Introduction: Database System Applications, Purpose of Database Systems, View of Data, components and structure, Database Users and Administrator, History of Database Systems. Data models: ER model, Entities, Attributes and Entity sets, Relationships and Relationship sets, Additional features of ER Model, Conceptual Design with the ER Model, Enhanced E-R Model.

INTRODUCTION :

Database :

- A database is a collection of related data, contains information relevant to an enterprise.
- Defining a database involves specifying the data types, structures, and constraints of the data to be stored in the database.
- It provides facilities for retrieving, adding, modifying, and deleting the data when required.

DBMS:

- ***A database-management system (DBMS) is a collection of interrelated data and a set of programs to access those data.***
- The DBMS is hence a **general-purpose software system** that facilitates the processes of **defining, constructing, and manipulating databases** for various applications.
- The primary goal of a DBMS is to provide a way to store and retrieve database information that is both convenient and efficient.
- **Eg.** Oracle, My Sql, Ms-access, Sybase, Informix, Foxpro

Database System:

- Database and DBMS together known as Database system.

1.1 DATABASE APPLICATIONS:

Databases are widely used. Here are some representative applications:

- **Enterprise Information**
  - **Sales:** For customer, product, and purchase information.
  - **Accounting:** For payments, receipts, account balances, assets and other accounting information.
  - **Human resources**: For information about employees, salaries, payroll taxes, and benefits, and for generation of paychecks.
  - **Manufacturing:** For management of the supply chain and for tracking production of items in factories, inventories of items in warehouses and stores, and orders for items.

- **Online retailers:** For sales data noted above plus online order tracking, generation of recommendation lists, and maintenance of online product evaluations.

- **Banking and Finance:**
  - **Banking:** For customer information, accounts, loans, and banking transactions.
  - **Credit card transactions**: For purchases on credit cards and generation of monthly statements.
  - **Finance:** For storing information about holdings, sales, and purchases of financial instruments such as stocks and bonds; also for storing real-time market data to enable online trading by customers and automated trading by the firm.
- **Universities:** For student information, course registrations, and grades (in addition to standard enterprise information such as human resources and accounting).
- **Airlines:** For reservations and schedule information. Airlines were among the first to use databases in a geographically distributed manner.
- **Telecommunication:** For keeping records of calls made, generating monthly bills, maintaining balances on prepaid calling cards, and storing information about the communication networks.
- **Manufacturing:** production, inventory, orders, supply chain

Advantaged of DBMS:

- ✓ Controlling Redundancy Restricting Unauthorized Access
- ✓ Providing Persistent Storage for Program Objects and Data Structures
- ✓ Providing Multiple User Interfaces
- ✓ Representing Complex Relationships Among Data
- ✓ Enforcing Integrity Constraints
- ✓ Providing Backup and Recovery

1.2 PURPOSE OF DATABASE SYSTEM:

The file-processing system is supported by a conventional operating system. The system stores permanent records in various files, and it needs different application programs to extract records from, and add records to, the appropriate files.

Purpose of Database Systems or Major **Disadvantages Of File-Processing** System are,

- ❖ **Data redundancy and Inconsistency:**

- **Data redundancy** - The same information may be **duplicated** in several places (files). This redundancy leads to higher storage and access cost.
  Storing the information several time leads to waste of storage space is called data redundancy .
  Eg: address and telephone number of a particular customer may appear in a file that consists of savings account record and in checking account records.
- **Data inconsistency** - the various copies of the same data may no longer use.
  Eg: a changed customer address may be reflected in savings-account records but not elsewhere in the system.

❖ **Difficulty in accessing data**

- Need to write a new program to carry out each new task.
- That conventional file-processing environments do not allow needed data to be retrieved in a convenient and efficient manner

❖ **Data isolation (i e. Multiple files and formats)**

Because data are scattered in various files, and files may be in different formats, writing new application programs to retrieve the appropriate data is difficult.

❖ **Integrity problems**

- The data values stored in the database must satisfy certain types of consistency constraints. However, when new constraints are added, it is difficult to change the programs to enforce them. The problem is compounded when constraints involve several data items from different files.
- **Eg:** The balance of certain types of bank accounts may never fall below a prescribed amount. ($25). Developers must enforce these constraints in the system by adding appropriate code in various application programs.

❖ **Atomicity problems**

- The transactions must be atomic – **It must happen completely or not at all.**
- It is difficult to ensure atomicity in a conventional file processing system. A Computer system is subjected to failure. If failure occurs, the data has to be restored to the consistent state that existed prior to failure.
- In DB approach, the DBMS ensures atomicity using the Transaction manager inbuilt in it. DBMS supports online transaction processing and recovery techniques to maintain atomicity.

❖ **Concurrent-access or Sharing of Data:**

- For the sake of overall performance of the system and faster response, many systems allow multiple users to update the data simultaneously.
- **Eg.** two people reading a balance and updating it at the same time

❖ **Security problems**

- Not every user of the database system should be able to access all the data.
- Eg: in a banking system, payroll personnel need to see only the information about the various employees, not about customer accounts.
- DBMS provide security and authorization subsystem

  The main characteristics of the ***database approach versus the file-processing*** approach are the following:
  - ✓ 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

1.3 VIEWS OF DATA

A database system is a collection of interrelated data and a set of programs that allow users to access and modify these data.

A major purpose of a database system is to provide users with an abstract view of the data. That is, the ***system hides certain details of how the data are stored and maintained***.

1.3.1 Data Abstraction:

Since many database-system users are not computer trained, developers hide the complexity from users through several levels of abstraction, to simplify users' interactions with the system.

Physical level

The lowest level of ***abstraction describes how the data are actually stored***. The physical level describes complex low-level data structures in detail.

Logical level

The next-higher level of ***abstraction describes*** **what data are stored** ***in the database, and*** **what relationships exist** ***among those data***.

The logical level describes the entire database in terms of a small number of relatively simple structures. Database administrators use the logical level of abstraction to decide what information to keep in the database.

Eg.

type *instructor* = **record**

ID : **char (5);**

name : **char (20);**
dept name : **char (20);**
salary : **numeric (8,2);**
end;

This code defines a new record type called *instructor* with four fields. Each field has a name and a type associated with it.

View level

The highest level of ***abstraction describes only part of the entire database***. Many users of the database system do not need all the information. Instead, they need to access only a part of the database. So, the view level of abstraction exists to simplify their interaction with the system.

A university organization may have several such record types, including

- **department,** with fields dept name, building, and budget
- **course,** with fields course id, title, dept name, and credits
- **student,** with fields ID, name, dept name, and tot_cred

✓ At the physical level, an instructor, department, or student record can be described as a block of consecutive storage locations.

✓ At the logical level, each such record is described by a type definition.

✓ Finally, at the view level, computer users see a set of application programs that hide details of the data types.

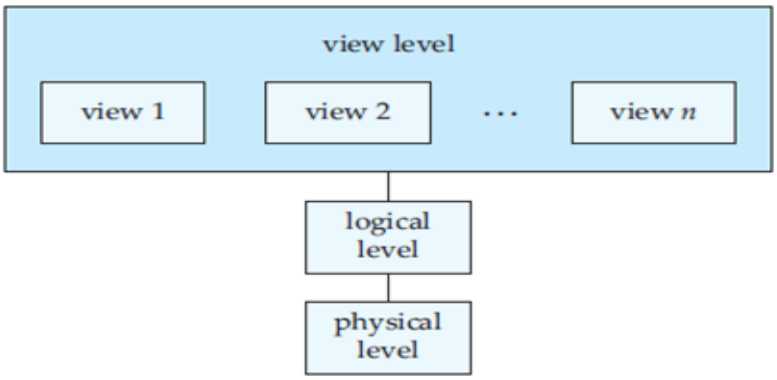


Fig. The three levels of data abstraction

1.3.2 Instances and Schemas:

Instances :

Databases change over time as information is inserted and deleted. The collection of information stored in the database at a particular moment is called an **instance** of the database.

Schema : The overall design of the database is called the database schema. Schemas are not changed frequently.

1.3.3 Three-Schema Architecture:

Database systems have several schemas, partitioned according to the levels of abstraction.

- ✓ **The internal level** has an **internal schema (Physical schema)**: It describes the database design at the physical level.
- ✓ **The conceptual level** has a **conceptual schema(Logical schema)**:
  It describes the database design at the logical level (i.e., describes the structure of the whole database for a community of users).
  The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints

Data Independence :

- **Physical Data Independence** – the ability to modify the physical schema without changing the logical schema
- **Logical data independence -** is the capacity to change the conceptual schema without having to change external schemas (View or subschema) or application programs.
- **The external** or **view level (Sub schemas):** describe different views of the database. The external level interacts directly with the user.

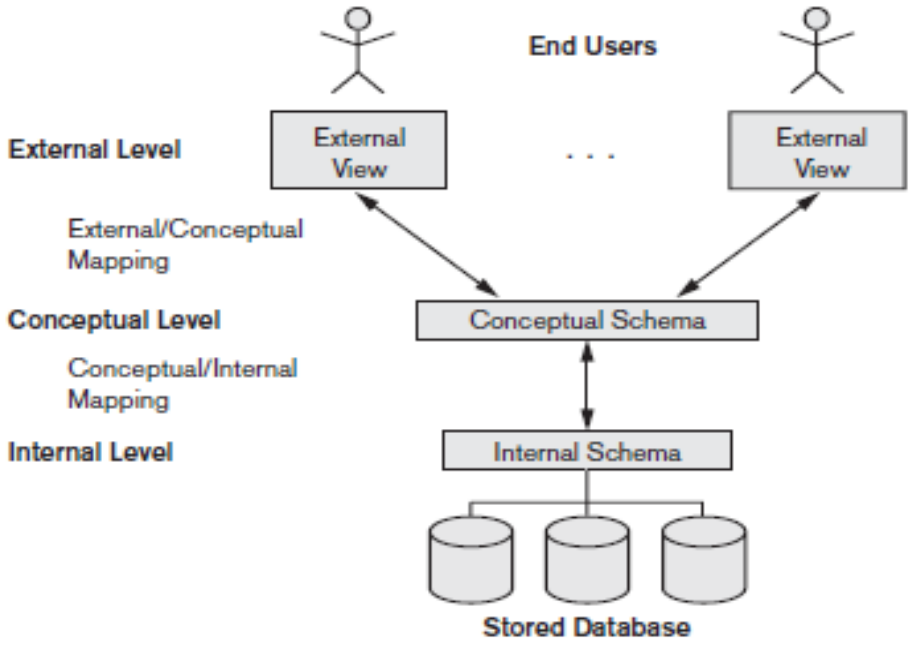


Fig: The three-schema architecture.

1.4 DATABASE ARCHITECTURE / COMPONENTS AND STRCUTURE/ DATABASE USERS & ADMINISTRATORS

Database systems can be centralized, or client-server, where one server machine executes work on behalf of multiple client machines. Database systems can also be designed to exploit parallel computer architectures. Distributed databases span multiple geographically separated machines.

Database Users:

Users are differentiated by the way they expect to interact with the system. 4 types of database users are,

- ✓ **Naive users**

  Naive users are unsophisticated users who interact with the system by invoking one of the application programs that have been written previously. The typical user interface for naive users is a forms interface, where the user can fill in appropriate fields of the form. For example, Bank tellers, Reservation agents for airlines, hotels, and car rental companies.

- ✓ **Application programmers:**

  Application programmers are computer professionals who write application programs. Application programmers can choose from many tools to develop user interfaces such as Rapid application development (RAD) tools . For example, software, developers or software engineers.

- ✓ **Sophisticated users**

  Sophisticated users interact with the system without writing programs. Instead, they form their requests in a database query language. For example , engineers, scientists, business analysts.

- ✓ **Specialized users**

  Specialized users are sophisticated users who write specialized database applications that do not fit into the traditional data-processing framework. For example , computer-aided design systems, geo-spatial systems.

- ✓ **Database Administrator(DBA)**

- Coordinates all the activities of the database system. The database administrator has a good understanding of the enterprise's information resources and needs.
- **Functions of Database administrator includes**:
  - **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**
  - **Schema and physical organization modification**

- **Granting user authority to access the database –** to decide which parts of the database various users can access.
- **Specifying integrity constraints.**
- **Monitoring performance and responding to changes in requirements**
- **Routine maintenance -** Periodically backing up the database, Ensuring that enough free disk space is available, Ensuring that performance is not degraded

Query Processor:

- The **query processor** subsystem compiles and executes DDL and DML statements.
- The query processor will accept query from user and solves it by accessing the database.

DDL interpreter

- This will interprets DDL statements and fetch the definitions in the data dictionary.

DML compiler

- This will translates DML statements in a query language into low level instructions that the query evaluation engine understands.
- A query can usually be translated into any of a number of alternative evaluation plans for same query result DML compiler will select best plan for query optimization.

Query evaluation engine

- This engine will execute low-level instructions generated by the DML compiler on DBMS.

