INSTITUTO SUPERIOR
UNIVERSITARIO
sucre

GUÍA GENERAL DE ESTUDIO DE BASE DE DATOS

x|

Guía General de Estudio de Base de Datos

Robinson Raúl Campaña Obando

Jorge Paolo Morales Jaya

Jorge Alfredo Yánez Intriago

Esta publicación ha sido sometida a revisión por pares académicos específicos por:

Livio Danilo Miniguano Miniguano

Universidad Técnica de Ambato

Corrección de estilo:

Miguel Ángel Naranjo Arcos - Docente - Sucre

Diseño y diagramación:

Freddy Javier Centeno Martínez - Docente - Sucre

Editorial RIMANA

Primera Edición
Quito – Ecuador

INSTITUTO SUPERIOR UNIVERSITARIO SUCRE

ISBN: 978-9942-590-16-9

MISIÓN

Ser una Institución Superior Universitaria con estándares de calidad académica e innovación, reconocida a nivel nacional con proyección internacional.

VISIÓN

Formamos profesionales competentes con espíritu emprendedor, capaces de contribuir al desarrollo integral del país.

Tabla de contenido

PRESENTACIÓN DE LA ASIGNATURA

La asignatura de Bases de Datos forma parte del plan de estudios de la carrera de Desarrollo de Software y tiene como finalidad proporcionar a los estudiantes los conocimientos teóricos y prácticos necesarios para diseñar, implementar y administrar bases de datos en distintos entornos de desarrollo.

A lo largo de la asignatura, se abordan tanto los fundamentos de las bases de datos relacionales como los principios básicos de las bases de datos orientadas a documentos (NoSQL), considerando la importancia de cada modelo en el desarrollo de aplicaciones modernas. Se profundiza en temas como el modelado de datos, el lenguaje de consulta estructurado (SQL), la optimización de consultas, la gestión de transacciones y la seguridad de los datos, así como en la exploración de bases de datos NoSQL y sus aplicaciones.

El enfoque metodológico de la asignatura combina el aprendizaje teórico con la práctica constante, permitiendo a los estudiantes aplicar los conocimientos adquiridos en proyectos y ejercicios que simulan situaciones reales del ámbito profesional. De esta manera, se busca que los futuros desarrolladores adquieran las competencias necesarias para enfrentar con éxito los desafíos tecnológicos que demanda la industria.

En síntesis, la asignatura de Bases de Datos constituye un pilar fundamental en la formación académica de los estudiantes, crítico y la capacidad de adaptación a las nuevas tendencias tecnológicas.

RESULTADOS DEL APRENDIZAJE

Al finalizar el estudio de esta guía, el estudiante será capaz de:

- Comprender la importancia de las bases de datos en el desarrollo de software y su papel en la gestión de la información.
- Reconocer las características y conceptos fundamentales de las bases de datos relacionales y NoSQL.
- Identificar y comparar los diferentes modelos de bases de datos y sus aplicaciones generales.
- Analizar de manera general las ventajas y desventajas de los distintos tipos de bases de datos.

- Distinguir entre las estructuras de datos, escalabilidad, consistencia y flexibilidad que ofrecen las bases de datos relacionales y NoSQL.
- Evaluar de forma general los factores a considerar al seleccionar un tipo de base de datos para un proyecto.
- Reconocer las tendencias y desafíos emergentes en el uso de bases de datos, fomentando el aprendizaje continuo.

UNIDAD 1 INTRODUCCIÓN

Definición y justificación

En el ámbito académico de las Ciencias de la Computación y el Desarrollo de Software, una base de datos se define como un conjunto estructurado de información organizada de manera que permite su almacenamiento, recuperación, manipulación y gestión eficiente (Coronel & Morris, 2021). Las bases de datos constituyen un elemento fundamental en el desarrollo de sistemas de información, facilitando el procesamiento y la administración de grandes volúmenes de datos. Estas pueden clasificarse en distintos modelos, entre los que destacan las bases de datos relacionales y las bases de datos NoSQL, cada una con características específicas que responden a diferentes necesidades y contextos de uso.

¿Cuán importante resultan las estructuras de datos en el ecosistema del desarrollo de software?

En el entorno tecnológico contemporáneo, la viabilidad de cualquier solución informática depende de su capacidad para gestionar flujos masivos de contenido. Las bases de datos actúan como el núcleo estructural que permite procesar y recuperar información con altos estándares de precisión.

Impacto en el Desarrollo y la Escalabilidad

Más allá del simple almacenamiento, la correcta administración de los sistemas de gestión de datos impacta directamente en:

- Optimización de Recursos: Permite una interacción fluida entre la lógica de negocio y las capas de presentación.
- Estabilidad y Seguridad: Garantiza la protección y la coherencia de los activos digitales frente a demandas crecientes de tráfico.
- Eficiencia Operativa: Facilita el despliegue de plataformas que se adaptan a las exigencias volátiles del sector comercial.

El Rol Estratégico del Desarrollador

La construcción de software moderno requiere que los ingenieros trasciendan la programación básica, integrando herramientas analíticas y de modelado de datos en su flujo de trabajo. Esta competencia es

indispensable, ya que la arquitectura de almacenamiento representa el cimiento sobre el cual se edifican las experiencias de usuario más confiables y robustas de la actualidad.

De acuerdo con Elmasri y Navathe (2016), estos sistemas constituyen la infraestructura esencial que sostiene prácticamente la totalidad de las innovaciones tecnológicas vigentes.

Perspectiva Histórica y Funcional de las Estructuras de Datos en la Ingeniería de Software

La capacidad de las organizaciones modernas para gestionar sus activos informáticos es el resultado de una transformación tecnológica que comenzó con sistemas rígidos y ha culminado en arquitecturas inteligentes y ubicuas.

Hitos de la Transformación Estructural

El panorama del almacenamiento ha transitado por diversas etapas críticas que definen el desarrollo de software actual:

- Orígenes y Rigidez (Década de 1960): Las primeras soluciones se basaban en modelos jerárquicos y de red. Aunque pioneros, estos sistemas carecían de la versatilidad necesaria para procesos de datos dinámicos.
- La Revolución Relacional (1970): Gracias a las propuestas teóricas de Edgar F. Codd, el sector adoptó un esquema lógico basado en la independencia de los datos. El uso de lenguajes declarativos como SQL simplificó radicalmente la interacción con la información.
- La Era del Big Data y NoSQL: La explosión de la web y el procesamiento masivo de información no estructurada impulsaron motores de búsqueda y almacenamiento altamente escalables, priorizando el rendimiento sobre la rigidez del esquema tradicional.

Tendencias Contemporáneas e Impacto Tecnológico

En la actualidad, el diseño de software no se limita a una única tecnología, sino que converge en ecosistemas híbridos. Esta integración permite a las compañías desarrollar infraestructuras que combinan la consistencia relacional con la flexibilidad de los modelos no relacionales.

