Database Management System

©Harendra Nath tiwari

4.1. INTRODUCTION

Traditionally data were kept in files (be it paper based or computerized) in the organizations. Different data were available with different departments, for example stores had the data relating to availability of stocks, the accounts department had data for cost structure, production department had data relating to production capabilities, finance department had data relating to availability of funds, etc.

The constantly changing business environment requires instant availability of the information to every department. Now suppose, the sales team receives an order to supply 10,000 units of a product. Before accepting the offer, the sales team would need to know about availability of stock, production possibilities, funds availability, cost structure, etc. Since, these information are available with different departments it may take a few days to reply to the offer which may lead to loss of the order. If these data are centrally available, the team would take only a few minutes to reply to the offer.

In a normal course of the day, many of us encounter several activities which involve interaction with database. For example if we go to bank the database is centrally managed and therefore we can use any branch of the bank for our banking transactions. It requires interaction with database. If we wish to book a railway ticket or air ticket it requires interaction with database which is constantly updated after every ticket booked.

Database have evolved from traditional paper file system to computer file system to modern database systems which are managed by sophisticated software. This chapter attempts to explain various concepts and models used in database approach giving due consideration to the traditional file system.

DATABASE SYSTEM

A database is a set of logically related data that is organised so that it can easily be retrieved, managed and updated. The modern database are managed by sophisticated software known as database management system (DBMS). The database which uses DBMS for data collection, retrieval and updating is called database system. A database system offers the following to the users :

- ⇒ an easy way to collect, update and retrieve stored information;
- ⇒ data stability by preventing unnecessary loss of data;
- ⇒ data protection by preventing unauthorised use of data; and
- ⇒ ensures data quality in terms of accuracy, availability and usability.

A database system is a computer based record keeping system which is a collection of tables, files and datasets.

The people with specialist skills are required to design, construct and maintain database systems. These people are called database programmers and database administrators. They use specialized database languages like database definition languages (DDL), data manipulation languages (DML) and query languages to construct and maintain the database. The most widely supported database language is SQL (structured query language) which combines the roles of both DDL/DML and a query language. It is developed for relational data models. A database is mostly used in information system is which it is an essential component. The database is used to generate information, it supports decision making process by making correct information available to the management at the right time. The database system is program independent, and hence, the same database can be used by a number of programs for their usage requirements.

The following are a few examples in which database system is used :

- ⇒ Computerized Library System
- ⇒ Online banking and Automated Teller Machines (ATMs)
- ⇒ Train and Flight reservations
- ⇒ Hotel bookings
- ⇒ Weather forecasting
- ⇒ Patient registrations in hospitals, etc.

Importance of Database Management System (DBMS)

DBMS is very important for an organisation as :

1. It makes data management more efficient and effective.
2. It provides easy access to better managed data.
3. By using query languages, database allows quick answers to sudden ad-hoc queries.
4. It maintains accuracy and consistency of data among different units of the organization.
5. It provides data security by preventing unauthorized use of sensitive data.
6. It reduces data redundancy and improves storage efficiency.
7. By integrating tables created by several users in a central database, it makes data sharing possible throughout the organization.
8. It helps in maintaining centralized control over data.
9. DBMS makes multiple view of the data possible with the help of query languages.
10. DBMS is program independent, therefore, it supports many programs and there is no need to collect different data for different programs.

