationship type



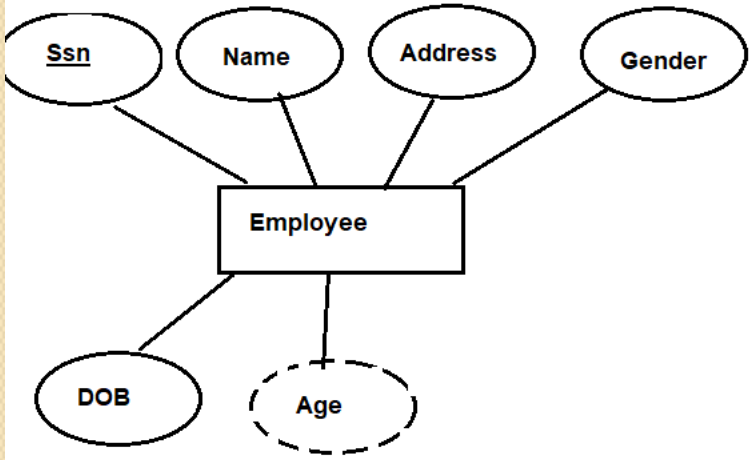
Identifying Relationship

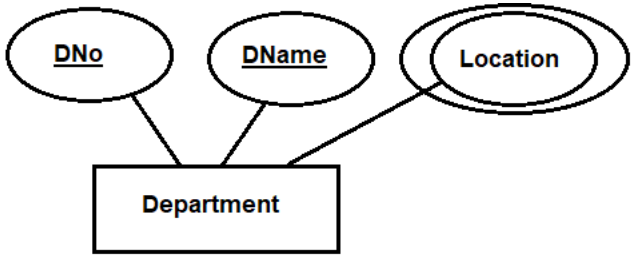
Example 1

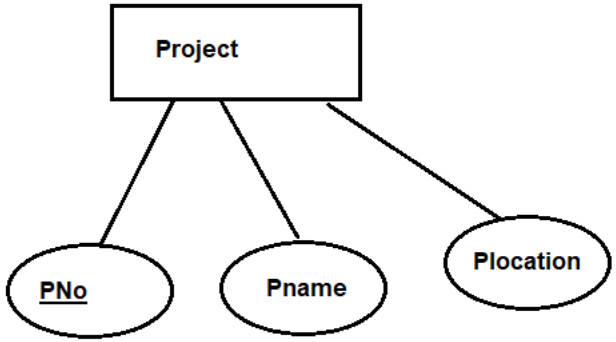
The company database keeps track of company's employee, department and projects We store Employee's name, ssn,address,salary,gender,date of birth, age. An employee is assigned to one department, but may work on several projects which are not necessarily controlled by the same department. A particular employee manages the department. Each department has a unique name,unique number and several locations. The department controls number of projects each of which has a unique name,unique number and a single location. We want to keep track of the dependents of each employee for insurance purpose. We keep each dependent's first name, sex,birth date and relationship to employee

The company database keeps track of company's employee, department and projects. We store Employee's name, ssn, address, salary, gender, date of birth, age. An employee is assigned to one department, but may work on several projects which are not necessarily controlled by the same department. A particular employee manages the department. Each department has a unique name,unique number and several locations. The department controls number of projects each of which has a unique name,unique number and a single location. We want to keep track of the dependents of each employee for insurance purpose. We keep each dependent's first name,sex,birth date and relationship to employee

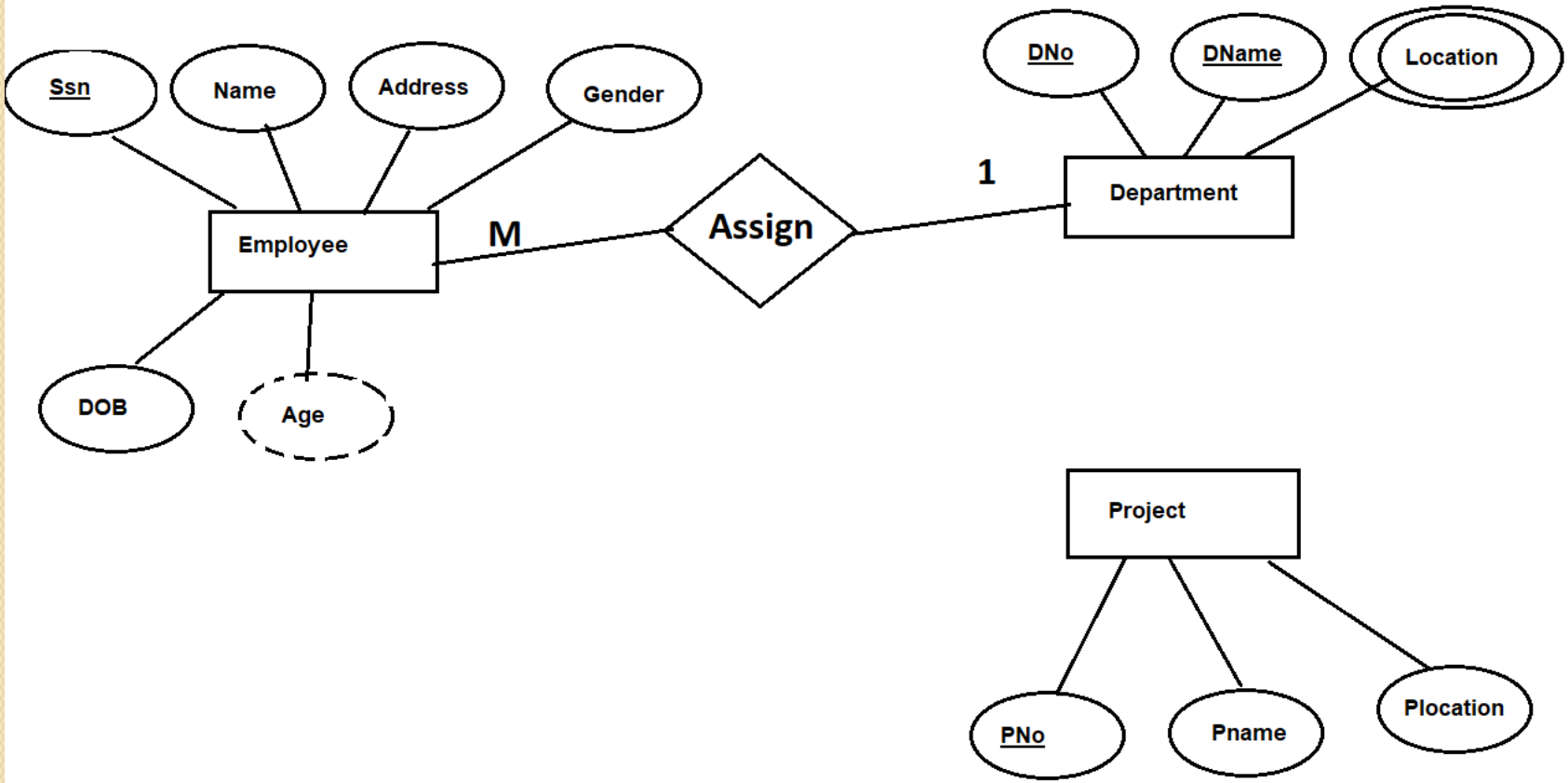
The company database keeps track of company's employee, department and projects. We store Employee's name,ssn,address,salary, gender,date of birth, age. An employee is assigned to one department, but may work on several projects which are not necessarily controlled by the same department. A particular employee manages the department. Each department has a unique name,unique number and several locations. The department controls number of projects each of which has a unique name, unique number and a single location. We want to keep track of the dependents of each employee for insurance purpose. We keep each dependent's first name, sex,birth date and relationship to employee



