

Keys

Meanwhile, it is also obvious that some of these combinations may have redundant attributes.

Therefore, in theory, we systematically classify these combinations of attributes using three types of Keys: superkey, candidate key, and primary key.

Keys

(1) **Superkey**: is a set of attributes whose value can uniquely identify an entity in the entity set.

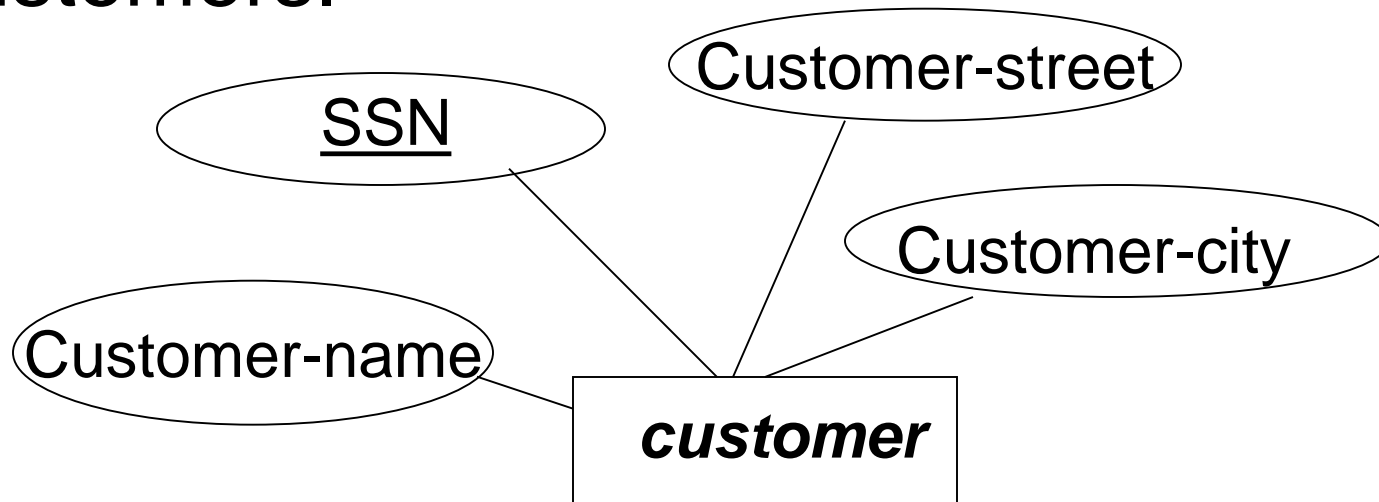
A superkey contains one or more attributes.

Example:

The combination of “SSN” and “Name” is a superkey of the following entity set ***customer***.

Because:

The value of attributes “SSN” and “Name”, such as 558-36-1234 and Susan, can uniquely identify that particular customer in *customer* entity set, which is the pool of all customers.



Notationally, we write,
Superkey: (SSN, name)

Other superkeys for the entity set *customer*:

Superkey: (SSN)

Superkey: (SSN, Street)

Superkey: (SSN, City)

Superkey: (SSN, Name, Street)

Superkey: (SSN, Name, City)

Superkey: (SSN, Street, City)

Superkey: (SSN, Name, Street, City)

Superkey is the broadest definition of unique identifiers of an entity in an entity set.

We are unsurprisingly very interested in the most economical combination(s) of attributes that can uniquely identify any particular entity.

Therefore, we introduce **Candidate Key** next.

(2) **Candidate key**: is a set of one or more attributes whose value can uniquely identify an entity in the entity set, and any attribute in the candidate key cannot be omitted without destroying the uniqueness property of the candidate key. (It is minimal superkey).

Example:

(*SSN*, *Name*) is NOT a candidate key, because taking out “*name*” still leaves “*SSN*” which can uniquely identify an entity. “*SSN*” is a candidate key of ***customer***.

Candidate key could have more than one attributes.

Also, while most entity sets have only one candidate key, some entity sets could have more than one candidate key.

Example: Both “SSN” and “License #” are candidate keys of *Driver* entity set.

In building a database in a database software, the software will only allow to use one candidate key to be the unique identifier of an entity for an entity set.

