## Class Modeling

- A class modeling captures static structure of a system by characterizing the objects in the system, the relationships between the objects, and the attributes and operations for each class of objects.
- Class model is most important among three models.
- Emphasizes on building a system around objects rather than functionality.
- Class Model closely corresponds to the real world and is consequently more flexible with respect to the change.
- The purpose of class modeling is to describe object.

### Object and Class Concepts

### Object:

- Object is a concept, abstraction, or a thing with identity that has meaning for an application E.g. Two apples each have identity and are distinguishable.
- Objects are instances of classes.
- It often appears as a proper nouns or specific references in problem descriptions
- Some objects have real world counterparts (name of a person/company).
- While some object have conceptual entity (formula for solving quadratic equation).
- Choice of object depends on judgment and the nature of a problem. There can be many correct representations. All Object have identity and distinguishable.
- Identity means objects are distinguished by their inherent existence and not by descriptive properties.
- Real-world objects share two characteristics: They all have attributes and behavior.

### Class:

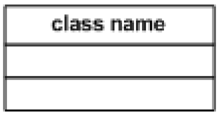
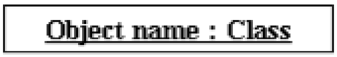
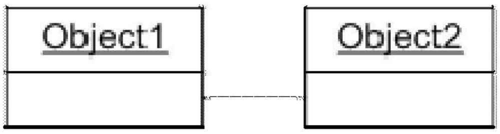
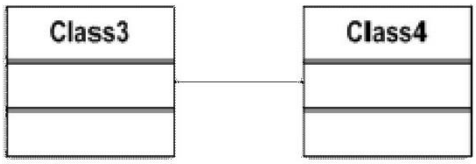
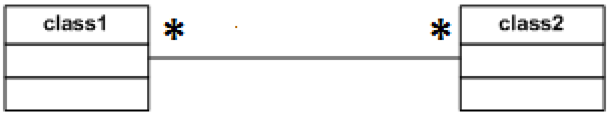
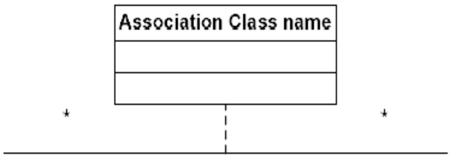
- Class describes a group of objects with the same properties (attributes), behaviour (operations), kinds of relationship, and Semantics. (E.g.: Person, Company, Process and Window).
- A software unit that implements one or more interfaces.
- Classes often appear as common noun and noun phrase in problem description.
- By grouping objects into class, we abstract a problem.

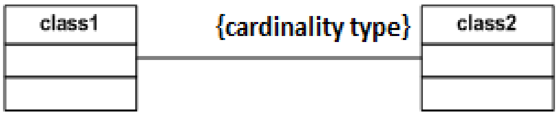
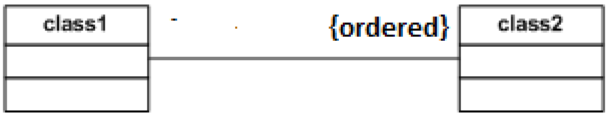
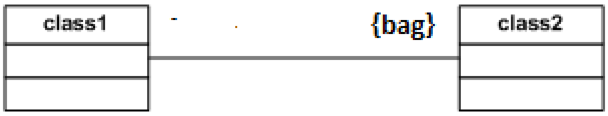
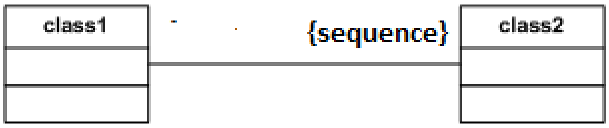
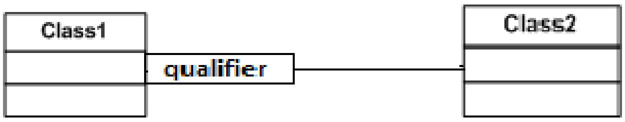
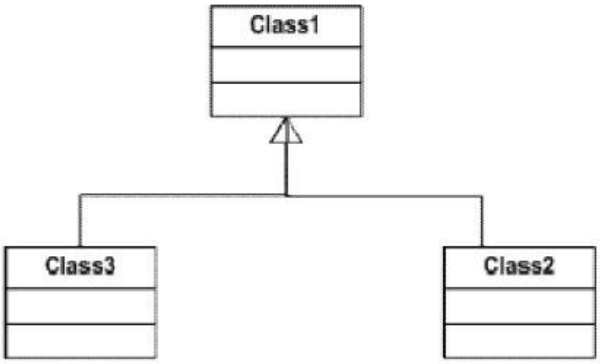
### Class Diagram

