UNIT-2

DATA MODELING AND RECORD STORAGE

IMPORTANT QUESTIONS

Section-A: (2 Marks)

1. Define specialization and Generalization

Generalization is the process of defining a general entity type from a set of specialized entity types by identifying their common characteristics.

Specialization is a process of defining one or more subclasses of a superclass by identifying their distinguishing characteristics.

2.What is Heap File? How pages are organized in a Heap File.

Files of Unordered Records (Heap Files)
In this simplest and most basic type of organization, records are placed in the file in the order in which they are inserted, so new records are inserted at the end of the file. Such an organization is called a **heap** or **pile file.**

3.Define 1)track 2)sector 3)cylinder

TRACK: Information is stored on a disk surface in concentric circles of small width, each having distinct diameter. Each circle is called track.

SECTOR: track usually contains a large amount of information; it is divided into smaller blocks or sectors. The division of a track into **sectors** is hard-coded on the disk surface and cannot be changed.

CYLINDER: In disk packs, tracks with the same diameter on the various surfaces are called a **cylinder** because of the shape they would form if connected in space.

4. Mention two approaches used for Database design.

BOTTOM-UP APPROACH :Generalization is a bottom-up approach as it starts with the specialized entity types (subclasses) and forms a generalized entity type (superclass).

TOP-DOWN APPROACH :Specialization is a top down approach.It starts with the general entity (superclass) and forms specialized entity types (subclasses) based on specialized attributes or relationships specific to a subclass.

5. What is attribute? Mention the difference between single valued and multi valued attribute

ANS: An attribute is a property that describes an entity.

For ex: attributes of a person entity are his name, address, job, salary.

DIFFERENCE BETWEEN SINGLE VALUED AND MULTI VALUED ATTRIBUTE:

-

SINGLE VALUED ATTRIBUTE	MULTIVALUED ATTRIBUTE
An attribute that can take only one value at a time. For a particular entity, each attribute will have single value. Ex: Age attribute will have single value	An attribute which contain more than one value for the same attribute. these attributes are called multi valued attribute. Ex: 1 Degree(BE,BCA,MRIE,MCA) 2 Carcolor(Red,Black)

6. How are storage devices classified?

There are two main categories of computer storage. They are:

>Primary storage

>secondary and tertiary storage

Primary storage.

*****This category includes storage media that can be operated on directly by the computer's *central processing unit* (CPU). such as the computer's main memory and smaller but faster cache memories.

* Primary storage usually provides fast access to data.

* The disadvantage is that it is of limited storage capacity.

*Examples include RAM. ROM, and CACHE memories.

Secondary and tertiary storage.

- Examples are magnetic disks, optical disks (CD-ROMs, DVDs, and other similar storage media), and tapes.

- Hard-disk drives are classified as secondary storage, whereas removable media such as optical disks and tapes are considered tertiary storage.

- The advantages are

*Larger storage capacity.

* Less Cost

*The disadvantage is that access is slow compared to primary memory.

- Data in secondary or tertiary storage cannot be processed directly by the CPU

7. Define spanned and unspanned records.

SPANNED RECORDS

When records can span more than one block they are called spanned records. This is possible when unused space of a block can be used by storing some part of records in one block and another part in another record.

UNSPANNED RECORDS

Records that do not cross block boundaries are said to be unspanned records. This is used with fixed length records because it makes each record start at a known location in the block,simplifying record and processing.

8. What is Hashing?

Hashing is a method which provides very fast access to records under certain search conditions. This organization is usually called a **hash file**.

9. Define (i) Entity (ii) Relationship (iii) ER Model .

ENTITY: An ***entity*** is an object that exists and which is distinguishable from other objects. An entity can be a person, a place, an object, an event, or a concept about which an organization wishes to maintain data.

For ex: in a school database, student, teachers, class and course offered can be considered as entities

RELATIONSHIP: The association among entities is called relationship.

For example, employee entity has relation works at with department.

ER MODEL:A logical representation of the data for an organization or for a business area is called E-R Model. It is also called has Entity-Relationship Model.

10. What is RAID?

RAID (originally **redundant array of inexpensive disks**; now commonly **redundant array of independent disks**) is a data storage that combines multiple disk drive components into a logical unit for the purposes of data redundancy or performance improvement.

11. What is blocking factor?

Blocking factor can be defined as the ratio of block size to record size.

Blocking factor bfr = Block size(B)/Record size(R) records per block.

12. What is Ordering field ?

The records of a file can be physically ordered based on the values of one of their fields called the ordering field. This is called an ordered or sequential file.

Section-B: (5 Marks and 10 Marks)

1. Explain Hashing techniques in Detail

Hashing is a method which provides very fast access to records under certain search conditions. This organization is usually called a **hash file**. The search condition must be an equality condition on a single field, called the **hash field**. In most cases, the hash field is also a key field of the file, in which case it is called the **hash key**.

The idea behind hashing is to provide a function *h*, called a **hash function** or **randomizing function**, which is applied to the hash field value of a record and yields the *address* of the disk block in which the record is stored. A search for the record within the block can be carried out in a main memory buffer. For most records, we need only a single-block access to retrieve that record.