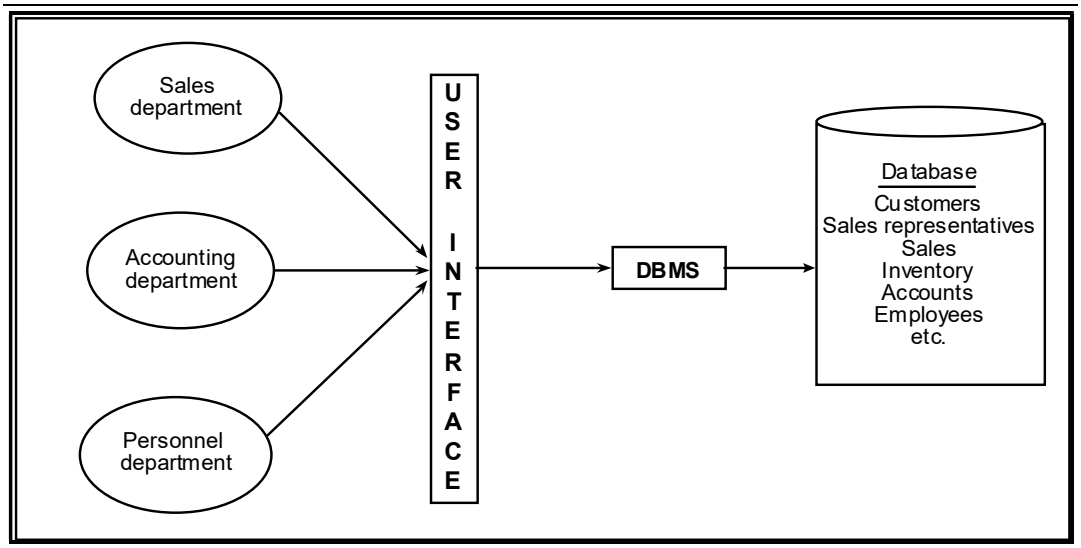


Figure 4.1 : Database System

Advantages of Using DBMS

(*1*) Reduced Data Redundancy: As compared to a conventional file system database management system (DBMS) allows the users to access centrally managed database that reduces the data redundancy and saves storage space.

(*2*) Elimination of inconsistencies: But, using DBMS the database is centrally managed that reduces the chances of having such inconsistencies.

(*3*) Easier Data Access to the Users: The database management system makes the data access by the users easier.

(*4*) Lower Overall Cost

(*5*) Data Security: The DBMS allows enforcement of authentications that reduces the possibility of unauthorized access of the database. This makes the data secure.

(*6*) Development of Data Model

(*7*) Easier Standards Enforcements

Dis-advantages of using Database

(*1*) High Initial Cost

(*2*) Complexity

(*3*) Security Issues

(*4*) Problems of Slower Response

COMPARISION BETWEEN TRADITIONAL FILE SYSTEM AND DATABASE SYSTEM

S. No.	Basis	Traditional File System	Database Management System
1.	Data-Program independence	Data definition is typically part of application program and therefore data and program are dependent to each other.	In DBMS data definition is not part of application program and therefore data and program are not dependent to each other.
2.	Data sharing	In file system, data is stored in separate files in different departments of the organization which makes data sharing throughout the organization difficult.	In DBMS the data is stored centrally in integrated tables created by several users which makes data sharing possible throughout the organization in a very quick time.
3.	Data redundancy	Since separate files are maintained in different departments for usage requirements it leads to duplication or multiplication of data, *i.e.,* data redundancy.	Since, in DBMS the data is stored centrally, it controls data redundancy by removing duplications.
4.	Data security	The degree of data security in file system is very low	The degree of data security is very high in case of DBMS as it prevents unauthorized use of centrally controlled data.
5.	Cost	Traditional file system is a low cost approach of database management	DBMS requires very heavy expenditure on setting up and requires training of personnel for handling the database.

6.	Error correction	If any error is found, may be miss-spelling or any other error, correcting it in all relevant files is very difficult and complex procedure.	The error correction is very fast and smooth in case of DBMS as the database is centrally managed.
7.	Data retrieval	Retrieval of stored data is difficult.	Data retrieval is very easy and quick.

Entity Relationship (ER) Model

An Entity Relationship Model (ER-Model) is a data model that is used to design relational databases. It's main component is Entity Relationship diagram (ER-Diagram) which creates graphical representation of the entities and the relationship between them. It is easy to transform ER-diagrams to relational data models.

The elements of ER-Model are described as follows:

Entity. An entity is a person, place, object, event or concept in the user environment about which the data is collected.

Weak Entity. A weak entity is the entity that cannot be uniquely identified by its attributes alone. For a weak entity to be meaningful it must be a part of one or more relationship set. The weak entity set is depicted by double rectangles.

Attribute. An attribute is a named property or characteristic of an entity that is of interest to an organization.

