ANALISIS COMPARATIVO DE BASES DE DATOS DE CODIGO ABIERTO VS CODIGO CERRADO (DETERMINACION DE INDICES DE COMPARACION)

ING. DIEGO JAVIER BURBANO PROAÑO

Quito-Ecuador, 5 de Mayo del 2006
dburbano@yahoo.com

Pongo a su disposición un análisis comparativo de bases de datos de código abierto vs bases de datos de código cerrado, este análisis pretende dar a los desarrolladores, gerentes y usuarios un documento que les permita realizar una evaluación entre motores de base de datos tanto libres como propietarios de manera que a la hora de elegir entre uno u otro tomen en cuenta puntos importantes.
Cualquier comentario o sugerencia, bienvenida.

Resumen
Esta monografía realiza un estudio comparativo de base de datos libres como Mysql versus bases de datos propietarias como Oracle, la comparación se da en la versión 5.0.18 de Mysql y la versión 10g de Oracle.

Se analizan varios puntos de comparación como soporte, rendimiento, funcionalidad, sistemas operativos soportados, interfaz de desarrollo, conectores, características de acceso a datos, tipos de datos, seguridades, tipos de almacenamiento.

Con este estudio los gerentes y desarrolladores pueden decidir según su proyecto que motor de base de datos usar de acuerdo a sus necesidades, presupuesto o afinidad.
Este estudio se realiza entre las dos bases de datos descritas, pero puede ser utilizado para comparar otras bases de datos existentes.

Es necesario en los estudios de evaluación de software tomar en cuenta las herramientas open source para que el usuario final, desarrolladores y gerentes puedan ver las ventajas y desventajas entre los productos propietarios y libres.

Indice

Indice de Gráficos

Indice de tablas

Anexos

Capítulo 1.- Fundamentos de bases de datos

1.1 Introducción a las bases de datos

Antes de las bases de datos se utilizaban archivos secuenciales como almacenes de datos. Estos daban un acceso muy rápido pero sólo de forma secuencial (para acceder a una posición, se debía recorrer el archivo entero). Más tarde aparecieron los archivos indexados, donde el acceso ya podía ser aleatorio (acceder de una vez a la posición deseada del archivo).

El sistema de **archivos**[1] era el sistema más común de almacenamiento de datos. Para compartir los datos entre varias máquinas surgió el **NFS**[2], y más tarde para evitar fallos en los sistemas de archivo aparecieron los sistemas **RAID**[3] (mirror).
Pero los programas y datos cada vez eran más complejos y grandes. Se requería de un almacenamiento que garantizara un cierto número de condiciones y que permitiera operaciones complejas sin que se violaran estas restricciones. Además cada usuario que accediera a los datos debía tener su trabajo protegido de las operaciones que hicieran el resto de usuarios.
Respondiendo a estas necesidades, surgieron las bases de datos jerárquicas donde los datos se situaban siguiendo una jerarquía.
Las bases de datos jerárquicas tenían el problema que los accesos a los datos eran unidireccionales, y era más complicado hacer el camino inverso (pero posible, aunque el tiempo de cálculo era mayor). Por ejemplo, era fácil saber que cuentas tenía un cliente, pero no tanto saber de que cliente era una cierta cuenta.
Para dar absoluta libertad a las relaciones entre tablas surgieron las bases de datos relacionales (**RDBMS**[4]).
Las RDBMS trajeron dos cosas muy importantes: las propiedades **ACID**[5] y un lenguaje común de acceso a los datos: **SQL**[6].
Las propiedades ACID son:

---

[1]Para más información: http://es.wikipedia.org/wiki/Sistema_de_archivos
[2]**Network file system**: es un sistema de archivos distribuido para un entorno de red de área local. Posibilita que distintos sistemas conectados a una misma red accedan a archivos remotos como si se tratara de locales. Originalmente desarrollado por Sun Microsystems.
[3]**Redundant Array Of Independent/Inexpensive Disks**: este término hace referencia a un conjunto de discos redundantes independientes, es utilizado para mejorar el rendimiento, la tolerancia fallos y errores en los discos, así como también mejora la integridad de los datos. http://es.wikipedia.org/wiki/RAID
[4]**Relational Data Base Management System**: Proporciona un ambiente adecuado para gestionar una base de datos.
[5]Atomicidad, Coherencia, aislamiento y Durabilidad.
[6]**Structured Query Language**: Es un lenguaje de acceso a las bases de datos, permite especificar todas las operaciones sobre la base de datos como por ejemplo: Inserción, Borrado, Actualización. Utiliza características de álgebra y cálculo relacional permitiendo de esta forma realizar consultas a la base de datos de forma sencilla. Más información en: http://es.wikipedia.org/wiki/SQL

- **Atomicidad**. Cada transacción del usuario debe tratarse de forma atómica. O se ejecuta todo o nada. En todo sistema la información es muy importante y no es posible realizar una **transacción**[7] a medias. Una transacción se ejecuta exactamente una vez y tiene carácter "atómico" (de subdivisión), es, decir, el trabajo se realiza en su totalidad o no se realiza en ningún caso.
- **Consistencia**. Las transacciones han de cumplir las restricciones definidas dentro la base de datos. Si no las pueden cumplir, se evita su ejecución. De esta forma se conserva la integridad y coherencia de los datos.
- **Aislamiento**. Una transacción es una unidad de aislamiento, permitiendo que transacciones concurrentes se comporten como si cada una fuera una única transacción que se ejecuta en el sistema. Las transacciones alcanzan el nivel más alto de aislamiento cuando se pueden serializar. En este nivel, los resultados obtenidos de un conjunto de transacciones concurrentes son idénticos a los obtenidos mediante la ejecución en serie de las transacciones.
- **Durabilidad**. Una vez se ha completado la transacción, los resultados de la misma han de ser permanentes y sobrevivir a posibles caídas del sistema o la base de datos.

Debido a que las RDBMS tienen que soportar todas estas propiedades, nunca serán tan rápidas como trabajar directamente sobre archivos, aunque internamente trabajen sobre ellos. La mayoría de desarrolladores prefieren hoy en día sacrificar la velocidad por las funcionalidades.

1.2 Qué son las Bases de Datos?

Según Henry F. Korth autor del libro "Fundamentos de Bases de Datos" se define una base de datos como una serie de datos organizados y relacionados entre sí, y un conjunto de programas que permitan a los usuarios acceder y modificar esos datos.
Las bases de datos proporcionan la infraestructura requerida para los sistemas de apoyo a la toma de decisiones y para los sistemas de información estratégicos, ya que estos sistemas explotan la información contenida en las bases de datos de la organización para apoyar el proceso de toma de decisiones o para lograr ventajas competitivas. Por este motivo es importante conocer la forma en que están estructuradas las bases de datos y su manejo.
Uno de los propósitos principales de un sistema de base de datos es proporcionar a los usuarios una visión abstracta de los datos. Es decir, el sistema esconde ciertos detalles de cómo se almacenan y mantienen los datos.

1.3 Porqué utilizar una base de datos?

Los sistemas tradicionales se denominaban sistemas orientados hacia procesos, debido a que, en ellos, se ponía el énfasis en los tratamientos que reciben los datos,

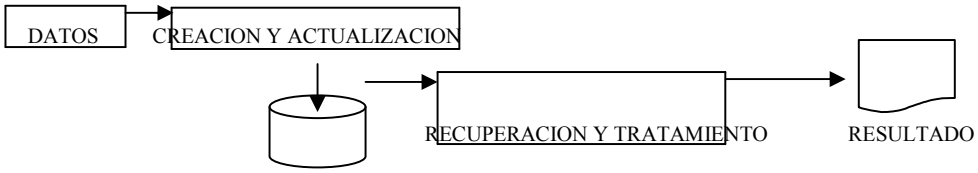
---

[7]**Una transacción** es un conjunto de procesos que se ejecutan uno después del otro, este conjunto de procesos que deben ejecutarse una sola vez en forma completa, si algún subproceso falla, lo realizado anteriormente debe reversarse para que los datos no se alteren, a este comportamiento se lo denomina Todo o nada.

los cuales se almacenan en archivos que son diseñados para una determinada aplicación.
Este planteamiento produce además de una ocupación inútil de memoria, un aumento de los tiempos de procesos, al repetirse los mismos controles y operaciones en los distintos archivos. Pero mas graves todavía son las inconsistencias que a menudo se presentan en estos sistemas, debido a que la actualización de los mismos datos, cuando estos se encuentran es más de un archivo, no se puede realizar de forma simultánea en todos ellos.

Con el fin de resolver estos problemas y de lograr una gestión mas racional del conjunto de datos, surge un nuevo enfoque que se apoya sobre una "base de datos" en la cual los datos son recogidos y almacenados, al menos lógicamente, una sola vez, con independencia de los tratamientos.

Gráfico # 1: Diagrama del flujo de la información al utilizar una base de datos



Fuente: Análisis y diseño detallado de aplicaciones informáticas de gestión
Autor: Diego Burbano

Según este enfoque se logran los siguientes cambios:

- **Independencia** de los datos respecto a los tratamientos y viceversa, lo que evita el importante esfuerzo que origina la reprogramación de las aplicaciones cuando se producen cambios en los datos.
- **Coherencia** de los resultados, con lo que se elimina el inconveniente de las divergencias en los resultados debidas a actualizaciones no simultaneas en todos los archivos.
- **Mejor disponibilidad** de los datos para el conjunto de los usuarios junto con una mayor transparencia respecto a la información existente.
- **Mayor valor informativo**, debido a que los distintos elementos están interrelacionados.
- **Documentación** de la información mejor y más normalizada, la cual está integrada con los datos.
- **Mayor eficiencia** en la recuperación, validación y entrada de los datos al sistema.

Además al momento de tomar una decisión hay que tomar en cuenta posibles inconvenientes que es necesario valorar antes de tomar una decisión relativa a un cambio en la orientación de sistema de información.

- Instalación costosa.
- Personal especializado.

- Falla de rentabilidad a corto plazo.
- Desfase entre teoría y práctica.

1.4 Componentes principales de una base de datos

1.4.1 Hardware.

El hardware se refiere a los dispositivos de almacenamiento en donde reside la base de datos, así como a los dispositivos periféricos (unidad de control, canales de comunicación, etc.) necesarios para su uso.

1.4.2 Software.

Está constituido por un conjunto de programas que se conoce como Sistema Manejador de Base de Datos (DMBS: Data Base Management System). Este sistema maneja todas las solicitudes formuladas por los usuarios a la base de datos.

1.4.3 Usuarios.

Existen tres clases de usuarios relacionados con una Base de Datos:

- El programador de aplicaciones, quien crea programas de aplicación que utilizan la base de datos.
- El usuario final, quien accesa la Base de Datos por medio de un lenguaje de consulta o de programas de aplicación.
- El administrador de la Base de Datos (DBA DataBase Administrator), quien se encarga del control general del Sistema de Base de Datos.

1.5 Ventajas en el uso de Bases de Datos.

Según Korth estas son las principales ventajas del uso de las bases de datos:

1.5.1 Globalización de la información.

Permite a los diferentes usuarios considerar la información como un recurso corporativo que carece de dueños específicos.

1.5.2 Eliminación de información redundante.

Información Duplicada que puede generar inconsistencias en la base de datos.

1.5.3 Eliminación de información inconsistente.

Si el sistema esta desarrollado a través de archivos convencionales, una cancelación de compra por ejemplo deberá operarse tanto en el archivo de facturas del Sistema de Control de Cobranza como en el archivo de facturas del Sistema de Comisiones.

1.5.4 Permite compartir información.

Varios sistemas o usuarios pueden utilizar una misma entidad.

1.5.5 Permite mantener la integridad en la información.

Solo se almacena la información correcta.

1.5.6 Independencia de datos y tratamiento.

La independencia de datos implica un divorcio entre programas y datos; es decir, se pueden hacer cambios a la información que contiene la base de datos o tener acceso a la base de datos de diferente manera, sin hacer cambios en las aplicaciones o en los programas. Lo que implica menor costo de mantenimiento.

1.5.7 Restricciones de seguridad.

En lo que tiene que ver con el acceso de usuarios a los datos y operaciones sobre los datos.

1.6 El sistema gestor de base de datos (DBMS)

El DBMS es un conjunto de programas que se encargan de manejar la creación y todos los accesos a las bases de datos.
El objetivo principal del sistema gestor de base de datos es proporcionar una forma de almacenar y recuperar la información de una base de datos de manera que sea tanto práctica como eficiente.
Se compone de un lenguaje de definición de datos (**DDL**: Data Definition Language), de un lenguaje de manipulación de datos (**DML**: Data Manipulation Language) y de un lenguaje de consulta (**SQL** : Structured Query Language).

1.6.1 El lenguaje de definición de datos (DDL)

Es utilizado para describir todas las estructuras de información y los programas que se usan para construir, actualizar e introducir la información que contiene una base de datos. El DDL permite al administrador de la base especificar los elementos de datos que la integran , su estructura y las relaciones que existen entre ellos, las reglas de integridad, los controles a efectuar antes de autorizar el acceso a la base.
Por ejemplo la siguiente instrucción de lenguaje sql define la tabla cuenta:

```
Create table cuenta
(
                numero_cuenta char(10),
                saldo integer
)
```

La ejecución de la instrucción DDL anterior crea la tabla cuenta. Además, actualiza un conjunto especial de tablas denominado **diccionario de datos**.

Un diccionario de datos contiene metadados, es decir, datos acerca de los datos.
Los valores de los datos almacenados en la base de datos deben satisfacer ciertas restricciones de consistencia de la información. Por ejemplo, supóngase que el saldo contable de una cuenta no puede ser mayor a 5.000,00 usd. El DDL proporciona facilidades para especificar tales restricciones. Los sistemas de base de datos comprueban estas restricciones cada vez que se actualiza la base de datos.

1.6.2 El lenguaje de manipulación de datos (DML)

Es utilizado para escribir programas que crean, actualizan y extraen información de las bases de datos. Siempre de acuerdo con las especificaciones y las normas de seguridad dictadas por al administrador.
Un lenguaje de manipulación de datos es un lenguaje que permite a los usuarios acceder o manipular los datos organizados mediante el modelo de datos apropiado. Hay dos tipos básicamente:

- **DMLs procedimentales**. Requieren que el usuario especifique qué datos se necesitan y cómo obtener esos datos.
- **DMLs declarativos** (o no procedimentales). Requiere que el usuario especifique qué datos se necesitan sin especificar cómo obtener esos datos.

Una consulta es una instrucción de solicitud para recuperar información. La parte de un DML se llama lenguaje de consultas.

Ejm:
Select nombre, dirección
from cliente
where id_cliente = 2

La secuencia conceptual de operaciones que ocurren para acceder cierta información que contiene una base de datos es la siguiente:

- El usuario solicita cierta información contenida en la base de datos.
- El DBMS intercepta este requerimiento y lo interpreta.
- El DBMS realiza las operaciones necesarias para acceder y/o actualizar la información solicitada.

1.7 El administrador de la base de datos (DBA)

El DBA es la persona encargada de definir y controlar las bases de datos corporativas, además proporciona asesoría a los usuarios y ejecutivos que la requieran.
Las principales funciones del administrador son:

- **La estructura de la base de datos** en el sentido de determinar que información va a ser necesario almacenar en la misma, después de haber analizado los requerimientos de los usuarios.
- **Los estándares** por los que se va a regir la organización en cuanto a documentación de la base de datos, metodologías de diseño de la misma.
- **La estrategia** de transición del sistema existente al nuevo sistema de

información soportado en una base de datos. El DBA deberá decidir sobre la posible puesta en marcha en paralelo del nuevo sistema con el antiguo, las fases de implantación del mismo, los controles necesarios. Todas estas decisiones habrán de tomarse en función de los objetivos marcados y de forma que se cause el mínimo trastorno a los usuarios.

- **Los permisos de explotación y uso**, es decir, establecer la normativa necesaria para la utilización de la base de datos, el modo de solicitar el acceso a la misma, su actualización, etc.
- **Los aspectos relativos a la seguridad**, incluidos los procedimientos de control y las auditorias.
- **Mantenimiento rutinario**. Algunos ejemplos de actividades rutinarias que el administrador de la base de datos debe revisar que se cumplan son:
    - Copia de seguridad periódica de la base de datos, bien sobre cinta o sobre servidores remotos, para prevenir la pérdida de datos en caso de desastres o imprevistos.
    - Asegurarse de que exista suficiente espacio libre en el disco duro para las operaciones normales y aumentar el espacio en el disco en caso de ser necesario.
    - Supervisión del los trabajos que se ejecuten sobre la base de datos y sobre todo asegurarse que el rendimiento no se degrade por tareas muy costosas realizadas por algunos usuarios.

Para que el DBA pueda cumplir con todas estas funciones deberá interactuar con todo el personal de la organización como se explica en la figura:

Gráfico # 2 : Diagrama de interacción del Dba con los usuarios

Fuente : Análisis y diseño detallado de aplicaciones informáticas de gestión

Autor: Diego Burbano

1.8 Modelo de base de datos

Además de la clasificación por la función de las bases de datos, éstas también se pueden clasificar de acuerdo a su modelo de administración de datos.
Un modelo de datos es básicamente una "descripción" de algo conocido como

contenedor de datos (algo en donde se guarda la información), así como de los métodos para almacenar y recuperar información de esos contenedores. Los modelos de datos no son cosas físicas; son abstracciones que permiten la implementación de un sistema eficiente de base de datos, por lo general se refieren a algoritmos, y conceptos matemáticos.

Algunos modelos con frecuencia utilizados en las bases de datos:

1.8.1 Bases de datos Jerárquicas

Estas son bases de datos que, como su nombre indica, almacenan su información en una estructura jerárquica. En este modelo los datos se organizan en una forma similar a un árbol (visto al revés), en donde un nodo padre de información puede tener varios hijos. El nodo que no tiene padres se le conoce como raíz, y a los nodos que no tienen hijos se les conoce como hojas.
Una de las principales limitaciones de este modelo, es su incapacidad de representar eficientemente la redundancia de datos.

1.8.2 Bases de datos de red

Este es un modelo ligeramente distinto del jerárquico, en donde su diferencia fundamental es la modificación del concepto de un nodo, permitiendo que un mismo nodo tenga varios padres (algo no permitido en el modelo jerárquico).
Fue una gran mejora con respecto al modelo jerárquico, ya que ofrecía una solución eficiente al problema de redundancia de datos, pero aun así, la dificultad que significa administrar la información en una base de datos de red, ha significado que sea un modelo utilizado en su mayoría por programadores más que por usuarios finales.

1.8.3 Bases de datos relacionales

Este es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente. Tras ser postuladas sus bases en 1970 por **Edgar Frank Codd**[8], de los laboratorios **IBM**[9] en San José California, no tardó en consolidarse como un nuevo paradigma en los modelos de base de datos. Su idea fundamental es el uso de "relaciones". Estas relaciones podrían considerarse en forma lógica como conjuntos de datos llamados "tuplas". Pese a que esta es la teoría de las bases de datos relacionales creadas por Edgar Frank Codd, la mayoría de las veces se conceptualiza de una manera más fácil de imaginar. Esto es pensando en cada relación como si fuese una tabla que está compuesta por registros (las filas de

---

8 Más información en : http://es.wikipedia.org/wiki/Edgar_Frank_Codd

9 Más información en : http://es.wikipedia.org/wiki/IBM

una tabla), que representarían las tuplas, y campos (las columnas de una tabla).
En este modelo, el lugar y la forma en que se almacenen los datos no tienen relevancia (a diferencia de otros modelos como el jerárquico y el de red). Esto tiene la considerable ventaja de que es más fácil de entender y de utilizar para un usuario casual de la base de datos. La información puede ser recuperada o almacenada por medio de "consultas" que ofrecen una amplia flexibilidad y poder para administrar la información.
El lenguaje más común para construir las consultas a bases de datos relacionales es SQL, Structured Query Language o Lenguaje Estructurado de Consultas, un estándar implementado por los principales motores o sistemas de gestión de bases de datos relacionales.
Las bases de datos relacionales pasan por un proceso al que se le conoce como **Normalización de una base de datos**[10].
En el modelo relacional se usan un grupo de tablas para representar los datos y las relaciones entre ellos. Cada tabla esta compuesta por varias columnas, y cada columna tiene un nombre único.

Tabla # 1 : Tabla de una base de datos relacional

Id_cliente	nombre_cliente	dirección_cliente	telefono_cliente
1	Diego Burbano	Av. 6 de Diciembre	2266171
2	Liliana Cisneros	Rio Coca	2439481
3	Pablo Monsserrath	Av.De los Granados	2546789
4	Javier Moncayo	12 de Octubre	2435987

numero_cuenta	estado_cuenta	saldo_contable	saldo_disponible
12005190253	Activa	5.689,15	5.000,15
12556456054	Cerrada	389,15	389,15
12489794549	Activa	89,15	89,15
12454894984	Activa	10,15	10,15

id_cliente	numero_cuenta
1	12005190253
1	12556456054
3	12489794549
4	12454894984

Autor : Diego Burbano

[10]El proceso de normalización consiste en aplicar una serie de reglas a las relaciones obtenidas luego de haber pasado del modelo Entidad Relación al modelo relacional.

Para más información :

http://es.wikipedia.org/wiki/Normalizaci%C3%B3n_de_una_base_de_datos

La Tabla # 1 presenta un ejemplo de base de datos relacional consistente en tres tablas, la primera muestra los clientes de un banco, la segunda las cuentas y la tercera las cuentas que pertenecen a cada cliente.

El modelo relacional es un ejemplo de un modelo basado en registros. Los modelos basados en registros se denominan así porque la base de datos se estructura en registros de formato fijo de varios tipos. Cada tabla contiene registros de un tipo particular. Cada tipo de registro define un número fijo de campos, o atributos.

El modelo de datos relacional se encuentra a un nivel de abstracción inferior al modelo de datos **E-R**[11]. Los diseños de bases de datos a menudo se realizan en el modelo E-R y después se traducen al modelo relacional.

1.8.3.1 El modelo Entidad Relación

Este modelo está basado en una percepción del mundo real, que consta de una colección de objetos básicos, llamados entidades, y de relaciones entre estos objetos. Una entidad es todo aquello que exhibe autonomía, diferenciación y existencia en el mundo real que es distinguible de otros objetos. Por ejemplo cada persona es una entidad y las cuentas bancarias pueden ser consideradas entidades.

Las entidades se describen en la base de datos mediante atributos. Por ejemplo los atributos nombre_cliente, dirección_cliente pueden describir una entidad cliente.

Adicionalmente id_cliente, se usa para identificar unívocamente a los clientes (dado que puede existir en el mundo real un cliente con el mismo nombre, dirección), de esta manera de asigna un identificador único para cada cliente. Una relación es una asociación entre varias entidades, Por ejemplo una relación puede asociar un cliente con cada cuenta que tiene.

El conjunto de todas las entidades del mismo tipo, y el conjunto de todas las relaciones del mismo tipo, se denominan respectivamente conjunto de entidades y conjunto de relaciones.

La estructura lógica general de una base de datos se puede expresar gráficamente mediante un diagrama E-R, q consta de los siguientes componentes:

- Rectángulos, que representan conjuntos de entidades.
- Elipses, que representan atributos.
- Rombos, que representan relaciones entre conjuntos de entidades.
- Líneas, que unen los atributos con los conjuntos de entidades y los conjuntos de entidades con las relaciones.

---

[11] **Entidad Relación**.- es un diagrama que consiste en objetos relacionados entre sí, que permiten identificar las entidades, atributos y relaciones que van a formar parte de la base de datos.

Cada componente se etiqueta con la entidad o relación que representa.

1.9 Bases de datos Orientadas a Objetos

Este modelo, bastante reciente, y propio de los modelos informáticos orientados a objetos, trata de almacenar en la base de datos los objetos completos (estado y comportamiento).
Las limitaciones impuestas por el modelo relacional han surgido como obstáculos. En consecuencia, los investigadores de bases de datos inventaron nuevos modelos de datos que resuelven las limitaciones del modelo de datos relacional.

Las aplicaciones de bases de datos tradicionales consisten en tareas de procesamiento de datos, tales como la banca, seguros, ventas, gestión de recursos humanos. Dichas aplicaciones presentan conceptualmente tipos de datos simples. Los elementos de datos básicos son registros bastante pequeños y cuyos campos son atómicos, es decir, no contienen estructuras adicionales y en los que se cumple la primera forma normal.

En los últimos años, la demanda ha incrementado la forma de abordar los tipos de datos más complejos. Por ejemplo, un conjunto de direcciones. Mientras una dirección completa puede ser vista como un elemento de datos atómico de tipo cadena de caracteres, esta forma de verlo escondería detalles como la calle, barrio, sector, provincia que pueden ser interesantes para una consulta. Por otra parte si una dirección se la presenta dividiéndole en varias componentes (calle, barrio, sector, provincia) las consultas escritas serían más complicadas, pues tendría que mencionar cada campo. Una alternativa mejor es permitir tipos de datos estructurados, que admiten un tipo dirección con subpartes calle, barrio, sector.

Las bases de datos orientadas a objetos (BDOO) almacenan y manipulan información que puede ser digitalizada por objetos, proporcionan una estructura flexible con acceso ágil, rápido, con gran capacidad de modificación.

Además combina las mejores cualidades de los archivos planos, las bases jerárquicas y relacionales. La BDOO representan el siguiente paso en la evolución de las bases de datos para soportar análisis, diseño y programación orientada a objetos.

Las BDOO permiten el desarrollo y mantenimiento de aplicaciones complejas ya que se puede utilizar un mismo modelo conceptual y así aplicado al análisis, diseño y programación, esto reduce el problema entre los diferentes modelos a través de todo el ciclo de vida, con un costo significativamente menor.

Las BDOO ofrecen un mejor rendimiento de la máquina que las bases de datos relacionales, para aplicaciones ó clases con estructuras complejas de datos. Sin embargo, las BDOO coexistirán con las bases de datos relacionales como una forma de estructura de datos dentro de una BDOO.
Una base de datos orientada ha objetos es una base de datos que incorpora todos los conceptos importantes de la programación orientada ha objetos:

- **Encapsulación.–** Ocultar datos del resto de los datos, impidiendo así accesos

incorrectos a conflictos.

- **Herencia.-** Reusabilidad del código.
- **Polimorfismo-** Sobrecarga de operadores o de métodos.

Se esta trabajando en SQL3, que es el estándar de SQL92 extendido que soportaría los nuevos conceptos orientados a objetos y tendría compatibilidad con SQL92.

1.9.1 Conceptos fundamentales

1.9.1.1 Objeto .-

Es cualquier cosa real ó abstracta acerca de la cual almacenamos datos y los métodos que controlan dichos datos. Por ejm. En una empresa EMPLEADO se aplica a los objetos que son personas empleadas por alguna organización alguna INSTANCIA podría ser Juan Pérez, Javier Proaño.

1.9.1.2 Tipo de Objeto.-

Es una categoría de objeto Ejm: EMPLEADO. Un objeto es una instancia de un tipo de objeto. PERSONA (Juan Perez).

1.9.1.3 Encapsulamiento.-

Es el resultado (o acto) de ocultar los detalles de implantación de un objeto respecto de su usuario.

1.9.1.4 Una solicitud.-

Invoca una operación específica, con uno o más objetos como parámetros. Es decir, es para que se lleve acabo la operación indicada y que se produzca el resultado. En consecuencia las implantaciones se refieren a los objetos como solicitudes.

1.9.1.5 Clase.-

Es una implantación de un tipo de objeto. Especifica una estructura de datos y los métodos operativos permisibles que se aplican a cada uno de sus objetos.

1.9.1.6 Herencia.-

Una clase implanta el tipo de objeto. Una subclase hereda propiedades de su clase padre, una subclase puede heredar la estructura y los métodos o algunos de los métodos.

En las BDOO los datos están encapsulados y se dice que estos son activos más que pasivos; debido a que por ejemplo: La clase mayor detecta si tiene un hijo (objeto) más o uno menos, es por esto que se dice que están activos ya que cuentan los hijos u objetos que tiene.

1.9.2 Características Fundamentales

1.9.2.1 Abstracción.-

Denota las características esenciales de un objeto que lo distinguen de todos los demás tipos objeto, y proporciona así fronteras conceptuales nítidamente definidas respecto a la perspectiva del observador. Una abstracción se centra en la visión extrema de un objeto, y, por lo tanto sirve para separar el comportamiento esencial de un objeto de su implantación.

1.9.2.2 Modularidad.-

Se basa en el concepto de fragmentación de los programas en componentes individuales para reducir su complejidad en algún grado, y para crear además una serie de fronteras bien definidas y documentadas dentro del programa, donde estas fronteras o interfaces tienen un incalculable valor cara a la comprensión del programa.

1.9.2.3 Jerarquía.-

Una clasificación u ordenación de abstracciones.

1.9.2.4 Tipos.-

Es un conjunto de objetos que tienen un mismo comportamiento (comparten una misma funcionalidad) que se puede observar desde afuera.

1.9.2.5 Genericidad.-

Permite construir clases genéricas para otras clases.

1.9.2.6 Objetos Complejos.-

Están construidos mediante algunos más simples ó mediante la aplicación de constructores a ellos. Los objetos más simples son objetos como: Integer, Carácter, String de bytes de cualquier longitud, booléanos, punto flotante y algunos pueden ser de tipo atómico.

1.9.3 Bases de Datos Orientadas a Objetos

1.9.3.1 Que es la O.O.

En la Orientación a objetos, el conocimiento se descentraliza en todos los objetos que lo componen, cada objeto sabe hacer lo suyo y no le interesa saber cómo el vecino hace su trabajo, pero sabe que lo hace y qué es lo que puede hacer.

“ La orientación a objetos proporciona una solución que conduce a un universo de

objetos 'bien educados' que se piden de manera cortés, concederse mutuamente sus deseos." Dan Ingalls de Smalltalk.

