TECNOLÓGICO
NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO
de Pabellón de Arteaga
TEC

Instituto Tecnológico de Pabellón de Arteaga, Ags.

UNIDAD 5

Lecturas

Tristán Nathaniel Huerta Valdivia

ITIC's 4to Semestre

Ingeniería de Software

Mtro. Eduardo Flores Gallegos

INDICE

Lectura 1 _____ 3

Things I Wished More Developers Knew About Databases (Cosas que deseaba que más desarrolladores supieran sobre las bases de datos) _____ *3*

Lectura 2 _____ 5

If You Want to Be a Senior Developer, Stop Focusing on Syntax (Si desea ser un desarrollador sénior, deje de centrarse en la sintaxis) _____ *5*

Lectura 3 _____ 6

Software Engineering Best Practices (Mejores prácticas de ingeniería de software) ___ *6*

Lectura 1

Things I Wished More Developers Knew About Databases (Cosas que deseaba que más desarrolladores supieran sobre las bases de datos)

La mayoría de los sistemas informáticos se encuentran en un determinado estado y pueden depender del sistema de almacenamiento. Mi conocimiento de la base de datos se ha acumulado con el tiempo, pero en el proceso, nuestros errores de diseño llevaron a la pérdida e interrupción de datos. En un sistema con una gran cantidad de datos, la base de datos es el núcleo de los objetivos y compromisos de diseño del sistema. Aunque no se puede ignorar la forma en que funciona la base de datos, los problemas que los desarrolladores de aplicaciones esperan y encuentran generalmente son solo la punta del iceberg. En esta serie de artículos, compartiré algunas de las ideas descubiertas, que son particularmente útiles para los desarrolladores que no son buenos en este campo.

ACID tiene muchos significados.

ACID significa atomicidad, consistencia, aislamiento y durabilidad. Estos son los atributos que las transacciones de la base de datos deben garantizar la validez de sus usuarios, incluso en el caso de fallas, errores, fallas de hardware, etc. Sin ACID o contratos similares, los desarrolladores de aplicaciones no podrán determinar cuál es su responsabilidad por el contenido proporcionado por la base de datos. La mayoría de las bases de datos transaccionales relacionales intentan cumplir con ACID, pero los nuevos métodos, como el movimiento NoSQL, han llevado a la implementación de muchas bases de datos no transaccionales ACID debido al alto costo de su implementación.

Cuando no puede mantener el bloqueo, puede utilizar el bloqueo optimista.

Los bloqueos no solo son costosos, no solo porque introducen más contención en la base de datos, sino que también pueden requerir una conexión consistente desde el servidor de aplicaciones a la base de datos. Las particiones de red afectarán el bloqueo exclusivo más seriamente y conducirán a puntos muertos que son difíciles de identificar y resolver. Si el bloqueo exclusivo no es fácil de mantener, puede elegir el bloqueo optimista.

El aumento personal puede ser dañino.

AUTOINCREMENT'ing es un método común para generar claves primarias. No es raro ver que la base de datos se usa como un generador de ID y que hay una tabla de generación de ID especificada en la base de datos.

Los datos desactualizados pueden ser útiles y no bloqueantes.

El control de concurrencia de versiones múltiples (MVCC) permite muchas de las características de coherencia que discutimos brevemente anteriormente. Algunas bases de datos (por ejemplo, Postgres, Spanner) usan MVCC para permitir que cada transacción vea una instantánea, que es una versión anterior de la base de datos. Las transacciones de instantáneas aún se pueden serializar para mantener la coherencia. Al leer una instantánea anterior, leerá datos obsoletos.

El crecimiento significativo de la base de datos introduce imprevisibilidad.

El crecimiento de la base de datos te hace encontrar problemas de escala impredecibles. Cuanto más sepamos acerca de las partes internas de la base de datos, menos podremos predecir cómo se expandirán, pero hay ciertas cosas que no podemos predecir.