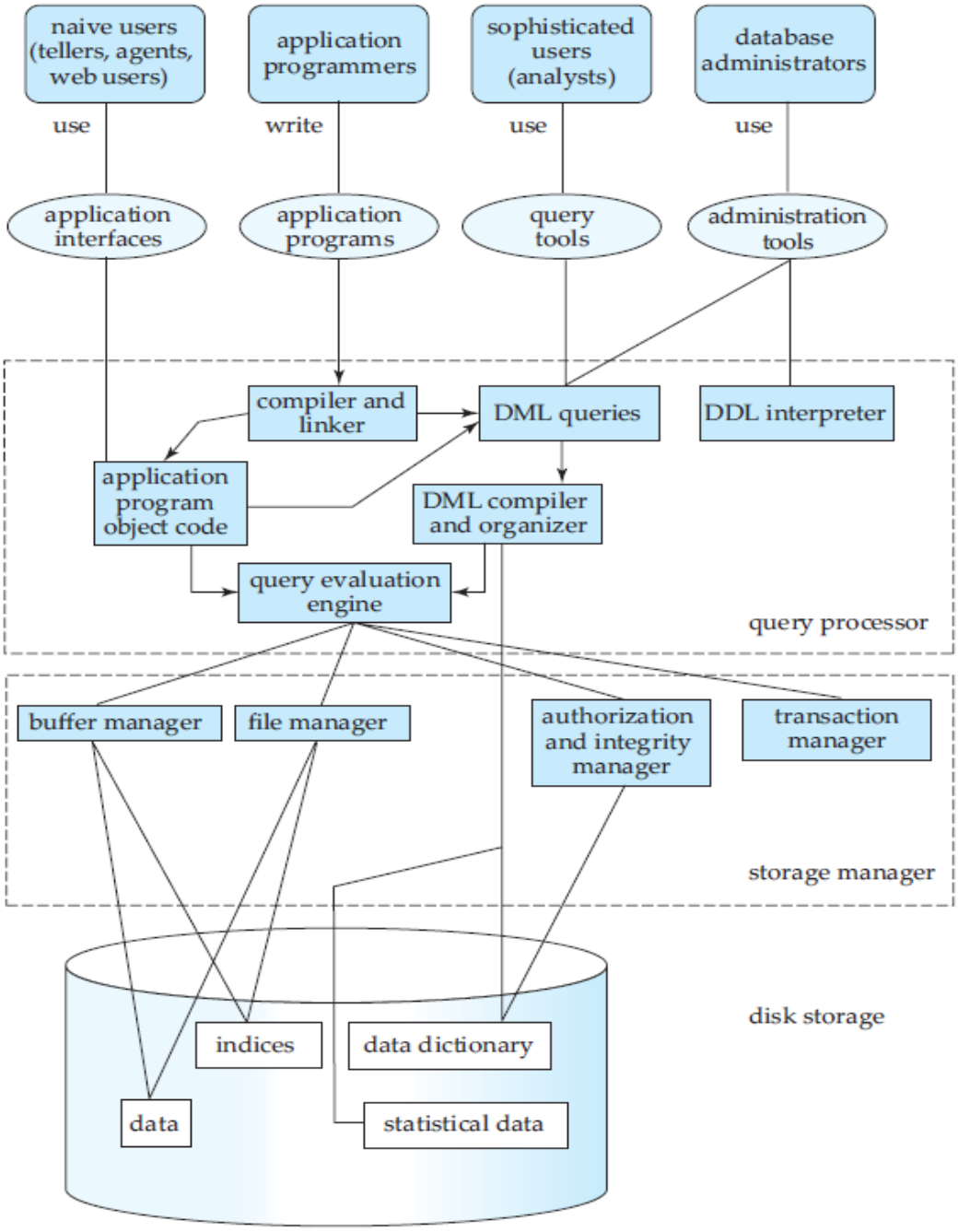
Storage Manager/Storage Management:

- A storage manager is a program module which acts like interface between the data stored in a database and the application programs and queries submitted to the system.
- Thus, the storage manager is responsible for storing, retrieving and updating data in the database.

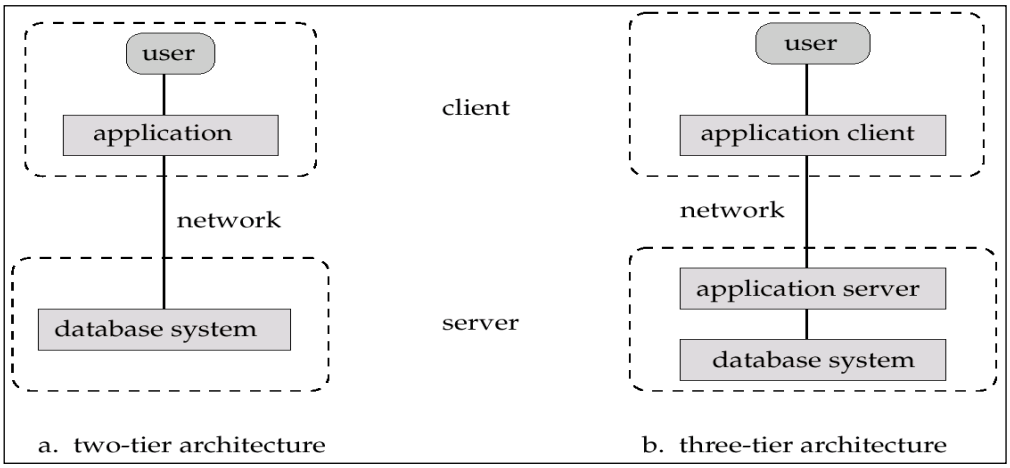
The storage manager components include:

- **Authorization and integrity manager:** Checks for integrity constraints and authority of users to access data.
- **Transaction manager:** Ensures that the database remains in a consistent state although there are system failures.
- **File manager:** Manages the allocation of space on disk storage and the data structures used to represent information stored on disk.

- **Buffer manager:** It is responsible for retrieving data from disk storage into main memory. It enables the database to handle data sizes that are much larger than the size of main memory.
- **Data structures implemented by storage manager.**
- **Data files:** Stored in the database itself.
- **Data dictionary:** Stores metadata about the structure of the database.
- **Indices:** Provide fast access to data items.



Fig, Database System Architecture



Two-Tier Architecture:

- The application resides at the client machine, where it invokes database system functionality at the server machine through query language statements.
- Application program interface standards like ODBC and JDBC are used for interaction between the client and the server.

Three-Tier Architecture:

- The client machine acts as merely a front end and does not contain any direct database calls. Instead, the client end communicates with an **application server,** usually through a forms interface.
- The application server in turn communicates with a database system to access data. The **business logic o**f the application, which says what actions to carry out under what conditions, is embedded in the application server, instead of being distributed across multiple clients. Three-tier applications are more appropriate for large applications, and for applications that run on the World Wide Web.
- E.g. web-based applications, and applications built using "middleware"

1.5 HISTORY OF DATABASE SYSTEMS:

Techniques for data storage and processing have evolved over the years:

1950s and early 1960s:

Magnetic tapes were developed for data storage. Data processing tasks such as payroll were automated, with data stored on tapes.

Data could also be input from punched card decks, and output to printers. Tapes (and card decks) could be read only sequentially, and data sizes were

much larger than main memory; thus, data processing programs were forced to process data in a particular order, by reading and merging data from tapes and card decks.

Late 1960s and 1970s:

Widespread use of hard disks in the late 1960s changed the scenario for data processing greatly, since hard disks allowed direct access to data.

With disks, network and hierarchical databases could be created that allowed data structures such as lists and trees to be stored on disk.

In the late 1960's, IBM (International Business Machines Corporation) developed the Integrated Management Systems which is the standard database system used till date in many places. It was developed based on the hierarchical database model.

In 1970:

E.F Codd in 1970 defined the r**elational model** and nonprocedural ways of querying data in the relational model and it hides implementation details completely from the programmer. The relational model was still in use by many people in the market.

1980s:

Initial commercial relational database systems called oracle was developed followed by, IBM DB2, Ingres, and DEC Rdb, all played a major role in advancing techniques for efficient processing of declarative queries.

Relational databases were so easy to use that they eventually replaced network and hierarchical databases.

Early 1990s:

Structured Query Language (SQL) was developed. It was declared as a standard language for the queries by ISO and ANSI.

The major event of the 1990s was the explosive growth of the World WideWeb. After this, Databases were deployed much more extensively than ever before.

2000s:

The first half of the 2000s saw the emerging of XML and the associated query language XQuery as a new database technology.

This period also saw a significant growth in use of open-source database systems, particularly PostgreSQL and MySQL

Late 2000's SQL databases were still extremely popular, but for those who needed scale, NoSQL had become a viable option. Google BigTable, HDFS, and Cassandra are a few examples.

1.6 DATA MODELS:

Definition :

A collection of conceptual tools for describing data, data relationships, data semantics, and consistency constraints.

A data model provides a way to describe the design of a database at the physical, logical, and view levels.

The data models can be classified into different categories are ,

- ✓ Entity-Relationship model -
- ✓ Relational model
- ✓ Other models:
- ✓ Object-oriented model
- ✓ Semi-structured data models, network model and hierarchical model

Entity-Relationship Model:

The entity-relationship (E-R) data model uses a collection of basic objects, called ***entities,*** and ***relationships*** among these objects.

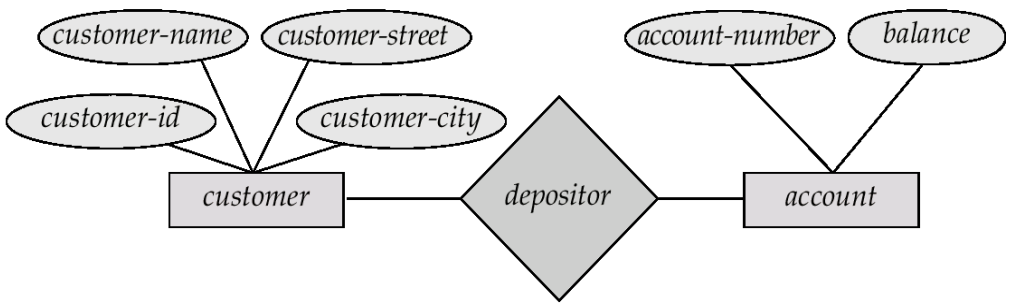
An **entity** is a "**thing" or "object"** in the real world that is distinguishable from other objects.

An entity has a real world property called **attribute** and attribute define by a set of values called **domain**. Every attribute has one piece of information about data. It is a particular property that describes it.

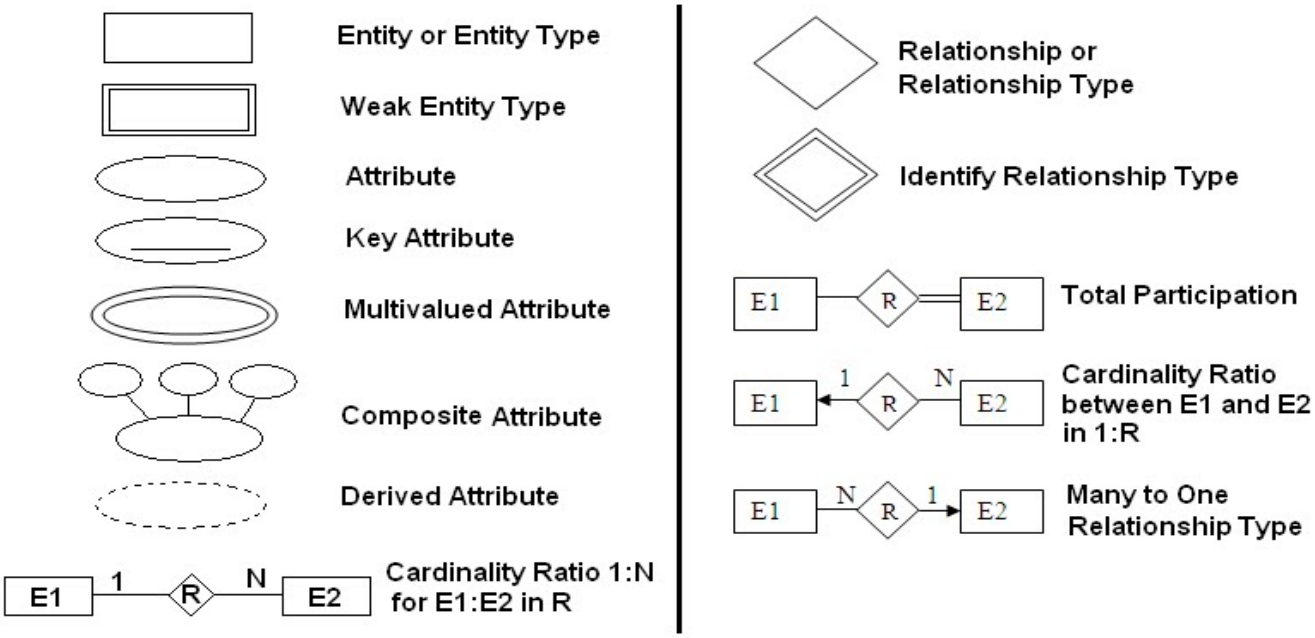
Instance: Single occurrence of an entity.

PRODUCT TYPE	Shoe
JOB	Electrician
SKILL LEVEL	Beginner

For example, in a university a *student is an entity, university is the database, name and age and sex are the attributes. Eg,* Fig, Example schema in entity relationship model.



