Object Orientated Class Relationships

Generalization is the process of extracting shared characteristics from classes, and combining them into a generalized superclass. Where characteristics can be shared though out attributes, associations, or methods

Inheritance enables new objects to take on the properties of existing objects. A class that is used as the basis for inheritance is called a superclass or base class. A class that inherits from a superclass is called a subclass or derived class

Realisation is the relationship between the Interface and a class that contains all of the details for the interface and how it will implement them. In simple terms realisation is the relationship between interface and the implementing class.

Dependency is the relationship between two or more objects that in which the objects depend on each other. Also they are linked together where if one thing changes on a single object the others will be impacted as well. The relationship is created when making the UML diagram using dashed arrows.

Aggregation can be confused with composition in many languages as many of the coding languages do not distinguish the difference between the two. So it is up to the user to make the decision. Aggregation is all to do with hiding certain classes within other classes and that they are only accessible though the selected class that it is being stored within. Aggregation also involves the dynamic creation of sub objects inside an object, in the process, expanding the properties and methods of that object.

Composition is the idea that classes should achieve the behaviours and be able to reuse code from their class that are within that have been implemented to do its selected function rather than inheritance from a base or a parent class. It goes to its composition source within. But with composition when the owner class is destroyed everything within also goes as it is only used within that class and cannot be used without the owner class.

Loose Coupling is mostly an independent class that will only get its knowledge from other classes by looking though its interface. It can only see what the other classes want it to see by displaying it onto there interfaces and keeping other private information safe within which the other classes cannot see.

Tight Coupling is when a selected amount of classes is highly dependent on each other. Meaning that they often change together and can see what the other classes are storing and be able to pull from them using their information when they need it.

Low cohesion is that the class does a lot of different actions and is not focused in one area so that it can be spread out and ran in many places or forms

High cohesion is when a class has been designed to focus on one thing and to put all of its time and action into completing it.

An object is a component that contains properties and methods needed to make a certain type of data useful otherwise the object would be no help. An object's properties are what it knows and its methods are what it can do.

A Sub Object is an object inside another object. This is the reasoning behind the name. whenever an instance of the object is created each of the sub objects within are also created. On the other hand, if you are to delete an object all of the sub objects will also be deleted.

A container is a class or a data structure that is designed to just hold objects. It can store objects and sub objects within itself..
You are able to look though this and seek out all the information you need within your objects and sub objects