4.2. Justificación Arquitectónica

Por qué se utilizó SQL para determinado tipo de información

SQL fue seleccionado para manejar datos que requieren una estructura clara, relaciones formales y un alto nivel de consistencia.

Este motor relacional permite definir esquemas estrictos que proporcionan control, validación y una organización precisa del modelo de datos.

Su soporte para transacciones ACID garantiza que operaciones críticas —como registros financieros, pedidos, inventarios o procesos administrativos—

se ejecuten correctamente aún bajo cargas altas. Además, SQL facilita la creación de consultas avanzadas mediante JOINs, agrupaciones,

subconsultas y operaciones analíticas, permitiendo obtener reportes confiables y completos.

En sistemas donde se necesita integridad referencial, trazabilidad y protección contra inconsistencias, SQL es la mejor opción.

Por qué se eligió MongoDB para otros tipos de datos

MongoDB fue elegido para datos que no encajan de forma natural en una estructura rígida, ya que permite un modelo flexible basado en documentos.

Esto facilita almacenar información que cambia con frecuencia o que tiene formatos diversos sin necesidad de modificar un esquema global.

MongoDB escala horizontalmente con facilidad, lo que lo hace ideal para manejar grandes volúmenes de datos distribuidos, como registros de actividad,

perfiles dinámicos, catálogos variables o información generada por los usuarios. Además, su rendimiento en lecturas y escrituras rápidas lo convierte

en una excelente herramienta para sistemas que requieren respuestas ágiles sin depender de relaciones complejas entre entidades.

La naturaleza del modelo JSON también hace más intuitive la integración con aplicaciones modernas.

Por qué Redis aporta valor en términos de rendimiento o simplicidad

Redis complementa la arquitectura al ofrecer operaciones extremadamente rápidas gracias a su almacenamiento en memoria.

Este motor se emplea principalmente como caché para reducir la cantidad de consultas realizadas a SQL y MongoDB, minimizando la latencia y aliviando la carga del sistema.

Redis permite guardar temporalmente información de uso frecuente, como sesiones de usuarios, resultados de consultas complejas o datos que deben actualizarse

en tiempo real. Además, ofrece estructuras ligeras como listas, conjuntos y contadores que permiten manejar colas de mensajes, sistemas de notificaciones

y estadísticas instantáneas. Su simplicidad en el uso como base de clave-valor y su velocidad lo vuelven fundamental para mejorar el desempeño general.

Beneficios y riesgos principales de la arquitectura de datos propuesta

La arquitectura híbrida ofrece un equilibrio entre rendimiento, flexibilidad y confiabilidad.

Entre los beneficios destaca la posibilidad de utilizar cada tecnología para el tipo de dato que mejor maneja,

logrando un sistema más eficiente y escalable. SQL aporta estabilidad y relaciones fuertes, MongoDB proporciona adaptabilidad

a datos cambiantes y Redis incrementa el rendimiento al acelerar las respuestas del sistema.

Este diseño facilita soportar crecimiento en volumen de información, nuevas funcionalidades y diferentes patrones de uso.

Sin embargo, también existen riesgos. La combinación de varios motores incrementa la complejidad del mantenimiento,

ya que requiere monitoreo, sincronización y estrategias claras para evitar inconsistencias.

También puede elevar los costos debido a la necesidad de infraestructura adicional y profesionales capacitados.

La correcta definición de qué datos van en cada sistema es clave para evitar duplicación de información o cuellos de botella.

Aun así, cuando se gestiona adecuadamente, la arquitectura híbrida ofrece un entorno robusto, flexible y altamente optimizado.