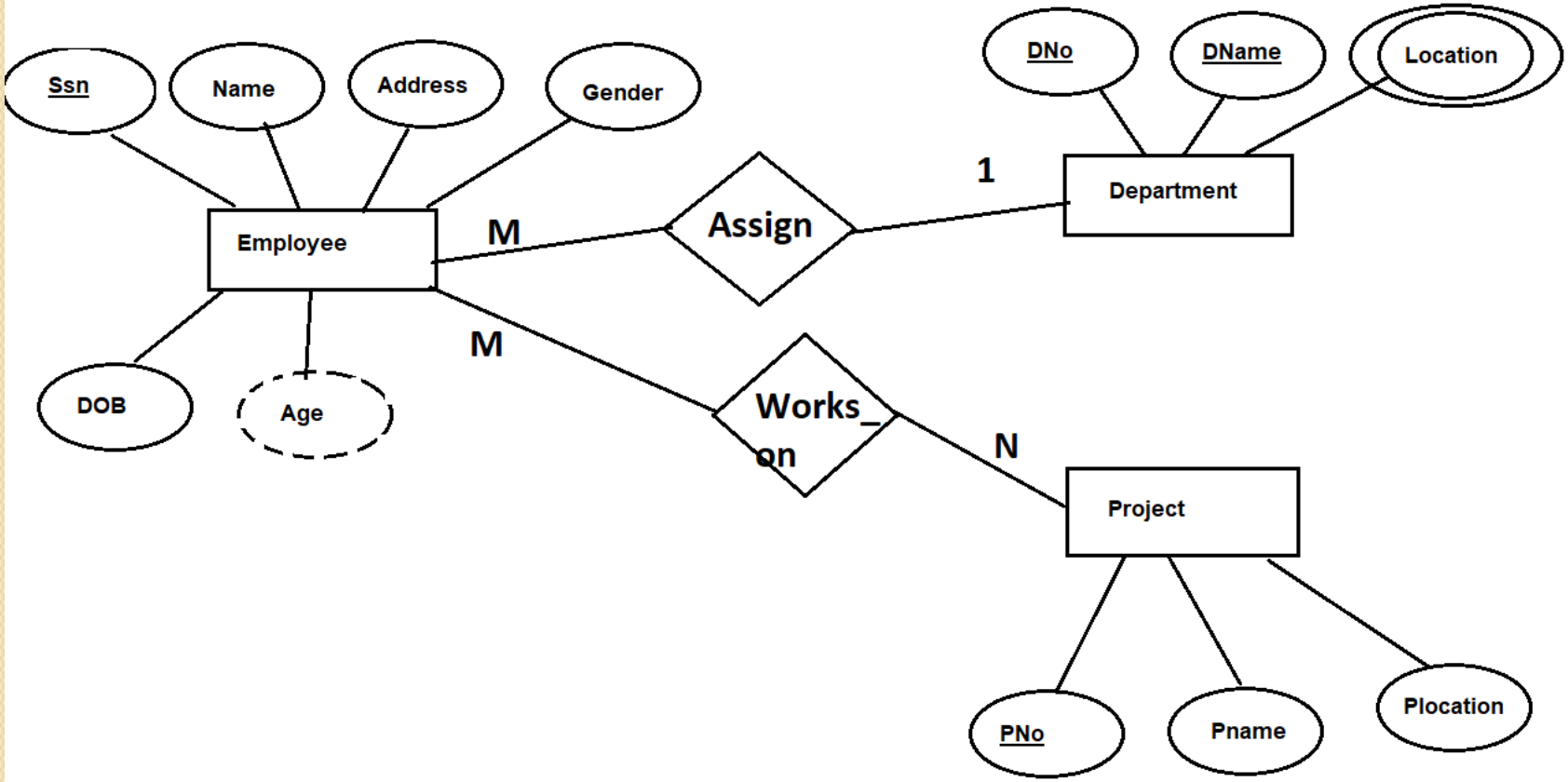




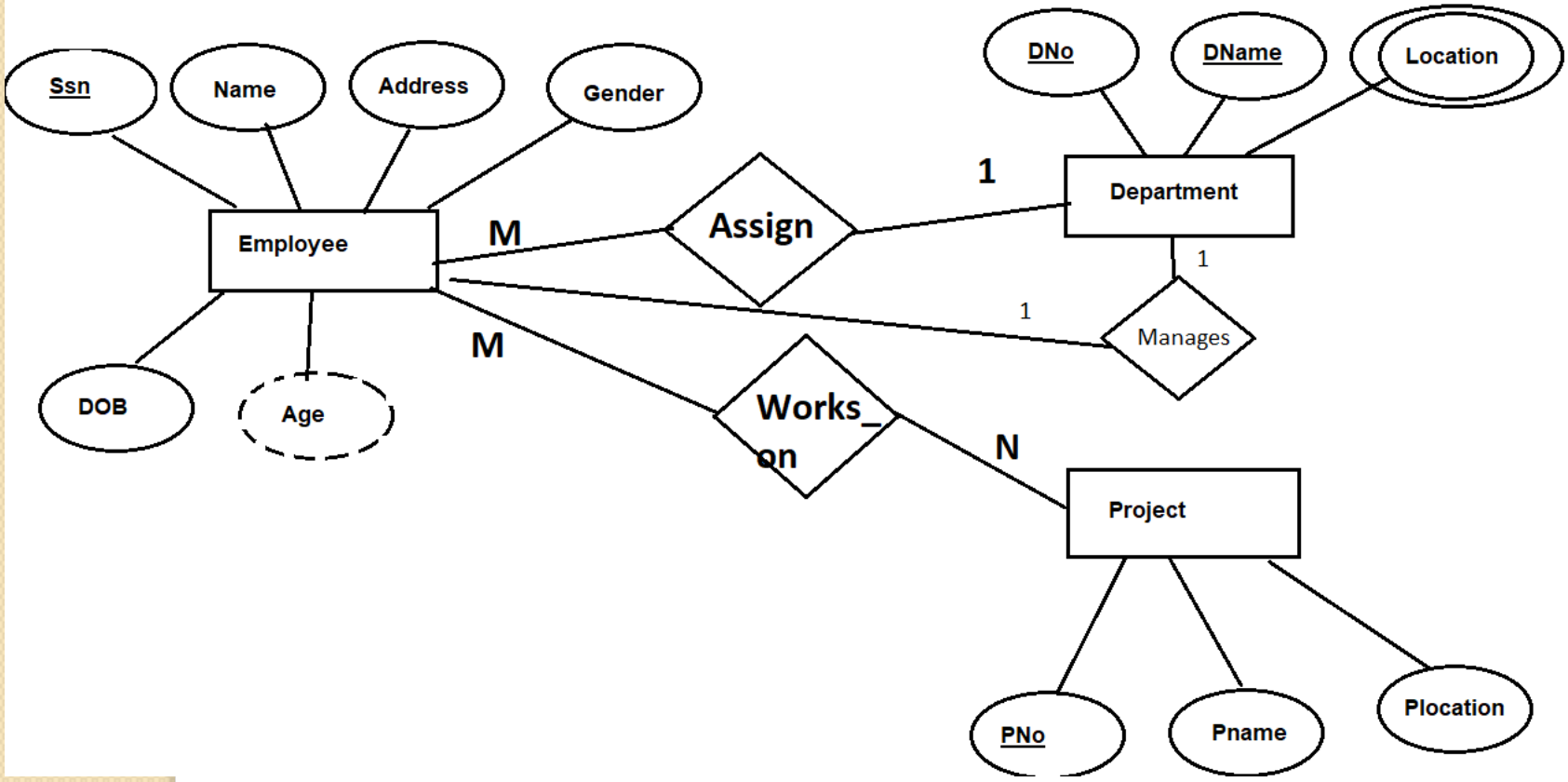
The company database keeps track of company's employee, department and projects. We store Employee's name,ssn,address,salary, gender,date of birth, age. An employee is assigned to one department, but may work on several projects which are not necessarily controlled by the same department. A particular employee manages the department. Each department has a unique name,unique number and several locations. The department controls number of projects each of which has a unique name, unique number and a single location. We want to keep track of the dependents of each employee for insurance purpose. We keep each dependent's first name, sex,birth date and relationship to employee



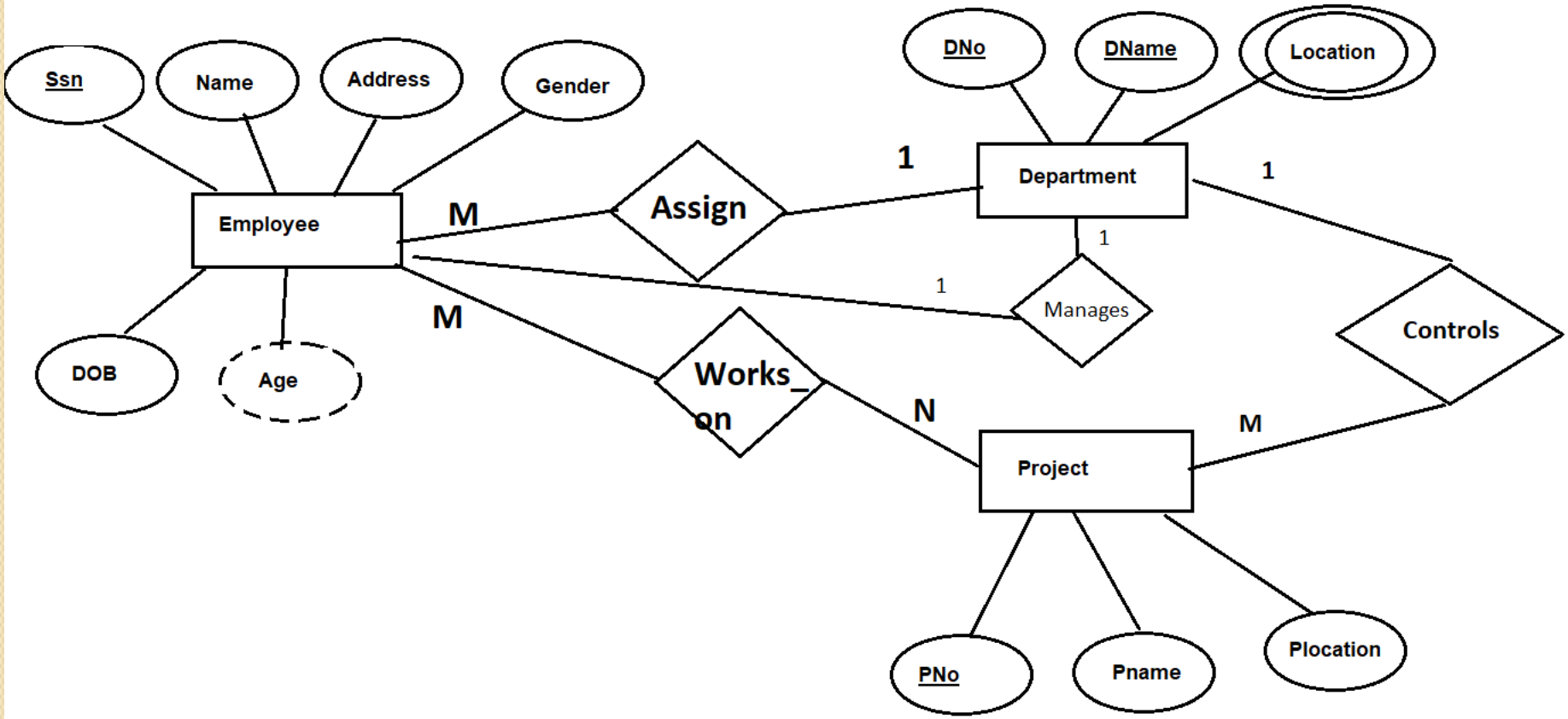
The company database keeps track of company's employee, department and projects. We store Employee's name,ssn,address,salary, gender,date of birth, age. An employee is assigned to one department, but may work on several projects which are not necessarily controlled by the same department. A particular employee manages the department. Each department has a unique name,unique number and several locations. The department controls number of projects each of which has a unique name, unique number and a single location. We want to keep track of the dependents of each employee for insurance purpose. We keep each dependent's first name, sex,birth date and relationship to employee



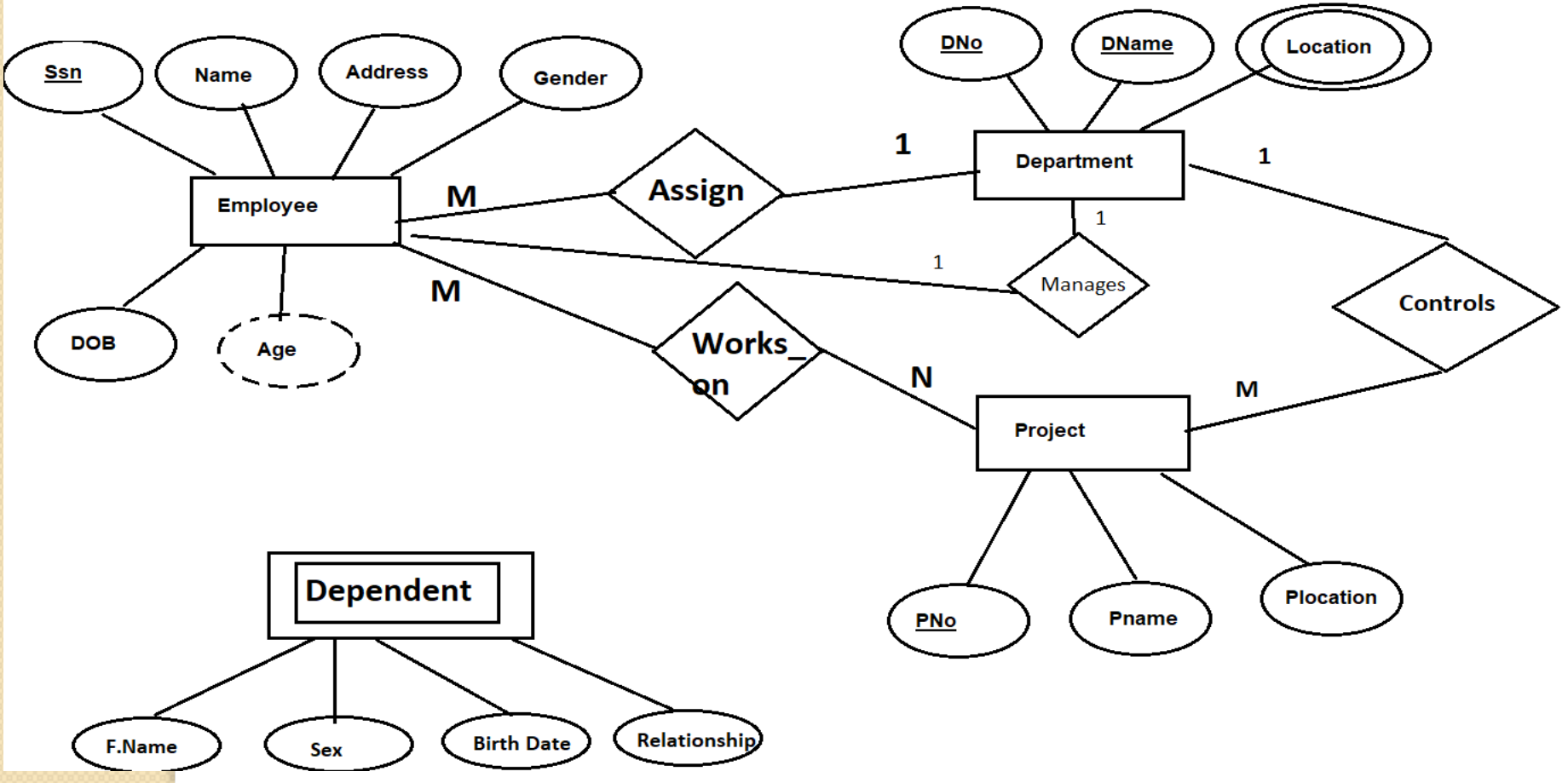
The company database keeps track of company's employee, department and projects. We store Employee's name,ssn,address,salary, gender,date of birth, age. An employee is assigned to one department, but may work on several projects which are not necessarily controlled by the same department. A particular employee manages the department. Each department has a unique name,unique number and several locations. The department controls number of projects each of which has a unique name, unique number and a single location. We want to keep track of the dependents of each employee for insurance purpose. We keep each dependent's first name, sex,birth date and relationship to employee



