Base de Datos Orientado a Columnas

Adrián Garcete
Ingeniería Informática
Matricula: 057546

Universidad Católica
"Nuestra Señora de la Asunción"
Asunción - Paraguay

Abstract. Este paper presenta, como su nombre lo indica, las bases de datos que están organizadas por columnas. Las Bases de Datos Orientadas a Columnas son sistemas de bases de datos que tienen la característica de almacenar los datos en forma de columna. La ventaja principal de este tipo de sistema es que permite el acceso a grandes volúmenes de datos de forma rápida porque se puede acceder como una unidad a los datos de un atributo particular en una tabla. Un SMBD orientada a columnas es un sistema de gestión de bases de datos que almacena su contenido por columnas (atributos) y no por filas (registros) como lo hacen los Sistemas de gestión de bases de datos relacionales. Veremos como esta implementado, las ventajas y desventajas de esta estructura columnar así como también las técnicas de compresión, indexación y paralelización.

1 Introducción

Las Bases de Datos Columnares se introdujeron por primera vez en 1970 en productos como Model 204 y ABABAS, este enfoque ha resurgido recientemente en Vertica y en cierta medida en QD Technology.
Como su nombre lo indica, las bases de datos están organizados de columna por columna en lugar de la fila: es decir, todos los casos de un solo elemento de datos (por ejemplo, Nombre de cliente) se almacenan de modo que se puede acceder

como una unidad. Esto los hace especialmente eficaz en las consultas analíticas, como la lista de selecciones, que a menudo lee unos pocos elementos de datos, pero necesitamos ver todas las instancias de estos elementos. En contraste, una convencional base de datos relacional almacena los datos por filas, por lo que toda la información de un registro (fila) es inmediatamente accesible. Esto tiene sentido para las consultas transaccionales, que suelen referirse a un registro a la vez.
Cada columna es almacenada contiguamente en un lugar separado en disco, usando generalmente unidades de lectura grandes para facilitar el trabajo al buscar varias columnas en disco. Para mejorar la eficiencia de lectura, los valores se empaquetan de forma densa usando esquemas de compresión ligera cuando es posible. Los operadores de lectura de columnas se diferencian de los comunes (de filas) en que son responsables de traducir las posiones de los valores en locaciones de disco y de combinar y reconstruir, si es necesario, tuplas de diferentes columnas.
Con este cambio ganamos mucha velocidad en lecturas, ya que si se requiere consultar un número reducido de columnas, es muy rápido hacerlo pero no es eficiente para realizar escrituras. Por ello este tipo de soluciones es usado en aplicaciones con un índice bajo de escrituras pero muchas lecturas. Típicamente en data warehouses y sistemas de inteligencia de negocios, donde además resultan ideales para calcular datos agregados. Cabe resaltar que parte del auge actual que está provocando NoSQL se debe a la adopcin de Cassandra (originalmente desarrollada por y para Facebook, luego donada a la fundacin Apache) por parte de Twitter y Digg. Apache Cassandra es la base de datos orientada a columnas más conocida y utilizada actualmente.

2 Características

2.1 Tiempo de carga

Cuánto tiempo se necesita para convertir datos de origen en el formato de columna? Esta es la pregunta más básica de todas. Tiempos de carga son a menudo medidos en gigabytes por hora, que puede ser extremadamente lento, cuando de decenas o cientos de gigabytes de datos se trata. La cuestión a menudo carece de una respuesta sencilla, porque la velocidad de carga puede variar en función de la naturaleza de los datos y las elecciones realizadas por el usuario. Por ejemplo, algunos sistemas pueden almacenar varias versiones de los mismos datos, ordenados en diferentes secuencias o en los diferentes niveles de agregación. Los usuarios pueden construir un menor número de versiones a cambio de una carga rápida, pero puede pagar un precio más adelante con consultas más lentas. Pruebas realistas basadas en sus propios datos son el mejor camino para una respuesta clara.

