Database Lab

CSE 3104

Session 05

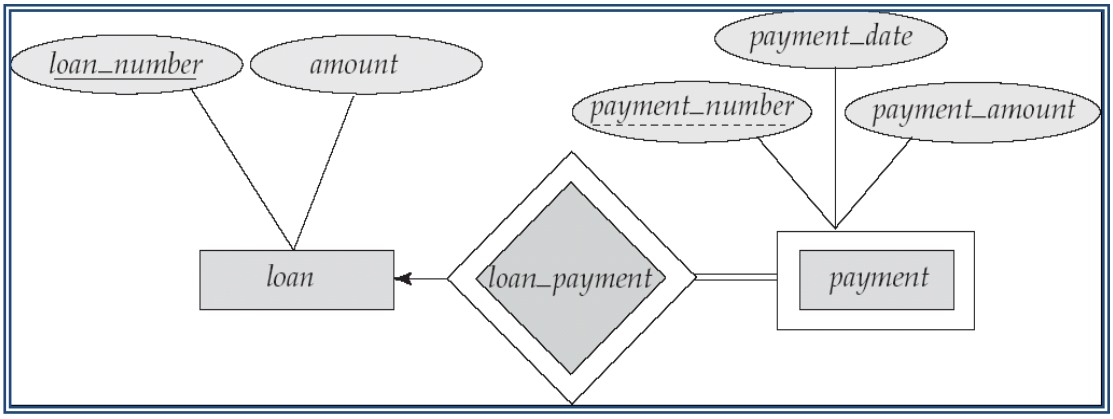
4 ERD (Cont.)

4.4 Weak Entity

- An entity set that does not have a primary key is referred to as a **weak entity set**.
- The **discriminator** (*or partial key)* of a weak entity set is the set of attributes that distinguishes among all the entities of a weak entity set.
- The primary key of a weak entity set is formed by the primary key of the strong entity set on which the weak entity set is existence dependent, plus the weak entity set's discriminator.

Representation of Weak Entity in ER Diagram:

- We depict a weak entity set by double rectangles.
- We underline the discriminator of a weak entity set with a dashed line.
- payment_number – discriminator of the *payment* entity set
- Primary key for *payment* – (*loan_number, payment_number*)

