# Chapter 4 – Domain Modeling

## Solutions to End-of-Chapter Problems

### Review Questions

**1. What are the two key concepts—one from Chapter 3 and one from this chapter—that define functional requirements?**

Use cases and Problem domain model (objects)

**2. What is the problem domain?**

The area of the user's business need. It is called “problem” because there is a need or something to be fixed. It is the domain, or the area of the business.

**3. What is a “thing” called in models used by traditional analysts and database analysts?**

Data entity

**4. What is a “thing” called in newer approaches that use UML?**

Object or object class

**5. What are two techniques for identifying things in the problem domain?**

The Brainstorming technique and the Noun technique

**6. What are some examples of tangible things in the problem domain of a restaurant?**

Meal, food item, menu...

**7. What are some sites or locations in the problem domain of a restaurant?**

Restaurant location (for a chain),

**8. What are some roles played by people in the problem domain of a restaurant?**

Customer, cook, waiter, supplier

**9. What are the main steps of the brainstorming technique?**

1. Identify a group of users and their related use cases.

2. Brainstorm with this group about all the things they need to keep track of with these use cases.

3. Expand the list of things by asking related questions about locations, roles, etc.

4. Return to step 1 with different groups of users.

5. Combine and merge, eliminating duplicates.

**10. Explain why identifying nouns helps identify things in the problem domain?**

Nouns are always “things.” So finding all the nouns will find all the things (and more, so it needs to be refined).

**11. What are the main steps of the noun technique?**

1. Using use cases, dialogs, conversations, etc. list all the nouns.

2. Using information from existing systems, etc. list all the nouns.

3. Refine the list by asking 3 questions for each noun – Include it? Exclude it? Research it more?

4. Create a master list out of step 3.

5. Review the list with users, stakeholders, and other team members.

**12. What is an attribute, an identifier or key, and a compound attribute?**

* An attribute is a a descriptor of a data entity (Object). For example a Customer (object) has a name (attribute). So although attributes are also nouns, they are descriptor or qualifier nouns.
* An identifier or key is an attribute that can uniquely identify a particular object.
* A compound attribute is an attribute consists of “sub-attributes” or components, like address consists of street, city, state, postal code.

**13. What is an association, and what system development standard defines it?**

An association is a relationship between things in the problem domain. It is the term used by UML.

**14. How would you describe or name the association between a ship and a captain?**

A captain is in charge of a ship. or A captain directs a ship.

**15. What is the term used for association by traditional analysts and database analysts?**

It is called a relationship.

**16. What is multiplicity, and what is the other term used by traditional analysts and database analysts?**

It is a measure of the number of links in an association between an object in one class and the objects in another class. In traditional analysis it is called cardinality.

**17. What is the minimum multiplicity for the association that reads a customer places zero or more orders?**

Zero

**18. What is the maximum multiplicity for the association that reads an order is placed by exactly one customer?**

One

**19. What are some examples of multiplicity constraints?**

Customer places one or more orders.

Customer has only one account.

Order is place by only one customer.

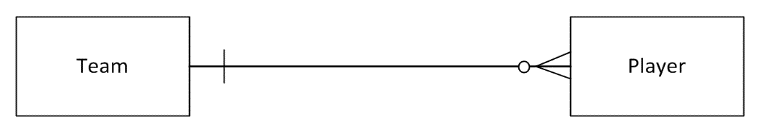
**20. What are the three types of associations, and which is the most commonly used?**

Binary, unary, ternary. Binary is most common.

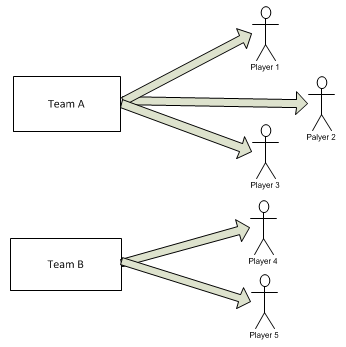
**21. What are the three key parts of an entity relationship diagram (ERD)?**

Data entities, relationships, and cardinality constraints.

**22. Sketch a simple ERD that shows a team has zero or more players and each player is on one and only one team.**



**23. Sketch a semantic net that shows two teams and five players based on your ERD.**



**24. What is a class, a domain class, and the key parts of a class diagram?**

A class is a set of objects that are similar in nature had have the same “classification.”

A domain class is a class in the problem domain.

A class diagram has classes with attributes, associations, and multiplicity constraints.

**25. What does a domain model class diagram show about system requirements, and how is it different from an ERD?**

A domain model shows the classes, i.e. the things, and their relationships and constraints. These are the specific system requirements that must be built into the database. The problem domain classes are the classes from the domain and are “persistent”, e.g. they must be stored in a database.

An ERD has different notation than a domain model. An ERD is not as powerful as a domain model to model specific real world conditions.