A medida que crece, las suposiciones o expectativas previas sobre el tamaño de los datos y los requisitos de capacidad de la red pueden quedar obsoletas. Esto es cuando se reescriben arquitecturas a gran escala, mejoras operativas a gran escala, problemas de capacidad, reconsideración de la implementación o migración a otras bases de datos para evitar interrupciones.

Lectura 2

If You Want to Be a Senior Developer, Stop Focusing on Syntax (Si desea ser un desarrollador sénior, deje de centrarse en la sintaxis)

Los nuevos desarrolladores de software pueden sentirse mal porque no están memorizando una sintaxis suficiente; todavía deben consultar la documentación. Ven esto como una señal de que son malos desarrolladores.

¿Es la sintaxis tan importante que tengo que memorizarla?

Hace unos meses, leí una publicación de un desarrollador senior en el equipo de Facebook. Mencionó que cuando el reclutador le preguntó si tenía experiencia técnica específica que no tenía, respondió: "Eso es solo otro Herramientas." Esto significa que es posible que no tenga la oportunidad de trabajar con él en el pasado, pero creo que puedo aprender.

Esto significa que es posible que no tenga la oportunidad de trabajar con él en el pasado, pero creo que puedo aprender. Es posible que no necesite unos meses para operar. Es posible que solo necesite unas pocas horas para leer la documentación y luego aprender más según sea necesario durante todo el proceso sin tener que recordar todo. Cuando comienzas a ir a tu destino, no quieres que se enciendan todas las luces verdes. Pasa a través de la luz verde actual y luego se detiene en la luz roja. Espere hasta que se encienda la luz verde y luego continúe.

Lectura 3

Software Engineering Best Practices (Mejores prácticas de ingeniería de software)

Escribir código limpio y modular

¿Qué es el código limpio y modular? Es un código de nivel de producción que cumple con los siguientes criterios.

Código de producción: software que se ejecuta en un servidor de producción para procesar usuarios y datos en tiempo real para el público objetivo. Esto es diferente de los códigos de calidad de producción, que describen códigos que cumplen con los requisitos de confiabilidad de producción, eficiencia, etc.

Limpio: legible, simple y conciso. Estas funciones son esenciales para la colaboración y el mantenimiento en el desarrollo de software.

Modular: Lógicamente dividido en funciones y módulos. Esto hace que su código sea más organizado, eficiente y reutilizable.

Módulo: un archivo. Estos módulos pueden reutilizar su código encapsulando el código en un archivo que puede importarse a otros archivos.

Escribir código eficiente

Saber cómo escribir código que funcione de manera efectiva es una habilidad básica en el desarrollo de software. Optimizar el código para aumentar la eficiencia puede significar:

- Ejecutar más rápido

- Ocupa menos memoria / espacio de almacenamiento

El proyecto en el que está trabajando actualmente determinará qué optimización es más importante. Si desea realizar muchas transformaciones diferentes en una gran cantidad de datos, puede hacer una gran diferencia en el rendimiento.

Agregar documentos significativos

Agregar información adicional de texto o ilustración que acompaña al código ayuda a aclarar las partes complejas del código, hace que el código sea más fácil de navegar y explica rápidamente cómo y por qué se utilizan los diferentes componentes del programa.

Se pueden agregar varios tipos de documentos en diferentes niveles del programa:

- Revisiones en línea: nivel de fila
- Docstrings-módulos y niveles funcionales
- Documento de proyecto-Nivel de proyecto

Comentarios en línea

Los comentarios en línea son texto que sigue los símbolos de barra diagonal en "//" todo el código. Se utilizan para explicar partes de su código y realmente ayudan a los futuros contribuyentes a comprender su trabajo.

Documentación de proyecto

La documentación del proyecto es crítica para otros, ya sean futuros usuarios de su proyecto o desarrolladores que quieran contribuir a su código, otros deben entender por qué y cómo su código es relevante para ellos. Un buen primer paso

en la documentación del proyecto es el archivo Léame. Por lo general, esta es la primera interacción que la mayoría de los usuarios tienen con su proyecto.

Como mínimo, esto debería explicar lo que hace su proyecto, enumerar sus dependencias y proporcionar instrucciones suficientemente detalladas sobre cómo usarlo.