Tema

BASES DE DATOS LIBRES
¿Qué alternativas existen?

Desarrolladores

- ***Acosta, Rodrigo Martín –*** *rodrimosca@hotmail.com*
- ***Remedi, Rolando Martin –*** *martinchito@gmail.com*
- ***Schumacher, Diego Leonel –*** *dschumacher1@gmail.com*

INDICE

PRESENTACIÓN

El siguiente trabajo pretende conocer las distintas alternativas en Base de Datos que existan en la actualidad en lo que respecta a Software Libre.

Las Bases de Datos incluidas en el informe son:

- **Firebird**
- **MySQL**
  - **MaxDB**
  - **MySQL Cluster**
- **POSTGRESQL**
- **HSQLDB**
- **SQLite**
- **Vulcan**
- **H2**

Asimismo, se presentan algunas alternativas a las Bases de Datos anteriores, que aun no cumplen con las características necesarias para denominarse motores de Base de Datos, pero creemos conveniente describirlas brevemente.

Ellas son:

- **KEXI**
- **GNOME-DB**

FIREBIRD

Firebird es un sistema de administración de base de datos relacional (o RDBMS) de código abierto, basado en la versión 6.0 de Interbase, cuyo código fue liberado por Borland en 2000. Su código fue reescrito de C a C++. El proyecto se desarrolla activamente y el 18 de abril de 2008 fue liberada la versión 2.1.

Firebird viene con un completo conjunto de utilitarios de línea de comandos que te permiten crear bases de datos, obtener estadísticas, ejecutar comandos y scripts SQL, realizar copias de seguridad y restauraciones, etc.

En Windows, puedes ejecutar Firebird como servicio o en modo aplicación. El instalador puede crear, en el panel de control, un icono para administrar el servidor (iniciarlo, detenerlo, etc.).

Firebird está siendo usado para grandes bases de datos con muchas conexiones.

La tecnología de Firebird ha estado en uso por 20 años, lo que lo hace un producto muy estable y maduro.

Sus principales características son:

- Es multiplataforma, y actualmente puede ejecutarse en los sistemas operativos: Linux, HP-UX, FreeBSD, Mac OS, Solaris y Microsoft Windows.
- Arquitectura Cliente/Servidor sobre protocolo TCP/IP y otros (embedded).
- Soporte de transacciones ACID y claves foráneas.
- Integridad Referencial.
- Arquitectura multi-generacional.
- Ejecutable pequeño, con requerimientos de hardware bajos.
- Completo lenguaje interno para procedimientos almacenados y disparadores (PSQL)
- Soporte para Funciones Externas (UDFs).
- Poca o ninguna necesidad de DBAs especializados.
- Prácticamente no requiere configuración.
- Gran comunidad y muchos sitios donde podes encontrar soporte gratuito.
- Versión incrustada, ideal para crear catálogos en CDROM, versiones mono usuario, de evaluación o portátiles de las aplicaciones.

- Docenas de herramientas de terceros, como herramientas de administración gráficas, herramientas de replicación, etc.
- Escritura segura - recuperación rápida.
- Muchas formas de acceder a tu base de datos: nativo/API, drivers dbExpress, ODBC, OLEDB, proveedor .Net, driver JDBC nativo tipo 4, módulo Python, PHP, Perl, etc.
- Copias de seguridad incrementales.
- Disponibilidad de binarios en arquitectura de 64bits.
- Implementación completa de cursores en PSQL.
- Tablas de Monitoreo.
- Disparadores a nivel de Conexión y Transacción.
- Tablas Temporales.
- Es medianamente escalable.
- Buena seguridad basada en usuarios/roles.
- Diferentes arquitecturas, entre ellas el Firebird incrustado (embedded server) que permite ejecutar aplicaciones monousuario en ordenadores sin instalar el software Firebird.
- Bases de datos de sólo lectura, para aplicaciones que corran desde dispositivos sin capacidad de escritura, como cd-roms.
- Requisitos de administración bajos, siendo considerada como una base de datos libre de mantenimiento, al margen de la realización de copias de seguridad.
- Pleno soporte del estándar SQL-92, tanto de sintaxis como de tipos de datos.
- Capacidad de almacenar elementos BLOB (Binary Large OBjects).

Tipos de servidor

Existen básicamente dos tipos de servidor Firebird para ser instalados: **Classic** y **Super server**. Si bien tienen varias diferencias menores entre si, la principal consiste en que el super server maneja hilos de ejecución individuales para cada conexión.

La edición Classic inicia un proceso servidor independiente por cada conexión que recibe.

Por lo tanto, para un número reducido de conexiones el recomendado sería el classic porque consumirá menor cantidad de recursos.

En caso de arquitecturas SMP, se debe utilizar el servidor **classic** porque el **Supersever** no tiene soporte para este tipo de arquitectura.

Los propios desarrolladores de Firebird recomiendan lo siguiente a la hora de decidirse por uno de estos servidores:

- En plataformas Windows, seleccionar el **Superserver**.
- En Linux, simplemente elegir cualquiera, según las conexiones estimadas. En la mayoría de las situaciones no se notará diferencias en la ejecución.

Podría considerarse un tercer tipo, el **Embedded**. Éste consiste en una única biblioteca de enlace dinámico DLL (de unos 2 MB de tamaño) que contiene todo el servidor. De esta forma se puede tener un DBMS completo disponible y distribuíble junto con aplicaciones de usuario sin requerir que este se instale por separado. Es un completo Servidor Firebird empacado en unos cuantos ficheros. Con él es muy fácil distribuir aplicaciones, puesto que no requiere instalación. Es ideal para catálogos en CDROM, demostraciones o aplicaciones de escritorio independientes.

POSTGRESQL

PostGreSQL es un sistema de gestión de bases de datos objeto-relacional (ORDBMS) basado en el proyecto POSTGRES, de la universidad de Berkeley. El director de este proyecto es el profesor Michael Stonebraker, y fue patrocinado por Defense Advanced Research Projects Agency (DARPA), el Army Research Office (ARO), el National Science Foundation (NSF), y ESL, Inc.

PostGreSQL es una derivación libre (OpenSource) de este proyecto, y utiliza el lenguaje SQL92/SQL99, así como otras características que comentaremos más adelante.

Fue el pionero en muchos de los conceptos existentes en el sistema objeto-relacional actual, incluido, más tarde en otros sistemas de gestión comerciales. PostGreSQL es un sistema objeto-relacional, ya que incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional. A pesar de esto, PostGreSQL no es un sistema de gestión de bases de datos puramente orientado a objetos.

Historia de PostGreSQL

La implementación de Postgres DBMS comenzó en 1986, y no hubo una versión operativa hasta 1987. La versión 1.0 fue liberada en Junio de 1989 a unos pocos usuarios, tras la cual se liberó la versión 2.0 en Junio de 1990 debido a unas críticas sobre el sistema de reglas, que obligó a su reimplementación. La versión 3.0 apareció en el año 1991, e incluyó una serie de mejoras como una mayor eficiencia en el ejecutor de peticiones. El resto de versiones liberadas a partir de entonces, se centraron en la portabilidad del sistema. El proyecto se dio por finalizado en con la versión 4.2, debido al gran auge que estaba teniendo, lo cual causó la imposibilidad de mantenimiento por parte de los desarrolladores.