The following Symbols or notations used in E-R diagram,



Relational Model:

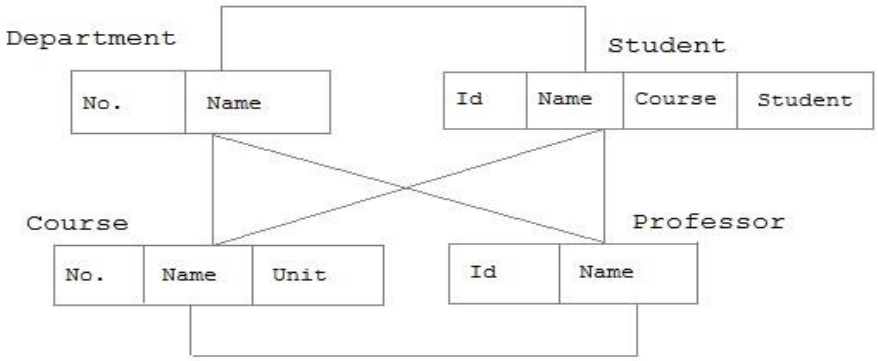
- The relational model uses **a collection of tables** to represent both data and the relationships among those data. Each table has multiple columns, and each column has a unique name. Tables are also known as **relations**. The relational model is an example of a record-based model.
- Each **record** type defines a fixed number of fields, or attributes.
- The **columns** of the table correspond to the **attributes** of the record type.
- Example of tabular data in the relational model,

Attributes

Customer-id	*customer-name*	*customer-street*	*customer-city*	*account-number*
192-83-7465	Johnson	Alma	Palo Alto	A-101
019-28-3746	Smith	North	Rye	A-215
192-83-7465	Johnson	Alma	Palo Alto	A-201
321-12-3123	Jones	Main	Harrison	A-217
019-28-3746	Smith	North	Rye	A-201

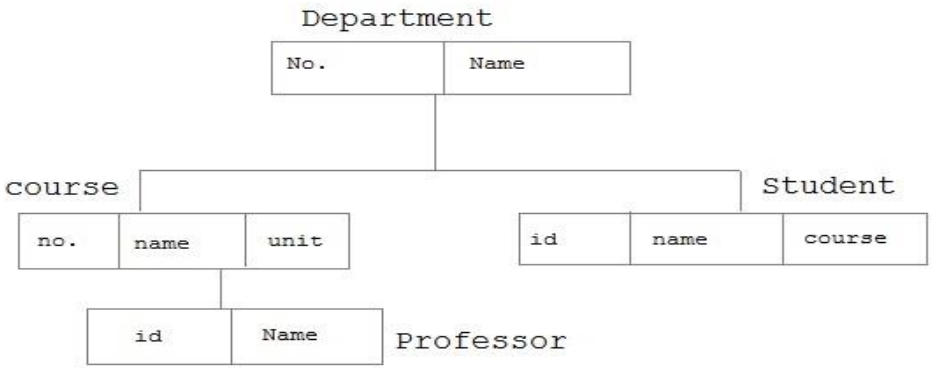
Network Model:

In the network model, entities are organized **in a graph,** in which some entities can be accessed through several paths.



Hierarchical Model:

- The hierarchical data model organizes data in **tree structure.**
- Collection of records logically organized to conform to the upside-down tree (hierarchical) structure.
- In this model each entity has only one parent but can have several children. At the top of hierarchy there is only one entity which is called **Root**.



Object-Based Data Model:

Object-oriented programming (especially in Java, C++, or C#) has become the dominant software-development methodology. This led to the development of an object-oriented data model that can be seen as extending the E-R model with notions of encapsulation, methods (functions), and object identity.

The object-relational data model combines features of the object-oriented data model and relational data model .

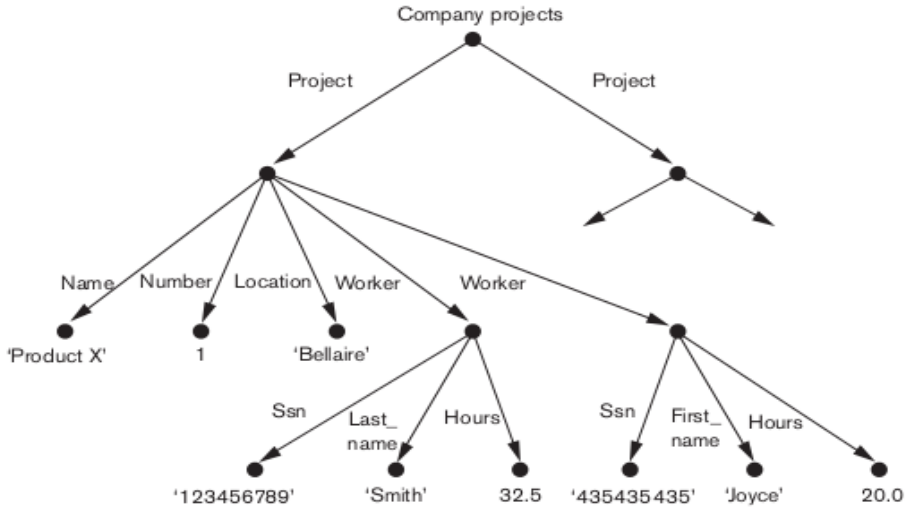
Semi structured Data Model:

- The semi structured data model permits the specification of data where individual data items of the same type may have different sets of attributes.

This is in contrast to the data models mentioned earlier, where every data item of a particular type must have the same set of attributes.

- The **Extensible Mark-up Language (XML)** is widely used to represent semi structured data.

For example,



1.7 ER MODEL(ENTITY-RELATIONSHIP MODEL)

- ER model stands for an Entity-Relationship model. It is a high-level data model.
- This model is used to define the data elements and relationship for a specified system.
- The entity-relationship (E-R) is **based on a perception of a real world** that consists of a collection of basic objects called **entities, attributes of entities and relationships** among these objects.
- It represents the overall logical structure of a database.

ER DIAGRAM:

Graphical components of ER Diagram consists of

- Entities
- Attributes
- Relationships

- **An entity** - is a "thing" or "object" in the real world.
- An *entity* is an object that exists and is distinguishable from other objects. An entity is described (in DB) using a set of **attributes.**
- An **entity** is a "**thing" or "object"** in the real world that is distinguishable from other objects.

Instance: Single occurrence of an entity.

Entities	Instances
PERSON	Mahatma Gandhi
PRODUCT	Nike Air Jordan
PRODUCT TYPE	Shoe
JOB	Electrician
SKILL LEVEL	Beginner
CONCERT	U2 at the Palladium
ANIMAL	Dog
CAR	Volkswagen Beetle

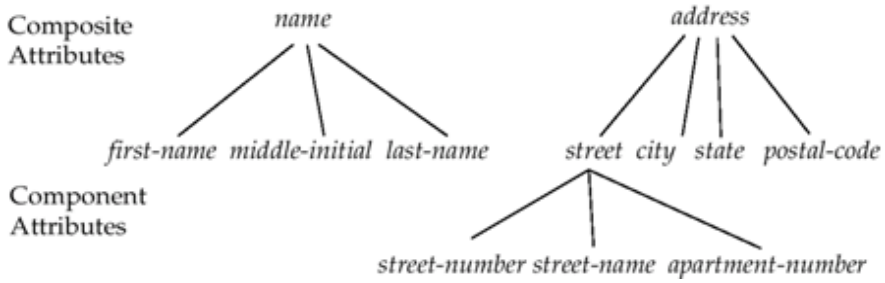
Attributes:

- Entities have ***attributes.***
- Every attribute has one piece of information about data. It is a particular property that describes it.

Entities	Attributes
CUSTOMER	family name, age, shoe size, town of residence, email
CAR	model, weight, catalog price
ORDER	order date, ship date
JOB	title, description
TRANSACTION	amount, transaction date
EMPLOYMENT CONTRACT	start date, salary

Types of Attributes:

❖ **Composite attribute**: can be divided into sub parts.



❖ **Simple attribut**e: attributes that are not divisible. Eg, customer ID

❖ **Single valued attribute:** Attributes that have single value for particular entity. Eg, Age.

❖ **Multi valued Attribute** : Attributes that have multiple values are called **multi valued attribute**. Eg, College_degree. (a person can have many degrees**).**

❖ **Stored attribute & Derived attribute:**

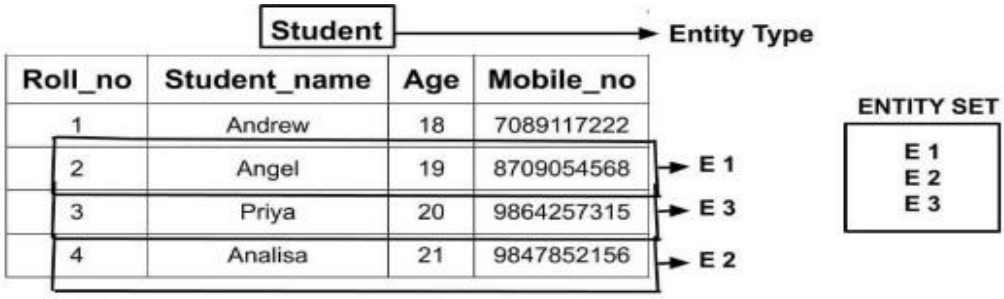
- ❖ The Age attribute is hence called a **derived attribute** and is said to be **derivable from** the BirthDate attribute, which is called a **stored attribute.**
- ❖ **Null Value:** In some cases a particular entity may not have an applicable value for an attribute. For example, the Apartment_Number attribute of an address

Entity Type :

- It is a collection(set) of entities that have the same attributes.
- Eg, EMPLOYEE is an entity set of SECRETARY, TECHNICIAN, MANAGER.

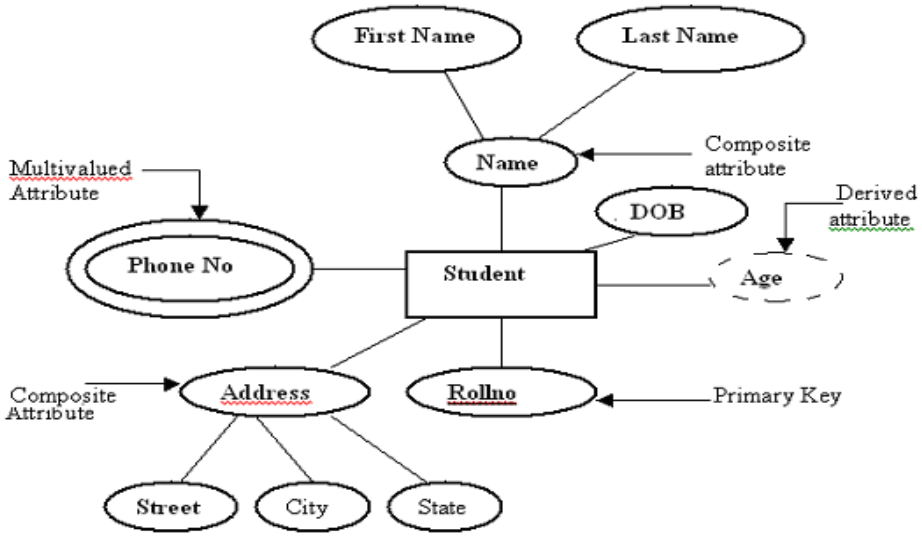
Entity Set:

- The collection of all entities of a particular entity type in the database at any point in time is called an **entity set.**



Key attributes of an Entity Type:

- An entity type usually has an attribute **whose values are distinct for each individual** entity in the collection.
- Eg, For the PERSON entity type, a typical key attribute is SocialSecurityNumber.
- In ER diagrammatic notation, each key attribute has its name underlined inside the oval.



Fig, ER Diagram representing composite multi valued and derived attribute

Relationship Set& Relationship Type:

Relationship: Attribute of one entity type refers to another entity type. Represent references as relationships not attributes. It is represented by **diamonds.**

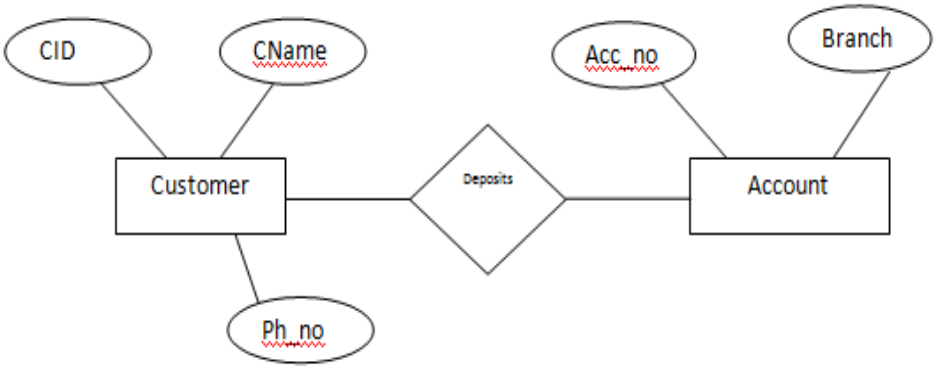
Relationship type R among n entity types E1, E2, ..., En defines a set of associations or relationship set among entities from these entity types.

In ER diagram, Relationship types are displayed as diamond shaped boxes which are connected by straight lines to a rectangular boxes representing the participating entity types.

Degree of a relationship type:

The degree of a relationship type is the **number of participating entity types.** If the degree is two, it is called binary. If one, it is called ternary.

Relationship Sets with Attributes:



Roles:

- The labels "employee" and "employer" are called roles; they specify how Person and company entities interact via the works-for relationship set.
- Roles are indicated in E-R diagrams by labelling the lines that connect diamonds to rectangles.
- Role labels are optional,.