1. **Composite vs. Simple attributes.** The composite attributes are those attributes which can be further subdivided to have independent meaning.

2. **Single Valued vs. Multivalued attributes.** A single valued attribute can take only a single value such as marital status of an employee can take only one value (married or unmarried, the name of the employee, phone number, etc.).

A multivalued attribute can take more than one value. For example, the qualification of the employee can take multiple values.

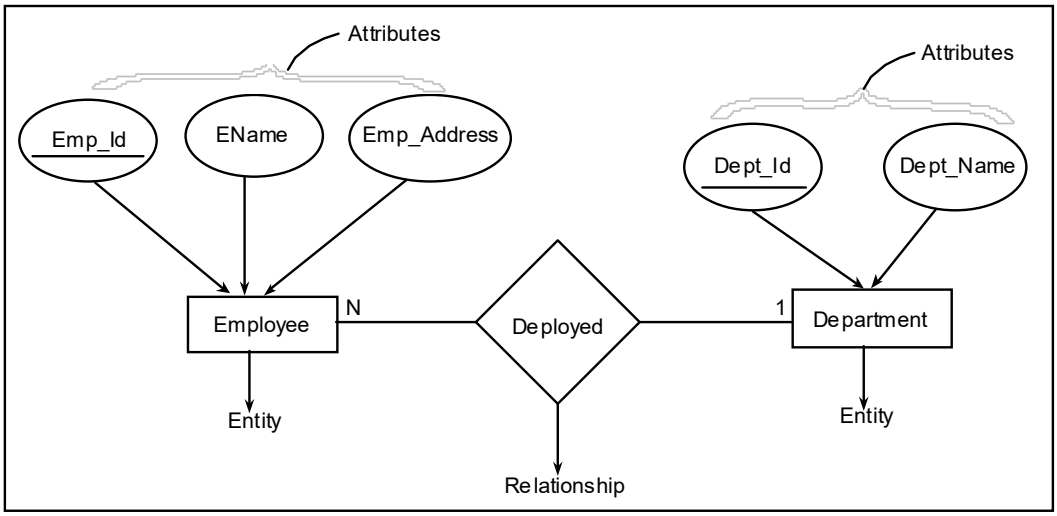
3. **Stored vs. Derived attributes.** Two or more attributes of an entity may relate to each other in such a manner that one or more of them becomes the basic attribute while the other becomes dependent on that basic attribute. The basic attributes which need to be stored directly are stored attributes while those attributes which depend on them becomes derived attributes.

.

Key attribute. A key attribute is the unique, distinguishing characteristic of the entity.

Relationship. A relationship represents some association between two or more entities. It defines the manner in which the entities interact.

Lets take the example of the employee, the employee can be deployed to a department of the organization to serve the organization. It can be shown as :



Relationship between entities

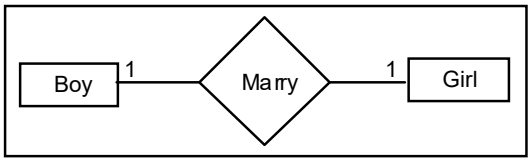
Types of Relationships

There are three types of relationships :

1. One-to-one (1 : 1) relationship
2. One-to-many (1 : N) relationship
3. Many-to-many (M : N) relationship

1. One-to-one relationship. A one-to-one relationship between two entities specifically means for every row of entity 1 there is only one row corresponding to entity 2. A one-to-one relationship is represented as (1 : 1)

For example, take two entities 'boys' and 'girl's. Assuming that one boy will marry only one girl and one girl will marry only one boy, the relationship represented between the entities is one-to-one shown as under in figure 4.5.

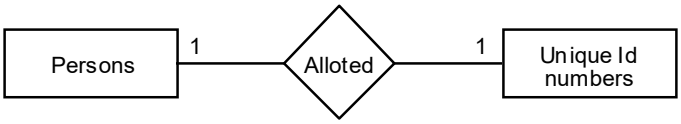


Another example of one-to-one relationship can be the unique identification

numbers issued by several countries (like UID-Aadhar in India).

One person can get only one Id number and one Id number issued to some one can not be issued to some body else.

The relationship represented is 1:1



Persons and their current valid passports do represent one to one relationships.

In some organizations parking spaces are assigned to different employees. It also represents a case of one-to-one relationship.

The nature of relationship that exists between automobiles and engines is also one to one.

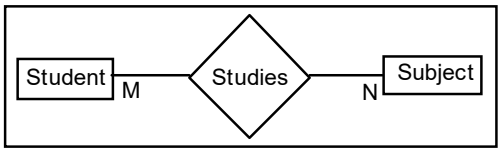
2. One-to-many : One-to-many relationship between two entities is said to exist if for every row in first table (Entity 1) there exist zero, one or many rows in the second table (Entity 2). One to many relationship is represented as (1 : N). In ER-diagram crowfoot (≼) is used to show this relationship in which crowfoot represents many rows.

The one-to-many relationship is shown by the department—employee relationship in figure 4.4.

3. Many-to-Many. Many-to-Many relationship between two tables is said to exist, if for every row in the first table there can be many rows in the second table and for every row of second table there can be many rows in the first table. In Microsoft Access, Many-to-Many relationship cannot be modelled directly. Instead, the relationships are broken in Multiple One-to-Many relationships.

In ER-diagram, Many-to-Many relationship is shown as (M : N).

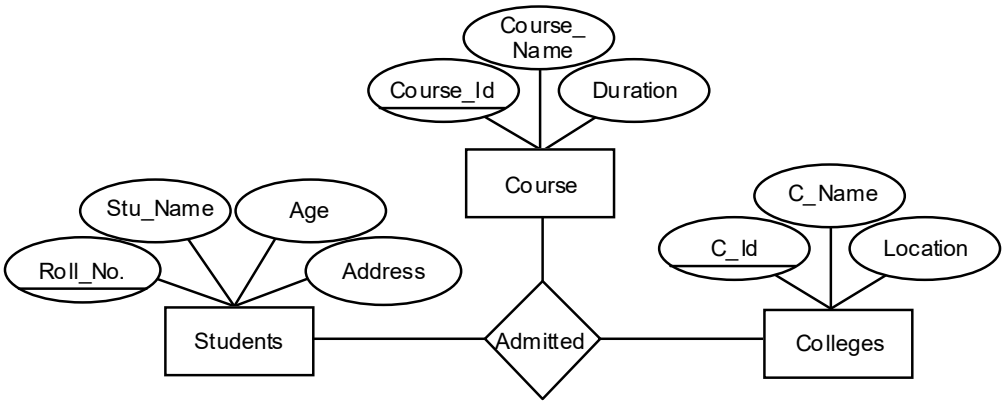
For example, a student of B.Com. (H) has to study more than one subject during the three years of his duration in the college at the same time the same subject can be studied by many students. This results in many-to-many relationship shown as follows :



Example: In a university students (Roll No, Name, Age, Address) are admitted

to different colleges (Id, Name, Location) for different courses (Id, Name, duration). Considering the above facts present an ER–Diagram.

Solution



Specialization

Specialization is a case in which a lower level entity set inherits all the attributes and relationship participation of a higher level entity set with which it is linked. The lower level entities may have additional attributes and they may participate in additional relationships.

The specialization is represented by a triangle in which 'ISA' is written.

Example 4.5: Consider the following:

⇒ Persons can be employees or suppliers of an organization.

⇒ The employees may be managers or the workers.

⇒ Each of them may have additional attributes accordingly.

Represent it by an ER-Diagram.