En 1994, Andrew Yu y Jolly Chen añadieron un intérprete de SQL a este gestor. Postgres95, como así se llamó fue liberado a Internet como un proyecto libre (OpenSource). Estaba escrito totalmente en C, y la primera versión fue un 25% más pequeña que Postgres, y entre un 30 y un 50% más rápida. A parte de la corrección de algunos bugs, se mejoró el motor interno, se añadió un nuevo programa monitor, y se compiló usando la utilidad GNU Make y el compilador gcc sin necesidad de parchearlo (como había hecho falta en versiones anteriores).

En 1996, los desarrolladores decidieron cambiar el nombre a al DBMS, y lo llamaron PostGreSQL (versión 6.0) para reflejar la relación entre Postgres y las versiones recientes de SQL. Se crearon nuevas mejoras y modificaciones, que repercutieron en un 20-40% más de eficiencia, así como la incorporación del estándar SQL92.

Características de PostGreSQL

A continuación se enumeran las principales características de este gestor de bases de datos:

1. Implementación del estándar SQL92/SQL99.
2. Soporta distintos tipos de datos: además del soporte para los tipos base, también soporta datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes (MAC, IP ...), cadenas de bits, etc. También permite la creación de tipos propios.
3. Incorpora una estructura de datos array.
4. Incorpora funciones de diversa índole: manejo de fechas, geométricas, orientadas a operaciones con redes, etc.
5. Permite la declaración de funciones propias, así como la definición de disparadores.
6. Soporta el uso de índices, reglas y vistas.
7. Incluye herencia entre tablas (aunque no entre objetos, ya que no existen), por lo que a este gestor de bases de datos se le incluye entre los gestores objeto-relacionales.
8. Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.

PostGreSQL es un magnífico gestor de bases de datos. Tiene prácticamente todo lo que tienen los gestores comerciales, haciéndo de él una muy buena alternativa GPL. Posee una gran escalabilidad, haciéndolo idóneo para su uso en sitios web que posean alrededor de 500.000 peticiones por día. Sin embargo, la sintaxis de algunos de sus comandos no es nada intuitiva. Por otro lado, la velocidad de respuesta que ofrece este gestor con bases de datos relativamente pequeñas puede parecer un poco deficiente, aunque esta misma velocidad la mantiene al gestionar bases de datos realmente grandes, cosa que resulta loable. También resulta engorroso las pequeñas variaciones que presenta este gestor en algunos de los tipos de datos que maneja, siendo el problema más comentado el referente al tipo "serial".

MySQL

MySQL es un sistema de gestión de bases de datos relacional, licenciado bajo la GPL de la GNU. Su diseño multihilo le permite soportar una gran carga de forma muy eficiente. MySQL fue creada por la empresa sueca MySQL AB, que mantiene el copyright del código fuente del servidor SQL, así como también de la marca.

Aunque MySQL es software libre, MySQL AB distribuye una versión comercial de MySQL, que no se diferencia de la versión libre más que en el soporte técnico que se ofrece, y la posibilidad de integrar este gestor en un software propietario, ya que de no ser así, se vulneraría la licencia GPL.

Este gestor de bases de datos es, probablemente, el gestor más usado en el mundo del software libre, debido a su gran rapidez y facilidad de uso. Esta gran aceptación es debida, en parte, a que existen infinidad de librerías y otras herramientas que permiten su uso a través de gran cantidad de lenguajes de programación, además de su fácil instalación y configuración.

MySQL surgió como un intento de conectar el gestor mSQL a las tablas propias de MySQL AB, usando sus propias rutinas a bajo nivel. Tras unas primeras pruebas, vieron que mSQL no era lo bastante flexible para lo que necesitaban, por lo que tuvieron que desarrollar nuevas funciones. Esto resultó en una interfaz SQL a su base de datos, con una interfaz totalmente compatible a mSQL.

Las principales características de este gestor de bases de datos son las siguientes:

1. Aprovecha la potencia de sistemas multiprocesador, gracias a su implementación multihilo.
2. Soporta gran cantidad de tipos de datos para las columnas.
3. Dispone de API's en gran cantidad de lenguajes (C, C++, Java, PHP, etc).
4. Gran portabilidad entre sistemas.
5. Soporta hasta 32 índices por tabla.
6. Gestión de usuarios y passwords, manteniendo un muy buen nivel de seguridad en los datos.

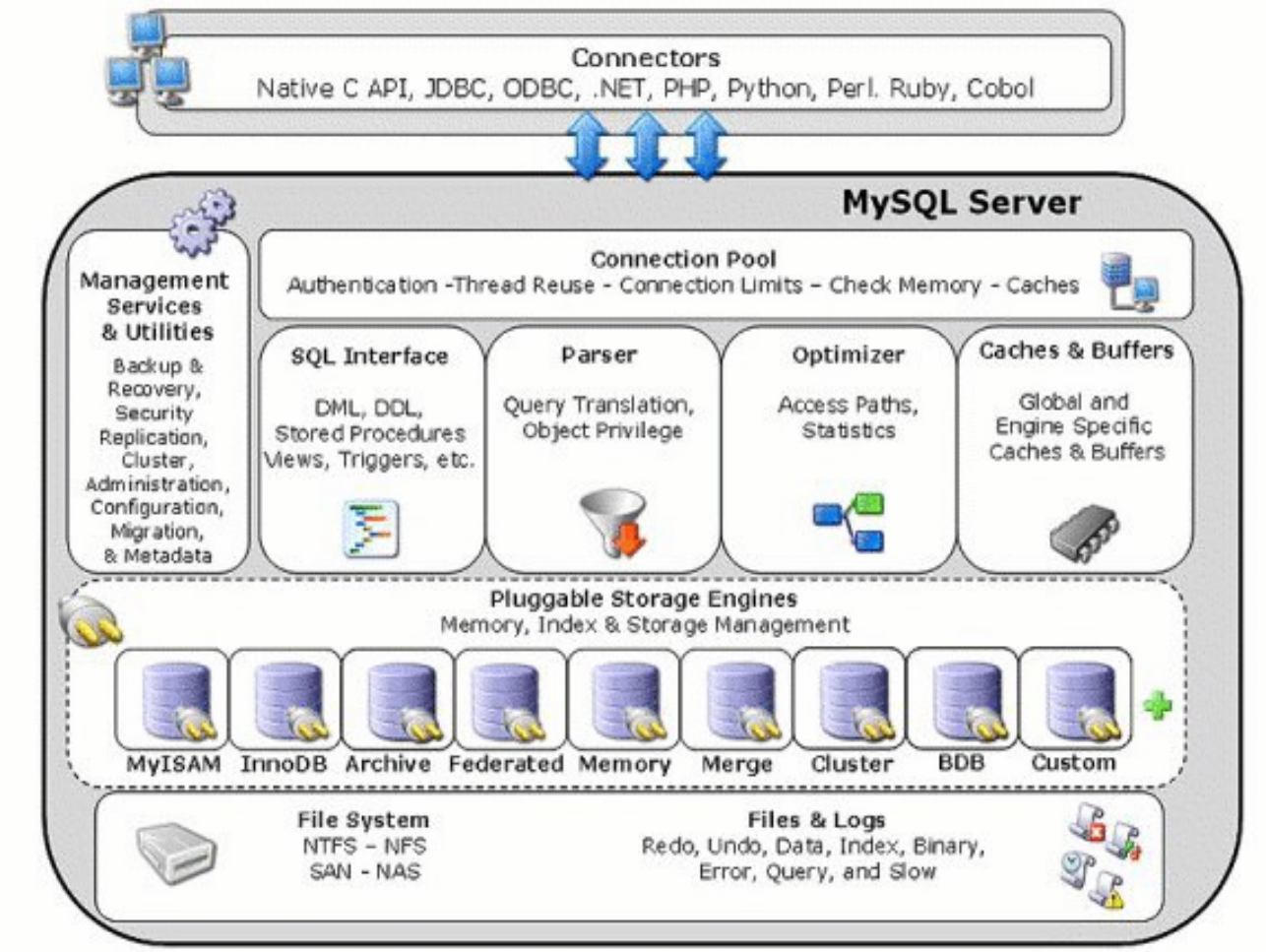
Aunque MySQL se incluye en el grupo de sistemas de bases de datos relacionales, carece de algunas de sus principales características:

1. Subconsultas: tal vez ésta sea una de las características que más se echan en falta, aunque gran parte de las veces que se necesitan, es posible reescribirlas de manera que no sean necesarias.

2. SELECT INTO TABLE: Esta característica propia de Oracle, todavía no está implementada.
3. Triggers y Procedures: Se tiene pensado incluir el uso de procedures almacenados en la base de datos, pero no el de triggers, ya que los triggers reducen de forma significativa el rendimiento de la base de datos, incluso en aquellas consultas que no los activan.
4. Transacciones: a partir de las últimas versiones ya hay soporte para transacciones, aunque no por defecto (se ha de activar un modo especial).
5. Integridad referencial: aunque sí que admite la declaración de claves ajenas en la creación tablas, internamente no las trata de forma diferente al resto de campos.

Los desarrolladores comentan en la documentación que todas estas carencias no les resultaban un problema, ya que era lo que ellos necesitaban. De hecho, MySQL fue diseñada con estas características, debido a que lo que buscaban era un gestor de bases de datos con una gran rapidez de respuesta. Pero ha sido con la distribución de MySQL por Internet, cuando más y más gente les están pidiendo estas funcionalidades, por lo que serán incluidas en futuras versiones del gestor.

La mayoría de las personas que utilizan MySQL saben que MyISAM e InnoDB son los dos motores de almacenamientos más comunes en MySQL. También es sabido, que la mayoría no toma en cuenta el motor de almacenamiento al crear una tabla y acepta el que viene por default en la base de datos.



El motor de almacenamiento (storage-engine) es quien almacenará, manejará y recuperará información de una tabla en particular. Comparando MyISAM vs InnoDB, ninguno se destaca como la solución para la mayoría de los casos. Cada uno tiene sus pros y sus contras, por lo tanto al momento de decidir que motor de almacenamiento a utilizar dependerá mucho del escenario donde se aplique.

Bases de Datos derivadas de MySQL

Dentro de MySQL, encontramos otras Bases de Datos, como MaxDB y MySQLCluster, que forman parte de ella y las describimos a continuación.

- MaxDB

MaxDB by MySQL

MaxDB opera como un producto cliente/servidor. Fue desarrollado para cubrir las demandas de las instalaciones que requieren de una gran volumen de procesamiento de transacciones. Tanto el back up en línea como expansión de la base de datos están soportados. El Servidor Cluster de Microsoft tiene soporte directo para múltiples instalaciones de servidores; otros requerimientos deben ser escritos manualmente. Las herramientas de administración son provistas tanto en aplicaciones Desktop como implementaciones basadas en browser.

Historia

La historia de SAP DB se remonta hacia los principios de 1980 cuando fue desarrollada como un producto comercial(ADABAS). Esta Base de Datos fue cambiando de nombre varias veces desde ese entonces. Cuando SAP AG, una compañía con sede en Walldorf, Alemania, se hizo cargo del desarrollo del producto, fue bautizada como SAP DB.

SAP desarrollo esta base de datos para funcionar como sistema de almacenamiento de todos los sistemas de misión critica de las Aplicaciones de SAP, llamadas R/3. SAP DB fue pensada para proveer una alternativa a sistemas de bases de datos como Oracle, Microsoft SQL Server, y DB2 de IBAM. En Octubre de 2000, SAP AG libero bajo la licencia GNU GPL, convirtiéndola en Software Open Source. En Octubre de 2003, mas de 2,000 clientes de SAP AG estaban utilizando SAP DB como su principal sistema de bases de datos y otros 2000 clientes estaban utilizando de forma separada como parte de la solución APO/Live cache

En Mayo de 2003 una alianza tecnológica fue conformada entre MySQL AB y SAP AG. Esa alianza fue llamada MySQL AB se encargara del futuro desarrollo de SAP DB, de su nuevo nombre, y de la comercialización de licencias de la re bautizada SAP DB para aquellos clientes que no quieran estar bajo las restricciones impuestas por utilizar esa base de datos bajo el licenciamiento GNU GPL. En Agosto de 2003, SAP DB fue re bautizada como MaxDB por MySQL AB.

Diferencias de prestaciones entre MaxDB y MySQL

La siguiente lista brinda un sumario de las principales diferencias entre MaxDB y MySQL; Esta lista es parcial.

- MaxDB corre en la modalidad de un sistema cliente/servidor. MySQL puede funcionar tanto cliente/servidor como tambien en un sistema ebedded.
- MaxDB no funciona en todas las platformas soportadas por MySQL. Por ejemplo, MaxDB no corre sobre el sistema operativo de IBM OS/2.
- MaxDB usa un protocolo de red propietario para la comunicación de cliente/servidor. MySQL usa TCP/IP (con o sin el cifrado SSL), enchufes (bajo sistemas Parecidos), o tubos llamados (bajo el Windows NT - sistemas de familia).
- MaxDB apoya procedimientos almacenados. Para MySQL, los procedimientos almacenados son puestos en práctica en la versión 5.0. MaxDB también apoya el programa de triggers por una extensión SQL, que es programada para MySQL 5.1. MaxDB contiene un depurador para lenguajes de procedimiento almacenado, y soporta múltiples triggers por acción y fila.
- MaxDB es distribuido con los interfaces de usuario que son a base de texto,

gráficos, o a base de Web. MySQL es distribuido con interfaces de usuario sólo a base de texto; la interfaz de usuario gráfico (MySQL Centro de Control, MySQL Administrador) es embarcada (transportada) separadamente de las distribuciones principales. Los terceros ofrecen interfaces de usuario a base de Web para MySQL.

