**Universidad Don Bosco**

Ingeniería en Ciencias de la Computación

Diagrama

Descripción generada automáticamente

**Foro 1**

**Carrera**: Ingeniería en Ciencias de la Computación

**Materia:** Diseño y Programación de Software Multiplataforma (G01T)

**Presentado por:**

* Villalta Reinoza, Luis Edgardo VR181981

**Entrega:** 29 de octubre, 2023

**Ciclo 02 – 2023**

**Contenido**

[Introducción 4](#_Toc149415343)

[Bases de datos SQL 5](#_Toc149415344)

[Ventajas de las bases de datos SQL 6](#_Toc149415345)

[Desventajas de las bases de datos SQL 7](#_Toc149415346)

[Bases de datos NoSQL 8](#_Toc149415347)

[Ventajas de las bases de datos NoSQL 9](#_Toc149415348)

[Desventajas de las bases de datos NoSQL 10](#_Toc149415349)

[¿Cuáles son las diferencias fundamentales entre bases de datos SQL y NoSQL? 11](#_Toc149415350)

[Modelo de Datos: 11](#_Toc149415351)

[Esquema de Datos: 11](#_Toc149415352)

[Escalabilidad: 11](#_Toc149415353)

[Consulta: 11](#_Toc149415354)

[Consistencia de Datos: 12](#_Toc149415355)

[Casos de Uso: 12](#_Toc149415356)

[¿Cuáles son las diferencias específicas entre Cloud Firestore y Realtime Database? 13](#_Toc149415357)

[Modelo de Datos: 13](#_Toc149415358)

[Consultas: 13](#_Toc149415359)

[Escalabilidad: 13](#_Toc149415360)

[Facturación: 14](#_Toc149415361)

[Conexiones en Tiempo Real: 14](#_Toc149415362)

[¿Cuál de estas bases de datos consideran que sería la mejor opción para implementar en una aplicación desarrollada en React Native? 15](#_Toc149415363)

[Cloud Firestore (Firestore) sería la mejor opción si: 15](#_Toc149415364)

[Realtime Database sería la mejor opción si: 15](#_Toc149415365)

[Pruebas de firestore 16](#_Toc149415366)

[Alumnos 16](#_Toc149415367)

[Notas 16](#_Toc149415368)

[Campus 17](#_Toc149415369)

[Catedraticos 17](#_Toc149415370)

# Introducción

Las bases de datos son una parte fundamental en el mundo de la informática, utilizadas para almacenar y gestionar información de manera estructurada. Dos enfoques principales son las bases de datos SQL (Structured Query Language) y NoSQL (Not Only SQL). Las bases de datos SQL son conocidas por su capacidad de mantener datos altamente estructurados y relaciones complejas, utilizando un lenguaje de consulta estructurado. En contraste, las bases de datos NoSQL son más flexibles y pueden gestionar datos no estructurados o semiestructurados, ideales para aplicaciones web escalables y entornos de big data. La elección entre SQL y NoSQL depende de los requisitos específicos del proyecto, y cada enfoque tiene sus propias ventajas y desafíos.

# Bases de datos SQL

Las bases de datos SQL, abreviatura de "Structured Query Language" en inglés, son sistemas de gestión de bases de datos relacionales (RDBMS) que utilizan un lenguaje de consulta específico llamado SQL para administrar y consultar datos. En una base de datos SQL, los datos se organizan en tablas con filas y columnas, y las relaciones entre los datos se definen mediante claves primarias y extranjeras. SQL permite realizar consultas complejas para recuperar, insertar, actualizar y eliminar datos de manera eficiente. Ejemplos de bases de datos SQL populares incluyen MySQL, PostgreSQL, Microsoft SQL Server y Oracle Database. Estas bases de datos son ideales para aplicaciones que requieren estructuras de datos bien definidas y relaciones sólidas entre los conjuntos de datos.

Una de las ventajas principales de las bases de datos SQL es su capacidad para garantizar la integridad de los datos y mantener relaciones complejas entre ellos. Además, ofrecen un alto nivel de consistencia y permiten realizar consultas sofisticadas a través del lenguaje SQL. Sin embargo, pueden ser menos flexibles en comparación con las bases de datos NoSQL y pueden no ser la elección ideal para aplicaciones que requieren escalabilidad horizontal extrema o que manejan datos no estructurados.

