Análisis, Diseño e Implementación de un Sistema de Administración y Planificación de Índices Basado en la Demanda de Operaciones de Consulta y Actualización de un Sistema de Base de Datos

Roberth Darío Chávez Jara, Juan Carlos Viscarra Jaramillo, Fabricio Echeverría Briones, Ing.
Facultad de Ingeniería de Electricidad y Computación
Escuela Superior Politécnica del Litoral
Campus "Gustavo Galindo V.", Km 30.5, Vía Perimetral, Guayaquil, Ecuador
dchavez@espol.edu.ec, jviscarr@espol.edu.ec, pechever@espol.edu.ec

Resumen

La tecnología ha revolucionado el mundo actual y su ritmo de crecimiento parece no tener límite. Una de las herramientas que cambió drásticamente la forma de hacer negocios son las bases de datos, con las cuales llegaron nuevos conceptos tales como CRM, ERP, DSS, GIS, entre otros. En la actualidad estos conceptos dejaron de ser eso y se han convertido en la herramienta con la cual muchas empresas hacen negocios y se mantienen en la competencia. Sin embargo, las bases de datos no pueden hacer todo el truco por si mismas. Es importante un buen diseño inicial y un monitoreo constante para poder obtener el mejor desempeño del motor de bases de datos reflejado en tiempos de respuesta muy cortos; la falta de monitoreo sobre estos objetos puede resultar en la degeneración del desempeño de las aplicaciones vitales para el negocio.

Nuestro sistema, MAEBD (Monitoreo y Análisis Estadístico de Base de Datos), se encarga de monitorear la demanda de uso de tablas basado en estadísticas y utilizando métodos heurísticos podemos predecir los índices que deberían crearse para mejorar el desempeño de la base de datos.

Palabras Claves: *Base de Datos, Monitoreo, Demanda, Índices, Desempeño, Métodos Heurísticos*

Abstract

Technology has revolutionized the present world y its rate of growth seems to have no limit. One of the tools that changed drastically the way to do business are databases, which brought along concepts such as CRM, ERP, DSS, GIS, among others. At the present time these concepts have stopped being so and have become one of the tools which most of enterprises are doing business and keeping up the pace. However, databases don't do the whole trick on their own. It is important a good initial design and constant monitoring so as to obtain the best performance from the RDBMS reflected as short response times; the lack of monitoring on these objects might result in degenerating critical business applications performance.

Our system, MAEBD (Monitoreo y Análisis Estadístico de Base de Datos), is meant to monitor the demand of tables usage based on statistics and using heuristic methods, we might be able to foresee what indexes ought to be created so as to enhance RDBMS performance.

1. Introducción

Las bases de datos son herramientas que nos permiten abstraer el modelo de negocios de una empresa y almacenar datos relevantes al mismo para posteriormente acceder a estos de forma rápida y estructurada, lo cual es aprovechado para desarrollar soluciones aplicables a las diferentes áreas del negocio, agilizando las operaciones del mismo.

Los administradores de bases de datos o también llamados DBA (por sus siglas en inglés, Database Administrator) tienen como una de sus labores diseñar un modelo acorde con el motor disponible de tal manera que puedan aprovechar al máximo la capacidad del mismo y monitorear el desempeño del motor para determinar si se necesita afinar ciertos parámetros que permitan obtener el máximo rendimiento. Sin embargo, cuando las empresas son grandes y complejas, los modelos de bases de datos por lo general son aún más grandes y complejos de tal manera que es virtualmente imposible para el DBA estar constantemente monitoreando todos los parámetros de desempeño que indiquen que todo marche como se ha previsto. En muchas ocasiones, las sentencias de consultas más simples implementadas en algún aplicativo de la empresa pueden convertirse en cuellos de botella que degeneran el motor e incrementan los tiempos de respuesta de las aplicaciones.

El propósito del presente artículo es exponer la solución implementada para resolver un problema de optimización de bases de datos utilizando herramientas propias de las bases de datos como son los índices, adicionando estadísticas y métodos heurísticos. Analizando el tipo de solución requerido y los recursos comprometidos para la misma, optamos por la implementación de una aplicación web utilizando JavaServer Faces (JSF), un estándar de desarrollo J2EE basado en el patrón MVC (Modelo Vista Controlador) [1].

La arquitectura de la solución implementada será tratada en la sección 2. En la sección 3 se describe la justificación del diseño de la aplicación. En la sección 4 el alcance de la aplicación. En la sección 5 la implementación de la aplicación. Finalmente las conclusiones y recomendaciones se incluyen en las secciones 6 y 7, respectivamente; y las referencias se describen en la sección 8.

2. Arquitectura de la solución

La solución implementada está formada por 4 componentes:

- Aplicación web
- Motor de replicación
- Agente de consultas
- Base de datos monitoreada

A continuación describiremos cada uno de los componentes y a su vez detallaremos los componentes que las integran:

Aplicación web: Es una aplicación web de 3 capas, en la cual se ha programado la lógica de análisis de demanda de uso de tablas, creación de índices y tareas de monitoreo que forman parte de la solución requerida. Fue implementada utilizando JSF, un servidor de aplicaciones y una base de datos.

Motor de replicación: Una aplicación Java tradicional que replica datos de estadísticas de uso de tablas desde la base monitoreada hacia el repositorio de datos de la solución. Utiliza una herramienta de programación de tareas para ejecutarse periódicamente.

Agente de consultas: Una aplicación Java tradicional que simula las consultas realizadas por las aplicaciones de una organización cualquiera contra sus bases de datos monitoreadas por la solución.

Base de datos monitoreada: Es una base de datos de producción de una empresa cualquiera monitoreada por la solución. Sobre ella se realiza las consultas de estadísticas y administración de índices.

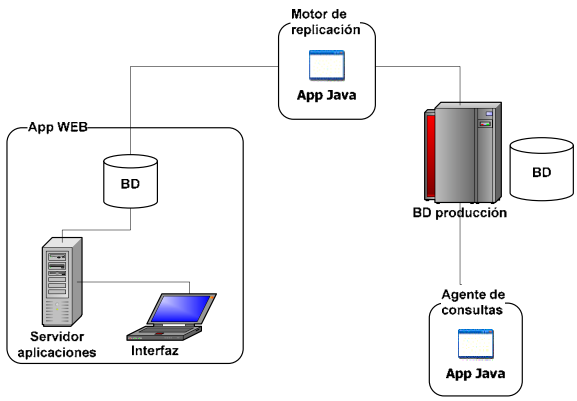

Figura 1. Arquitectura de la solución

2.1 Elementos de análisis de demanda

La aplicación web se encarga de interactuar con el usuario, la base de datos de la solución y la base de datos de producción. Para conseguir esto el usuario debe activar el monitoreo de la base de datos y las tablas de interés. Utilizando las estadísticas de demanda de las tablas monitoreadas y un algoritmo genético, llamado así porque se inspiran en la evolución genética y su base genético-molecular [2], se logra sugerir al usuario los posibles índices que se pueden crear para optimizar el acceso de las aplicaciones al repositorio de datos de producción. Las soluciones, conocidas como cromosomas, y representadas por los índices son seleccionadas en base a su aptitud o fitness (*F*). Los cromosomas están formados por genes, representados por las columnas de las tablas, y éstos también tienen una aptitud que puede ser cuantificada. La aptitud de los cromosomas es la sumatoria de las aptitudes de sus genes:

$$F = \sum_{i=1}^{k} F_i$$

Figura 2. Aptitud de un cromosoma

donde k representa los genes que forman el cromosoma. La aptitud de los genes es cuantificada a partir de su patrón genético, el cual está formado por dígitos binarios cuyo valor, 0 ó 1, es asignado en base al cumplimiento de un set de reglas definidas y diseñadas a partir de recomendaciones generales en el diseño de bases de datos, tanto a nivel teórico como práctico. El patrón genético tiene la siguiente composición:

DÍGITO A	DÍGITO B	DÍGITO 1	DÍGITO 2	DÍGITO 3	DÍGITO 4	DÍGITO 5	DÍGITO 6
Regla A	Regla B	Regla 1	Regla 2	Regla 3	Regla 4	Regla 5	Regla 6
1 - 0	1 - 0	1 - 0	1 - 0	1 - 0	1 - 0	1 - 0	1 - 0

Figura 3. Patrón genético de un gen

Las reglas definidas son de 2 tipos: estándar y de conocimiento. Las estándar obedecen, como mencionamos anteriormente, a recomendaciones generales en el diseño de bases de datos y son:

- Regla 1 – Factor de nulabilidad: mide la cantidad de valores nulos contenidos en la columna de una tabla. Se califica uno (1) si es igual a 0, caso contrario cero (0).
- Regla 2 – Clave foránea: Se califica con uno (1) si la columna es clave foránea (FK), caso contrario con cero (0).
- Regla 3 – Demanda de clave foránea: Se califica con uno (1) si la demanda es mayor a 1500 lecturas (demanda lógica más física)
- Regla 4 – Clave primaria: Columnas que forman parte de la clave primaria se califican con cero (0); caso contrario con uno (1).
- Regla 5 – Factor de selectividad: mide la cantidad de valores distintos contenidos en la columna. Se califica con uno (1) si el factor es mayor a 80%, caso contrario con cero (0).
- Regla 6 – Tipo de dato: Se califican con uno (1) las columnas cuyo tipo de dato está dentro de la siguiente clasificación:

Tabla 1. Tipo de datos válidos

Tipo de Dato	Long. MIN	Long. MAX
VARCHAR2	4	30
NUMBER	4	10
DATE	--	--

Las reglas de conocimiento están ligadas a la naturaleza del algoritmo utilizado y son muy específicas dependiendo del problema que se quiera enfocar. Éstas permiten cuantificar el conocimiento adquirido por aquellos cromosomas que ya constituyeron soluciones. Además hemos considerado que estas reglas tienen gran relevancia para el cálculo de la aptitud, de manera que constituyen los dígitos más significativos del patrón genético (dígito A y B). A continuación detallamos las reglas:

- Regla A – Regularidad: Es el número de veces que el índice (por ende las columnas) ha sido leído lógica o físicamente en un período de 30 días. Esta regla se expresa en forma de porcentaje y basta con que el índice haya sido una sola vez en el día para calificar. Esto es que si el índice se uso al menos 1 vez por 20 días, la regularidad será (20/30)*100 = 66,7%

  Regularidad = (# veces al menos 1 vez x día) / 30

  Las columnas con una regularidad mayor o igual al 60% se califican con 1.

- Regla B – Nivel de regularidad: El nivel de regularidad es el número de días que el índice (por ende las columnas) ha tenido una demanda superior al promedio calculado en base a un período de 30 días.

  Nivel de regularidad = (# días demanda excede promedio mensual) / 30

  Las columnas con un nivel de regularidad mayor o igual al 70% se califican con 1.

3. Justificaciones de Diseño

3.1 Aplicación Web vs. Aplicación de escritorio

Actualmente existen paquetes de desarrollo de software (SDK por sus siglas en inglés) para el desarrollo de aplicaciones de escritorio que permiten a los desarrolladores implementar una amplia gama de herramientas e interfaces que mejoran la funcionalidad y usabilidad de las aplicaciones. Entre las razones que motivan la implementación de estas soluciones tenemos [3]:

- Uso de recursos: las aplicaciones de escritorio pueden diseñarse de manera que aprovechen al máximo los recursos de los equipos host para procesar grandes cantidades de datos.
- Integración con el sistema host: este tipo de aplicaciones permiten integrarse con el sistema operativo del host permitiendo la ejecución de rutinas nativas como arrastrar archivos.
- Necesidad de interfaces consistentes: las interfaces de las aplicaciones de escritorio son diseñadas a la medida y no incluyen elementos estándar que pueden afectar como se ve la aplicación y como los usuarios interactúan con la misma.

Por otro lado, las aplicaciones web siguen ganando terreno y los gigantes del software como Microsoft se encuentran investigando y desarrollando nuevas APIs y frameworks que han permitido que las aplicaciones web evolucionen hasta convertirse en las recientes RIA (por su nombre en inglés, Rich Internet Application) con características y funcionalidades similares a las aplicaciones de escritorio [4]. Algunas de las razones para implementar este tipo de aplicaciones son [5]:

- Fácil de instalar y usar: no se necesitan instalar un cliente, pues los navegadores web nativos de los sistemas operativos pueden ser usados como clientes: escribe la dirección de la aplicación y a usar.
- Fácil de actualizar: dado que su naturaleza es cliente/servidor, una sola actualización y la aplicación se actualiza para todos los usuarios.
- Niveles de acceso avanzado no son necesarios: al solo necesitar el navegador del computador no se requieren accesos privilegiados para instalar o usar la aplicación.
- Acceso desde cualquier lugar: las comunicaciones de hoy en día permiten a los usuarios mantenerse conectados a sus oficinas inclusive desde sus teléfonos móviles; no solo el correo puede ser revisado remotamente, las aplicaciones web también puede ser publicadas en redes externas como Internet y accedidas por los usuarios como si accedieran localmente a través de una intranet.

Dado que los administradores de bases de datos deben estar habilitados para monitorear el estado de los sistemas de los cuales son responsables en cualquier momento y sumado a las ventajas ya mencionadas, escogimos una aplicación web lo cual nos permite enfocarnos en los aspectos propios del diseño de la aplicación sin invertir tiempo ni recursos innecesarios en detalles de integración con el sistema operativo nativo.

4. Alcance

La solución implementada constituye un prototipo cuya funcionalidad tiene los alcances a continuación detallados.

4.1 Monitorear una base de datos a la vez

Los usuarios pueden agregar varias bases de datos y activar el monitoreo de una base de datos a la vez.

4.2 Monitorear varias tablas de una misma base de datos a la vez

Los usuarios pueden activar el monitoreo de tablas para varias tablas o índices que una misma base de datos a la vez.

4.3 Ver demanda de tablas e histograma de demanda

Los usuarios pueden consultar la demanda acumulada de lecturas físicas y lógicas de las tablas de una base de datos. Así mismo puede consultar la demanda de una tabla específica por fecha.

4.4 Generar análisis de demanda

Esta opción permite a los usuarios ejecutar el proceso principal de la solución y calcula la aptitud de los genes (columnas) y cromosomas (posibles índices) y los presenta como soluciones permitiendo su creación.

4.5 Crear y eliminar índices

Los usuarios pueden crear índices en las bases de datos de producción desde la solución.

4.6 Visualizar estadísticas gráficas de base de datos, tablas e índices

Los usuarios pueden consultar estadísticas de base de datos, tablas e índices por fecha.

4.7 Impresión de estadísticas y listados

Los usuarios tienen la opción de imprimir muchas de las pantallas que visualizan e inclusive estadísticas.

4.8 Administración de usuarios y sistemas

Se pueden administrar (crear, editar y eliminar) los usuarios que pueden acceder a la aplicación y los sistemas a monitorear.

4.9 Historial de log de aplicación

Los usuarios pueden consultar el log del motor de replicación para verificar el estado del motor y si existen datos replicándose.