Solution

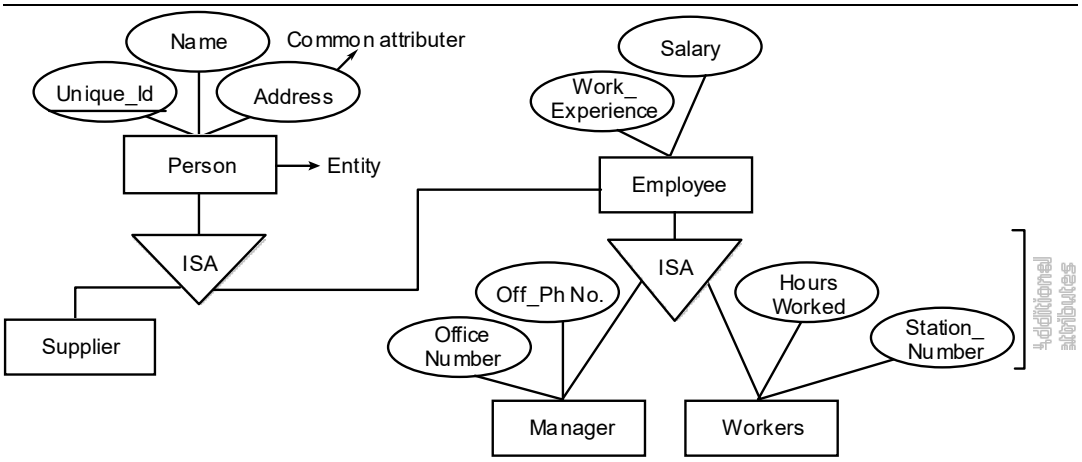


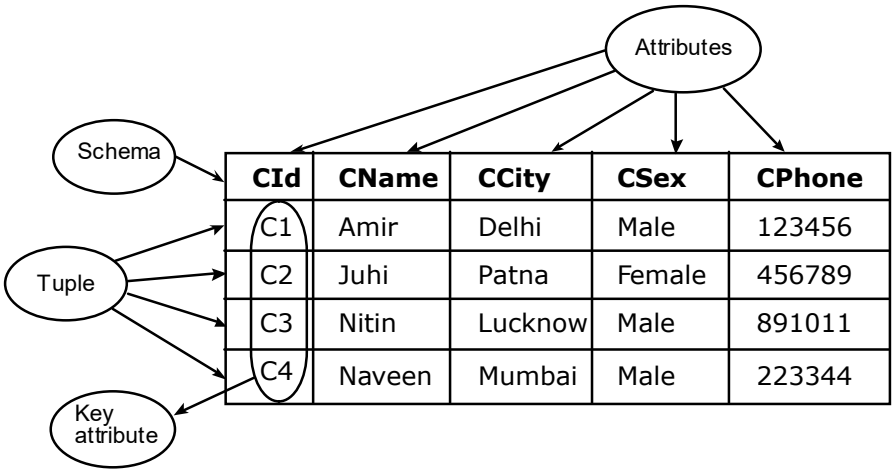
Figure 4.15 : ER-Diagram—Specialization

Relational Data Model

The relational data model was conceived by Edgar F. Codd, a researcher in IBM, in 1970. The Model was based on set theory. In this model, a database consists of a series of un-ordered two dimensional tables. The tables are known as **relations**. The tables are used to represent data. Each table (Relation) represents some real world person, place, thing, event or concept about which the information is collected. A relational table is a flat file composed of a set of named columns and an arbitrary number of unnamed rows. Each row in the table is called a **tuple** which represents a basic fact (data). The order in which the rows appear in a table is immaterial. Each column of the table is given a distinct name which is known as attributes. It provides a unique reference to a particular value of a tuple in the relation.

The set of permitted values which each row can take, for a particular attribute, is called domain.

The following is a customer table (relation) which explains these terms.



The definition of all the domains, and definition of all the relations comprises a schema.

The entity 'Customer' in the above relation has an attribute whose value are distinct for each individual entity in the collection, *i.e.,* CId (Customer Id).No two customers will have the same Customer Id. This attribute is called Key attribute or Identifier. It is used to identify each entity uniquely.

All relations (tables) in a relational database model have three components :

(*i*) **Name :** Name represents title of the entity. In the example given above name is 'Customer'.

(*ii*) **Degree :** Degree represents the number of columns or attributes in a relation (table). The degree of the relation given above, *i.e.,* 'Customer' is 5.

(*iii*) **Cardinality :** Cardinality represent the number of rows in a relation (table). In the example given earlier cardinality is 4.