Types of Relationship :

Relationship type	No of Entity
Unary	1

Binary	**2**
Ternary	**3**
N-Ary	N

1.Unary relationship: A Unary relationship exists when an association is maintained **with in a single entity.**

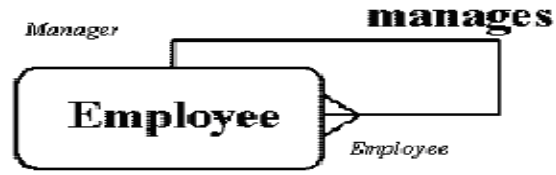
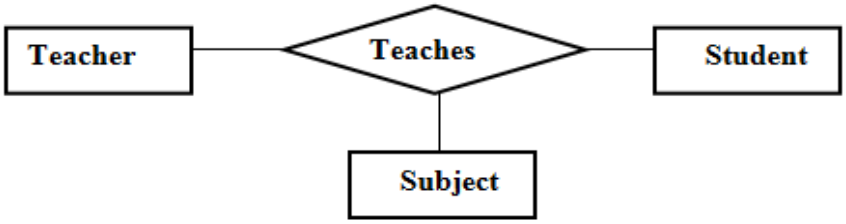


Fig: Association between two objects of the same entity set

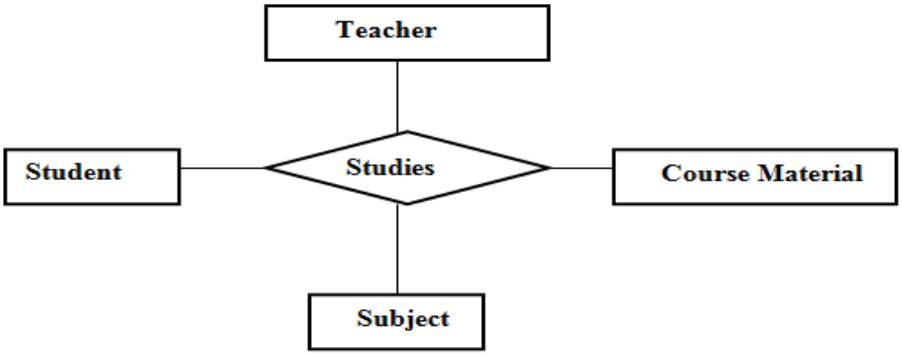
2.Binary Relationship: A binary relationship exists when **two entities** are associated. For Example ,the **Book-Publisher** relationship in the following fig is a binary relationship.

3.Ternary Relationship :A ternary relationship exists when there **are three entities associated .**

For example ,the entities teacher, subject and student are related using a ternary relationship called teaches as shown



4.Quaternary relationship: A quaternary relationship exists **when there are four entities associated**. An example of quaternary relationship is studies where 4 entities are involved –student, teacher, subject and course-material.



Cardinality Constraints:

Cardinality measures the quantity of something.

We express cardinality constraints by drawing either a directed line (→), signifying

"one," or an undirected line (—), signifying "many," between the relationship set and the entity set.

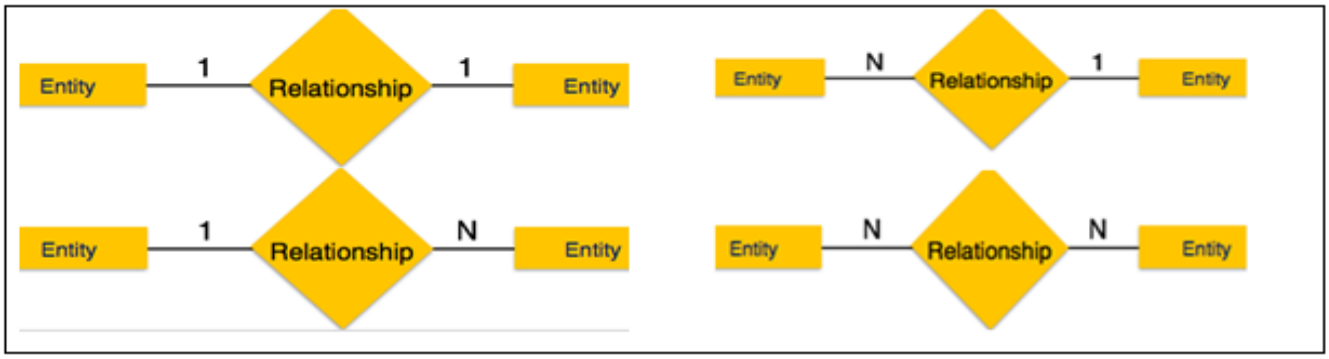
Constraints:

❖ **1:1(One to One)-** An entity in A is associated with at most one entity in B, and an entity in B is associated with at most one entity in A.

❖ **1:N (One to Many)** An entity in A is associated with any number (zero or more) of entities in B. An entity in B, however, can be associated with at most one entity in A.

❖ **N:1(Many to One)** An entity in A is associated with at most one entity in B. An entity in B, however, can be associated with any number (zero or more) of entities in A.

❖ **M:N(Many to Many)** An entity in A is associated with any number (zero or more) of entities in B, and an entity in B is associated with any number (zero or more) of entities in A.



Example:

One to one. (1:1)

One department has only one head and one head can be the head of only one department. Hence the cardinality is one to one. (1:1)

One to Many (1:N)

One department has multiple instructors. But instructor belongs to only one department. Hence the cardinality between department and instructor is One to Many (1:N)

Many to One (N :1)

One course is taught by only one instructor. But one instructor teaches many courses. Hence the cardinality between course and instructor is Many to One (N :1)

Many to Many (M:N)

One course can be enrolled by many students and one student can enroll for many courses. Hence the cardinality between course and student is Many to Many (M:N)

Participation of an Entity Set in a Relationship Set:

Total participation(also called Existence Dependency):

- Every entity in the entity set participates **in at least one relationship** in the relationship set. .It is indicated by double line==========)
- E.g. every entity in total set of employee entities must be related to a department entity via WORKS_FOR relationship type.

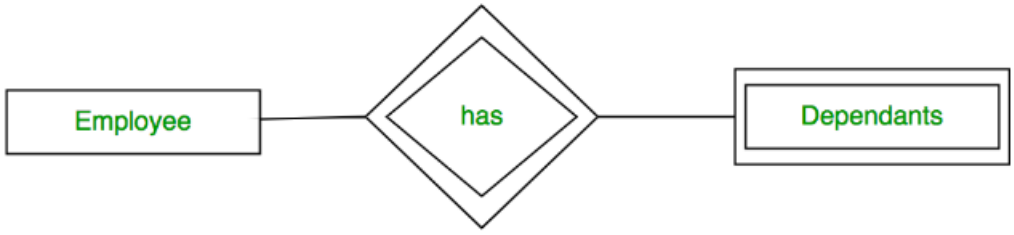
Partial participation:

- **Some entities may not participate in any relationship** in the relationship set.

E.g. we donot expect every employee to manage a department, so the participation of EMPLOYEE in MANAGES relationship type is partial.

Weak Entity Sets:

- An entity set that does not have key attributes of their own are called week entity set.
- It is represented using double rectangular box. Here, in the example dependent will not have any primary key to uniquely identify data.



- **Regular entity types** that do have a key attribute are called **strong entity types.**

Example 1: An ER schema Diagram for COMPANY Database :

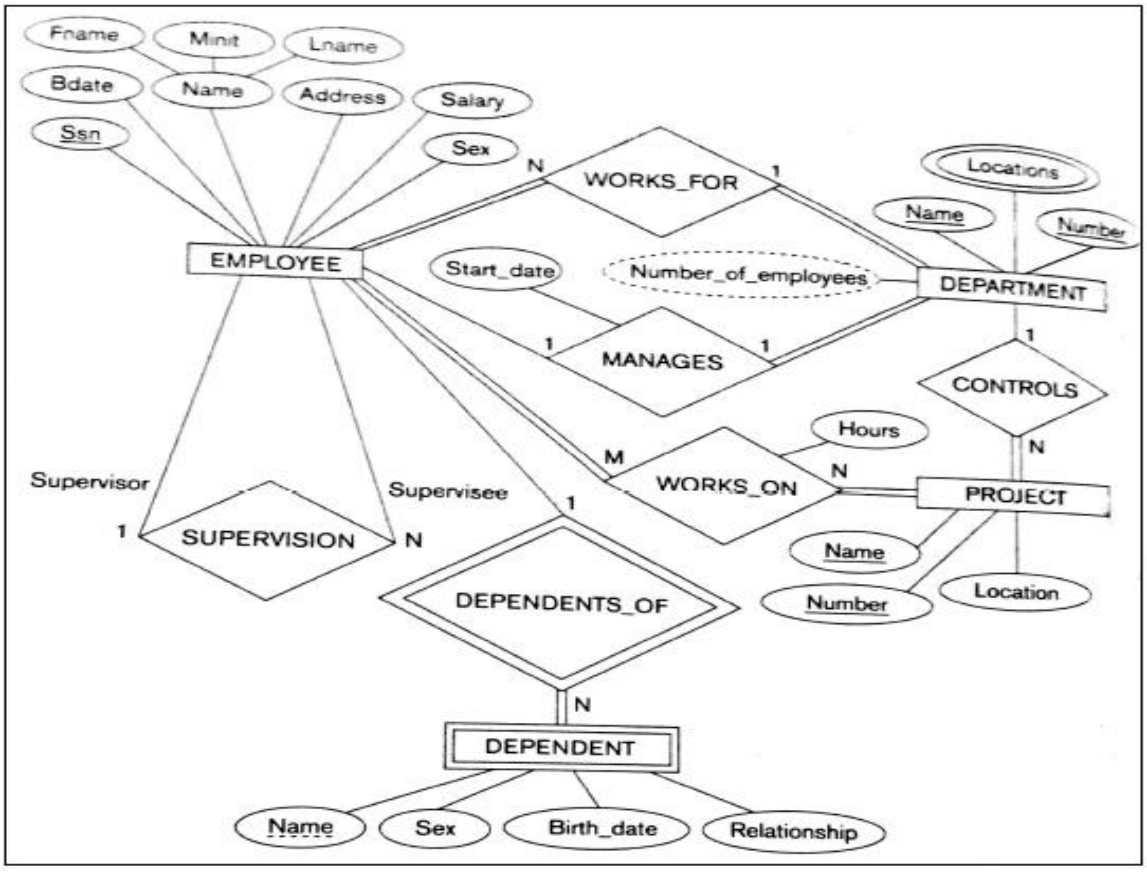
The COMPANY database keeps track of a company's employees, departments, and projects.

❖ 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.

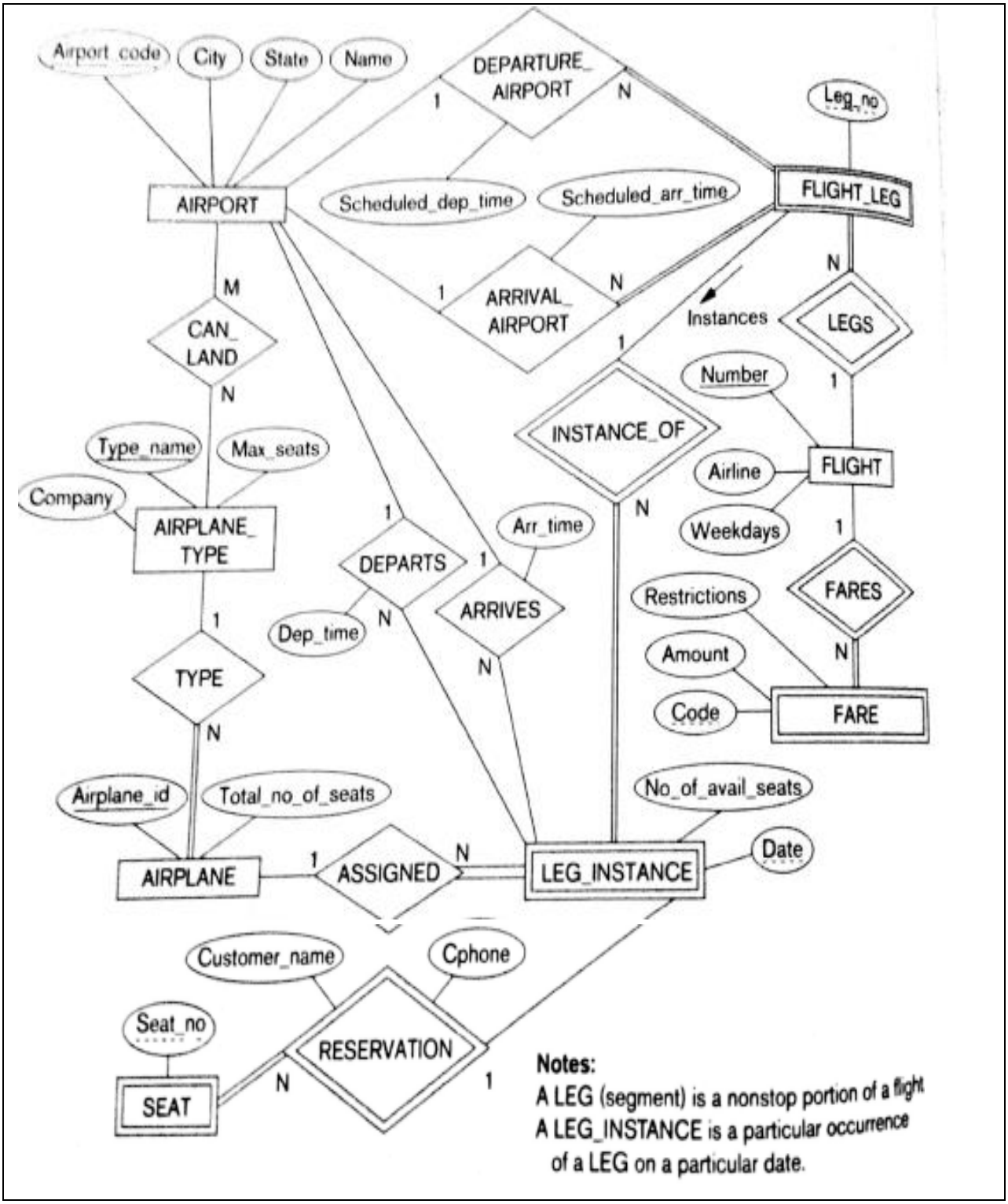
❖ A department controls a number of projects, each of which has a unique name, a unique number, and a single location.

