UNIT I

CHAPTER 1

Introduction to DBMS

Syllabus

- Introduction to Database Management Systems
- Purpose of Database Systems
- Database-System Applications
- View of Data
- Database Languages
- Database System Structure
- Data Models
- Database Design and ER Model : Entity, Attributes, Relationships, Constraints, Keys, Design Process, Entity Relationship Model
- ER Diagram
- Design Issues
- Extended E-R Features
- Converting ER & EER diagram into tables

Syllabus Topic : Introduction to Database Management Systems

1.1 Introduction to Database Management Systems

- **Data :** Data is the information which has been translated into a form that is more convenient to process or move.
- **Database :** The collection of related data is termed as Database which is organized in such a way that it can be easily retrieved and managed.
- **Database Management System**
  - A Database Management System (DBMS) is system software which manages the data. It can perform various tasks like creation, retrieval, insertion, modification and deletion of data to manage it in a systematic way as per requirement.
  - Database systems are designed to manage large amount of data by providing security from accidental crash of system and unauthorized access. DBMS provides convenient and efficient environment which used to handle the data.

Syllabus Topic : Purpose of Database Systems

1.2 Purpose of Database Systems

- Programming languages like Java, .Net are used to develop customized software's. Every software or application has its data to be stored permanently.
- Programming languages cannot store data permanently. For this purpose we have to use the Database Management System. The DBMS plays a significant role in storing and managing data.
- In an application we store data in DBMS and for operations like insertion, modification or deletion we write code in programming languages i.e. software is usually created with the help of both Programming language and Database.
- When the application is executed on client side, the client or user interacts with interface of application which is created in programming language.

- The database always remains backside and do not come in front of the user. Hence the database is known as backend while programming language is termed as frontend.
- To understand the purpose or need of database system, we need to study the previous option to store data which is called as File Processing System.

1.2.1 File Processing System

- In our day to day life, number of times we need to store data in such way that it should be easily accessible whenever required.
- The data may be of bank transaction details, daily expenses, employee details, product details etc. Before computers such data was stored with the help of papers.
- After invention of computers, it becomes easy to store data with the help of files. In the early days, database applications were built on top of file systems.
- Traditional File Processing System is a computer based system in which all the information is stored in various computer files.
- It stores data in a systematic way that the different departments of an organization can store their data in set of files which helps to manage and differentiate the data.
- Initially the Traditional File Processing System seems to be useful but as the requirement of data processing and the size of data increases, the drawbacks of this system comes in picture.

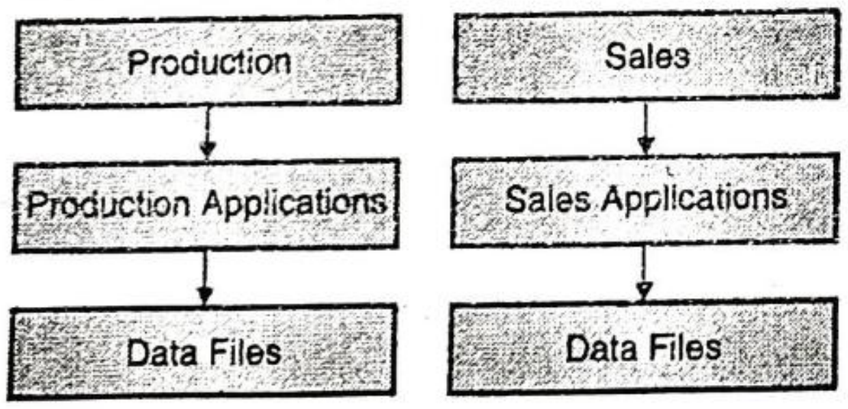


Fig. 1.2.1

1.2.2 Drawbacks of Traditional File Processing Systems

Consider an example of IT company database system where the data of all the employees is stored. In a professional IT firm the tasks are always done as teamwork. Different teams are assigned for different projects. Hence the data is stored in different files so as to differentiate it.

1. Data Redundancy

- Sometimes as per requirement same data may be stored in multiple files. Consider an employee having record in both Employee and Team files. The name and address of employee is stored in both of these files.
- Means the data get duplicated. If such data increases, it leads to higher storage and access cost. This duplication of data in various files is termed as data redundancy.
- In traditional file system, it is very difficult to avoid this data redundancy.

2. Data Inconsistency

- When data is to be updated the data redundancy may lead to data inconsistency. Data inconsistency occurs when data is not updated in all the files simultaneously.
- For example if the designation of employee get changed, then the respective changes should be made in both Employee and Team file. If for some reason, it is not done, then it leads to data inconsistency.
- Because for the sample employee, we may get different information which may create problems in the processing of data.

3. Limited Data Sharing

- It is difficult to share data in traditional file system. Each application has its own private files and users have little choice to share the data outside their own applications.
- To share data, we have to write complex programs.

4. Difficulty In Accessing Data

- The need of data access varies time to time. Means different types of information is needed at different situations.
- For example just consider that we want to retrieve the data of employees who do not have taken any leave throughout the quarter. In such case we have two options.
- We can access the data by manual method or we have to write an application program to

retrieve such customized data. Both the options are not convenient as both of them leads to wastage of time.

- If we do it, then also it may be possible that after some time we may require data with some another filter criteria. The data retrieval for customized information becomes difficult because the conventional files system does not provide any efficient and convenient way to retrieve the data.

5. Data Dependence

- In the files, data is stored in some specific format tab, semicolon or comma. If the format of any of the file is changed, then we have to make changes in program which processes the file.
- But sometimes there may be many programs related with the same file. In such case changes in all such programs should be done. Missing changes in single program may lead to failure of whole application.

6. Poor Data Control

- The Traditional File System does not have centralized data control; the data is de-centralized or distributed. In this system the same field may have different names in files of different departments of an organization.
- This situation may lead to different meaning of same data field in different context or same meaning for different fields. This causes poor data control.

7. Problem of Security

- It is very difficult to enforce security checks and access rights in a traditional file system. To the file we can set password protection.
- But what if we have to give access to only few records in the file? For example, in our database system, the project manager should be able to see all the data regarding teams under him.
- The team leader should be able to see data about his specific team. But payment details of one project manager should not be accessible to his team members or any another project manager. In the conventional file processing system, the application programs are added in ad hoc manner (for specific purpose) which makes it difficult to enforce security constraints.

8. Concurrency Problems

- Concurrency means access of same data by multiple users at the same time. This is very important aspect as it leads to increase in performance of a system and faster response. Many advanced systems allow the concurrent access and manipulation of data.
- For example, in our system, consider a record of an employee is accessed and updated by multiple users simultaneously at a time. This may lead to inconsistency of data, if the concurrency is not controlled in a proper manner.
- In another example if multiple transactions are make updations on a same bank account, then it may show incorrect balance, if any other transactions try to access balance amount in between.
- It is very difficult to implement concurrency control mechanism on file processing system, which leads to incorrect or wrong data retrieval.

9. Poor Data Modelling of Real World

- It is difficult for File Processing System to represent the complex data and interfile relationships. This results in poor data modelling properties.
- That means the real world applications are difficult to implement using File Processing System.

10. Data Isolation

- It is difficult to store the entire data in a single file. It is distributed in different files as per the category.
- These files may be in different formats because of which it becomes difficult to write application programs to access the desired data from these files.

11. Integrity Problems

- Every enterprise has its own constraints while maintain data in the files. Suppose in employee files the employee ID must start with 'E'. Such constraints can be added while writing application programs.

- But later on if any new constraints are introduced by the enterprise, and then it becomes difficult to add these constraints again. The File processing system does not provide any functionality to handle this situation.

12. Atomicity Problem

- Failure in a computer system may occur any time. When failure occurs, if any transaction is in its midway then it may lead to some incorrect data updation in the system.
- Consider another example of bank transaction where some amount is transferred from account A to account B. Initially the balance from account A is accessed and debited by Rs. 1000. Then we are going to credit it in account B. But before that system crash occurs which halts the transaction.
- Now this situation leads to incorrect data updation in the balance of account A. In file processing system, it very difficult to handle such situation to maintain the atomicity of database. The purpose of Database Management System is to solve all these problems and give functionality to store and manage data in efficient and convenient way.

1.2.3 Advantages of Database Management System

1. Controlling Data Redundancy

- In File Processing System the different applications has separate files for data storage. In this case, the duplicated copies of the same data are created at many places.
- In DBMS, all the data of an organization is integrated into a single database.
- The data is recorded at only one place in the database and it is not duplicated. For example, the Employee file and the Team file contain several items that are identical.
- When they are converted into database, the data is integrated into a single database so that multiple copies of the same data are reduced to-single copy.
- Controlling the data redundancy helps to save storage space. Similarly, it is useful for retrieving data from database using queries.

2. Data Consistency

- The data consistency is obtained by controlling the data redundancy, If a data item appears only once, any update to its value has to be performed only once and the updated value is immediately available to all users.
- For example if there is change in designation of employee, then the changes are made in single centralized file which is available to all the users.

3. Sharing of Data

- In DBMS, data can be easily shared by different applications. The database administrator manages the data and gives rights to users to access the data.
- Multiple users can be authorized to access the same data simultaneously. The remote users can also share same data.

4. Data Independence

- In DBMS we can completely separate the data structure of database and programs or applications which are used to access the data.
- This is called as data independence. If any changes are made in structure of database then there is no need to make changes in the programs. For example you can modify the size or data type of a data items (fields of a database table) without making any change in application program.

5. Data Control

- The DBMS provides centralized data storage. Hence keeping control on data is very much easy as compared to Traditional File Processing System.
- As data is common for all the application, no possibility of any confusion or complication.

6. Security

- In DBMS the different users can have different levels of access to data based on their roles. In the college database, students will have access to their own data only, while their teachers will have access to data of all the students whom they are teaching.

- Class teacher will be able to see the reports of all the students in that class, but not other classes. The principal will have access to entire data.
- Similarly, in a banking system, individual operator and clerk will have limited access to the data while the bank manager can access the entire data.
- All these levels of security and access are not allowed in file system.

7. Control over Concurrency

- In a computer file-based system, if multiple users are accessing data simultaneously, it is possible that it may lead to some irrelevant data generation. For example, if both users attempt to perform update operation on the same record, then one may overwrite the values recorded by the other.
- Most database management systems have sub-systems to control the concurrency so that transactions are always recorded with accuracy.

8. Data Modelling of Real World

- The DBMS has many functionalities are provided to represent the complex data and interfile relationships.
- This helps to map the database with real world applications.

1.2.4 Disadvantages of Database Management System

1. Increased Costs

- To install Database Systems, we require standard software and hardware. Also to handle the Database System, highly skilled personnel are required.
- The cost of maintaining the software, hardware, and personnel required to operate and manage the database system is more.
- The cost of training, license, and regulation compliance also increases the overall expenses.

2. Complexity

- Sometimes because of higher functionality expectations, the design of Database may become very complex.
- To utilize such database with complete efficiency, all the stakeholders like database designers, developers, database administrators and end-users must understand the functionality.
- Failure in understanding the system can lead to wrong design decisions, because of which serious consequences for an organization may occur.

3. Size

The DBMS becomes extremely large piece of software because of the complexity of functionality occupying large amount of disk space and requiring substantial amounts of memory to run efficiently.

4. Frequent Upgrade/Replacement Cycles

- New functionalities are often added into DBMS by their vendors. These new features often come bundled in new upgraded versions of the same software.
- Sometimes these versions require hardware upgrades which increases expenses. Also work to train database users and administrators to properly use and manage the new features get increased.

5. Higher Impact of a Failure

- The DBMS is placed at centralized location which increases the vulnerability of the system.
- That means the DBMS may get attacked and harmed. Since all users and applications rely on the centralized database, the failure of any component can bring operations to a halt.

6. Performance

- Usually, a File Based system is written for a specific application. Hence, the performance is generally very good.
- While the DBMS is written to be more general, to cater for multiple applications rather than any specific one.
- Because of which some applications may not run as fast as they used to.
- There are number of advantages of DBMS over File System.

1.2.5 Difference between File Processing and DBMS

SPPU : Dec. 13, May 14, Dec. 15

University Questions

Q. Explain significant difference between File Processing and DBMS. **(Dec. 2013, 6 Marks)**

Q. Explain the advantages of DBMS over normal file system in detail. **(May 2014, 8 Marks)**

Q. List significant difference between File Processing and DBMS. **(Dec. 2015, 5 Marks)**

Sr. No.	File Processing System	Database Management System
1.	Duplicate data may exist in multiple files which lead to data redundancy.	The data is integrated into a single database which avoids data redundancy.
2.	Data inconsistency occurs when data is not updated in all the files simultaneously.	The data consistency is obtained by controlling the data redundancy
3.	It is difficult to share data in traditional file system.	In DBMS, data can be easily shared by different applications.
4.	In the files, data is stored in specific format. If the format of any of the file is changed, then we have to make changes in program which processes the file.	In DBMS we can completely separate the data structure of database and programs or applications which are used to access the data.
5.	The Traditional File System does not have centralized data control; the data is de-centralized or distributed.	The DBMS provides centralized data storage. Hence keeping control on data is very much easy.
6.	It is very difficult to enforce security checks and access rights in a traditional file system.	In DBMS the different used can have different levels of access to data based on their roles which provides strong security to data.
7.	Concurrency problems means updation of same data by multiple users may generate irrelevant results.	DBMS have sub-systems to control the concurrency.
8.	It is difficult for File Processing System to represent the complex data and interfile relationships. This results in poor data modelling.	The DBMS has many functionalities are provided to represent the complex data and interfile relationships. This helps to map the database with real world applications.

Syllabus Topic : Database-System Applications

1.3 Database-System Applications

For any enterprise, its data is very important which helps to manage the business as well as decide some strategies to survive and grow the business in this competitive world. A Database Management System is a computerized record-keeping system. It works as a container for collection of computerized data files.

The overall purpose of DBMS is to provide functionality to the users to create, store, retrieve and update the information contained in the database as per requirement. Information can be of an individual or an organization.

Databases touch all aspects of our lives. Some of the major **areas of application** are as follows :

- **Telecom :** In Telecom sector database is maintained to keep track of the information about calls made, customer details, network usage etc. Without using database systems it is difficult to maintain such huge amount of data which keeps track of every second.
- **Banking :** In banking sector the data related to transactions of customers is very huge. Such data can be comfortably stored in the Database Management System. The DBMS is also used for storing customer personal information, tracking day to day credit and debit transactions, generating bank statements etc.

- **Industry :** In industry database management plays an important role. In departments like production, sales or account it is very essential to store the data in systematic format. For example in production department the data about manufactured products is stored which is very useful for sales department to keep track of orders.
- **E-commerce :** There are number of online shopping websites such as ebay, Flipkart Amazon, etc. These sites store the information about various products, user addresses, their preferences and credit details. It can be implemented using Database Management System only.
- **Airlines :** In airlines, the data of ticket booking, flight schedule, employee details, and customer details is very huge. Such data can be managed using the Database Management System.
- **Education System :** In education sector, the schools, colleges, private institutes have to store data of students, teachers, exam schedules, accounts etc which can be handles by Database Management System.
- **Railway Reservation System :** Database Management System helps in keeping record of ticket booking, departure and arrival status of train etc.
- **Library Management System :** In the library there are thousands of books which make it very difficult to keep record of all the books in a register. The DBMS can be used to maintain this information about book issue dates, names of the books, authors and availability of the book.
- **Social Media Sites :** The social media websites are used to share our views and connect with our friends. Daily millions of users signed up for these social media accounts like Facebook, Whatsapp, Twitter, and Google plus. The data of these millions of users and their chats can store using Database Management System.

Syllabus Topic : View Of Data

1.4 View of Data

As we have seen a data base system is collection of related data and system software which manages the data. The data is generally stored in a detailed and complex manner. It is important to provide an abstract view of data to the user.

To understand the view of data, first we have to learn the concept of abstraction.

1.4.1 Abstraction

- Abstraction is an important feature of Database Management System. Extracting the important data by ignoring the remaining irrelevant details is known as **abstraction**.
- Database systems are usually made-up of complex data structures as per their requirements.
- To make the user interaction easy with database, the internal irrelevant details can be hidden from users. This process of hiding irrelevant details from user is called data abstraction.
- The complexity of database can be hiding from user by different level of abstraction as follows.

1.4.1.1 Levels of Abstraction SPPU - Dec. 13

University Question

Q. Explain in detail the different levels of abstraction. **(Dec. 2013, 4 Marks)**

There are three levels of abstraction :

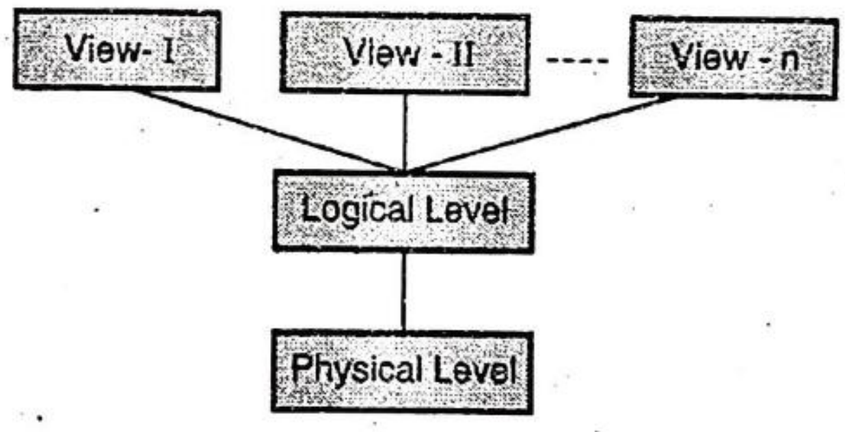


Fig. 1.4.1

1. Physical level

- In data abstraction Physical level is the lowest level. This level describes how the data is actually stored in the physical memory.
- The physical memory may be hard disks, magnetic tapes, etc. In Physical level the methods like hashing are used for organization purpose.
- Developer would know the requirement, size and accessing frequency of the records clearly in this level which makes easy to design this level.

2. Logical level

- This is the next higher level of abstraction which is used to describe what data the database stores,

and what relationships exist in between the data items. The logical level thus describes an entire database in terms of a small number of relatively simple structures.

- Although implementation of the simple structures at the logical level may involve complex physical level structures, the user of the logical level does not need to be aware of this complexity. This is considered as physical data independence.
- Database administrators use the logical level of abstraction to decide what information to keep in a database.

3. View level

SPPU - May 13

University Question

Q. Explain the need of view. **(May 2013, 3 Marks)**

- It is the highest level of data abstraction. This level describes the user interaction with database system. In the logical level, simple structures are used but still complexity remains because in the large database various type of information is stored.
- Many users are not aware of technical details of the system, and also they need not to access whole information from the database. Hence it is necessary to provide a simple and short interface for such users as per their requirements. Multiple views can be created for same database for multiple users.