**26. List appropriate UML class names by using the camelback notation for the following classes: graduate student, undergraduate major, course instructor, and final exam feedback.**

GraduateStudent

UndergraduateMajor

CourseInstructor

FinalExamFeedback

**27. List appropriate UML attribute names for the following attributes: student name, course grade, major name, and final exam quantity score.**

studentName

courseGrade

majorName

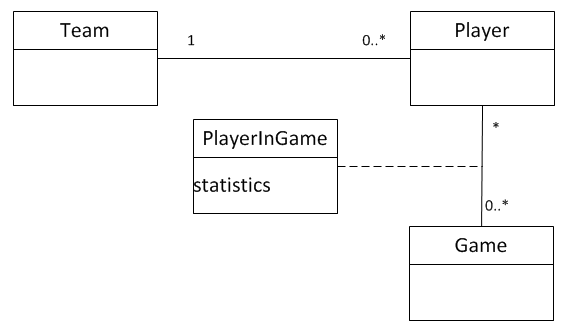
finalExamQuantityScore

**28. Draw a simple domain model class diagram for the example in question #22 where a team has zero or more players and each player is on one and only one team.**



**29. What is an association class? Extend the domain model class diagram for teams and players about to show a record of game statistics for each player in each game.**

An association class is an association that needs to also be treated as a class. It is an association that requires attributes just like any class.



**30. In UML, what are three types of relationships found on a class diagram?**

Regular association, Generalization/Specialization, Whole-part

**31. What is a generalization/specialization relationship, and what object-oriented terms does it illustrate?**

A generalization/specialization is a hierarchical relationship between classes, where some classes are subsets (e.g. subclasses) of other classes.

**32. Compare/contrast superclass and subclass. Compare/contrasts abstract class and concrete class.**

A superclass is higher in the relationship hierarchy and is a superset. A subclass is lower and is a subset of the superset.

An abstract class is a class with no objects allowed. It serves only as a template for attributes so that subclasses, which are concrete classes, will have objects with inherited attributes.

**33. What is a whole-part relationship, and why does it show multiplicity?**

A whole-part relationship in which a class is part of another class. It can have multiplicity constraints that allow multiple part-classes to belong to a single whole class.

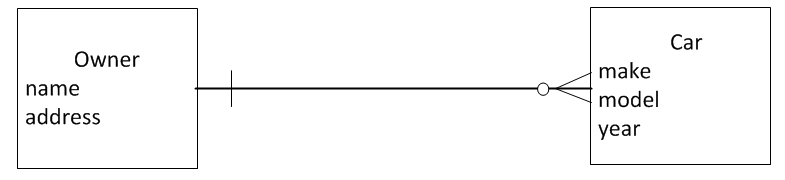
**34. Compare/contrast aggregation with composition for a whole part relationship.**

Aggregation is where the “part” objects may exist outside of the whole-part relationship. This often applies to physical devices that can exist prior to becoming part of the aggregate.

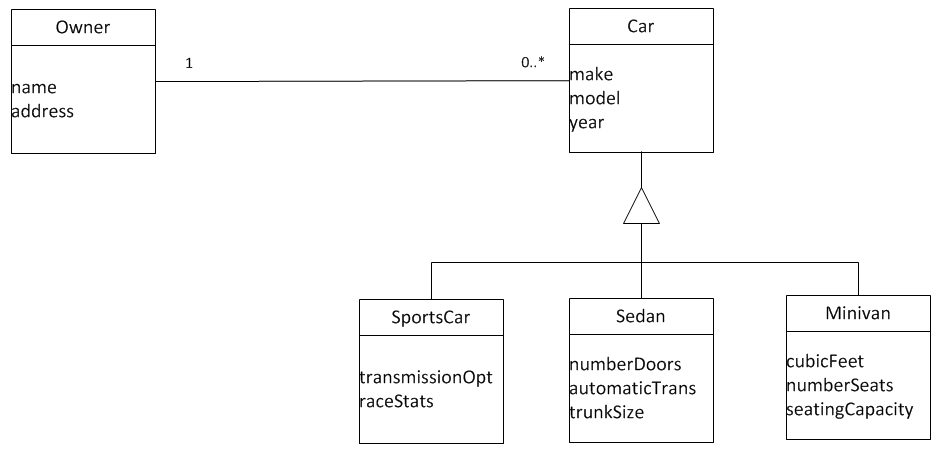
Composition is where the parts do not exist outside of the whole-part relationship. An example might be a sale which is “composed” of sale items. The sale items do not exist separate and apart from the “sale” object.

### Problems and Exercises

**1. Draw an entity-relationship diagram, including minimum and maximum cardinality, for the following: The system stores information about two things: cars and owners. A car has attributes for make, model, and year. The owner has attributes for name and address. Assume that a car must be owned by one owner and an owner can own many cars, but an owner might not own any cars (perhaps she just sold them all, but you still want a record of her in the system).**



**2. Draw a class diagram for the cars and owners described in exercise 1, but include subclasses for sports car, sedan, and minivan, with appropriate attributes.**



**3. Consider the domain model class diagram shown in Figure 4-16—the refined diagram showing course enrollment with an association class. Does this model allow a student to enroll in more than one course section at a time? Does the model allow a course section to contain more than one student? Does the model allow a student to enroll in several sections of the same course and get a grade for each enrollment? Does the model store information about all grades earned by all students in all sections?**

