UNIVERSIDAD GABRIELA MISTRAL
FACULTAD DE INGENIERIA

TENDENCIA DE USO
MOTOR Y MODELOS DE BASES DE DATOS

Tesis para optar al título de Ingeniero de Ejecución en Informática

Autor : Manuel Urra Rivera.
Profesor Guía : Roberto Caru.
Profesor Integrante : Jorge Tapia

Santiago – Chile

Septiembre, 2015

INDICE

AGRADECIMIENTOS

Agradezco a Dios y todas sus manifestaciones por permitirme Vivir y disfrutar las etapas de un proceso el cual me ha servido para conocer mis fortalezas y debilidades, las cuales he tenido que superar de forma consciente para lograr el cometido final. Además a mi familia, padres y en especial a mi Esposa e Hijas que sacrificaron su tiempo valioso en pro de mi desarrollo profesional, brindándome apoyo, comprensión y amor en estos años.

1 *Introducción Motor y Modelo de Base de Datos*

Dentro de las características de motores de bases de datos y modelos se pueden definir varios de estos, los cuales están orientados de una forma similar ya sea por su forma o por su estructura. Esto nos da como concepto al revisar y orientar la información sea por calidad de esta o cantidad tenemos que considerar cuál de estos motores de base de datos es el más apto para el planteamiento del modelo requerido, en base a la situación actual que se tienen cantidades enormes de información y las necesidad urgente de obtener el dato preciso en un tiempo mínimo para el usuario final de este.

Dentro de los motores de bases de datos podemos encontrar los modelo de estructura Vertical la cual es la de mayor uso y los modelos de estructura horizontal que son aplicada a grandes recipientes de información, esta estructura lleva a búsquedas más efectivas en tiempo, cantidad y en calidad de información.

A partir de este análisis debemos plantear y decidir que motor y modelo de base datos debemos usar para un mejor comportamiento en etapas críticas, esto nos da como resultado que a mayor cantidad de datos y que tipo de búsqueda realizaremos tendremos mejor respuesta.

Dentro de esta situación expuesta tendremos que seleccionar el mejor motor y modelo de base de datos para normar la información y conseguir respuestas acorde a la necesidad expuesta del entorno.

1.1 Motivación

Según los análisis generados en el transcurso del tiempo se puede determinar que las bases de datos y sus dimensiones se están incrementado en forma sustancial, por lo tanto esto genera la necesidad de realizar un planteamiento de cuál sería el motor y la base de datos a aplicar según las necesidades de fondos futuras.

Es así que por esto la motivación de este planteamiento es proporcionar una orientación según la necesidad del incremento de datos asociados a un motor y a una base de datos.

Dentro del mismo contexto para realizar la selección de una herramienta de motor y base de datos se ha incorporado dentro de este planteamiento dar la importancia necesaria al tamaño de la información base para poder plantear de buena forma que motor y modelo recomendar planteando una solución con soporte en el futuro.

Tenemos que la cantidad de información a contener dentro de este recipiente, es un elemento muy importante al momento de implementar estructuras de data en gran dimensión, por lo tanto nuestra decisión es un apoyo a la selección y adquisición de estos elementos, esto nos proporciona un marco de continuidad operacional más estructurada y muy necesaria en la tendencia actual de la necesidad de información.

1.2 Hipótesis

El Objetivo principal es poder realizar el análisis de todos los modelos posibles y evaluar cuál sería la tendencia a futuro de uso proyectando que los datos se están incrementado en forma exponencial en un tiempo muy corto.

En base a esto se aplicarán modelos de prueba los cuales por cantidad y disponibilidad del acceso a los datos responda en forma más eficiente a los requerimientos actuales de la información seleccionada dentro de Big data, no importando como éste su estructura de relación.

1.3 Objetivo General

El objetivo General es evaluar cómo se determina que un modelo especifico tiene un mejor comportamiento, esto será nuestro mejor indicador de cuál es la tendencia futura del tipo de motor y modelo de bases de datos y sus tendencias a futuro.

1.4 Objetivos Específicos

El objetivo específico es determinar cuál es el mejor modelo de motor y base de datos aplicados en diferentes estructuras y su motor de datos con la mejor respuesta para una data compleja pero común de mercado.

1.5 Alcances

En la actualidad existe un sin número de estándares en modelos de motores de base de datos y base de datos de calidad los cuales son factibles de aplicar en las organizaciones, pero con la necesidad de tener una base de datos y la premura se aplican los estándares como norma organizacional. Dado esto se requiere investigar en forma directa cuál es la necesidad real que genera el análisis del estándar se requiere según el producto a implementar. Este análisis se realiza por la necesidad de llegar a data de mayor cantidad(Big data) y con mejor respuesta en los requerimientos actuales en todos los requerimientos definidos.

1.6 Antecedentes

En la actualidad existen Muchas empresas con múltiples funciones, dentro de su organización recae la responsabilidad de liderar proyectos de desarrollo, mantención o adquisición de nuevas tecnologías de información. Estas han tenido que incluir mejoras en sus bases de datos debido a la gran cantidad de información que tienen que almacenar y mantener en forma histórica para todos sus procesos. Para asegurar el acceso al dato requerido en un instante de consulta y con esto promover el buen desempeño y existo del almacén de datos ya sea para sus clientes internos o externos.

Según la información que se recopila con respecto a esto, existen muchas falencias actualmente al suministrar el dato preciso en el momento requerido, pero

esto puede y tiene que ser solucionado con el análisis previo del tipo de data que será la que se requiere y poder así definir con mayor exactitud el planteamiento para dar la forma de qué motor y que base de datos es la más adecuada. Realizando esto nos dará como resultado que serán de mejor calidad las consultas para la ingeniería de software que actualmente se proponen.

1.7 Motores de bases de datos

Dentro de esta estructura podemos definir que existen motores de base de datos relacionales de orientación horizontales y verticales los cuales serán analizadas dentro de este contexto para tener mejor claridad en la el planteamiento final del tratamiento de la información .

2 Definición

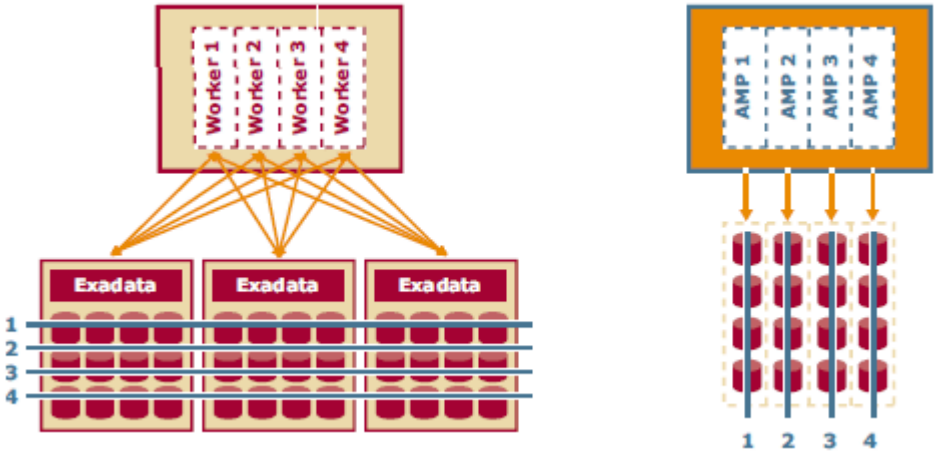
Dentro de este contexto tenemos que definir lo siguiente: existen muchas bases de datos relacionales las cuales por motivos de origen se crearon en forma vertical para su grabación y horizontal para su lectura y su motor fue predispuesto para esto sin pensar que podrían existir otras posibilidades de almacenamiento.

Todo esto nos lleva a revisar otra posibilidad de almacenamiento, la cual es de estructura Horizontal, en grabación y lectura. Esto nos deja varias incógnitas de por qué no se realizan con este motor de base de datos si muestra que es de mayor poder en la búsqueda y acceso a la información.

Dentro del contexto actual, tenemos que la importancia del dato es la mayor ganancia de cualquier ente si se tiene en el momento y minuto preciso, por lo tanto si esto continua en crecimiento, el modelo relacional vertical para su grabación y horizontal para su lectura tiende a ser lento para gran cantidad de informacion y

se requiere de mayor velocidad de la búsqueda de esta información. En base a lo anteriormente estipulado el concepto el motor de base de datos con orientación horizontal pasa a ser de mayor importancia por el crecimiento de la información y el acceso a esta.

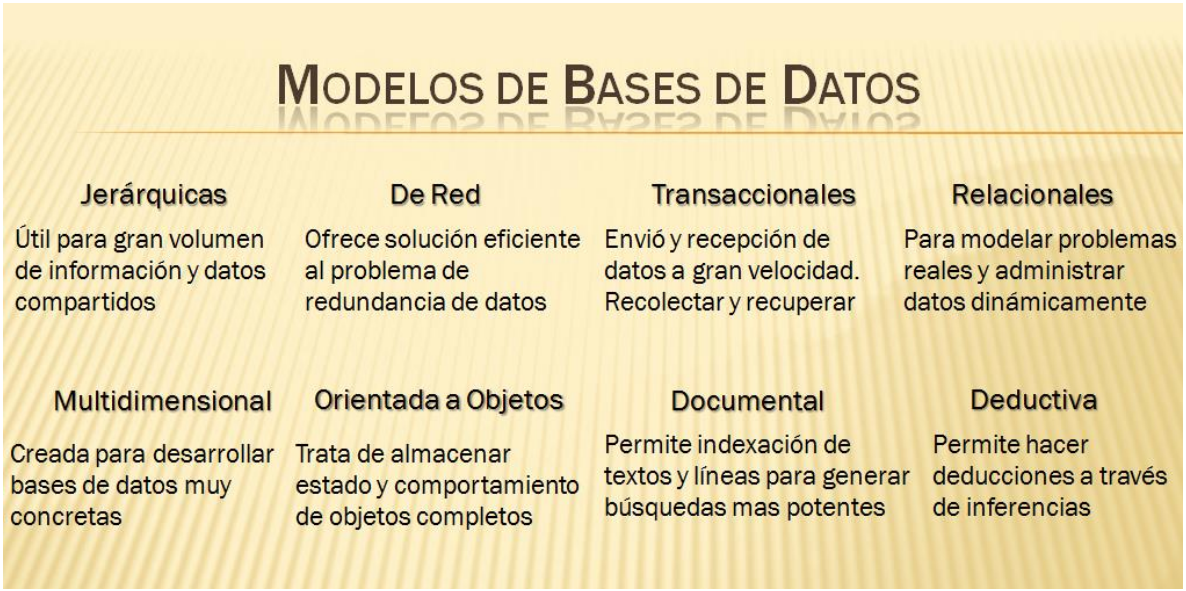
Otras bases de Datos (Horizontal) Teradata y Otras (Vertical)



3 *Modelo base de datos relacional*

Estas son estructuras de orden de datos en las que se puede optimizar al máximo su acceso y grabación de información, permitiendo un óptimo manejo de esta. Dentro de estos parámetros tenemos varios modelos los cuales pueden ser usados según el tipo de producto que se requiera realizar, ya sea por el tiempo de grabación o por el acceso en tiempo óptimo a la información requerida.

Como muestra se pueden indicar algunos de estos modelos conocidos actualmente.



3.1 Modelo de Base de Datos Jerárquicas

Útil para gran volumen y datos compartidos

Relación jerárquica

Esta relación está basada en niveles jerárquicos de superioridad o subordinación entre conceptos.

El concepto superior constituye una clase, mientras que los conceptos subordinados representan elementos o partes de esa clase.

Esta relación se indica mediante los siguientes símbolos:

- BT (broaderterm = término genérico), situado entre un concepto específico y un concepto genérico y acompañado de una cifra que indica el número de niveles jerárquicos que hay entre el término específico y cada uno de los términos genéricos que le corresponden.

Ejemplo:

Norma

BT1 normalización

BT2 reglamentación técnica

Los conceptos que carecen de término genérico se denominan términos cabecera (top terms).

NT (narrowerterm = término específico), situado entre un concepto genérico y un concepto específico y acompañado de una cifra que indica el número de niveles

jerárquicos que hay entre el término genérico y cada uno de los términos específicos que le corresponden.

Ejemplo:

Normalización

NT1 armonización de normas

NT1 homologación

NT2 certificación comunitaria

NT1 marca de conformidad CE

NT1 norma

NT2 norma de calidad

NT2 norma de producción

NT2 norma de seguridad

NT2 norma técnica

NT1 norma internacional

NT2 norma europea

La relación jerárquica permite al usuario de un sistema documental adaptar el nivel de especificidad del mismo navegando por la jerarquía. Puede ampliar o precisar su pregunta seleccionando conceptos que tengan un sentido más amplio (por ejemplo, "normalización", o "reglamentación técnica") o más estricto ("norma de producción", "norma de calidad" o "norma técnica").

La relación jerárquica precisa el sentido del concepto en su contexto. La acepción del concepto prensa, por ejemplo, queda determinada por su subordinación al concepto medio de comunicación de masas.

3.2 Modelo de Base de Datos Relación genérica

Identifica el vínculo entre una clase (término genérico o término cabecera) y sus elementos (términos específicos).

Ejemplo:

protectedarea

NT1 parque nacional

NT1 reserva natural

3.3 Modelo de Base de Datos Relación partitiva

Corresponde a algunas situaciones en las cuales el nombre de la parte indica el nombre del todo con independencia del contexto. El nombre del todo representa el término genérico y el nombre de la parte representa el término o términos específicos del concepto. En EuroVoc se aplica principalmente a las siguientes clases de términos:

— Lugares geográficos

Ejemplo:

Océano

NT1 Océano Antártico

NT1 Océano Ártico

NT1 Océano Atlántico

NT1 Océano Índico

NT1 Océano Pacífico

— Disciplinas o ámbitos de conocimiento

Ejemplo:

Química

NT1 bioquímica

NT1 química electroquímica

NT1 química fotoquímica

NT1 química analítica

— Estructuras sociales jerarquizadas

Ejemplo:

Mesa del Parlamento

NT1 cuestor

NT1 presidente del Parlamento

NT1 vicepresidente del Parlamento

3.4 Modelo de Base de Datos De Red

El modelo ofrece soluciones eficientes al problema de redundancia de datos.

Podemos considerar al modelo de bases de datos en red como de una potencia intermedia entre el jerárquico y la relacional que estudiaremos más adelante.

Su estructura es parecida a la jerárquica aunque bastante más compleja, con esto se consiguen evitar, al menos en parte, los problemas.

Los conceptos fundamentales que debe conocer el administrador para definir el esquema de una base de datos jerárquica, son los siguientes:

- Registro: Viene a ser como cada una de las fichas almacenadas en un fichero convencional.

- Campos o elementos de datos: Son cada uno de los apartados de que se compone una ficha.

- Conjunto: Es el concepto que permite relacionar entre sí tipos de registro distintos.

Podemos imaginar los registros simplemente como fichas de un fichero. Para ilustrar el concepto de conjunto, supongamos que tenemos un tipo de registro de clientes, y un tipo de registro de vuelos de avión, y queremos asociar ambas informaciones, de manera que para cada vuelo queremos saber cuáles son los pasajeros que viajan en él. La forma de hacerlo es a través de un conjunto. Este conjunto relaciona dos tipos de registro. Uno de ellos es el registro propietario del conjunto, y el otro es el miembro. Veamos el siguiente diagrama de Bachman que nos aclarará las cosas un poco más.

Cada tipo de conjunto, posee, a su vez, una serie de ocurrencias de conjunto, donde cada una de estas está formada por una instancia del tipo propietario, y una, varias o ninguna instancia del tipo miembro.

P.ej. una ocurrencia de conjunto puede ser:

33387698-K Juan Linares

83698637-H Pedro Hernández

24885764-G Luis Caro

64653627-J Pablo Mármol

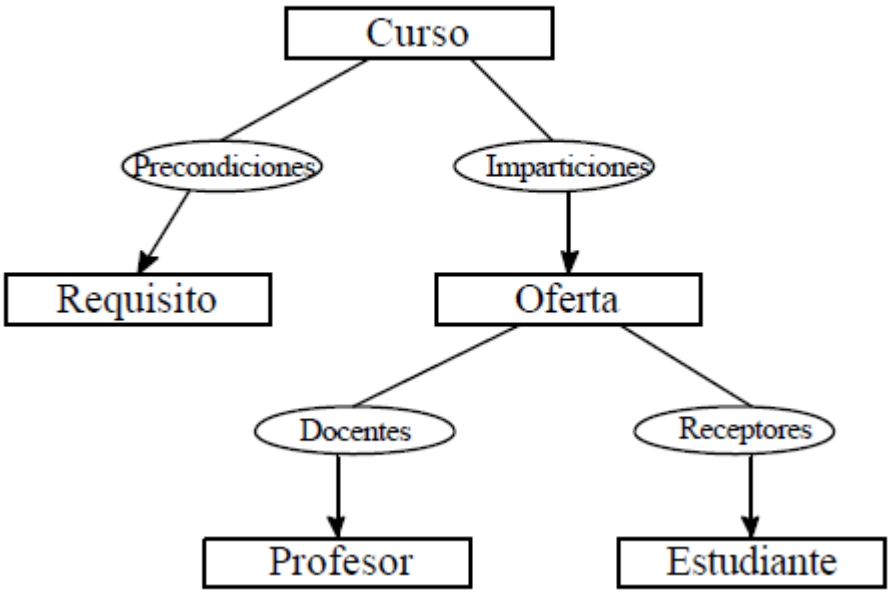
Una restricción bastante importante de este modelo, es que una ocurrencia de registro miembro puede pertenecer como máximo a una sola instancia de un determinado conjunto, aunque puede participar en varios tipos de conjuntos distintos.

Este modelo en red es más potente que el modelo jerárquico, ya que aquél puede simularse, aplicando una jerarquía de conjuntos en varios niveles. P.ej., el ejemplo jerárquico del punto anterior quedaría ahora como: Por otro lado, en un conjunto Concreto, el tipo de registro propietario no puede ser, a su vez, el mismo que el tipo de registro miembro, o sea, un mismo tipo de registro no puede intervenir en el mismo conjunto como propietario y como miembro a la vez.

Para ilustrar por qué el modelo en redes es más potente que el modelo jerárquico, basta con observar un conjunto como el siguiente

:

Cómo simular el ejemplo jerárquico mediante el modelo en red.



3.5 Modelo de Base de Datos Transaccionales

Envió y recepción de datos a gran velocidad recolectar y recuperar.

Son bases de datos cuyo único fin es el envío y recepción de datos a grandes velocidades, estas bases son muy poco comunes y están dirigidas por lo general al entorno de análisis de calidad, datos de producción e industrial. Es importante entender que su fin único es recolectar y recuperar los datos a la mayor velocidad posible, por lo tanto, la redundancia y duplicación de información no es un problema como con las demás bases de datos, por lo general, para poderlas aprovechar al máximo permiten algún tipo de conectividad a bases de datos relacionales.

Un ejemplo habitual de transacción es el traspaso de una cantidad de dinero entre cuentas bancarias. Normalmente se realiza mediante dos operaciones distintas, una en la que se decrece el saldo de la cuenta origen y otra en la que incrementamos el saldo de la cuenta destino.

Para garantizar la atomicidad del sistema es, decir, para que no aparezca o desaparezca dinero, las dos operaciones deben ser atómicas, es por esto que, el sistema debe garantizar que, bajo cualquier circunstancia (incluso una caída del sistema), el resultado final es que, o se han realizado las dos operaciones, o bien no se ha realizado ninguna.

3.6 Modelo de Base de Datos Relacionales

Es una base de datos que cumple con el modelo relacional, el cual es el más utilizado en la actualidad para implementar bases de datos ya planificadas.

Permiten establecer interconexiones (relaciones) entre los datos (que están guardados en tablas), y a través de dichas conexiones relacionar los datos de ambas tablas, de ahí proviene su nombre: “Modelo Relacional”.

Una Base de Datos Relacional, es aquella que cumple con el modelo relacional, el cual es el modelo más utilizado en la actualidad para implementar bases de datos ya planificadas. Permiten establecer interconexiones (relaciones) entre los datos (que están guardados en tablas), y a través de dichas conexiones relacionar los datos de ambas tablas, de ahí proviene su nombre: "Modelo Relacional". Tras ser postuladas sus bases en 1970 por Edgar Frank Codd, de los laboratorios IBM en San José (California), no tardó en consolidarse como un nuevo paradigma en los modelos de base de datos.

El primer paso para crear una base de datos, es planificar el tipo de información que se quiere almacenar en la misma, teniendo en cuenta dos aspectos: la información disponible y la información que necesitamos.

La planificación de la estructura de la base de datos, en particular de las tablas, es vital para la gestión efectiva de la misma. El diseño de la estructura de una tabla consiste en una descripción de cada uno de los campos que componen el registro y los valores o datos que contendrá cada uno de esos campos.

Los campos son los distintos tipos de datos que componen la tabla, por ejemplo: nombre, apellido, domicilio. La definición de un campo requiere: el nombre del campo, el tipo de campo, el ancho del campo, etc.

Los registros constituyen la información que va contenida en los campos de la tabla, por ejemplo: el nombre del paciente, el apellido del paciente y la dirección de este. Generalmente los diferentes tipos de campos que se pueden almacenar son los siguientes: Texto (caracteres), Numérico (números), Fecha / Hora, Lógico (informaciones lógicas si/no, verdadero/falso, etc.), imágenes.

En resumen, el principal aspecto a tener en cuenta durante el diseño de una tabla es determinar claramente los campos necesarios, definirlos en forma adecuada con un nombre especificando su tipo y su longitud.

3.7 Modelo de Base de Datos Multidimensionales

Este modelo fue creada para desarrollar bases de datos muy concretas

Las bases de datos multidimensionales son una variación del modelo relacional que utiliza cubos OLAP para organizar los datos y expresar las relaciones entre ellos. Las principales ventajas de este tipo de bases de datos son la versatilidad para cruzar información y la alta velocidad de respuesta. Esto las convierte en herramientas básicas para soluciones de Business Intelligence o de Big Data, donde el análisis de los datos resulta crucial.

En general, estamos acostumbrados a trabajar con bases de datos orientadas a transacciones (conocidas como bases de datos OLTP). A continuación veremos

las principales diferencias con las bases de datos multidimensionales (conocidas como bases de datos OLAP).

OLTP – On-Line Transactional Processing

Los sistemas OLTP son bases de datos relacionales (RDBMS) orientadas a transacciones. Una transacción es una secuencia de operaciones llevada a cabo por una base de datos de manera atómica.

Las operaciones pueden ser de cuatro tipos diferentes: SELECT, INSERT, DELETE y UPDATE. Al tratarse de un proceso atómico, cada transacción solo tiene dos posibles finales: commit (si se han llevado a cabo correctamente todas las operaciones) o rollback (cuando una operación de la secuencia ha fallado, en cuyo caso hay que deshacer los cambios producidos por el resto de las operaciones de la transacción y alertar del error). Las transacciones son el pilar de prácticamente cualquier programa de gestión o página web del mundo. Su necesidad se ve muy clara, por ejemplo, en el sector de la banca.

Esto garantiza las transacciones: las características ACID para que una base de datos OLTP pueda asegurar las transacciones es necesario que sea ACID compliant. ACID es un acrónimo de Atomicity, Consistency, Isolationand Durability:

Atomicidad: asegura que la operación se ha realizado correcta y completamente, de forma que no pueda quedar a medias en caso de que surja cualquier error.

Consistencia: asegura la integridad de la base de datos, es decir, que solo se ejecutan aquellas operaciones que no van a romper las reglas, restricciones y claves foráneas. La propiedad de consistencia sostiene que cualquier transacción llevará a la base de datos desde un estado válido a otro estado similar.

Aislamiento: asegura que dos transacciones que afectan a la misma información (tabla, fila o celda) son independientes y no generan errores ni dead locks. Existen

cuatro posibles niveles de aislamiento: READ UNCOMMITTED, READ COMMITTED, REPEATABLE READS y SERIALIZABLE, en función del tipo de bloqueos que se crean para proteger la información.

Durabilidad: asegura que una vez realizada la operación, ésta persistirá y no se podrá deshacer aunque falle el sistema. Será almacenada en disco, no solo en memoria.

Propiedades de las bases de datos OLTP .Los sistemas OLTP son la versión tradicional de una base de datos: se diseñan utilizando un modelo entidad-relación, se implementan en los motores típicos de base de datos (Oracle, SQLServer, MySQL… etc.) y dan soporte a la mayor parte del software del mercado.

Optimizadas para lecturas y escrituras concurrentes: gracias a las propiedades ACID, el acceso a los datos está adaptado para tareas frecuentes de lectura y escritura.

Organizadas según la capa de aplicación: las tablas y los datos se estructuran según el software que los maneja: programa de gestión a medida, ERP, CRM, BPM… etc.

Adaptadas a cada empresa o departamento: dado que en muchas ocasiones se utiliza software no integrado, los formatos de los datos no suelen ser uniformes en los diferentes departamentos. Es común la falta de compatibilidad y la existencia de islas de datos.

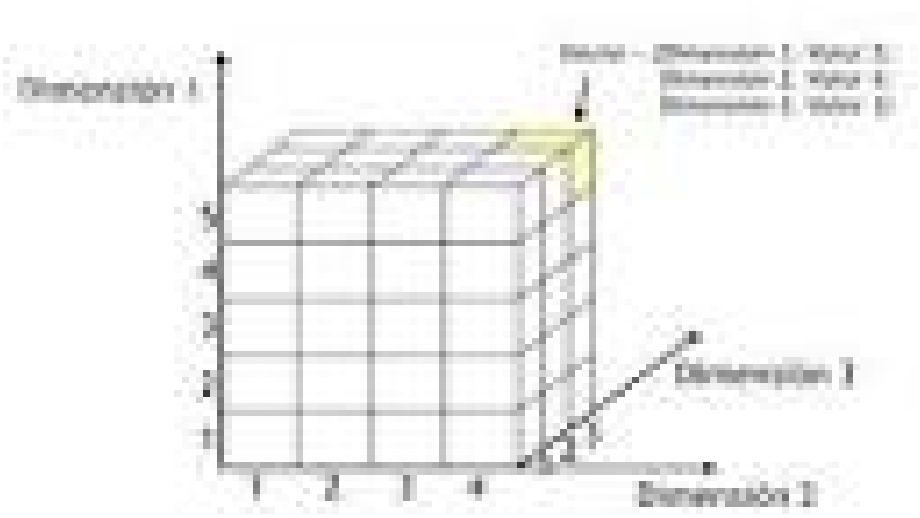
Consultas realizadas en SQL, modificaciones en DML: el SQL (Standard QueryLanguage) es el lenguaje de consulta universal para leer bases de datos relacionales (cláusula SELECT), mientras que DML (Data Manipulation Language) es el estándar para realizar modificaciones (cláusulas INSERT, UPDATE y DELETE).

Gestión de datos históricos inexistente: el historial de cambios suele limitarse a datos actuales o recientes. Salvo sistemas de backup, o que el software tenga una

funcionalidad específica para ello, no se suelen manejar valores históricos para cada campo.

OLAP: On-Line AnalyticalProcessing

Los sistemas OLAP son bases de datos orientadas al procesamiento analítico. Este análisis suele implicar, generalmente, la lectura de grandes cantidades de datos para llegar a extraer algún tipo de información útil: tendencias de ventas, patrones de comportamiento de los consumidores, elaboración de informes complejos… etc.



Cubo multidimensional.

3.8 Modelo de Base de Datos Orientadas a objetos

Este modelo trata de almacenar estado y comportamiento de objetos completos.

Una base de datos es una colección de datos que puede constituirse de forma que sus contenidos puedan permitirse el encapsular, tramitarse y renovarse sencillamente, elementos de datos, sus características, atributos y el código que opera sobre ellos en elementos complejos llamados objetos. Las bases de datos están constituida por objetos, que pueden ser de muy diversos tipos, y sobre los cuales se encuentran definidas unas operaciones donde interactúan y se integran con las de un lenguaje de programación orientado a objetos, es decir, que los componentes de la base de datos son objetos de los lenguajes de programación además que este tipo de base de datos están diseñadas para trabajar con lenguajes orientados a objetos también manipulan datos complejos de forma rápida y segura.

Las bases de datos orientadas a objetos se crearon para tratar de satisfacer las necesidades de estas nuevas aplicaciones. La orientación a objetos ofrece flexibilidad para manejar algunos de estos requisitos y no está limitada por los tipos de datos y los lenguajes de consulta de los sistemas de bases de datos tradicionales.

Los objetos estructurados se agrupan en clases. Las clases utilizadas en un determinado lenguaje de programación orientado a objetos son las misma que serán utilizadas en una base de datos; de tal manera, que no es necesaria una transformación del modelo de objetos para ser utilizado. De forma contraria, el modelo relacional requiere abstraerse lo suficiente como para adaptar los objetos

del mundo real a tablas. El conjunto de las clases se estructuran en subclases y superclases, los valores de los datos también son objetos.

Muchas organizaciones que actualmente usan tecnología orientada a objetos también desean los beneficios de los sistemas de gestión de base de datos orientados a objetos. En otras palabras, se desea la migración de bases de datos y aplicaciones de bases de datos relacionales a orientadas a objetos. La migración a la tecnología de objetos consiste de la ingeniería reversa de los programas de aplicación y la migración de la base de datos. El objetivo de la migración de la base de datos es tener un esquema equivalente y la base de datos disponibles. Esto desde luego puede ser logrado por medio de la transformación manual del código de los programas lo cual resulta demasiado complicado. Para esto existen tres enfoques que hacen uso de la tecnología de objetos para bases de datos relacionales.

a.- Construir una interface orientada a objetos sobre el sistema de base de datos relacional.

b.- La migración a un sistema de base de datos relacional/objetos.

c.- Conversión del esquema de base de datos relacional a uno orientado a objetos.

El primer enfoque retiene la base de datos relacional y crea una interface orientada a objetos encima de ésta. Este enfoque es el más fácil; no existe interrupción del sistema para la migración de datos y no existe perdida semántica de la información. Por otro lado, el rendimiento disminuye debido que no existe un buen acoplamiento entre los dos paradigmas en el tiempo de ejecución.

En el segundo enfoque, los datos deben ser migrados de acuerdo con el motor de base de datos, y las características orientadas a objetos solo pueden ser explotadas con la modificación o extensión del esquema.

El tercer enfoque es la migración de la base de datos en donde un nuevo esquema bajo el OODBMS es creado y los datos son migrados de la base de datos relacional a la orientada a objetos.

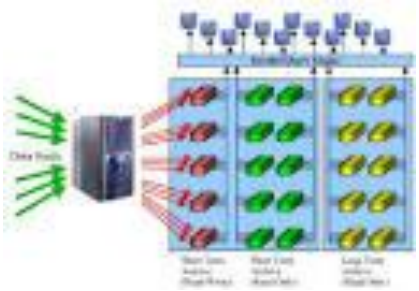
Una base orientada a objetos es una base de datos que incorpora todos los conceptos importantes del paradigma de objetos:

Encapsulación: Propiedad que permite ocultar información al resto de los objetos, impidiendo así accesos incorrectos o conflictos.

Herencia: Propiedad a través de la cual los objetos heredan comportamientos dentro de una jerarquía de clases.

Polimorfismo: Propiedad de una operación mediante la cual puede ser aplicada a distintos tipos de objetos.

En bases de datos orientadas a objetos, los usuarios pueden definir operaciones sobre los datos como parte de la definición de la base de datos. Una operación (llamada función) se especifica en dos partes. La interfaz (o signatura) de una operación incluye el nombre de la operación y los tipos de datos de sus argumentos (o parámetros). La implementación (o método) de la operación se especifica separadamente y puede modificarse sin afectar la interfaz. Los programas de aplicación de los usuarios pueden operar sobre los datos invocando a dichas operaciones a través de sus nombres y argumentos, sea cual sea la forma en la que se han implementado. Esto podría denominarse independencia entre programas y operaciones.



3.9 Modelo de Base de Datos Documentales

Este modelo permite indexación de textos y líneas para generar búsquedas más potentes.

Los Sistemas de Gestión Documental (SGD) - Text Retrieval Systems, en inglés- son un tipo de programas muy conocidos en el ámbito de la información y documentación, ya que están especialmente pensados para la gestión de información textual y de documentos cognitivos. Sus principales características se pueden sintetizar en lo siguiente: disponen de un modelo de registro flexible (campos de longitud variable, campos multivalor, etc.), facilitan el acceso a los registros a través del fichero inverso, contienen un conjunto de variadas prestaciones de recuperación de la información, y están dotados de diversos instrumentos para el control terminológico. Algunos de los sistemas más conocidos y extendidos son CDS/ISIS, FileMaker, Knosys, e Inmagic DB/Text.

Sobre ellos se han realizado aproximaciones teóricas de carácter global, entre las que se pueden destacar las de Sieverts y otros investigadores belgas (1991-93), autores de una serie de artículos muy completos y exhaustivos que describían las características de este tipo de programas, elaborando una tipología y presentando una evaluación muy detallada de unos treinta productos. Posteriormente, William Saffady, por dos veces, (1995) (2000), también realizó una aproximación actualizada a los SGD., se han publicado diversos trabajos de carácter global siendo los más recientes una monografía de Abadal y Codina (2005), este contiene informes y estudios diversos sobre programas de gestión de contenidos, entre los cuales se incluyen referencias a sistemas de gestión documental.

Los SGD han servido para que pequeñas y medianas organizaciones hayan podido crear bases de datos documentales de tipo referencial permitiendo a los

usuarios de estos centros la localización y consulta de sus fondos (ya se trate de libros, artículos de revista, fotografías u otro tipo de documentos).

Actualmente, estos programas a los que hacemos referencia tienen comercializada aplicaciones informáticas (denominadas vulgarmente pasarelas web) que permiten la consulta, desde un navegador web, de las bases de datos creadas por ellos.

Esto permite ampliar notablemente el espectro de usuarios potenciales de las bases de datos ya que no es necesario utilizar redes de área local para compartir el uso de las bases de datos ni, mucho menos, desplazarse a la ubicación física donde éstas residen.

El objetivo de este texto es, precisamente, mostrar la situación actual en el mercado de estas aplicaciones, valorarlas, y señalar tendencias de futuro. Para ello, sintetizaremos, en primer lugar, el funcionamiento básico de estas pasarelas web y, a continuación, valoraremos de forma comparativa aquellas que tienen un uso más extendido en el mercado.

Publicar bases de datos

Hasta hace pocos años, los productores y los distribuidores de bases de datos (estos últimos en particular) acostumbraban a tener un carácter especializado y a disponer, por tanto, de una potente estructura empresarial. Esta situación ha cambiado radicalmente con la eclosión de Internet y el desarrollo de distintas herramientas fácilmente configurables y adaptables que ponen al alcance de pequeños y medianos centros de información y documentación, e incluso de usuarios personales, la posibilidad de convertirse en productores y distribuidores de bases de datos.

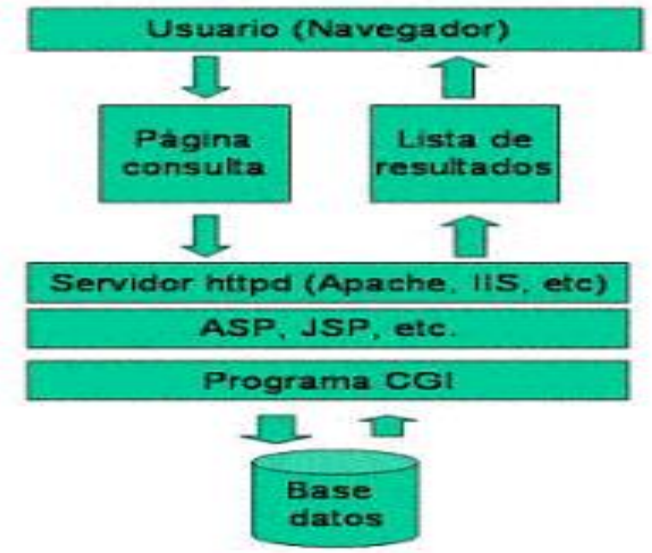
Pequeñas y medianas organizaciones que habían creado bases de datos documentales, ya las que nos hemos referido en el anterior apartado, están llevando a cabo un proceso generalizado de publicación de sus contenidos en la web. Esto permite que los usuarios sólo necesiten del navegador para poder

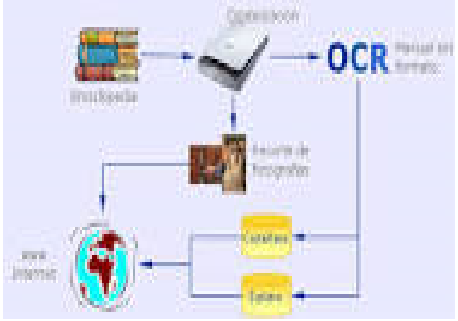
acceder a los registros de forma actualizada y que dispongan, en la mayoría de los casos, de las mismas prestaciones de consulta y explotación que tienen los sistemas de gestión documental cuando se consultan localmente o mediante redes de área local.

Elementos

Ahora bien, para que este método de acceso sea posible, es necesario disponer, en el lado del servidor web, de un programa o un conjunto de programas que permita establecer la comunicación entre dos entornos en principio incompatibles o distintos: la base de datos gestionada por el SGD, por un lado, y el servidor web, que es el que atiende a los navegadores que utilizan los usuarios y que sólo es capaz de interpretar páginas html transmitidas mediante el protocolo http, por el otro. Estos programas suelen recibir la denominación de pasarelas web ya que actúan como intermediarios entre los registros de la base de datos y los datos codificados en html que proceden del formulario de consulta que ha rellenado un usuario.

El siguiente esquema muestra los elementos básicos que intervienen en este proceso y su funcionamiento.





3.10 Modelo de Base de Datos Deductivas

Este modelo permite hacer deducción a través de inferencias

Modelos Semánticos.

Ciertos tipos de inferencia, integran a los sistemas de almacenamiento de datos. Modelos Orientados a Objetos. Objeto y versiones de objetos, consulta de ciertos datos. Aparecen las Bases de datos Deductivas.

Un sistema de base de datos deductivos, es un sistema de base de datos pero con la diferencia de que permite hacer deducciones a través de inferencias.

Se basa principalmente en reglas y hechos que son almacenados en la base de datos. También las bases de datos deductivas son llamadas base de datos lógica, a raíz de que se basan en lógica matemática. Una base de datos deductiva es, en esencia, un programa lógico; mapeo de relaciones base hacia hechos, y reglas que son usadas para definir nuevas relaciones en términos de las relaciones base y el procesamiento de consultas.

Los sistemas Bases de Datos Deductivas intentan modificar el hecho de que los datos requeridos residan en la memoria principal (por lo que la gestión de

almacenamiento secundario no viene al caso) de modo que un SGBD se amplíe para manejar datos que residen en almacenamiento secundario.

En un sistema de Bases de Datos Deductivas por lo regular se usa un lenguaje declarativo para especificar reglas. Con lenguaje declarativo se quiere decir un lenguaje que define lo que un programa desea lograr, en vez de especificar los detalles de cómo lograrlo. Se especifican de manera similar a como se especifican las relaciones, excepto que no es necesario incluir los nombres de los atributos.

Recordemos que una tupla en una relación describe algún hecho del mundo real cuyo significado queda determinado en parte por los nombres de los atributos.

En una Base de Datos Deductiva, el significado del valor del atributo en una tupla queda determinado exclusivamente por su posición dentro de la tupla. Se parecen un poco a las vistas relacionales.

Especifican relaciones virtuales que no están almacenadas realmente, pero que se pueden formar a partir de los hechos aplicando mecanismos de inferencia basados en las especificaciones de las reglas. La principal diferencia entre las reglas y las vistas es que en las primeras puede haber recursión y por tanto, pueden producir vistas que no es posible definirán términos de las vistas relacionales estándar. Las BDD buscan derivar nuevos conocimientos a partir de datos existentes proporcionando interrelaciones del mundo real en forma de reglas.

Utilizan mecanismos internos para la evaluación y la optimización. Tener la capacidad de expresar consultas por medio de reglas lógicas.

Permitir consultas recursivas y algoritmos eficientes para su evaluación.

Contar con negaciones estratificadas.

Soportar objetos y conjuntos complejos.

Contar con métodos de optimización que garanticen la seguridad en el acceso.

Brinda la posibilidad de inferir información a partir de los datos almacenados, modelando la base de datos como un conjunto de fórmulas lógicas, las cuales permiten deducir otras fórmulas nuevas.

Las principales ventajas al utilizar una BDD son las siguientes.

Almacenamiento de pocos Datos.

Permite crear consultas recursivas con los datos que se encuentran almacenados.

Capacidad de obtener nueva información a través de laya almacenada en la base de datos mediante inferencia.

La explotación de las reglas de deducción en una BDD plantea algunos problemas.

Encontrar criterios que decidan la utilización de una ley como regla de deducción o como regla de coherencia.

Replantear correctamente, en un contexto deductivo, las convenciones habituales en una base de datos (representaciones de informaciones negativas, eficacia de las respuestas a las interrogaciones, cierre del dominio).

Desarrollar procedimientos eficaces de deducción. La posibilidad de caer en bucles infinitos es un problema muy importante.

Las relaciones de una Base de Datos se define:

1. Intención.

2. Extensión.

Para una Base particular, la intención de las relaciones que la constituyen se define por un conjunto de leyes generales, mientras que cada estado de la Base proporciona una extensión (conjunto de tuplas) para cada una delas relaciones. Las tuplas constituyen, de hecho, informaciones elementales.

En un SGBD convencional, todas las leyes generales se explotan para mantener la coherencia de las informaciones elementales; a estas leyes se las denomina entonces restricciones de integridad. Por el contrario, en un Sistema deductivo, algunos (o todas) de estas leyes se utilizan como reglas de deducción para deducir nuevas informaciones elementales a partir de las introducidas explícitamente en la Base.

La explotación de las reglas de deducción en un SGBD plantea algunos problemas:

Encontrar criterios que permitan, para una ley dada decidir su utilización como regla de deducción o como regla de coherencia.

Replantear correctamente, en un contexto deductivo, las convenciones habituales en una base de datos (representaciones de informaciones negativas, eficacia de las respuestas a las interrogaciones, cierre del dominio).

Desarrollar procedimientos eficaces de deducción.

La explotación de las reglas de deducción puede analizarse de dos formas. La primera, consiste en su uso en fase de interrogación, buscando así informaciones deducibles implícitas; Una segunda forma consiste en su uso en fase de modificación, cuando se añaden informaciones deducibles. Según se utilicen en el primer o el segundo modo, las reglas se denominan de derivación o degeneración.

En Datalog (DatabaseLogic) no existen instrucciones de control. Su ejecución se basa en dos conceptos:

1. La unificación.

2. El backtracking.

Datalog selecciona el primer punto de elección y sigue ejecutando el programa hasta determinar si el objetivo es verdadero o falso.

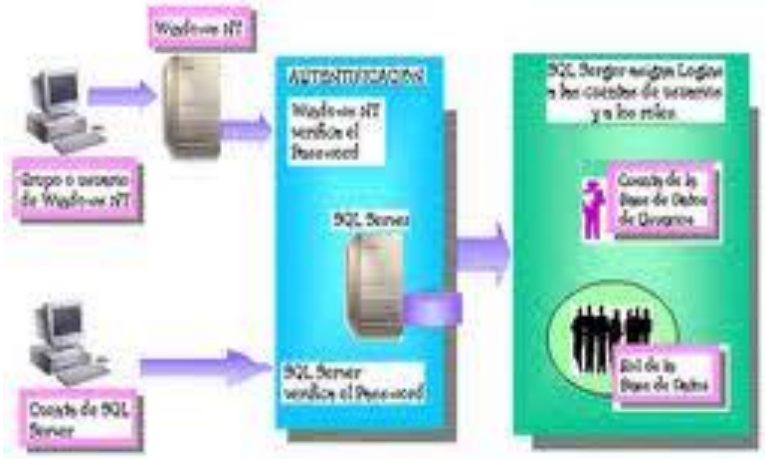
En caso de ser falso entra en juego el backtracking, que consiste en deshacer todo lo ejecutado situando el programa en el mismo estado en el que estaba justo antes de llegar al punto de elección. Entonces se toma el siguiente punto de elección que estaba pendiente y se repite de nuevo el proceso. Todos los objetivos terminan su ejecución bien en éxito ("verdadero"), bien en fracaso ("falso").

Existen principalmente dos tipos de inferencia computacional basados en la interpretación de las reglas por la teoría de la demostración:

Mecanismo de inferencia ascendente. También llamado encadenamiento hacia delante o resolución ascendente. La máquina de inferencia parte de los hechos y aplica las reglas para generar hechos nuevos. Conviene usar una estrategia de búsqueda para generar sólo los hechos que sean pertinentes a una consulta.

Mecanismo de inferencia descendente, También llamado encadenamiento hacia atrás o resolución descendente. Parte del predicado que es el objetivo de la consulta e intenta encontrar coincidencias con las variables que conduzcan a hechos válidos de la base de datos. Retrocede desde el objetivo buscado para

determinar hechos que lo satisfacen. Si no existieran los hechos que buscamos, el sistema entonces buscará la primera regla cuya cabeza (LHS) tenga el mismo nombre de predicado que la consulta.



4 Modelo Base datos Horizontal

La *partición horizontal* implementa sólo las tablas correspondientes a los subtipos, trasladando a las mismas todos los atributos del supertipo. El inconveniente principal de esta transformación es que sólo se pueden representar dos tipos de objetos: estudiantes ó profesores. Los objetos persona no pueden ser representados en este esquema.

DNI	Nombre	Apellido	Domicilio	Ciudad	Fecha_Nac	LU	Espec	Año_Ingreso
1234	Juan	Garcia	abc 12	XYZ	11/11/74	212	ISI	1999

DNI	Nombre	Apellido	Domicilio	Ciudad	Fecha_Nac	Legajo	Cargo	Fecha_Ingreso
6789	José	Perez	lmn 56	LPJ	9/4/60	994	JTP	1/4/1990

Base de datos normalmente usada en:

4.1 Sistemas de Gestión

En lo que se refiere a bases de datos Horizontales estas son adecuadas para aplicaciones en las que el resultado requerido sea de estructura horizontal de búsqueda rápida.

Las bases de datos horizontales son adecuadas para aplicaciones en las que el resultado solicitado es un conjunto de registros horizontales, pero no tanto para aplicaciones tales como la minería de datos, donde los investigadores suelen estar interesados en los resultados que se pueden expresar sucintamente.

Las P-árboles, por otra parte, son muy adecuadas para la minería de datos. Estos son generalmente creados por la descomposición de cada atributo o columna de una tabla de registro horizontal en vectores de bits independientes o estructuras de datos de una matriz. Los P-árboles pueden ser unidimensionales, bidimensionales o multidimensionales y si los datos que se almacenan en la base de datos tienen dimensiones físicas, por ejemplo, datos geoespaciales o información geográfica, las dimensiones de un P-árbol se hacen coincidir para esos datos.

4.2 Minería de datos

Las bases de datos horizontales son adecuadas para aplicaciones en las que el resultado solicitado es un conjunto de registros horizontales, pero no tanto para aplicaciones tales como la minería de datos donde los investigadores suelen estar interesados en los resultados que se pueden expresar sucintamente. Las P-árboles, por otra parte, son muy adecuadas para la minería de datos. Estos son generalmente creados por la descomposición de cada atributo o columna de una tabla de registro horizontal en vectores de bits independientes o estructuras de datos de una matriz. Los P-árboles pueden ser unidimensionales, bidimensionales o multidimensionales y si los datos que se almacenan en la base de datos tienen dimensiones físicas, por ejemplo, datos geoespaciales o información geográfica, las dimensiones de un P-árbol se hacen coincidir para esos datos.

4.3 Partición Horizontal

Esta partición es conveniente cuando en el universo a representar sólo existen objetos que pertenecen a los subtipos, por lo tanto, no existen objetos a nivel de súper tipos y no es necesario mantener una tabla para ellos.

Comparando la implementación relacional con la objeto-relacional, podemos decir que la mayor diferencia entre ambas es el manejo de comportamiento entre una y otra. La definición de las operaciones sobre los tipos que no es posible hacer en una implementación relacional, sí se pueden hacer en una objeto-relacional, con todas las ventajas que esto propone. Esta implementación aventaja a la relacional en las características de flexibilidad, herencia, polimorfismo. Son equivalentes en aspectos relacionados con el acceso a los datos y restricciones de integridad para los atributos.

5.- Modelo Base datos relacional vertical

La *partición vertical* implica que se generará una tabla para cada uno de los tipos que componen la jerarquía, cada una con sus atributos. La clave del supertipo se replica como clave foránea en cada uno de los subtipos.

DNI	Nombre	Apellido	Domicilio	Ciudad	Fecha_Nac
1234	Juan	García	Abc 12	XYZ	11/11/74
6789	José	Perez	Lmn 56	LPJ	9/4/60
4567	Luis	López	Zjk 34	KLO	24/10/65

Restricciones Clave Foránea

DNI	LU	Espec	Año_Ingreso
1234	212	ISI	1999

DNI	Legajo	Cargo	Fecha_Ingreso
6789	994	JTP	1/4/1990

Es aquella en la que la disposición física de los datos se organiza columna a columna en vez de fila a fila. En lugar de estar dispuestos en estructuras de registro horizontal y verticalmente procesados, los datos en una base de datos verticales se disponen en estructuras verticales, conocidos como árboles de predicados, o P-árboles (p-trees), y son procesados horizontalmente

5.1 Partición Vertical

En este modelo es posible analizar tres tipos de implementaciones: relacional, híbrida y objeto-relacional.

La implementación relacional tiene la dificultad de tener que realizar operaciones de join cuando se quiere recuperar la información completa de un estudiante y/o profesor. Esto, dependiendo del tamaño de las tablas y la distribución de las filas en el almacenamiento, puede ser muy costoso. Además tiene definido el

mecanismo de integridad referencial para administrar la integridad de las referencias manejadas por medio de las claves foráneas.

El esquema híbrido tiene una gran flexibilidad al momento de navegar por los datos. Definido de modo bi-direccional, es posible, por ejemplo acceder a un estudiante por medio de su documento de identidad, o su número de libreta, y siguiendo las respectivas referencias recuperar el resto de la información accediendo a la tabla de estudiante y luego a sus datos personales. En este esquema no existe un mecanismo que mantenga la integridad de las referencias como en el caso de las claves foráneas del modelo relacional. Esto implica que si el objeto apuntado por una variable REF se elimina, no hay nada definido por el estándar que diga qué hacer ante este caso. Esta implementación permite el manejo de funciones miembros, aunque no permite la posibilidad de herencia de atributos ni métodos, ni de polimorfismo, que sí los tiene la relación de generalización especialización definida por el estándar SQL: 1999.

La implementación objeto-relacional es la que presenta mayores ventajas porque puede cambiar flexiblemente las relaciones, hereda atributos y métodos, puede redefinir los métodos en los subtipos, puede definir índices para varios puntos de acceso a los datos, tiene la capacidad de definir restricciones sobre todos los atributos.

5.2 Rendimiento

Los datos de una base vertical se procesan a través de rápidos operadores lógicos tales como AND, OR, OR exclusiva y complemento. Por otra parte, mediante la presentación de datos de modo de columna en vez de filas es posible ejecutar consultas o búsquedas en los datos sin acceder a las páginas de un disco duro que no están afectadas por la consulta y por lo tanto, aumentan la velocidad en la

recuperación de los datos. Esta es una consideración importante a la hora de utilizar minería de datos en los repositorios de datos muy grandes.

5.3 Tamaño de la página

Otra ventaja de las bases de datos verticales es que permiten que los datos se almacenen en páginas grandes. Esto significa que un gran número de elementos de datos relevantes puede ser recuperado en una operación de lectura única. Por el contrario, una operación de lectura individual en una base de datos horizontal recupera no solo los elementos de datos relevantes, sino que también los atributos y las columnas que no son relevantes en la consulta en cuestión y favorece los pequeños tamaños de una página.

5. 4 Aplicaciones científicas

Las bases de datos verticales han recibido un renovado interés de la comunidad científica en los últimos años. El número de usuarios simultáneos en las aplicaciones de bases de datos científicas es generalmente mucho menor que en las aplicaciones comerciales, pero los usuarios tienden a presentar consultas más complejas e imprevistas. Además, las aplicaciones de bases de datos científicas generalmente deben dar una respuesta más automatizada para las consultas complejas debido a la ausencia de personal de base de datos y a los sistemas de apoyo. Los usuarios científicos generalmente prefieren trabajar con dedicación, en

el local de sistemas informáticos, las aplicaciones de base de datos para científicos deben ser portables entre distintos modelos de computadoras. Las bases de datos verticales son mejores, en todos estos aspectos, que sus contrapartes horizontales en bases de datos de menor tamaño.

6 Modelo base de datos y sus diferencias

La diferencia entre estructura de modelos de bases de datos vertical es la organización por una columna por cada dato, pero la búsqueda es de modo horizontal y existe una gran diferencia con respecto a él orden horizontal la cual es una línea para toda la datos relacionada y de búsqueda horizontal.

6.1 Forma de uso Base de vertical

Los datos en una base de datos verticales se disponen en estructuras verticales, conocidos como árboles de predicados, o P-árboles (p-trees), y son procesados horizontalmente, por lo tanto, este esquema de uso es dependiente de la forma de acceso horizontal.

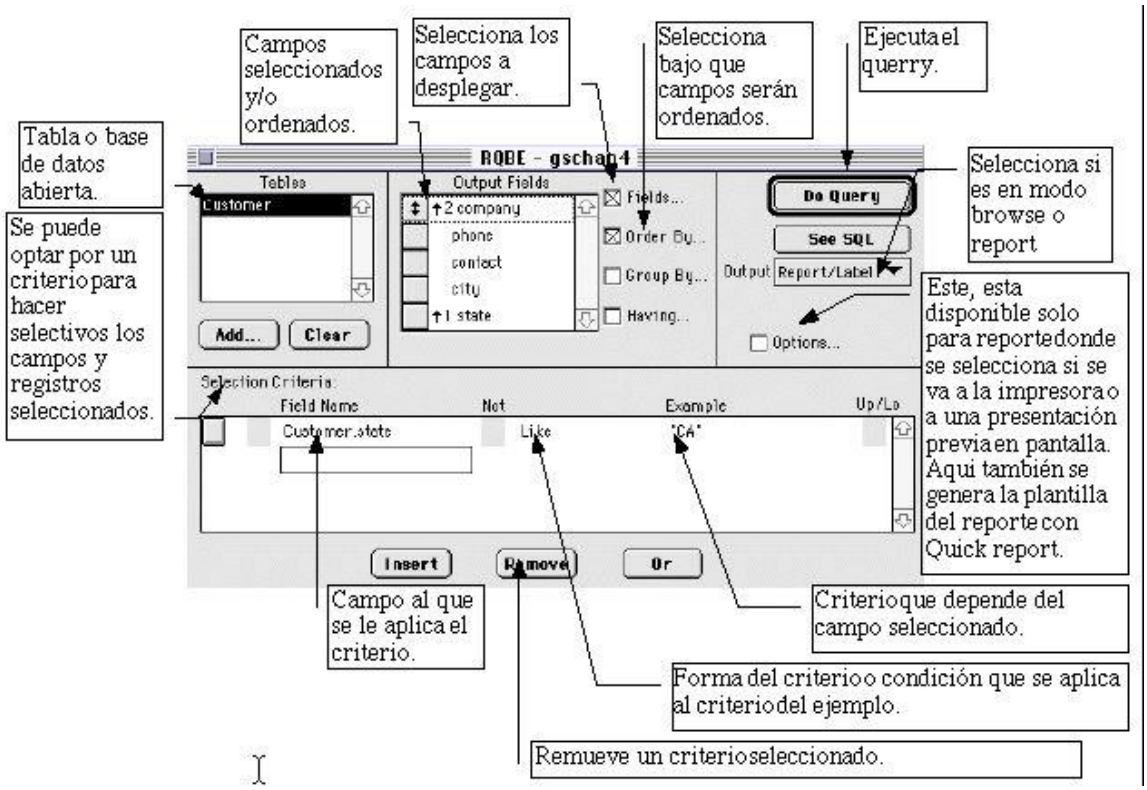
Los datos de una base vertical se procesan a través de rápidos operadores lógicos tales como AND, OR, OR exclusiva y complemento y estos mejoran el rendimiento de eso de la información.

Desde otro punto de vista, mediante la presentación de datos de modo de columna en vez de filas es posible ejecutar consultas o búsquedas en los datos sin incurrir en el acceso a las páginas de un disco duro, que no están afectadas por la consulta y por lo tanto aumentan la velocidad en la recuperación de los datos pero

dependiente de otros elementos que sean lo suficientemente rápidos para el requerimiento ejecutado en esa instancia.

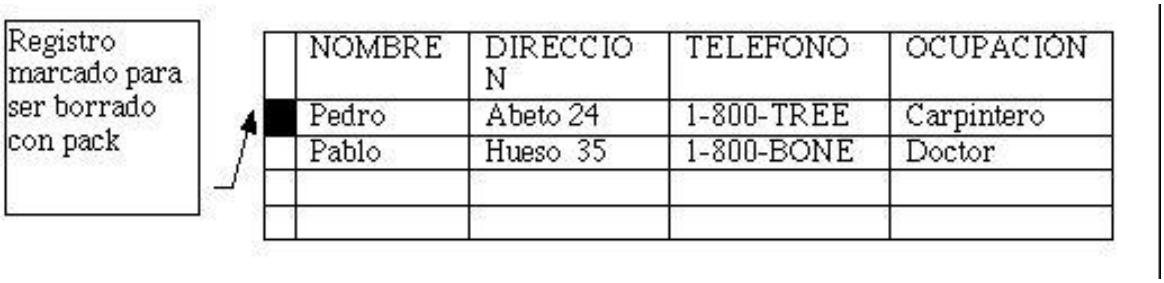
Esta es una consideración importante tener equipamiento acorde a la necesidad planteada a la hora de utilizar minería de datos en los repositorios de datos muy grandes.

Organización Base de datos vertical



6.2 Forma de uso Base de Horizontal

Los datos de una base de datos Horizontal se disponen en forma similar y son procesados en forma Horizontal, es por esto que se puede presentar como una tendencia a mejorar la velocidad de acceso a la información.



	NOMBRE	DIRECCIO N	TELEFONO	OCUPACIÓN
■	Pedro	Abeto 24	1-800-TREE	Carpintero
	Pablo	Hueso 35	1-800-BONE	Doctor

Fragmentación Horizontal:

La fragmentación horizontal se realiza sobre las tuplas de la relación. Cada fragmento será un subconjunto de las tuplas de la relación.

Existen dos variantes de la fragmentación horizontal: **la primaria y la derivada.**

La fragmentación horizontal primaria de una relación se desarrolla empleando los predicados definidos en la misma relación. Por el contrario, la fragmentación

horizontal derivada es el dividir una relación partiendo de los predicados definidos sobre alguna otra.

Información necesaria para la fragmentación horizontal

Información sobre la base de datos.

Esta información implica al esquema conceptual global. Es importante señalar cómo las relaciones de la base de datos se conectan con otras.

En una conexión de relaciones normalmente se denomina relación propietaria a aquella situada en la cola del enlace, mientras que se llama relación miembro a la ubicada en la cabecera del vínculo. Dicho de otra forma podemos pensar en relaciones de origen cuando nos refiramos a las propietarias y relaciones destino cuando lo hagamos con las miembro.

Definiremos dos funciones de relación:

Propietaria y miembro, las cuales proyectarán un conjunto de enlaces sobre un conjunto de relaciones. Además, dado un enlace, devolverán el miembro y el propietario de la relación, respectivamente. La información cuantitativa necesaria gira en torno a la carnalidad de cada relación, notada como card(R).

Información sobre la aplicación.

Necesitaremos tanto información **cualitativa como cuantitativa**. La información cualitativa guiará la fragmentación, mientras que la cuantitativa se necesitará en los modelos de asignación.

La principal información de carácter cualitativo son los predicados empleados en las consultas de usuario. Si no fuese posible investigar todas las aplicaciones para determinar estos predicados, al menos se deberían investigar las más importantes. Podemos pensar en la regla **"80/20"** para guiarnos en nuestro análisis, esta regla dice que el **20% de las consultas existentes acceden al 80% de los datos**. Llegados a este punto, sería interesante determinar los predicados simples.

A parte de los predicados simples, las consultas emplean predicados más complejos resultado de combinaciones lógicas de los simples.

Una combinación especialmente interesante es la conjunción de predicados simples, al predicado resultante se le denomina predicado mintérmino.

Partiendo de que siempre es posible transformar una expresión lógica en su forma normal conjuntiva, usaremos los predicados mintérmino en los algoritmos para no causar ninguna pérdida de generalidad.

Definir conjuntos de datos

Sobre la información cuantitativa necesaria relativa a las aplicaciones, necesitaremos definir dos conjuntos de datos.

Selectividad mintérmino.

Es el número de tuplas de una relación a las que accede una consulta de acuerdo a un predicado mintérmino dado. Notaremos la selectividad de un mintérmino mi como sel(mi).

Frecuencia de acceso.

Es la frecuencia con la que un usuario accede a los datos. Si Q = {q1, q2, ..., qq} es un conjunto de consultas de usuario, acc(qi) indica la frecuencia de acceso a la consulta qi en un periodo dado.

7 Orientación Modelo base de datos Horizontal

La ventaja de las bases de datos verticales es que permiten que los datos se almacenen en páginas grandes. Un tamaño de página grande significa que un gran número de elementos de datos relevantes puede ser recuperado en una operación de lectura única, esto lo hace dependiente de la estructura física en todos sus requerimientos.

Por el contrario, una operación de lectura individual en una base de datos horizontal recupera no solo los elementos de datos relevantes, sino que también los atributos y las columnas que no son relevantes en la consulta en cuestión y favorece los pequeños tamaños de una página y pasa a ser de mayor velocidad en consulta y depende de la estructura de almacenamiento si no de la velocidad de acceso.

7.1 Fragmentación horizontal primaria

Antes de presentar un algoritmo formal que lleve a cabo la fragmentación horizontal, intentaremos explicar de manera intuitiva los procesos de fragmentación horizontal primaria y derivada. La fragmentación horizontal primaria

se define como una operación de selección de las relaciones propietarias del esquema de la base de datos

Ahora definiremos la fragmentación horizontal más formalmente. Un fragmento horizontal Ri de una relación R contiene todas las tuplas de R que satisfacen un predicado mintérmino MI. Por tanto, dado un conjunto de predicados mintérmino M, existen tantos fragmentos horizontales de la relación R como predicados mintérmino. Este conjunto de fragmentos horizontales también se conocen como conjuntos de fragmentos mintérmino. En los párrafos siguientes se asumirá que la definición de fragmentos horizontales se basa en los predicados mintérmino. Además, el primer paso para el algoritmo de fragmentación consiste en establecer un conjunto de predicados con ciertas propiedades.

Un aspecto importante de los predicados simples es su compleción, así como su minimalidad. Un conjunto de predicados simples Pr se dice que es completo si y solo si existe una probabilidad idéntica de acceder por cada aplicación a cualquier par de tuplas pertenecientes a cualquier fragmento mintérmino que se define de acuerdo con Pr. Se puede apreciar como la definición de compleción de un conjunto de predicados simples difiere de la regla de compleción de la fragmentación.

El segundo paso en el proceso de fragmentación primaria consiste en derivar el conjunto de predicados mintérmino que pueden definirse sobre los predicados del conjunto Pr'. Estos predicados mintérmino establecen los fragmentos candidatos para el proceso de asignación. El establecimiento de los predicados mintérmino es trivial; la dificultad radica en el tamaño del conjunto de predicados mintérmino, que puede ser muy grande (de hecho, exponencial respecto al número de predicados simples). En el paso siguiente se presentarán formas de reducir el número de predicados mintérmino necesarios para la fragmentación.

El tercer paso aborda, como ya se ha citado, la eliminación de algunos fragmentos mintérmino que puedan ser redundantes. Esta eliminación se desarrolla

identificando aquellos mintérminos que puedan resultar contradictorios sobre un conjunto de implicaciones.

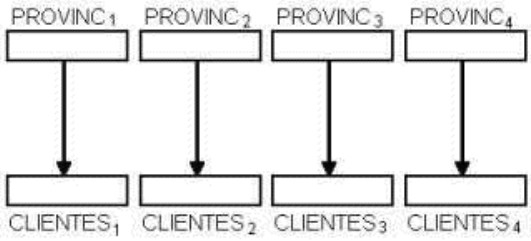
7.2 Fragmentación horizontal derivada

Una fragmentación horizontal derivada se define sobre una relación miembro de acuerdo a una operación de selección especificada sobre su propietaria. Se deben dejar claros dos puntos. Primero, el enlace entre las relaciones propietaria y miembro se define como un equi-yunto. Segundo, un equi-yunto puede desarrollarse a través de semiyuntos. Este segundo punto es especialmente importante para nuestros propósitos, ya que deseamos fraccionar una relación miembro según la fragmentación de su propietaria, además es necesario que el fragmento resultante se defina únicamente sobre los atributos de la relación miembro.

Las tres entradas necesarias para desarrollar la fragmentación horizontal derivada son las siguientes: el conjunto de particiones de la relación propietaria, la relación miembro y el conjunto es indicado como resultados de aplicar el semi-yunto entre la propietaria y la miembro. El algoritmo de fragmentación resulta tan trivial que no se ve la necesidad de entrar en detalles.

Existe una posible complicación que necesita nuestro estudio. En un esquema de base de datos, resulta frecuente que existan más de dos enlaces sobre una relación R. En este caso, aparece más de una posibilidad de fragmentación horizontal derivada. La decisión para elegir una u otra se basa en dos criterios: Uno, la fragmentación con mejores características de yunto. Dos, la fragmentación empleada en más aplicaciones.

Discutamos el segundo criterio primero. Resulta sencillo de establecer si tomamos en consideración la frecuencia con la que cada aplicación accede a los datos. Si es posible, deberíamos intentar facilitar el acceso a los usuarios que hagan mayor uso de los datos para, de esta manera, minimizar el impacto total del rendimiento del sistema.



7.3 Grafo de yuntos entre fragmentos.

Sin embargo, El primer criterio, no es tan sencillo. Considere, por ejemplo, la fragmentación expuesta. El objetivo de esta fragmentación consiste en beneficiar a la consulta que haga uso de las dos relaciones al poder realizarse el yunto de CLIENTES y PROVINC sobre relaciones más pequeñas (es decir, fragmentos), y posibilitar la confección de yuntos de manera distribuida. El primer aspecto resulta obvio. Los fragmentos de CLIENTES son más pequeños que la propia relación CLIENTES. Por tanto, resultará más rápido llevar a cabo el yunto de un fragmento de PROVINC con otro de CLIENTES que trabajar con las propias relaciones. El segundo punto, sin embargo, es más importante ya que es la esencia de las bases de datos distribuidas. Si, además de estar ejecutando un número de consultas en diferentes sitios, podemos ejecutar una consulta en paralelo, se espera que el

tiempo de respuesta del sistema aumente. En el caso de yuntos, esto es posible bajo ciertas circunstancias. Considere, por ejemplo, el grafo de yunto (los enlaces) entre los fragmentos de CLIENTES y la derivada PROVINC. Hay únicamente un enlace entrando o saliendo de un fragmento. De ahí, que se denomine a este grafo, grafo simple. La ventaja de este diseño donde la relación de yunto entre los fragmentos es simple, radica en la asignación a un sitio tanto de la propietaria como de la miembro y los yuntos entre pares diferentes de fragmentos pueden realizarse independientemente y en paralelo. Desgraciadamente, la obtención de grafos de yunto simples no siempre es posible. En tal caso, la mejor alternativa sería realizar un diseño que provoque un grafo de yuntos fragmentados. Un grafo fragmentado consiste en dos o más subgrafos que no están enlazados entre ellos. Por tanto, los fragmentos que se obtengan no se distribuirán para ejecuciones paralelas de un modo tan fácil como aquellos obtenidos a través de grafos simples, pero su asignación aún será posible.

Compleción. La compleción de una fragmentación horizontal primaria se basa en la selección de los predicados a usar. En la medida que los predicados seleccionados sean completos, se garantizará que el resultado de la fragmentación también lo será. Partiendo de la base que el algoritmo de fragmentación es un conjunto de predicados completos y mínimos Pr', se garantiza la compleción siempre que no aparezcan errores al realizar la definición de Pr'. La compleción de una fragmentación horizontal derivada es algo más difícil de definir. La dificultad viene dada por el hecho de que los predicados que intervienen en la fragmentación forman parte de dos relaciones. Definamos la regla de compleción formalmente. Sea R la relación miembro de un enlace cuya propietaria es la relación S, la cual está fragmentada como FS = {S1, S2, ..., Sw}. Además, sea A el atributo de yunto entre R y S. Entonces para cada tupla t de R, existirá una tupla t' de S tal que t[A] = t'[A].

Reconstrucción. La reconstrucción de una relación global a partir de sus fragmentos se desarrolla con el operador de unión tanto para la fragmentación horizontal primaria como para la derivada.

Disyunción. Resulta sencillo establecer la disyunción de la fragmentación tanto para la primaria como para la derivada. En el primer caso, la disyunción se garantiza en la medida en que los predicados mintérmino que determinan la fragmentación son mutuamente exclusivos. En la fragmentación derivada, sin embargo, implica un semi-yunto que añade complejidad al asunto. La disyunción puede garantizarse si el grafo de yunto es simple. Si no es simple, será necesario investigar los valores de las tuplas. En general, no se desea juntar una tupla de una relación miembro con dos o más tuplas de una relación propietaria cuando estas tuplas se encuentran en fragmentos diferentes a los de la propietaria. Esto no es fácil de establecer, e ilustra por qué los esquemas de la fragmentación derivada que generan un grafo de yunto simple son siempre más atractivos.

8 Usos y aplicabilidad de Modelo Base de datos Horizontal versus Vertical

Las bases de datos horizontales son adecuadas para aplicaciones en las que el resultado solicitado es un conjunto de registros horizontales y data de gran dimensión con consultas simultaneas, pero no tanto para aplicaciones tales como la minería de datos donde los investigadores suelen estar interesados en los resultados que se pueden expresar sucintamente.

Las P-árboles, por otra parte, son muy adecuadas para la minería de datos. Estos son generalmente creados por la descomposición de cada atributo o columna de una tabla de registro horizontal en vectores de bits independientes o estructuras de datos de una matriz.

Los P-árboles pueden ser unidimensionales, bidimensionales o multidimensionales y si los datos que se almacenan en la base de datos tienen dimensiones físicas, por ejemplo, datos geoespaciales o información geográfica, las dimensiones de un P-árbol se hacen coincidir para esos datos. Esta tendencia es para modelo de datos de gran cantidad de información

Las bases de datos verticales tienen como fuerte de es trabajar la relación de datos en forma más integral con respecto los modelos y es para datas de menor tamaño en su entorno.

9.1 Tendencia

Uso en modelos de base de datos horizontal y vertical

¿Base de datos Enlatada o a Medida?

Las Enlatadas, son bases cuyos datos pertenecen a los rubros más utilizados por la mayoría de nuestros clientes y estas comúnmente son de estructura vertical las cuales están arraigadas por tipo de estructura orientada a relaciones estándares, básicas y de menor cantidad de datos dentro de esta.

Pero esto no quita que Ud. no necesite ofrecer su producto o servicio a empresas y/o individuos fuera del contenido de las bases enlatadas. Todo lo contrario: Puede designar rubros y/o universos más específicos, partiendo de un modelo de base enlatada. Pero además puede designar rubros, universos, campos y potenciales a la medida de sus requerimientos. Esto se denomina Base de Datos a Medida, esto tiene grandes diferencias porque la orientación de esta propuesta se llevar la propuesta a una realidad de lo que realmente se requiere como motor y base de datos en el entorno actual del requerimiento y todas las potencialidades a rescatar de estos entes presentados para la solución final.

9.2 Demostración de usos de bases de datos Horizontales versus verticales

Información necesaria para la fragmentación horizontal

Antes de realizar los muestra de implementación de estas tendencias, es importante señalar cómo las relaciones de la base de datos se conectan con otras.

La información cuantitativa necesaria gira en torno a la cardinalidad de cada relación, notada como card(R), con esto claro podemos realizar pruebas de fondo para llegar a una buena demostración de uso.

Información sobre la aplicación.

Necesitaremos tanto información cualitativa como cuantitativa. La información cualitativa guiará la fragmentación, mientras que la cuantitativa se necesitará en los modelos de asignación.

La principal información de carácter cualitativo son los predicados empleados en las consultas de usuario. Si no fuese posible investigar todas las aplicaciones para determinar estos predicados, al menos se deberían investigar las más importantes. Podemos pensar en la regla "80/20" para guiarnos en nuestro análisis, esta regla dice que el 20% de las consultas existentes acceden al 80% de los datos.

Llegados a este punto, sería interesante determinar los predicados simples.

Fragmentación horizontal primaria

Un fragmento horizontal Ri de una relación R contiene todas las tuplas de R que satisfacen un predicado mintérmino mi. Por tanto, dado un conjunto de predicados mintérmino M, existen tantos fragmentos horizontales de la relación R como predicados mintérmino.

Este conjunto de fragmentos horizontales también se conocen como conjuntos de fragmentos mintérmino. En los párrafos siguientes se asumirá que la definición de fragmentos horizontales se basa en los predicados mintérmino.

Además, el primer paso para el algoritmo de fragmentación consiste en establecer un conjunto de predicados con ciertas propiedades.

El segundo paso en el proceso de fragmentación primaria consiste en derivar el conjunto de predicados mintérmino que pueden definirse sobre los predicados del conjunto Pr'. Estos predicados mintérmino establecen los fragmentos candidatos para el proceso de asignación.

El establecimiento de los predicados mintérmino es trivial; la dificultad radica en el tamaño del conjunto de predicados mintérmino, que puede ser muy grande (de hecho, exponencial respecto al número de predicados simples). En el paso siguiente se presentarán formas de reducir el número de predicados mintérmino necesarios para la fragmentación.

El tercer paso aborda, como ya se ha citado, la eliminación de algunos fragmentos mintérmino que puedan ser redundantes. Esta eliminación se desarrolla identificando aquellos mintérminos que puedan resultar contradictorios sobre un conjunto de implicaciones.

9.3 Fragmentación horizontal derivada

Una fragmentación horizontal derivada se define sobre una relación miembro de acuerdo a una operación de selección especificada sobre su propietaria.

Se deben dejar claros dos puntos. Primero, el enlace entre las relaciones propietaria y miembro se define como un equi-yunto. Segundo, un equi-yunto puede desarrollarse a través de semiyuntos.

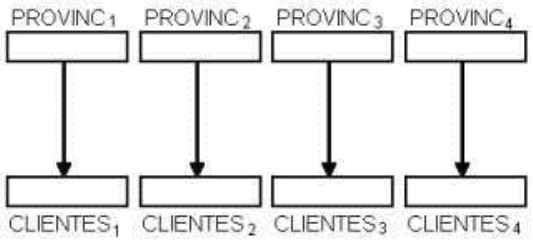
Este segundo punto es especialmente importante para nuestros propósitos, ya que deseamos fraccionar una relación miembro según la fragmentación de su propietaria, además es necesario que el fragmento resultante se defina únicamente sobre los atributos de la relación miembro.

Las tres entradas necesarias para desarrollar la fragmentación horizontal derivada son las siguientes: el conjunto de particiones de la relación propietaria, la relación miembro y el conjunto es predicado, este resultado de aplicar el semi-yunto entre la propietaria y la miembro. El algoritmo de fragmentación resulta tan trivial que no se ve la necesidad de entrar en detalles.

Existe una posible complicación que necesita nuestro estudio. En un esquema de base de datos, resulta frecuente que existan más de dos enlaces sobre una relación R. En este caso, aparece más de una posibilidad de fragmentación horizontal derivada.

La decisión para elegir una u otra se basa en dos criterios: Uno, la fragmentación con mejores características de yunto. Dos, la fragmentación empleada en más aplicaciones.

Discutamos el segundo criterio primero. Resulta sencillo de establecer si tomamos en consideración la frecuencia con la que cada aplicación accede a los datos. Si es posible, deberíamos intentar facilitar el acceso a los usuarios que hagan mayor uso de los datos para, de esta manera, minimizar el impacto total del rendimiento del sistema.



Grafo de yuntos entre fragmentos.

El primer criterio, sin embargo, no es tan sencillo. Considere, por ejemplo, la fragmentación expuesta en el ejemplo. El objetivo de esta fragmentación consiste en beneficiar a la consulta que haga uso de las dos relaciones al poder realizarse el yunto de CLIENTES y PROVINC sobre relaciones más pequeñas (es decir, fragmentos), y posibilitar la confección de yuntos de manera distribuida. El primer aspecto resulta obvio.

Los fragmentos de CLIENTES son más pequeños que la propia relación CLIENTES. Por tanto, resultará más rápido llevar a cabo el yunto de un fragmento de PROVINC con otro de CLIENTES que trabajar con las propias relaciones.

El segundo punto, sin embargo, es más importante ya que es la esencia de las bases de datos distribuidas. Si, además de estar ejecutando un número de consultas en diferentes sitios, podemos ejecutar una consulta en paralelo, se espera que el tiempo de respuesta del sistema aumente. En el caso de yuntos, esto es posible bajo ciertas circunstancias. Considere, por ejemplo, el grafo de yunto (los enlaces) entre los fragmentos de CLIENTES y la derivada PROVINC. Hay únicamente un enlace entrando o saliendo de un fragmento.

De ahí, que se denomine a este grafo, grafo simple. La ventaja de este diseño donde la relación de yunto entre los fragmentos es simple, radica en la asignación a un sitio tanto de la propietaria como de la miembro y los yuntos entre pares diferentes de fragmentos pueden realizarse independientemente y en paralelo. Desgraciadamente, la obtención de grafos de yunto simples no siempre es posible.

En tal caso, la mejor alternativa sería realizar un diseño que provoque un grafo de yuntos fragmentados. Un grafo fragmentado consiste en dos o más subgrafos que no están enlazados entre ellos. Por tanto, los fragmentos que se obtengan no se distribuirán para ejecuciones paralelas de un modo tan fácil como aquellos obtenidos a través de grafos simples, pero su asignación aún será posible.

Procederemos ahora a probar la corrección de los algoritmos presentados con respecto a los tres criterios enunciados anteriormente.

Compleción. La compleción de una fragmentación horizontal primaria se basa en la selección de los predicados a usar. En la medida que los predicados seleccionados sean completos, se garantizará que el resultado de la fragmentación también lo será. Partiendo de la base que el algoritmo de fragmentación es un conjunto de predicados completos y mínimos Pr', se garantiza la compleción siempre que no aparezcan errores al realizar la definición de Pr'. La compleción de una fragmentación horizontal derivada es algo más difícil de definir.

La dificultad viene dada por el hecho de que los predicados que intervienen en la fragmentación forman parte de dos relaciones. Definamos la regla de compleción formalmente. Sea R la relación miembro de un enlace cuya propietaria es la relación S, la cual está fragmentada como FS = {S1, S2, ..., Sw}. Además, sea A el atributo de yunto entre R y S.

Entonces para cada tupla t de R, existirá una tupla t' de S tal que t[A] = t'[A].

Reconstrucción. La reconstrucción de una relación global a partir de sus fragmentos se desarrolla con el operador de unión tanto para la fragmentación horizontal primaria como para la derivada.

Disyunción. Resulta sencillo establecer la disyunción de la fragmentación tanto para la primaria como para la derivada. En el primer caso, la disyunción se garantiza en la medida en que los predicados mintérmino que determinan la fragmentación son mutuamente exclusivos. En la fragmentación derivada, sin embargo, implica un semi-yunto que añade complejidad al asunto.

La disyunción puede garantizarse si el grafo de yunto es simple. Si no es simple, será necesario investigar los valores de las tuplas. En general, no se desea juntar una tupla de una relación miembro con dos o más tuplas de una relación propietaria cuando estas tuplas se encuentran en fragmentos diferentes a los de la

propietaria. Esto no es fácil de establecer, e ilustra por qué los esquemas de la fragmentación derivada que generan un grafo de yunto simple son siempre más atractivos.

9.4 Fragmentación Vertical:

Recuérdese que la fragmentación vertical de una relación R produce una serie de fragmentos R1, R2, ..., Rr, cada uno de los cuales contiene un subconjunto de los atributos de R así como la clave primaria de R. El objetivo de la fragmentación vertical consiste en dividir la relación en un conjunto de relaciones más pequeñas tal que algunas de las aplicaciones de usuario sólo hagan uso de un fragmento.

Sobre este marco, una fragmentación óptima es aquella que produce un esquema de división que minimiza el tiempo de ejecución de las aplicaciones que emplean esos fragmentos.

La partición vertical resulta más complicada que la horizontal. Esto se debe al aumento del número total de alternativas que tenemos disponibles.

Por ejemplo, en la partición horizontal, si el número total de predicados simples de Pr es n, existen 2n predicados mintérminos posibles que puedan definirse. Además, sabemos que algunos de estos predicados resultarán contradictorios con algunas de las aplicaciones existentes, por lo que podremos reducir el número inicial.

En el caso vertical, si una relación tiene m atributos clave no primarios, el número de posibles fragmentos es igual a B(m), es decir el m-ésimo número de Bell [3]. Para valores grandes de m, B(m) mm; por ejemplo, para m = 10, B(m) 115.000, para m = 15, B(m) 109, para m = 30, B(m) = 1023.

Estos valores indican que la obtención de una solución óptima de la fragmentación vertical resultará una tarea inútil, sino nos apoyamos en el uso

de heurísticos. Existen dos enfoques heurísticos para la fragmentación vertical de relaciones:

Agrupación. Comienza asignando cada atributo a un fragmento, y en cada paso, junta algunos de los fragmentos hasta que satisface un determinado criterio. La agrupación sugirió en principio para bases de datos centralizadas y se usó posteriormente para las bases de datos distribuidas.

Escisión. A partir de la relación se deciden que fragmentos resultan mejores, basándose en las características de acceso de las aplicaciones a los atributos. Esta técnica se presentó, también, para bases de datos centralizadas. Posteriormente, se extendió al entorno distribuido.

En este documento se tratará únicamente la técnica de escisión, ya que es más apropiada para la estrategia descendente y porque resulta más probable encontrar la solución para la relación entera que a partir de un conjunto de fragmentos con un único atributo. Además, la escisión genera fragmentos no solapados mientras que la agrupación normalmente produce fragmentos solapados.

Dentro del contexto de los sistemas de bases de datos distribuidos, son preferibles los fragmentos no solapados por razones obvias. Evidentemente, los fragmentos no solapados se refieren únicamente a atributos clave no primarios.

Antes de comenzar, vamos a aclarar un problema: la réplica de las claves de la relación en los fragmentos.

Esta es una característica de la fragmentación vertical que permite la reconstrucción de la relación global. Por tanto, la escisión considera únicamente aquellos atributos que no son parte de la clave primaria. La réplica de los atributos clave supone una gran ventaja, a pesar de los problemas que pueda causar.

La ventaja está relacionada con el esfuerzo para mantener la integridad semántica.

Tenga en cuenta que cada dependencia (funcional, multivaluada ...) es, de hecho, una restricción que influye sobre el valor de los atributos de las respectivas relaciones en todo momento.

También muchas de estas dependencias implican a los atributos clave de una relación. Si queremos diseñar una base de datos tal que los atributos clave sean parte de una fragmento que está ubicado en un sitio, y los atributos relacionados sean parte de otro fragmento asignado a un segundo sitio, cada petición de actualización provocará la verificación de integridad que necesitará de una comunicación entre esos sitios.

La réplica de los atributos clave de cada fragmento reduce esta problemática, pero no elimina toda su complejidad, ya que la comunicación puede ser necesaria para las restricciones de integridad que implican a las claves primarias, así como para el control de concurrencia.

Una posible alternativa a la réplica de los atributos clave es el empleo de identificadores de tuplas, que son valores únicos asignados por el sistema a las tuplas de una relación. Mientras el sistema mantenga los identificadores, los fragmentos permanecerán disjuntos.

9.5 Información necesaria para la fragmentación vertical

La principal información que necesitaremos se referirá a las aplicaciones.

Por tanto, este punto tratará de especificar la información que de una aplicación que funciona sobre la base de datos podamos extraer. Teniendo en cuenta que la fragmentación vertical coloca en un fragmento aquellos atributos a los que se accede de manera simultánea, necesitaremos alguna medida que defina con más precisión el concepto de simultaneidad.