1.9.3.2 Por qué O.O.

La meta es dejar la etapa en la que la construcción del software es una labor de artesanos, y pensar en la etapa en la que se pueda tener fábricas de software, con gran capacidad de reutilización de código y con metodologías eficientes y efectivas que se apliquen al proceso de producción.

1.9.3.3 Que es una B.D.O.O

A finales de los 80's aparecieron las primeras BDOO, es una base de datos inteligente. Soporta el paradigma orientado a objetos almacenando datos y métodos, y no sólo datos. Está diseñada para ser eficaz, desde el punto de vista físico, para almacenar objetos complejos. Evita el acceso a los datos; esto es mediante los métodos almacenados en ella. Es más segura ya que no permite tener acceso a los datos (objetos); esto debido a que para poder entrar se tiene que hacer por los métodos que haya utilizado el programador.

1.9.3.4 Arquitectura de una B.D.O.O

Tabla # 2: Tabla de algunas BDOO existentes y sus proveedores

Producto	**Proveedor**
Gemstone	Servio Corporation, Alameda,CA
Itasca	Itasca Systems,Inc.,Minneapolis,MN
Objectivity	Objectivity,Menlo Park,Ca
Object Store	Object Design,Inc.,Burlington,MA
Ontos	Ontos Inc.,Bellerica,MA
Versant	Versant Object Technology,Menlo Park,CA

Fuente: http://www.elrinconcito.com/articulos/BaseDatos/BasesDatos.htm
Autor : Diego Burbano

Los primeros se diseñaron como una extensión de los lenguajes de programación Smalltalk ó C++. El DML (lenguaje para la manipulación de datos; también conocido como DML) y el DDL construían un lenguaje OO común.

Algunas características son independientes de la arquitectura fundamental de una BDOO pero son comunes a la mayoría de ellas:

- **Versiones.-** La mayoría de los sistemas de bases de datos sólo permiten que exista una representación de un ente de la base de datos dentro de esta. Las versiones permiten que las representaciones alternas existan en forma simultánea.
- **Transacciones compartidas**.- Las transacciones compartidas soportan grupos de usuarios en estaciones de trabajo, los cuales desean coordinar sus esfuerzos en tiempo real, los usuarios pueden compartir los resultados

intermedios de una base de datos. La transacción compartida permite que varias personas intervengan en una sola transacción.

1.9.3.5 Impacto de la orientación a objetos en la Ingeniería de Software

En las BDOO, la organización "Gestión Manejadora de Datos Objeto (**ODMG**)" representa el 100% de las BDOO industriales y ha establecido un estándar de definición (ODL - Lenguaje de Definición de datos) y manipulación (OQL - Lenguaje de consulta) de bases de datos equivalente a SQL.

Respecto a las relacionales, todas (Oracle, Informix, etc.) están añadiendo en mayor o menor grado algunos aspectos de la orientación a objetos. **ANSI**(Instituto Nacional Estadounidense de Estándar), por su parte, está definiendo un SQL-3 que incorpora muchos aspectos de la orientación a objetos. El futuro del **SQL-3** es sin embargo incierto, ya que ODMG ha ofrecido a ANSI su estándar para que sirva de base para un nuevo SQL, con lo que solo habría un único estándar de base de datos.

El grupo ODMG (Grupo Manejador de Datos Objeto) nació de un grupo más grande, llamado "Grupo Manejador de Objetos (OMG)", donde están representados todas las cosas con alguna influencia en el sector. Este grupo esta definiendo un estándar universal por objetos. Este estándar permitirá que un objeto sea programado en cualquier lenguaje y sistema operativo. Esto facilitará enormemente el desarrollo de sistemas abiertos cliente-servidor.

1.9.3.6 Ventajas en BDOOs.

Está su flexibilidad, y soporte para el manejo de tipos de datos complejos. Por ejemplo, en una base de datos convencional, si una empresa adquiere varios clientes por referencia de clientes servicio, pero la base de datos existente, que mantiene la información de clientes y sus compras, no tiene un campo para registrar quién proporcionó la referencia, de qué manera fue dicho contacto, o si debe compensarse con una comisión, sería necesario reestructurar la base de datos para añadir este tipo de modificaciones. Por el contrario, en una BDOO, el usuario puede añadir una "subclase" de la clase de clientes para manejar las modificaciones que representan los clientes por referencia.

La subclase heredará todos los atributos, características de la definición original, además se especializará en especificar los nuevos campos que se requieren así como los métodos para manipular solamente estos campos. Naturalmente se generan los espacios para almacenar la información adicional de los nuevos campos. Esto presenta la ventaja adicional que una BDOO puede ajustarse a usar siempre el espacio de los campos que son necesarios, eliminando espacio desperdiciado en registros con campos que nunca usan.

La segunda ventaja de una BDOO, es que manipula datos complejos en forma rápida y ágilmente. La estructura de la base de datos está dada por referencias (o apuntadores lógicos) entre objetos.

1.9.3.7 Posibles Desventajas

Al considerar la adopción de la tecnología orientada a objetos, la inmadurez del mercado de BDOO constituye una posible fuente de problemas por lo que debe analizarse con detalle la presencia en el mercado del proveedor para adoptar su producto en una línea de producción sustantiva.

El segundo problema es la falta de estándares en la industria orientada a objetos. Sin embargo, el "Grupo Manejador de Objetos" (OMG[12]), es una organización Internacional de proveedores de sistemas de información y usuarios dedicada a promover estándares para el desarrollo de aplicaciones y sistemas orientados a objetos en ambientes de cómputo en red.

La implantación de una nueva tecnología requiere que los usuarios iniciales acepten cierto riesgo. Aquellos que esperan resultados a corto plazo y con un costo reducido quedarán desilusionados. Sin embargo, para aquellos usuarios que planean a un futuro intermedio con una visión tecnológica avanzada, el uso de tecnología orientada a objetos, paulatinamente compensará todos los riesgos.

1.9.3.8 Rendimiento

Las BDOO permiten que los objetos hagan referencia directamente a otro mediante apuntadores suaves. Esto hace que las BDOO pasen más rápido del objeto A al objeto B que las **BDR**, las cuales deben utilizar comandos **JOIN** para lograr esto. Incluso el JOIN optimizado es más lento que un recorrido de los objetos. Así, incluso sin alguna afinación especial, una BDOO es en general más rápida en esta mecánica de caza-apuntadores.

Las BDOO hacen que el agrupamiento sea más eficiente. La mayoría de los sistemas de bases de datos permiten que el operador coloque cerca las estructuras relacionadas entre sí, en el espacio de almacenamiento en disco. Esto reduce en forma radical el tiempo de recuperación de los datos relacionados, puesto que todos los datos se leen con una lectura de disco en vez de varias. Sin embargo, en una BDR, los objetos de la implantación se traducen en representaciones tabulares que generalmente se dispersan en varias tablas. Así, en una BDR, estos renglones relacionados deben quedar agrupados, de modo que todo el objeto se pueda recuperar mediante una única lectura del disco. Esto es automático en una BDOO. Además, el agrupamiento de los datos relacionados, como todas las subpartes de un ensamble, puede afectar radicalmente el rendimiento general de una aplicación. Esto es relativamente directo en una BDOO, puesto que representa el primer nivel de agrupamiento. Por el contrario, el agrupamiento físico es imposible en una BDR, puesto que esto requiere un segundo nivel de agrupamiento: un nivel para agrupar las hileras que representan a los objetos individuales y un segundo para los grupos de hileras que representan a los objetos relacionados.

---

[12]Para más información : http://www.omg.org

1.9.3.9 Características mandatorias de una BDOO

Un sistema de BDOO debe satisfacer dos criterios:

- Debe tener un SGBD
- Debe ser un sistema OO

Por ejemplo: para la extensión posible este debe ser consistente en los actuales cortes de lenguajes de programación OO.

El primer criterio se traduce en 5 características como son:

Persistencia, Manejador de almacenamiento secundario, Concurrencia, Recuperación, y Facilidad de Query.

La Segunda se traduce en 8 características: Objetos Complejos, Identidad del objeto, Encapsulación, Tipos ó Clases, Sobre paso con combinación retrasada, Extensibilidad y Completación Computacional.

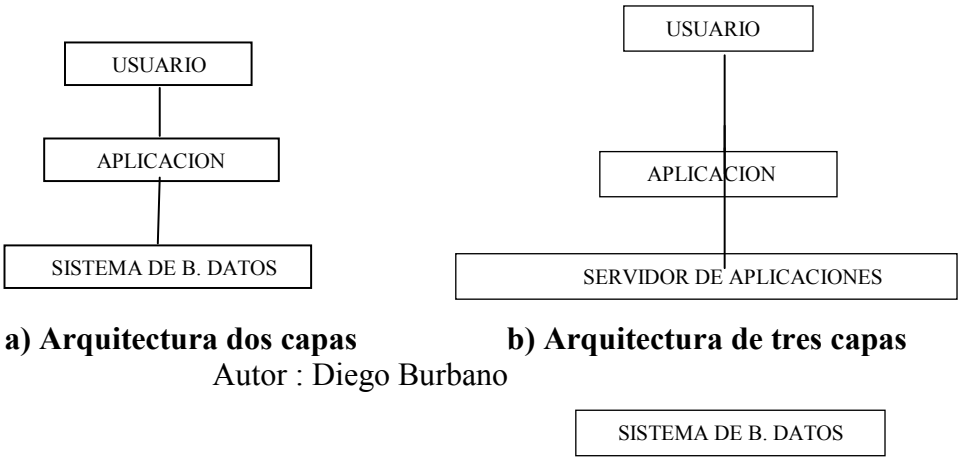
1.10 Bases de datos Distribuidas

Una base de datos distribuida (BDD) es la unión de las bases de datos con Redes distribuidas geográficamente.
La base de datos está almacenada en varias computadoras conectadas en red, (ya sea en el mismo lugar físicamente o distribuidas a lo largo de la red) lo que permite al acceso de datos desde diferentes máquinas. Está manejada por el Sistema de Administración de Datos Distribuida SABDD o Sistema de Gestión de Base de datos distribuida. Son la evolución de los Cliente-Sevidor.
La razón principal detrás de las BDD son los organismos descentralizados. Esto les da la capacidad de unir las bases de datos de cada localidad y acceder así a la información, sin tener todo centralizado en un solo punto. Ejemplo: bancos, cadenas de hoteles, campus de distintas universidades, sucursales de tiendas departamentales, etc.
Los principales problemas que se generan por el uso de la tecnología de bases de datos distribuidas son en lo referente a duplicidad de datos y a su integridad al momento de realizar actualizaciones a los mismos. Además, el control de la información puede constituir una desventaja, debido a que se encuentra diseminada en diferentes localidades geográficas.

1.11 Arquitecturas de Aplicaciones

Los usuarios de la base de datos no están situados actualmente junto al sistema de bases de datos, sino que se conectan a él a través de la red. Se puede diferenciar entonces las máquinas cliente, en donde trabajan los usuarios remotos de la base de datos, y las máquinas servidor en las que se ejecuta el sistema de base de datos.

Gráfico # 3: Diagrama de Arquitecturas



a) Arquitectura dos capas **b) Arquitectura de tres capas**

Autor : Diego Burbano

En una **Arquitectura de dos capas** la aplicación se divide en un componente que reside en la máquina cliente, que llama a la funcionalidad del sistema de base de datos en la máquina servidor mediante instrucciones del lenguaje de consultas.

Los estándares de interfaces de programas de aplicación como **ODBC**[13] y **JDBC**[14] se usan para la interacción entre el cliente y el servidor.

En cambio en una **arquitectura de tres capas**, la máquina cliente actúa simplemente como frontal y no contiene ninguna llamada directa a la base de datos. En su lugar, el cliente se comunica con **un servidor de aplicaciones**, usualmente mediante una interfaz de formularios. El servidor de aplicaciones, a su vez, se comunica con el sistema de base de datos para acceder a ellos.

---

[13]**ODBC** son las siglas de Open DataBase Connectivity, un estándar de acceso a bases de datos desarrollado por Microsoft Corporation, el objetivo de ODBC es hacer posible el acceder a cualquier dato de cualquier aplicación, sin importar qué Sistema Gestor de Bases de Datos (DBMS por sus siglas en Ingles) almacene los datos, ODBC logra esto al insertar una capa intermedia llamada manejador de Bases de Datos, entre la aplicación y el DBMS, el propósito de esta capa es traducir las consultas de datos de la aplicación en comandos que el DBMS entienda. Para que esto funcione tanto la aplicación como el DBMS deben ser compatibles con ODBC, esto es que la aplicación debe ser capaz de producir comandos ODBC y el DBMS debe ser capaz de responder a ellos.

[14]**JDBC** es el acrónimo de Java Database Connectivity, un API (Interfaz de Programación de Aplicaciones) que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java(Java es una plataforma de software desarrollada por Sun Microsystems, de tal manera que los programas creados en ella puedan ejecutarse sin cambios en diferentes tipos de arquitecturas y dispositivos computacionales) independientemente del sistema de operación donde se ejecute o de la base de datos a la cual se accede utilizando el dialecto SQL del modelo de base de datos que se utilice..

La **lógica del negocio** de la aplicación, que establece las acciones a realizar bajo determinadas condiciones, se incorpora en el servidor de aplicaciones, en lugar de ser distribuida a múltiples clientes.

1.12 Criterios de Calidad

1.12.1 Legibilidad

El diseño de una base de datos ha de estar redactado con la suficiente claridad para que pueda ser entendido rápidamente. El lenguaje utilizado debe ser lo suficientemente claro, conciso y detallado para que explique con total claridad el diseño del modelo, sus objetivos, sus restricciones, en general todo aquello que afecte al sistema de forma directa o indirecta.

1.12.2 Fiabilidad

Se trata de realizar un sistema de bases de datos lo suficientemente robusto para que sea capaz de recuperarse frente a errores o usos inadecuados. Se deben utilizar gestores con las herramientas necesarias para la reparación de los posibles errores que las bases de datos pueden sufrir, por ejemplo tras un corte inesperado de luz.

1.12.3 Portabilidad

El diseño deber permitir la implementación del modelo físico en diferentes gestores de bases de datos.

1.12.4 Modificalidad

Ningún sistema informático es estático, las necesidades de los usuarios varían con el tiempo y por lo tanto las bases de datos se deben adaptar a las nuevas necesidades, por lo que se precisa que un buen diseño facilite el mantenimiento, esto es, las modificaciones y actualizaciones necesarias para adaptarlo a una nueva situación.

1.12.5 Eficiencia

Se deben aprovechar al máximo los recursos de la computadora, minimizando la memoria utilizada y el tiempo de proceso o ejecución, siempre que no sea a costa de los requisitos anteriores. En este punto se debe tener en cuenta los gestores cliente / servidor de bases de datos. En muchas ocasiones es más rentable cargar de trabajo al servidor y liberar recursos de los clientes.

1.13 Tendencias Futuras

La explotación efectiva de la información dará ventaja competitiva a las organizaciones.
Las bases de datos orientadas a objetos empleadas para diseño y manufactura asistida por computadora CAD/CAM serán utilizados a un mismo nivel que las Bases se Datos relacionales de la actualidad.

Los lenguajes de consulta (SQL) permitirán el uso del lenguaje natural para solicitar información de la Base de Datos, haciendo más rápido y fácil su manejo.
El uso de las bases de datos distribuidas se incrementará de manera considerable en la medida en que la tecnología de comunicación de datos brinde más facilidades para ello. El uso de bases de datos facilitará y soportará en gran medida a los Sistemas de Información para la Toma de Decisiones

1.14 Arquitecturas

La arquitectura de un sistema de base de datos está influenciada por el sistema informático que soporta la instalación del SGBD, lo que reflejará muchas de las características propias del sistema subyacente en el SGBD.
Las redes de computadores permiten separar tareas en un esquema de clientes y servidores, el procesamiento paralelo dentro del computador permite acelerar algunas de las tareas de la base de datos así como la posibilidad de ejecutar más transacciones por segundo. Las consultas se pueden paralelizar permitiendo así que una consulta se pueda ejecutar por más de un procesador al mismo tiempo, esta característica ha llevado al estudio de las bases de datos paralelas.

La distribución de datos a través de distintos departamentos de una organización permite que ellos residan donde han sido generados (y donde se entiende que son más requeridos); la idea de mantener una copia de estos datos en otros lugares permite que puedan seguir las operaciones sobre los datos aún si alguno de estos sitios sufre algún desastre. El estudio de este tipo de descentralización de los datos lleva al desarrollo de los sistemas de base de datos distribuidos.

1.14.1 SGBD Centralizados

Un sistema de base de datos centralizado es aquel que se ejecuta en un único sistema computacional sin tener, para tal efecto, que interactuar con otros computadores. El rango de estos sistemas comprende desde los sistemas de bases de datos monousuario ejecutándose en computadores personales hasta los sistemas de bases de datos ejecutándose en sistemas de alto rendimiento.
Normalmente los sistemas de base de datos monousuarios no suelen proporcionar muchas de las facilidades que ofrecen los sistemas multiusuario, en particular no tienen control de concurrencia y tienen precarios o inexistentes sistemas de recuperación.
Dado que las máquinas en las cuales se utilizan los sistemas monousuarios son comúnmente computadores de propósito general, la arquitectura de estas máquinas es siempre parecida (de 1 a 2 procesadores que comparten la memoria principal) por tanto los sistemas de base de datos que se ejecutan sobre estas maáquinas no intentan dividir una consulta simple entre los distintos procesadores, sino que ejecutan cada consulta en un único procesador posibilitando así la concurrencia de varias consultas. Este tipo de sistemas dan la sensación de una mayor productividad (puesto que pueden ejecutar un mayor número de transacciones por segundo) a pesar de que cada transacción individualmente no se ejecute más rápido. Por el contrario las máquinas paralelas tienen un gran número de procesadores y los sistemas de base de datos que ahí se ejecutan siempre tenderán a paralelizar las tareas simples (consultas) que

solicitan los usuarios.

1.14.2 Cliente / Servidor

Esta arquitectura consta de un cliente inteligente que puede solicitar servicios de un servidor en red. En el lado del cliente de esta arquitectura se tiene una aplicación frontal bastante sencilla ejecutándose en un ordenador personal. A una aplicación cliente / servidor se le puede pedir que realice validaciones o que muestre listas de opciones válidas, pero la mayor parte de las reglas de integridad de los datos y de negocio se imponen en la propia base de datos: relaciones, índices, valores predeterminados, rangos, disparadores, procedimientos almacenados, etc. En el lado del servidor se encuentra un motor de servidor de bases de datos inteligentes. El servidor está diseñado para aceptar consultas SQL desde la aplicación frontal, generalmente en forma de llamadas a procedimientos almacenados que devuelven conjunto de resultados claramente definidos y de ámbito limitado.
Generalmente, la aplicación cliente es responsable, al menos, de la administración de la conexión, la captura de los datos, la presentación de datos y la administración de los errores.
El servidor es el responsable de la administración inteligente de los recursos, la administración de la seguridad, la administración de los datos, de las consultas y sobre todo de la integridad de los datos.
Con el crecimiento de los computadores personales (PC) y de las redes de área local (LAN), se han ido desplazando hacia el lado del cliente la funcionalidad de la parte visible al usuario de la base de datos (interfaces de formularios, gestión de informes, etc.) de modo que los sistemas servidores provean la parte subyacente que tiene que ver con el acceso a las estructuras de datos, evaluación y procesamiento de consultas, control de concurrencia y recuperación. Los sistemas servidores pueden dividirse en 2 tipos: los servidores transaccionales (que sirven para agrupar la lógica del negocio en un servicio aparte, proveen una interfaz a través de la cual los clientes pueden enviar peticiones como lo son los ODBC) y los servidores de datos(los cuales envían datos a más bajo nivel y que descansan en la capacidad de procesamiento de datos de las maquinas clientes).
Existen 2 arquitecturas dominantes en la construcción de motores de base de datos cliente-servidor: los motores multiprocesos y los motores multihilos.

1.14.2.1 Motores de base de datos multiprocesos (Multi-process database engines).

Algunos motores de base de datos confían en múltiples aplicaciones para realizar su trabajo. En este tipo de arquitectura, cada vez que un usuario se conecta a la base de datos, ésta inicia una nueva instancia de la aplicación de base de datos. Con el fin de coordinar a muchos usuarios que accesan los mismos conjuntos de datos estos ejecutables trabajan con un coordinador global de tareas que planifica operaciones para todos los usuarios.

El ejemplo más popular de motores de base de datos multiprocesos es el Oracle Server (Oracle corporation) el cual carga 16 tipos de ejecutables distintos que realizan distintas tareas. El sistema ejecuta sus aplicaciones que sirven para

administrar el acceso de múltiples usuarios a las tablas, el registro y control de versiones de una transacción y otras características como la replicación de datos, transacciones distribuidas. Por otro lado, cuando una conexión a la base de datos se establece, el sistema carga los ejecutables relacionados a tareas de usuario.

Cada vez que un usuario se conecta a una base de datos Oracle, esta carga un ejecutable con una nueva instancia de la base de datos, las consultas de usuario son transmitidas a este ejecutable, el cual trabaja en conjunto con otros ejecutables en el servidor que retornan conjuntos de datos, manejan los bloqueos y ejecutan todas las funciones necesarias para el acceso de datos.

La mayoría de los motores de base de datos multiprocesos fueron desarrollados antes de que los sistemas operativos soportaran características tales como hilos o planificación de tareas (scheduling). Como resultado de esto, el hecho de descomponer una operación significaba escribir un ejecutable distinto para manejar esta operación. Esta característica proporciona el beneficio de la fácil escalabilidad a través de la adición de más CPUs.

En un ambiente de multitarea el sistema operativo divide el tiempo de procesamiento entre múltiples aplicaciones asignándoles una porción de tiempo de CPU ("slice")[15] a cada una. De esta manera siempre hay una sola tarea ejecutándose a la vez, sin embargo el resultado es que múltiples aplicaciones aparenten estar corriendo simultáneamente en una sola CPU. La ventaja real, sin embargo, viene cuando el sistema operativo cuenta con múltiples CPUs.

1.14.2.2 Motores de base de datos multihilos (Single-Process multi-threaded database engines)

Los motores de base de datos multihilos abordan el problema del acceso multiusuario de una manera distinta, pero con principios similares. En lugar de confiar en que el sistema operativo comparta los recursos de procesamiento, el motor toma la responsabilidad por sí mismo, lo que en la práctica se asocia a una mejor portabilidad del sistema. Motores de base de datos comerciales como Sybase Adaptive Server o Microsoft Sql Server son ejemplos de este enfoque.
Las ventajas de este tipo de motores radican en una mayor eficiencia en el uso de recursos para determinadas plataformas. Mientas un sistema multiprocesos consume entre 500 Kb y 1 Mb por conexión, un motor multihilos consume entre 50 y 100 Kb de RAM diferencia que puede ser utilizada en caché de datos y procedimientos. Otra

---

[15] Time Slice: Fracción de tiempo. Intervalo fijo de tiempo que se asigna a cada usuario o programa en un sistema multitarea o de tiempo compartido.

ventaja es que no hay necesidad de un mecanismo de comunicación de interprocesos. De esta manera, la base de datos utiliza un elemento finito de trabajo, (el hilo) para una variedad de operaciones (instrucciones de usuarios, bloqueos de datos, E/S de disco, administración del caché, etc.) en vez de utilizar aplicaciones especializadas para cada tarea.
Las desventajas más reconocidas son dos: **escalabilidad y portabilidad**. La escalabilidad se centra en la habilidad que tengan los distintos motores de base de datos multihilos de descomponer una operación de manera que múltiples tareas puedan ejecutar esta operación.

Los problemas de portabilidad guardan relación con el SMP (multiprocesamiento simétrico) y se originan en el hecho de que dado que diferentes fabricantes de hardware dan soporte a SMP de diferentes maneras, estos motores de base de datos han tenido que implementar técnicas neutras que buscan funcionar bien en cualquier implementación física, lo que conlleva una sobrecarga en el motor y una limitación en la habilidad de escalar a un gran número de procesadores.

1.14.3 SGBD Paralelos

Los sistemas paralelos de base de datos constan de varios procesadores y varios discos conectados a través de una red de interconexión de alta velocidad. Para medir el rendimiento de los sistemas de base de datos existen 2 medidas principales: la primera es la productividad (**throughput**) que se entiende como el número de tareas que pueden completarse en un intervalo de tiempo determinado. La segunda es el tiempo de respuesta (response time) que es la cantidad de tiempo que necesita para completar una única tarea a partir del momento en que se envíe. Un sistema que procese un gran número de pequeñas transacciones puede mejorar su productividad realizando muchas transacciones en paralelo. Un sistema que procese transacciones más largas puede mejorar tanto su productividad como sus tiempos de respuesta realizando en paralelo cada una de las subtareas de cada transacción.
Las ganancias en este tipo de SGBD se pueden dar en términos de **velocidad** (menor tiempo de ejecución para una tarea dada) y **ampliabilidad** (capacidad de procesar tareas más largas en el mismo tiempo).
Existen varios modelos de arquitecturas para máquinas paralelas, los más mencionados son:

- **Memoria Compartida** : Todos los procesadores comparten una memoria común.
- **Disco Compartido**: Todos los procesadores comparten una disposición de discos común.
- **Sin Compartimiento**: Los procesadores no comparten ni memoria ni disco.
- **Jerárquico**: Compartimiento tanto de memoria como de disco.

1.14.4 SGBD Distribuidos

En un SGBD distribuido, la base de datos se almacena en varios computadores que se pueden comunicar a su vez por distintos medios de comunicación (desde redes de alta velocidad a líneas telefónicas). No comparten memoria ni discos y sus tamaños

pueden variar tanto como sus funciones pudiendo abarcar desde PC hasta grandes sistemas. Se denomina con el término de emplazamientos o nodos a todos aquellos computadores que pertenecen a un sistema distribuido.
Las principales diferencias entre las bases de datos paralelas y las bases de datos distribuidas son las siguientes: las bases de datos distribuidas se encuentran normalmente en varios lugares geográficos distintos, se administran de forma separada y poseen una interconexión más lenta. Otra diferencia es que en un sistema distribuido se dan dos tipos de transacciones, las locales y las globales. Una transacción local es aquella que accede a los datos del único emplazamiento en el cual se inició la transacción. Por otra parte una transacción global es aquella que o bien accede a los datos situados en un emplazamiento diferente de aquel en el que se inició la transacción, o bien accede a datos de varios emplazamientos distintos.
Un sistema de base de datos distribuido se conoce por:

- Los distintos emplazamientos están informados de los demás. Aunque algunas relaciones pueden estar almacenadas sólo en algunos emplazamientos, éstos comparten un esquema global común.
- Cada emplazamiento proporciona un entorno para la ejecución de transacciones tanto locales como globales.

1.15 Qué es el código abierto?

Richard Mathew Stallman[16] fue el precursor del movimiento de software libre (FSF), a la edad de 18 años ingresó en el laboratorio de inteligencia artificial del MIT, el estaba acostumbrado a trabajar en un entorno de software libre, donde todos compartían todo. El momento en que su comunidad empezó a desaparecer ya que una compañía contrató a casi todos los hackers de laboratorio de IA, además el laboratorio en el que trabajaba adquirió un PDP-10, sus administradores decidieron utilizar el sistema no libre en lugar del ITS que había sido diseñado en el MIT y que era libre.
Al desaparecer su comunidad, Stallman se vio obligado a tomar una elección, unirse al mundo de software propietario, firmar los acuerdos de no revelar, y prometer que no iría en ayuda de sus amigo hacker. El podía haber hecho dinero de esa forma, pero sabía que al final de su carrera, al regresar a ver atrás, sentiría que utilizó su vida para empeorar al mundo. La otra opción era dejar el campo de la computación.
Lo primero que hizo fue un sistema operativo. Le pudo el nombre GNU, este nombre se eligió siguiendo una tradición hacker, como acrónimo recursivo para GNU´s not UNIX.
Desde ese momento Stallman no paró, realizo muchos cambios programando software libre para todos con la posibilidad de modificar sus fuentes a su

[16]Más información en http://es.wikipedia.org/wiki/Richard_Stallman ;

http://www.z-labs.com.ar/docs/tif/3-stallman.html

conveniencia y sin estar atados a una empresa que es dueña de los códigos fuente.
El Software de Open Source exige la distribución libre y gratuita acompañada del código fuente. Código abierto (open source en inglés) es el término por el que se le conoce a software distribuido y desarrollado en una determinada forma. Este término empezó a utilizarse en 1998 por usuarios de la comunidad del software libre, tratando de usarlo como reemplazo al ambiguo nombre original del software libre (free software).
En inglés, "free software" puede significar diferentes cosas. Por un lado, permite pensar en "software por el que no hay que pagar", y se adapta al término de forma igualmente válida que el significado que se pretende (software que posee ciertas libertades).
Lamentablemente, el término no resultó apropiado como reemplazo para el ya tradicional free software, y en la actualidad es utilizado para definir un movimiento nuevo de software, diferente al movimiento del software libre, aunque no completamente incompatible con este, de modo que es posible (como de hecho ocurre) que ambos movimientos trabajen juntos en el desarrollo práctico de proyectos.
El significado obvio del término "código abierto" es "se puede mirar el código fuente", lo cual es un criterio más débil y flexible que el del software libre; un programa de código abierto puede ser software libre, pero también puede serlo un programa semi-libre o incluso uno completamente propietario.
El software de código abierto (OSS por sus siglas en inglés) es software para el que su código fuente está disponible públicamente, aunque los términos de licenciamiento específicos varían respecto a lo que se puede hacer con ese código fuente.