4.6.2.1. Properties of Relational Tables

The relational tables have five properties :

1. The values in a relational table are **atomic.** Any attribute, if in a group, is required to be converted into an atomic form. For example name attribute can be further divided into First_Name, Middle_Name and Last_Name. It simplifies the data manipulation logic.
2. Each row of a relational table is unique and it is identified by its primary key.
3. The column values are of the same kind. All values in a column come from the same domain.

4. The order or sequence of the columns and rows in a relational table is insignificant.
5. Each column (attribute) has a unique name within the table (Relation) it belongs to.

4.6.2.2. Creating Relationship between Tables

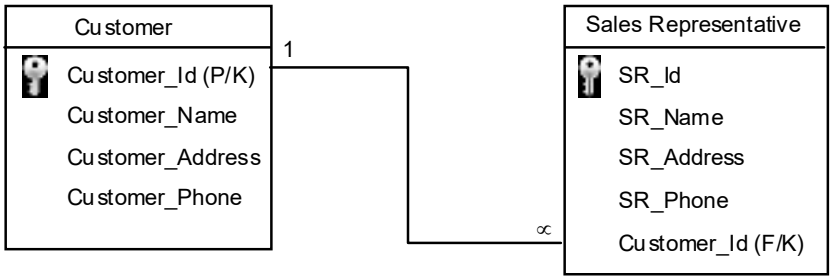
In Relational Database Management System (RDBMS), the data are stored in the form of tables and all the operations, *e.g.,* insert, delete, edit, retrieve etc. are carried out using these tables only.

Each table represents only one type of object. For example, in a database, there can be tables with names 'Customer', as we prepared earlier, 'Transactions', Inventories, Sales Representative, Receipts, etc. Each table contains only one type of data. In customer table, the details of sales representatives will not be stored. In case a link is required between any two of these a relationship need to be established.

Assume a database having two tables 'Customer', and 'Sales Representative' in the database. The details of customers will be recorded in the table 'Customer' and the details of sales representatives will be kept in the table 'Sales representative'.

To show an interaction between sales representatives and customers a foreign key needs to be inserted in Sales Representative Table which will show the link between these two tables.

This is shown as follows.



Every transaction will have a sales representative who will deal with a customer to sell a product. In each of the two tables, one attribute Customer_Id in Customer table and SR_Id in Sales Representative table will have a primary key to represent uniqueness of a record. Both the tables need to be connected so that data duplicacy can be controlled and the interaction between sales representatives and customers can be shown. This is achieved by setting a relationship between the tables. A relationship is a manner in which one table is linked with another table in the database. The relationship may be 'one-to-one', 'one-to-many', and 'many-to-many'

as already explained in ER-diagram part of the chapter. In the example given above the type of relationship is one to many.

For the sake of simplicity naming conventions have been ignored and the attributes are kept to a minimum level in the explained example.

Concept of Keys

A key is a data-item that is used to identify a record stored in a database. There are six types of keys used in RDBMS explained as under:

1. Super Key : A super key is a set of one or more attributes of a relation (table) whose combined value uniquely identifies a database record.

2. Candidate Key : A candidate key is an attribute or a set of attributes which uniquely identifies a record (tuple).

3. Primary Key : A primary key is an attribute or a group of attributes which uniquely identifies a tuple. A primary key can never be same for two or more records.

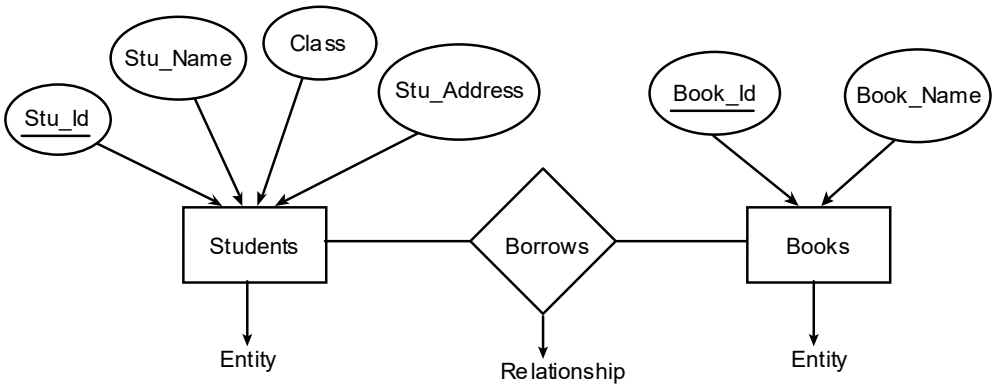
4. Secondary Key : A secondary key is an attribute or a combination of attributes which classifies the entity set on a particular characteristics

