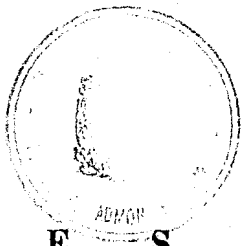
7
2g

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES

"A C A T L A N"

SERVIDORES DE BASES DE DATOS PARA LANs.



T E S I S

QUE PARA OBTENER EL TITULO DE:

LICENCIADO EN MATEMATICAS APLICADAS Y COMPUTACION

P R E S E N T A :

LUIS GUADARRAMA GONZALEZ

SANTA CRUZ ACATLAN, EDO. DE MEXICO 1991

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

INDICE.

4 CRITERIOS DE SELECCION.

INTRODUCCION

INTRODUCCION.

En la actualidad el procesamiento de información se ha convertido en una actividad muy importante, además su volumen día a día aumenta. En todas las etapas evolutivas de la computación, las organizaciones públicas, privadas y muchos ciudadanos se han visto beneficiados ascendentemente por la tecnología computarizada que almacena, maneja y procesa datos. Como es evidente el volumen de información crece muy rapidamente, por lo que es necesario que la tecnología evolucione para poder hacer frente a este vertiginoso avance en volumen de información. En los 60's la información era tratada por equipos grandes llamados Mainframes, posteriormente, aparecierón las Minicomputadoras, y en los 70's las Computadoras Personales (PCs). Los usuarios demandan cada vez aplicaciones más sofisticadas que traten a la información, ya que existe información que es recopilada y no es computarizada aún. El costo de equipo y programas decrece muy rápido, debido a la investigación en cuanto a Hardware y Software, asi el tratado de información es más satisfactorio ya que aumenta el número dispositivos que pueden almacenar más información, así mismo dispositivos que leen rápidamente la información y software que puede tratar imágenes, voz y datos conjuntamente. Puede entonces preverse que una vasta porción de los requerimientos actuales de datos son almacenados más económicamente en archivos de Sistemas Manejadores de Bases de Datos (DBMSs) que en papel.

Las dos áreas principales de la computación en el tratado de datos son : la tecnología de DBMSs y comunicaciones.

Esta Tesis está enfocada a ambientes de PCs, ya que en la actualidad es la tecnología que ha tenido mayor impacto en el mundo de la computación . Así mismo presenta una evaluación comparativa de lo último en tecnología de sistemas manejadores de bases de datos para PCs, junto con los fundamentos de esta nueva tecnología.

Como las PCs han tenido un impacto asombroso, han dado como consecuencia una gran demanda , por parte de las organizaciones, en cuanto a aplicaciones que compartan recursos hardware y software, en cuanto a hardware, que éste sea fácil de utilizar y en cuanto a software que pueda utilizarse concurrentemente, todo esto con vías a satisfacer las necesidades de información de los usuarios, que cada vez crecen más. Es por esto que en la evolución de las PCs se han hecho varios intentos para resolver el problema. Tales intentos son: el DISK-SERVER, el FILE SERVER y actualmente la arquitectura cliente-servidor (DATA BASE SERVER).

Esta tesis presenta conceptos y recomendaciones, así como una comparación de servidores de bases de datos. Cabe mencionar que no se trata de decir cual es el mejor, ya que ninguno es malo, pero tampoco bueno, es bueno en la medida que cumpla o se apegue a las necesidades de la organización que se trate.

Actualmente existe una gran confusión en las personas que toman decisiones en cuanto a equipo a utilizar y software que operará en él.

Los vendedores de software para el tratado de información (DBMS) son a veces exagerados o dicen que su producto es el mejor, por lo cual no son honestos, además las comparaciones se realizan de diferente manera para cada producto, por lo cual no son válidas.

Y dado que esta tecnología (cliente-servidor) tendrá un gran impacto en el mundo de las redes de área local (LANs), y muchas empresas están cambiando a este esquema de trabajo. De ahí mi interés por seleccionar este tema de tesis.

La tesis está comprendida en cuatro capítulos :

CAPITULO 1.
Generalidades de LANs y DBMSs.

Presenta conceptos fundamentales de redes de área local, así mismo las topologías básicas existentes , y métodos de compartir los medios de transmisión de datos.

En el apartado de DBMSs se describen los conceptos fundamentales de sistemas manejadores de bases de datos (DBMSs). En esta sección se hace más énfasis en el modelo relacional, así como a sus lenguajes de manipulación de datos, ya que es el modelo que utiliza la tecnología nueva de DBMS para redes de área local (arquitectura cliente-servidor).

CAPITULO 2.
Tecnología de DBMSs.

Se describe la forma en que los DBMSs manejan la información para el modelo relacional para que sea más rápida la respuesta a la consulta realizada por algún usuario y además sea confiable la información. En este capítulo también se describen los conceptos de la tecnología que actualmente se están difundiendo para ambientes LANs (tecnología cliente-servidor y bases de datos distribuídas).

CAPITULO 3.
Evaluación comparativa de los principales servidores de bases de datos con arquitectura cliente-servidor existentes en el mercado .

Se trata de hacer como el nombre del capítulo lo dice, una evaluación comparativa de los principales DBMSs con arquitectura cliente-servidor que pensamos que tendrán más impacto en México.

CAPITULO 4.

Criterios de selección.

Se analizan en conjunto todos los DBMS evaluados, posteriormente se trata de dar una serie de puntos que en algún momento pueden servir como guia para escoger un RDBMS con arquitectura cliente-servidor que se requiera en alguna organización, asi mismo se da un ejemplo de una dependencia pública. Finalmente se hace una pequeña crítica en cuanto a documentación existente para el tema de la arquitectura cliente-servidor.

El objetivo general de esta tesis es hacer una evaluación comparativa de los principales sistemas manejadores de bases de datos existentes en el mercado de México para redes de area local con arquitectura cliente-servidor; Asi mismo describir en que consiste esta nueva arquitectura y las ventajas que representa.

1. PANORAMA GENERAL DE LANs Y DBMSs

1.1 GENERALIDADES DE LANs.

1.1.1 Antecedentes históricos de LANs.

El análisis y almacenamiento de la información ha sido uno de los grandes problemas que ha enfrentado el hombre desde que inventó la escritura, uno de los primeros intentos para resolver el problema de análisis de información fue el de Blaise Pascal, filósofo y científico francés en 1642 cuando invento una máquina que a base de engranes permitía sumar y restar.

1694

El matemático alemán Gottifried W. Leibnitz diseñó una máquina que además de sumar y restar podía, multiplicar, dividir y extraer raíces cuadradas.

Liebniz fué el primero en introducir la notación binaria, con la cual se puede representar cualquier cifra con solo dos digitos (0 y 1 ; prendido y apagado).

1823

El matemático inglés Babbage diseño la máquina diferencial, su complejidad mecánica no permitia que este proyecto se terminara; sin embargo, en teoría podía resolver polinomios de hasta 8 términos.

Babbage después del primer fracaso, diseñó la máquina análitica que tampoco pudo concluir, debido a que funcionamiento requería de tecnología no disponible en ese tiempo. Además de estar diseñada para resolver cualquier operación matemática , se considera como la primera máquina de calcular programable, aún cuando sus programas eran externos.

La información de programas y datos se almacenaban mediante grandes tarjetas perforadas. Su mecanismo estaba constituído por engranes que giraban al ser activados por alambres que pasaban de un lado a otro de las tarjetas, al encontrar las perforaciones.

1889

Herman Hollerith patenta en Estados Unidos la primera tabuladora electrónica ; pudiéndose considerar como el inicio de las máquinas modernas para el tratamiento de datos.

El censo de la población de Estados Unidos se levantó en 1890 con la tabuladora de Hollerith, dos veces más rápida que si se hubiera usado otro sistema contemporáneo.

Los datos de cada familia se habían perforado previamente en una tarjeta, la cual era "leída" por agujas metálicas que al hacer contacto a través de las perforaciones, mandaban un impulso electrónico a los contadores localizados en el panel superior de la tabuladora.

1944

Jhon Von Newman propone la idea del programa interno, ademas es el quien desarrolla la teoria para la construcción del computador electrónico.

Este año se desarrolló el computador "electro-mecánico" denominado ASCC (Automatic Sequence Controlled Calculator) que pesaba 50 toneladas y su capacidad de cálculo era similar a la de una Calculadora portátil actual.

1951

En Nueva York, IBM muestra el computador SSEC (Selective Sequence Electronic Calculator). El principal trabajo que desarrolló ese año, fue recalcular la órbita de los planetas .

Remington Rand introduce el computador Univac. que es el primero que puede manejar cifras y letras.

1954

IBM inicia con éxito los experimentos para remplazar los bulbos por transistores.

En estos momentos parte del problema de análisis y almacenamiento de la información se habia resuelto.

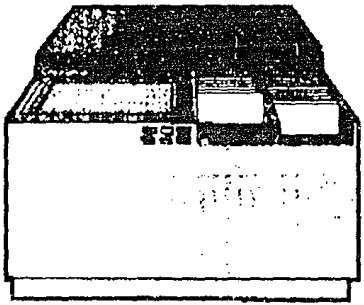
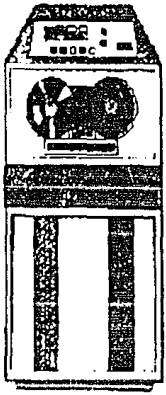


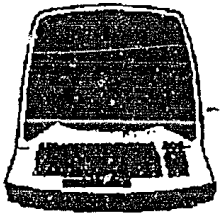
fig.1 lectora de tarjetas perforadas.

En la decada de los 50 s la información se almacenaba en tarjetas perforadas (figura 1), con este medio de almacenamiento ya era posible enviar información al centro de procesamiento de datos, el problema que surgió fué que las grandes cantidades de tarjetas perforadas se tenían que acarrear hasta el centro de procesamiento de datos.

La solución a este problema fué la aparición de las teminales tontas (figura 2) en la década de los 60 s, las cuales permitían la comunicación con la unidad central de procesamiento de datos, logrando con esto una comunicación más rápida y eficiente, pero con este tipo de computadoras (MAINFRAMEs) se encontró que a medida que se conectaban más terminales tontas al computador central la velocidad de respuesta era cada vez menor.



(a) Mainframe.



(b) terminal conectada al Mainframe

fig.2

hacia la mitad de la década de los setentas surge la primera computadora personal (figura 3). gracias, a la confiabilidad de operación ,reducción de costos, capacidad de memoria, y al gran desarrollo de software para PCs tales como procesadores de texto, hojas electrónicas manejadores de bases de datos y graficadores, pronto inundaron el mercado.

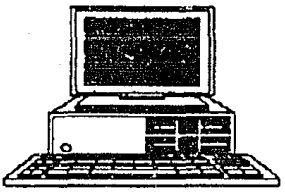


fig.3 computadora personal

A principios de los ochentas las microcomputadoras habían revolucionado por completo el concepto de computación electrónica así como también sus aplicaciones .

Las personas encargadas de los departamentos de cómputo fueron perdiendo el control de la información ya que ahora el proceso de la información no estaba centralizado; El nuevo problema que surge en las microcomputadoras es que ahora la información se tenía que acarrear en discos flexibles de una PC a otra , además la relativa poca capacidad de los diskettes hace mas difícil el manejo de grandes cantidades de información.

Con la llegada de la tecnología Winchester[1] se lograron dispositivos que podían almacenar grandes cantidades de información , una de las desventajas de esta tecnología era su alto costo.

En estos momentos fue cuando surge la idea de que varios usuarios compartieran el mismo disco duro Winchester, con este concepto nace la primera red local basada en un DISK SERVER. Estos equipos permitían a cualquier usuario accesar cualquier parte del disco, esto trajo obviamente problemas en la integridad y seguridad de los datos , para aliviar el problema se introduce el concepto de FILE SERVER en el que todos los usuarios pueden tener acceso a la misma información , compartiendo archivos pero con niveles de seguridad, lo cual permite que no sea violada la integridad y seguridad de la información, pero en cuanto a tratamiento de datos no es eficiente. Para solucionar el problema, surge la tecnología cliente -servidor (que se explica en el capítulo 2).

[1]La empresa IBM fue la primera que empezó a integrar estos dispositivos a sus computadoras personales. El proyecto original consistía de dos unidades de 30 Mbytes , por lo que al producto se le llamo 30-30, de donde se derivo la denominación Winchester, marca de un rifle de ese calibre.

1.1.2 *Que es una LAN*

"Una red de área local (LAN) es un sistema de comunicación de datos que permite que un número de dispositivos de tratamiento de información independientes como son PCs, minis o mainframes e impresores se comuniquen entre ellos en una área geográfica limitada que va de aproximadamente 3 metros A 10Km " [2].

El objetivo primordial de una red local es:
-Compartir recursos materiales (equipos y sus periféricos) y recursos informáticos (archivos de datos y programas), actualizándolos , organizándolos y explotándolos. Con las siguientes características:

-Debe asegurar la compatibilidad de productos fabricados por diferentes empresas.
-Debe estar estructurada en niveles de tal manera que un cambio en un nivel debe de afectar sólo a ese nivel (se explicará este inciso en el modelo OSI).
-Debe permitir la comunicación de estaciones de trabajo de bajo costo y ser ella misma un elemento de bajo costo.
- Posibilidad de comunicación directa entre dos nodos de la red local sin necesidad de almacenamiento y reenvío a través de un tercer nodo de la red.
- Las redes de area local deben permitir la supresión o adición de estaciones de trabajo de manera fácil.
- Cuando las estaciones de trabajo compartan recursos físicos tales como ancho de banda del medio físico la red local debe disponer de mecanismos que garanticen la justa compartición de recursos (explicado en el apartado de protocolos).

Las redes locales se clasifican en tres tipos :

- Sistemas de baja prestación y bajo costo. Este tipo de red local normalmente utiliza cable trenzado como medio de transmisión .
- Sistemas de prestacion media y costo medio , normalmente utilizan cable coaxial como medio de transmisión con codificación de señales en banda base.
-Sistemas de alta prestación y alto costo. Utilizan normalmente cable coaxial blindado.

Otro medio que está adquiriendo perspectivas muy interesantes son las redes de fibra óptica en las cuales se puede transmitir a alta velocidad y larga distancia.

[2] Teleinformática y Redes de Cómputo , Antonio Alabad Muños.

1.1.3 Partes de una LAN

Las LAN conectan computadores y otros dispositivos, tales como impresoras, terminales, modems, graficadores, utilizando algún tipo de sistema de cableado, interfaces de red (tarjetas) y apropiada comunicacion de la red (protocolo) y software de operación.

los principales componentes de la red son:

1.- El FILE SERVER que puede ser dedicado o no dedicado , cuando el server es dedicado, exclusivamente administra los recursos de la red. Cuando el server no es dedicado, además de administrar los recursos de la red, funciona como estación de trabajo.

En la PC definida como el server se carga el sistema operativo de red, además se conectan los dispositivos que van a compartir las estaciones de trabajo tales como impresoras , modems, graficadores, pero actualmente se está perdiendo este esquema con las nuevas versiones de sistemas operativos para LANs.

2.- ESTACIONES DE TRABAJO que están representadas por cada una de las microcomputadoras conectadas a la red.

3.- TARJETA INTERFACE que va instalada dentro de cada microcomputadora como es ETHERNET.

4.-CANAL DE COMUNICACION es un cable dedicado a la comunicación, mismo que puede ser:

a) De par trenzado.
b) De coaxial.
c) De fibra óptica.

5.- REPETIDORES sirven para reforzar la señal ya que algunos nodos de la red a veces se encuentran muy separados.

6.- SISTEMA DE CABLEADO cuya forma de conexión en los equipos (topologia), está en función de la tarjeta que se haya seleccionado.

7.- SISTEMAS OPERATIVOS los cuales pueden ser:

a) NETWARE DE NOVELL.
b) IBM PC NET.
c) NETWARE -OS.
d) URGERMANN-BASS.
e) TAPESTRY.
f) LAN MANAGER.

1.1.4 El modelo OSI

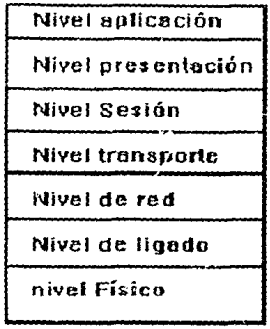
Muchos de los fabricantes de microcomputadoras en un tiempo comenzaron a fabricar productos que interconectaran

sus propios equipos , así cualquier otro equipo que no fuera de la familia del fabricante no podía comunicarse con él. Por lo que surge el modelo OSI (Open System Interconection) de ISO (International Standar Organitation).

El modelo OSI está dividido en siete niveles , entre sus características más importantes esta que un nivel inferior sirve a su superior , es decir el nivel K sirve al nivel K+1, pero el K+1 no puede servir al K por lo que si el nivel cuatro solicita algo de el nivel dos, el nivel cuatro lo tiene que pedir al nivel tres y el tres al dos.

Los niveles en que está estructurado el modelo OSI son:

-Nivel de aplicación.
-Nivel presentación.
-Nivel sesión.
-Nivel transporte.
-Nivel red.
-Nivel enlace.
-Nivel físico.



Nivel de aplicacion (nivel 7).

Es el nivel superior del modelo OSI, en él se resuelven todas las funciones de comunicación.

Entre las responsabilidades que tiene este nivel son:

- La inicialización de la transferencia de datos.
-El acceso general a la red.
-El control de flujo.
-La recuperación de errores.

Entre el software que opera en el nivel siete está:

-captación de emulación de una terminal.
-transferencia de programas.

Nivel de presentación (nivel 6).

Uno de los objetivos de este nivel es proveer servicios para el nivel superior.

Los servicios provistos por este nivel están orientados a la interpretación de la información. Como por ejemplo:
- Tiene un protocolo de conversion de datos.
- Una captación de datos.
-- Encriptación, cambios al conjunto de caracteres.

En otras palabras el nivel de presentación esta dedicado solo a la sintaxis (la forma de representar los datos y no a la semántica -el significado para este nivel). La semántica es conocida sólo por las entidades de aplicación.

Este nivel es provisto para la representación de la información.

La función formal del nivel de presentación incluye , establecer requerimientos de sesión , transferencia de datos, y transformación de datos.

Así pues en el nivel de presentación se concentran todas aquellas funciones que sean necesarias para la existencia de la heterogeneidad en la forma de intercambiar información.

Nivel de sesión (nivel 5).

De particular importancia para las redes de área local es el nivel 5, ya que una de las razones para implementar una LAN es la conectividad (que es la habilidad para que uno o más dispositivos se puedan conectar con otro).

El objetivo de los elementos de este nivel es proporcionar un soporte a la comunicación entre los entes del nivel presentación .

Cuando una liga entre dos dispositivos es hecha, una sesión es establecida , y se intercambian los datos con delimitadores de control. La sesión regula el diálogo entre ellos y deja de existir cuando finaliza la tranferencia de datos e información de control.

Nivel transporte (nivel 4).

El propósito del nivel de transporte es proveer un nivel adicional de conexión, más bajo que el nivel de sesión, el servicio del nivel de transporte provee una transferencia transparente de datos entre las entidades de sesión.

En el nivel de transporte se controla la transferencia de datos , así como el manejo de errores , y problemas involucrados con la transmisión y recepcion de paquetes (datos originados por un usuario e información que la red necesita para transportar los datos del usuario de un nodo a otro).

Nivel de red (nivel 3).

El objetivo de este nivel es proporcionar los elementos necesarios para intercambiar información entre los entes del nivel transporte a través de la red de transmisión de datos.

El nivel de red provee los mecanismos para establecer, mantener y terminar la conexión entre sistemas abiertos. Por

ejemplo cuando una LAN se quiere conectar con un Mainframe se conectan a través de este nivel .

El servicio más importante ofrecido por el nivel de red, es el proveer la transparencia de transferencia de datos entre entidades del nivel transporte .

El modelo de referencia ISO/OSI para la interconección de sistemas abiertos se ha tenido que modificar en el nivel de enlace para las LANs ya que en las grandes redes para transmitir información se utiliza un nodo intermediario para almacenar la información y despúes enviarla, en las redes de área local no sucede así. Los nodos se comunican extremo a extremo a nivel de enlace (nivel 2).

El nivel de enlace en las redes locales se ha subdividido en dos niveles ,LLC (control de acceso lógico) y MAC (control de acceso al medio).

Los objetivos que persigue la IEEE es que el nivel LLC sea independiente de la topologia usuada en la red local, del medio y método para accesarlo.

Nivel de enlace (nivel 2).

El nivel de enlace define los accesos estratégicamente para compartir el medio físico (cable).

En las LANs el medio se utiliza por varias comunicaciones de manera simultánea , por lo que se tiene que tener una estrategia de acceso al medio.

-Control de acceso al medio (MAC).

Este subnivel es el responsable de ejercer el control de acceso al medio, por lo que el subnivel LLC tiene un medio de comunicación aparentemente propio .

-Enlace lógico (LLC).

La función primordial de este subnivel es el de direccionamiento y envio sin error del mensaje, para ello hace lo siguiente:

-Un campo de direccionado para determinar el destino o destinos del mensaje ya que puede ser direccionado individual o por grupo.

--Un campo de control (para indicar el tipo de mensaje).

--Un campo de redundancia cíclica (para la detección de errores de transmisión).

Nivel físico (nivel 1).

Es el nivel que define las caracteristicas eléctricas y mecánicas de la red. Técnicas de modulación , frecuencia a la que la red opera y los voltajes empleados, ademas establecer la interface con el nivel de enlace.

1.2 TOPOLOGIAS DE LANs.

Las redes de area local se caracterizan por tres aspectos, el medio de transmisión, la topologia y el método de acceso.

1.2.1 Medio de transmisión.

Muchos medios son usados para proveer el servicio a la LAN en su nivel físico. los más comunes son:

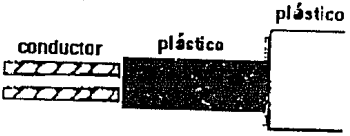
-Par trenzado.
-Cable coaxial (banda base y banda ancha).
-Fibra óptica.

Par trenzado(figura 4 a).
Por muchos años la comunicacion ha utilizado par trenzado . Sus principales caracteristicas son:
-Un canal puede transportar de 12 a 24 canales de 300-3khz.
-Son válidos en topologías, anillo, bus, estrella.
-Bajo costo.
-Alta taza de error a grandes velocidades.
-Alta interferencia.
-Alcance de hasta 3Kms
-Se utilizan repetidores para ampliar las distancias.

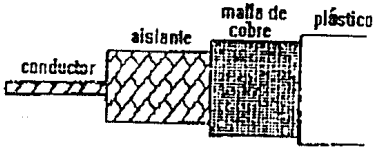
Cable coaxial de banda base(figura 4 b).
- No hay modulación de frecuencia.
-Uso de enchufes especiales para la conexión física.
-Se usa una tarjeta de interconexión de red.
-Se usa generalmente en topologías bus.
-Alcance de 1 a 10 Kms.
-Ancho de banda de 10Mbps.
-Bajo costo (simple de instalar).
-Poca inmunidad al ruido.

Cable coaxial de banda ancha.
-Se combina voz, datos y video simultáneamente.
-Los datos son modulados antes de la transmisión.
-Instalación mas difícil en comparación con el de banda base.
-Se utilizan amplificadores (para largas distancias).
-ALcance hasta 5kms.
-Topología soportada :bus.
-Ancho de banda máximo 400KHZ.
-Mayor inmunidad al ruido.
-Costo alto.

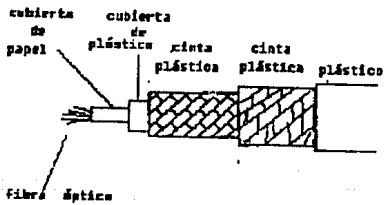
Fibra óptica(figura 4 c).
Esta es una alternativa para no usar par trenzado y cable coaxial en sus dos formas, además es el medio físico de transmisión para el futuro.
características:
-La señal no se modula.
-No es afectada por la interferencia eléctrica, ruido, temporales, radiación o agentes químicos.
-Ancho de banda de 50Mbps.
-Se puede transmitir datos, voz y video.
-Muy poca pérdida de señal.
-Topologías , anillo, estrella.
-Muy cara actualmente.



(a) par trenzado.



(b) cable coaxial.



(c) fibra óptica

fig 4.

1.2.2 Topologías de redes.

En las redes locales existen tres tipos básicos de topologías a saber.

-Estrella.
-Bus.
-Anillo.

El termino topología se refiere a la manera de agrupar las estaciones de trabajo y los cables de comunicación que componen la red.

1.2.2.1 Topología estrella

La topología estrella se caracteriza por un FILE SERVER centralizado con una conexión directa para cada estación de trabajo. Las estaciones de esta topología son bidireccionales y éstas se manejan por el FILE SERVER.

Las redes estrella fueron las primeras redes en desarrollarse debido a su estructura sencilla. Las desventajas son:

-en caso de fallar el FILE SERVER, todo el sistema deja de funcionar.
-La red puede crecer solo hasta alcanzar la capacidad del FILE SERVER.
-Resulta costosa esta configuración por la cantidad de cable a utilizar.

El protocolo que utiliza esta topología es el token.

Las ventajas son: si un nodo deja de funcionar la red sigue funcionando, asimismo la flexiblilidad es buena permitiendo adicionar o suprimir con sencillez estaciones de trabajo (figura 5).

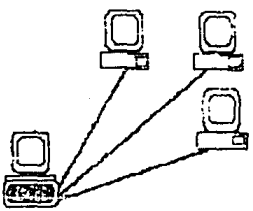


fig. 5 Topología Estrella.

1.2.2.2 Topología bus

Esta topología se considera la más sencilla de todas, donde las estaciones incluyendo al FILE SERVER están "colgadas" a un solo cable (figura 6), como por ejemplo un cable coaxial. un nodo (estación de trabajo) no depende de la siguiente para que el flujo de información continúe como sucede en la topología anillo. La red tipo bus permite que los mensajes sean transmitidos en todo el canal. Cuando un nodo reconoce que el mensaje es para él lo saca del canal . A consecuencia de que en el canal sólo puede viajar un solo mensaje el bus requiere que cada nodo pueda transmitir, y recibir datos y resolver colisiones. La topología bus utiliza el protocolo de contención CSMA/CD (Carrier Sense Multiple Access/Colision Detection). El protocolo de acceso multiple por sensibilidad de portadora / Detección de Colisiones requiere de un dispositivo para "escuchar " antes de transmitir el mensaje. El dispositivo puede enviar el mensaje sólo cuando esté libre de señal. En caso de que dos nodos comiencen a enviar el mensaje simultáneamente, se detectará la colisión y se detendrá la transmisión.

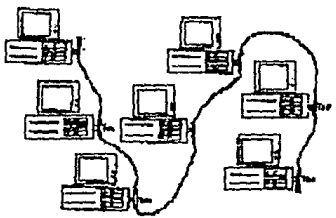


fig. 6 Topología Bus.

1.2.2.3 Topología anillo

En esta topología la información viaja en un solo sentido a través de un solo cable coaxial u otro medio. La información pasa de un nodo a otro a través de repetidores.

En la actualidad no existen verdaderas topologías de anillo en el mercado, ya que tiene una desventaja

fundamental. Si un nodo (estación de trabajo) falla, toda la red deja de funcionar.

Otro problema propio de la topología anillo es que a medida que se transmiten los mensajes puede disminuir la velocidad de la red. Por ejemplo si los datos van a la derecha y la terminal receptora se encuentra a la izquierda de la terminal emisora, el mensaje debe de viajar por toda la red antes de llegar al receptor.

La ventaja de la red anillo es que se requiere de un mínimo de inteligencia, siendo de un costo menor en comparación con las otras.

Debido a que en la topología de anillo si una estación falla se bloquea toda la red. Existe una topología llamada anillo modificado, la cual consta de una caja a la que se conectan las estaciones de trabajo , de esta forma si una estación de trabajo queda fuera de servicio, la red no se interrumpe (figura 7).

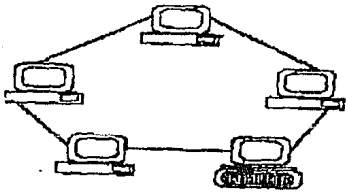


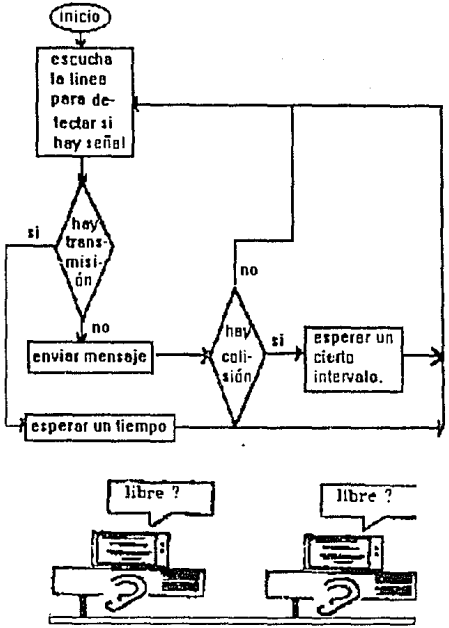
fig. 7 Topología anillo.

1.2.3 Métodos de acceso al media

1.2.3.1 CSMA/CD.

El original CSMA/CD fue desarrollado por Xerox como parte de la red local Ethernet (figura 8).

Este método de acceso al medio está asociado con la topología bus. Su forma de trabajo se puede resumir en el diagrama de flujo siguiente:



Protocolo CSMA/CD.

fig.8

1.2.3.2 Token bus.

Este metodo de acceso al medio es apropiado para la topología estrella. Aquí la estación que tiene el token es la dueña del canal y lo toma hasta que termina de transmitir su mensaje (figura 9).

Para entender este tipo de acceso al medio supongamos que tenemos una manecilla la cual apunta a una estación y pregunta si tiene mensaje. Si es así, esta estación toma el token y envia su mensaje el cual debe estar direccionado a otro nodo, si no, deja el token y lo toma otra.

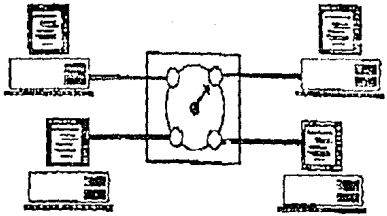


fig.9

1.2.3 Token ring

Este método fué originalmente propuesto en 1969 y fué conocido como ring Newhall.

La información es transferida secuencialmente, bit a bit , de un nodo activo al siguiente. Cada estación sirve como medio para atar uno c mas medios (estaciones de trabajo) para el anillo. Cada estación regenera y repite cada bit. La estación que tiene acceso al medio (tiene en su poder al token) transfiere la información dentro del anillo. El emisor de la información finalmente remueve la información del anillo. Cuando la transferencia de información finaliza , la estación receptora genera un nuevo token y lo transmite a la siguiente estación (figura 10).

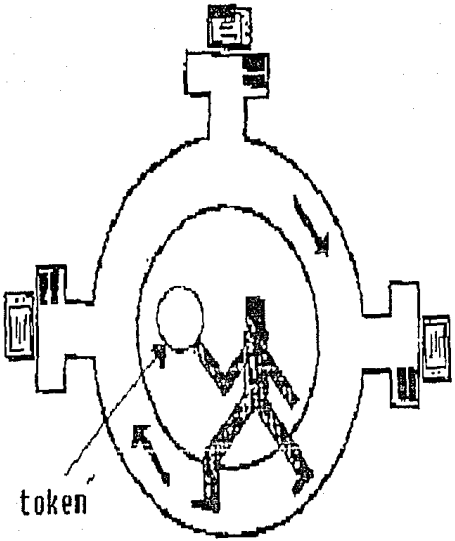


fig. 10

Aspectos importantes sobre las LANs.

Algunas de las preguntas que más se cuestionan los usuarios son:

-¿ hasta cuántos usuarios soporta la red?
-¿Cuántas estaciones se pueden conectar ?
-¿ Es posible agregar más usuarios a la red que se tenga instalada sin que se degrade el tiempo de respuesta?

Todas la preguntas planteadas arriba se refieren a la eficiencia de la red ("performance").

La eficiencia no depende de un solo factor , sino que es función de varios, tales como tipo de cable que se utiliza, tipo de tarjeta de interface a la red, forma de compartir el medio ,etc.

Toda combinación de servidor-red tiene un máximo ancho de banda esto es, existe un límite máximo de datos que pueden moverse a través de la red. Una vez que sea alcanzado este límite, la red ha llegado a su máximo de eficiencia.

Entonces el factor que más afecta el "performance " sobre una red con topología bus y protocolo de acceso al medio CSMA/CD, es el porcentaje de colisiones que ocurran. Mientras el número de colisiones aumente, la efectividad de la red decrece.

Pequeños paquetes usualmente implican mas transmision y por lo tanto incrementa la posibilidad de colisiones.

Con respecto al esquema de token conforme el número de estaciones crece el "performance" decrece.

Además otro factor que se tiene que considerar es el tipo de carga para la red, es decir, el tipo de usuarios que tendrá acceso a la red.

1.3 GENERALIDADES DE LOS DBMSs.

1.3.1 ¿Qué es un Sistema Manejador de Bases de Datos?

Un sistema manejador de bases de datos (Data Base Manager System), consiste de un conjunto de programas que permite al usuario interactuar con la base de datos sin tener conocimiento de como estan almacenados los datos.

El objetivo general de un DBMS es el poder actualizar (altas, bajas y cambios) y recuperar la información eficientemente así como mantener la información consistente, es decir que la información sea fiable.

definición.- una base de datos (DB) es una conjunto de datos organizados que modelan la actividad de la empresa.

1.3.2 Abstracción de la información.

Uno de los principales propósitos de un DBMS es proveer a los usuarios una vision abstracta de los datos[1], esto es, el sistema resguarda ciertos detalles de la forma en que se almacenan y mantienen los datos, ya que existen muchos usuarios que no cuentan con conocimientos computacionales, tal es el caso del gerente de la compañía que desea utilizar el DBMS para la toma de sus decisiones. Para obtener la abstracción en la base de datos es necesario dividirla en tres niveles a) nivel físico, b)conceptual, c)de visión. (Figura. 1.)