1.15.1 Los Beneficios del Open Source

La obvia ventaja monetaria es que no existen **costos** de licencia para el producto en sí mismo. El mayor diferenciador de todos modos es el que el usuario puede, además obtener el código fuente. Esto le brinda **independencia** del proveedor ("contribuyente original" en el lenguaje de Código Abierto). De este modo el usuario no depende de su existencia y prioridades.
Toda la **información** (estado, errores (bugs), etc.) es abierta también, no existe política de ocultamiento corporativa ni censura. Si algo no funciona, no tendrá inconveniente en averiguarlo rápidamente. Como consecuencia, los proyectos de Código Abierto son muy **rápidos para reaccionar**, si surgen problemas. Un desafío de todos modos es el que Usted necesita decidir la importancia de los inconvenientes que pudiesen surgir y el impacto en su propio proyecto.
La **comunidad** de usuarios (y desarrolladores) hacen una notable diferencia. Debido a la diversidad de usuarios, los productos están usualmente muy bien **probados** y Usted puede obtener ayuda y consejo rápidamente.

1.15.2 Los Problemas del Open Source

Los proyectos de Código Abierto funcionan bien cuando el alcance es el de herramientas básicas y dónde los requerimientos están claramente definidos. Especialmente, los proyectos de aplicaciones de negocio tienden a no llegar a nada

debido a grandes "diseños por comité", discusiones y desacuerdos respecto de prioridades.
La prueba de funciones y rendimiento, requiere de un enfoque muy estructurado y recursos, usualmente limitados en los proyectos de Código Abierto. Lo mismo sucede con el empaquetado (packaging), actualizaciones y mejoras. Otro inconveniente puede darse por el requerimiento de licencias de terceros.

1.15.3 La Motivación del Open Source

En una compañía típica de software de aplicación, cerca del 50-70% del costo final total está relacionado con el costo de pre-venta: Representantes de Venta, gastos de Marketing e Ingenieros de Pre-venta realizando demos y presentaciones.
Todo apunta que las tecnologías Open Source cambiarán radicalmente la industria del software en 2 o 3 años. Las bases de datos son parte de esta transformación. Poco a poco, pero sin descanso, el software OpenSource está adquiriendo una robustez y una potencia suficiente como para plantar cara al software comercial. La liberalización de Interbase y Mysql es sólo el principio. ¿Quién usa bases de datos OpenSource? Pues desarrolladores de Web y software, y pequeñas y medianas empresas. La mayoría de usuarios también lo integran con otras aplicaciones OpenSource. Es decir, normalmente, se tiende a radicalizar: o todo OpenSource, o todo comercial. Aunque el panorama irá cambiando.

1.15.4 El código abierto en las empresas

En el Ecuador el tema es controversial unos prefieren lo corporativo por que es seguro, además existe alguien que puede responder si algo falla, un punto importante es que las empresas tienen miedo al cambio. En fin, es una decisión que tarde o temprano será evaluada con mayor seriedad.
Una percepción usual es "Si es libre (o gratis), probablemente no **valga gran cosa**". De hecho, existen diferentes motivaciones para renunciar al ingreso de las licencias. No lograr el éxito comercial puede ser una, obtener mayor soporte es otra.
El principal temor que los usuarios pueden tener es el tener que confiar en medios informales tales como grupos de usuarios para obtener ayuda y soporte. Otra preocupación puede ser lo imprevisible del desarrollo de funciones y características.
Se puede pagar por un sistema operativo o programa, o no. Hoy día en la mayoría de casos hay un sistema operativo o programa, cuando menos equivalente, en los dos sectores tanto en código abierto como en código cerrado. Por ejemplo, hoy en día ya hay un gran número de aplicaciones con licencia **GPL**[17] que corren sobre varios sistemas operativos, tanto propietarios como abiertos.
A nivel laboral, a fin de cuentas es lo mismo. Tanto si contratan un ingeniero en sistemas, o en un negocio propio, para un tipo de producto u otro no quiere decir que

---

[17]**La GNU General Public License** (Licencia Pública General) es una licencia creada por la Free Software Foundation y orientada principalmente a los términos de distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es software Libre. Más información en: http://www.es.gnu.org/licencias/gples.html

se utilizará el mismo producto toda la vida.
Un ingeniero de sistemas es un recurso humano que en particular debe adaptarse a su entorno de trabajo, no importa el sector, lo importante es mantenerse, puede ser linux, unix, windows, pero lo importante es que el ingeniero este capacitado.
En el caso de las empresas, es una apuesta de negocio. En esta vida, hay que adaptarse al entorno y necesidades del mercado, y en el caso de las empresas, a veces, se tienen que arriesgar.
Las distribuciones de linux fueron un riesgo, y alto, ya que el mercado de los sistemas operativos es muy duro. La inversión de las empresas no fue una inversión a corto plazo, estaban concientes que iban a necesitar algunos años. Todas las empresas empiezan de una manera muy dura, es la ley del mercado, ni más ni menos.
Es el mercado el que decide quién sigue y quién no. Pero lo importante es que muchos clientes mirarán la diferencia entre un precio y otro, se encantarán por la opción de código abierto?
Una vez más será el mercado quién decida. Si el código abierto es igual o más fiable que el cerrado y se sabe promocionar, por la ley de la oferta y la demanda, tiene las de ganar. Aunque hay muchas variables en el mercado real que hacen que no se tan trivial.
En definitiva, a un profesional de la informática le es igual trabajar en un sistema abierto o cerrado. A fin de cuentas no tiene importancia. Si una persona desea utilizar un sistema u otro en su casa, eso es elección de cada uno. A nivel de empresa, viene a ser igual, es inapropiado pensar que las grandes del código cerrado desaparecerán si se impone el código abierto. Se adaptaran, o desaparecerán, pero como las empresas las llevan humanos, y si algo tenemos es instinto de supervivencia, seguramente se adaptarán al nuevo modelo de mercado de nuevas tecnologías que implica el Open Source.

Capítulo 2.- PRINCIPALES CARATERISTICAS TECNICAS DE LOS MOTORES DE BASES DE DATOS.

2.1 Procesamiento de Transacciones

Varias de las operaciones que se realizan sobre la base de datos forman a menudo una única unidad lógica de trabajo. Un ejemplo de esto es la acreditación de cuentas en un banco, en la que el valor acreditado se resta en la cuenta (A) y se debe actualizar tanto el saldo contable como el disponible, a parte de este proceso se debe insertar un registro en la tabla de auditoria.

Es esencial que se ejecuten todos los procesos o ninguno. Este requisito de todo o nada se denomina **atomicidad**. Además es fundamental que el valor restado a los saldos sea preservado, este requisito se conoce como **consistencia**. Finalmente tras la ejecución correcta de la transacción, los nuevos valores de los saldos debe persistir, a pesar de la posible falla del sistema. Este requerimiento del sistema se denomina **durabilidad**. También es importante y necesario si ocurre algún error durante la

ejecución de cualquiera de los procesos que forman parte de la transacción se ejecute un roolback[18] de forma que deje los valores como estaban al principio, es decir, como estaban antes de la ejecución.

El componente encargado de lograr la atomicidad de la transacción se conoce como administrador de transacciones y las operaciones COMMIT (comprometer o confirmar) y ROOLBACK (retroceder) son la clave de su funcionamiento.

La operación **COMMIT** indica el término exitoso de una transacción: le dice al administrador de transacciones que se ha finalizado con éxito, que la base de datos está o debería estar de nuevo en un estado consistente y que se pude comprometer o hacer permanentes todas las modificaciones efectuadas.

La operación **ROLLBACK**, en cambio, nos indica el término no exitoso de una transacción: le dice al administrador de transacciones que algo salió mal, que la base de datos podría estar en un estado no consistente y que todas las modificaciones realizadas hasta el momento deben retroceder o anularse.

Una **transacción** es una colección de operaciones que se lleva a cabo como una única función lógica en una aplicación de base de datos. Cada transacción es una unidad de atomicidad y consistencia. Así, se requiere que las transacciones no violen ninguna restricción de consistencia de la base de datos, Es decir, si la base de datos era consistente cuando la transacción comenzó, la base de datos debe ser consistente cuando la transacción termine con éxito. Sin embargo, durante la ejecución de una transacción puede ser necesario permitir inconsistencias temporalmente, ya que una de las operaciones que forma parte de la transacción se debe realizar una antes de la otra. Esta inconsistencia temporal, aunque necesaria, puede conducir a dificultades si ocurre un fallo.

Es responsabilidad del programador definir adecuadamente las diferentes transacciones, de tal manera que cada una preserve la consistencia de la base de datos.

Asegurar las propiedades de atomicidad y durabilidad es responsabilidad del propio sistema de base de datos, específicamente del **componente de gestión de transacciones**. En ausencia de fallos, toda transacción completada con éxito y atómica se archiva fácilmente.
El sistema de base de datos debe realizar la **recuperación de fallos**, es decir, detectar los fallos del sistema y restaurar la base de datos al estado que existía antes de que ocurriera el fallo.

---

[18]Esto ocurre cuando hay algún tipo de fallo en una transacción utilizando Bases de Datos, todos los datos que no hayan sido guardados (dar commit), regresan a su estado original, evitando así la perdida de datos cuando un cliente falla.

Finalmente cuando varias transacciones actualizan la base de datos concurrentemente, la consistencia de los datos puede no ser preservada, incluso aunque cada transacción individualmente sea correcta. Es responsabilidad del **gestor de control de concurrencia** controlar la interacción entre las transacciones concurrentes para asegurar la consistencia de la base de datos.
En esencia, lo que se persigue con el procesamiento de transacciones es, por una parte, obtener una transparencia adecuada de las acciones concurrentes a una base de datos y por otra, manejar adecuadamente las fallas que se puedan presentar en una base de datos.
La mayoría de medianas y grandes compañías modernas utilizan el procesamiento de transacciones para sus sistemas de producción, y es tan inprescindible que las organizaciones no pueden funcionar en ausencia de él. El procesamiento de transacciones representa una enorme y significativa porción del mercado de los sistemas informáticos (más de cincuenta billones de dólares al año) y es, probablemente, la aplicación simple más amplia de las computadoras. Además, se ha convertido en el elemento que facilita el comercio electrónico.
Como puede percibirse, el procesamiento de transacciones es una de las tareas más importantes dentro de un sistema de base de datos, pero a la vez, es una de las más difíciles de manejar debido a diversos aspectos, tales como:

- **Confiabilidad**.- Puesto que los sistemas de base de datos en línea no pueden fallar.
- **Disponibilidad**.- Debido a que los sistemas de base de datos en línea deben estar actualizados correctamente todo el tiempo.
- **Tiempos de Respuesta**.- En sistemas de este tipo, el tiempo de respuesta de las transacciones no debe ser mayor a diez segundos.
- **Throughput**.- Los sistemas de base de datos en línea requieren procesar miles de transacciones por segundo.
- **Atomicidad**.- En el procesamiento de transacciones no se aceptan resultados parciales.
- **Permanencia**.- No se permite la eliminación en la base de datos de los efectos de una transacción que ha culminado con éxito.

Para conseguir anular y recuperar las transacciones, el método más usado consiste en utilizar un archivo de log en el que se va guardando toda la información necesaria para deshacer (en caso de fracasar) o rehacer (en caso de recuperar) las transacciones. Este log consta de los siguientes datos:

- Identificador de la transacción
- Hora de modificación
- Identificador del registro afectado
- Tipo de acción
- Valor anterior del registro
- Nuevo valor del registro
- Información adicional

Otra alternativa es manejar 2 archivos de log, uno con la imagen anterior a las modificaciones y otro con la imagen posterior a las modificaciones. El archivo log es usualmente una pila que una vez llena va eliminado registros según van entrando

nuevos.
Un concepto relacionado con los archivos de log es el **CHECKPOINT,** que permite manejar en forma eficiente el contenido de los archivos log, ya que permiten no tener que recorrer todo el archivo de log, ante fallas.
Los puntos marcados como checkpoint, permiten la recuperación de la base de datos en caliente, es decir, después de la caída del sistema se obtiene la dirección del registro de recuperación más reciente y se recorre el archivo de log desde el punto marcado como checkpoint.

2.2 Recuperación de la Información

El objetivo del concepto de recuperación es proteger la base de datos contra fallas lógicas o físicas que destruyan los datos en forma total o parcial. Estas fallas pueden afectar al correcto almacenamiento de los datos.
Para asegurar que la base de datos siempre esté en un estado consistente, cada base de datos tiene un proceso para obtener copias de seguridad, esto ayuda a mantener un registro confiable de los datos ante desastres o posibles fallas del sistema.

Por otro lado, las bases de datos crean unidades de ejecución llamadas transacciones, que pueden definirse como una secuencia de operaciones que se ejecutan en forma atómica, es decir, se realizan todas las operaciones que comprende la transacción o no se realiza ninguna.

La transacciones, o terminan el proceso con éxito y son completadas en la base de datos, o fracasan y deben ser restaurado el estado anterior de la base de datos.
La recuperación en frío, consiste en disponer de un backup o respaldo de la BD, que permitirá junto con los archivos de log, que se han ido produciendo desde el ultimo backup, reconstruir la BD, para dejarla consistente.
El error fatal, se produce cuando se pierde el archivo de log. En este caso resulta imposible recuperar la base. La solución pasa por disponer los archivos de log en respaldos.
El DBA debe definir responsabilidades, procedimientos, situaciones y plazos en los que se deben realizar las copias de seguridad y el respaldo del archivo de log, especificando a los operadores los procedimientos de recuperación ante caídas. El principio básico en el que se basa la recuperación es la redundancia.

2.3 Tipos de Datos

En las bases de datos existentes en la actualidad se sigue una norma SQL que soporta un conjunto de dominios predefinidos, que incluye los siguientes:

- **Char**. (n) es una cadena de caracteres de longitud fija, con una longitud n especificada por el usuario. También se puede utilizar la palabra completa character.
- **Varchar**. (n) es una cadena de caracteres de longitud variable, con una

longitud n especificada por el usuario. También se puede utilizar la forma completa character varying.

- **Int.** Es un entero (un subconjunto finito de los enteros, que es dependiente de la máquina). También se puede utilizar la palabra completa integer.
- **Smallint.** Es un entero pequeño (un subconjunto del dominio de los enteros, tambien dependiente de la maquina).
- **Numeric**. (p,d) es un numero en coma flotante, cuya precisión la especifica el usuario. El número está formado por p dígitos (más el signo), y de esos p dígitos, d pertenecen a la parte decimal. Así, numeric(3,1) permite que el número 56,5 se almacene exactamente, mientras que los números 444,5 y 0,32 no se pueden almacenar exactamente en un campo de este tipo.
- **Real, double precision**. Son respectivamente números en coma flotante y números en coma flotante de doble precisión , con precisión dependiente de la maquina.
- **Float**. (n) es un número en coma flotante , cuya precisión es de al menos n dígitos.
- **Date** es una fecha del calendario, que contiene un año (de cuatro dígitos), un mes y un día del mes. Ejm: "2004-02-01" La fecha se debe especificar en formato aaaa-mm-dd.
- **Time** es la hora del día, expresada en horas, minutos y segundos. Se puede usar una variante, time (p), para especificar el número de dígitos decimales para los segundos (el número predeterminado es cero). También es posible almacenar la información del uso horario junto al tiempo. Ejm: "09:30:00".
- **Timestamp** es una combinación de date y time. Se puede usar una variante, timestamp (p), para especificar el número de dígitos decimales para los segundos (el número predeterminado es 6). Ejm: "2004-02-01 09:30:01.45".

SQL permite incluir en la declaración de dominio de un atributo la especificación **not null** de este modo se prohíbe la inserción de un valor nulo para ese atributo.
SQL permite realizar operaciones de comparación sobre todos los dominios que se listan aquí, y permite realizar operaciones aritméticas y de comparación sobre los diferentes dominios numéricos. SQL también proporciona un tipo de datos llamado **interval** y permite realizar cálculos basados en fechas, horas e intervalos.

2.4 Integridad y Seguridad

Las restricciones de integridad proporcionan un medio de asegurar que las modificaciones hechas a la base da datos por los usuarios autorizados no provoquen la pérdida de la consistencia de los datos. Por tanto las restricciones de integridad protegen a las bases de datos de daños accidentales.
La integridad tiene como función proteger la BD contra operaciones que introduzcan inconsistencias en los datos. Se habla de integridad en el sentido de corrección, validez o precisión de los datos.
El subsistema de integridad de un SGBD debe por tanto detectar y corregir, en la medida de lo posible, las operaciones incorrectas. En la práctica es el punto débil de los SGBD comerciales, ya que casi toda la verificación de integridad se realiza

mediante código de procedimientos escritos por los usuarios.
Habrá operaciones cuya falta de corrección no sea detectable, por ejemplo, introducir un fecha de nacimiento 25/12/1945 cuando en realidad era 25/12/1954.
En lo que tiene que ver con la seguridad también se protege los datos frente al acceso de personas no autorizadas y destrucción o alteración malintencionada.

2.4.1 Operaciones semánticamente inconsistentes

Son las que transgreden las restricciones que ha definido el administrador al diseñar la base de datos, tales como:

- Restricciones sobre dominios por ejemplo, el dominio edad esté comprendido entre 18 y 65 años.
- Restricciones sobre los atributos, por ejemplo, la edad de los empleados ingenieros debe ser mayor de 21 años.

Estas restricciones pueden ser estáticas, como las anteriores o dinámicas por ejemplo, el sueldo de un empleado no puede disminuir.
Otra forma de clasificar las restricciones es en:

- **simples**: si se aplican a una ocurrencia de un atributo con independencia de los demás, por ejemplo, el sueldo de un empleado tiene que ser mayor que 60000.
- **compuestas**: si implican más de una ocurrencia, como es el caso de las restricciones de comparación, por ejemplo, el sueldo de un empleado debe ser menor que el de su jefe. O bien , las llamadas de globalidad, por ejemplo, el sueldo medio de los empleados de un determinado departamento debe ser menor de 250000.

Los SGBD tienen que ofrecer en su lenguaje de definición, facilidades que permitan describir las restricciones con una sintaxis adecuada.
Por ejemplo, CREATE DOMAIN, CREATE ASSERTION, CREATE INTEGRITY RULE
En general, una regla de integridad está compuesta por tres componentes:

- La restricción propiamente tal, que establece la condición que deben cumplir los datos.
- La respuesta a la trasgresión, que especifica las acciones a tomar, como rechazar las operaciones, informar al usuario, corregir el error con acciones complementarias, etc.
- Condición de disparo, que especifica cuándo debe desencadenarse la acción especificada en la restricción de integridad: antes, después o durante cierto evento.

Los **triggers**, son casos especiales de reglas de integridad. Un trigger es un procedimiento que se activa o dispara al ocurrir un evento, que tienen muchas utilidades.
Las reglas de integridad deben almacenarse en el diccionario de datos, como parte integrantes de los datos(control centralizado de la semántica), de modo que no han de incluirse en los programas. Esto trae algunas ventajas:

- Las reglas de integridad son más sencillas de entender y de cambiar, facilitando su mantenimiento.
- Se detectan mejor las inconsistencias.
- Se protege mejor la integridad, ya que ningún usuario podrá escribir un

programa que las transgreda, llevando a la BD a estados inconsistentes.

El subsistema de integridad del SGBD debe realizar las siguientes funciones:

- Comprobar la coherencia de las reglas que se definen.
- Controlar las distintas transacciones y detectar las transgresiones de integridad.
- Cuando se produce una transgresión, ejecutar las acciones pertinentes.

2.4.2 Integridad Referencial

En una base de datos es necesario asegurar que un valor x se encuentre en las tablas que determinan una relación, a esta condición se le denomina integridad referencial.

2.4.2.1 Restricciones de Integridad

En el mundo real existen ciertas restricciones que deben cumplir los elementos en él existentes; por ejemplo, una persona sólo puede tener un número de identificación y una única dirección oficial. Cuando se diseña una base de datos se debe reflejar fielmente el universo del discurso que estamos tratando, lo que es lo mismo, reflejar las restricciones existentes en el mundo real.

Los componentes de una restricción son los siguientes:

La operación de actualización (inserción, borrado o eliminación) cuya ejecución ha de dar lugar a la comprobación del cumplimiento de la restricción.
La condición que debe cumplirse, la cual es en general una proposición lógica, definida sobre uno o varios elementos del esquema, que puede tomar uno de los valores de verdad (cierto o falso).

La acción que debe llevarse a cabo dependiendo del resultado de la condición.

En general, se puede decir que existen tres tipos de integridad:

Integridad de dominio: restringimos los valores que puede tomar un atributo respecto a su dominio, por ejemplo EDAD >= 18 - 65.
Integridad de entidad: la clave primaria de una entidad no puede tener valores nulos y siempre deberá ser única, por ejemplo la cedula de identidad.

2.4.3 Disparadores

Un disparador es una orden que se ejecuta de manera automática como efecto secundario de la modificación de la base de datos. Para diseñar un mecanismo disparador se debe tomar en cuenta lo siguiente:
Especificar las condiciones en las que se va a ejecutar el disparador. Esto se

descompone en un evento que causa la comprobación del disparador y una condición que de debe cumplir para ejecutar el disparador.
Especificar las acciones que se va a realizar cuando se ejecute el disparador.
Este modelo de disparadores se denomina modelo **evento-condición-acción.**
Una vez que el disparador es almacenado en la base de datos, el sistema asume la responsabilidad de ejecutarlo cada vez que se cumpla el evento especificado y se satisfaga la condición correspondiente.

Los disparadores son mecanismos útiles para alertar a los usuarios o para realizar de manera automática ciertas tareas cuando de cumplen determinadas condiciones.
Por ejemplo, en el caso de un almacén que desee mantener un inventario mínimo por cada producto; cuando el nivel de inventario de un producto cae por debajo del mínimo, se debe solicitar un pedido automáticamente. Para la implementación de este disparador se debe tomar en cuenta cuando se modifique el nivel de inventario de un producto, el disparador debería comparar el nivel con el mínimo y si el nivel es inferior al mínimo se añadiría un nuevo pedido.

2.5 Seguridad y autorización

Los datos almacenados en la base de datos deben estar protegidos contra accesos no autorizados, de la destrucción o alteración malintencionada además de la introducción de inconsistencias que evitan las restricciones de integridad.

2.5.1 Violaciones de la Seguridad

En este grupo tenemos:

- Lectura no autorizada de los datos (robo de información).
- Modificación no autorizada de datos.
- Destrucción no autorizada de datos.

Un punto muy importante es lograr la seguridad de la base de datos, aunque no es posible la protección absoluta, pero se puede elevar lo suficiente para disuadir lo suficiente la mayor parte, si no la totalidad, de los intentos de tener acesso a la base de datos sin la debida autorización.

Para lograr tal nivel de seguridad hay que adoptar medidas en varios niveles:

- **Sistema de base de datos**. Dar acceso a los datos a usuarios de acuerdo al tipo de usuario, esto quiere decir, que se debe dar los permisos correspondientes a una parte limitada de la base. Por ejemplo, a ciertos usuarios de la base de datos se les puede dar permiso para consulta pero no se les permite la modificación. Es responsabilidad del sistema gestor de base de datos asegurarse de que no se violen estas restricciones de autorización.
- **Sistema Operativo.** La debilidad del sistema operativo puede servir como medio de acceso no autorizado a los datos. Lo importante aquí es que el sistema operativo debe ser seguro para minimizar la posibilidad de que se pueda ingresar a la base de datos.

- **Red.** Este es un punto muy importante porque hoy en día casi todos los sistemas de base de datos permiten el acceso remoto desde terminales, la seguridad a nivel de red juega un papel muy importante.
- **Físico.** Los sitios que contienen los sistemas informáticos como el lugar donde están los servidores por ejemplo, deben tener seguridades contra intrusos.
- **Humano.** Los usuarios administradores de la base de datos deben ser cuidadosamente elegidos para reducir la posibilidad de que alguno de ellos dé acceso a personas no autorizadas.

Hay que tomar en cuenta que la debilidad de los dos últimos puntos puede burlar las medidas de seguridad tomadas para los niveles superiores.

2.5.2 Autorizaciones

Los usuarios pueden tener varios tipos de autorización.

- **Autorización de lectura.-** permite la lectura de los datos, pero no su modificación.
- **Autorización de inserción.-** permite la inserción de nuevos datos, pero no la modificación de los existentes.
- **Autorización de actualización.-** permite la modificación de los datos, pero no su borrado.
- **Autorización de borrado**.- permite el borrado de los datos.

Los usuarios pueden recibir todos los tipos de autorización, ninguno de ellos o una combinación determinada de los mismos.
Además de estas autorizaciones, los usuarios pueden recibir autorización para modificar el esquema de la base de datos:

- **Autorización de índices.-** permite la creación y borrado de índices.
- **Autorización de recursos.-** permite la creación de relaciones nuevas.
- **Autorización de alteración.-** permite el añadido o borrado de atributos de las relaciones.
- **Autorización de eliminación**.- permite el borrado de las relaciones.

Las autorizaciones de eliminación y de borrado se diferencian en que la autorización de borrado sólo permite el borrado de registros. Si un usuario borra todos los registros de una relación, la relación sigue existiendo, pero está vacía. Si se elimina una relación deja de existir.

La capacidad de crear nuevas relaciones queda regulada mediante la autorización de recursos. El usuario con la autorización de recursos que crea una relación nueva recibe automáticamente todos los privilegios sobre la misma.
La autorización de índices puede parecer innecesaria, dado que la creación o borrado de un índice no afecta a los datos de las relaciones. Más bien, los índices son una estructura para la mejora del rendimiento. Sin embargo los índices también ocupan espacio y se exige que todas las modificaciones de las bases de datos actualicen los

índices. Si se concediera a todos los usuarios la autorización de índices, los que llevaran a cabo actualizaciones estarían tentados a borrar los índices, mientras que los que formulan consultas estarían tentados a crear numerosos índices. Para permitir al administrador de la base de datos que regule el uso de los recursos del sistema es necesario tratar la creación de índices como un privilegio.

2.6 Trazas de Auditoria

Muchas aplicaciones de bases de datos seguras requieren que se mantenga una traza de auditoria. Una traza de auditoria es un registro histórico de todos los cambios (inserciones, borrados o actualizaciones) de la base de datos, junto con información con el usuario que realizo el cambio y en que momento.

La traza de auditoria ayuda a la seguridad de formas diferentes. Por ejemplo, si el saldo de una cuenta es incorrecto, el banco desearía revisar todas las actualizaciones realizadas sobre la cuenta para encontrar las actualizaciones incorrectas (o fraudulentas), así como las personas que realizaron los cambios.

2.7 Indexación a Asociación

Un índice sirve para encontrar datos específicos en la base de datos se forma rápida y eficiente. Muchas consultas solamente hacen referencia a una pequeña porción de los registros de una tabla. Para reducir el gasto adicional en la búsqueda de estos registros se puede construir índices.

Tenemos dos tipos básicos de índices:

- **Indices ordenados.** Estos índices están basados en una disposición ordenada de los valores.
- **Indices asociados.** (hash índices). Estos índices están basados en una distribución uniforme de los valores a través de una serie de cajones (buckets). El valor asignado a cada cajón está determinado por una función, llamada función de asociación.

Se consideran varias técnicas de indexación y asociación. Ninguna es la mejor. Sin embargo, cada técnica es la más apropiada para una aplicación específica de bases de datos. Cada técnica debe ser valorada bajo los siguientes criterios:

- **Tipos de acceso.** Los tipos de acceso que se soportan eficazmente. Estos tipos podrían incluir la búsqueda de registros con un valor concreto en un atributo, o buscar los registros cuyos atributos contengan valores en un rango especificado.
- **Tiempo de acceso.** El tiempo que se tarda en buscar un determinado elemento de datos, o conjunto de elementos, usando la técnica en cuestión.
- **Tiempo de inserción.** El tiempo empleado en insertar un nuevo elemento de datos. Este valor incluye el tiempo utilizado en buscar el lugar apropiado donde insertar el nuevo elemento de datos, así como el tiempo empleado en actualizar la estructura del índice.
- **Tiempo de borrado.** El tiempo empleado en borrar un elemento de datos.

Este valor incluye el tiempo utilizado en buscar el elemento a borrar, así como el tiempo empleado en actualizar la estructura del índice.

2.8 Control de Concurrencia

El control de concurrencia en las bases de datos permite que la información se maneje en forma eficiente, permite además la ejecución de transacciones en paralelo, accesando a información compartida y, por lo tanto, interfiriendo potencialmente unas con otras. El hecho de reservar un pasaje aéreo por internet, cuando miles de personas pueden reservarlo también, nos da la idea de lo importante que es el manejo concurrente de la base de datos.
El procesamiento de transacciones en línea es utilizado por entidades como bancos, aerolíneas ya que la forma de negocio que este tipo de entidades tiene así lo requiere, para que todo funcione correctamente, es necesario, que las bases de datos estén actualizadas todo el tiempo.
Es una preocupación del programador que sus aplicaciones sean confiables y funcionen apropiadamente, pero a pesar de que estas aplicaciones son probadas antes de salir a producción, son vulnerables a ciertos errores que están fuera de su control. Estos errores potenciales surgen debido a dos factores: concurrencia y fallas.
Un algoritmo de control de concurrencia asegura que las transacciones se ejecuten atómicamente, controlando la intercalación de transacciones concurrentes, estas se ejecutan una después de otra.
El concepto principal es el de transacción. Informalmente, una transacción es la ejecución de ciertas instrucciones que accedan a una base de datos compartida. El objetivo del control de concurrencia y recuperación es asegurar que dichas transacciones se ejecuten atómicamente, es decir:
Cada transacción accede a información compartida sin interferir con otras transacciones, y si una transacción termina normalmente, todos sus efectos son permanentes, caso contrario no tiene efecto alguno.