5. Composite Key : A composite key is a combination of attributes which uniquely identifies a record in a table

6. Foreign Key : A foreign key is that attribute of a table (relation) whose values correspond to the values of a primary key of another table.

ER-diagram to Relational Mapping

An ER-diagram is prepared by the designer to capture a real world situation and to give a pictorial representations of the interaction between the entities. Later the ER-diagram need to be mapped to the relational model. To explain the process, the example taken is that a library issues books to the students. For simplicity, two entities student and books are taken. Diagramatically it can be presented as :

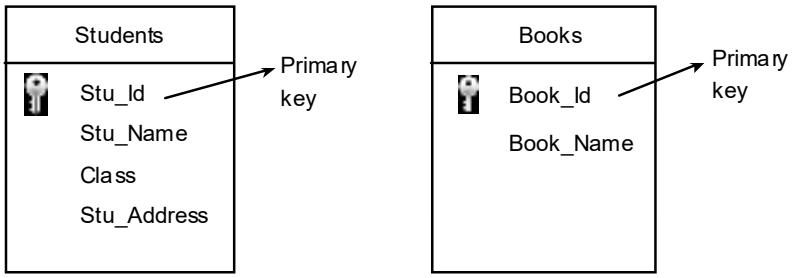


The designer proceeds with the mapping, preserving as many integrity constraints as possible. There are three steps to be taken in the process :

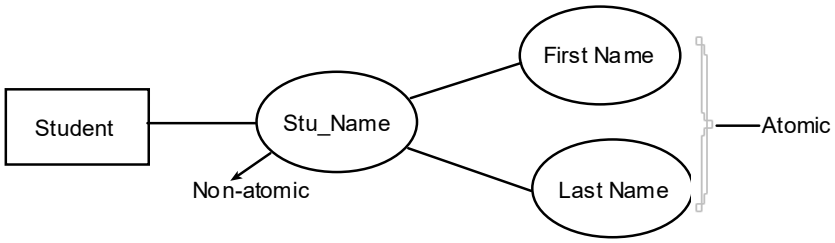
1. Conversion of entities,
2. Conversion of relationships, and
3. Conversion of multivalued attributes.

Entity Conversion

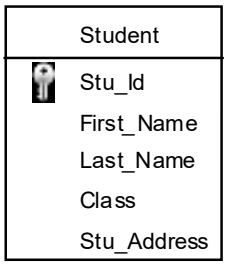
The entity sets are translated into relations (tables). The attributes of entities become the attributes of the relations. Any key attribute is assigned a primary key. In the above example, there will be two tables (Relations).



During this conversion all non-atomic attribute are converted into atomic attributes. For example, if Student Name contains first name and last name then Stu_Name should be replaced with atomic attributes.



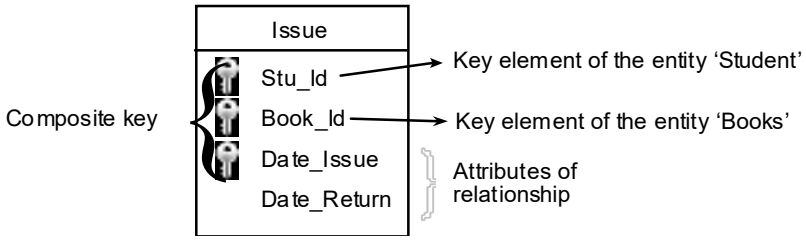
The relation will be re-written as



Conversion of Relationships

The relationship sets between the entities are translated into tables (relations).This table assumes the attributes of the relationship set and the key attributes of participating entities. To continue with the example of the library, the relationship between the entities is borowing of books and the key attributes of participating entities are Book_Id, and Stu_Id.