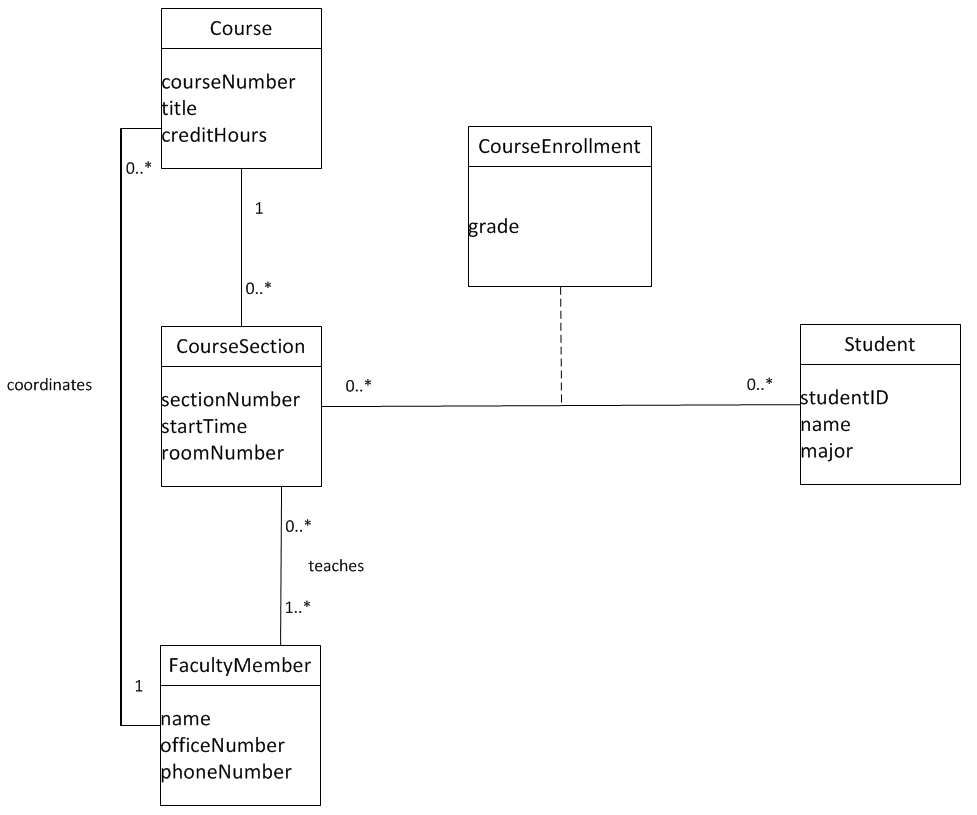
More than one section? → Yes, he/she can enroll in zero to many

Course section more than one student? → Yes, zero to many students in a section

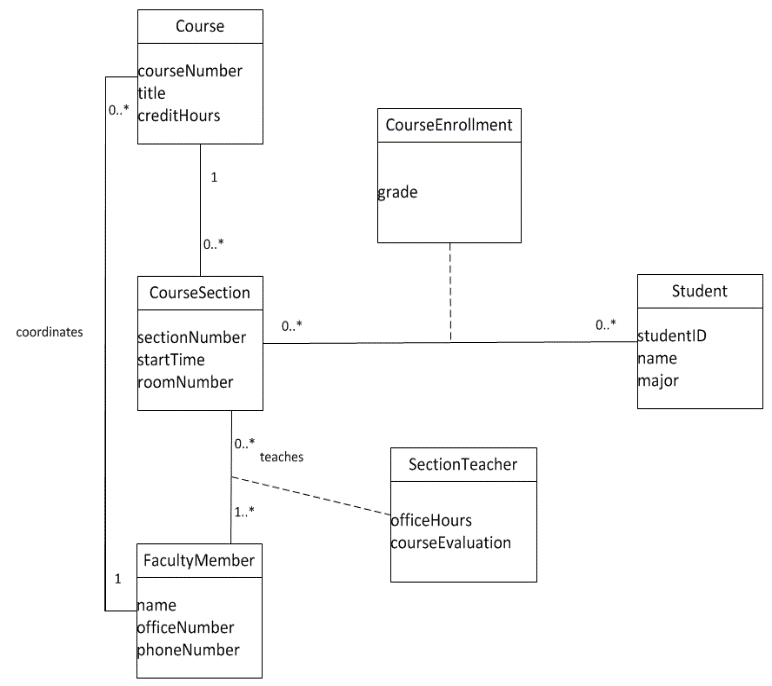
More than one section of same course? → Yes, there is no constraint to limit it

Store of grades? → Yes it captures all student grades

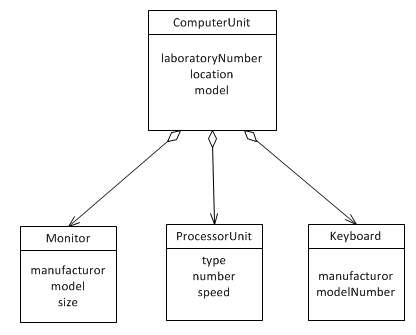
**4. Again consider the domain model class diagram shown in Figure 4-16. Add the following to the diagram and list any assumptions you had to make: A faculty member usually teaches many course sections, but some semesters, a faculty member may not teach any. Each course section must have at least one faculty member teaching it, but sometimes, faculty teams teach course sections. Furthermore, to make sure that all course sections are similar, one faculty member is assigned as course coordinator to oversee the course, and each faculty member can be the coordinator of many courses.**



**5. If the domain model class diagram you drew in exercise 4 showed a many-to-many association between faculty member and course section, a further look at the association might reveal the need to store some additional information. What might this information include? (Hint: Does the instructor have specific office hours for each course section? Do you give an instructor some sort of evaluation for each course section?) Expand the domain model class diagram to allow the system to store this additional information.**



**6. Consider a system that needs to store information about computers in a computer lab at a university, such as the features and location of each computer. What are the domain classes that might be included in a model? What are some of the associations among these classes? What are some of the attributes of each class? Draw an domain model class diagram for this system.**



**7. Consider the domain model class diagram for the RMO CSMS Sales subsystem shown in Figure 4-21. If an InStoreSale is created, how many attributes does it have? If an OnlineSale is created, how many attributes does it have? If an existing customer places a telephone order for one item, how many new objects are created overall for this transaction? Explain.**

InStoreSale has 6 + 3 = 9 attributes

OnlineSale has 6 + 2 = 8 attributes

For a telephone sale to an existing customer:

* TelephoneSale
* SaleItem
* Sale Trans
* No *Sale* is created because it is an abstract class.