2.2 Carga Incremental

Una vez que un conjunto de datos se ha cargado, todo debe ser recargado cada vez que hay una actualización. Muchos sistemas columnares permiten carga incremental, teniendo sólo los registros nuevos o modificados y la fusión de los datos anteriores. Pero la atención al detalle es fundamental, ya que las funciones de carga incremental varían ampliamente. Algunas cargas incrementales tardan hasta una completa reconstrucción y algunos resultados son el rendimiento más lento, algunos pueden agregar registros, pero no cambiar o suprimirlos. Las Cargas incrementales a menudo deben completarse periódicamente con una reconstrucción completa.

2.3 Compresión de datos

Algunos sistemas columnares pueden comprimir mucho la fuente de datos y archivos resultantes a fin de tomar una fracción de espacio en el disco original. Puede ocasionar en estos casos un impacto negativo en el rendimiento por la descompresión de datos a realizar la lectura. Otros sistemas utilizan menos compresión o almacenan varias versiones de los datos comprimidos, teniendo más espacio en disco, pero cobrando otros beneficios a cambio. El enfoque más adecuado dependerá de sus circunstancias. Tenga en cuenta que la diferencia de los requisitos de hardware pueden ser sustanciales.

2.4 Limitaciones estructurales

Las bases de datos columnares utilizan diferentes técnicas para imitar una estructura relacional. Algunos requieren la misma clave principal en todas las tablas, es decir, la jerarquía de la base de datos está limitada a dos niveles. Los límites impuestos por un sistema en particular no parece tener importancia, pero recuerde que sus necesidades pueden cambiar mañana. Limitaciones que parece aceptable ahora podra evitar que la ampliación del sistema en el futuro.

2.5 Técnicas de acceso

Algunas bases de datos de columnares sólo se pueden acceder utilizando su propio proveedor de lenguaje de consultas y herramientas. Estos pueden ser muy poderosos, incluyendo capacidades que son difíciles o imposibles usando el estándar SQL. Pero a veces faltan funciones especiales, tales como las consultas que comparan valores con o en los registros. Si necesita acceder al sistema con herramientas basadas en SQL, determine exactamente qué funciones SQL y dialectos son compatibles. Es casi siempre un subconjunto completo de SQL y, en particular, rara vez se dispone de las actualizaciones. También asegúrese de encontrar si el rendimiento de las consultas SQL es comparable a los resultados con el sistema de la propia herramienta de consulta. A veces, el ejecutar consultas SQL mucho más lento.

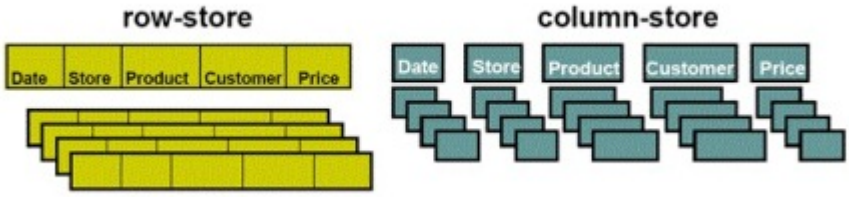
2.6 Rendimiento

Los sistemas columnares por lo general superan a los sistemas de relaciones en casi todas las circunstancias, pero el margen puede variar ampliamente. Las consultas que incluyen cálculos o acceso individual a los registros puede ser tan lento o más que un sistema relacional adecuadamente indexado.

2.7 Escalabilidad

El punto de las bases de datos columnares es obtener buenos resultados en grandes bases de datos. Pero no puede asumir todos los sistemas pueden escalar a decenas o centenares de terabytes. Por ejemplo, el rendimiento puede depender de determinados índices de carga en la memoria, de modo que su equipo debe tener memoria suficiente para hacer esto. Como siempre, en primer lugar preguntar si el vendedor tiene en ejecución los sistemas existentes a una escala similar a la suya y hablar con las referencias para obtener los detalles. Si el suyo sería más grande que cualquiera de las instalaciones existentes, asegúrese de probar antes de comprar.