2.8.1 El problema del control de concurrencia

En sistemas multiusuario, es necesario, un mecanismo para controlar la concurrencia, se pueden producir inconsistencias importantes derivadas del acceso concurrente, como por ejemplo, el problema de la operación perdida.
En un sistema de biblioteca por ejemplo, existe un campo que almacena el número de copias disponibles de cada ejemplar que esta disponible. Este campo debe incrementarse en uno cada vez que una persona devuelve un libro y disminuye uno cada vez que se realiza un préstamo.
El problema se da cuando existen varias bibliotecas, ubicadas en diferentes puntos alrededor de la ciudad, una de ellas inicia la transacción t1, leyendo la variable número de ejemplares (n) que se almacena en la variable n1. Tiempo después, otra biblioteca podría leer la misma variable incrementándola en una unidad, transacción t2. Después, la transacción t1 añade una unidad a esta variable y la actualiza, el resultado es erróneo, ya que la variable n debería haber aumentado en 2 unidades, y solo ha aumentado en una. La transacción t2 se ha perdido.

2.8.1.1 Técnicas de control de concurrencia

Existen varias técnicas:

- Pesimistas: bloqueo y marcas de tiempo
- Optimistas

2.8.1.1.1 Técnicas de bloqueo

Es una variable asociada a cada elemento de datos que describe el estado de dicho elemento respecto a las posibles operaciones (recuperación o actualización) que se pueden realizar sobre ellos en cada momento.
Las transacciones pueden llevar a cabo bloqueos, impidiendo a otros usuarios la recuperación o actualización de los elementos bloqueados, para evitar inconsistencias en el acceso concurrente.
Los SGBD tienen bloqueos (por registro, por tabla) para asegurar la consistencia. Los usuarios también pueden bloquear explícitamente los objetos, impidiendo el acceso por parte de otros usuarios.

2.8.1.1.1.1 Tipos de Bloqueo

- **Exclusivos:** cuando una transacción mantiene un bloqueo de este tipo, ninguna otra transacción puede acceder al objeto bloqueado, ni bloquearlo, hasta que sea liberado por la transacción que lo había retenido. Se utiliza cuando se quiere actualizar datos.
- **Bloqueo compartido:** cuando una transacción bloquea en este modo, permite que otras transacciones retengan también el objeto en bloque compartido, pero no exclusivo. Este tipo se utiliza cuando no se requiere actualizar datos, pero se desea impedir cualquier modificación mientras los datos son consultados.

El algoritmo que se utiliza se llama bloqueo de dos fases (two phase locking).
El problema de las técnicas de bloqueo es que puede producirse un interbloqueo (deadlock), dos o mas transacciones están esperando cada una de ellas que la otra libere algún objeto antes de seguir
Se puede solucionar:

- **Prevenir el deadlock:** obliga a que las transacciones bloqueen todos los elementos que necesitan por adelantado. En caso de no poder conseguir todos esos elementos no bloquea ninguno y se queda en espera hasta volver a intentarlo.
- **Detectar el deadlock:** Se controla de forma periódica si se ha producido un deadlock. Se construye un grafo en espera, cada nodo es una transacción en ejecución y un arco de una transacción Ti a Tj, en caso que Ti esté esperando un elemento que ocupa Tj. Si existe un ciclo en el grafo tenemos un deadlock. La solución es escoger transacciones víctimas y deshacerlas, hasta que desaparezca el deadlock. Cada SGBD tiene políticas diferentes para escoger víctimas.

Este tema influye notoriamente en el rendimiento de los sistemas. Los SGBD pueden bloquear:

- un campo de un registro (un atributo de una tabla)
- un registro (una tupla)

- un archivo (una tabla)
- la BD total

Esto se llama granularidad del bloqueo.

Granularidad.- muy gruesa implica gestionar menor número de bloqueos, pero retrasa la ejecución de muchas transacciones (los objetos no se van liberando). Una granularidad muy fina, permite mayor concurrencia, pero aparecen más situaciones de deadlock que han de ser resueltas.

2.8.1.1.1.2 Protocolo de bloqueo de dos fases

Indica el momento en que una transacción puede bloquear y desbloquear cada uno de los elementos de datos.

Un protocolo que asegura la secuencialidad en cuanto a conflictos es el protocolo de bloqueo en dos fases. Este protocolo exige que cada transacción realice las peticiones de bloqueo y desbloqueo en dos fases:

1. **Fase de crecimiento.-** Una transacción puede obtener bloqueos pero no puede liberarlos.
2. **Fase de decrecimiento.-** Una transacción puede liberar bloqueos pero no puede obtener ninguno nuevo.

2.8.1.1.1.3 Técnicas de marca de tiempo (timestamping)

Las marcas de tiempo son identificadores únicos que se asignan a las transacciones, que se consideran como el tiempo de inicio de una transacción. Con esta técnica no existen bloqueos. Ordena las transacciones. Se retrasan.

2.8.1.1.2 Técnicas optimistas

Las transacciones acceden libremente a los elementos, y antes de finalizar se determina si ha habido interferencias.

Este tipo de técnicas considera que las transacciones tienen 3 fases:

- **Lectura:** las transacciones realizan operaciones sobre copias privadas de los objetos (accesibles solo por la transacción)
- **Validación :** en la que se comprueba si el conjunto de objetos modificados por una transacción se solapa con el conjunto de objetos modificados por alguna otra que haya hecho la validación durante la fase de lectura de dicha transacción
- **Grabación:** en el caso de no detectar interferencias se graban las modificaciones, convirtiendo las versiones privadas de los objetos en versiones actuales.

2.9 Procesamiento de consultas

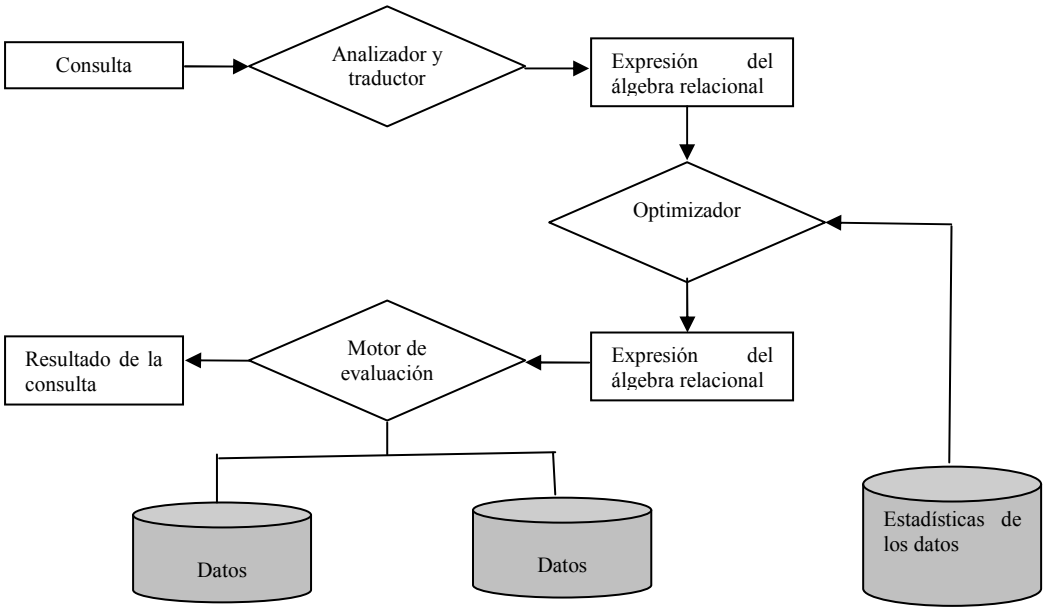
El procesamiento de consultas hace referencia a la serie de actividades implicadas en la extracción de la información de la base de datos. Estas actividades incluyen la traducción de consultas expresadas en lenguajes de bases de datos de alto nivel en

expresiones implementadas en el nivel físico del sistema, así como transformaciones de optimización de consultas y la evaluación real de las mismas.
Los pasos básicos para el procesamiento de una consulta son:

- Análisis y traducción
- Optimización
- Evaluación

Antes de realizar el procesamiento de la consulta, el sistema debe traducir la consulta en una utilizable. Un lenguaje como SQL es adecuado para el uso humano, pero es poco apropiado para una representación interna en el sistema de la consulta. Así, una representación más útil es la basada en el álgebra relacional extendida.
Entonces, la primera acción que el sistema realiza es esta traducción de la consulta al formato interno. Este proceso es similar al que realiza el analizador de un compilador. Durante este proceso, el analizador comprueba la sintaxis de la consulta del usuario, verifica que sean nombres de las relaciones en la base de datos, etc.
Para especificar completamente cómo evaluar una consulta, no basta con proporcionar la expresión del álgebra relacional, además hay que anotar en ellas las instrucciones que especifiquen cómo evaluar cada operación. Estas anotaciones podrían ser el algoritmo a usar para una operación específica o el índice o índices concretos a utilizar. Las operaciones de álgebra relacional anotadas con instrucciones sobre la evaluación recibe el nombre de primitivas de evaluación. Una secuencia de operaciones primitivas que se pueden utilizar para evaluar una consulta establecen un plan de ejecución de la consulta o plan de evaluación de la consulta.

Gráfico # 4: Pasos en el procesamiento de una consulta



Fuente: Fundamentos de bases de datos
Autor: Diego Burbano

El motor de ejecución de consultas toma un plan de evaluación, lo ejecuta y devuelve su respuesta a la consulta.

Los diferentes planes de evaluación para una consulta dada pueden tener costes distintos. No se puede esperar que los usuarios escriban las consultas de manera que sugieran el plan de evaluación más eficiente. En su lugar, es responsabilidad del sistema construir un plan de evaluación de la consulta que minimice el costo de la evaluación de la consulta.
La secuencia de pasos descrita para procesar una consulta es representativo, no todas las bases de datos lo siguen exactamente. Por ejemplo, en lugar de utilizar la representación del álgebra relacional, varias bases de datos usan una representación anotada del árbol de análisis basada en la estructura de la consulta SQL. Sin embargo, los conceptos que se describen, forman la base del procesamiento de consultas en las bases de datos.

2.10 Protección de los datos

La protección de los datos es un aspecto muy relacionado con el propósito mismo para el que fue creada una base de datos. La protección de datos debe realizarse contra todo tipo de fallos, físicos, lógicos y humanos.
En este contexto aparecen los temas de: recuperación, concurrencia, seguridad e integridad de la base de datos.
Los problemas de recuperación y concurrencia están muy relacionados con lo que se conoce como procesamiento de transacciones que ya lo habíamos revisado anteriormente.

Capítulo 3.- PRINCIPALES CARATERISTICAS DE LOS GESTORES DE BASES DE DATOS.

Existen otras bases de datos relacionales, tanto bases de datos comerciales y de código abierto están disponibles, para este estudio se va a analizar Mysql como base de datos de código abierto y Oracle como base de datos comercial.

3.1 Mysql

3.1.1 Introducción

Mysql en los últimos años ha tenido un crecimiento vertiginoso. Es la base de datos de código abierto más popular del mundo. Código abierto significa que todo el mundo puede acceder al código fuente, es decir, al código de programación de Mysql, esto significa que también todos pueden contribuir con ideas, elementos, mejoras o sugerir optimizaciones. Y así es que Mysql ha pasado de ser una pequeña base de datos a una completa herramienta. Su rápido desarrollo se debe en gran medida a la contribución de mucha gente al proyecto, así como la dedicación del equipo de Mysql.

A diferencia de los proyectos propietarios, en los que el código fuente es desarrollado por un número reducido de personas y se protege atentamente, los proyectos de código abierto no excluyen a nadie interesado en aportar ideas, si disponen de los conocimientos necesarios.

Lo que en un tiempo se consideró como un sencillo juguete para uso en sitios Web, se ha convertido en la actualidad en una solución viable y de misión crítica para la administración de datos.

Mysql es un sistema de administración de bases de datos relacional (RDBMS). Se trata de un programa capaz de almacenar una enorme cantidad de datos de gran variedad y de distribuirlos para cubrir las necesidades de cualquier tipo de organización, desde pequeños establecimientos comerciales a grandes empresas y organismos administrativos. Mysql compite con sistemas RDBMS propietarios como Oracle, Sql Server y Db2.

Mysql incluye todos los elementos necesarios para instalar el programa, preparar diferentes niveles de acceso de usuario, administrar el sistema y proteger los datos. Puede desarrollar sus propias aplicaciones de bases de datos en la mayor parte de lenguajes de programación utilizados en la actualidad y ejecutarlos en casi todos los sistemas operativos, incluyendo algunos de los que probablemente no ha oído hablar nunca. Mysql utiliza el lenguaje de consulta estructurado (SQL).

Antes Mysql se consideraba como la opción ideal de sitios web; sin embargo, ahora incorpora muchas de las funciones necesarias para otros entornos y conserva su gran velocidad. Mysql es una base de datos robusta que se la puede comparar con una base de datos comercial, es incluso más veloz en el procesamiento de las transacciones y dispone de un sistema de permisos elegante y potente, y ahora, además, incluye un motor de almacenamiento InnoDb[19][17] compatible con ACID[20][18], además dispone de store procedures, triggers, vistas.

Mysql es rápido, y una solución accesible para administrar correctamente los datos de una empresa. MysqlAB es la compañía responsable del desarrollo de Mysql, dispone de un sistema de asistencia eficiente y a un precio razonable, y, como ocurre con la mayor parte de las comunidades de código abierto, se puede encontrar una gran cantidad de ayuda en la Web.
Son muchas las razones para escoger a Mysql como una solución de misión crítica para la administración de datos:

- **Costo:** Mysql es gratuito para la mayor parte de los usos y su servicio de asistencia resulta económico.

---

[19]**INNODB** es un tipo de tabla de Mysql que permite trabajar con transacciones, y definir reglas de integridad referencial.

[20]**ACID** son las propiedades que una base de datos debe cumplir para que el Sistema administrador de base de datos (DBMS) maneje correctamente la transaccionalidad, el término ACID viene de Atomicidad, Consistencia, Aislamiento, Durabilidad.

- **Asistencia:** MysqlAB ofrece contratos de asistencia a precios razonables y existe una nutrida y activa comunidad Mysql.
- **Velocidad:** Mysql es mucho más rápido que la mayoría de sus rivales.
- **Funcionalidad:** Mysql dispone de muchas de las funciones que exigen los desarrolladores profesionales, como compatibilidad completa con ACID, compatibilidad para la mayor parte de SQL ANSI[21][19], volcados online, duplicación, funciones SSL e integración con la mayor parte de los entornos de programación.
- **Portabilidad:** Mysql se ejecuta en la inmensa mayoría de sistemas operativos y, la mayor parte de los casos, los datos se pueden transferir de un sistema a otro sin dificultad.
- **Facilidad de uso:** Mysql resulta fácil de utilizar y de administrar. Las herramientas de Mysql son potentes y flexibles, sin sacrificar su capacidad de uso.

3.1.2 Tipos de Datos y tipos de tabla

Mysql utiliza varios tipos de tablas. El tipo de tabla predeterminado es MyISAM que está optimizado para la velocidad del comando SELECT.
La mayor parte de los sitios Web utilizan esta tabla, ya que estos sitios suelen utilizar la instrucción SELECT mucho más que las instrucciones INSERT o UPDATE.

3.1.3 Análisis de los distintos tipos de columnas

Existen tres tipos fundamentales de columnas: numéricas, de cadena y de fecha.
Por regla general se debe seleccionar el tipo de columna de menor tamaño, ya que de esta forma se ahorra espacio y se logra una mayor velocidad de acceso y actualización. Sin embargo, si se selecciona un tipo de columna demasiado pequeño, puede dar como resultado la pérdida de datos o que se recorten al introducirlos.

3.1.3.1 Tipos de columna numéricos

Las columnas numéricas están diseñadas para almacenar todo tipo de datos numéricos, como precios, edades y cantidades. Hay dos tipos principales de tipos numéricos: tipos enteros y de punto flotante.

Tabla # 3: Tipos de datos numéricos Mysql

Tipo
TINYINT
BIT
BOOL
SMALLINT
MEDIUMINT
INT
INTEGER
BIGINT

[21]**SQL ANSI** es un estándar para el manejo del lenguaje estructurado de consultas.

FLOAT
DOUBLE
DEC
NUMERIC

Fuente: Mysql Avanzado
Autor: Diego Burbano

3.1.3.2 Tipos de columna de cadena

Los tipos de columna de cadena se utiliza para almacenar todo tipo de datos compuestos de caracteres como nombres, direcciones.

Tabla #4: Tipo de datos Cadena Mysql

Tipo
CHAR
VARCHAR
TINYBLOB
TINYTEXT
BLOB
TEXT
MEDIUMBLOB
MEDIUMTEXT
LONGBLOB
LONGTEXT
ENUM
SET

Fuente: Mysql Avanzado
Autor: Diego Burbano

3.1.3.3 Tipos de cadena de fecha y hora

Los tipos de columna de fecha y hora están diseñados para trabajar con las necesidades especiales que exigen los datos de tipo temporal y se puede utilizar para almacenar datos tales como la hora del día o fechas de nacimiento.

Tabla #5: Tipos de datos fecha y hora Mysql

Tipo
DATETIME
DATE
TIMESTAMP
TIME
YEAR

Fuente: Mysql Avanzado
Autor: Diego Burbano

3.1.4 Análisis de los distintos tipos de tablas

Existen dos tipos de tablas de transacción segura (Innodb y BDB). El resto (ISAM; MyISAM, MERGE y HEAP) no son de transacción segura. La elección del tipo de tabla adecuado puede afectar enormemente al rendimiento.

3.1.4.1 Tablas ISAM

Las tablas de tipo de Método de acceso secuencial indexado (ISAM) era el estándar antiguo de Mysql. Estas fueron sustituidas por las tablas MyiSAM en la versión 3.23. Por lo tanto, es probable que solo se tope con este tipo de tablas si está trabando con bases de datos antiguas. La principal diferencia entre las dos, es el índice de las tablas MyISAM es mucho más pequeño que el de las tablas ISAM, de manera que un SELECT con un índice sobre una tabla MyISAM utilizará mucho menos recursos del sistema.

3.1.4.2 Tablas MyISAM

3.1.4.2.1 Tablas estáticas

Las tablas estáticas tienen longitud fija. Cada registro tiene asignado exactamente 10 Bytes.
Este tipo de tablas se caracterizan por:

- Ser muy rápidas (ya que Mysql sabe que el segundo nombre comienza siempre en el carácter número once)
- Resultan sencillas de almacenar en caché.
- Resultan sencillas para reconstruir tras un fallo.
- Requieren más espacio de disco

3.1.4.2.2 Tablas Dinámicas

Las columnas de las tablas dinámicas tienen diferentes tamaños. Auque este tipo de dato ahorra espacio, resulta sin embargo más complejo.
Las tablas de tipo dinámico presentan las siguientes características:

- Todas las columnas de cadena son dinámicas.
- Por regla general, ocupan mucho menos espacio de disco que las tablas fijas.
- Las tablas requieren un mantenimiento regular para evitar su fragmentación.
- No resulta tan sencillo de reconstruir tras un fallo, especialmente si las tablas están muy fragmentadas.

3.1.4.2.3 Tablas comprimidas

Las tablas comprimidas son tablas de solo lectura que utilizan mucho menos espacio en disco.
Son ideales para su uso con datos comprimidos que no cambien (que solo se pueden leer y no escribir) y donde no exista mucho espacio disponible.
Las tablas comprimidas presentan las siguientes características:

- Las tablas son mucho más pequeñas.
- Como cada registro se comprime de forma separada, la carga de acceso es reducida.
- Cada columna se podría comprimir de forma diferente, utilizando distintos algoritmos de compresión.

- Se puede comprimir formatos de tabla fija y dinámica.

3.1.4.3 Tablas Merge

Las tablas Merge son la fusión de las tablas MyISAM iguales.
Por lo general se usa cuando las tablas MyISAM comienzan a resultar demasiado grandes.
Entre las ventajas de estas tablas se pueden mencionar las siguientes:

- Resultan más rápidas en determinadas situaciones.
- El tamaño de la tabla es más pequeño

Desventajas de la tabla Merge:

- Resultan mucho más lentas en búsquedas.
- El comando REPLACE no funciona sobre ellas.

3.1.4.4 Tablas Heap

Las tablas Heap son el tipo de tabla más rápido porque se almacena en memoria y utilizan un índice asignado. La contrapartida es que como se almacenan en memoria, en el caso de una falla del sistema, los datos se pierden.

3.1.4.5 Tablas Innodb

Las tablas Innodb son tablas de transacción segura (lo que significa que dispone de las funciones COMMIT y ROLLBACK). En una tabla MyISAM, la tabla entera se bloquea al realizar funciones de inserción. Durante esa fracción de segundo, no se puede ejecutar ninguna otra instrucción sobre la tabla. Innodb utiliza funciones de bloqueo a nivel de fila de manera que solo se bloquee dicha fila y no toda la tabla, y se puedan seguir aplicando instrucciones sobre otras filas.

3.1.5 Transacciones y bloqueos

Las consultas sobre la base de datos se ejecutan una después de otra. En el caso de un sitio Web que sirva páginas, da lo mismo el orden en que la base de datos realice las consultas, siempre y cuando lo haga rápidamente. Sin embargo, ciertos tipos de consultas necesitan realizarse en un orden dado, como las que dependen de los resultados de una consulta anterior, o grupos de actualizaciones que necesitan realizarse en conjunto. Todos los tipos de tabla pueden utilizar la función de bloqueo, pero sólo los tipos innodb y BDB disponen de funciones transaccionales integradas.

3.1.6 Las transacciones en las tablas Inodb

La potencia de las tablas innodb procede del uso de transacciones o instrucciones SQL agrupadas en una. Un ejemplo típico son las transacciones bancarias.
Por ejemplo, si se transfiere una cantidad de dinero desde la cuenta de una persona a otra, se realizan al menos dos consultas:

```
UPDATE Persona1 SET Valor = ValorDisponible – ValorADebitar;
UPDATE Persona2 SET Valor = ValorDisponible + ValorADebitar;
```

El proceso parece claro, pero que ocurriría si algo sale mal durante el proceso y el sistema falla entre las dos consultas sin que llegue a completarse la segunda. Se habrá retirado los fondos de la cuenta de la primera persona, se creerá que el pago se ha realizado. Sin embargo, la segunda persona no estará muy contenta porque el pago no se ha realizado. En este tipo de transacciones, resulta fundamental asegurarse de que las dos consultas se levan a cabo o que no se hace ninguna de las dos. Para ello, se empaquetan en los que se conoce como una transacción, con una instrucción BEGIN para indicar el inicio de la transacción y una instrucción COMMIT para indicar el final. Solo tras procesar la instrucción COMMIT, las consultas se habrán convertido en permanentes. Si algo sale mal a media ejecución, podemos utilizar el comando ROLLBACK para invertir la parte incompleta de la transacción.

3.1.7 Lecturas coherentes

De manera predeterminada, las tablas innodb realizan una lectura coherente. Esto significa que al realizar una consulta de selección, Mysql devuelve los valores presentes de la base de datos hasta la última transacción completada. Si en el momento de realizar la consulta existe alguna transacción en progreso, los resultados de las instrucciones UPDATE o INSERT no se reflejarán, con una excepción: la transacción abierta puede modificarse (puede que haya observado que al realizar la consulta BEGIN-INSERT-SELECT, se visualizó el resultado insertado). Para poder verlo, necesita tener dos ventanas abiertas y estar conectado a la base de datos.

3.1.8 Lectura de Bloqueos para actualizaciones

Las lecturas coherentes no siempre resultan adecuadas. Por ejemplo, que ocurriría si varios usuarios están intentando agregar un nuevo registro en una tabla innotest? Cada nuevo registro inserta un número ascendente exclusivo, este campo no es clave principal o un campo de incremento automático, por lo tanto nada impide que cree el registro duplicado. Sin embargo, no queremos que eso ocurra. Lo que desearíamos es leer el valor actual e insertar un nuevo valor, incrementando en una unidad. Pero esta acción no garantiza un valor único.
La forma de evitar resultados erróneos es realizando un bloqueo de actualización sobre la operación de selección. Si indicamos a Mysql, que estamos realizando una lectura de actualización, no permitirá que nadie más lea el valor hasta que nuestra transacción se haya completado.

3.1.9 Bloqueos de lectura en modo compartido

Existe otro tipo de bloqueo de lectura que no devuelve un valor si el valor que está leyendo ha sido modificado por una transacción incompleta. Devuelve el último valor, pero no forma parte de una transacción cuya intención es modificar el valor.

3.1.10 Transacciones en tablas BDB

Las tablas BDB procesan las transacciones de forma ligeramente diferente a las tablas Innodb. Si una persona está realizando una transacción sobre una tabla x, si esta transacción no está completa, ninguna persona podrá consultar los datos de esta

tabla mientras la transacción no finalice.
El periodo de tiempo que puede significar al llevar a cabo esta consulta es demasiado. El hecho de que no se trate de una consulta de selección "rápida" en las tablas BDB significa que todas las transacciones que se pospongan pueden dar lugar a graves problemas de rendimiento.
Como en el caso de las tablas innodb, el modo predeterminado de AUTOCOMMIT = 1. Esto significa que a menos que coloque sus cambios dentro de una transacción (comenzando con BEGIN), se completarán inmediatamente.

3.1.11 Otros comportamientos transaccionales

Existe una serie de comandos adicionales que finalizan automáticamente una transacción (en otras palabras, que se comportan como si hubiéramos realizado una operación de confirmación):

- BEGIN
- ALTER TABLE
- CREATE INDEX
- RENAME TABLE
- TRUNCATE
- DROP TABLE
- DROP DATABASE
- LOCK TABLES

Hay que tener mucho cuidado cuando se realizan transacciones ya que al ejecutar cualquiera de estos comandos cuando las transacción este a medias, automáticamente se realizará un COMMIT.

3.1.12 Bloqueo de tablas

Mysql maneja bloqueo a nivel de fila como lo revisamos anteriormente en el análisis de las tablas innodb y BDB. Los bloqueos a nivel de fila son mucho más eficaces cuando se necesita realizar una gran cantidad de inserciones o actualizaciones en la tabla. El bloqueo a nivel de fila, sin embargo, solo está disponible para los tipos de tabla de transacción segura (BDB e Innodb). Mysql incorpora la función de bloqueo a nivel de tablas, que está disponible para todos los tipos de tabla.
Existen dos tipos de bloqueos de tabla: los bloqueos de lectura y los bloqueos de escritura. Los bloqueos de lectura solo permiten realizar lecturas sobre la tabla, quedando bloqueadas las operaciones de escritura. Los bloqueos de escritura impiden la realización de operaciones de lectura o escritura sobre la tabla durante el bloqueo.

3.1.13 Cómo evitar los bloqueos de tabla

Se debe evitar los bloqueos sobre tablas que necesiten realizar un gran volumen de actualizaciones, ya que, en el caso de los bloqueos de escritura, no se puede leer o escribir ningún registro de la tabla durante el bloqueo.
Además, como los bloqueos de escritura tienen prioridad sobre los de lectura de

manera predeterminada, no se puede leer ningún registro hasta que todas las operaciones de actualización e inserción se completen, lo que puede provocar que Mysql se atasque de forma terrible. Existen varias formas de evitar los bloqueos de tabla. Una de ellas consiste en realizar la operación de lectura y actualización dentro de la misma instrucción (es lo que se conoce como actualización incremental).

3.1.14 Niveles de transacción

Se puede modificar el comportamiento predeterminado al trabajar con transacciones mediante el establecimiento del nivel de transacción. Existen varios niveles de transacción en Mysql. En concreto admite los siguientes niveles de aislamiento de transacción.

3.1.14.1 READ UNCOMMITTED

Este nivel permite transacciones para leer datos sin confirmar desde otras transacciones (es lo que se conoce como lectura sucia).

3.1.14.2 REPEATABLE READ

Este nivel no permite lecturas no susceptibles de repetición (que son las que se dan cuando otra transacción ha modificado los datos, incluso si se han confirmado).

3.1.14.3 SERIALIZABLE

Este nivel no permite lecturas fantasma, que tienen lugar cuando otra transacción ha confirmado una nueva fila que coincide con los resultados de nuestra consulta. Los datos serán los mismos en cada ocasión.

3.1.15 Indices y optimización de consultas

Es posible agilitar la velocidad en las consultas mediante el uso de métodos básicos.
El uso inteligente de índices ayudará a que la consulta, la actualización sea más rápida.
El ajuste correcto del servidor también contribuye a lograr mejoras notables.

3.1.16 Compresión de los índices

Al realizar una consulta sobre una tabla X de la base de datos, los registros son recuperados mediante un barrido completo de la tabla si la misma no utiliza índices, el problema es notorio cuando queremos buscar información con cierto criterio de búsqueda, haciendo que la consulta se demore demasiado.
La operación de recorrer la tabla de esta forma (de principio a fin, examinando todos los registros) se conoce como examen completo de la tabla. Cuando las tablas son de gran tamaño, esta operación resulta poco eficiente ya que la labor de examinar tablas compuestas de varios cientos de miles de registros puede resultar muy lenta. Para evitar este problema, se debe ordenar los registros. Sin embargo, puede ocurrir que deseemos buscar registros de la tabla utilizando otro criterio de búsqueda. La

solución es crear listas separadas para cada campo que se desee ordenar, a este ordenamiento se le conoce como índice.
En Mysql existen cuatro tipos de índice: clave primaria, un índice exclusivo, un índice de texto completo, y un índice ordinario.

3.1.17 Tipos de Indice

3.1.17.1 Clave primaria

Una clave primaria es un índice establecido sobre un campo en el que cada valor es exclusivo y ninguno de los valores puede ser nulo.
Para establecer una clave primaria al crear la tabla, se debe utilizar la instrucción PRIMARY KEY al final de las definiciones del campo, es importante la palabra clave NOT NULL, es además obligatoria al crear un campo primario, esto indica a la base de datos que no se admiten valores nulos ni valores duplicados ya que la clave primaria es única.