**8. Again consider the domain model class diagram shown in Figure 4-21. How many attributes does an active cart object have? Can an on-reserve cart contain cart items? Explain.**

An active cart object has 4 + 1 = 5 attributes

OnReserveCart → Yes it can have CartItems. An OnReserveCart is also an *OnlineCart* (It is a subset).

**9. A product item for RMO is not the same as an inventory item. A product item is something like a men’s leather hunting jacket supplied by Leather ‘R’ Us. An inventory item is a specific size and color of the jacket—like a size medium brown leather hunting jacket. If RMO adds a new jacket to its catalog and six sizes and three colors are available in inventory, how many objects need to be added overall? Explain.**

New jacket requires 1 product item object and 3 \* 6 = 18 inventory item objects. Total of 19.

**10. Consider the domain model class diagram shown in Figure 4-24, which includes classes for college, department, and faculty members.**

**a. What kind of UML relationships are shown in the model?**

There are two binary association relationships

**b. How many attributes does a “faculty member” have? Which (if any) have been inherited from another class?**

FacultyMember has 5 attributes. None inherited.

**c. If you add information about one college, one department, and four faculty members, how many objects do you add to the system?**

You add 6 objects.

**d. Can a faculty member work in more than one department at the same time? Explain.**

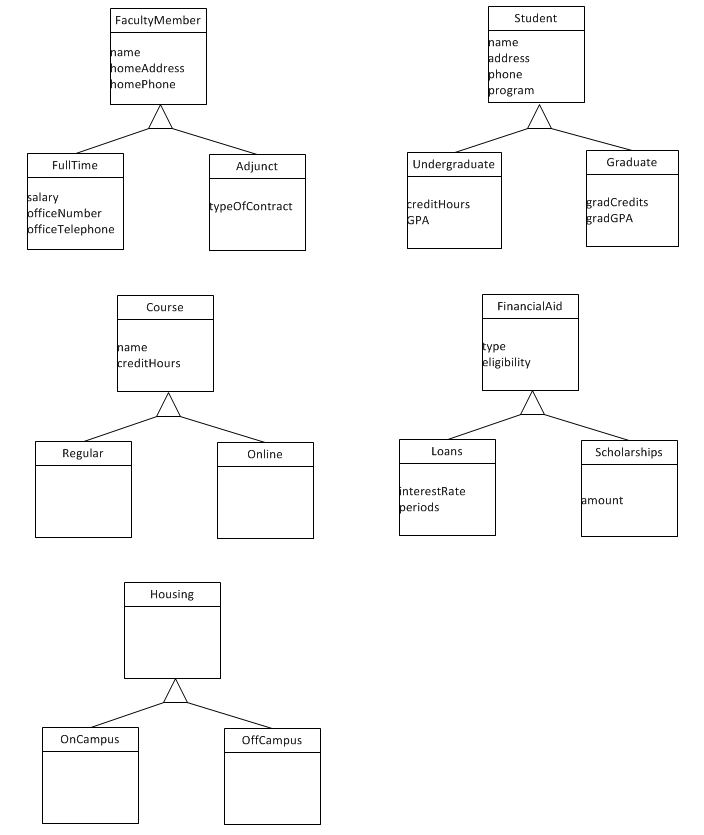
Using minimum and maximum multiplicity (cardinality), we can say that a faculty member can be in more than one department at the same time. In the real world, for example, one teacher can be part of both the computer science and computer information systems departments (a split appointment is possible).

**e. Can a faculty member work in two departments at the same time, where one department is in the college of business and the other department is in the college of arts and sciences? Explain.**

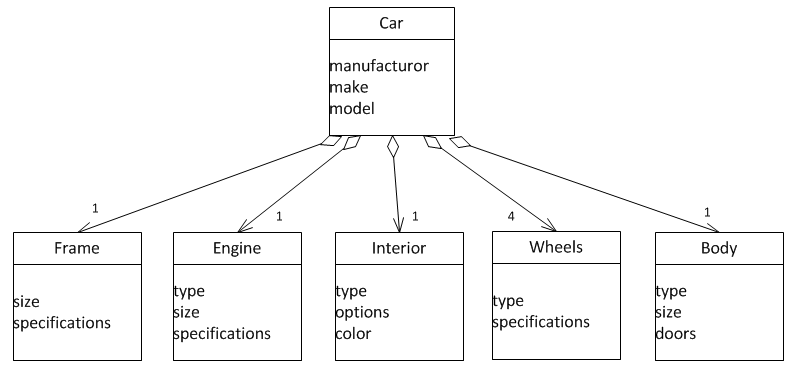
This is a good question to discuss in class. Draw a circle for a FacultyMember object (Billy Bob) and two circles for two separate Department objects (computer science and CIS). Connect the FacultyMember to each Department with two lines. These lines associate one FacultyMember object with two Department objects. Next, draw two College objects (business and sciences). Connect computer science to sciences and CIS to business. Now it should be clear that a faculty member can be part of two departments at the same time.

**11. Review information about your own university. Create generalization/specialization hierarchies by using the domain model class diagram notation for (1) types of faculty, (2) types of students, (3) types of courses, (4) types of financial aid, and (5) types of housing. Include attributes for the superclass and the subclasses in each case.**

Answers will vary. Types of faculty might include tenure track or lecturer, and tenure track might include tenured or non-tenured. Types of students might include undergraduate or graduate or, in some cases, day student (full time) or evening student (part time). Financial aid might include scholarships, loans, or work-study. Housing might include on campus or off campus. On campus might include dorm, suite, or apartment. Please note that a subclass is included only if there are additional attributes (or methods) for it that do not apply to the superclass. Also, the subclass and its attributes are only included if the system being described requires keeping track of the distinction. Should there be a subclass? Only if the system must remember the additional specific details it provides.

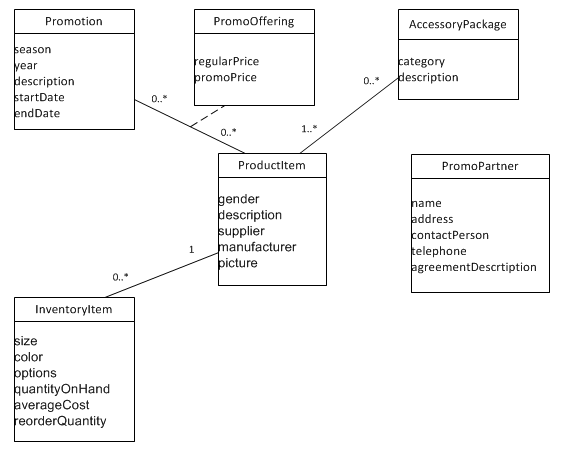


**12. Consider the classes involved when modeling a car and all its parts. Draw a domain model class diagram that shows the whole-part relationships involved, including multiplicity. Which type of whole-part relationships are involved?**



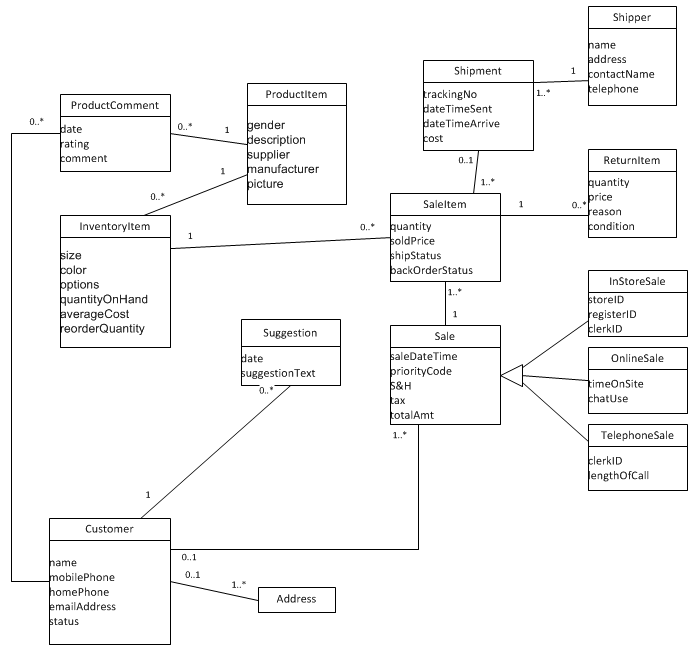
**13. Refer to the complete RMO CSMS domain model class diagram shown in Figure 4-23. Based on that model and on the discussion of subsystems in Chapter 3, draw a domain model class diagram for the CSMS Marketing subsystem.**

Answers may vary depending on how they interpret the use cases.



**14. Again based on the complete RMO CSMS domain model class diagram shown in Figure 4-23, draw a domain model class diagram for the CSMS Order Fulfillment subsystem.**

