Buenas Prácticas en Bases de Datos

María del Pilar Angeles.
Posgrado de la Facultad de Ingeniería, UNAM.
mpilar_angeles@exalumno.unam.mx

Algunos Tópicos de Base de Datos

- Modelado y Diseño
- Programación
- Seguridad
- Respaldo y Recuperación
- Configuración, Rendimiento y Afinación
- Calidad
- OLTP,DWH, BI,DG,DP

Modelado y Diseño

Titulo: Estandarización de términos

Área de aplicación: Modelado y Diseño

Problema que resuelve: Confusión y pérdida de semántica de los objetos a crear.

Contexto: Acceso y mantenimiento a los objetos de base de datos.

Descripción: Establecer convenciones en el nombrado de la base de datos y sus objetos, estandarizarlo y apegarse a el.

Beneficio: Mantenimiento e interoperabilidad de sistemas mas rápido y efectivo.

Titulo: Normalización y Rendimiento

Área de aplicación: Modelado y Diseño

Problema que resuelve: Incongruencia entre el esquema de la base de datos y el modelo de datos.

Contexto: Acceso a la información

Descripción: Asegurarnos de normalizar los datos cuando menos hasta la tercera forma normal, sin comprometer el rendimiento

Beneficio: Mejor tiempo de respuesta y congruencia del modelo físico con las bondades del manejador.

Modelado y Diseño

Titulo: Análisis de requerimientos y utilización de índices

Área de aplicación: Modelado y Diseño

Problema que resuelve: Acceso secuencial en lugar de acceso por índices.

Contexto: Análisis de requerimientos, acceso a la información

Descripción: Es muy importante que mediante el análisis de requerimientos se obtengan las consultas mas populares, de tal forma que la información pueda indexarse y aprovechar el acceso prácticamente directo que nos permiten los arboles al generar los índices correspondientes y actualizar el plan de acceso. Por tanto, se recomienda que al diseñar las entidades se identifiquen sus consultas, llaves primarias, foráneas e índices.

Beneficio: Mejora en el tiempo de respuesta al obtener la información.

Modelado y Diseño

Titulo: Uso de Diccionario de Datos

Área de Aplicación: Modelado y Diseño

Problema que resuelve: Problemas de integridad e inconsistencia.

Contexto: Procesamiento de la Información

Descripción: Usar tipos de datos definidos por usuario si una columna en particular se repite en varias tablas de tal forma que el dominio será consistente en todas esas tablas.

Beneficio: Optimización y estandarización del dominio de datos a utilizarse en los atributos y por ende se minimiza el riesgo de capturar valores incongruentes en éstos.

Titulo: Restricción de dominio según requerimientos

Área de aplicación: Modelado y Diseño

Problema que resuelve: Problemas de integridad e inconsistencia

Contexto: Acceso y procesamiento de información

Descripción: Definir aquellos tipos de datos cuya longitud y precisión (dominio) se ajuste mejor al rango de valores requeridos validos.

Beneficio: Optimización y estandarización del dominio de datos a utilizarse en los atributos.

Modelado y Diseño

Titulo: Manejo de Textos y BLOB

Área de aplicación: Modelado y Diseño

Problema que resuelve: Tiempo de acceso inaceptable y limitada o nula manipulación de datos

Contexto: Acceso y procesamiento de información

Descripción: Un Manejador de Bases de datos ha sido programado para manipular atributos con cierto tipos de datos. Posteriormente, se permitió definir texto, BLOB, XML, por ejemplo. Sin embargo, no es posible un manejo optimo de dichos tipos de datos. Se recomienda por tanto evitar guardar estos tipos de datos en la base de datos y guardar en su lugar la ruta de donde se encuentra a fin de que el software correspondiente lo manipule correctamente.

Beneficio: Reducción de tiempo de respuesta.

Modelado y Diseño

Titulo: Generación y Ejecución de SQL dinámico

Área de aplicación: Programación SQL y Seguridad

Problema que resuelve:

Contexto: Acceso y procesamiento de información

Descripción: Al utilizar SQL dinámico existe la posibilidad de el usuario inyecte código y elimine o corrompa información. Por otro lado, como es SQL dinámico, cada vez que ejecuta requiere la compilación y la generación del plan de ejecución, lo cual afecta el rendimiento. Por lo tanto se recomienda evitar SQL dinámico.

Beneficio: Sistemas mas seguros, predecibles y rápidos.

Titulo: Manejo de valores nulos.

Área de aplicación: Modelado y Diseño

Problema que resuelve: Información incompleta, malinterpretación de la información ausente, tiempo de acceso al convertir un campo inexistente a nulo al desplegarse.

Contexto: Acceso y procesamiento de información

Descripción: Minimizar el uso de valores nulos

Beneficio: Captura mas completa de la información.

Programación

Titulo: Documentación de objetos SQL

Área: Programación SQL

Problema que resuelve: Dada la ilegibilidad de programas, se presenta lentitud y confusión en el mantenimiento de programas

Contexto: Mantenimiento de Sistemas

Descripción: Documentar los procedimientos almacenados, triggers, y demás programas.

Beneficio: Tiempo de mantenimiento reducido.

Titulo: Detalle de atributos en la inserción, consulta o actualización

Área: Programación SQL

Problema que resuelve: Contención en disco

Contexto: Consulta de Información

Descripción: Evitar SELECT * en las consultas. Procurar mencionar los campos que se requieren

Beneficio: reducción de acceso a disco y por ende mejor tiempo de respuesta

Programación

Titulo: Uso de cursores como simulación de programación estructurada

Área: Programación SQL

Problema que resuelve: El optimizador realiza búsqueda secuencial a nivel exponencial, provocando problemas de concurrencia y tiempos de respuesta inaceptables.

Contexto: Consulta de Información (conjunto vs. procedural)

Descripción: Evitar usar cursores en el servidor. El uso de cursores es una simulación a la programación procedural, sino se hace con cuidado y se empiezan a anidar cursores se empiezan a formar exponencialmente arboles, gastando recursos y por tanto tiempo de ejecución.

Beneficio: Se aprovecha el manejo de conjuntos en el manejador y por tanto el optimizador puede proporcionar vías de acceso con menor costo.

Programación

Titulo: Uso de comodines

Área: Programación SQL

Problema que resuelve: El optimizador realiza búsqueda secuencial, provocando problemas de concurrencia y tiempos de respuesta inaceptables.

Contexto: Uso del optimizador

Descripción: Evitar usar comodines al inicio de la palabra, LIKE ‘%pples vs LIKE ‘A%s’ (índex scan vs. índex search)

Titulo: Uso de operadores

Área: Programación SQL

Problema: El optimizador realiza búsqueda secuencial, provocando problemas de concurrencia y tiempos de respuesta inaceptables.

Contexto: Consulta de Información

Descripción: Evitar buscar por operadores de desigualdad (<>, not). Dado que el optimizador no tiene una constante a buscar en el árbol, realiza un barrido de todo el árbol de índices.

Programación

Titulo: Uso de estadísticas y uso de optimizador

Área de aplicación: Programación SQL

Problema que resuelve: Acceso secuencial a datos

Contexto: Acceso a Información

Descripción: Checar siempre el plan de acceso a ejecutarse y verificar que usa los índices adecuadamente.

Beneficio: Uso de índices y por tanto, acceso mas rápido.

Titulo: Envío de mensajes innecesarios

Área: Programación SQL

Problema que resuelve: Trafico de red innecesario y por tanto, tiempo de respuesta lento.

Contexto: Acceso y procesamiento de información.

Descripción: Usar SET NOCOUNT ON/ FEEDBACK OFF al inicio de batches, procedimientos almacenados y triggers

Beneficio: Evitar tráfico inútil en la red.

Programación

Titulo: Generación inútil de plan de ejecución

Área de aplicación: Programación SQL

Problema que resuelve: Al realizar una consulta complicada implica su generación de plan de ejecución, la cual toma tiempo. Si no existen cambios en índices o estructura de las tablas, no se necesita generar el plan cada vez.