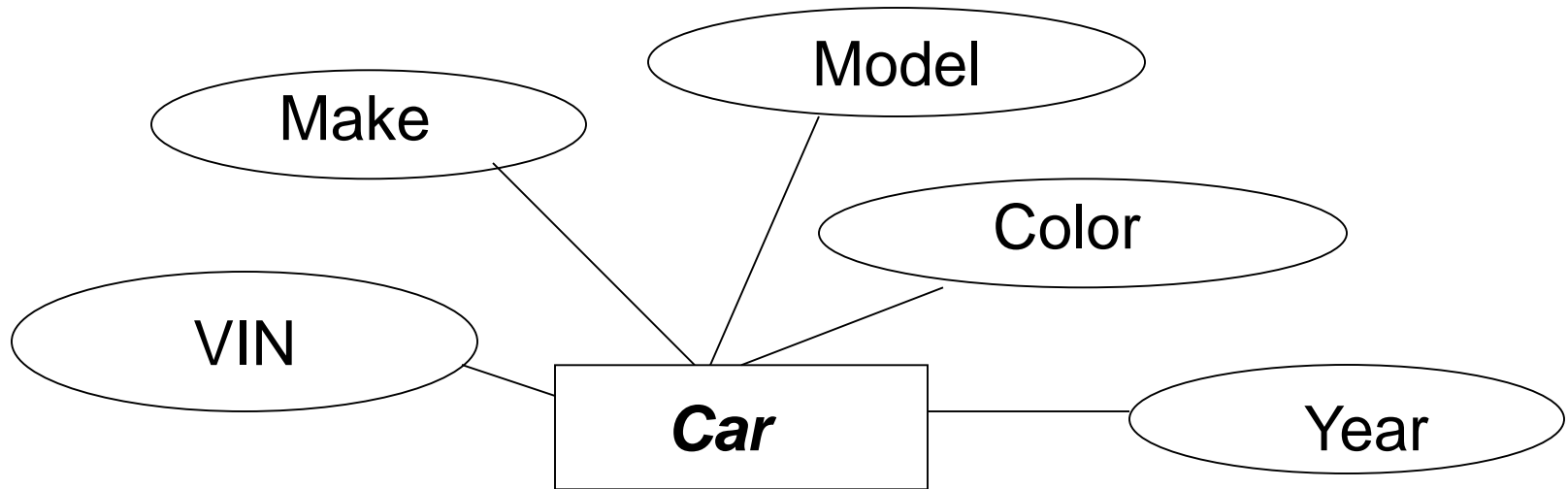
(3) **Primary Key**: is the candidate key that is chosen by the database designer as the unique identifier of an entity. The database designer chooses only one candidate key as the primary key in building the system.

Example: “SSN” and “License #” are both candidate keys of *Driver* entity set. The database designer can choose either one as the primary key.

In an E-R diagram, we usually underline the primary key attribute(s).

Overall, superkey is the broadest unique identifier; candidate key is a subset of superkey; and primary key is a subset of candidate key.

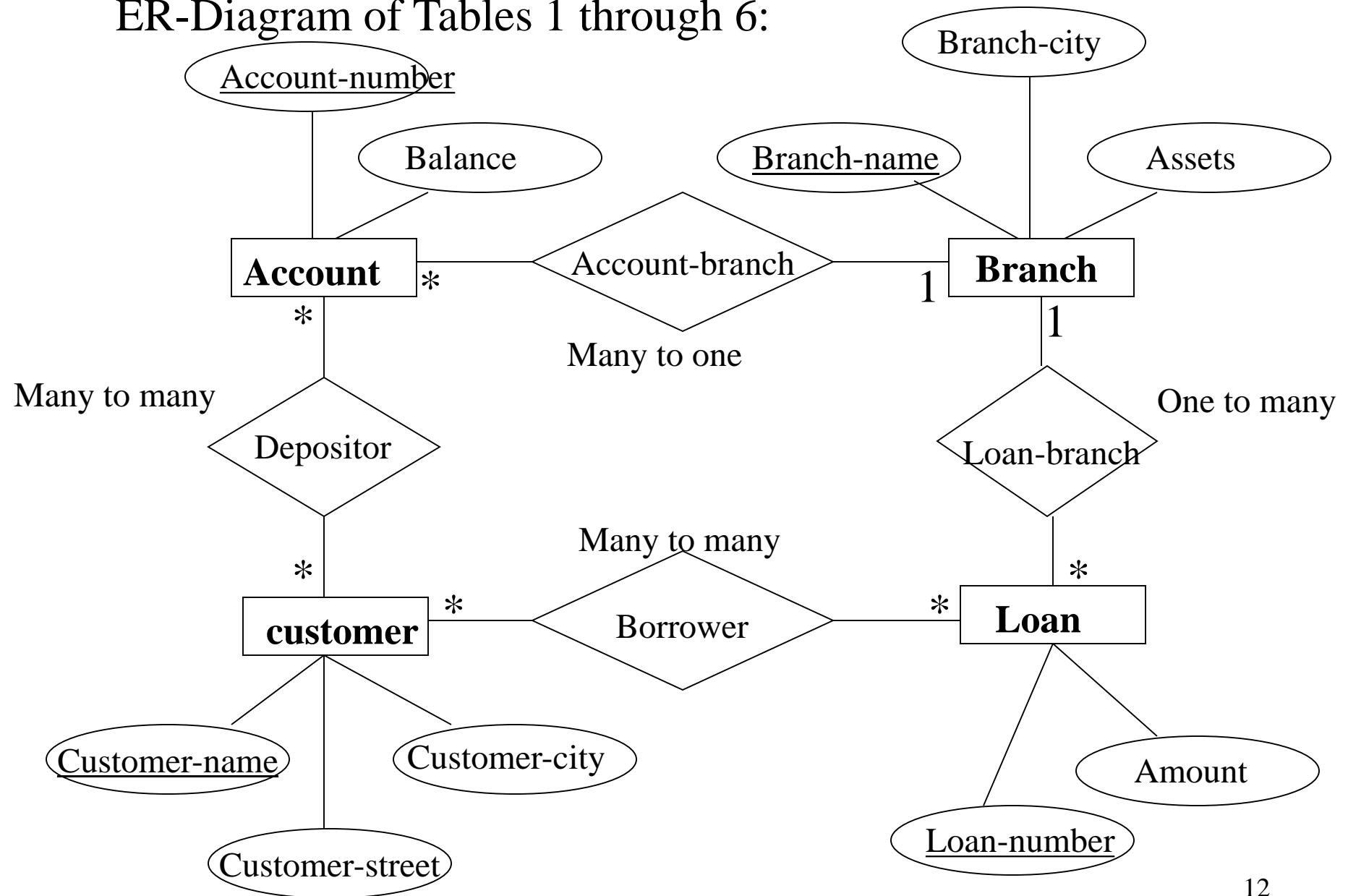
For example, think about the mental process of finding the primary key of the following entity set.



