DBMS

INTRODUCTION

- Database is a collection of related data.
- Database management system is software designed to assist the maintenance and utilization of large scale collection of data.
- DBMS came into existence in 1960 by Charles Bachman. Integrated data store which is also called as the first general purpose DBMS.
- Again in 1960 IBM brought IMS-Information management system.
- In 1970 Edgor Codd at IBM came with new database called RDBMS.
- In 1980 then came SQL Architecture- Structure Query Language.
- In 1980 to 1990 there were advances in DBMS e.g. DB2, ORACLE.

- Data
  - Data is raw fact or figures or entity.
  - When activities in the organization takes place, the effect of these activities need to be recorded which is known as Data.
- Information
  - Processed data is called information
  - The purpose of data processing is to generate the information required for carrying out the business activities.

In general data management consists of following tasks

• Data capture: Which is the task associated with gathering the data as and when

they originate.

• Data classification: Captured data has to be classified based on the nature and

intended usage.

• Data storage: The segregated data has to be stored properly.

• Data arranging: It is very important to arrange the data properly

• Data retrieval: Data will be required frequently for further processing,

Hence it is very important to create some indexes so that data can be retrieved easily

Cont....

- Data maintenance: Maintenance is the task concerned with keeping the data upto-date.
- Data Verification: Before storing the data it must be verified for any error.
- Data Coding: Data will be coded for easy reference.
- Data Editing: Editing means re-arranging the data or modifying the data for presentation.
- Data transcription: This is the activity where the data is converted from one form into another.
- Data transmission: This is a function where data is forwarded to the place where it would be used further.

Metadata

- **Metadata** (meta data, or sometimes meta information) is "data about data", of any sort in any media.
- An item of metadata may describe a collection of data including multiple content items and hierarchical levels, for example a database schema.
- Eg: A text document's metadata may contain information about how long the document is, who the author is, when the document was written, and a short summary of the document.
- Metadata within web pages can also contain descriptions of page content, as well as key words linked to the content.[14] These links are often called "Metatags",

Database:

- Database may be defined in simple terms as a collection of data
- A database is a collection of related data.
- The database can be of any size and of varying complexity.
- A database may be generated and maintained manually or it may be computerized.

Database Management System - A Database Management System (DBMS) is a collection of program that enables user to create and maintain a database.

Difference between File system & DBMS

File System

1. File system is a collection of data. Any management with the file system, user has to write the procedures
2. File system gives the details of the data representation and Storage of data.
3. In File system storing and retrieving of data cannot be done efficiently.
4. Concurrent access to the data in the file system has many problems like
   a. Reading the file while other deleting some information, updating some information
5. File system doesn't provide crash recovery mechanism.
6. Protecting a file under file system is very difficult.

DBMS

1. DBMS is a collection of data and user is not required to write the procedures for managing the database.
2. DBMS provides an abstract view of data that hides the details.
3. DBMS is efficient to use since there are wide varieties of sophisticated techniques to store and retrieve the data.
4. DBMS takes care of Concurrent access using some form of locking.
5. DBMS has crash recovery mechanism, DBMS protects user from the effects of system failures.
6. DBMS has a good protection mechanism.

Advantages of DBMS.

1. Data independency:

Application program should not be exposed to details of data representation and storage

DBMS provides the abstract view that hides these details.

2. Efficient data access.:

DBMS utilizes a variety of sophisticated techniques to store and retrieve data

efficiently.

3. Data integrity and security:

Data is accessed through DBMS, it can enforce integrity constraints.

E.g.: Inserting salary information for an employee.

Cont....

4. Data Administration:

When users share data, centralizing the data is an important task, Experience

professionals can minimize data redundancy and perform fine tuning which reduces

retrieval time.

5. Concurrent access and Crash recovery:

DBMS schedules concurrent access to the data. DBMS protects user from the effects of

system failure.

6. Reduced application development time.

DBMS supports important functions that are common to many applications.

Functions of DBMS

- Data Definition: The DBMS provides functions to define the structure of the data in the application. These include defining and modifying the record structure, the type and size of fields and the various constraints to be satisfied by the data in each field.
- Data Manipulation: Once the data structure is defined, data needs to be inserted, modified or deleted. These functions which perform these operations are part of DBMS.
- Data Security & Integrity: The DBMS contains modules which handle the

security and integrity of data in the application.

Cont.....

- Data Recovery and Concurrency: Recovery of the data after system failure and

concurrent access of records by multiple users is also handled by DBMS.

- Data Dictionary Maintenance: Maintaining the data dictionary which contains the

data definition of the application is also one of the functions of DBMS.

- Performance: Optimizing the performance of the queries is one of the important

functions of DBMS.

Role of Database Administrator.

Typically there are three types of users for a DBMS:

1. The END User who uses the application. Ultimately he is the one who actually

puts the data into the system into use in business. This user need not know

anything about the organization of data in the physical level.

2. The Application Programmer who develops the application programs. He/She has

more knowledge about the data and its structure. He/she can manipulate the data

using his/her programs. He/she also need not have access and knowledge of the

complete data in the system.

3. The Data base Administrator (DBA) who is like the super-user of the system.

The role of DBA is very important and is defined by the following functions.

- Defining the schema: The DBA defines the schema which contains the structure of the data in the application. The DBA determines what data needs to be present in the system and how this data has to be presented and organized.
- Liaising with users: The DBA needs to interact continuously with the users to understand the data in the system and its use.
- Defining Security & Integrity checks: The DBA finds about the access restrictions to be defined and defines security checks accordingly. Data Integrity checks are defined by the DBA.
- Defining Backup/Recovery Procedures: The DBA also defines procedures for backup and recovery. Defining backup procedure includes specifying what data is to be backed up, the periodicity of taking backups and also the medium and storage place to backup data.
- Monitoring performance: The DBA has to continuously monitor the performance of the queries and take the measures to optimize all the queries in the application.

Simplified Database System Environment:

A database management system (DBMS) is a collection of programs that enables users to create and maintain database.

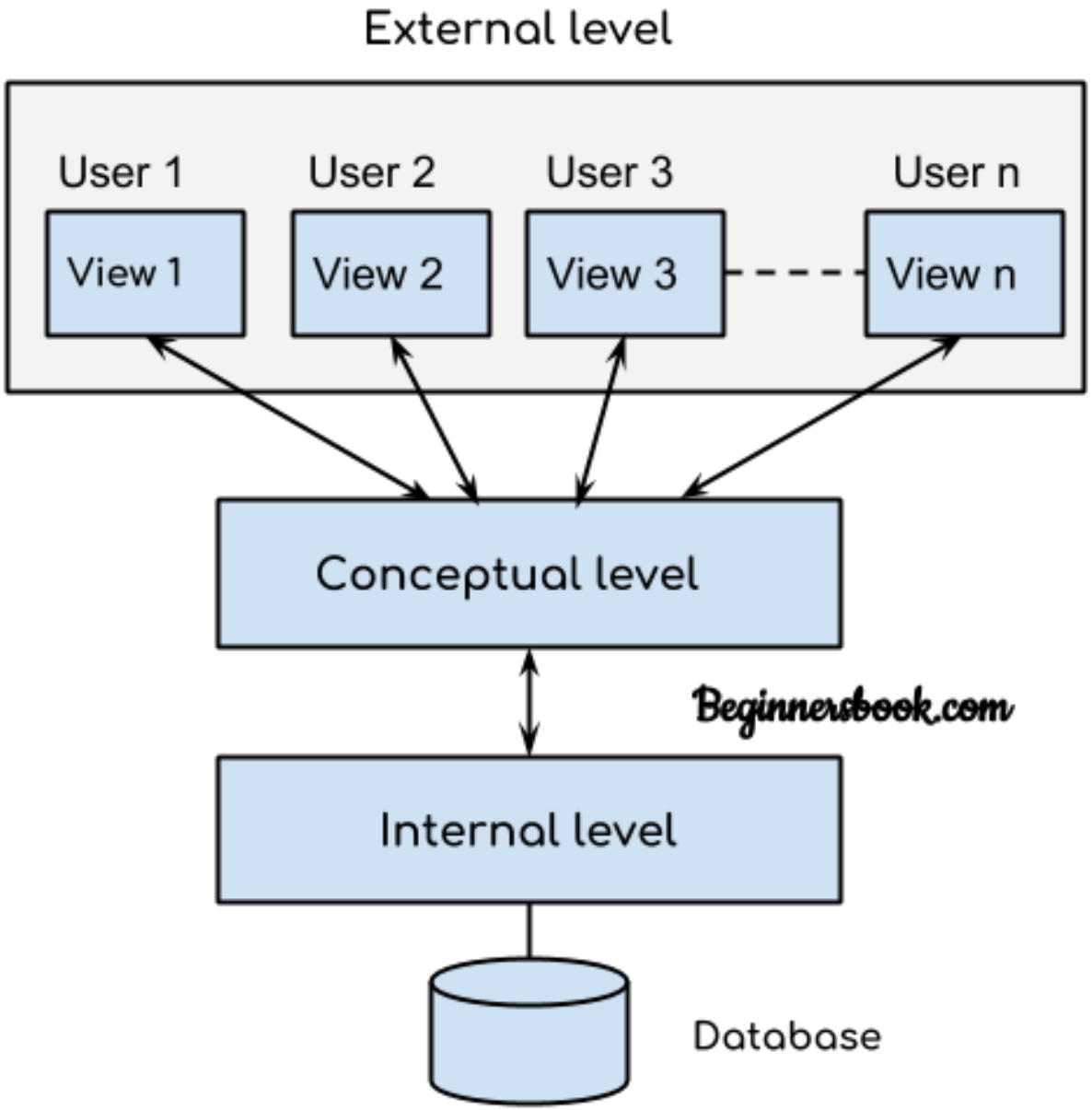
The DBMS is a general purpose software system that facilitates the process of defining, constructing, manipulating and sharing databases among various users and applications.