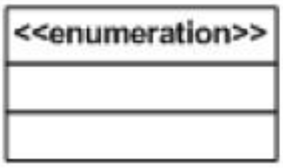
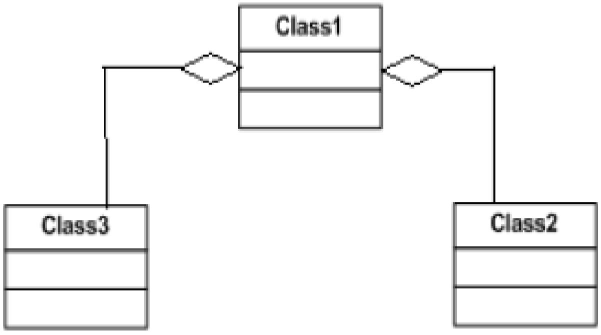
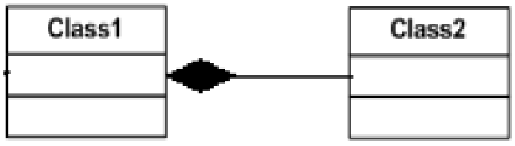
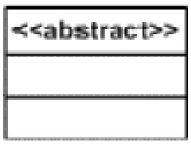
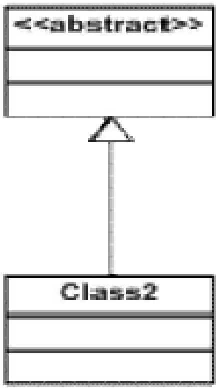
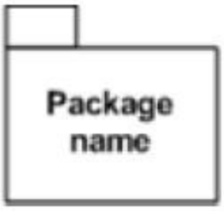
- Class Diagram provides a Graphical notation for modeling classes and their relationships, thereby describing possible objects.
- The most widely used diagram of UML.
- Models the static design view of a system.
- Useful in modeling business objects.
- Used to specify the structure, interfaces and relationships between classes that underlie the system architecture.
- Primary diagram for generating codes from UML models.