In-class exercise:

Construct an E-R diagram for a university's registrar's office. The office maintains data about each class, including the instructor, the enrollment, and the time and place of the class meetings. For each student-class pair, a grade is recorded. Specify the mapping cardinalities between all related entity sets. (hint: there are three entity sets, instructor, student, and class)

ER-Diagram of Tables 1 through 6:



Creating Tables from E-R Diagram

Two Steps:

1. Each entity set has its own table.

Example: *account*: (account-number, balance)

branch: (branch-name, branch-city, assets)

Note: Primary key attribute(s) is (are) underlined.

Creating Tables from E-R Diagram (cont'd)

Two Steps:

2. Each relationship should be accounted for.

How to account for the relationships? →

- **One to One relationship**

We can do either:

(1) put the primary key of one entity set into the table of the other side.

Exception

If one entity set occurs prior to the other entity set, you can only put the primary key of the first entity set into the table of the second entity set, and NOT the other way around.

Or

(2) create a new table that contains the primary keys of both entity sets.

- **One to Many or Many to One.**

Put the primary key of the entity set on the “one” side into the table of the entity set on the “many” side.

Example: The relation of *account-branch* is accomplished by putting the primary key of *branch* into the table of *account*, and the relationship does not require a separate table. Now the *account* table becomes:

account: (account-number, balance, branch-name*)

“branch-name” is called **Foreign Key** in the *account* table and it is used to link the *account* table and the *branch* table.

Foreign key has an * beside it.

The same thing happens to the “loan-branch” relationship.

- **One to many or Many to one relationships (Exception).**

If the many entity set occurs first, you **cannot** put the primary key from the one side into the many side.

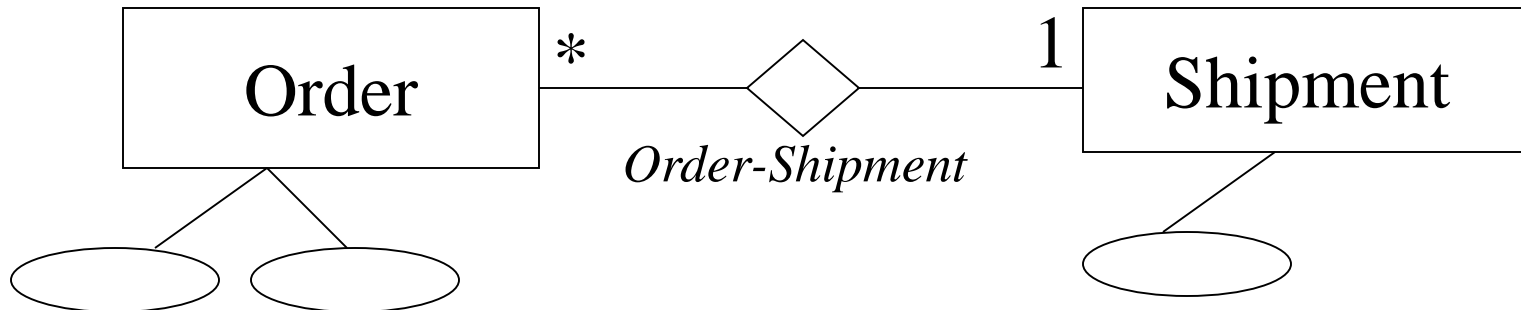
For example, the following model describes the relationship between customer order and shipment. It is a many-to-one relationship, meaning that each customer order can be delivered once, while each delivery can have several customer orders.