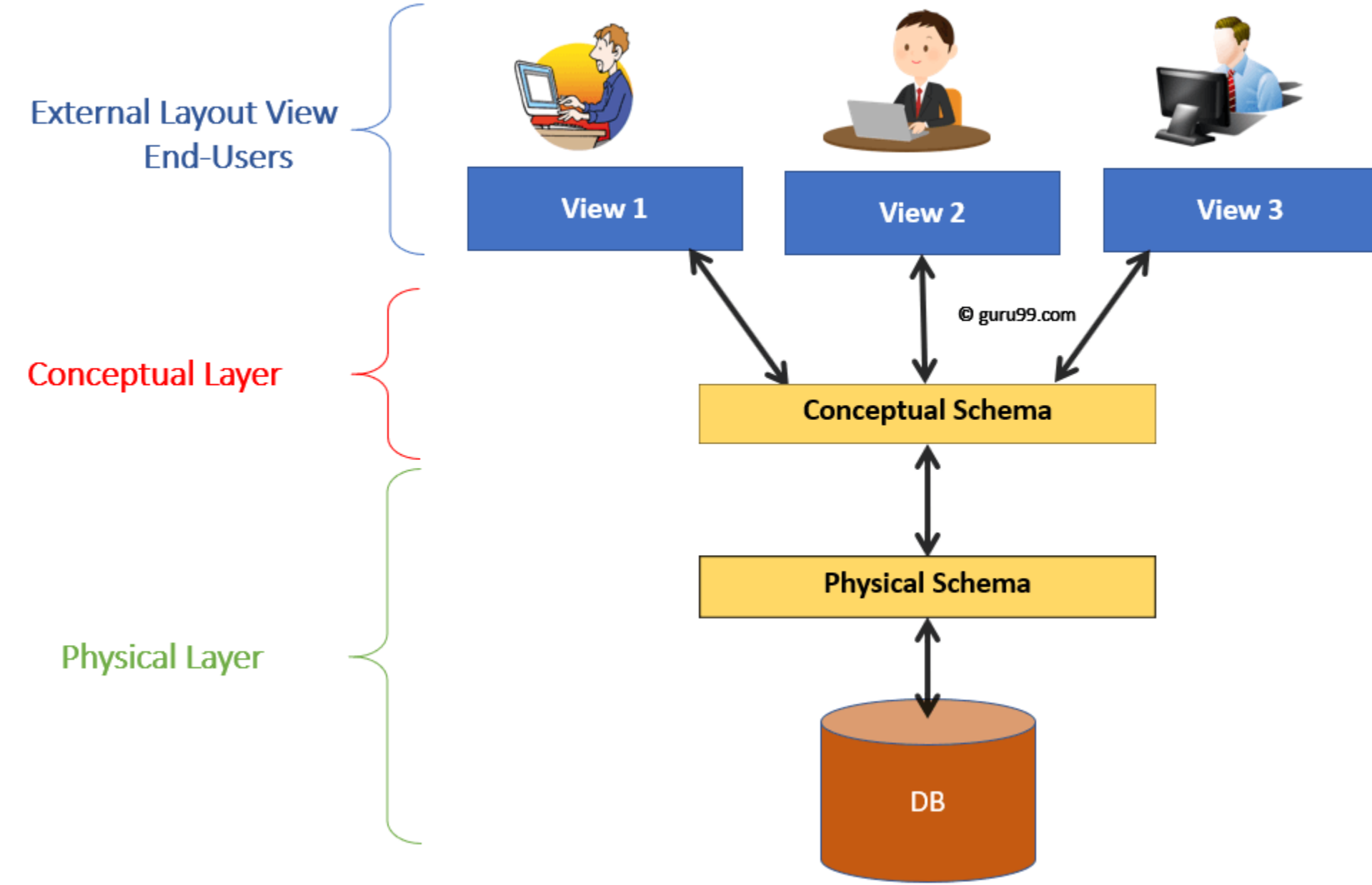
Defining a database specifying the database involves specifying the data types, constraints and structures of the data to be stored in the database.

The descriptive information is also stored in the database in the form database catalog or dictionary; it is called meta-data.

Manipulating the data includes the querrying the database to retrieve the specific data. An application program accesses the database by sending the qurries or requests for data to DBMS.

The important function provided by the DBMS includes protecting the database and maintain the database.


External level
User 1
User 2
User 3
User n
View 1
View 2
View 3
View n
Conceptual level
Beginnersbook.com
Internal level
Database


External Layout View
End-Users
Conceptual Layer
Physical Layer
View 1
View 2
View 3
© guru99.com
Conceptual Schema
Physical Schema
DB

Architecture of DBMS

A commonly used views of data approach is the three-level architecture suggested by ANSI/SPARC (American National Standards Institute/Standards Planning and Requirements Committee).

The three levels of the architecture are three different views of the data:

External - individual user view

Conceptual - community user view

Internal - physical or storage view

external level

- The external level is the view that the individual user of the database has. This view is often a restricted view of the database and the same database may provide a number of different views for different classes of users.
- For example, a department head may only be interested in the departmental finances and student enrolments but not the library information. The librarian would not be expected to have any interest in the information about academic staff.
- s/w and h/w independent

The conceptual view

- The conceptual view is the information model of the enterprise and contains the view of the whole enterprise without any concern for the physical implementation. This view is normally more stable than the other two views.
- In a database, it may be desirable to change the internal view to improve performance while there has been no change in the conceptual view of the database.
- The conceptual view is defined by the conceptual schema which includes definitions of each of the various types of data.
- Overall view keeping in consideration od DBMS software to be used, hence this view is dependent on s/w but not on h/w

The internal view

- The internal view is the view about the actual physical storage of data. It tells us what data is stored in the database and how.
- At least the following aspects are considered at this level:
- Storage allocation e.g. B-trees, hashing etc.
- Access paths e.g. specification of primary and secondary keys, indexes and pointers and sequencing.
- Miscellaneous e.g. data compression and encryption techniques, optimization of the internal structures.
- At least the following aspects are considered at this level: Storage allocation e.g. B-trees, hashing etc. Access paths e.g. specification of primary and secondary keys, indexes and pointers and sequencing. Miscellaneous e.g. data compression and encryption techniques, optimization of the internal structures.
- Both s/w and h/w dependent.

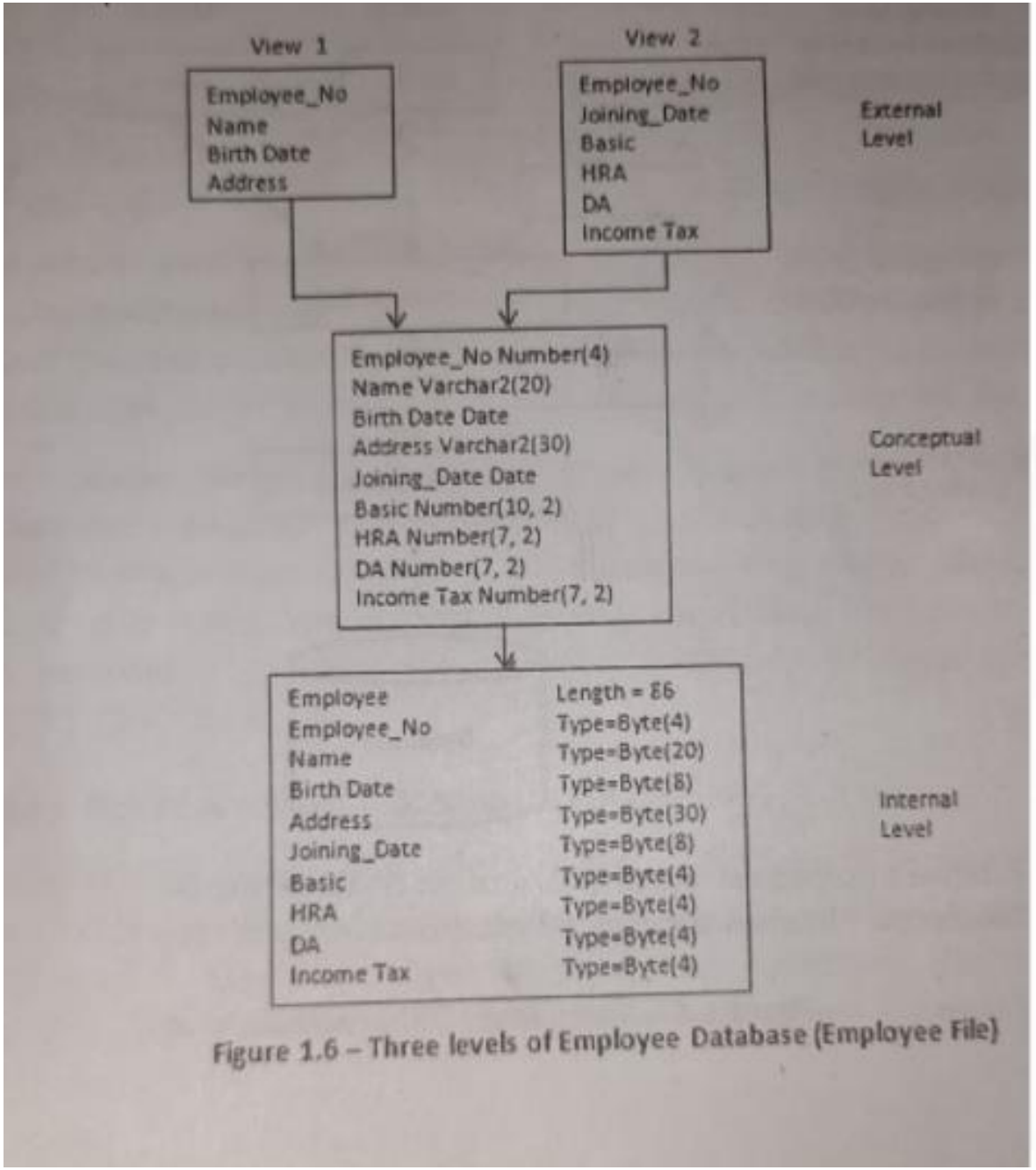


Figure 1.6 – Three levels of Employee Database (Employee File)

Data Modeling and Data Models

- **Data modeling**: It is a process of creating a data model for the data to be stored in a Database i.e. a conceptual representation of
  - Data objects
  - The associations between different data objects
  - The rules.
- Data modeling helps in the visual representation of data and enforces business rules, regulatory compliances, and government policies on the data. Data Models ensure consistency in naming conventions, default values, semantics, security while ensuring quality of the data.