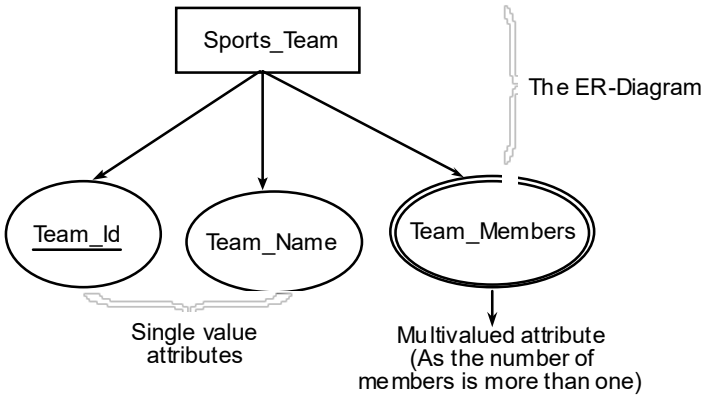
An 'Issue' table will be created as given below:



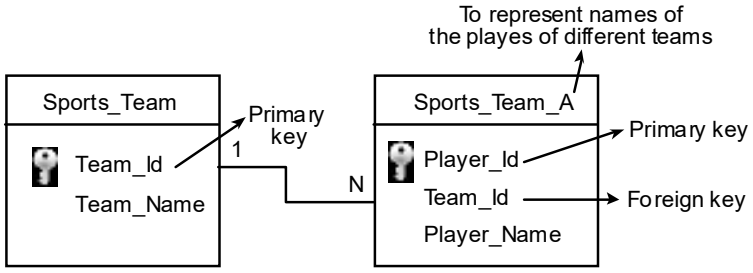
Conversion of Multivalued Attributes

In conventional database design, each row and column must store only one value. For example, in a relation if you add a column (attribute) qualification of employees, this attribute may assume more than one value like matriculation, graduation, masters, etc. Since only one row belongs to one employee, all these values are added in one cell only that is known as multivalued attributes. But in conventional database a multivalued attribute need to be converted into a single value attribute. To convert multivalued attributes into single value attributes, a new relation (table) is created which contains the attribute and the Primary Key of the relation.

For example, the sports team of a college has more than one players (data value). In this case, the table will have the following attributes.



To map this in a relational database, two relations (tables) will be created in the following manner.



Normalization of Relational Database

Normalization is the process of efficiently organizing data. It is the process adopted to structure the data in a manner which removes or controls data redundancy. It contains a set of rules which are concerned with

⇒ Identifying relationships among the attributes

⇒ Converting those relationships in the form of relations (tables), and

⇒ Combining these relations to form a database.

There are two objectives of data normalization

1. Controlling redundant data, and
2. Making sure that the data dependencies make sense, *i.e.* only related data are stored into tables.

It provides the following **advantages** :

1. Maintains data integrity,
2. Optimizes query response, and
3. Improves process of updating data.

There are different stages of data normalization, but the discussion in this chapter is limited to only first three forms which are often used by the database designers.

First Normal Form (1NF)

The first normal form sets the very basic rules for organizing database. A relation (table) said to be in first normal form if all the key attributes are defined, attribute values are atomic and the table doesn't contain any repeating group.

There are two steps for a making tables in the first normal form.

1. Eliminate duplicative columns from the table, and
2. Create separate table for related data of each group and identity each row with the primary key.

Second Normal Form (2NF)

A relation (table) is said to be in 2nd normal form

⇒ If it is in first normal form (1NF) and

⇒ If it contains a multi-attribute key (composite key) then all the non key attributes must be dependent on the complete set of the key attribute (composite key) and not on any part of it.

Third Normal Form (3NF)

A table is said to be in Third Normal Form (3NF) if

⇒ it is in first and second normal form, and

⇒ every non-key attribute is directly dependent on the key attribute.

In a table which is in third normal form, all the non-key attributes are independent of each other and directly depend upon the key attribute.