3.1.17.2 Indice exclusivo

Los índices que no son primarios, permiten valores duplicados (a menos que los campos se especifiquen como únicos). Un índice exclusivo permite realizar búsquedas por un solo registro de la tabla que no necesariamente es único pero que la gran cantidad de registros almacenados amerita la creación de este índice.

3.1.17.3 Indice de texto completo

Se puede crear este tipo de índice sobre cualquier campo Char, Varchar o Text. Los índices de texto completo están diseñados para facilitar la búsqueda sobre palabras clave en capos de texto de tablas grandes.
Para devolver los resultados de una búsqueda de texto completo, se utiliza la función MATCH(), y se busca la correspondencia de un campo con un valor.

3.1.17.4 Indice ordinario

Un índice ordinario es aquel que por necesidad del programador o de la consulta es necesario la creación del índice, este puede ser de uno o varios registros. Hay que tener cuidado en la creación de este tipo de índices, ya que pueden degradar el correcto funcionamiento de la base de datos, no se puede crear porque sí, es necesario examinar detenidamente si es óptimo o no.

3.1.18 Tipos de Tabla e índices

Cada tipo de tabla tiene su propio comportamiento en materia de índices y cada una de ellas lo procesa de manera diferente. No todos los tipos de índices están disponibles para los distintos tipos de tabla. Es importante tener claro como se va a utilizar una tabla y los índices que se van a necesitar ante de seleccionar el tipo de

tabla. En ocasiones, lo que parece ser el tipo de tabla perfecta se convierte en la peor elección porque o se puede utilizar un determinado tipo de índice en ella.

A continuación se enlista las funciones y las diferencias de índices para cada tipo de tabla. Las tablas MyISAM presentan las siguientes características:

- Los índices se almacenan con la extensión .MYI
- Los índices de número se almacenan con el byte alto primero par permitir una mejor compresión del índice.
- Se puede utilizar índices BLOB y TEXT.
- Se permiten valores nulos en los índices.
- Los datos y el índice se pueden incluir en directorios diferentes (lo que permite una mayor velocidad).

Las tablas MERGE presentan las siguientes características:

- Las tablas MERGE no contienen índices propios.
- El archivo .MRG contiene una lista de los archivos .MYI de índice procedentes de las tablas MyISAM integrantes.
- Sigue siendo necesario especificar los índices al crear la tabla MERGE.

Las tablas HEAP presentan las siguientes características:

- Utilizan un índice de asignación almacenado en memoria, que resulta muy rápido.
- Solo puede utilizar índices con los operadores = y <=>.
- No puede usar un índice en una columna que permita valores nulos.
- Los índices no se pueden utilizar con la cláusula ORDER BY.
- Mysql no puede determinar el número aproximado de filas que existen entre los dos valores (este resultado es utilizado por el optimizador de consultas para seleccionar el índice más eficaz que utilizar).

Las tablas ISAM utilizan un índice B-TREE almacenado en archivos con la extensión .ism.
Las tablas InnoDB no pueden utilizar índices de texto completo.

3.1.19 Uso eficaz de los índices

Las tablas con pocos índices devolverán los resultados muy rápido. Pero la inclusión de demasiados índices, aunque no suele ser normal, también ocasiona degradación de la base de datos. Los índices ocupan espacio de disco y, como están ordenados, cada vez que se realice una operación de inserción o de actualización, es necesario volver a organizar el índice para incluir los cambios, lo que da como resultado una carga de trabajo adicional significativa. La eficiencia en el uso de los índices depende también de la configuración de Mysql.

3.1.20 Dónde utilizar los índices

El uso más común de un índice consiste en recuperar filas que cumplan una

condición incluida en la cláusula WHERE.
Es muy importante crear el índice correcto, sobre el campo correcto, revisando siempre que forme parte de la condición.
Al buscar valores máximos o mínimos, Mysql, sólo necesita tomar el primer valor o el último de una tabla ordenada con un índice, lo que resulta extremadamente rápido.
Si se solicita con frecuencia valores máximos o mínimos, resulta muy útil crear un índice sobre el campo pertinente.

3.1.21 Sistema de prefijación más a la izquierda

Comenzando por la parte izquierda de la lista de campos del índice, Mysql puede utilizar cada uno de ellos, uno tras otro, siempre y cuando sigan la secuencia empezando por la izquierda.
Para una mejor comprensión de este concepto se va a utilizar el siguiente índice:
Tabla: Customer, INDEX(surname, inicial, first_name).
Si se realiza una consulta incluyendo los tres campos que forman parte del índice, lograríamos el mayor provecho del índice creado;
SELECT * FROM customer WHERE surname = 'Clegg' AND inicial = 'X' AND first_name = 'Yvonne'.

También sacaríamos el máximo partido del índice si buscamos por el apellido y la inicial.
SELECT * FROM customer WHERE surname = 'Clegg' AND inicial = 'X'.
O simplemente por el apellido:
SELECT * FROM customer WHERE surname = 'Clegg'.
Sin embargo, si se rompe la secuencia de disposición más a la izquierda y realiza la búsqueda por el nombre o la inicial o por ambos campos, Mysql no utilizará el índice:
SELECT * FROM customer WHERE inicial = 'X' AND first_name = 'Yvonne'.
SELECT * FROM customer WHERE first_name = 'Yvonne'.
SELECT * FROM customer WHERE inicial = 'X'.

Ninguna de las anteriores consultas utiliza el índice.

3.1.22 Selección de Indices

Se debe tomar en cuenta los siguientes consejos antes de seleccionar un índice:

- Los índices solo deben crearse en aquellas consultas que lo utilicen (por ejemplo, sobre campos de la condición WHERE) y no sobre campos que o vayan a utilizarlos (como en el caso en el que el primer carácter de la condición sea un comodín).
- Crear índices que devuelvan el menor número de filas posibles. El mejor lugar es sobre una clave primaria ya que estas se asignan de manera exclusiva sobre un registro. De manera similar, los índices sobre campos enumerados no resultan particularmente útiles (por ejemplo, un índice sobre un campo que contenga valores si o no, solo serviría para reducir la selección a la mitad, con toda la carga que supone el mantenimiento de un índice).

- Utilizar índices cortos (crear un índice sobre los 10 primeros caracteres de un nombre, por ejemplo, en lugar de hacerlo sobre el campo completo).
- NO se debe crear demasiados índices. Los índices aumentan el tiempo necesario para actualizar o agregar un registro. Por ello, si el índice se crea para una consulta que se utilice en contadas ocasiones y pueda aceptarse un rendimiento ligeramente más lento, considere la opción de no crear el índice.
- Utilice el sistema de prefijación más a la izquierda.

3.1.23 Optimización de Selecciones

Mientras mayor sea el número de tablas que se combinan, mayor será la cantidad de filas examinadas. Parte del buen diseño de las bases de datos consiste en hallar un equilibrio entre las tablas pequeñas de las bases de datos que necesitan más combinaciones y las tablas de mayor tamaño que resultan más difíciles de mantener.
Lo principal para que una consulta sea óptima es la selección correcta del índice y la cláusula WHERE correctamente definida, tomando en cuenta la prefijación más a la izquierda.
En Mysql un detalle importante es el comando EXPLAIN, que ayuda al programador a conocer en detalle que índice se está utilizando, cuantos registros revisó para sacar la consulta, como es procesada la instrucción SELECT, esta instrucción es de gran ayuda para escribir consultas optimas y seleccionar el índice adecuado a la consulta.
Cuando más complejos sean los permisos, mayor será la carga de trabajo que experimentan las consultas.

3.1.24 Optimización de actualizaciones, eliminaciones e inserciones

Una operación de actualización es prácticamente igual a una operación de selección con la diferencia de que se realiza una operación de escritura al final.
Es posible optimizar una instrucción UPDATE de la misma forma con la que haríamos con la instrucción SELECT. Así mismo, hay que tener en cuenta que cuando menor sea el número de índices y el número de datos, más rápida resultará la operación.
La velocidad de la instrucción DELETE depende del número de índices. Al eliminar registros, resulta necesario suprimir, cada uno de ellos de todos los índices.
El mejor método para insertar datos consiste en utilizar LOAD DATA en lugar de INSERT, ya que puede resultar 20 veces más rápido.

3.2 Oracle

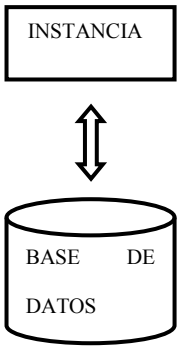
3.2.1 Introducción

Oracle es un sistema de administración de base de datos (o RDBMS Relational Data Base Management System por las siglas en inglés), fabricado por Oracle corporation, básicamente una herramienta cliente/servidor para la gestión de Bases de Datos. Es un producto vendido a nivel mundial, aunque la gran potencia que tiene y su elevado precio hace que sólo se vea en empresas muy grandes y multinacionales, por norma general. En el desarrollo de páginas web pasa lo mismo: como es un sistema muy

caro no está tan extendido como otras bases de datos, por ejemplo, Access, Mysql, Sql Server, etc.
Para desarrollar en Oracle utilizamos PL/SQL un lenguaje de 5ª generación, bastante potente para tratar y gestionar la base de datos, también por norma general se suele utilizar SQL.
Oracle es sin duda una de las mejores bases de datos que tenemos en el mercado, es un sistema gestor de base de datos robusto, tiene muchas características que nos garantizan la seguridad e integridad de los datos; que las transacciones se ejecuten de forma correcta, sin causar inconsistencias; ayuda a administrar y almacenar grandes volúmenes de datos; estabilidad, escalabilidad y es multiplataforma.
Aunque su dominio en el mercado de servidores empresariales ha sido casi total hasta hace poco, recientemente sufre la competencia de gestores de bases de datos comerciales y de la oferta de otros con licencia Software Libre como PostgreSql, Mysql o FireBird. Las últimas versiones de Oracle han sido certificadas para poder trabajar bajo Linux.

3.2.2 Aspectos generales de Oracle
3.2.3 Arquitectura de un servidor Oracle

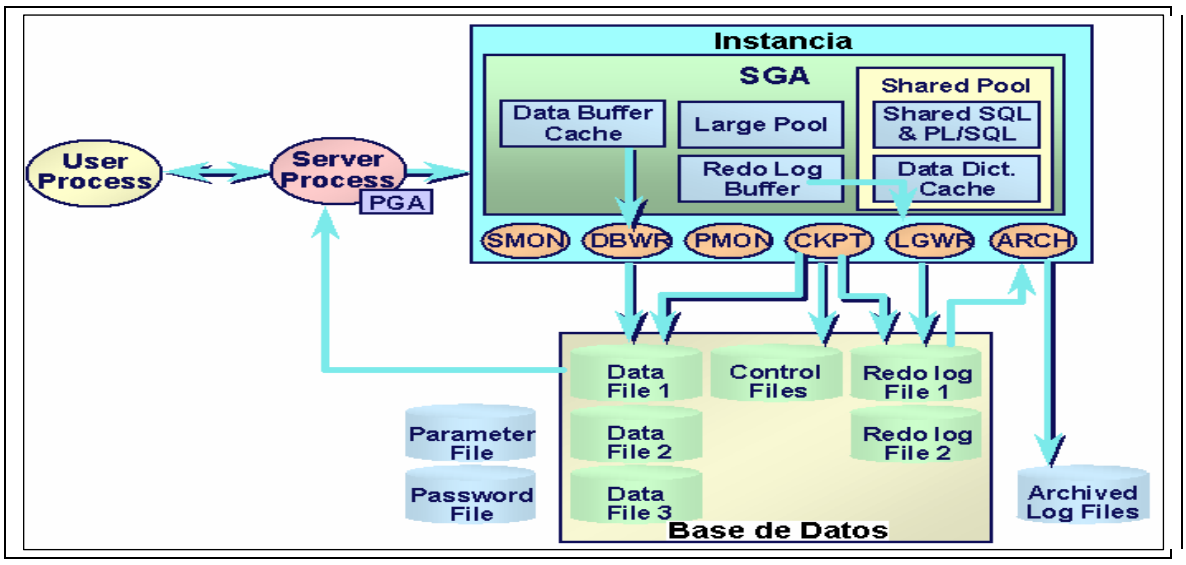
Gráfico # 5: Instancia de Oracle



Fuente: Manual de Oracle
Autor: Diego Burbano

Por cada instancia de Oracle se tiene una sola base de datos.
En un servidor se pueden crear varias instancias, pero no es recomendable ya que cada instancia consume muchos recursos.

Gráfico # 6: Diagrama de arquitectura Oracle



Fuente: Manual de Oracle
Autor: Diego Burbano

Una instancia de Oracle está conformada por varios procesos de fondo y espacios de memoria compartida denominada System Global Area (SGA) que son necesarios para acceder a la información contenida en la base de datos.
La instancia está conformada por procesos del usuario, procesos que se ejecutan en el background de Oracle y los espacios de memoria que comparten estos procesos.
El SGA es utilizado para el intercambio de datos entre el servidor y los clientes.
Una instancia de oracle solo puede abrir una sola base de datos a la vez.

3.2.3.1 Procesos de instancia

- **PMON.-** Process Monitor, Monitorea los procesos de los usuarios en caso de que la conexión falle. Su misión es monitorizar los procesos del servidor y tomar acciones correctivas cuando alguno de ellos se interrumpe en forma abrupta, limpiando la caché y liberando los posibles recursos que pudieran estar asignados en ese momento. También es responsable por el restablecimiento de aquel proceso que se ha interrumpido bruscamente.
- **SMON.-** System Monitor, este proceso es el encargado de recuperar la instancia y abrir la base da datos en caso de que ocurra alguna falla. Levanta una instancia cuando se le da la instrucción de partida (al comienzo del trabajo, encontrándose previamente en shutdown). Enseguida limpia los segmentos temporales y recupera las transacciones que pudieran haberse interrumpido debido a una falla del sistema. Además disminuye la fragmentación del sistema agrupando aquellas extensiones libres que existen dentro de la base de datos.
- **CKPT.-** CheckPoint Process, Sintoniza las tareas de grabación en la base de datos. Es el responsable de advertir al proceso DBWR de efectuar un proceso de actualización en el disco de los datos mantenidos en memoria,

incluyendo los datafiles y control files (para registrar el checkpoint). Este proceso es opcional, si no está presente, es el proceso LGWR quien asume la responsabilidad de la tarea.

- **DBWR.-** DataBase Writer, escribe los bloques de datos de la memoria en la base de datos. Es el responsable de la escritura en disco de toda la información almacenada en los buffers de bloques que no se han actualizado.
- **LGWR.-** Log Writer, graba los bloques del redo log del buffer a los archivos Redo Log File.
- **ARCH (archiver)**: La función de este proceso es la de respaldar la información almacenada en los archivos redo log cuando éstos se llenan. Este proceso está siempre activo cuando se ha establecido el modo ARCHIVELOG. Si el sistema no está operando en este modo se hace más difícil recuperar el sistema sin problemas luego de una falla general.

3.2.3.2 Area Global del sistema (SGA)

El SGA es un área de memoria compartida que se utiliza para almacenar información de control y de datos de la instancia. Se crea cuando la instancia es levantada y se borra cuando ésta se deja de usar (cuando se hace shutdown). La información que se almacena en esta área consiste de los siguientes elementos, cada uno de ellos con un tamaño fijo:

El buffer de caché (database buffer cache): almacena los bloques de datos utilizados recientemente (se hayan o no confirmado sus cambios en el disco). Al utilizarse este buffer se reducen las operaciones de entrada y salida y por esto se mejora el rendimiento.
El buffer de redo log: Guarda los cambios efectuados en la base de datos. Estos buffers escriben en el archivo físico de redo log tan rápido como se pueda sin perder eficiencia. Este último archivo se utiliza para recuperar la base de datos ante eventuales fallas del sistema.
El área shared pool: Esta sola área almacena estructuras de memoria compartida, tales como las áreas de código SQL compartido e información interna del diccionario. Una cantidad insuficiente de espacio asignado a esta área podría redundar en problemas de rendimiento. En resumen, contiene las áreas del caché de biblioteca y del caché del diccionario de datos.

- El caché de biblioteca se utiliza para almacenar código SQL compartido. Aquí se manejan los árboles de parsing y el plan de ejecución de las queries. Si varias aplicaciones utilizan la misma sentencia SQL, esta área compartida garantiza el acceso por parte de cualquiera de ellas en cualquier instante.
- El caché del diccionario de datos está conformado por un grupo de tablas y vistas que se identifican la base de datos. La información que se almacena aquí guarda relación con la estructura lógica y física de la base de datos. El diccionario de datos contiene información tal como los privilegios de los usuarios, restricciones de integridad definidas para algunas tablas, nombres y tipos de datos de todas las columnas y otra información acerca del espacio asignado y utilizado por los objetos de un esquema.

3.2.3.3 Area Global de programas (PGA)

Esta área de memoria contiene datos e información de control para los procesos que se ejecutan en el servidor de Oracle (relacionados con la base de datos, por supuesto). El tamaño y contenido de la PGA depende de las opciones del servidor que se hayan instalado.

3.2.3.4 La Base da Datos

Dentro de los procesos que forman parte de las base de datos tenemos:

- **Control File**, que contiene la información para controlar y mantener la integridad de la base de datos.
- **Data files**, son lo archivos en los cuales se almacenan los datos de las aplicaciones.
- **Redo Log Files**, almacena los cambios hechos en la base de datos con propósito de recuperarlos en caso de falla.

3.2.3.5 Estructuras adicionales

- **Parameter File**, contiene parámetros y valores que definen las características de la instancia y de la base de datos.
- **Password File**, se utiliza para validar al usuario que puede bajar y subir la instancia Oracle.
- **Achived Log Files**, son copias fuera de línea de los archivos Redo Log Files que son necesarios para el proceso de recovery en caso de falla del medio de almacenamiento.

3.2.4 Conexión a la instancia Oracle

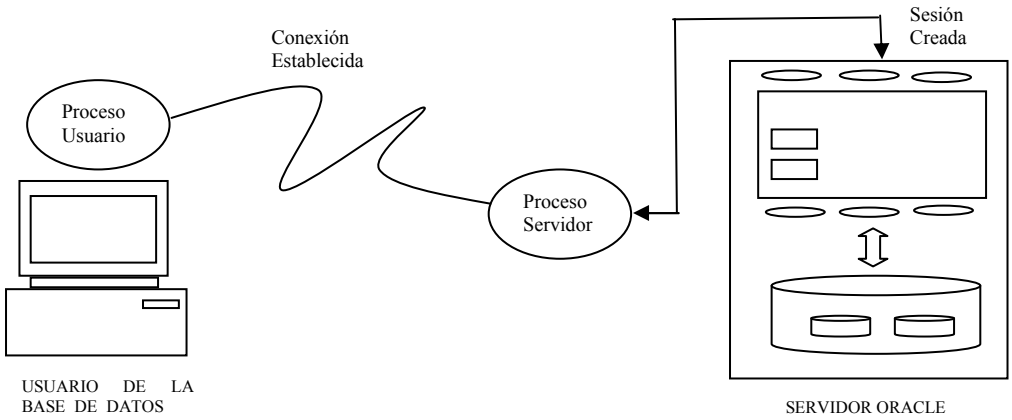
De la relación de servicios creados durante la instalación de Oracle, por ahora nos interesa básicamente dos.

- El servicio relacionado con la instancia y la base de datos, cuyo nombre tiene la siguiente estructura: OracleServiceXXX, donde XXX representa el nombre de la instancia.
- El servicio relacionado con la disponibilidad de servidor para el acceso remoto, el nombre del servicio es: OracleOraHome92TNSListener.

Para una correcta conexión, estos dos servicios deben estar ejecutándose.

3.2.4.1 Esquema General

Gráfico # 7: Esquema General

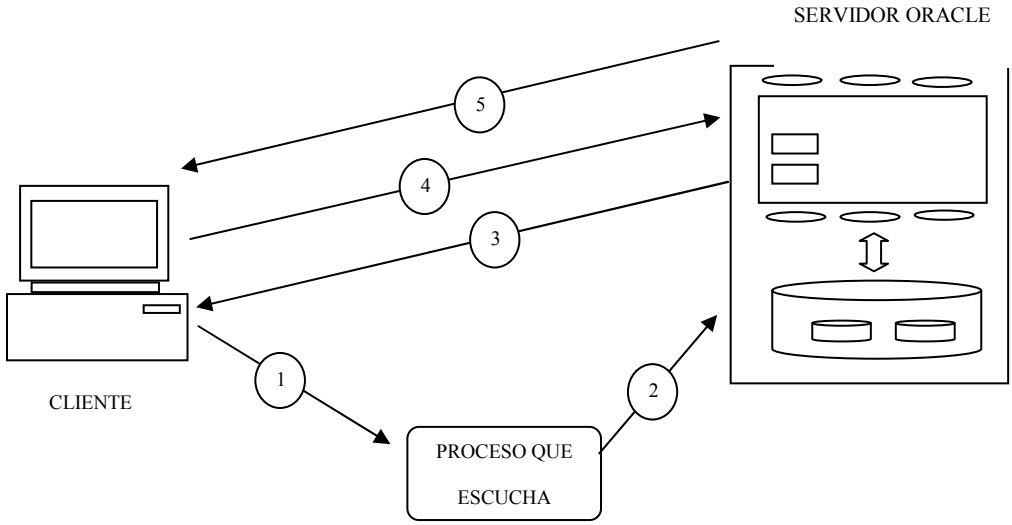


Fuente: Manual de Oracle
Autor: Diego Burbano

- **Proceso Usuario.** Programa, aplicación ó herramienta que usa el usuario para iniciar un proceso de usuario y establecer una conexión.
- **Proceso Servidor.** Una vez que el proceso del usuario estable ce conexión, un proceso servidor es iniciado, el cual manejará las peticiones del usuario. Un proceso servidor puede ser dedicado, es decir solo atiende las peticiones de un solo proceso usuario, ó puede ser compartido, con lo cual puede atender múltiples procesos usuario.
- **Sesión.** Una sesión es una conexión específica de un usuario a un servidor Oracle. Se inicia cuando el usuario es validado por el servidor Oracle. Finaliza cuando el usuario termina la sesión en forma normal (logout) ó aborta la sesión.

3.2.4.2 Conexión Remota

Gráfico # 8: Conexión Remota



Fuente: Manual de Oracle
Autor: Diego Burbano

Oracle tiene su herramienta de red que permite a las aplicaciones en general conectarse a servidores Oracle. Para que una aplicación pueda conectarse a un servidor Oracle, es necesario que el Proceso Escucha se encuentre ejecutándose en el servidor(OracleOraHome92TNSListener).
El esquema de conexión remota se puede apreciar en el gráfico # 8.
El proceso se describe a continuación:

1. El cliente establece una conexión al proceso Escucha usando el protocolo configurado y envía un paquete CONNECT.
2. El proceso Escucha comprueba que el SID esté definido. Si es así, genera un nuevo proceso para ocuparse de la conexión. Una conexión, se establece en el proceso Escucha y el nuevo proceso del servidor para pasarle la información del proceso de inicialización. Luego la conexión es cerrada.
3. El proceso del servidor envía un paquete al cliente.
4. Un nuevo paquete CONNECT es enviado al proceso servidor dedicado.
5. El proceso del servidor dedicado acepta la conexión entrante y remite un mensaje de ACEPTADO al nuevo cliente.

3.2.5 Tipos de datos

Los tipos de datos Oracle se agrupan en los siguientes conjuntos:

Tabla # 6: Tipos de datos Oracle

Alfanuméricos	Numéricos	Fecha	Binarios	Otros
CHAR	NUMBER	DATE	RAW	ROWID
VARCHAR(2)	FLOAT	TIMESTAMP	LONG RAW	UROWID
VARCHAR		TIMESTAMP WITH TIME ZONE	BLOB	
NCHAR		INTERVAL	CLOB	
NVARCHAR(2)			NLOB	
LONG			BFILE	

Fuente: Manual de Oracle
Autor: Diego Burbano

Los valores alfanuméricos van encerrados entre comilla simple.
Los valores numéricos son número simple: 123
Las fechas van encerradas entre comillas simples ejm: ‘14/09/1977’.
Los valores binarios no pueden ser representados, son videos, fotos.

Para los textos, oracle dispone de los siguientes tipos:

- VARCHAR(2), para textos de longitud variable de hasta 4000 caracteres.
- CHAR, para textos de longitud fija de hasta 2000 caracteres.
- NCHAR, para el almacenamiento de caracteres nacionales, de texto fijo.
- NVARCHAR(2), para el almacenamiento de caracteres nacionales de longitud variable.

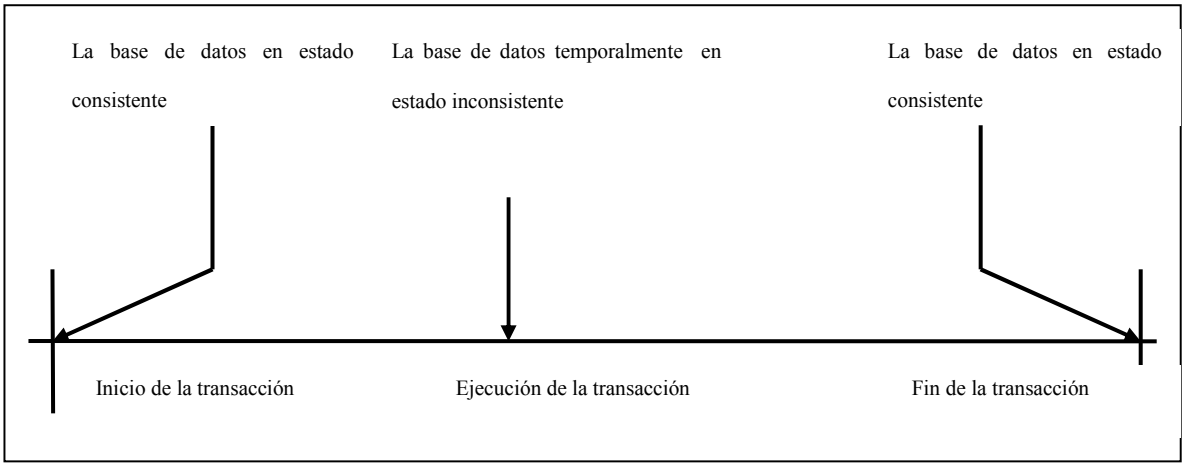
En todos estos tipos se indica el tamaño en paréntesis tras el nombre del tipo, este tamaño, en el caso de los tipos VARCHAR(2), es obligatorio, en el caso de los tipos CHAR son opcionales (si no se pone toma uno como default).
El tipo de dato Number es un tipo de dato versátil, que permite representar todo tipo de números de entre 10E-130 y 9,99999999999 * 10E128. Fuera de estos rangos, oracle devuelve un error.
Los números decimales, se indican con NUMBER (p, s), donde p es la precisión máxima y s es la escala.
El tipo de dato DATE, permite almacenar fechas. Las fechas se puede escribir en formato día, mes, año entre comillas. El separador puede ser /, - y casi cualquier símbolo.
El tipo de dato TIMESTAMP, es una extensión del anterior, almacena valores de día, mes, año, junto con la hora, minutos y segundos, con lo que representa un instante concreto en el tiempo.
El tipo de dato INTERVAL representa intervalos de tiempo.
El tipo de dato RAW sirve para almacenar valores binarios de hasta 2000 bytes (se puede especificar el tamaño máximo entre paréntesis).
LOB, son varios tipos de datos que permiten almacenar valores muy grandes. Incluye BLOB, CLOB, NCLOB y BFILE.
El tipo ROWID, es un valor hexadecimal que representa la dirección única de una fila en su tabla.

3.2.6 Transacciones y bloqueos

3.2.7 Transacciones

Se denomina transacción al espacio de tiempo que transcurre desde la primera sentencia DML que no sea SELECT (INSERT, UPDATE O DELETE) hasta que damos por finalizada la transacción explícitamente (con las sentencias apropiadas) o implícitamente (finalizando la sesión). Una base de datos está en estado consistente si, obedece todas las restricciones de integridad definidas sobre ella. Durante la transacción, todas las modificaciones realizadas sobre la base de datos, no son definitivas, más concretamente, se realizan en un TABLESPACE especial que se denomina ROOLLBACK o RBS (Roolback segment). Este tablespace, tiene reservado un espacio para cada sesión activa en el servidor, y es en este espacio donde, se almacenan cada una de las modificaciones de la transacción. Una vez que la transacción se ha finalizado, las modificaciones temporales almacenadas en el RBS, se vuelcan al tablespace original, donde está almacenada nuestra tabla. Esto permite que ciertas modificaciones que se realizan en varias sentencias, se puedan validar todas a la vez, o rechazar todas a la vez. Es importante asegurar siempre que la base de datos nunca este en un estado de inconsistencia. Sin embargo, durante la ejecución de una transacción, la base de datos puede estar temporalmente en un estado inconsistente. El punto importante aquí, es que la base de datos regrese al estado consistente al final de la transacción.

Gráfico #9: Estado transaccional de la base de datos

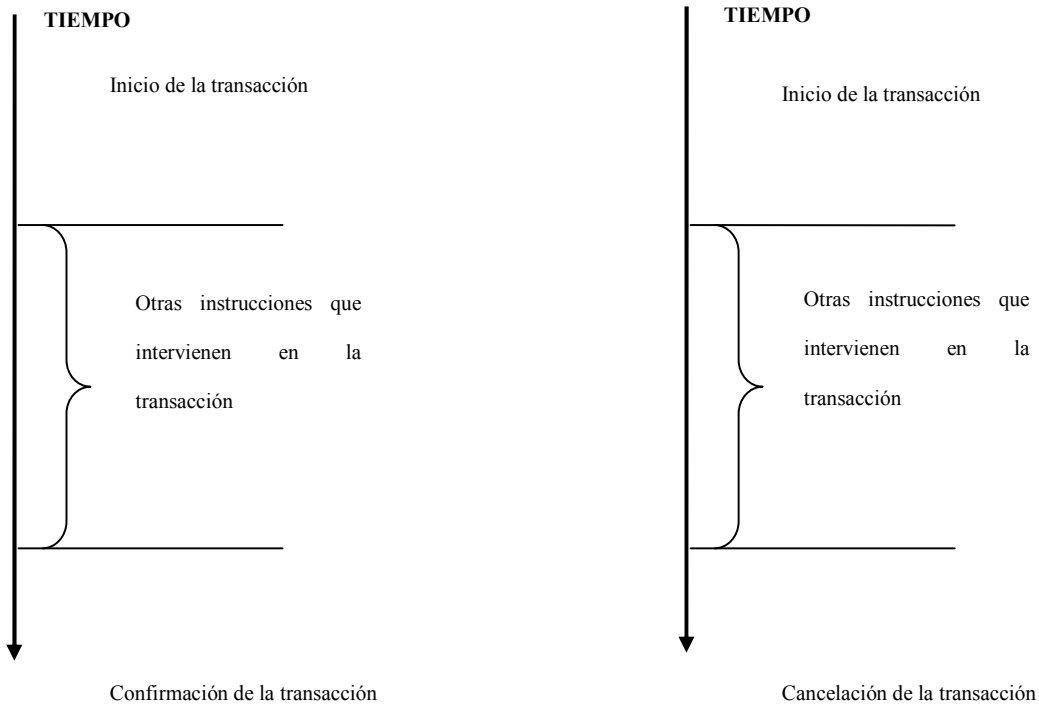


Fuente: Manual Administración de Oracle
Autor : Diego Burbano

3.2.7.1 Operación de transacciones

El siguiente gráfico, ilustra el funcionamiento de una transacción, cuando es confirmada y cuando es cancelada.

Gráfico #10: Ejecución de transacciones



Fuente: Manual Administración de Oracle
Autor : Diego Burbano

El inicio de una transacción es de manera automática cuando ejecutamos una sentencia insert, update o delete. La ejecución de cualquiera de estas sentencias da inicio a una transacción. Las instrucciones que se ejecuten a continuación formarán parte de la misma transacción.
Para confirmar los cambios realizados durante una transacción, utilizamos la sentencia COMMIT.
Para cancelar los cambios realizados durante una transacción, utilizamos la sentencia ROLLBACK.
Opcionalmente existe la sentencia SAVEPOINT para establecer puntos de transacción.
Si el número de sentencias es tan grande que el RBS se llena, Oracle hará un rollback, por lo que perdemos todos los datos. Así que es recomendable hacer COMMIT cada vez que el estado de la base de datos sea consistente.
Si terminamos la sesión, con una transacción pendiente, Oracle consultará el parámetro AUTOCOMMIT, y si este está en TRUE, hará COMMIT, si está en FALSE hará ROLLBACK.
Hay que tener en cuenta que cualquier instrucción DDL (como alter table por ejemplo), o una instrucción DCL (como grant), si es ejecutada durante la transacción, ese instante pasan a ser definitivas y finaliza toda la transacción.
También es importante tomar en cuanta que si el usuario que inicia la transacción, realiza una consulta a los datos, está, mostrará los datos ya modificados, el resto de usuarios, ve los datos tal y como estaban antes de la transacción, de hecho, los registros afectados por la transacción, aparecen bloqueados hasta que la transacción finalice. Tras la transacción, todos los usuarios ven los datos que quedan al final de la transacción. Los bloqueos son liberados y los puntos de ruptura borrados.

3.2.7.2 Bloqueos

Se pueden diferenciar dos clases de cloqueos:

- Un bloqueo de lectura que da acceso de solo lectura a un objeto y evita que cualquier otra transacción actualice el objeto, esta clase de bloqueo se llama a menudo, de lectura compartida puesto que varias transacciones pueden tener este tipo de bloqueo al mismo tiempo.
- Un bloqueo de escritura que otorga un acceso exclusivo de lectura-escritura y previene a la fuerza, que otras transacciones lean o escriban sobre el mismo objeto.

Es importante notar que no es necesario bloquear un registro de la base de datos durante la duración de la transacción ya que esto incrementaría la posibilidad de que ocurra un bloqueo mutuo (dead lock), por esta razón, el bloqueo se lo podría hacer solamente durante el acceso real al objeto.
Para resolver el conflicto creado por bloqueo mutuo, lo más usual es deshacer alguna de las transacciones. En Oracle se escoge aquella que tenga la menor cantidad de trabajo realizado.

3.2.7.2.1 Dead Lock

Consideremos el siguiente ejemplo:

- T1 efectúa un bloqueo de escritura en el objeto A.
- T2 efectúa un bloqueo de escritura en el objeto B.
- T1 solicita un bloqueo sobre el objeto B pero debe esperar porque T2 lo tiene bloqueado.
- T2 solicita un bloqueo sobre el objeto A pero debe esperar porque T1 lo tiene bloqueado.

En este punto, ni T1 ni T2 pueden proceder porque están bloqueadas mutuamente. La estrategia para resolver este tipo de problemas, es , permitir que los bloqueos mutuos ocurran, detectarlos y resolverlos. La técnica de detección de un bloqueo mutuo (dead lock) puede ser muy costosa si el nivel de granularidad es alto.
El bloqueo en Oracle es completamente automático y no requiere intervención del usuario. No obstante, también se le permite al usuario bloquear cuando lo considere necesario, deshabilitando el bloqueo por defecto.
La forma de hacerlo es:
LOCK TABLE tabla1 IN {SHARE | SHARE UPDATE | EXCLUSIVE} MODE [NOWAIT].

Candado.- impide que una transacción, accese a un registro reservado previamente por otra transacción, hasta que la transacción libere el registro o el campo.
Granularidad.- es el tamaño del item sobre el cual se ejerce el bloqueo. Este item, puede ser una página, bloque, una tabla o toda la base de datos.

3.2.8 Características relacionales orientadas a objetos

Oracle tiene soporte extensivo para constructores relacionales orientados a objetos, incluyendo:

- **Tipos de objetos**. Se soporta un único modelo de herencia para las jerarquías de tipos.
- **Tipos de colecciones**. Oracle soporta varrays, que son arrays de longitud variable, y tablas anidadas.
- **Tablas de objetos.** Se utilizan para almacenar objetos mientras se proporciona una vista relacional de los atributos de los objetos.
- **Funciones de tablas**. Son funciones que producen conjuntos de filas como salida y se pueden utilizar en la cláusula From de una consulta.
- **Vistas de objetos.** Proporcionan una vista de tablas de objetos virtuales de datos almacenados en una tabla relacional normal. Permite acceder a ver los datos en un estilo orientado a objetos incluso si los datos están realmente almacenados en un formato relacional tradicional.
- **Métodos.** Se pueden escribir en PL/SQL, Java o C.
- **Tipos de datos XML.** Se pueden utilizar para almacenar e indexar documentos XML.

Oracle tiene dos lenguajes procedimentales principales, PL/SQL y Java. PL/SQL fue el lenguaje original de Oracle para los procedimientos almacenados y tiene una sintaxis similar al utilizado en el lenguaje Ada. Java se soporta mediante una

máquina virtual Java dentro del motor de base de datos. Oracle proporciona un paquete para encapsular procedimientos, funciones y variables relacionadas en unidades únicas.

3.2.9 Disparadores

Oracle proporciona varios tipos de disparadores y varias opciones para el momento y forma en que se invocan. Los disparadores se pueden escribir en PL/SQL o java o como llamadas a C.
Para los disparadores que se ejecutan sobre instrucciones DML tales como insert, update o delete, Oracle soporta disparadores de filas (row) y disparadores de instrucciones (statement). Lo disparadores de filas se pueden ejecutar una vez por cada fila que se vea afectada (actualización o borrado, por ejemplo) por la operación DML. Un disparados de instrucciones, se ejecuta solamente una vez por instrucción. En cada caso, el disparador se puede definir tanto como un disparador "before" o "alter" dependiendo de si se va a invocar antes o después de que se lleva a cabo la operación DML.
Oracle también tiene disparadores que ejecutan otros eventos, tales como el inicio o finalización de la base de datos, mensajes de error del servidor, inicio o finalización de sesión de un usuario e instrucciones DDL tales como instrucciones create, alter o drop.

3.2.10 Almacenamiento e indexación

3.2.10.1 TableSpace (Espacio de tablas)

La base de datos Oracle se divide en unidades lógicas denominadas TABLESPACES. Cada TABLESPACE, consiste en una o más estructuras físicas denominadas DATAFILES.
Un TableSpace no es un fichero físico en disco, simplemente es el nombre que tiene un conjunto de propiedades de almacenamiento que se aplican a los objetos (tablas, secuencias…) que se crean en la base de datos bajo el tablespace indicado.

Un objeto en la base de datos debe estar almacenado obligatoriamente dentro de un tablespace.

Las propiedades que se asocian a un tablespace son:

- Localización de los ficheros de datos.
- Especificación de máximas cuotas de consumo de disco.
- Control de la disponibilidad de los datos (en línea o fuera de línea).
- Backup de los datos.

Cuando un objeto se crea dentro de un tablespace, este objeto adquiere todas las propiedades antes descritas del tablespace utilizado.

Si creamos por ejemplo una tabla llamada Artículo, esta se almacena dentro del tablespaceA, y que por lo tanto tendrá todas las propiedades del TableSpaceA que pueden ser:

- Sus ficheros de datos están en $ORACLE_HOME/datos/tablespaceA
- Los objetos no pueden ocupar más de 10Mb de espacio de la base de datos.
- En cualquier momento se puede poner fuera de línea todos los objetos de un cierto tablespace.
- Se puede hacer copia de seguridad solo de ciertos tablespace.

Si nos fijamos, se puede apreciar que es posible tener una tabla en un Tablespace, y los índices de esta tabla en otro tablespace. Esto es debido a que los índices no son más que objetos independientes dentro de la base de datos, como lo son las tablas. Y al ser objetos independientes, pueden ir en tablespaces independientes.
En las bases de datos Oracle encontramos el tablespace SYSTEM es un tablespace por defecto en todas las bases de datos Oracle. En él se almacenan todos los datos del sistema, el catálogo y todo el código fuente y compilado de procedimientos PL/SQL. También es posible utilizar el mismo tablespace para guardar datos del usuario. Este tablespace contiene además, el diccionario de datos, almacenamiento para disparadores y los procedimientos almacenados.
Por otra parte, también existe un tablespace Temporal. Este tablespace representa las propiedades que tendrán los objetos que la base de datos cree temporalmente para sus cálculos internos (normalmente para ordenaciones y agrupaciones).
El tablespace se puede crear con ciertas restricciones como por ejemplo un tablespace de solo lectura (Read Only), que por lo tanto, todos los objetos en el contenidos pueden recibir ordenes de consulta de datos, pero no de modificación de datos.
Un tablespace puede estar en línea o fuera de ella (OnLine o OffLine), esto es que el tablespace completo está a disposición de los usuarios o está desconectado para restringir el uso.

3.2.10.2 DataFile (Archivo de datos)

Un datafile es la representación física de un tablespace. Son los "Archivos de datos" donde se almacena la información físicamente.
Un datafile puede tener cualquier nombre y extensión (siempre dentro de las limitaciones del sistema operativo), y puede estar localizado en cualquier directorio del disco duro, aunque su localización típica suele ser $ORACLE_HOME/Database.
Un datafile tiene un tamaño predefinido en su creación (por ejemplo 100Mb) y este puede ser alterado en cualquier momento.
Cuando se cree un datafile, este ocupará tanto espacio en disco como hayamos indicado en su creación, aunque internamente esté vacío. Oracle hace esto para direccionar espacio continuo en disco y evitar así la fragmentación. Conforme se vayan creando objetos en el tablespace, se irá ocupando el espacio definido.
Un datafile está asociado a un solo tablespace y un tablespace está asociado a uno o varios datafiles.
Es decir, la relación lógica entre tablespaces y datafiles es de 1-N, maestro-detalle.
En la estructura manejada por Oracle un TableSpace está compuesto (físicamente) por uno o más. Estos datafiles son los ficheros físicos que soportan los objetos contenidos dentro del tablespace.
Aunque siempre se dice que los objetos están dentro del tablespace, en realidad las tablas están dentro del datafile, pero tienen propiedades asociadas al tablespace.
Cada uno de los datafiles utilizados está ocupado un tamaño en disco (50 mb los dos

primeros y 25 mb el último) aunque en realidad solo contenga dos objetos y estos objetos no llenen el espacio asignado para los datafiles.
Los datafiles tienen un propiedad llamada AUTOEXTEND, que si está activa se encarga de que el datafile crezca automáticamente (según un tamaño indicado) cada vez que se necesite espacio y no exista.
Al igual que los tablespaces, los datafiles también pueden estar en línea o fuera de ella.

3.2.10.3 Segment (Segmento, trozo, sección)

El espacio en un espacio de tablas se divide en unidades denominadas **segmentos,** cada una de las cuales contiene los datos para una estructura de datos específica.
Hay cuatro tipos de segmentos:

- **Segmento de datos.-** cada tabla en un espacio de tablas tiene su propio segmento de datos donde se almacenan los datos de la tabla a menos que ésta se encuentre dividida; si esto ocurre, existe un segmento de datos por división (regularmente esta división se da por partición de las tablas).
- **Segmento de índices.-** cada índice en un espacio de tablas posee su propio segmento de índices, excepto los índices divididos, los cuales mantienen un segmento de índice por división.
- **Segmentos temporales.-** son segmentos utilizados cuando una operación de ordenación necesita escribir datos al disco o cuando éstos se insertan en una taba temporal.
- **Segmento de retroceso.-** se trata de segmentos que contienen información para deshacer los cambios de las transacciones de forma de que pueda deshacer una copia no terminada. También juegan un papel muy importante en el modelo de control de concurrencia en Oracle y para la recuperación de la base de datos.

Capítulo 4.- COMPARATIVA DE LAS BASES DE DATOS

4.1 Comparativa General

La siguiente tabla muestra una comparación general de las bases de datos.

Tabla # 7: Comparativa general de las bases de datos

	Mysql	Oracle
Nombre de la Empresa	Mysql AB	Oracle Corporation
1er Realease Público	1996	1977
Ultima versión estable	5.0	10g realease 2
Tipo de licenciamiento	Gpl o Propietario	Propietario

Fuente: Internet
Autor: Diego Burbano

Mysql es muy popular por su velocidad de procesamiento, además el tipo de licencia que maneja es Gpl (General Public Licence).
Oracle es una base de datos robusta, es una de las bases de datos más confiables que existen en el mercado.

4.2 Sistemas Operativos soportados

Tabla # 8: Sistemas Operativos Soportados

	Peso	Mysql	Peso	Oracle	Peso
Windows	**10**	Si	10	Si	10
Linux	**10**	Si	10	Si	10
Mac OS x	**10**	Si	10	Si	10
Free BSD	**10**	Si	10		0
IBM AIX	**10**	Si	10	Si	10
Solaris	**10**	Si	10	Si	10
HP - UX	**10**	Si	10	Si	10
QNX	**10**	Si	10	Si	10
SCO Unix	**10**	Si	10	Si	10
Novell Netware	**10**	Si	10	Si	10
SGI Irix	**10**	Si	10	Si	10
	110		110		100

Fuente: Internet
Autor: Diego Burbano

Free BSD (Berkeley Software Distribution).- Es un sistema operativo open source de la familia de sistemas operativos Unix, fue desarrollado por Research Group, UC Berkeley.
Hp-UX (Hewlett Packard Unix).- es un sistema operativo de la familia Unix, desarrollado por Hewlett-Packard.
Novell Netwate.- es un sistema operativo de red desarrollado por Novell, Inc.
Sco Open Server.- La empresa que lo diseñó es The SCO Group, pertenece a la familia de sistemas operativos Unix, el tipo de licencia es propietaria, antes se llamaba Sco Unix desarrollado por Santa Cruz Operation (SCO) y ahora mantenido por Sco Group.
Iris.- La empresa que lo diseño es Silicon Graphics, pertenece también a la familia Unix.
Qnx.- la empresa que le desarrollo es QNX Software systems, el tipo de licencia es propietaria.

4.3 Interfaces (Api's) / Conectores soportados

Tabla # 9: Interfaces / Conectores Soportados

Interfaces					
	Peso	**Mysql**	**Peso**	**Oracle**	**Peso**
C	**10**	Si	10	Si	10
C++	**10**	Si	10		0
TCL	**10**	Si	10		0
Delphi	**10**	Si	10	Si	10
Perl	**10**	Si	10	Si	10
Python	**10**	Si	10		0
Php	**10**	Si	10	Si	10
Java	**10**	Si	10	Si	10
Ruby	**10**	Si	10		0
Conectores					
.Net	**10**	Si	10	Si	10
Odbc	**10**	Si	10	Si	10
Jdbc	**10**	Si	10	Si	10
	120		120		80

Fuente: Internet
Autor: Diego Burbano

Mysql ofrece los conectores indicados en la tabla que pueden ser usados para desarrollar aplicaciones utilizando mysql como base de datos. Cuando una aplicación es desarrollada con Php, Java, .net, perl, ODBC, Mysql dispone de un driver que se encarga de realizar este trabajo.
Java.- es una plataforma independiente, es un lenguaje orientado a objetos desarrollado por Sun Microsystems Inc. Un compilador java crea al código binario y la maquina virtual Java (JVM) convierte el código binario en lenguaje de máquina. Java al ser una plataforma independiente, los programas compilados en Java en una

plataforma, pueden ser utilizados en otra plataforma sin cambio alguno.
Oracle maneja PL/SQL que es un lenguaje de programación propio de Oracle, la base de datos incluye un compilador Java y JVM con la ingeniería de la base de datos. Esto permite a los desarrolladores escribir procedimientos almacenados, triggers y funciones en el estándar de programación Java incluido en lenguaje Pl/sql. Los desarrolladores compilan los programas Java directamente en la base de datos o leer una clase java utilizando la utilidad de Oracle llamada LoadJava. Mysql no permite almacenar o ejecutar programas java en la base de datos.

4.4 Comparativa de Características de las Bases de Datos

Tabla # 10: Comparativa de Características de las bases de datos

Categoria	Peso	Mysql	Peso	Oracle	Peso	Anexo
Versión		**5.0.18**		**10gR2**		
Seguridad	**100**		80		100	
Control de Acceso a Usuarios	**10**	Si	10	Si	10	Anexo 01
Backups	**10**	Si	10	Si	10	Anexo 02
Hot Backups	**10**	Si	10	Si	10	Anexo 02
SSL	**10**	Si	10	Si	10	Anexo 03
Importación / Exportación de Datos	**10**	Si	10	Si	10	
Privilegios de acceso a objetos	**10**	Si	10	Si	10	
Privilegios de Acceso en grupo	**10**	No	0	Si	10	
Perfiles de Usuario	**10**	Si	10	Si	10	
Encriptación de Datos	**10**	Si	10	Si	10	Anexo 04
Roles	**10**	No	0	Si	10	
Características Fundamentales	**110**		100		110	
ACID	**10**	Si	10	Si	10	Anexo 05
Bloqueo a Nivel de Registro	**10**	Si	10	Si	10	
Bloqueo a nivel de Tabla	**10**	Si	10	Si	10	
Bloqueo de lectura	**10**	No	10	No	10	
Writers lock readers	**10**	No	10	No	10	Anexo 06
Readers lock writers	**10**	No	10	No	10	
Escalating row level locking	**10**	No	10	No	10	
Bloqueo a Nivel de Campo	**10**	No	0	Si	10	
Procesamiento distribuido de Transacciones	**10**	Si	10	Si	10	Anexo 07
Detección de Deadlock	**10**	Si	10	Si	10	
Unicode	**10**	Si	10	Si	10	Anexo 08

Categoria	Peso	Mysql	Peso	Oracle	Peso	Anexo
Acceso a Datos	**160**		**150**		**160**	
Indices	**10**	Si	10	Si	10	Anexo 09
Diccionario de Datos	**10**	Si	10	Si	10	Anexo 10
Vistas	**10**	Si	10	Si	10	Anexo 11
Vistas Actualizables	**10**	Si	10	Si	10	
Secuencias	**10**	Si	10	Si	10	
SubSelect	**10**	Si	10	Si	10	
Triggers	**10**	Si	10	Si	10	Anexo 12
Creación de Nuevos Tipos de Datos	**10**	No	0	Si	10	
Tipos de datos especiales	**10**	Si	10	Si	10	
TableSpace	**10**	Si	10	Si	10	
Cursores	**10**	Si	10	Si	10	Anexo 13
Data Integrity	**10**	Si	10	Si	10	
SavePoint	**10**	Si	10	Si	10	
Uso de Alias	**10**	Si	10	Si	10	
Funciones	**10**	Si	10	Si	10	Anexo 14
Procedimientos Almacenados	**10**	Si	10	Si	10	Anexo 15
Alta Disponibilidad	**30**		0		30	
Incremental Backup	**10**	No	0	Si	10	
FlashBack Table	**10**	No	0	Si	10	Anexo 16
Recovery de transacciones erróneas	**10**	No	0	Si	10	Anexo 16
Funcionalidad	**150**		117		150	
Replica	**10**	Si	10	Si	10	Anexo 17
Cluster	**10**	Si	10	Si	10	Anexo 18
Particionamiento de Tablas	**10**	Si	10	Si	10	Anexo 19
Automatic Storange Managment	**10**	Si	10	Si	10	Anexo 20
DataWareHouse	**10**	Si	10	Si	10	
Federated Tables	**10**	Si	10	Si	10	Anexo 21
Grid Computing	**10**	No	0	Si	10	
Businness Intelligence	**10**	Si	10	Si	10	
Gateways	**10**	No	0	Si	10	
Xml	**10**	Si	10	Si	10	Anexo 22
Características de Orientación a Objetos	**10**	No	0	Si	10	Anexo 23
Precision Math	**10**	Si	10	Si	10	Anexo 24
Herramientas de Migración de datos	**10**	Si	10	Si	10	
Identificar Sentencias Sql mal programadas	**10**	Si	7	Si	10	
Archive Engine	**10**	Si	10	Si	10	Anexo 25

Autor: Diego Burbano

4.5 Aplicaciones Administrativas de Mysql

Tabla # 11: Aplicaciones Administrativas de Mysql

Aplicación	Descripción
Mysql Migration Toolkit	Mysql Migration Toolkit administra y realiza tareas de migracion de datos a Oracle, Sql Server, Access, Jdbc Generico .
Mysql Administrator	Administra las instancias, backup, usuarios, performance, logs, replica
Mysql Workbench	Es el nuevo dbDesigner con al cual se puede hacer el proceso de ingenieria inversa
Mysql Monitor	Un Monitor del Servidor y manejador de las instancias
Mysql Query Browser	Administra las consultas Sql, creación de procedimientos, vistas, manipulación de sripts, ayuda sobre la sintaxis de mysql
Plone	Herramienta Case
Ruby on Rails	Herramienta Case, Open Source Web framework que optimiza la programación y productividad, es una herramienta que sirve para desarrollar aplicaciones Web con bases de datos.

Autor: Diego Burbano

4.6 Aplicaciones Administrativas de Oracle

Tabla # 12: Aplicaciones Administrativas de Oracle

Aplicación	Descripción
Enterprice resource Planning (ERP)	Finanzas, proyestos, Rrhh, nominas, Mantenimiento, distribución y manufactura.
Customer Relationship Management (CRM)	Ventas, servicio, mercadotecnia, Call Centers
Supply Chain Management (SCM)	Administración de la cadena de suministro, desarrollo, planeación, Procura, manufactura.
Business Inteligence	Balance Score Card, Activity Base Management, Inteligencia Operativa, Portal ejecutivo.
Oracle Developer	Forms Developer, Jdeveloper, Reportes
Oracle Designer	Creación y desarrollo de la ingenieria de Software
Oracle WareHouseBuilder	
Oracle Discoverer	
Oracle Enterprice Manager (OEM)	Es una aplicación gráfica ue está incluida en todas las ediciones, los administradores la usan para manejar las instancias de Oracle.
Oracle Express Server	
People Soft	

Autor: Diego Burbano

4.7 Comparativa de Tipos de Datos

Tabla # 13: Comparativa de Tipos de Datos

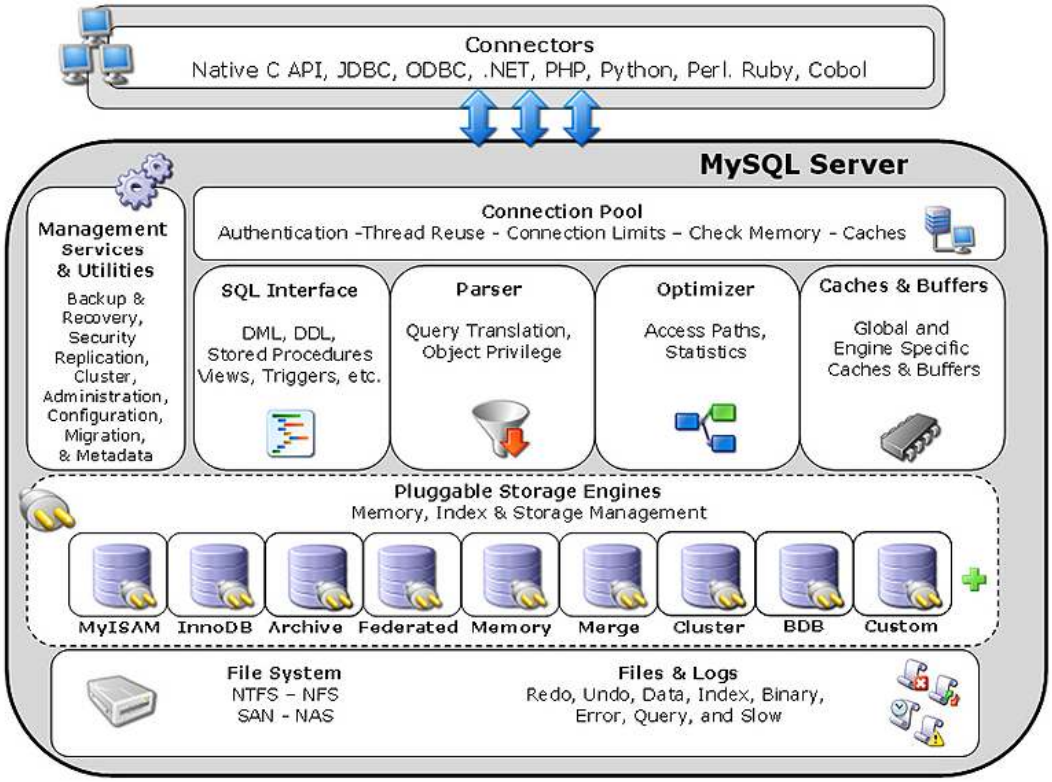
Tipo de Dato	Peso	Mysql	Peso	Oracle	Peso	Descripción
Cadena de longiud fija	10	Sobre 255 bytes	8	Sobre 2000 Bytes	10	Oracle soporta más volumen de información
Cadena de longitud variable	10	Sobre 255 bytes	8	Sobre 4000 Bytes	10	Oracle soporta más volumen de información
Long Text	10	Sobre 4 gigabytes. Limitado a 16 mb con algunas tablas.	8	Sobre 4 gigabytes	10	Oracle no tiene límite de tamaño para objetos Long Text.
Large Binary	10	Sobre 4 gigabytes. Limitado a 16 mb con algunas tablas.	8	Sobre 4 gigabytes	10	Oracle no tiene límite de tamaño para objetos binarios grandes.
Integer	10	Sobre 2e64 digitos aproximadamente y 20 digitos de precisión.	8	Sobre 38 digitos de precisión.	10	Oracle soporta enteros más grandes
Floating Point	10	-1.8e308 a -2.2e308, 2.23-308 a 1.83308'	10	Sobre 38 digitos de precisión.	8	Mysql Soporta enteros de punto flotante y de presición doble más grandes.
Date	10	Fecha y hora con precisión de segundos de Enero 1,1000 a Diciembre 31, 9999	8	Fecha y hora con precisión de segundos de Enero 1,4712 BCE a Diciembre 31, 4712 CE	10	Oracle asegura que los valores de fecha sean fechas válidas y puede almacenar fechas antes de 1000 CE. Mysql puede almacenar fechas después de 4712 CE.
Time	10	Hora con presición de segundos. -838:59:59 a 838:59:59	10	N/A	10	Oracle no tiene un tipo de dato Time
TimeStamp	10	Precisión de segundos sin local time zone.	8	Precisión de milisegundos con local time zone.	10	Oracle soporta milisegundos y conversiones automáticas de time zone.
Year	10	1901 a 2155	10	N/A	10	Oracle no tiene un tipo de dato para almacenar solo años.
Row Identifier	10	N/A	10	Local y universal	8	Mysql no soporta identificadores de fila que son utiles para actualizaciones rápidas.
Enumeration	10	Sobre 65535 valores diferentes.	10	N/A	10	Oracle no soporta este tipo de dato; sin embargo soporta constraints de chequeo de multiples valores
Set	10	Puede contener de cero a 64 valores.	10	N/A	0	Oracle no tiene este tipo de dato.
Tipos de dato definidos por el usuario	10	N/A	0	Los desarrolldores pueden definir sus propios tipos de datos	10	Oracle soporta tipos de datos complejos.
	140,00		116		126	

Autor: Diego Burbano

4.8 Almacenamiento de Datos

4.8.1 Almacenamiento de Datos en Mysql

Gráfico #11: Arquitectura de Almacenamiento en Mysql



Fuente: Manual de Mysql
Autor: Diego Burbano

Esta eficiente arquitectura provee beneficios para el tipo de aplicación que se necesite como data warehousing, procesamiento de transacciones, situaciones de alta disponibilidad, etc. Todo para que el motor de la base de datos funcione eficientemente. Esto va de la mano con las ventajas de utilizar un set de interfaces y servicios.

