



UNIVERSIDAD NACIONAL AUTÓNOMA DE  
MÉXICO  
FACULTAD DE INGENIERÍA

DIVISIÓN DE INGENIERÍA ELÉCTRICA  
COORDINACIÓN DE COMPUTACIÓN

## **BASES DE DATOS**

### **Tarea 1**

## **Introducción a las Bases de Datos**

**Nombre del alumno:** Yukioayax Canek Gabriel Hernández

**Profesor:** Fernando Arreola Franco

**Grupo:** 1

**Semestre en curso:** 2026-2

# 1. Modelo orientado a objetos

## 1.1. Descripción

El modelo orientado a objetos (MOO) aplica los conceptos de la programación orientada a objetos a las bases de datos, los datos y sus relaciones se encapsulan en objetos que contienen tanto atributos como métodos, cada objeto tiene un identificador único (OID) y pertenece a una clase la cual define la estructura y comportamiento de sus instancias y además, las clases pueden organizarse en jerarquías de herencia, permitiendo la reutilización y especialización.

## 1.2. Ventajas

- Representación natural de entidades del mundo real.
- Reutilización de código mediante herencia y polimorfismo.
- Integración directa con lenguajes de programación orientados a objetos.
- Soporte para tipos de datos complejos (colecciones, multimedia, etc.).

## 1.3. Desventajas

- Mayor complejidad en el diseño y la implementación.
- Curva de aprendizaje pronunciada para equipos acostumbrados a modelos relacionales.
- Menor estandarización en comparación con el modelo relacional (SQL).
- Rendimiento potencialmente inferior en consultas simples debido a la navegación entre objetos.

## 1.4. Casos de uso

- Sistemas CAD/CAM (diseño asistido por computadora).
- Aplicaciones científicas con datos complejos (genómica, física).
- Sistemas de información geográfica (SIG).
- Telecomunicaciones y sistemas de tiempo real.

# 2. Modelos NoSQL

Los sistemas NoSQL (Not only SQL) surgen para manejar grandes volúmenes de datos no estructurados o semiestructurados, escalabilidad horizontal y esquemas flexibles, se clasifican principalmente en cuatro tipos.

## 2.1. Modelo clave-valor

### Descripción

Es el modelo NoSQL más simple. Los datos se almacenan como un diccionario o tabla hash, donde cada ítem tiene una clave única y un valor asociado (que puede ser opaco para el gestor) algunos ejemplos son Redis, Riak, Amazon DynamoDB.

### Ventajas

- Alto rendimiento y baja latencia.
- Escalabilidad horizontal excelente.
- Modelo extremadamente simple y fácil de usar.

### Desventajas

- No es adecuado para consultas complejas o relaciones entre datos.
- La lógica de negocio debe manejar la interpretación del valor.
- Limitado soporte para transacciones ACID (depende del producto).

### Casos de uso

- Almacenamiento de sesiones de usuario.
- Caché de objetos.
- Carritos de compra.
- Sistemas de recomendación en tiempo real.

## 2.2. Modelo documental

### Descripción

Almacena datos en documentos semiestructurados (JSON, BSON, XML), cada documento tiene una estructura interna y puede contener subdocumentos y arreglos; los documentos se agrupan en colecciones como por ejemplo MongoDB, Couchbase, Firebase Firestore.

### Ventajas

- El esquema es flexible pues los documentos de una misma colección pueden tener diferentes campos.
- Tiene un mapeo natural a objetos de lenguajes de programación modernos.
- Sus consultas son ricas sobre el contenido de los documentos.
- Tiene indexación sobre cualquier campo.

## Desventajas

- Puede haber redundancia y duplicación de datos.
- Las operaciones de unión (join) no están soportadas nativamente (se manejan mediante agregaciones o embebido).
- Consistencia eventual en configuraciones distribuidas.

## Casos de uso

- Catálogos de productos (e-commerce).
- Sistemas de gestión de contenidos (CMS).
- Blogs y plataformas de publicación.
- Aplicaciones con datos que evolucionan rápidamente (eventos, logs).

## 2.3. Modelo de grafos

### Descripción

Representa los datos mediante nodos (entidades), aristas (relaciones) y propiedades, está optimizado para consultas que explotan las conexiones entre los datos, por ejemplo Neo4j, Amazon Neptune, OrientDB.

### Ventajas

- Excelente para datos altamente interconectados.
- Consultas de relaciones profundas y patrones de grafos muy eficientes.
- Modelo intuitivo para problemas como redes sociales, recomendaciones, detección de fraudes.

### Desventajas

- No es adecuado para escenarios con muchos agregados o consultas sin relaciones.
- Escalamiento horizontal más complejo que en otros modelos NoSQL.
- Curva de aprendizaje del lenguaje de consulta.

### Casos de uso

- Redes sociales (amistades, interacciones).
- Motores de recomendación (basados en conexiones).
- Detección de fraudes y análisis de riesgos.
- Gestión de identidades y accesos.

## 2.4. Modelo columnar

### Descripción

Almacena los datos en columnas en lugar de filas, optimizando el acceso a atributos específicos, las columnas se agrupan en familias de columnas y pueden tener versiones temporales como Apache Cassandra, HBase, Google Bigtable.

### Ventajas

- Alta compresión y rendimiento en consultas que involucran pocas columnas.
- Escalabilidad lineal y alta disponibilidad.
- Ideal para escrituras masivas y datos con alta cardinalidad.

### Desventajas

- Modelo más complejo de diseñar que el relacional.
- No soporta consultas ad-hoc complejas de manera eficiente (depende del diseño).
- Operaciones de agregación pueden ser costosas si no se modelan correctamente.

### Casos de uso

- Almacenamiento de series temporales (logs, métricas IoT).
- Sistemas de mensajería y redes sociales (grandes volúmenes de escritura).
- Analítica de datos a gran escala (data warehouses columnares).
- Aplicaciones que requieren baja latencia en consultas por clave primaria.

## Referencias

- [1] C. J. Date, *An Introduction to Database Systems*, 8th ed. Massachusetts: Addison Wesley, 2003.
- [2] R. Elmasri y S. Navathe, *Fundamentos de sistemas de bases de datos*, 3ra ed. Pearson Prentice Hall, 2003.
- [3] D. Kroenke, *Procesamiento de bases de datos*, 8a ed. México: Pearson / Prentice Hall, 2003.
- [4] P. Rob y C. Coronel, *Database Systems: Design, Implementation and Management*, 6th ed. Course Technology, 2004.
- [5] A. De Miguel Martínez, M. Piattini et al., *Diseño de bases de datos relacionales*. México: Alfamaga, 2000.
- [6] J. Johnson, *Bases de datos, modelos, lenguajes, diseño*. México: Oxford University Press, 2000.