# 12 – Class Modeling

## Class Diagram Notations

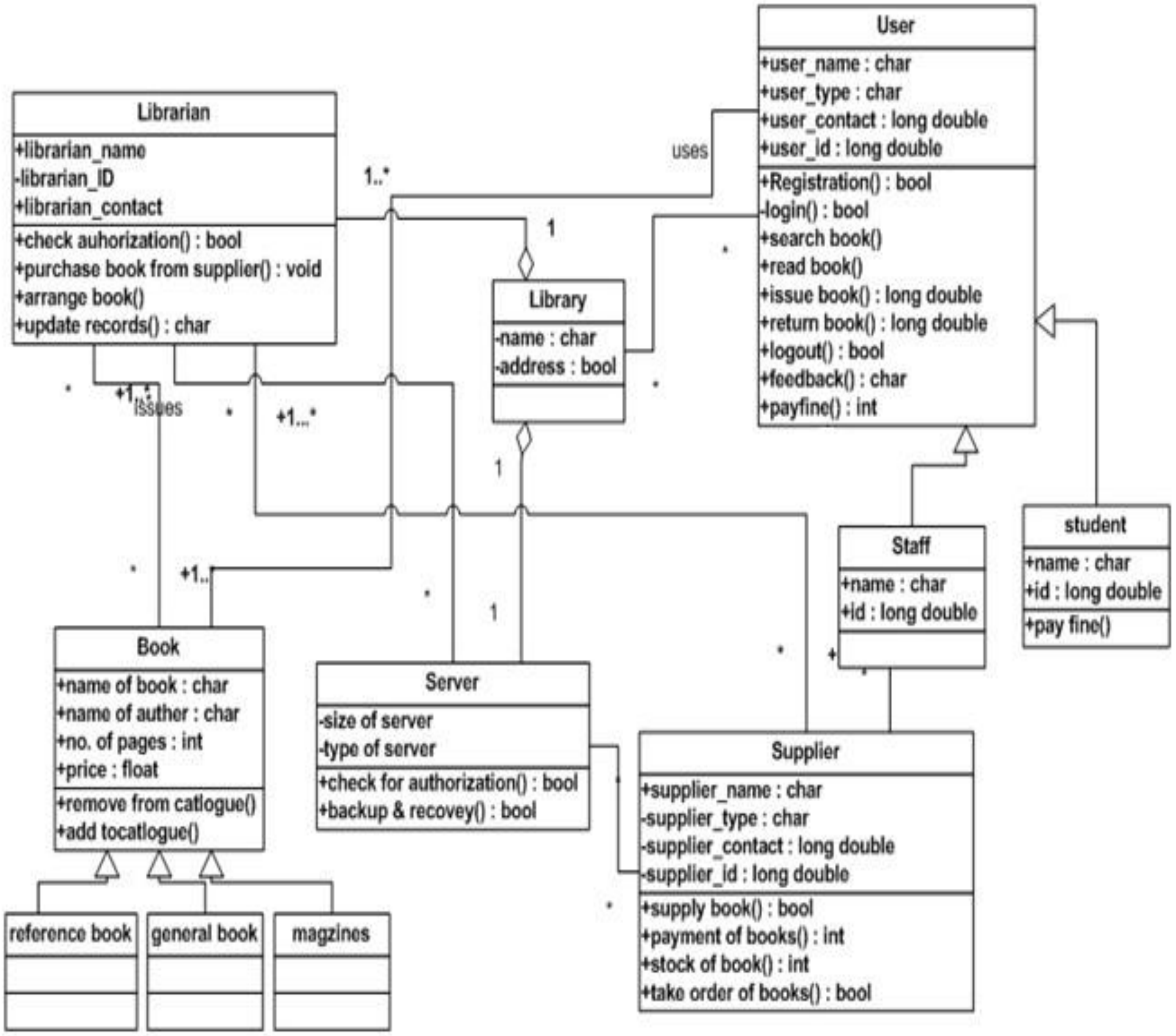
| Sr. No. | Name | Symbol | Meaning |
|---|---|---|---|
| 1 | Class |  | Class is an entity of the class diagram. It describes a group of objects with same properties & behavior. |
| 2 | Object |  | An object is an instance or occurrence of a class. |
| **3** | Link |  | A link is a physical or conceptual connection among objects. |
| **4** | Association |  | An association is a description of a links with common structure & common semantics. |
| **5** | Multiplicity | Ex. 1 to 1<br>1 to *<br>* to *<br>* to 1<br>1 to 1<br>0….2<br> | Multiplicity specifies the Number of instances of one class that may relate to a single instance of an associated class. It is a constraint on the Cardinality of a set. |
| **6** | Association class |  | It is an association that is a class which describes the association with attributes. |

| | | | |
|---|---|---|---|
| **7** | cardinality |  | It describes the count of Elements from collection. |
| **8** | ordering |  | It is used to indicate an ordered set of objects with no duplication allowed. |
| **9** | bag |  | A bag is a collection of unordered elements with Duplicates allowed. |
| **10** | sequence |  | A sequence is an ordered collection of elements with duplicates allowed. |
| **11** | qualified association |  | Qualification increases the precision of a model. It is used to avoid many to many multiplicities and it converts into one to one multiplicity. |
| **12** | generalization |  | Generalization organizes classes by their superclass and sub-class relationship. |