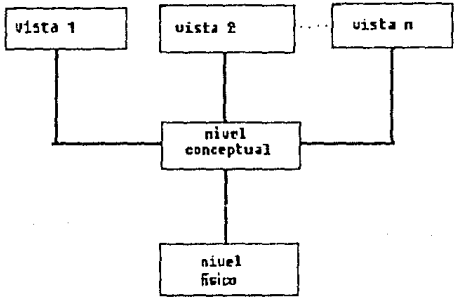


fig.1 Nivel de abstracción de la información (cuando los datos se encuentran dentro de la base de datos del DBMS).

[1]valores registrados fisicamente como son, datos de productos, cuentas bancarias, nombres de proveedores.

-Nivel fisico.

A este nivel concierne la manera de almacenamiento de los datos en el dispositivo.

-Nivel conceptual.

Este es el nivel intermedio, en el se definen cuales son los datos reales (entendibles a las personas) almacenados, y cuales son sus relaciones. Este nivel se utiliza solo por el administrador de la base de datos (DBA), ya que él decide qué información tiene relación con otra.

-Nivel de visión.

Este es el nivel más alto de la abstraccion de los datos ya que no todos los usuarios están interesados por la totalidad de la base de datos, por lo qué se definen vistas (por el DBA). A este nivel le atañe el cómo visualizará los datos cada usuario.

Para ilustrar las diferencias entre los tres niveles se hace una analogia con el lenguaje de programacion "c".

Supongamos que declaramos un registro proveedor que contiene los campos, numero de proveedor, ciudad, direccion, y nombre del representante, el cual se define como sigue.

```
struct{

        char num-prov       [10];
        char ciudad         [30];
        char direccion      [30];
        char nom_repres     [30];

    }proveedor;
```

Los resultados del ejemplo se muestran en la figura 2.

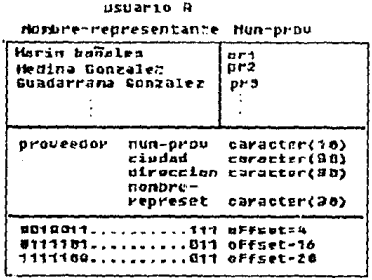


Fig. 2(a)

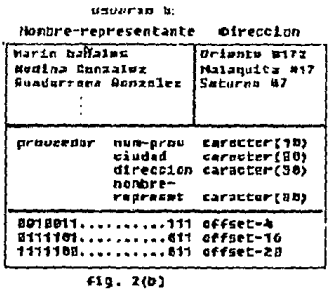


Fig. 2(b)

fig.2

En la figura.2(a) al usuario A sólo le interesa saber el nombre del proveedor y su número de identificación, mientras que al usuario B le interesa el nombre y la dirección del proveedor, pero ambos tienen en común dos niveles (nivel físico y nivel conceptual), se puede decir que el nivel de visión es el que cambia constantemente.

1.3.3 Objetivos de un sistema manejador de bases de datos.

Los objetivos principales de la tecnología de los DBMS son:

- Independencia de los datos.
- Habilidad de compartir datos.
- Habilidad para relacionar.
- Integridad.
- Flexibilidad de acceso.
- Seguridad.
- Eficiencia.
- Control y administración.

La habilidad para modificar la definición de un nivel del esquema de abstracción sin afectar al nivel superior, se le conoce como independencia de datos.

En la independencia de datos existen dos niveles de independencia, independencia física e independencia lógica.

- Independencia física.- Es la propiedad que permite aislar las aplicaciones de los cambios en la organización de los datos empleados en estas aplicaciones, por ejemplo cambio de dispositivo de almacenamiento.

- Independencia lógica.- Se refiere a la capacidad de aislar las aplicaciones de los cambios en el nivel conceptual.

La independencia lógica es más difícil de lograr. Para entender esto explicaremos el fenómeno opuesto poniendo como ejemplo a dBASE III.

Los programas hechos en DBASE III son dependientes de los datos, esto significa que la manera como se accesan los datos depende de los requerimientos de la aplicación (por qué atributo se indexará). Por ejemplo la aplicación debe saber qué índice existe, de modo que la estructura interna de la aplicación (programa) se construirá con base a este conocimiento.

Se dice que una aplicación como esta es dependiente de los datos por que es imposible cambiar la estructura de acceso (quitar el índice e indexar por otro atributo) sin afectar la aplicación.

Por lo que es necesario que un DBMS proporcione la independencia de datos.

- Habilidad de compartir datos.

El objetivo es permitir a las aplicaciones compartir una base de datos, eliminando así la redundancia. El DBMS debe permitir visualizar los mismos datos a varios usuarios concurrentemente, así como el control de acceso (este párrafo se explica más ampliamente en el capítulo dos).

En el párrafo anterior se dijo que se elimina la redundancia, ya que a veces es necesaria para efectos de rendimiento en términos de tiempo de acceso.

- Habilidad para relacionar.

La habilidad para relacionar se refiere precisamente a la habilidad para definir relaciones entre registros a nivel lógico de manera conveniente. Las relaciones son muy importantes en el diseño de las bases de datos.

- Integridad.

Un DBMS debe proporcionar mecanismos que garanticen que los datos de la base de datos sean confiables. Por ejemplo tomemos las tablas de la figura 9, la tabla proveedor registra todos los posibles proveedores de alguna parte, la tabla partes registra todas las partes que son útiles a la empresa, y la tabla embarques registra que proveedor surte determinada parte; por lo que el DBMS se debe de encargar que no se registren en la tabla embarques números de proveedores inexistentes, es decir que no puede entrar en esa tabla un proveedor que no se encuentre en el dominio de la tabla proveedor, así mismo con las partes.

Entre los mecanismos que debe proveer el DBMS para la integridad es que el DBA pueda definir procedimientos de validación que habrán de ejecutarse cada vez que se intente una operación de actualización (altas, bajas, cambios). Explicado con más detalle en el capítulo dos.

- Flexibilidad de acceso.

La flexibilidad de acceso se refiere a navegar por toda la base de datos, en base a cualquier llave de acceso y calificación de acceso, mediante un lenguaje de consultas no procedural.

- Seguridad.

Deben de existir mecanismos que asignen y revoquen derechos a usuarios para el acceso a los datos (tratado más ampliamente en el capítulo dos).

- Eficiencia.

Debido a los grandes volúmenes de datos y los requerimientos de acceso a la base de datos, la eficiencia es un factor muy importante.

A medida que la base de datos crece la actualización y consulta tardan más (tratado en el capítulo dos).

- Control y administración.

En los DBMS debe existir un administrador de bases de datos, el cual debe tener todos los derechos que otorga el DBMS.

En el DBA recaen las responsabilidades del diseño de la base de datos, dar derechos de uso a los usuarios de la base de datos, revocar derechos, entre otros.

Un DBMS como se mencionó anteriormente está constituido por software que permite manipular los datos, este software es el siguiente:

- Un lenguaje de definicion de datos (Data Definition Languaje -DDL), que permite especificar el esquema de la base de datos, el resultado de la compilación de las proposiciones en el DDL son datos sobre la estructura de los archivos los cuales se almacenan en el diccionario de datos por lo que el contenido del diccionario de datos son datos de los datos.

-Un lenguaje de manipulacion de datos (Data Management Languaje -DML), que tiene como tarea actualizar y recuperar datos.

Existen dos tipos de DML los cuales son procedurales y no procedurales.
En el DML procedural el usuario especifica cuales datos quiere y cómo debe obtenerlos, en el DML no procedural el usuario indica qué datos desea sin decir como obtenerlos.

Otra característica de los no procedurales es que no contienen ciclos iterativos como son for, while, etc.

1.3.4 Clasificación de los usuarios.

Existen tres tipos de usuarios en una base de datos y se distinguen por la manera que piensan comunicarse con el DBMS.

- Familiarizados con conceptos de programación.

Se caracteriza al usuario que no es temeroso al computador y ha adquirido lógica o habilidad para resolver problemas mediante algoritmos. Además estos usuarios hacen llamadas al DML por medio de un lenguaje de programación tal como "c", pascal, cobol, basic, y las APIs (Aplication Programing Interface), por ejemplo realizar un programa que imprima un reporte que no sea muy común.

Las APIs son transformadas por medio de un precompilador a instrucciones reconocibles al compilador del lenguaje de programacion anfitrión.

- Uso frecuente de lenguajes de consulta.

Estos usuarios interactúan con el sistema (DBMS) escribiendo sus consultas en el procesador de consultas , las cuales son tomadas y transformadas para que las pueda entender el manejador de la base de datos.

- Usuarios ingenuos.

Son usuarios que interactúan con el sistema invocando algunos programas permanentes, desarrollados por los que tienen conocimientos de programación. Por ejemplo una secretaria que desea imprimir un reporte especifico.

1.3.5 Modelos de bases de datos.

El ámbito de las bases de datos está dividido en tres etapas, éstas tres etapas son, nivel conceptual (como los entiende el humano en la realidad), nivel lógico (para su definicion, procesamiento y almacenamiento) y la tercera es el nivel fisico de los datos.figura 3.

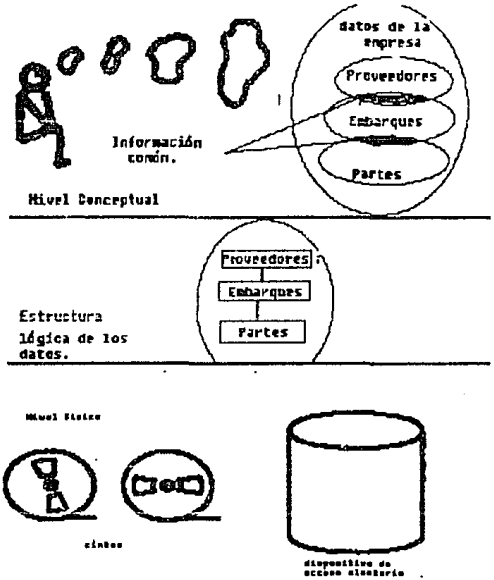


fig.3.

En el nivel dos se tiene el nivel uno, pero con la diferencia que aqui están organizados los datos de acuerdo con las reglas de un modelo particular de datos.

Existen tres tipos principales de modelos de datos para modelar el nivel uno, estos modelos son: el modelo jerárquico, de red, y relacional (los cuales se explican a continuación).

1.3.5.1 Modelo jerárquico

El modelo jerárquico no es más que una colección de arboles. Un árbol es una gráfica en la que el número de nodos es igual al número de arcos más uno, es decir el número de arcos que llegan al nodo son uno, cada padre puede tener uno o más hijos, el nodo hijo no puede tener más de un padre, el nodo superior se llama "raiz" y los nodos que no tienen sucesor se les llama hojas (figura 4).

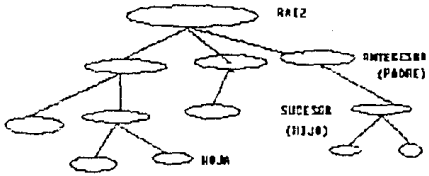


Fig. 4

En las bases de datos jerárquicas se considera a un nodo como un registro y al arco como una liga que une a dos registros.

Las relaciones de mapeo pueden ser solo 1 a 1 o 1 a M es, decir uno a uno y uno a muchos, pero no puede ser N a M (muchos a muchos). En otras palabras un hijo sólo puede tener un padre, un padre puede tener varios hijos (1 a M), asimismo, un padre puede tener un solo hijo (1 a 1).

En este modelo el contenido de un registro puede repetirse en varios lugares, por ejemplo, en el sistema de proveedores y partes, una parte puede ser surtida por varios proveedores, la información correspondiente al proveedor se tendrá que repetir para cada pieza que surta (figura.5).

Esto trae una gran desventaja, al actualizar la información puede suceder alguna inconsistencia de la información y por supuesto un desperdicio de espacio en el almacenador.

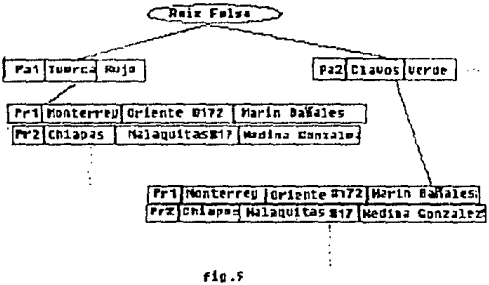


fig.5

1.3.6.2 Modelo de red.

Los diagramas para representar los modelos de red están constituidos por cuadros que representan a los registros y lineas que corresponden a las ligas, por lo que se puede decir que el modelo jerárquico y el de red se representan igual. La diferencia está en que la estructura de red es más general que la jerárquica por que una ocurrencia de registro especifico puede tener cualquier número de antecesores y también cualquier antecesor puede tener varios sucesores de esta manera se tendra un mapeo N a M , M a 1 , 1 a M , M a N.

Tomando el ejemplo de los proveedores y partes un diagrama de red sería como se encuentra en la fig.6.

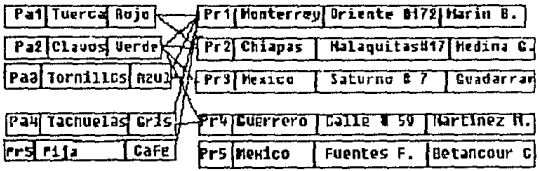


fig.6.

Antecedentes Historicos del modelo relacional.

La historia de las bases de datos de tipo relacional comenzo en la década de los 70's cuando el matemático E.F. Codd publicó en junio de 1970 un escrito llamado " a relational model of data for large shared data bank" . Codd estableció 12 conceptos en los que se basa el modelo relacional (las cuales listamos posteriormente).

La investigación realizada por Codd influyó al desarrollo de un lenguaje prototipo que concretizaba las características del modelo relacional. Uno de estos

lenguajes se le llamo Structure English Query Languaje (SEQUEL), el cual fue definido por D.D. Chamberlim y el departamento de investigación de IBM en 1974. En 1976 se define el SEQUEL/2 y posteriormente el SQL (que trataremos con detalle más adelante).

En 1982 IBM anuncia el primer manejador relacional comercial llamado SQL/DS; Para el siguiente año, introduce DB2 el cual cubre el ámbito tipo mainframe, y es aun, la mejor base de datos para sistemas grandes de IBM.

Las doce reglas definidas por Codd para el modelo relacional, se utilizan como parametros, para verificar qué tanto se apegan las bases de datos relacionales a sus conceptos originales.

1.3.3 Modelo Relacional

La mayoría de bases de datos para microcomputadoras se basan en el concepto del modelo relacional. La gran popularidad que ha adquirido este modelo en todos los ambientes (mainframes, minis y microcomputadores) se debe a su gran facilidad de manejo para sumar nuevas relaciones entre tablas, la habilidad para cambiar la estructura de los registros y la flexibilidad de su lenguaje.

Definición.- Una relación es un subconjunto del producto cartesiano de una lista de dominios.

Un dominio es simplemente un conjunto de valores. Por ejemplo; el conjunto de enteros es un dominio, el conjunto de caracteres (A..Z) es un dominio, etc.

Sean los dominios A1,A2,A3,..An se dice que $A_1XA_2X..A_n$. forman el producto cartesiano y es el conjunto de todas las tuplas $(a_1,a_2,a_3,..a_n)$ tal que a_1 está en A_1, a_2 esta A_2 ... a_n está en A_n.

Un modelo relacional consta de tablas bidimensionales, las cuales contienen columnas y renglones. Como se muestra en la figura 7(b).

El concepto de tabla tiene una similitud con el concepto de relación, ya que dados los dominios, en la tabla (atributos), su producto cartesiano forma una relación por lo que los renglones son llamados tuplas.

Para ser más explicitos supongamos que tenemos un dominio de piezas y un dominio de colores, el producto cartesiano de estos dominios forman una tabla como se muestra en la figura 7.

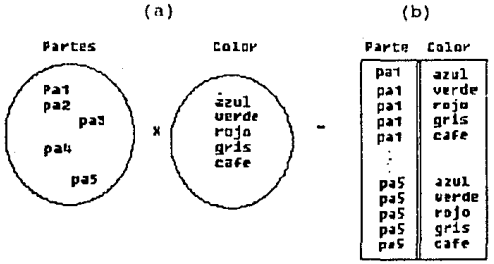


Fig . 7

La normalización auxilia a modelar la información decidiendo cúando particionar los datos dentro de tablas relacionales. La meta es asignar los datos eficientemente. La figura 8(a) muestra tablas sin normalizar o sea en su primera forma formal (First Formal Form), la cual contiene redundancia de datos y la figura 8(b) muestra la misma información, pero sin redundancia, particionada en dos tablas (segunda forma formal).

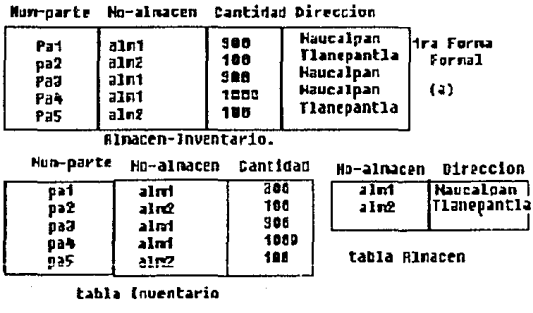


Fig 8.

Num-prov	Ciudad	Direccion	Nombre-repres
Pr1	Monterrey	Oriente 8172	Marin Bañales
Pr2	Chiapas	Malaquitas#17	Medina Gonzalez
Pr3	Mexico	Saturno # 7	Guadarrama G.
Pr4	Guerrero	Calle 8 SD	Martinez M.
Pr5	Mexico	Fuentes F.	Montanchuer C.
Pr6	Toluca	Casa Aleman	Majera Amador
Pr7	Oaxaca	Nueva Vida	Gonzalez Campos

Tabla Proveedor. (a)

Num-parte	Nombre	Color
Pa1	Tuerca	Roja
pa2	Ciruca	Verde
Pa3	Tornillos	Azul
Pa4	Tachuelas	Gris
Pa5	Pija	Cafe

Tabla Partes (b).

Num-Prov	Num-parte
pr1	pa1
pr1	pa2
pr1	pa3
pr1	pa4
pr1	pa5
pr2	pa1
pr2	pa2
Pr3	pa2
Pr4	pa2
pr4	pa3
pr4	pa3
Pr7	pa4

Tabla Embarques (c)

fig 9 .

Como un ejemplo adicional, piénsese en las relaciones proveedor, partes y embarques.

su representación notacional esta dada por
proveedor(num-prov,ciudad,dirección, nombre-representante).
partes(numero-parte, nombre, color).
embarque(num-prov, número-parte).

Su significado es el siguiente , la relación proveedor contiene información referente al proveedor.

La relación partes contiene información de las características de las piezas.

La relación embarque contiene información sobre las piezas que surte cada proveedor (figura 9).

Supogamos que deseamos saber el nombre y la dirección de los proveedores que surten la parte uno. Primero, se examina la tabla embarques para encontrar los números de los proveedores que surten la parte uno (quedando un resultado como se muestra en la figura 10(a)). Después se examinará la relación proveedor para averiguar el nombre y la dirección

de los seleccionados de la tabla de embarques, dando un resultado como se muestra en la figura.10(b).

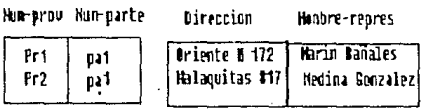


Fig. 10.

1.3.5.3.1 Reglas de Codd para el modelo relacional.

1.- REGLA DE INFORMACIÓN.- Todos los datos deben ser puestos en tablas y deben definirse en el nivel conceptual, además toda la información de control de datos (diccionario de datos) debe presentarse también por medio de tablas.

2.- REGLA DE ACCESO.- Esta regla explica que se debe de poder recuperar la información que se desee si se conoce:
- El nombre de la tabla donde se encuentra dicha información.
- El valor de la clave por la cual se recuperará.
- El nombre de la columna donde se encuentra esta clave.
por ejemplo.
SELECT nombre_repres FROM proveedor WHERE num_prov = 'pr1'.

3.- TRATAMIENTO DEL CARACTER NULO.- la distinción del caracter nulo con otros caracteres como string con blancos o la distinción del cero con otros números.

4.- CATALOGO EN LINEA .- El catálogo que controla la base de datos debe ser gobernado por las mismas reglas de control de los datos comunes, por lo tanto, para insertar a un nuevo usuario se puede hacer mediante los mismos procedimientos que se realizan con la inserción de datos comunes.

5.- REGLA DE SUBLENGUAJES.- Un RDBMS soporta al menos un lenguaje el cual debe de tener una sintaxis bien definida, este lenguaje debe soportar lo siguiente:

- Definición de datos (creando las relaciones de la base de datos).
- Definición de vistas (vistas externas).
-Manipulación de datos (iterativo y por programa).
- Establecer integridad en los cambios.
- Establecer seguridad.
-Manejo de transacciones (BEGIN, COMMIT, ROLLBACK).

6.- REGLA DE ACTUALIZACION DE VISTAS.- En esta regla se establece que un RDBMS debe tener la habilidad de poder actualizar las tablas a través de las vistas.

7.- ALTO NIVEL DE INSERCION, MODIFICACION Y BORRADO. - El RDBMS debe tener la habilidad de operar con muchos registros simultaneamente. Con un solo comando el RDBMS debe ser capaz de ejecutar una operación (insert, update, delete o select) sobre cualquier número de renglones en cualquier tabla.

8.- INDEPENDENCIA FISICA DE LOS DATOS.- Los cambios físicos de los datos del RDBMS no debe afectar a las aplicaciones existentes. Los cambios pueden ser:
- Adición o supresión de índices.
- Mover tablas a diferentes drives.
- Cambiar el orden de los renglones en la tabla.

9.- INDEPENCIA LOGICA DE LOS DATOS.- Las aplicaciones no deben ser afectadas cuando se haga algún cambio lógico, pero el nivel conceptual esta muy cerca de la aplicación, por lo que la aplicación es muy sensible en cualquier cambio o alteración. Por ejemplo en una tabla donde se le suma una columna .No pasará nada a una aplicación que haga referencia a esta tabla, pero si se suprime una columna a la cual se hace referencia en alguna aplicación está si se vera afectada por la eliminación de la columna.

10.- REGLA DE INTEGRIDAD .- La integridad debe ser definida por el lenguaje de consultas y almacenada en el catálogo (no en la aplicación).

11.- INDEPENDENCIA DE DISTRIBUCION.- Esta regla es una extensión de la regla ocho. Un RDBMS maneja el concepto de bases de datos distribuidas en la medida que las aplicaciones no sufran modificaciones por la distribucion de los datos.

12.- Esta regla se refiere a que el RDBMS deben tener la habilidad de poder comunicarse con el usuario via algun lenguaje tal como COBOL, FORTRAN, C etc.

1.3.5.3.2 Lenguajes de consulta.

Un lenguaje de consultas (Query Languaje) se define como un lenguaje de computación de alto nivel para la recuperación y modificación de los datos de la base de datos. Es usualmente en linea , y capaz de soportar preguntas no predefinidas.

Los lenguajes de consulta pueden clasificarse en procedurales y no procedurales. En los lenguajes de procedimientos el usuarios dice qué datos desea obtener y como obtenerlos; En contraste con los no procedurales el

usuario dice que es lo que necesita, pero no dice como hacerlo.

La mayor parte de los DBMS permiten los dos tipos de lenguajes.

Dentro de los lenguajes de consulta no procedurales para las DBMS relacionales se encuentran: el álgebra relacional y el SQL.

El álgebra relacional está compuesta por 5 operadores básicos que son: Unión, Diferencia, Producto cartesiano, Proyección y Selección (figura 5).

NOMBRE	SIMBOLO.
Unión	U
Diferencia	-
Producto cartesiano	X
Selección	&
Proyecta	¶

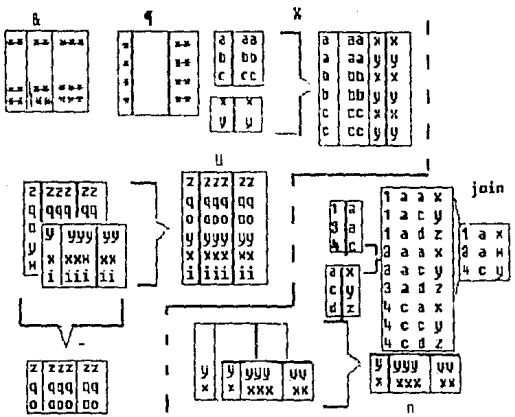
Los tres primeros operadores (U, -, X) se denominan bidireccionales ya que operan bajo dos relaciones o más . Los dos operadores se llaman unarios (&,¶), estos operan sólo sobre una relación.

Las cinco operaciones fundamentales se explican a continuación:

Selección(&).-Selecciona tuplas de una relación que cumple cierto predicado.
Proyección(¶).-Selecciona ciertas columnas solamente (las que se le indique).
Producto cartesiano(X).-Combina información de dos o varias relaciones.
Unión(U) .-Lee renglones que se encuentre en una u otra relación.
Diferencia(-).-Lee los renglones que se encuentran en una pero no en otra relación.

Existen otros operadores que son adicionales estos permiten simplificar las expresiones del álgebra relacional básica. Tal es el caso del join y la intersección.
Intersección(∩).- Permite encontrar las tuplas que se encuentran repetidas en ambas relaciones.

Join .- Lee los posibles pares de renglones que satisfacen cierta condición.



Después de que Codd publicó su documento sobre los conceptos del modelo relacional, IBM lanzo un proyecto para implementar esta teoría en un producto comercial pero para esto necesitaba un lenguaje de consultas no procedural que ejecutara todas las operaciones del álgebra relacional (& , ¶, U, -, X), asi nace el SQL.

El SQL (Structure Query Languaje) consta de tres niveles.

1) Definición de datos (DDL- Data Definition Languaje).
2) Manipulación de datos (DML - Data Manipulation Languaje).
3) Control de datos (DCL- Data Control Languaje).

1)Definición de datos.

Son Proposiciones que definen la estructura de base de datos. Estas proposiciones permiten que se definan tablas o vistas en la base de datos. Tales proposiciones son : CREATE TABLE , CREATE INDEX, DROP Y ALTER TABLE.
La sintaxis de los comandos arriba mencionados es:

CREATE TABLE nombre de tabla (nombre atributo atributo tipo(longitud),....atributo tipo(longitud)).

CREATE nombre-indice INDEX ON nombre de tabla (atributo).

CREATE nombre-indice UNIQUE INDEX ON nombre de tabla(atributo).

donde : los paréntesis son parte de la sintaxis y las mayusculas son comandos SQL; Por ejemplo CREATE TABLE proveedor(prov_parte decimal (10)).

La combinación del unique Index y not null a un campo especifico forman la llave primaria de la tabla.

ALTER nombre de tabla ADD atributo tipo (longitud).

CREATE VIEW nombre de la vista AS (SELECT atributos FROM tablas WHERE predicado.

Esto permite que el usuario vea los datos que le atañen.

DROP TABLE nombre de la tabla.

El diccionario de datos es creado automáticamente por el DDL.

2)Manipulación de Datos (DML - Data Management Languaje).

Después de que una tabla se ha definido en la base de datos, los datos pueden ser cargados dentro de la tabla. Se utilizan las proposiciones de manipulación de datos para colocar datos dentro del RDBMS asi mismo estas proposiciones permiten cambiar, borrar y recuperar los datos de la base de datos . UPDATE, INSERT, DELETE, Y SELECT son proposiciones de manipulación de datos.

La principal proposición del DML es el SELECT que se usa para todas las operaciones de recuperación. El SELECT tiene la forma general:

SELECT [DISTICT]·[ALL] atributo(s) FROM tabla(s)[WHERE predicado][GROUP BY atributo][HAVING expresión][ORDER BY atributo].

donde:

[ ] son parte opcional de la sintaxis.

( ) son obligados en la sintaxis.

Un ejemplo para esta proposición y retomando la pregunta de seleccionar el nombre y dirección de los proveedores que surten la parte uno , en SQL seria como se muestra a continuación.

SELECT NOMBRE,DIRECCION FROM PROVEEDOR WHERE NUM_PROVE=(SELECT NUM_PROVE FROM EMBARQUE WHERE NUM_PARTE='pal').

Para modificar uno o varios registros de alguna tabla existente en la base de datos se utiliza la propocisión update y su sintaxis es la siguiente:

UPDATE nombre de tabla SET atributo= valor WHERE....

Para borrar algun renglón o renglones que cumple con algún predicado se utiliza la proposicion delete y su sintaxis es:

DELETE FROM nombre de tabla WHERE predicado.

Para insertar renglones dentro de una tabla se utiliza la proposición insert :

INSERT INTO nombre de la tabla VALUES(valor(s)).

El DML tambien incluye las proposiciones COMMIT Y ROLLBACK para el manejo de transacciones. Una unidad lógica de trabajo (UNT) es un grupo de proposiciones que deben ejecutarse juntas para prevenir la actualización parcial de la base de datos. El comando BEGIN comienza una UTL y el COMMIT termina la UTL.

Si una UTL termina antes del COMMIT, o si se usa el ROLLBACK ,todos los datos actualizados son automaticamente cancelados, regresando a la base de datos a su estado original (antes de comenzar la UTL). Esto es importante para preservar la integridad de las bases de datos.

3) Control de datos (DCL- Data Control Lenguaje).

SQL provee una clase de proposiciones involucradas con la administración de la base de datos. Estas proposiciones definen el tipo de privilegio que se le dara a los usuarios, tales proposiciones son , GRANT Y REVOKE.

La sintaxis para la proposicion GRANT es:
GRANT -un privilegio(select, update, insert)
ON -una tabla o vista.
TO -un usuario.

ejemplo.

GRANT SELECT ON PROVEEDOR TO LUIS.

Este ejemplo otorga privilegios para recuperar información de la tabla proveedores al usuario Luis.

REVOKE.

REVOKE -un privilegio(select, update, insert)
ON -una tabla o vista.
TO -un usuario.

ejemplo.
REVOKE SELECT ON PROVEEDOR TO LUIS.

Este ejemplo anula el privilegios de recuperar informacion de la tabla de proveedores al usuario Luis.

1.3.6 Por Que?

Después de haber descrito qué es una base de datos y para qué sirve , los modelos de las bases de datos, y el lenguaje de consultas SQL. Nos preguntamos ¿ por qué cambiar a un DBMS, por que utilizar el modelo de datos relacional, y como lenguaje de consultas al SQL? . La respuesta se explica a continuación.

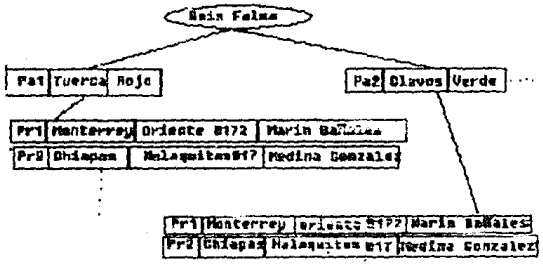
Por qué cambiar a un DBMS?

Un sistema de bases de datos proporciona a las empresas un control centralizado de los datos de operación, en contraste con la situación en la que cada aplicación tiene sus propios archivos (y algunas veces sus discos particulares). De modo que los datos se hallan muy dispersos y por lo tanto son difíciles de controlar.

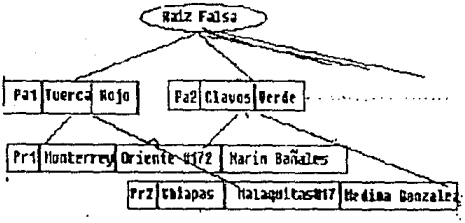
La centralización de la información da la ventaja de evitar la redundancia de información, los datos pueden compartirse, puede aplicarse seguridad, y se puede conservar la integridad (punto muy valioso).

Por qué utilizar el modelo relacional?

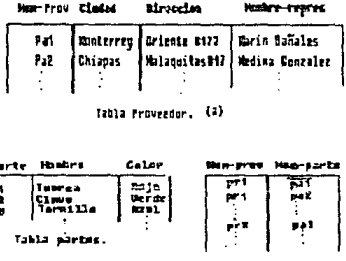
Cualquier estructura jerárquica se puede convertir en estructura de red, y a su vez cualquier estructura de red puede transformarse a estructura tipo tabla. Como se muestra en la figura 13 donde se transforma la estructura jerárquica a red y de red a tabla.



ARBOL. (fig.13 (a)).



RED.



Modelo Relacional.
Fig. 13 .

por qué SQL.

Los estándares son una necesidad primordial en el mundo de las computadoras, así como en las comunicaciones.

La estandarización del lenguaje entre las bases de datos relacionales es de vital importancia, ya que esto permite la interconexión de bases de datos que residan en mainframes, minis, y micros. Obteniendo así una interconectividad entre diferentes ambientes, y el proceso distribuído de información.

Para evitar que cada RDBMS hable su propio lenguaje se introduce al SQL como un estándar.

por primera vez IBM nombro al SQL como un estandar para los RDBMS, desafortunadamente no existe un solo SQL estandar.

El estándar oficial lo define el American National Standar Institute (ANSI), pero se considera que tiene limitaciones de uso. Por lo que el estándar más usado es el de IBM.

2. TECNOLOGIA DE DBMSs

Introducción.

Como se vió en el capitulo uno en la seccion generalidades de DBMSs , un sistema manejador de bases de datos consta de varios componentes.

Entre los componentes se encuentran, el optimizador de consultas, el manejador de autorizaciones e integridad, el control de concurrencia y un manejador de recuperaciones de los cuales hablaremos a continuación.

2.1 EL OPTIMIZADOR DE CONSULTAS.

Para el caso de los manejadores que utilizan el modelo relacional y como lenguaje de consultas al SQL (utilizados en nuestro estudio) , la petición de datos se puede hacer de diferentes maneras, esto es que la consulta puede expresarse de diferentes formas, cada una de las formas de expresar la consulta sugiere una estrategia para encontrar la respuesta . Estas estrategias son llamadas trayectorias de acceso. El procesamiento de recursos que son necesarios para recorrer las diferentes trayectorias de acceso para la recuperación de datos puede variar. Consecuentemente, el optimizador es el componente del DBMS responsable de transformar la pregunta en una forma equivalente que pueda resolverse más eficientemente.