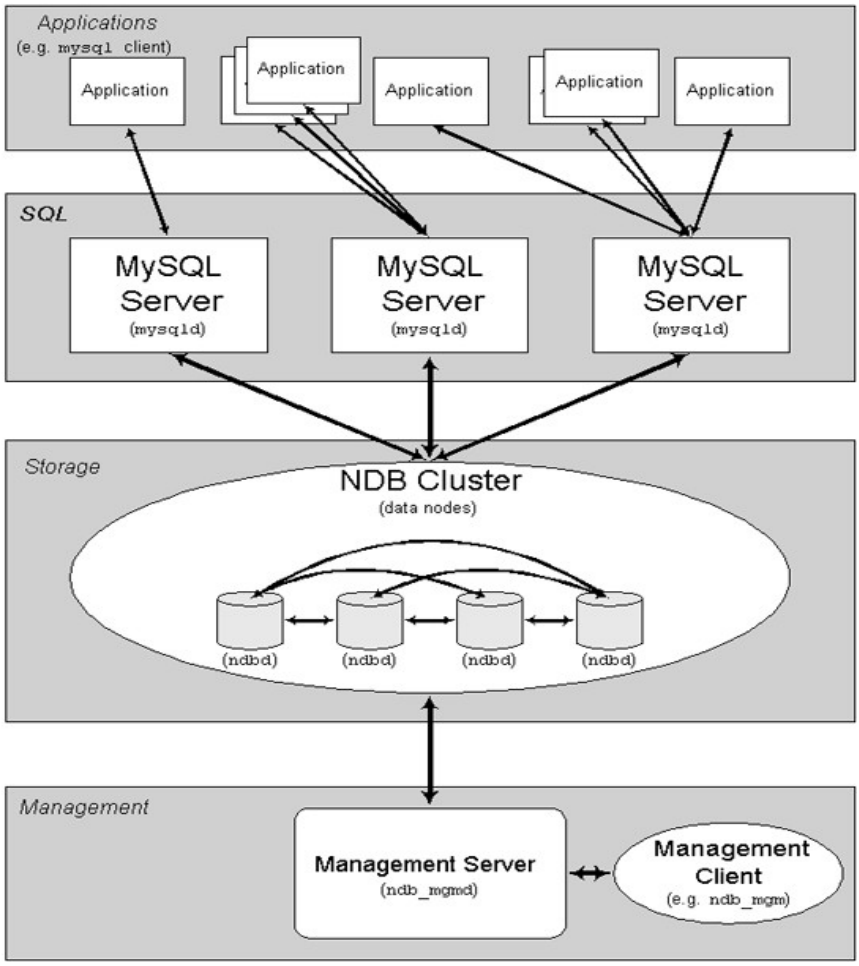
- MaxDB soporta un número de interfaces de programa que también son soportadas por MySQL. Sin embargo, MaxDB no soporta la RADIO, LA ALHARACA, o .NET, todo lo cual es soportado por MySQL. MaxDB soporta integramente SQL sólo con C/C ++.

MySQL Cluster

MySQL Cluster es una tecnología que permite clustering de bases de datos en memoria en un entorno de no compartición. La arquitectura de no compartición permite que el sistema funcione con hardware barato, y sin ningún requerimiento especial de hardware o software. Tampoco tienen ningún punto único de fallo porque cada componente tiene su propia memoria y disco.

MySQL Cluster integra el servidor MySQL estándar con un motor de almacenamiento clusterizado en memoria llamado **NDB**. En nuestra documentación, el término **NDB** se refiere a la parte de la inicialización específica al motor de almacenamiento, mientras que **MySQL Cluster** se refiere a la combinación de MySQL y el nuevo motor de almacenamiento.

Un MySQL Cluster consiste en un conjunto de máquinas, cada una ejecutando un número de procesos incluyendo servidores MySQL , nodos de datos para NDB Cluster, servidores de administración, y (posiblemente) programas especializados de acceso a datos. La relación de estos componentes en un cluster se muestra aquí:



Todos estos programas funcionan juntos para formar un MySQL Cluster. Cuando se almacenan los datos en el motor NDB Cluster, las tablas se almacenan en los nodos de datos. Tales tablas son directamente accesibles desde todos los otros servidores MySQL en el cluster. Por lo tanto, en una aplicación de pago que almacene datos en un cluster, si una aplicación actualiza el salario de un empleado, todos los otros servidores MySQL que acceden a estos datos pueden ver el cambio inmediatamente.

Los datos almacenados en los nodos de datos de MySQL Cluster pueden replicarse: el cluster puede tratar fallos de nodos de datos individuales sin otro impacto a parte de abortar unas pocas transacciones debido a la pérdida de estado de transacción. Como las aplicaciones transaccionales se suponen que tratan fallos transaccionales, esto no debería ser un problema.

Al llevar MySQL Cluster al mundo Open Source , MySQL propociona tratamiento de datos clusterizado con alta disponibilidad, alto rendimiento, y escalabilidad disponible para todo el que lo necesite.

Conceptos básicos

NDB es un motor de almacenamiento en memoria que ofrece alta disponibilidad y caracterísitcas de persistencia de datos.

El motor NDB puede configurarse con un rango de opciones de fallo y balanceo de carga, pero es más sencillo arrancarlo con el motor de almacenamiento a nivel de cluster. El motor de MySQL Cluster NDB contiene un conjunto completo de datos, dependiente sólo de otros datos dentro del propio cluster.

Ahora describiremos cómo inicializar un MySQL Cluster consistente de un motor NDB y algunos servidores MySQL .

La porción de cluster del MySQL Cluster está configurada independientemente de los servidores MySQL . En MySQL Cluster, cada parte del cluster se considera como un **nod**.

Nota: En muchos contextos, el término "nodo" se usa para indicar una máquina, pero cuando se discute MySQL Cluster significa un *proceso*. Puede haber cualquier número de nodos en una máquina, para los que se usa el término **máquina cluster**.

Hay tres tipos de nodos cluster, y en una configuración MySQL Cluster mínima, al menos habrán tres nodos, uno de cada tipo:

- El nodo de administración (MGM) : El rol de este tipo de nodo es administrar los otros nodos dentro del MySQL Cluster, tal como proporcionar datos de configuración, iniciar y parar nodos, ejecutar copias de seguridad, y así. Como este tipo de nodo administra la configuración de otros nodos, un nodo de este tipo debe arrancarse primero, antes de cualquier otro nodo. Un nodo MGM se arranca con el comando **ndb_mgmd**.
- El **nodo de datos**: Este es el tipo de nodo que almacena los datos del cluster. Hay tantos nodos de datos como réplicas, multiplicado por el número de fragmentos. Por ejemplo, con dos réplicas, cada uno teniendo dos fragmentos, necesita cuatro nodos de datos. No es necesario tener más de una réplica. Un nodo de datos se arranca con el comando **ndbd**.
- El **nodo SQL**: Este es el nodo que accede a los datos del cluster. En el caso de MySQL Cluster, un nodo cliente es un servidor MySQL tradicional que usa el motor NDB Cluster . Un nodo SQL típicamente se arranca con el comando **mysqld --ndbcluster** o símplemente usando **mysqld** con ndbcluster añadido a my.cnf.