DBMS Schemas: Internal, Conceptual, External

- The **database schema** of a **database** is its structure described in a formal language supported by the **database** management system (DBMS). The term "**schema**" refers to the organization of data as a blueprint of how the **database** is constructed (divided into **database** tables in the case of relational **databases**).
- The design of a database at physical level is called **physical schema**, how the data stored in blocks of storage is described at this level.
- Design of database at logical level is called **logical schema**, programmers and database administrators work at this level, at this level data can be described as certain types of data records gets stored in data structures, however the internal details such as implementation of data structure is hidden at this level (available at physical level).
- Design of database at view level is called **view schema**. This generally describes end user interaction with database systems.

Data Modeling and Data Models

Data model emphasizes on (1) what data is needed and (2) how it should be organized instead of what operations need to be performed on the data.

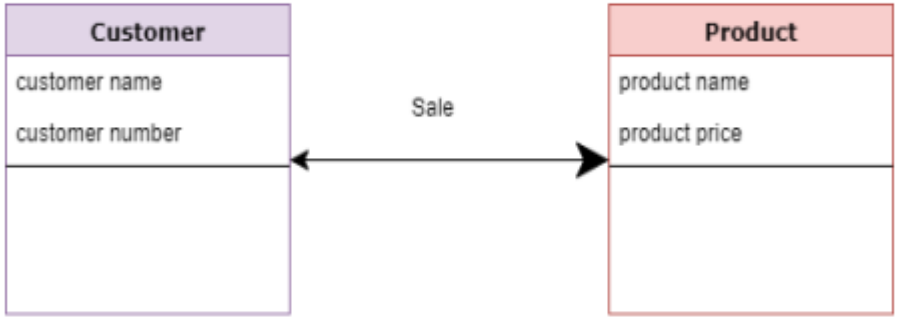
- **Data models**: Simple representations of complex real-world data structures
  - Useful for supporting a specific problem domain
- **Model** - Abstraction of a real-world object or event

- The primary goal of using data model are:

1. Ensures that all data objects required by the database are accurately represented. Omission of data will lead to creation of faulty reports and produce incorrect results.
2. A data model helps design the database at the conceptual, physical and logical levels.
3. Data Model structure helps to define the relational tables, primary and foreign keys and stored procedures.
4. It provides a clear picture of the base data and can be used by database developers to create a physical database.
5. It is also helpful to identify missing and redundant data.
6. Though the initial creation of data model is labor and time consuming, in the long run, it makes your IT infrastructure upgrade and maintenance cheaper and faster.

Data Model Basic Building Blocks

- **Entity**: Unique and distinct object used to collect and store data
  - **Attribute**: Characteristic of an entity
- **Relationship**: Describes an association among entities
  - **One-to-many (1:M)**
  - **Many-to-many (M:N or M:M)**
  - **One-to-one (1:1)**



- **Constraint**: Set of rules to ensure data integrity

Hierarchical and Network Models

Hierarchical Models

- Manage large amounts of data for complex manufacturing projects
- Represented by an upside-down tree which contains segments
  - **Segments**: Equivalent of a file system's record type
- Depicts a set of one-to-many (1:M) relationships

Network Models

- Represent complex data relationships
- Improve database performance and impose a database standard
- Depicts both one-to-many (1:M) and many-to-many (M:N) relationships

Hierarchical Model

Advantages

- Promotes data sharing
- Parent/child relationship promotes conceptual simplicity and data integrity
- Database security is provided and enforced by DBMS
- Efficient with 1:M relationships

Disadvantages

- Requires knowledge of physical data storage characteristics
- Navigational system requires knowledge of hierarchical path
- Changes in structure require changes in all application programs
- Implementation limitations
- No data definition
- Lack of standards

Network Model

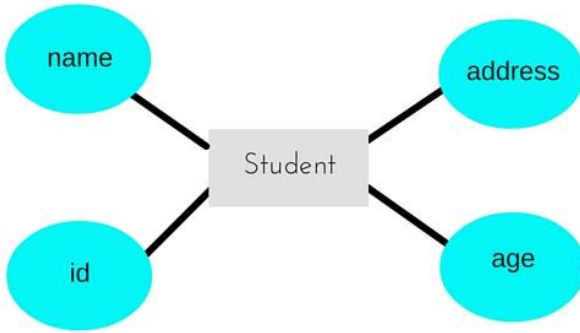
Advantages	Disadvantages
▪ Conceptual simplicity ▪ Handles more relationship types ▪ Data access is flexible ▪ Data owner/member relationship promotes data integrity ▪ Conformance to standards ▪ Includes data definition language (DDL) and data manipulation language (DML)	▪ System complexity limits efficiency ▪ Navigational system yields complex implementation, application development, and management ▪ Structural changes require changes in all application programs

Entity-relationship Model

- In this database model, relationships are created by dividing object of interest into entity and its characteristics into attributes.
- Different entities are related using relationships.
- E-R Models are defined to represent the relationships into pictorial form to make it easier for different stakeholders to understand.
- This model is good to design a database, which can then be turned into tables in relational model(explained below).
- Let's take an example, If we have to design a School Database, then **Student** will be an **entity** with

  attributes name, age, address etc.

  As **Address** is generally complex, it can be another **entity** with **attributes** street name,pincode, city etc, and there will be a relationship between them



ER diagram : Components

ER diagram shows the complete logical structure of a database.

1. Entity

An entity is an object or component of data. An entity is represented as rectangle in an ER diagram.

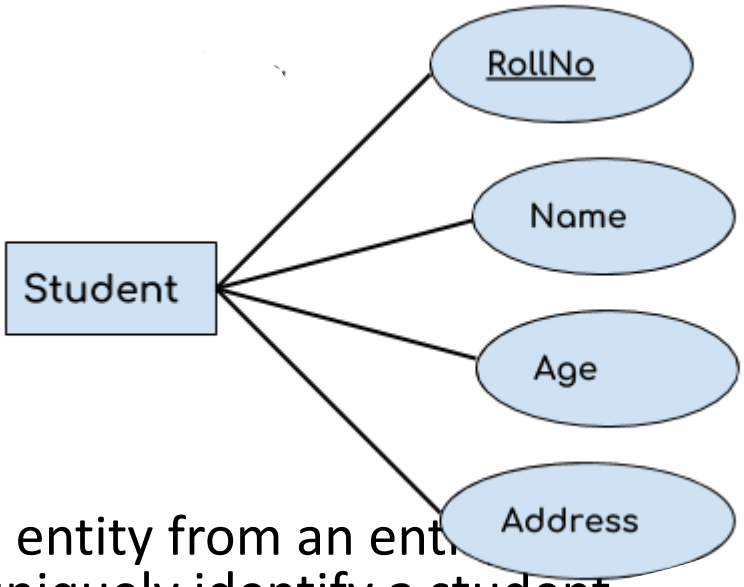
- An entity might be
- An object with physical existence (e.g., a lecturer, a student, a car)
- An object with conceptual existence (e.g., a course, a job, a position)
- An *entity set* is a collection of entities of an entity type at a particular point of time. An **entity set** is a **set** of **entities** of the same type (e.g., all persons having an account at a bank).

Weak Entity:

An entity that cannot be uniquely identified by its own attributes and relies on the relationship with other entity is called weak entity. The weak entity is represented by a double rectangle. For example – a bank account cannot be uniquely identified without knowing the bank to which the account belongs, so bank account is a weak entity.

ERD- Components

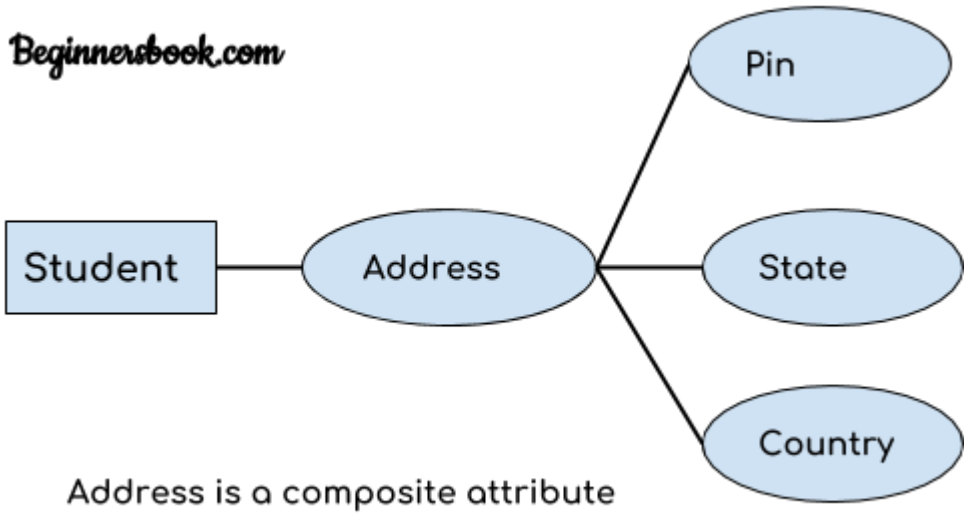
- **2. Attribute**
- An attribute describes the property of an entity. An attribute is represented as Oval in an ER diagram. There are four types of attributes:
- 1. Key attribute
  2. Composite attribute
  3. Multivalued attribute
  4. Derived attribute



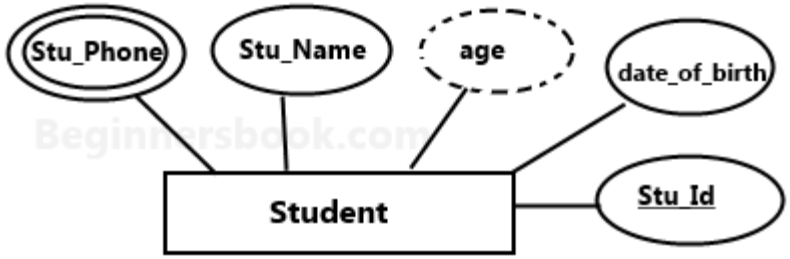
- **1. Key attribute:**
- A key attribute can uniquely identify an entity from an enti
  For example, student roll number can uniquely identify a student from a set of students. Key attribute is represented by oval same as other attributes however the **text of key attribute is underlined**.