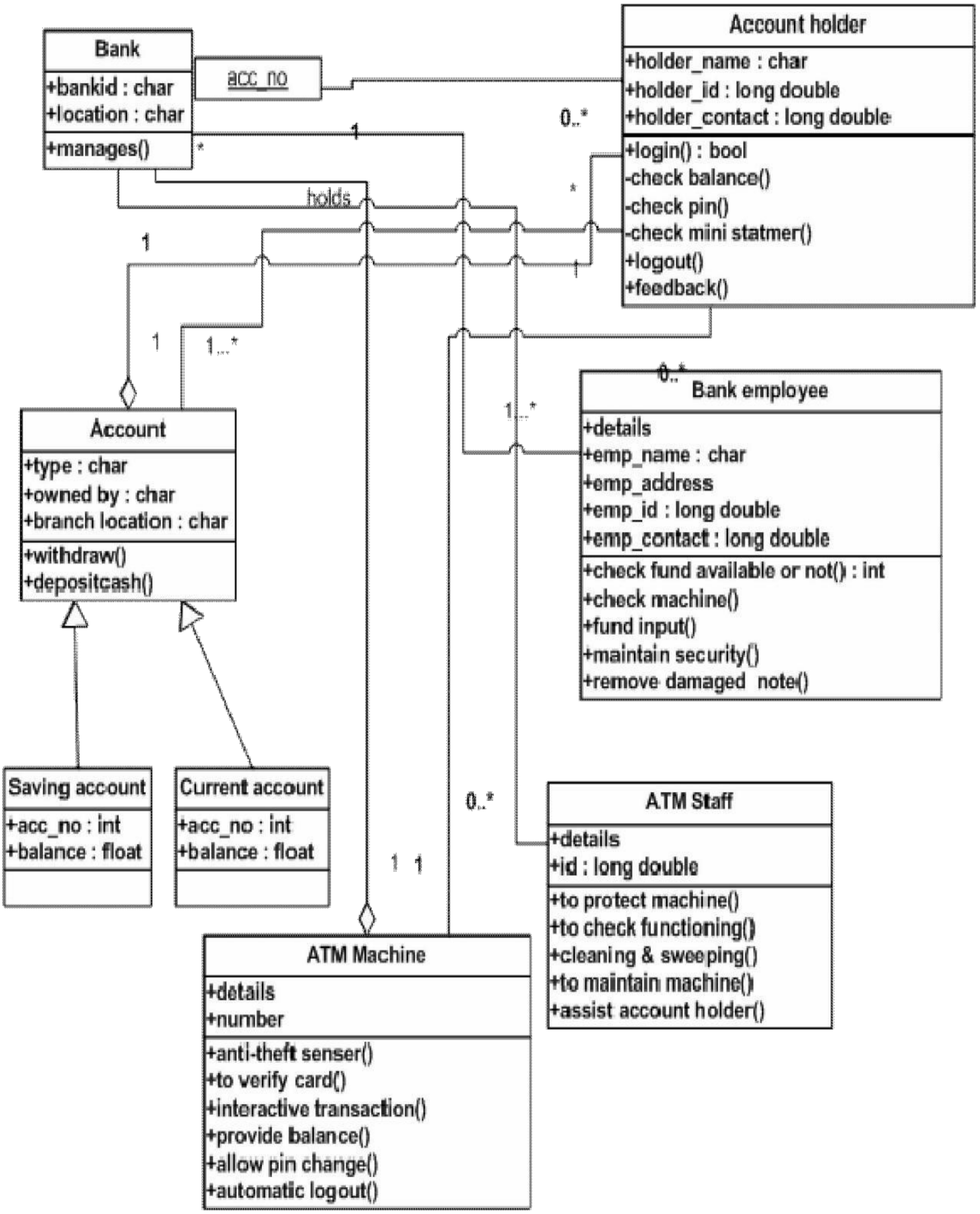
| | | | |
|---|---|---|---|
| **13** | Enumeration |  | An enumeration is a data type that has a finite set of values. |
| **14** | Aggregation |  | It is a strong form of association in which an aggregate object is made of constituent parts. |
| **15** | Composition |  | It is a form of aggregation. Composition implies ownership of the parts by the whole. |
| **16** | Abstract class |  | It is a class that has no direct instances. |
| **17** | Concrete class |  | It is a class that is intangible; it can have direct instances. Class-2 is example of concrete class |
| **18** | package |  | A package is a group of elements with common theme. |