## Ventajas de las bases de datos SQL

* **Integridad de Datos:** Las bases de datos SQL garantizan la integridad de los datos mediante la aplicación de restricciones y reglas, como claves primarias, claves foráneas y restricciones de unicidad, lo que ayuda a mantener la calidad y la coherencia de los datos.
* **Transacciones ACID**: Utilizan el modelo ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad) para garantizar la fiabilidad de las transacciones.
* **Lenguaje SQL**: SQL proporciona un lenguaje de consulta poderoso y estandarizado que facilita la recuperación, actualización y manipulación de datos.
* **Soporte para Relaciones:** Las bases de datos SQL son ideales para representar y gestionar relaciones complejas entre los datos.
* **Seguridad:** Proporcionan capacidades de seguridad sólidas, incluida la autenticación de usuarios, control de acceso y cifrado de datos, lo que ayuda a proteger la información confidencial.
* **Escalabilidad Vertical:** Aunque su escalabilidad es más limitada en comparación con las bases de datos NoSQL, las bases de datos SQL pueden escalarse verticalmente agregando recursos a un servidor, lo que las hace adecuadas para aplicaciones empresariales con cargas de trabajo moderadas o predecibles.

## Desventajas de las bases de datos SQL

* **Esquema Rígido:** El esquema de una base de datos SQL es rígido y debe definirse de antemano.
* **Escalabilidad Limitada:** Las bases de datos SQL tienden a escalar verticalmente, lo que significa que para aumentar su capacidad, se requieren servidores más potentes.
* **Rendimiento en Lectura/Escritura Masiva:** En comparación con las bases de datos NoSQL diseñadas específicamente para grandes volúmenes de lectura/escritura, las bases de datos SQL pueden ser menos eficientes en entornos con alta concurrencia y carga de trabajo intensiva.
* **Modelo de Datos Complejo:** Aunque la estructura en tablas es beneficiosa para mantener relaciones complejas, también puede volverse complicada en sistemas con múltiples tablas y esquemas.
* **Costo Inicial y Licenciamiento:** Las soluciones de bases de datos SQL comerciales a menudo requieren inversiones iniciales significativas en licencias y hardware especializado, lo que puede ser un obstáculo financiero para algunas organizaciones.
* **No es la mejor opción para datos no estructurados:** Aunque pueden manejar datos semiestructurados, las bases de datos SQL no son la elección ideal para datos no estructurados o altamente variables, como texto sin formato o datos de sensores en tiempo real.

# Bases de datos NoSQL

Las bases de datos NoSQL, que significa "Not Only SQL" o "No Solo SQL", son sistemas de gestión de bases de datos diseñados para gestionar datos que no se ajustan bien a la estructura tabular de las bases de datos SQL tradicionales. A diferencia de las bases de datos SQL, las NoSQL son altamente flexibles y pueden manejar datos no estructurados, semiestructurados o estructurados de manera menos rígida. Estos sistemas de bases de datos NoSQL a menudo se utilizan en aplicaciones web modernas, análisis de big data, entornos de rápido crecimiento y proyectos donde la escalabilidad horizontal es fundamental.

Un aspecto destacado de las bases de datos NoSQL es su capacidad para escalar horizontalmente de manera eficiente, lo que significa que se pueden agregar nuevos servidores para manejar un mayor volumen de datos y tráfico, lo que las hace ideales para aplicaciones con crecimiento impredecible o masivo. Además, los sistemas NoSQL utilizan diversos modelos de datos, como bases de datos de documentos, bases de datos de gráficos, bases de datos clave-valor y bases de datos de columnas, lo que permite a los desarrolladores seleccionar el modelo que mejor se adapte a las necesidades específicas de su aplicación.

