

EXERCISE: CAR OWNERSHIP AND LOANS

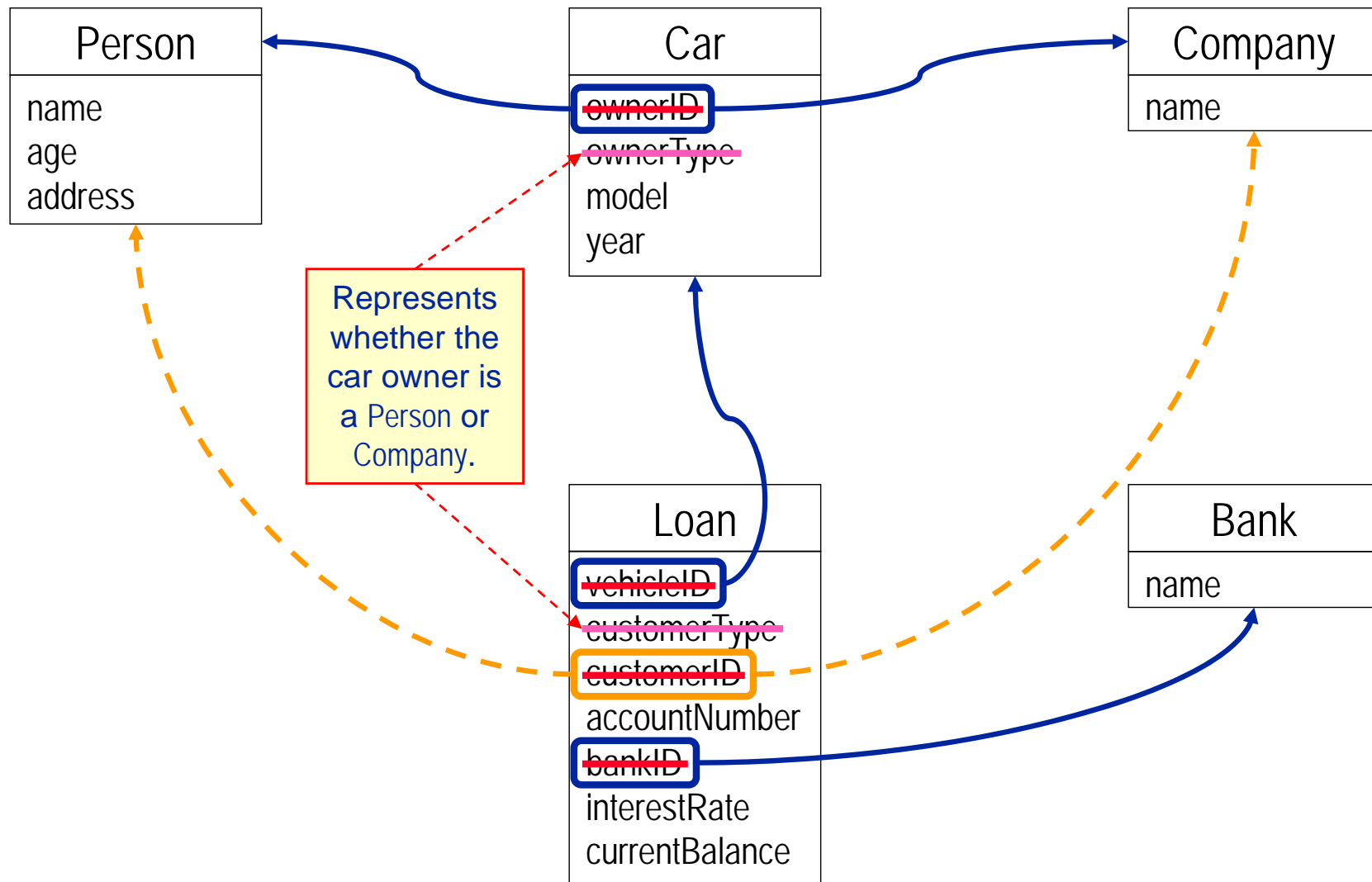
The classes shown below are required to represent information about car ownership and car loans. However, they have some attributes that are **internal object identifiers (OIDs)** that are being used to represent relationships and that *should not appear* in a class diagram. All such attributes conveniently have names ending in ID. 😊

Persons or companies may own cars. The car owner ID represents either the person or company that owns the car. A car may have only one owner (person or company). A car may have no loan or multiple loans. A bank provides a loan to a person or a company for the purchase of a car. Only the car owner may obtain a loan on the car. The car owner type and the loan customer type indicate whether the car owner/loan holder is a person or company.