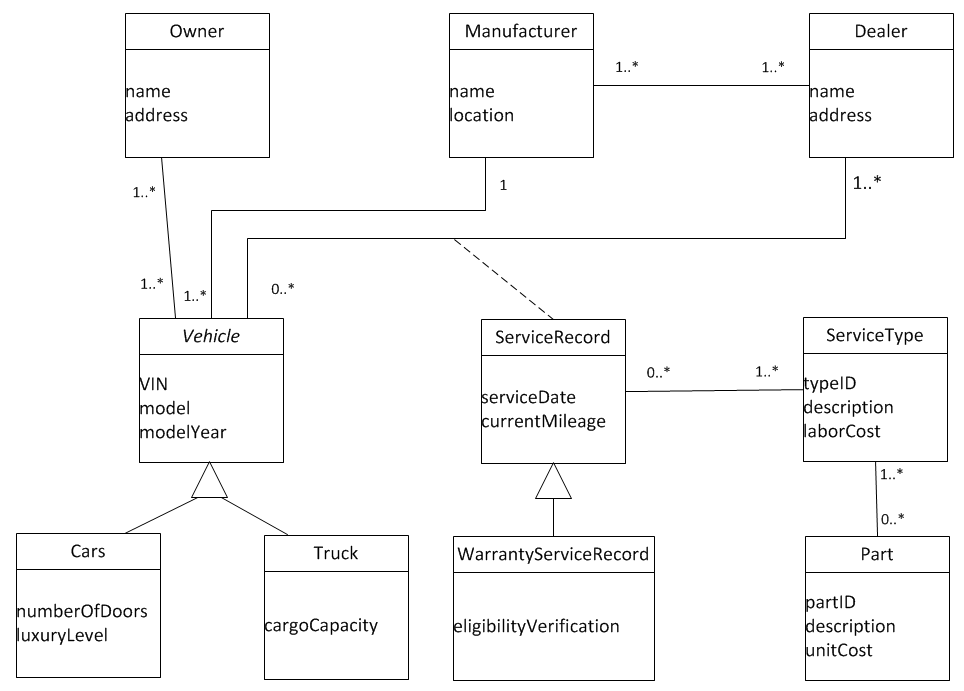
Answers may vary. But this subsystem reads a lot of database classes, although it does not necessarily create or update all the data.



## Solutions to End-of-Chapter Cases

### Case Study: Metropolitan car Service Bureau

**1. Draw a UML domain model class diagram for the system as described here. Be as specific and accurate as possible, given the information provided. If needed information is not given, make realistic assumptions.**



**2. Answer True or False to the following statements, which are based on the domain model. You may want to draw a semantic net to help you think through the questions**

**a. This domain model is for a single car dealer service department.**

False

**b. This domain model is for a single car manufacturer.**

False

**c. A vehicle can have service records with more than one dealer.**

True

**d. A dealer can service vehicles from more than one manufacturer.**

True

**e. Current mileage is recorded for service records and warranty service records.**

True

**f. An owner can have each of his or her cars serviced by a different dealer.**

True

**g. A warranty service for a car can include many parts.**

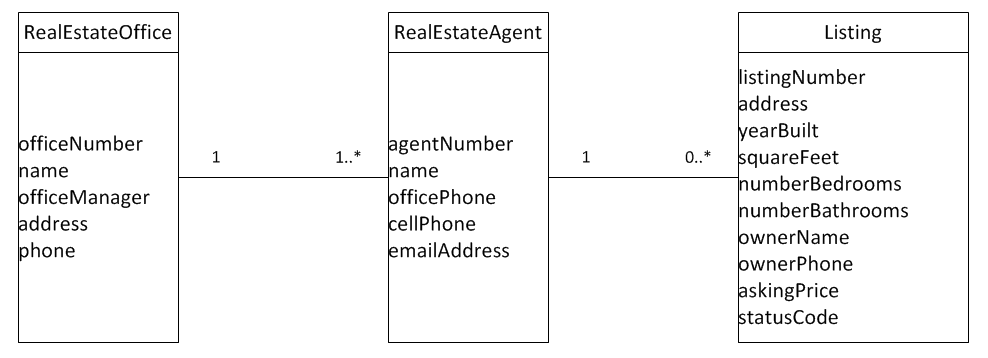
True

**h. A vehicle can be made by more than one manufacturer.**

False

### Running Cases: Community Board of Realtors

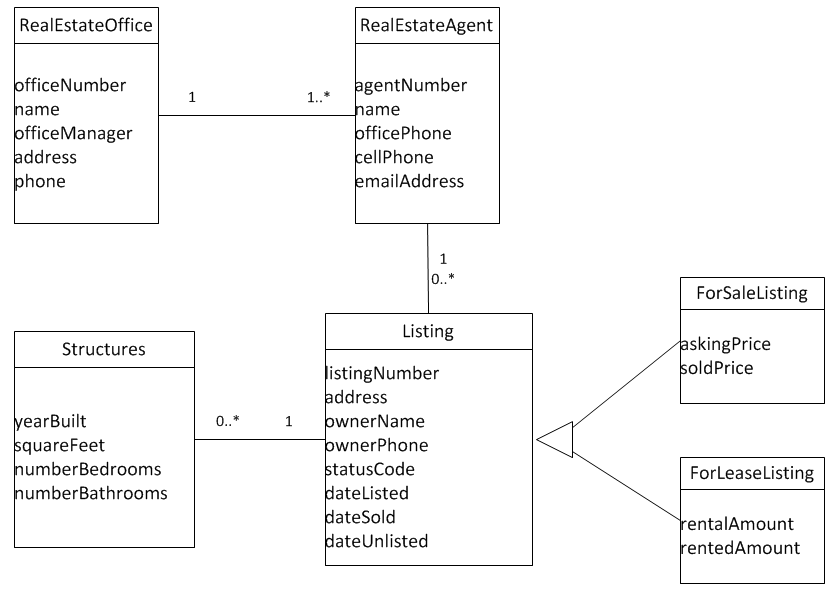
**1. Based on the information here, draw a domain model class diagram for the MLS system. Be sure to consider what information needs to be included versus information that is not in the problem domain. For example, is detailed information about the owner, such as his employer or his credit history, required in the MLS system? Is that information required regarding a potential buyer?**