El optimizador juega un papel muy importante en el rendimiento ("performance") del DBMS .

Desde hace tiempo los vendedores de DBMSs no difunden la tecnologia de sus optimizadores, sin embargo, es posible para los usuarios considerar un número de productos y evaluar su optimizador utilizando alguna base de datos propia.

El optimizador se puede considerar como un sistema que incorpora los conccimientos del administrador de bases de datos de determinada empresa y la experiencia de muchos investigadores de las ciencias del acceso a las bases de datos. Estos conocimientos toman la forma de estadisticas que describen a la base de datos (tamaño de la base de datos , índices, descripción de campos , etc).

Entonces antes que se pueda optimizar una pregunta se debe traducir a una forma interna reconocible al manejador de consultas, despues la consulta del usuario se le verifica la sintaxis y se checa que los nombres en la consulta sean nombres registrados en la base de datos.

Una vez que se ha traducido la consulta a una forma interna tal como el álgebra relacional, comenzara el proceso de optimización el cual esta dividido en varias fases las cuales son:

a) encontrar una estrategia detallada para procesar la consulta -> orden en que se ejecutarán los comandos

b) encontrar una estrategia que utilice menos accesos al disco.

A continuación se darán ejemplos de formas que utiliza el optimizador para hacer más rapidos los tiempos de respuesta de las consultas.

Consideremos la siguiente pregunta:
Encontra a todos los clientes que suministran la parte2, el nombre de la pieza y la cantidad que suministran , si recordamos las tablas de esta base son: proveedor,partes y embarques (figura 1).

Num-prov	Ciudad	Direccion	Nombre-repres
Pr1	Monterrey	Oriente #172	Marin Rafales
Pr2	Chiapas	Malaquitas#17	Medina Gonzalez
Pr3	Mexico	Saturno # 7	Guadarrama G.
.	.	.	.
.	.	.	.
.	.	.	.
.	.	.	.

Tabla Proveedor. (a)

Num-parte	Nombre	Color
Pa1	Tuerca	Rojo
pa2	Clavos	Verde
Pa3	Tornillos	Azul
.	.	.
.	.	.

Tabla Partes (b).

Num-Prov	Num-parte	cantidad
pr1	pa1	300
pr1	pa2	200
pr2	pa1	400
Pr3	pa2	200
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.

Tabla Embarques (c)

fig.1 tablas simplificadas para ejemplo.

La consulta traducida al álgebra relacional sera la siguiente:

¶proveedor.nombre,partes.nombre, embarques.cantidad (&
proveedor.num_prov=embarque.num_prov & embarques.parte =
pieza.parte &embarque.parte=par2(proveedorXpiezaXembarque)).

proveedorXpiezaXembarque.

pr1 Monterrey	Oriente # 172	Marin bañales Enrique	Pa1 Tuerca Rojo	Pr1 Pa1 300
pr1 Monterrey	Oriente # 172	Marin bañales Enrique	Pa1 Tuerca Rojo	Pr1 Pa2 200
pr1 Monterrey	Oriente # 172	Marin bañales Enrique	Pa1 Tuerca Rojo	Pr2 Pa2 400
pr1 Monterrey	Oriente # 172	Marin bañales Enrique	Pa1 Tuerca Rojo	Pr3 Pa1 200
pr1 Monterrey	Oriente # 172	Marin bañales Enrique	Pa2 Clavos Verde	Pr1 Pa1 300
pr1 Monterrey	Oriente # 172	Marin bañales Enrique	Pa2 Clavos Verde	Pr1 Pa2 200
pr1 Monterrey	Oriente # 172	Marin bañales Enrique	Pa2 Clavos Verde	Pr2 Pa2 400
pr1 Monterrey	Oriente # 172	Marin bañales Enrique	Pa2 Clavos Verde	Pr3 Pa1 200
pr1 Monterrey	Oriente # 172	Marin bañales Enrique	Pa3 Tornillo Azul	Pr1 Pa1 300
pr1 Monterrey	Oriente # 172	Marin bañales Enrique	Pa3 Tornillo Azul	Pr1 Pa2 200
pr1 Monterrey	Oriente # 172	Marin bañales Enrique	Pa3 Tornillo Azul	Pr2 Pa2 400
pr1 Monterrey	Oriente # 172	Marin bañales Enrique	Pa3 Tornillo Azul	Pr3 Pa1 200
pr2 Chiapas	Malaquitas # 1	Medina Gonzalez Ruben	Pa1 Tuerca Rojo	Pr1 Pa1 300
pr2 Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa1 Tuerca Rojo	Pr1 Pa2 200
pr2 Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa1 Tuerca Rojo	Pr2 Pa2 400
pr2 Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa1 Tuerca Rojo	Pr3 Pa1 200
pr2 Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa2 Clavos Verde	Pr1 Pa1 300
pr2 Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa2 Clavos Verde	Pr1 Pa2 200
pr2 Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa2 Clavos Verde	Pr2 Pa2 400
pr2 Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa2 Clavos Verde	Pr3 Pa1 200
pr2 Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa3 Tornillo Azul	Pr1 Pa1 300
pr2 Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa3 Tornillo Azul	Pr1 Pa2 200
pr2 Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa3 Tornillo Azul	Pr2 Pa2 400
pr2 Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa3 Tornillo Azul	Pr3 Pa1 200
pr3 México	Saturno # 7	Guadarrama G.	Pa1 Tuerca Rojo	Pr1 Pa1 300
pr3 México	Saturno # 7	Guadarrama G.	Pa1 Tuerca Rojo	Pr1 Pa2 200
pr3 México	Saturno # 7	Guadarrama G.	Pa1 Tuerca Rojo	Pr2 Pa2 400
pr3 México	Saturno # 7	Guadarrama G.	Pa1 Tuerca Rojo	Pr3 Pa1 200
pr3 México	Saturno # 7	Guadarrama G.	Pa2 Clavos Verde	Pr1 Pa1 300
pr3 México	Saturno # 7	Guadarrama G.	Pa2 Clavos Verde	Pr1 Pa2 200
pr3 México	Saturno # 7	Guadarrama G.	Pa2 Clavos Verde	Pr2 Pa2 400
pr3 México	Saturno # 7	Guadarrama G.	Pa2 Clavos Verde	Pr3 Pa1 200
pr3 México	Saturno # 7	Guadarrama G.	Pa3 Tornillo Azul	Pr1 Pa1 300
pr3 México	Saturno # 7	Guadarrama G.	Pa3 Tornillo Azul	Pr1 Pa2 200
pr3 México	Saturno # 7	Guadarrama G.	Pa3 Tornillo Azul	Pr2 Pa2 400
pr3 México	Saturno # 7	Guadarrama G.	Pa3 Tornillo Azul	Pr3 Pa1 200

&embarque.num_parte = pa2 (proveedor X pieza X embarque).

pr1 Monterrey	Oriente # 172	Marin bañales Enrique	Pa1 Tuerca	Rojo	Pr1 Pa2 200
pr1 Monterrey	Oriente # 172	Marin bañales Enrique	Pa1 Tuerca	Rojo	Pr2 Pa2 400
pr1 Monterrey	Oriente # 172	Marin bañales Enrique	Pa2 Clavos	Verde	Pr1 Pa2 200
pr1 Monterrey	Oriente # 172	Marin bañales Enrique	Pa2 Clavos	Verde	Pr2 Pa2 400
pr1 Monterrey	Oriente # 172	Marin bañales Enrique	Pa3 Tornillo	Azul	Pr1 Pa2 200
pr1 Monterrey	Oriente # 172	Marin bañales Enrique	Pa3 Tornillo	Azul	Pr2 Pa2 400
pr2 Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa1 Tuerca	Rojo	Pr1 Pa2 200
pr2 Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa1 Tuerca	Rojo	Pr2 Pa2 400
pr2 Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa2 Clavos	Verde	Pr1 Pa2 200
pr2 Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa2 Clavos	Verde	Pr2 Pa2 400
pr2 Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa3 Tornillo	Azul	Pr1 Pa2 200
pr2 Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa3 Tornillo	Azul	Pr2 Pa2 400
pr3 México	Saturno # 7	Guadarrama G.	Pa1 Tuerca	Rojo	Pr1 Pa2 200
pr3 México	Saturno # 7	Guadarrama G.	Pa1 Tuerca	Rojo	Pr2 Pa2 400
pr3 México	Saturno # 7	Guadarrama G.	Pa2 Clavos	Verde	Pr1 Pa2 200
pr3 México	Saturno # 7	Guadarrama G.	Pa2 Clavos	Verde	Pr2 Pa2 400
pr3 México	Saturno # 7	Guadarrama G.	Pa3 Tornillo	Azul	Pr1 Pa2 200
pr3 México	Saturno # 7	Guadarrama G.	Pa3 Tornillo	Azul	Pr2 Pa2 400

& embarques.parte - pieza.parte (& embarque.parte=par2 (proveedorXpiezaXembarque))

pr1	Monterrey	Oriente # 172	Marin Bañales Enrique	Pa2	Clavos Verde	Pr1	Pa2 200
pr1	Monterrey	Oriente # 172	Marin Bañales Enrique	Pa2	Clavos Verde	Pr2	Pa2 400
pr2	Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa2	Clavos Verde	Pr1	Pa2 200
pr2	Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa2	Clavos Verde	Pr2	Pa2 400
pr3	Mexico	Saturno # 7	Guadarrama G.	Pa2	Clavos Verde	Pr1	Pa2 200
pr3	Mexico	Saturno # 7	Guadarrama G.	Pa2	Clavos Verde	Pr2	Pa2 400

& proveedor.num_prov=embarque.num_prov & embarques.parte = pieza.parte &embarque.parte=par2(proveedorXpiezaXembarque)).

pr1	Monterrey	Oriente # 172	Marin Bañales Enrique	Pa2	Clavos Verde	Pr1	Pa2 200
pr2	Chiapas	Malaquitas # 17	Medina Gonzalez Ruben	Pa2	Clavos Verde	Pr2	Pa2 400

¶proveedor.nombre,partes.nombre, embarques.cantidad (& proveedor.num_prov=embarque.num_prov & embarques.parte = pieza.parte &embarque.parte=par2(proveedorXpiezaXembarque)).

Marin Bañales Enrique	Clavos 200
Medina Gonzalez Ruben	Clavos 400.

Como se observa el producto cartesiano de estas tres relaciones produce un resultado muy grande y sólo interesan tres atributos del conjunto. Dado que la relación resultante es grande, lo más probable es que no pueda mantenerse en memoria RAM por lo que se tiene que generar una relación temporal en disco duro, esto implica que además de los accesos a disco para leer las tres relaciones, también se tiene que hacer para leer la tabla temporal, por lo que si se reduce la tabla temporal la consulta se podria resolver más rápidamente.

Puesto que solo (para este caso) interesan las tuplas para las cuales: parte.num_parte = Par2, nombre de parte y número de proveedor. No es necesario tomar en cuenta las tuplas de la relación embarques en las que no se cumpla embarque.pieza=pa2. Si se reduce la relación embarques a las que solo son necesarias , la tabla temporal se reducirá, la consulta entonces puede ser expresada como se muestra a continuación:

¶proveedor.nombre,partes.nombre, embarques.cantidad (& proveedor.num_prov=embarque.num_prov (& embarques.parte = pieza.parte (&embarque.num_parte = pa2 X proveedor X partes))).

```
&embarque.num_parte = par2Xproveedor  X  partes .

pr1 Monterrey Oriente # 172   Marin bañales Enrique    Pa1 Tuerca   Rojo   Pr1 Pa2 200
pr1 Monterrey Oriente # 172   Marin bañales Enrique    Pa1 Tuerca   Rojo   Pr2 Pa2 400
pr1 Monterrey Oriente # 172   Marin bañales Enrique    Pa2 Clavos   Verde  Pr1 Pa2 200
pr1 Monterrey Oriente # 172   Marin bañales Enrique    Pa2 Clavos   Verde  Pr2 Pa2 400
pr1 Monterrey Oriente # 172   Marin bañales Enrique    Pa3 Tornillo Azul   Pr1 Pa2 200
pr1 Monterrey Oriente # 172   Marin bañales Enrique    Pa3 Tornillo Azul   Pr2 Pa2 400
pr2 Chiapas   Malaquitas # 17 Medina Gonzalez Ruben    Pa1 Tuerca   Rojo   Pr1 Pa2 200
pr2 Chiapas   Malaquitas # 17 Medina Gonzalez Ruben    Pa1 Tuerca   Rojo   Pr2 Pa2 400
pr2 Chiapas   Malaquitas # 17 Medina Gonzalez Ruben    Pa2 Clavos   Verde  Pr1 Pa2 200
pr2 Chiapas   Malaquitas # 17 Medina Gonzalez Ruben    Pa2 Clavos   Verde  Pr2 Pa2 400
pr2 Chiapas   Malaquitas # 17 Medina Gonzalez Ruben    Pa3 Tornillo Azul   Pr1 Pa2 200
pr2 Chiapas   Malaquitas # 17 Medina Gonzalez Ruben    Pa3 Tornillo Azul   Pr2 Pa2 400
pr3 México    Saturno # 7     Guadarrama G.           Pa1 Tuerca   Rojo   Pr1 Pa2 200
pr3 México    Saturno # 7     Guadarrama G.           Pa1 Tuerca   Rojo   Pr2 Pa2 400
pr3 México    Saturno # 7     Guadarrama G.           Pa2 Clavos   Verde  Pr1 Pa2 200
pr3 México    Saturno # 7     Guadarrama G.           Pa2 Clavos   Verde  Pr2 Pa2 400
pr3 México    Saturno # 7     Guadarrama G.           Pa3 Tornillo Azul   Pr1 Pa2 200
pr3 México    Saturno # 7     Guadarrama G.           Pa3 Tornillo Azul   Pr2 Pa2 400
```

```
&  embarques.parte = pieza.parte (&embarque.num_parte = par2
X  proveedor  X  partes).

pr1 Monterrey Oriente # 172   Marin bañales Enrique    Pa2 Clavos Verde  Pr1 Pa2 200
pr1 Monterrey Oriente # 172   Marin bañales Enrique    Pa2 Clavos Verde  Pr2 Pa2 400
pr2 Chiapas   Malaquitas # 17 Medina Gonzalez Ruben    Pa2 Clavos Verde  Pr1 Pa2 200
pr2 Chiapas   Malaquitas # 17 Medina Gonzalez Ruben    Pa2 Clavos Verde  Pr2 Pa2 400
pr3 México    Saturno # 7     Guadarrama G.           Pa2 Clavos Verde  Pr1 Pa2 200
pr3 México    Saturno # 7     Guadarrama G.           Pa2 Clavos Verde  Pr2 Pa2 400
```

```
& proveedor.num_prov=embarque.num_prov  (&  embarques.parte
= pieza.parte (&embarque.num_parte = par2  X  proveedor  X
partes)).

pr1 Monterrey  Oriente # 172   Marin bañales Enrique   Pa2  Clavos Verde  Pr1 Pa2 200
pr2 Chiapas    Malaquitas # 17  Medina Gonzalez Ruben  Pa2  Clavos Verde  Pr2 Pa2 400
```

```
¶proveedor.nombre,partes.nombre, embarques.cantidad (&
proveedor.num_prov=embarque.num_prov  (&  embarques.parte =
pieza.parte (&embarque.num_parte = par2  X  proveedor  X
partes))).

Marin  Bañales Enrique     Clavos  200.
Medina Gonzalez Ruben      Clavos  400.
```

Como se vió en este caso el optimizador tomara la segunda opción para resolver la consulta ya que la tabla temporal ocupa menos espacio, consecuentemente se tendra menos acceso a disco.

Otra manera de hacer más pequeña la tabla temporal es ejecutar el producto en diferente orden , dando un resultado más pequeño como se muestra a continuación.

Consideremos la expresion:

¶proveedor.nombre_repres,partes.nombre,embarques.cantidad (&proveedor.num_prov=embarque.num_prov(&partes.num_parte= embarque.num_parte(&embarque.parte=pa2XproveedorX partes))).

Podría optarse por calcular primero proveedor X partes y después unir el resultado con &embarque.parte= pa2 , sin embargo puede ser probable que la relación proveedor X parte sea grande puesto que tiene una tupla por cada proveedor y una tupla por cada parte. Sin embargo &embarque.parte= pa2 es una relación pequeña, además se puede reducir mas aún el tamaño de la tabla temporal quitando atributos que no aparecerán en el despliegue o que no se utilizarán para procesar las operaciones subsecuentes , en este ejemplo se podrían quitar los atributos ciudad, dirección, color de la pieza , de la relación &embarque.parte= pa2 X prov eedorX partes.

La siguiente fase de optimización es encontrar una estrategia detallada para procesar la consulta (plan de acceso), así que en primera instancia se estima el costo de procesamiento de la consulta . La estrategia que se elija dependera en gran parte de la información que tiene el diccionario de datos ya que el diccionario contiene el tamaño de cada relación (número y tamaño de las tuplas) y el número de valores distintos que aparecen en la relación para cada atributo.

La información cambia constantemente ya que se hacen modificaciones a las relaciones y dado que los DBMS no actualizan la información del diccionario cada vez que se realiza una modificación a las tablas, la información que toma el optimizador no es del todo exacta por lo que la estrategia tomada puede no ser la mejor.

La información estadística es muy importante cuando se tienen índices en las relaciones. Los índices permiten tener acceso rápido a las tuplas que cumplen con un valor determinado en la llave de indexación, además existen índices que son "clostered" , lo que permiten estos es que el rendimiento sea mejor ; Ademas permiten que los registros de un archivo se puedan leer en un orden que corresponda aproximadamente al orden físico por lo que permiten leer las tuplas en bloques.

Como también es de pensarse se utiliza tiempo para leer los índices que se encuentran en el disco, por lo que es

necesario considerar estos accesos cuando se estima el costo de la consulta tomando en cuenta índices.

Para ilustrar lo anterior supongase que deseamos encontrar : todos los números de proveedor que viven en la ciudad de Monterrey y su nombre sea Luis .

Supongamos también que se encuentra la siguiente información en el diccionario de datos.

- Cada 20 tuplas de proveedor es igual a un bloque.
- Número de ciudades diferentes es igual a 50.
- Nombres de proveedores diferentes es igual a 200
- La relación proveedor contiene 10,000 tuplas.

Además contiene un índice "clostered" por ciudad y uno sin "clostered" por nombre.

Puesto que el número de ciudades es igual a 50 es de esperarse que (10,000/50) 200 correspondan a Monterrey , si se emplea el índice por ciudad se tendrían que hacer (200/20) 10 accesos a disco para cargarlos al buffer. Además se tiene que contar cuantos bloques de índices se leerán.

En total se acumulan los bloques a leer de la tabla proveedor y los bloques de índices, por lo que nos podríamos percatar que la solución será tal vez mejor si utilizamos el otro índice (por nombre).

Si se utiliza el índice por nombre el número de accesos a disco será el siguiente:

Puesto que el diccionario de datos contiene 200 nombres diferentes de proveedor en 10,000 registros es de esperarse, en el peor de los casos que (10,000/200) 50 correspondan a Luis y como el índice no es "clostered", es decir que los registros no se podrán leer en un orden que corresponde aproximadamente al orden físico, es de esperarse pues, que se hará acceso a disco 50 veces por que tal vez se requiera de la lectura de un bloque por cada tupla, más los bloques de índices que existen para accesar los datos, suman más de 50. Llegando a la conclusión que es mejor utilizar el índice "clostered".

Si ninguno de los dos índices fuera "clostered" se tendría como mejor opción esta última (por nombre), ya que en la otra opción se tendrían que hacer 200 accesos a disco.

Existen otras técnicas que hacen que el tiempo de respuesta sea más rápido tal es el caso de la denormalización.

Los resultados de la normalización es usualmente muchas tablas pequeñas de datos. Cuando estos datos son unidos dinámicamente, el resultado es un incremento en la entrada y salida en disco.

Se pueden tener ganancias significativas en tiempo de respuesta si se realiza la denormalización en el diseño de la base de datos.

La denormalización consiste en tomar datos normalizados y producir un diseño que reordene los datos para el óptimo acceso y manipulación. Mediante una matriz se checa que uniones se realizan más frecuente y entonces se unen esas tablas físicamente.

2.2 SEGURIDAD E INTEGRIDAD.

2.2.1 Seguridad.

El término seguridad se refiere a la protección de los datos que contiene la base de datos contra accesos ilegales, modificaciones o destrucción no autorizada. Existe un gran interés y dedicación al aspecto de seguridad de las bases de datos. Uno de los objetivos primordiales de un Sistema Manejador de Bases de Datos Relacionales (RDBMS) es proporcionar un nivel de seguridad bastante confiable en los datos.

Como los manejadores de bases de datos tienen la habilidad de compartir datos, además la característica de que debe existir irredundancia de datos ó por lo menos tenerla controlada , se desprende la pregunta : ¿que tan buena es la seguridad que proporciona el RDBMS?.

Consideremos una base de datos donde varias aplicaciones pueden tener derecho a leer solo algunas columnas, otras aplicaciones puede que tengan más autoridad y pueden leer toda la tabla de la base de datos . Ahora imaginemos que todas las aplicaciones tienen derecho de accesar todos los campos de la tabla ¿quién sera el responsable de la introducción de datos erróneos?, o de la eliminación de información importante ,¿ como se puede mantener el estado correcto de la tabla si todo mundo tiene acceso a ella ?. Al punto que queremos llegar es que el RDBMS debe tener mecanismos que controlen el acceso a los datos más capaces que los proporcionados por el sistema operativo de la máquina (D.O.S. ,O.S/2) y el de la red (Novell Netware, Lan Manager, etc).

ALgunas formas de acceso no autorizado pueden ser :
- Lectura de datos sin la autorización debida.
- Modificación no autorizada.
- Borrado de datos no autorizado.
- Inserción no autorizada.

Para proteger los datos de actualización y lectura no autorizada el usuario debe de pasar por los siguientes pasos mínimos:

1.- Acceso a la máquina PC.

El usuario debe proporcionar una clave de acceso que es secreta para que la máquina realice el "boot -strap " .

2.- Acceso a la red.

El usuario deberá proporcionar un identificador de usuario. Y mas aún para que el sistema verifique que el usuario es autorizado debe pedir una palabra de paso secreta.

3.- Acceso al RDBMS.

Una vez que el usuario ha pasado los dos niveles anteriores se procede a accesar al RDBMS , que es de manera análoga a la descrita anteriormente, el RDBMS determinará la validez del nombre del usuario y la palabra de paso secreta, y si el usuario puede o no tener acceso a determinada parte de la base de datos.

Cada vez que se ejecuta un comando es verificado por el RDBMS con la información que se encuentra en su catálogo interno. Determina si el nombre de la relación o relaciones y los campos (atributos) que se nombraron en el SELECT , UPDATE,INSERT o DELETE , son accesibles al usuario y además si este tipo de comando es permitido para él, ya que cada vez que se ejecuta un comando es verificado por el RDBMS .

Una forma de implementar la seguridad es generando vistas, las cuales son ventanas que permiten ver o modificar un subconjunto de la información contenida dentro de una tabla o tablas de la base de datos, por ejemplo una ferretería puede tener una base de datos que contenga una tabla con los atributos: num_proveedor , ciudad, dirección y nombre de representante, una vista de la tabla podría contener solo las columnas num_proveedor , nombre y dirección. Usuarios que no necesiten saber la ciudad pueden usar esta vista .

Otra forma es la implementación a nivel relación . En esta parte el DBA puede permitir o no que el usuario tenga acceso directo a una relación. Más aún, en una vista o relación, un usuario puede tener acceso a la información, pero no tendrá los privilegios de modificar, borrar o insertar información.

Para realizar los tipos de seguridad mencionados arriba en bases de datos relacionales y con lenguaje de consultas SQL, se pueden generar vistas de la siguiente manera:

```
CREATE VIEW prov_pieza AS (SELECT num_prov, nombre, ciudad
FROM proveedor).
```

En esta vista se utilizan las tablas de la figura 9. del capítulo uno . El efecto que tiene esta vista es que el usuario solo verá algunos de los atributos de la relación proveedor.

Como se mencionó, el usuario puede entonces contar con distintas formas de autorizacion respecto a partes de la base de datos , entre ellas se encuentran:

- Autorización de lectura-> Permite leer y no modificar la base de datos.
- Autorización de inserción -> Que permite insertar datos nuevos pero no eliminar.
- Autorización de borrado -> Que permite eliminar datos.

Además puede o no el usuario tener privilegios de cambiar la base de datos, entre estos privilegios se encuentran:
- Creación y eliminación de índices.
- Creación de relaciones nuevas.
- Privilegios para agregar o eliminar atributos de una relación.
- Privilegios de eliminación de relaciones.

En un RDBMS la máxima autorización es otorgada al DBA. Al instalarse el RDBMS ya trae un DBA definido. Este DBA puede autorizar a nuevos usuarios, crear tablas, crear índices, autorizar a nuevos usuarios para que ellos creen tablas, índices, etc.

Para poder realizar lo mencionado en el parrafo anterior mediante SQL, existen dos comandos, GRANT y REVOKE.

GRANT INSERT,DELETE ON proveedor TO Luis.
En este comando estamos permitiendo al usuario Luis insertar y borrar en la tabla proveedores.

GRANT CREATE TABLE TO Luis.
En este ejemplo estamos permitiendo al usuario Luis crear tablas en la base de datos.

GRANT ALL ON proveedor TO Luis.
En este ejemplo al usuario Luis le estamos otorgando todos los permisos existentes para esa tabla (seleccción y actualización).

EL comando REVOKE nos permite eliminar los permisos que hayamos otorgado con el comando GRANT.

Los ejemplos anteriores quedan de la siguiente manera:
REVOKE INSERT, DELETE ON proveedor TO Luis .
REVOKE CREATE TABLE TO Luis.
REVOKE ALL ON proveedor TO Luis.

2.2.2 Integridad.

Es la forma de garantizar que los cambios que se hagan por los usuarios autorizados no resulte una pérdida de validez y exactitud en la información de la base de datos.

" de acuerdo con un estudio realizado con respecto al desarrollo de sistemas, se estimó que el 80% de los lenguajes de tercera generacion (Cobol, Basic, Pascal, Fortran, etc.) tienen más código con respecto al mantenimiento de archivos o programas relacionados con el procesamiento de transacciones , y aún más código para el acceso de archivos -incluyendo definición , recuperación , selección y creación de registros- contando la edición de datos para validación , menos del 20% de este código se refiere a cálculos y transformaciones de datos después de la lectura y antes de la escritura" [1].

De acuerdo con lo antes citado todos los RDBMS proporcionan alguna forma para garantizar que los cambios que se hagan por los usuarios autorizados no resulte una pérdida de validez y exactitud de la información de la base de datos.

Existen valores inválidos que son introducidos al momento de estar capturando los datos, en algunos casos, puede ser prácticamente imposible evitar estos valores inválidos; por ejemplo, puede no haber manera de detectar que un embarque de piezas que se introdujo por 2,000, en realidad debería de ser 20,000, aunque el volumen de 1,200,000 para alguna situación particular podria ser un error detectable.

Los mecanismos de integridad se dividen en dos categorías: integridad referencial e integridad semántica.

Integridad semántica.
Se refiere a las reglas internas de alguna empresa en particular. Por ejemplo puede requerirse que el valor de un campo particular quede comprendido dentro de un cierto rango.

Por ejemplo el atributo de embarques, cantidad, debe tener valores comprendidos entre 100 y 1,000.
Se puede requerir que un campo tome solamente ciertos valores especificos. Por ejemplo el campo de nombre de pieza de la tabla pieza debe tomar solamente los valores rojo, verde, azul, gris, café . De tal manera que al capturarse otro color, marca error el RDBMS.

[1] Paul Conte.
"Understandig Relational Data Bases"
Computer Languaje Mayo 87, Pag.47.

Integridad referencial.

Es garantizar que ciertas relaciones entre los renglones de tablas diferentes sea mantenido. Estas relaciones deben ser especificadas por el DBA cuando se crean las tablas en la base de datos. Por ejemplo consideremos la tabla de embarque, siempre que se capture un embarque, el cual consiste de número de proveedor, número de pieza y cantidad de piezas se debe checar que estas dos claves introducidas (número de proveedor y número de pieza) existan en las tablas proveedor y partes, de tal manera que no se podra introducir un proveedor que no este registrado en la tabla proveedor , o de igual forma que una pieza no este registrada en la tabla pieza.

Otra forma de garantizar la integridad y además fácil de implementar es crear indices únicos en los atributos , estos indices únicos permiten que solo pueda existir un registro por cada valor de la clave , por ejemplo un número de proveedor no debe de existir dos veces , entoces se crea un indice del tipo único bajo el atributo número de proveedor de la tabla. Cuando se inserta un registro a la tabla se realiza una búsqueda para checar si existe otro valor igual al introducido.

Existe otra forma de conservar la integridad y que también es fácil de implementar ; Esta es la de limitante de dominio.

La prueba de limitante de dominio es la que permite en el dominio de algún atributo valores vacios, pero en algunos atributos prohibirse.

Por ejemplo tomemos la tabla de proveedores a la cual le aplicaremos el tipo de integridad de dominio en los campos num_proveedor y nombre de representante prohibiendo la entrada de valores vacios. ahora nosotros introduciremos un registro de la siguiente manera :

```
INSERT INTO PROVEEDOR VALUES ("pr8", "sonora", "estrella
# 12" ," Martin Arellano").
```



```
INSERT INTO PROVEEDOR VALUES ("pr8", "  ", "estrella #
12" ," Martin Arellano").
```

al insertarse estos valores el RDBMS no marcara error, pero si introducimos los siguientes valores:

```
INSERT INTO PROVEEDOR VALUES ("  ", "sonora", "estrella #
12" ," Martin Arellano").
```

si marcara error. por la limitante de dominio.

Existen otros métodos para conservar la integridad de los datos, entre ellos estan los disparadores ("triggers").

Un disparador es un procedimiento que contiene comandos SQL que ejecuta el RDBMS en forma automática cada vez que se modifican datos relacionado con el disparador.

Para diseñar un disparador se tiene que realizar lo siguiente:
- Especificar las condiciones en las que se va a ejecutar el disparador (una condición en SQL)
- Especificar las condiciones que se deben realizar cuando se ejecute el disparador (comandos SQL).

Por ejemplo el DBA realizó un disparador que se ejecuta cuando se quiere borrar un registro de la tabla proveedores , el cual consiste de lo siguiente: se debe checar que no exista algún registro en la tabla embarques que contenga el mismo número de proveedor y si llegara a existir se desplegará el mensaje "no se puede borrar este registro".

Supongamos que nosotros en algun momento deseamos borrar el proveedor con clave pr1 , pero existen registros en la tabla embarques con ese mismo número de proveedor, entonces al ejecutarse el disparador se desplegara el mensaje " no se puede borrar este registro".

2.3 CONTROL DE CONCURRENCIA

Dado que algunos RDBMS operan bajo red varios usuarios pueden hacer peticiones al mismo tiempo o sea, se pueden ejecutar varias transacciones a la vez (concurrentemente).

Es necesario que el RDBMS controle la interaccion entre las transacciones concurrentes para evitar que se pierda la consistencia de los datos, ya que algún usuario puede iniciar su transacción con un estado consistente y terminar con un estado inconsistente en la información, iniciar con un estado inconsistente de la base de datos y terminar con un estado inconsistente , o iniciar con un estado consistente, pero en el transcurso de la transacción por lo mismo que los datos son accesados por varios usuarios a la vez puede que la transacción tome datos que están temporalmente inconsistentes.

Existen varias técnicas para controlar la concurrencia y consistencia de los datos, por lo que no existe un estandar para el manejo de este problema. Entre las técnicas que existen está la de candados, la cual consiste en marcar los datos como ocupados al momento que se accesan. Existen otras técnicas como la de probar si la transacción es seriable[1] por medio de gráficos como se verá posteriormente.

[1]Se refiere a que el resultado de procesar un conjunto de transacciones de manera concurrente sea el mismo que el que se produciría si se ejecutara una por una.

Tomando en cuenta las tablas de la fig.9 del capítulo uno , supongamos que hay devolución de piezas y se cambian unas por otras además existen en pa1 100 piezas y en pa2 100.

Sea la transacción uno (T1) -> descontar 10 unidades a la pieza uno y agregar 10 unidades a la pieza 2.

```
T1:
lee  (pa1).
pa1=pa1 -10
escribe(pa1).
lee (pa2)
pa2=pa2 +10
escribe  (pa2).
```

la transacción dos (T2) transfiere el 5% del total de unidades de la pieza uno a la pieza dos por lo que queda de la siguiente manera:

```
T2:
lee (pa1)
var_temporal = pa1 * 0.05.
pa1=pa1 - var_temporal.
escribe (pa1)
lee (pa2)
pa2= pa2 + var_temporal.
escribe (pa2).
```

Supongamos que T1 se ejecuta primero que T2, el total de pa1 + pa2 se conserva en ambos casos , de manera similar sí se ejecuta primero T2 y despues T1 se conserva aún el total (pa1+ pa2) para cada transacción.

Cuando se ejecutan las transaciones en paralelo pueden suceder estados incorrectos como se muestra a continuación:

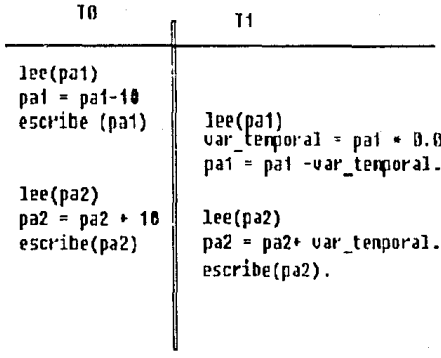


Fig 3 .

Después de llevarse a cabo la ejecución de la figura 3. se llega a un estado donde los valores de pa1 y pa2 son, 275 y 125 respectivamente los cuales son correctos , pero existen transacciones que pueden resultar en un estado inconsistente como lo muestra la figura 4.