3 Column-Oriented vs Row-Oriented



La base de datos orientada a filas debe leer toda la fila con el fin de acceder a los atributos necesarios. Como resultado, las consultas analíticas y de inteligencia de negocios terminan leyendo más datos de lo necesario para satisfacer su consulta. Además este tipo de bases de datos habiendo sida diseãda para actividades transaccionales, es a menudo construida para la recuperación óptima y unión de conjunto de datos pequeños en lugar de grandes, cargando asi los subsistemas de entrada y salida que soportan el almacenamiento analítico. En respuesta, los administradores de base de datos tratan de ajustar el entorno de las diferentes consultas mediante la construcción de índices adicionales asi como la creación de vistas especiales. Esto requiere mayor tiempo de procesamiento y

consumo adicional de almacenamiento de datos.
Debido a que cada columna puede ser almacenado por separada, para cualquier consulta, el sistema puede evaluar las columnas que se están accediendo y recuperar sólo los valores solicitados en las columnas específicas. En lugar de exigir los índices separados para las consultas de forma óptima los datos se valora dentro de cada forma de columna del índice, reduciendo los sistemas de entrada y salida lo que permite un acceso rápido a los datos mejorando el tiempo y el rendimiento de las consultas.

Ventajas

1. La principal ventaja de este tipo de sistemas es el rápido acceso a los datos: esto ya lo hemos demostrado con el modelo DSM el cual nos permite consultar rápidamente los datos columna a columna, al guardarse físicamente de manera contigua.
2. Un BBMS en una base de datos orientada a columnas, lee solo los valores de columnas necesarios para el procesamiento de una consulta determinada por lo cual las bases de datos orientadas a columnas tienen una mayor eficiencia en entornos de almacenes, donde las consultas, típicas incluyen los agregados realizados por un gran número de elementos de datos
3. Se comprime la información asignable de cada columna con el fin de mejorar el procesamiento desde el ancho de banda del acceso a disco
4. Cambios en el esquema tiene menor impacto y por lo tanto el coste de realizarlos es menor

Desventajas

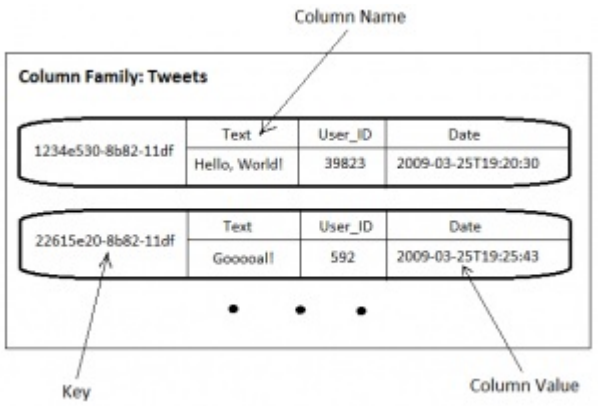
1. No orientado a transacciones: este es el factor más débil de esta tecnología. El hecho de tener los datos guardados columna a columna nos permite retornarnos las filas más rápidamente, pero al insertar, actualizar o borrar un registro, se deberá hacer en más de una ubicación (al tener que actualizar todos los pares clave-valor asociados a una relación). Por esta razón, este tipo de bases de datos no se recomienda para sistemas de tipo OLTP orientados a transacciones y alta concurrencia.
2. Reportes operacionales: también llamados reportes de seguimiento en los que se desea ver toda la información de una relación que puede contener muchas tuplas. En algunos casos esto puede resultar ineficiente comparado con los Row-Stores
3. No existe un modelo de datos que soporte teóricamente este modelo de base de datos
4. No existe un estándar que unifique los criterios de implementación de este modelo de base de datos.