**Examples:**

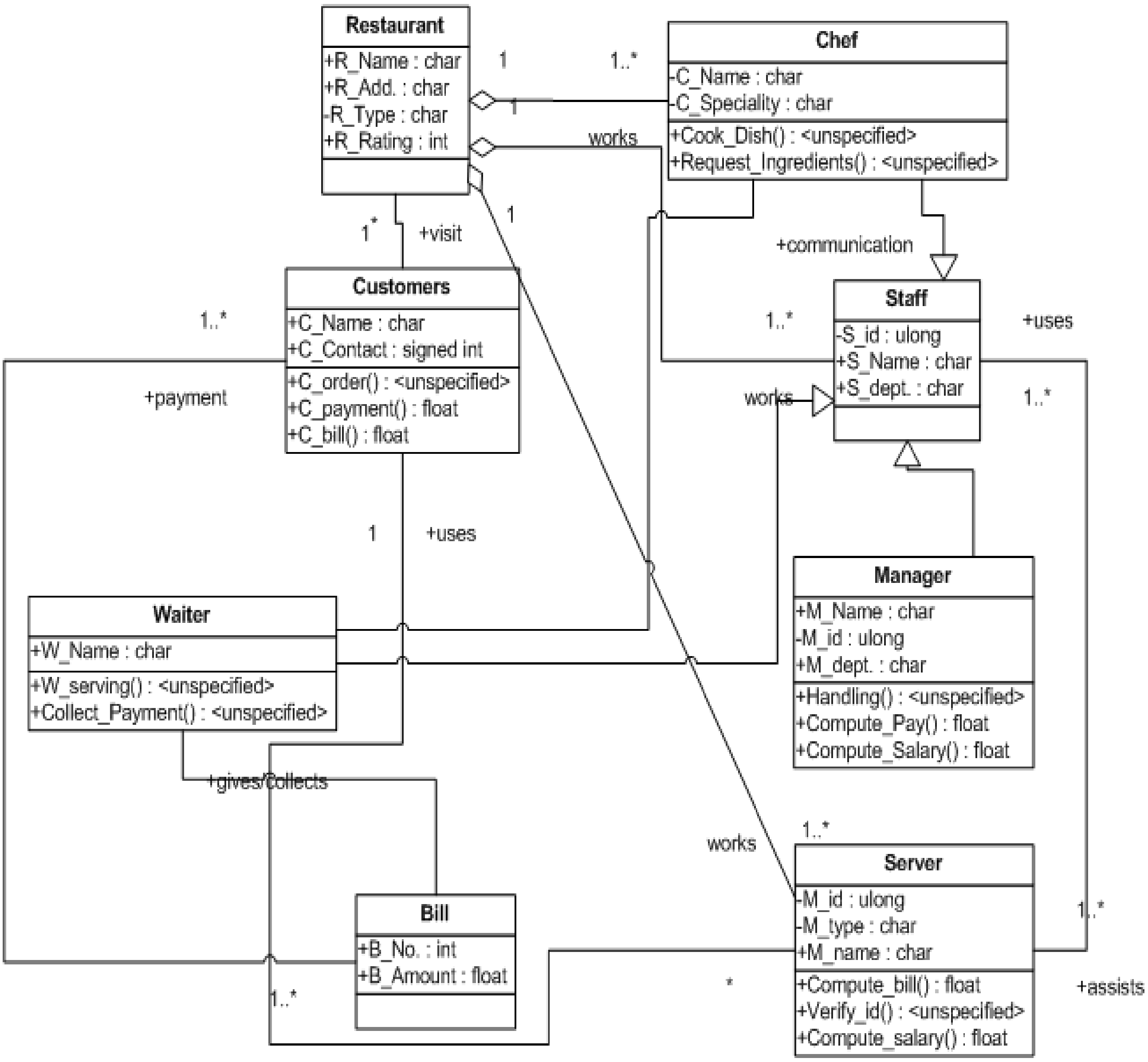
### Class Diagram for Library Management System