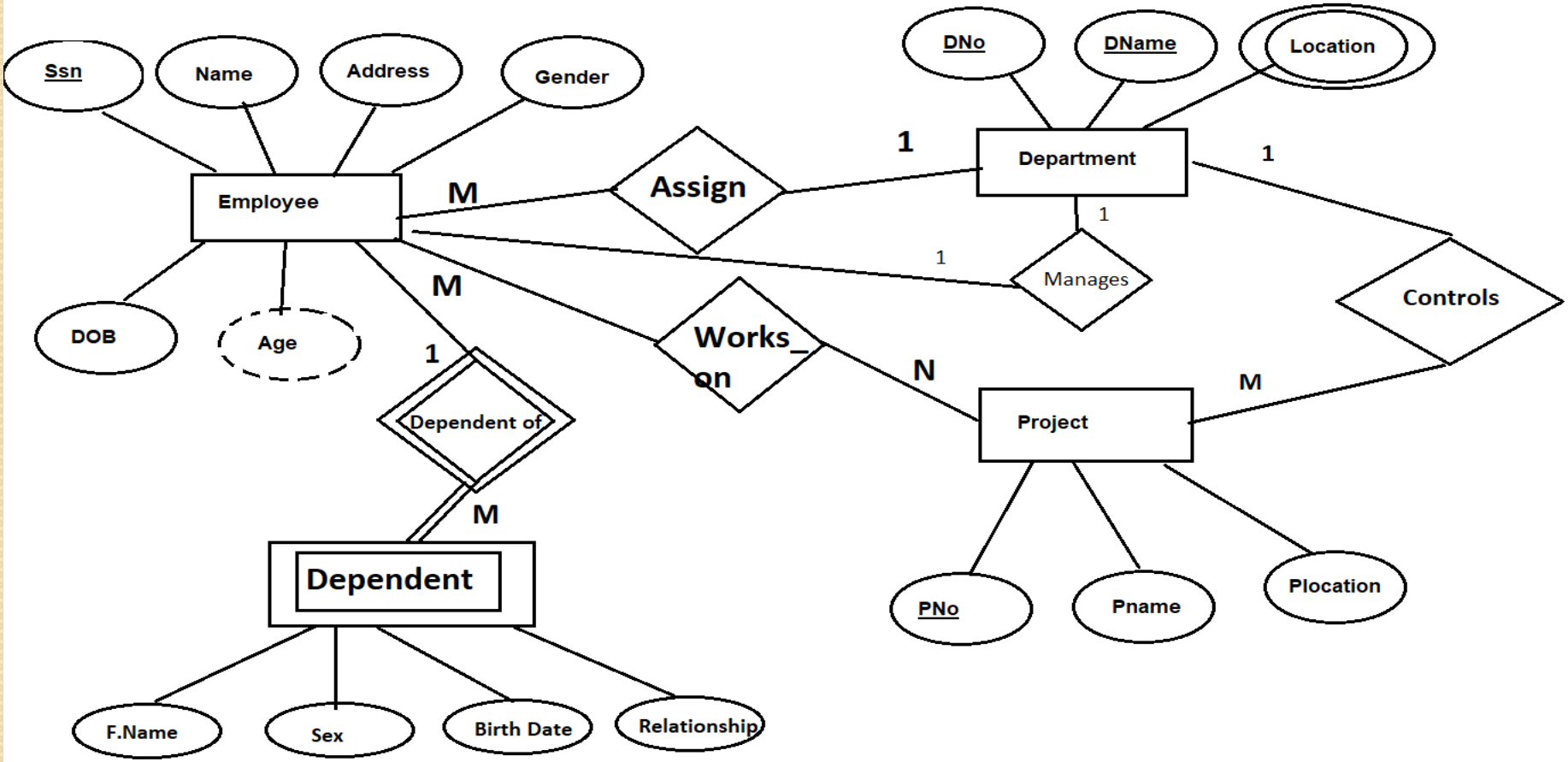
The company database keeps track of company's employee, department and projects. We store Employee's name,ssn,address,salary, gender,date of birth, age. An employee is assigned to one department, but may work on several projects which are not necessarily controlled by the same department. A particular employee manages the department. Each department has a unique name,unique number and several locations. The department controls number of projects each of which has a unique name, unique number and a single location. We want to keep track of the dependents of each employee for insurance purpose. We keep each dependent's first name, sex,birth date and relationship to employee



The company database keeps track of company's employee, department and projects. We store Employee's name,ssn,address,salary, gender,date of birth, age. An employee is assigned to one department, but may work on several projects which are not necessarily controlled by the same department. A particular employee manages the department. Each department has a unique name,unique number and several locations. The department controls number of projects each of which has a unique name, unique number and a single location. We want to keep track of the dependents of each employee for insurance purpose. We keep each dependent's first name, sex,birth date and relationship to employee



The company database keeps track of company's employee, department and projects. We store Employee's name,ssn,address,salary, gender,date of birth, age. An employee is assigned to one department, but may work on several projects which are not necessarily controlled by the same department. A particular employee manages the department. Each department has a unique name,unique number and several locations. The department controls number of projects each of which has a unique name, unique number and a single location. We want to keep track of the dependents of each employee for insurance purpose. We keep each dependent's first name, sex,birth date and relationship to employee

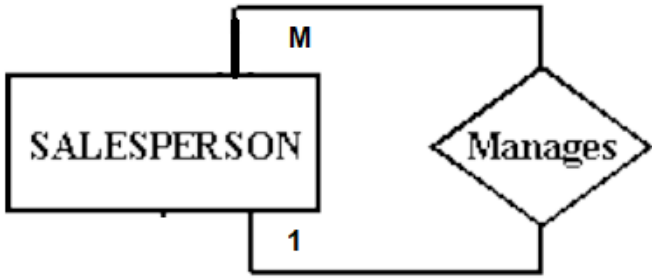


Example 2

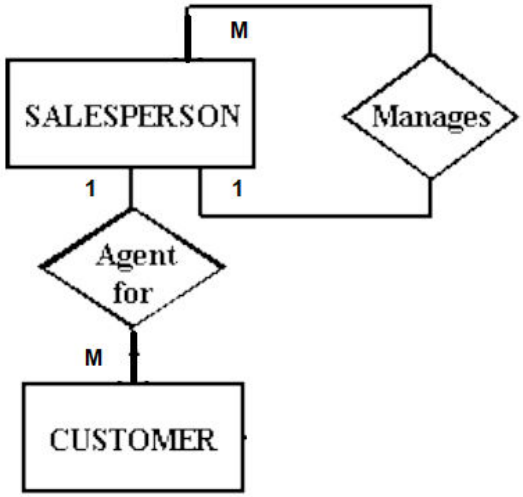
A company has the following scenario: There are a set of salespersons. *Some* of them manage other salespersons. However, a salesperson *cannot* have more than one manager. A salesperson can be an agent for many customers. A customer is managed by *exactly*one salesperson. A customer can place any number of orders. An order can be placed by *exactly*one customer. Each order lists one or more items. An item may be listed in many orders. An item is 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 different parts in certain quantities. A part may be supplied by different suppliers.

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 one manager. A salesperson can be an agent for many customers. A customer is managed by *exactly*one salesperson. A customer can place any number of orders. An order can be placed by *exactly*one customer. Each order lists one or more items. An item may be listed in many orders. An item is 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 different parts in certain quantities. A part may be supplied by different suppliers.

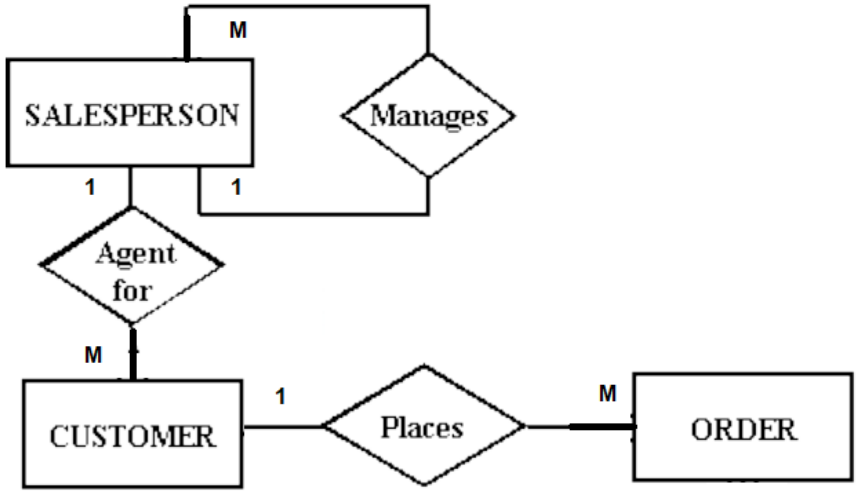
A company has the following scenario: There are a set of salespersons. *Some* of them manage other salespersons. However, a salesperson *cannot* have more than one manager. A salesperson can be an agent for many customers. A customer is managed by *exactly*one salesperson. A customer can place any number of orders. An order can be placed by *exactly*one customer. Each order lists one or more items. An item may be listed in many orders. An item is 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 different parts in certain quantities. A part may be supplied by different suppliers.



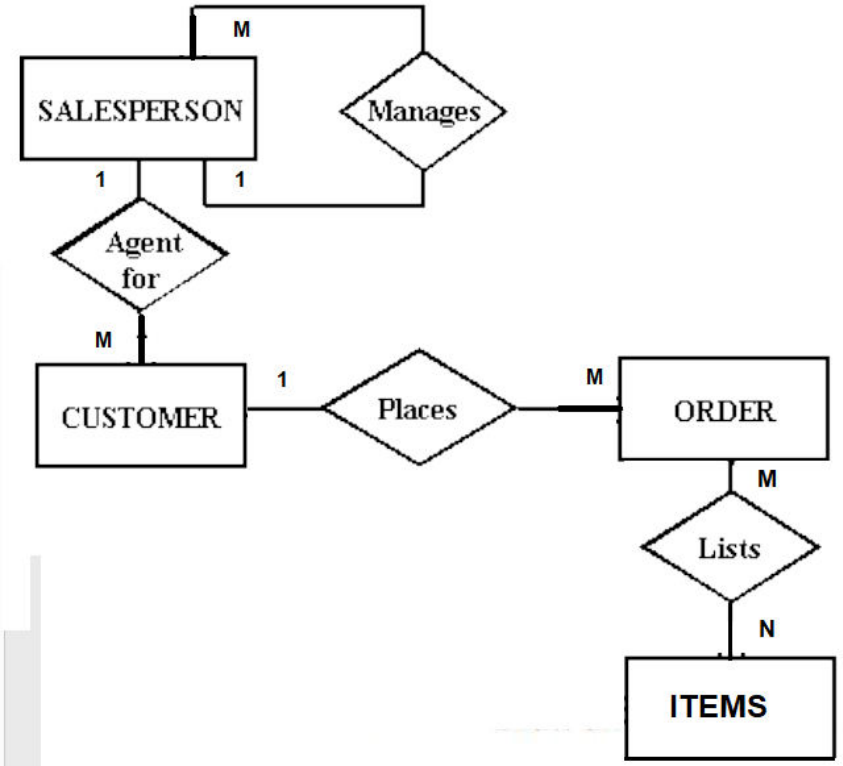
A company has the following scenario: There are a set of salespersons. *Some* of them manage other salespersons. However, a salesperson *cannot* have more than one manager. A salesperson can be an agent for many customers. A customer is managed by *exactly*one salesperson. A customer can place any number of orders. An order can be placed by *exactly*one customer. Each order lists one or more items. An item may be listed in many orders. An item is 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 different parts in certain quantities. A part may be supplied by different suppliers.



