Modelo orientado a objetos

Def. Una base de datos orientada a objetos es un sistema de gestión de base de datos mediante el cual representamos la información en forma de objetos que son utilizados en programación orientada a objetos.

Características

- Cada objeto tiene un nombre atributos y operaciones.
- Es una tecnología para producir modelos que reflejen un dominio d negocio y utiliza la terminología propia de tal dominio.
- Cuenta con cinco conceptos subyacentes: objeto, mensajes, clases, herencia y polimorfismo.
- Un objeto tiene un estado, un comportamiento y una entidad.
- Los mensajes brindan comunicación entre los objetos.
- Las clases son un tipo de plantilla usada para definir objetos, los cuales son instancias del mundo real.

Uso

Este tipo de bases de datos se utilizan en los sistemas multimedia, como los geográficos y de medio ambiente, los de gestión de imágenes y documentos y los de apoyo a las decisiones necesitan de modelos de datos complejos, difíciles de representar como tuplas de una tabla, como se hacia en el modelo relacional.

En general, estas aplicaciones necesitan manipular objetos y los modelos de datos deben permitirles expresar su comportamiento y las relaciones entre ellos.

Persistencia en el modelado orientado a objetos

La persistencia es una característica necesaria de los datos en un sistema de bases de datos. Recordemos que consiste en la posibilidad de recuperar datos en el futuro. Esto implica que los datos se almacenan a pesar del término del programa de aplicación. En resumen, todo administrador de base de datos brinda persistencia a sus datos.

En el caso de los sistemas de gestión de base de datos orientada a objetos (OODBMS por sus siglas en inglés), la persistencia implica almacenar los valores de atributos de un objeto con la transparencia necesaria para que el desarrollador de aplicaciones no tenga que implementar ningún mecanismo distinto al mismo lenguaje de programación orientado a objetos.

Lo anterior traería como ventaja que no seria necesario el uso de dos lenguajes de programación para construir una aplicación; es decir, actualmente, el desarrollo de aplicaciones se hace con lenguajes de programación orientada a objetos almacenamiento de datos en bases relacionales, por lo que el desarrollador debe de utilizar un lenguaje para la aplicación (Java,PHP,C++) y otro para la base de datos (SQL).

Ventajas y Desventajas de un modelo de datos orientado a objetos en comparación con bases de datos relacionales

Ventajas	Desventajas
• Modelo de objetos intuitivamente más cercano el mundo real. • Extensibilidad - herencia. • Valores complejos. • Eliminación de la impedancia incorrecta. • Lenguaje de consulta más expresivo • El estrechamiento acoplamiento entre datos y aplicaciones permite que el esquema capture más el significado de las aplicaciones. • Soporte para transacciones largas. • Mejor soporte para aplicaciones como ingeniería del software o diseño asistido por computadora (CAD) • Podría decirse que tienen mejor rendimiento, aunque los benchmarks se han aplicado principalmente en áreas como el soporte de ingeniería, a las que los los sistemas de gestión base de datos orientados a objetos están mejor adaptados.	• La falta de un fundamento teórico, por lo que el significado exacto de modelo de datos orientado a objetos no está bien definido. • Con un modelo de datos orientado a objetos es más difícil conseguir personal experimentado. • Falta de estándares. • La competencia de los sistemas de gestión de base de datos relacionales y objeto-relacionales. • La encapsulación está comprometida para optimizar las consultas. • Un sistema de gestión de base de datos orientado a objetos generalmente controla la concurrencia bloqueando. Bloquear una jerarquía de herencia es difícil y puede afectar al rendimiento. • Un modelo de datos orientado a objetos es inherentemente más complejo que el modelo de datos relacional; el sistema de gestión de base de datos orientado objetos proporciona más complejidad que el sistema de gestión de un modelo de datos relacional. La complejidad lleva a mayores costos de implementación y mantenimiento. • Falta de vistas, pero ¿son las vistas necesarias con un modelo de objetos? • Los sistemas de gestión de bases de datos orientados a objetos generalmente proporcionan control de acceso de grano grueso. Se necesita un mecanismo de seguridad más fino para la mayoría de las aplicaciones comerciales.

Modelo NoSQL

Se puede decir que la aparición del término NoSQL aparece con la llegada de la web 2.0 ya que hasta ese momento sólo subían contenido a la red aquellas empresas que tenían un portal, pero con la llegada de aplicaciones como Facebook, Twitter o Youtube, cualquier usuario podía subir contenido, provocando así un crecimiento exponencial de los datos. Es en este momento cuando empiezan a aparecer los primeros problemas de la gestión de toda esa información almacenada en bases de datos relacionales. En un principio, para solucionar estos problemas de accesibilidad, las empresas optaron por utilizar un mayor número de máquinas pero pronto se dieron cuenta de que esto no solucionaba el problema, además de ser una solución muy cara. La otra solución era la creación de sistemas pensados para un uso específico que con el paso del tiempo han dado lugar a soluciones robustas, apareciendo así el movimiento NoSQL. Por lo tanto hablar de bases de datos NoSQL es hablar de estructuras que nos permiten almacenar información en aquellas situaciones en las que las bases de datos relacionales generan ciertos problemas debido principalmente a problemas de escalabilidad y rendimiento de las bases de datos relacionales donde se dan cita miles de usuarios

concurrentes y con millones de consultas diarias. Además de lo comentado anteriormente, las bases de datos NoSQL son sistemas de almacenamiento de información que no cumplen con el esquema entidad–relación. Tampoco utilizan una estructura de datos en forma de tabla donde se van almacenando los datos sino que para el almacenamiento hacen uso de otros formatos como clave–valor, mapeo de columnas o grafos (ver epígrafe 'Tipos de bases de datos NoSQL').