4 Principales Base de Datos Columnares

4.1 APACHE CASSANDRA

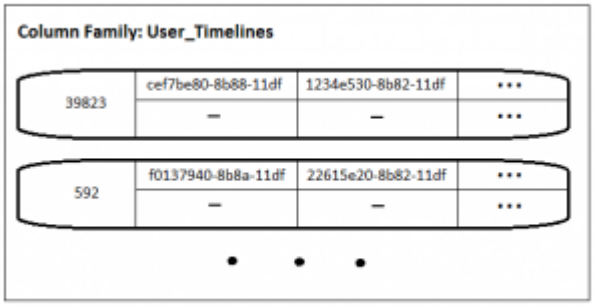
Apache Cassandra es una base de datos no relacional distribuida y basada en un modelo de almacenamiento de clave-valor, escrita en Java. Permite grandes vólumenes de datos en forma distribuida. Por ejemplo, lo usa Twitter para su plataforma. Su objetivo principal es la escalabilidad lineal y la disponibilidad. La arquitectura distribuida de Cassandra está basada en una serie de nodos iguales que se comunican con un protocolo P2P con lo que la redundancia es máxima.
Cassandra es desarrollada por la Apache Software Foundation.
En las versiones iniciales utilizaba un API propia para poder acceder a la base de datos. En los últimos tiempos están apostando por un lenguaje denominado CQL (Cassandra Query Language) que posee una sintaxis similar a SQL aunque con muchas menos funcionalidades. Esto hace que iniciarse en el uso de la misma sea más sencillo. Permite acceder en Java desde JDBC.

Caso Twitter.
La Columna de la familia Tweets contiene registros que representan tweets. La clave de un registro es de tipo Hora y UUID generado cuando el tweet se recibe (vamos a utilizar esta característica en las familias de columnas User_Timelines siguientes). Los registros se componen de columnas. Las columnas representan simplemente atributos de tweets. Así que es muy similar a cómo se podría almacenar en una base de datos relacional.

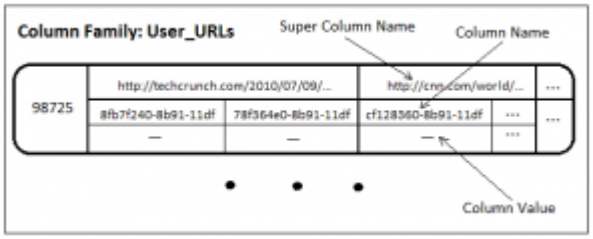


El siguiente ejemplo es User_Timelines (es decir, mensajes de twitter Publicado por un usuario). Los registros están codificados por los ID de usuario (referenciado por columnas User_ID en la familia de columna Tweets). User_Timelines demuestra cómo los nombres de columna se pueden utilizar para almacenar los valores de los identificadores de tweet para este caso. El tipo de nombres de columna se define como UUID Time. Esto significa que los ID de los tweets se

mantienen ordenadas por el momento de su publicación. Esto es muy útil, ya que por lo general quieren mostrar los últimos N tweets para un usuario. Los valores de todas las columnas se establece en una matriz de bytes vaca (denotado "-"), ya que no se utilizan.



Para demostrar columnas súper supongamos que queremos recoger información sobre URLs enviados por cada usuario. Para eso necesitamos agrupar todos los tweets publicados por un usuario de URLs contenidas en los tweets. Se puede guardar utilizando súper columnas como sigue:



En User_URLs los nombres de las columnas súper se utilizan para almacenar las direcciones URL y los nombres de las columnas anidadas son los correspondientes tweets IDs.

4.2 PROJECT GEMINI

Ese es el nombre que recibe la interesante propuesta que nos hacen desde Microsoft, para renovar su base de datos OLAP. Se trata de un almacenamiento en

memoria y orientado a columnas de Analysis Services para ser explotado desde Excel, entre otras opciones. Se trata de empezar a jugar fuerte en el campo del BI, aprovechando la herramienta basada en la popular hoja de cálculo que se incluye dentro del portfolio ofrecido actualmente por Microsoft.
De esta forma se consigue que los usuarios no técnicos puedan acceder al mundo del BI (sin necesidad de contar con IT). Una versión Wii para el BI.
Nigel Pendse, del Olap Report, hace una revisión bastante positiva, indicando que el Project Gemini es como un caballo de Troya, donde a través del uso de Excel (conocido por todos), se quita la complejidad de cubos, MDX, etc. y se democratiza su uso y el de AS a todos.

4.3 INFOBRIGHT

Infobright combina una base de datos orientada a la columna con la red de conocimiento para ofrecer una arquitectura de auto-gestión de Data Warehouse optimizado para el análisis. Este software sofisticado elimina el tiempo y el esfuerzo que suelen participar en la ejecución y la gestión de un Data Warehouse, liberando su tiempo y su presupuesto.
Infobright Analytic Data Warehouse est basado en los siguientes conceptos:

- Orientación a Columnas.
- Paquetes de Datos.
- Conocimiento de Red.
- La Optimización.

Infobright es, en su núcleo, es un comprimido de bases de datos orientadas a la columna. Esto significa que en lugar de los datos que se almacena la fila por fila, sino que se almacena la columna por columna. Infobright, organiza cada columna en paquetes de datos, tiene más compresión que otras bases de datos orientadas a la columna, ya que se aplica un algoritmo de compresión basado en el contenido de cada paquete de datos, no sólo cada columna.

La mayoría de las consultas sólo implican un subconjunto de las columnas de las tablas y por lo que una base de datos orientada a la columna se centra solo en recuperar los datos que se requieren.

4.4 VERTICA

Vertica es el único DBMS habilitado para gestionar terabytes de datos ms rápido y ms fiable que cualquier otro producto de almacenamiento de datos. Obtiene rápidamente BI con las siguientes características:

- Orientación a columnas. 50x - 200x más rápido, eliminando los costos de IO.
- Escala a arquitectura MPP. Escala ilimitadamente solo por la adición de nuevos servidores a la red.
- Agresiva compresin de datos. Reduce los costos de almacenamiento hasta en un 90%.
- Alta disponibilidad inmediata. Corre sin parar con replicación automática, resistente a fallos y recuperación.

Fact	
Vertica is faster	✓
Vertica is less costly	✓
Vertica runs on commodity servers	✓
Vertica scales on less hardware	✓
Vertica is cloud and virtualization friendly	✓
You can install and start using Vertica today...	✓

Vertica cambia completamente la economía de la BI, que permite rápidamente iniciar un espectro mucho más amplio de análisis del negocio:

- Ver mucho mayores volúmenes de datos históricos.
- Analizar los datos en cualquier nivel de detalle.
- Realizar análisis en tiempo real.
- Conducta ad-hoc y de corta duración de análisis de proyectos de negocios.
- Construir Análisis de Negocio con Software as a Service (SaaS).

4.5 QD TECHNOLOGY

QD Technology Base de Datos de respuesta rápida es una solución de base de datos relacional que permite a los ejecutivos de negocios y analistas de datos fácil y cómodamente obtener respuestas rápidas a consultas de base de datos de

copias locales de su base de datos. Consultas correr más rápido que con otras soluciones y ya que se ejecutan en los ordenadores de usuario, que no interfieran con otros las actividades de los usuarios, y se puede ejecutar en cualquier lugar.

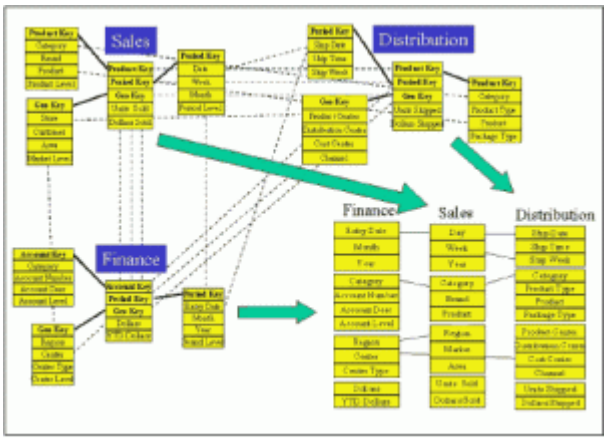
1. Compresión.
   - Mejora del rendimiento de consultas a través de una compresión inteligente.
   - Rápido acceso a través de una compresión optimizada.
   - Aplica una técnica de compresión especialmente seleccionada de una biblioteca para cada columna en cada tabla.
   - Comprime los datos y tablas basndose en patrones.
2. Compatibilidad.
   - Compatible con ODBC.
   - Consultas con el estándar SQL-92.
   - Datos almacenados en filas y columnas.
   - Solamente datos basados en ASCII.
3. Tecnología. Tablas antiguas nunca mueren. La compresión de QD es compatible con todas las versiones anteriores de QD. Plataforma y Despliegue. Plataforma. La configuración mínima para QD server es:
   o Windows server 2003 o 2000, XP o Vista.
   o 2 GB RAM
   o Espacio en disco suficiente para mantener la base de datos.
4. Tiempo de implementación.
   - Instalación y configuración inicial de QD, en general se puede completar en 30 min.
5. Habilidades requeridas.
   - QD es compatible con ODBC sin especial formacin a nivel de usuario.
6. Seguridad.
   - Datos en la PC.
   - Encriptación.

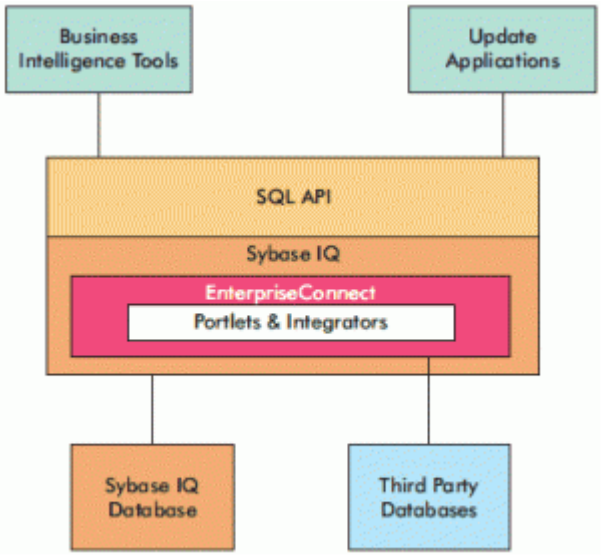
4.6 SYBASE

Sybase es una base de datos relacional basada en columnas que es intrínsecamente más apropiado para el adecuado procesamiento de consultas que un enfoque basado en filas. Debido a que está basado en columnas, Sybase IQ aprovecha las características de cada columna en la tabla, en un número de diferentes caminos.
Sybase soporta los esquemas relacionales tradicionales, incluyendo la normalización de esquemas usados para procesos de transacción.

Como se puede ver Sybase incluye una API SQL que permite el acceso a SQL, también incluye ODBC, JDBC y XML, provee java para que puede ser usado para escribir procedimientos almacenados y funciones de usuario.

* Ofrece una serie de índices especializados para el adecuado rendimiento de las consultas.
Una consecuencia de utilizar el almacenamiento columnar en conjunción con la