La configuración de un cluster implica configurar cada nodo individual en el cluster y inicializar los enlaces de comunicación individual entre los nodos. MySQL Cluster está diseñado con la intención que los nodos de almacenamiento son homogéneos en términos de procesador, espacio de memoria, y ancho de banda. Además, para proporcionar un punto único de configuración, todos los datos de configuración del cluster entero se guardan en un único fichero de configuración.

El servidor de administración (nodo MGM) administra el fichero de configuración del cluster y el log. Cada nodo en el cluster recibe los datos de configuración del servidor de administración, y necesita una forma de determinar dónde reside el servidor de administración. Cuando ocurren eventos interesantes en los nodos de datos, los nodos transfieren información acerca de estos eventos al servidor de administración, que guarda la información en el log del cluster.

Además, puede haber cualquier número de procesos clientes del cluster o aplicaciones. Hay de dos tipos:

- **Clientes MySQL estándar**: No son diferentes para MySQL Cluster que para cualquier MySQL (no cluster). En otras palabras,MySQL Cluster puede ser accedido para aplicaciones MySQL existentes escritas en PHP, Perl, C, C++, Java, Python, Ruby, y así.
- **Clientes de administración**: Estos clientes conectan al servidor de administración y proporcionan comandos para arrancar y parar nodos, arrancar y parar traceo de

mensajes (sólo en versiones de depuración), mostrar versiones y estatus de nodos, arrancar y parar copias de seguridad, y así.

HSQLDB

HSQLDB es una base de datos hecha totalmente en Java, pero que tiene características muy interesantes para usarla con Applets o en test automáticos.

Por un lado, se puede arrancar un servidor de base de datos HSQLDB y conectarse a él desde Java como a cualquier otra base de datos, utilizando, por supuesto, el driver suministrado por HSQLDB.

Por otro lado, sin necesidad de arrancar un servidor y sobre la marcha, una aplicación java puede conectarse a una base de datos HSQLDB ficticia que se guardará en fichero. De esta forma, sin servidor de base de datos, podemos crear tablas y llenarlas de datos que se guardarán en un fichero, y luego recuperarlos.

Finalmente, podemos hacer lo mismo que en el caso anterior, pero sobre memoria. No se escribe ningún dato en ningún sitio y cuando la aplicación termine se perderán los datos. Esta posiblidad es muy útil para Applets o para test automáticos de prueba. En un test automático de prueba, podemos crear la base de datos en memoria con sus tablas y comprobar el comportamiento de la clase bajo test.

HSQLDB es la principal base de datos relacional SQL motor escrito en Java. Tiene un controlador JDBC y apoya un rico subconjunto de ANSI-92 SQL (BNF árbol formato), además de SQL 2003 y 99 mejoras. Ofrece un pequeño (menos de 100k en una versión para los applets), rápido motor de base de datos que ofrece-tanto en la memoria y el disco a base de cuadros y apoya incorporados y los modos de servidor. Además, incluye herramientas tales como un mínimo del servidor web, en memoria de consulta y herramientas de gestión (se puede ejecutar como applets) y una serie de ejemplos de demostración.

El producto está siendo utilizado como una base de datos y motor de persistencia en muchos programas informáticos de código abierto y los proyectos, incluso en proyectos comerciales y productos.. En su versión actual es muy estable y fiable. Es mejor conocido por su pequeño tamaño, la capacidad de ejecutar completamente en memoria, su flexibilidad y velocidad.

Esta característica repleto de software es completamente libre de utilizar y distribuir con arreglo a nuestras licencias, basado en el estándar de la licencia BSD. Completamente libre de costo o restricciones y plenamente compatible con todas las principales licencias de código abierto.

SQLite

SQLite es una de las herramientas de Bases de Datos que ha crecido significativamente en los últimos tiempos, y es utilizada por grandes programadores y en diversos proyectos informáticos.
SQLlite es una pequeña librería (de aproximadamente 500kb) programada en lenguaje C que implementa un completo motor de base de datos multiplataforma, que no necesita configuración. Su creador es D. Richard Hipp. Se distribuye bajo licencia de dominio público. Es muy rápida, versátil y robusta, y la ventaja fundamental es que permite utilizar un amplio subconjunto del lenguaje estándar SQL.

Uno de las primeras diferencia con los motores de Bases de datos convencionales es su arquitectura cliente/servidor, pues SQLite es independiente; simplemente, se realizan llamadas a sub rutinas o funciones de las propias librerías de SQLite, lo cual reduce ampliamente la latencia en cuanto al acceso a las bases de datos. Con esto, se puede decir que las bases de datos compuesta por la definición de las tablas, índices y los propios datos son guardados por un solo fichero estándar y en un solo ordenador. Esto hace que cada usuario pueda crear tantas bases de datos como desee sin la necesidad de la intervención de un administrador de bases de datos que gestione los espacios de trabajo, usuarios y permisos de acceso. El hecho de almacenar toda la base de datos en un único archivo, facilita la portabilidad de los datos, y solamente tiene la restricción del espacio de disco asignado al usuario en el servidor.

Su potencia se basa fundamentalmente en la simplicidad, lo que hace que no sea una buena solución en entornos de tráfico muy elevado y/o alto acceso concurrente a datos. SQLite encapsula toda la base de datos en un único fichero.

En su versión 3, SQLite soporta bases de datos de hasta 2 Terabytes de tamaño, y también permite la inclusión de campos tipo Blob. Es por ello, entre otras cosas, que grandes empresas como Adobe, Firefox, Google, McAfee, Toshiba, Sun Microsystem, etc., hacen uso de SQLite para el desarrollo de varios de sus productos, demostrando de esta manera la confianza y el gran rendimiento de la misma.

Se puede utilizar SQLite de dos formas:

- **Como gestor de base de datos local en una PC.** De esta forma, se puede gestionar bases de datos con SQLite igual que si se estuviera trabajando con un sistema gestor de base de datos como MySQL sin necesidad de instalar nada, ya que SQLite se compone de un único archivo ejecutable.
- **Como una extensión más de PHP, utilizando las funcionalidades de SQLite configuradas, o bien como módulo de PHP, o como librería; sin necesidad de tener instalado o conectar con un servidor de base de datos.** Esta opción, ofrece una rápida interfaz de base de datos, al igual que ofrecen otras bases de

datos como MySQL, pero con la ventaja de no tener la necesidad de tener instalado o conectar con un servidor de base de datos. SQLite tiene prácticamente las mismas funcionalidades y rapidez que el resto de gestores de base de datos, y los datos se almacenan en un archivo de texto plano.