In this case, the following tables do **not** work.

Shipment: (Shipment-number, ...)

Order: (Order-number, Shipment-number*, ...)

**This field has to be left blank
Until shipment occurred
This is wrong.**



- In this case, we create a new table including the primary keys on both sides.

That is,

Shipment: (Shipment-number, ...)

Order: (Order-number, ...)

Order-Shipment: (Order-number*, Shipment-number*)

- **Many to many relationship**

Requires a separate table (**bridge**) to represent the relationship. The primary key of this bridge table consists of the primary keys of the entity sets on both sides of the relationship.

Example: The relationship “depositor” is implemented by table:

depositor: (customer-name*, account-number*)

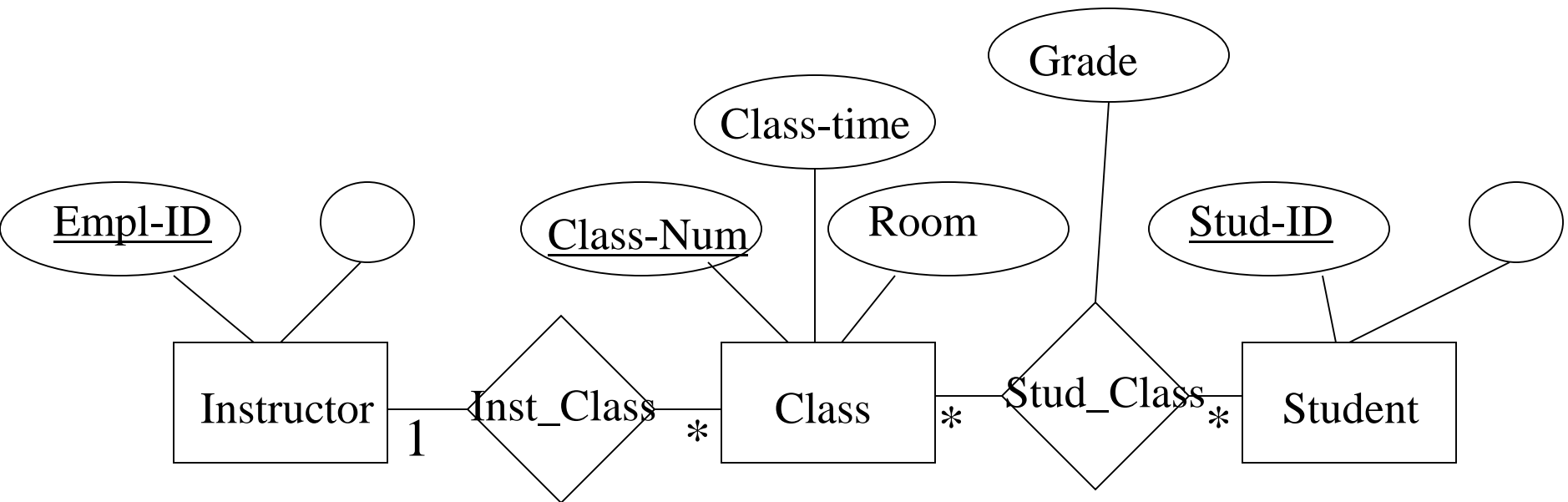
The same happens to the “borrower” relationship.

Revisit the Example of Registrar's Office

Construct an E-R diagram for a university's registrar's office. The office maintains data about each class, including the instructor, the enrollment, and the time and place of the class meetings. For each student-class pair, a grade is recorded. Specify the mapping cardinalities between all related entity sets.

(hint: there are three entity sets, instructor, student, and class)

E-R Diagram and Mapping Cardinalities:



Note: A relationship sometimes has its own attribute(s). In such case, the relationship has to be a separate table.

Suggested Structure of Tables:

Instructor:

(Empl-ID, Empl-name, Empl-address, Empl-phone)

Class:

(Class-Num, Class-time, Room, **Empl-ID***)

Student:

(Stud-ID, Stud-name, Stud-address)

Stud_Class:

(**Class-Num***, **Stud-ID***, Grade)

Homework (no need to turn in):

Create the structure of tables for the E-R diagram of the customer database in the bank.