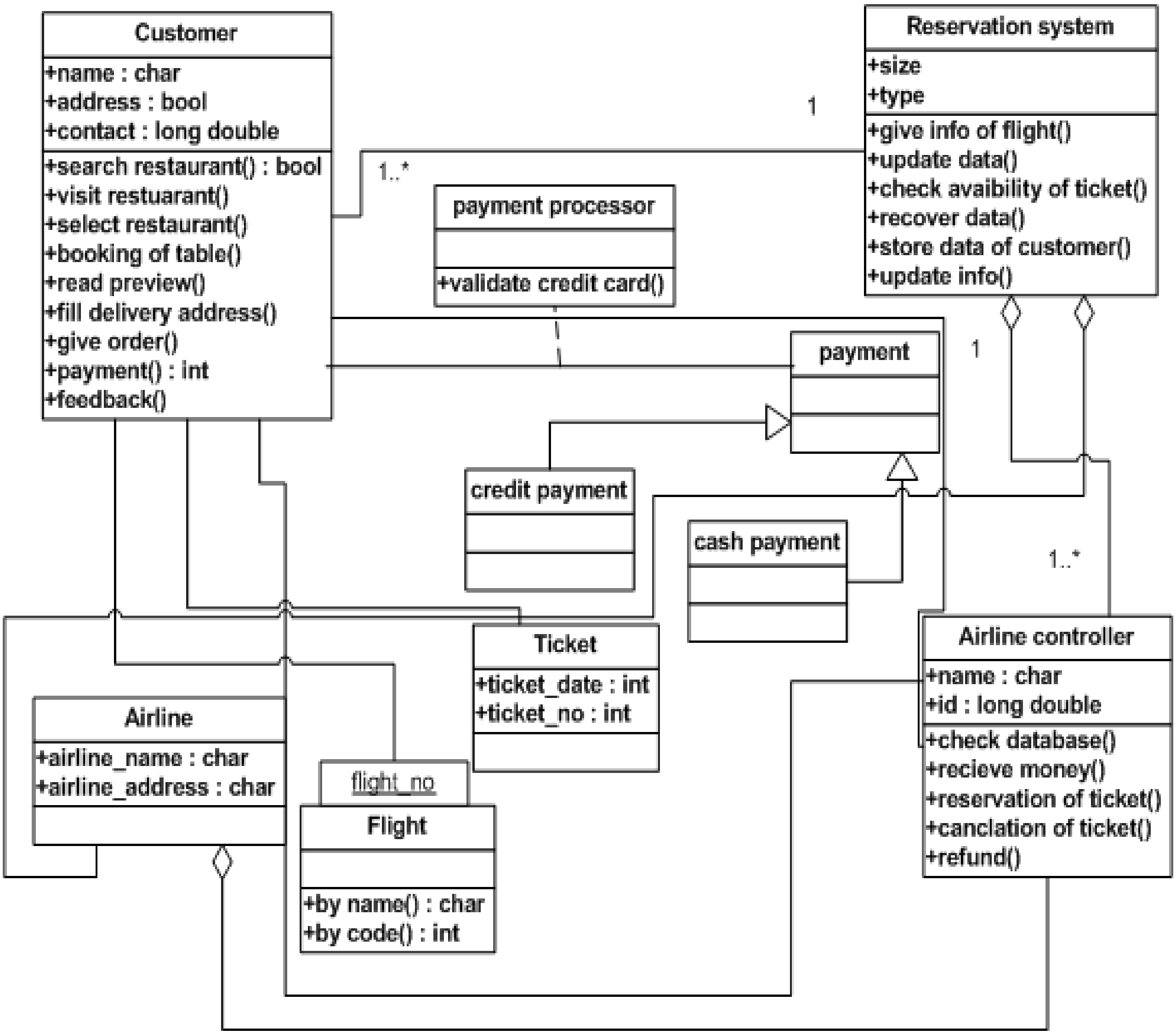
**Class Diagram for ATM**



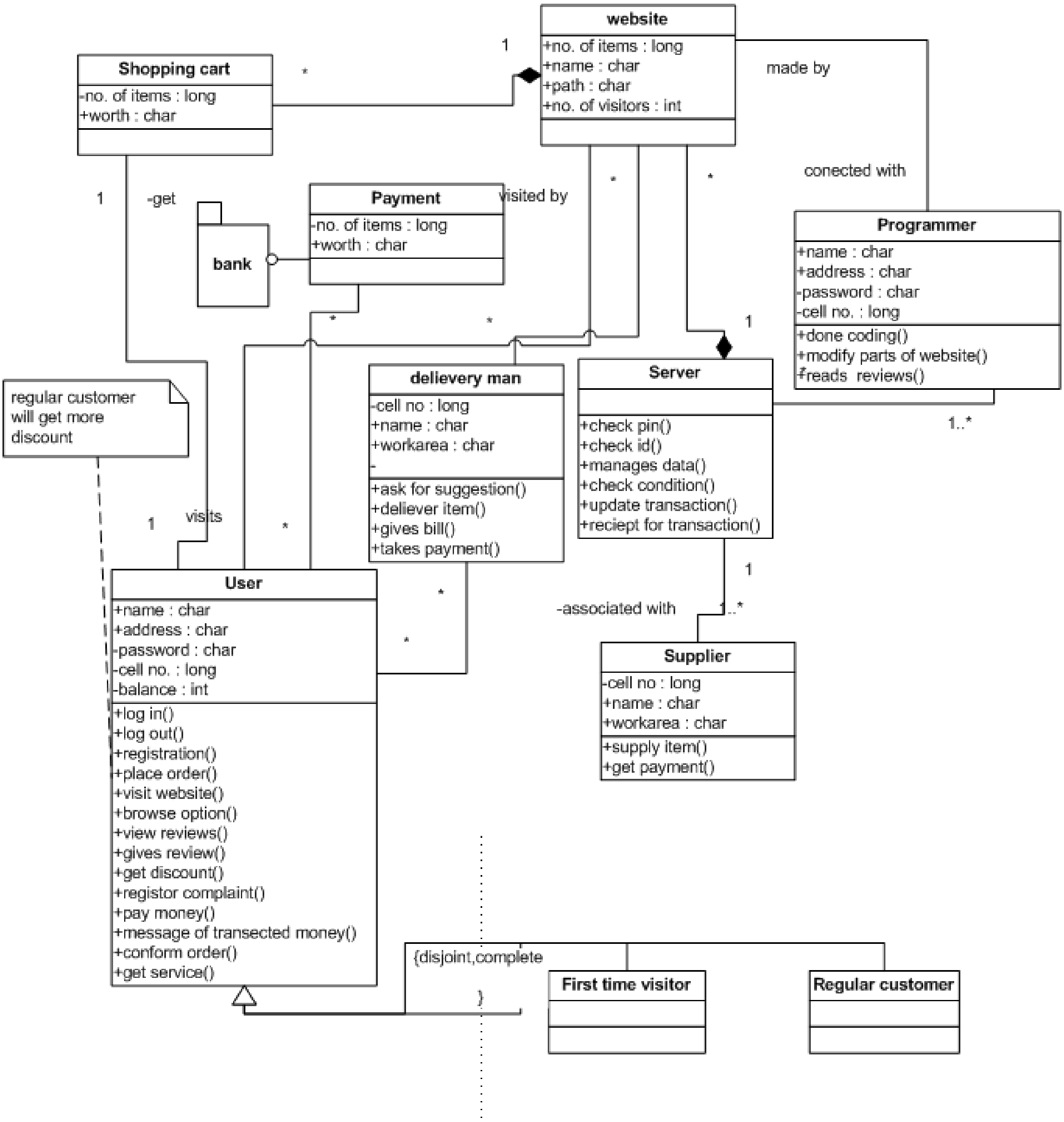
**Class Diagram for Online  Restaurant System**



**Class Diagram for Online Reservation System**



## Class Diagram for Online Shopping System



**Purpose of class diagram**

- □ Analysis and design of the static view of an application
- □ Describes responsibilities of a system.
- □ Base for component and deployment diagrams.

## Link and Association

- □ Link and Association are the means for establishing relationship among objects and classes.
- □ Link and Association often appears as verbs in problem statement.
- □ Link is a physical / conceptual connection among objects most links relate two objects, but some links relate three or more object. It is an instance of association as shown in figure below.
- □ Association is a description of a group of links with common structure and common semantics as in the class diagram shown in figure below.