Sales
Distribution
Finance
Finance
Sales
Distribution


Business Intelligence Tools
Update Applications
SQL API
Sybase IQ
EnterpriseConnect
Portlets & Integrators
Sybase IQ Database
Third Party Databases

indexación de Sybase IQ Bit Wise es que las agrupaciones pueden hacerse bajo la marcha. Dado que una parte significativa de extraer, trasformar y cargar es la anterior agrupación de transacciones.
Compresión de datos es mucho más fcil de implementar en un enfoque basado en columnas que cuando se utilizan los métodos convencionales. Es significativamente más eficiente. En la práctica Sybase IQ ha demostrado una compresión de datos de un 50% a un 70% del conjunto de datos original.
Es fácil agregar y cargar una columna de datos a una tabla como seria agregar una fila a una base de datos relacional convencional.
Un enfoque basado en columnas es mucho más fcil de mantener y requiere menos sintonización que unDWH convencional.
Multihilo y alta disponibilidad 24 x 7.
A parte de las características ya mencionadas, también apoya RCube, estructura plana que puede proveer importantes beneficios en comparación con los esquemas convencionales. En particular RCubepuede acelerar significativamente la implementación, así como el rendimiento en tiempo de ejecución y proporcionar una mayor flexibilidad. * Sybase ha sido creado para soportar el mayor numero de consultas posible corriendo en paralelo en lugar de concentrarse en el uso del paralelismo para optimizar el rendimiento de una consulta en particular.

5 Conclusión

A Medida que las organizaciones siguen empleando grandes almacenes de datos para fines que van desde la presentación de informes estándar para análisis de negocios estratégicos, el procesamiento de eventos complejos y profundos de buceo de minería de datos, la necesidad de que el rendimiento sea eficaz seguirá superando las capacidades de las tradicionales bases de datos relacionales.

Diferentes tipos de organizaciones reconocen cada vez más los beneficios potenciales de la forma de bases de datos analíticas que pueden apoyar la presentación y análisis estratégico, y otras actividades de inteligencia de negocios. Y a medida que los volúmenes de datos utilizados para el análisis de aumento, los tamaos de los almacenes de datos utilizados para apoyar las actividades de inteligencia de negocios también deben crecer para satisfacer las necesidades de la organización, en términos de tamaõ, rendimiento y escalabilidad. Con el fin de satisfacer la necesidad rápidamente explotando para un rendimiento analítico, un enfoque alternativo de base de datos, que comienza por el almacenamiento de datos orientados por columnas en lugar de filas, se ha demostrado para sostener el rendimiento y los requisitos de crecimiento rápido de aplicaciones analíticas. Además, las características de simplicidad y el rendimiento del enfoque de columnas ofrecen una alternativa costo-efectiva en la aplicación de la especialidad de análisis de servidores para soportar una amplia gama de usuarios y tipos de consulta.

6 Bibliografía

References

1. WebSite, Setiembre 2012
   http://abd-ucv-computacion.wikispaces.com/Sistemas+de+Base+de+datos+Orientadas+a+Columnas

2. WebSite, Setiembre 2012
   http://db.csail.mit.edu/projects/cstore/abadi-sigmod08.pdf

3. WebSite, Setiembre 2012
   http://db.lcs.mit.edu/projects/cstore/vldb.pdf

4. WebSite, Setiembre 2012
   http://en.wikipedia.org/wiki/Column-oriented_DBMS

5. WebSite, Setiembre 2012
   http://www.stratebi.es/todobi/abr12/DBColumn_OpenSource.pdf

6. WebSite, Setiembre 2012
   http://www.information-management.com/issues/2007_42/10000432-1.html

7. WebSite, Setiembre 2012
   http://www.gravitar.biz/index.php/bi/base-datos-columnar/

8. WebSite, Setiembre 2012
   http://www.youtube.com/watch?v=u5LfQp4vQKs&feature=related

9. WebSite, Setiembre 2012
   http://en.wikipedia.org/wiki/LucidDB

10. WebSite, Setiembre 2012
    http://repositorio.utp.edu.co/dspace/bitstream/11059/2473/1/005756A992.pdf

11. WebSite, Setiembre 2012
    http://es.wikipedia.org/wiki/Apache_Cassandra

12. WebSite, Setiembre 2012
    http://maxgrinev.com/2010/07/09/a-quick-introduction-to-the-cassandra-data-model/