5. Implementación de la solución

Como previamente se mencionó, utilizamos JSF para implementar la solución. Solo usuarios autenticados pueden acceder al menú principal:

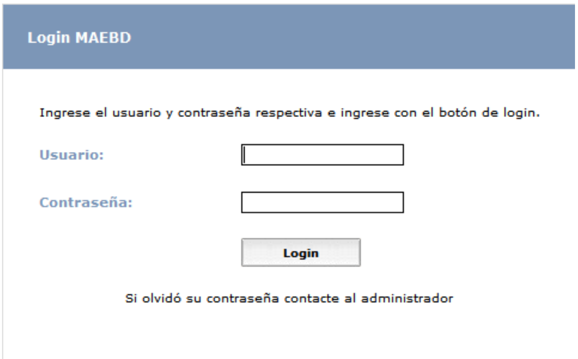

Figura 4. Pantalla de login de la aplicación

Figura 5. Menú de la aplicación

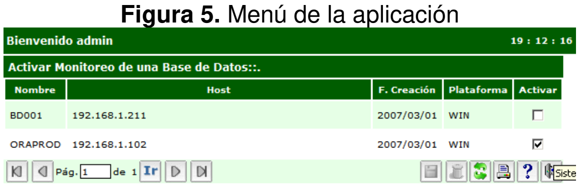

Figura 6. Activar base de datos

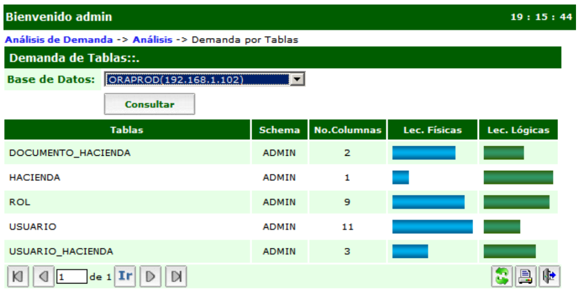

Figura 7. Consultar demanda de tablas de una base

El framework JSF permite un desarrollo rápido y eficiente, el cual se combinó con estándares de programación que permiten modularizar la aplicación en paquetes de clases de acuerdo a su naturaleza. Algunos de los conceptos aplicados para la implementación de la solución son DAO (Data Access Object), DTO (Data Transfer Object) y Backing Beans, los cuales son patrones de diseño J2EE y son parte de las buenas prácticas de diseño y programación.

6. Conclusiones

Las herramientas nativas de monitoreo de base de datos pueden consumir grandes cantidades de recursos que podrían afectar el tiempo de respuesta de las aplicaciones. Los índices pueden mejorar mucho el desempeño del motor si son bien diseñados; sin embargo parte del diseño es el monitoreo lo cual es complejo para empresas grandes cuyos sistemas utilizan grandes bases de datos en cuanto a volumen y estructura. La solución implementada refleja la utilidad de los métodos heurísticos aplicados a la solución de problemas de optimización y demuestra que dependiendo del problema a enfocar, se debe escoger el algoritmo a aplicar.

Esta aplicación facilita el monitoreo de la demanda de tablas de una base de datos y automatiza la recolección de estadísticas que permiten al administrador de base de datos crear índices para mejorar el desempeño del motor y disminuir los tiempos de respuesta de las aplicaciones.

7. Recomendaciones

Como recomendación general sería conveniente seguir adelante con el desarrollo de esta solución y habilitar el monitoreo de más bases de datos al mismo tiempo y agregar funcionalidades en base a las necesidades de los administradores de bases de datos aprovechando la ventaja de las herramientas utilizadas para su desarrollo.

8. Referencias

[1] Schalk, Cris. "*Introduction to Javaserver Faces - What is JSF?*", Abril 2005. Oracle Technology Network, <http://www.oracle.com/technology/tech/java/newsletter/articles/introjsf/index.html>

[2] Wikipedia La Enciclopedia Libre, "*Algoritmo Genético*", Artículo, <http://es.wikipedia.org/wiki/Algoritmo_gen%C3%A9tico>

[3] Steward, Ryan. "*Desktop vs. Browser - when to deploy applications for each*", Abril 2007. ZDNet, < http://blogs.zdnet.com/Stewart/?p=329 >

[4] Wikipedia La Enciclopedia Libre, "*Rich Internet Application*", Artículo, < http://en.wikipedia.org/wiki/Rich_Internet_application>

[5] Wikipedia La Enciclopedia Libre, "*Web Application*", Artículo, <http://en.wikipedia.org/wiki/Web_application>