Universidad Nacional Autónoma de México

Facultad de Ingeniería

Bases de Datos | Grupo 1

Tarea 1

Lara Hernández Emmanuel – 321189536

1. Modelo de Bases de Datos Orientadas a Objetos

El modelo de base de datos orientada a objetos (OODBMS, por sus siglas en inglés) organiza la información en **objetos**. A diferencia del enfoque relacional, donde los datos suelen distribuirse en múltiples tablas, en este modelo **los atributos de un registro se agrupan en una sola unidad**. Así, la información relacionada permanece integrada y disponible de manera directa dentro del objeto.

Además de almacenar atributos, los objetos pueden incluir **métodos** (comportamiento). Esto refleja la afinidad de este modelo con la programación orientada a objetos: cada objeto no solo representa datos, sino también acciones que pueden ejecutarse sobre ellos.

1.1. Clases, jerarquías y relaciones

Los objetos se describen mediante **clases**. Un objeto es una instancia concreta de una clase, mientras que la clase define de forma abstracta su estructura y comportamiento. Esta organización facilita construir **jerarquías de clases y subclases**, donde una subclase puede heredar propiedades de su clase superior y extenderlas con nuevos atributos o métodos.

Aunque la herencia forma una estructura jerárquica, los objetos también pueden establecer **relaciones con otras clases**, lo que permite construir redes de relaciones más flexibles. Asimismo, es posible combinar objetos simples para formar **objetos compuestos** (más complejos).

1.2. Identificación y persistencia

Para administrar objetos de manera consistente, un SGBD orientado a objetos asigna a cada registro un **identificador único** (OID). Este identificador permite recuperar los objetos almacenados sin depender de claves compuestas o joins extensos.

2. Utilidad del modelo orientado a objetos

Los sistemas gestores de bases de datos evolucionan para responder a nuevas necesidades tecnológicas. Aunque los SGBD relacionales (RDBMS) dominan el mercado por su madurez y estandarización, existen aplicaciones donde el modelo tabular resulta limitado, especialmente cuando se requiere representar **datos complejos** y relaciones ricas.

Este modelo es particularmente útil en áreas como CAD/CAM, sistemas multimedia, sistemas geográficos y ambientales, gestión de imágenes y documentos, y sistemas de apoyo a decisiones, donde los datos no se describen fácilmente como tuplas simples en una tabla.

En general, estas aplicaciones requieren manipular objetos complejos; por ello, el modelo debe permitir representar no solo datos, sino también su **comportamiento** y las **relaciones** entre ellos.

2.1. Operaciones y requerimientos comunes en OODBMS

Un manejador de bases de datos orientado a objetos suele considerar, entre otros, los siguientes requerimientos:

- Definir tipos de datos propios (extensibilidad del modelo).
- Manejar volúmenes de datos potencialmente muy grandes.
- Soportar transacciones de larga duración.
- Recuperar eficientemente objetos complejos (incluyendo objetos compuestos).
- Proveer lenguajes de consulta orientados a objetos, por ejemplo OQL.
- Incorporar mecanismos de seguridad basados en la noción de objeto.
- Permitir funciones para definir reglas o mecanismos deductivos.

3. Ventajas y limitaciones

El uso de bases de datos orientadas a objetos suele ser adecuado cuando el desarrollo se realiza con lenguajes orientados a objetos (por ejemplo, Java), ya que facilita integrar objetos del código con objetos persistentes en la base de datos.

Sin embargo, su adopción es menor en comparación con el modelo relacional: existen menos SGBD orientados a objetos, menor estandarización y una comunidad más reducida. Además, aunque la complejidad del modelo puede acelerar consultas sobre estructuras complejas, en escenarios donde los procesos y datos son simples, esa complejidad puede añadir sobrecosto.