## Ventajas de las bases de datos NoSQL

* **Escalabilidad Horizontal:** Las bases de datos NoSQL están diseñadas para escalar horizontalmente, lo que significa que pueden manejar grandes volúmenes de datos y tráfico agregando nuevos servidores de manera eficiente.
* **Flexibilidad en el Esquema:** Las bases de datos NoSQL permiten un esquema dinámico o flexible, lo que facilita la adaptación a cambios en la estructura de datos sin necesidad de una reestructuración completa de la base de datos.
* **Rendimiento en Lectura/Escritura Masiva:** Estas bases de datos suelen ofrecer un alto rendimiento en entornos de lectura y escritura intensiva, lo que es beneficioso para aplicaciones con grandes cantidades de datos en tiempo real, como redes sociales y sistemas de seguimiento.
* **Modelos de Datos Variados:** Las bases de datos NoSQL admiten diversos modelos de datos, como documentos, gráficos, clave-valor y columnas.
* **Economía de Almacenamiento:** Para aplicaciones que manejan datos no estructurados o semiestructurados, las bases de datos NoSQL pueden resultar más económicas en términos de almacenamiento, ya que no requieren una estructura tabular fija.
* **Alta Disponibilidad:** Muchas bases de datos NoSQL están diseñadas para funcionar en entornos distribuidos y ofrecen mecanismos de alta disponibilidad, lo que garantiza la disponibilidad de los datos incluso en caso de fallos de hardware o red.

## Desventajas de las bases de datos NoSQL

* **Falta de Estandarización:** Dado que existen varios tipos de bases de datos NoSQL, no hay un estándar único para el lenguaje de consulta o la administración de datos.
* **Complejidad de Desarrollo:** La flexibilidad en el esquema puede llevar a una mayor complejidad en el desarrollo de aplicaciones, ya que los desarrolladores a menudo deben lidiar con la gestión de relaciones y la coherencia de datos a nivel de aplicación.
* **Menos Soporte para Transacciones ACID:** Aunque algunas bases de datos NoSQL admiten transacciones, en general, no ofrecen el mismo nivel de garantía de transacciones ACID que las bases de datos SQL, lo que puede ser un problema en aplicaciones donde la integridad de los datos es crítica.
* **Menos Adecuadas para Datos Relacionales:** Las bases de datos NoSQL suelen ser menos adecuadas para aplicaciones que requieren relaciones complejas y consultas de unión entre múltiples tablas, como las aplicaciones empresariales tradicionales.
* **Curva de Aprendizaje:** Para los desarrolladores acostumbrados a trabajar con bases de datos SQL, la transición a bases de datos NoSQL puede requerir tiempo para aprender nuevos modelos de datos y enfoques de consulta.
* **Seguridad y Control de Acceso:** Algunas bases de datos NoSQL pueden tener limitaciones en términos de autenticación y control de acceso, lo que podría plantear problemas de seguridad en ciertos escenarios.

# ¿Cuáles son las diferencias fundamentales entre bases de datos SQL y NoSQL?

## Modelo de Datos:

* **SQL:** Utilizan un modelo de datos tabular o relacional, donde la información se organiza en tablas con filas y columnas, y las relaciones entre los datos se mantienen mediante claves primarias y extranjeras.
* **NoSQL:** Emplean diversos modelos de datos, como bases de datos de documentos, bases de datos de gráficos, bases de datos clave-valor y bases de datos de columnas.

## Esquema de Datos:

* **SQL:** Requieren un esquema fijo y rígido, que debe definirse antes de ingresar los datos. Cambiar el esquema puede ser complicado.
* **NoSQL:** Ofrecen un esquema dinámico o flexible, lo que permite agregar campos o cambiar la estructura de los datos sin necesidad de reestructuración completa.

## Escalabilidad:

* **SQL:** Escalan principalmente verticalmente, lo que implica agregar recursos más potentes a un solo servidor. La escalabilidad horizontal es limitada.
* **NoSQL:** Escalan horizontalmente, lo que permite agregar nuevos servidores para manejar un mayor volumen de datos y tráfico de manera más eficiente.

## Consulta:

* **SQL:** Utilizan el lenguaje SQL (Structured Query Language) para realizar consultas complejas y ad-hoc.
* **NoSQL:** Las consultas pueden variar según el modelo de datos utilizado y a veces no ofrecen la misma flexibilidad que SQL.