  Example : Consider that we are storing information of all the employees of an organization in employee table. At **physical level** these records can be described as blocks of storage (bytes, gigabytes, terabytes etc.) in memory. These details are usually hidden from the developer.
- The records can be described as fields and attributes along with their data types at the **logical level**. The relationship between these fields can be implemented logically. Usually the developer works at this level because they have knowledge of such things about database systems.
- End user interacts with system with the help of GUI and enters the details at the screen at **view level**. User is not aware of how the data is stored and what data is stored; such details are hidden from them.

1.4.2 Schema

- The design of a database is called the schema. To understand schema we can consider an example of a program of an application. A variable or array declared with its structure (data type and/or size) is schema. The changes in schema or not frequent.
- **Types of Schema :** According to the level of abstraction, the database schema is divided into three types : Physical schema, Logical schema and View schema.

  (i) Physical schema is the design of a database at physical level, i.e. how the data stored in the blocks of storage is described in this level.

  (ii) Logical schema is the design of database at logical level. Developers and database administrators work at this level. Here the data can be described as certain types of data records gets stored in data structures, however the internal details like the implementation of data structure are hidden at this level.

  (iii) View schema refers to design of database at view level. This usually describes the end user interaction with database systems. There may be multiple schemas at view level.

1.4.3 Instance

In database changes are quite frequent i.e. insertion, deletion or updation are the frequent operations on database. The data is stored in the database at particular moment is called as instance of the database. In the example of application program, the value of a variable at particular time or situation is called as instance of database schema.

Syllabus Topic : Database Languages

1.5 Database Languages

In general in a database system, the Data Definition Language is used to specify schemas (design) of a database while the data manipulation language is used to fire queries (commands) on database in order to manipulate it. Both are the main pillars of database language like SQL.

With DDL and DML the database system also has the languages like DCL and TCL for different functionalities. The database languages are categorized as DDL, DML, DCL and TCL.

1.5.1 Data Definition Language (DDL)

- The DDL specify the schema of database by set of definitions.
- This language allows the users to define data and their relationship to other types of data. It is used to create data tables, dictionaries, and files within databases.
- The DDL is also used to specify the structure of each table, set of associated values with each attribute, integrity constraints, security and authorization information for all the tables and physical storage structure of all the tables on the disk.
- The data values stored in the database should satisfy some constraints for consistency purpose. For example the salary of an employee should never be negative or the employee id should start with 'E' etc. The Data Definition Languages provides functionality to specify such constraints.

1.5.2 Data Manipulation Language (DML)

- The Data Manipulation Language (DML) is used for accessing and manipulating data in a database. DML provides a set of functionalities to support the basic data manipulation operations on the data stored in the database.
- The DMS is basically of two types.
  1) **Procedural DML** – There is a requirement of user to specify which data is required and how get this data.
  2) **Declarative DML** – Here is also requirement of user to specify which data is required and without specifying how get this data. This is easier to understand and useful than procedural DML.
- A concept of query is used for processing. Query is a statement or command which requests the retrieval of information from the database. The part of DML which involved information retrieval is known as query language. The three levels of abstraction are used in defining structure of data as well as to manipulate the data.

Syllabus Topic : Database System Structure

1.6 Database System Structure

SPPU Dec. 13

University Question

Q. Explain component and overall structure of DBMS. **(Dec. 2013, 10 Marks)**

- DBMS (Database Management System) acts as an interface between the user and the database.
- The user requests the DBMS to perform various operations (retrieve, insert, delete and update) on the database.
- The components of DBMS perform these requested operations on the database and provide necessary data to the users. The various components of DBMS are as shown below :

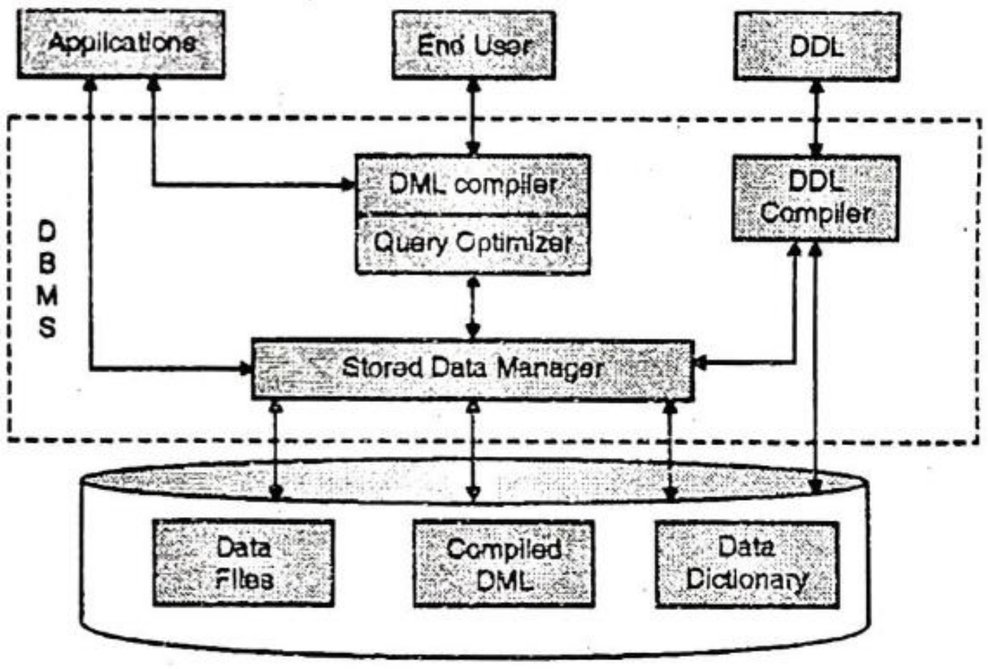


Fig. 1.6.1 : Database System Structure

Structure of DBMS

The Structure of DBMS contains following components :

1. DDL Compiler

- The DDL Compiler converts DDL commands into set of tables containing metadata stored in a data dictionary.
- The metadata information is name of the files, data items, storage details of each file, mapping information and constraints etc.

2. DML Compiler and Query optimizer

- The DML commands such as retrieve, insert, update, delete etc. from the application program are sent to the DML compiler for compilation. It converts these commands into object code for understanding of database.
- The object code is then optimized in the best way to execute a query by the query optimizer and then send to the data manager.

3. Data Manager

- The Data Manager is the central software component of the DBMS also known as Database Control System.
- The main functions of Data Manager are :
  - It converts the requests received from query optimizer to machine understandable form. It makes actual request inside the database.
  - Controls DBMS information access that is stored on disk.
  - It controls handling buffers in main memory.
  - It enforces constraints to maintain consistency and integrity of the data.
  - It synchronizes the simultaneous operations performed by the concurrent users.
  - It also controls the backup and recovery operations.

4. Data Dictionary

Data Dictionary is a repository of description of data in the database. It contains information about

- Data - names of the tables, names of attributes of each table, length of attributes, and number of rows in each table.
- Relationships between database transactions and data items referenced by them which are useful in determining which transactions are affected when certain data definitions are changed.
- Constraints on data i.e. range of values permitted.
- Detailed information on physical database design such as storage structure, access paths, files and record sizes.
- Access Authorization - is the Description of database users their responsibilities and their access rights.
- Usage statistics such as frequency of query and transactions.
- Data dictionary is used to actually control the data integrity and accuracy. It may be used as an important part of the DBMS.

Importance of Data Dictionary

- Data Dictionary is necessary in the databases due to following reasons:
- It improves the control of DBA over the information system and user's understanding for the use of the system.
- It helps in documenting the database design process by storing documentation of the result of every design phase and design decisions.
- It helps in searching the views on the database definitions of those views.
- It provides great assistance in producing a report of which data elements (i.e. data values) are used in all the programs.

5. Data File

It contains the data portion of the database i.e. it has the real data stored in it. It can be stored as magnetic disks, magnetic tapes or optical disks.

6. Compiled DML

- The DML complier converts the high level Queries into low level file access commands known as compiled DML.
- Some of the processed DML statements (insert, update, delete) are stored in it so that if there is similar requests, the data can be reused.

7. End Users

They are the real users of the database. They can be developers, designers, administrator or the actual users of the database.

Syllabus Topic : Data Models

1.7 Data Models

Basic Concept of Data Models

The process of analysis of data object and their

relationships to other data objects is known as data modeling. It is the conceptual representation of data in database. It is the first step in database designing. Data models define how data is connected to each other and how they are processed and stored inside the system. A data model provides a way to describe the design of a database at the physical, logical and view levels.

1.7.1 Types of Data Models SPPU - May 13

University Question

Q. Explain various Data Models used in DBMS. **(May 2013, 10 Marks)**

There are different types of data models :

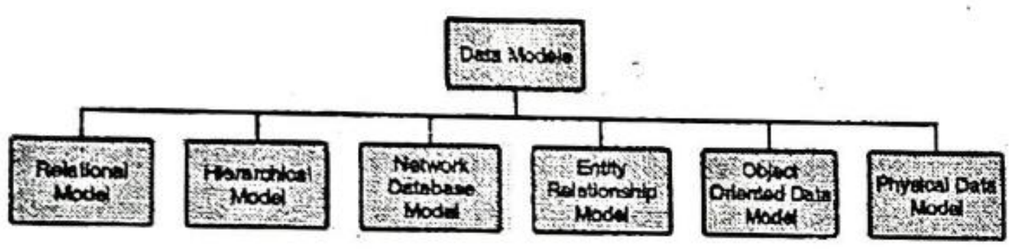


Fig. 1.7.1

1.7.1.1 Relational Model

- The relational model is developed by E.F. Codd. Relational database is a type of record-based relations.
- Relational database is an attempt to simplify the data structure by making use of tables. Tables are used to represent the data and their relationships. Table is a collection of rows and columns. Tables are also known as relations.
- Records are known as tuples and fields are known as attributes.
- The relational model is called as record based model because the database is structured in fixed format records of different types. A record is consists of fields or attributes.
- In the relational model, every record must have a unique identification or key based on the data.
- In following table Stud_ID is the key through which we can identify the record uniquely in the relation. Relational data model is the most widely used record-based data model.

Key

Tuple →

Stud_ID	Stud_Name	DOB
101	Prajakta	03/03/1995
102	Rakesh	13/01/1996
103	Rahul	16/08/1995

Advantages of Relational Data Model

a) Supports SQL

- For accessing the data in Relational data model we have a language known as Structured Query Language (SQL).
- This language is used to access, insert, update or delete the data from the table. By using relational data model we can execute the complex queries.

b) Flexible

- We can easily manipulate the information which is linked with various tables.
- We can extract the information from different table simultaneously by using this model.

1.7.1.2 Hierarchical Model

- A data model in which the data is organized into a tree structure is known as hierarchical data model.
- Hierarchical data model structure contains parent-child relationship where root of the tree is a parent which then branches into its children. It is another type of record based data model.
- The data is stored in the form of records. These records are connected to one another.
- A record is a collection of fields; each field contains only one value. The hierarchical data models represent data according to its hierarchy.

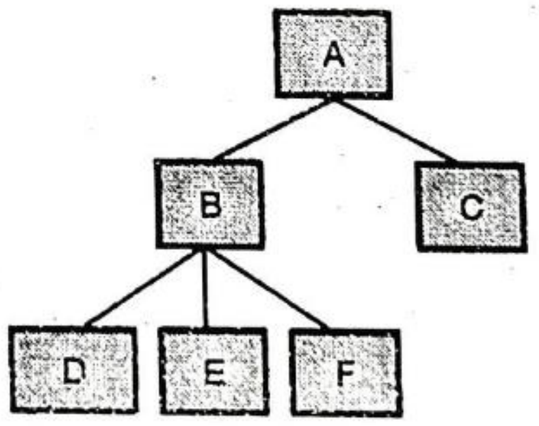


Fig. 1.7.2 : Hierarchical Model

- In hierarchical model there is a business rule. The rule is that, one parent node can have many child nodes but one child nodes can have only one parent node. This type of model is not in use currently.

Advantages of Hierarchical Model

a) Simple to understand : Due to its hierarchical structure it is easy to understand. Most of the time data have hierarchical relationship. Therefore, it is easy to arrange the data in that manner.

b) **Database Integrity :** In hierarchical data model there is always a parent child association between different records on different level. Due to this inherent structure integrity gets maintained.

c) **Efficient :** The performance of this model is very efficient due to its hierarchical structure when database contain large amount of data which has various related records.

1.7.1.3 Network Database Model

- It is extended type of hierarchical data model. This data model is also represented as hierarchical, but any child in the tree can have multiple parents.
- In network data model there is no need of parent child association. There is no downward tree structure.

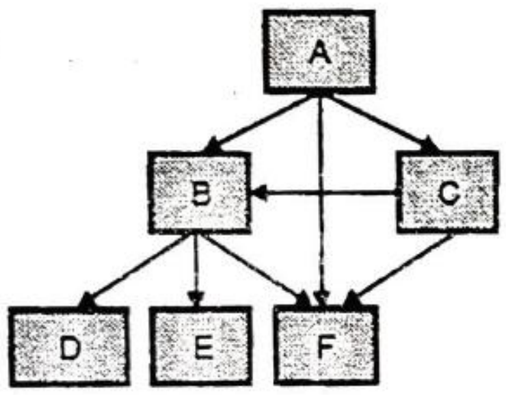


Fig. 1.7.3 : Network Model

- It is the flexible way of representing the objects and their relationship. A network data model allows multiple records linked in the same file.
- Basically, network database model forms a network like structure between the entities.

Advantages of Network Model

a) **Design is simple :** The network model is simple and easy to design and understand. There is no complex structure in network model.

b) **It has capability to handle various relationships :** The network model can handle the one to many and many to many relationships which is useful to develop the database.

c) **Easy to access :** The data access is easy and flexible than the hierarchical data model. In network model there is no any hierarchy in the objects and their relations therefore it is very easy to access the data in network model.

1.7.1.4 Entity Relationship (E-R) Model

- This model describes inter-related things of interest in a specific domain of knowledge. An ER model is composed of entity types (which classify the things of interest) and specifies relationships that can exist between instances of those entity types. In the next Sections 1.10 and 1.11 we will study this model in detail.

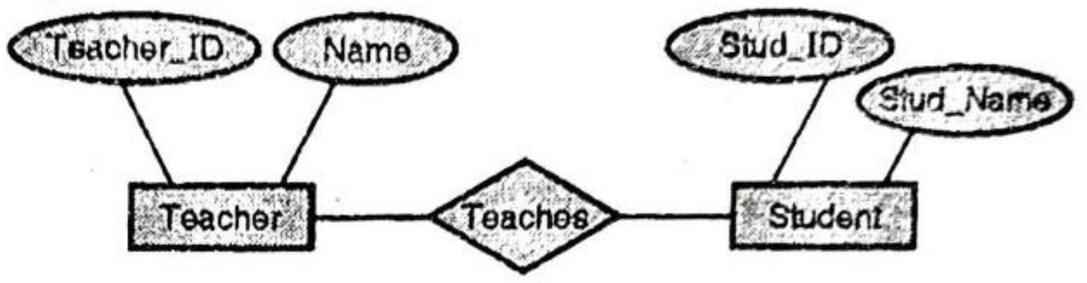


Fig. 1.7.4 : ER Model

Advantages of Entity Relationship Model

a) **Simple Design :** The ER model is simple and easy to design. It shows the logical view of the data and its relationships. This model is easy to understand.

b) **Effective representation :** The presentation of Entity Relationship Model is very simple and effective. The programmer and designer can easily understand the flow of the system by referring the ER Model.

c) **Connected with Relational Model :** The Entity Relationship Model is connected with the relational model. Due to this advantage we can develop a well-structured design.

1.7.1.5 Object Oriented Database Model

- Object oriented data model is nothing but the collection of objects or elements with its methods.
- Now a days all advanced programming languages supports the concepts of Object Oriented Programming.
- This model is based on the Object-oriented paradigm. This paradigm is defined for the database.
- It is extension to E-R model with integration of concepts like object, method and encapsulation.
- Applications of data modelling contain the key concepts of object-oriented.

Advantages of Object- Oriented Data Model

a) Object oriented data Model supports inheritance. Due to this we can reuse attributes and functionalities. It reduces the cost of maintaining data multiple times.

b) Due to inheritance, and encapsulation this model become more flexible.

c) We can bind each class with its attributes and functionality. It helps in representing the real world objects.

d) We can see each object as a real world entity in this model. Hence it is more understandable.

1.7.1.6 Physical Data Model

- The description of data storage in the computer in the form of information is provided by Physical data models.
- Physical data model shows the table structure which includes column name, column data, constraints and relationship between tables. Physical data models define all the components of logical database.

Advantages of Physical Model

a) We can specify the all tables and columns using physical data model.

b) We can use foreign key to join the tables. By using foreign key we can identify the related data from the different tables.

c) By using physical data model we can convert entity into the table.

Syllabus Topic : Database Design and ER Model

1.8 Database Design and ER Model

1.8.1 Database Design

In the creation of database system, the design of database is the most important and initial part. The overall success of the system is completely depends upon the design of database. The design is mainly focused on the security and access of data by the application program. The requirements of user play an important role in database design.

The process of doing database design generally consists of a number of steps which will be carried out by the database designer. Usually, the designer must:

- Determine the data to be stored in the database.
- Determine the relationships between the different data elements.
- Superimpose a logical structure upon the data on the basis of these relationships.

Determine the data to be stored in the database

- Number of times, the database designer is a person with expertise in the area of database design. It is not necessary that he should be expertise in the domain from which is the data is retrieved e.g. financial information, biological information etc.
- Hence it is important that the data to be stored in the database should be determined in cooperation with the domain expertise that has knowledge that which data must be stored within the system.
- This process is a part of requirements analysis and required that the database designer should get the required data from domain expertise.
- The reason is that number of times the domain expertise cannot express exactly what their system requirements for the database are unfamiliar to thinking in terms of the discrete data elements which are needed to be stored.
- Data to be stored can be determined by Requirement Specification. In the life cycle of software it is known as requirement gathering phase.

Determining data relationships

- Once a database designer is aware of the data which is to be stored within the database, they must then determine where dependency is within the data. Sometimes when data is changed you can be changing other data that is not visible.
- For example, in a list of names and addresses, assuming a situation where multiple people can have the same address, but one person cannot have more than one address; the address is dependent upon the name.
- When provided a name and the list the address can be uniquely determined; however, the inverse does not hold - when given an address and the list, a name cannot be uniquely determined because multiple people can reside at an address.
- Because an address is determined by a name, an address is considered dependent on a name.

Logically structuring data

- Once the relationships and dependencies amongst the various pieces of information have been determined, it is possible to arrange the data into a logical structure which can then be mapped into

the storage objects supported by the database management system.

- Now the important part in design is that how to represent things like person, product, place etc. These things can be represented in term of entities. The various entities are related with each other by one or other way.

In database design schema, it is necessary to avoid two major problems.

1. Redundancy

- Sometimes as per requirement same data may be stored in multiple files. Consider an employee having record in both Employee and Team files:
- The name and address of employee is stored in both of these tables. Means the data get duplicated. If such data increases, it leads to higher storage and access cost.
- This duplication of data in various files is termed as data redundancy. Such redundancy can also occur in relational schema also. In the schema of the database, such information may be repeatedly shown.

2. Incompleteness

- An incomplete design is very difficult to model. Suppose in a IT enterprise database system, if a department like R& D is there, to which no employee is still appointed, then it becomes very difficult to represent information about this department.

1.8.2 E-R Model

- The Entity Relationship (E-R) model allows specifications of an enterprise schema and represents the overall logical structure of the database.
- Now a days the database related to real world applications becomes very vast and complex. Representing relations between the different elements of the database becomes difficult.
- ER Model simplifies this task. It is nothing but the design technique for database. It is a graphical technique which helps to understand and organize the complex data which should not depend upon the actual database implementation.
- The real world objects can be easily mapped with entities of E-R model.
- In Entity Relationship Model a graphical representation of a database system is generated. Diagrams are used in this model. These diagrams are known as entity-relationship diagrams, ER diagrams or ERDs.
- Basic concepts of ER Model are as follows :

Entity, attribute, Relationship, constraints and keys.

Syllabus Topic : Entity

1.8.2.1 Entity and Entity Set

- The E-R model consists of entities and relationships between those entities. An entity is nothing but a thing having its own properties. These properties helps to differentiate the object (entity) from other objects.
- An entity is a thing that exists either physically or logically. An entity may be a physical object such as a house or a car, or an entity may be a logical concept like an event such as a house sale or a car service, or a concept such as a customer transaction or order.
- An entity set is a set of entities which share the same properties. In a Company employee is the entity set which has similar properties like Employee_ID, emp_name, salary etc.
- There is difference between an entity and an entity-type. An entity-type is a category. An entity, strictly speaking, is an instance of a given entity-type. There are usually many instances of an entity-type
- There are two types of entities in Database management system.

1. Strong Entity or Regular Entity

- If an entity having it's own key attribute specified then it is a strong entity. Key attribute is used to indentify that entity uniquely among set of entities in entity-set.
- **Example :** In a parent/child relationship, a parent is considered as a strong entity.
- Strong entity is denoted by a single rectangle.

- The relation between two strong entities is denoted by a single diamond simply called relationship.

2. Weak Entity

- The entity which does not have any key attribute is known as weak entity. The weak entity has a partial discriminator key. Weak entity depends on the strong entity for its existence. Weak entity is denoted with the double rectangle.
- **Example :** In a parent/child relationship, a child is considered as a weak entity which is completely depends upon the strong entity 'parent'.

Comparison of an Object is OOP and Entity in E-R Model SPPU - May 13

University Question

Q. How does the concept of an object in the object oriented model differ from the concept of an entity in the E-R model ? **(May 2013, 8 Marks)**

Sr. No.	Entity	Object
1.	An entity is a thing that exists either physically or logically. An entity may be a physical object such as a house or a car (they exist physically), an event such as a house sale or a car service, or a concept such as a customer transaction or order (they exist logically as a concept)	Object in Object oriented programming is nothing but anything which has its own properties. E.g. flower is an object having properties like color, fragrance etc.
2.	Even if data in two entity instances is the same, we don't deem them as equivalent.	Objects don't have an identifier field and if two objects have the same set of attributes we can treat them interchangeably.
3.	Entities live in continuum that means they have a history of what happened to them and how they changed during their lifetime.	Objects, at the same time, have a zero lifespan. We create and destroy them with ease.
4.	Entities are mutable. That means we can change them.	Objects should be immutable that means we cannot change them rather we construct a new object based on the existing object.

Syllabus Topic : Attributes

1.8.2.2 Attribute

- An attribute is a characteristic of an entity. Entities are represented by means of their attributes. All attributes have their own specific values. For example, an employee entity may have Employee_ID, emp_name, salary as attributes.
- In a database management system an attribute is a database component, such as field or column of a table.

Example :

The entity student has attributes like student_id, student_name. In this every attribute has a value. Here 101 is the value for the attribute student_id, Kunal is the value for attribute student_name.

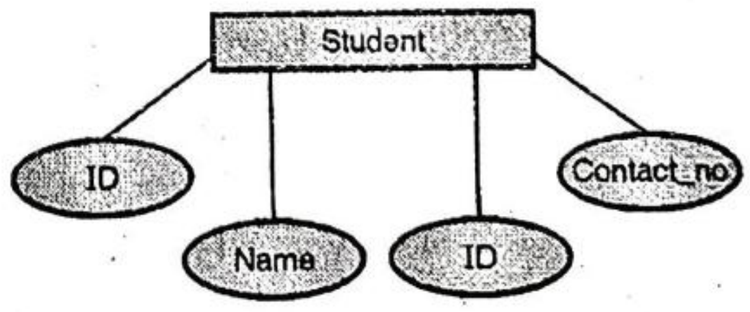


Fig. 1.8.1

There are five different types of attributes in Database Management System :

1. Single-valued Attribute

- A single-valued attribute is the attribute which can hold a single value for the single entity.
- **Example :** In the entity student, student_name is the single-valued attribute since a student have a single value for name attribute.

2. Multi-valued Attribute

- A multi-valued attribute is the attribute which can hold multiple values for the single entity.
- **Example :** In the entity student, the attribute student_contact no could be considered a multi-value attribute since a student could have multiple contact numbers.

3. Simple Attribute

- An attribute whose value cannot be further divided is known as simple attribute. That means it is atomic in nature.
- **Example :** In the entity student, the attribute student_age cannot be divided. Therefore student_age is the simple attribute of student entity.

4. Composite Attribute

- The composite attributes are the attributes which can be further divided into sub parts. These sub parts represent the basic entities with their independent meaning.
- **Example :** In the entity student, student_name is the composite attribute, we can divide this attribute in three different sub parts: First_name, Middle_name and Last_name.

5. Derived Attribute

- The attribute which is not physically exist in database, but its value can be calculated from the other present attributes is known as derived attribute.
- **Example :** In the entity student, we can calculate the average age of students. This average age is not physically present in the database but it can be derived from the attribute student_age;

Syllabus Topic : Relationships

1.8.2.3 Relationships

The association between two different entities is called as relationship. In the real world application, what does one entity do with the other, how do they connect to each other ?

For example : An employee works at a department, a student enrolls for a course. Here, works at and Enrolls are called relationships.

The Degree of Relationships

The degree of relationship refers to number of entities participated in the relation.

1. Unary Relationship

- A unary relationship exists when there is relation between single entity. A unary relationship is also known as recursive relationship in which an entity relates with itself.
- **Example :** A person can be in the relationship with another person, such as :
- A woman who can be someone's mother
- A person that is a someone's child.

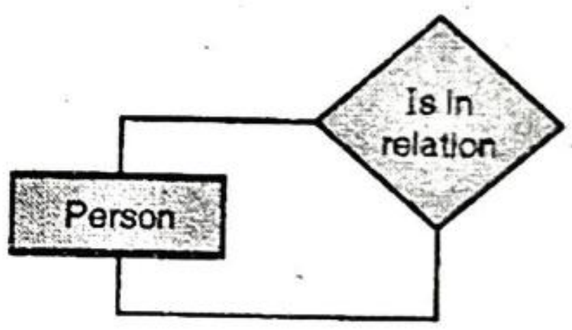


Fig. 1.8.2

2. Binary Relationship

- A binary relationship exist only when there is relation between only two entities. In this case the degree of relation is two.
- **Example :** A teacher teaches student. In this teacher and student are two different entities which are connected with each other via relation Teaches.

Fig. 1.8.3

3. Ternary Relationship

- A ternary relationship exists when there are relations between three entities. In ternary relation the degree of relation is six.
- **Example :** A person can be a student and a person also can be teacher. Here teacher, student and person are three entities which are related to each other.

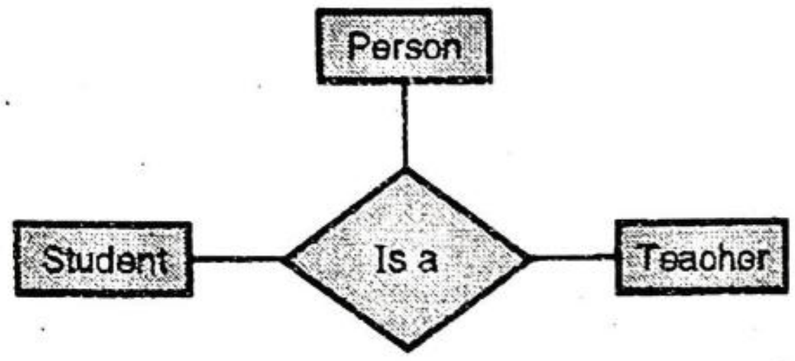


Fig. 1.8.4

4. Quaternary Relationship

- A quaternary relationship exists when there are relations between four entities. In quaternary relation the degree of relation is eight.
- **Example :** The four entities Employee, Management Faculty, Teaching Faculty, and Non-Teaching Faculty are connected with each other via is a relationship.

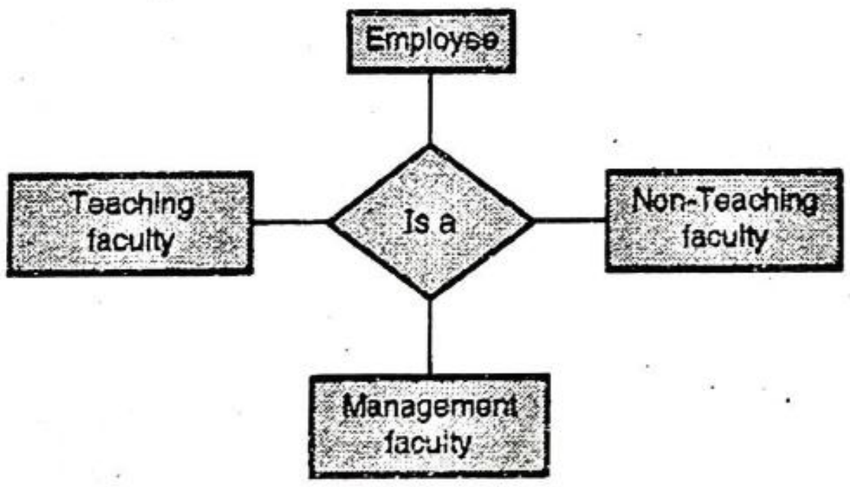


Fig. 1.8.5

Syllabus Topic : Constraints

1.8.2.4 Constraints

There are certain constraints in E-R enterprise schema to which the contents of a database must conform.

Two types of constraints are

1. Mapping cardinalities
2. Participation constraints

1. Mapping Cardinalities SPPU - Dec. 13, May 14

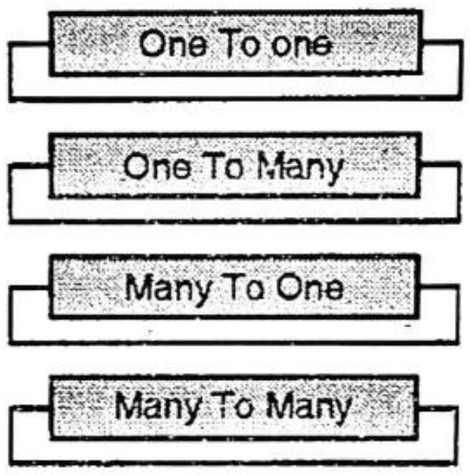
University Questions

Q. What is meant by mapping cardinality? **(Dec. 2013, 2 Marks)**

Q. What is meant by Mapping Cardinality? Explain different types of Cardinalities for a binary relationship with example. **(May 2014, 4 Marks)**

Cardinality in DBMS

- In Database Management System the term 'Cardinality' refers to the uniqueness of data values contained in a column.
- If the column contains a large percentage of totally unique values then it is considered as high cardinality while if the column contains a lot of "repeats" in its data range, it is known as Low cardinality.



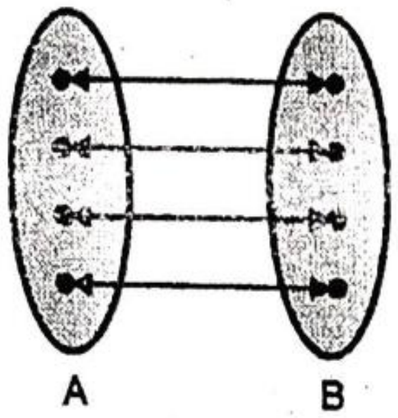
- Sometimes cardinality also refers to the relationships between tables. Cardinality between tables can be one-to-one, many-to-one or many-to-many.
- A relationship where two entities are participating is called a **binary relationship**.

Mapping Cardinalities

- Mapping cardinality expresses the number of entities to which another entity can be associated with in a relationship-set.
- It defines the relationship between two entities via a relationship set. Consider in a binary relationship set R (relation) between entity of set A and set B. For this relationship mapping cardinalities are as Follows :

One to One

- An entity of entity-set A can be associated with at most one entity of entity-set B and vice versa that means an entity in entity-set B can be associated with at most one entity of entity-set A.



(a)

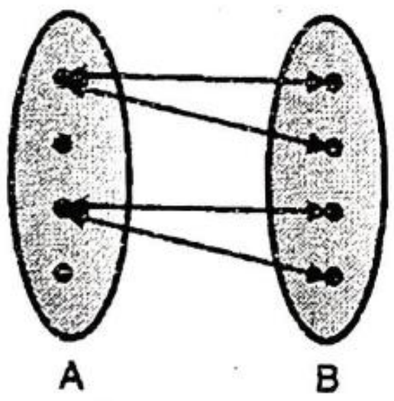
Example : In following example one state have only one capital.

(b)

Fig. 1.8.6

One to Many

- In this type an entity in set A is associated with many other entities in set B.
- But an entity in entity set B can be associated with maximum of one entity in entity set A.



(a)

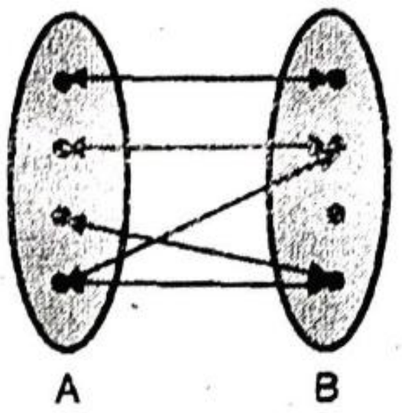
Example : In following example one teacher can teach many students.

(b)

Fig. 1.8.7

Many to One

- In this type an entity in set A is associated with at most one entity in set B. And an entity in set B can be associated with number of entities in set A.



(a)

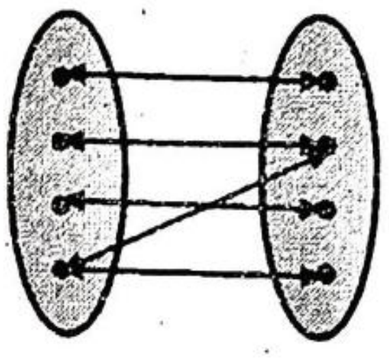
Example : In following example many students can enroll in one school.

(b)

Fig. 1.8.8

Many to Many

- In this type any entity in entity set A is associated with number of entities in entity set B.



(a)

- An entity in entity set B is associated with number of entities in set entity A.
- **Example :** Many students learn many subjects.

(b)

Fig. 1.8.9

The appropriate mapping cardinality for a particular relationship set is depending upon the real world situation to which the relationship set is modeling.

2. Participation Constraints

There are two types of participation constraints.

i. Total participation

ii. Partial participation

i. **Total Participation :** The participation of an entity set E in a relationship set R is said to be total if every entity in E participates in at least one relationship in R.

ii. **Partial Participation :** The participation of an entity set E in a relationship set R is said to be partial if only some entities in E participates in relationships in R.

- For example, in a college database system, consider teachers are assigned as project guides to every student. In this case every Student entity is related with teacher entity through the relationship "Guide".
- Hence participation of student in the relationship guidance is total. But it is not compulsory that every teacher should guide the students.
- Hence it is possible that not all the teacher entities are related with the student through the relationship "Guide". Here the participation of teacher in the "guide" relationship set is partial.

Syllabus Topic : Keys

1.8.2.5 Keys

SPPU - Dec. 16

University Question

Q. Explain the distinction among the terms primary key, candidate key, and super key. **(Dec. 2016, 5 Marks)**

The specification of differentiation of entities in a given entity set is very important. The entities can be differentiated in terms of attributes. Here the values of attributes come in picture. These values should be different to identify the attributes uniquely. It is necessary that in an entity set, no two entities should have same values for all the attributes.

In the database schema the notion of key is directly applies to entity sets. The key for an entity in entity set is an attribute or set of attributes which is used to distinguish entities from each other.

Keys are also used to identify relationships uniquely and differentiate these relationships from each other.

There are six types of keys available in DBMS :

1. Primary Key

- Primary key uniquely identify each entity in the entity set. It must have unique values and cannot hold null values. Let, R be a relationship set having entity sets E1,E2,....En. Consider primary key (Ei) denotes the set of attributes that forms the primary key for entity ser Ei. The set of attributes associated with the relationship set R is responsible for composition of primary key for that relationship set.
- **Example :** In Bank database, the account_number entity should be primary key. Because this field cannot be kept NULL as well as no account_number should be repeated.

2. Super Key

- This key is formed by combining more than one attributes for the purpose of uniquely identifying entities.
- **Example :** In student database having attributes Student_reg_id, Student_roll_no, Sutdent_name, Address, Contact_no.
- The Super keys are :

  {Student_reg_id}

  {Student_roll_no}

  { Student_reg_id, Student_roll_no }

  { Student_reg_id, Sutdent_name }

  { Student_roll_no, Sutdent_name }

  {Student_reg_id, Student_roll_no, Sutdent_name }
- It means super key can be any combination of attributes, so that identifying the record becomes easier.

3. Candidate Key

- Candidate key is formed by collection of attributes which hold unique values. A super key without redundant values is known as candidate key. Candidate keys are selected from the set of super keys.
- Candidate key are also known as minimal super key having uniqueness property. The attribute which do not contain duplicate value, may be a candidate key.