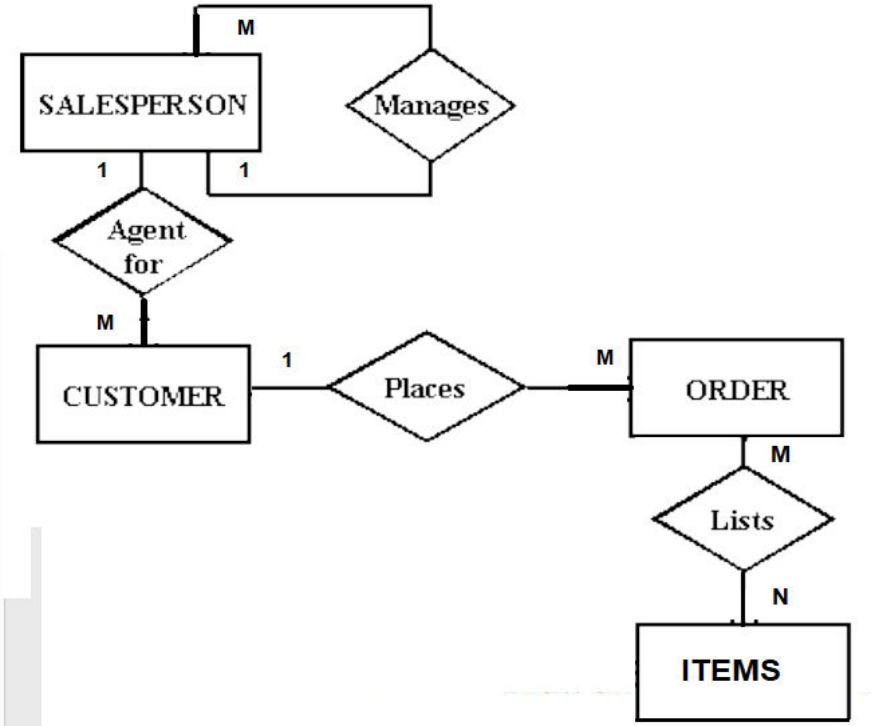
A company has the following scenario: There are a set of salespersons. *Some* of them manage other salespersons. However, a salesperson *cannot* have more than one manager. A salesperson can be an agent for many customers. A customer is managed by *exactly*one salesperson. A customer can place any number of orders. An order can be placed by *exactly*one customer. Each order lists one or more items. An item may be listed in many orders. An item is 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 different parts in certain quantities. A part may be supplied by different suppliers.



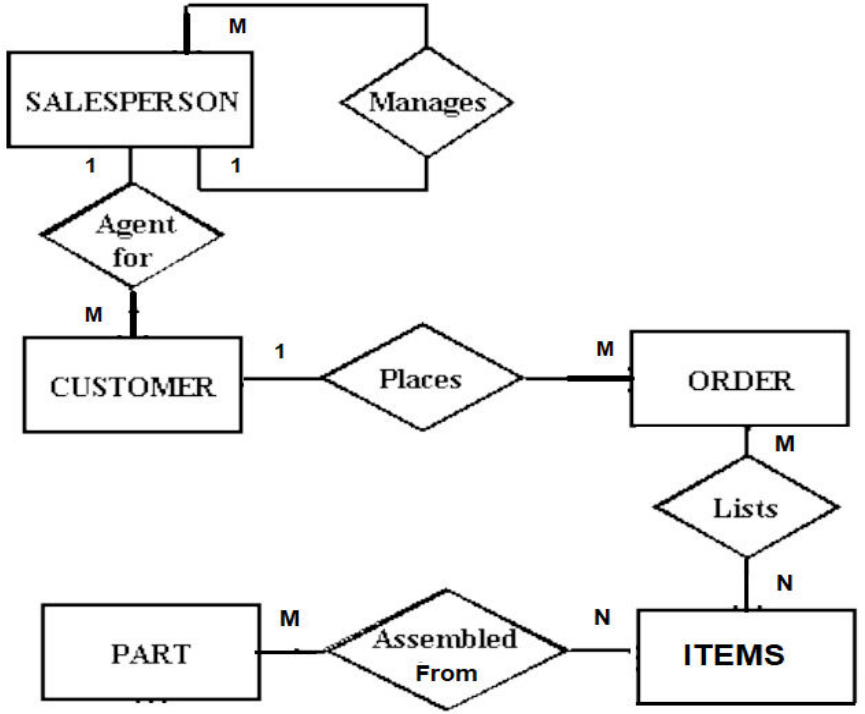
A company has the following scenario: There are a set of salespersons. *Some* of them manage other salespersons. However, a salesperson *cannot* have more than one manager. A salesperson can be an agent for many customers. A customer is managed by *exactly*one salesperson. A customer can place any number of orders. An order can be placed by *exactly*one customer. Each order lists one or more items. An item may be listed in many orders. An item is 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 different parts in certain quantities. A part may be supplied by different suppliers.



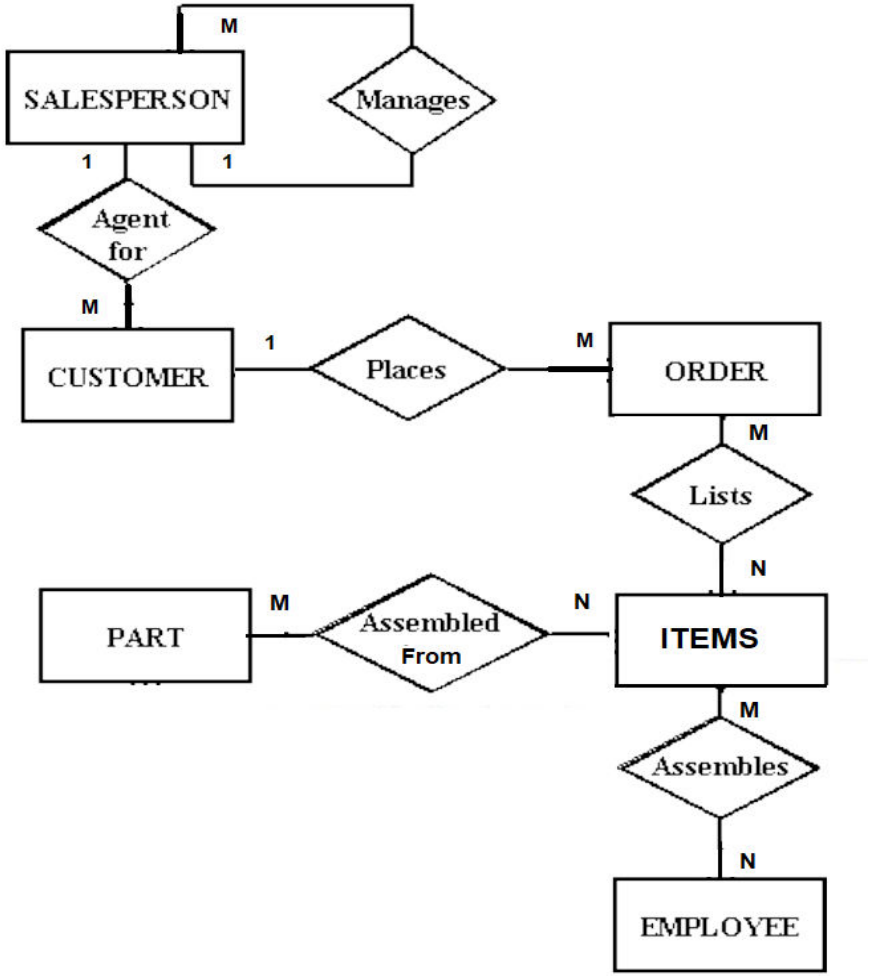
A company has the following scenario: There are a set of salespersons. *Some* of them manage other salespersons. However, a salesperson *cannot* have more than one manager. A salesperson can be an agent for many customers. A customer is managed by *exactly*one salesperson. A customer can place any number of orders. An order can be placed by *exactly*one customer. Each order lists one or more items. An item may be listed in many orders. An item is 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 different parts in certain quantities. A part may be supplied by different suppliers.



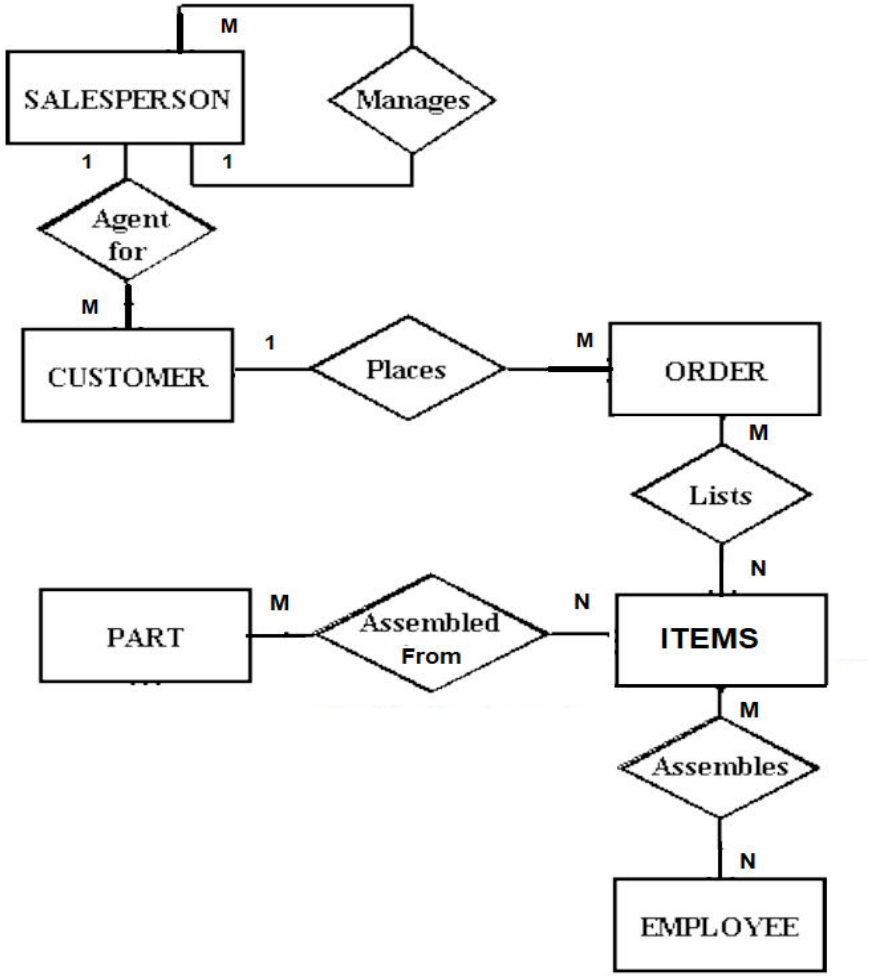
A company has the following scenario: There are a set of salespersons. *Some* of them manage other salespersons. However, a salesperson *cannot* have more than one manager. A salesperson can be an agent for many customers. A customer is managed by *exactly*one salesperson. A customer can place any number of orders. An order can be placed by *exactly*one customer. Each order lists one or more items. An item may be listed in many orders. An item is 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 different parts in certain quantities. A part may be supplied by different suppliers.