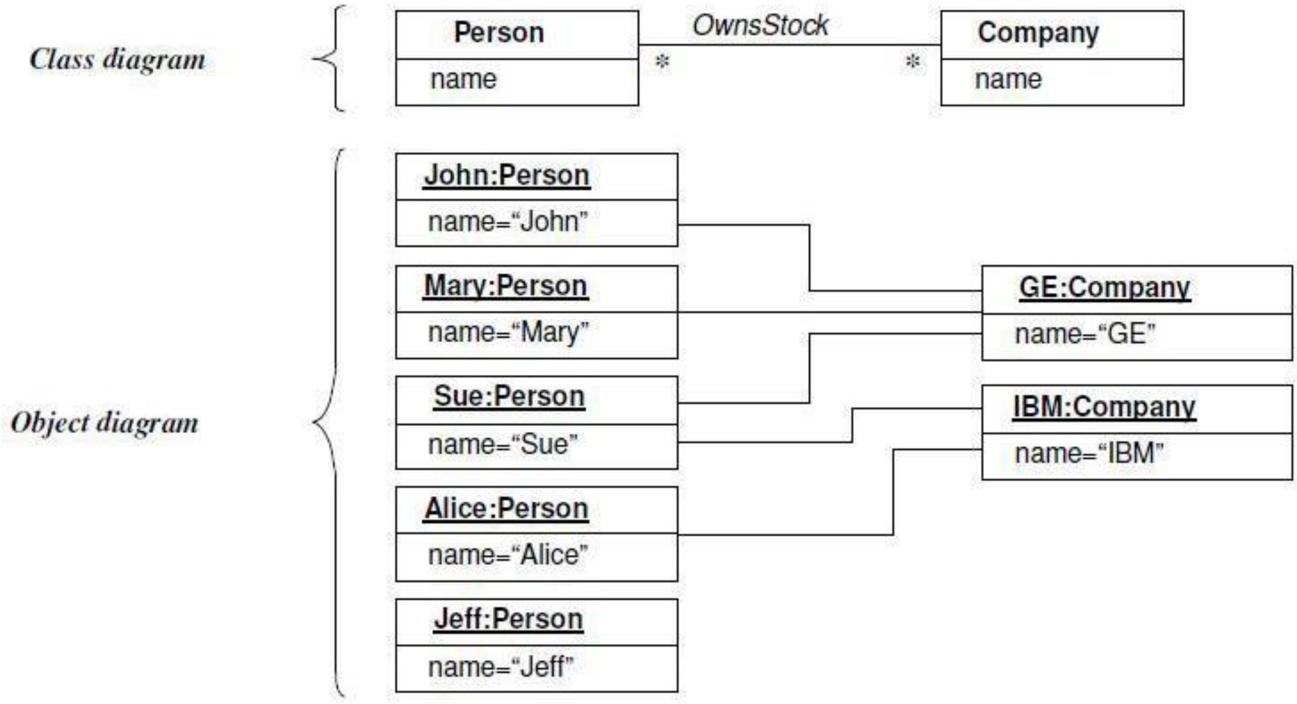


**Figure: Many-to-many Association**

- □ Associations are indirectly bidirectional.
- □ Both the direction of traversal is equally meaningful.
- □ It is the only name of the association that establishes the direction.
- □ Developers often implements associations in programming language as a references from one object to another.
- □ Associations are important, precisely because they break encapsulation.
- □ Associations cannot be private to a class, because they go beyond classes.

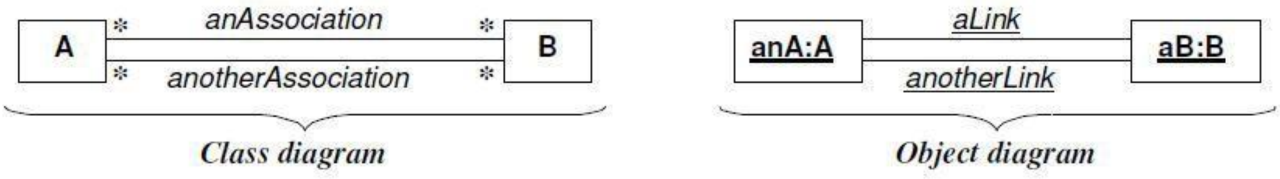


**Figure: Association Vs. Link**

## Generalization and Inheritance

- Generalization is the relationship between a class (the super class) and one or more variations of the class (sub class).
- Super class holds the common attributes, operations and associations. Subclass adds specific attributes.
- Each subclass inherits features of super class Ancestor and descendents.
- Use of Generalization serves three purposes:
    i. Support for polymorphism. (call at super class level automatically resolved)
    ii. Second purpose is to structure the description of objects. (a taxonomy is formed)
    iii. Third purpose is to enable reuse of code.
- The terms generalization, specialization and inheritance all refer to aspects of the same idea.
- Generalization: derives from the fact that the sub class generalizes to super class
- Specialization: refers to the fact that the subclasses refine or specialize the super-class.
- Inheritance: Is the mechanism for sharing attributes, operations, and associations via generalization specialization relationship which is useful for parent child relationship.