Example : In student database with attributes Student_reg_id, Student_roll_no, Sutdent_name, Address, Contact_no.

- The Candidate keys are :

  {Student_reg_id}

  {Student_roll_no}

  { Student_reg_id, Student_roll_no }

- It means candidate key can be any combination of key attributes, so that identifying the record from the table becomes easier.

4. Alternate Keys

- From all candidate keys, only one key gets selected as primary key, remaining keys are known as alternative or secondary keys.
- **Example :** In student table Student_address, Contact_no, Date_Of_Birth are the alternative keys.

5. Composite Key

- A key which consists of more than one attributes to uniquely identify rows in a table is called composite key. It is also known as compound key.
- **Example :** In student databse having attributes Student_reg_id, Student_roll_no, Sutdent_name, Address, Contact_no.
- The composite keys are :

  { Student_reg_id, Student_roll_no }

  { Student_reg_id, Sutdent_name }

  { Student_roll_no, Sutdent_name }

  { Student_reg_id, Student_roll_no, Sutdent_name }

These sets of keys are used to uniquely identify the record from the table.

Syllabus Topic : Design Process

1.9 Database Design Process

The database design process includes following six stages.

1. Requirement analysis
2. Conceptual database design
3. Choice of the DBMS
4. Logical Database Design
5. Physical design
6. Implementation

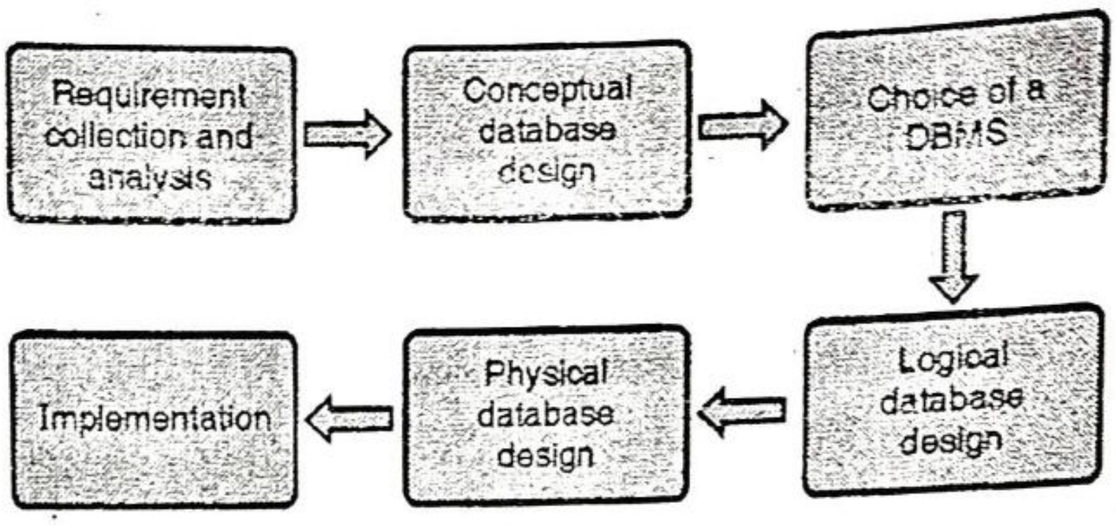


Fig. 1.9.1 : Steps of database design

1. Requirement analysis

- This stage involves the data gathering of requirements from user. Here we discover DATA and OPERATIONS requirements by interaction with the customer.
- The discussion about what current data is and which operations user expect to perform on the database.
- Here we also discuss how data elements are related with each other.

2. Conceptual database design

- The main Purpose of this stage is to produce a conceptual schema of the database using concepts of the high level data model.
- This stage does not include implementation details. It is expected that this design should understood by non-technical users.
- It has detail information about the all the objects of the domain.
- The conceptual database design is independent of the DBMS to be used.
- This design cannot be used directly to implement the database.

3. Choice of the DBMS

- The main purpose of this stage is to select the best suitable database framework for implementing the produced schema.
- The schema includes :
  - Type of DBMS (relational, network, deductive, Object Oriented, ...)
  - User and programmer interfaces
  - Type of query language

- The choice of database is also depends upon technical factors like whether the DBMS has support the required tasks or not.
- The economic factors are also get considered which includes
  - Software and hardware purchase and maintenance.
- Training of staff
- The organisational factors considered are :
  - Platforms supported, availability of vendor services

4. Logical Design

- The main Purpose of logical design is to transform the generic, DBMS independent conceptual schema in the data model of the chosen DBMS. This technique is also called as data model mapping.
- There are two stages of mapping
  1. System independent mapping : In this stage, there is no consideration of characteristics of database i.e. it is totally independent of database system.
  2. Tailoring to DBMS : The same model can be implemented by different DBMS.
- Logical design produces set of DDL statements in the language of the chosen DBMS

5. Physical Design

- The Purpose Physical Design is to select the specific storage structures and access paths for the database files.
- The performances factor is mainly considered.
- Performance of system is based on following factors.
  - **Response time :** The database access time should be less for data items which are frequently used by transactions.
  - **Space utilisation :** Less frequently used data and queries may be archived to save the space.
  - **Transaction throughput :** The number of transactions that can be processed per minute should be more.

6. Implementation

- This is the most important and last stage in design process. In it database is created.
- The DDL statements are compiled and executed.
- In this stage application programs are written with embedded DML statements
- From here operational phase will be initiated.

Syllabus Topic : Entity Relationship Model

1.10 Entity Relationship Model

- In 1976 Entity relationship model was developed. It is useful in conceptual design. It is the high level data model.
- This model defines the data objects and their relationship. It is the popular model in database.
- This model consists of entities and relationships between those entities. An entity is nothing but a thing having its own properties. These properties helps to differentiate the object (entity) from other objects.

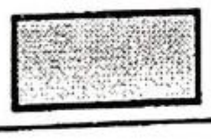
Syllabus Topic : ER Diagram

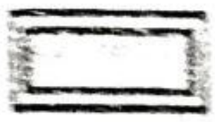
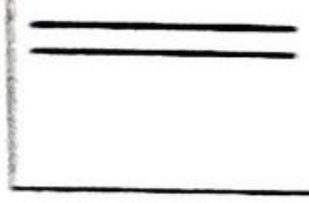
1.11 ER Diagram

The pictorial representation of data using different conventions which state that how these data are related with each other is known as Entity Relationship Diagram. E-R diagrams express the logical structure of database in graphical manner. Special symbols are used to draw an ER-Diagram. Every symbol has its own meaning.

Example of various symbols used in ER Diagram

Following are the symbol / notations used in ER-Diagram :

Symbol	Symbol Name	Symbol Description
Entities		
	Rectangle	**Entity**

Symbol	Symbol Name	Symbol Description
	Double rectangle	Weak entity
Attributes		
	Ellipse	**Attribute**
	Ellipse with Under Line	**Key Attribute**
	Double Ellipse	**Multi-valued Attribute**
	Dashed Ellipse	**Derived attribute**
Relationships		
	Diamond	**The relationship**
	Line	**Links attributes to entity sets and entity sets to relationship sets.**
	Double Line	**Represents total participation of an entity in relationship set**

Representations

1. Entity

An **Entity** is any object, place, person or class. In E-R Diagram, an **entity** is represented using rectangles. Consider an example of an Organization. Employee, Manager, Department, Product etc. are considered as entities.

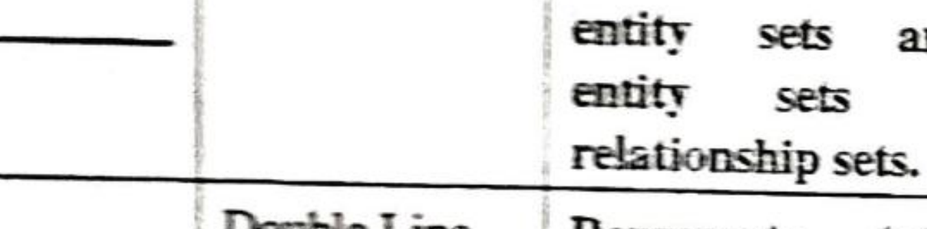


Fig. 1.11.1

Here employee and department are entities.

2. Weak Entity

Weak entity is an entity which depends upon another entity. Weak entity is represented by double rectangle. Subject is the weak entity. Because subject is depends on course.

Fig.1.11.2

3. Attribute

Attributes are nothing but the properties of entity. Here Stud_id, Name and address are attributes of entity Student.

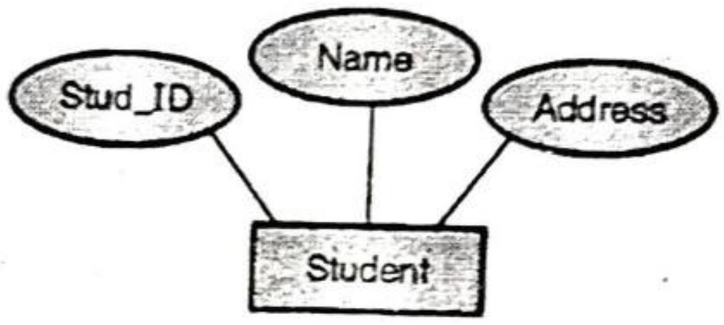


Fig. 1.11.3

4. Key Attribute (Primary key)

To identify attribute uniquely we set the key to the attribute. It is denoted by underline.

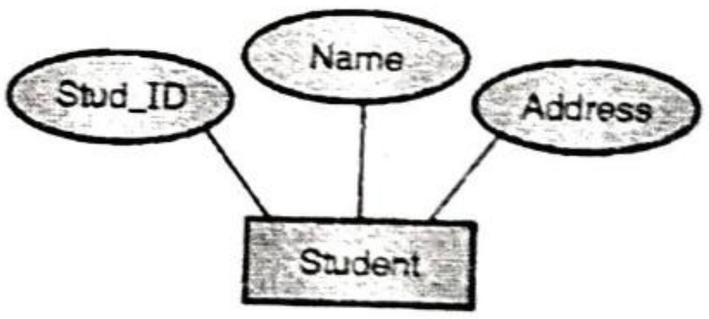


Fig. 1.11.4

5. Multi valued Attribute

The attribute which have multiple values is known as multi valued attribute.

Here Phone No is multi valued attribute as a person can have more than one phone numbers.

Fig. 1.11.5

6. Derived Attribute

Derived attributes are the attributes that do not exist physically in the database, but their values can be derived from other attributes present in the database.

For example : Age can be derived from data_of_birth.

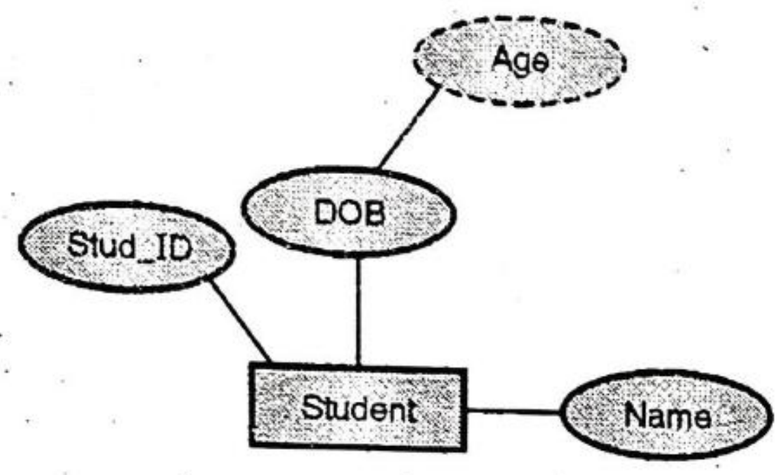


Fig. 1.11.6

7. Relationship

A relationship describes how entities interact with each other. For example, the entity "carpenter" may be related to the entity "table" by the relationship "builds". Relationships are represented by diamond shapes and are labeled using verbs.

Fig. 1.11.7

8. Recursive Relationship

If the same entity participates more than once in a relationship it is known as a recursive relationship. Consider an example where an employee can be a supervisor and be supervised by manager, so there is a recursive relationship.

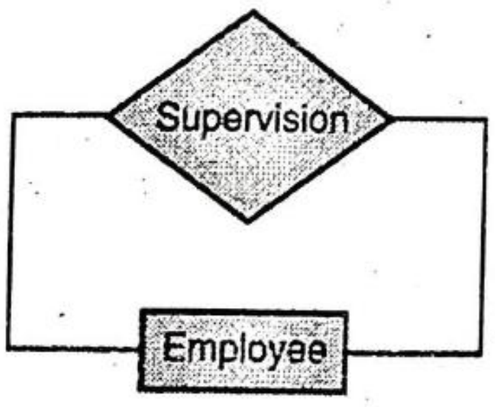


Fig. 1.11.8

Following E-R diagram represent the relationship between two entity sets teacher and student related through binary relationship guide.

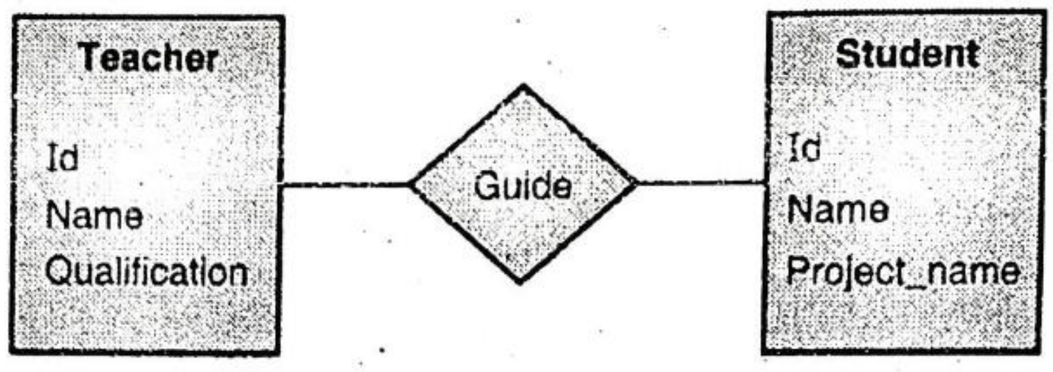


Fig. 1.11.9

The attributes of entity set teacher are

- Id
- Name
- Qualification

The attributes of entity set student are

- Id
- Name
- Project_name

1.11.1 Mapping Cardinality in E-R Diagram

In the two entities Teacher and Student, the relationship *Guide* may be

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

1. One to One

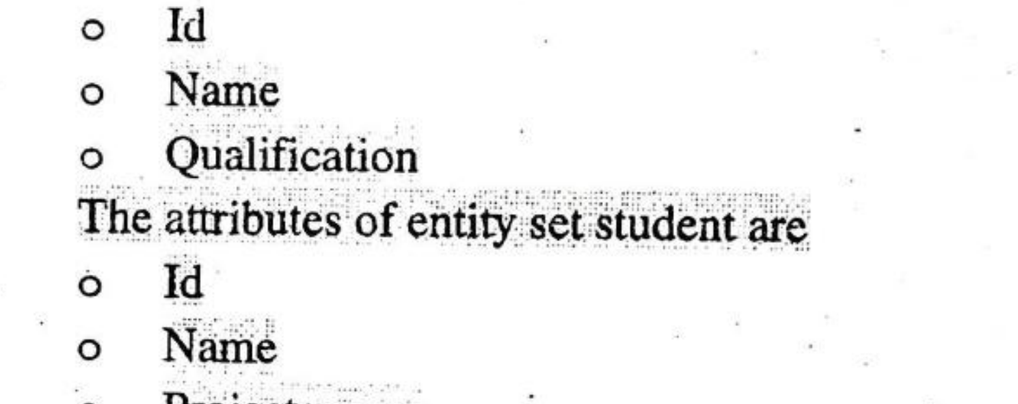


Fig. 1.11.10

In one to one mapping cardinality directed lines from relationship *Guide* are drawn towards both entity sets Teacher and Student. In this, the teacher can guide at most one student and the student can take guidance from at most one teacher.

2. One to Many

In one to many mapping cardinality directed line from relationship *Guide* to entity set Teacher is drawn and undirected line from relationship *Guide* to entity set Student is drawn. In this, the teacher can guide many students but a student can take guidance from at most one teacher.

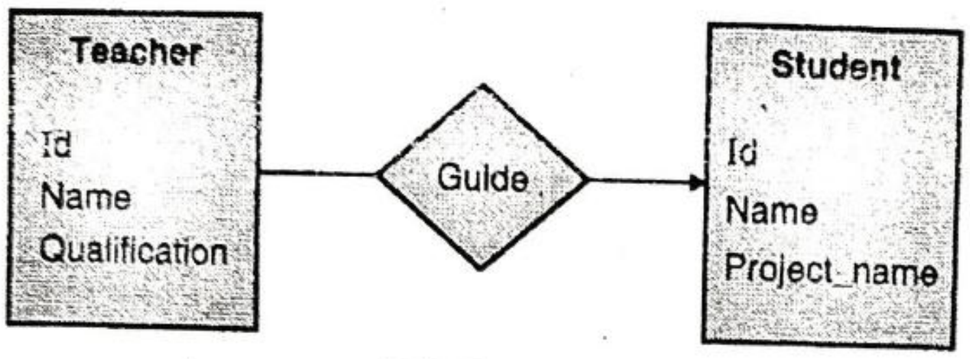


Fig. 1.11.11

3. Many to One

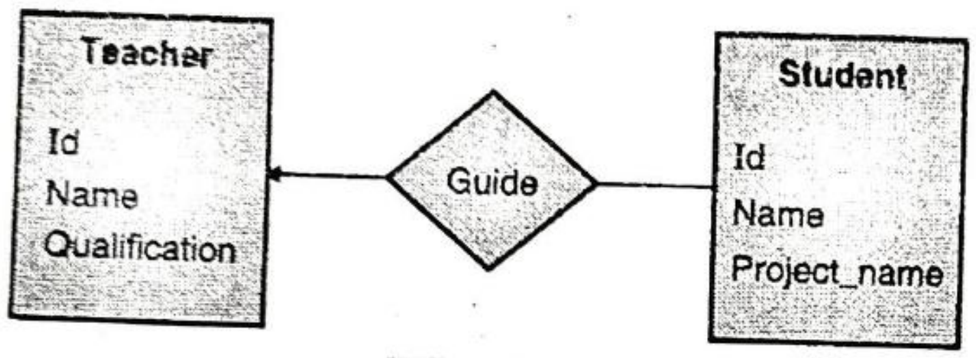


Fig. 1.11.12

In many to one mapping cardinality undirected line from relationship *Guide* to entity set Teacher is drawn and directed line from relationship *Guide* to entity set Student is drawn. In this, the teacher can guide at most one student but a student can take guidance from many teachers.