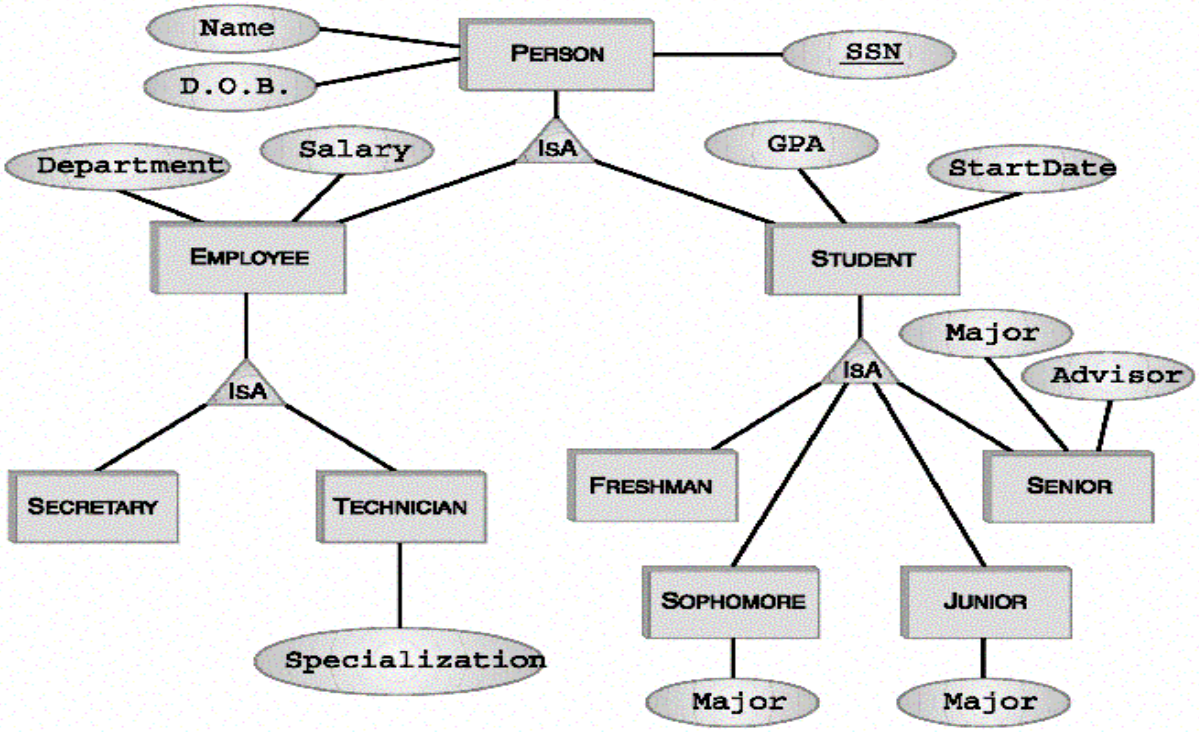


4.5 Recursive Relationship

A **relationship between two entities of a similar entity** type is called recursive relationship. Example: Let us suppose that we have an **employee** table. A manager supervises a subordinate. Every employee can have a supervisor. One employee may be the boss of more than one employee.

4.6 Extended E-R Feature:

- **Attribute inheritance –** a lower-level entity set inherits all the attributes and relationship participation of the higher-level entity set to which it is linked.
- Top-down design process: we designate sub groupings within an entity set that are distinctive from other entities in the set
- Depicted by a *triangle* component labeled **ISA** (E.g. *employee* "is a" *person*).



5 Converting ERD to Relational Model

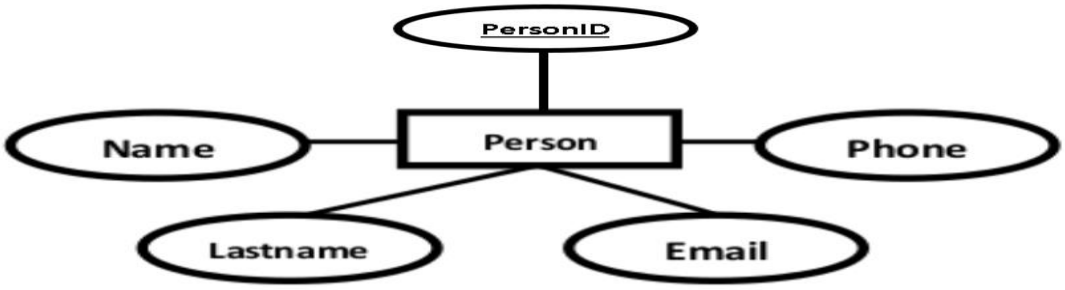
The ER Model is intended as a description of real-world entities. Although it is constructed in such a way as to allow easy translation to the relational schema model, this is not an entirely trivial process. The ER diagram represents the conceptual level of database design meanwhile the relational schema is the logical level for the database design. We will be following the simple rules:

5.1 Entities and Simple Attributes:

An entity type within ER diagram is turned into a table. You may preferably keep the same name for the entity or give it a sensible name but avoid DBMS reserved words as well as avoid the use of special characters.
Each attribute turns into a column (attribute) in the table. The key attribute of the entity is the primary key of the table which is usually underlined. It can be composite if required but can never be null.

Note: It is highly recommended that every table should start with its primary key attribute conventionally named as TablenameID.