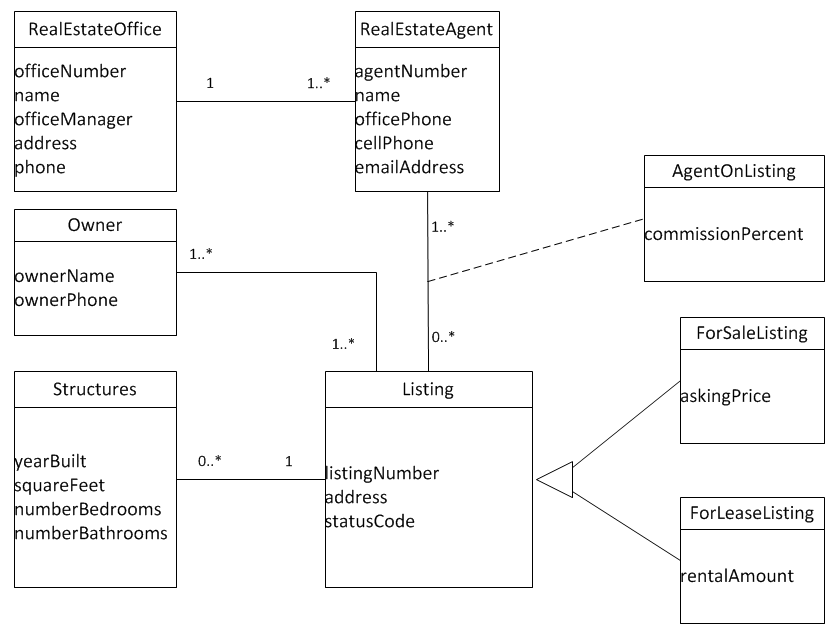
**2. Draw a second domain model class diagram that adds the following specifications. First, there are two types of listings: a listing for sale and a listing for lease. Additionally, a listing might include no structures, such as vacant land, or it might include more than one structure, such as a main house and a guest house, each with separate values for square footage, number of bedrooms, and number of bathrooms.**



**Note: Here is an additional solution to Question 2 with a few more attributes added that are not requested by the case, but that will be helpful for later chapters. It adds some attributes to produce more reports.**



**3. Draw a third domain model class diagram that assumes a listing might have multiple owners. Additionally, a listing might be shared by two or more agents, and the percentage of the commission that each agent gets from the sale can be different for each agent.**

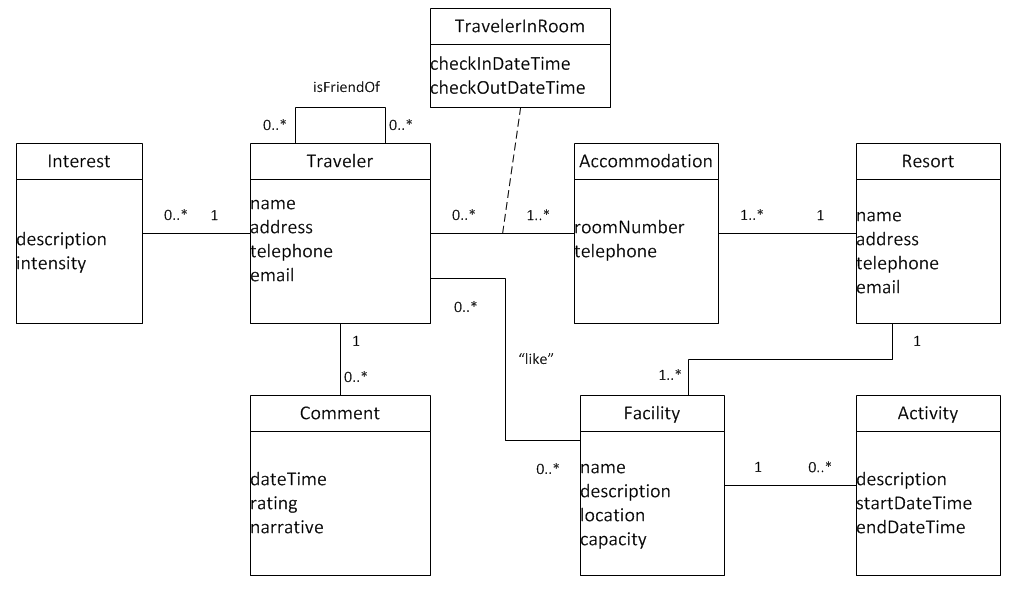


### Running Cases: The Spring Breaks 'R' Us Travel Service

**1. For the Social Networking subsystem as described here, list the domain classes and their attributes that should be included in the Social Networking subsystem. Be creative and add those you think should be included to make the system useful and appealing.**

See solution below

**2. Based on the domain classes you identified, draw a domain model class diagram showing domain classes with attributes and associations with multiplicity.**



Note: Student solutions will vary. “Comment” will probably not appear on student solutions.

Note: Some students might try to make an accommodation class as an association class between Traveler and Resort, e.g. on an association such as *Traveler stays at Resort.*  However, this will only work if it is defined as CurrentAccommodation and only one is allowed in the database at a time. In other words, a given Traveler may stay at the same Resort on multiple dates. In that case an Accommodation association would not be unique by the keys TravelerID-ResortID, and the above solution is more general.