4. Many to Many

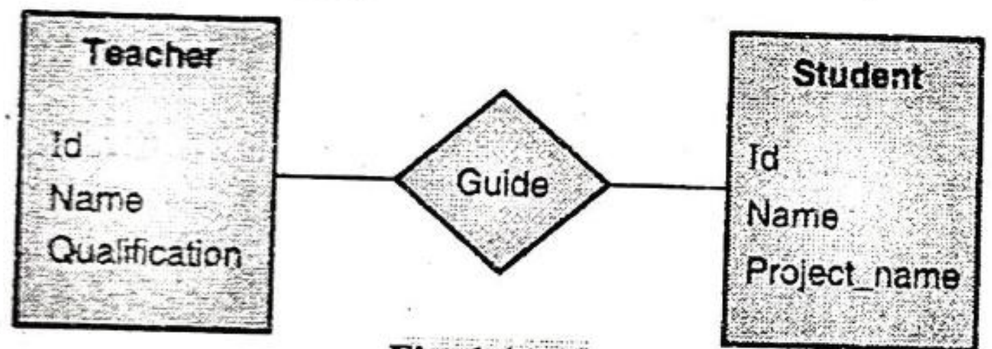


Fig. 1.11.13

In one to one mapping cardinality directed lines from relationship *Guide* are drawn towards both entity sets Teacher and Student. In this, the teacher can guide many students and the student can also take guidance from many teachers.

Points to remember while drawing an ER diagram

1. Initially identify all entities and their relationships with each other in the given database system.
2. No entity should be repeated in a particular diagram.
3. Provide a precise and appropriate name for each entity, attribute, and relationship in the diagram. Try to give user friendly words while naming. The name should also be meaningful, unique and easily understandable
4. Do not set unclear, redundant or unnecessary relationships between entities.
5. Never connect a relationship to another relationship.
6. Using colors helps to make the diagram easily understandable. It helps in differentiation and classification

1.11.2 Examples of ER Diagram

E-R diagram with multi valued and derived attributes

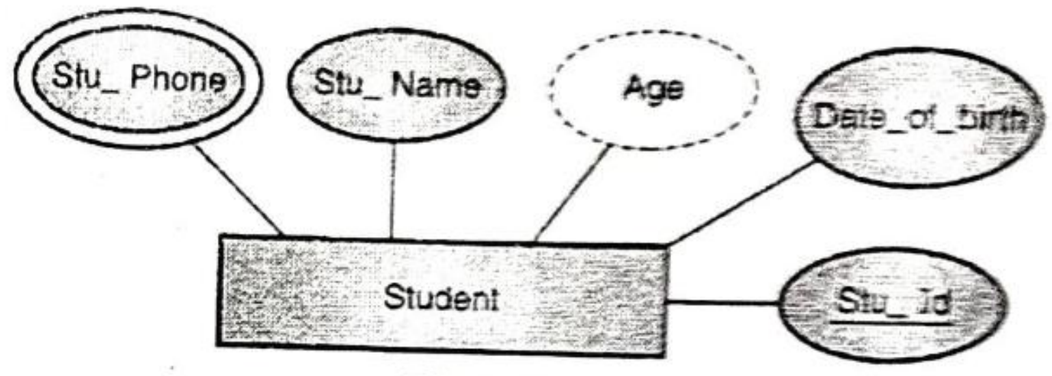


Fig. 1.11.14

Total Participation of an Entity set

If in a relationship set, every entity in entity set has one relationship then it called as total participation of entity set.

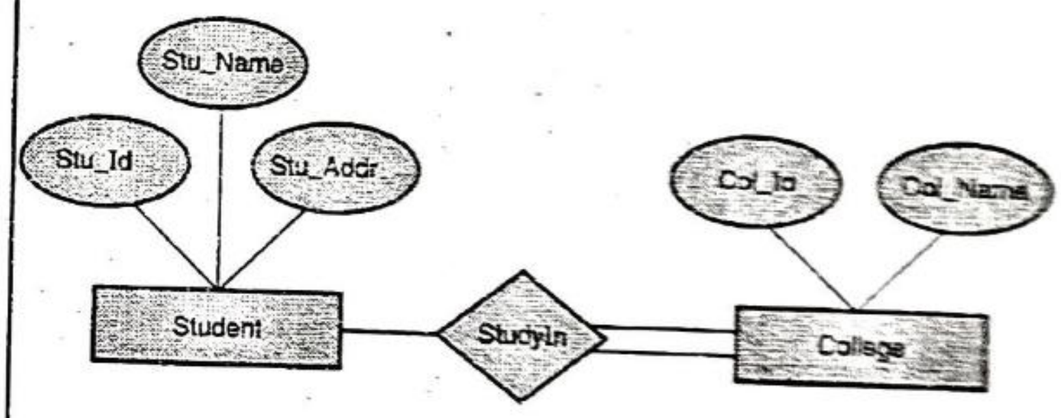


Fig. 1.11.15

In the Fig. 1.11.15, each college must have at least one associated student. A Total participation of an entity set represents that all the entities in the entity set should have minimum one relationship in a relationship set. For example: In the above diagram each college must have at-least one associated Student.

ER diagram for Database of employee

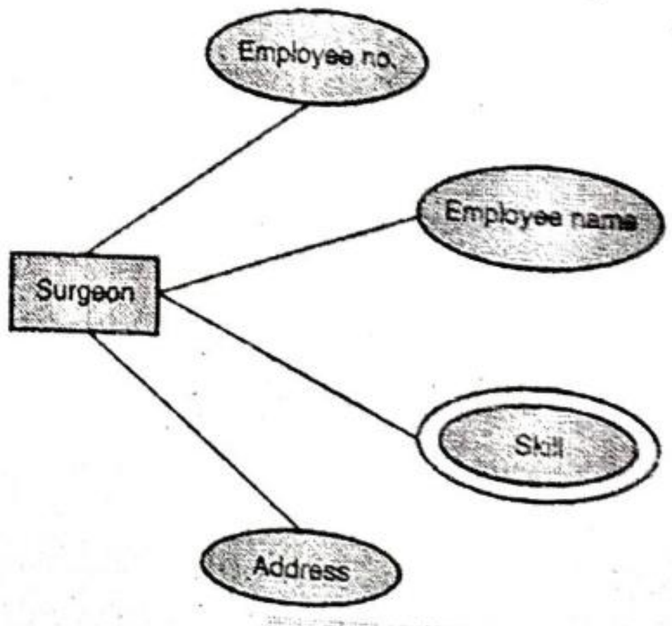


Fig. 1.11.16

Ex. 1.11.1

Following requirements are given for a database of the National Hockey League The NHL has many teams,

Each team has a name, a city, a coach, a captain, and a set of players, Each player belongs to only one team,

Each player has a name, a position (such as left wing or goalie), a skill level, and a set of injury records, A team captain is also a player, A game is played between two teams (referred to as host_team and guest_team) and has a date (such as May 11th, 1999) and a score (such as 4 to 2). Construct an ER diagram for the NHL database.

Soln. :

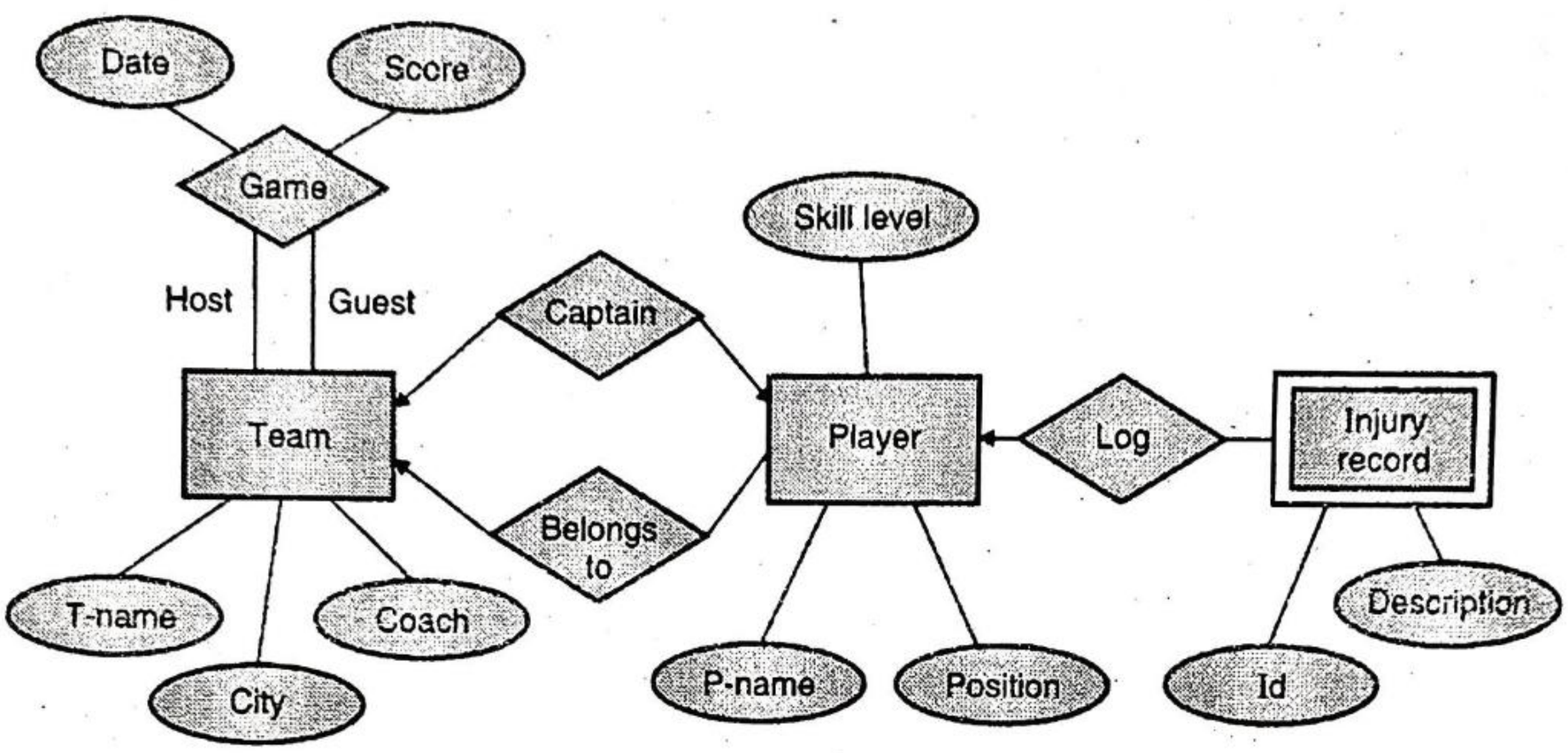


Fig. P. 1.11.1

Ex. 1.11.2 SPPU - Oct. 2016(In sem), 5 Marks

Draw an ER-Diagram for online Book Shop which should consist of entity set, attribute, relationship, mapping cardinality and keys, it will maintain information about all Customers, books, book author, publisher, billing etc.

Soln. :

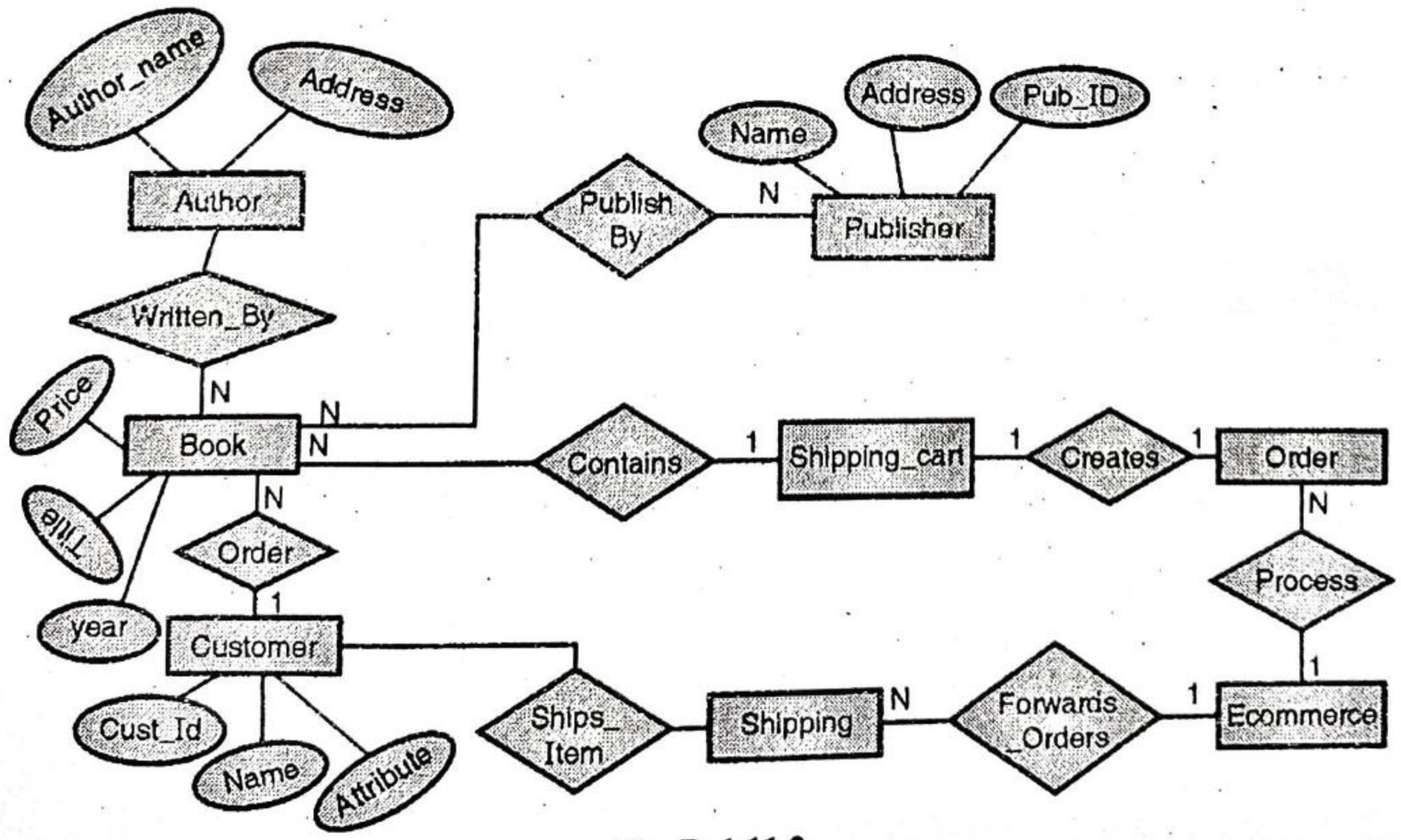


Fig. P. 1.11.2

Ex. 1.11.3

Draw an ER-Diagram for hospital management where patient take treatments from physician and also he/she can claim a medical insurance.

Soln. :

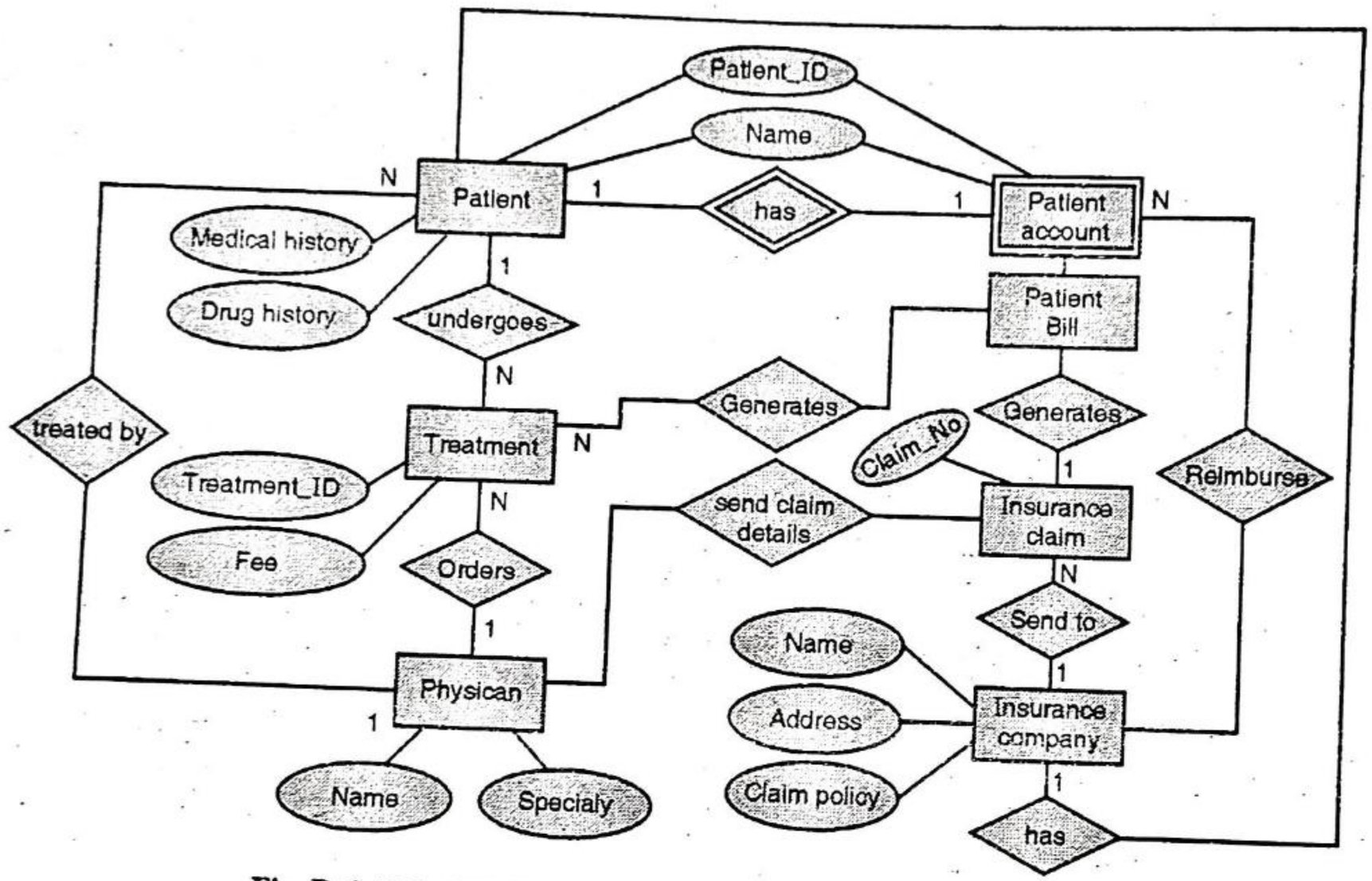


Fig. P. 1.11.3 : ER diagram for Hospital management system

Ex. 1.11.4 :

Draw an ER-Diagram for Online Sales system in which customer can order terms online and pay through credit card.

