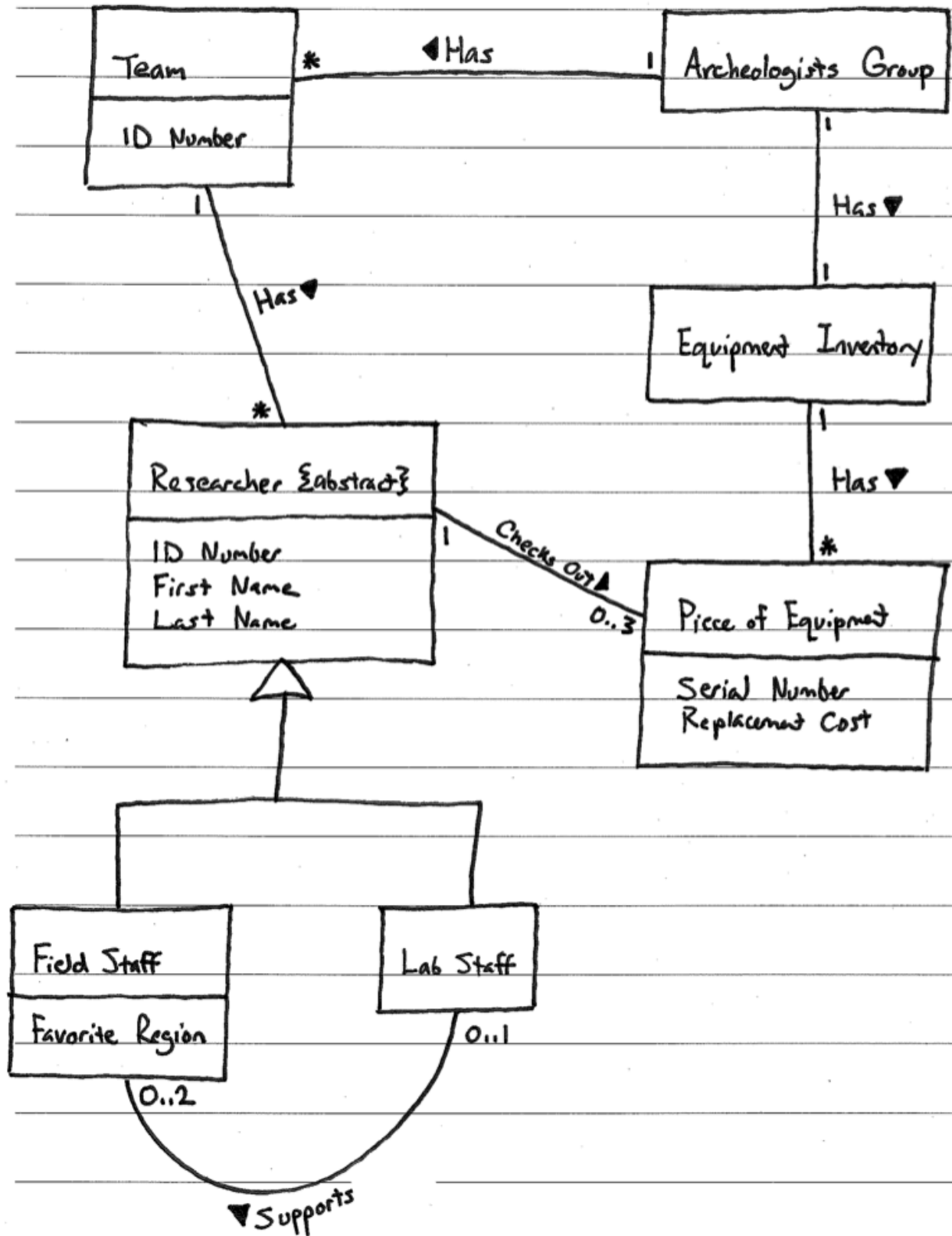


Problem: Create a domain model (using class diagram notation) based on the following description.