De igual forma, el auge de la Inteligencia Artificial (IA) y el Machine Learning ha forzado una evolución hacia el procesamiento distribuido y analítica en tiempo real en entornos de nube (Coronel

& Morris, 2018). Según señalan Elmasri y Navathe (2016), esta progresión histórica es la que dota a los especialistas de las herramientas analíticas necesarias para resolver los desafíos de almacenamiento en escenarios de alta complejidad.

Diversidad Arquitectónica y Ecosistemas de Datos en la Ingeniería Contemporánea

En el panorama tecnológico de vanguardia, la arquitectura de una solución de software se define por su capacidad para procesar activos de información bajo criterios de alta disponibilidad. La elección del motor de datos no es una decisión trivial, sino una respuesta estratégica a la naturaleza del flujo informativo y los requisitos de escalabilidad del producto final.

Taxonomía de los Sistemas de Gestión de Datos

La especialización de las bases de datos permite segmentar las soluciones en dos grandes vertientes que dominan el mercado:

- Modelos Relacionales (RDBMS): Basados en la estructura tabular y el rigor del esquema definido por Codd, estos sistemas priorizan la consistencia transaccional e integridad referencial. Su lenguaje estándar, SQL, sigue siendo la herramienta analítica predominante para entornos corporativos complejos.
- Estructuras NoSQL: Diseñadas para romper la rigidez de los esquemas fijos, estas variantes (orientadas a documentos, grafos, clave-valor o columnas anchas) facilitan la escalabilidad horizontal. Son el componente esencial en arquitecturas de microservicios y despliegues en la nube.

Impacto en el Desarrollo y la Competitividad

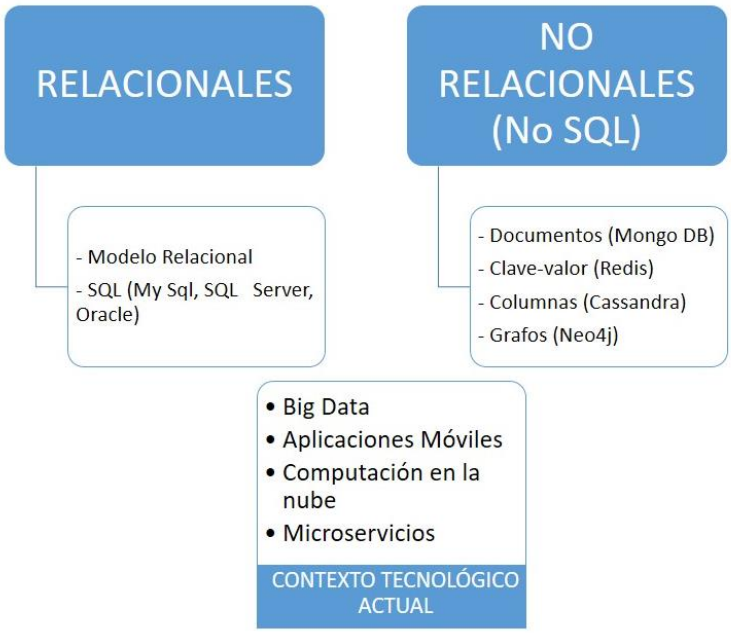
Para el profesional de la ingeniería, dominar este abanico tecnológico es imperativo. La selección precisa entre un modelo relacional o uno no relacional determina la viabilidad técnica y el rendimiento operativo de las aplicaciones en mercados globales altamente exigentes.

Esta capacidad de discernimiento técnico permite a los desarrolladores estructurar soluciones que no solo almacenan información, sino que potencian la agilidad del negocio. Como señalan Coronel y

Morris (2018), la comprensión profunda de estas tipologías y su aplicabilidad práctica es el factor diferenciador para resolver los desafíos de gestión de datos en la era de la transformación digital.

FIGURA 1

Clasificación de sistemas de gestión de datos en el ecosistema digital actual



Nota. Adaptación de las arquitecturas de almacenamiento dominantes, segmentadas en modelos relacionales (SQL) y no relacionales (NoSQL), vinculadas a paradigmas de computación en la nube y microservicios. Elaboración propia Campaña (2025).

Cuestionario de Autoevaluación - Unidad 1: Fundamentos de Sistemas de Datos

1. ¿Cómo definen Coronel y Morris (2021) el concepto de base de datos?

A) Un entorno operativo para la ejecución de diversas apps.

B) Un repositorio de información estructurado lógicamente para optimizar su guardado, administración y acceso. ✓

C) Un aplicativo enfocado en la creación de recursos gráficos.

D) Una plataforma para la gestión de mensajería electrónica.

2. En el ecosistema del desarrollo de software, ¿cuál es el propósito fundamental de una base de datos?

A) El diseño y despliegue de redes sociales.

B) Ofrecer el soporte técnico necesario para el gobierno y almacenamiento de la información. ✓

C) La generación automatizada de componentes de interfaz.

D) La ejecución de procesos estadísticos complejos.

3. ¿Qué arquitectura de datos fue introducida por Edgar F. Codd a principios de la década de 1970?

A) El paradigma NoSQL.

B) El esquema de organización jerárquica.

C) El modelo de tipo relacional. ✓

D) La estructura basada en grafos.

4. ¿Cuál es el estándar de lenguaje que se consolidó para la comunicación con sistemas relacionales?

A) JavaScript.

/B) HTML.

C) SQL (Structured Query Language). ✓

D) XML.

5. ¿Qué atributo técnico es propio de los sistemas de bases de datos NoSQL?

A) Rigidez en la disposición tabular de los datos.

B) Capacidad de crecimiento horizontal y gestión de contenidos no estructurados. ✓

C) Dependencia de claves foráneas y tablas vinculadas.

D) Restricción de uso en entornos de programación web.

6. Basándose en Elmasri y Navathe (2015), ¿cuáles son las ventajas competitivas del modelo relacional?

A) Problemas en la gobernanza de grandes activos de datos.

B) Preservación de la integridad, independencia de datos (lógica y física) y cumplimiento de reglas ACID. ✓

C) Inflexibilidad operativa frente a costos de implementación.

D) Vulnerabilidades críticas en la seguridad informática.

7. ¿Hacia dónde se dirige la evolución actual en la gestión de bases de datos?

A) La hegemonía absoluta de las soluciones relacionales.

B) La obsolescencia total de los esquemas relacionales.

C) La adopción de arquitecturas híbridas que integran capacidades relacionales y NoSQL. ✓

D) El retorno a los modelos de bases de datos jerárquicas.

8. De los siguientes ejemplos, ¿cuál corresponde a un motor NoSQL?

A) Oracle.

B) MySQL.

C) PostgreSQL.

D) MongoDB. ✓

9. ¿Cuál es la mayor virtud técnica que justifica el uso de NoSQL?

A) Eficiencia superior al procesar datos amorfos y facilidad de escalado. ✓

B) Rigor extremo en las normativas de normalización.

C) Unificación global mediante un solo lenguaje de consulta.

D) Limitaciones en la gestión de transacciones atómicas.