❖ We store each employee's name, social security number (Note 1), address, salary, sex, 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 number of hours per week that an employee works on each project. We also keep track of the direct supervisor of each 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.



Example 2: E-R Diagram for a AIRLINE Database Schema:



Example 3:

ER Diagram for **University Database:**

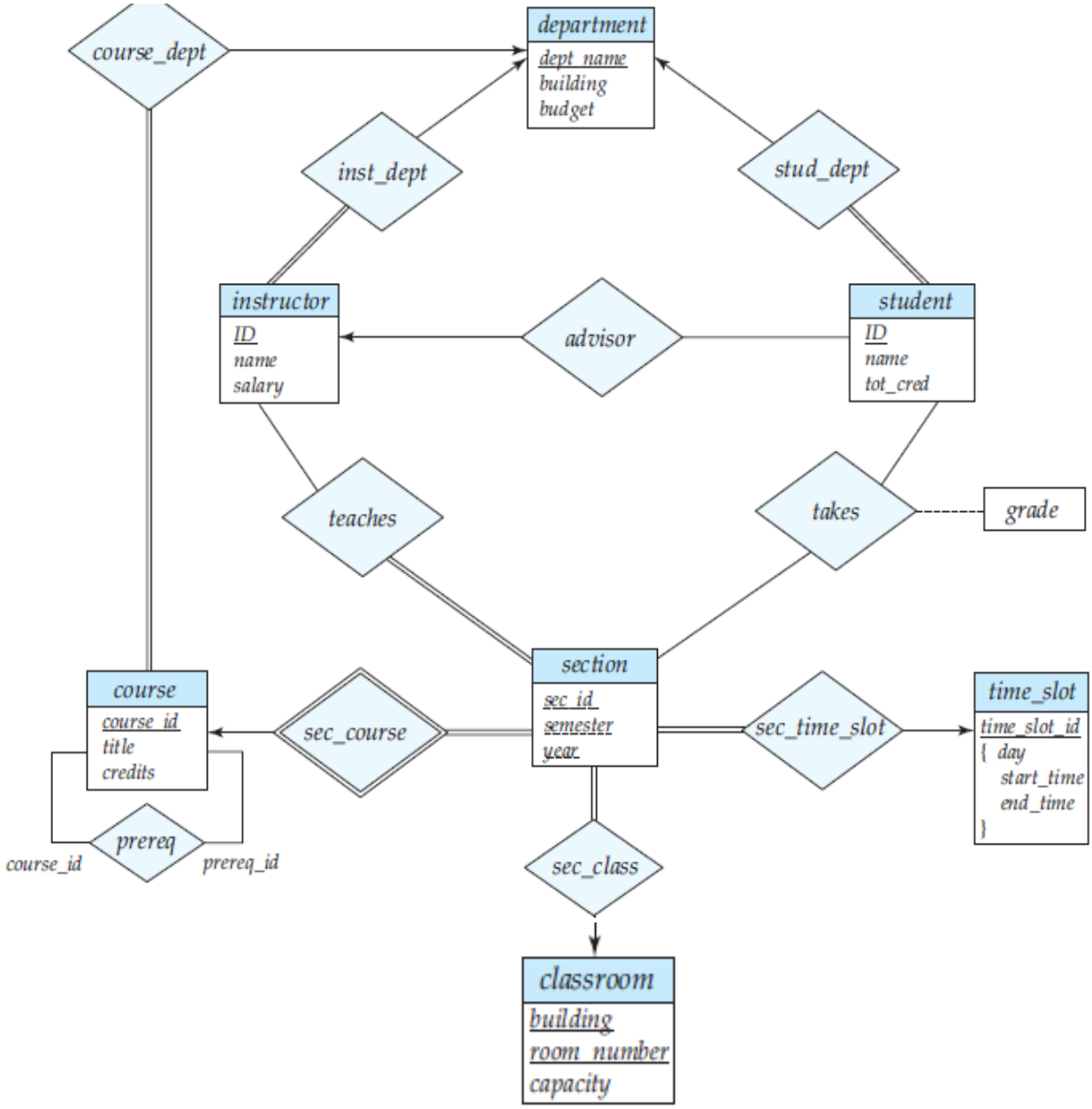
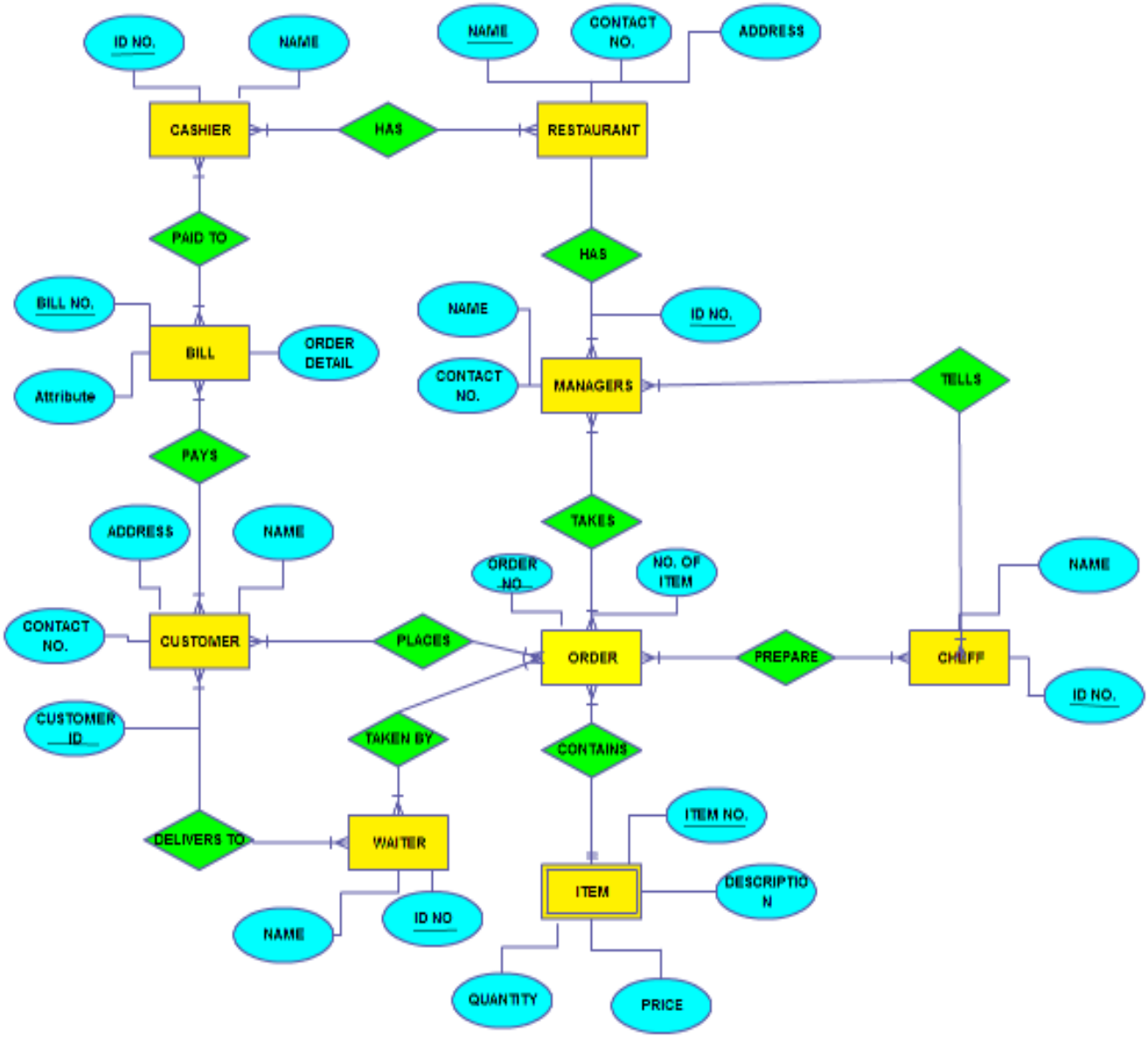


Figure E-R diagram for a university enterprise.

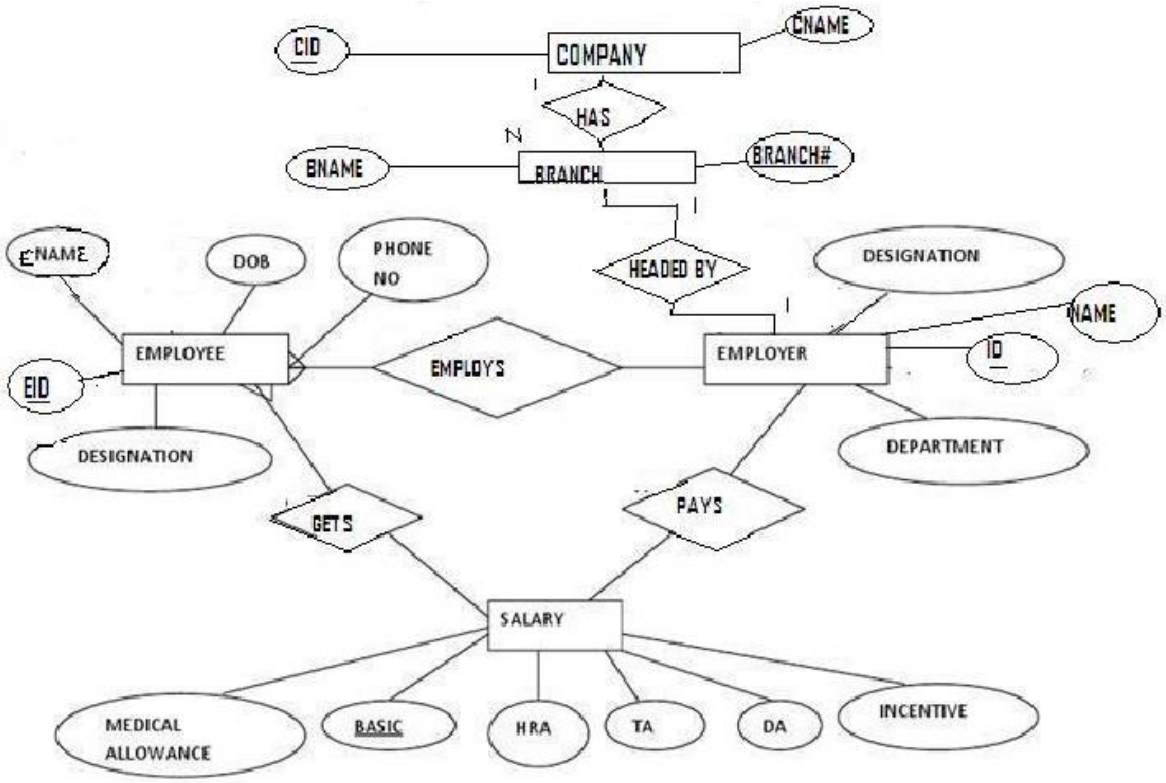
Example 4:

Draw ER Diagram for **" Restaurant menu ordering system**", which will facilitate the food items ordering and services within restaurant. The entire scenario is detailed below. The customer is able to view the food items menu, call the waiter, place orders and obtain final bill through the computer kept in their table. The waiters through their wireless tablet PC are able to initialize a table for customers , control the table functions to assist customers , orders, send orders to food preparation staff(chef) and finalize the customers bill.