2. Composite attribute:

An attribute that is a combination of other attributes is known as composite attribute. For example, In student entity, the student address is a composite attribute as an address is composed of other attributes such as pin code, state, country.



Address is a composite attribute



3. Multivalued attribute:

- An attribute that can hold multiple values is known as multivalued attribute. It is represented with **double ovals** in an ER Diagram. For example – A person can have more than one phone numbers so the phone number attribute is multivalued.

4. Derived attribute:

- A derived attribute is one whose value is dynamic and derived from another attribute. It is represented by **dashed oval** in an ER Diagram. For example – Person age is a derived attribute as it changes over time and can be derived from another attribute (Date of birth).

Relationship

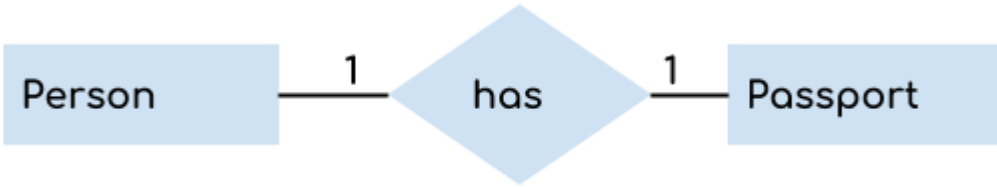
3. Relationship

A relationship is represented by diamond shape in ER diagram, it shows the relationship among entities. There are four types of relationships:

1. One to One
2. One to Many
3. Many to One
4. Many to Many

1. One to One Relationship

When a single instance of an entity is associated with a single instance of another entity then it is called one to one relationship. For example, a person has only one passport and a passport is given to one person.



- **One to Many Relationship**

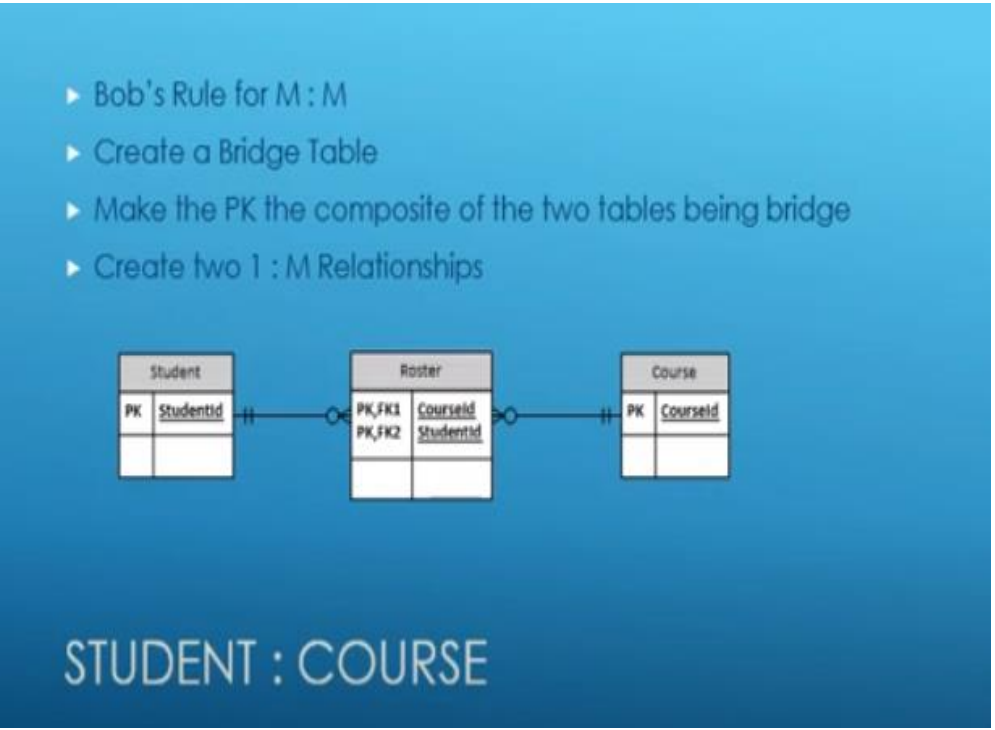
When a single instance of an entity is associated with more than one instances of another entity then it is called one to many relationship. For example – a customer can place many orders but a order cannot be placed by many customers.

3. Many to One Relationship

When more than one instances of an entity is associated with a single instance of another entity then it is called many to one relationship. For example – many students can study in a single college but a student cannot study in many colleges at the same time.

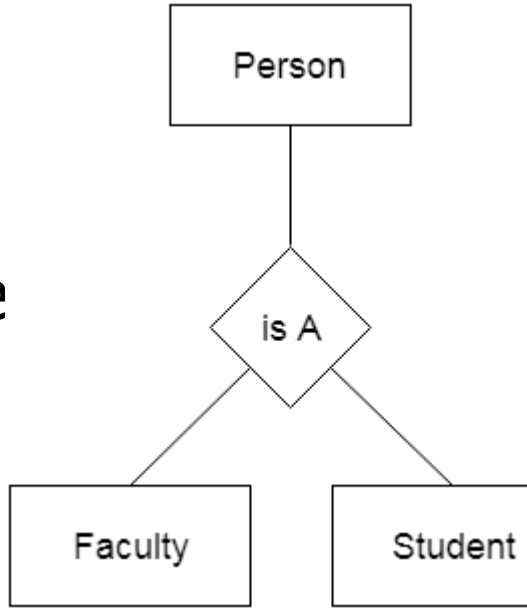
Bridge-entity

- A bridge entity is used to capture a many-to-many relationship that cannot be accommodated by the natural granularity of a single fact or dimension entity. A bridge entity serves to bridge between the fact and dimension entities to support many-valued dimension attributes.

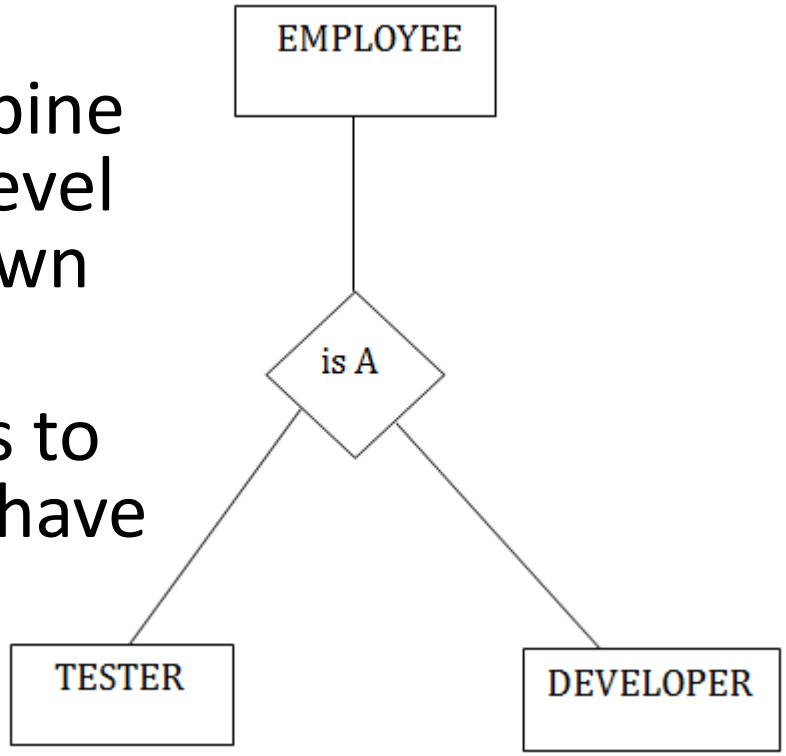


A composite entity is also known as a "bridge" entity. This "bridge" is used to handle the many-to-many relationships that the traditional entity could not handle. This entity lies between the two entities that are of interest and this composite entity shares the primary keys from both the connecting tables.

- **Generalization** is a process in which the common attributes of more than one entities form a new entity. This newly formed entity is called generalized entity.
- 1. Generalization uses bottom-up approach where two or more lower le entities combine together to form a higher level new entity.
  2. The new generalized entity can further combine together with lower level entity to create a further higher level generalized entity.