10. Atendiendo a la perspectiva de Coronel y Morris (2015), ¿qué variables determinan qué base de datos elegir para un software?

A) Únicamente la disposición de las tablas en el esquema.

B) El lenguaje de programación utilizado por el equipo.

C) La naturaleza de la información y los requerimientos funcionales del sistema. ✓

D) La localización geográfica de los servidores físicos.

UNIDAD 2 BASES DE DATOS RELACIONALES

Sistemas Relacionales: Arquitectura, Atributos y Fundamentos Operativos

En el ecosistema del desarrollo de software, la implementación de estructuras relacionales constituye un estándar para la administración de activos informáticos que requieren alta fidelidad. Su relevancia radica en la capacidad de transformar flujos de información dispersos en un repositorio centralizado, lógico y de alta disponibilidad.

Principios de Estructuración y Gestión

El núcleo de este paradigma se basa en una organización geométrica y lógica que define la interacción con el sistema:

- Organización Bidimensional: La información se dispone en esquemas de tablas, donde las tuplas representan las entradas de datos y los atributos definen las propiedades de cada entidad.
- Representación de Entidades: Este modelo facilita el mapeo de relaciones complejas entre distintos conjuntos de datos, asegurando una administración coherente de la información.
- Abstracción de Datos: Permite una autonomía operativa tanto en el plano físico como en el lógico, optimizando la recuperación de registros sin comprometer la estructura base.

Criterios de Calidad y Seguridad Transaccional

Para garantizar la robustez en entornos de producción, estas arquitecturas se rigen por estándares de cumplimiento estricto. Un aspecto determinante es la adherencia a las propiedades ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad), las cuales actúan como el protocolo de seguridad para la ejecución de transacciones complejas.

Según exponen Elmasri y Navathe (2016), la eficacia de estos sistemas no reside únicamente en su capacidad de almacenamiento masivo, sino en la integridad y seguridad con la que se procesa cada unidad de información en entornos concurrentes.

Fundamentos y Operatividad del Paradigma Relacional

En la ingeniería de software actual, la arquitectura de sistemas robustos y la escalabilidad de las soluciones dependen directamente de un diseño de datos coherente. El paradigma relacional ofrece el soporte teórico necesario para gestionar esta complejidad, garantizando que la información no solo se almacene, sino que mantenga su utilidad lógica a lo largo del tiempo.

Mecanismos de Estructuración y Álgebra de Datos

La operatividad de este modelo se articula a través de varios componentes esenciales que permiten una manipulación precisa de los activos informáticos:

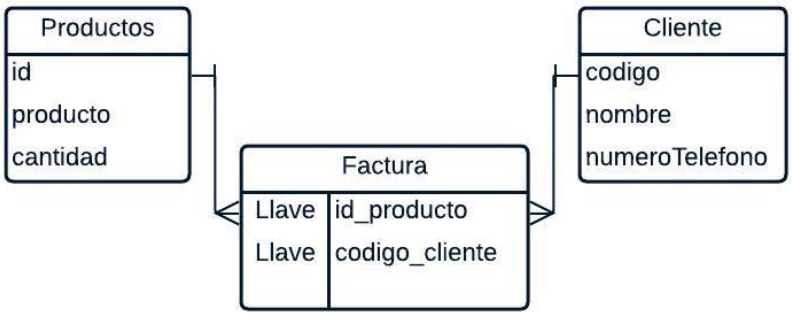
- Organización mediante Relaciones: Bajo este esquema, la información se segmenta en tablas, donde cada una actúa como una relación matemática definida.
- Procesamiento Analítico: El uso del álgebra relacional permite ejecutar operaciones fundamentales como la proyección, la unión y la selección, optimizando la recuperación de datos.
- Consistencia Sistémica: La aplicación de restricciones de integridad, específicamente mediante el uso de identificadores únicos (claves primarias) y vínculos referenciales (claves foráneas), asegura la estabilidad del ecosistema de datos.

Legado y Evolución Teórica

Este enfoque, cuya arquitectura lógica fue introducida originalmente por Edgar F. Codd en 1970, revolucionó la informática al separar la representación de los datos de su almacenamiento físico. De acuerdo con Coronel y Morris (2018), la adopción de este marco normativo es lo que permite a los desarrolladores actuales dotar de herramientas analíticas a sus aplicaciones, asegurando que la integridad de la información permanezca inalterable ante el crecimiento del sistema.

FIGURA 2

Esquema de integridad referencial y mapeo de entidades en un modelo relacional



Nota. Representación lógica de la interacción entre entidades (Productos, Factura, Cliente) mediante el uso de vínculos de dependencia y claves de relación. Elaboración propia basada en la metodología de Campaña (2025).

Instrumentos para el Control Transaccional: Motores Relacionales en la Industria

En el ecosistema tecnológico actual, el desarrollo de aplicaciones modernas exige el uso de motores que garanticen la seguridad, la replicación de activos y la gestión de transacciones complejas. Los Sistemas de Gestión de Bases de Datos Relacionales (RDBMS) funcionan como el software especializado que dota de herramientas analíticas a los desarrolladores para operar bajo el paradigma de tablas y relaciones.

Principales Soluciones y Entornos de Aplicación

La selección de un motor de datos depende directamente de los requerimientos de escalabilidad y el entorno operativo de la organización:

- Oracle Database: Se posiciona como el estándar para infraestructuras corporativas de gran envergadura debido a su alta capacidad de procesamiento y escalabilidad.
- PostgreSQL: Es reconocido en la comunidad técnica por su estricto cumplimiento de estándares, robustez estructural y capacidad de extensión.
- Microsoft SQL Server: Ofrece una sinergia nativa con los entornos de desarrollo del ecosistema Microsoft, facilitando el despliegue de soluciones empresariales integradas.
- MySQL: Representa la opción predilecta en el desarrollo web y proyectos de escala intermedia gracias a su naturaleza de código abierto y eficiencia.

Impacto en el Desarrollo de Software

La relevancia de estas tecnologías reside en su capacidad para ofrecer un marco de gestión integral de la información. De acuerdo con Coronel y Morris (2018), estos sistemas constituyen piezas fundamentales para asegurar la integridad de los datos en plataformas competitivas, permitiendo una administración eficiente y segura de los recursos digitales.

Evaluación de Capacidades y Restricciones en Entornos Relacionales

En el panorama del desarrollo de software, la elección de una arquitectura de datos depende del equilibrio entre el rigor normativo y la agilidad de respuesta. Los sistemas relacionales se consolidan

como el estándar para infraestructuras que priorizan la estabilidad y la gobernanza de activos informáticos sobre la flexibilidad absoluta.

Fortalezas y Atributos Operativos

La adopción de este modelo se justifica por la implementación de mecanismos que garantizan la fiabilidad del sistema:

- Garantía Transaccional: El cumplimiento del protocolo ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad) asegura la ejecución de operaciones seguras y predecibles.
- Normalización Estructural: La organización de los datos bajo esquemas normalizados reduce la redundancia y optimiza el almacenamiento lógico.
- Estandarización de Consultas: El uso del lenguaje SQL facilita una interacción analítica universal y compatible entre diversas plataformas.
- Preservación de la Integridad: Los motores relacionales aplican restricciones que mantienen la coherencia de la información en todo momento.