Soln. :

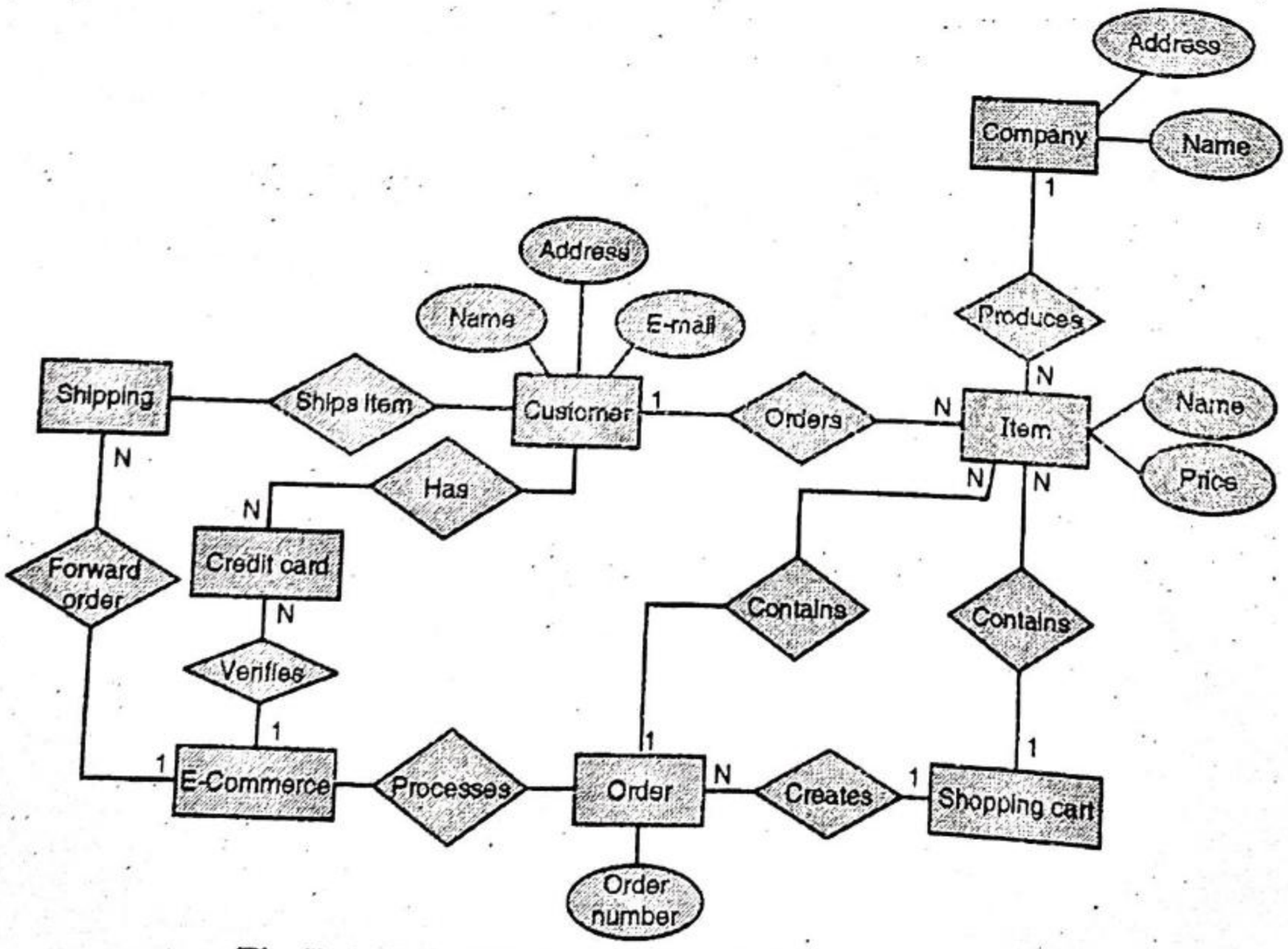


Fig. P. 1.11.5 : ER diagram for Online sales system

Ex. 1.11.5 **SPPU - May 2016, 5 Marks**

Construct an E-R diagram for a car insurance company that has a set of customers each of whom owns one or more cars. Each car has associated with it zero to any number of recorded accidents.

Soln. :

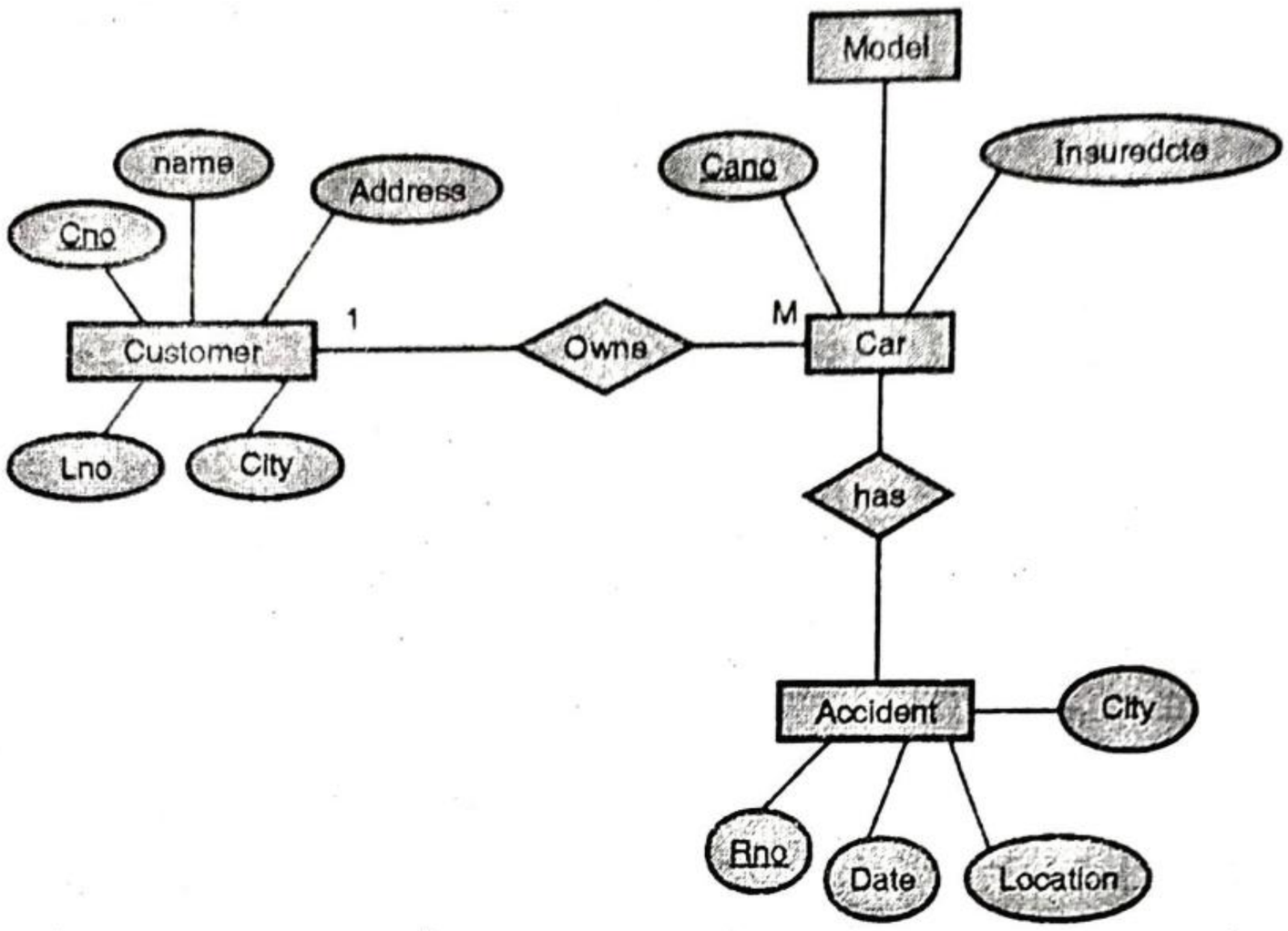


Fig. P.1.11.5

Ex. 1.11.6 :

Following information is maintained for online bookstore

(i) Books(ISBN,price,title,year)

(ii) Author(name,address,url)

(iii) Publisher((name,address,phone,url)

(iv) Customer(name,address,email,phone) (name is discriminating attribute)

(v) Shoppingbasket(basketID)

Construct ER diagram with following constraints :

Each book should have author and publisher. Book may have more than one author. Each customer have a dedicated shopping basket. Books can further be categorized as books, music, cassette or compact disks.

Soln. :

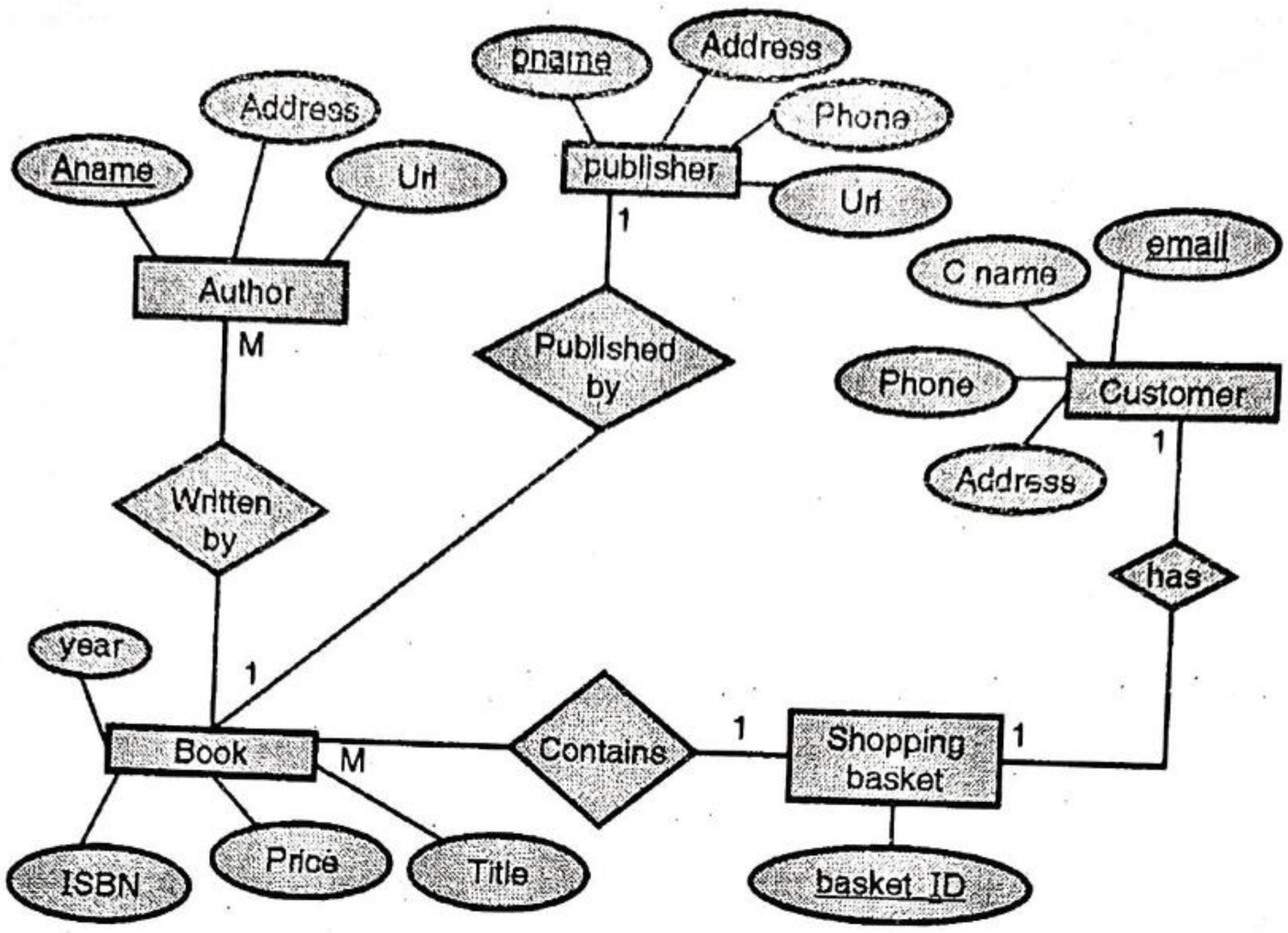


Fig. P.1.11.6

Syllabus Topic : Design Issues

1.12 Design Issues

It is possible to define a set of entities and the relationships among them in a number of different ways. But sometimes it is difficult to decide on the best way to model the application. That means it is hard to find the best option to design. Sometimes it is hard to decide whether something should be represented as an Entity, an attribute or a relationship. These basic issues in the design of an E-R database schema are described as below :

Use of entity sets vs. attributes

- While designing ER-Diagram the question arises as whether a value is represented as a separate entity-set or an attribute?
- The Choice mainly depends on the structure of the enterprise being modeled, and on the semantics associated with the attribute in question.
- Representing a value as a separate entity-set provides a scope to add details later.

For Example

- As shown in Fig. 1.12.1 employee entity has 5 attributes(emp_id, name, phone, city, street).
- An employee has single name but he/she can have multiple phone numbers so it is good to represent phone as a separate entity with two attributes phone_number and location; rather than an attribute. The location stands for office or home. If the phone numbers with different locations are important then we cannot add the phone as attribute, rather we will add it as separate entity having its own attributes.

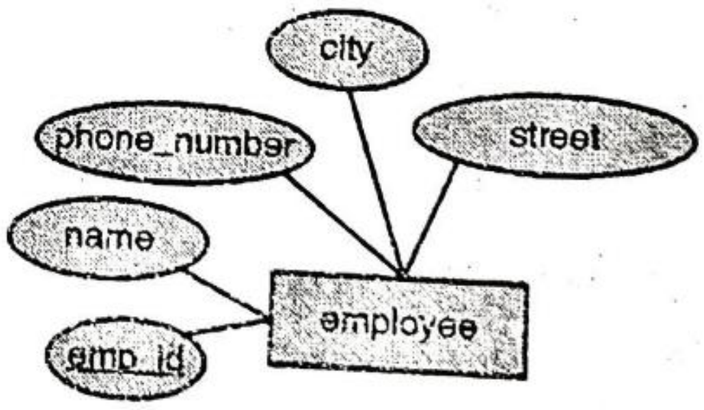


Fig. 1.12.1

- The relationship *emp_phone* can be created between entities employee and phone as follows :

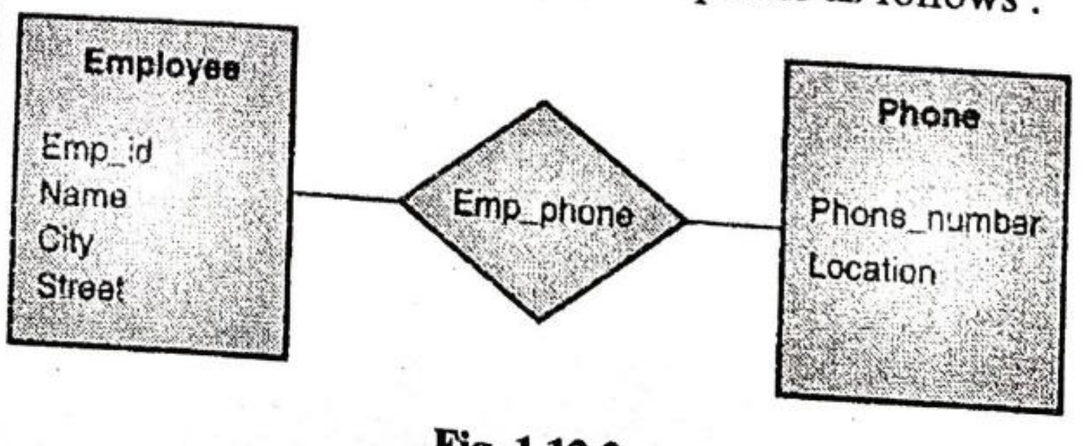


Fig. 1.12.2

- But if the phone number is not so important or detailed concept in the system, then it can be taken as multi-valued attribute.
- Number of times it becomes difficult to set the importance of any element, which makes it complicated to decide whether to make it an attribute or entity.

Use of entity sets vs. relationship sets

- In the situation where ER Diagram is designed to represent loan given to exactly one customer and each loan is given at a particular bank branch.
- We assumed that a loan is modeled as an entity. An alternative is to model a loan as a relationship between customers and branches, with loan-id and amount as descriptive attributes as follows :

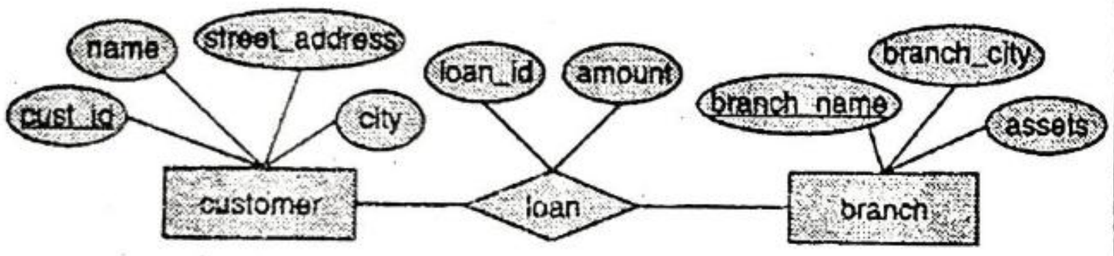


Fig. 1.12.3 : ER diagram for Loan

- Each loan is represented by a relationship between a customer and a branch.
- But in another situation in which several customers hold a joint loan, we must define a separate relationship for each joint-loan-holder.
- So each relationship has the same value for the loan-number attribute and amount attribute. That is more than one customer can hold a loan so the value of loan- id and amount attribute are same for those customers; if for every customer separate relationship is defined then this attribute values are replicated in each relationship.
- This replication causes two problems as :
  1. Waste of storage space
  2. Data in an inconsistent state, if update operation is not well-performed.
- To solve this issue one possible guideline for determining whether to use an entity set or a relation- ship set in ER diagram design, is to choose an entity set to describe an thing and choose a relationship set to describe an action that occurs between entities.
- This approach can also be useful in deciding whether certain attributes may be more appropriately expressed as relationships.

