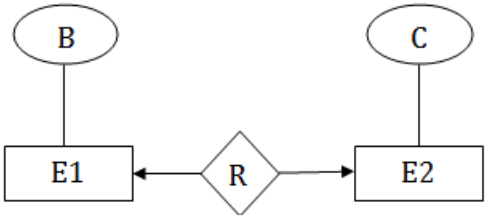
Mapping Constraints

- A mapping constraint is a data constraint that expresses the number of entities to which another entity can be related via a relationship set.
- It is most useful in describing the relationship sets that involve more than two entity sets.
- For binary relationship set R on an entity set A and B, there are four possible mapping cardinalities.

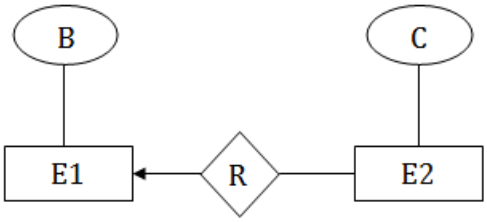
One-to-one

In one-to-one mapping, an entity in E1 is associated with at most one entity in E2, and an entity in E2 is associated with at most one entity in E1.



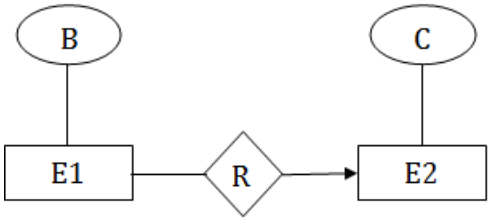
One-to-many

In one-to-many mapping, an entity in E1 is associated with any number of entities in E2, and an entity in E2 is associated with at most one entity in E1.



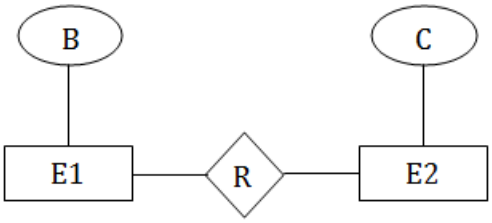
Many-to-one

In one-to-many mapping, an entity in E1 is associated with at most one entity in E2, and an entity in E2 is associated with any number of entities in E1.



Many-to-many

In many-to-many mapping, an entity in E1 is associated with any number of entities in E2, and an entity in E2 is associated with any number of entities in E1.



Data Models

Underlying the structure of a database is the **data model**: a collection of conceptual tools for describing data, data relationships, data semantics, and consistency constraints.

There are two main data models - the entity-relationship model and the relational model. Both provide a way to describe the design of a database at the logical level.

Entity-Relationship Model

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

- An entity is a “thing” or “object” in the real world that is distinguishable from other objects. For example, each person is an entity, and bank accounts can be considered as entities.
- Entities are described in a database by a set of **attributes**.
- For example, the attributes *account-number* and *balance* may describe one particular account in a bank, and they form attributes of the *account* entity set.
- Similarly, attributes *customer-name*, *customer-street* address and *customer-city* may describe a *customer* entity. customer-id is used to uniquely identify customers.
- A unique customer identifier must be assigned to each customer.

A **relationship** is an association among several entities.

For example, a *depositor* relationship associates a customer with each account that he/she has. The set of all entities of the same type and the set of all relationships of the same type are termed an **entity set** and **relationship set**, respectively.

Entity-Relationship Diagram

The overall logical structure (schema) of a database can be expressed graphically by an *E-R diagram*, which is built up from the following components:

• **Rectangles**, which represent entity sets

• **Ellipses**, which represent attributes

• **Diamonds**, which represent relationships among entity sets

• **Lines**, which link attributes to entity sets and entity sets to relationship sets

• **Double ellipses**, which represent multivalued attributes

• **Dashed ellipses**, which denote derived attributes

• **Double lines**, which indicate total participation of an entity in a relationship set

• **Double rectangles**, which represent weak entity sets.

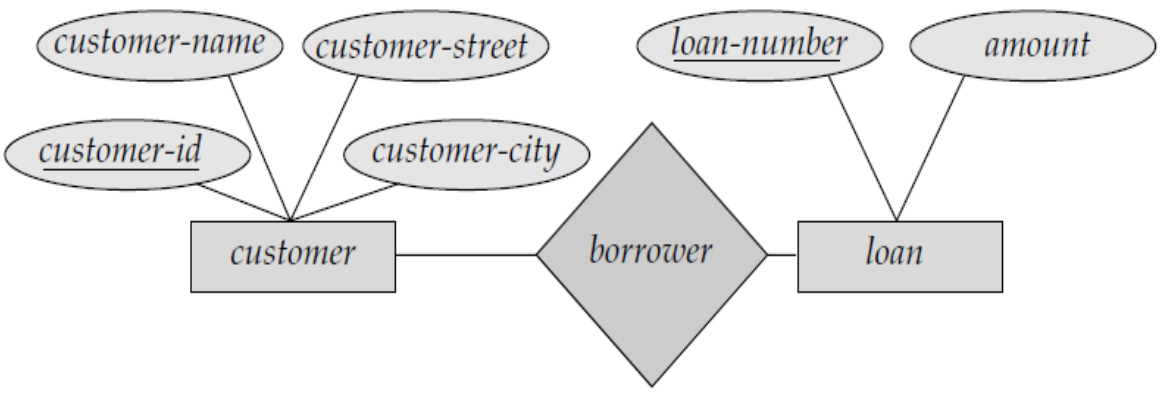


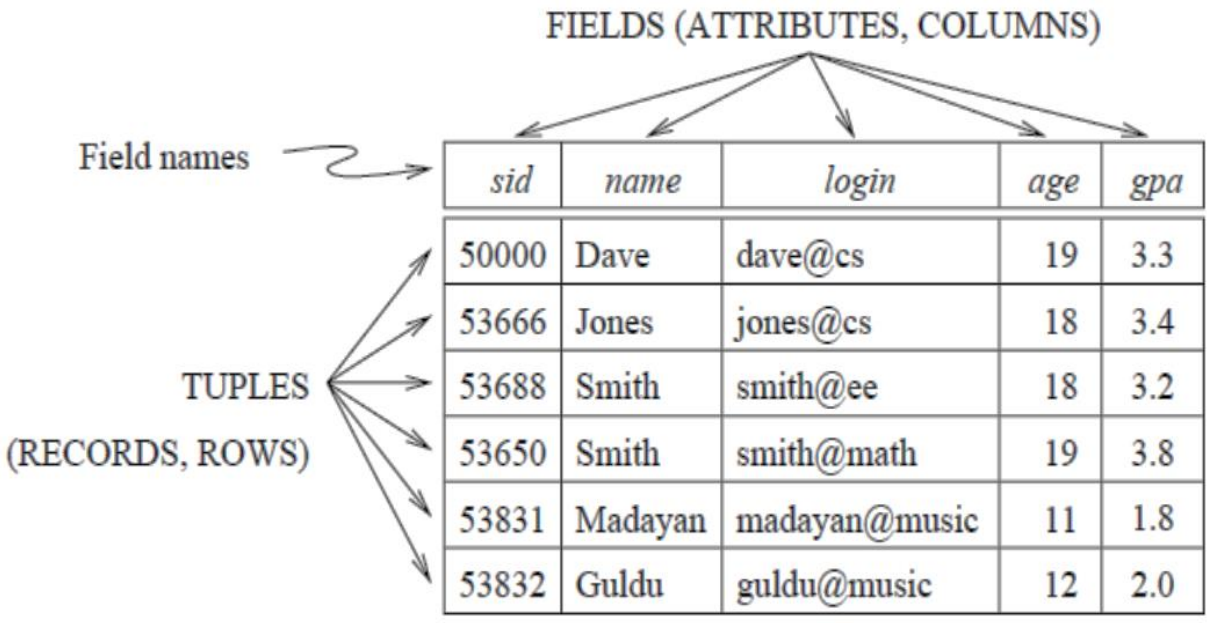
Figure - A sample E-R diagram

Relational Model

- The relational model uses a collection of tables to represent both data and the relationships among those data.
- Each table has multiple columns, and each column has a unique name. Tables are also known as **relations**.
- The relational model is an example of a **record-based model**.
- Record-based models are so named because the database is structured in fixed-format records of several types.
- Each table contains records of a particular type. Each record type defines a fixed number of fields, or attributes. The columns of the table correspond to the attributes of the record type. The relational data model is the most widely used data model.

Example:

NAME	LOCATION	CITY	PHONE NO.	ACCOUNT No.
ANAND	KORAMANGALA	BANGALORE	534278	401
VIKRAM	AUDOGODI	BANGALORE	546678	402

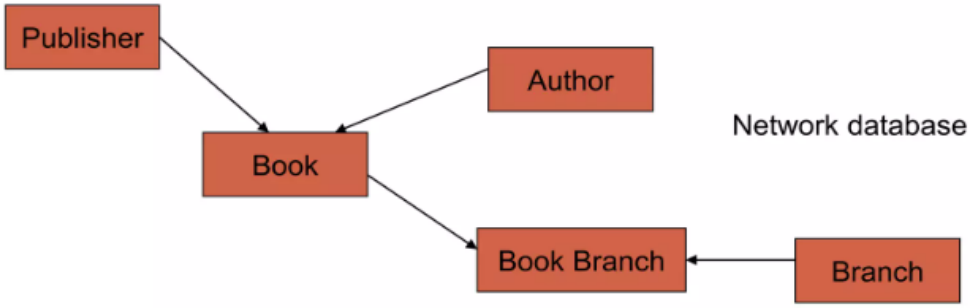


sid	*name*	*login*	*age*	*gpa*
50000	Dave	dave@cs	19	3.3
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith@math	19	3.8
53831	Madayan	madayan@music	11	1.8
53832	Guldu	guldu@music	12	2.0

Network Data Model

Data is represented as a collection of records and relationship between data is represented by links which can be viewed as pointers. The record in the database are organized as collection of arbitrary graphs.

Example:



Network database structure

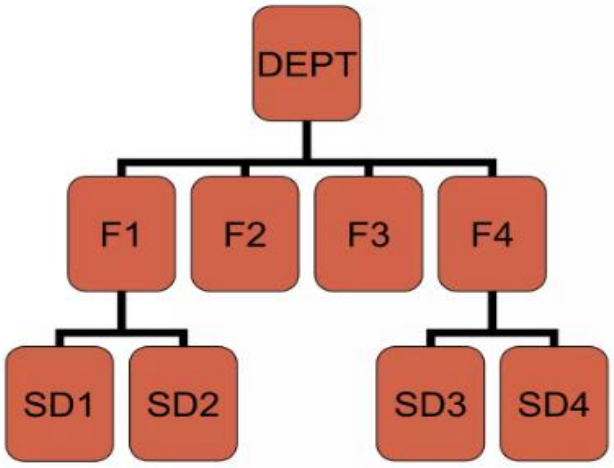
Network database model

Hierarchical Data Model

It is different from network model in the way that records are organized into a tree like structure.

For eg. An organization might store information about an employee, such as name, dep, sal. The organization might also store information about employee's family. The employee and the family data forms hierarchy.

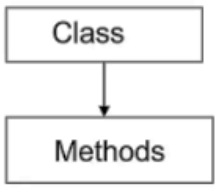
Example:



F – Faculty
S - Student

Object Oriented Data Model

It defines the database in terms of objects, their properties and their operations.



Objects with some structure and behavior.

Operations of each class in terms of predefined procedure.

- Support the basic elements of the object approach used in object oriented programming languages like inheritance, use of methods, and encapsulation.
- Some object-oriented databases are designed to work well with object oriented programming languages such as Java, C++, C# etc.
- OODBMS use exactly the same model as object-oriented programming languages.

Database Languages

A database system provides a **data-definition language** to specify the database schema and a **data-manipulation language** to express database queries and updates. There are four types of database languages.

Data-Definition Language

We specify a database schema by a set of definitions expressed by a special language called a **data-definition language** (**DDL**).

We specify the storage structure and access methods used by the database system by a set of statements in a special type of DDL called a **data storage and definition** language.

Data-Manipulation Language

A **data-manipulation language (DML)** is a language that enables users to access or manipulate data as organized by the appropriate data model. The types of access are:

• Retrieval of information stored in the database

• Insertion of new information into the database

• Deletion of information from the database

• Modification of information stored in the database

There are basically two types:

Procedural DML require a user to specify *what* data are needed and *how* to get those data.

Non-Procedural DML require a user to specify *what* data are needed *without* specifying how to get those data.

Data Control Language

DCL works to deal with SQL commands that are used to permit a user to access, modify and work on a database. it is used to access stored data. It gives access, revokes access, and changes the permission to the owner of the database as per the requirement.

Commands used in DCL are Grant, Revoke.

Transaction Control Language

It can be grouped into a logical transaction and is used to run the changes made by the DML command in the database.

Commands used in TCL are Commit, Rollback.

Query

A **query** is a statement requesting the retrieval of information. The portion of a DML that involves information retrieval is called a **query language**.

Domain Constraints

A domain of possible values must be associated with every attribute (for example, integer types, character types, date/time types).

Declaring an attribute to be of a particular domain act as a constraint on the values that it can take. Domain constraints are the most elementary form of integrity constraint. They are tested easily by the system whenever a new data item is entered into the database.

Database Administrator

One of the main reasons for using DBMS is to have central control of both the data and the programs that access those data. A person who has such central control over the system is called a **database administrator (DBA)**.

The functions of a DBA include:

- **Schema definition** - The DBA creates the original database schema by executing a set of data definition statements in the DDL.

- **Storage structure and access-method definition** – The DBA creates the Storage structure and access-method definition.

- **Schema and Physical-Organization Modification** - The DBA carries out changes to the schema and physical organization to reflect the changing needs of the organization, or to alter the physical organization to improve performance.

- **Granting of Authorization for Data Access** - By granting different types of authorization, the database administrator can regulate which parts of the database various users can access.
  The authorization information is kept in a special system structure that the database system consults whenever someone attempts to access the data in the system.

- **Routine maintenance** - Examples of the database administrator's routine maintenance activities are:

1) Periodically **backing up the database**, either onto tapes or onto remote servers, to prevent loss of data in case of disasters such as flooding.
2) Ensuring that **enough free disk space is available** for normal operations, and upgrading disk space as required.
3) **Monitoring jobs running on the database** and ensuring that performance is not degraded by very expensive tasks submitted by some users.