You have been asked to build a management system for a group of archeologists. The group is comprised of multiple teams of researchers. Each team has a letter ID (e.g., team A, team B). Each researcher belongs to one of the teams, and has an ID number, a first name, and a last name. There are two types of researchers: field and lab staff. Each field staff member has a favorite region (string). Each lab researcher supports up to 2 field researchers. Some researchers may not be supported by a lab researcher. The company also manages an inventory of equipment. Researchers of any type may check out up to 3 pieces of equipment. Each piece of equipment has a serial number and replacement cost.

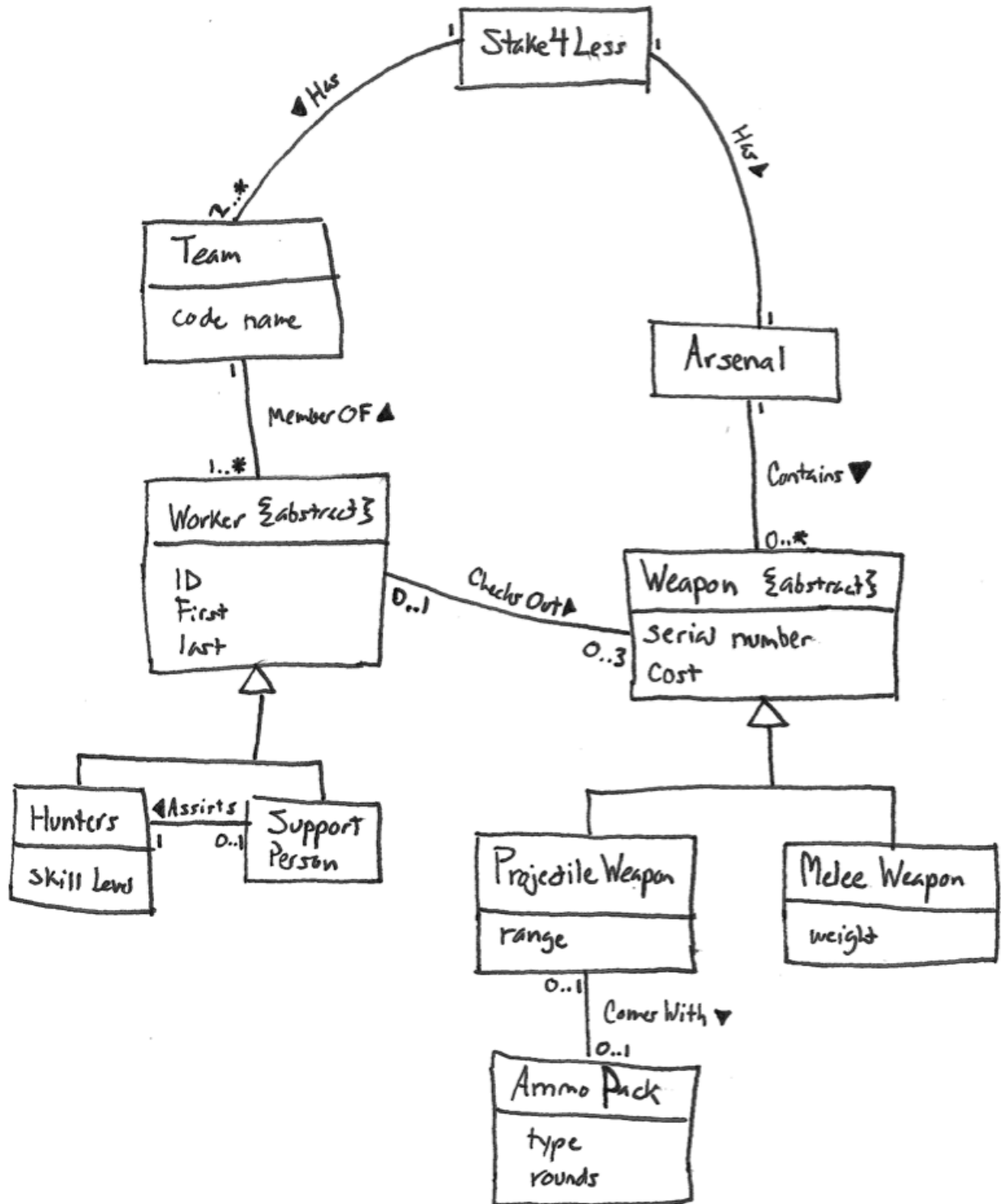
Solution:



Problem: Create a domain model (using class diagram notation) based on the following description.

