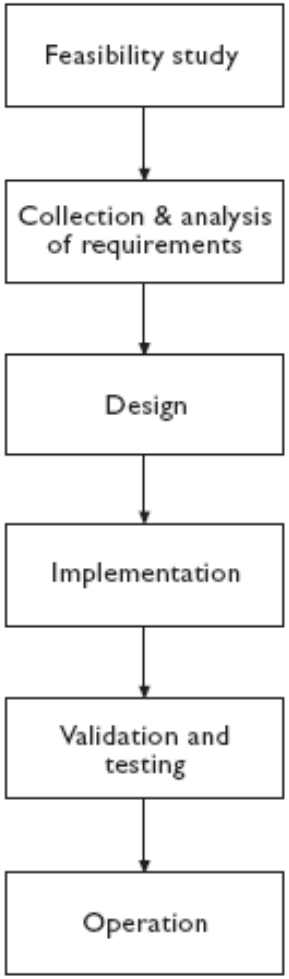
Database Management Systems

Database Design (1)

Topics

- Information Systems Life Cycle
- Data Base Design
  - Logical Design
  - Physical Design
- Entity Relationship (ER) Model
  - Entity
  - Relationship
  - Attributes
  - Cardinality of Relationships
- Physical Design

Information System Life Cycle



Database Design

- Database design is the process of producing a detailed data model of a database.
- Database design includes:
  - Determining data to be stored
  - Determining main entities and their attributes
  - Determining relationships between entities
  - Designing a suitable model to store them

Logical and Physical Design

- Logical design is about gathering requirements and converting those requirements into a model.
- Logical design includes the activities and the relationships between units.
- Physical design is the process of converting the logical model into database tables.

The Entity Relationship Model

- The **E-R** (entity-relationship) data model views the real world as a set of basic **objects** (**entities**) and **relationships** among these objects.

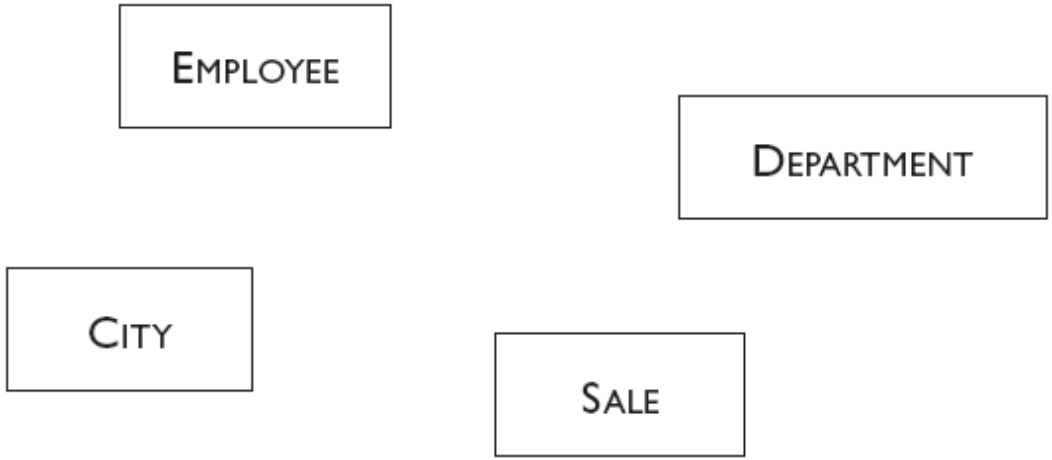
- E-R is used for logical design of a database

Entities

- An **entity** is an object that exists and is distinguishable from other objects. For instance, John Harris with S.S.N. 890-12-3456 is an entity, as he can be uniquely identified as one person in the universe.

- An **entity** may be a physical object such as a house or a car, or a concept such as a customer transaction, or order.