Limitaciones Técnicas y Desafíos de Modernización

A pesar de su robustez, estas arquitecturas enfrentan obstáculos ante las demandas de la computación contemporánea:

- Restricciones de Escalabilidad: Estos sistemas presentan dificultades intrínsecas para el escalado horizontal masivo en comparación con alternativas distribuidas.
- Gestión de Flujos no Estructurados: La rigidez del esquema tabular complica el procesamiento eficiente de grandes volúmenes de datos con formatos variables.
- Rendimiento en Tiempo Real: El mantenimiento de la consistencia estricta puede impactar la latencia en aplicaciones que requieren una respuesta inmediata y un crecimiento acelerado.

De acuerdo con Elmasri y Navathe (2016) y Coronel y Morris (2018), comprender estas ventajas y carencias es fundamental para que los ingenieros puedan diseñar soluciones que soporten eficazmente los retos de la transformación digital.

Cuestionario de Autoevaluación - Unidad 2: Sistemas de Datos Relacionales

1. ¿Qué arquitectura estructural emplea el paradigma relacional para la organización de los activos de información?

A) Una jerarquía de nodos ramificados.

B) Un esquema de almacenamiento basado en listas enlazadas.

C) Una matriz bidimensional compuesta por registros (tuplas) y campos (atributos). ✓

D) Un grafo de interconexiones lógicas.

2. De acuerdo con la cronología técnica, ¿quién fue el precursor que propuso el modelo relacional en 1970?

A) Donald D. Chamberlin.

B) Jim Gray.

C) Michael Stonebraker.

D) Edgar F. Codd. ✓

3. Según la perspectiva de Elmasri y Navathe (2016), ¿cuál es una propiedad crítica que distingue a los sistemas relacionales?

A) El procesamiento nativo de grandes volúmenes de datos no estructurados.

B) La garantía de integridad y coherencia de la información. ✓

C) La capacidad de escalado horizontal sin restricciones de esquema.

D) La gestión exclusiva de contenido multimedia semiestructurado.

3. En el mercado de soluciones de software, ¿cuál de los siguientes ejemplos NO se clasifica como un motor relacional (RDBMS)?

A) Microsoft SQL Server.

B) MongoDB. ✓

C) Oracle Database.

D) MySQL.

4. Dentro del marco del álgebra relacional, ¿cuál operación es fundamental para la recuperación selectiva de registros?

A) Procedimiento de replicación.

B) Técnica de normalización.

C) Operación de selección. ✓

D) Indexación de campos.

5. Según Coronel y Morris (2018), ¿qué mecanismo de control asegura la estabilidad de los vínculos entre distintas entidades?

A) El protocolo de atomicidad.

B) La integridad referencial gestionada mediante claves foráneas. ✓

C) La replicación de activos digitales.

D) El control de concurrencia lógica.

6. ¿Qué factor representa una ventaja competitiva de las bases de datos relacionales en entornos corporativos?

A) La adopción de una estructura normalizada que minimiza la redundancia. ✓

B) El soporte nativo para flujos informativos de rápido crecimiento y sin esquema.

C) Una escalabilidad horizontal ilimitada en sistemas distribuidos.

D) La baja dependencia de transacciones seguras bajo estándares ACID.

7. ¿Qué solución de gestión de datos es reconocida por su robustez técnica, extensibilidad y cumplimiento de estándares?

A) Neo4j.

B) Oracle Database.

C) PostgreSQL. ✓

D) Redis.

8. En el contexto de la seguridad transaccional, ¿cuál es el significado correcto del acrónimo ACID?

A) Autonomía, Coherencia, Independencia y Disponibilidad.

B) Acceso, Control, Integridad y Durabilidad.

C) Atomicidad, Consistencia, Aislamiento y Durabilidad. ✓

D) Administración, Concurrencia, Integridad y Disponibilidad.

9. ¿Cuál es una limitación reconocida de las arquitecturas relacionales frente a las demandas de la transformación digital?

A) La carencia de independencia lógica en la gestión de datos.

B) La inexistencia de un lenguaje de consulta estandarizado.

C) La vulnerabilidad estructural en procesos de seguridad.

D) La complejidad técnica para administrar volúmenes masivos de datos no estructurados. ✓

UNIDAD 3 BASES DE DATOS NOSQL

Sistemas NoSQL: Arquitectura Adaptable y Escalabilidad Distribuida

En el ecosistema tecnológico contemporáneo, la viabilidad de las plataformas de alta demanda, como redes sociales o infraestructuras de comercio electrónico, depende de su capacidad para procesar flujos masivos de activos digitales heterogéneos. Las bases de datos NoSQL han surgido como el componente esencial para estos entornos, permitiendo una gestión de la información que trasciende las limitaciones de los esquemas tradicionales.

Atributos Operativos y Capacidades Técnicas

A diferencia de los modelos convencionales, las arquitecturas no relacionales dotan de herramientas analíticas para responder a los retos del Big Data mediante las siguientes propiedades:

- Escalabilidad Horizontal: Facilita el crecimiento del sistema mediante la adición de nodos genéricos en clústeres, evitando la dependencia de un único servidor de altas prestaciones.
- Ausencia de Esquema Rígido: Permite el almacenamiento de registros sin la necesidad de definir estructuras predeterminadas, adaptándose orgánicamente a datos cambiantes o semiestructurados.
- Optimización del Rendimiento: Estos motores están diseñados para garantizar latencias mínimas y altas tasas de transferencia, incluso bajo condiciones de concurrencia masiva.

Versatilidad de Almacenamiento: Según el caso de uso, se emplean diversos paradigmas lógicos:

- Documentales: Gestión de información mediante estructuras tipo JSON (ej. MongoDB).
- Clave-Valor: Almacenamiento basado en diccionarios de alta velocidad (ej. Redis).
- Orientadas a Grafos: Modelado de interconexiones y redes de relaciones complejas (ej. Neo4j).
- Columnares: Estructuras optimizadas para el análisis masivo de datos (ej. Cassandra).

Impacto en el Desarrollo Contemporáneo

En esencia, el paradigma NoSQL se define por su agilidad operativa y su capacidad para el procesamiento distribuido en tiempo real. Este enfoque permite a las organizaciones diseñar

infraestructuras resilientes que prescinden de normativas estructurales estrictas en favor de un rendimiento superior y una flexibilidad total en la gestión de sus activos informáticos.

Como señalan Redmond y Wilson (2012), este modelo proporciona el marco teórico necesario para superar la rigidez del almacenamiento tabular, consolidándose como la columna vertebral de los sistemas que manejan volúmenes de datos en constante expansión.