Taking the following simple ER diagram:

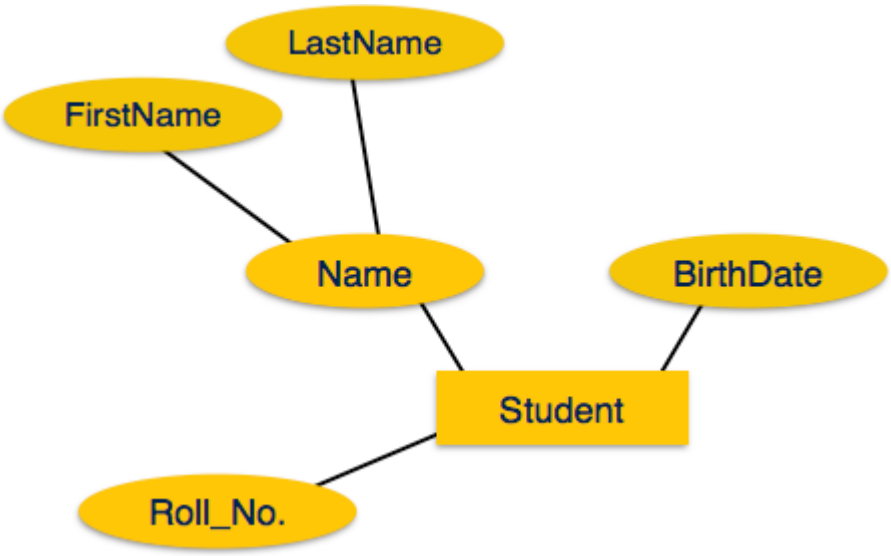


The initial relational schema is expressed in the following format writing the table names with the attributes list inside a parentheses as shown below for

Person(PersonID, Name , Email, Lastname, Phone)

5.2 Composite Attribute:

If there is composite attribute, omit the composite attribute and add the component attributes as columns in the table.



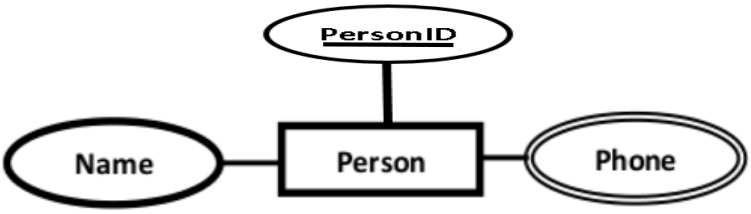
Corresponding Schema:

Student(Roll_NO , FirstName, Lastname, BirthDate)

5.3 Multivalued Attribute:

If you have a multi-valued attribute, take the attribute and turn it into a new entity or table of its own. Then make a 1:N relationship between the new entity and the existing one. In simple words:

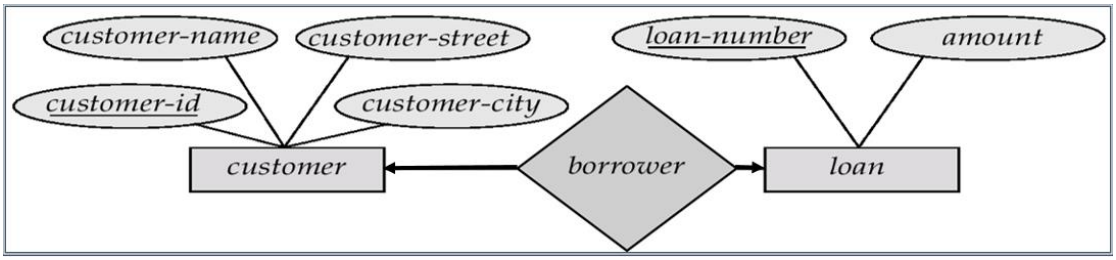
1. Create a table for the attribute.
2. Add the primary (id) column of the parent entity as a foreign key within the new table as shown below:



Person(PersonID , Name)
Phones (phoneid , ***personid***, phone)

5.4 One-to-One Relationship:

To keep it simple and even for better performances at data retrieval, I would personally recommend using attributes to represent such relationship. For instance, let us consider the case where the Customer has or optionally has one loan. You can place the primary key of the **loan** within the table of the **customer** which we call in this case Foreign key as shown below.



customer(customer-id , customer-name, customer-street, customer-city , ***loan-number***)
loan (loan-number , amount)