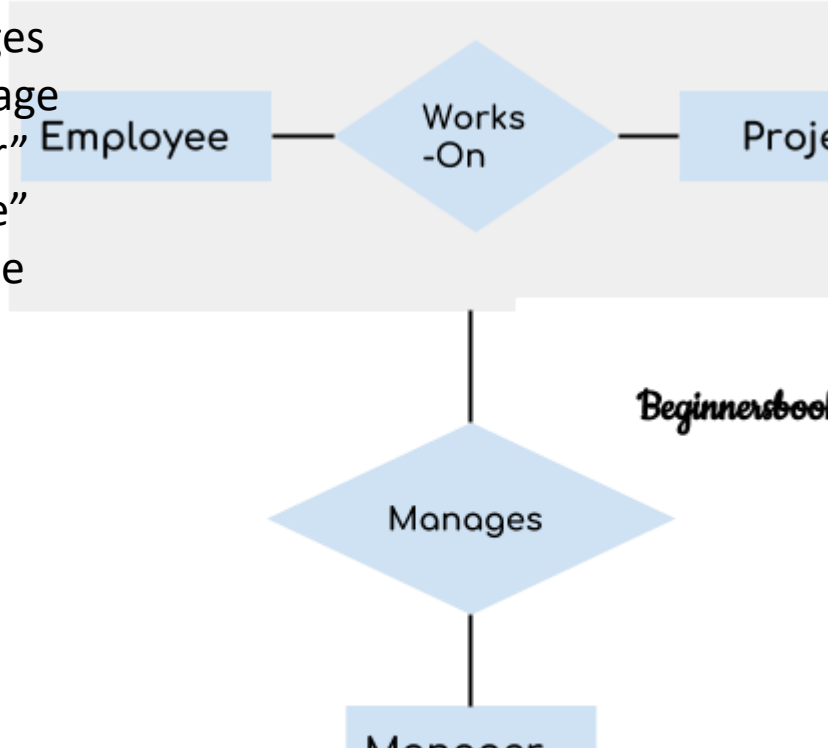
- **Specialization** is a process in which an entity is divided into sub-entities. You can think of it as a reverse process of **generalization**, in generalization two entities combine together to form a new higher level entity. Specialization is a top-down process.
- The idea behind Specialization is to find the subsets of entities that have few distinguish attributes. For example – Consider an entity employee which can be further classified as sub-entities Technician, Engineer & Accountant because these sub entities have some distinguish attributes.



Aggregation

- In aggregation, the relation between two entities is treated as a single entity. In aggregation, relationship with its corresponding entities is aggregated into a higher level entity.

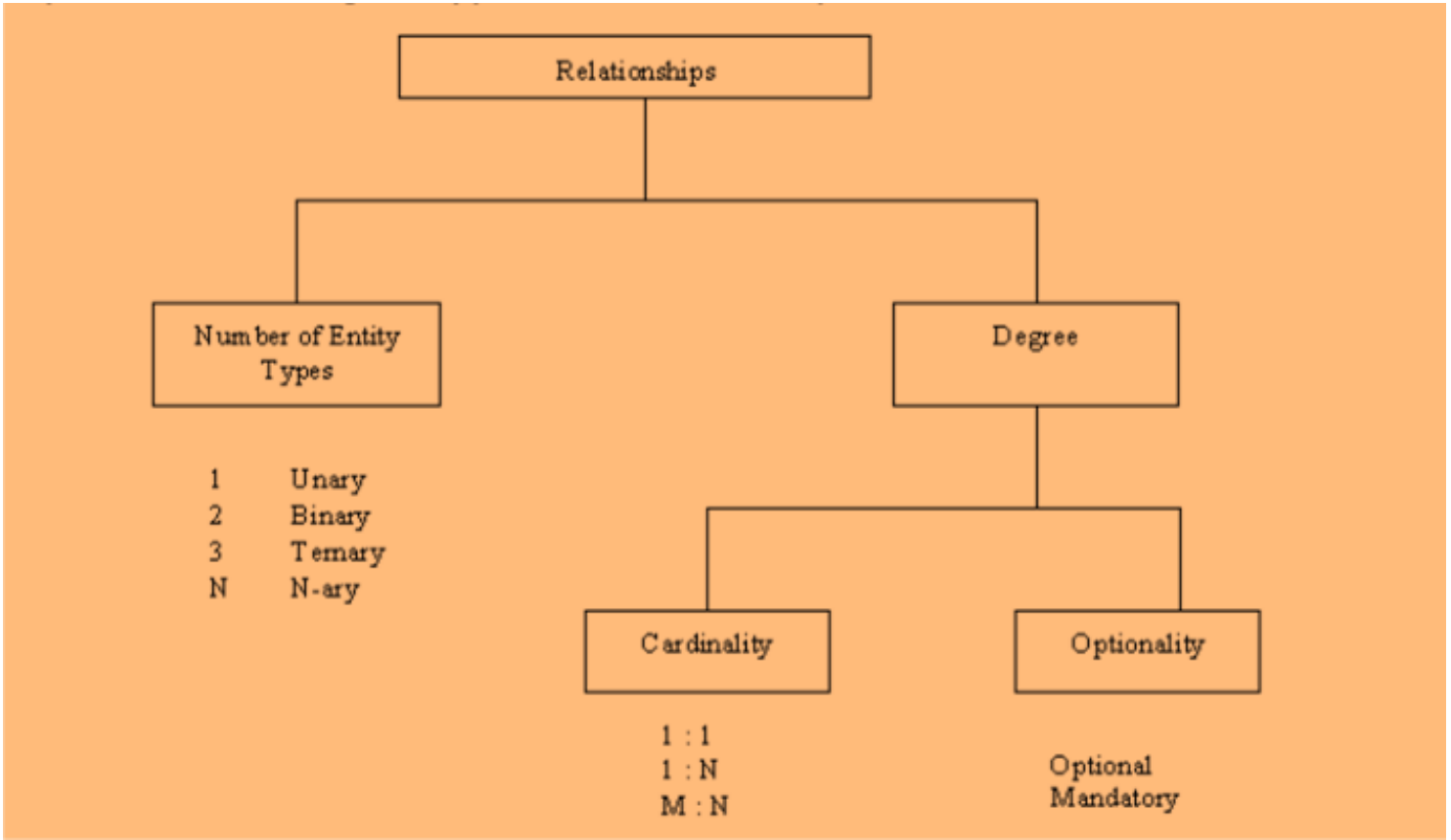
In real world, we know that a manager not only manages the employee working under them but he has to manage the project as well. In such scenario if entity “Manager” makes a “manages” relationship with either “Employee” or “Project” entity alone then it will not make any sense because he has to manage both. In these cases the relationship of two entities acts as one entity. In our example, the relationship “Works-On” between “Employee” & “Project” acts as one entity that has a relationship “Manages” with the entity “Manager”.



Relationships

Ways of Classifying Relationships Types

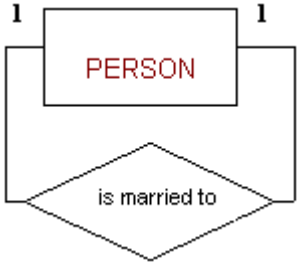
- A relationship type can be classified by the number of entity types involved, and by the degree of the relationship type.



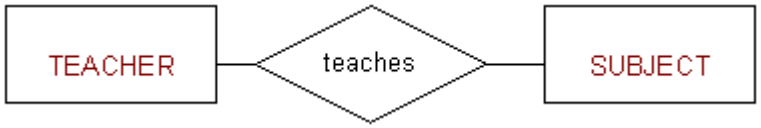
Degree

The ***degree*** of a relationship is the number of entity types that participate in the relationship. The three most common relationships in ER models are ***Binary, Unary*** and ***Ternary***

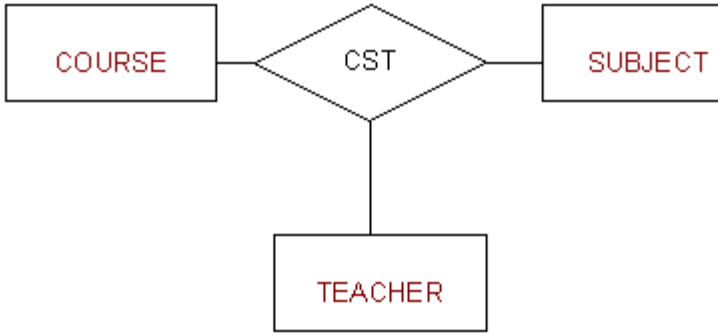
A **unary relationship** is when both participants in the relationship are the same entity.



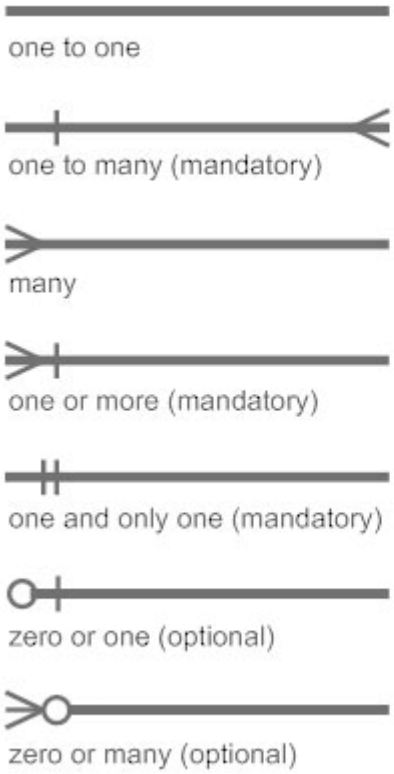
A **binary relationship** is when two entities participate and is the most common relationship degre



A **ternary relationship** is when three entities participate in the relationship

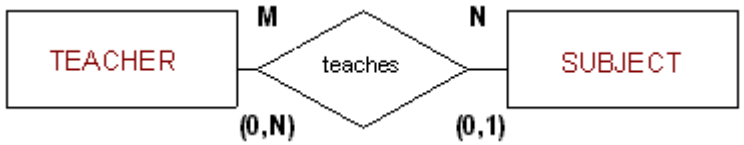


Notations for ERD



- Connectivity :
  - connectivity is used to describe the relationship classification. Just to show the relation existence and type of relation between entities.
- Cardinality: expresses the minimum and maximum number of entity occurrences associated with one occurrence of the related entity. In the ERD, cardinality is indicated by placing the appropriate numbers beside the entities, using the format (min , max).

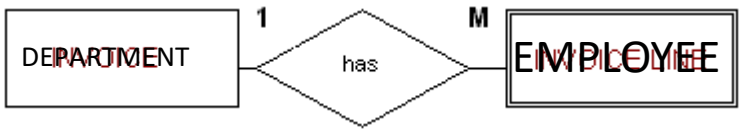
Teachers may teach 0 subjects if they are involved in non teaching projects. Therefore, the cardinality limits for TEACHER are **(O,N)**. Each Subject is taught by only one teacher, but it is possible to have Subjects that have not yet been assigned a teacher. Therefore, the cardinality limits for SUBJECT are **(0,1)**.



Existence Dependency

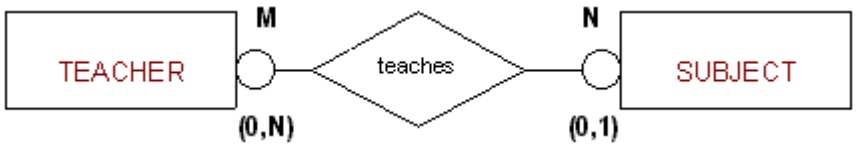
Weak Entity

A ***weak entity*** is an entity type that, in addition to being existence dependent, has a primary key that has been totally or partially constructed from the entity it depends on

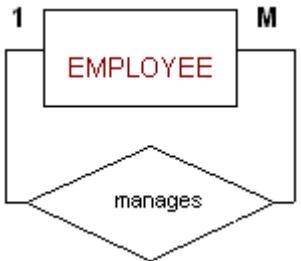


Mandatory/Optional Relationships:

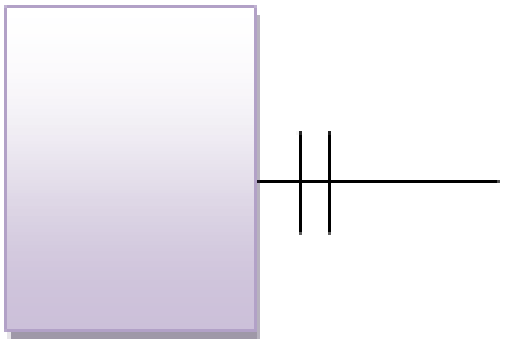
Participation by an entity in a relationship may be **optional** or **mandatory**.



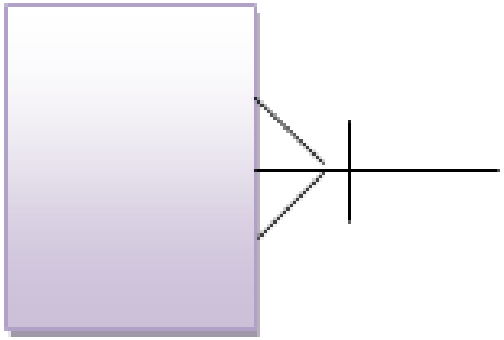
Recursive Relationships



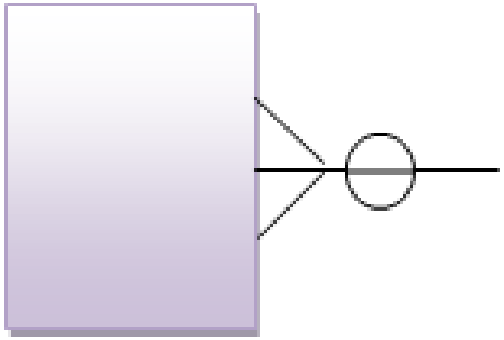
Crow Foot Notation Symbols



One and Only One (1)



One or Many (1 .. *)

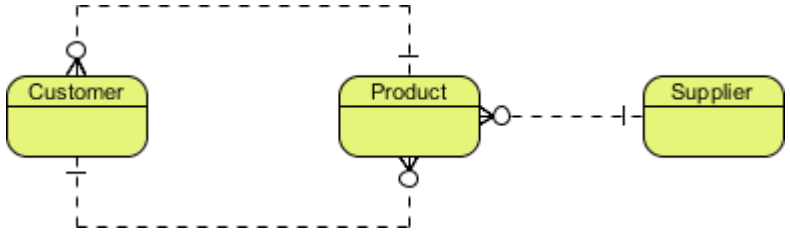


Zero or One or Many (0 .. *)

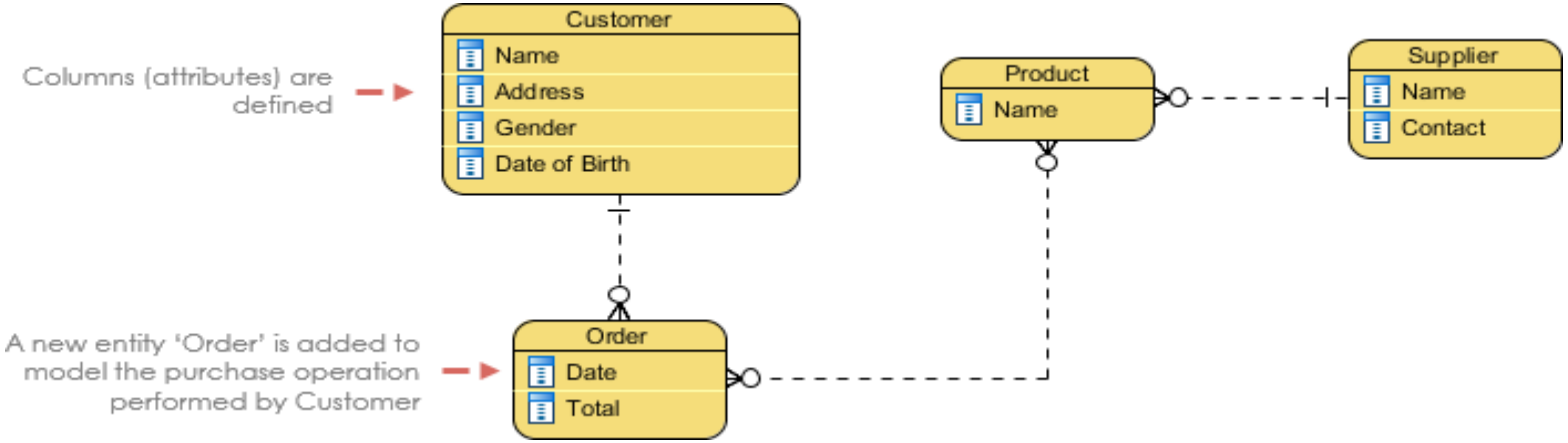
ERDS

- An ER model is typically drawn at up to three levels of abstraction:
- **Conceptual ERD / Conceptual data model**
- **Logical ERD / Logical data model**
- **Physical ERD / Physical data model**
- While all the three levels of an ER model contain entities with attributes and relationships, they differ in the purposes they are created for and the audiences they are meant to target.

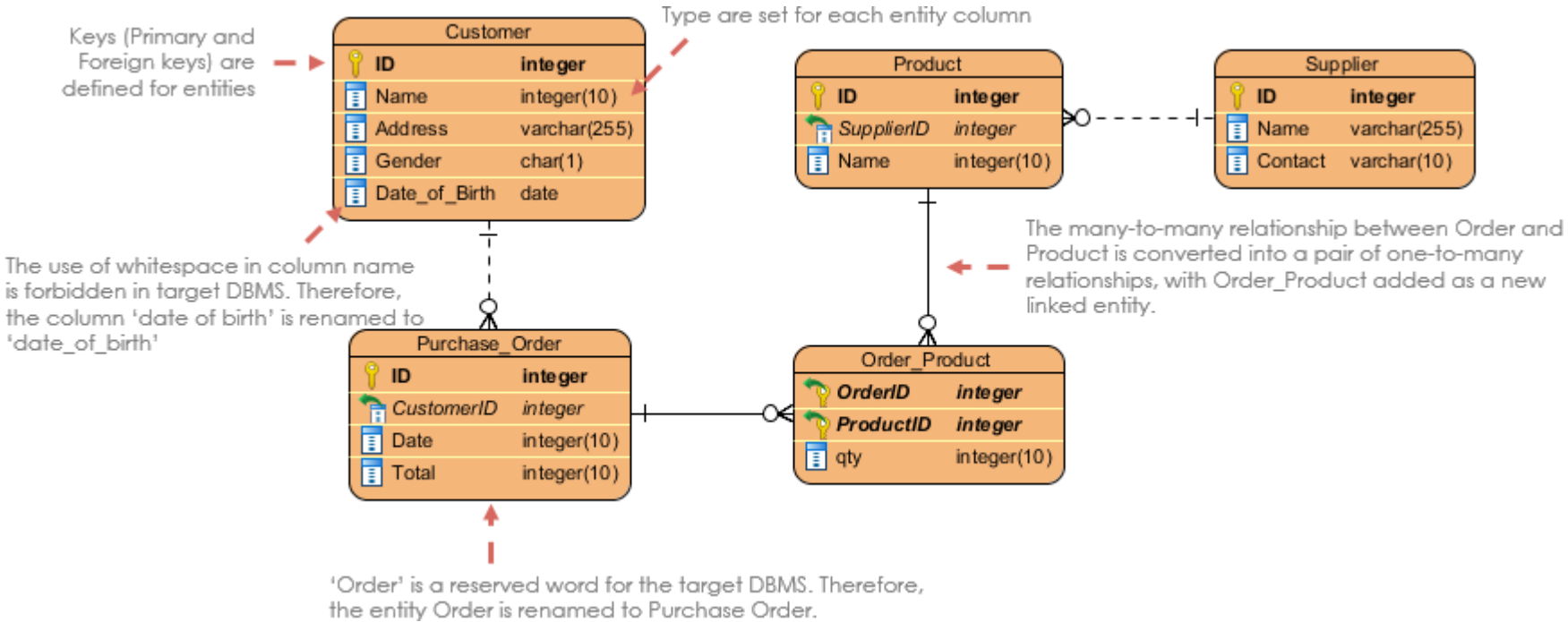
- Conceptual ERD models the **business objects that should exist in a system and the relationships between them**.