Clasificación y Escenarios de Aplicación de las Arquitecturas NoSQL

En el ecosistema tecnológico actual, la selección de una infraestructura de datos depende directamente de la naturaleza de la información y los requisitos de rendimiento. El paradigma NoSQL se fragmenta en cuatro vertientes especializadas que dotan de herramientas analíticas para resolver desafíos de escalabilidad y flexibilidad.

Modelos de Almacenamiento y Casos de Uso

- Arquitecturas Documentales: Operan mediante el almacenamiento de registros en formatos semiestructurados como JSON, BSON o XML, emulando la gestión de objetos en el desarrollo de software. Son el estándar para plataformas móviles y web debido a su capacidad de albergar esquemas heterogéneos dentro de una colección única sin comprometer la estabilidad del sistema. Referentes en este ámbito son MongoDB y CouchDB.
- Sistemas de Clave-Valor: Estructuran la información mediante pares asociativos (diccionarios), optimizados para procesos que requieren una latencia mínima. Su eficiencia es superior en la gestión de cachés, sesiones de usuario y configuraciones de acceso instantáneo. Tecnologías como Redis, Amazon DynamoDB y Riak lideran este segmento.
- Modelos Orientados a Grafos: Diseñados para representar entidades como nodos y sus interconexiones como aristas, facilitando el mapeo de relaciones de alta complejidad. Son indispensables en el análisis de redes sociales, algoritmos de recomendación y optimización de rutas, destacando herramientas como Neo4j y ArangoDB por su agilidad en la navegación de vínculos profundos.
- Sistemas de Almacenamiento Columnar: A diferencia de los métodos tradicionales, organizan los activos de información por columnas, lo que permite ejecutar agregaciones y análisis

masivos sobre volúmenes de datos a gran escala. Motores como Apache Cassandra y HBase son fundamentales en soluciones de Business Intelligence y procesamiento analítico en tiempo real.

Impacto en la Ingeniería de Datos

La diversidad de estos modelos permite a los administradores de sistemas y desarrolladores implementar arquitecturas resilientes que se adaptan a la volatilidad del mercado digital. Como señalan Redmond y Wilson (2012), la comprensión de estas tipologías es crítica para garantizar que el rendimiento de la aplicación sea coherente con la estructura de sus datos.

Ecosistemas de Implementación: Análisis de Soluciones NoSQL

En la ingeniería de software contemporánea, la eficiencia operativa no depende únicamente del motor de datos, sino del conjunto de herramientas analíticas que facilitan su gestión. La transición hacia servicios gestionados y entornos visuales ha permitido que los desarrolladores optimicen el ciclo de vida de las aplicaciones de alta demanda.

Entorno de Trabajo y Herramientas para MongoDB

Más allá de su arquitectura documental, este sistema se integra en un ecosistema que automatiza la administración de la infraestructura y simplifica la interacción con los datos:

- Servicios en la Nube (MongoDB Atlas): Representa el paradigma de Database as a Service (DBaaS), delegando la configuración y el mantenimiento de servidores a una plataforma automatizada para centrar el esfuerzo en la lógica de negocio.
- Interfaces de Gestión Visual (GUI): Facilitan la exploración y manipulación de colecciones sin requerir exclusivamente el uso de la consola de comandos.
- Studio 3T y Robo 3T: Herramientas diseñadas para la construcción de consultas complejas y la visualización rápida de información, respectivamente.
- DbSchema: Especializada en la representación gráfica de esquemas y la vinculación lógica entre colecciones.

Entornos de Desarrollo Integrados (IDE)

Soluciones como DataGrip de JetBrains permiten centralizar la administración de sistemas relacionales y no relacionales bajo una misma interfaz operativa.

Optimización de Latencia y Despliegue en Redis

Redis se consolida como el estándar para escenarios que exigen una respuesta inmediata, operando primordialmente sobre la memoria volátil para eliminar los cuellos de botella del almacenamiento físico. Su implementación es fundamental en sistemas de mensajería instantánea, gestión de sesiones y analítica en tiempo real.

Modalidades de Despliegue:

- Redis Cloud / Enterprise: Versiones gestionadas que ofrecen alta disponibilidad y escalabilidad automática en la nube, reduciendo la complejidad de la configuración manual.
- Redis Open Source: Variante comunitaria óptima para el auto-hospedaje en servidores locales, experimentación y proyectos de escala reducida.

Versatilidad de Infraestructura

El motor permite una orquestación flexible mediante contenedores (Kubernetes), servidores físicos o nubes públicas, gestionándose a través de APIs REST o interfaces de línea de comandos.

Gestión Visual y Conectividad

Aunque su naturaleza es técnica y basada en terminal, la integración con plugins de IDEs y clientes ligeros dota de herramientas analíticas visuales que agilizan el monitoreo de datos en memoria.

Análisis de Capacidades y Restricciones en Ecosistemas NoSQL

En el marco de la arquitectura de software contemporánea, la elección de una solución no relacional responde a la necesidad de gestionar volúmenes masivos de activos digitales con alta eficiencia operativa. Este paradigma permite a los ingenieros diseñar infraestructuras que priorizan la agilidad y el rendimiento distribuido sobre la rigidez normativa de los modelos tradicionales.

Beneficios Operativos y Escalabilidad

- La implementación de estos sistemas dota de herramientas analíticas superiores para afrontar los retos de la transformación digital, destacando los siguientes puntos:
- Versatilidad de Estructura: La ausencia de esquemas predefinidos facilita el almacenamiento de datos heterogéneos, permitiendo que la base de datos evolucione junto con los requisitos de la aplicación sin interrupciones estructurales.
- Crecimiento Horizontal: En lugar de depender de la actualización de hardware único, estos sistemas permiten la expansión mediante clústeres de servidores estándar que distribuyen la carga de trabajo de manera equitativa.
- Rendimiento en Alta Concurrencia: Están optimizados para garantizar latencias mínimas y tiempos de respuesta extremadamente bajos, incluso ante picos de tráfico masivo de usuarios.
- Adaptabilidad a Casos de Uso: Gracias a la disponibilidad de diversos modelos (documentales, clave-valor, orientados a grafos o columnares), es posible seleccionar la arquitectura más coherente para escenarios específicos como el análisis de redes sociales o el e-commerce.

Limitaciones Técnicas y Desafíos Estratégicos

A pesar de su potencial, la adopción de infraestructuras NoSQL conlleva compromisos técnicos que deben ser evaluados con rigor:

- Fragmentación de Estándares: La carencia de un lenguaje de consulta universal dificulta la interoperabilidad y puede generar dependencia hacia un proveedor o tecnología específica (vendor lock-in).
- Consistencia Eventual: Con el fin de maximizar la disponibilidad y la velocidad, estos motores suelen sacrificar la consistencia inmediata, lo que puede derivar en la lectura temporal de datos no actualizados.
- Curva de Aprendizaje y Soporte: La transición desde el paradigma SQL hacia modelos distribuidos requiere una especialización técnica profunda. Asimismo, aunque el ecosistema es robusto, ciertas herramientas de soporte y comunidades de desarrolladores son menos extensas que las de los sistemas relacionales clásicos.

De acuerdo con Pandora FMS (2024) y Dongee (2023), la comprensión de este balance entre flexibilidad y rigor transaccional es imperativa para garantizar la sostenibilidad técnica de cualquier proyecto informático moderno.

Cuestionario de Autoevaluación - Unidad 3: bases de datos NoSQL

1. ¿A qué concepto hace referencia la denominación técnica "NoSQL" en el ámbito de la ingeniería de datos?

A) Nodo Seguro para Consultas Lógicas.

B) Nueva Orientación de Sistemas Lineales.

C) Not Only SQL (No solo SQL). ✓

D) Operación de Red de Secuencia Lógica.

2. ¿Bajo qué esquema organizativo operan convencionalmente los sistemas de gestión de datos relacionales (RDBMS)?

A) Registros basados en pares clave-valor.

B) Estructura de matrices bidimensionales (tablas con registros y campos). ✓

C) Documentos de texto semiestructurados.

D) Arquitecturas de grafos con interconexiones.

3. ¿Cuál es un rasgo distintivo de los motores de datos basados en el modelo relacional?

A) Ausencia total de atributos y entidades.

B) Implementación de esquemas predefinidos y estrictos. ✓

C) Exclusión de vínculos entre conjuntos de datos.

D) Especialización exclusiva en activos multimedia.

4. ¿Qué naturaleza de flujos informativos impulsó la necesidad de nuevas arquitecturas tras el auge de la Web 20 y la interacción digital masiva?

A) Registros estáticos y estructuras invariables.

B) Contenido heterogéneo (comentarios, registros de actividad, metadatos). ✓

C) Solo transacciones de carácter contable.

D) Unidades de texto plano sin codificación.

5. ¿Qué obstáculos técnicos limitaron el desempeño de las bases de datos tradicionales frente a los requisitos del Big Data?

A) Restricciones en agilidad, crecimiento distribuido y tasas de transferencia. ✓

B) Dificultades en el acceso a recursos de ofimática.

C) Incompatibilidad con protocolos de red inalámbrica.

D) Exceso de redundancia en sistemas de hojas de cálculo.

6. ¿Cuáles son las vertientes lógicas que integran el ecosistema de las soluciones NoSQL?

A) Exclusivamente almacenamiento en la nube.

B) Modelos de documentos, orientados a grafos, clave-valor y columnares. ✓

C) Formatos de archivos portátiles y binarios.

D) Sistemas basados en algoritmos de lógica difusa.

7. ¿Qué motor de datos es un referente en la administración de información mediante el paradigma documental (JSON/BSON)?

A) Neo4j.

B) Redis.

C) Cassandra.

D) MongoDB. ✓

8. Dentro de las infraestructuras de baja latencia, ¿cuál de los siguientes sistemas se especializa en el modelo de clave-valor?

A) PostgreSQL.

B) Redis. ✓

C) MongoDB.

D) Microsoft SQL Server.

9. ¿Qué solución tecnológica es la más adecuada para gestionar redes de relaciones complejas y análisis de interconexiones?

A) Neo4j. ✓

B) Apache Cassandra.

C) CouchDB.

D) MariaDB.

10. ¿Cuál es el beneficio estratégico de las arquitecturas NoSQL en términos de crecimiento infraestructural?

A) Optimización mediante la ampliación de recursos en un servidor central.

B) Despliegue de escalabilidad horizontal mediante la distribución en clústeres. ✓

C) Incremento manual del volumen de las particiones de disco.

D) Reducción del procesamiento de lenguajes de consulta.

UNIDAD 4 DISPARIDADES ESTRUCTURALES: ARQUITECTURAS RELACIONALES FRENTE A NO RELACIONALES

En el diseño de sistemas de información, la elección del modelo de datos determina la viabilidad de la infraestructura frente a cambios en los requerimientos del negocio. Mientras que el enfoque tradicional prioriza la consistencia mediante esquemas estrictos, las soluciones modernas ofrecen una maleabilidad orientada a objetos y flujos de datos dinámicos.

Respecto a:

Análisis Comparativo de la Organización de Datos

La tabla número 1 sintetiza las divergencias fundamentales en la gestión de esquemas y la vinculación de entidades:

Tabla 1

Diferenciación de arquitecturas lógicas: SQL vs. NoSQL.

Dimensión Técnica	Modelo Relacional (SQL)	Ecosistemas NoSQL
Paradigma Organizativo	Estructuras tabulares con esquemas rígidos; cada registro debe alinearse estrictamente a una definición de campos preestablecida.	Modelado orientado a colecciones de objetos o documentos; cada entrada permite una configuración de atributos heterogénea.
Gobernanza del Esquema	Estructura fija: Cualquier alteración en la definición de los datos exige una reestructuración global de la entidad (Alter Table).	Esquema dinámico: Facilita la inserción de nuevos campos en registros individuales sin comprometer la estabilidad del conjunto.
Interconexión Lógica	Basada en integridad referencial mediante identificadores únicos y claves ajenas para vincular conjuntos dispares.	Determinada por la topología del modelo (grafos, clave-valor, etc.), permitiendo relaciones embebidas o enlaces no restrictivos.
Contexto de Aplicación	Registros de carácter administrativo o contable con atributos invariables (ej. sistemas de facturación).	Entornos de contenido volátil y perfiles de usuario con datos opcionales (ej. plataformas de interacción social).

Nota. El paradigma SQL se fundamenta en la consistencia y la normalización de datos dentro de casillas predefinidas, mientras que NoSQL se caracteriza por una poli morfía que permite a cada unidad de información poseer una estructura independiente y adaptativa.

Estrategias de Crecimiento y Escalabilidad: SQL frente a NoSQL

En la ingeniería de plataformas digitales, la capacidad de respuesta ante el incremento de la demanda operativa define la viabilidad a largo plazo de un sistema. La elección entre el escalamiento de recursos centrales o la distribución de la carga de trabajo es el factor determinante para la arquitectura de datos.

La Tabla 2 detalla las divergencias técnicas entre la potenciación de hardware único y el despliegue de infraestructuras distribuidas:

Tabla 2

Diferencias en modelos de escalado de datos: Enfoque vertical vs. horizontal

Criterio Técnico	Modelo Relacional (SQL)	Ecosistemas NoSQL
Mecanismo de Expansión	Escalamiento Vertical: Se basa en la optimización de los recursos de un servidor central mediante el incremento de CPU, memoria volátil y almacenamiento físico.	Escalamiento Horizontal: Implementa una arquitectura distribuida que fragmenta la carga operativa entre múltiples nodos que operan de manera paralela.
Arquitectura de Carga	Depende de una unidad centralizada de procesamiento. Al aumentar el tráfico, se requiere hardware con mayores capacidades de cómputo.	Utiliza una red de clústeres donde se añaden máquinas estándar para procesar volúmenes masivos de peticiones simultáneas.
Beneficios Críticos	Garantiza una consistencia transaccional absoluta, al centralizar el control de la información en un repositorio único y robusto.	Ofrece una elasticidad superior, permitiendo un crecimiento prácticamente ilimitado mediante la adición de hardware común.
Restricciones Técnicas	Presenta un techo físico y económico: la actualización de	La gestión de datos entre múltiples servidores puede comprometer la consistencia

	hardware único llega a un límite de coste-beneficio insostenible.	inmediata en favor de la disponibilidad.

Nota. El paradigma SQL se asemeja a una estructura que gana altura y resistencia de forma centralizada, mientras que el modelo NoSQL funciona como una red urbana que se expande horizontalmente mediante la interconexión de múltiples unidades funcionales. Ambas arquitecturas resuelven problemas distintos bajo criterios de ingeniería específicos.

Modelos de Coherencia Informativa: Rigor Transaccional vs. Sincronización Eventual

En el desarrollo de sistemas críticos, la gestión de la integridad de los datos representa un factor decisivo. La elección entre una validación inmediata y una convergencia diferida determina no solo la fiabilidad del software, sino también su capacidad de respuesta en arquitecturas globales y distribuidas.

Análisis de la Gestión de la Integridad de Datos

La Tabla 3 presenta las divergencias fundamentales en los mecanismos de validación y sincronización de los activos digitales:

Tabla 3

Diferenciación de paradigmas de consistencia: SQL frente a NoSQL.

Dimensión Técnica	Enfoque Relacional (SQL)	Ecosistemas NoSQL
Protocolo de Validación	Consistencia Estricta: Se fundamenta en el cumplimiento riguroso de las propiedades ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad) para cada operación.	Consistencia Eventual: Los nodos del sistema pueden presentar estados temporales divergentes hasta alcanzar la convergencia final de la información.
Mecanismo de Control	Implementa procesos de verificación inmediata, asegurando que cualquier cambio sea validado y reflejado en el	Prioriza la disponibilidad mediante la propagación asíncrona de actualizaciones entre

	repositorio central sin excepciones.	los servidores que integran el clúster.
Fortalezas Operativas	Ofrece precisión absoluta, siendo la arquitectura indispensable para plataformas bancarias, sistemas de auditoría o registros clínicos.	Brinda agilidad y resiliencia en entornos con alta latencia, optimizando la experiencia del usuario en plataformas de escala masiva.
Compromisos Técnicos	La exigencia de una integridad constante puede degradar el rendimiento y elevar la complejidad en infraestructuras de gran envergadura.	Introduce un margen de incertidumbre temporal, donde los datos pueden no ser idénticos en todos los puntos de acceso durante breves intervalos.

Nota. Mientras que el modelo SQL actúa como un sistema de certificación en tiempo real que prohíbe las discrepancias, el modelo NoSQL funciona bajo un esquema de actualización distribuida que favorece la continuidad del servicio frente a la uniformidad inmediata de los registros.

Dinámicas de Adaptabilidad: Rigidez Estructural frente a Esquemas Polimórficos

En la ingeniería de software moderna, la capacidad de una base de datos para asimilar cambios en los requisitos del negocio sin degradar la operatividad es un factor determinante. La elección entre un diseño predefinido o uno adaptativo influye directamente en la velocidad de despliegue y en la resiliencia del sistema ante datos de naturaleza variable.

Evaluación de la Versatilidad del Esquema

La Tabla 4 desglosa las discrepancias técnicas entre la gobernanza estricta de SQL y la libertad estructural de los modelos NoSQL:

Tabla 4

Comparativa de maleabilidad y gestión de cambios: SQL vs. NoSQL.

Criterio Técnico	Paradigma Relacional (SQL)	Ecosistemas NoSQL

Configuración de Datos	Arquitectura Invariable: La persistencia se rige por esquemas fijos; cada entidad debe ajustarse a una definición de campos estandarizada.	Arquitectura Dinámica: Los registros o documentos poseen autonomía estructural, permitiendo atributos únicos por entrada.
Respuesta ante Modificaciones	La alteración del esquema (migraciones) es un proceso complejo que exige validación global y puede comprometer la disponibilidad del servicio.	La evolución de los datos es orgánica; se pueden integrar nuevas propiedades de forma inmediata sin reestructurar el repositorio existente.
Fortaleza Estratégica	Asegura una uniformidad absoluta, optimizando procesos donde la estabilidad de los metadatos es innegociable.	Facilita la iteración acelerada, resultando indispensable en proyectos con requerimientos volátiles o prototipado rápido.
Limitación Operativa	Presenta una baja tolerancia a la variabilidad informativa y a las actualizaciones de esquema frecuentes.	La carencia de un molde estricto puede derivar en una fragmentación de la coherencia si no se gestiona mediante lógica de aplicación.

Nota. Mientras que el modelo SQL se comporta como una estructura cuadriculada de alta precisión que demanda conformidad total, el modelo NoSQL funciona como un espacio multiforme donde cada unidad informativa puede adoptar una morfología independiente según las necesidades del flujo de datos.

Escenarios de uso

Criterios de Selección para Sistemas SQL

La decisión de implementar un motor relacional debe fundamentarse en la naturaleza de los flujos de trabajo. Este paradigma es la solución idónea bajo las siguientes condiciones:

- Rigurosidad en la Integridad: Esencial para ecosistemas financieros, contables o de gestión de nóminas donde la consistencia de los registros es innegociable.
- Seguridad Transaccional: Imprescindible en operaciones que requieren validación inmediata, tales como transferencias de activos o transacciones de comercio electrónico de alta fidelidad.
- Estabilidad de Esquemas: Recomendado cuando los metadatos (clientes, catálogos de productos, facturación) presentan estructuras invariables y bien definidas.
- Estandarización y Soporte: Ideal cuando se busca interoperabilidad, aprovechando que el lenguaje de consulta estructurado es un protocolo universalmente adoptado.
- Gestión de Carga Vertical: Óptimo para infraestructuras donde el crecimiento se puede absorber mediante la potenciación de los recursos de hardware del nodo central.

Fundamentos Operativos del Modelo Tradicional

El éxito persistente de las bases de datos relacionales, vigente desde su concepción teórica, radica en una arquitectura diseñada para la coherencia. Según postula IBM (s.f.), estas estructuras permiten la optimización de las dinámicas operativas y la detección de tendencias mediante la correlación de datos distribuidos en múltiples entidades lógicas.

Este rendimiento se sustenta en pilares técnicos específicos:

- Organización Matricial: Disposición de activos en esquemas de tablas con atributos y tuplas estrictamente delimitados.
- Soporte Transaccional ACID: Protocolos que aseguran la atomicidad, consistencia, aislamiento y durabilidad de cada proceso.
- Vínculos Referenciales: Mecanismos que, a través de claves primarias y ajenas, garantizan que la interconexión entre tablas sea lógica y sin errores.
- Protocolo Universal de Consulta: Empleo del lenguaje SQL para una administración y recuperación de información eficiente y estandarizada.