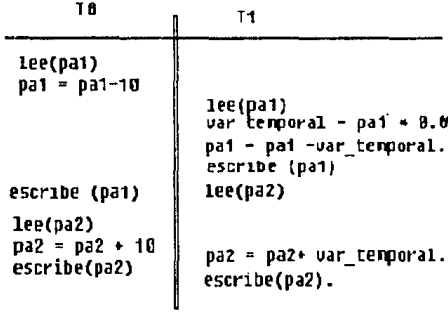


fig 4.

Después de haber ejecutado las dos transacciones pa1 es igual a 290 y pa2 es igual a 115 lo cual es un estado inconsistente ya que se ha ganado 5 piezas en el transcurso de las transacciones. por lo que el cuadro de la figura 4 no conserva la consistencia, es decir no es seriable.

2.3.1 lectura antes de escritura

Para probar si una serie de transacciones que son ejecutadas concurrentemente de alguna forma cumplen con la seriabilidad existe el metodo llamado lectura antes de escritura , para hacerlo se construye una gráfica dirigida, llamada gráfica de precedencia , esta gráfica consiste de la pareja G (V,E), donde :

V - es el conjunto de vértices.
E - es el conjunto de aristas.

El conjunto de vertices es tomado como el conjunto de transacciones que participan concurrentemente.

El conjunto de aristas consiste de todos los Ti -> Tj para los cuales se cumplen las dos condiciones siguientes:

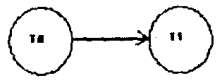
a) Ti ejecuta escribe (dato) antes de que Tj ejecute lee (dato).
b) Ti ejecuta lee (dato) antes de que Tj ejecute escribe (dato).

Por ejemplo :

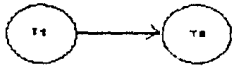
T0	T1
lee(pa1) pa1 = pa1-10 escribe (pa1) lee(pa2) pa2 = pa2 + 10 escribe(pa2)	
	lee(pa1) var temporal = pa1 * 0.05 pa1 = pa1 -var_temporal. escribe (pa1) lee(pa2) pa2 = pa2+ var_temporal. escribe(pa2).

Fig. 5.

La gráfica correspondiente al cuadro de arriba es la siguientes:



ya que la transacción T0 se ejecuta (en su totalidad) antes de ejecutarse la transacción T1, si se ejecutara la transacción T1 primero y después T0 la gráfica seria la siguiente:

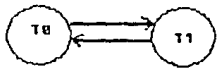


pero si las transacciones se ejecutan concurrentemente como se muestra a continuación puede suceder inconsistencia .

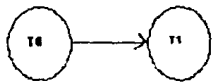
Sea el cuadro :

T0	T1
lee(pa1)	
pa1 = pa1-10	
	lee(pa1)
	var temporal = pa1 * 0.05
	pa1 = pa1 -var_temporal.
	escribe (pa1)
	lee(pa2)
escribe (pa1)	
lee(pa2)	
pa2 = pa2 + 10	
escribe(pa2)	pa2 = pa2+ var_temporal.
	escribe(pa2).

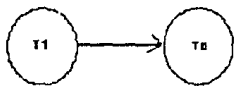
su gráfica correspondiente es:



ya que el cuadro contiene la arista :



por que T0 ejecuta una lectura (pa1) antes de que T1 ejecute escribe (pa1) y también contiene la arista :



ya que T1 ejecuta la lectura de pa2 antes de que T0 ejecute escribe (pa2).

Entonces podemos ver que sí la gráfica resultante contiene un ciclo por lo que no es seriable, es decir se llegará a un estado inconsistente de la información.

Para probar si ei cuadro de ejecución es seriable se requiere de la construccción de la gráfica y aplicar un algoritmo de detección de ciclos.

2.3.2 Candados

Existen muchas formas de controlar la consistencia de las transacciones concurrentes, entre ellas esta la de

candados, esta técnica consiste en que los datos se hagan mutuamente excluyentes, es decir que mientras una transacción accesa un dato, ninguna otra transacción puede modificarlo. El método que se utiliza con mayor frecuencia para implantar esto, es permitir que una transacción accese un dato solamente si tiene actualmente un candado en ese dato.

En esta tesis se tomará en cuenta Bloqueo compartido (candado compartido) y bloqueo exclusivo (candado exclusivo).

Candado Compartido-> significa que al mismo tiempo, más de un usuario puede poner un candado y accesar un registro ó bloque de registros. Cuando se pone este tipo de candado sobre los registros sólo se pueden leer.

Candado exclusivo -> son puestos mientras la transacción dura (hasta un "commit"), cuando se tiene un candado exclusivo se puede leer y actualizar (modificar, inserta y borrar).

Una transacción debe poner un candado al registro mientras lo está accesando. Existen ocasiones en que no se debe liberar al registro inmediatamente, ya que existe la posibilidad de que se pierda la consistencia.

Para ilustrar lo anterior, sea la transacción T3 que quita diez unidades a la pa2 y agrega estas diez unidades a pa1 , además supongamos que existen para la pa1 300 unidades y para la pa2 100 unidades. Para denotar candado compartido , candado exclusivo y liberación de candado tomaremos las abreviaturas CC, CE, LC para cada uno respectivamente.

```
T3:
     CE (pa2)
     lee (pa2)
     pa2=pa2 -10
     escribe  (pa2)
     LC (pa2)
     CE (pa1)
     lee (pa1)
     pa1= pa1 +10
     escribe (pa1)
     LC (pa1).
```

La transacción T4 despliega el total en unidades de la pa1 y de pa2.

```
T4:
     CC (pa1)
     lee (pa1)
     LC (pa1)
     CC (pa2)
     lee (pa2)
     LC (pa2)
     despliega (Pa1 + pa2).
```

Si estas transacciones se ejecutan T3 y T4 ó T4 y después T3 el resultado siempre es 400. Pero si estas transacciones se ejecutan de manera concurrente se puede tener el cuadro siguiente:

T3	T4
CE(pa2) lee(pa2) pa2 = pa2 -10 escribe(pa2) LC(pa2)	
	CC(pa1) lee(pa1) LC(pa1) CC(pa2) lee(pa2) LC(pa2) despliega(pa1 + pa2)
CE(pa1) lee(pa1) pa1 = pa1 +10 escribe(pa1) LC(pa1)	

Si observamos el cuadro anterior podemos percatarnos que se a llegado ha un estado inconsistente por liberar el candado demasiado rápido. El valor desplegado es 390. La duración de los candados determina el grado de consistencia que se pondrá a la base de datos.Para resolver esto, existen varios niveles de aislamiento entre transacciones los cuales son: "Cursor Stability" , "Repeatable Read" y "Read Consistency".

Cursor Stability (CS) son llamadas asi a páginas que adquieren un candado compartido mientras se esta leyendo es decir mientras el cursor esta en "on" . Bajo el metodo Cursor Stability los candados compartidos son eliminados hasta que el cursor libera la página, pero los candados exclusivos son retenidos hasta que la transacción es completada. Por ejemplo si el usuario uno lee la pagina 10A y entonces lee la pagina 10B, el candado compartido.que está puesto en 10A es liberado. Asi permitiendo al usuario dos actualizar la página. Sin embargo, si el usuario uno actualiza un renglón sobre la pagina 10A, el usuario dos no podrá disponer de esta página sino hasta que el usuario uno utilice el "commit".

Repeatable Read (RR) bajo este método los candados exclusivo y compartido son retenidos hasta que la transacción se termina, es decir que se garantiza que aquellos bloques de registros liberados y que son leidos por

otros usuarios no podrán ser modificados sino hasta que la transacción es cerrada(commit).

Read Consistency (RC) este método no involucra candados, por lo que no hay que esperar; puede ser utilizado solamente para lectura de datos y no para actualización. Este método da una vista instantánea (instancia) de la base de datos consistente de los datos que existieron en el momento de ejecutar el query. Esta caracteristica es utilizada especialmente donde se tienen que imprimir reportes grandes, y así no esperan otros usuarios que quieren actualizar. Esto no quiere decir que sea una copia de la tabla o tablas, ya que el RDBMS utiliza información de los segmentos del Rollback.

Por ejemplo, el usuario Martin hace una consulta a la tabla proveedor, los resultados que son regresados , producto de su consulta son los datos que existieron en la tabla proveedor al tiempo en que comenzó su consulta

Cursor Stability provee una concurrencia más alta en comparación con "Repeatable Read", con Cursor Stability sin embargo, los datos que son leídos como parte de la transacción pueden ser cambiados por otros usuarios del sistema por cada página liberada y así la consistencia es menor.

2.3.2.1 Deadlocks.

Un deadlock ocurre cuando dos usuarios tienen un candado distinto sobre objeto distintos. Cada uno necesita poner un candado al objeto del otro usuario. Cuando esto ocurre, el primer usuario esta esperando a que el segundo usuario libere el candado, pero el segundo usuario no libera el candado sino hasta que el objeto que tiene el primer usuario este libre para tomarlo. Para ilustrar el deadlock consideremos el siguiente cuadro:

T5	T6
CE(pa2) lee(pa2) pa2 = pa2 -10 escribe(pa2)	
	CC(pa1) lee(pa1) CC(pa2)
CE(pa1)	

Como se puede observar T5 tiene un candado en modo exclusivo en pa2 y T6 pide un candado en modo compartido en pa2, T6 espera a que T5 libere a pa2 del candado exclusivo, por otro lado T6 tiene un candado de modo compartido en pa1 y T5 pide un candado de modo exclusivo en pa1. T5 espera que T6 libere a pa1 del candado compartido y T6 espera a qué T5 libere a pa2, así se llega a un deadlock por qué nunca se podra terminar una de las dos transacciones.

Cuando sucede un deadlock el sistema debe detectarlo y eliminar una de las dos transacciones, una vez que se elimina una de las dos transacciones, los datos quedan libres y se puede entonces completar la transacción y posteriormente la transacción eliminada se puede ejecutar en su totalidad.

2.4 ARQUITECTURA CLIENTE - SERVIDOR.

En toda organización los datos deben ser accesibles, confiables , y estar bajo control. Porque las organizaciones deben compartir información entre sus empleados y otras organizaciones los datos deben ser centralizados. Para resolver estos requerimientos, las organizaciones tienen su información centralizada en una base de datos de Mini o Mainframe. los usuarios accesan la información mediante terminales tontas (figura 1).

La centralización de la información permite a los usuarios accesarla fácilmente en todas las áreas de la organización y el administrador de la base de datos asegura la restricción de los datos. En un ambiente en línea, un sistema manejador de base de datos centralizado tiene provista la integridad de datos.

Mientras esta arquitectura es adecuada para un conjunto restringido de aplicaciones, no lo es para aplicaciones que requieren de gráficos por ejemplo, ya que las terminales tontas no pueden soportar gráficos (como windows de Microsoft).

Cuando las PCs arribaron al mercado por primera vez, los usuarios rápidamente descubrieron su potencial. Desarrolladores de software liberaron una avalancha de nuevo software para realizar las aplicaciones que no se podían realizar en el mundo de los Mainframes y Minis y sus terminales tontas.

Aplicaciones tales como hojas electrónicas, procesadores de palabra, software de graficación, DBMSs personales, etc. abrieron nuevas oportunidades para los usuarios aburridos por la limitada capacidad de las terminales tontas.

No tomaron mucho tiempo las PCs para proliferar y mover el mundo centralizado a ellas.

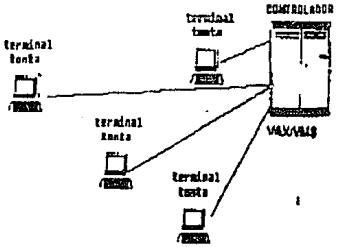


fig.1 Mundo Centralizado.

Desafortunadamente, este paso también crea problemas en términos de la forma en que se comparten los datos, su protección e integridad.

Además los usuarios encuentran dificultad para compartir la información almacenada en las PCs. Los usuarios tienen que mover información de su base de datos a su hoja electrónica y paquetes de graficación y finalmente, posiblemente a un procesador de palabras.

Cuando se tiene un número de PCs (en donde corre un sistema operativo mono-usuario tal como D.O.S.) conectado a un computador servidor, donde se encuentra un sistema operativo de red tal como Novell Netware o Lan Manager, se le llama red (File Server y sus nodos).

El File Server fué el primer intento para resolver el problema de compartir datos por diferentes usuarios y tener un control de ellos y de los datos. Los usuarios pueden compartir la información y programas que existen centralizados en el disco duro del File Server.

El File Server funciona de la siguiente manera:

Supongamos que una PC requiere de información que se encuentra en un File Server con sistema operativo Novell, y en él se encuentra el manejador de archivos dBase, entonces sucede lo siguiente:

1.- Se manda el requerimiento a D.O.S usando la interrupción 21h.
2.- NET3 examina la llamada 21h para determinar si es local o es un requerimiento a la red. Si es local el requerimiento, el BIOS (Basic Input Output System) maneja el requerimiento.
3.- Si es un requerimiento a la red, NET3 lo redirecciona a IPX diciendole que lo envie al File Server.
4.- IPX, pasa el requerimiento a la tarjeta de la PC del usuario.
5 .-La tarjeta (NIC) pone el requerimiento dentro de uno o más paquetes de datos y pasa el paquete al sistema de cable. La tarjeta incluye cierta información con cada paquete, incluyendo dirección fuente y dirección destino. Toda tarjeta de red tiene una unica dirección.
6.- La tarjeta del File Server recibe el paquete (s) y pasa los datos a la memoria del File Server.
7.- La tarjeta del File Server manda un mensaje de reconocimiento a la tarjeta de la PC del usuario por cada paquete que recibe.
8.- El software del sistema operativo de red examina y procesa el requerimiento.
9.- El sistema operativo de la red pasa los datos requeridos de memoria a la tarjeta del File Server.
10.- La tarjeta paquetiza los datos, entonces pasa los paquetes uno a la vez al cable.
11.- la tarjeta de la PC del usuario recibe los paquetes, uno a la vez despaquetizando y poniendo los datos en memoria.
12.- La tarjeta de la PC manda un reconocimiento a la tarjeta del File Server por cada paquete recibido.

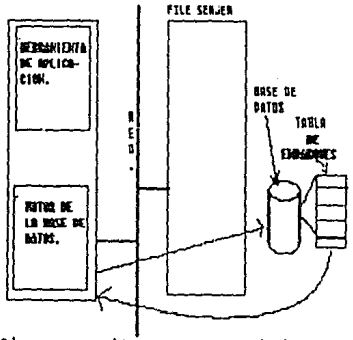


fig.2 Arquitectura Tradicional.

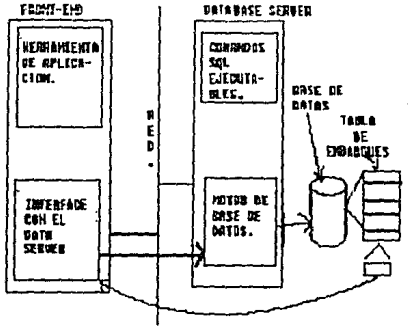


fig.3 Arquitectura Cliente-Servidor.

En primera instancia el File Server fué una buena solución pero en la medida que los datos (no programas) tienen más demanda el File Server deja de ser una buena solución , ya que datos y programas son tratados de igual manera (como se describió anteriormente). El mayor problema con el File Server es su limitado soporte de aplicaciones que requieren de datos seguros, exactos y disponibles a la vez, tal es el caso de un banco. El File Server no garantiza la integridad de datos o recuperación de transacciones en caso de fallas del sistema .

Los File Server son básicamente discos virtuales, por lo que ellos no tiene la inteligencia de un DBMS.

Sobre una red, una aplicación de base de datos no es diferente a cualquier otra aplicación, ya que una red no diferencia los archivos.

En la arquitectura tradicional de DBMSs para red, el File Server envía una copia del software del DBMS a la estación de trabajo y si la estación de trabajo hace una petición de datos, aunque el resultado de la petición sea un solo registro, se envía en su totalidad el archivo o archivos que tiene que ver en la petición (figura 2).

Por ejemplo si el usuario requiere la suma de todas las piezas que se tienen en el archivo embarques, entonces el archivo viajará por el cable de la red y al llegar a la estación de trabajo las cantidades se iran tomando , para después dar el resultado. Por lo antes dicho, si los archivos son muy grandes y varios usuarios hacen peticiones existirá mucho tráfico en la red, degradando el tiempo de respuesta. Además el File Server no puede soportar automáticamente funciones de manejo de datos, especialmente operaciones de multiusuario tales como:

-Automático bloqueo de registros.
-Manejo de transacciones (Rollback ,Commit).
-Recuperación de fallas del sistema.

Los servidores de datos son tecnología relativamente nueva , el primer intento en servidores de bases de datos surgió en 1986 y fué introducido al mercado por Gupta Tecnologies. Pero ha sido hasta estos dos últimos años (1989 y 1990), en que se estan dando a conocer diferentes servidores de bases de datos de diferentes compañías tales como Oracle , Microsoft y Novell.

Los servidores de bases de datos más comerciales se fundamentan en los principios del modelo relacional , y además es el modelo mas difundido en Minis y Mainframes actualmente.

La arquitectura CLIENTE-SERVIDOR (figura 3) mezcla el control y seguridad que caracterizan a los equipos grandes (Minis y Mainframes) con la economía, productividad y flexiblidad que caracterizan a las PCs. En este aspecto la arquitectura cliente-servidor tiene lo mejor de dos mundos.

En la arquitectura cliente-servidor se separan las funciones en dos partes:
el lugar donde se realizan las funciones del manejo de datos (BACK-END) y el lugar donde son mostrados los datos al usuario (FRONT-END) . En la arquitectura tradicional las dos partes se encuentran juntas.

En la arquitectura cliente-servidor cada estación de trabajo PC corre un FRONT-END (software de interface para el usuario) como puede ser una hoja electrónica, alguna aplicación hecha en un lenguaje de programación, etc, mientras otra máquina PC conectada a la red procesa el requerimiento hecho por el usuario, este BACK-END es también llamado motor de base de datos o servidor de datos ("DATABASE SERVER"). Con esta arquitectura, solo los registros que cumplen con la petición serán transferidos del servidor de datos a la estación de trabajo.

Por ejemplo cuando un usuario hace un requerimiento mediante su front-end , el requerimiento es transformado a comandos SQL y posteriormente estos comandos SQL viajan por la red los cuales son atrapados o direccionados al servidor de datos y entonces el servidor los analiza y procesa para posteriormente enviar los resultados. Supongamos que la consulta es: dame el numero de piezas que existe en la tabla embarques, entonces el requerimiento convertido a SQL viajara por el cable y sera recibido por el servidor de datos y enviara solo un número a la estación de trabajo y no enviara toda la tabla para que la estación de trabajo la analize.

Un Data Server puede ser implementado de la siguiente manera:
1.- Puede ser un proceso secundario del File Server de la LAN.
2.- Puede ser un Value-Added Process (VAP) sobre el File Server trabajando como parte del sistema operativo de la LAN.
3.- Puede implementarse un coprocesador en el File Server de la LAN.
4.- Puede ser una PC dedicada en la LAN.
5.- puede ser un proceso " BACKGROUND" sobre una estación de la LAN.
6.- Puede ser un coprocesador en la estación de trabajo de la LAN.

1.- Proceso secundario del File Server de la LAN.
Este es un intento que actualmente recibe mucha atención . El CPU del File Server divide su atención entre las funciones del File Server y las funciones del Data Server.

2.- Servicios de dases de datos como un VAP.
Un Value Added Process es una extensión del software del File Server. Un VAP ejecuta Tareas cuyos resultados son integrados con el trabajo del File Server.

3.- Coprocesador en el File Server
Esta es una tarjeta procesadora separada, instalada en el File Server, los requerimientos de bases de datos son direccionadas al coprocesador.

4.- Servidor de base de datos dedicado.
Un servidor de base de datos dedicado es una segunda PC conectada en la red y usada solo para procesar servicios de bases de datos.

5.- Servicios de bases de datos trabajando como "background" en una estación de trabajo.
Bajo OS/2 se puede usar una estación de trabajo (PC) como un servidor de bases de datos, donde las funciones de servidor de datos son ejecutadas como tareas"background".

Este tiene muchas caracteristicas de los atributos de un servidor dedicado, pero tendrá menor rendimiento debido a el compartimiento del CPU de la estación de trabajo y tarjeta de interface.

6.- Coprocesador en una estación de trabajo.

Se instala un coprocesador en una estación de trabajo PC, y todos los requerimientos de bases de datos son direccionados a esa PC. Este tipo tiene características de un servidor de base de datos separado.

Entre las formas de implementar un servidor de bases de datos mencionados anteriormente, el más usado es el servidor de bases de datos dedicado.

Muchos front-ends de diferentes tipos pueden accesar directamente los datos del Data Server. Los usuarios pueden construir aplicaciones usando lenguajes tales como "C" y dBase IV ver.1.1 o software tales como hojas electrónicas . Las aplicaciones entonces consultan y actualizan datos del Database Server. Cada uno de estos productos (front-ends) tiene sus propias capacidades de manejo de mono-usuario, pero en la arquitectura cliente-servidor, estas facilidades son desviadas y todas las funciones de manejo de datos son asumidas por el Back-End.

Toda la información puede ser compartida entre los usuarios de la red. Una hoja electrónica puede analizar y graficar datos , un usuario puede crear un reporte para exportarlo a una hoja electrónica mediante un lenguaje de programación.

Para conectar esta variedad de aplicaciones con el Database Server, la arquitectura cliente-servidor necesita de un lenguaje común . La industria de bases de datos ha estandarizado al SQL como lenguaje de consultas para el modelo relacional.

Como se vió en el ejemplo de la consulta del usuario: dame la cantidad de piezas que existen en la relación embarques, la solución dada por el File Server y el Data Server es la misma, pero en la arquitectura cliente - servidor existen las siguientes ventajas:

- La estación de trabajo puede accesar múltiples Database Servers. Así que la separación física de los Database Server puede ser tratada por los usuarios como una base de datos distribuida (punto tratado posteriormente). Un usuario puede consultar datos sin saber dónde se encuentran físicamente.
- Además la arquitectura cliente-servidor releva a las aplicaciones (front-end) de todo manejo de datos.
- Otra de las ventajas es que el usuario obtiene un mejor rendimiento ("performance"), ya que el Database Server solo envía los renglones que cumplen con la consulta hecha por el usuario eliminando el sobre tráfico de la red.

-Soporta aplicaciones en linea, teniendo controlada la integridad y seguridad que normalmente en un ambiente tradicional de LAN no se tiene.

-Escalabilidad, permite a las organizaciones remplazar Database Server que están en alguna plataforma por otra ,por ejemplo un Oracle Server que se tiene bajo OS/2 se puede mover a una plataforma UNIX ya que existen versiones de Oracle para UNIX .

- Aunque el Back-End cambie de plataforma el front-end permanece imperturbado por lo tanto las aplicaciones realizadas no requieren de modificaciones, éstas ventajas no existen en Minis y Mainframes.

Las aplicaciones realizadas en los front-ends no requieren de modificaciones.

-Los servidores de bases de datos ofrecen la posibilidad de conectividad de redes heterogéneas . Por ejemplo si un servidor de datos se encuentra en una red Lan Manager y está esta conectada con una red Novell Netware, mediante algún Gateway, alguna estación de trabajo de Novell puede realizar consultas al servidor de datos que se encuentra en Lan Manager.

-Otro de los beneficios de los servidores de bases de datos es que sólo el que servirá como servidor de bases de datos puede tener un procesador rápido (80386) , un disco rápido y mucha memoria RAM, ya que las estaciones no procesan nada con respecto a los datos y por lo tanto pueden ser PCs XTs con 256KB de memoria RAM.

Las investigaciones de mercado indican que la nueva tecnología cliente-servidor, hará que muchos usuarios migren de Minis y Mainframes a LANs.

"Aproximadamente 16,000 configuraciones cliente-servidor se venderán en 1989, y para 1993 serán 179,000"[2].

[2] Data Comunication Marzo 21 1989.

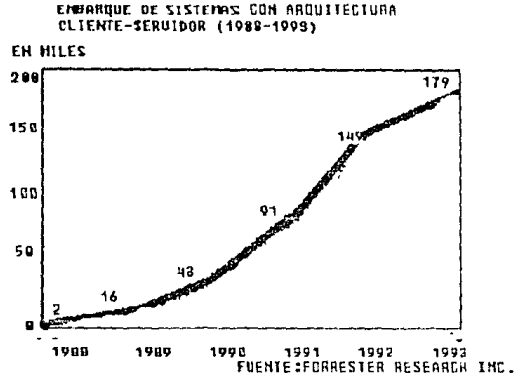


fig.4

2.5 BASES DE DATOS DISTRIBUIDAS

introducción.

Un proceso es distribuido porque se ejecuta en varios procesadores es decir, por ejemplo un programa se separa en partes y son ejecutadas en diferentes procesadores. La arquitectura cliente-servidor se puede decir que es entonces un proceso distribuido, ya que cuando un usuario conectado a red hace un requerimiento a la base de datos , la tarea se divide en dos partes, el Back-end y el Front-end. El Back-end realiza las operaciones de manejo de datos y el Front-end hace la presentacion al usuario. Con la arquitectura cliente-servidor existe la posibilidad de tener varios servidores de base de datos conectados a la red, entonces cualquier estacion de trabajo puede accesar los datos de cualquier servidor. Pero esto no quiere decir que se tienen bases de datos distribuidas.

2.5.1 Bases de datos distribuidas.

Son varios los factores que hacen que una base de datos sea distribuida:

1.- Los datos son almacenados en multiples sitios (PCs) conectados a la red.

2.- Cada sitio es consciente que existen los demas sitios.

3.- Cada sitio permite ejecutar transacciones tanto locales como globales.

En la tecnologia de los sistemas manejadores de bases de datos distribuidas(DDBMS) el optimizador de consultas y control de concurrencia requieren de una inteligencia global.

Una base de datos distribuida se puede diseñar en cuanto a su almacenamiento de tres maneras diferentes: fragmentación, repiticion y fragmentado-repetida.

2.5.1.1 Fragmentación.

La relación (tabla) es dividida en varios fragmentos y almacenados en diferentes sitios de la red. La tabla se puede fragmentar de varias formas las cuales son, fragmentación horizontal y fragmentación vertical. La fragmentacion horizontal de una relación contiene todos sus atributos y parte de sus tuplas. Por ejemplo, tomemos como ejemplo la relacion proveedor, pero cambiando el dominio de la ciudad para efectos explicativos; La tabla se muestra en la figura 5.

Num_Prov	Ciudad	Dirección	Nombre_Repres
pr1	Monterrey	Oriente # 172	Marin Bañales
pr2	Chiapas	Malaquitas #17	Medina Gonzalez
pr3	Chiapas	Saturno # 17	Guadarrama G.
pr4	Monterrey	Calle # 59	Martinez M.
pr5	Monterrey	Fuentes F.	Betancour C.
pr6	Chiapas	Casa Aleman	Majera Amador
pr7	Chiapas	Nueva Vida	Gonzalez Campos.

TABLA PROVEEDOR.

fig.5.

Los siguientes comandos SQL crearán una fragmentación horizontal para almacenarse en el lugar apropiado.

```
SELECT  * FROM  proveedor WHERE ciudad = Monterrey
SELECT  * FROM  proveedor WHERE ciudad =  Chiapas
```

La fragmentación quedaría como se muestra en la fig. 6. La figura 6 (a) se almacenará en el sitio Monterrey la figura 6 (b) se almacenará en el sitio de Chiapas.

Num_Prov	Ciudad	Dirección	Nombre_Repres
pr1	Monterrey	Oriente # 172	Marin Bañales
pr4	Monterrey	Calle # 59	Martinez M.
pr5	Monterrey	Fuentes F.	Betancour C.

(a)

Num_Prov	Ciudad	Dirección	Nombre_Repres
pr2	Chiapas	Malaquitas #17	Medina Gonzalez
pr3	Chiapas	Saturno # 17	Guadarrama G.
pr6	Chiapas	Casa Aleman	Najera Amador
pr7	Chiapas	Nueva Vida	Gonzalez Campos.

(b)

fig. 6.

Una fragmentación vertical por otro lado, incluirá un subconjunto de los atributos de la relación y todas las tuplas.

El siguiente comando SQL realizará una fragmentación vertical.

SELECT num - prov, ciudad FROM proveedor
SELECT num - prov, direccion, nombre - repres FROM proveedor

El resultado es mostrado en la figura 7.

Num_Prov	Ciudad
pr1	Monterrey
pr2	Chiapas
pr3	Chiapas
pr4	Monterrey
pr5	Monterrey
pr6	Chiapas
pr7	Chiapas

(a)

Num_Prov	Dirección	Nombre_Repres
pr1	Oriente # 172	Marin Bañales
pr2	Malaquitas #17	Medina Gonzalez
pr3	Saturno # 17	Guadarrama G.
pr4	Calle # 59	Martinez M.
pr5	Fuentes F.	Betancour C.
pr6	Casa Aleman	Najera Amador
pr7	Nueva Vida	Gonzalez Campos.

(b)

fig.7

En la creación de la fragmentación horizontal, se incluyen todas las columnas (Select *) y se usa una cláusula WHERE para especificar los renglones. Con la fragmentación vertical, sin embargo, se usa una lista de campos para crear un subconjunto de atributos y se seleccionan todos los renglones. Se debe incluir Num - proveedor en ambas fragmentaciones verticales, para identificar cada registro, además se debe incluir en todas las fragmentaciones verticales que se hicieran.

La fragmentación mixta, combina la fragmentacion vertical con la horizontal. Para ilustrar supongamos que deseamos fragmentar la tabla proveedor en tres partes para almacenarlas según nuestras necesidades.

Sean los comandos SQL:

```
SELECT  num- prov, ciudad FROM proveedor WHERE ciudad =
Monterrey
SELECT  num - prov, ciudad FROM proveedor WHERE ciudad =
Chiapas
SELECT  num - prov, direccion, nombre - repre FROM
proveedor
```

Así la relación proveedor se divide en tres fragmentos, fig8(a) , fig 8(b) , fig 8(c). Cada una de ellas puede residir en un sitio diferente.

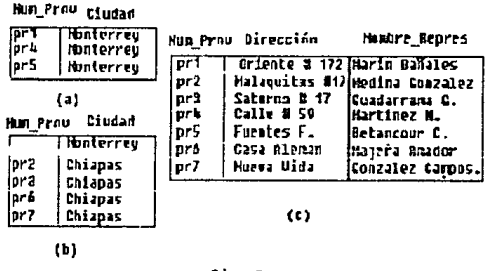


fig.8.

2.5.1.2 Repetición de relaciones

La repetición es duplicar las relaciones através de la red. Se duplican las tablas por dos razones: para maximizar la disponibilidad de los datos, así se reduce el movimiento de información entre sitios y para tener una copia de los datos en caso de que un sitio de la red falle y contenga determinada tabla.

2.5.1.3 Repetición y fragmentación de los datos

Las dos técnicas descritas arriba pueden aplicarse de manera sucesiva a la misma relación o fragmento, es decir las relaciones y fragmentos pueden repetirse y las relaciones y fragmentos pueden fragmentarse.

Como se mencionó anteriormente una relación puede ser almacenada en diferentes sitios de diferente manera, por lo que es necesario que el DDBMS(sistema manejador de bases de datos distribuidas) reduzca al mínimo la necesidad de que el usuario conozca como esta almacenada la relación.

Para dar la transparencia al usuario se debe tener fragmentación, localización y repetición transparente.

2.5.1.3.1 Fragmentación Transparente.

La relación o relaciones que están fragmentadas y depositadas en diferentes sitios aparecerán al usuario como si las relaciones completas residieran en el sitio donde se encuentra.

El DDBMS mapeará a la relación en su fragmento apropiado según la consulta.

Si se cambiara el esquema de almacenamiento de la base de datos no se afectará a la aplicación del usuario. Por ejemplo si se tuviera alguna relación fragmentada en tres partes y cada parte estuviera almacenada en diferente sitio, digamos Ciudad Universitaria, Acatlán y Cuautitlán, pero se decidiera que el fragmento localizado en Ciudad Universitaria se transladará a Aragón, no se afectará a la aplicación del usuario ya que el DDBMS actualizará sus catálogos y para el usuario es trasparente, por que el DDBMS mapeara según la pregunta.

2.5.1.3.2 Localización Trasparente.

Una de las metas de un sistema manejador de bases de datos distribuidas es el concepto de localización trasparente. La intención es que el usuario especifique una pregunta y no tenga la necesidad de saber la localización de los datos que se accesaran, por ejemplo la pregunta en SQL:

```
SELECT num_prov, ciudad, nombre_repres, partes.nombre
FROM proveedor , partes, embarques WHERE
embarque.num_parte = partes.numero_parte and
embarques.num_prov= proveedor.num_prov.
```

y supongamos que las relaciones estan en diferentes sitios ; la transparencia de localización se logra poniendo sinónimos al nombre real de la relación, de tal manera que el nombre real (el que maneja el DDBMS) para proveedor podría ser: servidor1.proveedor, y para las relaciones partes y embarques serían servidor2.partes y servidor1.embarques respectivamente, por lo que si la aplicación del usuario se refiriera a la relación proveedor, partes y embarques el DBA al definir las relaciones debio haber creado sinónimos o ligas para transformar digamos proveedor a servidor1.proveedor, como se muestra a continuación:

```
CREATE SYNONYM server1.proveedor AS proveedor
```

Así si el DBA cambia de sitio una relación de las tres , por ejemplo a la relación proveedor la pasa al server4 , entonces con sólo cambiar el sinónimo la aplicación no se afecta.

2.5.1.3.3 Repetición Transparente.

No es conveniente que los usuarios especifiquen que copia(si es que la base de datos tiene copias) se debe accesar. El DDBMS debe ser el que determine qué copia debe ser accesada cuando se solicite una lectura o actualización(como se verá en las técnicas de optimizador para DDBMS) ya que tiene que ver con la distancia y el medio de comunicación que exista. Además se deben de actualizar todas las copias cuando se solicite una

modificación. Como se mencionó anteriormente el control de concurrencia , integridad de datos y el optimizador de consultas requieren de mayor inteligencia , ya que se debe tener una visión global de toda la base de datos.

