

UNIVERSIDAD PERUANA LOS ANDES

FACULTAD DE INGENIERÍA

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS Y
COMPUTACIÓN



UPLA

ASIGNATURA: Base de Datos II
ESTUDIANTE: Luis Enrique Espejo Quispe
DOCENTE: Mg. Raúl Fernández Bejarano
CICLO: V
SECCIÓN: A1

HYO-2025

1. Bases de Datos Relacionales (SQL)

La base de las bases de datos relacionales (RDBMS), ejemplificada por PostgreSQL, es el **Modelo Relacional** desarrollado por E.F. Codd en los años 70, que se basa en la teoría de conjuntos y la lógica de predicados. El principio fundamental es la organización de los datos en **tablas** bidimensionales, donde cada fila es una **tupla** (registro) y las columnas representan **atributos** con tipos de datos estrictos. La fortaleza de este modelo reside en la capacidad de definir **relaciones** claras entre tablas usando claves primarias y foráneas, lo que permite realizar *JOINS* complejos. La piedra angular de su fiabilidad son las propiedades **ACID** (**A**tomicidad, **C**onsistencia, **I**slamiento, **D**urabilidad), que garantizan que las transacciones sean procesadas de manera segura y que la integridad de los datos se mantenga en todo momento, haciéndolas ideales para sistemas transaccionales críticos.

2. Bases de Datos No Relacionales (NoSQL)

Las bases de datos NoSQL, como MongoDB, surgieron para abordar las limitaciones de escalabilidad y la rigidez de los esquemas relacionales, especialmente con el auge de **Big Data** y las aplicaciones web modernas. A diferencia de las SQL, no se limitan a un único modelo tabular, sino que utilizan estructuras diversas como **documentos** (JSON/BSON), **clave-valor**, o **grafos**. Su diseño favorece el **escalamiento horizontal** (distribuyendo la carga en múltiples servidores) y la **flexibilidad de esquema**, permitiendo a los desarrolladores iterar rápidamente sin necesidad de un plan de esquema rígido. Teóricamente, a menudo se adhieren al modelo **BASE** (**B**ásicamente Disponible, **E**stado Suave, **C**onsistencia **E**ventual), priorizando la **disponibilidad** y el **rendimiento** sobre la consistencia inmediata, una compensación vital para aplicaciones de alta concurrencia.

3. Bases de Datos Multimodelo (Híbridas)

El enfoque multimodelo, representado por bases de datos como ArangoDB, es una evolución que busca la eficiencia y la simplicidad operacional al unificar varios modelos de datos distintos (Documento, Grafo, Clave-Valor) dentro de **un solo núcleo de motor de base de datos**. El principio central es que una aplicación moderna a menudo necesita diferentes modelos de datos para diferentes tareas (ej. documentos para perfiles de usuario y grafos para relaciones entre ellos). El multimodelo permite gestionar todo esto con una única base de datos y un **lenguaje de consulta unificado** (como AQL), evitando la complejidad y la sobrecarga de latencia de tener que integrar y sincronizar múltiples bases de datos de un solo modelo (*persistencia políglota*). Esto resulta en un desarrollo más ágil y en la capacidad de ejecutar consultas que combinan las fortalezas de cada modelo.

Cuadro comparativo

Características	PostgreSQL (Relacional)	MongoDB (NoSQL - Documentos)	ArangoDB (Híbrido/Multimodelo)
Modelo de Datos	Relacional: Tablas, filas, columnas, esquemas fijos.	Documentos (JSON/BSON): Colecciones flexibles, esquema dinámico.	Multimodelo: Soporta Documento, Grafo y Clave-Valor nativamente.
Lenguaje de Consulta	SQL (Structured Query Language).	MongoDB Query Language (MQL) (basado en JSON) y el Aggregation Framework .	AQL (ArangoDB Query Language), declarativo y optimizado para multimodelos.
Ventajas	<ul style="list-style-type: none"> - ACID estricto: Alta integridad y consistencia de datos. - Soporte de <i>JOINS</i> complejos y transacciones. - Madurez, robustez y amplio ecosistema. 	<ul style="list-style-type: none"> - Gran flexibilidad de esquema. - Escalabilidad horizontal superior. - Ideal para datos cambiantes y grandes volúmenes de datos. 	<ul style="list-style-type: none"> - Flexibilidad y rendimiento de NoSQL con capacidad de JOINS de datos relacionales/gráficos. - Un solo sistema para múltiples necesidades de datos.
Limitaciones	<ul style="list-style-type: none"> - Baja escalabilidad horizontal (principalmente vertical). - Poca flexibilidad para manejar datos no estructurados o esquemas cambiantes. 	<ul style="list-style-type: none"> - Consistencia eventual (no estricta). - Transacciones complejas limitadas. - Dificultad para manejar relaciones complejas entre documentos. 	<ul style="list-style-type: none"> - Menos maduro que los sistemas SQL tradicionales. - La optimización y el modelado pueden ser más complejos al usar múltiples modelos.
Escenarios de Uso	<ul style="list-style-type: none"> - Sistemas Financieros/Bancarios (alta integridad). - Sistemas ERP/CRM. - Aplicaciones con datos altamente estructurados y relaciones fijas. 	<ul style="list-style-type: none"> - Catálogos de productos (con atributos variables). - Big Data y IoT. - Plataformas de contenido dinámico (ej. blogs, CMS). 	<ul style="list-style-type: none"> - Aplicaciones que requieren datos de documentos y relaciones de grafos (ej. redes sociales, motores de recomendación, análisis de fraude). - Proyectos que necesitan consolidar bases de datos.