En resúmen, algunas de las ventajas más relevantes de SQLite son las siguientes:

- **Rendimiento de base de datos.** SQLite realiza operaciones de manera eficiente y es más rápido que MySQL y PostgreSQL.
- **No posee configuración. N**o necesita ser instalado. No prender, reiniciar o apagar un servidor, e incluso configurarlo. Esta cualidad permite que no haya un administrador de base de datos para crear las tablas, vistas, asignar permisos.
- **Portabilidad.** Se ejecuta en diversas plataformas y sus bases de datos pueden ser fácilmente portadas sin ninguna configuración o administración. La portabilidad no está dada en sí por el software, sino por la base de datos condensada en un solo fichero, que puede estar situado en cualquier directorio, trayendo como ventaja que la base de datos puede ser fácilmente copiada a algún dispositivo USB o ser enviada vía correo electrónico.
- **Registros de longitud variable.** El uso de registros de longitud variable por SQLite, tiene una serie de ventajas, entre ellas el resultado de un pequeño archivo de base de datos y optimización de la velocidad de la misma, puesto que hay menos información desperdiciada que leer y recorrer.
- **Estabilidad:** SQLite es compatible con ACID, reunión de los cuatro criterios de Atomicidad, Consistencia, Aislamiento y Durabilidad.
- **SQL:** implementa un gran subconjunto de la ANSI - 92 SQL estándar, incluyendo sub-consultas, generación de usuarios, vistas y triggers.
- **Interfaces:** cuenta con diferentes interfaces del API, las cuales permiten trabajar con C++, PHP, Perl, Python, Ruby, Tcl, groovy, etc.
- **Costo:** SQLite es de dominio público, y por tanto, es libre de utilizar para cualquier propósito sin costo y se puede redistribuir libremente.
- **Tamaño:** SQLite tiene una pequeña memoria y una única biblioteca es necesaria para acceder a bases de datos, lo que lo hace ideal para aplicaciones de bases de datos incorporadas.

SQLite también cuenta con algunas limitaciones:

- **Limitaciones en Where.** Esta limitación está dada por el soporte para clausuras anidadas.
- **Falta de Clave Foránea.** Se hace caso omiso de las claves foráneas; esto quiere decir, cuando se realice la creación de la tabla desde el modo consola, está permitiendo el uso de la clausura, aunque no realizara el chequeo de la misma.
- **Escasa documentación en español.** Si bien ya se cuenta con una comunidad latino americana de SQLite, sería importante encontrar mucha más documentación, libros, review, etc., como muchos otros motores de bases de datos cuentan hoy en

día.

Vulcan

Vulcan es un brazo de desarrollo nacido del DBMS Firebird.

La base de datos en línea Vulcan le permite contratar dirección física y de administrativas disposición cuestiones antes de tomar una entrega fuera de pista.

Vulcan hace que sea fácil de realizar el seguimiento de una contratación de personal de información y resultados de fitness.También puede utilizar Vulcan para generar informes personalizables que acortar reunión de preparación y simplificar la fuerza de evaluación de las presentaciones.

Use Vulcan para:

- Ingresar, actualizar y controlar la información sobre contratación de preparación
- Generar informes sobre IET disposición en los planos nacional, estatal, sitio, y contratar a los niveles
- Garantizar la nave antes de la formación de todos los reclutas

Vulcan es una única fuente para reclutar y RSP información, que combina:

- Instalación básica personal de la División de Sistema (SIDPERS)
- Contratar sistema de cuotas (Solicitar)
- Ejército de necesidades de capacitación y recursos del sistema (ATRRS)
- Total Ejército de base de datos del personal de guardia (TAPDB-G)
- Defensa sistema de contabilidad de las finanzas (DFAS)

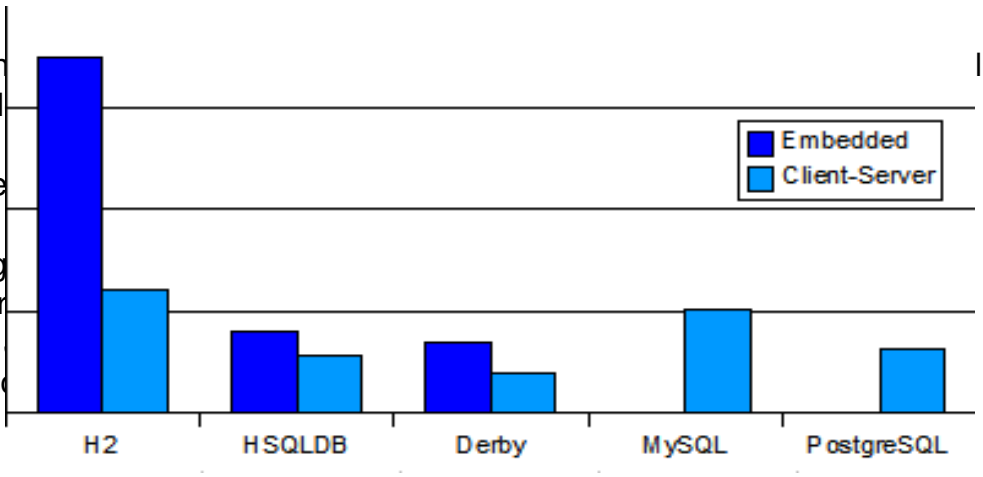
H2

H2 es un sistema administrador de bases de datos relacionales programado en Java. Puede ser incorporado en aplicaciones Java o ejecutarse de modo cliente-servidor. Una de las características más importantes de H2 es que se puede integrar completamente en aplicaciones Java y acceder a la base de datos lanzando SQL directamente, sin tener que pasar por una conexión a través de sockets.

Está disponible como software de código libre bajo la Licencia Pública de Mozilla o la Eclipse Public License.

Incluye con lustering y
está basad

Algunos de
- Alta
- Es g
- Escr
- API
- Sop
- Etc.