El optimizador de consultas debe considerar, además de lo explicado en el capitulo uno el costo de transmisión y el tiempo que se podria ahorrar al ejecutar la consulta en diferentes sitios.

Cuando el optimizador estima el costo de una junta, incluyendo relaciones localizadas en sitios diferentes , por ejemplo se debe de considerar la diferencia en el número de renglones que tendrán que ser transferidos dependiendo sobre qué sitio se ejecute la junta.

Por ejemplo supongamos la consulta SQL:

```
SELECT num_prov, ciudad, nombre_repres FROM
proveedor , embarque WHERE  proveedor.num_prov =
embarque.num_prov  AND  embarque.num_prov='pr1';
```

la consulta es ejecutada en el sitio donde se encuentra la tabla proveedor, y además la tabla de proveedor contiene 2,000 registros y embarques contiene 10,000 registros y supongamos que el resultado son 300 registros. El optimizador debe encontrar la mejor solución para que el costo de transmisión sea menor. Para lograr esto el optimizador debe de optar por enviar una copia de la tabla de proveedor (2,000) al sitio donde se encuentra la relación embarques y como el resultado son sólo 300 registros, la cantidad de registros que se movieron a través del medio de transmisión fueron 2,300, por otro lado, si se hubiera mandado una copia de la relación embarque se hubieran movido 10,300 y no sólo 2,300 .Pero si el optimizador tuviera más información, como por ejemplo las características de cada máquina , tal vez se ejecutaría de otra manera ó si el optimizador pudiera analizar y separar las partes de la consulta para que se ejecute en cada sitio la parte que le corresponde y mandar solo el resultado la consulta se ejecutaría de diferente manera.

En el ejemplo anterior supusimos que las tablas no estaban fragmentadas ni duplicadas en diferentes sitios. Supongamos que un usuario realiza una consulta , la cual requiere de información de alguna relación que tiene varias copias, entonces el optimizador debe escojer la copia más cercana para que el costo de transmisión sea más bajo. Por ejemplo supongamos que el usuario realiza la siguiente consulta :

```
SELECT * FROM proveedor.
```

proveedor se encuentra en la zona1 y en la zona5 la zona1 se encuentra en Naucalpan y la zona5 se encuentra en Tlanepantla; y supongamos que la consulta se realiza en Acatlán, el optimizador debe de optar por la copia que se encuentra en Naucalpan.

Para realizar este tipo de decisiones el optimizador de un sistema manejador de bases de datos distribuídas, debe tener una descripción total de la base de datos distribuida, llamada una vista global. Sin embago cada sitio de la red debe tener acceso a esa vista global.

2.5.2 Integridad de datos distribuidos

Otra característica esencial de un DDBMS es la habilidad de proteger la integridad de una transacción distribuida. Cuando una transacción consiste de multiples operaciones, es imperativo que el DDBMS ejecute a todas ellas o ninguna de ellas.

Supongamos la compañía de distribución de piezas y un DDBMS que ofrece fragmentación transparente. Se puede transferir un proveedor de un sitio a otro con el siguiente comando SQL :

```
UPDATE proveedor SET ciudad = 'Monterrey' WHERE
num_prov = 'pr3';
```

Además supongamos que la tabla proveedor tiene una copia en Sonora, en la actualización se realizan los siguientes pasos:

1-. Adicionar el proveedor Guadarrama G. al sitio ubicado en Monterrey.

2.- Borrar al proveedor Guadarrama G. del sitio ubicado en Chiapas..

3.- Actualizar la copia que existe en Sonora cambiando el atributo ciudad del registro del proveedor "pr3".

Así como este ejemplo pueden existir actualizaciones que en un DBMS son triviales, pero en un DDBMS se pueden convertir en complejas.

Como se vió existen varios sitios involucrados en la actualización, pero estos sitios son suceptibles de fallas, además puede que deje de existir comunicación por ruptura del medio de transmisión, por lo que deben de existir métodos que detecten este tipo de fallas para que exista consistencia en los datos. El método más aceptado para garantizar la integridad de las transacciones en un ambiente de bases de datos distribuídas es el método de dos fases ("two phase commit"). Un sitio conocido como el coordinador controla la transacción distribuida , los otros sitios (los participantes) responden a los comandos del coordinador. La secuencia de acciones es la siguiente:

FASE 1.

- El coordinador envía un mensaje de prepararse para hacer el "commit" a los registros de cada uno de los sitios participantes , al recibir el mensaje cada una de los

manejadores de transacciones de cada uno de los sitios determina si esta dispuesto para hacer el "commit" a la parte de transacción que le corresponde. Si la respuesta es negativa se envía un mensaje al coordinador y se abortará la transacción.
- Si la respuesta es satisfactoria de todos los participantes se continúa con la segunda fase.

FASE 2.
-Si ningún participante está fuera de tiempo (es decir tiene un tiempo para responder) y su respuesta no es de abortar, entonces el coordinador envía un mensaje de hacer el "commit" a la parte que le corresponde.
-Los participantes al recibir el mensaje hacen el "commit", y regresan un mensaje al coordinador de satisfactorio.
-El coordinador hace el "commit " final o segundo, quedando la transacción completada.

Dado que cualquiera de los sitios participantes puede quedar fuera de servicio, el protocolo esta preparado para que se aborte la transacción, así mismo el coordinador puede quedar fuera de servicio, por lo que es preciso decidir si se hará el "commit" o se abortará cada una de las transacciones que estaba manejando el coordinador.

Para manejar el control de concurrencia en esquemas donde no se repite la información se pueden usar las técnicas expuestas en el tema de control de concurrencia (2.3) ya que cada sitio maneja las peticiones de candado exclusivo y compartido, pero estas técnicas no son aplicables cuando se tienen varias copias distribuídas en diferentes sitios. Para este tipo de esquemas existen otras técnicas tales como la del coodinador de candados.

El coordinador de candados funciona de la siguiente manera:
el DDBMS mantiene un coordinador que reside en un solo sitio, todas las solicitudes de colocación y liberación de candado se hacen en ese sitio. Si una transacción necesita poner candado a un dato, se enviará una solicitud de colocación de candado al coordinador, él otorgará o negará el candado según el candado que tenga el dato en ese momento, es decir si existe un candado exclusivo y se pide un candado exclusivo se negará así mismo si la petición es de candado compartido.Si se tiene un candado compartido y se pide candado compartido ó si el dato no tiene candado y se pide un candado exclusivo, se otorgará la petición. Una vez que se otorgue el permiso, el coordinador manda un mensaje a la estación que hizo el requerimiento informandole que se coloco el candado. La transacción puede leer el dato de cualquier sitio donde resida una copia y el optimizador autorize. Cuando es una petición de actualización se debe poner candado a todos los sitios donde reside una copia del dato.

Este método tiene las siguientes ventajas:
-Sencillez de implantar ->solo requiere de dos mensajes (colocación y liberación).
-sencillez de manejo de "deadlock" -> ya que es un solo sitio donde se aplican los candados por lo que se pueden aplicar las técnicas expuestas en el tema 2.3 de este capítulo.

Desventajas:
-Si el coordinador falla se perderá el control de concurrencia.
-Dado que las solicitudes candado son enviadas al coordinador, este se puede convertir en un cuello de botella.

3. EVALUACION COMPARATIVA DE LOS PRINCIPALES CLIENTE-SERVIDOR EXISTENTES EN EL MERCADO

Introducción.

Para hacer esta evaluación comparativa, se dividirá en apartados que consideramos importantes.

Los productos que se evaluarán son aquellos que consideramos desempeñarán un papel importante en el mercado Mexicano, Existen varios más que se distribuyen en Estados Unidos, pero éstos no entrarán en nuestro estudio.

Nosostros identificamos dieciseis puntos a evaluar en los diferentes productos:
- Consumo de recursos.
- Sistemas operativos soportados por el servidor.
-Sistemas operativos soportados en la estación de trabajo (CLIENTE)
-Protocolos soportados.
-Facilidad de instalación.
-Documentación adecuada.
-Soporte de SQL.
-Control de concurrencia.
-Mecanismos de recuperación.
-Mecanismos de seguridad.
-Rendimiento.
-Front -ends soportados.
-Lenguajes de alto nivel soportados.
-Manejo de APIs.
-Conectividad a Minis y Mainframes.
-Capacidad de distribución.

En seguida bosquejaremos cada uno de los puntos arriba mencionados para tratar de ser lo más objetivos posible al momento de aplicar dichos puntos de evaluación, asi mismo describiremos el ambiente de hardware y software que se utilizó en la evaluación de los productos.

Cada uno de los productos serán corridos sobre la red con sistema operativo NETWARE V.2.15 de NOVELL, La red consistirá de cinco estaciones de trabajo , un FILE SERVER, y un DATABASE SERVER.
Las pruebas fueron codificadas en MicroSoft C versión 5.1 usando las APIs de cada producto.

-Consumo de recursos.

El consumo de recursos es un punto donde los servidores de bases de datos varia. Además, es un punto importante ya que el rendimiento es función de esta caracteristica, por lo que es necesario saber cuánta memoria RAM y cuánto espacio en disco duro requiere como mínimo, así mismo el tipo de procesador requerido.

-Sistemas operativos de LANs soportados y sistemas operativos donde se instala en BACK-END (servidor de bases de datos).

En esta sección se describirán los sistemas operativos que soporta el servidor de datos. Este punto es importante para las empresas que cuentan con una o varias LANs instaladas, así mismo se describirá el tipo de sistema que utiliza el servidor para su instalación, ya que esto va de la mano con el sistema operativo de la LAN.

-sistemas operativos soportados por la estación de trabajo (CLIENTE).

En esta sección se dirá qué tipo de sistemas operativos (DOS,OS/2) permite el servidor para las estaciones de trabajo.

-Protocolos soportados.

Los protocolos de comunicación son una parte importante para la conectividad hacia otros ambientes por lo que se describirá qué tipo de protocolos soporta cada uno de los productos a evaluar.

-Facilidad de instalación.

La instalación es relativamente fácil en algunos casos de LANs, ya que en ciertos tipos para las que en un principio no fué desarrollado el DBMS es un poco más compleja.

En este punto se tomará en cuenta que el proceso de instalación sea lo más intuitivo posible, que el tiempo de instalación sea breve y los mensajes de error sean entendibles.

-Documentación adecuada.

Los Manuales deben tener un enfoque didáctico.

Un aspecto muy importante es que tenga casos prácticos (ejemplos) que ayuden al entendimiento del funcionamiento del producto. Dado que esta prueba requiere de varios usuarios y no se cuenta con los recursos humanos para realizarla, la metodología que se seguirá es la siguiente: se tomará información de revistas especializadas en el tema, tomando en cuenta la metodología siguiente:usuarios novatos y usuarios experimentados estudian los manuales y material adicional proporcionado por el producto, una vez estudiados realizan un programa en un lenguaje procedural , así mismo crean una base de datos.

-Soporte del SQL.

El producto debe soportar el SQL ANSI además proveer facilidades que mejoren al ANSI SQL.

-Control de concurrencia.

En esta sección describiremos el tipo de interbloqueo (candados) con el que cuenta el producto.

-Mecanismos de recuperación.

Todos los manejadores de bases de datos deben soportar recuperación automática de datos ya que pueden suceder catástrofes tales como : fallas en el hardware, fallas en el sistema o fallas en el programa.

En este punto se indicará con qué mecanismos de control cuenta el servidor evaluado.

-Mecanismos de seguridad.

Este es otro punto importante en la administración de la base de datos. En esta sección mencionaremos las facilidades que tiene cada uno de los servidores evaluados para el control de usuarios, así como los privilegios para accesar la información .

-Rendimiento

Este es factor importante en el proceso de evaluación. Dicha evaluación se realizará con programas escritos en lenguaje "C" y las APIs de cada producto (dichos programas se encuentran en el anexo del capítulo), para determinar el tiempo que tarda cada producto en resolver el requerimiento de algún usuario.

Las pruebas que desarrollaremos serán para comprobar la rapidez del optimizador para resolver peticiones. Los "QUERYs" serán ejecutados en un máquina Hewlett Packard Vectra RS/25c con 8MB. en RAM y disco duro De 150MBytes para el FILE SERVER, para el Database Server es la misma configuración que la del File Server en cuanto a Hardware. Las estaciones de trabajo serán máquinas Olivetti M290 con 4MB. en RAM y 20MB en disco duro.

Las tablas a las que haremos referencia en el transcurso de las pruebas contienen 10,000 registros con longitud aproximada de 60 Bytes de información, los registros son de longitud fija. La información fué extraida de un sistema de información de la Secretaría de Comercio y Fomento Industrial contenido en un Mainframe.

```
QUERY 1   SELECT * FROM MOV WHERE MOV_XVOLUM ='90801'
          AND FRA_FRACCI < '0302-6999'

QUERY 2   SELECT * FROM MOV WHERE MOV_XVOLUM = '90801'
          OR FRA_FRACCI = '0202-6999'

QUERY 3   SELECT * FROM MOV WHERE FRA_FRACCI > '0303-7999'
          ORDER BY MOV_XVOLUM.

QUERY 4   SELECT SUM(MOV_VALCOM) FROM MOV.

QUERY 5   SELECT MOV_XVOLUM FROM MOV GROUP BY MOV_XVOLUM HAVING
          COUNT(*) > 5

QUERY 6   SELECT MOV.FRA_FRACCI , FRA.FRA_DESFRA FROM MOV,FRA
          WHERE MOV.FRA_FRACCI =FRA.FRA_FRACCI AND
          MOV.MOV_XVOLUM LIKE '90%'
```

Fig.1

La figura 2 muestra la descripción de las tablas que crearemos en el servidor de base de datos, La Figura 1, muestra Los seis "QUERYs" que serán aplicados. Cabe hacer mención que la técnica que utilizaremos para la evaluación en cuanto a rendimiento es la de David Dewitt[2].

```
DESCRIPCION DE LA TABLA FRACCION
CREADA PARA LA PRUEBA DE RENDI-
MIENTO.

FRA_MOVI    VARCHAR(2)
FRA_FRACCI  VARCHAR(9)
FRA_DESFRA  VARCHAR(40)
FRA_UNIMED  VARCHAR(2)
```

```
DESCRIPCION DE LA TABLA MOVIMI
ENTOS CREADA PARA LA PRUEBA DE
RENDIMIENTO.
FRA_MOVI     VARCHAR(1)
FRA_FRACCI   VARCHAR(9)
PAI_PAIS     VARCHAR(2)
EMP_RNIE     VARCHAR(10)
MOV_VALCOM   DECIMAL(14)
MOV_XVOLUM   VARCHAR(11)
MOV_UMED     DECIMAL(1)
```

Fig.2

[2] El Benchmark DeWitt fue desarrollado por David DeWitt y otros de la Universidad de wisconsin en 1983. Su articulo " Benchmarking Database system: A Systematic Approach" define una metodología para probar bases de datos relacionales. el Benchmark DeWitt fue diseñado para medir pequeños sistemas de bases de datos (de 5,000 a 10,000 renglones), y es usado para aplicaciones de solo lectura.

-Lenguajes de alto nivel soportados.

En esta sección se indicará qué tipo de lenguajes de alto nivel soporta como front - end el servidor de bases de datos evaluado.

-Manejo de APIs.

En esta sección se mencionará que tipo de interface tiene el producto para los lenguajes de alto nivel, ya que puede ser un precompilador o una interface directa.

-Conectividad a Minis y Mainframe.

Es importante que el servidor evaluado posea facilidad de comunicación con otros ambientes a un nivel donde dicha comunicación sea transparente para el usuario . Este concepto es nuevo pero puede ser importante ya que se tendrá más posibilidad que en algún momento dicho servidor pueda establecer comunicación con una máquina grande (Mini o Mainframe).

-Capacidad de distribución.

Para Nuestro estudio entenderemos por capacidad de distribución a la facilidad de manejar varios servidores de bases de datos en una sola red, pudiendo distribuir la información así como el control de esta entre dichos servidores.

3.1 ORACLE SERVER (versión1.0)

Oracle Corporation, uno de los vendedores más grandes de DBMSs, actualmente distribuye Oracle Server basado en la arquitectura cliente-servidor, con el lenguaje de consultas SQL., y el modelo relacional para el manejo de información. Oracle Server fué desarrollado por DECVAX y el sistema UNIX.

consumo de recursos.

Oracle Server Corre bajo OS/2 , los requerimientos en cuanto a Hardware son:

Database server.
-Una pc AT (procesador 80286 ó 80386)
-30 MB. de espacio mínimo en disco duro.
-Memoria RAM óMB.

Estación de trabajo.
-Una pc AT (procesador 80286 o 80386) para poder ejecutar cualquier productos de Oracle.
-640 KB. de memoria base y 896 de memoria extendida para estaciones con sistema operativo DOS.
-Para estaciones OS/2 se requiere memoria RAM según la versión de OS/2.

Protocolos soportados.

Oracle Server puede correr sobre los tres protocolos, ipx/spx , Named -Pipes, y Netbios. Por los que puede correr en redes tales como Novell Netware que utiliza el protocolo IPX/SPX y sobre Lan Manager que utiliza el protocolo Named Pipes .

Sistemas operativos soportados en la estación de trabajo.
Los sistemas operativos que soporta la estación de trabajo que utiliza como back-end a Oracle Server son DOS y OS/2.

Facilidad de instalación

El proceso de instalación de Oracle Server bajo Novell Netware es un poco complejo ya que, como funciona bajo OS/2 se tiene que instalar un software adicional llamado "requestor" para que pueda haber comunicación de la máquina que funcionará como database server (back-end) al FILE SERVER . Una vez que se ha instalado el "requestor" se procede a instalar Oracle Server, la instalación de Oracle Server no es muy intuitiva ya que al estar instalando, los mensajes son muy austeros y no presenta ayuda en línea, así mismo el tiempo de instalación es un poco más tardado en comparación con los otros productos.

A continuación se da el procedimiento para instalar Oracle Server sobre Novell Netware ver. 2.15.

1.- Se instala el sistema operativo Os/2 Ver. 1.1 + sobre la máquina que funcionará como back-end.
2.- Una vez instalado el OS/2 versión 1.1 sobre el servidor se instala el OS/2 Requestor de Novell Netware.
3.- Después de probar que existe comunicación de la estación OS/2 que se utilizará como back-end con el FILE SERVER se procede a instalar el Oracle Server tecleando el siguiente comando [c:] a:orainst.

4.- Una vez instalado el Oracle Server, editar el archivo net.cfg e introducir las siguientes dos lineas
PROTOCOL STACK SPX
SESSION #.

6:- Para arrancar el oracle server teclee los siguientes comandos.
>sqldba startup <authorization= nombre>
>spxsrv.
nota: spxsrv se utiliza cuando se usa el protocolo IPX/SPX.

instalación de Oracle sobre la estación de trabajo.
1.- c:>a:orainst.
2.- Insertar los discos que pide.
3.-Una vez que termina de instalar la interface al back-end de Oracle Server pedirá que se inserte el producto que se desea instalar.
4.- insertar el producto que se desea instalar (Sqlplus, Sqlforms, Sqlmenu, Sqlreport writer, etc).

Una vez instalado el Oracle server en la estación de trabajo (cliente), se procede a comunicarse con el database server de la siguiente manera.
>ipx.
> netx ;x= 2,3,4.
>sqlpme
>sqlspx.

Soporte de SQL

Oracle Server soporta el SQL lo que permite comunicarse con bases de datos de mainframe tales como DB2 de IBM.

Oracle Server además de tener el SQL estandar tiene extensiones por lo que los usuarios pueden escribir preguntas complejas con menos comandos.

Oracle Server también tiene los comandos MINUS e INTERSECCION que son iguales a los operadores del álgebra relacional diferencia e intersección. Evitando la necesidad de subpreguntas en estas operaciones. Los trigger de Oracle Server son incluídos en una aplicación o ligados para un campo de una entrada de datos o forma de actualización (Sqlforms).

Oracle server contiene un diccionario de datos el cual contiene datos acerca de los datos (metadatos) entre las tablas que se encuentran en el diccionario de datos estan : un catálogo que contiene el nombre de las tablas y vistas definidas en la base de datos, un catálogo que describe los campos de las tablas definidas, un catálogo que describe los índices (tipo único o normal) , un catálogo que describe las vistas que existen en la base de datos, un catálogo de los sinónimos, tres catálogos de autorizaciones (usuarios, columnas y tablas).

Oracle Server asigna información estadística en el diccionario de datos, esta información es utilizada por el optimizador para hacer más rápida la consulta y la actualización de datos.

Control de Concurrencia.

Oracle Server permite el automático bloqueo por tabla, pero esto degrada el rendimiento ya que la tabla se desbloquea hasta que la transacción termina y sólo permite una transacción.

El modo compartido de Oracle Server coloca un candado compartido sobre una tabla , que permite a los usuarios leer datos concurrentemente aunque algunos usuarios esten leyendo los mismos datos. Los programadores que usan el modo exclusivo tienen el control total de la tabla. El modo compartido del update se activa cuando se usa un comando SELECT con una cláusula FOR UPDATE.

Oracle Server posiciona un candado a nivel bloque sobre todas las páginas recuperadas por el comando SELECT del SQL . Cuando los registros están siendo actualizados Oracle Server coloca un candado exclusivo sobre la tabla, que no es borrado sino hasta que la transacción termina.

Oracle Server contiene un modo especial de lectura llamado read consistency el cual permite a los usuarios leer cualquier registro bloqueado o no, por lo que provee un alto nivel de concurrencia.

Oracle Server tiene una detección automática de DEADLOCK y aleatoriamente finaliza una transacción.

Mecanismos de recuperación.

Para la recuperción , oracle server utiliza el comando AFTER IMAGE JOURNAL (AIJ) , el cual es posible si se suma el parámetro AFTER -IMAGE en el archivo init.ora para que al inicializar la base se inicialice este archivo. Este archivo es usado para restaurar el proceso perdido en la falla en la estación de trabajo.

El ROLLFORWARD se utiliza cuando se cae el sistema y se quiere regresar a la base a su estado anterior.

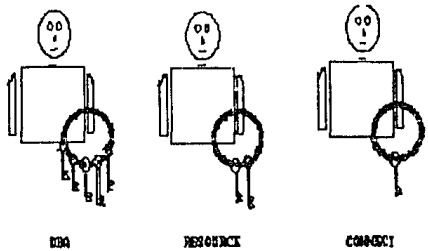
Una transacción en Oracle Comienza cuando encuentra un comando ejecutable del SQL. Un comando ejecutable del SQL es el que generalmente llama a la base de datos, e incluye comandos del DML, DDL, o DCL.

Una Transacción termina cuando uno de los siguientes puntos ocurre:

-Un COMMIT o ROLLBACK es invocado.
-Con el uso de un comando del DDL (CREATE, DROP, RENAME, ALTER) hace que la transacción presente sea cerrada, así mismo al término del comando del DDL la transacción es cerrada automáticamente.
-Una terminación anormal de un proceso regresa a la base de datos a su estado anterior (ROLLBACK).
-Terminación del programa.
-desconección del front-end que se esté utilizando (hoja electrónica, dbms, etc).

Mecanismos de seguridad.

Oracle Server permite tres tipos de usuarios para la base de datos los cuales son: CONNECT, RESOURCE, Y DBA. El usuario DBA tiene todos los privilegios que otorga Oracle Server, el usuario RESOURCE tiene más privilegios en comparación con el CONNECT .



CONNECT .- Permite a los usuarios consultar tablas.
RESOURCE.-Permite a los usuarios crear tablas, consultar y actualizar.
DBA .-Ejecuta comandos de administración alta,baja de usuarios , etc.

Además Oracle Server provee GRANT Y REVOKE para privilegios de tablas.

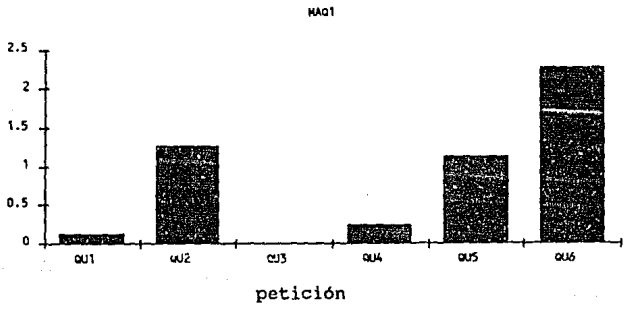
Rendimiento.

A continuación mostramos los resultados de los programas aplicados para medir el rendimiento del optimizador, es decir la velocidad de respuesta a los requerimientos de varios usuarios, así mismo las gráficas

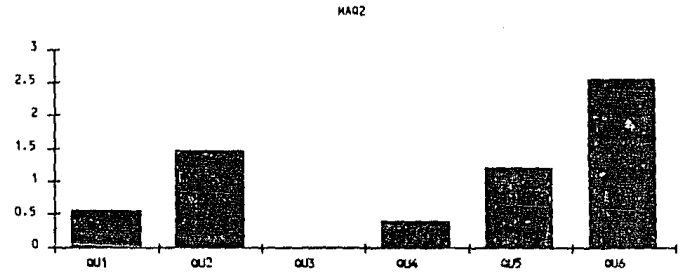
(los programas fuentes se encuentra en el anexo al final del capítulo).

ORACLE					
	MAQ1	MAQ2	MAQ3	MAQ4	MAQ5
QUERY 1	.12 MIN.	.56 MIN	.87 MIN.	.15 MIN.	.87 MIN.
QUERY 2	1.26 MIN.	1.46MIN	.52 MIN	1.22 MIN.	1.24 MIN.
QUERY 3	ERROR	ERROR	ERROR	ERROR	ERROR.
QUERY 4	.23 MIN.	.39 MIN	.39 MIN	.27 MIN	.43 MIN.
QUERY 5	1.13 MIN	1.22MIN	1.37MIN	1.03 MIN	1.01 MIN.
QUERY 6	2.27 MIN	2.58MIN	3.01MIN	2.06MIN	2.02MIN.

TABLA DE RESULTADOS .

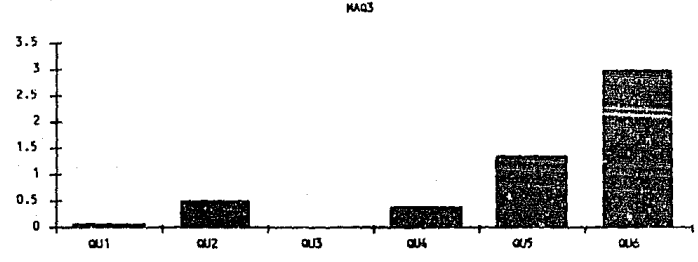


minutos



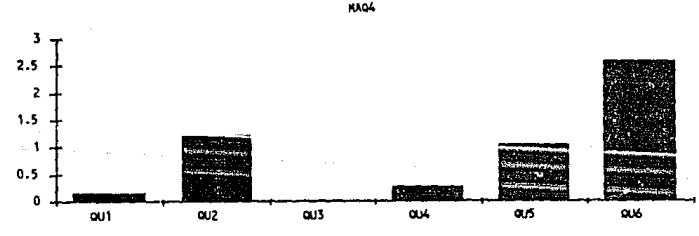
petición

minutos

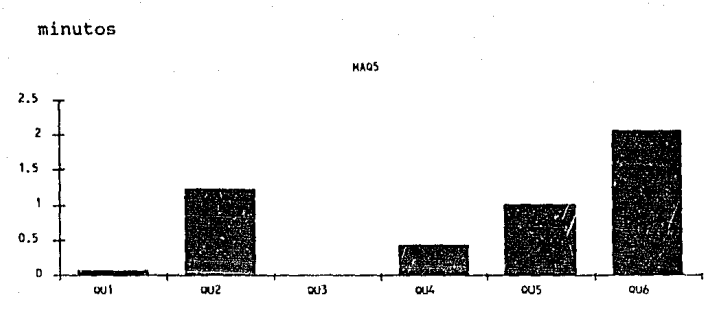


petición

minutos



petición



Front-ends soportados, uso de APIs y lenguajes de alto nivel.

Oracle Server posee un conjunto de herramientas desarrolladas por Oracle Corporation para el desarrollo de aplicaciones en SQL pudiéndose así desarrollar sofisticadas y amigables aplicaciones en ambiente LAN , Sqlforms, Sqlplus, Sqlmenu,Sqlcalc, Sqlreport Writer , Oracle Para lotus , son algunas de las herramientas que ofrece Oracle. Asimismo permite tener como FRONT-END a las aplicaciones hechas en lenguajes tales como: C , Cobol , Fortran , Pascal , Ada, Pl/ I interactuando con sus APIs.

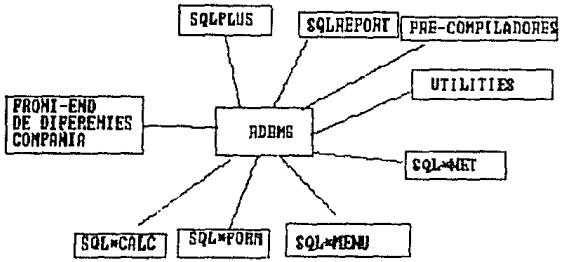
Para interactuar con el database server mediante los lenguajes anteriormente mencionados , Oracle Server utiliza un precompilador el cual hace más rápida la programación y fácil, pero el código ya precompilado es muy grande.

Muchas compañías desarrolladoras de software están liberando FRONT-ENDs para que trabajen con Oracle Server tales compañías se mencionan a continuación.

COMPAÑIA	NOMBRE DEL PRODUCTO.
Aspray	Aspray Business Software.
Borland International	Paradox DBMS.
Clarion	Clarion Developer.
Conceptual Software	DBMS copy & DBMS Stat.
Dataflex	Entry Point Software.
MegaHouse	Forms System.
Microsoft Corp.	Ms Excel Spreadsheet.
Neuron Data	Nexpert Object al Tool.
Picture Ware	Picture Power.
Pioneer	Q&E.
Progress	Progress Software.
Revelation Tech.	Advanced Revelation DBMS.

Software Group	Enable-Wp, Spreadsheet, DBMS.
Wordtech Systems	Quicksilver- Dbase Compiler.
Xy Quest	Xy Index-DBMS.
Gupta Technologies	Sql Windows.

CUERPO DE ORACLE SERVER



Oracle Server soporta los siguientes tipos de datos:

VARCHAR	255 caracteres.
NUMBER	9.99×10^{124}
DATE	dd-mon-aa.
LONG	65,535 caracteres.
RAW	Orientado a byte similar a varchar y long.

Conectividad a Minis y Mainframes.

Oracle Server tiene la habilidad de conectarce a Minis y Mainframes ya que aproximadamente en noventa sistemas operativos tiene un Oracle tales como Oracle para MVs (de IBM 3090), Oracle para Wang , Oracle para Unix, etc. Mediante la utileria de Oracle Sql*Conect que funciona como gateway se pueden accesar datos de diferentes plataformas.

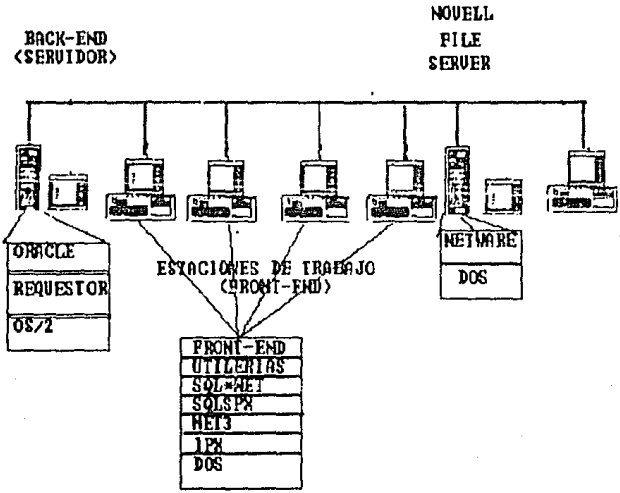
Con respecto a la migración el proveedor dice que es muy fácil ya que Oracle opera igual en todas sus plataformas, por lo que si un producto tal como el Oracle Server ya no es suficiente para el manejo de datos se puede migrar a un equipo mini o mainframe (que contenga a Oracle) sin tener que modificar las aplicaciones.

Capacidad de distribución.

Oracle Server permite tener varios servidores de datos en la misma red , pero el desarrollador tiene que especificar el server al que se quiere conectar y no se

pueden hacer actualizaciones que involucren a dos servidores , lo que quiere decir que el acceso a los datos no es transparente.

Sqlplus .- Es un ambiente de manejo de datos en linea para el RDBMS de Oracle Server , los usuarios pueden accesar los datos de la base via el Sqlplus, lo que quiere decir que el usuario debe estar familiarizado con el lenguaje SQL ; A través del Sqlplus el usuario con privilegios de DBA puede borrar, crear y alterar tablas, asimismo puede realizar consultas no programadas. También los usuarios con diferentes privilegios pueden hacer uso de esta herramienta básica.



ARQUITECTURA BAJO LA QUE SE REALIZARON LAS PRUEBAS DE ORACLE SERVER.

3.2 SQLBASE (versión 3.2.1).

Sqlbase es desarrollado por Gupta Technologies, Gupta Technologies es un pionero en los servidores de bases de datos que utiliza el concepto Cliente-Servidor. El servidor de bases de datos ofrecido por Gupta Technologies corre bajo OS/2 ó D.O.S.

Consumo de Recursos.

Requerimientos para el BACK-END(database server):
-Una pc AT (procesador 80286, 80386).
-10MB de disco duro.
-640KB de Ram mínimo para DOS. Para máquinas con sistema operativo OS/2 según lo que requiera la versión de OS/2.
-Versión para DOS 3.1 o más alta y para OS/2 versión 1.1 o más alta.

Requerimientos para la estación de trabajo:
-Una pc compatible .
-256 KB para DOS mínimo.
-Para Oc/2 RAM según lo que requiera la versión.

Sqlbase provee software que puede correr sobre memoria base y memoria extendida, este software se corre dependiendo de la configuración del equipo (hardware)

Para poder correr Sqlbase sobre una red, se necesitan dos PCs compatibles con IBM y que trabajen como nodos de la red. La PC servidor (BACK-END) es dedicada, es decir se utilizará solo como servidor de datos.