Escenarios de Aplicación y Pertinencia Técnica

La transición hacia modelos no relacionales es imperativa cuando el rendimiento y la adaptabilidad superan la necesidad de esquemas rígidos. De acuerdo con **Coursera (2024)**, este enfoque es la elección lógica en entornos donde la volatilidad de los datos y la velocidad de respuesta son factores críticos para el éxito operativo.

Las situaciones que demandan esta tecnología incluyen:

- Administración de Big Data: Procesamiento eficiente de registros provenientes de sensores (IoT) y archivos de registro (logs).
- Agilidad en el Ciclo de Vida: Implementación de esquemas polimórficos que facilitan actualizaciones inmediatas sin migraciones complejas.
- Crecimiento Distribuido: Capacidad de escalar horizontalmente mediante la incorporación de nodos en clústeres de servidores.
- Gestión de Atributos Heterogéneos: Almacenamiento de perfiles o activos informáticos con propiedades variables.

Flexibilidad Estructural en el Ecosistema Moderno

A diferencia de la organización tabular convencional, estas bases de datos operan mediante diversos modelos lógicos —como documentos, grafos, columnas distribuidas o pares clave-valor—, dotando de herramientas analíticas para enfrentar la complejidad del software contemporáneo (IBM, s.f.).

Esta versatilidad resulta determinante en sectores específicos:

- Plataformas de Interacción Social: Gestión de millones de interacciones y metadatos que mutan instantáneamente.
- Comercio Electrónico: Catálogos de productos con especificaciones técnicas disímiles y sistemas de valoraciones dinámicas.

- Servicios de Contenido Multimedia: Motores de recomendación que analizan patrones de consumo en tiempo real.
- Internet de las Cosas (IoT): Captura de señales provenientes de dispositivos periféricos sin formatos preestablecidos.

Como bien sostiene **Unzué (2025)**, la realidad tecnológica actual ha trascendido las limitaciones de las tablas y las operaciones de unión (JOINs), exigiendo arquitecturas que reflejen la diversidad y la celeridad del flujo informativo global. En definitiva, la adopción de NoSQL responde a una necesidad de resiliencia frente a la naturaleza cambiante del dato moderno (Coursera, 2024).

Marco Comparativo para la Selección de Arquitecturas

Para concluir el análisis técnico, es imperativo reconocer que la selección de una infraestructura de datos no responde a una jerarquía de superioridad, sino a una alineación estratégica con los requerimientos operativos del proyecto. En la ingeniería de software contemporánea, la decisión entre un modelo u otro define la capacidad de respuesta y la integridad de los activos digitales.

La **Tabla 5** sintetiza las divergencias fundamentales que deben orientar la toma de decisiones arquitectónicas:

Tabla 5

Evaluación técnica diferencial entre sistemas relacionales y no relacionales.

Criterio Técnico	Paradigma Relacional (SQL)	Ecosistemas NoSQL
Arquitectura de Datos	Esquemas tabulares de carácter rígido	Modelos de persistencia polimórficos y flexibles
Estrategia de Crecimiento	Escalabilidad de tipo vertical (potenciación de nodo)	Escalabilidad de tipo horizontal (distribución de carga)

Gobernanza de Integridad	Coherencia inmediata y estricta	Sincronización de naturaleza eventual
Áreas de Aplicación	Auditoría, finanzas y registros contables	Analítica masiva, redes sociales y Big Data

Nota. Como sostiene **IBM (s.f.)**, aunque las estructuras relacionales mantienen su vigencia en infraestructuras de legado y sistemas de alta precisión, la agilidad de los entornos NoSQL es fundamental para enfrentar los desafíos de la computación distribuida y la ausencia de esquemas predefinidos.

Tendencias futuras

El futuro de las bases de datos se parece mucho a cómo evoluciona nuestra vida diaria: seguimos necesitando lo clásico y confiable, pero también buscamos nuevas herramientas que nos den más libertad y rapidez. SQL, con su orden y precisión, seguirá siendo el guardián de los sistemas críticos, como un contador meticuloso que nunca se equivoca. Al mismo tiempo, NoSQL crecerá porque el mundo moderno está lleno de datos desordenados y cambiantes, como las publicaciones en redes sociales o los registros de sensores. Y lo más emocionante es que la inteligencia artificial empezará a darle "vida" a las bases de datos, ayudándolas a detectar patrones y anticiparse a lo que necesitamos (Romanowski, 2025; DbaExperts, 2025).

También veremos cómo la nube híbrida se convierte en el nuevo hogar de la información: un espacio flexible que combina lo mejor de lo público y lo privado, como tener tu propia casa, pero con acceso a un edificio compartido cuando necesitas más espacio. A esto se suma el Big Data y la analítica avanzada, que permitirán procesar información en tiempo real y tomar decisiones más inteligentes. En definitiva, el futuro de las bases de datos no es reemplazar lo que ya existe, sino unir lo tradicional con lo innovador, creando un ecosistema

donde la disciplina, la creatividad y la velocidad trabajen juntas para que la información se convierta en un recurso estratégico (IEBS Business School, 2025).

Conclusiones

NoSQL no reemplaza a SQL, sino que lo complementa. Muchas empresas usan ambos: SQL para lo estructurado y crítico, NoSQL para lo flexible y masivo. La decisión final debe basarse en la pregunta: ¿necesitas orden y consistencia absoluta, o velocidad y adaptabilidad?

Referencias

Campaña, R. R. (2025). Figuras de bases de datos. Elaboración propia.

Coronel, C., & Morris, S. (2015). Database systems: Design, implementation, and management. Cengage Learning.

Coronel, C., & Morris, S. (2018). Database systems: Design, implementation, and management. Cengage Learning.

Coronel, C., & Morris, S. (2021). Database systems: Design, implementation, and management. Cengage Learning.

Dongee. (2023). Desventajas de NoSQL. Recuperado de https://dongee.com/

Elmasri, R., & Navathe, S. B. (2015). Fundamentals of database systems (7th ed.). Pearson.

Guru99. (2025). MongoDB tutorial. Recuperado de https://www.guru99.com/

IBM. (s.f.). Redis overview. Recuperado de https://www.ibm.com/

JetBrains. (s.f.). DataGrip. Recuperado de https://www.jetbrains.com/datagrip/

MongoDB. (s.f.). MongoDB Atlas. Recuperado de https://www.mongodb.com/

Pandora FMS. (2024). Ventajas de NoSQL. Recuperado de https://pandorafms.com/

Redmond, E., & Wilson, J. R. (2012). Seven databases in seven weeks: A guide to modern databases and the NoSQL movement. Pragmatic Bookshelf.

Redis. (s.f.). Redis Enterprise. Recuperado de https://redis.io/

Stackscale. (2023). Características de bases de datos NoSQL. Recuperado de https://www.stackscale.com/es/

Wikipedia. (2025). Redis. Recuperado de https://es.wikipedia.org/wiki/Redis

INSTITUTO SUPERIOR UNIVERSITARIO sucre

ISBN: 978-9942-590-16-9

SUCREInstitutooficia @SUCREInstituto @SUCREInstituto