Justificación Arquitectónica de Bases de Datos para un Sistema Bancario Simple

Este documento explica las decisiones técnicas detrás de la selección de PostgreSQL, MongoDB y Redis para optimizar el almacenamiento de datos en nuestro sistema bancario simple. Cada elección se alinea con el tipo de información y los requisitos de rendimiento y consistencia específicos de la banca.

PostgreSQL se utiliza para la gestión de datos críticos que demandan alta consistencia y transaccionalidad, tales como cuentas bancarias, transacciones, saldos, movimientos y la auditoría de operaciones financieras.

MongoDB se implementa para almacenar datos con esquemas flexibles y de alto volumen, incluyendo preferencias de usuario bancario, notificaciones de transacciones, historial de login y configuración de alertas.

Redis es esencial para operaciones de alta velocidad y tiempo real, como el caché de saldos frecuentes, tokens de sesión de usuarios y contadores de transacciones en tiempo real.

La Columna Vertebral de los Datos Bancarios: Una Visión General

La elección de PostgreSQL, MongoDB y Redis para nuestro sistema bancario simple no es aleatoria; cada uno se selecciona por su adecuación a las particularidades de los datos financieros y las exigencias de rendimiento, consistencia y seguridad. A continuación, detallamos la justificación para cada motor de base de datos en su rol específico dentro de nuestra infraestructura bancaria.

```
/sql/
   modelo_conceptual.pdf
   modelo_relacional.pdf
   create_tables.sql
   insert_data.sql
   queries_avanzadas.sql

/mongodb/
   diseño_colecciones.md
   inserts.json
   consultas_aggregation.md

/redis/
   comandos_basicos.txt
   operaciones_estructuras.txt
   casos_de_uso_redis.md

/documentacion/
   arquitectura_de_datos.pdf
   conexion_entre_las_3_bases.m

README.md
```

PostgreSQL (SQL): Integridad para Cuentas y Transacciones Bancarias

PostgreSQL, un robusto sistema de gestión de bases de datos relacional, es el pilar para gestionar información bancaria crítica. Su cumplimiento con las propiedades ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad) lo hace insustituible para el manejo de cuentas bancarias, el registro preciso de transacciones y saldos, los movimientos de dinero, y la auditoría rigurosa de todas las operaciones financieras, asegurando la máxima fiabilidad y seguridad.

PostgreSQL: Pilar para la Integridad de Datos Bancarios

Cuentas Bancarias y Saldos

Requieren integridad absoluta para saldos y movimientos en un sistema bancario. PostgreSQL impone restricciones como llaves primarias y foráneas, garantizando que cada cuenta y transacción mantenga su consistencia y fiabilidad sin comprometer los datos financieros.

Transacciones (ACID) y Auditoría

Operaciones bancarias críticas como depósitos, transferencias o pagos necesitan completarse íntegramente o no ejecutarse. PostgreSQL garantiza la seguridad transaccional (ACID), fundamental para la auditoría de operaciones financieras y prevenir inconsistencias como "dinero fantasma" o registros incompletos.

PostgreSQL: Roles y Auditoría en el Sistema Bancario

Usuarios y Roles Bancarios

En un sistema bancario, el control de acceso granular es vital. PostgreSQL ofrece un modelo de seguridad robusto que define con precisión qué empleados bancarios pueden acceder, modificar o administrar la información de cuentas y transacciones, garantizando la privacidad del cliente y la seguridad operativa.

Auditoría de Operaciones Financieras

El registro detallado de todas las operaciones financieras es crucial para la banca. PostgreSQL facilita la auditoría mediante triggers y logs, permitiendo un seguimiento exacto de cada movimiento, desde depósitos hasta retiros, esencial para el cumplimiento normativo y la detección de fraudes.

En resumen, para el sistema bancario, PostgreSQL es la mejor opción cuando se requieren altas garantías de integridad, seguridad y consistencia en datos financieros críticos como saldos de cuentas y registros de transacciones.

MongoDB (NoSQL – Documentos): Gestión Flexible de Datos del Cliente Bancario

MongoDB destaca en el almacenamiento de información en documentos JSON, lo que lo hace idóneo para gestionar datos dinámicos y semi-estructurados, como preferencias de usuarios bancarios, notificaciones de transacciones, historial de inicios de sesión y configuraciones de alertas. Su capacidad para adaptarse a esquemas cambiantes es fundamental para la agilidad en el desarrollo de un sistema bancario, permitiendo una evolución rápida de las funcionalidades centradas en el cliente.

MongoDB: Adaptabilidad para la Experiencia del Usuario Bancario

Preferencias del Usuario Bancario

Los usuarios bancarios tienen preferencias personalizadas (tipo de alertas, idioma preferido, configuración de visualización). MongoDB almacena estas configuraciones flexibles sin un esquema estricto, permitiendo una experiencia de usuario altamente adaptada y escalable.