Protocolos soportados.

Sqlbase puede correr sólo con el protocolo netbios, por lo que puede correr sobre LAN MANAGER Y NOVELL NETWARE (y todas las LANs que soporten netbios).

Facilidad de instalación.

La instalación de Sqlbase bajo Novell Netware es relativamente sencilla ya que solo se copian los programas a un subdirectorio llamado Sqlbase, el tiempo de instalación es corto y no ofrece grandes problemas.

El procedimiento de instalación de Sqlbase en el BACK-END se describe a continuación.

1.- Crear un subdirectorio bajo la raiz del disco duro llamado Sqlbase dentro de él crear un subdirectorio llamado BIN.
2.- Copiar los archivos .EXE a este subdirectorio (BIN).
3.- copiar los archivos con extensión: CFG, DBS , SQL , BAT a l subdirectorio Sqlbase.
4.- Inicializar la base con el comando INIT seguido del nombre de la base de datos.

Instalación de Sqlbase sobre la estación de trabajo (CLIENTE).

1.- Generar un Subdirectorio llamado sqlbase
2.- Copiar los archivos netbios , dbxroutr, y dbrouter.
3.- Copiar el front-End que se desea utilizar para interactuar con el servidor de datos (Sqltalk, SqlWindows, etc.).

Como se ve Sqlbase es simple para instalar bajo Novell Netware ; En el archivo de configuración dbxservr.cfg ó dbxserver.cfg se especifica: las bases de datos que manejará Sqlbase sobre la red , su localización, máximo número de estaciones de trabajo, longitud del buffer, y el número de transacciones que el servidor permitirá a la vez.

Para inicializar el servidor se teclea la instrucción dbxservr o dbserver según el tipo de memoria, una vez instalada la base de datos la pantalla de estado despliega la actividad de la base de datos y procesos activos.

Soporte de SQL.

Sqlbase soporta el SQL ANSI y además tiene Extensiones tales como manejo de string, substring y de búsqueda. Un ejemplo de este tipo de extensiones es:

```
SELECT   .. substr (FRA_FRACCI,0,4) FROM FRA.
```

Sqlbase no contiene alguna instrucción parecida a los triggers de Oracle, pero este tipo de procedimientos para la integridad referencial de la base de datos se puede implementar al tiempo que se desarrolla una aplicación en el lenguaje SAL (Sql Application Languaje). Este lenguaje contiene instrucciones tales como if , else , loop.

SqlWindows trabaja bajo un ambiente gráfico, por lo que para poder correr SqlWindows debe estar corriendo WIndows de Microsoft.

Sqlbase genera un diccionario de datos al correr la instrucción INIT en el prompt del sistema operativo, el diccionario de datos contiene tablas que son utilizadas para asignar información acerca de las tablas , índices, vistas , columnas e información de usuarios.

```
SYSCOLUMNS .-Describe las columnas de cada tabla de la base
            de datos.
SYSINDEXES .-Nombre de los índices de las tablas.
SYSVIEW    .-Las vistas que existen en la base de datos,
            nombres y campos.
SYSSINONYMS .-sinónimos de las tablas y vistas.
SYSKEYS     .-Las columnas de los índices.
SYSUSERAUTH .-Cada usuario y su tipo de autorización en las
             tablas.
SYSTABAUTH .-Cada usuario y sus privilegios para cada tabla.
SYSCOLAUTH .-Cada usuario y las columnas que puede accesar.
```

Control de concurrencia.

Sqlbase soporta tres niveles de aislamiento en transacciones, repeatable read , cursor stability y read consistency.

Repeatable read.- Bloquea las páginas recuperadas o actualizadas con un comando SQL y mantiene el bloqueo hasta que es cerrada (commit) la transacción.

cursor stability.- Bloquea un solo registro a la vez dependiendo sobre que página el programa esta leyendo. El modo cursor stability permite a dos o más programas leer el mismo conjunto (uno ó más) de registros, pero previene a los programas de actualizar simultáneamente el mismo registro.

Cursor stability es usado cuando un programa lee varios renglones y trabaja sólo con uno. Permitiendo mas alta concurrencia pero menos consistencia en comparación con el metodo repeatable read.

El método read consistency permite a los usuarios leer cualquier registro bloqueado o no.

Sqlbase tiene detección automática de deadlock y termina una de las dos transacciones arbitrariamente.

Mecanismos de recuperación.

Sqlbase para la recuperación hacia atrás tiene un archivo llamado BEFORE -IMAGE, este archivo contiene una extensión .BI , el archivo before-image crea una imagen de los datos antes de ser cambiados hasta que ocurre alguno de los siguientes casos:

-El usuario usa el comando commit .
-Se inicia otro comando SQL.
-Termina el Programa de aplicación.

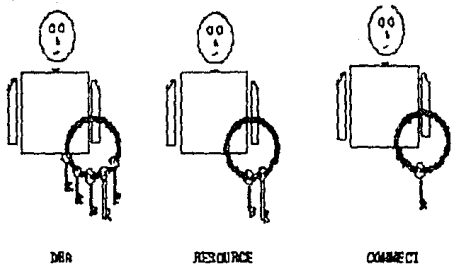
Sqlbase utiliza para la recuperación hacia adelante un archivo con la extensión ".JOR" definido por el usuario. Este archivo contiene todos los comandos SQL que cambiaron la base de datos. Este archivo puede ser utilizado después que haya ocurrido una falla en el sistema.

Mecanismos de seguridad.

Sqlbase implementa la seguridad a nivel usuario protegiendo las tablas, vistas y registros.

Sqlbase define tres tipos de usuarios para la base de datos los cuales son:

CONNECT, RESOURCE, Y DBA. El usuario DBA tiene todos los privilegios que otorga Sqlbase, el usuario RESOURCE tiene más privilegios en comparación con el CONNECT .



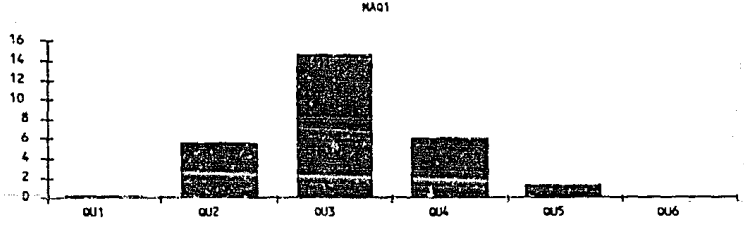
CONNECT .-Permite a los usuarios consultar tablas.
RESOURCE .-Permite a los usuarios crear tablas, consultar y actualizar.
DBA .-Ejecuta comandos de administración alta,baja de usuarios,etc.

Rendimiento.

Al igual que en Oracle Server , a continuación se muestran los resultados de aplicar los programas hechos en lenguaje "C" utilizando las APIs de Sqlbase para medir la velocidad de respuesta a su optimizador.

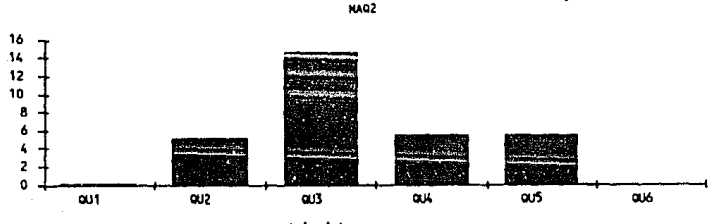
SQLBASE					
	MAQ1	MAQ2	MAQ3	MAQ4	MAQ5
QUERY 1	.11 MIN.	.13MIN	.13MIN	.11MIN	.11MIN
QUERY 2	5.57MIN	5.18MIN	5.59MIN	5.10MIN	5.57MIN
QUERY 3	14.57MIN	14.59MIN	14.01M.	14.15MIN	13.56MIN
QUERY 4	6.08MIN	5.51MIN	3.99MIN	5.51MIN	11.99MIN
QUERY 5	1.36MIN	5.51MIN	1.38MIN	.57MIN	1.03MIN
QUERY 6	ERROR	ERROR	ERROR	ERROR	ERROR

minutos

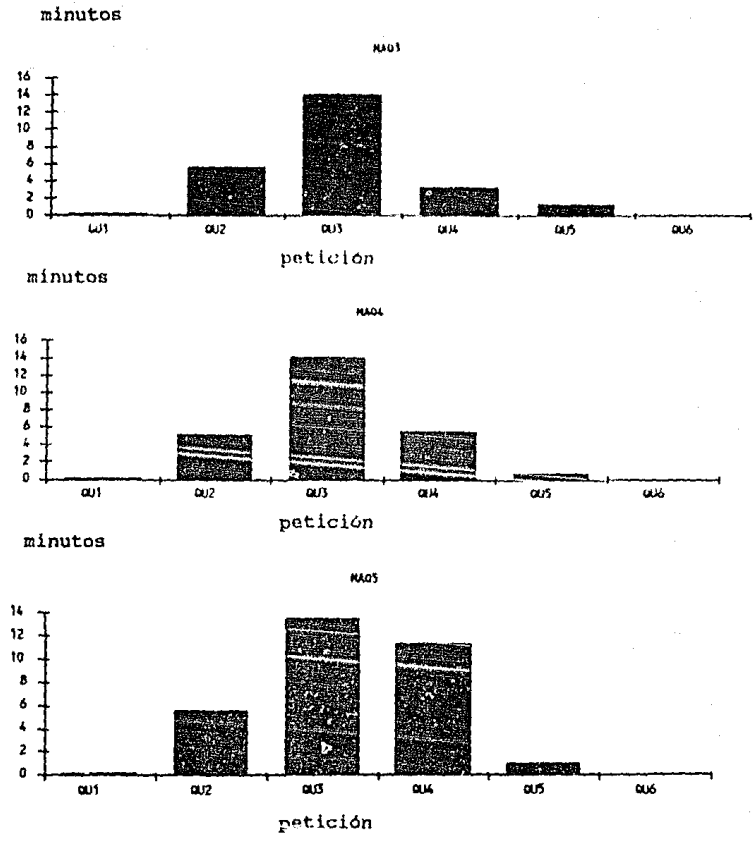


petición

minutos



petición



Front-ends soportados, uso de APIs y lenguajes de alto nivel.

Sqlbase provee una serie de FRONT-ENDs gráficos que corren bajo windows, tal es el caso de sqltalk windows y SqlWindows. Asimismo provee la herramienta Sqltalk para poder trabajar en línea. Mediante Sqltalk se pueden ejecutar comandos del SQL.

Sqltalk/Windows .- trabaja sobre un ambiente de ventaneo como trabaja windows de Microsoft, Sqltalk/windows permite escribir comandos SQL que son ejecutados y los resultados son mostrados en una pantalla tipo scroll, asimismo permite salvar el query, cargar uno ya existente, e imprimirlo. Además contiene comandos de control de sesión tales como conectarse , desconectarse, instalar, desinstalar una base de datos , herramientas para copiar tablas de una base a otra. Todas estas opciones explicadas arriba son realizadas a través de menues utilizando el teclado o el mouse.

Sqlbase provee una herramienta de cuarta generación llamada Express Windows.

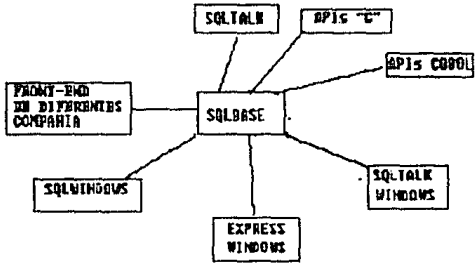
Express Windows está compuesto por express form y express tables, los cuales generan aplicaciones sobre una sola tabla existente en la base de datos ; Sólo se tiene que especificar el nombre de la tabla y los campos que contendrá la aplicación, se salva y se compila, obteniendo una aplicación en la cual se pueden hacer recuperaciones y actualizaciónes.

Sqlbase ofrece una interface con los lenguajes de programación C y Cobol mediante sus APIs , asimismo cuenta con su propio lenguaje de programación que es llamado SAL.

Para poder interactuar con el BACK-END (database server) de Sqlbase mediente los lenguajes "C" y Cobol se utilizan sus APIs las cuales son comandos directos en el programa, esto es, no utiliza un precompilador como lo hace Oracle Server , haciendo más difícil la programación .

Sqlbase tiene la posibilidad de interactuar con FRONT-ENDs de diferentes compañías tales como:

COMPAÑIA	DESCRIPCION DEL PRODUCTO.
Jyacc	Precompiler for IBM Mainframe.
Lotus Development	Lotus DBMS.
Must Software	Nodad Mainframe Front-end.
PaperBack Software	Up Expert and Planner Spreadsheet.
Planet Software	Clipper Library.
Revelation Tech.	Advanced Revelation DBMS.
Wordtech System	QuickSilver-Dbase Compiler.
Zanthe	Zim Mainframe 4GL.



CUERPO DE SQLBASE.

Sqlbase soporta los siguientes tipos de datos.

VARCHAR	.-254 bytes.
LONG VARCHAR	.- 254 caracteres y no puede entrar en comparaciones.
NUMBER	.-22 digitos de presicion.
DECIMAL	.-(D,P) ; D=digitos ,P=precisión.
INTEGER	.-sin digitos de precisión (5 digitos).
SMALLINT	.- " " " " (2 digitos).
REAL	.-22 digitos de precisión.

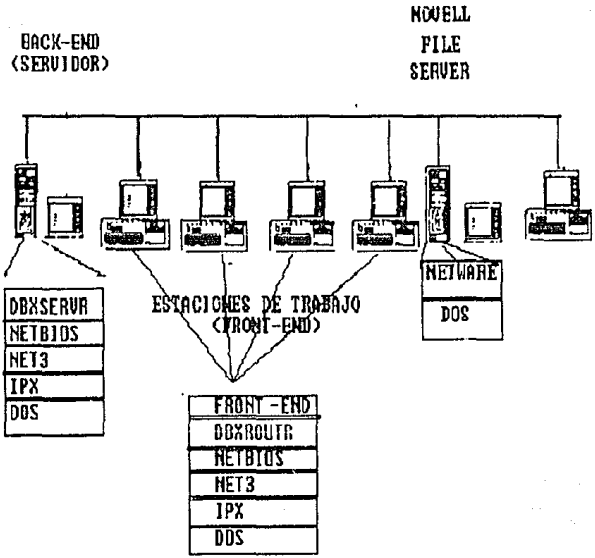
DOUBLE PRECISION.-Especifica que tiene punto flotante.

Conectividad a Minis y Mainframes.

Sqlbase sólo tiene la posibilidad de conectarse a DB2 de IBM, con la utilería SQL NETWORK.

Capacidad de distribución.

Sqlbase con la utilería SQLNET puede conectar dos o varios servidores de bases de datos que trabajen bajo una red. Los desarrolladores tienen que hacer referencia al lugar físico donde se encuentra el servidor de datos. Por lo que se puede decir que Sqlbase no soporta actualización de bases de datos distribuidas . Los programadores sólo pueden actualizar una base de datos en las transacciónes.



ARQUITECTURA BAJO LA CUAL SE REALIZARON LAS PRUEBAS DE SQLBASE

3.3 NETWARE SQL (versión 2.01b)

Netware Sql fué desarrollado por Novell , además es uno de los primeros en implementar el servidor de bases de datos (BACK-END) en el file server, es decir se utiliza la misma máquina para tener el FILE y el DATA SERVER.

Netware Sql corre bajo el sistema operativo de red Novell Netware versión. 2.1 o más alto.

Consumo de recursos.

Para poder instalar Netware Sql se debe contar con los siguientes recursos:

-Novell Netware ver 2.1 o más
-Btrieve ver. 5.0 o más .
-Tener instalado el TTS (Transaction Tracking System) para garantizar la integridad de datos.
-Mínimo 2 MB de memoria RAM.
-Espacio en disco duro 650 KB.

Los Requerimientos para la estación de trabajo son mínimos ya que sólo debe tener memoria disponible para cargar el sistema operativo , el shell de Netware y el SQL Requestor, La aplicación o el FRONT-END es la que consume más memoria tal es el caso de XQL. La estación de trabajo puede ser DOS o OS/2.

Protocolos soportados.

Netware Sql puede ser accesado vía el protocolo nativo de Novell Netware IPX/SPX. Ningún otro más es permitido.

Facilidad de instalación.

La instalación de Netware Sql es realmente fácil ya que consiste de un solo disco de 1.2 MB, así mismo el tiempo de instalación es muy reducido y demasiado intuitivo.

Un simple programa setup instala Netware Sql . El setup se encarga de colocar un archivo llamado NW$SQL.VPO dentro del directorio system de Novell Netware.

Antes de instalar Netware Sql debe estar cargado Btrieve para red, es decir deben de existir los VAPs[3], BROUTER.VAP y BSERVER.VAP en el subdirectorio system de Novell Netware. De tal forma que el Netware Sql es cargado junto con Btrieve para red al momento de cargar el sistema operativo de Novell ver. 2.15 , en la versión 386 no porque se carga como un NLM[4].

3 VAP (Value-Added Process) - Son procesos que residen en el file server y usan el protocolo de Netware para accesar los recursos.

4 NLM (Netware Load Module)- modulo cargable y descargable del sistema operativo.

Entre los parámetros para configurar Netware Sql estan: Longitud máxima de mensaje, Máximo tamaño de Buffer, y número de sesiones activas.

Soporte de SQL.

Netware Sql soporta el ANSI estandar en su primer nivel , pero soportará en versiones venideras el segundo nivel del ANSI el cual contempla la implementación de triggers para la integridad referencial . Por el momento no incluye triggers.

Control de Concurrencia.

Netware Sql tiene un bloqueo automático a nivel tabla, al menos que los programadores usen explicitamente bloqueo por registro. En Netware Sql existen dos comandos para manejar el bloqueo en registros los cuales son: WAIT y NOWAIT.

El comando WAIT suspende el programa hasta que el registro requerido este libre.

El comando NOWAIT regresa el control al programa cuando trata de recuperar registros bloqueados., entonces se tiene que volver a ejecutar el comando SQL y/o mandar un mensaje al operador del programa.

Mecanismos de recuperación.

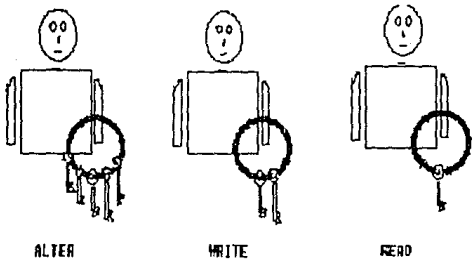
Netware Sql no tiene un manejo de recuperaciones transparente como Oracle Server y Sqlbase con sus archivos AFTER IMAGE y BEFORE IMAGE. Para poder controlar la integridad de datos y el manejo de transacciones usa el TTS (Transaction Tracking System) de Novell el cual usa tecnicas especiales de entrada y salida ("I/O") .

Mecanismo de seguridad.

Netware Sql tiene buenas facilidades para la administración de usuarios que comparadas con otros productos es mejor . Netware Sql tiene la capacidad de administrar por grupo o a nivel usuario.

Netware Sql soporta los comandos SQL estandar: GRANT y REVOKE para privilegios de tablas solamente. Los tres privilegios existentes en Netware Sql son: Read, Write, y Alter.

WRITE.-Abarca los comandos INSERT, UPDATE, y DELETE. Por lo que permite modificar el contenido de la base de datos.
ALTER.-Permite modificar la estructura de la base de datos.
READ .-Permite solo leer el contenido de la base de datos.

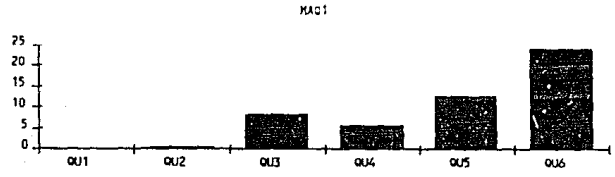


Rendimiento.

A continuación se muestran los resultados arrojados en las pruebas aplicadas a la velocidad de respuesta a las peticiones aplicadas.

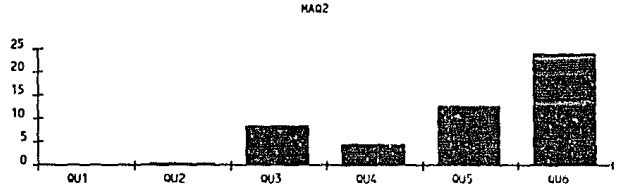
NETWARE SQL					
	MAQ1	MAQ2	MAQ3	MAQ4	MAQ5
QUERY 1	.01 MIN	.01 MIN	.01 MIN	.01 MIN	.01 MIN
QUERY 2	.13 MIN	.14 MIN	.11MIN	.11 MIN	.13 MIN
QUERY 3	8.29 MIN	8.25MIN	8.26MIN	9.54MIN	8.25MIN
QUERY 4	5.59MIN	4.58MIN	4.55MIN	4.55MIN	4.01MIN
QUERY 5	12.54MIN	12.53MIN	12.45MIN	12.45MIN	12.54MIN
QUERY 6	24.01	24.00	23.54	23.54	24.03

minutos



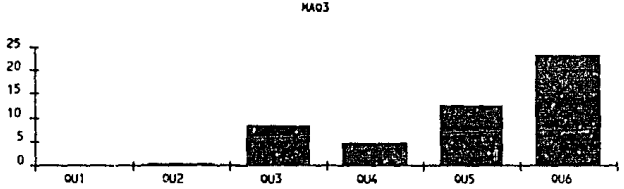
petición

minutos



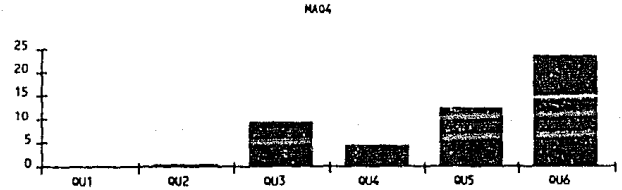
petición

minutos



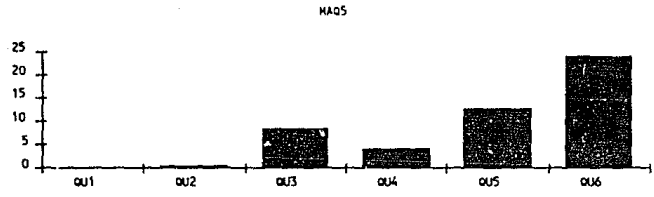
petición

minutos



petición

minutos

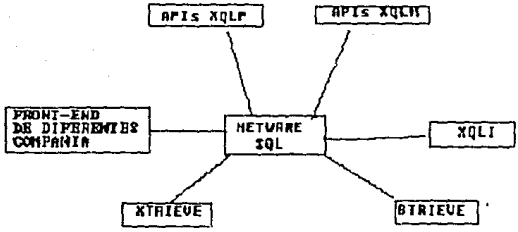


petición

Front-ends soportados,uso de APIs y lenguajes de alto nivel.

Al igual que Oracle Server y Sqlbase, Netware Sql permite trabajar con diferentes FRONT-ENDs desarrollados por diferentes compañias, además de los propios. Tales FRONT-ENDs son:
XQL, Xtrieve, Btrieve.

COMPAÑIA	DESCRIPCION DEL PRODUCTO.
Word Tech System	DBXL.
Word Tech System	Quick Silver.
Gupta Technologies	SqlWindows.
Borland	Paradox.
Zanthe Information	Zim.
Revelation Technologies	Advanced Revelation 4GL.
ABM	Platinum Accounting Package.
Lotus	Lotus 123.
Alfha Software	Alpha Four Spreadsheet.



CUERPO DE NETWARE SQL.

Los programadores pueden interactuar con el servidor de bases de datos mediante los lenguajes de programación C. Pascal, Cobol , y Basic. Las APIs usadas son comandos escritos directamente en el programa o sea no tiene un precompilador , consecuentemente la programación se dificulta más.

XQLP.- Es un nivel bajo para poder comunicarse con Netware Sql. Consiste de cuarenta comandos, cada uno de ellos puede ser colocado en programas escritos en algún lenguaje de programación tal como "C".

XQLM.- Sirve como una liga de alto nivel entre la aplicación desarrollada en algún lenguaje de alto nivel y Netware Sql. Consiste de un grupo de funciones que permite que la aplicación utilice comandos del SQL directamente.

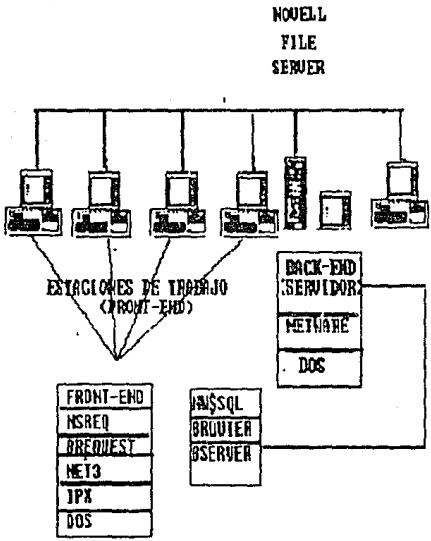
XQLI.- El XQLI acepta comandos del SQL en linea. Para poder usar el XQLI, se deben tener cargados en memoria los programas XQLP y XQLM, ya que los comandos introducidos en XQLI son trasformados a XQLM, posteriormente a XQLP y después son transformados a Btrieve ya que este es el que maneja todos los requerimientos.

Conectividad a Minis y Mainframes.

Como Netware Sql corre como un VAP bajo el sistema operativo de Novell Netware no puede comunicarse con máquinas grandes o minis solo atraves de gateway's.

Capacidad de distribución.

Netware Sql el cual trabaja como VAP puede soportar tantos servidores de bases de datos como tantos file server existan en la red, pero no soporta operaciones distribuidas.



ARQUITECTURA BAJO LA CUAL SE REALIZARON LAS PRUEBAS DE NETWARE SQL.

3.4 SQL SERVER (versión1.0)

Sql Server es un RDBMS desarrollado con tecnología de Sybase (tecnología de bases de datos para minicomputadora) y tecnología de Microsoft, y comercializado por Asthon Tate, cuenta con la arquitectura cliente-servidor y con el lenguaje de consultas SQL.

Consumo de recursos.

Sql Server corre bajo OS/2. Los requerimientos en cuanto a hardware son:

servidor de bases de datos
- Una PC AT compatible con IBM con procesador 80286 ó 80386.
- 6 MB de memoria RAM mínimo (OS/2 más Sql Server).
- MS OS/2 1.0 o más alta.
- 30 MB en disco duro.

Estación de trabajo.

- PC compatible con IBM.
- 640 KB de RAM más la Memoria adicional que ocupe la version de OS/2.
- Requestor (para la LAN de Novell).
- 5 MB. de espacio en disco duro.

Protocolos soportados.

Sql Server soporta el protocolo Named Pipes, por lo que puede correr bajo LAN MANAGER Y NOVELL NETWARE con su requestor, que carga en un nivel más arriba del protocolo ipx/spx al Named Pipes.

Sistemas operativos soportados en la estación de trabajo.

Las estaciones de trabajo que utilizan como BACK-END (servidor de bases de datos) a Sql Server y que estan conectadas a una red Lan Manager pueden soportar los sistemas operativos MS-DOS y OS/2. Para estaciones que estan conectadas a una LAN Novell solo pueden soportar el sistema operativo OS/2 por el momento.

Facilidad de instalación.

La instalación de Sql Server no puede ser hecha como se realiza bajo Lan Manager , por lo que a continuación describiremos el proceso que se sigue para instalar Sql Server bajo ambiente Novell Netware, cabe mencionar que esté procedimiento no está incluido en los manuales de instalación que se proporcionan con el software, esté procedimiento fue obtenido de un boletin técnico especializado (vea referencias bibliograficas).

Para soportar a Sql Server Novell desarrolla un producto llamado requestor que es utilizado para la conexión

de estaciones OS /2 a la red y además incluye el protocolo Named Pipes.

Cuando se instala el requestor sobre la estación de trabajo que actuará como servidor de datos, el archivo config.sys se debe modificar en una línea donde se escribe el nombre del servidor . Esto permite a Novell Netware imitar el esquema que utiliza Lan Manager.

Pasos a seguir para instalar Sql Server sobre la red con sistema Novell Netware.

1.- Instalar el sistema operativo OS/2 versión 1.1 + sobre la máquina que funcionará como BACK-END.
2.- Una vez instalado el OS/2 sobre la estación que funcionará como servidor de datos se instala el OS/2 requestor de Novell Netware.
3.- Después de probar que existe comunicación entre la estación servidor y el file server de Novell y haber modificado el parámetro del config.sys "computer-name", se procede a instalar el Sql Server tecleando los siguientes comandos (como se mencionó anteriormente Sql Server no se puede instalar como en Lan Manager por lo que se tiene que hacer a mano) :
4.- Crear la estructura que sigue:
[c:] MD SQL
[C: SQL] MD BINP
[C: SQL] MD DATA
[C: SQL] MD DLL
[C: SQL] MD LOG

5.- Copiar los archivos de Sql Server a los directorios siguientes:

* copiar los archivos .exe y .hlp al directorio c:/sql/binp
* copiar los archivos DBLIBP.DLL y Mshelp.dll al directorio c: /sql/dll
*copiar los archivos .sql al directorio c:/sql/install.
6.- Crear el Sql Server ejecutable de la siguiente manera:

Posicionarse en el directorio c:/sql/binp y digitar el siguiente comando:
copy sqlserv1.exe /b + sqlserv2.exe /b sqlservr.exe/b.

7.- Modificar el path y libpath del config.sys e inicializar la máquina para que se realicen los cambios.

8.- Crear la base de datos maestra de la siguiente manera:

posicionarse en el directorio c:/sql/data y teclear la siguiente línea.
bldmastr /dMASTER.DAT /s5120 /C.

9.- ejecutar el Sql Server con la siguiente instrución:

SQLSERVR /dC: SQL DATA MASTER.DAT /eC: SQL LOG ERROR.LOG
10.- correr otro proceso de OS/2 y teclear los siguientes comandos en el subdirectorio
c: sql install.
isql /Usa /P <instmstr.sql
isql /Usa /P <instmodl.sql
isql /Usa /P <instpubs.sql

Instalación de Sql Server sobre la estación de trabajo (cliente).

1.- instalar OS/2 ver 1.1+
2.- instalar el requestor de Novell y activar el Named Pipes.
3.- insertar en el drive A, el disco de Sql Server de instalación para OS/2.
4.- en la pantalla que aparece seleccionar instalación de estación de trabajo.
5.- salir del menú.

Una vez que se ha instalado el Sql Server en la estación de trabajo se procede a comunicarse con el servidor inicializando la estación de trabajo.

Para la instalación de Sql Server bajo Lan Manager Sql Server checa si el servidor esta propiamente configurado. Si no se tiene la memoria necesaria en RAM o espacio en disco, o por alguna razón el Named Pipes no puede ser localizado, la instalación será automáticamente terminada.

Soporte de SQL.

Sql Server además de soportar el SQL ANSI tiene varias extensiones, por lo que se pueden realizar consultas complejas con pocas sentencias SQL.

Sql Server tiene un catálogo que controla la integridad de datos mediante "triggers", los cuales son definidos con el Transact-SQL , el cual incluye comandos de control tales como IF - THEN y GO TO . Transact-SQL puede construir rutinas que soporten la integridad referencial y reglas definidas por el usuario, asignandolas en el catálogo. Los triggers son automáticamente invocados cuando una tabla es actualizada . Ellos (trigger y reglas) aseguran que cuando un valor es modificado, automáticamente se cheque la integridad referencial e integridad semántica.

Sql Server permite definir procedimientos en el Transact-SQL, estos procedimientos son compilados y asignados en un catálogo solo una vez, este catálogo puede ser llamado por algún programa, haciendo llamadas al servidor vía el procedimiento, disminuyendo el flujo de comandos y así disminuir el tráfico en la red.

Sql Server al igual que ORACLE SERVER soporta subpreguntas.

Sql Server no soporta la instrucción UNION que es esencial para la mezcla de tablas, por lo que crea tablas temporales.

Control de concurrencia.

Sql Server emplea múltiples niveles de bloqueo, dependiento del tipo de requerimiento.

Los candados compartidos son utilizados para operaciones de solo lectura. Si un candado compartido ha sido aplicado a los datos una segunda transacción de solo lectura puede también recuperar los datos aunque la transacción primera no haya sido terminada, los candados compartidos son borrados de los bloques de datos cuando ya no se necesitan.

Sql Server usa un candado exclusivo para operaciones de actualización (borrar, insertar, modificar). Cuando un bloqueo exclusivo es puesto ninguna otra transacción puede obtener un bloqueo de ningún tipo, sino hasta que el bloqueo exclusivo sea borrado, y es borrado hasta que termina la transacción . Así mismo ninguna transacción puede obtener un bloqueo exclusivo a datos que tienen un bloqueo compartido. Por lo antes dicho Sql Server tiene mecanismos que detectan los deadlocks y livelocks.

Sql Server detecta los deadlocks y elimina cualquiera de las dos transacciones mediante un algoritmo especial.

Para eliminar los livelocks Sql Server los detecta y después de cuatro intentos del bloqueo exclusivo Sql Server refuta el requerimiento. Entonces el bloqueo exclusivo es otorgado tan pronto como se termine el bloqueo compartido.

Mecanismos de recuperación.

Sql Server provee un mecanismo de recuperación automático que no requiere de la intervención del administrador del sistema (sa) . Las rutinas de recuperación automática están corriendo todo el tiempo, éstas rutinas aseguran que todas las transacciones terminadas se escriban a disco antes de que el sistema se "caiga" y las que aún no se terminan se les aplica el rollback.

La recuperación automática es hecha en cosa de pocos minutos ya que el Sql Server puede usar el CHECKPOINTS con suficiente frecuencia.

Un CHECKPOINTS escribe todos los bloques marcados (páginas que han sido modificadas a partir del último checkpoints) a la tabla de la base de datos. Un checkpoints toma cerca de un minuto, lo que significa que todas las transacciones terminadas serán escritas en la base de datos.

