Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 1er. Semestre 2021.

PROYECTO1 INTRODUCCIÓN A LA PROGRAMACIÓN Y COMPUTACIÓN 2

201801005 – Fabio Josué Hernández Martinez

Resumen

En este proyecto se busca minimizar los costos de una transmisión en una base de datos, por ejemplo, digamos que tiene una aplicación que utiliza tablas de base de datos para almacenar datos de registro y de flujo de clics. Puede almacenar sus datos en una base de datos relacional para facilitar las tareas de desarrollo y administración. Cuando inicia su aplicación, la base de datos es manejable al principio, pero crece a cientos de gigabytes por semana. Tan solo el almacenamiento y la recuperación de datos consumen el 20 por ciento de IOPS (es una unidad de benchmark usada para medir el rendimiento de dispositivos informáticos) y CPU en su instancia de base de datos relacional. Además, las aplicaciones almacenan documentos XML, JSON y binarios en las tablas de bases de datos junto con datos transaccionales. Los datos históricos siguen creciendo cada mes. Sus costos tradicionales de infraestructura y licencias de la base de datos local están aumentando, y escalar la base de datos se ha convertido en un gran desafío. ¿Qué puede hacer?

Palabras clave

- Matriz
- TDA
- POO
- Estructura
- Datos

Abstract

This project seeks to minimize the costs of a stream to a database, for example, let’s say you have an application that uses database tables to store log and click flow data. You can store your data in a relational database to facilitate development and management tasks. When you start your application, the database is manageable at first, but grows to hundreds of gigabytes per week. Only data storage and retrieval consume 20 percent IOPS (it's a benchmark drive used to measure computing device performance) and CPU on your relational DB instance. In addition, applications store XML, JSON, and binary documents in database tables along with transactional data. Historical data continues to grow every month. Your traditional infrastructure costs and on-premises database licensing are increasing and scaling the database has become a big challenge. What can you do?

Keywords

- Matrix
- TDA
- POO
- Structure
- Data

Introducción

Se desarrollará la solución a una base de datos en la cual se espera minimizar los costos de transmisión de datos, sobre una base de datos, pero ¿Qué es una base de datos? Para simplificar la definición podemos decir que una base de dato es un tipo de "almacén" que nos permite guardar grandes cantidades de información de forma organizada para que luego podamos encontrar y utilizar fácilmente.

Con esta breve definición no podemos prever la magnitud e importancia de una base de datos hoy en día, y el porque de la importancia de su eficiencia esperando maximizar productividad y disminuir los problemas y costos de esta.

Desarrollo del tema

Este problema consiste en alojar objetos de bases de datos en sitios distribuidos, de manera que el costo total de la transmisión de datos para el procesamiento de todas las aplicaciones sea minimizado. Un objeto de base de datos es una entidad de una base de datos, esta entidad puede ser un atributo, un set de tuplas, una relación o un archivo. Los objetos de base de datos son unidades independientes que deben ser alojadas en los sitios de una red. Una definición formal del problema se presenta en la figura No. 1.

La figura No. 1 muestra un set de objetos de bases de datos O = {o1, o2, …, ono}, una red de computadoras que consiste en un set de sitios S = {s1, s2, …, sns}, donde un set de consultas Q = {q1, q2, … qnq} son ejecutadas, los objetos de base de datos requeridos por cada consulta, un esquema inicial de alojamiento de objetos de bases de datos, y las frecuencias de acceso de cada consulta desde cada sitio en un período de tiempo. El problema consiste en obtener un nuevo esquema replicado de alojamiento que se adapte a un nuevo patrón de uso de la base de datos y minimice los costos de transmisión.

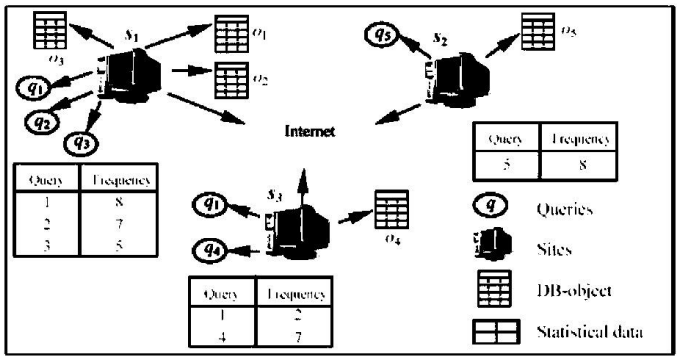

Figura No.1 – Problema de diseño de distribución en una base de datos

El problema de diseño de distribución consiste en determinar el alojamiento de datos de forma que los costos de acceso y comunicación son minimizados. Como muchos otros problemas reales, es un problema combinatorio NP-Hard. Algunas de las situaciones comunes que hemos observado cuando se resuelven instancias muy grandes de un problema NP-Hard son: Fuerte requerimiento de tiempo y fuerte demanda de recursos de memoria. Un método propuesto para resolver este tipo de problemas consiste en aplicar una metodología de agrupamiento.