### Running Cases: On the Spot Courier Services

**1. Using the noun technique, read through this case and identify all the nouns that may be important for this system. You may also find it helpful to read back through the case descriptions in the previous chapters.**

The noun technique yielded the following list.

pickup requests - class

package pickups – class ?

mobile phone - ?

packages - class

driver - class

warehouse - ?

warehouse employee – class ?

deliveries – class ?

desk – self: not important

delivery van - ?

business – self: not important

services – ?

customers – class

businesses – class

individuals – subclass

address – attribute

contact information – attribute

contact person – attribute

monthly statements – output

shipments – class ? duplicate

bill month – attribute

cost – attribute

cash – attribute

running account – class

outstanding balances – attribute

payments – class

invoices – output

type of payment – attribute

amount – attribute

delivery request – class

delivery order – class ? duplicate

pickup location – attribute

date and time – attribute

delivery order – class ? duplicate

delivery person – duplicate

delivery to name – attribute

type of delivery – attribute

weight – attribute

names – duplicate

dateTimeStamp - attribute

delivery trip – class

**2. Once you have identified all the nouns, identify which are classes and which are attributes of these primary classes. Begin constructing a class diagram based on the classes and attributes you have identified.**

Note: In the analysis, it was determined that dateTimeStamp was an attribute of package movements. However, no Movement noun was identified. So we added a general class called “MovementEvent” to track package movements, such as pickups, deliveries, warehouse movements.

Also a trip noun was not specifically mentioned but it was strongly implied by the driver delivering packages. “Delivery Trip” was included above, which has been added as RouteTrip since it handles both deliveries and pickups.

See below for solution.

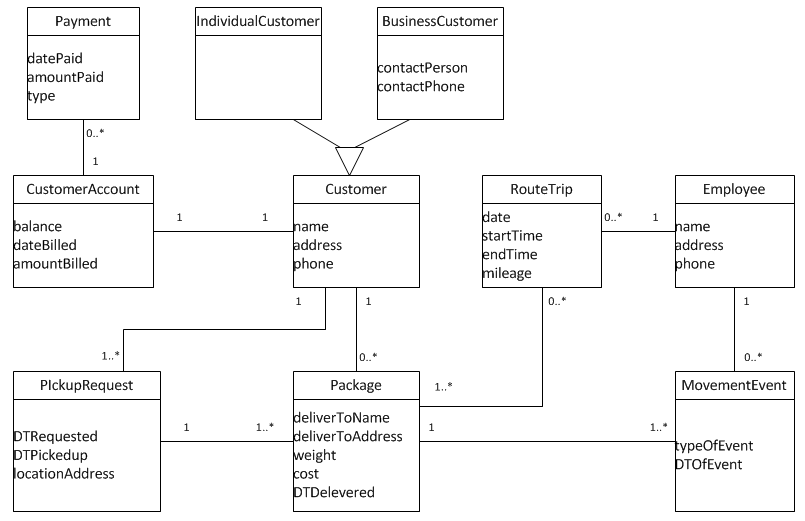
**3. Now that you have identified the classes, determine what the relationships should be among the classes. Add multiplicity constraints, being especially cognizant of zero-to-many versus one-to-many differences.**

See below for solution.

**4. Finalize the class diagram, including all your classes, attributes, primary keys, relationships, and multiplicity constraints.**

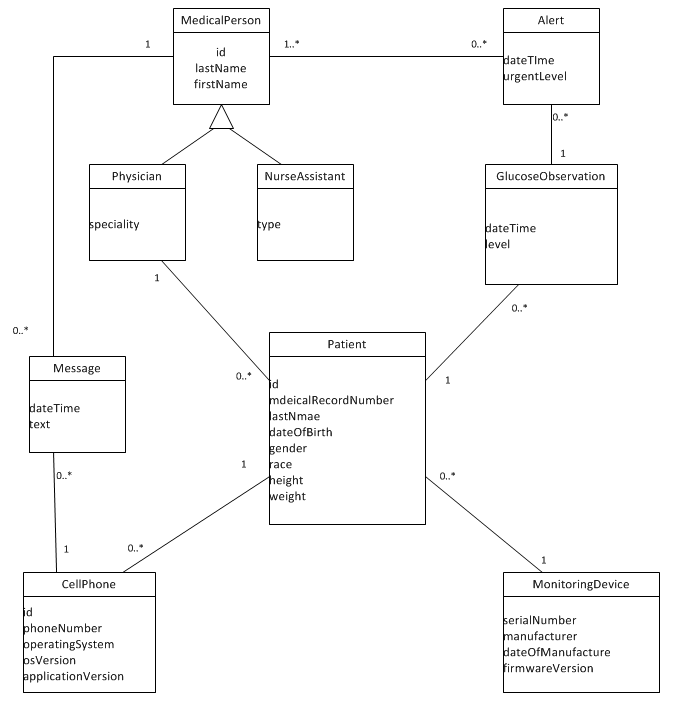
Note: With this domain model it becomes evident that the use cases identified in Chapter 3 are insufficient. A CRUD analysis will yield additional use cases.

Note: DT means Date & Time, such as DTRequested.



### Running Cases: Sandia Medical Devices

**1. Modify the diagram (Figure 4-25) to incorporate the changes under consideration. You may need to use association classes and generalization/specialization (inheritance).**



**2. Are a set of abstract and concrete classes needed to represent variations among cell phones? Why or why not?**

Answers will vary.

Possibly to distinguish between different types of cell phones, a general abstract class and specialized concrete classes might be necessary.