Notificaciones de Transacciones

Las notificaciones de transacciones (compras, depósitos, retiros) son eventos rápidos y de corta duración. MongoDB, con su alta velocidad de escritura y consulta, es ideal para gestionar el gran volumen de estos datos transitorios, que no requieren una consistencia ACID estricta.

MongoDB: Gestión Flexible para Perfiles y Registros Bancarios

Configuración de Perfil de Usuario Bancario

Los perfiles de usuario en banca, incluyendo preferencias y configuración de alertas, a menudo requieren campos opcionales y estructuras anidadas. La naturaleza basada en documentos de MongoDB maneja estos datos de forma natural, facilitando la personalización sin un esquema rígido.

Historial de Login y Notificaciones de Transacciones

El historial de login de los usuarios y las notificaciones de transacciones se generan con alta frecuencia y pueden variar en estructura. MongoDB almacena millones de estos registros con excelente rendimiento, gestionando eficientemente el volumen sin afectar los procesos bancarios críticos.

En conclusión, MongoDB es ideal para gestionar información bancaria flexible, masiva y variable, como preferencias de usuario y logs, donde la estructura no siempre es fija y la velocidad de escritura es una prioridad.

Redis: Caché Ultra-Rápido para Operaciones Bancarias

Redis es un motor de base de datos en memoria que destaca por su increíble velocidad y eficiencia. Su arquitectura clave-valor y su capacidad para operar como caché lo convierten en una pieza fundamental para optimizar el rendimiento de sistemas bancarios, gestionando eficazmente el caché de saldos frecuentes, tokens de sesión de usuarios y contadores de transacciones en tiempo real.

Redis: Potenciando la Eficiencia en Sistemas Bancarios

Caché de Alto Rendimiento para Banca

Redis se utiliza para almacenar en caché **saldos de cuentas frecuentes** y **datos transaccionales de alta demanda**. Esto reduce significativamente la carga sobre **PostgreSQL** (donde residen las cuentas y transacciones principales) y **MongoDB** (con preferencias y notificaciones del usuario), garantizando una respuesta inmediata en operaciones bancarias críticas.

Estructuras de Datos Bancarias Versátiles

Desde cadenas para **tokens de sesión de usuario** hasta hashes para **perfiles de riesgo temporal** o listas para **colas de notificación de transacciones**, Redis ofrece estructuras de datos que permiten implementar soluciones rápidas para **contadores de transacciones en tiempo real** o la gestión de **alertas de fraude**.

Sencillez y Rapidez en Operaciones Financieras

Su modelo de datos simple y su operación **en memoria** garantizan una latencia extremadamente baja, lo que es crucial para **experiencias de usuario fluidas** en la banca móvil, la **validación de transacciones instantáneas** y otras funcionalidades que exigen respuestas inmediatas en el sector financiero.

Conexión entre las Bases de Datos para un Sistema Bancario Integrado

La sinergia entre PostgreSQL, MongoDB y Redis crea una arquitectura de datos robusta y eficiente, optimizada para diferentes necesidades de datos y rendimiento en un entorno bancario. Cada base de datos juega un rol fundamental, trabajando en conjunto para soportar la agilidad y escalabilidad del sistema bancario.

Sincronización de Datos Bancarios

Aunque operan de forma independiente para sus datos primarios, puede haber escenarios donde los datos de PostgreSQL (cuentas bancarias, saldos y movimientos), al ser la fuente de verdad, necesiten replicarse o sincronizarse parcialmente con MongoDB, por ejemplo, para enriquecer preferencias de usuario bancario con información transaccional clave para el envío de notificaciones personalizadas o la configuración de alertas.

Estrategia de Caché para Banca en Redis

Redis almacena datos de acceso frecuente, como saldos frecuentes de cuentas, tokens de sesión de usuarios o contadores de transacciones en tiempo real. Esto reduce la carga en PostgreSQL y MongoDB, y acelera drásticamente los tiempos de respuesta para los usuarios al consultar sus saldos o realizar operaciones rápidas.

Flujo de Lectura y Escritura Bancario

Las solicitudes de lectura de datos bancarios suelen ir primero a Redis (para saldos rápidos o tokens). Si el dato no está cacheado, se busca en la base de datos correspondiente (PostgreSQL para cuentas, transacciones y auditoría; MongoDB para preferencias de usuario y notificaciones). Las escrituras de datos críticos como transacciones y movimientos van directamente a PostgreSQL, mientras que las de datos flexibles como la actualización de preferencias de usuario se dirigen a MongoDB.

Consistencia Eventual en la Experiencia Bancaria

Adoptamos un modelo de consistencia eventual, especialmente para datos no críticos en MongoDB (historial de login, configuración de alertas) y Redis (caché de saldos). Esto significa que la propagación de algunos datos puede tardar un breve tiempo en todo el sistema, una compensación aceptable para ofrecer alta disponibilidad y rendimiento en la consulta de información no transaccional inmediata o la visualización de saldos frecuentes.