You have been contracted to build an asset-management system for a vampire-hunting company, *Stake4Less*. The organization consists of multiple teams of workers. Each team has a code name. Each worker belongs to one of the teams, and has an ID number, a first name, and a last name. There are two types of workers: hunters and support personnel. Hunters have a skill level. Each support person assists exactly one hunter on the team. Some hunters on the team may not be assigned a support person. The company also manages an arsenal of weapons. Workers of any type may check out up to 3 weapons from the arsenal. Each weapon has a serial number and replacement cost. There are two types of weapons in the arsenal: projectile and melee. Projectile weapons have a range, and may come with an ammo pack. An ammo pack has an ammo type, and includes some number of rounds. Melee weapons have a weight.

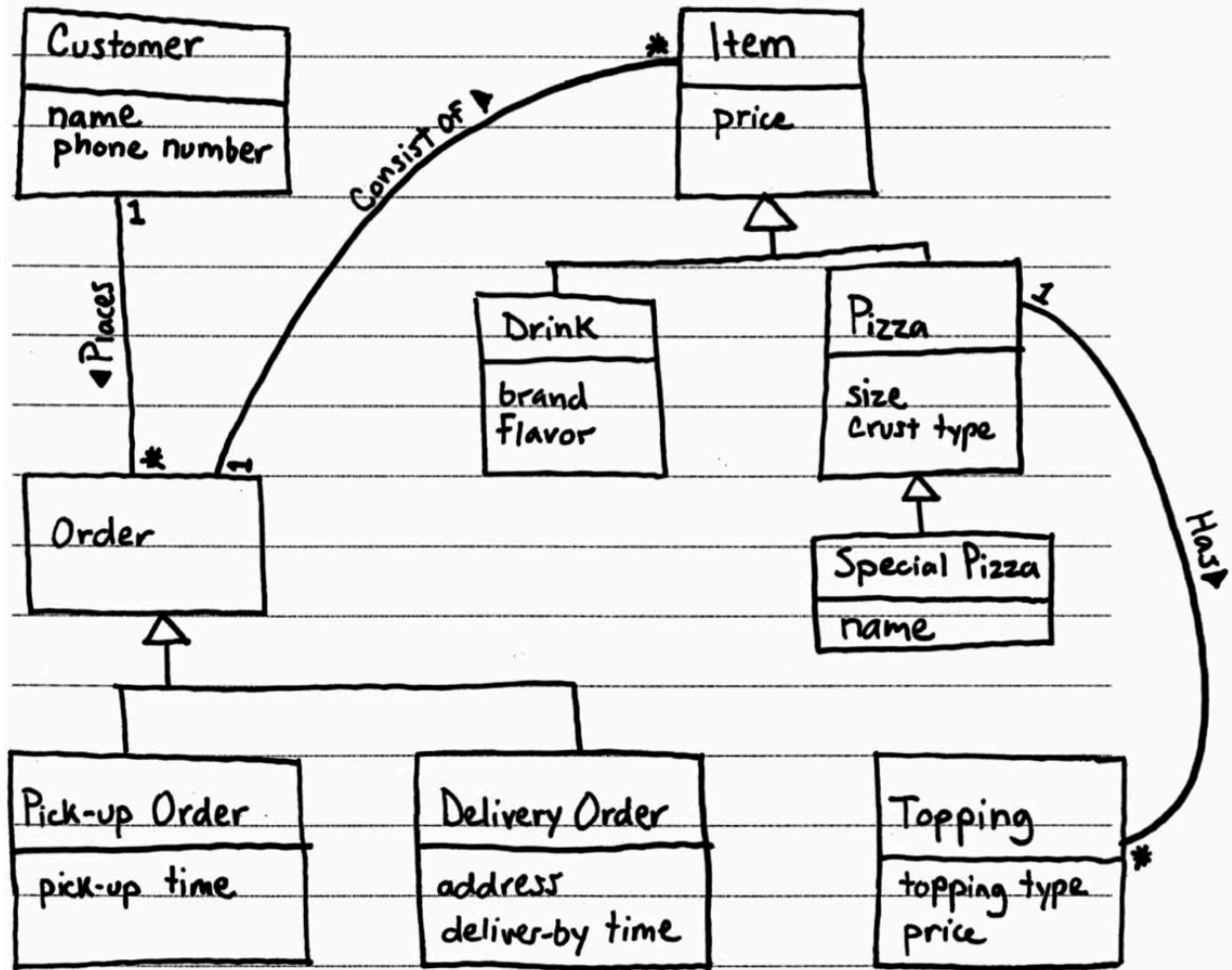
Solution:



Problem: Imagine that you are tasked with developing a system for a pizza shop. Given the following description, create a Domain Model (in the form of a UML class diagram). Include all conceptual classes, attributes, associations, and generalization relationships mentioned in the descriptions. Label all associations and include all multiplicities.

A customer places orders. A customer has a name and phone number. There are two types of orders: pick-up and delivery. A pick-up order has a pick-up time. A delivery order has an address and deliver-by time. All orders consist of a set of items. There are two types of items: pizzas and drinks. All items have a price. A pizza has a size and a crust type. A pizza also has a number of toppings. A topping has a topping type and a price. Some pizzas are special pizzas that have a name (e.g., “Hawaiian” or “Meat Lovers”). A drink has a brand and a flavor.

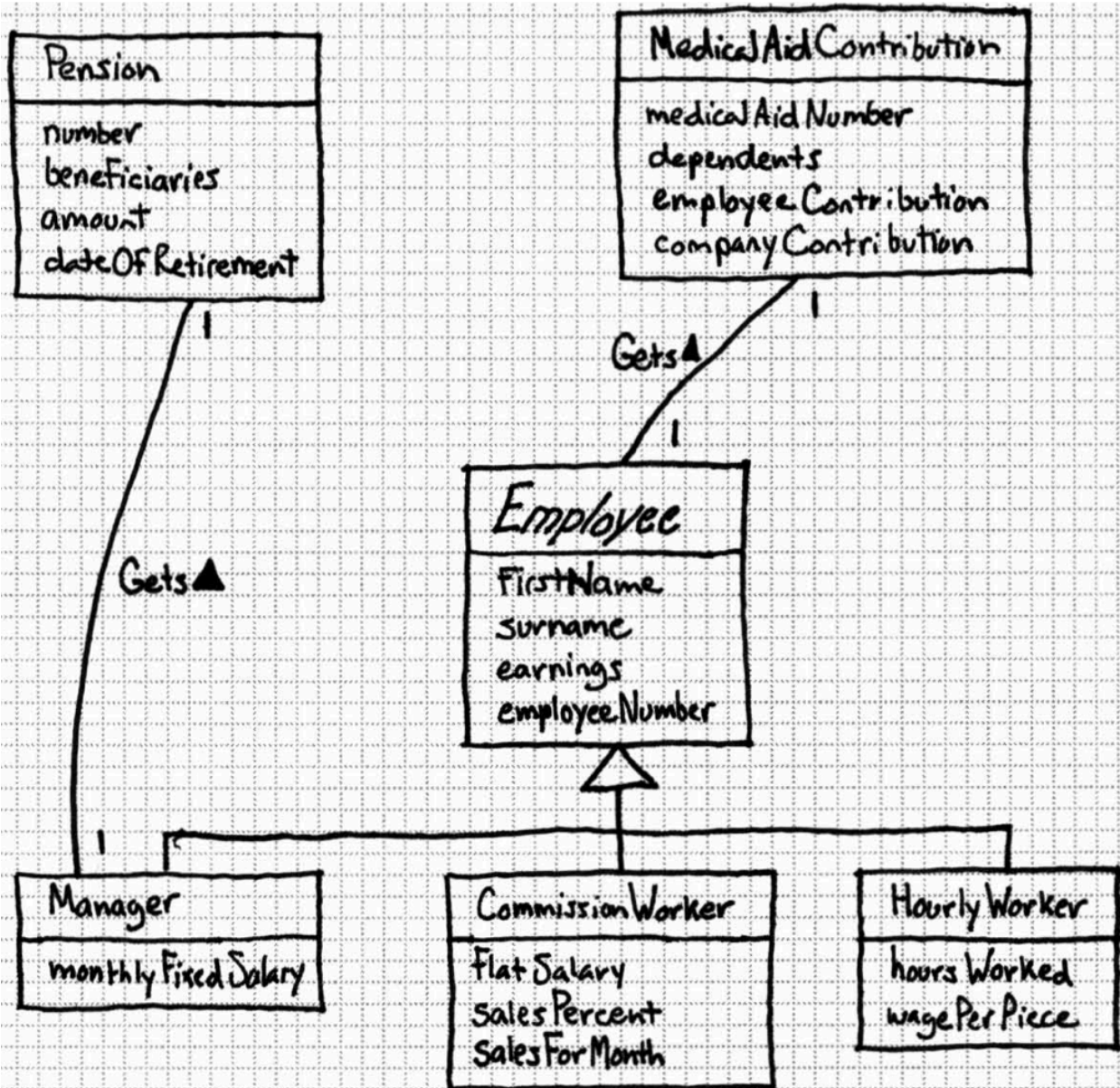
Solution:



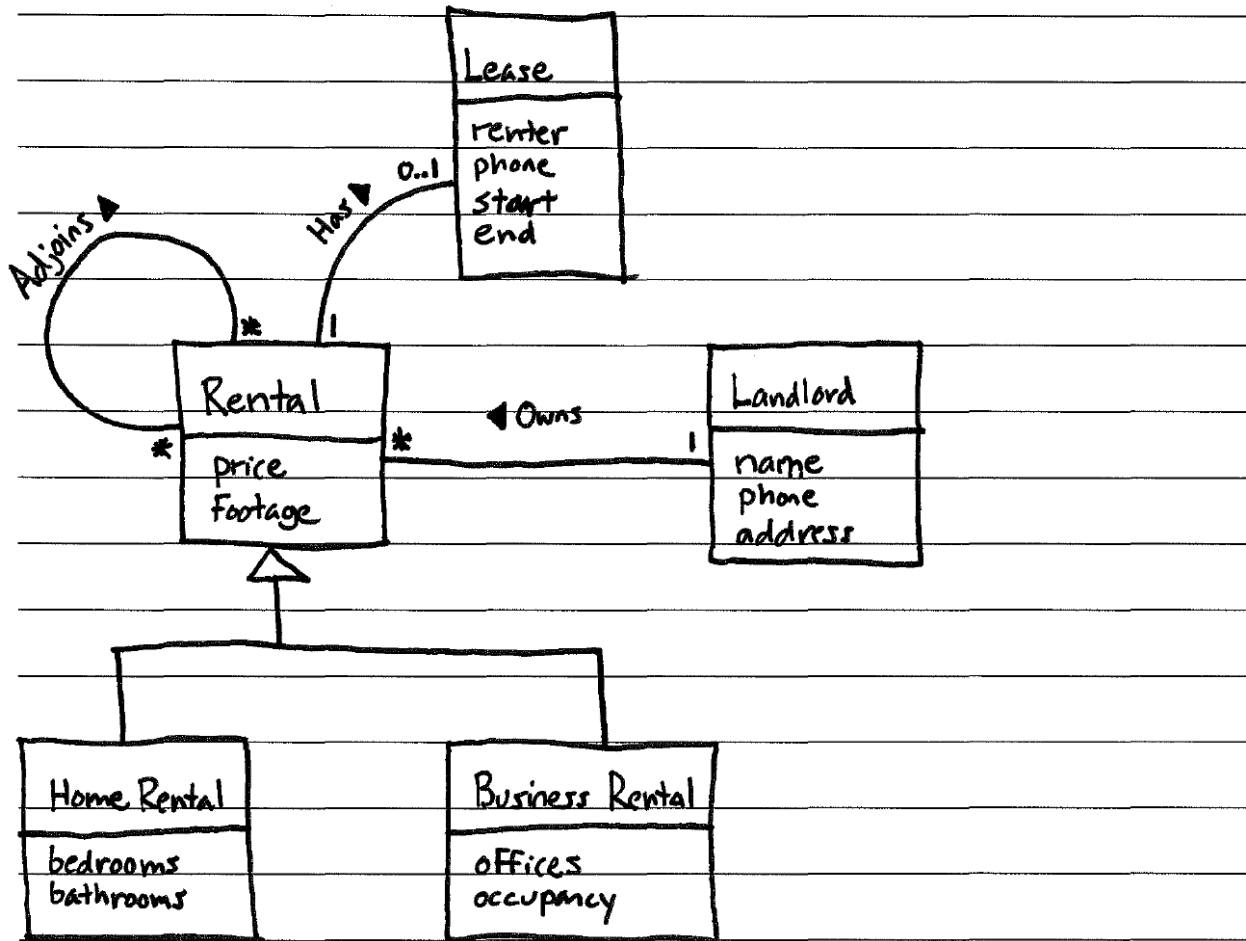
Problem: Create a domain model in the form of a class diagram based on the following description. Model only things that are specifically described. Do not model “the system.”