4. Comparación: bases de datos orientadas a objetos vs. relacionales

Criterio	**Orientada a objetos (OODBMS)**	**Relacional (RDBMS)**
Unidad principal de almacenamiento	Objeto que integra atributos y, en algunos casos, métodos	Tabla formada por filas (tuplas) y columnas (atributos)
Representación de relaciones	Referencias directas entre objetos y asociaciones entre clases	Relaciones mediante claves primarias y foráneas con uso de joins
Modelado de datos complejos	Adecuado para estructuras complejas, jerarquías y objetos compuestos	Requiere normalización y múltiples tablas para representar complejidad
Rendimiento en consultas complejas	Eficiente cuando se accede a objetos completos como una unidad	Puede verse afectado por joins costosos en esquemas grandes
Integración con programación orientada a objetos	Alta, ya que el modelo coincide con el paradigma de POO	Menor, suele requerir mapeo objeto-relacional (ORM)
Estandarización y adopción	Menor estandarización y menor número de gestores disponibles	Altamente estandarizado y ampliamente utilizado
Lenguaje de consulta	Lenguajes orientados a objetos (por ejemplo, OQL en algunos sistemas)	SQL como lenguaje estándar
Casos de uso típicos	CAD/CAM, multimedia, sistemas geográficos y apoyo a decisiones	Sistemas transaccionales, aplicaciones empresariales y analítica

Table 1: Comparación de OOBDMS vs Relacionales

5. Modelos de Bases de Datos NoSQL

Las bases de datos NoSQL surgen como una alternativa a los sistemas relacionales tradicionales cuando las aplicaciones requieren manejar grandes volúmenes de información, alta disponibilidad y escalabilidad horizontal. De acuerdo con IBM y Amazon Web Services, este tipo de bases de datos está pensado para operar en entornos distribuidos, especialmente en aplicaciones modernas desplegadas en la nube.

Una de las principales características de los modelos NoSQL es que no imponen un esquema rígido. Esto permite que la estructura de los datos pueda cambiar con el tiempo sin necesidad de rediseñar toda la base de datos, lo cual resulta útil en sistemas donde los requerimientos evolucionan rápidamente.

5.1. Modelo clave–valor

El modelo clave–valor es el más simple dentro de las bases de datos NoSQL. En este enfoque, cada elemento se almacena como un par formado por una clave única y un valor asociado. El sistema únicamente utiliza la clave para acceder a la información, sin interpretar el contenido del valor, lo que permite un acceso extremadamente rápido a los datos.

Entre sus principales ventajas se encuentra la alta eficiencia en operaciones de lectura y escritura, así como su facilidad para escalar horizontalmente en entornos distribuidos. Esta simplicidad reduce la complejidad del diseño y permite manejar grandes volúmenes de datos con baja latencia.

Sin embargo, debido a su estructura minimalista, presenta limitaciones importantes. La capacidad para realizar consultas complejas es reducida y resulta difícil representar relaciones entre distintos elementos, ya que el sistema depende principalmente del acceso directo mediante la clave.

Por estas características, el modelo clave–valor suele emplearse en escenarios donde la rapidez es prioritaria y no se requiere establecer relaciones complejas entre los datos, como el almacenamiento de sesiones de usuario, sistemas de caché o información temporal.

5.2. Modelo documental

El modelo documental almacena la información en documentos, generalmente en formatos como JSON. Cada documento agrupa datos relacionados y puede tener una estructura distinta a otros documentos, incluso dentro de la misma colección. Esto proporciona una gran flexibilidad y facilita representar información jerárquica.

Una de sus principales ventajas es la flexibilidad del esquema, que permite adaptar fácilmente la estructura de los datos conforme evolucionan los requerimientos del sistema. Además, los documentos suelen corresponder naturalmente con los objetos utilizados en aplicaciones modernas, lo que facilita la integración con el desarrollo de software.

No obstante, esta flexibilidad también puede representar una desventaja, ya que la ausencia de un esquema rígido puede provocar duplicación de información y trasladar parte de la validación hacia la lógica de la aplicación. Asimismo, las relaciones complejas entre documentos pueden resultar menos eficientes que en otros modelos especializados.

Este modelo resulta especialmente adecuado para aplicaciones web, sistemas de gestión de contenido, APIs y plataformas donde la estructura de los datos puede variar con frecuencia.