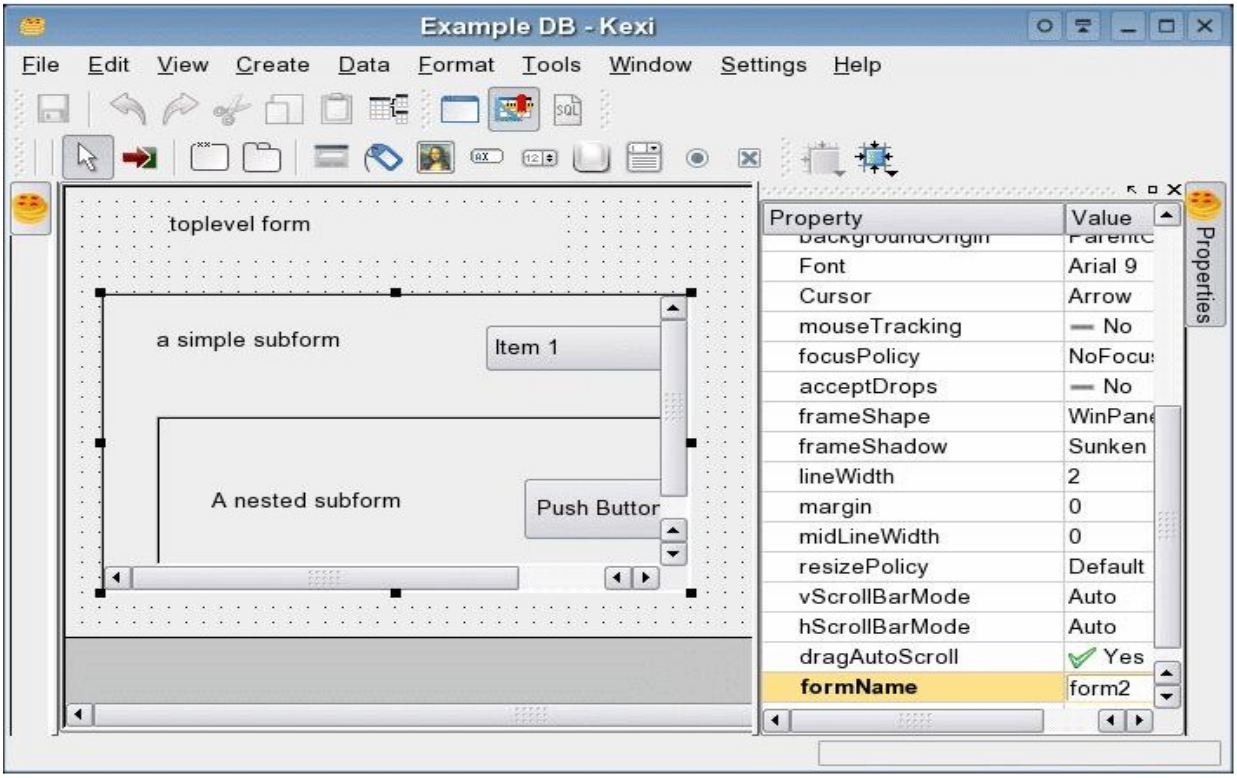


Bases de Datos Alternativas

KEXI

Kexi es una hoja de cálculo y, a la vez, una completa aplicación de gestión de base de datos.Se puede utilizar para crear los esquemas de la base de datos, insertar los datos, procesarlos y realizar las consultas.

Pueden crearse los formularios para proporcionar una interfaz personalizada de los datos. Estos formularios cuentan con toda una gama de controles para facilitar la entrada y consulta de datos, tales como: etiquetas, cuadros de texto, rejillas, etc. Concretamente, Kexi organiza toda la información que provenga desde servidores de bases de datos (MySQL o PostgresSQL), como también soporta ficheros en formato CSV para las importaciones y exportaciones, las cuales se realizan fácilmente debido a que las tablas y formularios de las bases de datos se guardan en un único archivo. Además, posee una completa compatibilidad con lenguajes de programación como Phyton o Ruby, para programar scripts. Todos los archivos que se van editando y procesando son almacenados en la base de datos que luego, si se lo desea, se podrán compartir con otros usuarios mediante cualquier tecnología. En definitiva, una de las mejores alternativas, para los más avanzados. En muchos aspectos, el modelo de trabajo es similar al de Microsoft Access, por ejemplo en los macros.
En resumen, Kexi es una herramienta para el manejo de datos muy completa y versátil, con grandes posibilidades de interacción con otros programas y tecnología.



GNOME-DB

GNOME-DB es una arquitectura de acceso a bases de datos orientada a ser usada por aplicaciones para el proyecto GNOME.

GNOME-DB ha sido diseñado de forma modular, estableciendo una serie de capas para cada una de las partes de la arquitectura. Esto facilita enormemente el desarrollo a los programadores, pues cada parte es independiente de las otras y puede ser modificada fácilmente. Además, permite el uso individual de algunos de los módulos sin necesidad de los de las capas superiores.

Así, en el núcleo mismo de la arquitectura se encuentra GDA (GNU Data Access), que es la parte más importante de todas y de la que dependen el resto de módulos. Es la parte que implementa las comunicaciones via CORBA y la que se encarga del intercambio de datos entre los clientes y los servidores de base de datos. A su vez, GDA está dividida en dos partes, una es la que implementa todo el acceso a los servidores de bases de datos (conocidos en la terminología de GNOME-DB como proveedores para no mezclarlos con los servidores de BBDD), que son servidores CORBA que implementan un determinado interfaz mediante el cual los clientes se comunican con ellos. Cada uno de estos proveedores ofrece acceso a un servidor de base de datos diferente (PostgreSQL, MySQL, ODBC, Oracle, Interbase, LDAP), de forma que los clientes sólo tienen que activar un proveedor diferente para así, sin darse cuenta, acceder a una base de datos distinta. El otro componente de GDA es la parte cliente, que consiste en una encapsulación del acceso a los proveedores CORBA desde los clientes en una librería muy fácil de usar.

La implementación de GDA, libgda, ha sido recientemente separada del módulo principal del proyecto con la idea de permitir su uso en aplicaciones no desarrolladas para GNOME. Para ello, se han eliminado todas las dependencias de GNOME.

En el nivel siguiente de la arquitectura se encuentran las librerías específicas de cada escritorio. Esta afirmación ahora mismo no tiene mucho sentido, pues sólo existe la implementación de este nivel para el proyecto GNOME, pero se ha estructurado la arquitectura de forma que GDA no sea dependiente de ningún escritorio en concreto, por lo que no sería una idea descabellada el implementar esta parte para, por ejemplo, el proyecto KDE. Estas librerías, en el caso de GNOME, contienen una serie de 'widgets' creados por los desarrolladores de GNOME-DB para facilitar el acceso a GDA desde aplicaciones GNOME, así como una serie de componentes Bonobo que pueden ser usados desde cualquier aplicación GNOME que soporte componentes.

Finalmente, en el nivel superior de la arquitectura, se encuentra un conjunto de aplicaciones y utilidades que hacen de GNOME-DB un potente entorno para el desarrollo de aplicaciones de acceso a bases de datos. Este conjunto de aplicaciones incluye varias utilidades de línea de comandos para distintos menesteres, como por ejemplo la ejecución

de ficheros SQL o la importación/exportación de datos entre distintas bases de datos, así como aplicaciones con interfaz gráfico de usuario para el acceso a las bases de datos y la configuración de todo el entorno GNOME-DB.

Desarrollo con GNOME-DB

El objetivo principal de GNOME-DB es el crear toda una infraestructura para el desarrollo de aplicaciones de acceso a bases de datos. Por ello, la parte más completa la componen las distintas librerías que componen el sistema.

La primera de ellas, libgda-server, es una encapsulación de la implementación de los interfaces CORBA para su uso en proveedores. Esta librería ha sido creada hace poco, pero desde que existe, la vida de los programadores de GNOME-DB ha cambiado, pues ahora, añadir soporte para otra base de datos distinta es cuestión de horas, y no como antes, que después de pensarlo, se prohibió la creación de nuevos proveedores durante un tiempo para evitar que el código fuente de GNOME-DB fuera corrompiéndose más y más.

gda-client, de la que ya hemos hablado anteriormente, es otra encapsulación más, pero esta vez de la parte cliente de los interfaces CORBA, es decir, de la parte que realiza las llamadas a los métodos implementados por los distintos proveedores. Esta librería es la que usan los clientes. Está orientada a objetos (aunque implementada en C), por lo que su uso resulta muy intuitivo. Se están desarrollando distintos enlaces con otros lenguajes de esta librería; entre ellos, el más avanzado es C++, mientras que Python y Pascal están aún en continuo desarrollo. Para el futuro se esperan enlaces con otros muchos lenguajes, principalmente Perl.