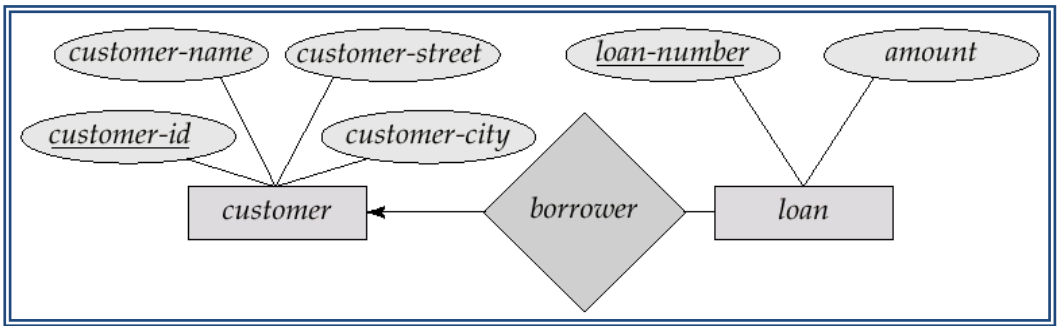
Or vice versa to put the **customer-id** as a foreign key within the loan table as shown below:

customer(customer-id , customer-name, customer-street, customer-city)
loan (loan-number , amount , ***customer-id***)

5.5 One-to-Many Relationship:

This is the tricky part!

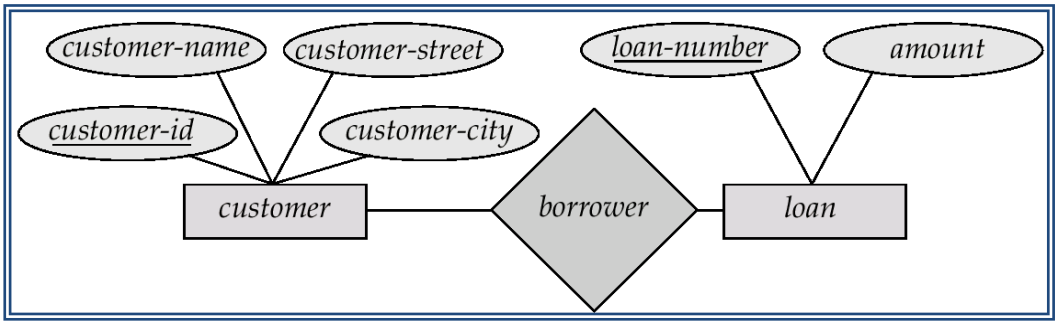
For simplicity, use attributes in the same way as one-to-one relationship but we have only one choice as opposed to two choices. For instance, the Customer can have a **loan** from zero to many, but a **loan** can have only one **customer**. To represent such relationship the **customer-id** as the Parent node must be placed within the Child table as a foreign key but **not the other way around as shown next.**



It should convert to :
customer(customer-id , customer-name, customer-street, customer-city)
loan (loan-number , amount , ***customer-id***)

5.6 Many-to-Many Relationship:

We normally use tables to express such type of relationship. This is the same for N – ary relationship of ER diagrams. For instance, The Customer can take or borrow many loans. Also, a loan can be owned by many customers. To express this relationship within a relational schema we use a separate table as shown below:

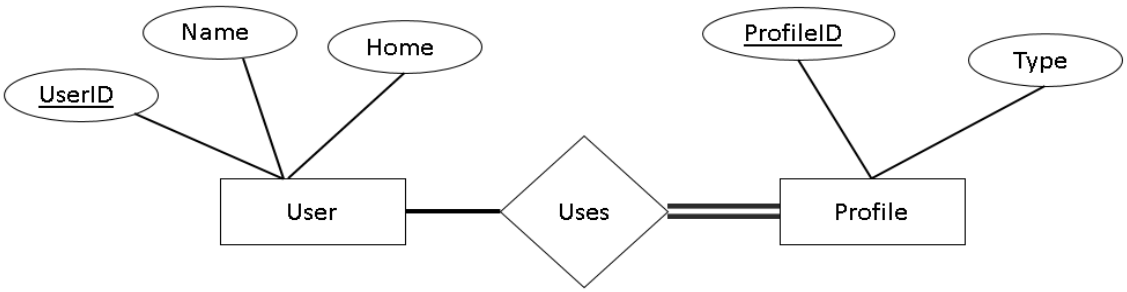


It should convert into:

customer(customer-id , customer-name, customer-street, customer-city)
loan (loan-number , amount)
borrower (borrower-id , **customer-id , loan-number**)