- Logical ERD is a **detailed version of a Conceptual ERD**. A logical ER model is developed to enrich a conceptual model by defining explicitly the columns in each entity and introducing operational and transactional entities.



- Physical ERD represents the **actual design blueprint of a relational database**. A physical data model elaborates on the logical data model by assigning each column with type, length, nullable, etc. Since a physical ERD represents how data should be structured and related in a specific DBMS it is important to consider the convention and restriction of the actual database system in which the database will be created.



ERD

ER diagram of Company has the following description :

Company has several departments.

Each department may have several Location.

Departments are identified by a name, D_no, Location.

A Manager control a particular department.

Each department is associated with number of projects.

Employees are identified by name, id, address, dob, dat e_of_joining.

An employee works in only one department but can work on several project.

We also keep track of number of hours worked by an employee on a single project.

Each employee has dependent

Dependent has D_name, Gender and relationship.

AN ENTITY RELATIONSHIP DIAGRAM METHODOLOGY

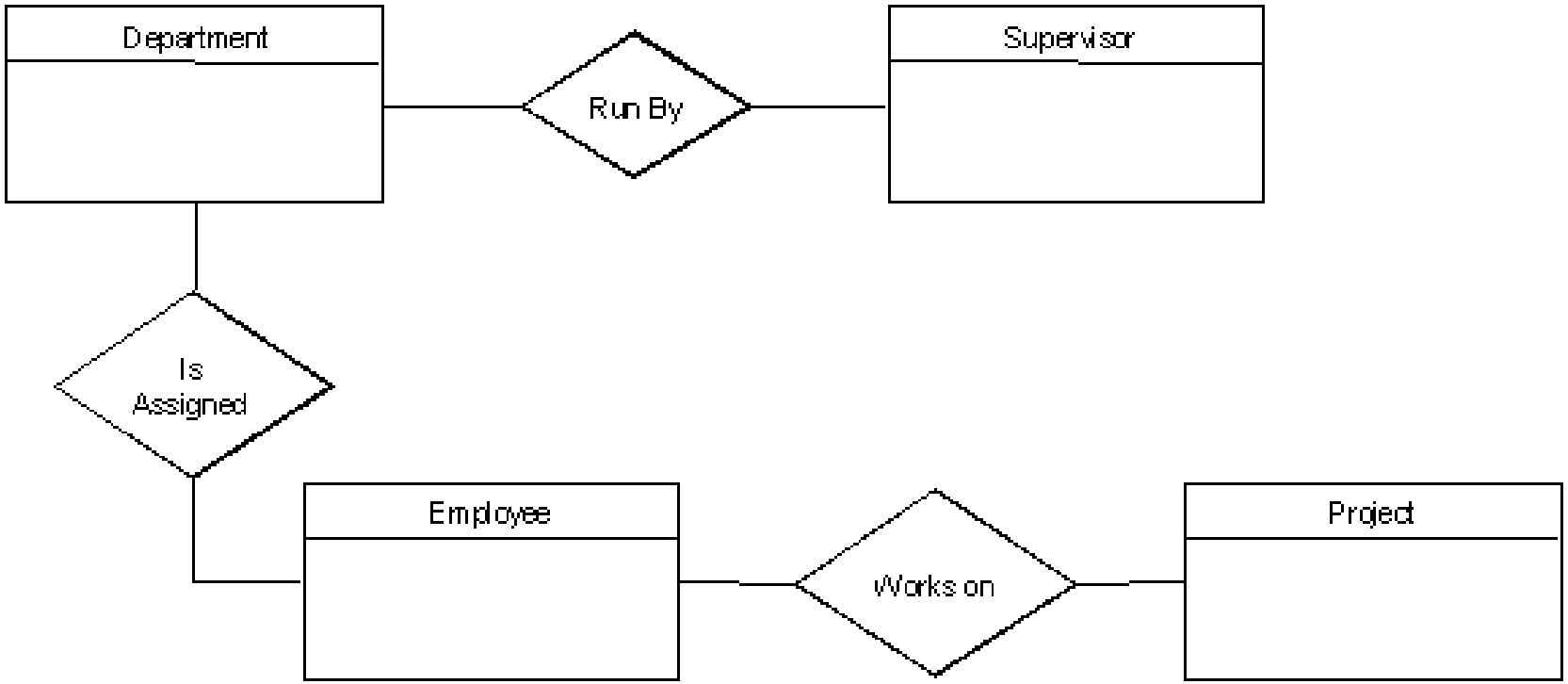
1. Identify Entities	Identify the roles, events, locations, tangible things or concepts about which the end-users want to store data.
2. Find Relationships	Find the natural associations between pairs of entities using a relationship matrix.
3. Draw Rough ERD	Put entities in rectangles and relationships on line segments connecting the entities.
4. Fill in Cardinality	Determine the number of occurrences of one entity for a single occurrence of the related entity.
5. Define Primary Keys	Identify the data attribute(s) that uniquely identify one and only one occurrence of each entity.
6. Draw Key-Based ERD	Eliminate Many-to-Many relationships and include primary and foreign keys in each entity.
7. Identify Attributes	Name the information details (fields) which are essential to the system under development.
8. Map Attributes	For each attribute, match it with exactly one entity that it describes.
9. Draw fully attributed ERD	Adjust the ERD from step 6 to account for entities or relationships discovered in step 8.
10. Check Results	Does the final Entity Relationship Diagram accurately depict the system data?

A company has several departments. Each department has a supervisor and at least one employee. Employees must be assigned to at least one, but possibly more departments. At least one employee is assigned to a project, but an employee may be on vacation and not assigned to any projects. The important data fields are the names of the departments, projects, supervisors and employees, as well as the supervisor and employee number and a unique project number.

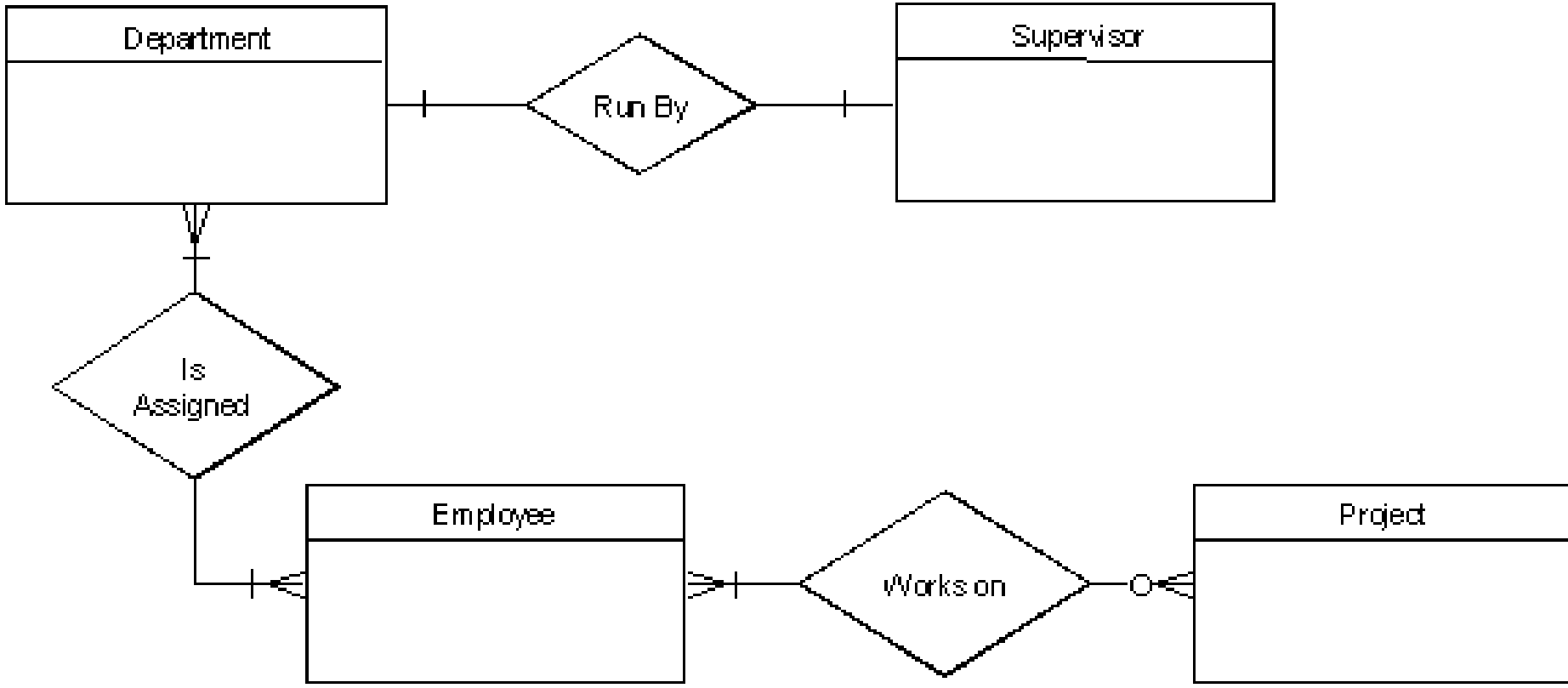
- We construct the following Entity Relationship Matrix:

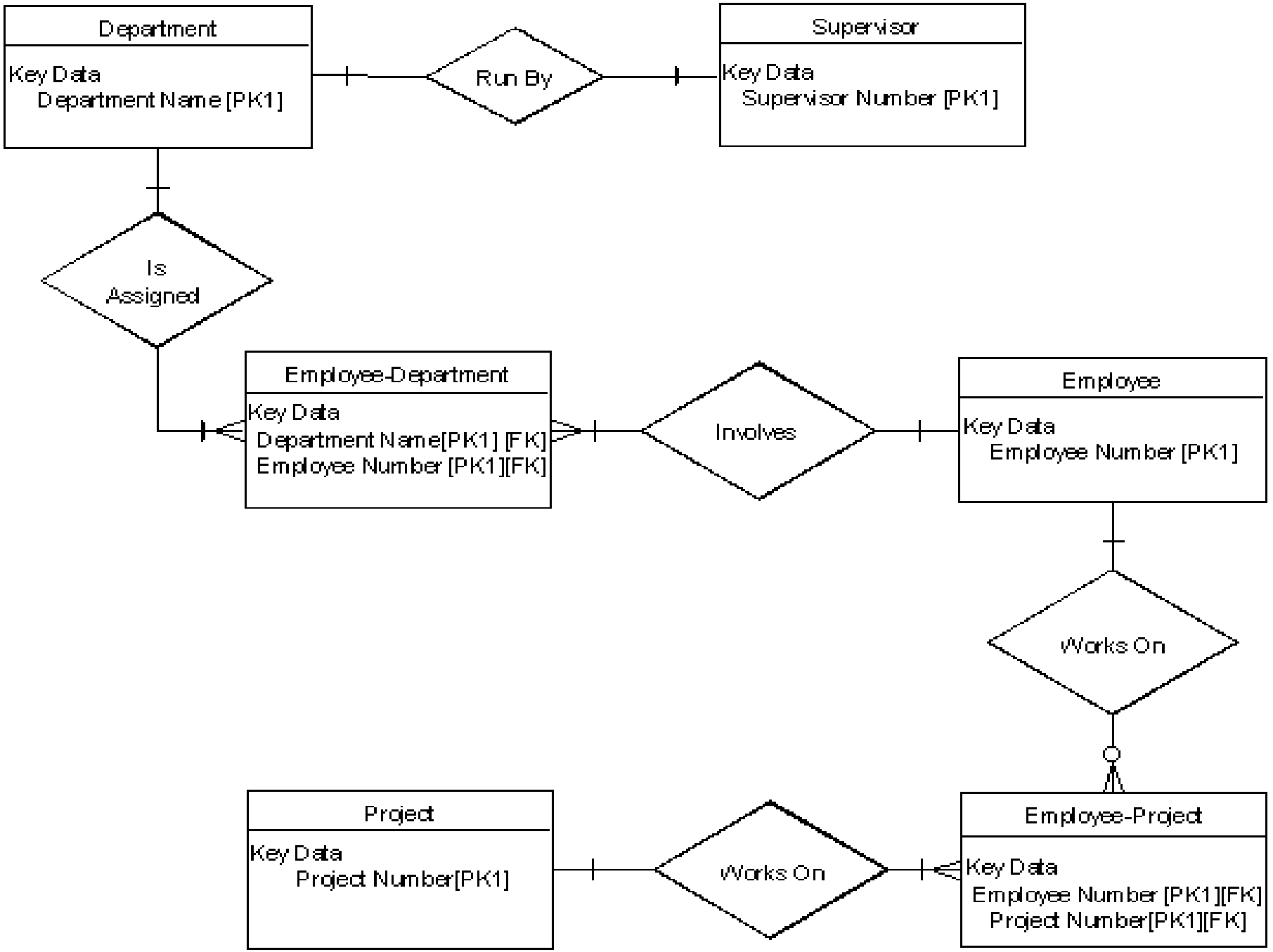
	Department	**Employee**	**Supervisor**	**Project**
Department		is assigned	run by	
Employee	belongs to			works on
Supervisor	runs			
Project		uses		

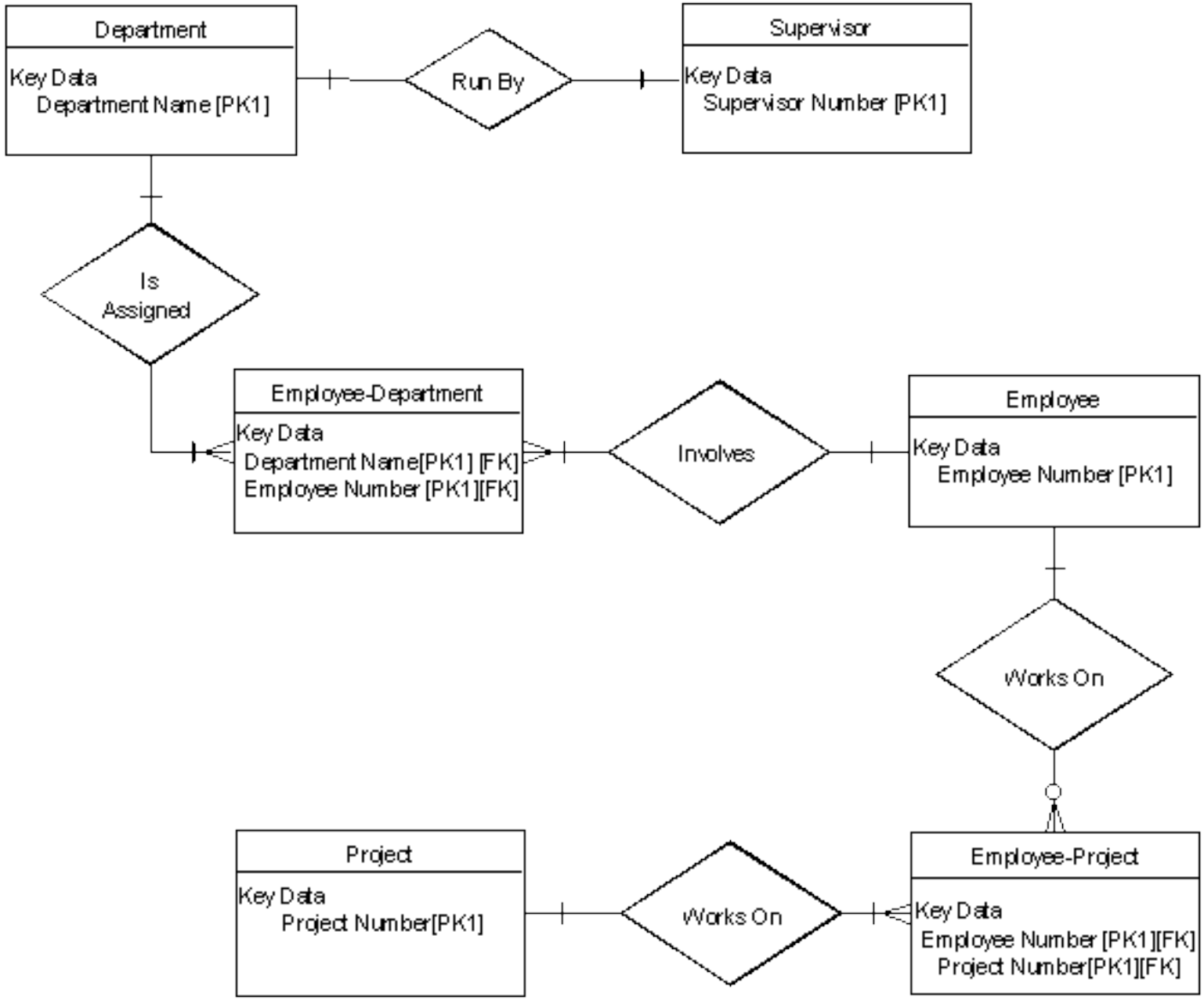
Rough ERD


Department
Run By
Supervisor
Is Assigned
Employee
Works on
Project

- **Fill in Cardinality**
- From the description of the problem we see that: Each department has exactly one supervisor.
- A supervisor is in charge of one and only one department.
- Each department is assigned at least one employee.
- Each employee works for at least one department.
- Each project has at least one employee working on it.
- An employee is assigned to 0 or more projects.


Department
Run By
Supervisor
Is
Assigned
Employee
Works on
Project


Department
Key Data
Department Name [PK1]
Run By
Supervisor
Key Data
Supervisor Number [PK1]
Is
Assigned
Employee-Department
Key Data
Department Name[PK1] [FK]
Employee Number [PK1][FK]
Involves
Employee
Key Data
Employee Number [PK1]
Works On
Project
Key Data
Project Number[PK1]
Works On
Employee-Project
Key Data
Employee Number [PK1][FK]
Project Number[PK1][FK]


Department
Key Data
Department Name [PK1]
Run By
Supervisor
Key Data
Supervisor Number [PK1]
Is
Assigned
Employee-Department
Key Data
Department Name[PK1] [FK]
Employee Number [PK1][FK]
Involves
Employee
Key Data
Employee Number [PK1]
Works On
Project
Key Data
Project Number[PK1]
Works On
Employee-Project
Key Data
Employee Number [PK1][FK]
Project Number[PK1][FK]

Data Independence

- Data independence can be defined as the capacity to change the schema at one level without changing the schema at next higher level. There are two types of data Independence.
- They are
  - 1. Logical data independence.
  - 2. Physical data independence.

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

  2. Physical data independence is the capacity to change the internal schema without changing the conceptual schema.

- Types of Databases and Database Applications • Traditional Applications:
  - Numeric and Textual Databases
- More Recent Applications:

  Multimedia Databases

  Geographic Information Systems (GIS)

  Data Warehouses Real-time

  and Active Databases Many other application