Análisis de Caso: Migración de Base de Datos para Empresa de Tecnología

Rol: Arquitecto de Bases de Datos

1. Resumen Ejecutivo

La empresa enfrenta problemas de **rendimiento y escalabilidad** en su base de datos relacional tradicional debido al crecimiento en clientes y transacciones. Se propone migrar a una solución **híbrida**:

- **Bases de datos NoSQL** (MongoDB o Cassandra) para datos de alto volumen y baja estructura (ej: logs, interacciones de usuarios).
- **RDBMS** (PostgreSQL) para transacciones críticas que requieren ACID. **Justificación:** Equilibrio entre escalabilidad horizontal (NoSQL) y consistencia (RDBMS), reduciendo costos operativos.

2. Análisis del Problema

Situación actual (RDBMS tradicional):

- **Cuellos de botella:** Consultas lentas por JOINs en tablas con millones de registros.
- **Escalabilidad vertical limitada:** Costos elevados al aumentar hardware.
- **Alta latencia:** En operaciones concurrentes (ej: picos de tráfico). **Causas raíz:**
- Modelo rígido para datos semiestructurados (ej: metadata de usuarios).
- Ineficiencia en distribuir carga en sistemas monolíticos.

3. Comparación de Tecnologías

Tecnología	Ventajas	Desventajas	Caso de Uso Ideal
PostgreSQL	ACID, soporte para JSON, escalabilidad read-replica.	Escalabilidad horizontal compleja.	Transacciones financieras, catálogos.
MongoDB	Escalabilidad horizontal, esquema flexible.	Consistencia eventual, JOINs ineficientes.	Perfiles de usuarios, contenido dinámico.
Cassandra	Alta disponibilidad, tolerancia a particiones.	Sin soporte para transacciones complejas.	Time-seri

Tabla Comparativa:

Criterio	PostgreSQL	MongoDB	Cassandra
Escalabilidad	Vertical	Horizontal	Horizontal
Consistencia	Fuerte (ACID)	Eventual	Eventual
Costo Operativo	Alto	Medio	Bajo

4. Propuesta de Solución

Arquitectura Híbrida:

- **RDBMS (PostgreSQL):**
  - Usar para **transacciones críticas** (pagos, gestión de inventario).
  - Aprovechar particionamiento y réplicas de lectura para mejorar rendimiento.
- **NoSQL (MongoDB):**
  - Almacenar **datos de perfil de usuarios** y **logs de interacciones**.
  - Escalabilidad automática en la nube (ej: MongoDB Atlas).
- **Caché (Redis):**
  - Acelerar consultas frecuentes (ej: productos populares).

Beneficios:

- **Reducción de costos:** Escalabilidad horizontal en NoSQL vs. costos de hardware en RDBMS.
- **Rendimiento:** Consultas rápidas para datos no estructurados.
- **Flexibilidad:** Adaptación a futuros crecimientos.

Desafíos:

- Complejidad en sincronización entre sistemas (ej: consistencia eventual).
- Curva de aprendizaje para equipos acostumbrados a SQL.

5. Conclusiones y Recomendaciones

- **Priorizar MongoDB** para datos no críticos y alta escalabilidad.
- **Mantener PostgreSQL** para operaciones con requerimientos ACID.
- **Implementar Redis** como capa de caché.

Pasos siguientes:

1. Realizar un **POC** (Prueba de Concepto) con MongoDB para un módulo no crítico.
2. Monitorear métricas de rendimiento (latencia, throughput) post-migración.
3. Capacitar al equipo en modelos de datos NoSQL.