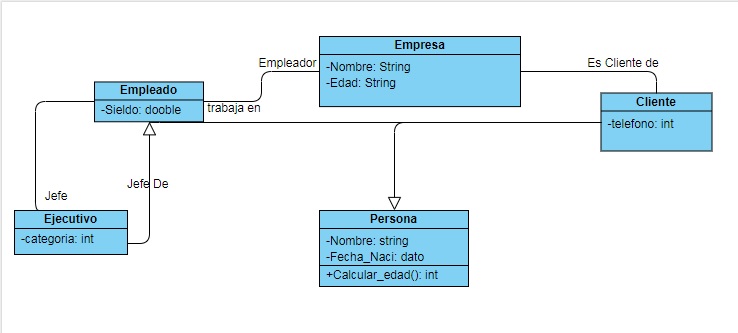
**PRIMER EJERCICIDE DIAGRAMA DE CLASES**

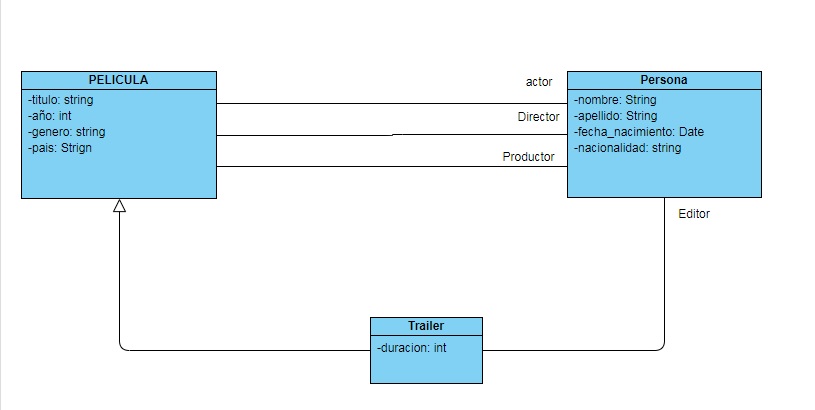
En un diagrama de clases represente el siguiente ejercicio

1. Un almacén necesita almacenar toda la información de sus empleados los clientes y datos del almacén y caracterizar ambos por nombre y edad
2. Los empleados tienen que tener sueldo los empleados con mejores cargos tienen que estar categorizados, y un conjunto de empleados subordinados
3. De los clientes se necesita conocer su teléfono de contacto 

**SEGUNDO DIAGRAMA DE CLASES**

Representa mediante un diagrama de clases la siguiente especificación sobre las personas que participan en una película

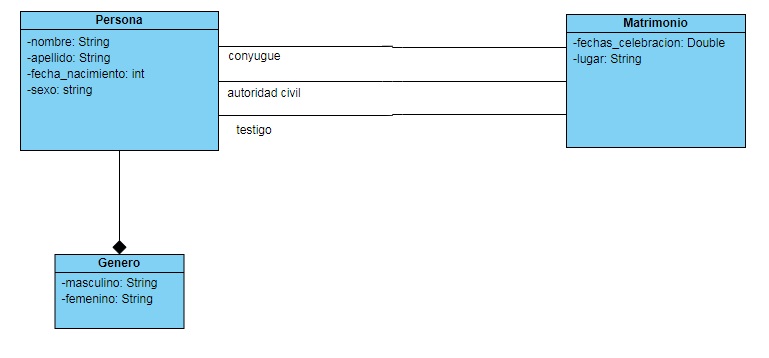
* De cada película se almacena el título, la sinopsis, el año, el género al que pertenece (drama, comedia, acción, terror, romance, aventura, scifi) y el país. "
* Sobre las personas que participan en la película es necesario conocer el nombre, los apellidos, la fecha de nacimiento y la nacionalidad.
* Una persona puede participar en una película como actor, director, productor o guionista.
* Una película tiene al menos un director y un guionista.
* Una persona se considera actor si ha actuado al menos en una película.
* Una película puede tener asociados varios trailers que son editados por una o más personas. No puede existir el trailer de una película hasta que existe la película.



**TERCER EJERCICIO**

Representa mediante un diagrama de clases la siguiente especificación sobre los matrimonios civiles realizados en la ciudad de Salamanca.

* De cada matrimonio se almacena la fecha, el lugar de la celebración y los datos personales nombre, apellidos, edad, sexo de los contrayentes.

****

**QUE ES UN PATRON Y DISEÑO DE SOFTWARE**

Los patrones y diseños de software son las soluciones a los problemas típicos y recurrentes que nos encontramos a la hora de generar el desarrollo de una aplicación ya sea web Android o cualquier base que uses

Es muy común que la aplicación que se esté desarrollando sea única en el mundo pero siempre se va a encontrar con problemas o partes con muchas similitudes a otras aplicaciones entre ellas podemos encontrar acceso a datos, creación de objetos, operaciones entre sistemas y muchas cosas más.

En lugar de quemarnos la cabeza con los problemas que tenemos, podemos usar las soluciones que ya se han usado en algunas aplicaciones que hayan tenido algún problema similar o que tuvieran un estancamiento parecido en el pasado podemos tomar esos problemas y soluciones como ejemplo y encontrar una solución fácil y funcional que ya sueles estar documentadas u con el problema solucionado por multitudes de programadores que ya en algún momento le encontraron una solución a este problema

**QUE TIPO DE PATRONES EXISTEN ?**

Existen varios tipos de patrones y diseños de software de pronto la más extendida o más larga es agruparlos a todos por propósito, en este caso tendríamos las siguientes 3 categorías

1. **Patrones Creacionales:** son más que todo para el tema de instanciar objetos y así poder separar el tema del cliente y los objetos que utilice el programa o la aplicación, de esta manera podemos separar la lógica de creación del objeto y se puede generar un encapsulamiento
2. **Patrones de Comportamiento:** Se utilizan a la hora de ver como las clases y los objetos ya establecidos en el programa que estamos desarrollando se comunican o cómo interactúan entre ellos
3. **Patrones Estructurales:** este patrón se usa para la creación de clases u objetos que se desean incluir en la estructura más difícil o compleja del patrón o programa

**ARQUITECTURA DE SOFTWARE**

En el mundo de la programación cada vez es más común escuchar el término de arquitectura de software

El termino arquitectura de software se refiere a una estructuración del sistema que idealmente es creado en etapas tempranas del desarrollo. Esta estructuración genera una estructuración de alto nivel del sistema en cuestión que usualmente suele tener dos propósitos primarios que son satisfacer los atributos de calidad entre ellos están diseño, seguridad y modificalidad entre otros y pueden servir como una guía para la persona que esté desarrollando el software las decisiones críticas que se tomas sobre el diseño general del software se deben tener en cuenta desde un principio. El no acogerse a este plan desde un principio es posible que se genere una limitación bastante sebera en el producto final que puede llegar a la insatisfacción del cliente final del producto generando que cubra las expectativas y necesidades del cliente, teniendo también en cuenta que el costo de la corrección teniendo en cuenta el problema que se va a generar al intentar corregir la arquitectura como tal son costos demasiado elevado tanto de dinero como de tiempo requerido por esta razón la arquitectura del software juega un papel fundamental en el momento del desarrollo de la aplicación

**CARACTERISTICAS DE ARQUITECTURA DE SOFTWARE**

La característica principal de la arquitectura de software es que es la columna vertebral del sistema de software en cuestión en general es la que se encarga de dar acceso y denegar el acceso a ciertos atributos del sistema que se destacan por llevar una buena confiabilidad y un buen rendimiento del software además que puede ser un modelo que se puede reutilizar en cualquier momento que se desee hacer, que se puede transferir de un sistema a otro y que como tal también general y representa un medio de comunicación entre los dos sistemas que también puede generar un intercambio entre los dos desarrolladores siempre y cuando se tenga un fin en común

**QUE ES TDD**

Partamos del punto que es la forma fácil de decir **Test Driven Development** o en español desarrollo dirigido por pruebas es una práctica de programación que como su nombre lo indica consiste es escribir un código de prueba generalmente se escribe de forma unitario para después poder tener una base y poder escribir el código fuente teniendo en cuenta que el código que se escriba tiene y debe pasar las pruebas efectuadas satisfactoriamente y para finalizar se tiene que generar el código escrito y lo que podemos conseguir con este tipo de prácticas entre otras cosas en código más fácil de entender un código más robusto un código mantenible y un código con una mayor rapidez en el tema de su desarrollo