Graphical Model of Entities



Examples of Entities in the E-R Model

- CITY, DEPARTMENT, EMPLOYEE, PURCHASE and SALE are examples of entities in a commercial application
- STUDENT, COURSE, INSTUCTOR are example entities of university application
- BOOK, MAGAZINE, EMPLOYEE, CLIENT are example entities of library application

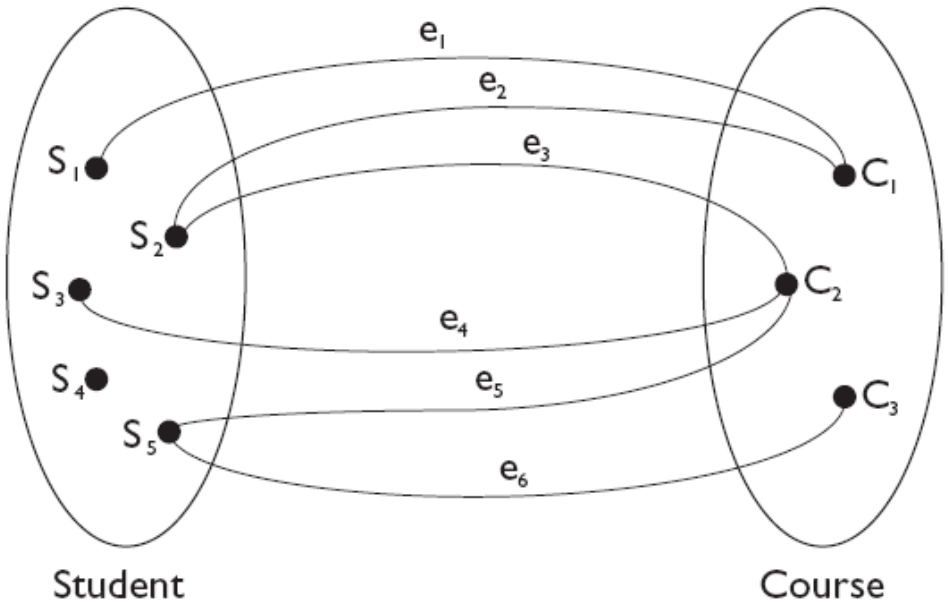
Relationships

- A **relationship** is an association between several entities.

- For example, consider the two entity sets *customer* and *account*. The relationship *CustAcct* defines the relation between customers and their accounts.

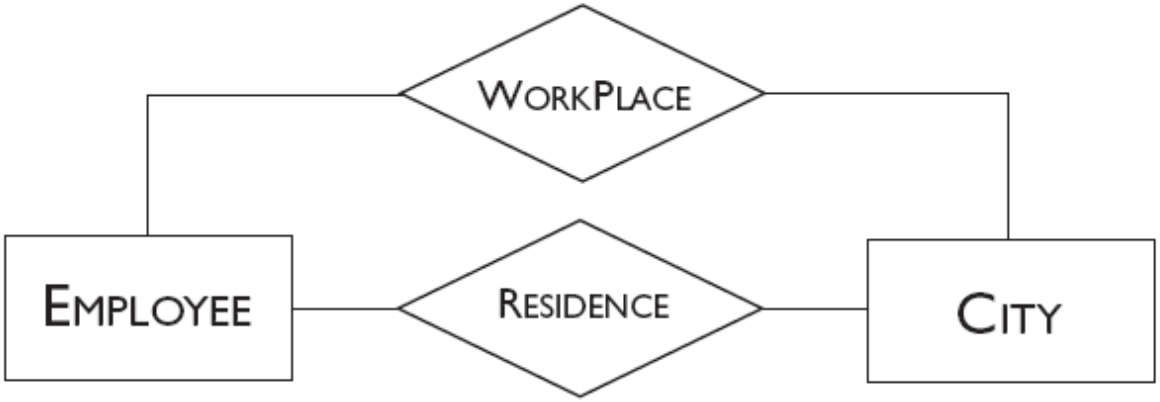
Example Relationships

- Each student takes some courses each semester. Therefore **TakesCourse** is the relationship between student and course entities



