Class Diagram

A class diagram models the static view of a system. It comprises of the **classes**, **interfaces**, and **collaborations of a system**; and the **relationships** between them.

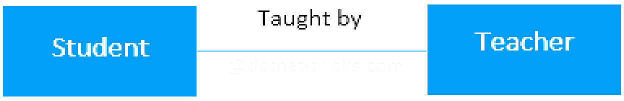
Relationships

- ✓ Association
- ✓ Aggregation
- ✓ Composition

Association

It represents a relationship between two or more objects where all objects have their own lifecycle and there is no owner. The name of an association specifies the nature of relationship between objects. This is represented by a solid line.

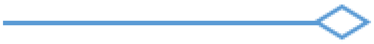
Let’s take an example of relationship between Teacher and Student. Multiple students can associate with a single teacher and a single student can associate with multiple teachers. But there is no ownership between the objects and both have their own lifecycle. **Both can be created and deleted independently**.



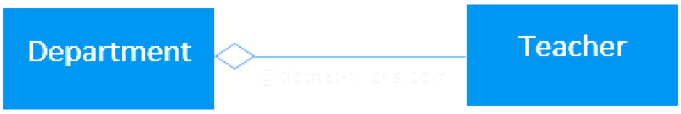
Association

Aggregation

It is a specialized form of Association where all object have their own lifecycle but there is ownership. This represents **“whole-part or a-part-of” relationship**. This is represented by a hollow diamond followed by a line.



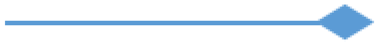
Let’s take an example of relationship between Department and Teacher. A Teacher may belongs to multiple departments. Hence Teacher is a part of multiple departments. **But if we delete a Department, Teacher Object will not destroy**.



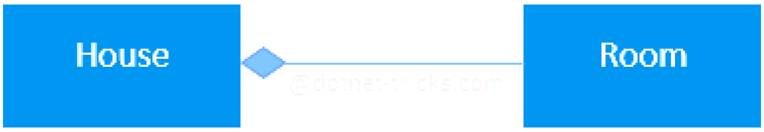
Aggregation

Composition

It is a specialized form of Aggregation. It is **a strong type** of Aggregation. In this relationship child objects does not have their lifecycle without Parent object. If a parent object is deleted, all its child objects will also be deleted. This represents **“death” relationship**. This is represented by a solid diamond followed by a line.



Let’s take an example of relationship between House and rooms. House can contain multiple rooms there is no independent life of room and any room cannot belongs to two different house **if we delete the house room will automatically delete.**



Composition

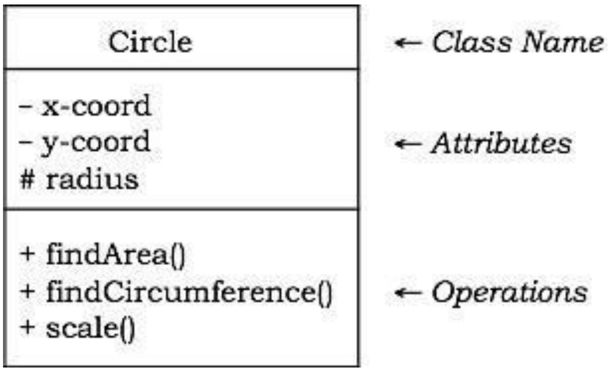
The visibility of the attributes and operations can be represented in the following ways

Public − A public member is visible from anywhere in the system. In class diagram, it is prefixed by the symbol „+‟.

Private − A private member is visible only from within the class. It cannot be accessed from outside the class. A private member is prefixed by the symbol „−‟.

Protected − A protected member is visible from within the class and from the subclasses inherited from this class, but not from outside. It is prefixed by the symbol „#‟.

Example − Let us consider the Circle class introduced earlier. The attributes of Circle are x-coord, y-coord, and radius. The operations are findArea(), findCircumference(), and scale(). Let us assume that x-coord and y-coord are private data members, radius is a protected data member, and the member functions are public. The following figure gives the diagrammatic representation of the class.

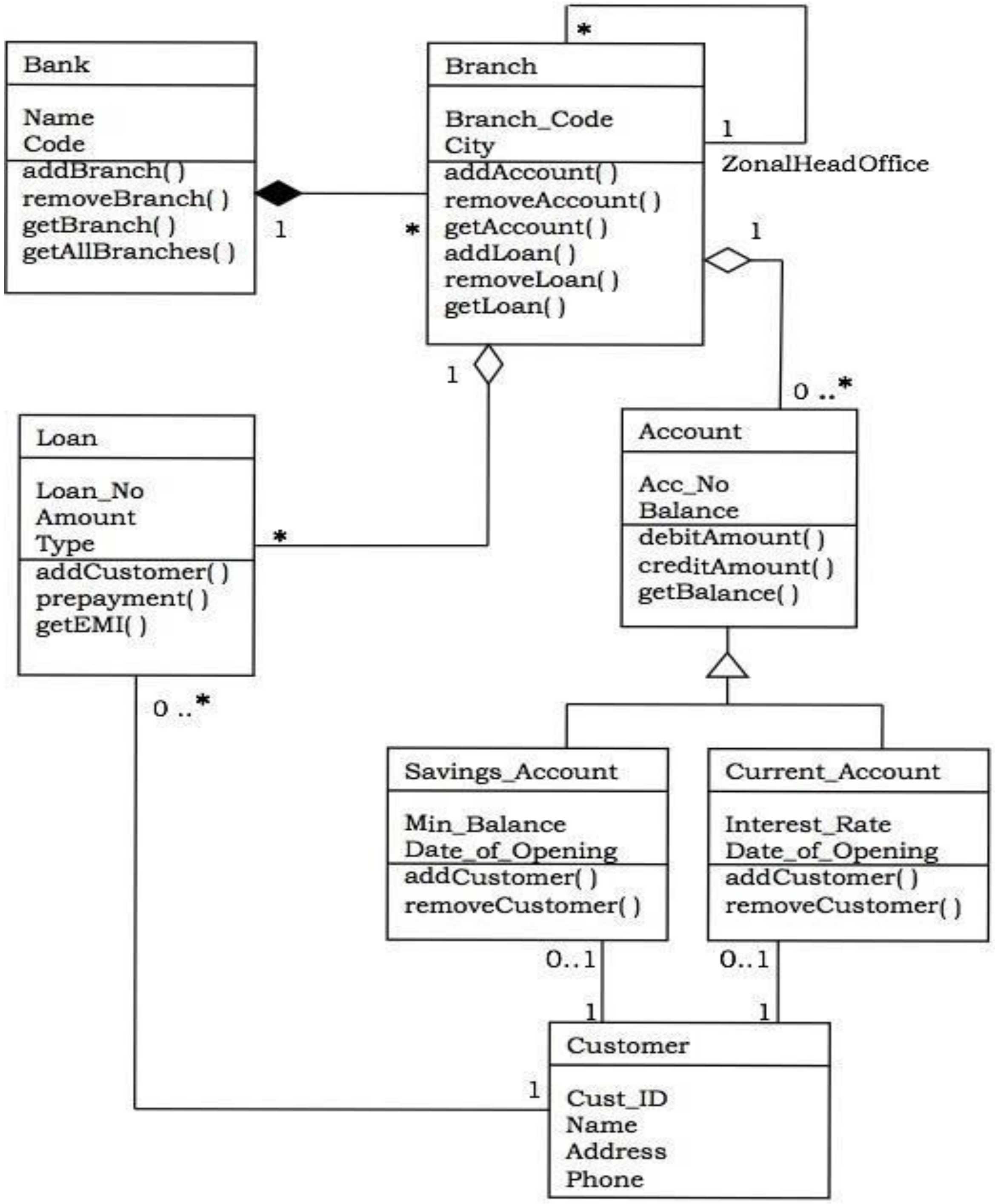


Class Diagram

Let us consider a simplified Banking System.

A bank has many branches. In each zone, one branch is designated as the zonal head office that supervises the other branches in that zone. Each branch can have multiple accounts and loans. An account may be either a savings account or a current account. A customer may open both a savings account and a current account. However, a customer must not have more than one savings account or current account. A customer may also procure loans from the bank.

The following figure shows the corresponding class diagram.



Use case diagrams

Use case diagrams comprise of −

- Use cases
- Actors
- Relationships like dependency, generalization, and association

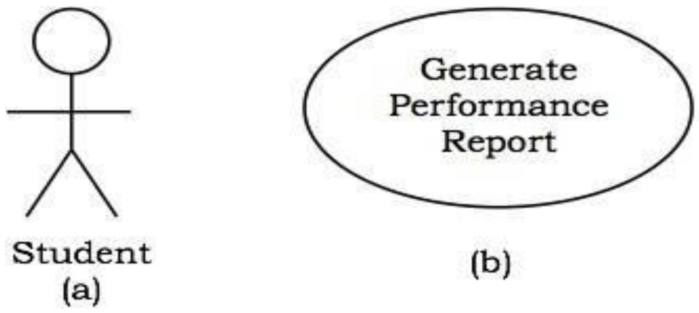
1. USE CASE

A use case **describes the actions of a system performs** yielding visible results. It shows the interaction of things outside the system with the system itself. Use cases may be applied to the whole system as well as a part of the system.

2. ACTOR

An actor represents the roles that the users of the use cases play. An actor may be a person (e.g. student, customer), a device (e.g. workstation), or another system (e.g. bank, institution).

The following figure shows the notations of an actor named Student and a use case called Generate Performance Report.



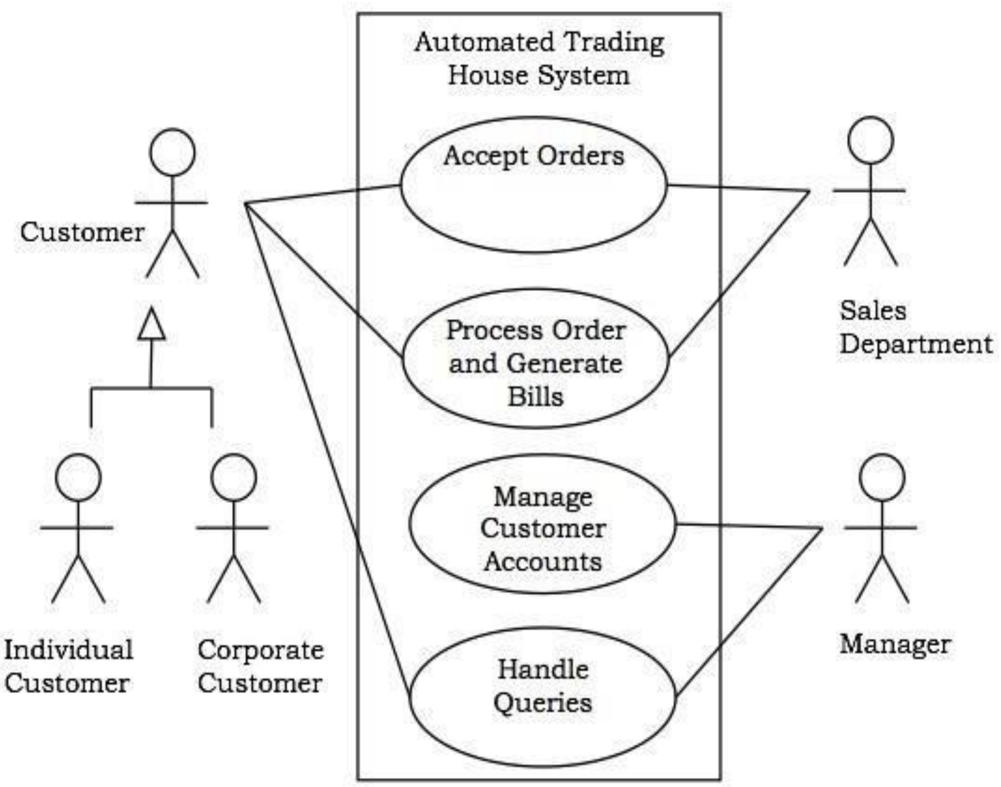
Example Use Case:

Let us consider an Automated Trading House System. We assume the following features of the system −

1- The trading house has transactions with two types of customers, individual customers and corporate customers.

2- Once the customer places an order, it is processed by the sales department and the customer is given the bill.

3- The system allows the manager to manage customer accounts and answer any queries posted by the customer.



Sequence Diagrams

Sequence diagrams are interaction diagrams **that illustrate the ordering of messages according to time.**

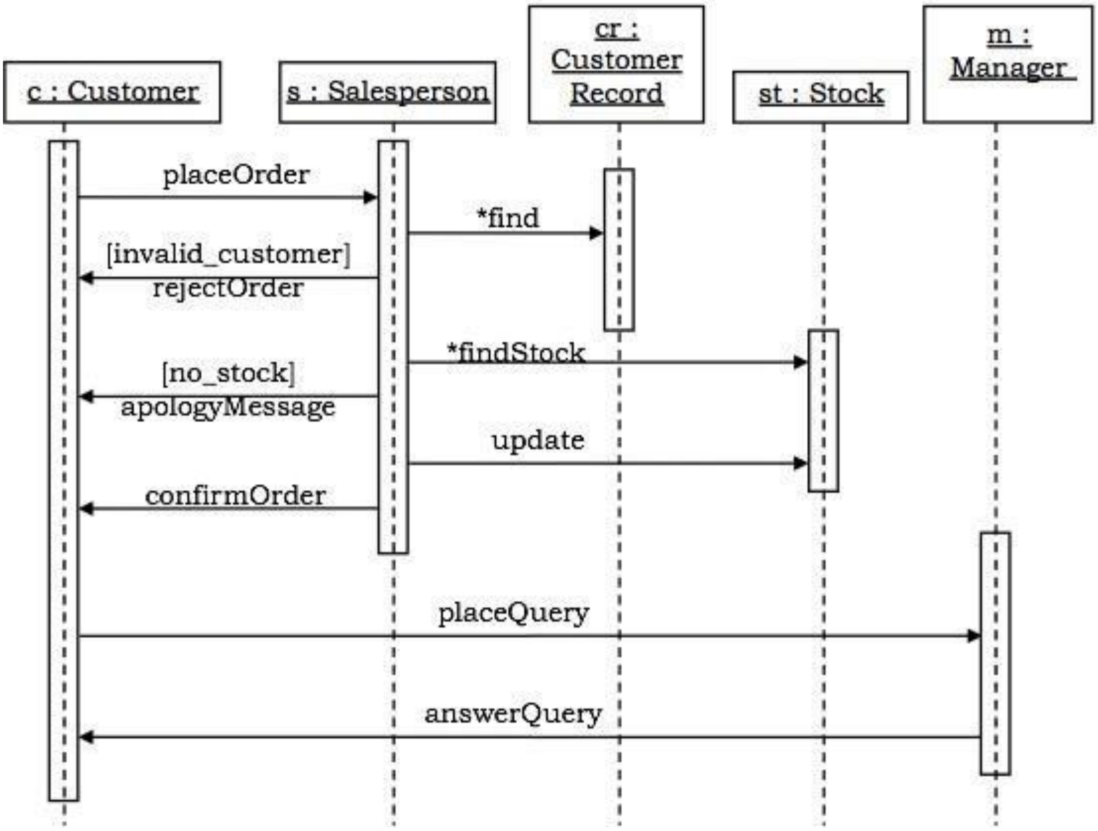
Let us consider an Automated Trading House System. We assume the following features of the system −

1- The trading house has transactions with two types of customers, individual customers and corporate customers.

2- Once the customer places an order, it is processed by the sales department and the customer is given the bill.

3- The system allows the manager to manage customer accounts and answer any queries posted by the customer.

Example − A sequence diagram for the Automated Trading House System is shown in the following figure.


c : Customer
s : Salesperson
cr : Customer Record
st : Stock
m : Manager
placeOrder
*find
[invalid_customer] rejectOrder
*findStock
[no_stock] apologyMessage
update
confirmOrder
placeQuery
answerQuery