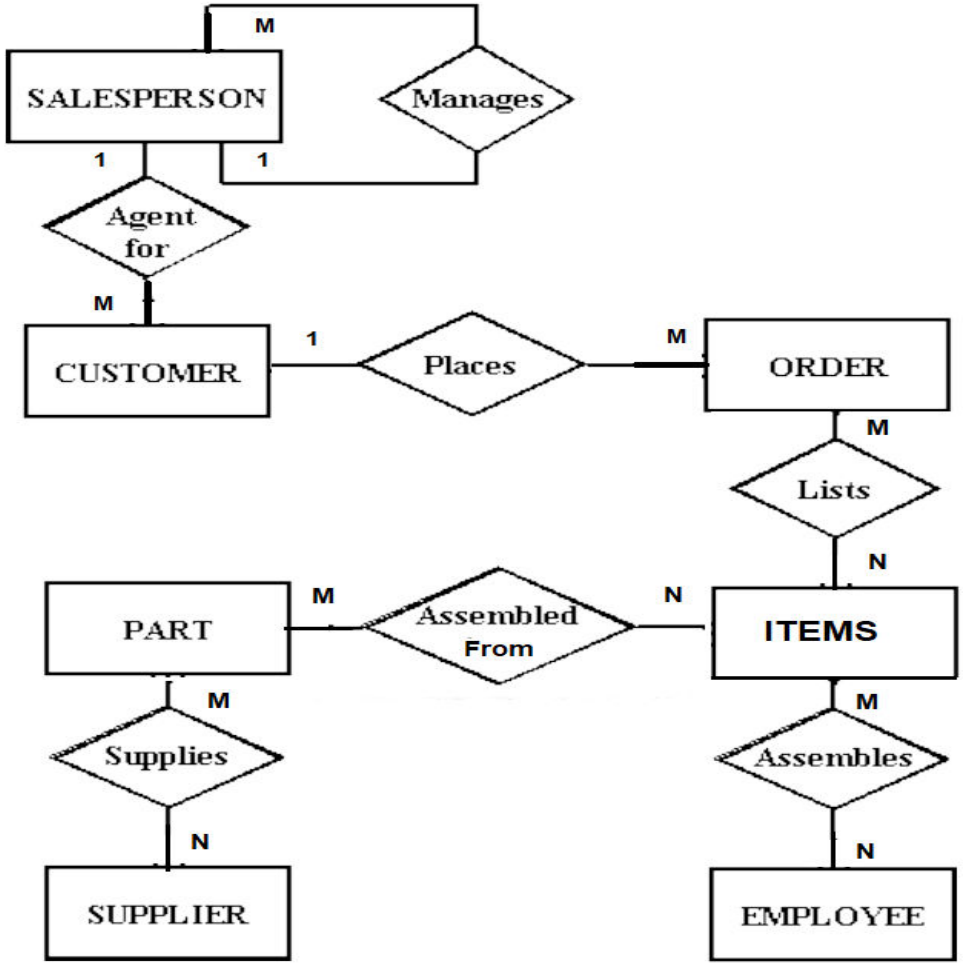
A company has the following scenario: There are a set of salespersons. *Some* of them manage other salespersons. However, a salesperson *cannot* have more than one manager. A salesperson can be an agent for many customers. A customer is managed by *exactly*one salesperson. A customer can place any number of orders. An order can be placed by *exactly*one customer. Each order lists one or more items. An item may be listed in many orders. An item is 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 different parts in certain quantities. A part may be supplied by different suppliers.



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 one manager. A salesperson can be an agent for many customers. A customer is managed by *exactly*one salesperson. A customer can place any number of orders. An order can be placed by *exactly*one customer. Each order lists one or more items. An item may be listed in many orders. An item is 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 different parts in certain quantities. A part may be supplied by different suppliers.



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 one manager. A salesperson can be an agent for many customers. A customer is managed by *exactly*one salesperson. A customer can place any number of orders. An order can be placed by *exactly*one customer. Each order lists one or more items. An item may be listed in many orders. An item is 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 different parts in certain quantities. A part may be supplied by different suppliers.



Lect 9 Quiz Answers

1 . In E-R Diagram Entity is represented as ----------

(a) Rectangle
(b) Oval
(c) Diamond

2. Attributes that can be divided into sub parts is called ----------

(a) Derived Attribute
(b) Composite Attribute
(c) Multivalued attribute

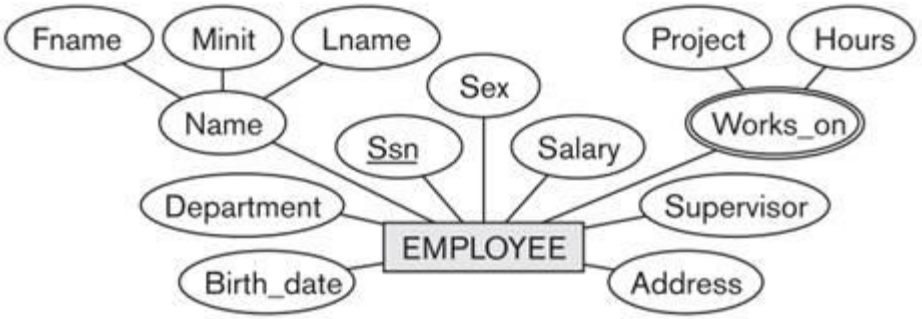
3. Attributes that can have more than one value for a given entity is ----------

(a) Multivalued Attribute

(b) Composite Attribute

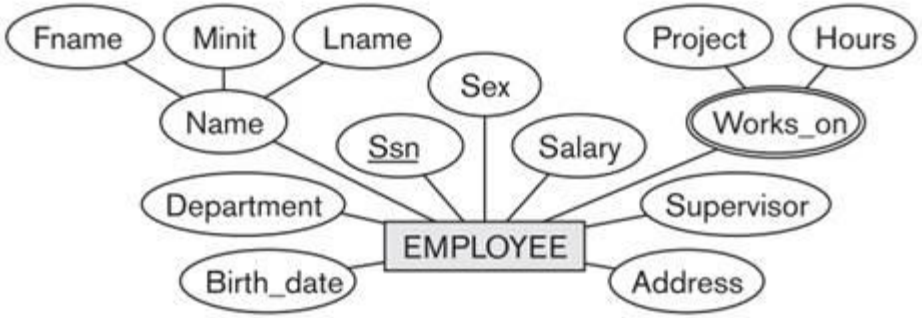
(c) Complex Attribute

4. In the following figure Identify the Key attribute



(a) Name
(b) Salary
(c) Ssn

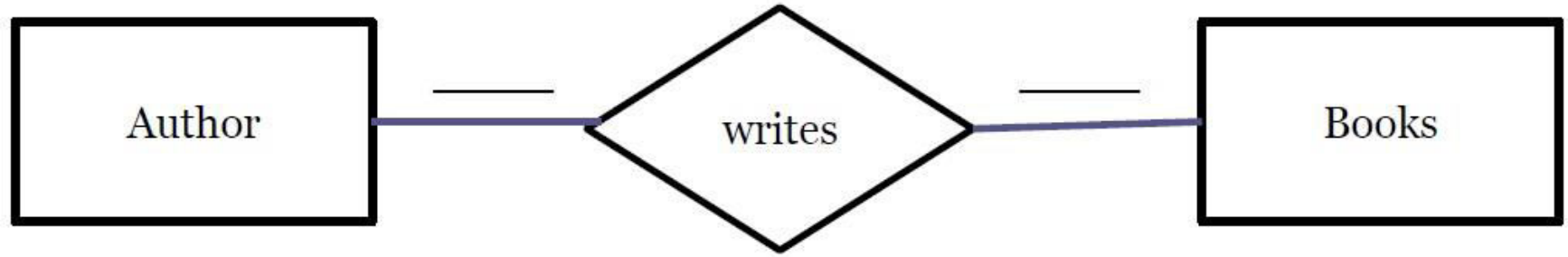
5. In the following E R Diagram identify the multivalued attributes