Graphical Model of Relationships

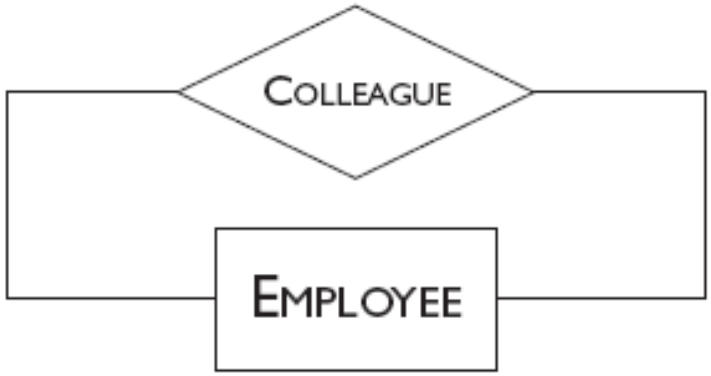
Example Entity-Relationships



Recursive Relationships

- If a relation connects an entity to itself, it is called recursive relationship.
- e.g. An Employee is a colleague of another employee
- A person can be the father of another person. Both father and child are from the same entity set.

Example for Recursive Relationships



Ternary Relationships

- If a relation connects three entities to each other, it is called a ternary relationship.

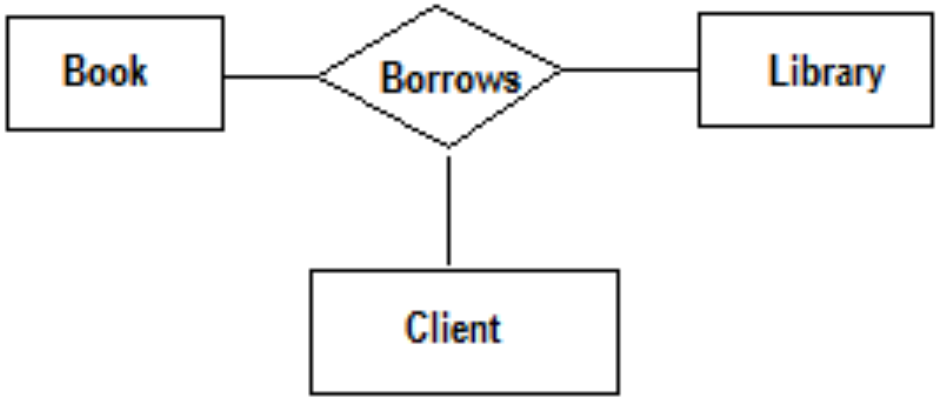
- e.g. A client borrows a book from a library

  entities are: Client, Book, Library

  Book belongs to **Library**

  Client borrows the **Book**

Ternary Relationship

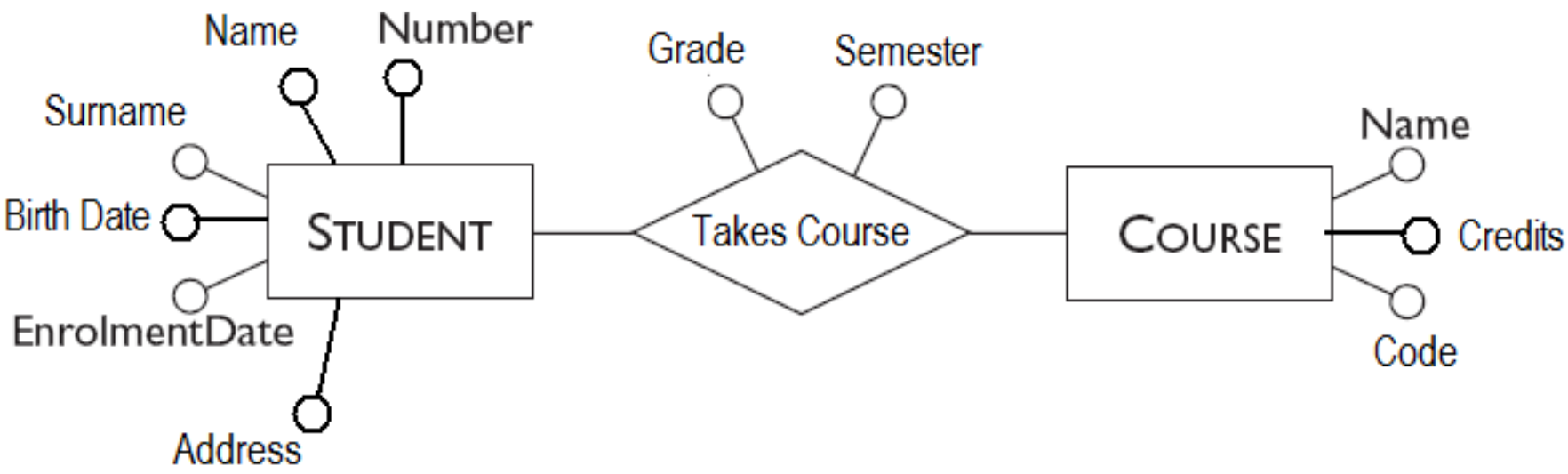


Attributes

- Attributes describe the properties of entities or relationships.
- e.g. Surname, Salary and Age are possible attributes of the EMPLOYEE entity.

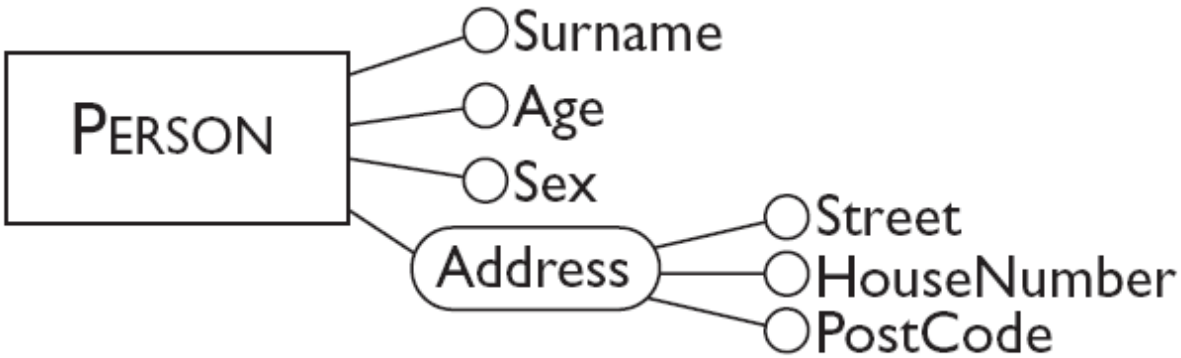
- Semester, Grade are attributes of relationship **Takes Course**

Example

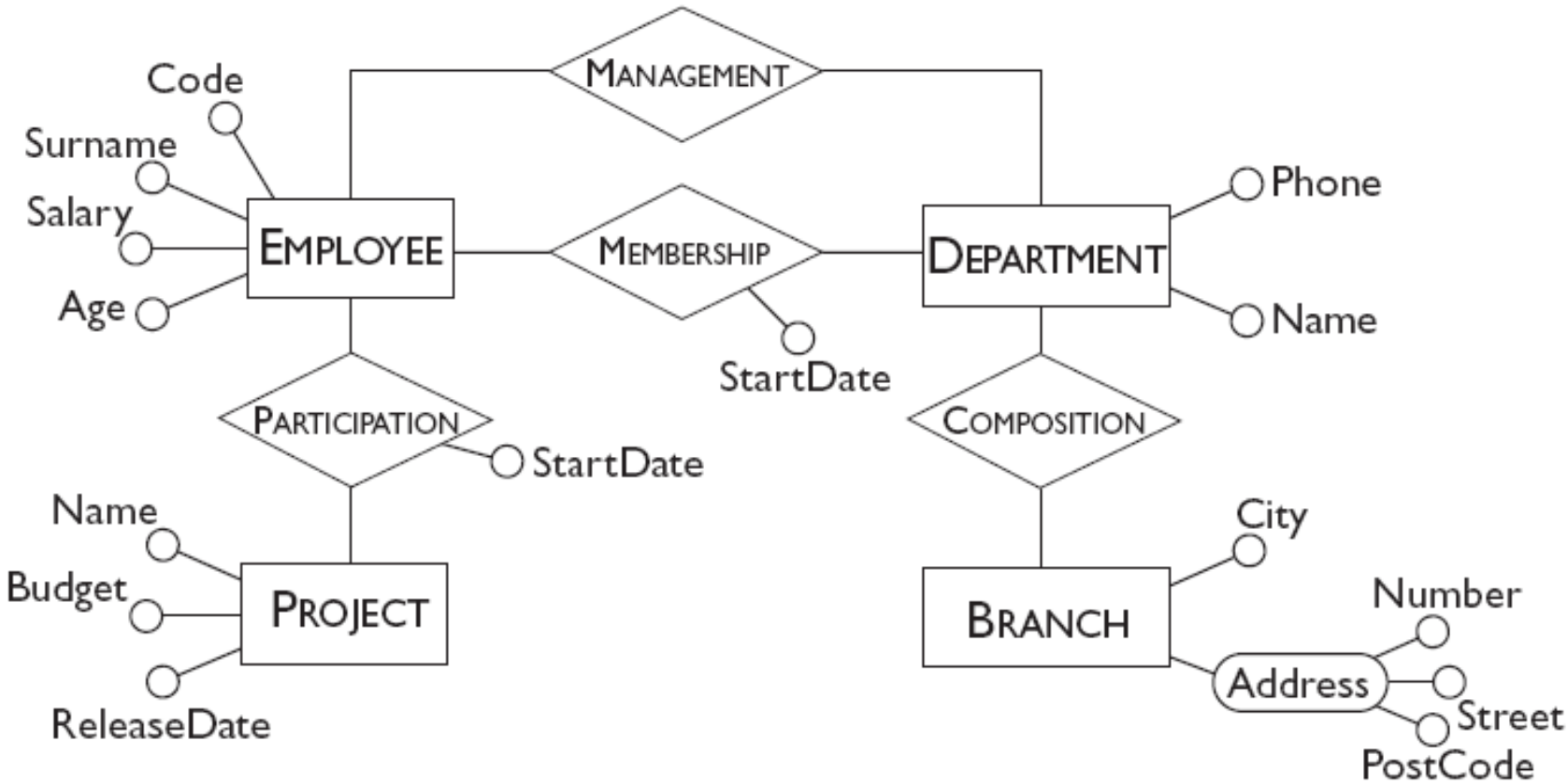

Name
Number
Surname
Birth Date
EnrolmentDate
Address
STUDENT
Grade
Semester
Takes Course
COURSE
Name
Credits
Code

Composite Attributes

- If an attribute contains many sub-parts it is called a composite attribute
- E.g. An address is made of Street, House Number, Post Code, …



An Entity-Relationship Schema



Cardinality of Relationship

- Cardinality shows how many entities participate in a relationship.
- Example 1. : many students are taking CS1356
- A student may also take many courses

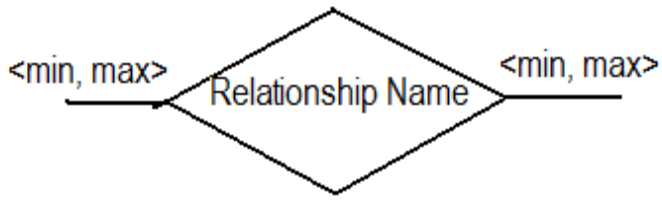
- Example 2. : each student number is given to only one student, each student has only one number

Relationship Cardinality Types

- A relationship can be:
  - One-to-one (Student ➔ Number)
  - One-to-many (Department ➔ Student)
  - Many-to-many (Student ➔ Course)

Optional Relationships

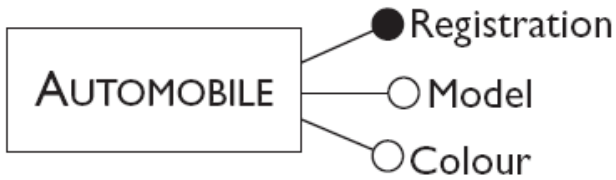
- Some members of an entity set may have no relationship with other entity members.
- e.g. A library client may borrow zero or many books but a book can be borrowed by only one client.
- Optional relationship is shown as follows:



Cardinality of Relationship Example

Identifier

- Identifiers are attributes that help us to uniquely identify an entity.
- Identifiers are given with filled in circles.



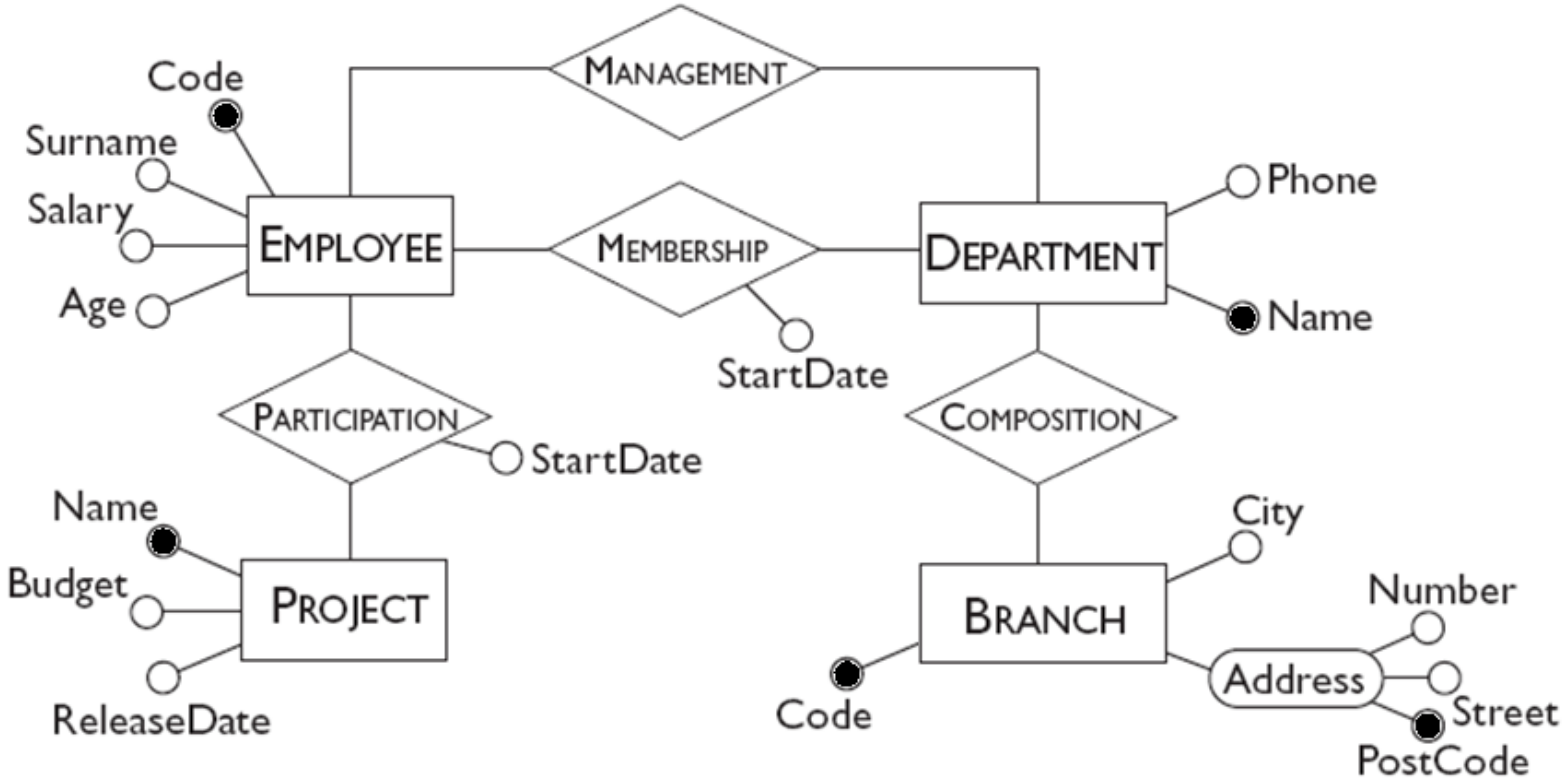
Graphical Symbols for E-R Constructs

Construct	Graphical representation
Entity	
Relationship	
Simple attribute	
Composite attribute	
Cardinality of a Relationship	(m_1,M_1) (m_2,M_2)
Cardinality of an attribute	(m,M)
Identifier	

Converting E-R Model to Relational Model

- Each entity in an E-R model is converted to a table in relational model.
- The attributes of the entity become fields of the table
- Primary key is given by the identifiers of the entity

Sample E-R Model



Example

```
CREATE Table Employee
(
    Code Integer PRIMARY KEY,
    Surname Char(30),
    Salary Integer,
    Age Integer
)
```

One-to-One Relationships

- One-to-one relationships are defined as attributes.
  - e.g. Book ➔ ISBN : Each book has only one ISBN and each ISBN represents only one book. ISBN should be an attribute for book.

- For restricting access to sensitive data, some attributes can be stored in a second table.
  - User and Password have a one-to-one relationship. Password should be an attribute for the user but we generally create a new table as <UsrID,Password>

One-to-Many Relationships (1)

- One-to-many relationships are defined as foreign keys.
  - e.g. An employee is a member of one department

    A department has many members

  In employee table define attribute ***dept*** to show the department of each employee. ***dept*** is a foreign key referring to Department.

One-to-Many Relationships (2)

- If relation has attributes then we have to define it as a table.
  - e.g. The relationship ***Membership*** between ***Employee*** and ***Department*** has "*start date*" attribute.
  - The relationship which is defined as a table can store the history of a relationship.
  - The primary keys of both entities and the attributes of the relationship are added to the table

Example

```
Create Table Membership
(
    EmpCode Number references Employee(code),
    DeptName  char(20) references department(name),
    startDate  date,
    position   char(30),
    PRIMAR KEY ( EmpCode, DeptName, startDate)
)
```

Many-to-Many Relationships

- All many to many relationships are defined as tables.
  - e.g. Student ➔ Course
  - A student takes many courses and a course is taken by many students
  - The primary key of both entities are added to the table
  - <StudentID, CourseCode, Year-Semester, Grade>

Recursive Relationships

- Recursive relationships are defined as a foreign key attribute.
  - e.g.

  Human < ID, Name, Surname, Birth Date, Sex, Father >

  Father is a foreign key to table Human (recursive)

Ternary Relationships

- Ternary relationships are defined as a table.
- The primary key of three tables are included in the table
- e.g. A client borrows a book from a library

  entities are: Client, Book, Library

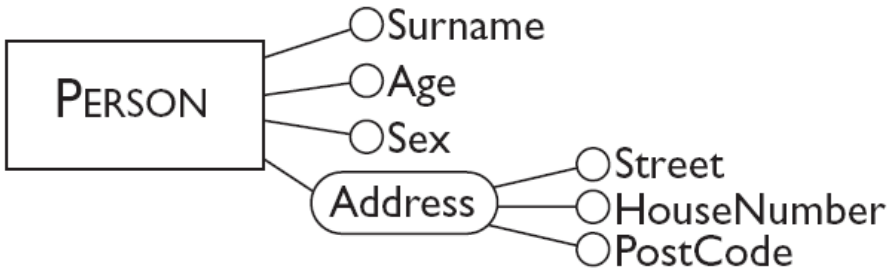
  Borrow < BookID, ClientID, LibID, Date,..>

Composite Attributes

- Composite attributes are defined as a table
- The primary key of the composite attribute table is added to the table using it as foreign key.
  - e.g. Address<Street, HouseNumber, PostCode>

    PostCode is **primary key**
  - Person <Surname, Age, Sex, PostCode >

    PostCode is **foreign key**



Summary

- Database design is the process of modeling data in an information system.
- E-R model is a method for database design.
- E-R model is based on finding entities in a system and the relationships between them.
- E-R model is converted to relational tables in physical design.
- Entities and many-to-many relationships, and one-to-many relationships with attribute in E-R model are converted to tables.

Questions?