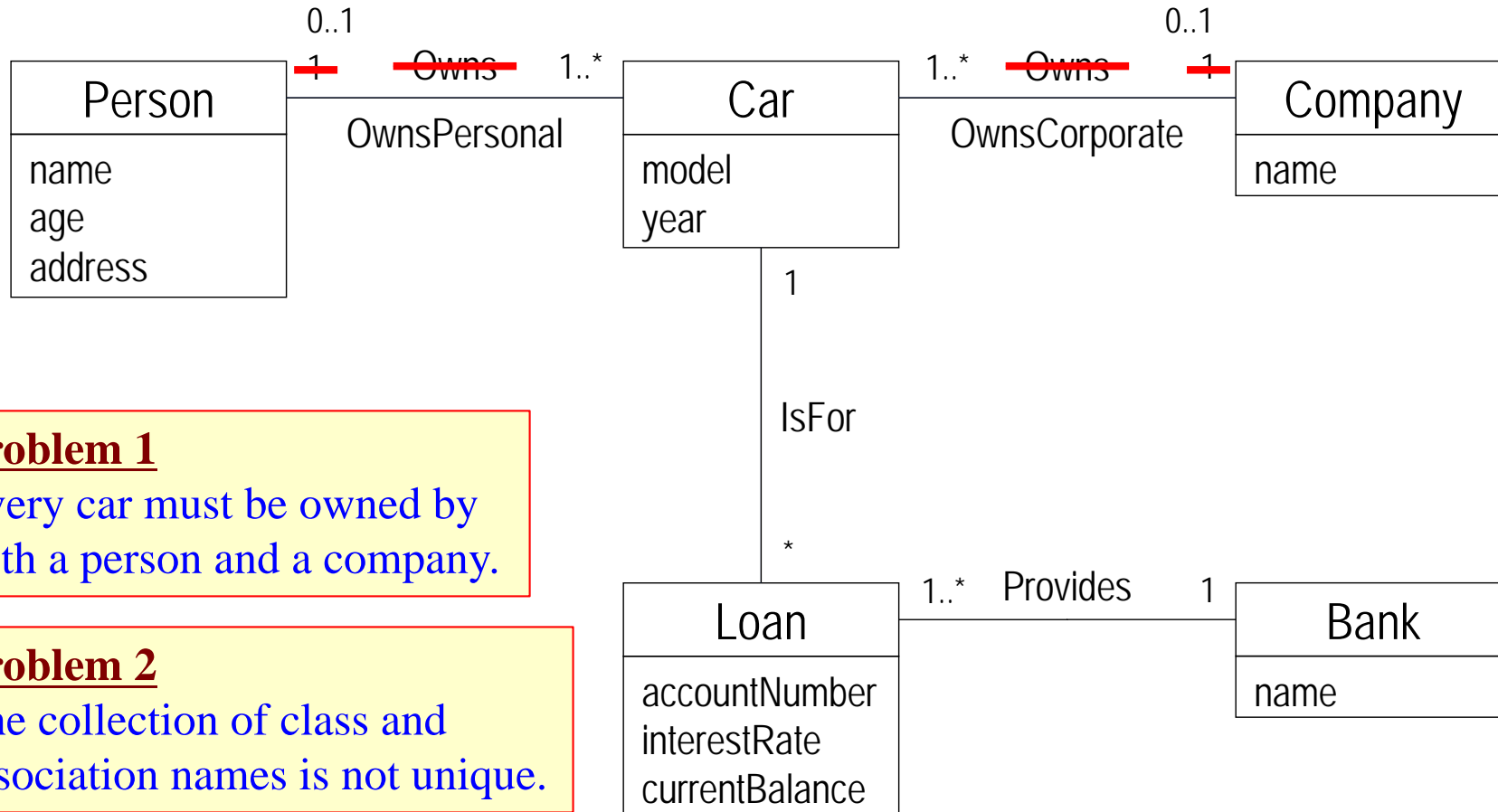
On the class diagram below:

- 1. Replace all OIDs with associations.**
- 2. Indicate the most likely multiplicities for all associations.**
- 3. Cross out any unnecessary attributes.**

EXERCISE: CAR OWNERSHIP AND LOANS — SOLUTION



EXERCISE: CAR OWNERSHIP AND LOANS — SOLUTION



Problem 1

Every car must be owned by both a person and a company.

Problem 2

The collection of class and association names is not unique.