El programador de aplicaciones y el dba pueden interactuar con la base de datos a través de los conectores Api's y capas de servicios que se están disponibles para los tipos de almacenamiento. Es posible cambiar entre los tipos de almacenamiento sin necesidad de modificar mucho código o procesos durante el cambio.

4.8.1.1 Ingeniería de almacenamiento Mysql (Pluggable Storange Engine)

Es el componente de Mysql que es el responsable de administrar las operaciones de entrada/salida de la base de datos, así como de habilitar y reforzar ciertas características de acuerdo a la particular aplicación, esto hace más eficiente y alto al performance de la base de datos. Esta es una de las razones por las que Mysql es ahora conocido por tener alto performance comparado con las bases de datos

propietarias.

Desde una perspectiva técnica, los componentes de almacenamiento son:

- **Concurrencia.-** algunas aplicaciones requieren de estrategias de bloqueos más rígidas que otros (por ejemplo, bloqueo a nivel de fila). Seleccionar la correcta estrategia de bloqueo puede reducir en gran medida la saturación del sistema y ayuda en mucho al performance de la base de datos. Esta área incluye soporte para características como control de concurrencia multiversión o "snapshot" read.

- **Soporte de transacciones.-** no todas las aplicaciones necesitan transaccionalidad, pero para las que lo usan, está muy bien definido la característica como ACID compliance.

- **Integridad referencial.-** la necesidad de tener en un servidor de base de datos relacional integridad referencial a través de claves foráneas y DDL.

- **Almacenamiento Físico.-** esto abarca todo lo que tiene que ver con el tamaño de página para tablas e índices, así como el formato usado para guardar los datos físicamente en disco.

- **Soporte de indexamiento.-** los diferentes escenarios n las aplicaciones, se benefician de las diferentes estrategias de índices, y cada uno de los tipos de almacenamiento tiene generalmente sus propios métodos de indexamiento (b-tree por ejemplo).

- **Caché de Memoria.-** las diferentes aplicaciones responde mejor de acuerdo a la estrategia de almacenamiento en caché que utilizan.

4.8.1.2 Comparación de los diferentes tipos de almacenamiento en Mysql

Tabla # 14: Comparativa de almacenamiento de Datos en Mysql

Característica	MyIsam	BDB	Memory	InnoDb	Archive	NDB
Limites de Almacenamiento	x	x	Si	64TB	x	Si
Transacciones (Commit, roolback, etc.)	x	Si	x	Si	x	Si
Granularidad de Bloqueo	Tabla	Página	Tabla	Registro	Registro	Registro
MVCC (Multi-version concurrency control) / Snapshot Read	x	x	x	Si	Si	Si
Soporte Geoespatial	Si	x	x	x	x	x
Indices B-Tree	Si	Si	Si	Si	x	Si
Indices Hash	x	Si	x	Si	x	Si
Indice de búsqueda Full text	Si	x	x	x	x	x
Clustered Index	x	x	x	Si	x	x
Cache de Datos	x	x	Si	Si	x	x
Cache de Indices	Si	x	Si	Si	x	Si
Compresión de Datos	Si	x	x	x	Si	x
Encriptación de datos	Si	Si	Si	Si	Si	Si
Costo de Almacenamiento (espacio utilizado)	Bajo	Bajo	N/A	Alto	Muy Bajo	Alto
Costo de Memoria	Bajo	Bajo	Medio	Alto	Bajo	Alto
Velocidad de Inserción	Alto	Alto	Alto	Bajo	Muy Alto	Alto
Cluster DataBase Support	x	x	x	x	x	Si
Soporte para Replica	Si	Si	Si	Si	Si	Si
Soporte para claves foraneas	x	x	x	Si	x	x
Backup / Point-in-time Recovery	Si	Si	Si	Si	Si	Si
Soporte para Query Cache	Si	Si	Si	Si	Si	Si
Estadísticas de Actualización para diccionario de datos	Si	Si	Si	Si	Si	Si

Fuente: Mysql
Autor: Diego Burbano

Mysql ofrece varios tipos de almacenamiento propios de Mysql que han sido diseñados de acuerdo a las necesidades de las aplicaciones, esto permite a los profesionales de bases de datos, seleccionar un tipo de almacenamiento especializado a las necesidades de una aplicación particular. Una ellas es Innodb, la cual permite el manejo de transacciones, MyIsam es otra que contiene características y capacidades que soportan alto tráfico en ambientes web. Al ser una herramienta Open Source, lo clientes pueden crear su propia ingeniería de almacenamiento que se ajuste a sus requerimientos.

Como se puede ver en la figura, tenemos varios tipos de almacenamiento que pueden ser usados con el servidor de base de datos:

- **MyIsam**, es el default de los tipos de almacenamiento, usa indexamiento B-tree, es uno de los más usados en aplicaciones web, data warehousing y otros tipos de aplicación. Nótese que el tipo de almacenamiento puede ser fácilmente cambiado alterando la configuración de la variable STORAGE_ENGINE.
- **InnoDb,** usado para aplicaciones transaccionales, soporta algunas

características incluida ACID, boqueo a nivel de fila, los usuarios que escriben información no bloquean a otros usuarios que leen información y viceversa, soporta claves foráneas.

- **BDB,** es otra alternativa de manejo transaccional como innodb que soporta Commit, Rollback y otras características transaccionales.
- **Memory,** almacena todos los datos en RAM para proporcionar un acceso rápido en ambientes que requieren una vista rápida de la información.
- **Merge,** permite al DBA o al desarrollador, realizar un grupo lógico de series de tablas MyIsam idénticas y referenciadas estas como un único objeto. Bueno para ambientes VLDB como data warehousing.
- **Archive,** provee una solución perfecta de almacenamiento y recuperación de grandes tablas históricas, archivos o información de auditoria.
- **Federated,** ofrece la habilidad de realizar una especie de link entre los datos de servidores Mysql creando una base de datos lógica de varios servidores ubicados físicamente en distintos lugares. Muy bueno para ambientes distribuidos con un manejo inteligente de los datos.
- **Cluster/NDB,** fue particularmente creado para ambientes con alto nivel de rendimiento que requieren un alto nivel de disponibilidad.
- **Otras,** incluye otros tipos de almacenamiento, incluye csv (archivos separados por comas como tablas en la base de datos), también incluye un tipo de almacenamiento de ejemplo que pude ser usado para iniciar el proceso de creación de un tipo nuevo de almacenamiento.

En la tabla # 14 tenemos un análisis más completo de lo que soporta cada uno de los tipos de almacenamiento, esta tabla tiene algunas de las características más importantes, pero en el manual de referencia existe más información.
Es posible utilizar varios tipos de almacenamiento en una aplicación, mysql no limita el uso de un tipo de almacenamiento en una base de datos particular. Para que la aplicación funcione siempre con el máximo rendimiento lo más importante es seleccionar adecuadamente el tipo de almacenamiento a usar.
Por ejemplo, para una aplicación de Business Inteligente (BI), la aplicación tendría los siguientes requerimientos:

- Pesadas consultas y lectura de datos con el mínimo tiempo entre lecturas.
- Las operaciones son solo de lectura por naturaleza, esto implica que no se necesita altos niveles de concurrencia de datos, solamente el bloqueo compartido puede ser usado entre lecturas .
- La aplicación puede usar un host en una interfase web que requiere la capacidad de búsqueda full-text.
- El soporte de transacciones no es necesario.
- La integridad referencial de los datos es asumida por el código del sistema transaccional.
- Los datos deben ser repicados a varios sitios geográficos.

Para este tipo de aplicación, la opción correcta puede ser MyIsam.

4.8.2 Almacenamiento de datos en Oracle

Oracle utiliza Espacios de tablas (tablespaces) que consiste en uno o más archivos de datos (datafiles) en disco para cada una de las tablas creadas. Cada tablespace puede

usar un diferente tamaño de bloque (Block Size) para optimizar el almacenamiento. Todas las tablas mantienen la misma estructura en la base de datos. Oracle sopota tablas temporales para una sesión individual o global para todos los usuarios. Oracle también soporta el manejo de tablas externas que permite a los usuarios realizar consultas en archivos planos como si los datos estuvieran en la misma base de datos.

Sistema de Archivos:
Archivos de Control
TableSpaces
Archivos de datos
Segmentos de rollback
Grupos RedoLog

4.9 Comparativa de Soporte

4.9.1 Oracle

El servicio de soporte Oracle incluye:

- Asistencia técnica las 24 horas al día 7 días a la semana (24 x 7)
- Acceso a OracleMetaLink – sistema de soporte al cliente basado en el internet.
- Habilidad para registrar requerimientos a través de OracleMetaLink.
- Actualizaciones de software, versiones de mantenimiento.

Asistencia técnica a los problemas y preguntas de los clientes con relación a los productos oracle. Se presta a través de su CSI (código de identificación de soporte) desde el centro de soporte en Orlando, al cual se accede por llamada telefónica ó por la conexión con Oracle MetaLink. Este proceso requiere el registro por parte del cliente de las solicitudes de asistencia técnica (TAR's) la cuales se transfieren directamente y se resuelven por analistas técnicos que tienen acceso a un amplio rango de herramientas de software de diagnóstico. Cada TAR recibe una prioridad establecida por el analista y el cliente, en función del impacto que cause en el negocio del cliente. Todas las solicitudes son registradas, procesadas, solucionadas y sólo se cierran cuando el analista y cliente así lo acuerden.
Actualizaciones de productos (upgrades) y versiones de mantenimiento. Oracle tiene el compromiso de la continua investigación y desarrollo de mejoras a sus productos que aseguren a sus clientes cumplir con los continuos retos que enfrentan es su negocio, por medio de las actualizaciones de productos.

4.9.2 Mysql

Mysql Network provee soporte y servicios directos de los desarrolladores de Mysql asegurando una respuesta confiable, segura y rápida. En la página web las personas interezadas en la herramienta tienen acceso a listas de correo, foros, documentación en línea, información técnica de temas específicos.
El soporte de Mysql se divide en tres tipos para los clientes con aplicaciones de base de dato crítica:

Mysql Network Basic.- Provee consultores independientes para pequeños negocios con servicios que mejoran la productividad. Tiene acceso a software certificado y optimizado, mantenimiento, updates, ingreso a la base de conocimiento de Mysql. Si tiene incidentes, puede realizar un máximo de dos incidencias, acceso web y tiene un tiempo máximo de respuesta de 2 días. Este tipo de soporte cuesta $595.00 por servidor por año.

Mysql Network Silver.- provee los mismos servicios que el básico en lo que se refiere a software, pero en la parte de resolución de problemas, puede tener un numero de incidentes ilimitado, acceso web, acceso telefónico 8x5 y un limite de 4 horas en el tiempo de respuesta. El costo es 1995.00 por servidor y por año.

Mysql Network Gold.- tiene las mismas particularidades del básico y plata, pero en la resolución de problemas tiene numero de incidentes ilimitado, acceso web, acceso telefónico 24x7, máximo tiempo de repuesta a incidencias 2 horas, además, en caso de emergencias, tiene un tiempo máximo de respuesta de 30 minutos, también dispone de resolución de problemas en forma remota. El costo es de 2995.00 por servidor por año.

Mysql Network Platinum.- Además de las particularidades de software de las anteriores, tiene como opción el desarrollo de proyectos especiales, tiene un número de incidentes ilimitado, acceso web, acceso a llamadas telefónicas 24x7, máximo tiempo de respuesta en incidencias 30 min, en caso de emergencia tiene también 30 minutos de tiempo base. Para soporte de consultaría tiene resolución de problemas remotamente, revisión de código, revisión de performance, revisión de código para manejar las diferentes api's que soporta Mysql y funciones definidas por el usuario. El costo es de 4995.00 por servidor por año.

4.10 Comparativa de Licenciamiento

Tabla # 15: Comparativa de Licenciamiento

Tipo de Licenciamiento	Mysql	Oracle
Gpl	Si	No
Comercial	Si	Si

Autor: Diego Burbano

4.11 Empresas que utilizan Oracle

General Motors, General Electric, Intel Corporation, Hp, Philips, Nike, Banco de crédito del Perú, Alcatel, British Gas, Andinatel Ecuador, Mercado Libre, Mastercard Internacional, Boing, Sri Ecuador, Concep, Superintendencia de Bancos del Ecuador, Toyota Casa Baca, Iess.
El mercado de la base de datos Oracle no está segmentado solo a un tipo de aplicaciones, como se puede observar, existen entidades financieras, empresas de comunicaciones
Comunicaciones, portales web.

4.12 Empresas que utilizan Mysql

Dentro de las empresas que utilizan Mysql en sus ambientes de producción tenemos las siguientes:
Sonny, Suzuki, Lycos, Yahoo, Dell, PortaOne, Nasa, Unicef, Mcafee, Aizawa Securities, Google.
Como se puede observar, la mayoría son empresas que tienen su aplicación en el internet con base de datos, es decir, aplicaciones web.

4.13 Calidad del Software Mysql

Un proyecto financiado por el departamento de seguridad interna de EEUU con 1.2 millones de euros, es un proyecto completamente independiente (http://www.whitehouse.gov/pcipb/). El estudio realizado por la empresa Coverity (http://www.coverity.com/main.html) especializada en auditoria de código y evaluación de riesgos y Symantec, en colaboración con la universidad de Stanford. Este proyecto, a parte de los resultados es un proyecto completamente independiente no como otros que han sido financiados por empresas y consultoras afines a una tendencia u otra. En este caso la finalidad era la de saber la verdad sobre el software libre para que ayude a la toma de decisiones.
Para el proyecto se analizaron 32 proyectos de software libre uno por uno frente a las aplicaciones propietarias, entre ellos la base de datos libre Mysql, la conclusión fue que el software libre tiene menos errores de programación que el software propietario.
Software libre: 0.434 errores por cada 1,000 líneas de código
Propietario: de 20 a 30 errores por cada 1,000 líneas de código
Incluso en el caso más desastroso, se ha comprobado que en muchas ocasiones las Comunidades de Software Libre tienen pocos problemas en hacer cambios mayores en el código, incluso partiendo desde cero, para solventar errores graves de diseño, cosa que es complicado ver en productos propietarios en producción, que arrastran errores hasta el final del ciclo de vida.

4.14 Precios Oracle

Tabla # 16 Precios Oracle

Producto	Usuario Nombrado	Update, Licencia y Soporte	Licencia por procesador	Update, Licencia y Soporte
Base de datos				
Standard Edition One	149.00	32.78	4995	1098.90
Standard Edition	300.00	66.00	15000	3300.00
Enterprice Edition	800.00	176.00	40000	8800.00
Opciones de Enterprice Edition				
Real application Cluster	400	88.00	20000	4400.00
Partitioning	200	44.00	10000	2200.00
OLAP	400	88.00	20000	4400.00
Data Mining	400	88.00	20000	4400.00
Spatial	200	44.00	10000	2200.00
Advanced Segurity	200	44.00	10000	2200.00
Label Segurity	200	44.00	10000	2200.00
Aplicaciones para Mantenimiento				
Diagnostic Pack	60	13.20	3000	660.00
Tunning Pack	60	13.20	3000	660.00
Change Management Pack	60	13.20	3000	660.00
Configuration Management Pack	60	13.20	3000	660.00
Data WareHouse				
Express Server	800	176.00	40000	8800.00
Express Analizer	800	176.00	n/a	n/a
Express Objects	5000	1100.00	n/a	n/a
Intenet Application Server				
Standard Edition One	149	32.78	4995	1098.90
Standard Edition	200	44.00	10000	2200.00
Enterprice Edition	600	132.00	30000	6600.00
Businness Inteligence	400	88.00	20000	4400.00

Fuente: Global Price List Oracle 6 Enero 2006
Autor: Diego Burbano

Dentro de los productos más vendidos de Oracle, tenemos los relacionados con la base de datos, es decir,

Oracle Database Standar Edition One
Oracle Database Standar Edition
Oracle Database Enterprice Edition

Claro está que cada uno de ellos tiene acogida en mercados muy diferentes.

Oracle Database Standar Edition One.- Este producto está orientado a empresas pequeñas, para poder utilizar esta versión, el equipo sobre el que va a ser instalado no puede contar con capacidad para más de dos procesadores. Como se puede apreciar en la tabla, el costo de este producto es de $4995 por procesador (máximo dos), o de $149 por usuario nombrado (mínimo 5).

Oracle Database Standar Edition.- Este producto cuenta con las mismas características que la versión Standar Edition One, pero esta puede ser configurada en un equipo con capacidad hasta 4 procesadores, el costo es de $15000 por procesador (máximo 4) o de $300 por usuario nombrado (mínimo 5).

Oracle Database Enterprice Edition.- Orientado a empresas grandes o que requieren características especiales, puede ser configurado sin restricción de procesadores, y su costo es de $40000 por procesador o de $800 por usuario (mínimo 25 usuarios por procesador instalado).

Los clientes de Oracle Database Enterprice Edition encuentran muy útil licenciar de manera adicional opciones como:

Real Application Cluster
Partitioning
Diagnostic Pack
Tunnig Pack
Change Management Pack
Configuration Management Pack

Real Application Cluster.- Su finalidad es poder tener múltiples servidores soportando a la misma base de datos, tiene un costo de $20000 por procesador, o de $400 por usuario nombrado.
Obviamente el número de procesadores o usuarios a licenciar debe ser igual al número de procesadores o usuarios licenciados para Oracle Enterprice Edition.

Partitioning.- Una opción que permite administrar las tablas grandes de la base de datos en diferentes tablespaces, para ayudar al rendimiento de la base de datos. Su coso es de $10000 por procesador o $200 por usuario nombrado.

Orcle Packs.- Diagnostic, Tunning, Change Management, Configuration Managemet son características adicionales que permiten facilitar la administración del motor de base de datos Oracle, así como potenciar su rendimiento. El costo de cada uno de estos 4 paquetes es de $3000 por procesador o de $60 por cada usuario nombrado.

Otro de los productos que tienen buena salida es el Internet Application Server:

Internet Application Server Standar Edition One

Internet Application Server Standar Edition
Internet Application Server Enterprice Edition

Internet Application Server Standar Edition One.- A un costo de $4995 (máximo 2cpu) por procesador o de $149 (mínimo 5 usuarios) por usuario nombrado, está orientado a empresas pequeñas que requieran un servidor de aplicaciones, se puede configurar en equipos de máximo 2 procesadores.

Internet Application Server Standar Edition.- Para empresas un poco más grandes cuyo servidor pueda escalar hasta 4 CPUs a un costo de $10000 por CPU (máximo 4), o de $200 por usuario nombrado (mínimo 5).

Internet Application Server Enterprice Edition.- Para empresas grandes, no tiene límite de escalabilidad a nivel de CPUs, y tiene un costo de $30000 por CPU o de $600 por usuario nombrado.
A nivel de servidor de aplicaciones, existe un sin número de opciones pero ninguna que se venda en forma masiva, son todas más bien puntuales para el cliente.
Otro producto que se vende es el:

Developer Suite.- Para aquellos encargados del desarrollo de aplicaciones a un costo de $5000 por usuario nombrado (no se puede licenciar por procesador).

4.15 Precios Mysql

Mysql es Glp (General Public Licence) no tiene costo, en lo que gana la empresa Mysql Ab es en el soporte y entrenamiento.
Al ser una empresa que maneja sus códigos con el tipo de licencia Gpl reduce los costos de desarrollo, administración.

Tabla # 17 Precios Mysql

Producto	Por servidor / por año
Base de datos	
Mysql Migration Toolkit	0
Query Browser	0
Administrator	0
Mysql 5.0	0
Opciones	
Hot Backup	350
Soporte	**Por usuario**
Mysql Network Basic	595
Mysql Network Silver	1995
Mysql Network Gold	2995
Mysql Network Platinum	4995
Capacitación	
Mysql High Availability	1495
Mysql Performance Tunning	1995
Mysql Mew Features	1995
Mysql For DataBase Administrators	2495
Mysql For Developers	1495
Mysql In-Depth	995
Performance Tunnign And Optimization	995
Mysql Cluster	1500
Mysql For Beginners	2495
Developing Application with Mysql an Php	2495
Certificación	
Core Certification	200
Professional Exam	200
Core and Prof Exam	380

Fuente: shop.mysql.com Enero 2006
Autor: Diego Burbano

4.16 Algunas Pruebas de Rendimiento

Las pruebas realizadas sobre la base de datos Mysql (versión 5.0.18) se realizaron en una notebook pentium 4 de 2.4 Ghz, 1 GB Ram, el sistema operativo es Xp Professional.
Cabe señalar que para realizar un test más exhaustivo de los motores de base de datos existen empresas que dedican sus recursos y tiempo solo a este tipo de análisis.

Tabla # 18 Pruebas de rendimiento

Almacenamiento	# Registros Insertados	T1(s)	T2(s)	T3(s)	T Promedio(s)	T Promedio(min)
Innodb	100	0,000	0,010	0,010	**0,007**	0,000
Innodb	1.000	0,030	0,040	0,030	**0,033**	0,001
Innodb	10.000	0,360	0,380	0,350	**0,363**	0,006
Innodb	1.000.000	132,750	124,415	116,020	**124,395**	2,073
Innodb	3.000.000	227,510	228,980	355,740	**270,743**	4,512
MyIsam	100	0,100	0,040	0,010	**0,050**	0,001
MyIsam	1.000	0,050	0,070	0,050	**0,057**	0,001
MyIsam	10.000	0,500	0,430	0,450	**0,460**	0,008
MyIsam	1.000.000	46,130	45,520	45,230	**45,627**	0,760
MyIsam	3.000.000	144,390	142,910	143,710	**143,670**	2,395
Oracle	100	0,200	0,200	0,200	**0,200**	0,003
Oracle	1.000	0,291	0,311	0,301	**0,301**	0,005
Oracle	10.000	3,155	3,157	3,194	**3,169**	0,053
Oracle	1.000.000	364,244	333,250	340,340	**345,945**	5,766
Oracle	3.000.000	1046,115	1134,270	1098,345	**1092,910**	18,215

Autor: Diego Burbano

Para las pruebas se creó una base de datos llamada “Salud”, 3 tablas relacionadas entre sí. La versión de Mysql es el 5.0.18 para windows, la versión de Oracle fue la 10.2.0.1.0 también para windows.

Se generó procedimientos almacenados para cada motor de base de datos, el mismo que genera en forma secuencial el número de registros requeridos, para tomar los tiempos de respuesta en el caso de Oracle, como maneja el tipo de dato TimeStamp con soporte para milisegundos, se creó un campo fecha con tipo de dato Timestamp, en el caso de Mysql se utilizó el command line Client que con cada ejecución indica el tiempo que se demoró.

Según los resultados obtenidos en las pruebas, mysql es muchísimo más rápido que oracle cuando se utiliza el tipo de almacenamiento MyIsam, el cual, no maneja transacciones ni integridad referencial.

Al realizar las mismas pruebas con el tipo de almacenamiento Innodb (este tipo de almacenamiento maneja transaccciones), el tiempo de respuesta decayó en un 50% con respecto a Myisam. Con respecto a Oracle, sigue siendo más rápido (4 veces más rápido).

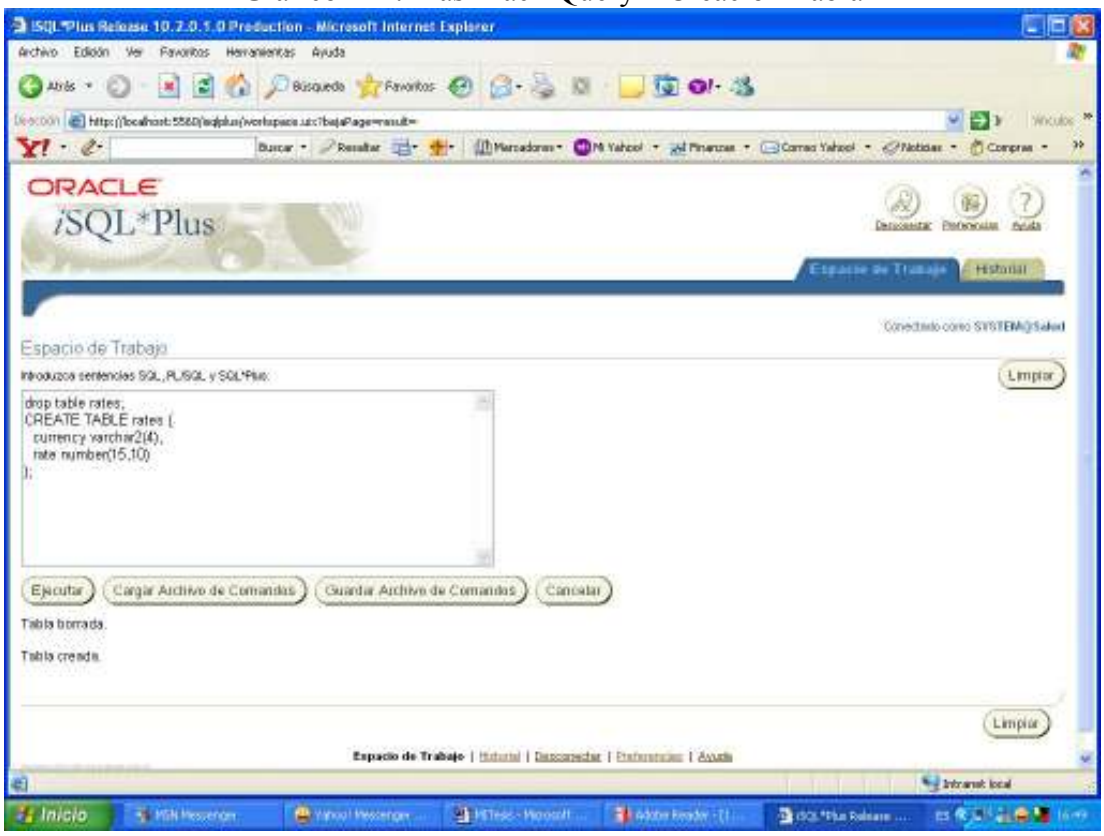
Innodb es el tipo de almacenamiento que maneja transaccionalidad, integridad referencial, es por esto que al comparar Myisam con Innodb, esta última es más lenta.

Si nos ponemos a analizar el porque de tanta diferencia entre Oracle y Mysql, esto se debe a que Oracle realiza muchos más controles, por ejemplo, Oracle 10g tiene

nuevas y novedosas características como Flashback Query, FlashBack Table, Flash Back TransactionUqery , lo que hacen es guardar en la misma base de datos una tabla histórica con la información de inserción, actualización o borrado que el usuario realizó sobre una determinada tabla, esta operación es costosa en tiempos de respuesta ya que por cada cambio en la tabla actualiza 2 veces, una en la tabla padre y otra en la tabla histórica de la siguiente manera:

- El primer paso es crear la tabla:

Gráfico #12: FlashBack Query – Creación Tabla



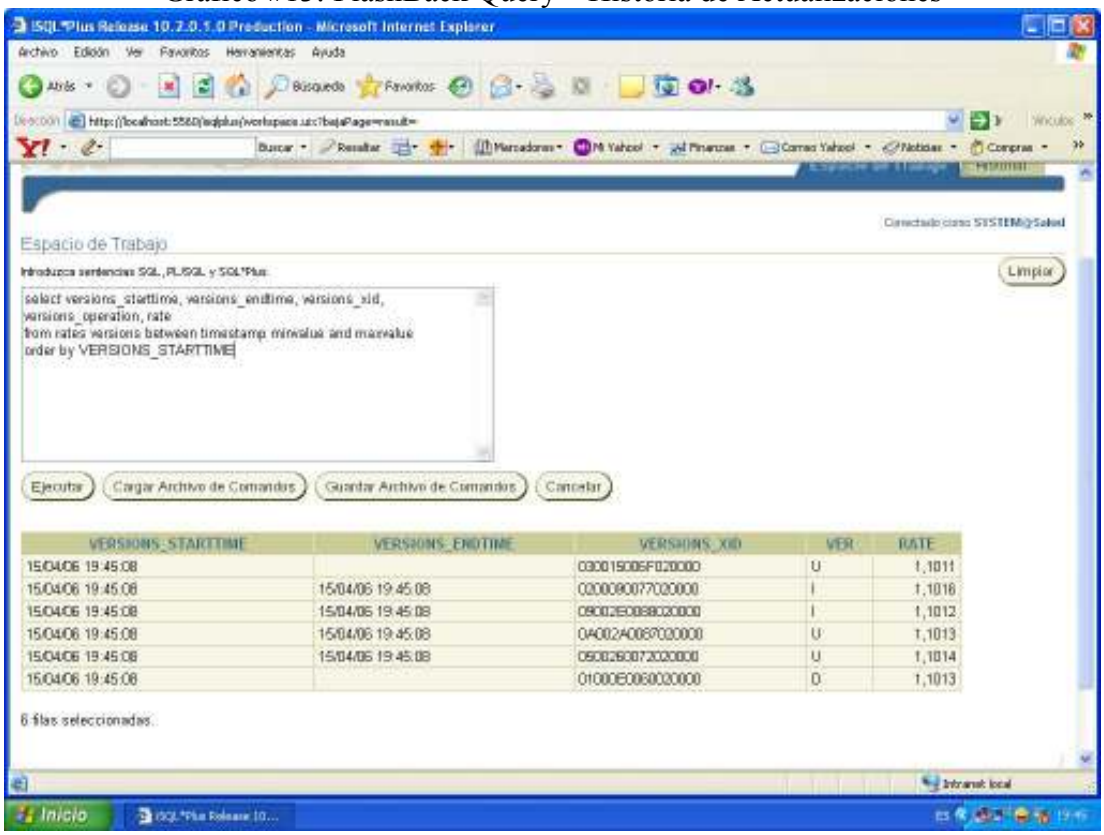
Autor: Diego Burbano

- Luego realzamos el siguiente proceso para actualizar, borrar e insertar datos sobre esta tabla:

```
insert into rates values ('EURO',1.1012);
commit;
update rates set rate = 1.1014;
commit;
update rates set rate = 1.1013;
commit;
delete rates;
commit;
insert into rates values ('EURO',1.1016);
commit;
update rates set rate = 1.1011;
commit;
```

- Los resultados obtenidos al consultar en la base de datos Oracle, son los siguientes:

Gráfico #13: FlashBack Query – Historia de Actualizaciones



Autor: Diego Burbano

Tendríamos que preguntarnos si esta característica es tan útil en un entorno de producción en el que lo primordial es la respuesta rápida a los usuarios, en el caso de un banco esto resulta muy bueno ya que permite recuperar los datos del valor de una divisa por ejemplo en un tiempo específico, esto ayuda a un correcto funcionamiento del negocio pero a la vez disminuye el performance de la base de datos. En todo caso para ambientes tan críticos como un banco, el poder recuperar los registros de una tabla (Flashback Table) que por accidente fue eliminada es de mucha ayuda ya que no se necesita parar la atención el público y el proceso es rápido.
El mismo análisis que hace Oracle (FlashBack Query) es programable en cualquier base de datos, se puede crear tablas de auditoria que permitan determinar que cambios se realizaron sobre un campo, pero esto va a degradar el tiempo de respuesta.