(a) Ssn
(b) Salary
(c) Works_on

Lect 10 Quiz Answers

1 . Identify the cardinality Ratio of the following Relationship

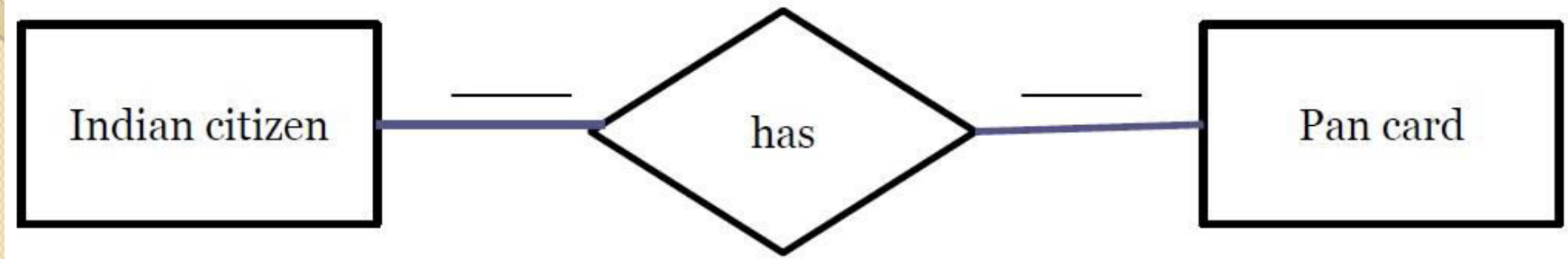


(a) One-one

(b) One-Many

(c) Many-Many

2. Identify the cardinality Ratio of the following Relationship

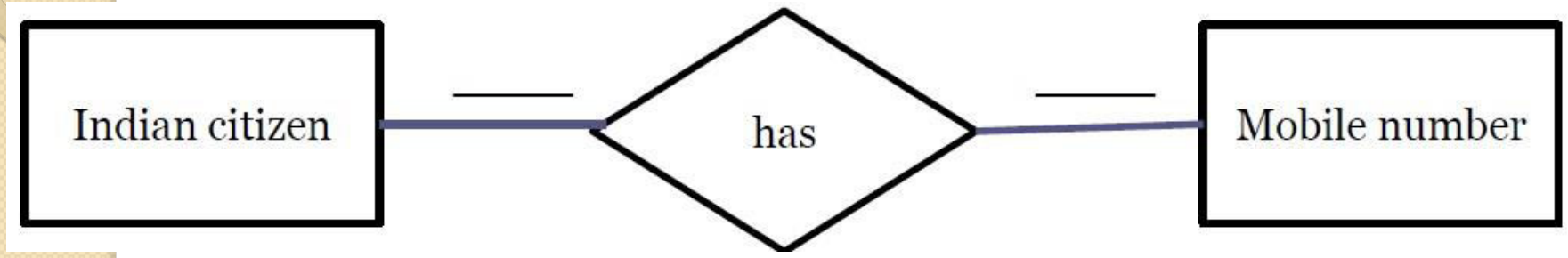


(a) One-one

(b) One-Many

(c) Many-Many

3. Identify the cardinality Ratio of the following Relationship

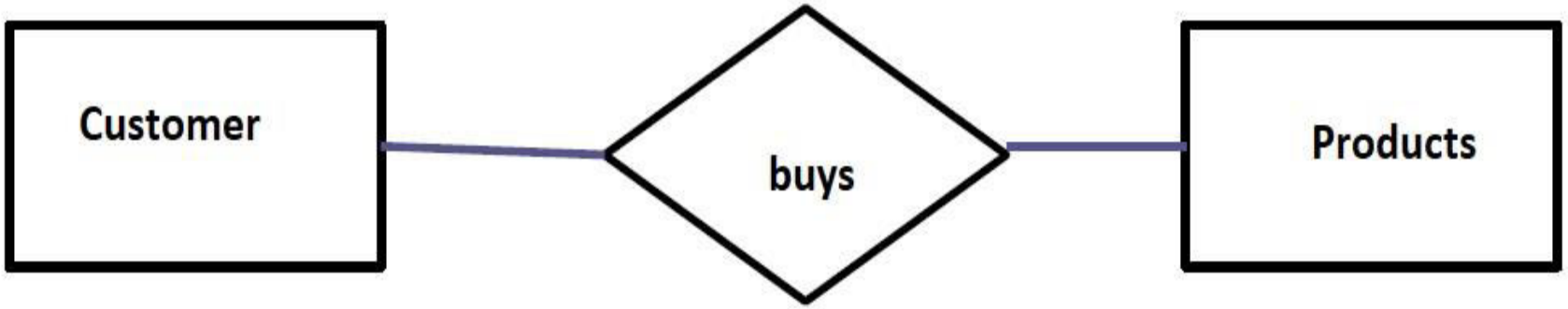


(a) One-one

(b) One-Many

(c) Many-Many

4. Identify the degree of the relationship

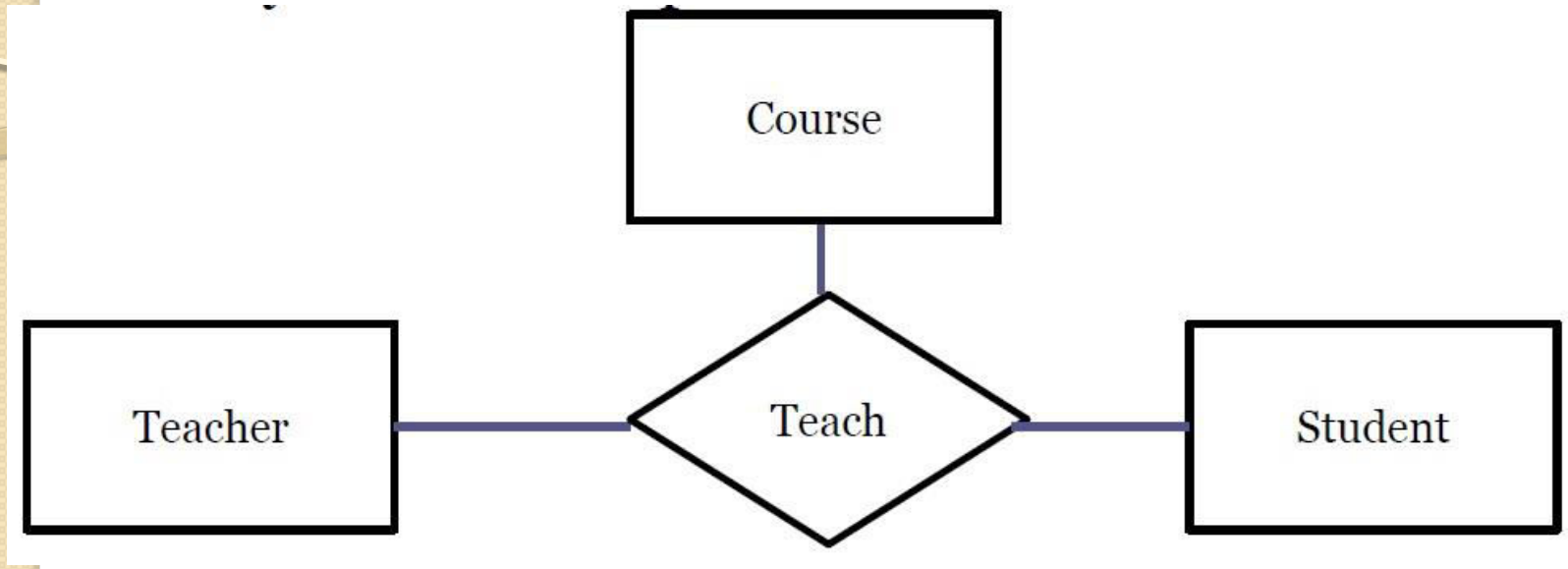


(a) Unary

(b) Binary

(c) Ternary

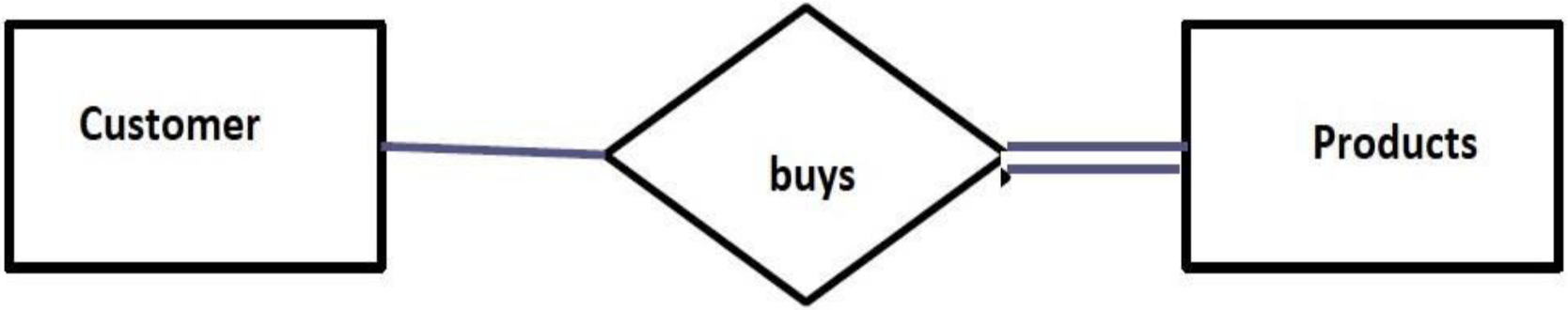
5. Identify the degree of the relationship



(a) Unary
(b) Binary
(c) Ternary

Lect 11 Quiz Answers

1. In the E R Diagram, Which Entity can have Total Participation?



(a) Customer

(b) Products

(c) both

2. In E-R Diagram Partial participation is represented in ----------

(a) Single line
(b) Double line
(c) Double diamond

3. Weak entity type ----------

(a) Does not have key attribute
(b) Have Key attribute

4. Understand the ER diagram and identify the weak & strong entity

(a) Weak Entity type is Employee,Strong Entity is Insurance record

(b) Weak Entity type is Insurance record, Strong Entity is Employee

(c) Both are weak entity type

5. Relationship type that links a strong entity type and a weak entity type is ----------

(a) Relationship set

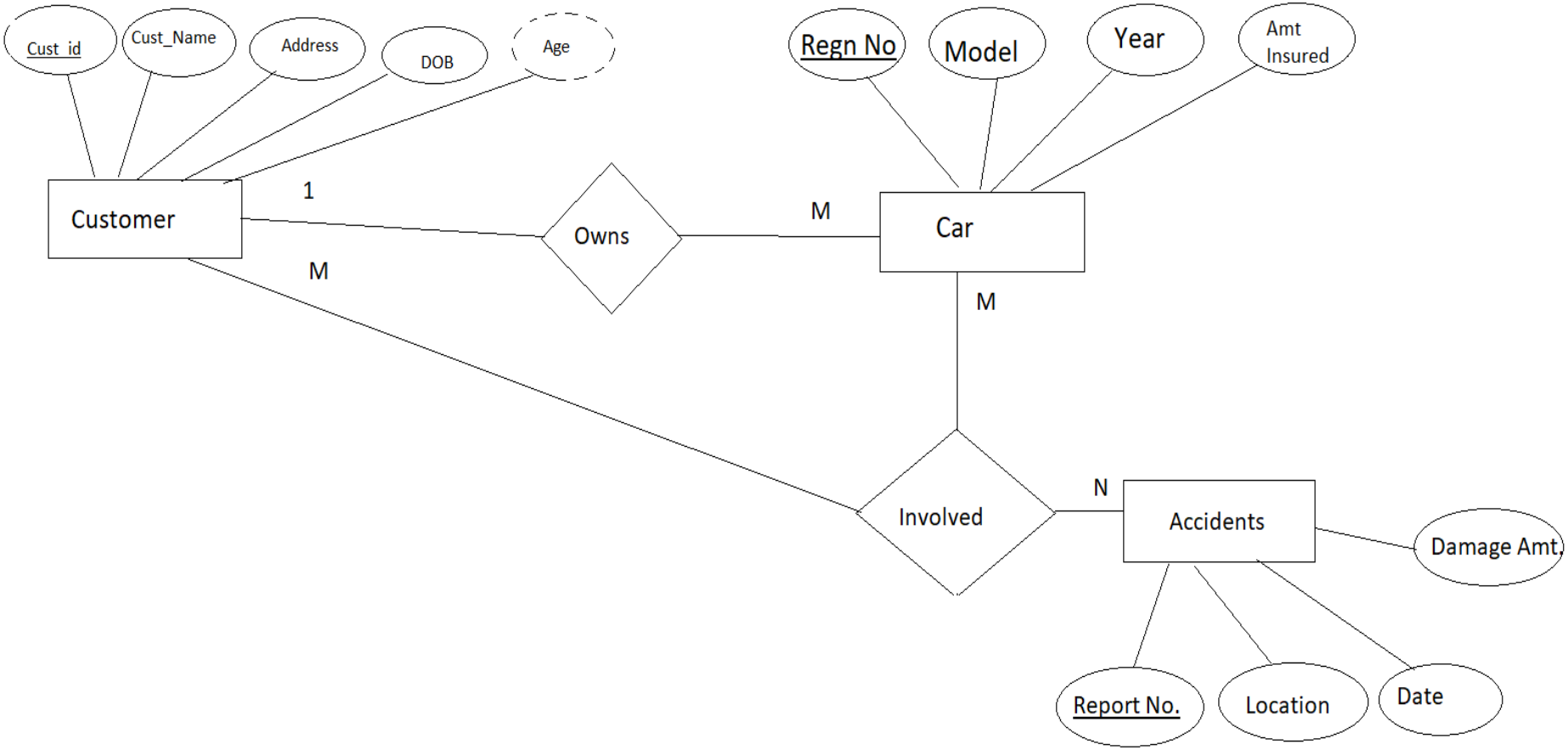
(b) Identifying relationship

(c) Normal Relationship

Lect 12 Assignment

- For a car-insurance company whose customers has a Unique Customer_id, Cust_name, Address, Date of birth,age. A customer owns many cars, A car can be owned by a single customer .The car is identified by unique regn no, model, year & amount insured. A person as well as a car can be involved in zero to any number of accidents. The accident is recorded by a unique report number,location ,date & damage amount. Construct the E-R diagram mentioning all the attributes and also the cardinalities of relationship.
-

For a car-insurance company whose customers has a Unique Customer_id, Cust_name, Address, Date of birth, age. A customer owns many cars, A car can be owned by a single customer .The car is identified by unique regn no, model, year & amount insured. A person as well as a car can be involved in zero to any number of accidents. The accident is recorded by a unique report number,location ,date & damage amount. Construct the E-R diagram mentioning all the attributes and also the cardinalities of relationship.



Alternative Notations for ER Diagrams

- One alternative ER notation for specifying structural constraints on relationships, which replaces the cardinality ratio (1:1, 1:N, M:N) with (min,max) notation
- The minimum number of times an entity can appear in a relation is represented by min
- The maximum time it an entity can appear in a relation is max.
- If m = 0 it denotes partial participation,
- , if m >0 it denotes total participation of the entity.