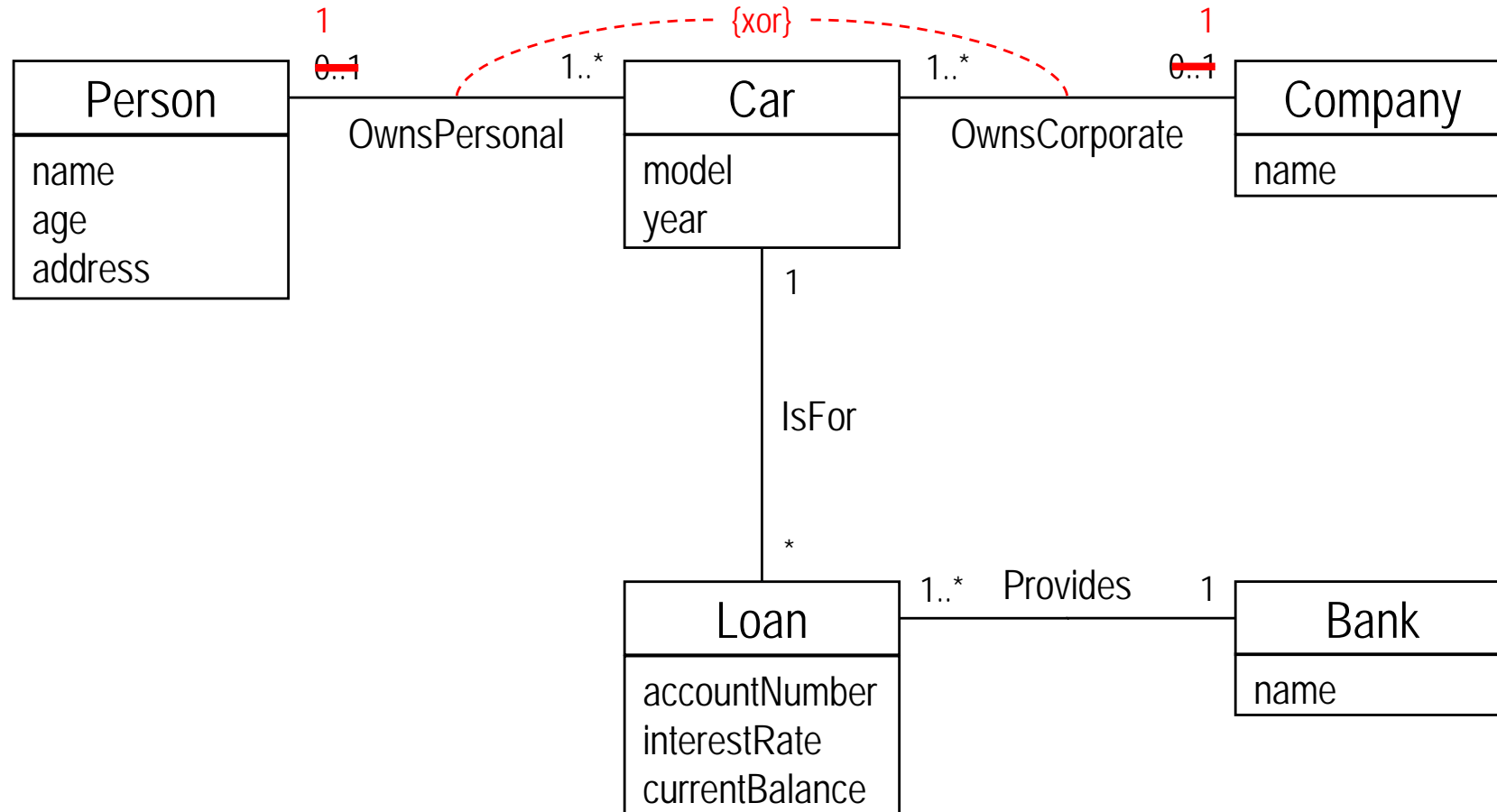
Problem 3

It is still possible for a car to be owned by both a person and a company or to not be owned at all!



Requires an **exclusive** or constraint (see later).

EXERCISE: CAR OWNERSHIP AND LOANS — SOLUTION REVISITED



Problem 3

It is still possible for a car to be owned by both a person and a company or to not be owned at all!



Requires an **exclusive** or constraint (see later).

EXERCISE: CAR OWNERSHIP AND LOANS — COMMON ERRORS

- Missing classes or associations
- Extra classes or associations not specified in the requirements
 - Stick to what is stated in the user requirements; don't be creative.
- Incorrect associations
- Redundant associations
 - Two paths in the class diagram that result in the same objects.
- Missing association names
 - While associations names are not required, it is always a good idea to name them.
- Missing or wrong multiplicities
 - (e.g., Owns)
- Using incorrect notation
 - (e.g., entity-relationship notation)