4.17 Pruebas de caídas

Las bases de datos están expuestas a problemas como fallas en comunicaciones, falla del ups. En estos casos, que no son muy repetitivos pero que suceden en ambientes reales es necesario proteger los datos.

Las bases de datos manejan este problema con las llamadas transacciones y las propiedades ACID.

Una transacción es un grupo de instrucciones sql que se manejan en forma atómica, como una sola unidad de trabajo, la atomicidad permite que las transacciones se ejecuten en su totalidad o no se ejecuta nada, esto permite que los datos se mantengan consistentes.

Tanto Mysql como Oracle manejan transacciones y ACID.
El procedimiento almacenado utilizado para las pruebas en Mysql es el siguiente:

```
DELIMITER $$
DROP PROCEDURE IF EXISTS `salud`.`spCreaCliente2` $$
CREATE PROCEDURE `spCreaCliente2`()
BEGIN
DECLARE CTR INTEGER;
DECLARE ch1 CHARACTER(30);
DECLARE ch2 CHARACTER(30);
Start transaction;
set CTR = 1;
set ch1 = 'djburbano@hotmail.com';
set ch2 = '2439481';
WHILE CTR<= 3000000 DO
 insert into salud.clientes values (ctr,'sierra','av granados',ch2,ch1,'T','0501890255');
  SET CTR = CTR + 1;
END WHILE;
END $$
DELIMITER ;
```

Al ejecutar el procedimiento almacenado spCreaCliente2, el momento en que termina el proceso se realiza un commit automático y los datos están consistentes y son permanentes, pero si en el transcurso de la ejecución se apaga el equipo, no se procesa ninguna inserción.

En oracle es de manera similar, la transacción empieza con el primer DML ejecutado y finaliza con alguno de los siguientes eventos:

- Si una sentencia Commit o Rollback es utilizado.
- Si una sentencia DDL o DCL que tiene commit automático es ejecutada.
- Si el usuario sale del iSql*Plus (rollback automático)
- Por una caída del sistema. (rollback automático)

El procedimiento almacenado utilizado para las pruebas en Oracle es el siguiente:

```
CREATE OR REPLACE PROCEDURE SP_CREACLIENTESORA2 is
BEGIN
FOR i IN 1..1000 LOOP
insert into clientes values (i,'sie','dici','243941','dja@hotmail.com','T','0501890255',
LOCALTIMESTAMP);
END LOOP;
ENd;
```

Al ejecutar el procedimiento almacenado SP_CREACLIENTESORA2, el momento en que termina el proceso se realiza un commit automático y los datos están consistentes y son permanentes, pero si en el transcurso de la ejecución se apaga el equipo, no se procesa ninguna inserción, se realizó un roolback.

4.18 Comparativa de Pesos desde el punto de vista del usuario

Tabla # 19: Comparativa de pesos - usuario

	Peso T	%	Peso Mysql	P1	Peso Oracle	P2
Costo y Licenciamiento	20,00	12%	20,00	12,00%	5,00	3,00%
Rendimiento	10,00	11%	10,00	11,00%	9,00	9,90%
Funcionalidad	150,00	10%	117,00	7,80%	150,00	10,00%
Características Fundamentales	110,00	10%	100,00	9,09%	110,00	10,00%
Acceso a Datos	160,00	10%	150,00	9,38%	160,00	10,00%
Seguridad	100,00	9%	80,00	7,20%	100,00	9,00%
Soporte	10,00	7%	10,00	7,00%	10,00	7,00%
Almacenamiento	50,00	7%	50,00	7,00%	30,00	4,20%
Tipos de Datos	140,00	7%	116,00	5,80%	126,00	6,30%
Alta Disponibilidad	30,00	7%	0,00	0,00%	30,00	7,00%
Sistemas Operativos	110,00	5%	110,00	5,00%	100,00	4,55%
Apis	120,00	5%	120,00	5,00%	80,00	3,33%
	1010,00	100%	883,00	86,27%	910,00	84,28%

Autor: Diego Burbano

4.19 Comparativa de pesos desde el punto de vista del desarrollador

Tabla # 20: Comparativa de pesos - desarrollador

	Peso T	%	Peso Mysql	P1	Peso Oracle	P2
Sistemas Operativos	110,00	12%	110,00	12,00%	100,00	10,91%
Apis	120,00	12%	120,00	12,00%	80,00	8,00%
Soporte	10,00	10%	10,00	10,00%	10,00	10,00%
Funcionalidad	150,00	10%	117,00	7,80%	150,00	10,00%
Rendimiento	10,00	10%	10,00	10,00%	9,00	9,00%
Características Fundamentales	110,00	8%	100,00	7,27%	110,00	8,00%
Tipos de Datos	140,00	8%	116,00	6,63%	126,00	7,20%
Acceso a Datos	160,00	8%	150,00	7,50%	160,00	8,00%
Seguridad	100,00	8%	80,00	6,40%	100,00	8,00%
Almacenamiento	50,00	7%	50,00	7,00%	30,00	4,20%
Alta Disponibilidad	30,00	7%	0,00	0,00%	30,00	7,00%
	990,00	100%	863,00	86,60%	905,00	90,31%

Autor: Diego Burbano

Anexo # 1: **Control de Accesos a Usuarios**

Mysql Provee altas características de seguridad, contiene un sistema de autentificación capaz de aceptar o denegar las conexiones, contiene también una herramienta para administración de los privilegios y métodos de encriptación.

Anexo # 2: **Backups**

Mysql puede ser respaldada con las utilidades en ambiente gráfico disponibles y además tiene la capacidad de sacar respaldos en caliente, esto es permitido gracias a online backup cuando la aplicación está en normal actividad.

Anexo # 3: **SSL:** Secure Sockets Layer

Es la misma tecnología que mantiene seguros los datos de los números de tarjetas de crédito cuando se realizan compras a través del internet.

Mysql lo soporta, el administrador de la base de datos cuenta con nuevos niveles de seguridad a nivel de base de datos, dependiendo de las necesidades en la seguridad de las aplicaciones en particular.
En los casos más básicos, quizá se desee encriptar solamente las sesiones, dejando al protocolo SSL que proteja as contraseñas de los usuarios. También se puede obligar a los usuarios que se conecten a través de SSL utilizando los argumentos opcionales que proporciona el comando Grant.

Anexo # 4: **Encriptación de Datos**

En las aplicaciones que almacenan datos muy delicados, como por ejemplo, registros de cuentas bancarias, quizá es necesario almacenar estos datos con un formato encriptado. Al hacer esto, resulta muy complicado para alguien tratar de utilizar los datos y se asegura la privacidad y no divulgación de los mismos.
En aplicaciones que son menos delicadas, se usa la encriptación para proteger algunas piezas de información, como por ejemplo la base de datos con las contraseñas para otra aplicación. En realidad, las contraseñas nunca deberían almacenarse en la base de datos sin encriptar.
A diferencia de la encriptación tradicional, que puede ser revertida, la serialización es un proceso de un solo sentido que no puede ser revertido. La única forma de averiguar la contraseña que generó una serialización en particular es utilizar un ataque de cálculo por fuerza bruta (probando todas las combinaciones para al entrada).
Mysql ofrece cuatro funciones para serializar contraseñas: Password(), Encrypt(), Sha1() y Md5(). La mejor forma de ver los resultados de cada función es probándolas utilizando el mismo texto original

Anexo # 5: **ACID**

Mysql soporta Atomicidad, consistencia, aislamiento y durabilidad, es un modelo transaccional que asegura la integridad y validez de los datos. Soporta también Commit y Roolback que asegura la consistencia de los datos.
Oracle si soporta transaccionalidad, utiliza Roolback, Commit, SavePoint.

Anexo # 6: **Control de Concurrencia**
Oracle maneja un control de concurrencia multiversión que difiere de los mecanismos de concurrencia utilizados por otros fabricantes de bases de datos.
Oracle soporta la consistencia de lectura en un nivel de instrucción y de transacción. Oracle determina el número de cambio del sistema (System Change Number, SCN) actual. El SCN esencialmente actúa como una marca de tiempo donde el tiempo se mide en términos de compromisos de la base de datos en lugar del tiempo de reloj.
Si en el curso de una consulta se encuentra que un bloque de datos tiene un SCN mayor que el que está siendo asociado con la consulta, es evidente que se ha modificado el bloque de datos después del SCN de la consulta original mediante alguna otra transacción y puede o no haberse comprometido. Por ello los datos en el bloque no se pueden incluir en una vista consistente de la base de datos como existía a la hora del SCN de la consulta. En su lugar, se debe utilizar una versión anterior de los datos en el bloque; específicamente aquel que tenga el SCN mayor que no exceda

al SCN de la consulta.
En el modelo de concurrencia de Oracle las operaciones de lectura no bloquean las operaciones de escritura y las operaciones de escritura no bloquean las operaciones de lectura, una propiedad que permite alto grado de concurrencia.

Anexo # 7: **Procesamiento Distribuido de Transacciones**

Muchas veces una simple transacción necesita escalar entre múltiples servidores de bases de datos. Esta situación es solventada por Mysql 5.0. Usando la ingeniería de almacenamiento disponible llamada Innodb, asegurando la integridad de la transacción. Mysql soporta Distributed Transaccions (XA), permite distribuir las transacciones a través de los diferentes servidores.

Anexo # 8 : **Unicode**

Unicote es un estándar industrial cuyo objetivo es proporcionar el medio por el que un texto en cualquier forma o idioma pueda ser codificado para el uso informático.
El establecimiento de Unicode ha involucrado un ambicioso proyecto para reemplazar los esquemas de codificación de caracteres existentes, muchos de los cuales están muy limitados en tamaño y son incompatibles con entornos multilingües. Unicode se ha vuelto el más extenso y completo esquema de codificación de caracteres, siendo el más dominante en la internacionalización y adaptación local del software informático. El estándar ha sido implementado en un número considerable de tecnologías recientes, que incluyen XML, Java y sistemas operativos modernos.

Anexo # 9: **Indices**

Mysql con el uso de las tablas innodb soporta el estándar b-tree indexing.
Oracle soporta el tipo de indexamiento b-tree indexing, esta característica mejora el rendimiento cuando accesa a una tabla,

Anexo # 10: **Diccionario de Datos**

Esta característica de la base de datos permite obtener fácilmente la información de los objetos que pertenecen a la base de datos como son las tablas, vistas, procedimientos almacenados, etc.
Mysql soporta esta característica desde la versión 5.

Anexo # 11: **Vistas**

Es una característica que permite visualizar los datos de una o varias tablas.
Mysql soporta esta característica desde la versión 5.
Oracle soporta esta característica que es un gran mecanismo para crear consultas con cierto formato o con ciertos campos de la tabla, esto ayuda a proteger la información que se presenta al usuario.

Anexo # 12: **Trigger**

Los disparadores ejecutan acciones antes o después de que se realice algún proceso de DML, un ejemplo de uso puede ser para sacar datos de auditoria, o para encriptar datos.
Mysql soporta esta característica desde la versión 5.

Anexo # 13: **Cursores**

Son utilizados dentro de los procedimientos almacenados para seleccionar o modificar datos con cierta lógica de negocio.
Mysql soporta esta característica desde la versión 5.

Anexo # 14: **Funciones**

Las funciones difieren de los procedimientos almacenados en que el propósito de la función es retornar un valor o información de una llamada a rutina o de una consulta, esto ayuda mucho a la reutilización de código.
Mysql soporta esta característica desde la versión 5.

Anexo # 15: **Procedimientos Almacenados**

Los procedimientos almacenados programados eficientemente ayudan a reducir el tráfico de red ya que utiliza una sola llamada a la base de datos.
Mysql soporta esta característica desde la versión 5 para satisfacer las necesidades de muchas empresas, una diferencia con los procedimientos almacenados de las bases de datos propietarias es que solamente necesita la cláusula SELECT para obtener una salida de datos, esto no ocurre con otras bases de datos que necesitan llamadas a paquetes o cursores con apuntadores a la cláusula SELECT.
Los procedimientos almacenados ayudan a eliminar tráfico excesivo en la red porque solamente una simple llamada al servidor es requerida en lugar de múltiples llamadas.
Oracle también soporta esta característica, además permite construir sentencias dinámicas.

Anexo # 16: **Recovery de Transacciones Erróneas**

Es una herramienta que sirve para retornar al estado anterior las transacciones que por error se ejecutaron sobre la base de datos.
Mysql no soporta esta característica.
En oracle, esta característica se la conoce como flashback query, que permite a un usuario establecer un cierto número SCN o tiempo de reloj en su sesión y ejecutar consultas sobre los datos que existían en esa fecha (supuesto que los datos todavía existían en el segmento de retroceso). Normalmente en un sistema de bases de datos,

una vez realizado el cambio no hay forma de retroceder al estado anterior de los datos a menos que se realicen restauraciones desde copias de seguridad, sin embargo, realizar esto es bases de datos muy grandes, resulta ser muy costoso.
Oracle dispone de una herramienta que realiza el trabajo en forma casi automática (necesita de un DBA para su correcta ejecución), Primero identifica la tabla afectada, la operación realizada Inserción, actualización o borrado, luego examina los cambios que realizó la transacción y si afectó a otras tablas, el dba examina el resultado del análisis de oracle, copia el la sentencia SQL y la ejecuta. Además puede recuperar la información con especifico time point.

Anexo # 17: **Replica**

Es el proceso de copia y mantenimiento de bases de datos múltiples que hacen un sistema distribuido. Consiste en dos servidores: uno Maestro y uno esclavo, El maestro registra todas las consultas que provocan cambios en la base de datos y las almacena en un fichero de registro binario. El servidor esclavo se conecta al maestro, lee las condiciones en el fichero binario y las ejecuta contra su copia local.
Mysql Maneja replica maestro-esclavo (en una sola vía).
Generalmente 3 productos o herramientas de Oracle cubren el área de replicación. Todos ellos están incluidos en el Oracle Enterprise Edition.

Anexo # 18: **Cluster**

Esta característica ayuda al balanceo de carga, la idea básica que está por detrás del balanceo de carga es bastante sencilla. Se dispone de un cluster de dos o más servidores y queremos compartir la carga de trabajo tanto como sea posible. Además de unos servidores finales, un balanceador de carga (e menudo un elemento especializado de hardware) se encarga de encaminar las conexiones entrantes a los servidores disponibles que estén menos ocupados.
Este grupo de servidores independientes o nodos, conectados vía una red privada que trabajan en grupo como un sistema unificado. Esta característica de la base de datos garantiza la disponibilidad de los datos en caso de falla de hardware por ejemplo. Mysql Cluster implementa recoverability automático del nodo para asegurar una aplicación automáticamente, es decir, si uno o más nodos de la base de datos deja de operar, pasa a otro nodo de la base de datos que contiene un set consistente de datos.
Mysql es usada para proveer el máximo tiempo en producción para sistemas de negocio críticos. En particular, Mysql Cluster es especialmente utilizado para proporcionar alta disponibilidad y respuesta rápida en búsquedas con sql. Mysql Cluster utiliza el Query Cache, el cual es una especie de memoria de cache que almacena consultas repetitivas, esto ayuda mucho a la respuesta rápida y al performance de Mysql Cluster cuando consultas idénticas son enviadas al servidor.
Oracle utiliza Real Apllication Clusters (RAC) como una opción de la base de datos, que soporta hardware clusters.

Anexo # 19: **Particionamiento de Tablas**

Se trata de una característica del las bases de datos actuales que permite dividir físicamente una tabla de la base de datos en particiones de acuerdo a la lógica del negocio que se necesite, de forma que, reduce significativamente los tiempos de respuesta de la aplicación.
En Oracle, la versión 10g soporta esta característica, permitiendo dividir los datos por fecha por ejemplo. La aplicación afecta directamente al diccionario de datos, creando la tabla interina de los datos e indicando qué datos se encuentran en una partición. Este mismo aplicativo permite realizar cambio en la base de datos en caliente sin reprogramar el aplicativo.
En Mysql, las tablas InnoDb manejan particiones, es decir, si las tablas son muy grandes, las descompone en pequeñas y más manejables piezas llamadas particiones.

Anexo # 20: **Automatic Storange Management**

Mysql automáticamente expande el espacio de almacenamiento de la base de datos.
Oracle también realiza este proceso automático desde la versión 10g, esto permite liberar al administrador de la base de datos de este tipo de análisis.

Anexo # 21: **Federated Tables**

En algunas ocasiones se necesitan referenciar a tablas que pertenecen a otras bases de datos remotas que se encuentran en otros servidores. Esta técnica hace que las distintas bases de datos físicas se vean como una base de datos lógica para la comunidad de usuarios, esto simplifica el acceso a datos.

Anexo # 22: **XML**

Extensible Markup language (XML) es un formato de archivo estándar usado para transferir datos entre sistemas, es similar al hypertext markup language (html) que almacena datos en un formato de jerarquía después de extraerlo de la base de datos, transformar y leerlo en otra base de datos. Un validador puede verificar que un documento xml pasa os requerimientos de estructura antes de procesar los datos. Xml es un componente clave en la computación basada en internet para transferir datos entre organizaciones.

Anexo # 23: **Características Orientadas a Objetos**

Esta característica en bases de datos tiene que ver con la posibilidad de acceder, utilizar, modificar los datos de una base de datos utilizando propiedades de la orientación a objetos: Hoy en día, ha aumentando uso de multimedios (la animación de vídeo, audio) en aplicaciones, especialmente en aplicaciones de la Internet. Los requisitos aumentados en la búsqueda, el índice por ejemplo, que optimiza eficazmente la búsqueda de datos, se pone en duda al utilizar tipos de datos más complejos.

Anexo # 24: **Precision Math**

En el caso de Mysql, soporta para los clientes que necesitan gran precisión en sus cálculos matemáticos. Dispone de 56 dígitos de precisión.
Anexo # 25: **Archive Engine**
Esta herramienta permite almacenar de forma eficiente los datos históricos.
Esta utilidad permite comprimir los datos en un 80% y reducir el tamaño de almacenamiento de los mismos.

Capítulo 5.- Conclusiones y Recomendaciones

5.1.1 Conclusiones

- El comercio electrónico ofrece opciones de automatización en todos los negocios de las empresas, los documentos digitales son el futuro, al ser este servicio tan imprescindible, se hace necesario analizar las posibilidades existentes en el mercado de bases de datos y las características que una base de datos libre puede solventar frente a las bases de datos comerciales.
- La inversión en una base de datos de código abierto es una alternativa válida que gerentes y desarrolladores deberían considerar el momento de escoger una solución informática.
- El desarrollo de MySql y la forma en la que los miles de usuarios aportan con la solución pronta a problemas existentes en el motor de base de datos, la hacen ver como una seria competidora ante las bases de datos propietarias, poco a poco sus funcionalidades envidian casi nada a estas últimas.
- Hay que tomar en cuenta que la compra del motor de base de datos no es el único gasto que las empresas tendrán que realizar para automatizar sus procesos, existen otros costos como administración, mantenimiento, capacitación, el costo es entonces un punto muy importante para tomar en cuenta a la hora de seleccionar un sistema gestor de bases de datos.
- Las bases de datos de código abierto están ganando terreno en el mercado rápidamente, esto ha preocupado mucho a las firmas propietarias de software como Oracle a tal punto que esta empresa ya lanzó su versión libre de la base de datos, además Oracle compró recientemente la empresa Innodb que es la empresa que maneja la ingeniería de almacenamiento de Mysql, esto indica que las empresas saben que Mysql es un buen competidor.
- Según las pruebas realizadas, el motor de base de datos Mysql es mucho más rápido que Oracle en el procesamiento de transacciones, esta es otra característica válida que en ambientes críticos de producción toma gran importancia el tiempo de respuesta de la base de datos en lo que satisfacción del cliente se refiere y en ambientes grandes de información como dataware house para la toma de decisiones.
- La base de datos Oracle es una herramienta muy confiable y segura, tiene opciones de auditoria, backup y aplicaciones para la toma de decisiones que

la diferencian de sus competidores libres y propietarios. En ocasiones es mejor sacrificar los recursos (memoria, disco) para obtener a cambio integridad en los datos.

- Si comparamos las herramientas de administración que posee Oracle frente a las de Mysql, nos damos cuenta que a Mysql le falta mucho por recorrer aún, pero con el transcurso del tiempo las distancias se van acortando.
- El tiempo de presencia en el mercado de cada una de las bases de datos es un factor determinante en su desarrollo y características de madurez, más de 10 años de ORACLE frente a 5 de MySql hacen que existan diferencia sin embargo el avance de MySql es notorio y robusto.
- Al tener el código fuente e instaladores de la base de datos disponibles, cualquier persona puede analizarlo y utilizarlo para desarrollar otro producto. La ventaja, sobre el modelo de software tradicional radica en que al tener disponible el código fuente e instaladores, la velocidad de respuesta ante errores y la aparición de los mismos es sumamente alta gracias al test que realizan miles de personas alrededor del mundo.
- El problema del open source ha sido siempre la falta de soporte técnico garantizado al cual acudir si los manuales y ayuda en línea no son suficientes, hoy en día, existen empresas como Mysql Ab que se ha preocupado por este problema y tiene varios servicios de soporte especializado. De hecho, esta es la fuente de sus ingresos, con el producto libre, lo que vende es la solución de problemas.
- En la página web http://www.tpc.org se puede observar varios benchmarcks de bases de datos, según este análisis realizado por la organización: Transaction Processing Performance Council, no se toma en cuenta a Mysql ya que la organización solamente avalúa los productos de miembros asociados y afiliados, este tipo de evaluaciones no son reales.
- Es complicado realizar un estudio de performance de las bases de datos ya que depende en gran medida de la experiencia de desarrolladores y del administrador de la base de datos, también del volumen de información, laboratorios especializados, simulación de casos reales.

5.1.2 Recomendaciones

- Oracle debe abrir el código fuente de sus aplicaciones de manera que permita a los usuarios un mejor manejo de la herramienta y conocer sobre su funcionamiento.
- Las pequeñas empresas deben tomar en cuenta las bases de datos de código abierto como Mysql para sus desarrollos ya que tiene muchas funcionalidades como las bases de datos comerciales y que está al alcance de su economía.
- En este estudio se detallan buenas razones para usar bases de datos de código

abierto en las empresas, en el caso de mysql, cumple con muchas de las características de las bases de datos propietarias, además el soporte es muy bueno y lo principal es que el costo se reduce en gran medida.

- Realizar las pruebas entre los motores de bases de datos en otros sistemas operativos y con servidores de características óptimas. Este estudio no las realiza por lo complicado que resulta en términos de infraestructura y costos.

Capítulo 6.- Bibliografía

- http://opensource.org/docs/definition.php
- http://mx.grulic.org
- http://wikipedia.org
- http://www.tpc.org
- http://www.mysql.com
- http://www.oracle.com
- Mysql Avanzado, Jeremy D. Zawodny y Derek J. Balling. Editorial O'reilly.
- Bases de Datos desde Chen hasta Codd con Oracle, Irene Ruiz, Miguel Angel Gómez, Enrique López Espinosa, Gonzalo Cerruela García, Editorial Alfaomega.
- Mysql para Windows y Linux, César Pérez, Editorial Alfaomega.
- La biblia de Mysql, Ian Gilfillan, Editorial Anaya.
- Fundamentos de Bases de Datos, Cuarta Edición, Abraham Silberschatz, Henry F. Korth, S. Sudarshan, Editorial McGraw Hill.

Glosario de Términos y Siglas

ACID, 15
Atomicidad, Consistencia, Aislamiento, Durabilidad.
ANSI, 40
Instituto Nacional Estadounidense de Estándares
Api, 150
Application programmign Interface - Interfaz de programación de aplicaciones.
Backup, 171
Copia de Seguridad

Tipo de almacenamiento utilizado por la base de datos Mysql para el manejo de transacciones y características Acid.
Java, 134
Lenguaje de programación.
JDBC, 47
Java DataBase Conectivity, un Api que permite la ejecución de operaciones sobre una base de datos desde el lenguaje de programación Java.
JVM, 143
Maquina virtual Java, permite la ejecución de programas sin depender de la plataforma.
LAN, 53
Red de área local, interconexión de varios ordenadores y periféricos.
LGWR, 120
Log Writer, proceso de la instancia Oracle.

Md5, 172
Algoritmo de resumen del mensaje 5, es un algoritmo de encriptación.
MysqlAB, 94
Empresa desarrolladora de la base de datos Mysql.
NFS, 14
Sistema de archivos de red (Network File System), posibilita que distintos sistemas conectados a una misma red accedan a ficheros remotos como si se tratara de locales.
ODBC, 47
Estándar de acceso a bases de datos (Open DataBase Conectivity), permite acceder a los datos sin importar que sistema gestor de base de datos almacene los datos.
ODMG, 40
Es el acrónimo de Object DataBase Management Group, Se usa tanto para definir el grupo de personas y empresas encargadas de desarrollar el modelo de objetos para persistencia, así como para la definición de dicho estándar.
Este modelo especifica los elementos que se definirán, y en qué manera se hará, para la consecución de persistencia en las BBDDOO que soporten el estándar. Consta de un lenguaje de definición de objetos, ODL, que especifica los elementos de este modelo.
OMG, 42
El **Object Management Group** u **OMG** (de sus siglas en inglés *Grupo de Gestión de Objetos*) es un consorcio dedicado al cuidado y el establecimiento de diversos estándares de tecnologías orientadas a objetos, tales como UML, XML, CORVA. Es una organización NO lucrativa que promueve el uso de tecnología orientada a objetos mediante guías y especificaciones para tecnologías orientadas a objetos
Php, 143
Es un lenguaje de programación usado generalmente para la creación de contenido para sitios web. PHP es el (acrónimo recursivo de "**P**HP: **H**ypertext **P**reprocessor", inicialmente PHP Tools, o, *P*ersonal *H*ome *P*age Tools) es un lenguaje interpretado usado para la creación de aplicaciones para servidores, o creación de contenido dinámico para sitios web, y últimamente también para la creación de otro tipo de programas incluyendo aplicaciones con interfaz gráfica.
PL/SQL, 116

Lenguaje de programación Simple de Oracle. Lenguaje procedural que amplía las capacidades del estándar SQL y poder definir secuencias de control, de flujo y toma de decisiones.

RAID, 14

RAID (Redundant Array Of Independent/Inexpensive Disks) es un término inglés que hace referencia a un conjunto de discos redundantes independientes/baratos. Este tipo de dispositivos se utilizan para aumentar la integridad de los datos en los discos

RDBMS, 15

Relational DataBase Management System, Sistema gestor de bases de datos relacionales.

Roolback, 65

Un instrucción sql que indica que la transacción debe reverarse.

Sha1, 172

La familia **SHA** (*Secure Hash Algorithm*, Algoritmo de Hash Seguro) es un sistema de funciones hash criptográficas relacionadas de la Agencia Nacional de Seguridad estadounidense y publicadas por el *National Institute of Standards and Technology* (NIST).

SMP, 56

Es el acrónimo de *Symmetric Multi-Processing*, multiproceso simétrico. Se trata de un tipo de arquitectura de ordenadores que se encuentra en las computadoras personales y servidores de baja gama.

SQL, 15

El Lenguaje de Consulta Estructurado (**S**tructured **Q**uery **L**anguage).

SSL, 171

Secure Sockets Layer (**SSL**) y **Transport Layer Security** (**TLS**), su sucesor, son protocolos criptográficos que proporcionan comunicaciones seguras en Internet. Existen pequeñas diferencias entre SSL 3.0 y TLS 1.0, pero el protocolo permanece sustancialmente igual. El término "SSL" según se usa aquí, se aplica a ambos protocolos a menos que el contexto indique lo contrario.

Throughput, 56

Se entiende como el número de tareas que pueden completarse en un intervalo de tiempo determinado

UNIX, 59

Es un sistema operativo portable, multitarea y multiusuario; desarrollado en principio por un grupo de empleados de los laboratorios Bell de AT&T

VLDB, 153

Very Large Data Bases, son bases de datos muy grandes como ambientes DataWareHouse.

XML, 134

Es la sigla del inglés e**X**tensible **M**arkup **L**anguage (lenguaje de marcado ampliable o extensible) desarrollado por el World Wide Web Consortium (W3C).

Trabajo realizado por :
Ing. Diego Javier Burbano
Quito-Ecuador Mayo 2006
dburbano@yahoo.com
Edad: 28 años
Titulo: Comparativa de base de datos