El principal dato necesario relativo a las aplicaciones es la frecuencia de acceso. Sea Q = {q1, q2, ..., qq} el conjunto de consultas de usuario (aplicaciones) que funcionan sobre una relación R(A1, A2, ..., An).

Los vectores uso pueden definirse muy fácilmente para cada aplicación siempre que el diseñador conozca las aplicaciones existentes en el sistema.

La regla 80/20 expuesta páginas atrás podría resultar útil para el desarrollo de esta tarea.

Los valores del uso de los atributos en general no son suficientes para desarrollar la base de la escisión y la fragmentación de los atributos, ya que estos valores no representan el peso de las frecuencias de la aplicación.

La dimensión de esta frecuencia puede incluirse en la definición de la medida de los atributos afines afd(Ai, Aj), la cual mide el límite entre dos atributos de una relación de acuerdo a cómo las aplicaciones acceden a ellos.

10 Demostración de usos

10.1 Aplicabilidad Horizontal

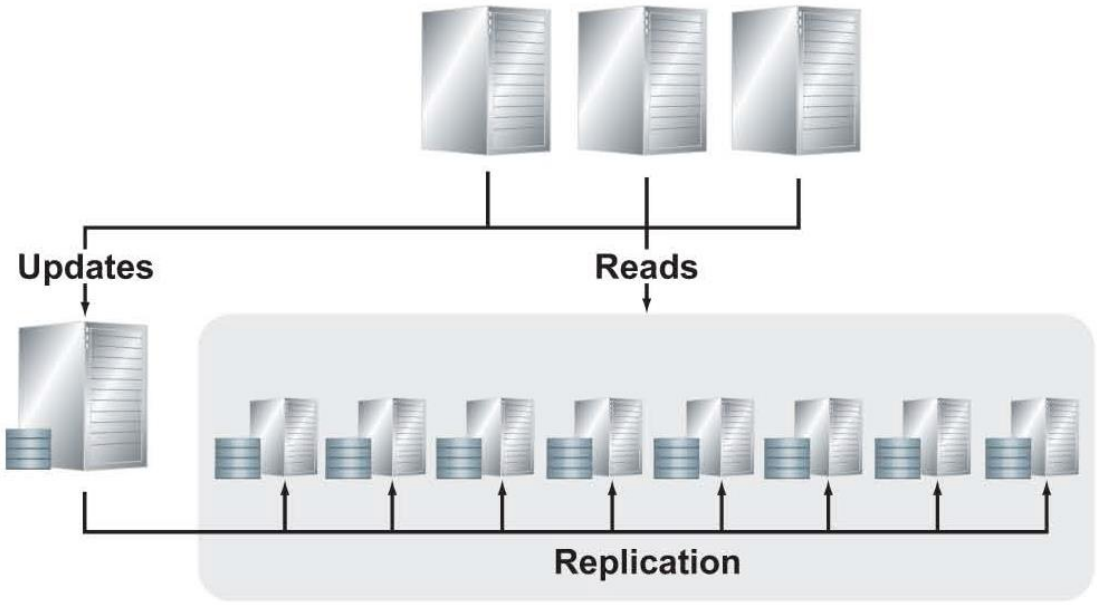
La forma de usar este modelo es hacer referencia a la capacidad de servir un número cada vez mayor de peticiones contra la base de datos.

El tiempo específico de ejecución de una petición una vez insertado en la base de datos, se ve afectado por muchos factores como la lógica de aplicaciones y la potencia de la CPU, cantidad de memoria e interconexiones de E/S subyacentes.

El tiempo ha demostrado que las aplicaciones se hacen populares y la capacidad de una instancia SQL llega a no ser suficiente para responder a las necesidades.

En el entorno de las bases de datos, el término Data Horizontal tiene referencias a la mejora en rendimiento y escalabilidad de las aplicaciones de forma incremental, a medida que se genera la propia demanda, se añaden servicios y servidores de datos para trabaja entre sí ya sea en la dato como en los servicios.

Modelos de acceso a los servicios En forma Horizontal



10.2 Escalada Horizontal para operaciones de lectura

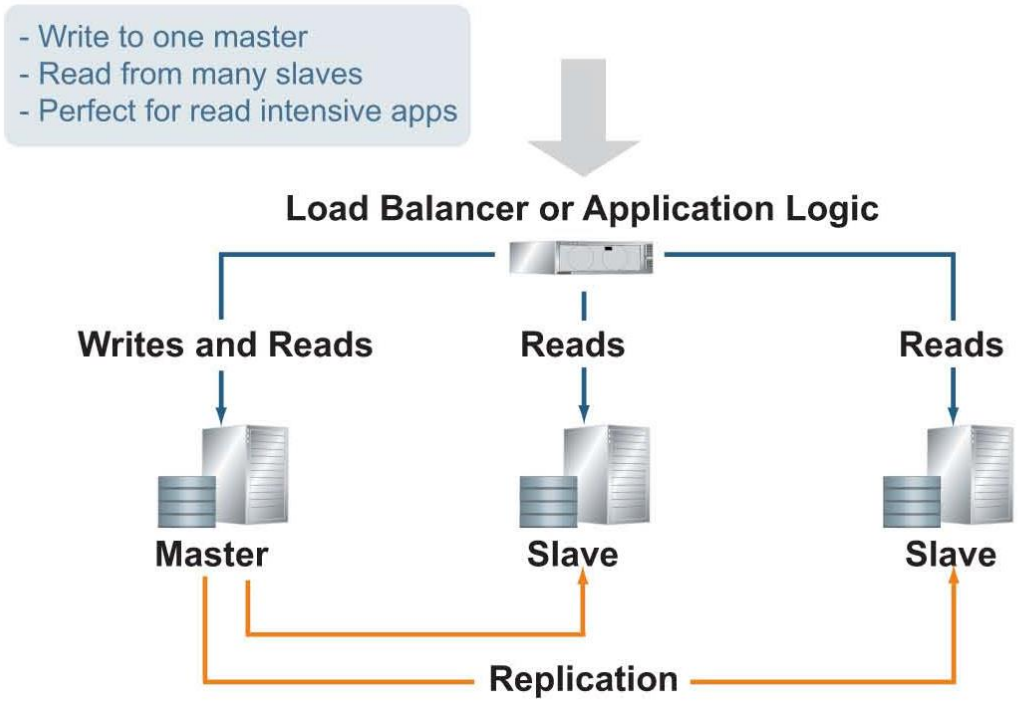
El uso más habitual de las bases de datos es para server, la información almacenada en ellas en respuesta a las peticiones de los usuarios. Un problema muy común en las bases de datos, a medida que crece su volumen es que se deben ser capaces de realizar muchas más operaciones de lectura que de escritura.

Esta situación se presenta cuando la aplicación que accede a la base de datos se convierte en popular y sobre pasa la capacidad del hardware que se aprovisionó originalmente.

EL problema, por lo tanto, es el de ser capaz de soportar cada vez más operaciones de lectura, con solo un pequeño aumento en las operaciones que modifique el contenido de las bases de datos.

Este planteamiento es conocido como red scaling, o escalada de lectura y es la más común de las formas de replicación.

Modelos de acceso a los DATOS En forma Horizontal



CONCLUSIONES

Se han mostrado diversas implementaciones de relaciones de generalización-especialización sobre un ORDBMS. Las implementaciones se estudiaron sobre un modelo relativamente simple pero de gran tamaño, que no por ello pierde generalidad. Las experiencias se realizaron sobre una base de datos objeto-relacional (vertical) pensado en el mejor comportamiento de si es comparativamente con una de orden horizontal.

Se analizaron tres posibles transformaciones: **modelo plano**, **partición vertical** y **partición horizontal**, bajo un esquema relacional y bajo un esquema objeto-relacional que obedece al estándar SQL: 1999. **Para la partición vertical se incluyó también un esquema híbrido, utilizando la capacidad de definir referencias de la tecnología relacional**. Se tuvieron en cuenta las siguientes características: facilidad de implementación, flexibilidad para incorporar nuevos objetos a las estructuras de almacenamiento, mecanismos de acceso y restricciones, posibilidad de manejar comportamiento de los objetos (funciones miembros de los objetos) y facilidad para navegar por los datos.

En principio, no existe una transformación que sea universal y que se adapte a los requerimientos de todos los sistemas que puedan presentarse a quien diseña las aplicaciones. Pero en líneas generales, se puede concluir que, obviamente las transformaciones que obedecen al estándar SQL: 1999 están basadas en conceptos orientados a objetos, que intentan dar una respuesta a la complejidad de las relaciones que se deben abordar en los sistemas de información en nuestros días.

Sus fortalezas más destacables son la capacidad de definir comportamiento, herencia de atributos y métodos, polimorfismo, una mayor flexibilidad para introducir cambios en los modelos originales.

Por su parte, la tecnología relacional presenta características muy arraigadas, que no se pueden considerar obsoletas. El manejo de restricciones sobre los atributos, la capacidad para definir caminos de acceso que aceleren la recuperación de información, es muy importante para algunas aplicaciones que son de menor tamaño.

El **esquema híbrido de partición horizontal** presentado se destaca por la facilidad de navegar por los datos en varias direcciones, facilitando el acceso a los mismos. Su debilidad es la falta de mecanismos que ayuden a mantener la integridad de las referencias, pero realizando un buen esquema de manejo de data se puede llegar a controlar de mejor forma la información, este esquema es el más aconsejable para data de gran tamaño y en la actualidad el mayor problema de todo la información es exactamente la Big data.

En base a este análisis se puede definir que la tendencia de la forma de tratar la información es en esta modalidad porque cada día estamos generando más información y el sostener esto en la forma y en el fondo es colocar este esquema en todos los entes posibles de almacenamiento futuros.

Consideraciones:

A lo largo de este documento se ha intentado dar una visión global y genérica de los problemas y características que contiene el diseño de una base de datos.

Se ha hecho especial hincapié en las técnicas de fragmentación horizontal y vertical a través de métodos y algoritmos muy frecuentes en la literatura referida al tema.

Se espera que el lector no haya tenido demasiados problemas para su comprensión, las técnicas son sencillas y se ha procurado incluir distintos ejemplos para facilitar el entendimiento. Igualmente, la puesta en práctica de los algoritmos, es decir, su codificación, no es un proceso complicado si se poseen nociones en el desarrollo de algoritmos.

Piense, por ejemplo, que los algoritmos de partición vertical, no hacen más que manipular matrices y en el caso de fragmentación horizontal en forma similar.

También debería tenerse presente la existencia de enfoques de fragmentación distintos y, posiblemente, más complejos, pero se debe pensar que son más eficientes para los requerimientos actuales de las big data.

11 GLOSARIO

Motor de base datos

Sistema propietario donde se generan herramientas de uso para almacenar y procesar información

Modelo de base de datos

La forma de distribuir la información en un recipiente de datos

Broadeterm

Término Genérico de un Concepto específico

Narrowerterm

Termino específico de un Concepto

EuroVoc

La relación jerárquica se establece a partir de situaciones lógicas.

Diagramas de Bachman

Describe gráficamente el esquema lógico de una base de datos jerárquica. - Los rectángulos representan los tipos de registro

commit

Marca el final de una transacción correcta

Rollback

Revierte una transacción explícita o implícita

SGBD

Sistema de gestión de bases de datos

tupla

Una secuencia ordenada de objetos

deadlocks

Un interbloqueo se produce cuando dos o más procesos están esperando en el mismo cada proceso de recursos

Html

Tabla para referencia rápida de caracteres y símbolos ascii

Mintérmino

Obtener a partir de la tabla de verdad las expresiones Maxtérminos y Mintérminos en su forma canónica

Subgrafos

En teoría dos grafos, Un subgrafo de un grafo

equi-yunto

Enlace entre las relaciones propietarias

Minimalidad

Registro mínimo requiere que la tabla de destino cumple las condiciones

join

La sentencia en SQL permite combinar registros de dos o más tablas

Backtracking

Vuelta atrás

12 Bibliografía

Biblioteca de Microsoft SQL Server - MSDN

SQL Server 2012 - SQL, Transact SQL

MySQL | The Most Popular Open-Source Database | Oracle

MySQL Workbench: Data Modeling & Development (Oracle Press)

Libro sobre MySQL: La Biblia MySQL

Teradata Spain Global Information

Teradata 12 Certification Study Guide

Búsqueda y Análisis particionada de información