La parte GNOME del proyecto también incluye librerías que permiten el desarrollo de aplicaciones basadas en GNOME-DB. Estas librerías incluyen libgnomedb, que incluye una gran variedad de 'widgets' orientados al acceso a datos para su uso en aplicaciones GNOME. Estos 'widgets' van desde la típica rejilla en la que se muestran los resultados de un comando enviado a la base de datos, hasta todo un potente navegador de la base de datos, que permite la visualización de toda la estructura de la base de datos.

Por último, está libgnomedbcomponents, que en un principio fue creada para uso interno, pero que puede ser también utilizada fácilmente en otras aplicaciones. Su creación se debió al continuo cambio que experimentaba Bonobo hace unos meses, cambio que conllevaba más disgustos que otra cosa pese a significar que estaba siendo desarrollado muy activamente por varias personas. Así, se decidió añadir otra capa de encapsulación a la arquitectura debido a que el código relativo a Bonobo se usa en numerosas partes de GNOME-DB, por lo que tener todo ese código fácilmente 'cambiante' en una librería fue todo un alivio para los programadores de GNOME-DB. Pero, como se decía anteriormente, esta librería puede ser usada por otras aplicaciones para la creación y el uso de componentes Bonobo. De hecho, los integrantes del proyecto GNOME-DB animan a que sea utilizada para ello.

Varias aplicaciones hacen ya uso de estas librerías para sus accesos a datos:

- gASQL: herramienta de administración de bases de datos. Es la que incluye el soporte más completo de GNOME-DB de todas las aplicaciones que lo usan. Esto se debe a que Vivien Malerba, el desarrollador de gASQL, forma parte del equipo de desarrollo de GNOME-DB.

- GNOME Office es una "suite" ofimática que incluye varias aplicaciones GNOME orientadas a la productividad. GNOME-DB ofrece el acceso de datos de la suite.
- Glade: el famoso diseñador de pantallas Glade incluye soporte para los "widgets" de GNOME-DB. El soporte aún no es lo suficientemente completo como para facilitar el desarrollo de aplicaciones de BBDD, pero se está trabajando en una total integración.
- GNU Enterprise es un software orientado a todo tipo de empresa, suministrando la infraestructura necesaria para la informatización de cualquier empresa. También constituye el primer ejemplo de aplicación que utiliza únicamente GDA. Estos hechos están acelerando enormemente el desarrollo de esta parte, de forma que sea realmente útil para todo tipo de aplicaciones, desde sencillos clientes a todo un sistema, por ejemplo, de facturación o control de stock.

Estado actual del proyecto

En estos momentos GNOME-DB se encuentra en el punto más álgido de su desarrollo, pues para finales de este año se va a hacer la primera liberación 100% estable. Hasta ahora ha estado en buen estado y, pese a algunos fallos, era bastante estable, pero debido al uso de librerías no "estándar" del proyecto GNOME (como Bonobo, OAF, etc), no se ha podido realizar una liberación oficial. A finales de este año, con la liberación de GNOME 1.4 veremos esta primera liberación estable y se espera que GNOME-DB forme parte del paquete "extra" de aplicaciones que acompañará a GNOME 1.4, a finales de este año.

Además, cada vez más y más gente se acerca a GNOME-DB con muchas ganas de ayudar a hacer posible este proyecto, algo que ha significado un empuje importante en el desarrollo, especialmente con la entrada de los desarrolladores de GNU Enterprise. No sólo han decidido usar GDA par su proyecto, sino que están colaborando activamente en su desarrollo.

Así pues, el futuro de GNOME-DB se muestra esperanzador y su inclusión en el proyecto GNOME Office no hace más que dar un nuevo empujón hacia la consecución de sus objetivos.

GNOME-DB sigue en continuo desarrollo, y entre las novedades que se esperan para un futuro cercano se encuentran una buena cantidad de nuevos proveedores (MS Access, Sybase, IBM DB2, así como un entorno integrado de desarrollo basado en Glade para el desarrollo de aplicaciones de acceso a datos. Además, continúan añadiéndose enlaces con nuevos lenguajes, como por ejemplo Perl, Pascal, Python, Java....

BIBLIOGRAFÍA

- http://translate.google.com.ar/translate?hl=es&sl=en&u=http://www.h2database.com/&sa=X&oi=translate&resnum=1&ct=result&prev=/search%3Fq%3DH2%26hl%3Des%26sa%3DG
- http://translate.google.com.ar/translate?hl=es&sl=en&u=http://www.innodb.com/&sa=X&oi=translate&resnum=2&ct=result&prev=/search%3Fq%3Dinnodb%26hl%3Des%26sa%3DG
- http://64.233.179.104/translate_c?hl=es&sl=en&u=http://dba.openoffice.org/&prev=/search%3Fq%3Dopenoffice%2Bbase%26hl%3Des%26client%3Dfirefox-a%26rls%3Dorg.mozilla:es-ES:official%26sa%3DX&usg=ALkJrhiy5aPnzSizaKFTYz6N0HaQHXe4Sw
- http://es.wikipedia.org/wiki/H2
- http://www.mysql.com/company/legal/licensing/.
- http://www.firebirdnews.org/docs/fb2min_es.html
- http://translate.google.com.ar/translate?hl=es&sl=en&u=http://www.stayguard.com/product_vulcan.html&sa=X&oi=translate&resnum=4&ct=result&prev=/search%3Fq%3Dvulcan%2Bdatabase%26hl%3Des%26sa%3DX
- http://es.wikipedia.org/wiki/Firebird
- www.firebirdsql.org
- http://dev.mysql.com/doc/refman/5.0/es/ndbcluster.html
- http://www.mysql.com/products/maxdb.
- http://bugs.mysql.com
- http://www.netpecos.org/docs/mysql_postgres/x15.html
- http://translate.google.com.ar/translate?hl=es&sl=en&u=http://www.openoffice.org/product/base.html&sa=X&oi=translate&resnum=3&ct=result&prev=/search%3Fq%3Dopenoffice%2Bbase%26hl%3Des%26client%3Dfirefox-a%26rls%3Dorg.mozilla:es-ES:official%26sa%3DX
- http://translate.google.com.ar/translate?hl=es&sl=en&u=http://conference2005.kde.org/cfp/develconf/jaroslaw.staniek
- http://www.gnome.org/projects/gnome-db/
- http://www.gnome.org
- http://barrapunto.org/gnome/

- http://www.firebirdsql.org/index.php?op=files&id=engine
- http://www.arcoe.es/2008/05/24/hsqldb-instalacion-y-uso/