Binary versus n-ary relationship sets

Mostly databases use binary relationships. Some non-binary relationships could actually be better represented by several binary relationships. For example,

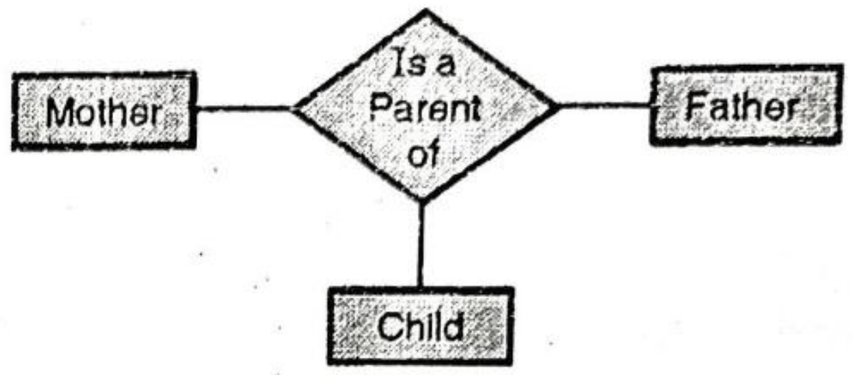


Fig. 1.12.4

Fig. 1.12.4 shows ternary relationship Parent, which relates a Child with his/her Mother and Father. However, it could also be represented by two binary relationships, Mother and Father, relate a Child to his/her Mother and Father separately as follows :

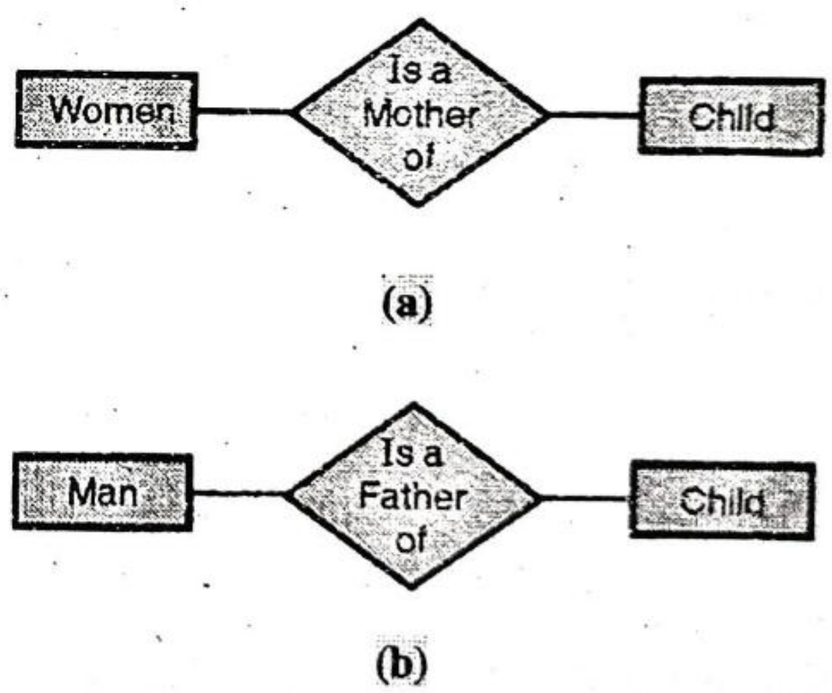


Fig. 1.12.5

Advantages of using two relationships Mother and Father allows us to record a child's mother, even if child's father's identity is not known; a null value would be required if the ternary relationship parent is used. This null representation is avoided in binary relationship.

For simplicity, consider the abstract ternary relationship set R, which relates entity sets A, B, and C.

We replace the relationship set R by an entity set E, and create three relationship sets :

- Relationship set RA relates entities E and A.

- Relationship set RB relates entities E and B
- Relationship set RC relates entities E and C

Restricting the E-R model to include only binary relationship sets is not always desirable. It is not possible to translate constraints on the ternary relationship into constraints on the binary relationships.

For example, consider a constraint that says that R is many-to-one relationship from A, B to C; that is, each pair of entities from A and B is associated with at most one in C entity. This constraint cannot be expressed by using cardinality constraints on the relationship sets RA, RB, and RC

Placement of relationship attributes

- Relationship set may have attributes which describe each relation with entities.
- The problem arises where to place this descriptive attribute; whether it is places with one of the entity or it will be placed with the separate relationship-set.

For example

- Consider *loan* relation is related with customer and account entity. If loan relationship has attributes as loan number and amount. We can associate these attributes in the account table or in the loan table.
- The choice of attribute placement is more clear-cut for many-to-many relationship sets. For example: a customer may have one or more number of accounts, and there may be one or more customers for single account.
- To express the date of a specific customer's last account access, then the access date must be an attribute of the depositor relationship set, it should not be one of the participating entities.
- If access-date were an attribute of account, in that case , we could not determine which customer made the most recent access to a joint account.
- When an attribute is determined by the combination of participating entity sets, instead of either entity independently, that attribute must be associated with the many-to-many relationship set.
- In such cases the decision of design: where to place descriptive attributes - as a relationship or entity attribute - should reflect the properties of the business being modeled.
- The cardinality ratio of a relationship can affect the placement of relationship at- tributes. Thus, attributes of one-to-one or one-to-many relationship sets can be associated with one of the participating entity sets, rather than with the relationship set.

Syllabus Topic : Extended ER Features

1.13 Extended ER Features

SPPU - Dec. 13, May 14, Dec. 16

University Questions

Q. Explain the different constraints on specialization/generalization with suitable example. **(Dec. 2013, 4 Marks)**

Q. Explain extended ER features Specialization, Generalization and Aggregation with Example and diagrams. **(May 2014, 8 Marks)**

Q. Explain the concept of specialization & generalization in E-R Model using suitable example. **(Dec. 2016, 5 Marks)**

We can use basic E-R concepts to develop most database features, but to express database deeply some extended features available in ER Model which are known as **Extended ER-Features**. These are as follows :

1.13.1 Specialization

- In an entity set, sometimes further sub grouping is also possible depending upon certain characteristics. For example, a subset may have some attributes which are not shared by other subset in the entity set. In E-R diagram these distinct subsets can be represented by some means.
- Consider an example of entity set person having attributes name, city and street. The entity person may be further classify as follows:
  - o customer
  - o employee
- Both of these types of person are described by a set of attributes that contains all the attributes of entity set person with some additional attributes of their own.
- For example, in the description of customer entities, we may add new attribute like customer-

id, whereas in the description of employee entities we may add new attributes like employee-id and salary.

- This process of creating subgroups within an entity set is called **specialization**. The specialization of entity person helps us to distinguish whether a person is an employee or customer depending upon attributes.
- An entity set may be specialized by more than one distinguishing characters. In the above example, job performed by an employee may be a distinguishing character from customer. The employee may be further divided as permanent or temporary employee depending upon some characteristics.

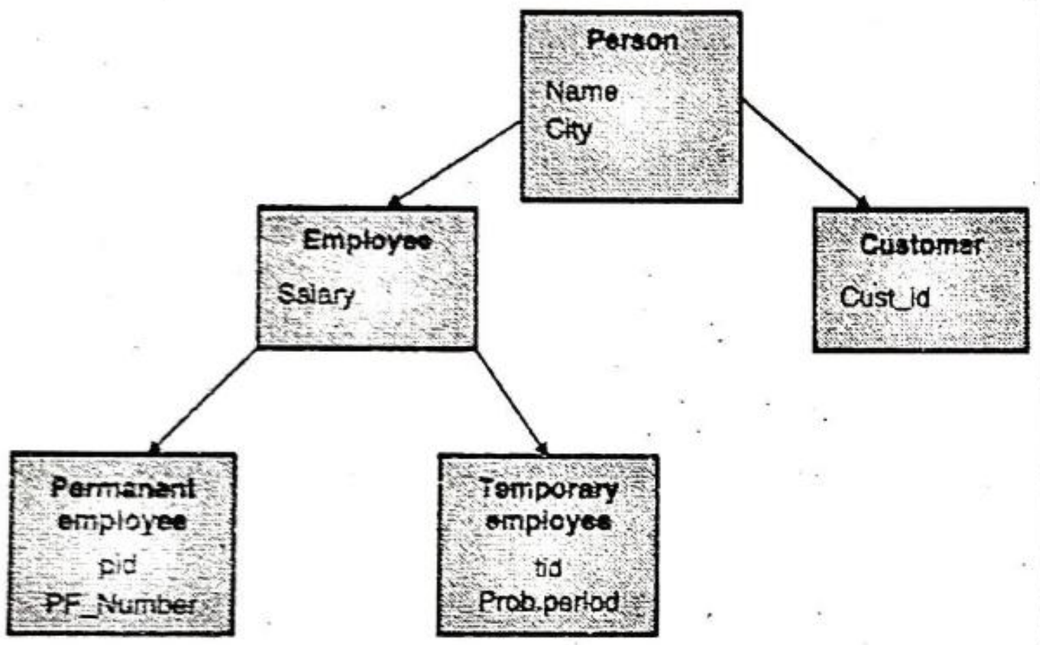


Fig. 1.13.1 : Specialization

1.13.2 Generalization

- In the Fig. 1.13.1, we can observe that the refinement from an initial entity set into subgroups(multiple entity sets) depending upon distinct features shows top-down approch.
- The design process may also proceed into opposite bottom-up approach , in which multiple entity sets are grouped into higher level single entity set depending upon the common characteristics in between these entity sets.
- In out example the entity 'Permananent emplyoee' has following attributes
  - Name
  - City
  - pid
  - Salary
  - Pf_Number
- The entity 'Temporary emplyoee' has following attributes
  - Name
  - City
  - eid
  - Salary
  - Prob_period
- Now if we observe, there are some common attributes between entities 'Permananent emplyoee' and 'Temporary emplyoee'.
- This commonality can be termed as **Generalization** which is containment relationship that exists between higher level entity set and one or more lower level entity set.

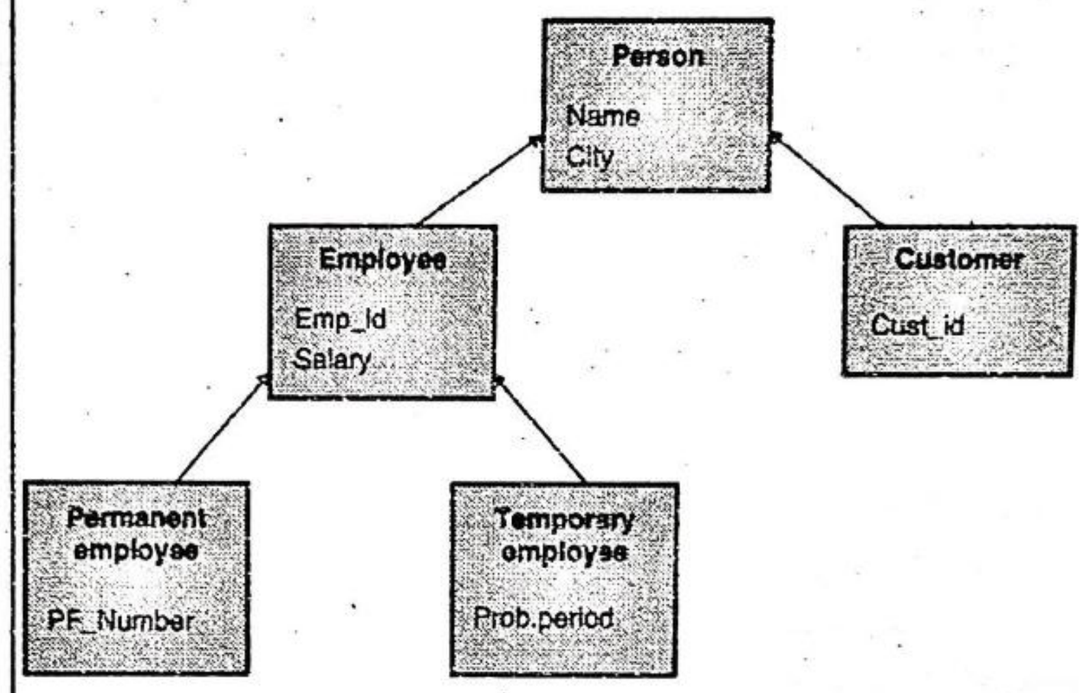


Fig. 1.13.2 : Generalization

- Here the employee is higher level entity set while the entities 'Permananent emplyoee' and 'Temporary emplyoee' are the lower level entity set.
- In this example, the attributes which are conceptually same has different names. e.g. the pid and eid are both nothing but the employee ids of permanent and temporary employees. To create generalization such attributes can be given a common name like emp_id and must be represented in higher level entity *employee*.
- The higher level entity set is also known as Superclass while the lower level entity set also known as subclass.

Attribute Inheritance

- The higher- and lower-level entities created by specialization and generalization has the important property - attribute inheritance
- The attributes of the higher-level entity sets are generally considered to be inherited from the lower-level entity sets.
- In the above example customer and employee both inherit the attributes of person entity set. The description of customer contains name, street, city and additional customer-id attribute. The descriptoin of employee contains name, street, and city and additional employee-id and salary attributes.
- Participation in the relationship sets of higher-level entity is inherted by the lower level entity set.
- Attribute inheritance applies through all tiers of lower-level entity sets. The *employee* and *customer* entity sets can participate in any relationships in which the *person* entity set participates.
- The outcome is basically the same even though the given portion of an E-R model was arrived at by specialization or generalization.
  - A higher-level entity set with attributes and relationships that apply to all of its lower-level entity sets.
  - Lower-level entity sets with unique characters that apply only within a particular lower-level entity set.

Constraints on Generalizations

While designing a database system, the database designer may place certain constraints on a particular generalization.

There are number of such constraints.

(A) One is to detemine which entities can be members of a given lower-level entity set. Such membership may be one of the following :

- **Condition-defined :** In this entity set, it is observed that whether the entity satisfies the specific condition or not.
- Consider an example, assume that the higher-level entity set account contains attribute account-type. Now all account entities are evaluated depending upon the account-type attribute. Only those entities which has account-type as "savings account" are allowed to belong to the lower-level entity set person.
- All entities that has account-type as "current "are included in current account. Here all the lower-level entities are evaluated on the basis of the common attribute as account-type, this type of generalization is called as attribute-defined.
- **User-defined :** This type of lower level entity sets are not constrained by a membership condition. The end user of the database assigns entities to a given entity set. Consider groups are created of students for creating projects. In such case the the teams may be decided by the class teacher.

(B) A second type of constraint check whether entities belong to one or more lower-level entity set within a single generalization.

The lower-level entity sets may be one of the following :

- **Disjoint :** In this generalization , the entity must belong to single lower-level entity set. In the example, the acount_type attribute of account entity satisfies only one condition that it may be saving or current, but not both.
- For a disjointness constraint, it is necessary that an entity should not belong to more than one lower-level entity set.
- **Overlapping :** In this generalizations, One entity may belong to more than one lower-level entity.

Consider an example of IT firm database, where a manager may involve and guide to more than one teams. Hence it appears in more than one lower level team entity sets. Here the generalization is overlapping.

(C) **Completeness :** It specifies whether or not an entity in the higher-level entity set belongs to at least one of the lower-level entity sets within the generalization/ specialization or not.

This constraint may be one of the following :

- **Total generalization or specialization :** Here each higher-level entity must belong to a lower-level entity set.
- **Partial generalization or specialization :** Here higher-level entities may not belong to any lower-level entity set.

1.13.3 Aggregation

In E-R model there is limitation that is it cannot express relationships among relationships. Consider the ternary relationship 'works-on' between a employee, branch, and job. Consider we want to record managers for (employee, branch, job) combinations. An entity set 'manager' is available.

To reprsent this relationship, we can set a quaternary relationships manages between employee, branch, job, and manager. Using the basic E-R modeling constructs, we obtain the following E-R diagram.

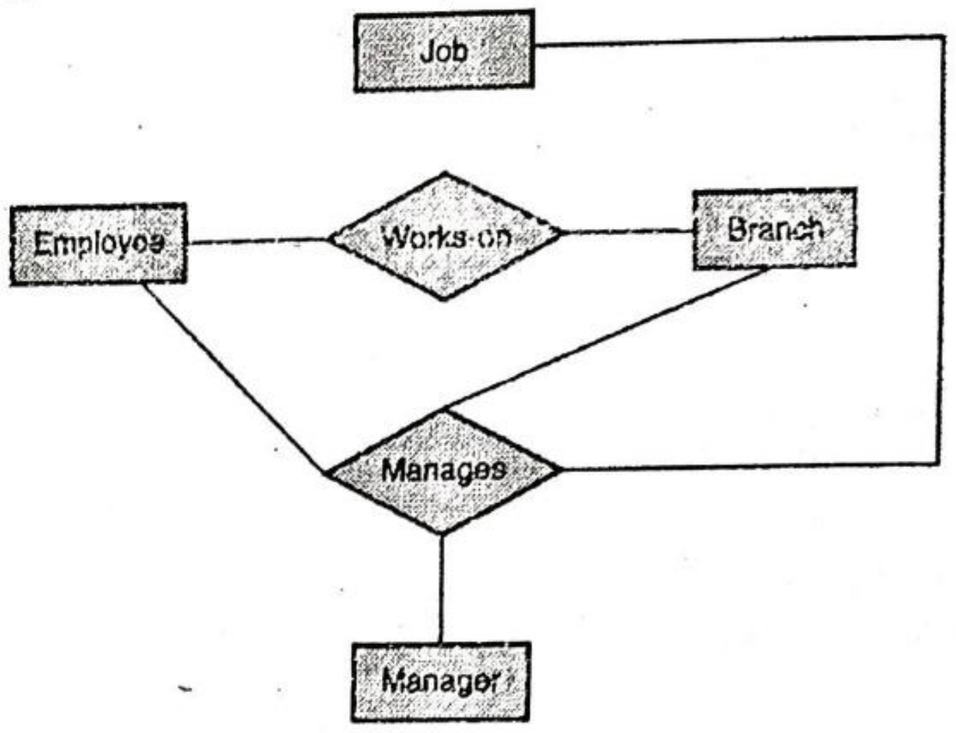


Fig. 1.13.3 : Aggregation

Here the relationship sets 'works-on' and 'manages 'can be combined into one single relationship set. But combining the mis difficult as some employee, branch, job combinations many not have a manager.

Ex. 1.13.1 SPPU - Dec. 2015, 5 Marks

Construct an ER Diagram for a banking Database System. Consider various entities such as Account, Customer, Branch, Loan, Deposit, Borrower, etc. Design Specialization and Generalization EER Feature.