Ventajas de los Sistemas NoSQL

Esta forma de almacenar la información ofrece ciertas ventajas sobre los modelos relacionales. Entre las ventajas más significativas podemos destacar:

- **Se ejecutan en máquinas con pocos recursos:** Estos sistemas, a diferencia de los sistemas basados en SQL, no requieren de apenas computación, por lo que se pueden montar en máquinas de un coste más reducido.
- **Escalabilidad horizontal:** Para mejorar el rendimiento de estos sistemas simplemente se consigue añadiendo más nodos, con la única operación de indicar al sistema cuáles son los nodos que están disponibles.
- **Pueden manejar gran cantidad de datos:** Esto es debido a que utiliza una estructura distribuida, en muchos casos mediante tablas Hash.
- **No genera cuellos de botella:** El principal problema de los sistemas SQL es que necesitan transcribir cada sentencia para poder ser ejecutada, y cada sentencia compleja requiere además de un nivel de ejecución aún más complejo, lo que constituye un punto de entrada en común, que ante muchas peticiones puede ralentizar el sistema.

Principales Diferencias con las bases de datos SQL

Algunas de las diferencias más destacables que nos podemos encontrar entre los sistemas NoSQL y los sistemas SQL están:

- **No utilizan SQL como lenguaje de consultas.** La mayoría de las bases de datos NoSQL evitan utilizar este tipo de lenguaje o lo utilizan como un lenguaje de apoyo. Por poner algunos ejemplos, Cassandra utiliza el lenguaje CQL, MongoDB utiliza JSON o BigTable hace uso de GQL.
- No utilizan estructuras fijas como tablas para el almacenamiento de los datos. Permiten hacer uso de otros tipos de modelos de almacenamiento de información como sistemas de clave–valor, objetos o grafos.
- **No suelen permitir operaciones JOIN.** Al disponer de un volumen de datos tan extremadamente grande suele resultar deseable evitar los JOIN. Esto se debe a que, cuando la operación no es la búsqueda de una clave, la sobrecarga puede llegar a ser muy costosa. Las soluciones más directas consisten en desnormalizar los datos, o bien realizar el JOIN mediante software, en la capa de aplicación.
- **Arquitectura distribuida.** Las bases de datos relacionales suelen estar centralizadas en una única máquina o bien en una estructura máster–esclavo, sin embargo, en los casos NoSQL la información puede estar compartida en varias máquinas mediante mecanismos de tablas Hash distribuidas.

Tipos de Bases de Datos NoSQL

Dependiendo de la forma en la que almacenen la información, nos podemos encontrar varios tipos distintos de bases de datos NoSQL, algunos de ellos son:

1. **Bases de datos clave – valor:** Son el modelo de base de datos NoSQL más popular, además de ser la más sencilla en cuanto a funcionalidad. En este tipo de sistema, cada elemento está identificado por una llave única, lo que permite la recuperación de la información de forma muy rápida, información que habitualmente está almacenada como un objeto binario (BLOB). Se caracterizan por ser muy eficientes tanto para las lecturas como para las escrituras. Algunos ejemplos de este tipo son Cassandra, BigTable o HBase.
2. **Bases de datos documentales:** Este tipo almacena la información como un documento, generalmente utilizando para ello una estructura simple como JSON o XML y donde se utiliza una clave única para cada registro. Este tipo de implementación permite, además de realizar búsquedas por clave–valor, realizar consultas más avanzadas sobre el contenido del documento. Son las bases de datos NoSQL más versátiles. Se pueden utilizar en gran cantidad de proyectos, incluyendo muchos que tradicionalmente funcionarían sobre bases de datos relacionales. Algunos ejemplos de este tipo son MongoDB o CouchDB.
3. **Bases de datos en grafo:** En este tipo de bases de datos, la información se representa como nodos de un grafo y sus relaciones con las aristas del mismo, de manera que se puede hacer uso de la teoría de grafos para recorrerla. Para sacar el máximo rendimiento a este tipo de bases de datos, su estructura debe estar totalmente normalizada, de forma que cada tabla tenga una sola columna y cada relación dos. Este tipo de bases de datos ofrece una navegación más eficiente entre relaciones que en un modelo relacional. Algunos ejemplos de este tipo son Neo4j, InfoGrid o Virtuoso.
4. **Bases de datos orientadas a objetos:** En este tipo, la información se representa mediante objetos, de la misma forma que son representados en los lenguajes de programación orientada a objetos (POO) como ocurre en JAVA, C# o Visual Basic .NET. Algunos ejemplos de este tipo de bases de datos son Zope, Gemstone o Db4o.

¿Cuándo utilizar bases de datos NoSQL?

Algunas de las razones que nos pueden llevar a decantarnos por el uso de las bases de datos NoSQL en lugar de las clásicas SQL son:

- Cuando el volumen de los datos crece muy rápidamente en momentos puntuales, pudiendo llegar a superar el Terabyte de información.
- Cuando la escalabilidad de la solución relacional no es viable tanto a nivel de costes como a nivel técnico.
- Cuando tenemos elevados picos de uso del sistema por parte de los usuarios en múltiples ocasiones.
- Cuando el esquema de la base de datos no es homogéneo, es decir, cuando en cada inserción de datos la información que se almacena puede tener campos distintos.

Bibliografía:

[1]Anonimo. "Modelo Orientado a Objetos " [Online].Aveilable: https://programas.cuaed.unam.mx/repositorio/moodle/pluginfile.php/782/mod_resource/content/8/contenido/index.html

[2] Anonimo. "Bases de datos NoSQL. Qué son y tipos que nos podemos encontrar**"[Online].Aveilable: https://www.acens.com/wp-content/images/2014/02/bbdd-nosql-wp-acens.pdf**