The chef with their touch display interface to the system, are able to view orders sent to the kitchen by waiters. During preparation they are able to let the waiter know the status of each item , and can send notifications when items are completed. The system should have full accountability and logging facilities and should support supervisor actions to account for exceptional circumstances such as meal being refunded or walked out on. **(M/J Soln:**

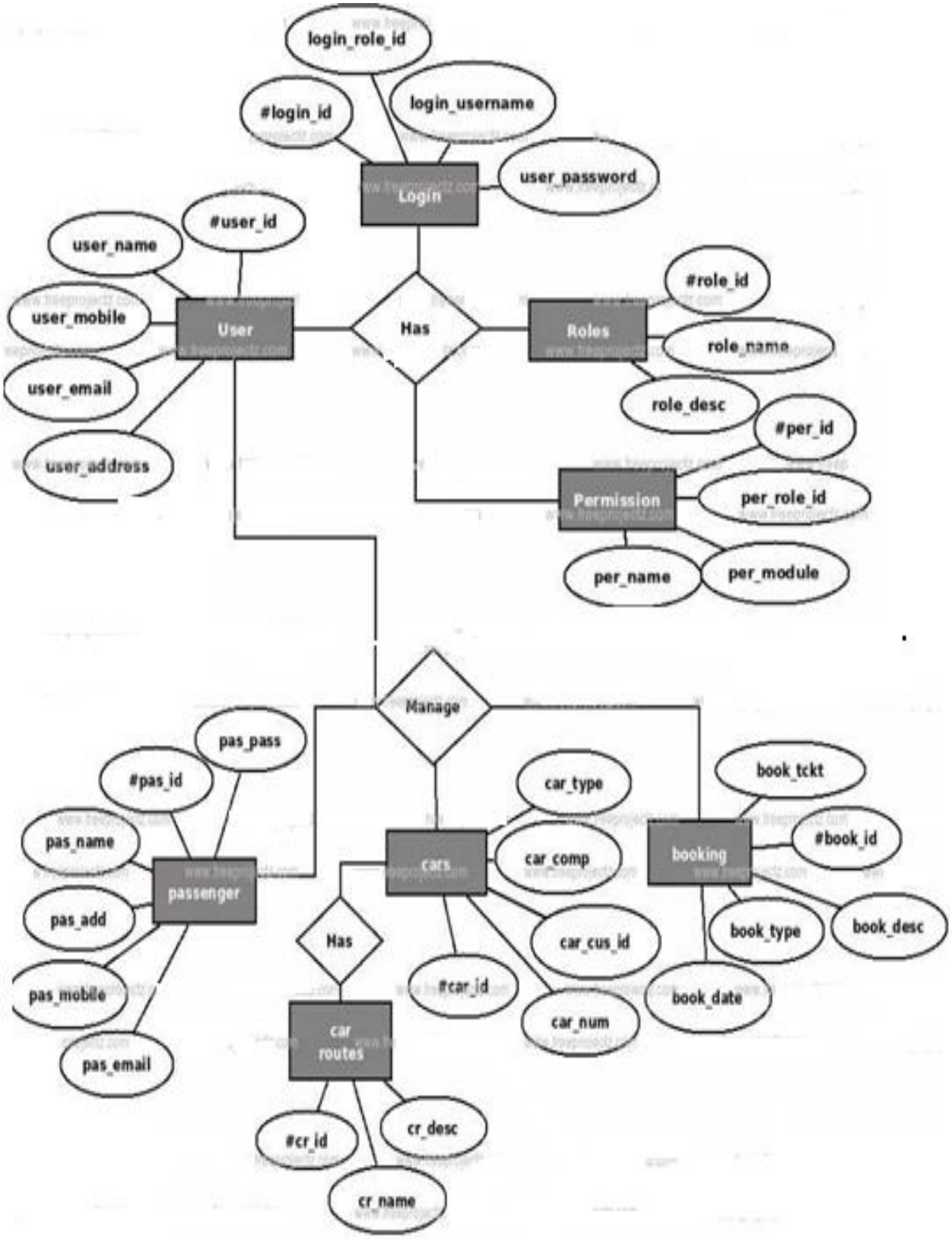


Example: 5 ER Diagram for **Employee Payroll system**: **. (**April/May 2010)



Example 6: ER Diagram for **Insurance company:**

Example 7: ER model for **car rental company** database. (Nov/Dec'15)



1.8 EXTENDED ER DIAGRAM:

Extended ER incorporates the concepts of ***Super class/subclass relationships*** , ***Generalization, Specialization*** and ***type inheritance*** in the existing ER model.

Features:

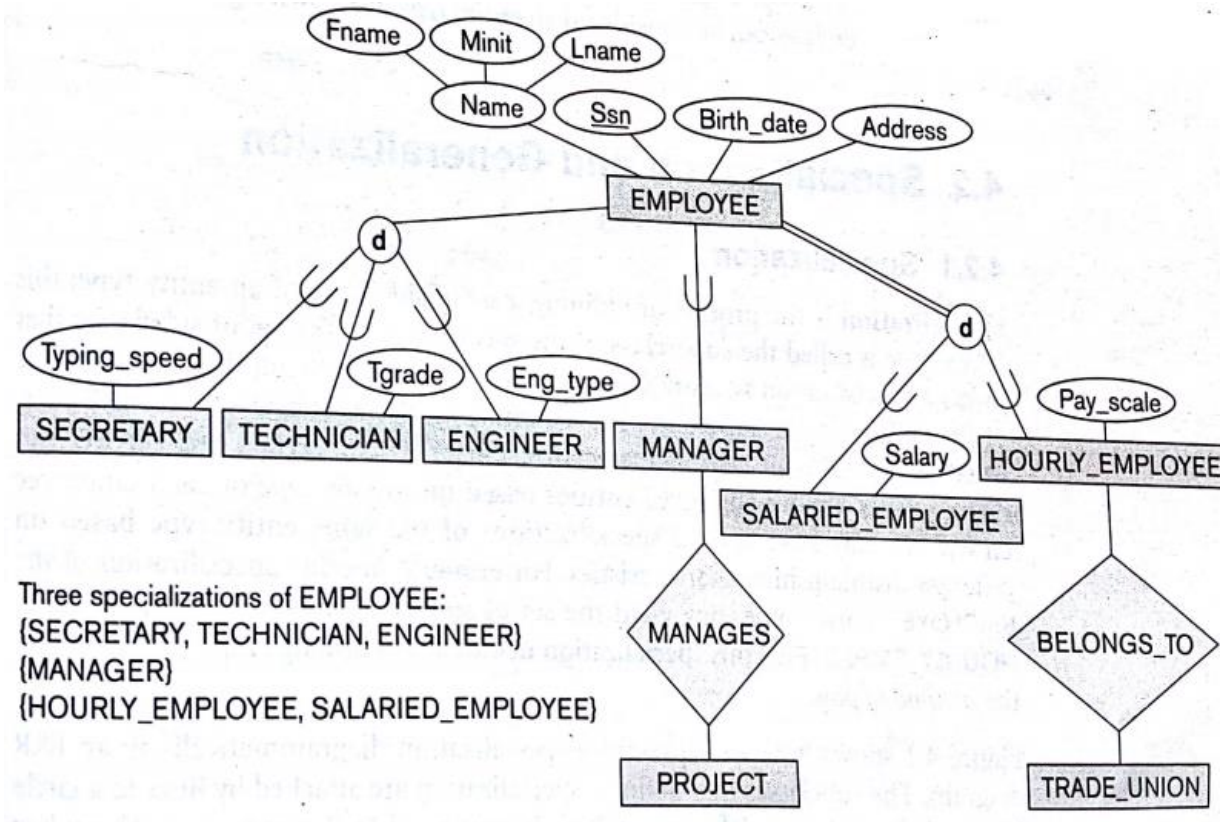
(i) Subclass & Super class:

- An entity type is used to represent both a type entity and the entity set or collection of that type exists in the database.
- For example, the entities that are members of the EMPLOYEE entity type may be grouped further into SECRETARY, ENGINEER, MANAGER, TECHNICIAN, SALARIED_EMPLOYEE, HOURLY_EMPLOYEE, and so on.

We call each of the subgroupings as a **subclass** of the EMPLOYEE entity type, and the EMPLOYEE entity type is called the **superclass** for each of these subclasses.

i,e EMPLOYEE Entity is a super class and SECRETARY,TECHNICIAN & ENGINEER are SubClass.

Relationship between a superclass and any one of the subclass is called a superclass/subclass or simply class/subclass relationship.



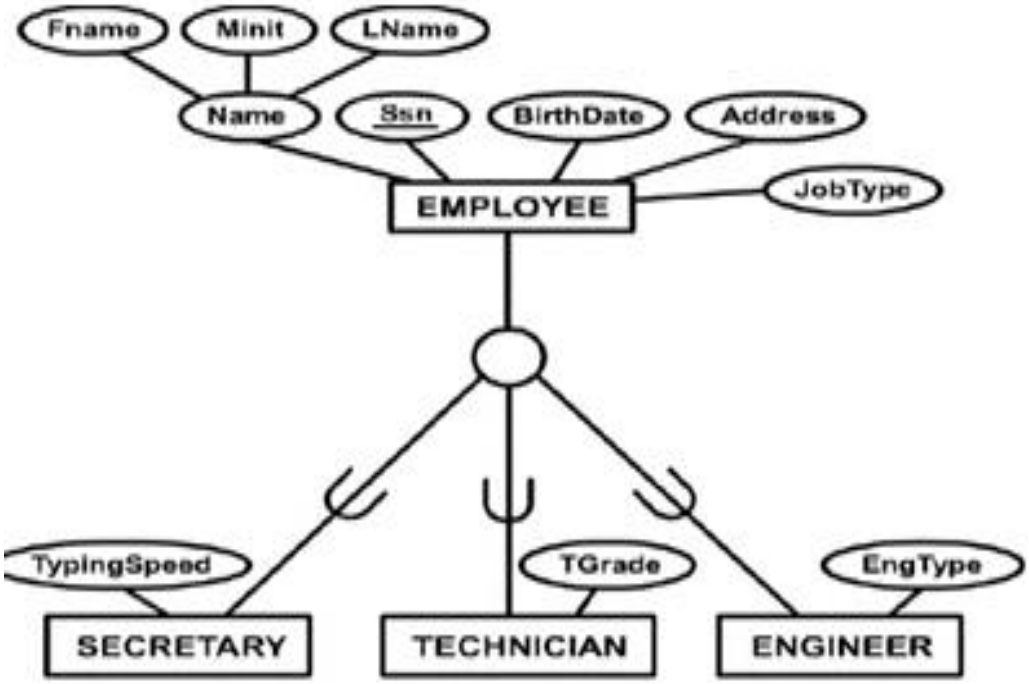
Fig, EER Diagram to represent Subclass and specialization

(ii) Type Inheritance:

- All entity that is a member of a subclass inherits all the attributes of the entity as a member of the superclass.

(iii) Specialization:

- **Specialization** is **a top-down process.**
- **Specialization** is the process of defining a *set of subclasses* of an entity type; this entity type is called the **superclass** of the specialization.
- For example, the set of subclasses {SECRETARY, ENGINEER, TECHNICIAN} is a specialization of the superclass EMPLOYEE that distinguishes among EMPLOYEE entities based on the *job type* of each entity.
- Following diagram shows how we represent a specialization diagrammatically in an **EER diagram.**



Fig, EER Notation for specialization on Job Type.

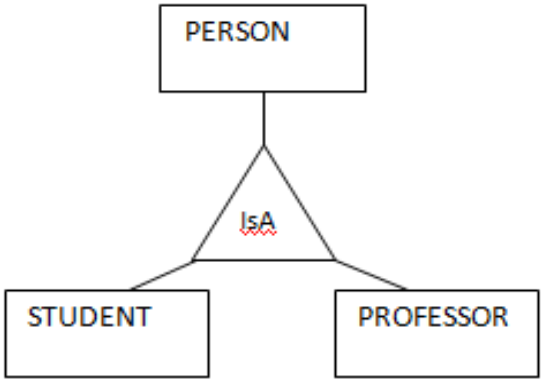
The subclasses that define a specialization are attached by lines to a circle(SECRETARY,TECHNICIAN,ENGINEER), which is connected to the superclass(EMPLOYEE).

The ***subset symbol*** $\in$ on each line connecting a subclass to the circle indicates the direction of the superclass/subclass relationship.

Attributes that apply only to entities of a particular subclass—such as TypingSpeed of SECRETARY—are attached to the rectangle representing that subclass. These are called **specific attributes** (or **local attributes**) of the subclass.

specialization is represented by triangle components labeled ISA

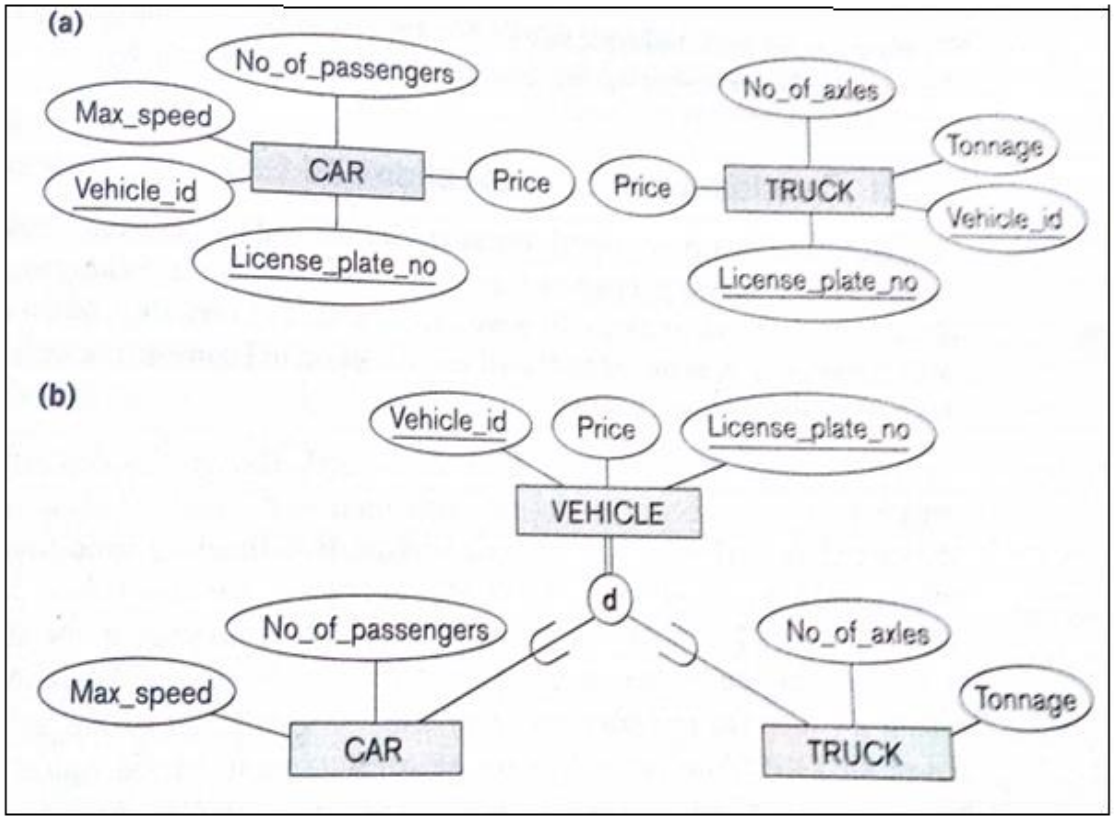
EER IsA Notation



The ISA relationship also referred to as **superclass - subclass** relationship

(iv)Generalization:

- It is a result of computing union of two or more entity sets to produce a higher level entity set.
- It is **a bottom up design process.**
- It converts subclasses to superclasses. This process combines a number of entity sets that share the same features into higher-level entity sets.