## Consistencia de Datos:

* **SQL:** Garantizan la consistencia de datos utilizando el modelo ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad) en transacciones.
* **NoSQL:** Algunas bases de datos NoSQL sacrifican la consistencia estricta en aras de la disponibilidad y la tolerancia a fallos (modelo BASE: Basically Available, Soft state, Eventually consistent).

## Casos de Uso:

* **SQL:** Son ideales para aplicaciones donde la integridad de los datos y las relaciones complejas son críticas, como sistemas financieros y aplicaciones empresariales tradicionales.
* **NoSQL:** Se utilizan en aplicaciones web escalables, análisis de big data, aplicaciones de redes sociales, sistemas de seguimiento y otros escenarios donde la flexibilidad y la escalabilidad son esenciales.

# ¿Cuáles son las diferencias específicas entre Cloud Firestore y Realtime Database?

## Modelo de Datos:

* **Cloud Firestore:** Utiliza un modelo de base de datos de documentos en colecciones. Los datos se organizan en documentos que pueden anidarse en colecciones. Cada documento se almacena como un objeto JSON y admite campos más complejos y consultas más avanzadas.
* **Realtime Database:** Emplea un modelo JSON de árbol jerárquico, donde los datos se almacenan en nodos. A diferencia de Firestore, no ofrece documentos o colecciones, lo que puede hacer que la estructuración de datos más compleja sea un desafío.

## Consultas:

* **Cloud Firestore:** Ofrece consultas más avanzadas y flexibles, incluyendo filtrado, ordenación y combinación de múltiples campos en una sola consulta. Las consultas pueden recuperar datos con mayor precisión.
* **Realtime Database:** Las consultas son más limitadas y no admiten combinación de filtros en una sola consulta. Esto puede requerir estructuras de datos más complejas para ciertos tipos de consultas.

## Escalabilidad:

* **Cloud Firestore:** Es altamente escalable y permite un mayor rendimiento en consultas y lectura/escritura de datos. Es una opción sólida para aplicaciones con altos volúmenes de tráfico y necesidades de escalabilidad.
* **Realtime Database:** Aunque es escalable, su escalabilidad no es tan avanzada como la de Firestore, lo que puede resultar en problemas de rendimiento en aplicaciones extremadamente populares.

## Facturación:

* **Cloud Firestore:** Se factura según la cantidad de operaciones de lectura, escritura y almacenamiento de datos, lo que puede ser más predecible en términos de costos.
* **Realtime Database:** La facturación se basa en el ancho de banda de datos transferidos, lo que puede ser más difícil de prever y controlar.

## Conexiones en Tiempo Real:

* **Cloud Firestore:** Permite la sincronización en tiempo real de datos, pero no es tan eficiente como Realtime Database para aplicaciones que requieren actualizaciones extremadamente rápidas.
* **Realtime Database:** Ofrece sincronización en tiempo real altamente eficiente, lo que lo hace ideal para aplicaciones en tiempo real, como chats y juegos.

# ¿Cuál de estas bases de datos consideran que sería la mejor opción para implementar en una aplicación desarrollada en React Native?

La elección entre Cloud Firestore y Realtime Database para una aplicación desarrollada en React Native dependerá de los requisitos específicos de tu aplicación. Ambas bases de datos son compatibles con React Native, por lo que puedes utilizar cualquiera de ellas. Aquí hay algunas consideraciones que pueden ayudarte a tomar una decisión:

## Cloud Firestore (Firestore) sería la mejor opción si:

1. Necesitas realizar consultas más complejas y avanzadas en tus datos.
2. Tu aplicación tiene requisitos de escalabilidad y esperas un alto volumen de tráfico.
3. Deseas una base de datos que se adapte bien a aplicaciones en crecimiento y escalables.
4. Valoras la capacidad de estructurar tus datos en documentos y colecciones, lo que facilita la organización y consulta de la información.

## Realtime Database sería la mejor opción si:

1. Tu aplicación se centra en características en tiempo real, como chats, juegos multijugador o colaboración en tiempo real.
2. Valoras la sincronización en tiempo real de datos, ya que Realtime Database es especialmente eficiente en este aspecto.
3. La estructura jerárquica de datos en formato JSON se adapta a las necesidades de tu aplicación.

# Pruebas de firestore

## Alumnos

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

## Notas

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

## Campus

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

## Catedraticos

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente