UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ALUMNO
SILVERIO MARTÍNEZ ANDRÉS

MATERIA
BASES DE DATOS

PROFESOR
FERNANDO ARREOLA FRANCO

GRUPO
1

TAREA 1
MODELO ORIENTADO A OBJETOS Y MODELOS NOSQL

SEMESTRE 2024 - 2

Modelo Orientado a Objetos

El modelo orientado a objetos surge en la década de los 90´s, la cual es considerada la “era de la programación orientada a objetos”, principalmente porque este tipo de programación empezó a agarrar fuerza por esos años. Los usuarios demandaban programas y entornos de trabajo amables y fáciles de utilizar, naciendo de ahí el modelado orientado a objetos.
El modelo orientado a objetos es una técnica utilizada en el desarrollo de software. Esta permite representar de manera eficiente, y a su vez, de manera estructurada, los objetos, atributos, relaciones y comportamientos de un sistema. Cuando se aplica esta metodología, es más fácil el desarrollar y construir sistemas más flexibles, modulares y fáciles de mantener. Dicho modelado orientado a objetos se basa en la representación de entidades del mundo real como “objetos” que tienen ciertos atributos, así como métodos o comportamientos. Los objetos que se representan en el modelado interactúan entre ellos a través de mensajes, permitiendo así, simular sistemas de mayo complejidad. Representa al sistema como una colección de objetos interconectados.
La ventaja que tiene un modelo orientado a objetos a diferencia que un modelo relacional, es que el modelo orientado a objetos, los atributos de cada objeto están disponibles de inmediato, aunque también puede llegar a ser complejo por esto.

Algunas de las ventajas de utilizar el modelo orientado a objetos son:

- **Reutilización de código.** Cuando se utilizan clases y objetos, entra la posibilidad de volver a utilizar código, debido a que se pueden crear instancias de una clase en partes diferentes del sistema.
- **Flexibilidad y mantenibilidad.** La estructura modular del modelado orientado a objetos, hace que sea más fácil la incorporación de cambios y mejoras en el sistema sin afectar parte de este.
- **Abstracción.** Representar entidades del mundo real, lo más fiel posible, facilitando en la comprensión y análisis.
- **Encapsulación.** Permite ocultar la implementación interna de un objeto, brindando mayor seguridad y simplicidad en el desarrollo
- **Herencia y polimorfismo.** Ayudan a compartir atributos y comportamientos entre clases relacionadas, haciendo que se agilice el desarrollo, además de tener una mejor organización en el código.
- **Persistencia.** Consiste en la posibilidad de recuperar datos en el futuro, es decir, los datos se almacenan a pesar del término del programa de aplicación.

Como se puede apreciar, estas ventajas que tiene el modelado orientado a objetos, son los mismos que los principios de la programación orientada a objetos, la cual se llevó a cabo en semestres anteriores de la carrera.

Mientras que, algunas de las desventajas del modelado orientado a objetos, son:

- **Complejidad de aprendizaje.** Si bien, aprender el modelado orientado a objetos es fácil a estas alturas de la carrera, para cualquier otra persona que no tenga ni idea de como funciona lo orientado a objetos, puede que le cueste más aplicar este modelado.
- **Mayor consumo de memoria.** Como el modelado esta orientado a objetos, es normal que se requiera demasiado espacio en la memoria para poder almacenar a los objetos, y los datos que los conforman.
- **Resistencia al cambio.** Una vez definida una jerarquía de clases, y las relaciones entre los objetos, es muy difícil el realizar cambios significativos sin afectar otras partes importantes del sistema

Algunos de los pasos principales que componen el proceso de modelado orientado a objetos son:

- **Identificación de objetos.** En esta etapa, se identifican los objetos que conformarán parte del sistema. A su vez, como ya se mencionó, los objetos tienen un estado, un comportamiento y una identidad.
- **Definición de clases.** Cuando ya se tienen ubicados los objetos a utilizar, se empiezan a definir a qué clases pertenece cada objeto. En donde la clase definirá las propiedades y comportamientos comunes a un conjunto determinado de objetos.
- **Relaciones entre clases.** Las clases tienen la capacidad de relacionarse entre sí mediante diferentes tipos de relaciones, como la asociación, la composición y la herencia. Las relaciones de clases permiten establecer cómo interactúan los objetos y cómo se relacionan entre sí en el sistema.
- **Diagramas de clases.** Ayudan a representar visualmente las clases y las relaciones entre ellas. Tienen una visión global del sistema, y facilitan la comunicación entre los miembros del equipo de desarrollo.

El caso de uso de un modelado de procesos de negocio puede ser ejemplificado por la siguiente imagen:

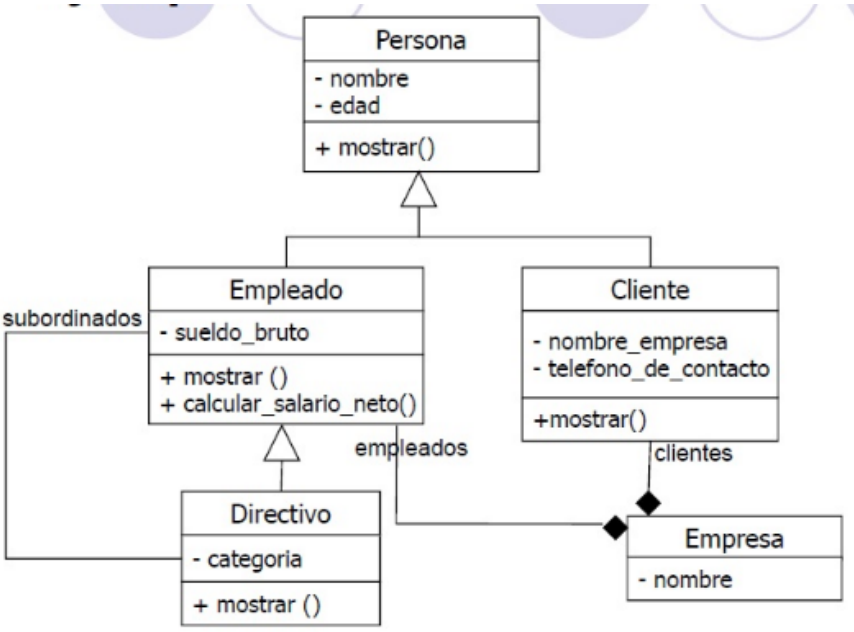

Imagen 1. Caso de uso de un modelado orientado a objetos

Donde la parte de hasta arriba de cada rectángulo es la entidad, los que están acotados con un guión corto son los atributos, y los que tienen un símbolo de suma, son los métodos que realizan cada uno de las entidades. Cada uno de los objetos está asociado mediante líneas de unión, para saber cómo están relacionadas entre sí.

Modelos NoSQL

La historia de los modelos NoSQL empieza en la década de 1970, con el surgimiento de SQL, ya que gracias la creación de SQL, en el año 1998, Carlo Strozzi acuñaría el término de NoSQL, llamando a su base de datos “Base de datos relacional de código abierto Strozzi NoSQL”. Strozzi aclaraba que la forma en la que su base de datos funcionaba de manera diferente a las bases de datos relacionales en conjunto. Posteriormente, en el año 2000, se reanudaría el desarrollo de NoSQL, teniendo como objetivo el superar las limitaciones de SQL, en cuanto a escalabilidad y el potencial para la recopilación de datos multi estructurados.
En sí, los modelos SQL, fueron desarrollados con la esperanza de resolver los problemas clásicos de las bases de datos SQL.
Los modelos NoSQL se refieren a tipos de modelos no relacionales que almacenan datos en un formato distinto a las tablas relacionales. Los modelos NoSQL se utilizan de forma generalizada en aplicaciones web en tiempo real y big data, debido a que una de sus principales ventajas son los elevados niveles de escalabilidad y disponibilidad que tienen.
Los modelos NoSQL tienen una naturaleza ágil, debido a su rápida adaptación a los requisitos que suelen estar en constante cambio. También, este modelo permite almacenar datos de forma más intuitiva y fácil de entender, o mejor dicho, de manera más cercana a la forma en la que las aplicaciones utilizan los datos. Alguna de las diferencias que tienen lo modelos NoSQL

respecto a los modelos SQL y los modelos de datos relacionales, es que, como ya se mencionó, NoSQL es una forma de almacenamiento no estructurado

Entre las ventajas que encontramos de utilizar un modelo NoSQL, están:

- **Flexibilidad.** Permiten un almacenamiento de datos más libre sin la rigidez de los esquemas predefinidos de los modelos SQL. Esto impulsa la innovación y el rápido desarrollo de aplicaciones.
- **Escalabilidad.** Pueden escalarse horizontalmente utilizando hardware básico, lo que permite admitir un mayor volumen de tráfico para satisfacer la demanda sin tiempo de inactividad. Siendo
- **Alto rendimiento.** Su arquitectura garantiza tiempos de respuesta rápidos y predecibles de milisegundos de un solo dígito. Son útiles en aplicaciones que recopilan terabytes de datos todos los días y que requieren una experiencia de usuario altamente interactiva.
- **Disponibilidad.** Replican datos de forma automática en múltiples servidores, centros de datos o recursos en la nube, lo que minimiza la latencia de los usuarios.
- **Alta funcionalidad.** Están diseñadas para almacenes de datos distribuidos que precisan una capacidad de almacenamiento de datos extremadamente grande. Son ideales para big data, aplicaciones web en tiempo real, cliente 360, compras en línea, juegos en línea, Internet de las cosas, redes sociales y aplicaciones de publicidad en línea.

Y, entre las desventajas de utilizar modelos NoSQL, están:

- **Soporte bajo para consultas complejas.** A diferencia de SQL, el utilizar NoSQL implica que no puedes hacer consultas complejas que impliquen múltiples tablas y relaciones.
- **Incompatibilidad con consultas SQL.** Muchas bases de datos NoSQL no son compatibles con consultas SQL, lo que requiere un lenguaje de consulta manual y puede hacer que los procesos sean más lentos y complejos.
- **Escasez de herramientas y recursos.** A pesar de que los modelos NoSQL sean más sencillos que los SQL, no cuentan con demasiadas herramientas para su elaboración y menos recursos adecuados para su correcta implementación.
- **Falta de funciones de fiabilidad.** A diferencia de las bases de datos relacionales, la mayoría de las bases de datos NoSQL no admiten características de fiabilidad como atomicidad, consistencia, aislamiento y durabilidad. En su lugar, ofrecen consistencia a cambio de rendimiento y escalabilidad.

Los tipos de modelos de NoSQL que podemos encontrar son:

- **Valor - clave**
  Es el tipo más flexible de base de datos NoSQL, ya que la aplicación tiene un control completo sobre lo que se almacena en el campo de valor sin restricción alguna. Se basa

en una tabla de solamente dos columnas. En una de estas columnas, se guarda el valor, y en otra, la llave o clave que es el identificador del valor, y la cuál es única. El valor puede ser una cadena de caracteres, un número entero, e inclusive objetos complejos. Siendo ArangoDB, Apache Ignite, Oracle NoSQL Database, Couchbase, Dynamo, Redis y Riak los que utilizan este modelo.

- **Documentos**
  También conocida como modelo orientado a documentos o almacenamiento de documentos, se utilizan para almacenar, recuperar y gestionar datos con semiestructura. Siendo Apache CouchDB, ArangoDB, BaseX, Clusterpoint, Couchbase, Cosmos DB, IBM Domino, MarkLogic, MongoDB, OrientDB, Qizx y RethinkDB los que utilizan este modelo.
- **Gráfico o de grafos**
  Este modelo organiza los datos como nodos y relaciones, permitiendo una representación de datos muchísimo más completa. Se utilizan principalmente para redes sociales, sistemas de reserva y detección de fraudes. Siendo Accumulo, Cassandra, Scylla y HBase los que utilizan este modelo.
- **Columna ancha**
  Estos almacenan y gestionan datos en forma de tablas, filas y columnas. Son comunes en aplicaciones que ocupan un formato de columna para la captura de datos. Siendo AllegroGraph, ArangoDB, InfiniteGraph, Apache Giraph, MarkLogic, Neo4J, OrientDB y Virtuoso los que utilizan este modelo.