Internal Hashing

Hashing is typically implemented as a **hash table** through the use of an array of records. Let the array index range is from 0 to M–1, as shown in Figure(a); then we have *M* **slots** whose addresses correspond to the array indexes. We choose a hash function that transforms the hash field value into an integer between 0 and M-1. One common hash function is the $h(K) = K$ **mod** M function, which returns the remainder of an integer hash field value K after division by M; this value is then used for the record address.

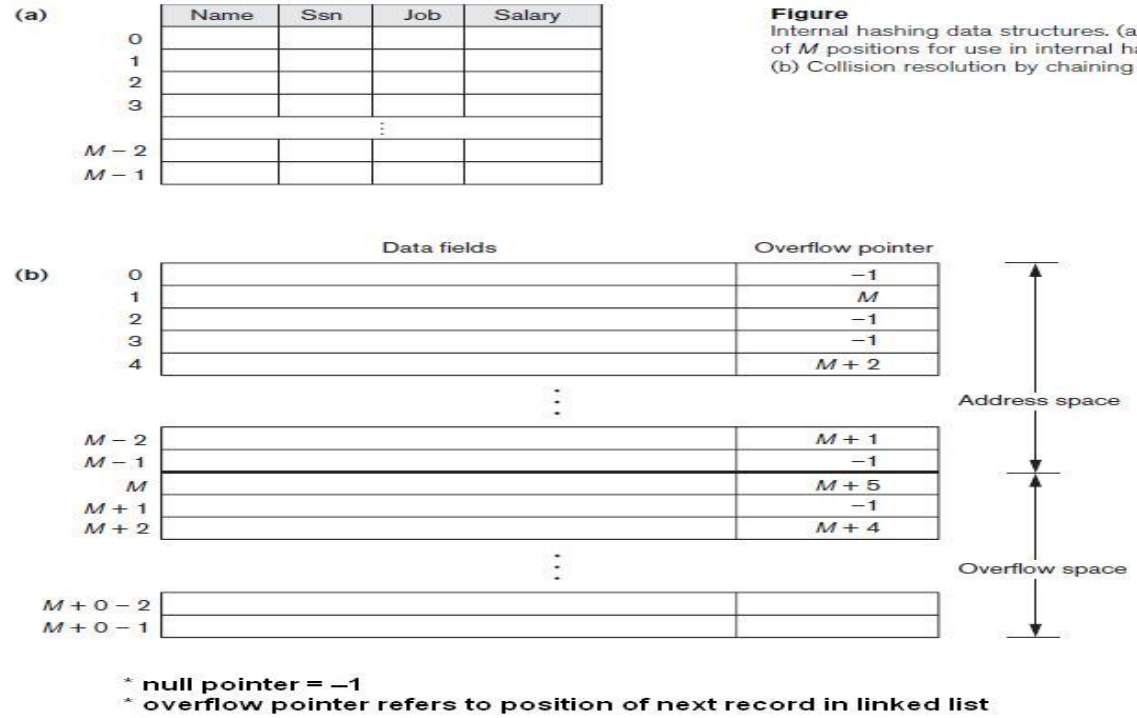


Figure
Internal hashing data structures. (a) Array of *M* positions for use in internal hashing. (b) Collision resolution by chaining records.

One technique, called **folding**, involves applying an arithmetic function such as *addition* or a logical function such as *exclusive or* to different portions of the hash field value to calculate the hash address.

A **collision** occurs when the hash field value of a record that is being inserted hashes to an address that already contains a different record. In this situation, we must insert the new record in some other position. The process of finding another position is called **collision resolution**.

There are numerous methods for collision resolution, including the following:

***Open addressing*.** Ponce a position specified by the hash address is found to be occupied, the program checks the subsequent positions in order until an unused (empty) position is found. Algorithm (b) may be used for this purpose.

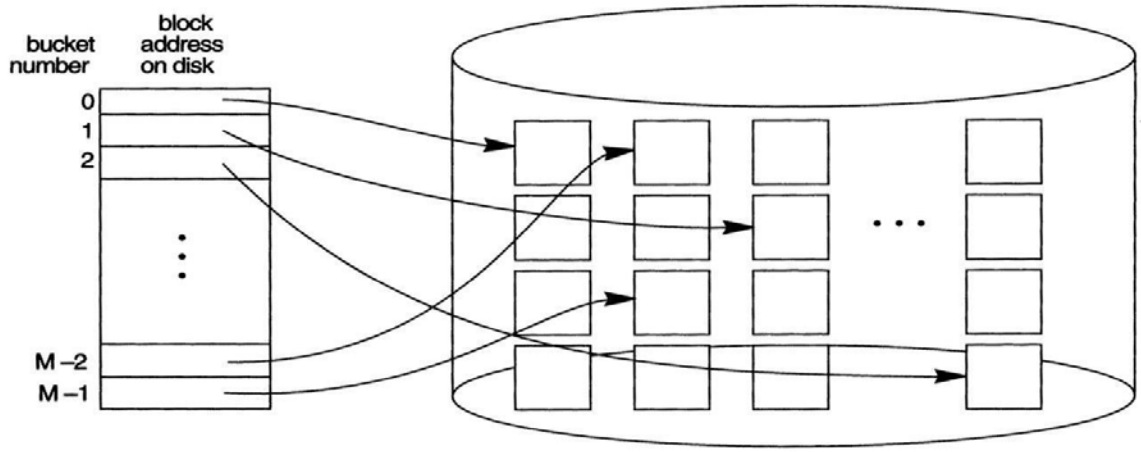
Chaining.For this method, various overflow locations are kept, usually by extending the array with a number of overflow positions. Additionally, a pointer field is added to each record location. A collision is resolved by placing the new record in an unused overflow location and setting the pointer of
the occupied hash address location to the address of that overflow location. A linked list of overflow records for each hash address is thus maintained, as shown in Figure (b).

***Multiple hashing*.**The program applies a second hash function if the first results in a collision. If another collision results, the program uses open addressing or applies a third hash function and then uses open addressing if necessary.

External Hashing for Disk Files

Hashing for disk files is called **external hashing**. the target address space is in external hashing is made of **buckets**, A bucket is either one disk block or a cluster of contiguous blocks. The hashing function maps the indexing field's value into a relative bucket number. A table maintained in the file header converts the bucket number into the corresponding disk block address as shown in below Figure

.



The hashing scheme is called **static hashing** if a fixed number of buckets is allocated.
A major drawback of static hashing is that the number of buckets must be chosen large enough that can handle large files. That is, it is difficult to expand or shrink the file dynamically.

2. Explain various methods for allocating the blocks of Files on Disk?

Allocating File Blocks on Disk

There are several standard techniques for allocating the blocks of a file on disk.

In ***contiguous allocation***, the file blocks are allocated to consecutive disk blocks. This makes reading the whole file very fast using double buffering, but it makes expanding the file difficult.

In ***linked allocation***, each file block contains a pointer to the next file block. This makes it easy to expand the file but makes it slow to read the whole file. A combination of the two allocates **clusters** of consecutive disk blocks, and the clusters are linked. Clusters are sometimes called **file segments** or **extents**.

In ***indexed allocation***, where one or more **index blocks** contain pointers to the actual file blocks. It is also common to use combinations of these techniques.

3.Explain various types of attributes with Examples

Attributes

An attribute is a property that describes an entity. All attributes have values. For example, a student entity may have name, class, age as attributes. There exists a domain or range of values that can be assigned to attributes. For example, a student's name cannot be a numeric value. It has to be alphabetic. A student's age cannot be negative, etc.

Types of attributes:

Simple attribute:
Simple attributes are atomic values, which cannot be divided further. For example, student's phone-number is an atomic value of 10 digits
.

***Composite attribute*:**
Composite attributes are made of more than one simple attribute.
For example, a student's name may have Firstname and Lastname.

Derived attribute:
Derived attributes are attributes, which do not exist physical in the database, but there values are derived from other attributes presented in the database.

For another example, Age can be derived from DOB.

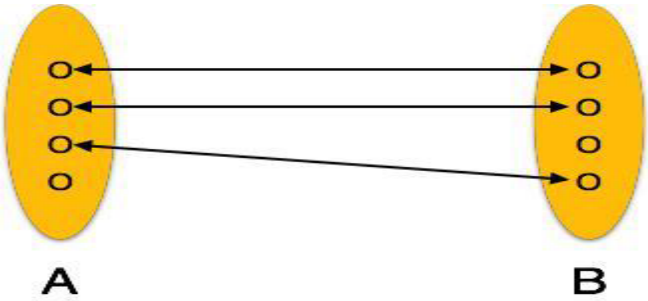
Stored attribute:
An attribute whose value cannot be derived from the values of other attributes is called a stored attribute. For example, DOB

***Single-valued attribute*:**
Single valued attributes contain on single value. For example: SocialSecurityNumber.

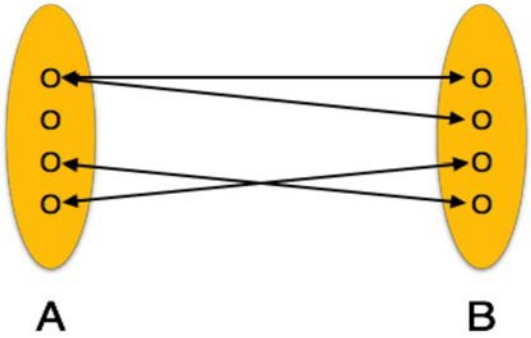
Multi-value attribute:
Multi-value attribute may contain more than one values.
For example, a person can have more than one phone numbers, EmailId etc.

4. Explain different types of relationships with examples

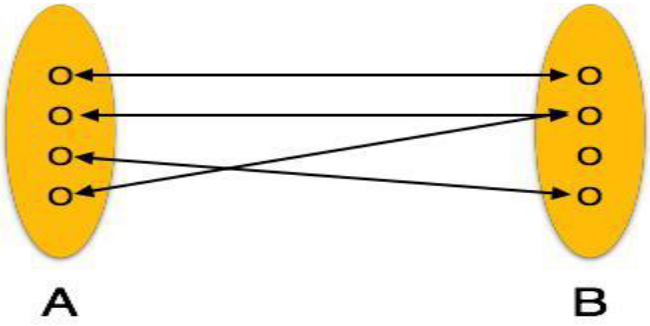
Relationship
The association among entities is called relationship. For example, employee entity has relation works at with department. Another example is for student who enrolls in some course. Here, Works_at and Enrolls are called relationship.
One-to-one: one entity from entity set A can be associated with at most one entity of entity set B and vice versa.