Sql Server permite hacer backups en línea, es decir no se tiene que dar de baja el server para poder hacer un backup de la información .

Mecanismos de seguridad.

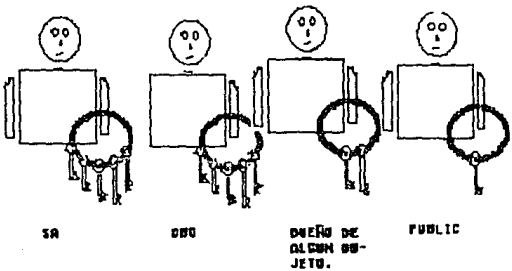
Los permisos para accesar tablas, vistas y crear bases de datos son otorgados sólo por el administrador del sistema o el dueño de la base de datos.

Existen dos clases de permisos usados con el grant y revoke , permisos sobre objetos y permisos sobre comandos. Los permisos sobre objetos regulan el uso de ciertos comandos sobre objetos de una base de datos; ellos son otorgados o revocados por el dueño del objeto. Ejemplos de permisos sobre objetos son: tablas, vistas, columnas y procedimientos (select , update, insert, delete, execute).

Los permisos sobre comandos son otorgados por el administrador del sistema o el dueño de la base de datos, ejemplo de permisos sobre comandos son los siguiente: create data base, create procedure, create rule, create table , create view.

Sql Server tiene los siguientes tipos de usuarios:

- Administrador del sistema (sa)
- Dueño de la Base de Datos (dbo)
- Dueño de algún Objeto (oo)
- Usuarios Públicos (public)



SA
Es un super usuario el cual tiene todos los derechos que otorga Sql Server .

DBO
Es el siguiente en la jerarquía del sqlserver, el dueño de la base de datos tiene los mismos derechos que el sa.

OO
Es el siguiente nivel al dueño de la base, éste usuario puede ser dueño de algún objeto tal es el caso de una tabla , vista o regla. El puede otorgar permisos a otros usuarios para que lo puedan usar.

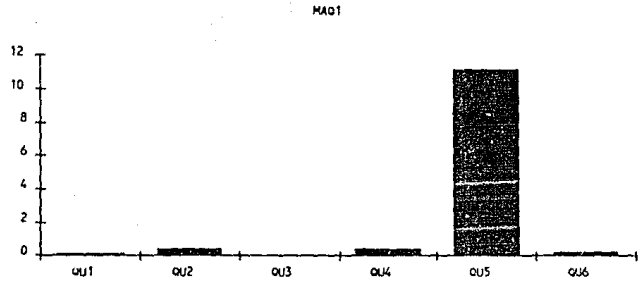
PUBLIC
A estos usuarios se les puede otorgar y revocar permisos para poder consultar una vista, consultar una tabla, etc.

Rendimiento.

A continuación mostramos los resultados de los programas aplicados para medir el rendimiento, es decir la velocidad de respuesta a los requerimientos de varios usuarios, asi mismo las gráficas (los programas fuente y resultados se encuentran en el anexo al final del capitulo).

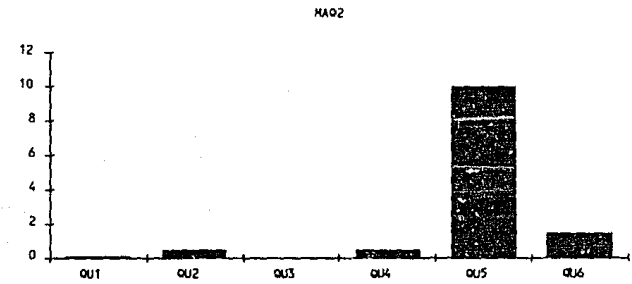
SQL SERVER					
	MAQ1	MAQ2	MAQ3	MAQ4	MAQ5
QUERY 1	.08 MIN	.09 MIN.	10 MIN	.01 MIN	02 MIN.
QUERY 2	.36 MIN	.46 MIN	.46 MIN	1.34 MIN	.39 MIN.
QUERY 3	ERROR	ERROR	ERROR	ERROR	ERROR
QUERY 4	.38 MIN	.41 MIN	.40 MIN	.10 MIN	.42 MIN.
QUERY 5	11.19 MIN.	10.01 MIN	10.39 MIN	2.25 MIN	8.50 MIN.
QUERY 6	.22 MIN	1.43 MIN.	1.0 MIN	.08 MIN	2.23 MIN.

minutos



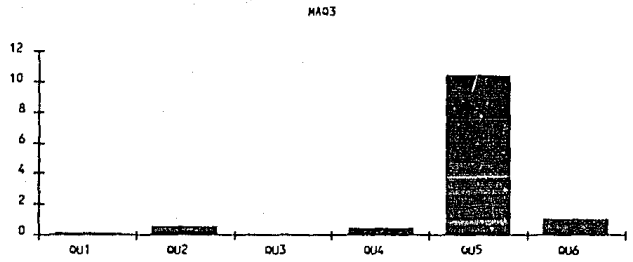
petición

minutos



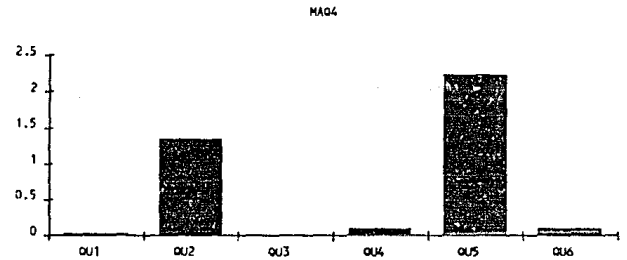
petición

minutos



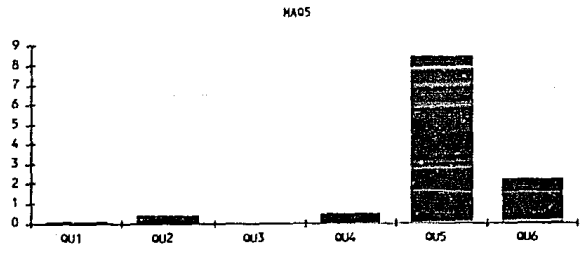
petición

minutos



petición

minutos



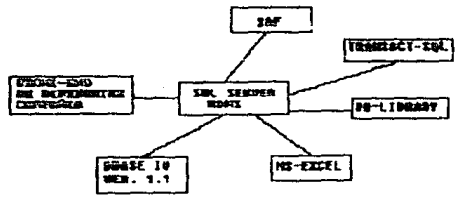
petición

Front-ends soportados, uso de APIs y lenguajes de alto nivel.

Sql Server posee un conjunto de herramientas desarrolladas por Microsoft, tal es el caso de MS-EXCEL (hoja electrónica) y de Asthon-Tate Dbase IV versión 1.1 y el propio Transact-Sql que es un 4GL, éstos front-ends permiten desarrollar complejas y amigables aplicaciones para ambiente LAN. Así mismo Sql Server permite tener como front-end a las aplicaciones hechas en lenguajes tales como: C,Cobol, Pascal, Fortrán y Macro ensamblador,por el momento solo "C".

Para interactuar con el database server (BACK-END) mediante los lenguajes mencionados anteriormente, Sql Server utiliza sus APIs llamadas DB-LIBRARY , las cuales son incrustadas directamente dentro del lenguaje anfitrión. Muchos desarrolladores de software están realizando herramientas que pueden ser utilizadas como FRONT-ENDs en Sql Server, tales compañías son:

COMPAÑIA	DESCRIPCION DEL PRODUCTO.
Applied Research	RealtimeApps for Instruments
Borland International	Paradox DBMS.
Computer Associate	Project Management and Accounting.
Data Ease Int	DataEase SqlDbms.
Information Builders	Focus DBMS.
Internetics	DBMS & Cominicatión Internatión.
Jhonson & Higgins	Complete Claims System.
Mdbs Inc.	Database and Expert System.
Popkin Software	Case Tool.
Revelation Tech.	Advanced Revelatión DBMS.
Saros Inc.	Document library System.
Solutions by Design	Objeted Oriented Front -end tool.
TLB Inc.	Solomon Accounting Package.
Wordtech Systems	Quick Silver - Dbase compiler.



Sqlserver soporta los siguientes tipos de datos:

int	-2,147,483,647 a +2,147,483,647
small int	-32,768 a +32,767
tinyint	entre 0 a 255
float	valores de punto flotant
char (n)	máximo 255 caracteres de longitud fija
varchar (n)	máximo 255 caracteres de longitud variable
text	2,147,483,648 caracteres
binary (n)	datos binarios de longitud fija
varbinary(n)	de longitud variabl
imagen	2,147,483,648 bytes
bit	0 o 1
money	asignación de dólares y centavos (+/- 922,337,203,685,447.5807 dólares)
data time	día y hora
reglas	lista de valores que debe tomar la variable
timestamp	utiliza la hora cuando el registro es actualizado

tipos definidos por el usuario (tales como:código char (2))

Conectividad a Minis y Mainframes.

Sqlserver tiene la facilidad de poder conectarse a minis: DEC, VAX , PYRAMID y SUN-MINI, ya que el Sql Server es también disponible por Sybase para éstas minis mencionadas.

Capacidad de distribución.

Sql Server soporta múltiples servidores sobre la red. El manejo de las bases de datos distribuidas se realiza a través del método "TWO FASES COMMIT", pero no es transparente el uso de éste método. Los desarrolladores necesitan utilizar funciones especiales del DB-LIBRARY para aquellos programas que hacen uso de bases de datos distribuidas. Una vez programada propiamente la aplicación, es consciente de que son operaciones distribuidas y toma en cuenta que servidores están involucrados en la transacción.

Two fases commit.

El método de Two Fases Commit es utilizado para realizar transacciones que involucren a varios servidores el cual funciona de la siguiente manera:

Fase I

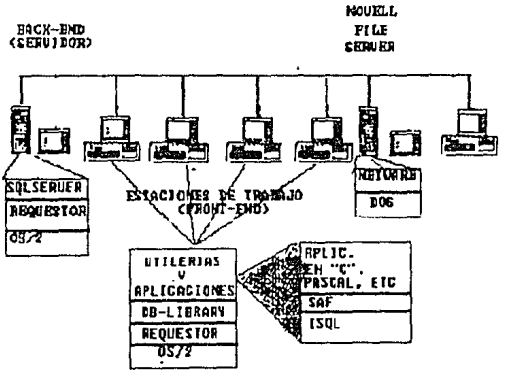
La primera fase inicializa la extracción de datos de cada servidor de datos involucrado en la transacción distribuida. Una vez que termina el servidor la parte de transacción que le corresponde éste envía un mensaje al commit server indicando que está listo para confirmar esa porción de la transacción.

Fase II
Cuando todos los Sql Servers involucrados indican que han terminado su proceso (se ha preparado la transacción), el commit server envía un mensaje indicando que todos los registros y transacciones están confirmadas para llamar a una utileria de servicio, donde se encuentra el commit especial. Una vez que ésto sucede, la transacción está totalmente confirmada.

Si un Sql Server falla antes de haber realizado el "prepare transactión" el commit server cancela la transacción totalmente, notificando a los servidores involucrados realizar un rollback a su proceso respectivo.

SAF (Sqlserver Administration Facility)
Es una utileria que sirve como front-end y que puede ser utilizada con el mouse.
El SAF es un sistema de menues y dialog boxes que permiten que el administrador del sistema pueda ejecutar tareas de administración. Además el SAF permite a otros tipos de usuarios hacer requerimientos ("querys") a la base viendo los resultados del query, así mismo permite editar, guardar, copiar, imprimir, etc. , un query y sus resultados.

ISQL
Es una utileria que sirve para el manejo de datos en linea. Los usuarios pueden accesar los datos via ISQL tecleando el comando SQL y después la palabra GO.



ARQUITECTURA BAJO LA QUE SE REALIZARON LAS PRUEBAS DE SQL SERVER.

4.- CRITERIOS DE SELECCION

4.1 PUNTOS.

En esta sección analizaremos a los servidores de base de datos evaluados en el capítulo tres, para posteriormente dar los puntos que servirán como referencia para la selección de un servidor de base de datos acorde a las necesidades que se tengan. En el análisis, revisaremos en conjunto a todos los servidores de base de datos evaluados, tomando en cuenta los puntos de evaluación del capítulo tres.

4.1.1 Requerimientos de hardware

Influyen muchos factores para seleccionar un servidor de base de bases de datos, entre ellos el sistema operativo y los recursos requeridos que como ya se dijo en el capitulo tres en el esbozo de los puntos a evaluar, son punto importantes, porque son función del rendimiento del RDBMS. A continuación mostramos en la fig.1 los requerimientos de cada producto.

Ver.	PRODUCTO	COMPAÑIA	SIS.OPER.	HARD DISK(MIN.)	RAM	LANs
1.0	ORACLE SERVER	ORACLE CORPORATION.	OS/2.	30MB.	6MB	TODAS LAS QUE SOPORTEN IPX/SPX, NETBIOS, NAMED-PIPES
3.0	SQL BASE SERVER.	GUPTA TECHNOLOGIES.	DOS 3.1+ OS/2.	1MB.	640 KB	TODAS LAS QUE SOPORTEN NETBIOS COMPATIBLE.
2.01b	NETWARE SQL	NOVELL.	NOVELL NETWARE VER.215+	2MB.	512 KB	SOLO NETWARE
1.0	SQL SERVER.	ASHTON-TATE/MICROSOFT/SYBASE.	OS/2.	20MB.	6MB	TODAS LAS QUE SOPORTEN NAMED-PIPES.

fig.1.

Si observamos la figura 1 podemos percatarnos que los que utilizan más recursos son Oracle Server y Sql Server, por lo que se tomará en cuenta para el análisis de rendimiento.

El siguiente punto en la evaluación es el sistema operativo de LAN, bajo el cual trabaja el back-end. Este punto es de mayor importancia para corporaciones donde ya se tiene un esquema de redes. En la figura 1 se muestra el sistema operativo de red, bajo el cual puede correr el back-end. Asi mismo el sistema operativo donde se instala el back-end, en la figura 2 encontramos los sistemas operativos

que soportan los clientes (estación de trabajo) para accesar al back-end (database server).

Ver.	PRODUCTO	COMPAÑIA	SIS.OPER.	HARD DISK(MIN.)	RAM.
7.0	ORACLE SERVER	ORACLE CORPORATION.	OS/2. VER 1.1 DOS	5MB 5MB	4MB. 2MB.
3.4	SQL BASE SERVER.	GUPTA TECHNOLOGIES.	DOS 3.1+ OS/2. VER 1.1	MINIMO.	640KB. 4MB.
2.01b	NETWARE SQL	NOVELL.	DOS.	MINIMO.	640KB.
1.0	SQL SERVER.	ASHTON-TATE/MICROSOFT/SYBASE.	OS/2.VER 1.1. DOS	5MB 5MB	4MB. 640KB.

fig.2

analizando la figura 1 podemos observar que si alguna empresa o institución cuenta con un esquema de redes tal como Novell, Lan Manager o Ungermann-Bass. observamos que el que tiene más posibilidades de conexión en cuanto a protocolos es Oracle Server, ya que este puede correr en redes que soportan netbios, ipx/spx y named-pipes, lo que los otros manejadores no tienen, consecuentemente tienden a "casarse" con determinadas LANs.

Con respecto a la instalación todos los RDBMS evaluados tienden a ser fáciles, sólo que cuando no están diseñados para correr en determinadas LANs la instalación es un poco compleja. Aquellas LANs para las cuales el paquete (software) está diseñado, el programa de instalación oculta muchos detalles.

El servidor que tiene una instalación muy sencilla es Netware Sql ya que es muy intuitiva, así mismo el tiempo de instalación es muy breve. En Netware Sql los errores son "entendibles", cosa que no sucede con los otros tres servidores evaluados (Oracles Server,Sql Base y Sql Server), vea la figura 3.

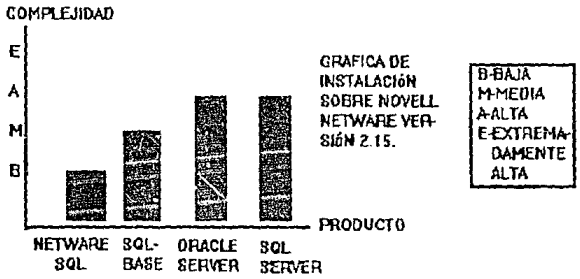


fig3.

4.1.2 Documentación adecuada.

En este punto podemos observar que existe una gran diferencia entre los manuales de cada producto, así mismo nos percatamos que en algunos manejadores de base de datos tal es el caso de Oracle Server (14 manuales), Sql Server (6 manuales) tienen demasiados manuales y otros tienen muy pocos, tal es el caso de Netware sql. Para el caso de Sql Base contiene solo tres manuales.

La puntuación de la figura 4 indica qué tan rápido el usuario puede trabajar realizando programas, creando tablas y bases de datos una vez que ha entendido las funciones básicas. Los programas que están incluídos en los manuales y que requieren de un mínimo de comandos a ejecutar para realizar una función son más fáciles de recordar por lo que tienen menos dificultad.

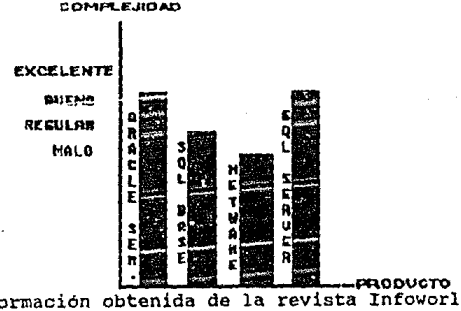


fig.4 información obtenida de la revista Infoworld Marzo 5, 1990 Volumen 12, edición 10.

La gráfica muestra los resultados despues de que ha revisado y utilizado un programa que ejecute una serie de tareas y de haber estudiado el tutorial y ejemplos.

Si revisamos la figura 4 esta favorece a Oracle Server y Sql Server pero Oracle contiene más manuales, y SQL SERVER no contiene tutorial.

4.1.3 Soporte del SQL

Todos estos RDBMS evaluados soportan el SQL ANSI estandar lo que permite tomar aplicaciones hechas en PCs y transportarlas a minicomputadoras y mainframes tal como DB2 de IBM La figura 5 muestra los comandos que soporta cada uno de los servidores de bases de datos evaluados.

Al utilizar el estandar SQL garantiza la portabilidad a otro ambiente, pero las extensiones de cada uno de los productos reducen el esfuerzo de programación e incrementa el poder del lenguaje.

COMANDOS SQL ANSI ESTANDAR

COMANDO SQL-DML	ORACLE SERVER	SQL BASE SERVER	NETWARE SQL	SQL SERVER
SELECT	SI	SI	SI	SI
COLUMNS	SI	SI	SI	SI
EXPRESSIONS	SI	SI	NO	SI
DISTINCT	SI	SI	NO	SI
FROM	SI	SI	SI	SI
WHERE	SI	SI	SI	SI
GROUP BY	SI	SI	SI	SI
HAVING	SI	SI	SI	SI
ORDER BY	SI	SI	SI	SI
SUBQUERIES	SI	SI	SI	SI
UPDATE SET	SI	SI	SI	SI
WHERE	SI	SI	SI	SI
SUBQUERIES	SI	SI	NO	SI
INSERT INTO	SI	SI	SI	SI
SUBQUERY	SI	SI	NO	SI
DELETE FROM	SI	SI	SI	SI
SUBQUERY	SI	SI	SI	SI
UNION	SI	SI	NO	SI
DML PREDICADOS				
BETWEEN	SI	SI	SI	SI
LIKE	SI	SI	SI	SI
IS NULL	SI	SI	SI	SI
EXISTS	SI	SI	NO	SI
ALL	SI	SI	NO	SI
ANY	SI	SI	NO	SI
SOME	NO	NO	NO	NO
[NOT]	SI	SI	SI	SI
DML FUNCIONES				
AVG	SI	SI	SI	SI
COUNT(*)	SI	SI	SI	SI
COUNT	SI	SI	SI	SI
MAX	SI	SI	SI	SI
MIN	SI	SI	SI	SI
SUM	SI	SI	SI	SI
DDL				
ALTER TABLE	SI	SI	SI	SI
CREATE TABLE	SI	SI	SI	SI
CREATE INDEX	SI	SI	SI	SI
CREATE UNIQUE INDEX	SI	SI	SI	SI
CREATE VIEW	SI	SI	NO	SI
DROP TABLE	SI	SI	SI	SI
DROP INDEX	SI	SI	SI	SI
DCL.				
GRANT	SI	SI	SI	SI
REVOKE	SI	SI	SI	SI
MANEJO DE TRANSACCIONES				
COMMIT	SI	SI	SI	SI
ROLLBACK	SI	SI	SI	SI

fig .5. tabla obtenida de la revista Pc Tech Journal, Marzo 1989.

4.1.4 Control de concurrencia.

Una función importante de los servidores de base de datos es el bloqueo automático para prevenir las colisiones entre transacciones que puede llevar a la inconsistencia y pérdida de datos.

La figura 6 presenta los resultados (sumario) de bloqueo con los que cuenta cada producto.
Todos los servidores de base de datos manejan "locks" automáticos en las tablas (nivel tabla), página (nivel block), ó a nivel registro. Como la información está centralizada, los servidores permiten tener "locks" compartidos, permitiendo a los programas leer datos aunque esten bloqueados o bloquear datos cuando se esta actualizando previniendo que otros usuarios traten de actualizar antes de que termine la transacción.

	ORACLE SERVER	SQL BASE	NETWARE SQL	SQL SERVER
NIVEL BLOQUE				
CURSOR STABILITY	•	•	•	•
REPEATABLE READ		•		•
READ CONSISTENCY	•	•		
BLOQUEO AUTOMATICO				
NIVEL TABLA	•		•	
NIVEL BLOQUE		•		•
NIVEL REGISTRO				
LOCKING CONTROLADO POR EL PROGRAMADOR.	•			
NIVEL TABLA				
NIVEL BLOQUE				
NIVEL REGISTRO			•	
DETECCION DE COLISIONES.	•	•		•
DEADLOCKS				
LIVELOCKS				•

fig.6.

Oracle server y Netware Sql bloquean automáticamente toda la tabla por lo que, cuando se realiza una transacción se bloquea la tabla degradando el "performance". Sql Server y Sql Base bloquean automáticamente a nivel página lo que permite que exista mayor concurrencia.
Netware Sql es el unico que permite que los desarrolladores bloquen a nivel registro, por medio del wait y nowait.

Sql Server, Oracle Server y Sql Base permiten la detección de deadlocks eliminando los overhead* cosa que no puede hacer Netware Sql.

Sql Server es el único que permite la detección de livelock, por lo que elimina aún más el overhead*.

Tanto Sql Server, Sql Base, Oracle Server y Netware Sql soportan la forma de aislamiento cursor stability lo que permite que varios usuarios lean y/o actualisen datos a la vez.

*Overhead. Factores que causan el degrado del "performance" ideal.

Solo Sql Server y Sql Base permiten el "repeatable read" asegurando la integridad de datos cuando se leen concurrentemente.
Para el read consintency los servidores que lo soportan son : Oracle Server y Sql Base.
Concluyendo: Sql Base es el que tiene mejor manejo de concurrencia y Sql Server el que tiene mejor detección de overhead.

4.1.5 Mecanismos de recuperación.

Cada uno de los servidores de base de datos tienen sus propias formas de recuperación, pero si se puede decir que todos soportan la recuperación automática por caidas del sistema, fallas en el hardware, fallas de energia electrica, etc. Los mecanismos de recuperación en cuanto a programas son los mismos (rollback transaction).

Netware Sql y Sql Server, tienen comandos que permiten iniciar y terminar una transacción (begin transaction, end transaction). Si la transaccion no termina se hace un rollback.

Para recuperar actualizaciones todos los servidores de base de datos evaluados contienen técnicas para el "forward recovery journal". Todos los servidores contienen un archivo que registra todas las operaciones que se hacen durante una sesión.Si el log (archivo registrador) tiene alguna transacción parcialmente hecha, el manejador remueve las transacciónes y el archivo de "forward recover log" completa las transacciones correctamente.

4.1.6 Seguridad.

El servidor de bases de datos que tiene el mejor control en usuarios y seguridad en la información es Sql Server, posteriormente se encuentra Netware Sql, ya que Netware Sql puede controlar a los usuarios a nivel grupo o individual, pero no puede otorgar derechos en las tablas a nivel registro. Oracle Server y Sqlbase solo pueden manejar a los usuarios a nivel individual de tal manera qué en orden de mayor importancia en control de usuarios es:
-Sql Server.
-Netware Sql.
-Oracle Server y Sqlbase.
En cuanto a control en tablas es:
-Sql server.
-Oracle Server y Sqlbase.
-Netware Sql.

4.1.7 Rendimiento.

El rendimiento es un factor muy importante para que se acepte un RDBMS, ya que para los usuarios es lo principal.

Los "query´s" ejecutados en los RDBMS evaluados son los que se muestran en la figura 2 del capitulo 3, los cuales explicaremos a continuación, así mismo analizaremos

los resultados obtenidos, estos resultados se muestran en los cuadros del capitulo tres.

Un comentario importante es que solo supondremos lo que hace el optimizador dado que ningún RDBMS proporciona información de su optimizador. Por lo mismo nos basaremos en la teoria expuesta en el capitulo dos, sección optimizador.

QUERY 1.
Select * from mov where mov_xvolum='90001' and fra_fracci < '0302_6999'

Prueba la habilidad del optimizador para cambiar al índice correcto en este caso, el rendimiento es incrementado por usar el índice mov_x volum . Por lo siguiente:
lo primero que suponemos que se realiza es convertir las proposiciones del usuario a una forma de bajo nivel entendible al RDBMS, en segundo lugar el optimizador toma la proposición y trata de transformarla a una forma equivalente pero que sea más eficiente en su ejecución.

Después de haber transformado la proposición del usuario a una forma reconocible al RDBMS y haberla transformado a una forma equivalente, pero más eficiente, el optimizador toma información del diccionario de datos el cual contiene información tal como:
-- Número de índices y tipo.
-- Atributos por los que estan conformados los índices.
-- Número de tuplas que caben en un bloque.
-- Número de tuplas con que cuenta la tabla.
-- Número de veces que se repite un atributo.

Como se mencionó en el capítulo dos, el optimizador que tome mayor información de su diccionario realizará más rápido la consulta.

Este query favorece a Netware Sql.

QUERY 2.
Select * from mov where mov_xvolum='90001' or fra_ fracci ='0302_6999'

Prueba la habilidad del optimizador para utilizar índices en el or lógico. En este caso, usando ambos índices reduce la pregunta para seleccionar justamente aquellos renglones que conocen mov_xvolum ó la fracción, o posiblemente algún servidor solo barrera la tabla tomando más tiempo.

Para la primera opción tal vez entren Oracle Server y Sql Server y para la segunda opción entraria Sql Base y Netware Sql. La diferencia en tiempos entre Sql Base y Netware Sql es que Netware Sql utiliza los recursos del "file server" de Novell, por lo que tal vez carga a memoria principal en su "file caching" las tablas, tardando menos tiempo que Sql Base. Por lo tanto el mejor en esta prueba es Netware Sql.

QUERY 3.
Select * from mov where fra.fracci > '0303_7999' order by mov_xvolum.

Prueba la habilidad del optimizador para usar el indice fra.fracci ya que el RDBMS puede tener buenos algoritmos de clasificación y entonces no usar el indice y al mismo tiempo ordenar los datos. Para este query el que presenta mejores algoritmos de ordenamiento es Netware Sql y posteriormente Sql Base, ya que Oracle Server y Sql Server genera tablas temporales por lo que si no se tienen memoria RAM y espacio suficiente en disco, marcará error de espacio en disco o de memoria.

QUERY 4.
Select sum (mov_ valcom) from mov
Prueba la facilidad del servidor para usar funciones matemáticas.
En este query los resultados favorecen a Oracle Server y posteriormente a Sql Server y al que menos favorece es a Netware Sql.

QUERY 5.
Select mov_xvolum from mov group by mov_xvolum having count (*) >5.

Este query agrupa el requerimiento mediante un sort y además otras funciones de agrupamiento. Los paquetes con algoritmos de agrupamiento eficientes pasaran mejor la prueba tal es el caso de Oracle y porteriormente Sql Base.

QUERY 6.
Select mov.fra.fracci, fra.fra_desfra from mov, fra where mov.fra_fracci = fra.fra_fracci and mov.mov_xvolum like '900%'.

Prueba la junta de dos tablas con el comando like. Aqui hay varias formas de ejecutar el query. Por ejemplo los renglones que contienen 900 pueden ser escogidos primero y entonces hacer la junta ó el optimizador puede juntar todos los renglones y seleccionar solo aquellos que cumplan con los tres caracteres puestos en el like.

Este query favorece a Oracle Server y Sql Server, Netware Sql tiene muy malos algoritmos para tratar al like ya que si lo quitamos tardará menos tiempo en realizarse la pregunta.

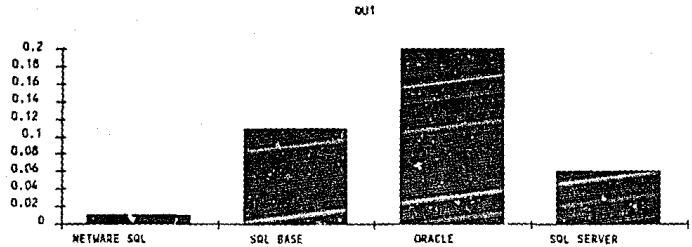
Tomando en cuenta que este tipo de query es el más común en la vida real, Netware Sql tendrá que mejorar este punto.

A continuación mostramos gráficas que son el resultado de la media obtenida en cada máquina para cada query ejecutado .

Esto es:

$$\sum_{i=0}^{5} \frac{\text{MQi}}{5}$$

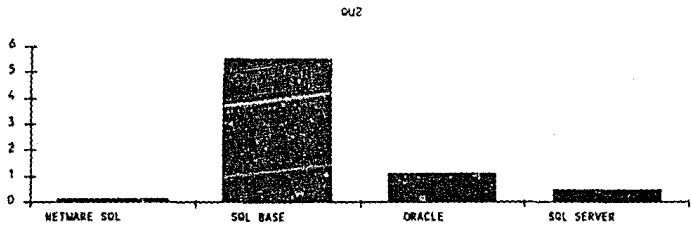
minutos



producto.

$$\sum_{i=0}^{5} \frac{\text{MQi}}{5}$$

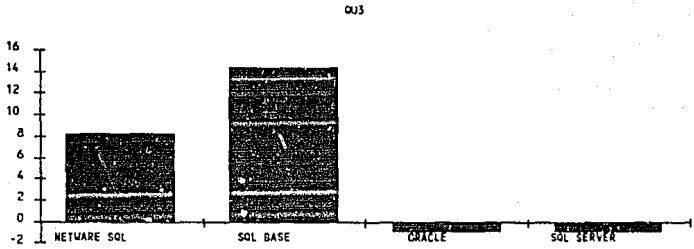
minutos



producto.

$$\sum_{i=0}^{5} \frac{MMQi}{5}$$

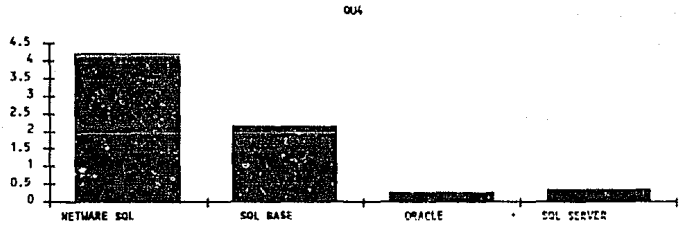
minutos.



producto

$$\sum_{i=0}^{5} \frac{MMQi}{5}$$

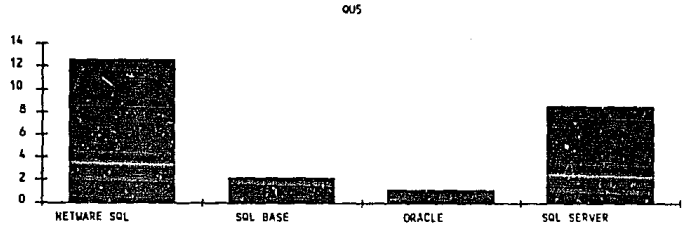
minutos.



producto

$$\sum_{i=0}^{5} \frac{MRQ1}{5}$$

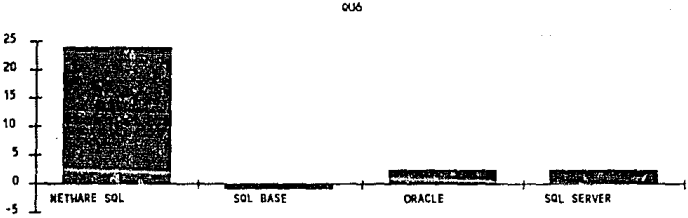
minutos.



producto.

$$\sum_{i=0}^{5} \frac{MRQ1}{5}$$

minutos.



producto.

4.1.8 Front-ends.

Los FRONT- ENDs enumerados en cada uno de los servidores se diferencian porque algunos son basados en gráficos y otros son basados en caracteres. Un ejemplo de los que estan basados sobre gráficos es Sql Windows de Gupta Technologies. Esta herramienta esta diseñado para construir aplicaciones basadas sobre gráficos con el ambiente de Microsoft Windows y Presentation Manager para OS/2. Este front_end usa "dialog boxes", "pull_down" manejados con o sin mouse.

Para el otro caso (front_ends basados en caracteres) esta el dBase IV Ver 1.1.

Existen front_ends que trabajan con su propio servidor solamente. Ejemplos de front_ends que trabajan con su propio servidor de datos están:

front_end	servidor
Sql forms	Oracle Server.
Sql windows	Sql Base.
Xql	Netware Sql.
Dbase IV	Sql Server.

Ejemplos de front_ends que trabajan con más de un servidor son:
Paradox DBMS, Advanced Revelation DBMS, Quick Silver _Dbase Compiler de Word Tech Systems, MS_Excel Spreadssheet de Microsoft Corp, entre otros.