5.7 Total Participation:

If there is total participation from any entity to other entity, include the primary key attribute of the other entity as foreign key in the table which is taking part in total participation.
For example, Profile entity has total participation to the User entity. So while creating Profile table, we will include one foreign key which comes from the primary key (**UserID**) of User entity.

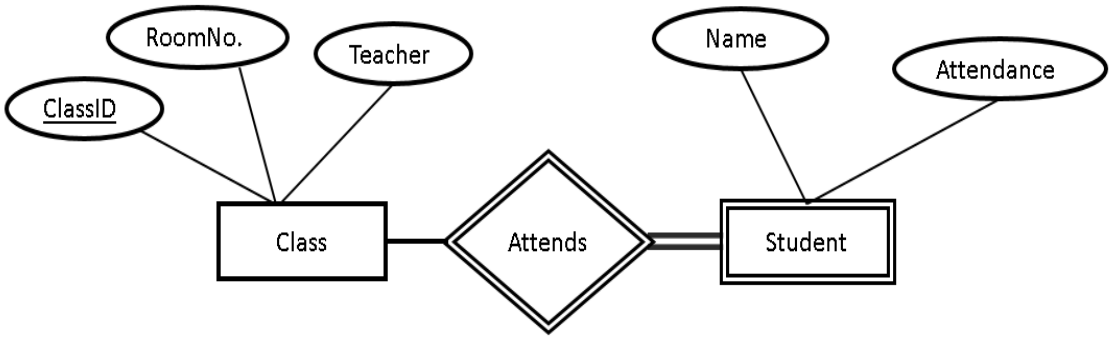


It should convert into:

User(UserID , Name, Home)
Profile (ProfileID , Type, **UserID**)

5.8 Weak Entity:

If one of the entity is weak entity, include the primary key attribute of the strong entity as foreign key in the table of the weak entity.

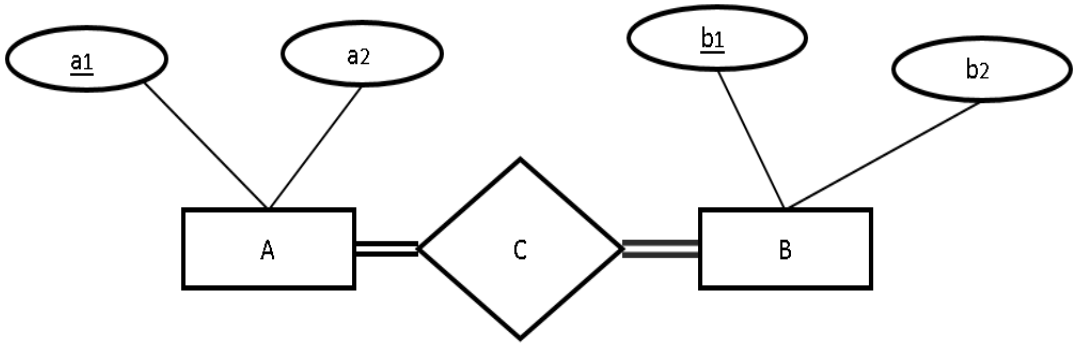


It should convert into:

Class(ClassID , RoomNo., Teacher)
Student (Name, Attendance, **ClassID**)

5.9 Total Participation from Both Sides:

If there is total participation from both the entities while creating a relationship, just create a single table containing all the attributes from both of the entities.



It should convert into:

ABC(a1, a2, b1, b2)