Contexto: Acceso a Información

Descripción : Incorporar joins complicados, frecuentes y con cálculos en vistas, de tal forma que al seleccionar los datos de la vista, evitemos la generación del plan de ejecución.

Beneficio: Reducción en tiempo de ejecución.

Programación

Titulo: Centralización de la lógica del negocio en el Manejador.

Área de aplicación: Programación SQL

Problema que resuelve: Trafico de red innecesario.

Contexto: Arquitectura cliente-servidor

Descripción: No permitir a las aplicaciones front-end consultar o manipular datos directamente usando sentencias SQL. Crear procedimientos almacenados, permitiendo así un acceso consistente en todos los módulos de la aplicación y centralizando la lógica del negocio dentro de la base de datos.

Beneficio: Mejor tiempo de respuesta al mandar del back-end al front-end solamente los datos resultados.

Programación

Titulo: Uso de constraints para chequeo de integridad referencial.

Área de aplicación: Programación SQL

Problema que resuelve: La programación por triggers envuelve procedimiento procedural, por tanto es mas lenta su respuesta.

Contexto: Procesamiento de información

Descripción: El uso de constraints es una forma de reducir el dominio de los datos. Checar integridad referencial vía constraints, si son complicados entonces usar triggers.

Beneficio: Mejora de rendimiento al usar teoría de conjuntos .

Titulo: Control de deadlocks

Área de aplicación: Programación SQL

Problema que resuelve: Interbloqueo, donde los dos procesos esperan la liberación del recurso que utiliza el otro proceso.

Contexto: Acceso y procesamiento de información.

Descripción: Accesar tablas siempre en el mismo orden en los procedimientos almacenados y triggers para evitar deadlocks (abrazos mortales/interbloquos)

Programación

Titulo: Manejo de Transacciones

Área de aplicación: Programación SQL

Problema que resuelve: Bloqueo de recursos por tiempo prolongado, afectando a los demás procesos que desean utilizar dichos recursos.

Contexto: Acceso y procesamiento de Información

Descripción: Cuando se modifica/inserta/borra registros en tablas. El manejador aloja candados para ese recurso a nivel registro/pagina/tabla. Si existe otro proceso que desea accesar a dicha pagina, no necesariamente al mismo recurso y dicho recurso esta bloqueado, el proceso no puede ejecutarse. Por tanto, se recomienda que las transacciones sean lo mas cortas posibles a través de commit cada determinado numero de registros. Existen otras opciones como particionamiento, cambio de esquema de bloqueo a uno menos restrictivo, siempre y cuando se tengan los recursos para manejar dicho esquema.

Beneficio: Mayor concurrencia.

Programación

Titulo: Centralización de tareas sencillas en el front-end

Área de aplicación: Programación SQL

Problema que resuelve: Poca concurrencia

Contexto: Arquitectura cliente servidor

Descripción: Realizar tareas de validación sencillas en el front-end.

Beneficio: Liberación de carga innecesaria al Manejador de bases de datos.

Titulo: Centralización de tareas sencillas en el front end

Área de aplicación: Programación SQL

Problema que resuelve: El Manejador se libera de trabajos sencillos y puede atender a mas procesos que requieren manejo de información residente en la base de datos

Contexto: Arquitectura cliente servidor

Descripción: Realizar manipulaciones de cadenas, concatenaciones, numerado de registros, conversiones, en el front-end.

Beneficio: Liberación de carga innecesaria al Manejador de bases de datos.

Programación

Titulo: Manejo de recursos

Área de aplicación: Programación SQL

Problema que resuelve: Tiempo de ejecución, uso de memoria excesivos, poca concurrencia.

Contexto: Procesamiento de información

Descripción: No hacer llamadas a funciones repetidamente dentro de un programa. Hacer la llamada una vez y guardarlo en una variable.

Beneficio: Utilización optima de memoria y procesamiento, mejor tiempo de respuesta.

Conclusión

Programar con vistas hacia el rendimiento general del sistema y considerando toda la infraestructura que se encuentra a disposición.

Crea, Verifica, Valida, Acepta, Mide, Mejora y Controla

…en cualquier etapa CORRIGE