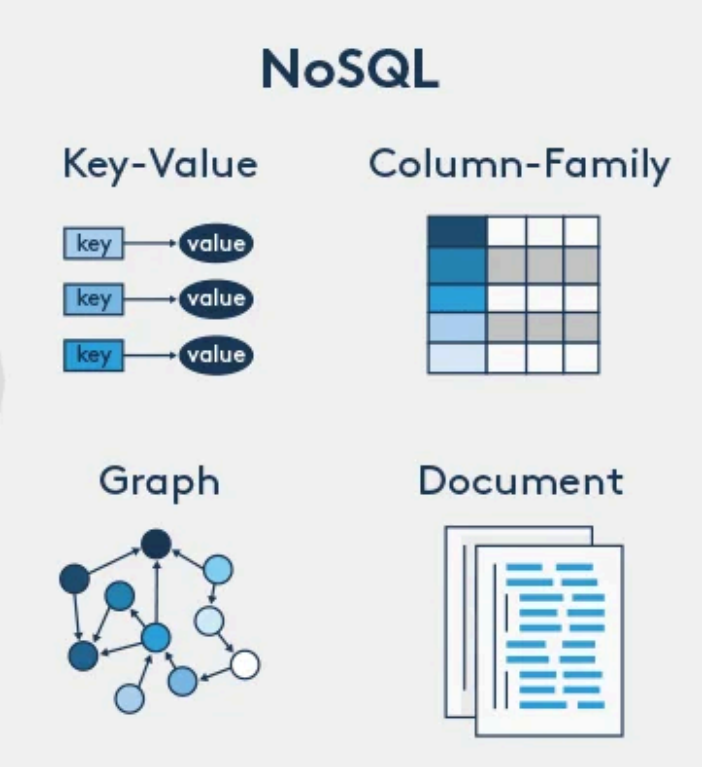

Imagen 2. Casos de uso de los modelos NoSQL agrosomodo

Referencias

[1] A. Jiménez. “Modelado orientado a objetos: clave en el desarrollo de software”. El Blog de Python. Accedido el 31 de enero de 2024. [En línea]. Disponible: https://elblogpython.com/tecnologia/modelado-orientado-a-objetos-clave-en-el-desarrollo-de-software/

[2] N. a. “Programación orientada a objetos - ventajas y desventajas | El Mundo Infinito”. El Mundo Infinito. Accedido el 31 de enero de 2024. [En línea]. Disponible: https://elmundoinfinito.com/programacion-orientada-a-objetos-ventajas-desventajas/?expand_article=1

[3] W. Díaz. “13019 – Diseño de bases de datos Capítulo 2 – Modelado orientado a objetos”. Accedido el 31 de enero de 2024. [En línea]. Disponible: https://informatica.uv.es/iiguia/DBD/Teoria/capitulo_2a.pdf

[4] Equipo editorial de IONOS. “Bases de datos clave-valor”. IONOS Digital Guide. Accedido el 31 de enero de 2024. [En línea]. Disponible: https://www.ionos.mx/digitalguide/hosting/cuestiones-tecnicas/bases-de-datos-clave-valor/

[5] N. a. “¿Por qué los desarrolladores prefieren las bases de datos NoSQL?” Oracle | Cloud Applications and Cloud Platform. Accedido el 1 de febrero de 2024. [En línea]. Disponible: https://www.oracle.com/mx/database/nosql/what-is-nosql/#:~:text=Existen%20cuatro%20tipos%20principales%20de%20bases%20de%20datos,...%203%20Gráfico%20...%204%20Columna%20ancha%20

[6] N. a. “Bases de datos NoSQL: Guía con las ventajas y desventajas”. Pandora FMS - The Monitoring Blog. Accedido el 1 de febrero de 2024. [En línea]. Disponible: https://pandorafms.com/blog/es/bases-de-datos-nosql/