Soln. :

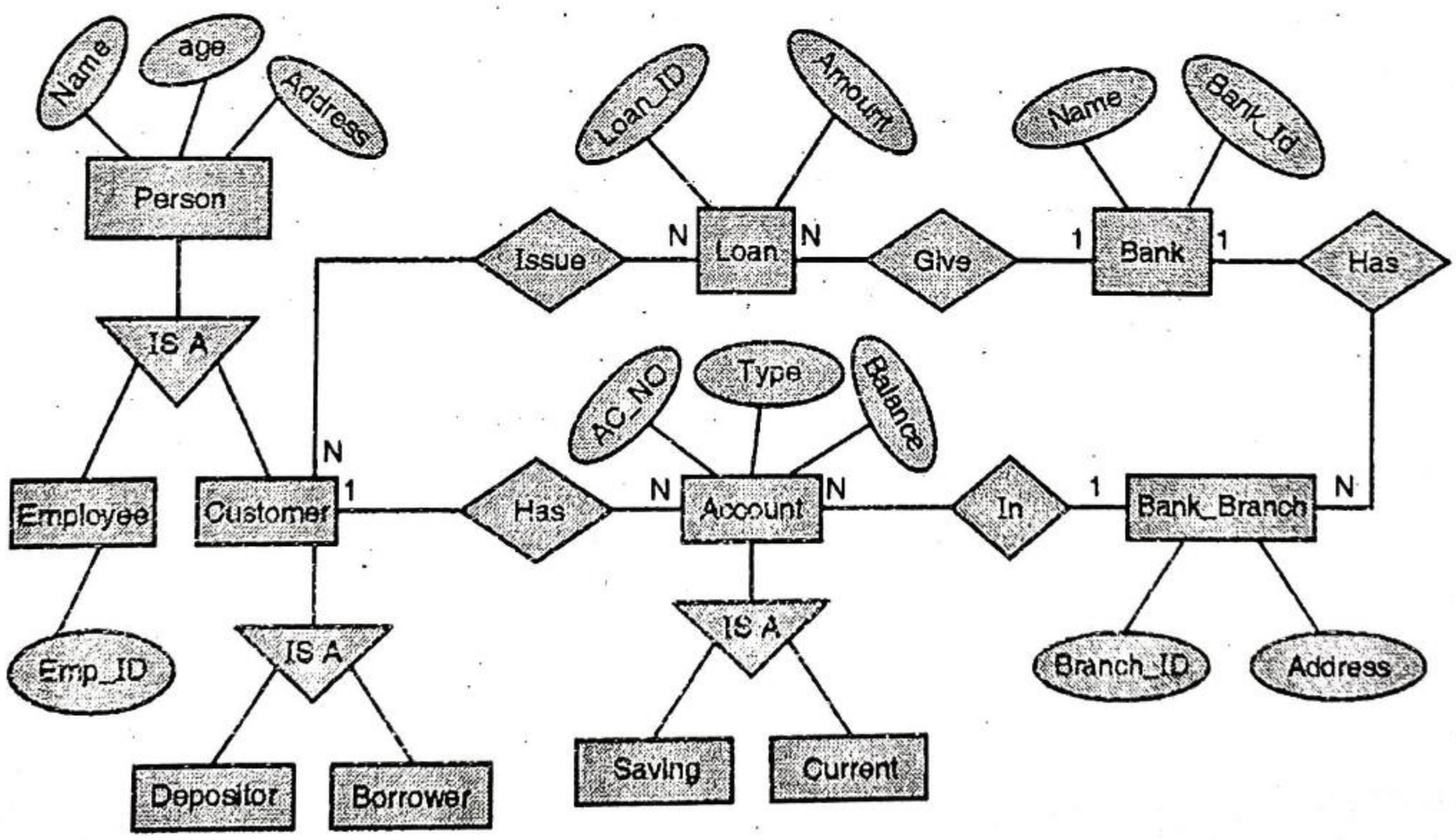


Fig. P. 1.13.1

Ex. 1.13.2 :

Construct an ER Diagram for a Travel Agency. Consider various entities such as travel agency, passenger, Branch, seat, bus, employee, tours etc. Design Specialization and Generalization EER Feature.

Soln. :

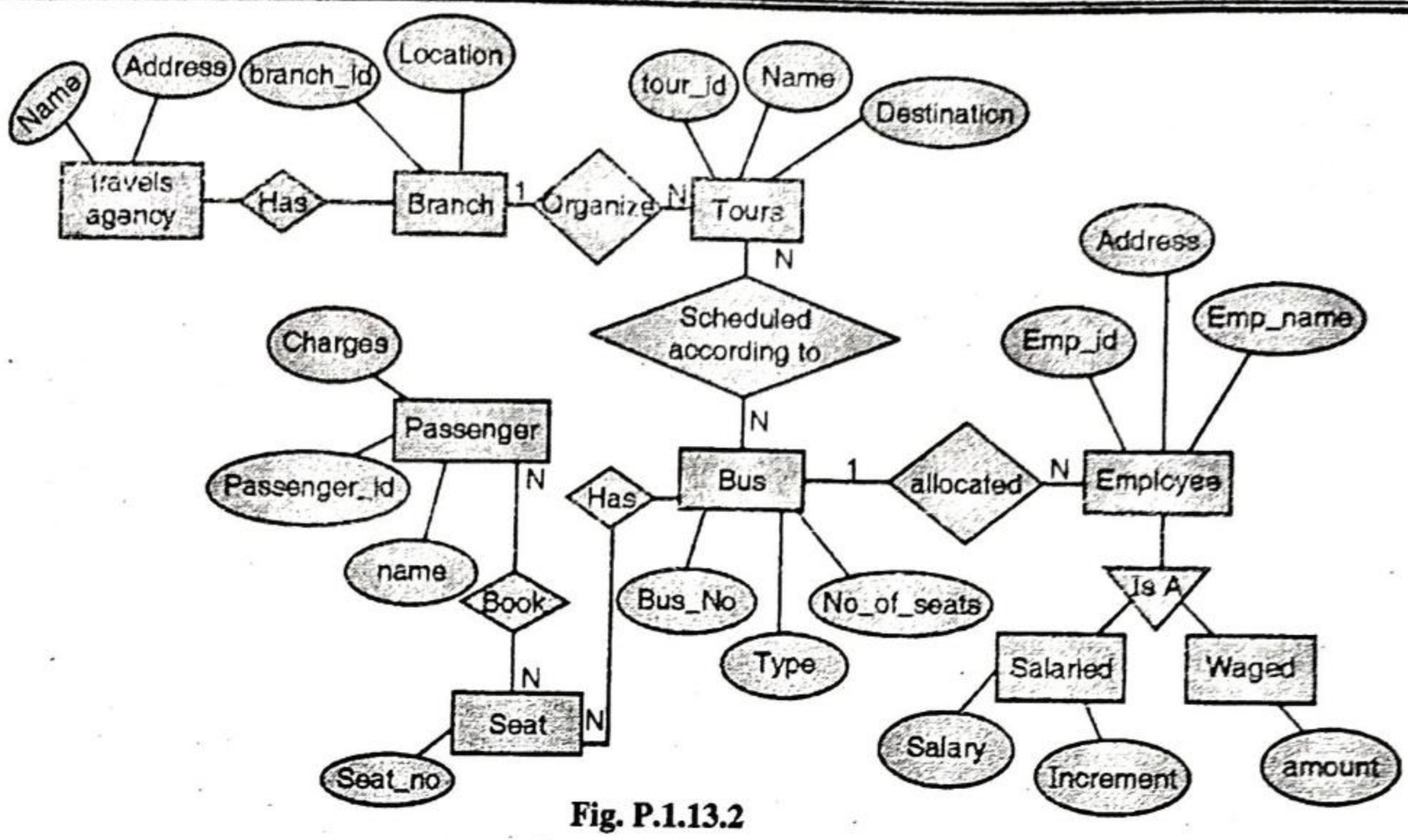


Fig. P.1.13.2

Ex. 1.13.3

For a Library Management System following information is maintained.

Books (Accession_no, Title, Author, Price, Booktype, publisher)

Borrower(Membership_no, Name, Address, Category, max_no_of_books_issued, Accession_no)

Draw E-R Diagram for the above taking into consideration following

Constraints and by making use of at least one Extended ER feature :

i) A book may have more than one author.

ii) There may be more than one copy of a book.

iii) Borrower can be staff or a student. Depending on this category max number of books that can be issued will vary. [i.e. student can ask for max 3 books whereas staff can ask for 10 books]

Soln. :

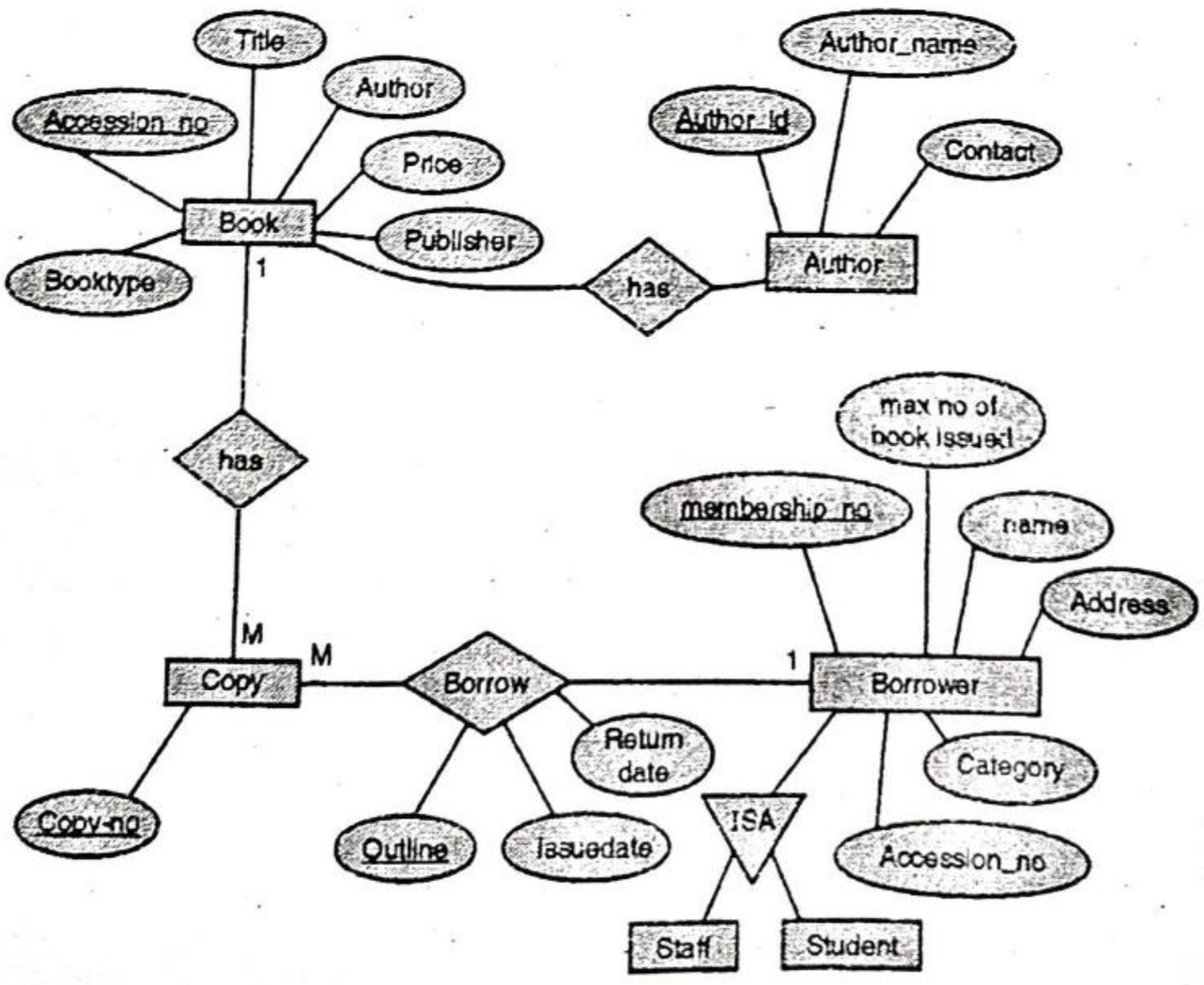


Fig. P.1.13.3 : ER diagram for library management system

Syllabus Topic : Converting ER & EER Diagram into Tables

1.14 Converting ER & EER Diagram Into Tables

SPPU - May 15

University Question

Q. Explain with example how E-R diagrams are converted into tables. **(May 2013, 6 Marks)**

As we know ER Diagram gives us good knowledge of entities and their relations. We can understand various mapping cardinalities from ER-Diagram. Using ER Diagram we can easily create Relational Data Model. It is nothing but the logical view of the database. We can convert these ER Diagrams into the tables. There are various steps involved in conversion of ER diagrams into the table.

An ER diagram consists of :

- Entity sets which are represented by rectangular box.
- Relationship sets which are represented by diamond.

We will convert each entity/relationship set into a tables by using following steps:

Consider example see following ER-Diagram which we are going to convert into table using following steps :

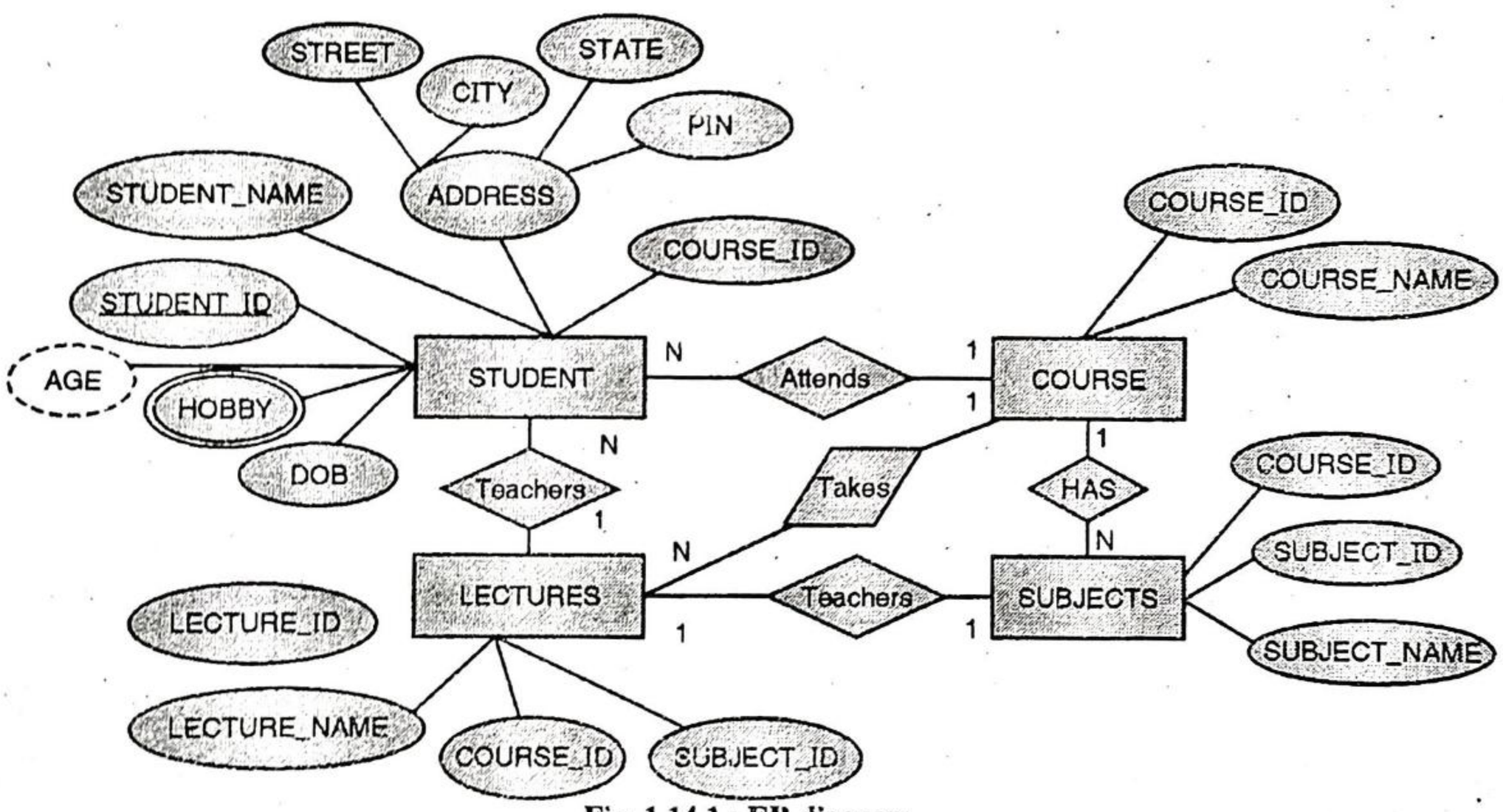


Fig. 1.14.1 : ER diagram

Entity Set ⇒ Table

While converting An ER-Diagram into tables (relational database) first task is to design individual table for each entity in ER diagram. Attributes of entity are treated as fields / columns of tables.

- **Single-valued attribute :** These attributes, whose value is unique, are considered as columns of that table. In the STUDENT Entity, STUDENT_ID, STUDENT_NAME form the columns of STUDENT table. Similarly, LECTURER_ID, LECTURER_NAME form the columns of LECTURER table.
- **Composite Attribute :** Composite attributes are represented in table as individual columns. In above ER diagram, in STUDENT entity ADDRESS is a composite attribute; it is formed by combining STREET, CITY, STATE, and PIN attributes so while converting it into table the table contain columns for STREET, CITY, STATE, and PIN.

- **Multi-valued Attribute :** If any entity has multi-valued attribute then to convert it in table needs to form an individual table for that attribute. In above diagram, in the Student table , hobby attribute is multi-valued. A student can have more than one hobby. So it is difficult to represent multiple values in a single column of STUDENT table. It must be stored separately, so that it is possible to store multiple hobbies. Redundancy in the system should not be created by adding/ removing / deleting hobbies. A separate table STUD_HOBBY with STUDENT_ID and HOBBY as its columns can be created. We can create a composite key using both the columns.
- **Primary key :** In above diagram, STUDENT_ID, LECTURER_ID, COURSE_ID and SUBJECT_ID are the key attributes of the entities. Hence they are the primary keys of respective table.
- **Weak entity :** In above ER diagram no weak entity is present.
- As shown in Fig. 1.14.1 ER-Diagram, it contains 4 entities (STUDENT, COURSE, LECTURER and SUBJECT) which are become independent tables as follows :

Table 1.14.1 : Student

Student_Id	Student_Name	Street	City	State	Pin	Course_Id	Dob

Table 1.14.2 : Stud_Hobby

Student_Id	Hobby

Table 1.14.3 : Lecturer

Lecturer_Id	Lecturer_Name	Course_Id	Subject_Id

Table 1.14.4 : Course

Course_Id	Course_Name

Table 1.14.5 : Subject

Subject_Id	Subject_Name	Course_Id

❑❑❑