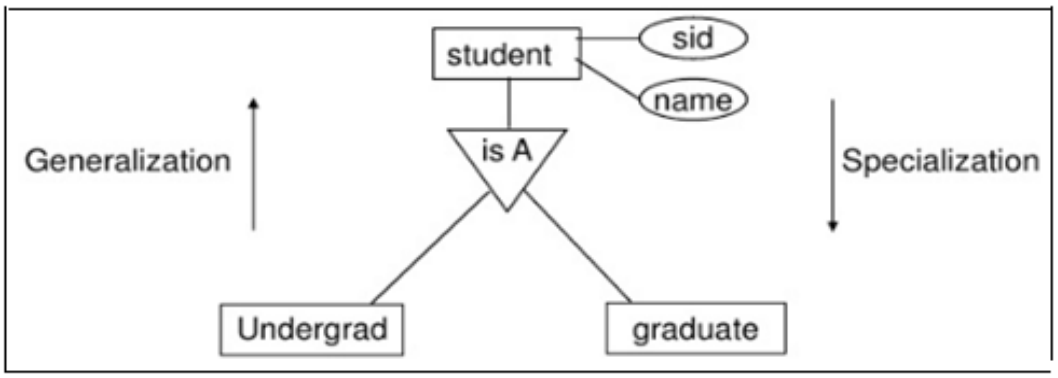


(a) Two entity types CAR and TRUCK

(b) (b) Generalizing CAR and TRUCK in to SuperClass VEHICLE

(V)Generalization and Specialization:

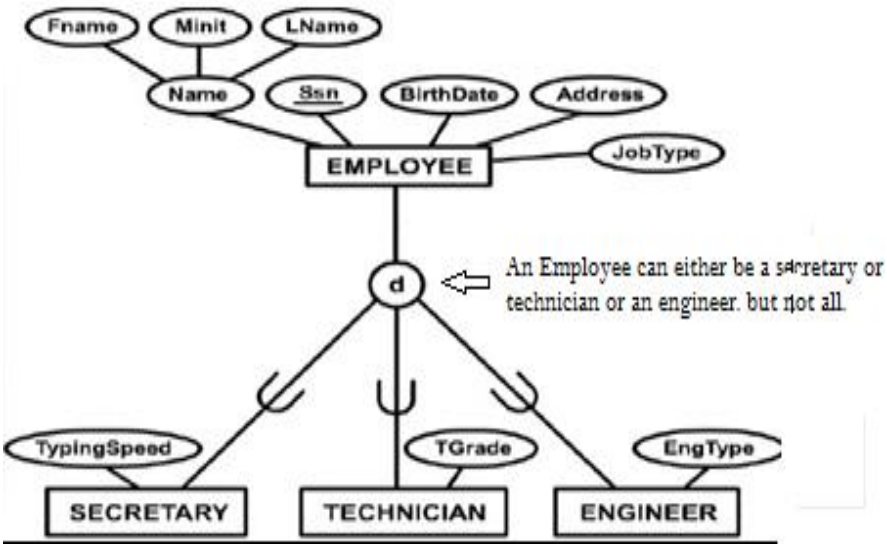
An arrow pointing to the generalized superclass represents a generalization, whereas arrows pointing to the specialized subclasses represent a specialization.



Constraints on Specialization/Generalization:

Disjointness constraint:

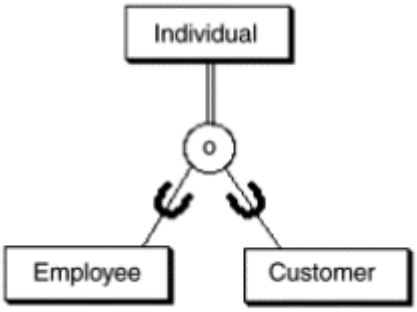
- This means that an entity can be a member of *at most one(maximum one)* of the subclasses of the specialization.
- **d** in the circle stands for disjoint.



NonDisjoint (Overlap):

If the subclasses are not constrained to be disjoint, their sets of entities may **overlap;** that is, the same (real-world) entity may be a member of more than one subclass of the specialization.

This case, which is the default, is displayed by placing an **o** in the circle. Here, Individual can be employee

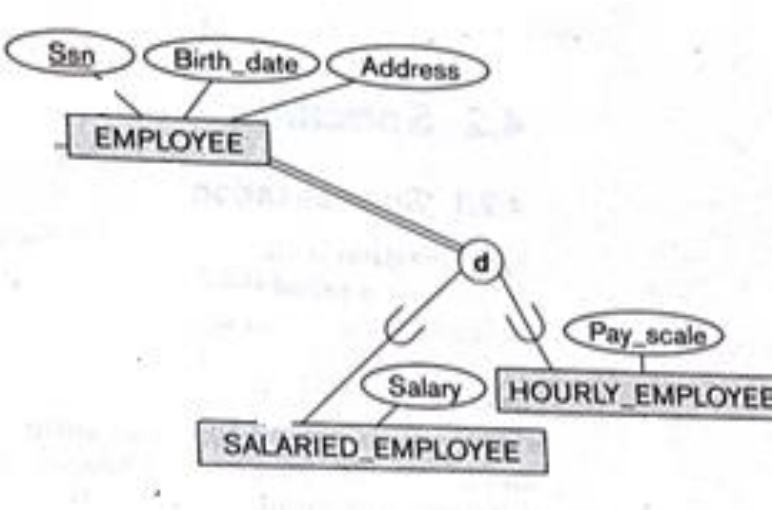


Fig, EER Diagram notation for overlapping specialization

Completeness Constraint :

i) Total Specialization: (double line)

- *Every* entity in the superclass must be a member of some subclass in the specialization.
- For example, if every EMPLOYEE must be either an HOURLY_EMPLOYEE or a SALARIED_EMPLOYEE, then the specialization {HOURLY_EMPLOYEE, SALARIED_EMPLOYEE} is a total specialization of EMPLOYEE;
- This is shown in EER diagrams by using a double line to connect the superclass to the circle.

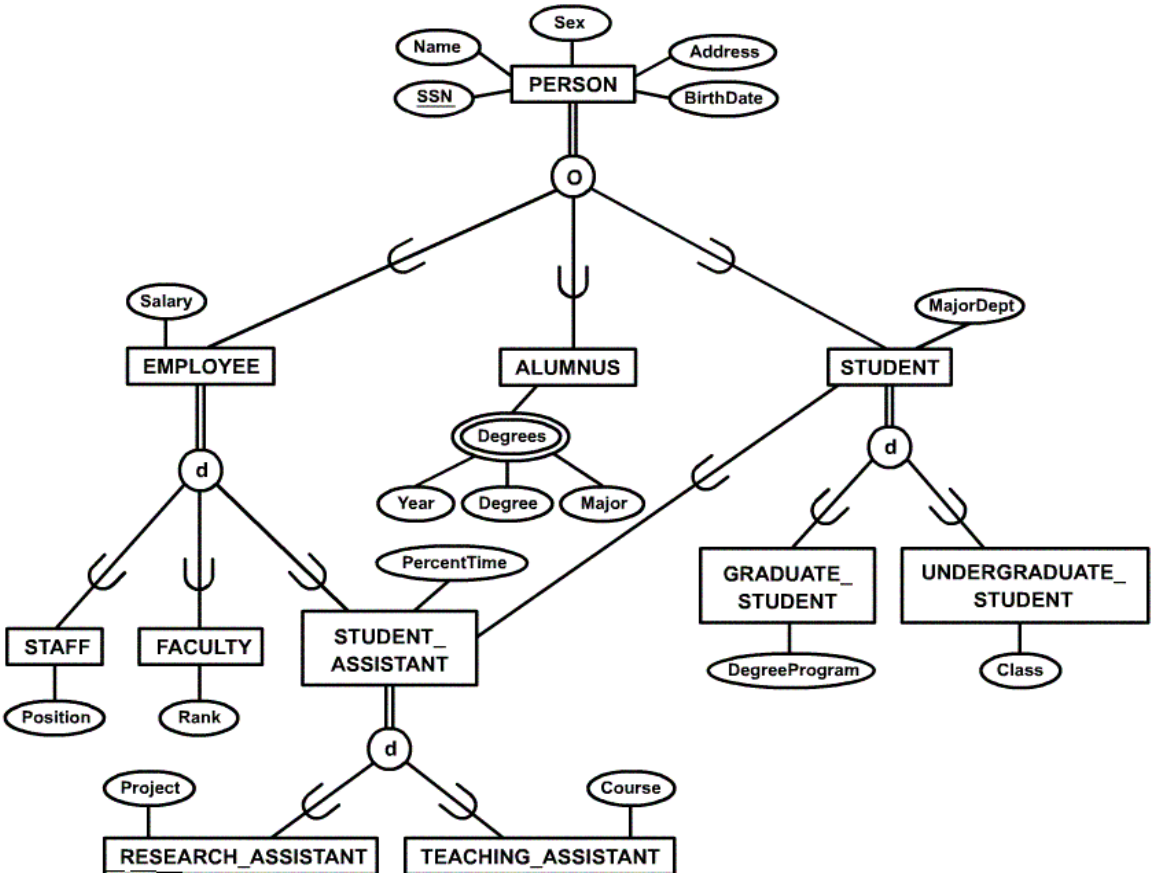


Fig, Total Specialization

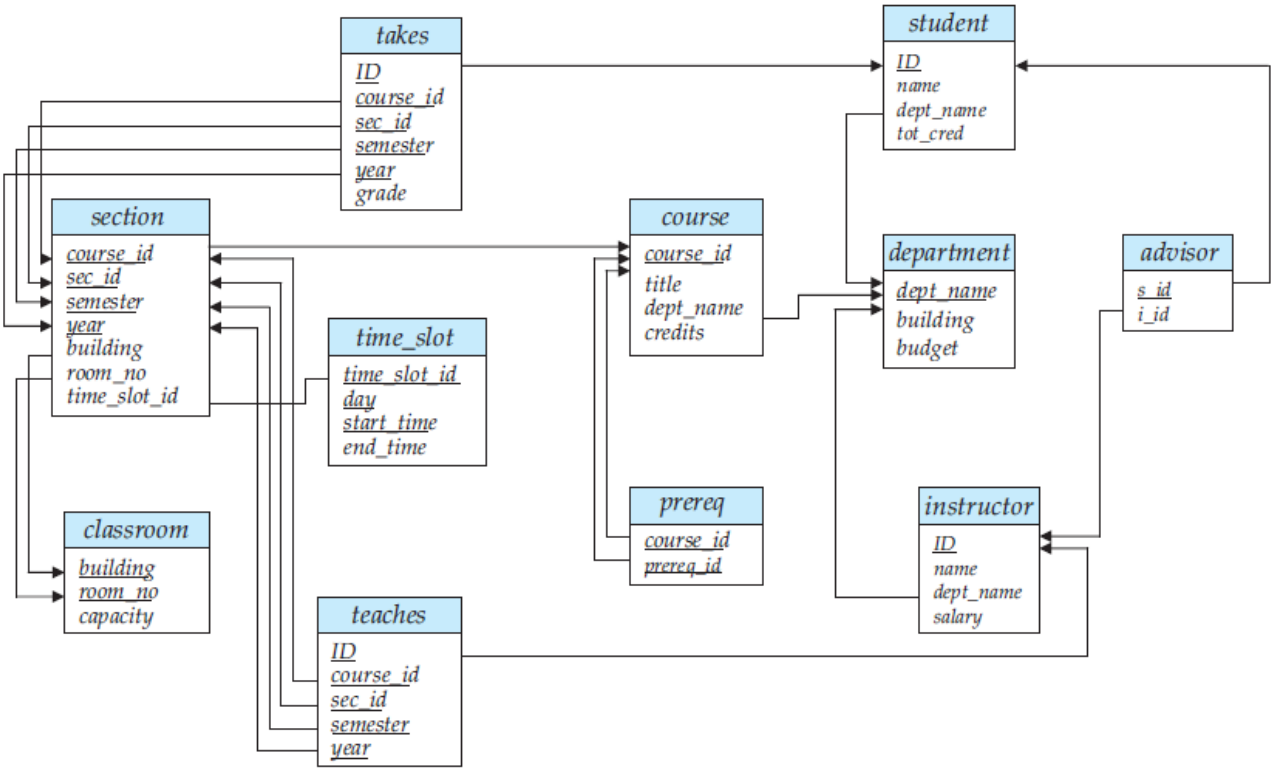
ii)Partial Specialization (single line)

- A single line is used to display a **partial specialization,** which allows an entity not to belong to any of the subclasses.
- For example, if some EMPLOYEE entities do not belong to any of the subclasses {SECRETARY, ENGINEER, TECHNICIAN} then that specialization is partial.