A company requires a payroll system. The company has three types of employees, namely, a manager, a commission worker, and an hourly worker. Information that needs to be stored for each employee is the employee’s first name, surname, earnings and employee number. Additional information that must be stored for a manager is his/her monthly fixed salary; for a commission worker his/her flat salary, sales percent and the sales made for the month; for an hourly worker the number of hours worked and the wage per piece. All employees get a medical aid contribution. In addition to this managers get a pension as a fringe benefit. Each medical aid instance specifies the medical aid number, dependents, employee contribution and company contribution, while each pension instance specifies the pension number, beneficiaries, amount and date of retirement.

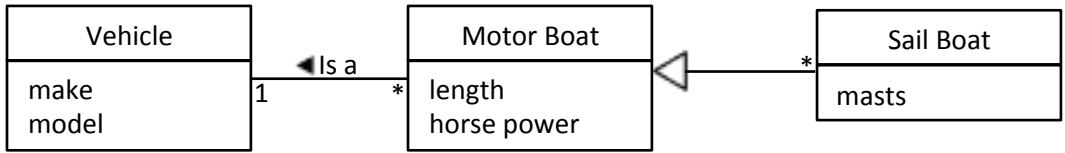
Solution:



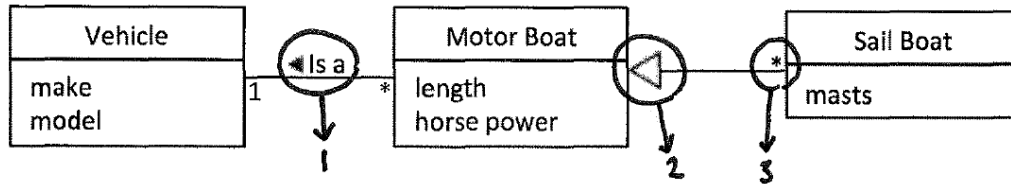
Solution:



Problem: Name three problems with this domain model (esp. with respect to class diagram notation).



Solution:



1. An "is-a" relationship should be modeled with a generalization (i.e., an arrow like this: \leftarrow).
2. This generalization violates both the 1s-a and 100% rules:
Not all sail boats have motors (with horse power).
3. Generalization relationships should not have multiplicities.