Using Min Max notation

- Database Management System

Module 1

Lect 13 :

PREV UNIVERSITY QUESTION

Module 1 University Question

1. List any three categories of database users, highlighting any one important characteristic of each category (3 Marks)

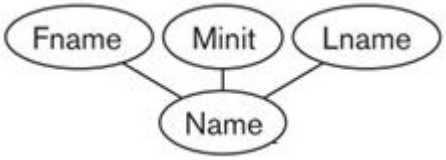
Hint :

- Database Administrators
- Database Designers
- End Users
  - Casual end users
  - Naive or parametric end users
  - Sophisticated end users
  - Standalone users
- System Analysts and Application Programmers

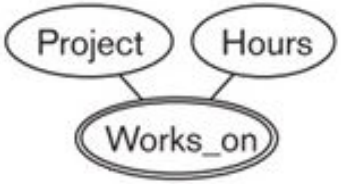
2. Give suitable examples for multivalued, composite & multivalued composite attributes (3 Marks)

Hint : Multivalued –

Composite-



Composite multivalued -

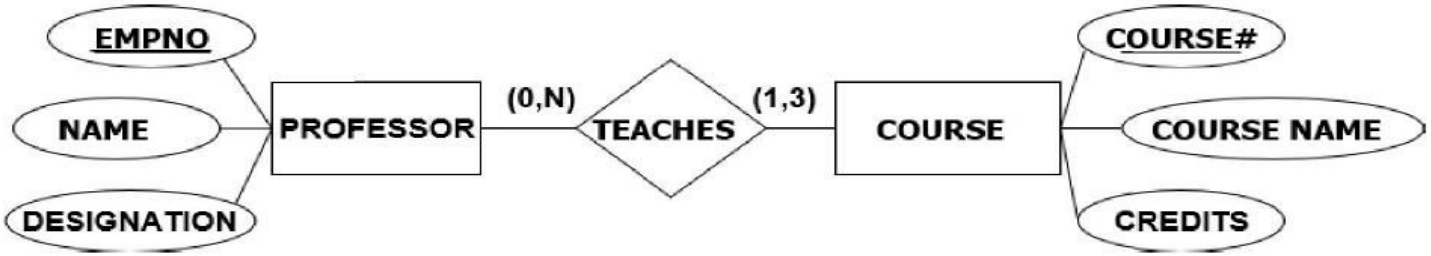


3. Distinguish between total participation & partial participation with example (4 Marks)

Hint :

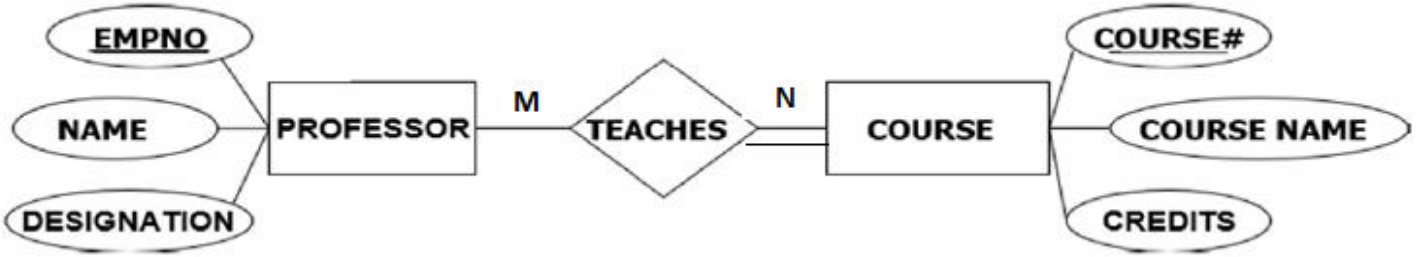
- **Total Participation** :
  - If every entity in the entity type participates in at least one relationship in the relationship type
  - Represented by double lines in ER Diagram
- **Partial Participation :**
  - Some entities may not participate in any relationship in the relationship type
  - Represented by single line in ER Diagram

4. Describe the real-world situation described by the following E-R diagram (3 Marks)



Re-draw the ER diagram replacing the *(min,max) notation* with the conventional notation showing cardinality and participation

Soln:



5. List out the three features of database systems (3 Marks)

Soln : (Characteristicts of DB)

- 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 multiuser transaction processing

6. How is DML different from DDL

(3 Marks)

Hint : Refer Database Languages topic

8. Justify weak entity set with the help of an example (5 Marks)

Refer

9. What are the responsibilities of DBA (4 Marks)

Hint

- Schema definition.
- Storage structure and access-method definition.
- Schema and physical-organization modification
- Granting of authorization for data access.
- Routine Maintenance:

11. With the help of neat diagram explain 3 schema architecture (9 Marks)

Hint : Explain about Internal Level, Conceptual level & View Level

12. Explain the term participation Constraint in detail (3 Marks)

Hint : Explain about Total Participation & Partial Participation

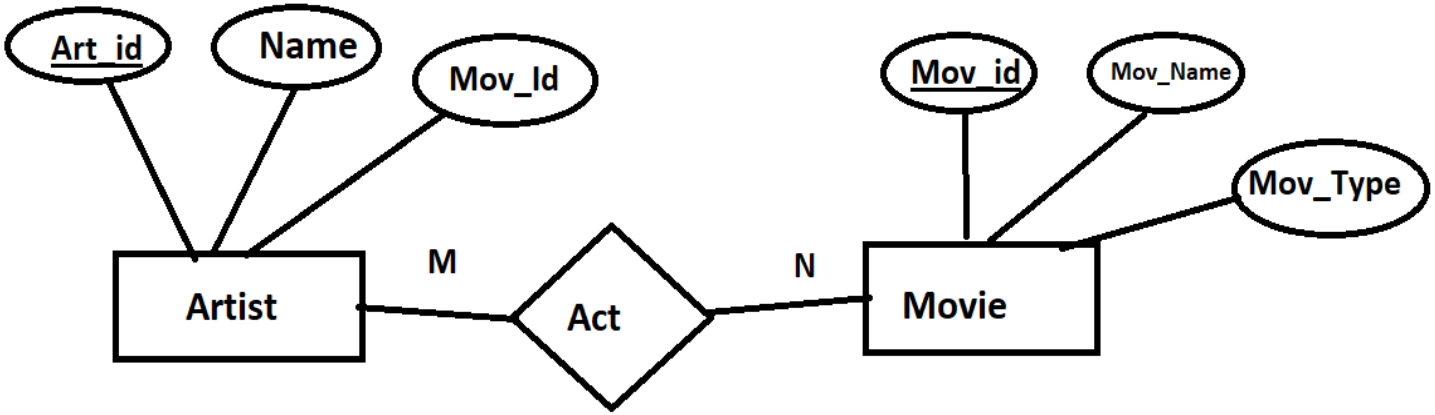
13. List out any three responsibilities of database administrators (3 Marks)

- Hint :
- Schema definition.
- Storage structure and access-method definition
- Schema and physical-organization modification
- Granting of authorization for data access

14. Give good examples (using ER notation) for unary and ternary relationships with a very brief explanation

Hint : Explain about Unary & Ternary relationship

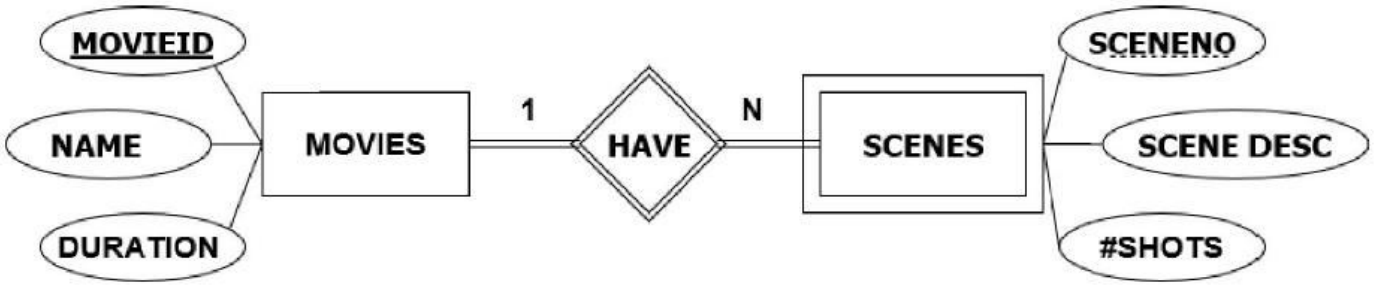
15. Consider a scenario where artists act in movies an artist can act in different movies and movie can have many. artists Assuming suitable attributes show how the situation can be represented using relations with foreign keys. (5 Marks)



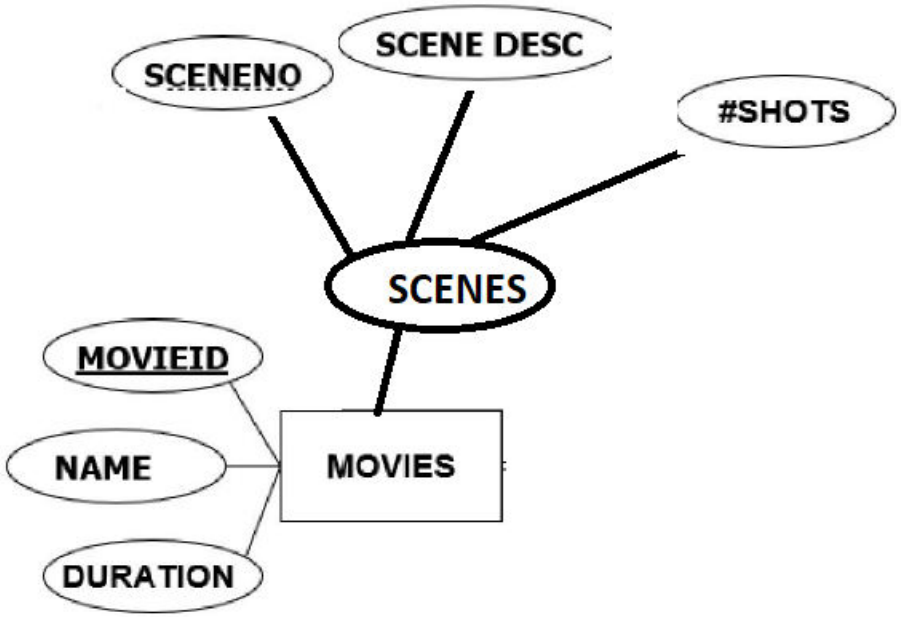
16. Briefly explain the concepts of Physical data independence & logical data independence with a typical real world example (5 Marks)

Hint : Refer Data Independence Topic

17. In the following ER diagram how can we replace the entity set SCENE with an attribute of the entity set MOVIE? Draw the new ER diagram. (5 Marks)



Soln :