Example: Extented ER Diagram For UNIVERSITY Database:



Example 1: , Schema diagram for the university database,



Eg,2 : Schema for Airline Reservation System:

AIRPORT

Airport_code	Name	City	State

FLIGHT

Flight_number	Airline	Weekdays

FLIGHT_LEG

Flight_number	Leg_number	Departure_airport_code	Scheduled_departure_time
		Arrival_airport_code	Scheduled_arrival_time

LEG_INSTANCE

Flight_number	Leg_number	Date	Number_of_available_seats	Airplane_id
	Departure_airport_code	Departure_time	Arrival_airport_code	Arrival_time

FARE

Flight_number	Fare_code	Amount	Restrictions

AIRPLANE_TYPE

Airplane_type_name	Max_seats	Company

CAN_LAND

Airplane_type_name	Airport_code

AIRPLANE

Airplane_id	Total_number_of_seats	Airplane_type

SEAT_RESERVATION

Flight_number	Leg_number	Date	Seat_number	Customer_name	Customer_phone

****

PART A

1. What is the purpose of Database management system? [Nov/Dec'14]

- Database management systems were developed to handle the following difficulties of typical file processing systems.
- Data redundancy and inconsistency
- Difficulty in accessing data
- Data IsolationDCL
- Enforcing Integrity constraints
- Atomicity problems:
- Concurrent access or Sharing of Data
- Security
- Multiple Views of data

2. Write the characteristics that distinguishes database system from file system. [May/June '15] / List anyfour significant differences between a file processing system and DBMS[April/May'12]/ Compare the features of file system with database system.[April/May'11][Nov/Dec'16]

Data Base System	File System
Data can be Shared.	Data cannot be shared
Redundancy can be reduced	Data duplication is allowed.
Flexible to access data.(ie queries)	Pre-determined access to data(ie compiled programs)
Multi users accessing the same data at same time	No two programs can concurrently access same file
Expensive	Less Expensive
Inconsistency can be avoided	Inconsistency cannot be avoided
Data Integrity is maintained	Data Integrity is not maintained.
Standards can be enforced	No Standards can be maintained
More Secure-(Restricting Un authorized access)	Less Security
Backup and Recovery is possible	Not possible
Required Skilled Administrators to maintain.	More knowledge is not required.
It is suitable for medium to large sized organizations	It is suitable for small organizations

3. What are the disadvantages of file processing system? [May/June '16]

- Data redundancy and inconsistency
- Difficulty in accessing data
- Data Isolation

- Enforcing Integrity constraints
- Atomicity problems:
- Concurrent access or Sharing of Data
- Security
- Multiple Views of data

4. **Explain the basic structure of a relational database with an example. [April/May '2010]**
   - Relational databases store data in the form of tables (logically).
   - The rows of a table are called as ***tuples.***
   - The columns of a table are known as attributes.
   - Tables are called as ***relations.***
   - For eg, EMPLOYEE_ID,FIRST_NAME, LAST_NAME,DEPARTMENT_ID are **attributes.**

EMPLOYEE (table name)

Row → Tuples

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	DEPARTMENT_ID
100	Steven	King	90
101	Neena	Kochhar	90
102	Lex	De Haan	90
200	Jennifer	Whalen	10
205	Shelley	Higgins	110

↑ Column (Attributes)

5. **What are the functions of DBA? [April/May '2010] [Nov/Dec'10][April/May'11] [Nov/Dec 2023]**
   - The person who has central control over the system is called a **database administrator(DBA**).
   - In a database environment, the primary resource is the database itself and the secondary resource is the DBMS and related software.
   - Administering these resources is the responsibility of the **database administrator (DBA).**
   - The DBA is responsible for authorizing access to the database, for coordinating and monitoring its use, and for acquiring software and hardware resources as needed.

6. **What is data model? List the types of Data model?[April/May'11][Nov/Dec'11] [April/May '12][April 2019]**

   A data model provides a way to describe the design and structure of a database at the physical, logical and view level.

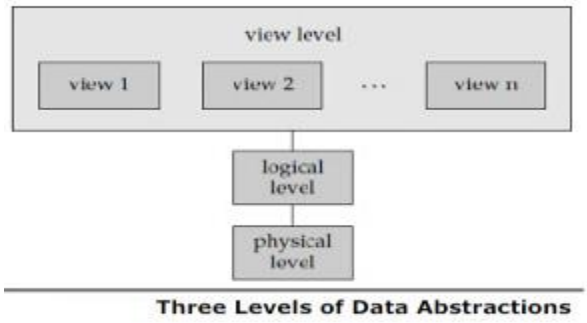
   Data Model is a collection of tools for describing Data , Data relationships , Data semantics & Data constraints.

   Types of Data Models:

- High level or conceptual data model
- Entity Relationship Model
- Relational Model
- Hierarchical Model
- Network Model
- Object Based Data model

7. What are the three levels of data abstraction? [May/June'14][Nov/Dec'17] [Nov/Dec 2023]

- **Physical Level:**The lowest level is the physical Level of abstraction that describes how the data areactually stored.
- **Logical Level:** The next-higher level of abstraction describes what data arestored in the database, and what relationships exist among those data.
- **View Level**: the highest level of abstraction describes only part of the entire database. The variety of information stored in a large database. The view level of abstraction exists to simplify their interaction with the system.



Three Levels of Data Abstractions

8. List any 8 Applications of DBMS. [April 2019]

Sector	Use of DBMS
Banking	For customer information, account activities, payments, deposits, loans, etc.
Airlines	For reservations and schedule information.
Universities	For student information, course registrations, colleges and grades.
Telecommunication	It helps to keep call records, monthly bills, maintaining balances, etc.
Finance	For storing information about stock, sales, and purchases of financial instruments like stocks and bonds.
Sales	Use for storing customer, product & sales information.

Manufacturing	It is used for the management of supply chain and for tracking production of items. Inventories status in warehouses.
HR Management	For information about employees, salaries, payroll, deduction, generation of paychecks, etc.

9. State three levels of abstraction [June 2021]

- Physical Level
- Logical Level
- View Level

10. Define the two levels of data independence [Nov/Dec'10]

Physical data independence:

The ability to modify the **physical schema** without causing application programs to be rewritten or **with out changing** the **conceptual schema.**

Modifications at this level are usually to improve performance.

Logical data independence:

The ability to modify the **conceptual (logical) schema** without causing application programs to be rewritten or without changing the **external schema.**

Usually done when logical structure of database is altered.

11. Is it possible for several attributes to have a same domain? Illustrate your answer with suitable example. [Nov/Dec'15]

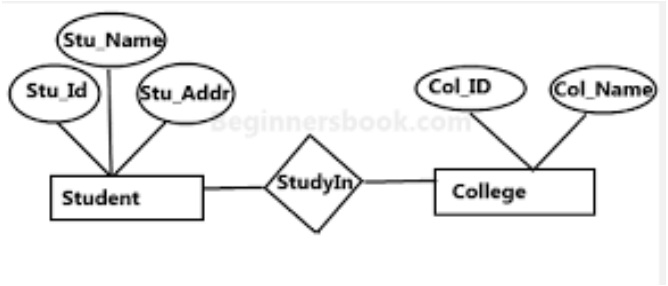
A **domain is a pool of values** , from which a values of specific attributes of specific relations are taken. For Eg, the**domain department** is a set of possible department names, and **the domainemp_name** is a set of all employee names. Thus each and every attribute has its own domain. Hence it is **not possible for several attributes to have Same domain.**

12. Explain Entity relationship model. [May/June '16]/ What is ER Diagram? [June 2021]

The E-R data model is based on a perception of a real world that consists of a collection of basic objects called entities and of relationships among these objects.

The graphical components of ER Model consists of

- Entities
- Attributes
- Relationship
- Cardinality



13. Define the term Entity Set Relationship Set. [April 2019]

Entity Set:

An **entity set** is a **set of same type** of **entities**. Types of entity sets are,

- Strong entity set
- Weak entity set

Relationship Sets :

A relationship set is a **set of relationships of same type.**

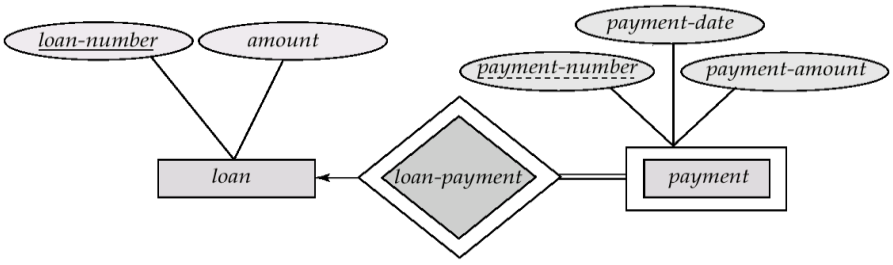
Types of Relationship sets are,

- Unary relationship set
- Binary relationship set
- Ternary relationship set
- N-ary relationship set

14. What do you mean by weak entity set? Give Example [April/May'2010] [Nov/Dec'16]

An entity set that does not have a primary key is referred to as a ***weak entity set***.

Eg,



15. Give example for one to one and one to many Relationship [April/May'13]/ What does cardinality ratio specify?

One to one. (1:1)

One department has only one head and one head can be the head of only one department. Hence the cardinality is one to one. (1:1)

One to Many (1:N)

One department has multiple instructors . But instructor belongs to only one department. Hence the cardinality between department and instructor is One to Many (1:N)

Many to One (N :1)

One course is taught by only one instructor. But one instructor teaches many courses. Hence the cardinality between course and instructor is Many to One (N :1)

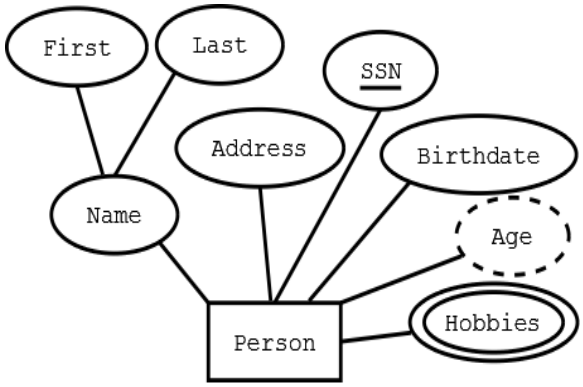
Many to Many (M:N)

One course can be enrolled by many students and one student can enroll for many courses. Hence the cardinality between course and student is Many to Many (M:N)

16. With an example what is derived attribute is? [Nov/Dec'11]

Derived Attribute: An attribute that's value is derived from a stored attribute.

Example : age, and it's value is derived from the stored attribute Date of Birth.



17. What is an entity?[May/June'14][April 2019]

An entity - is a "thing" or "object" in the real world. An *entity* is an object that exists and is distinguishable from other objects. An entity is described (in DB) using a set of **attributes.**

Entities	Attributes
CUSTOMER	family name, age, shoe size, town of residence, email
CAR	model, weight, catalog price
ORDER	order date, ship date
JOB	title, description
TRANSACTION	amount, transaction date
EMPLOYMENT CONTRACT	start date, salary

PART B

1. Write Short Notes on Data Model and its Types /Explain three different groups of data models.[**Nov/Dec'14] [Apr/May'10] [Nov/Dec'10][April 2019]**
2. With neat block diagram explain basic architecture of DBMS. **[Nov/Dec'15]**/Briefly explain about database system architecture.**[April/May'12][May/Jun'15][Nov/Dec'17]/ [Nov/Dec 2023]/**[**May/June'14][April/May'17**]/With a neat sketch discuss the three-schema architecture of a DBMS.**[Nov/Dec'11]** /Explain Database System Structure **[April/May'10] [April/May'11] [June 2021]**
3. **Compare and Contrast File Processing system with DBMS [Nov/Dec 2023]**
4. Explain the differences between physical level, conceptual level and view level of data abstraction. (4) **[April/May'11]**
5. What are the advantages of having a centralized control of data ? Illustrate your answer with example. **[Nov/Dec'15]**
6. Briefly explain about Views of Data. **[May/June '16]**
7. Explain various disadvantages of file system and how it can be overcome by database system.**(Apr/May'10**)/ Explain the purpose of DBMS. **[April/May'13]**
8. Illustrate and explain ER diagram for Banking System**[Nov/Dec'14] [May/June '15]** / Draw ER diagram corresponding to customers and Loans **[May/June'14]**
9. Draw ER Diagram for " Restaurant menu ordering system", which will facilitate the food items ordering and services within restaurant. The entire scenario is detailed below. The customer is able to view the food items menu, call the waiter, place orders and obtain final bill through the computer kept in their table. The waiters through their wireless tablet PC are able to initialize a table for customers , control the table functions to assist customers , orders, send orders to food preparation staff(chef) and finalize the customers bill. The chef with their touch display interface to the system, are able to view orders sent to the kitchen by waiters. During preparation they are able to let the waiter know the status of each item , and can send notifications when items are completed. The system should have full accountability and logging facilities and should support supervisor actions to account for exceptional circumstances such as meal being refunded or walked out on. **[May/June '15]**
10. Construct an ER Diagram for Car insurance company whose customers own one or more cars each. Each car has associated with zero to any number of recorded accidents. Each insurance policy covers one or more and has one or more premium payments associated with it. Each payment is for a particular period of time and has an associated due date and the date when the payment was received**. [Nov/Dec'16]**
11. Construct ER diagram of employee payroll system. **[Apr/May'10]**
12. Describe the components of Entity relationship diagram with suitable example. **[Nov/Dec'10]**

PART C

1. Draw ER diagram for banking systems(home loan application) (13) **[Nov/Dec'17]**
2. A car rental company maintains a database for all vehicles in its current fleet. For all vehicles, it includes the vehicle id number, license no, manufacturer, model, date of purchase,&color. Special data are included for certain type of vehicles.

   Trucks : cargo capacity

   Sports car: horse power, renter age requirement

   Vans: no of passengers

   Off Road vehicles: ground clearance, drive train (four or two wheel drive)

   Construct an ER model for car rental company database. **[Nov/Dec'15]**