Para "nt" tuplas y "ns" sitios, el método consiste en tener la matriz de frecuencia de acceso en los sitios F[nt][ns] de la instancia objetivo, transformarla en una matriz de patrones de acceso y agrupar las tuplas con el mismo patrón.

¿Qué es un problema NP-Hard?

Un problema np-hard se refiere cuando para cada problema L en NP, hay una reducción polinomio-tiempo de varios-uno de L a H. Una definición equivalente es exigir que cada problema L en NP pueda ser resuelto en tiempo polinomio por una máquina oracle con un oráculo para H. Informalmente, se puede pensar en un algoritmo que llama a una máquina de oráculos como una subrutina para resolver H y resuelve L en tiempo polinomio si la llamada a la subrutina toma sólo un paso para calcular.

Otra definición es exigir que haya una reducción del tiempo polinomio de un problema np-complete G a H. Como cualquier problema L en NP reduce en tiempo polinomio a G, L reduce a su vez a H en tiempo polinomio por lo que esta nueva definición implica la anterior. Torpemente, no restringe la clase NP difícil de tomar decisiones problemas, y también incluye problemas de búsqueda o problemas de optimización.

Un ejemplo de un problema np-hard es el problema de suma del subconjunto de decisiones: dado un conjunto de enteros, ¿algún subconjunto no vacío de ellos suma cero? Ese es un problema de decisión y resulta ser np-complete. Otro ejemplo de un problema np-hard es el problema de optimización de encontrar la ruta cíclica de menor costo a través de todos los nodos de un gráfico ponderado. Esto se conoce comúnmente como el problema del vendedor ambulante.

Hay problemas de decisión que son difíciles de np pero no np-complete como el problema de la detención. Ese es el problema que se pregunta "dado un programa y su aporte, ¿funcionará para siempre?" Eso es un sí/no hay duda y también lo es un problema de decisión. Es fácil demostrar que el problema de la detención es difícil de np pero no np-complete. Por ejemplo, el problema de satis-fiabilidad booleana se puede reducir al problema de detención transformándolo en la descripción de una máquina Turing que intenta todas las asignaciones de valor de verdad y cuando encuentra una que satisface la fórmula se detiene y de lo contrario entra en un bucle infinito. También es fácil ver que el problema de la paralización no está en np ya que todos los problemas en NP son indecisos en un número finito de operaciones, pero el problema de la detención, en general, es indeciso. También hay problemas difíciles de NP que no son ni np-complete ni indeciso. Por ejemplo, el lenguaje de las fórmulas booleanas cuantificadas verdaderas es decidirle en el espacio polinomio, pero no en tiempo polinomio no determinista (a menos que NP = PSPACE).

Existen diferentes estrategias para reducir los costos de bases de datos, al mismo tiempo que mejoramos la disponibilidad y simplificamos la administración de la base de datos. Antes que nada, debemos conocer los diferentes tipos de almacenes de datos que hay disponibles en AWS.

Sabemos que los RDBMS (Gestión de bases de datos relacionales) son utilizados por la mayoría de las aplicaciones que generan procesamiento de transacciones. Entre los motores de bases de datos más populares de este estilo, tenemos MySql, SQL Server y Oracle. Este tipo de motores de base datos puede:

- Generar problemas de desempeño. A menos que tengamos los recursos para ir aumentando la capacidad de procesamiento y disco de nuestros servidores de forma constante, las tablas más grandes siempre necesitaran más tiempo para ser consultadas y la combinación de transacciones de tipo lectura/escritura de este tipo de tablas, puede generar problemas de lentitud.
- Aumento del costo total de propiedad (CTO). Para mejorar algunas consultas lentas y mejorar el tráfico lectura/escritura en nuestra base de datos, debemos escalar verticalmente agregando más capacidad de cómputo y almacenamiento. Si necesitáramos aumentar la capacidad solamente por una temporada del año, tendríamos que hacer la inversión necesaria pero no podremos reducir la escala, lo cual aumenta el CTO.

Conclusiones

1. La importancia de la optimización de una base de datos reduce costos.

2. El problema NP-Hard se puede reducir a como crece el coste computacional de resolver un determinado problema con relación a lo que crece el tamaño de dicho problema.