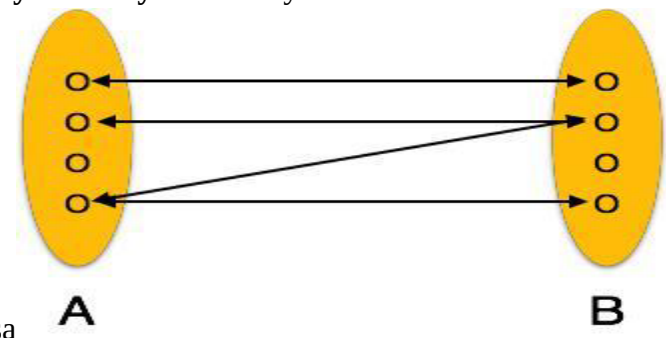
One-to-many: One entity from entity set A can be associated with more than one entities of entity set B but from entity set B one entity can be associated with at most one entity.



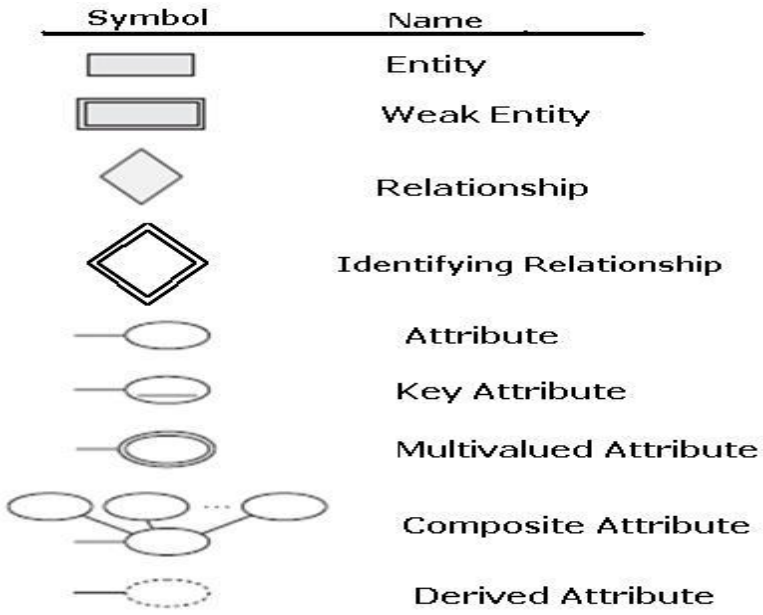
Many-to-one: More than one entities from entity set A can be associated with at most one entity of entity set B but one entity from entity set B can be associated with more than one entity from entity set A.



Many-to-many: one entity from A can be associated with more than one entity from B and vice versa



5. Explain various notations in ER diagram.

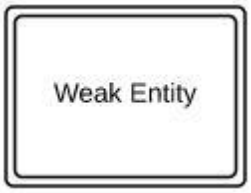


Entity:an entity can be any object, place person or class.

In E-R diagram entity is represented using rectangles.

For example Student is an entity. 

Strong Entitiesare independently from other entity types. They always possess one or more attributes that uniquely(primary key) distinguish each occurrence of the entity. For example Student is an entity.

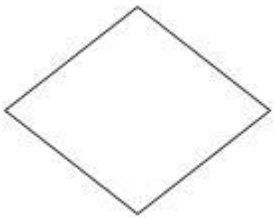
Weak Entitiesdepend on another entity. 

Weak entity doesn't have key attribute of their own.

Double rectangle represents weak entity.

Relationship

A relationship describes relations between entities. Relationship is represented using diamonds.

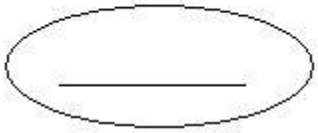


Attributes:

An attributes describes a property or characteristic of an entity. An attribute is represented using eclipse For example regno, name, course can be the attribute of student entity.

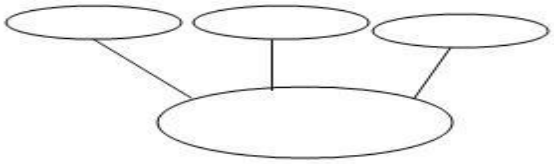
Key Attribute

Key attribute represents the main characteristic of an entity. It is used to represent Primary key. Ellipse with underlying lines represent key attribute.



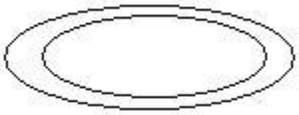
Composite Attribute

An attributes can be sub divided. These attributes are known as composite attribute.



Multivalued Attribute

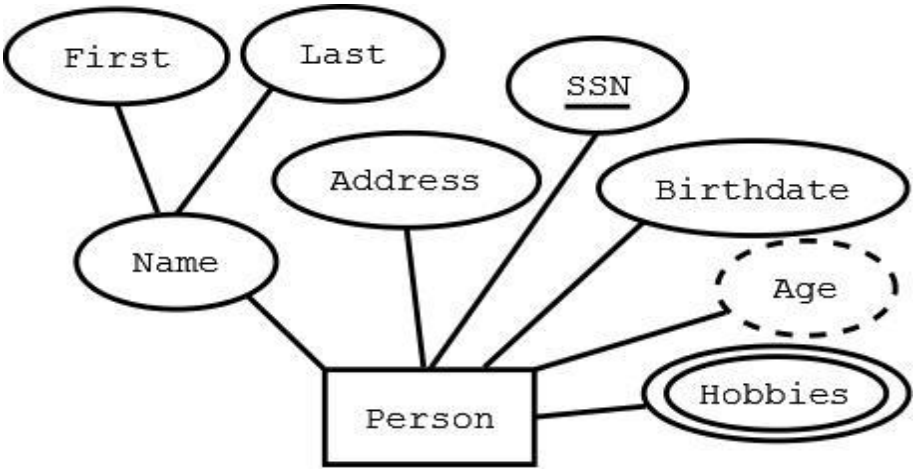
Multivalued attributes are those that are capable of taking on more than one value. It is represented by double Ellipse.



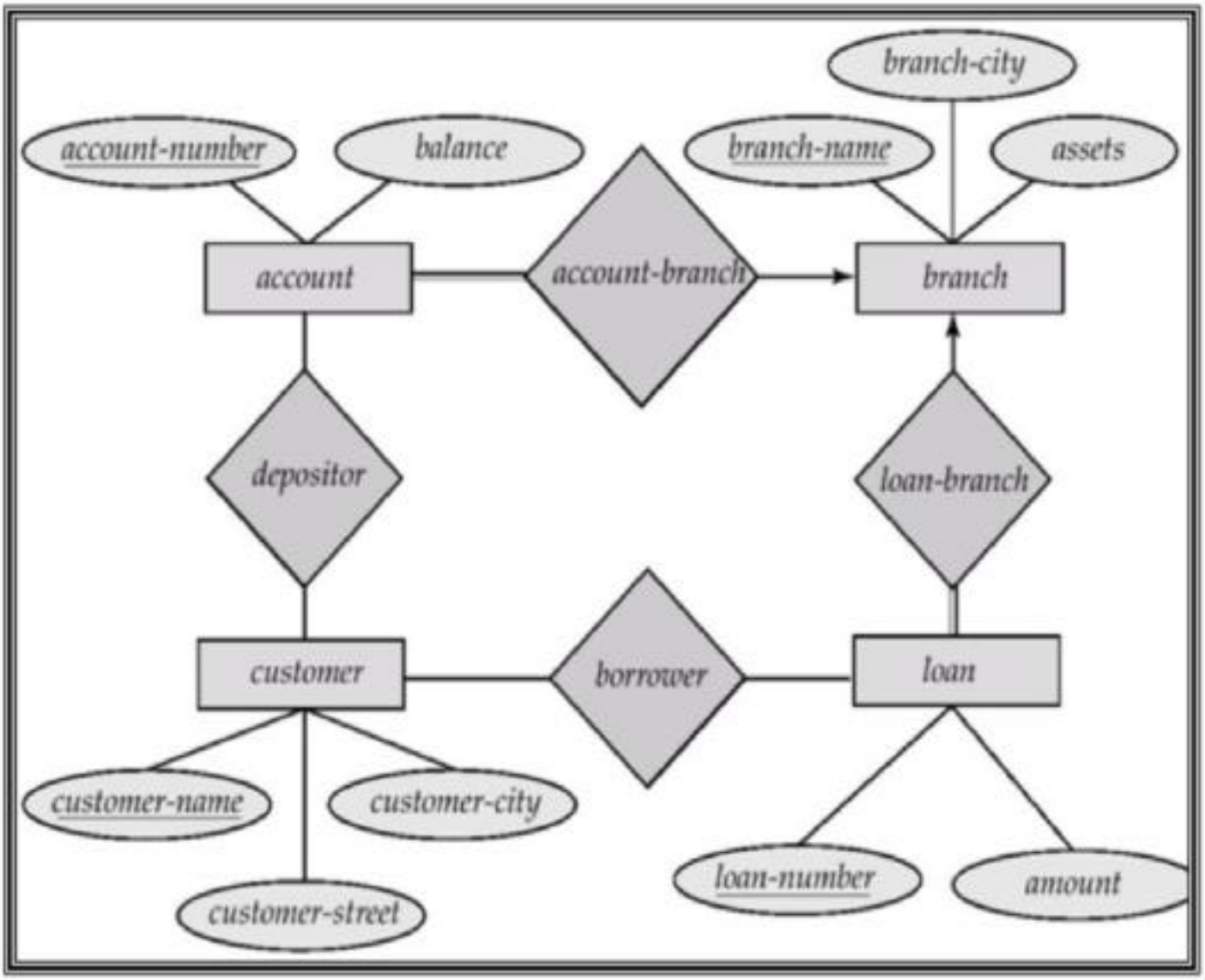
Derived Attribute

Derived attributes are attributes whose value can be calculated from related attribute values. They are represented by dotted ellipse.

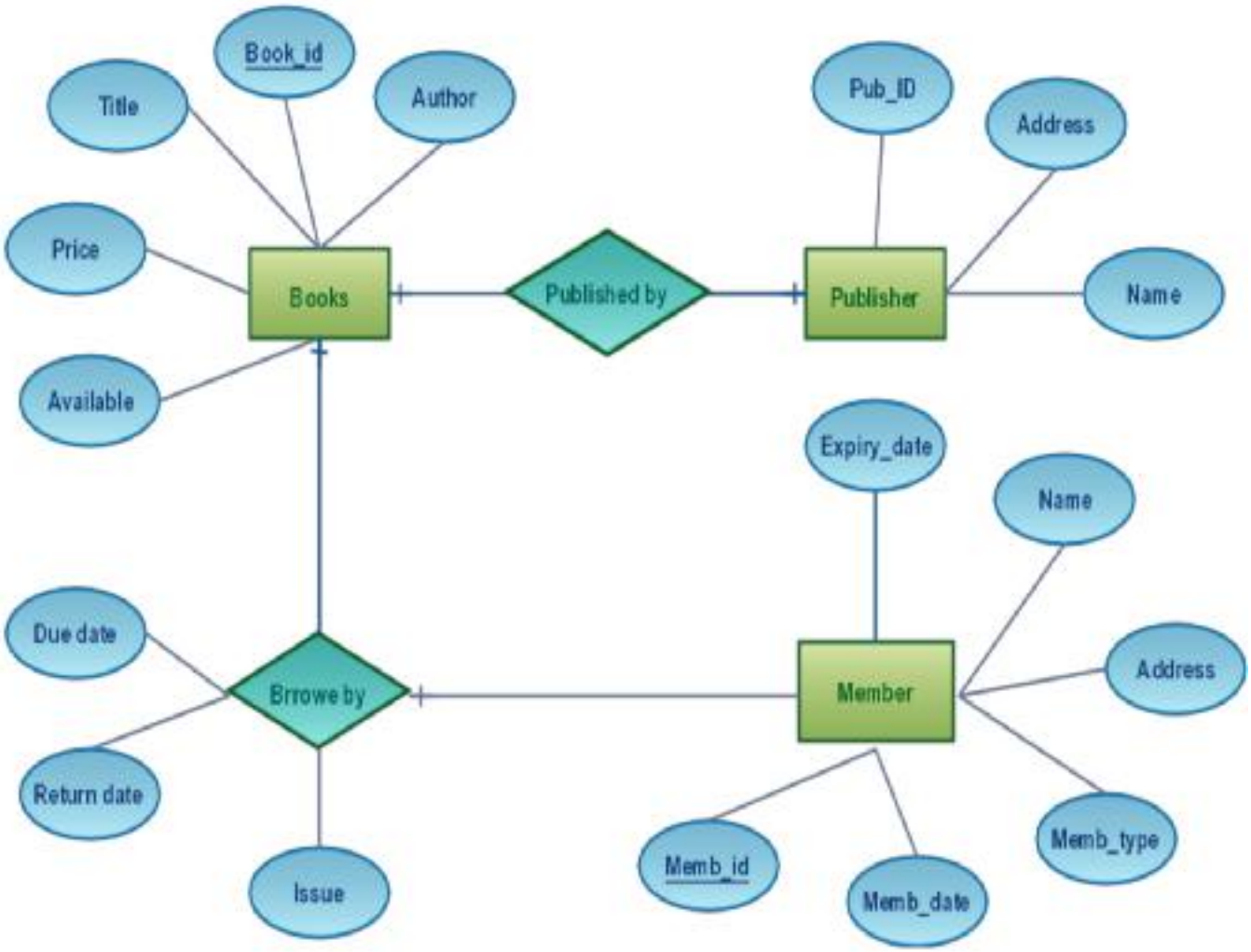
Example of E-R diagram



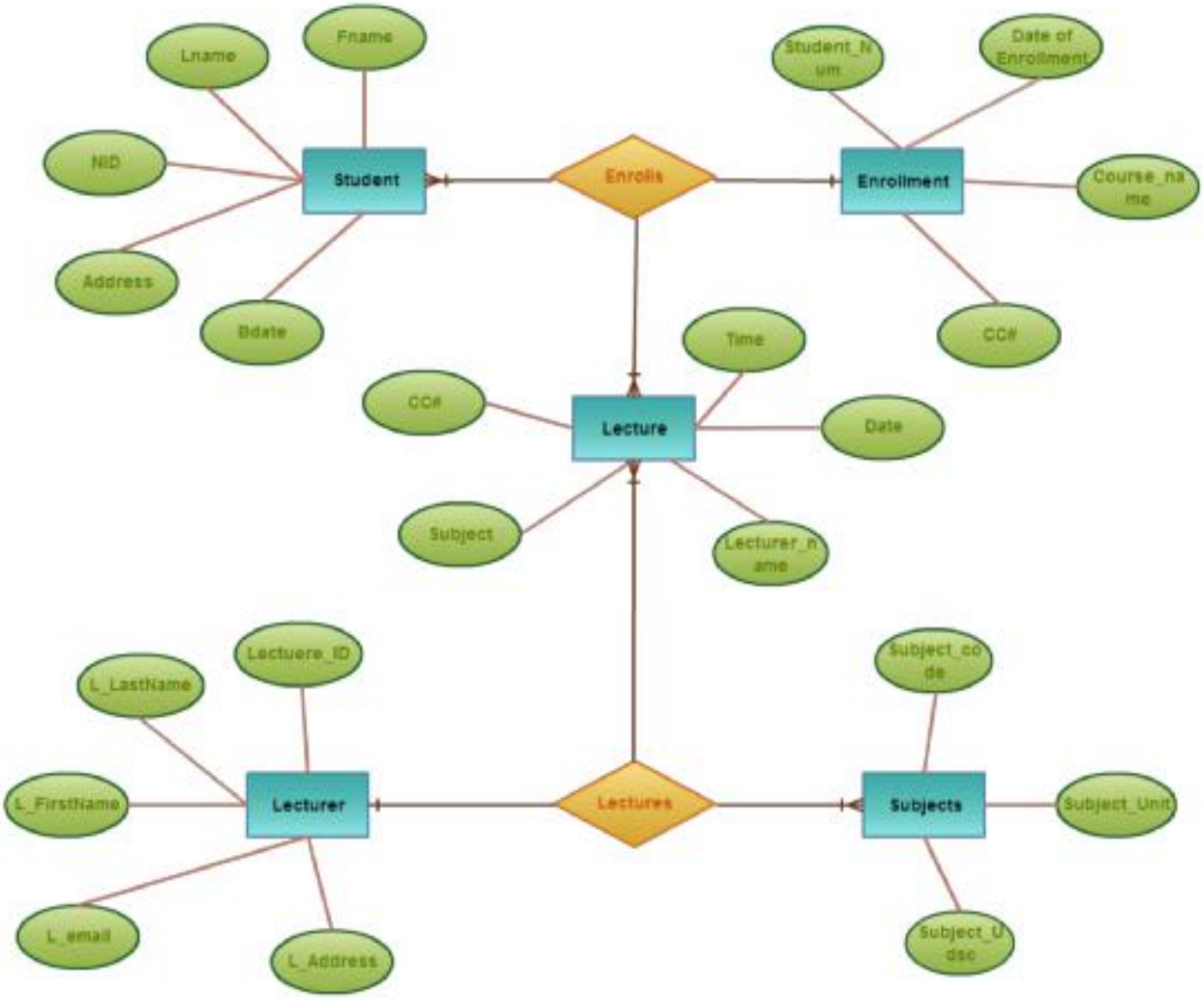
E-R Diagram for the Banking Enterprise



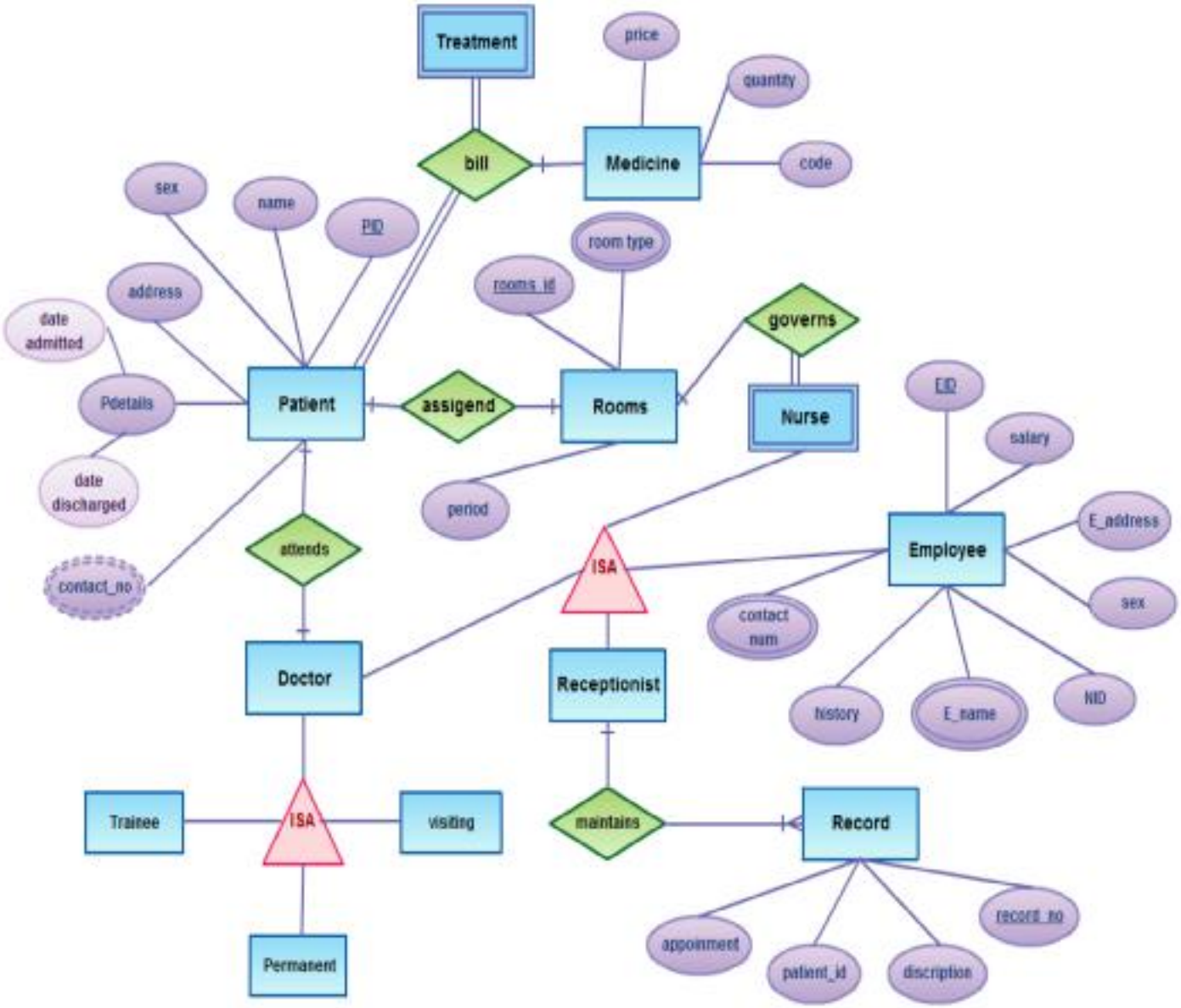
E-R Diagram for Library Management System

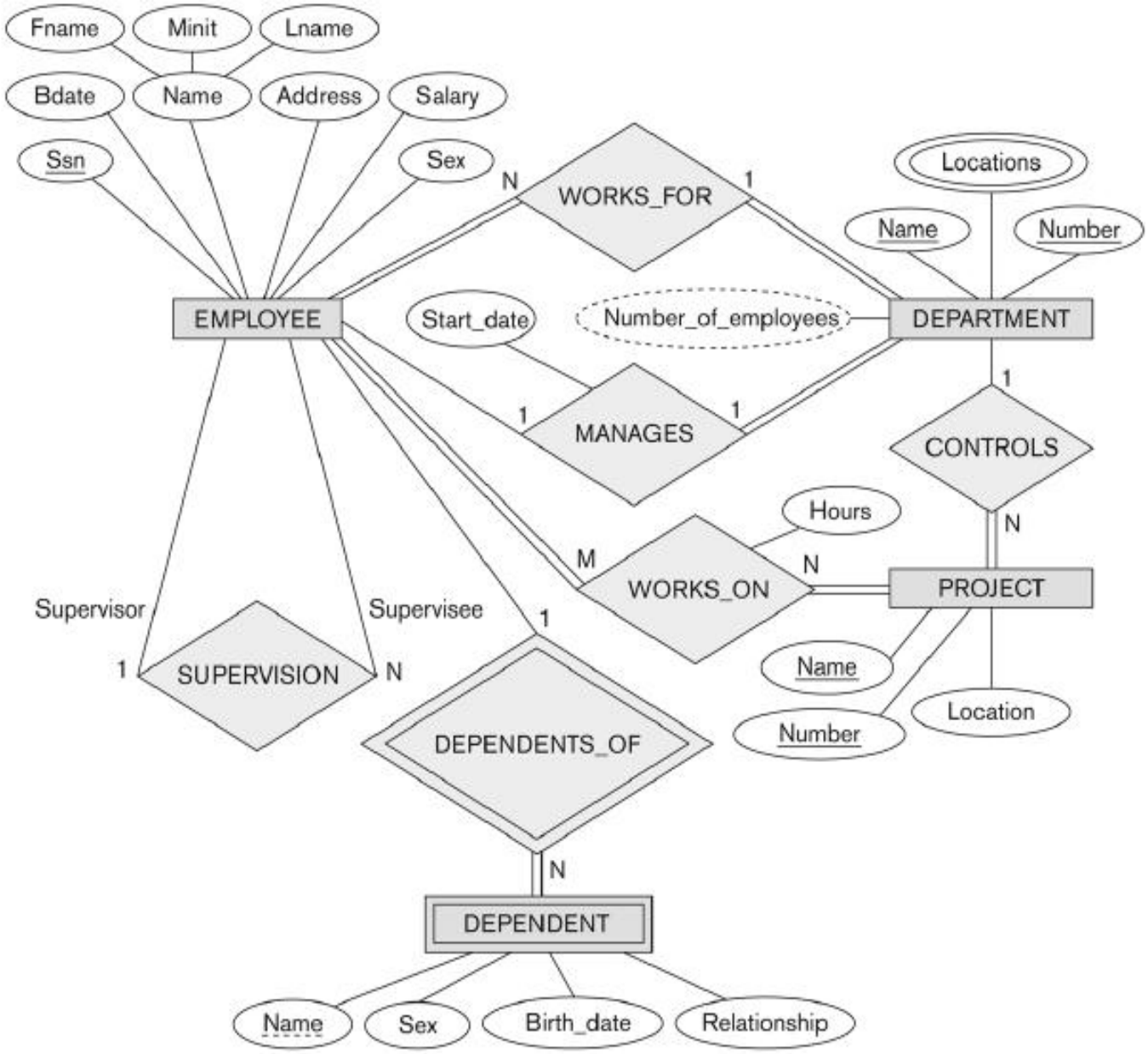


ER DIAGRAM FOR STUDENT ENROLLMENT SYSTEM



E-R Diagram for Hospital Management System





An ER schema diagram for the COMPANY database.