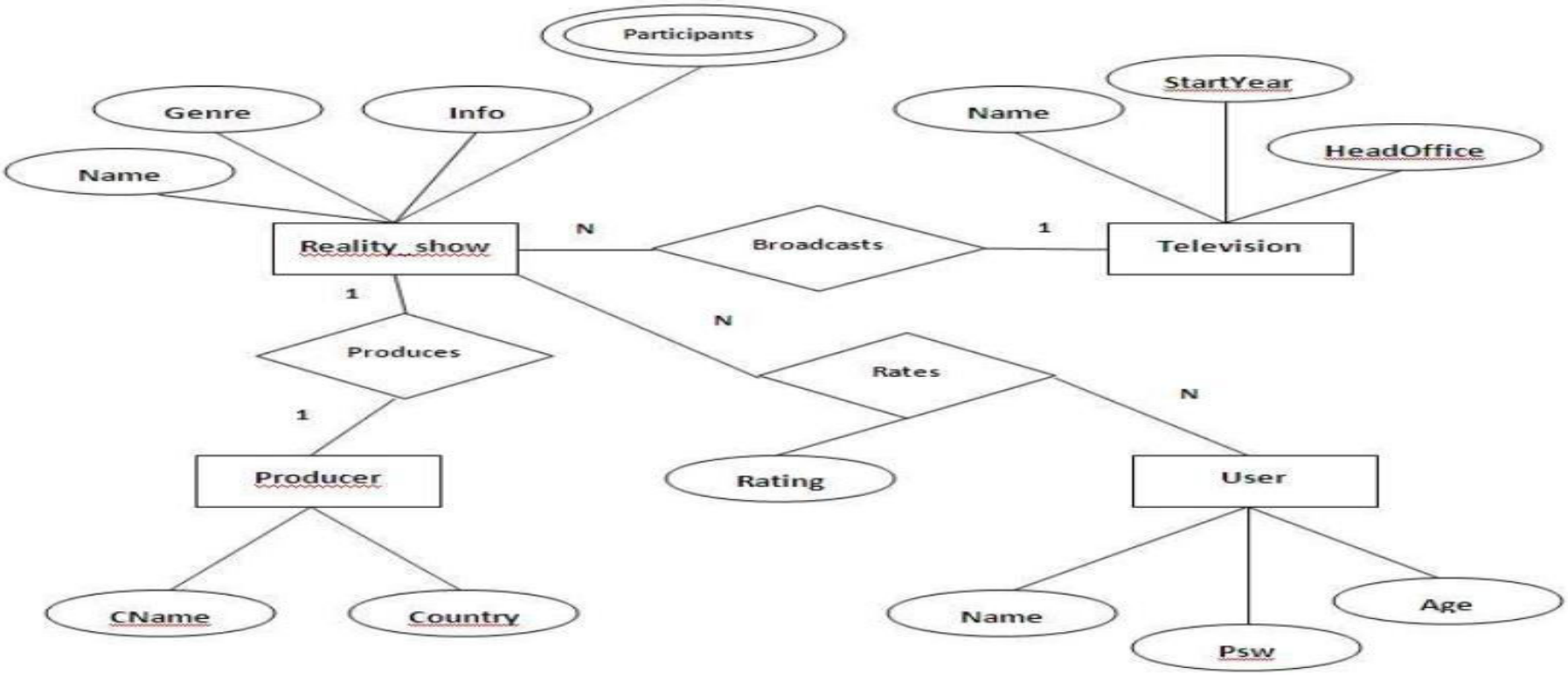


18. With suitable example, define integrity constraint?

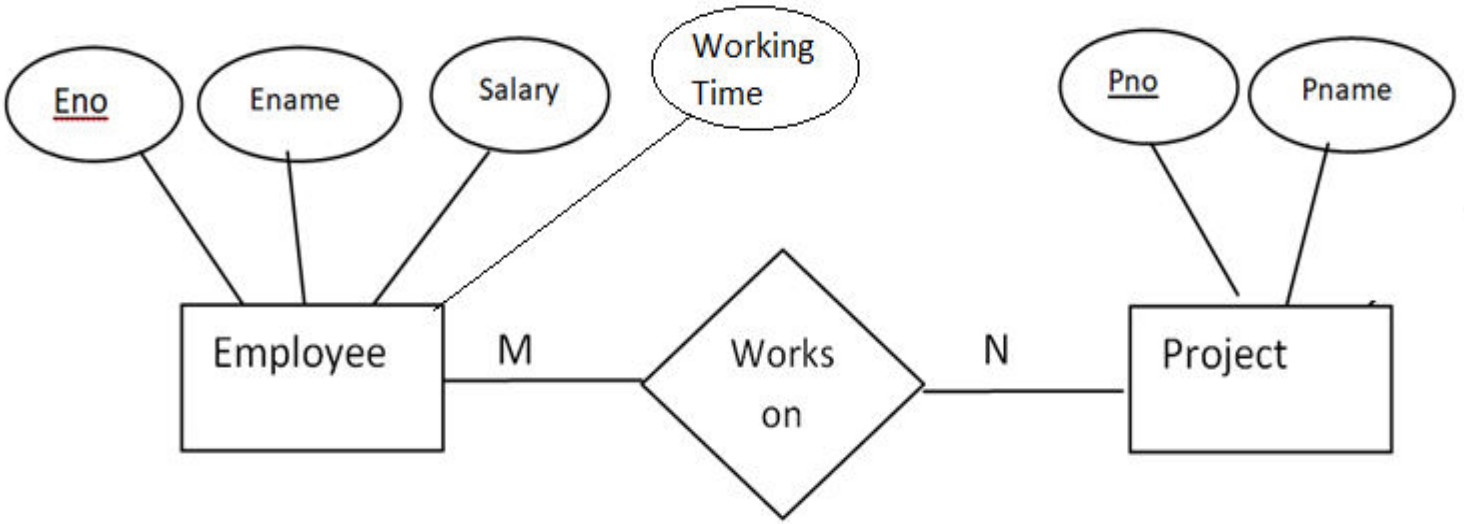
Hint : Explain about

- **referential integrity constraint**
- **key or uniqueness constraint**

- 19. Design an ER diagram for the given scenario; Suppose that you are designing a schema to record information about reality shows on TV. Your database needs to record the following information: _ For each reality show, its name, genre, basic_info and participants name. Any reality show has at least two or more participants. - For each producer, the company name, company country. A show is produced by exactly one producer. And one producer produces exactly one show. - For each television, its name, start year, head office. A television may broadcasts multiple shows. Each show is broadcasted by exactly one television. -For each user, his/her username, password, and age. A user may rate multiple shows, and a show may be rated by multiple users. Each rating has a score of 0 to 10. (9 Marks)



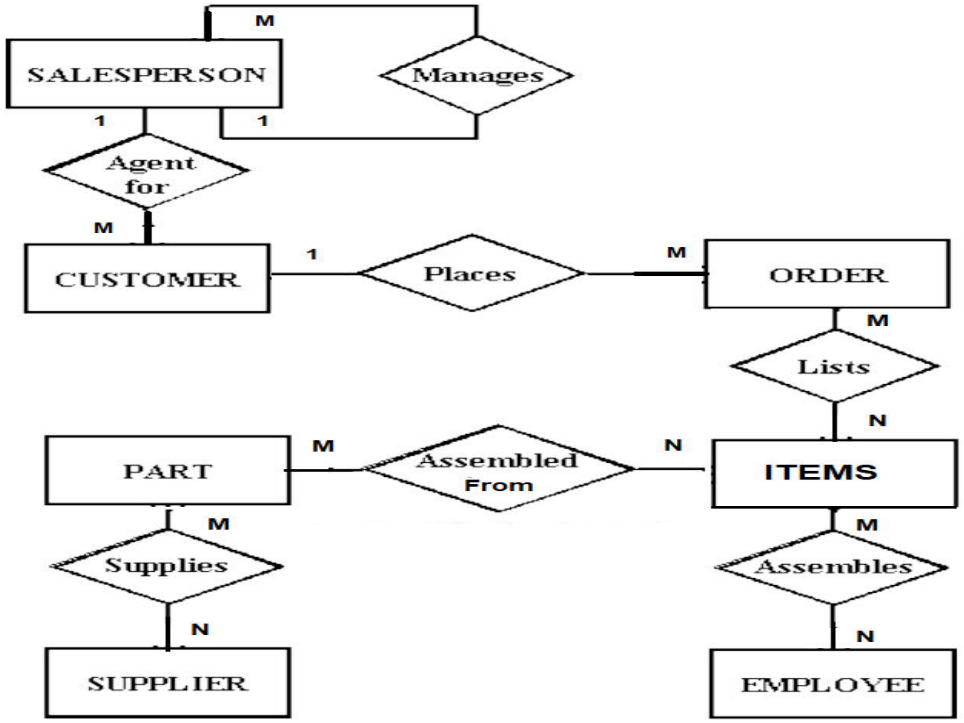
A company has many employees working on a project. An employee can be part of one or more projects. Each employee works on a project for certain amount of time. Assume suitable attributes for entities and relations. Mark the primary key(s) and the cardinality ratio of the relations.

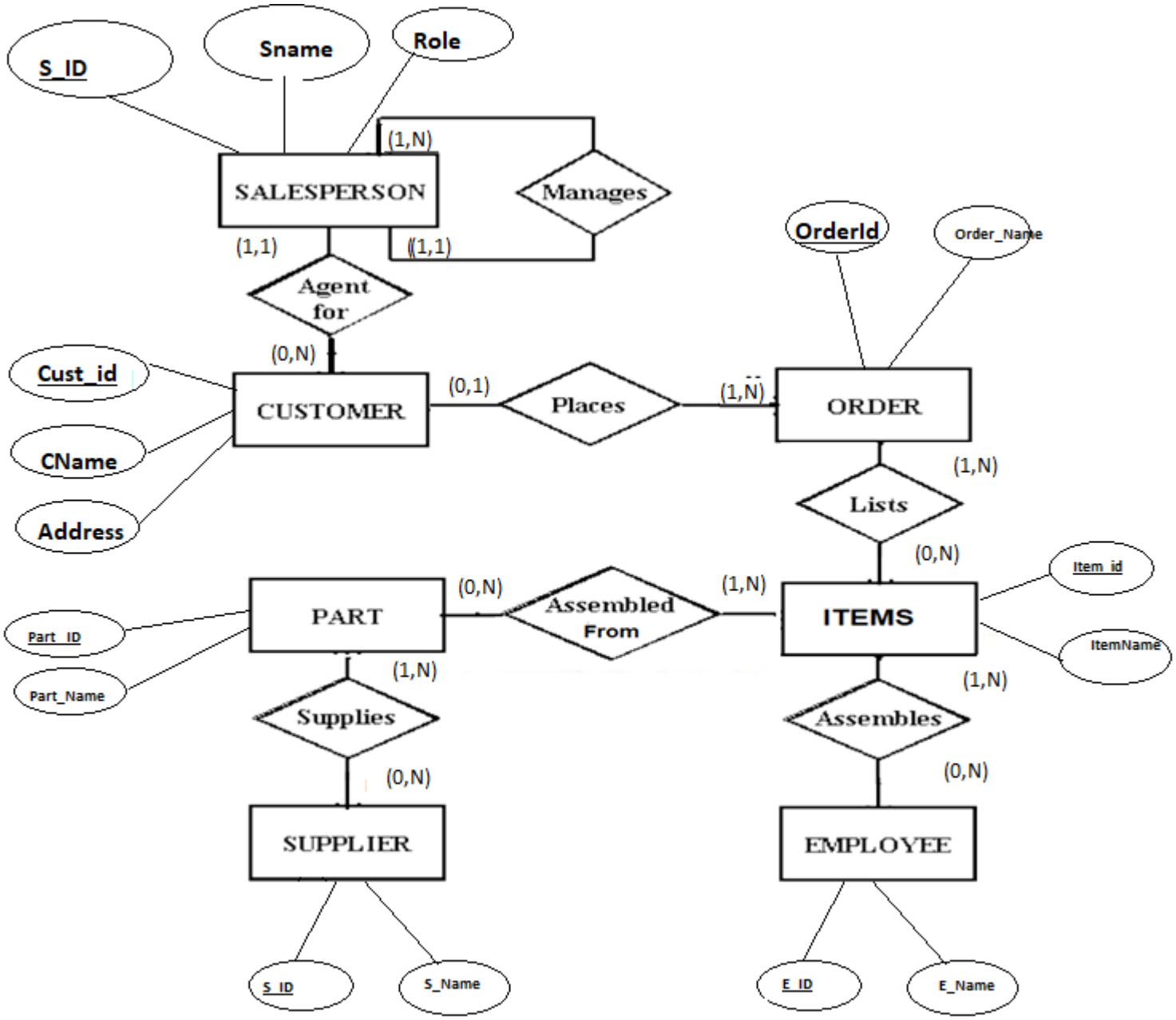


6. A company has the following scenario: There are a set of salespersons. *Some* of them manage other salespersons. However, a salesperson *cannot* have more than one manager. A salesperson can be an agent for many customers. A customer is managed by *exactly*one salesperson. A customer can place any number of orders. An order can be placed by *exactly*one customer. Each order lists one or more items. An item may be listed in many orders. An item is 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 different parts in certain quantities. A part may be supplied by different suppliers. (9)

(i) Identify and list entities, suitable attributes, primary keys, foreign keys and relationships to represent the scenario.

(ii) Draw an ER diagram to model the scenario using *min-max* notation.




S_ID
Sname
Role
SALESPERSON
(1,N)
Manages
(1,1)
(1,1)
Agent for
(0,N)
Cust_id
CUSTOMER
CName
Address
(0,1)
Places
(1,N)
OrderId
Order_Name
ORDER
(1,N)
Lists
(0,N)
Item_id
ITEMS
ItemName
(1,N)
Assembles
(0,N)
EMPLOYEE
E_ID
E_Name
(1,N)
Assembled From
(0,N)
PART
Part_ID
Part_Name
(1,N)
Supplies
(0,N)
SUPPLIER
S_ID
S_Name