5.3. Modelo de grafos

Las bases de datos orientadas a grafos representan la información mediante nodos y relaciones explícitas entre ellos. Este enfoque permite modelar de forma directa las conexiones entre entidades, lo cual facilita la exploración de relaciones complejas.

Entre sus ventajas principales se encuentra la capacidad de realizar consultas basadas en relaciones de manera eficiente, evitando joins costosos y permitiendo analizar estructuras altamente conectadas. Esto resulta especialmente útil cuando las relaciones entre los datos son tan relevantes como los propios datos.

Sin embargo, su implementación puede resultar más compleja en comparación con otros modelos NoSQL, y no siempre es la opción más adecuada para datos tabulares o estructuras simples. Además, su

adopción es menor en algunos entornos, lo que puede limitar herramientas disponibles.

Este modelo destaca en aplicaciones como redes sociales, sistemas de recomendación, análisis de fraude y cualquier escenario donde sea necesario analizar conexiones, dependencias o recorridos entre distintos elementos.

5.4. Modelo columnar

El modelo columnar organiza la información en columnas en lugar de filas y está diseñado para operar sobre sistemas distribuidos. Este enfoque permite procesar grandes volúmenes de datos de forma eficiente, especialmente en consultas analíticas que requieren recorrer solo ciertas columnas.

Entre sus principales ventajas se encuentra el alto rendimiento en consultas analíticas y procesamiento masivo de datos, ya que reduce la cantidad de información que debe leerse cuando solo se requieren ciertos atributos. Además, su arquitectura facilita la escalabilidad en entornos distribuidos y aplicaciones de Big Data.

No obstante, este modelo puede presentar limitaciones en sistemas que requieren transacciones frecuentes o escrituras en tiempo real, debido a que su diseño está orientado principalmente al análisis y no al procesamiento transaccional intensivo. Asimismo, el diseño del esquema requiere planificación cuidadosa para obtener un rendimiento óptimo.

Este tipo de bases de datos es común en sistemas de análisis de datos, monitoreo, procesamiento de logs y data warehousing distribuido.

6. Resumen comparativo de modelos NoSQL

Modelo	Ventajas	Desventajas	Casos de uso
Clave–valor	Acceso muy rápido mediante clave única, alta escalabilidad y simplicidad de implementación.	Limitado para consultas complejas y difícil representar relaciones entre datos.	Caché de aplicaciones, sesiones de usuario, almacenamiento temporal y sistemas de alta velocidad.
Documental	Flexibilidad de esquema, representación natural de datos jerárquicos e integración sencilla con aplicaciones modernas.	Posible duplicación de datos y manejo menos eficiente de relaciones complejas.	Aplicaciones web, APIs REST, sistemas de gestión de contenido y catálogos de información.
Grafos	Modelado eficiente de relaciones complejas y consultas basadas en conexiones.	Mayor complejidad conceptual y menor adopción general.	Redes sociales, sistemas de recomendación, detección de fraude y análisis de relaciones.
Columnar	Alto rendimiento en consultas analíticas y procesamiento de grandes volúmenes de datos.	No ideal para transacciones intensivas o escrituras frecuentes.	Big Data, analítica, monitoreo, procesamiento de logs y data warehousing distribuido.

Table 2: Comparación general de modelos de bases de datos NoSQL

References

[1] E. D. K. Hernández, "Modelo orientado a objetos." https://repositorio-uapa.cuaed.unam.mx/repositorio/moodle/pluginfile.php/2731/mod_resource/content/1/UAPA-Modelo-Orientado-Objetos/index.html

[2] "Base de datos orientada a objetos: la información en unidades," *IONOS Digital Guide*, Jan. 18, 2023. https://www.ionos.mx/digitalguide/hosting/cuestiones-tecnicas/base-de-datos-orientada-a-objetos/

[3] Amazon Web Services, Inc., "¿Qué es NoSQL?," https://aws.amazon.com/es/nosql/

[4] IBM, "Bases de datos NoSQL," Nov. 27, 2025. https://www.ibm.com/mx-es/think/topics/nosql-databases