FRONT-END	ORACLE SERVER	SQL BASE	NETWARE SQL	SQL SERVER
ADVANCED REVELATION DBMS(REVELATION TECH)	●	●	●	●
QUICK SILVER-DBASE COMPILER (WORD TECH SYSTEMS)	●	●	●	●
PARADOX DBMS (BORLAND INTERNATIONAL)	●		●	●
MS EXCEL SPREADSHEET (MICROSOFT CORP.)	●			●
SQL WINDOWS. (GUPTA TECHNOLOGIES)	●	—	—	●

*(entre otros)

información obtenida de 3+open Lan Manager Applications associates Marketing. Febrero 24 de 1989.

fig.7

Como se ve en la figura 7 Sql Windows, que es un front_end basado en gráficos trabaja sobre los cuatro servidores evaluados y Quick Silver-Dbase Compiler de Word Tech Systems trabaja en los cuatro servidores, pero a diferencia de Sql Windows trabaja en base a caracteres.

Concluyendo: los servidores que más front-end's tienen es Sql Server y Oracle Server.

4.1.9 Lenguajes de alto nivel y APIs.

Cada servidor de datos viene con sus propias APIs (application program interface), algunos son precompiladores tal es el caso de Oracle Server, o rutinas que permiten a los desarrolladores de aplicaciones interactuar con el servidor de datos como Sql Base, Sql Server y Netware Sql . Estas APIs permiten tener como front_end lenguaje tales

como C, Cobol, Pascal, Basic, etc (como lo muestra la figura 8).

El más favorecido en cuanto a lenguajes de programación es Oracle Server, posteriormente Netware Sql, después Sql Base y finalmente sql Server, por el momento.

LENGUAJES DE PROGRAMACION QUE SE PUEDEN UTILIZAR COMO FRONT-ENDs.

SERVIDOR / LENGUAJE	ORACLE SERVER	SQL BASE	NETWARE SQL	SQL SERVER
BASIC	NO	NO	SI	SI *
FORTRAN	SI	NO	NO	SI *
COBOL	SI	SI	SI	SI *
PASCAL	SI	NO	SI	SI *
C	SI	SI	SI	SI
ADA	SI	NO	NO	NO
PL/I	SI	NO	NO	NO

* posteriormente a liberar.

fig.8.

Cuando se escriben aplicaciones en lenguajes tal como C y dentro del programa existen comandos Sql existe un problema por que SQL retorna tablas y C por ejemplo retorna registros. Para resolver este problema se definen cursores que son punteros a los registros que fueron seleccionados. Todos los servidores es sus APIs contienen cursores como lo muestran los fragmentos de programas.

En los fragmentos se encuentra código de una aplicación Oracle y es equivalente a cada uno de los demás pero los comandos de Oracle son más fáciles de entender y ocupan menos código, pero al precompilarlo se genera un código muy grande que es poco entendible al programador.

APIs DE ORACLE SERVER.

```
EXEC SQL DECLARE C1 CURSOR FOR
SELECT  FRA_MOVI,FRA_FRACCI,PAI_PAIS,EMP_RNIE,MOV_VALCOM,MOV_XVOLUM,
        MOV_UNED FROM MOV
           WHERE MOV_XVOLUM='90001' AND FRA_FRACCI < '0302-6999';

EXEC SQL OPEN C1;
EXEC SQL WHENEVER NOT FOUND GOTO endloop
num_ret=0;
while(1)
{
EXEC SQL FETCH C1 INTO :sFraMovi,:sFraFracci,:sPaiPais,:sEmpRnie,
                       :fMovValcom,:sMovXvolum,:sMovUnimed;
print_rows(sqlca.sqlerrd[2] - num_ret);
num_ret=sqlca.sqlerrd[2];
}
```

APIs DE SYBASE.

```
static    char      updItem[] =
"select * from mov where mov_xvolum='90001' and fra_fracci <'0302-6999'";
.
.
  status=sqlnsi(cur, &nsi);                    /* obtiene # de atributos seleccionados*/
  if(status)
     exit(1);
  for (sic = 1; sic <= nsi; sic++)       /* obtiene informacion de cada columna*/
  {
    sqldes(cur, sic, &ddt, &pdi,        /*      */
        SQLNPTR,SQLNPTR,SQLNPTR,SQLNPTR);
    sqlssb(cur, sic, pdt, cp,
        pdi, 0, SQLNPTR, SQLNPTR);
   cp+=(pdi+1);
  }
    EJECUTA EL COMANDO SELECT
status =sqlexe(cur);
if(status)
{
  sqlerr(status,errmsg);
  printf("fallo ejecucion %s\n",errmsg);
  exit(1);
}
```

XQLM DE NETWARE SQL

```
status   = XQLCursor (&cursorid);
  if (status)
   {
    printf ("XQLCursor failed, status: %d\n", status);
    quit (); return (0);
   };

  status   = XQLCursor (&cursorid2);
  if (status)
   {
    printf ("XQLCursor failed2, status: %d\n", status);
    quit (); return (0);
   };

  /*                                                          */
  /*   Compila el comando sql                                 */
  /*                                                          */

   strcpy (statement,
   "select * from mov where mov_xvolum = '90001' and fra_fracci < '0302-6999' ");
 statlen = strlen (statement);
   printf ("%s\n",statement);
   status  = XQLCompile (cursorid, &statlen, statement);
   if (status)
    {
     printf ("XQLCompile failed, status: %d\n", status);
     quit (); return (0);
    };
```

DB-LIBRARY DE SQL SERVER

```
   /*
 dbcmd(dbproc,"select * from mov where mov_xvolum='90001' and fra_fracci <'0302-6999'");
  dbsqlexec(dbproc);

  while ((return_code = dbresults(dbproc)) != NO_MORE_RESULTS)
  {
     if (return_code == SUCCEED)
```

```
{
   dbbind(dbproc,1,STRINGBIND, (DBINT) 0,fra_movi);
   dbbind(dbproc,2,STRINGBIND, (DBINT) 0,fra_fracci);
   dbbind(dbproc,3,STRINGBIND, (DBINT) 0,pai_pais);
   dbbind(dbproc,4,STRINGBIND, (DBINT) 0,emp_rnie);
   dbbind(dbproc,5,FLTBBIND, (DBINT) 0,(BYTE *) &mov_valcom);
   dbbind(dbproc,6,STRINGBIND, (DBINT) 0, mov_xvolum);
   dbbind(dbproc,7,SMALLBIND, (DBINT) 0,(BYTE *) &mov_umed);

   while (dbnextrow(dbproc) != NO_MORE_ROWS)
   {
      printf
         ("%s  %s %s %s %f %s %d \n",
          fra_movi,fra_fracci,pai_pais,emp_rnie,mov_valcom,mov_xvolum,mov_umed);
```

Los front_end (incluyendo los lenguajes de programación C, pascal,etc) descritos son solo parte de los que existen para los servidores de bases de datos, asi mismo se incrementara su mercado.

4.1.10 Conectividad a Minis y Mainframes.

Este punto favorece grandemente a Oracle Server ya que corre en un amplio rango de sistemas operativos de microcomputadoras, minicomputadores y mainframes.

Posteriormente está Sql Server que es compatible con Sql Server para algunas minis, en seguida esta sql base el cual se puede conectar a DB2 de IBM.

Al servidor que más afecta este punto es a Netware Sql ya que sólo corre bajo el sistema operativo Netware de Novell.

4.1.11 Capacidad de distribución.

Como ya se había mencionado este punto es relativamente nuevo por lo que todos los servidores evaluados tienen una pequeña idea de lo que es base de datos distribuidas. Al Servidor de base de datos que favorece este punto es a Sql Server con su técnica "two fases commit" que permite tener una consistencia en los datos cuando se hacen actualizaciones en más de un servidor de datos. Cabe señalar que tanto para este servidor y para los demás el grado de distribución que tienen aún no es transparente para los usuarios y desarrolladores.

4.2. RECOMENDACIONES.

Nosotros identificamos una serie de puntos qué en algún momento podrían servir como ayuda para determinar que RDBMS con arquitectura cliente-servidor podría ser la mejor opción para determinada empresa.

Dado que ninguno de los productos evaluados es mejor que otro en todos los ambientes LAN y en todos los aspectos, proponemos algunas recomendaciones que como se mencionó arriba serviran en algún momento para que se logre una

buena decisión sobre el servidor que se adquirirá, y estas son las siguientes:

1. Decidir si se necesita una arquitectura cliente-servidor.
2. Comparar distintas alternativas en cuanto al servidor de bases de datos.
3. Comparar distintas alternativas en cuanto a Front-end.
4. Selección del DBA evaluando sus conocimientos.
5. Planear para crecer.

1. Decidir si se necesita una arquitectura cliente-servidor.

Si una empresa está dedicada a tareas transaccionales con actualizaciones frecuentes como son las aplicaciones contables, compañías de seguros, aerolíneas, instituciones bancarias así como gubernamentales donde se tienen que manejar grandes cantidades de información y la información será accesada por muchos usuarios al mismo tiempo o se desee eliminar máquinas grandes (minis o mainframes) y se cuente con redes, así mismo se requiere que esta información sea tratada de manera diferente como por ejemplo análisis de información, información tratada por procesadores, gráficos de la información, etc y además se tiene un gran capital para hacer este tipo de compra, ya que se requiere de máquinas muy poderosas tal como una máquina con procesador 386 con gran cantidad de memoria RAM (8 MB o más). Un servidor de base de datos tiene sentido.

2. Comparar distintas alternativas en cuanto a servidor de bases de datos.

Para poder hacer esta comparación nosotros identificamos los puntos mencionados en el capítulo tres con los cuales fueron evaluados los motores de base de datos y se les dará el peso necesario según la empresa que se trate.

¿cual es realmente el costo ?

Este es un factor muy importante en cualquier empresa. El costo de cada software de servidor de datos varía grandemente de producto a producto. Los vendedores dicen que su servidor corre con determinadas especificaciones mínimas pero esto también es factor del rendimiento, posiblemente para que un servidor de base de datos funcione idealmente se necesite de una configuración costosa en cuanto a hardware y software.

Algunos servidores de bases de datos corren en la misma máquina que es utilizada como file server bajando el costo de tal configuración (cliente-servidor).

Con respecto a los front _ends, Los vendedores tienen diferentes precios y esquemas de licencia. Algunos front_ends son vendidos por usuario y algunos son vendidos por data server. Aunque hay ligeras variaciones a las dos formas mencionadas .

Así que se tiene que poner mucho énfasis a este punto en cualquier corporación que se trate y por ende se tiene que dar gran peso.

sistemas operativos.

Con el advenimiento de OS/2 y el creciente interés en UNIX y XENIX, este punto, viene a ser muy importante . Si se planea moverse a otros ambientes operativos y aún no se tiene instalada ninguna red, será bueno considerar llevar las aplicaciones y archivos a Lan Manager y por ende a Sql Server. Si se tiene Novell y se considera moverse a un "file server" Netware 386 se debe considerar seriamente Netware sql. Además actualmente no hay otro servidor de base de datos que corra directamente en el file server como lo hace Netware Sql. Si bien Oracle Server, Sql Server y Sql Base pueden correr en un ambiente Netware, cada uno requiere que se dedique una PC AT como el servidor de datos para tener un rendimiento razonable.

Si se tienen LANs Netware y se requiere de alguna característica que no tenga Netware Sql como por ejemplo alguna característica de bloqueo de datos, la opción sera Sql Base ya que cuenta con buena tecnología de bloqueo .

Sistemas operativos soportados por la estación de trabajo.

Si se tiene una gran proliferación de PCs con sistema operativo D.O.S, memoria RAM menor de 1MB la opción es Netware Sql o Sql Base, pero si se tiene arriba de 1 MB de memoria RAM (2MB +) la opción puede ser Oracle Server. Si se tienen estaciones poderosas, por ejemplo con las siguientes características:

-- Procesador 80286
-- Memorias RAM mayor o igual a 2MB
-- Proliferación de sistema operativo OS/2 o miras a tener las estaciones de trabajo con OS/2.

La solución es Oracle Server o Sql Server .

Facilidad de instalación y documentación adecuada.

Este punto desprende una pregunta ¿Qué hay acerca del soporte?, ya que a medida que los manuales sean poco didácticos se necesitará más soporte.

Los planes de soporte varían tanto como los servidores. Los vendedores de servidores de bases de datos dan ilimitada asistencia de instalación y configuración, pero sin embargo no dan soporte a los diseñadores de aplicaciones. Algunos vendedores sólo dan soporte por algún tiempo, por lo que se tiene que poner énfasis a este punto y se debe tomar en cuenta qué tipo de personal se tiene, es decir, cuenta con conocimientos de SQL y del sistema operativo de la red?.

En resumen se debe tomar en cuenta que exista soporte técnico en México. Todos los productos evaluados en el capítulo tres tienen soporte en México. Pero unos tienen una infraestructura de soporte menor que otros, tal es el caso

de Sql Base. Oracle Server, Sql Server y Netware Sql, tienen una infraestructura mejor aquí en México.

Soporte del SQL

Dado que el SQL es el lenguaje estandar para bases de datos el servidor debe soportar por lo menos el SQL ANSI estandar, además proveer extensiones para mejorarlo.

¿Que hay acerca del control de concurrencia? ¿ Cuales son los métodos de bloqueo?.

Si este es un factor importante en algún ambiente y además se requiere omisión de reportes y actualizaciones al mismo tiempo se recomienda que se tome en cuenta Sql Base y Sql Server ya que este punto tiene que ver mucho con el "performance" del servidor .

¿Que hay acerca de la recuperación de datos contra fallas?

Este nivel de seguridad es esencial para aplicaciones transaccionales; tales como procesamiento en linea de operaciones bancarias, en las cuales el servidor es bombardeado constantemente para hacer actualizaciones, si este es el caso de la empresa o parecido se debe pensar seriamente en el servidor que cuente con la mejor forma de recuperación de datos automática, por lo que se debe considerar a Sql Server, Oracle Server y SqlBase seriamente, o considerar aquel que se apegue a las necesidades.

¿ Que hay acerca de la seguridad?

Dependiendo del ambiente LAN y el tipo de aplicación (datos que solo determinadas personas puedan consultar, acutailizar), se debe considerar al servidor. Tal es el caso de los bancos o instituciones militares, si se desea tener una rigurosa seguridad en el servidor se sugiere que se considere en primer lugar a Sql Server posteriormente a Oracle Server y Sql Base.

¿Que hay acerca del rendimiento?

El desempeño y tiempo de respuesta (recuperacion de datos salvar/actualizar y reporte) de los servidores de base de datos puede variar de alto a bajo, en general varios factores pueden afectar el desempeño de los servidores de bases de datos, incluyendo a la LAN y tipo de procesador del servidor y formas de bloqueo de datos.
Por lo que se recomienda tomar en cuenta estos factores para cuando se mida el rendimiento del servidor.

3. Comparar distintas alternativas en cuanto a Front-end.

Para poder tomar una decisión de que front-end será el adecuado se tienen que considerar varios puntos, los cuales mencionamos a continuación:

¿ quien diseña las aplicaciones ?.

Si todos o muchos usuarios de la empresa crean aplicaciones para su propio uso, se debe poner enfasis en herramientas de diseño que requieran poco tiempo de programación.

Si un grupo de desarrolladores se dedica a crear aplicaciones que son usadas por otros, para recuperar y actualizar, se debe considerar la flexibilidad y programabilidad primordialmente.

Otro punto que debe considerarse es si el programador puede controlar completamente el diseño de pantallas, y si este control es disponible por medio de herramientas fáciles.

¿ El front-end puede trabajar con el servidor seleccionado?

¿Cumple con el estilo del diseño de aplicaciones de la organizacion?

En algunas organizaciones sus diseños estan orientados a ventanas gráficas, si es así, se debe considerar en primera instancia a Sql Windows y si es orientada a programación tipo Dbase (orientado a caracter), se debe considerar a Dbase IV ver.1.1.

Por ejemplo si se tiene como servidor a Sql Server y en la organización se cuenta con personal que maneja dBase se debe considerar a dBase IV ver 1.1 como front-end.

Para empresas que requieren que sus aplicaciones sean orientadas a gráficos sin importar el servidor que se adquiera, nuevamente es Sql Windows y para empresas donde no se requiera que las aplicaciones sean orientadas a gráficos la opción es Advanced Revelation DBMS de Revelation Tech.

Para organizaciones que requieran de aplicaciones que corran con múltiples servidores se debe tomar en cuenta que el front-end este diseñado para trabajar en ese ambiente, considérece en primer lugar a Paradox.

Lenguajes de Programación.

¿hay interfaces de lenguajes de programación ?

Todos los servidores de bases de datos tienen esta posibilidad pero algunos tienen más posibilidad para interactuar con lenguajes tales como C, Pascal, Ada, etc.

Si se desea trabajar con algún lenguaje de programacion se debe de considerar la complejidad de sus APIs.

¿Que hay acerca de la conectividad?

Existen aplicaciones en máquinas Mainframes o Minis que por alguna razón no se pueden transportar en esos

momentos a los equipos con arquitectura cliente-servidor. Si este es el caso se debe considerar al servidor de más posibilidades de conectividad a Minis y Mainframes .Para esta situación se debe considerar a Oracle Server y posteriormente a Sql Server dependiendo que tipo de Mainframe o Mini se tenga.

4.- Seleccionar al DBA y evaluar sus conocimientos.

Con esta nueva arquitectura (cliente-servidor) vienen nuevas consecuencias. Los servidores de bases de datos requieren administradores expertos, tal como los administradores de bases de datos de Mainframes.

Los administradores deben conocer que y donde los datos serán almacenados, ellos deben monitorear y afinar el rendimiento. Deben ser capaces de diseñar bases de datos para que ellas puedan manejar apropiadamente la amplia variedad de aplicaciones de los usuarios; además en la medida que tengan mejores conocimientos de LANs y bases de datos será más fácil capacitarlos.

Así mismo el DBA escogido debe tener características de iniciativa y responsabilidad.

5.- Planear para crecer.

Este punto desprende una pregunta: ¿soporta el concepto de bases de datos distribuidas?.

Para organizaciones donde se tenga redes de diferentes tipos tales como Novell, Lan Manager, etc. y se desee en algún momento interconectar estas redes o si se tienen redes homogéneas (todas las redes Novell, o todas Lan Manager, etc) y se desea conectarlas mediante un "Backbone" u otra forma, y se desea tener la información en diferentes máquinas las cuales funcionan como servidores. Se debe buscar el servidor que ofrezca si no del todo el concepto, sí el que mejor se aproxime, en este punto se debe considerar a Sql Server.

4.2.1 Un ejemplo.

A continuación mostramos un ejemplo de la vida real .

La Secretaría de Comercio y Fomento Industrial (SECOFI) tiene interés en adquirir la arquitectura Cliente -Servidor para sus redes Novell Netware versión 2.15 ya que desea eliminar los equipos Mini con los que cuenta.

Dadas las características de la SECOFI, los directivos desean saber cuál es la mejor opción en cuanto a servidor de bases de datos (BACK-END) y FRONT-END.

Las características de esta Secretaría son las siguientes:

No. de caracte-rística.	Descripción
BACK-END.	
1	Desean tener conectividad principalmente con UNIX.
2	Planean seguir con Novell Netware con sus versiones venideras (386 +).
3	Se desea que el producto cuente con robustes.
4	Se tienen cerca de 700 PCs con sistema operativo D.O.S. Ver.3.3 Y 1MB de memoria RAM.
5	Se debe tener soporte en México.
6	Se cuenta por lo general con personal operativo nivel técnico.
7	Que el producto cuente con el ANSI estandar.
8	La actualización concurrente es muy poca.
9	Se requiere que tenga recuperación automática.
10	La información que se tiene no es confidencial, es decir que cualquier usuario puede accesarla.
11	Tiempo de respuesta aceptable.
12	Se requiere sea maduro (que garantice que se haran mejoras al producto).
13	Buen servicio.
14	Manejo de bases de datos distribuidas, ya que existe un Backbone.
15	Posibilidad de interactuar con el lenguaje "C", Windows y sus APIs.
16	Que el Front-End selecionado trabaje con el Servidor de bases de datos seleccionado.
FRONT-END:	
1	La mayoría de usuarios realizan sus aplicaciones.
2	Que el Front-end sea gráfico.

Para poder seleccionar el mejor manejador de bases de datos con arquitectura cliente-servidor para la Secretaría de Comercio Y Fomento Industrial (SECOFI), realizamos una tabla (Figura10). Como resultado de aplicar un juicio de importancia para la Secretaría y poniendo calificación de tres a cero a cada caracteristica, esto con el fin de que al final se pueda totalizar 30 puntos si resultara excelente. Para SECOFI definitivamente como front_end es SqlWindows pero tienen que escoger entre Oracle Server y Sqlbase, ya que Sqlbase no tiene posibilidades de comunicación con UNIX y si se escoge Oracle Server se tiene que extender la memoria RAM de las PCs que se utilizarán como clientes.

No. CARACTE-RISTICA.	SERVIDOR. ORACLE	SQLBASE	NETWARE SQL	SQL SERVER	VALOR
1	3	0	0	0	3
2	2	2	2	2	2
3	2	2		2	2
4		3	3		3
5	2	2	2	2	2
6	2	2	*	*	2
7	1	1	1	1	1
8	1	1	1	1	1
9	1	1	1	1	1
10	1	1	1	1	1
11	3	3		3	3
12	2				2
13	1				1
14	2	1	1	3	3
15		1		1	1
16	2	2	2	2	2
TOTAL	24	22	14	19	30

* no tutorial.

No. CARACTE-RISTICA.	FRONT-END. SQLFORM	SQLWINDOWS	XQL	DBASE IV V.1.1
1	*	*		*
2		*		

fig.10

4.3 CRITICAS.

A través de ésta tesis se estuvo en contacto con los vendedores de cada uno de los productos, así mismo ellos proporcionarón información de su producto, esta documentación, junto con todas las notas editadas por expertos en base de datos, sirvió en parte como apoyo .

En toda la documentación proporcionada y presentaciones de su producto, los representantes lo ubicaban como el mejor en comparación con otros, además hacen notorias las características que tiene su producto y que no tienen los demás o contra el que se esté comparando.

Un ejemplo de esto es la documentación proporcionada por Oracle Server donde se compara con Sql Server haciendo énfasis a sus posibilidades de comunicación con otros sistemas operativos cosa que no tiene Sql server, pero Oracle Server no hacia comparación por lógica de las buenas características que tiene Sql Server como son los trigger almacenados en la base de datos cosa que no puede hacer Oracle Server ni Sqlbase . Con respecto a los expertos que publican artículos en las revistas sobre la arquitectura cliente-servidor, algunos hacen énfasis en algún producto.

Algunos escritores son imparciales o algunos opinan contrariamente, a lo que publica cada uno de los productos.

Cada corporación dice que su producto es mejor, haciendo comparaciones con otros mediante el TP1, el cual describimos a continuación :

El TP1 fue derivado del benchmark debit_credit desarrollado por el banco de América para determinar si un sistema podría hacer 100 transacciones por segundo sobre una red bancaria con 1000 sucursales,10,000 banqueros y 10,000,000 de cuentas, cada cajero simulado en este benchmark obtendría un promedio de 100 transacciones por segundo.

La base de datos del benchmark TP1 contienen las siguientes tablas: cuentas, sucursales, cajeros y tabla histórica.

CUENTAS.

Esta tabla debe de contener al menos 100,000 registros de cuentas para todas las transacciones del TP1 por segundo. En otras palabras, si un RDBMS hace diez transacciones por segundo la tabla de cuentas debe contener al menos 1,000,000 de registros. Cada registro de cuenta es de 100 Bytes de longitud, la tabla tiene los siguientes atributo: número de cuenta, número de cliente, número de sucursal y balance de cuenta.

SUCURSAL.

La tabla contiene los siguientes atributos: número_sucursal, nombre_sucursal, balance de la sucursal, la tabla tiene una llave unica en el número_sucursal.

CAJERO.

Cada registro tiene 100 bytes de longuitud. Cada registro de cajero representa un cajero del banco en una sucursal y contiene el número_cajero, nombre_cajero, número_sucursal y balance_cajero. Tiene una llave única en el atributo número_cajero.

HISTORIA.

La tabla contiene un registro por cada transacción ejecutada contra la base de datos . Cada registro contiene 50 bytes de longuitud y contiene una sumaria de las transacciones. No tiene índices.

ESPECIFICACIONES DEL TP1.

-Escala de la base de datos.

En la base de datos se requiere de al menos 100,000 registros en la tabla de cuentas, para que se tome una transacción por segundo. Por ejemplo, si en el benchmark se dice que se realizarón 100 transacciones por segundo se deben tener 10,000,000 de registro. Estos registros deben estar en el disco duro.

-Red.

Un RDBMS que sera utilizado en red debe ser probado en red, incluyendo la máquina que se usará como servidor de datos separada del "files server" y de las estaciones de trabajo; el servidor de datos y las estaciones deben estar conectadas via la red.

-Cantidad de usuarios.

El RDBMS debe contar con un gran número de usuarios para medir realmente el rendimiento.

-Software.

Todos los componentes del Benchmark deben de estar disponibles comercialmente y deben estar soportados por los vendedores.

-Código del Benchmark.

Los usuarios deben de tener la posibilidad de poder adquirir el código fuente para verificar lo que dice el vendedor y para adaptarlo a su propio ambiente.

-Chequeo de errores.

Debe de checarse la integridad de datos ya que en el Benchmark se incluye la actualización y una inserción. Un ejemplo de integridad es que el número de cuenta no sea negativo.

-Tabla histórica.

La tabla histórica debe ser solo una tabla y no varias ya que en la vida real solo sera una.

-Cuentas.

Se deben generar números aleatorios en el TP1, la selección de cuenta usadas en una transacción deben ser generadas aleatoriamente .

-Se deben de asignar procedimientos para checar que la transacción sea válida, estos procedimientos contendran

comandos SQL para checar la integridad y comandos para la detección de errores.
-Archivo registrados (LOG).
este archivo debe estar trabajando en el disco duro, nunca debe de estar en RAM ya que puede que ocurran fallas y perderse.
-El comando Checkpoint debe de estar en "ON" y debe de ser de un periodo corto.
-Tiempo de respuesta.
El 95% de la transación debe ser realizada bajo un segundo.
-Estado estable.
Es crucial que los resultados del Benchmark sean reportados en condiciones de estado estable.

	BENCHMARK TP1 RIGUROSO.
ESCALA DE LA BASE	100,000 MINIMO.
RED	CLIENTE-SERVIDOR. O TERMINAL DE UN HOST.
NUMERO DE USUARIOS	MUCHOS.
SOFTWARE	COMERCIALMENTE DISPONIBLE.
CODIGO DEL BENCH-MARK	TOTALMENTE DISPONIBLE.
CHECADO DE ERRORES	MULTIPLE CHECADO DE ERRORES.
TABLA HISTORICA	SOLO UNA TABLA.
NUMEROS DE CUENTA	ALEATORIOS.
CARACTERISTICAS	ASIGNACION DE PROCEDIMIENTOS.
ARCHIVO REGISTRADOR (LOG).	PERMANENTE EN DISCO DURO.
CHECKPOINT	ON.
TIEMPO DE RESPUESTA	95% DE LA TRANSACCION EN 1 SEG.
ESTADO ESTABLE	SI

caracteristicas del benchmark TP1.

Oracle Server dice que su producto realiza 10.9 transacciones por segundo mientras en el benchmark de Sql Server se publican 10.5 transacciones por segundo pero, cada uno tuvo diferentes formas de aplicar el Benchmark, por ejemplo Oracle Server solo utilizó una máquina como cliente, Sql Server solo utilizó cuatro máquinas como clientes conectadas a un Backbone.

En el artículo publicado en el mes de Marzo de 1990 por la revista DBMS se analizan los resultados editados por cada producto y se dice que no son comparables, ya que el TP1 no funciona para la arquitectura cliente-servidor, además cada producto utilizó diferentes configuraciones en cuanto a disco, así mismo cada producto es afinado de tal manera que se exploten todas sus virtudes y las formas de registrar las transacciones son diferentes.

Por todo lo antes dicho el objetivo de la documentación proporcionada por los proveedores y articulos publicados en revistas dedicadas a este tema, mas bien que servir como marco para el usuario norme su criterio de elección del producto que más le conviene, sirve para confundirlo y puede suceder que no elija el más adecuado ya que él no sabe realmente quién es el que dice la verdad y quién no esta siendo honesto. Además las pruebas que se realizan a los servidores de datos no cubren todas las posibilidades de todas las empresas, por lo que es necesario que el usuario pruebe los servidores de bases de datos y el que más se apege a sus necesidades, es el mejor.

CONCLUSION.

La tecnología expuesta en ésta tesis junto con los DBMSs, evaluados dan una nueva visión para el desarrollo, ya que en la actualidad las necesidades informáticas en cuanto a comunicaciones crecen, debido a que se tienen computadores y sistemas operativos en las organizaciones de diferentes tipos, por lo que tal vez, la información requerida por algún usuario no se encuentre en un solo lugar, es decir, en un solo sistema operativo, tal vez el usuario para satisfacer su necesidad de información tenga que conocer diferentes sistemas operativos, esto ocasiona problemas. Una respuesta a este tipo de necesidades ha sido el de conectar grupos de PCs y ofrecer un gateway o un emulador de terminal, para conectarse a un computador heterogeneo, tal es el caso de un Mainframe, donde mucha de la información de la organización esta almacenada, sin embargo, ésta solución no es la mejor, debido a que cuando se emula una terminal y se desea información, digamos del Mainframe, primero se tiene que extraer a la PC ó servidor vía el "FILE TRANSFER" y esto trae como consecuencia que el usuario tenga que conocer el sistema operativo del Mainframe, ademas cuando se esta utilizando un gateway el intercambio de información es lenta.

Más aún cuando ya se tiene la informacion vía el "FILE TRANSFER", el formato de la información puede no ser compatible con la hoja electrónica o manejador de bases de datos utilizado por el usuario.

Por lo antes mencionado el usuario debe tener un acceso transparente y rápido a los datos. Lo cual como mencionamos antes, la arquitectura cliente- servidor hoy en día comienza a responder a las necesidades de los usuarios que tienen este tipo de problemática.

El SQL definitivamente se convertirá en un estándar para las bases de datos relacionales en los 90's, ya que hasta nuestros días, como se mencionó a través de la tesis, el modelo relacional en DBMSs ha cubierto un gran rango de hardware, por lo que las empresas tienen diferentes marcas de DBMSs instaladas.

Uno de los problemas de hoy es como manejar la multiplicidad de productos y datos. Por lo que el SQL permitirá que todos los DBMSs hablen el mismo lenguaje y así todos puedan intercambiar datos.

Además la heterogeneidad de hardware y software como hemos mencionado, aunado al aumento de datos emanan la necesidad de poder tener bases de datos distribuidas.

Hoy en dia la alta competividad del mercado de DBMSs con arquitectura cliente-servidor comienza a incrementarse, esto es bueno para el consumidor ya que tiene una amplia variedad de productos para escojer, pero los vendedores tratan de engañar al consumidor, por lo que es recomendable que el consumidor realize sus benchmarck y además revise los

productos y seleccione el que más se apegue a sus necesidades.

Otra de las conclusiones que se desprenden de este trabajo es que la tecnología actual de manejadores de bases de datos con arquitectura cliente-servidor son una aproximación al concepto de bases de datos distruibuidas, pero esto está apenas en proceso por que al momento, si bien han resuelto problemas que existen en la arquitectura tradicional de bases de datos para LANs no se puede suplantar a los equipos mainframes que manejan grandes volúmenes de información por redes de área local y arquitectura cliente-servidor . Tal vez con la nueva tecnología de los discos ópticos que pueden almacenar grandes cantidades de información, los procesadores 80486 y software que aproveche este hardware se puedan sustituir a los equipos Mainframes.

En cuanto a software la nueva tendencia del mercado es desarrollar DBMSs que cumplan con necesidades ahora prioritarias como son:

- Almacenamiento de imágenes y texto en la misma base de datos.
- Almacenamiento de reglas para los datos.
-almacenamiento de datos.
-Libre acceso de cualquier lenguaje de programación tales como c, cobol, pascal, etc.
-Interface con paquetes de graficación, hojas electrónicas, etc.
- Habilidad de poder correr la aplicación en máquinas ajenas a la de la base de datos.

Pienso que este es el momento para que las empresas privadas e instituciones publicas empiezen a considerar ésta tecnología (arquitectura cliente-servidor y bases de datos distribuidas) para prepararse a cambiar los Mainframes por LANs, ya que en esta década se tendrá lo mejor de los dos mundos(la gran cantidad de interfaces fáciles de usar de las PCs, combinado con el manejo de datos del mundo Mainframe).

La utilidad de ésta tesis podría ser en algún momento, servir como guia a compradores potenciales de DBMSs con arquitecura cliente-servidor, además pienso que puede servir como apoyo a mis compañeros de carrera que tomen la materia de bases de datos.

En lo particular a mi me sirvió esta investigación para conocer más sobre bases de datos, asi mismo para poder desarrollar más rapido y eficientemente mi trabajo cotidiano.

Dado que los 90's estarán gobernados por la arquitectura cliente-servidor para LANs, pienso que alguna futura tesis podría ser un estudio a fondo sobre bases de datos distribuidas y las reglas que las gobernarán.

GLOSARIO.

Amplificador
Un componente electrónico que levantara la intensidad ó amplitud de lo transmitido, usualmente signos análogos; Funcionalmente equivalentes a un repetidor en transmisión digital.

Ancho de banda
El rango de frecuencias entre dos límites definidos, expresado es Hertz. El ancho de banda determina el porcentaje de información que puede ser transmitida.

Algoritmo
Un conjunto finito de pasos bien definidos para la solución de un problema.

APIs
Conjunto de subrutinas que permiten a uno ó varios lenguajes de programación interactuar con algún otro programa ejecutable.

Banda base
Referencia a señales en su forma original y no cambiadas por modulación.

Banda ancha
Se refiere a los medios que pueden soportar un amplio rango de frecuencias electromagnéticas moduladas.

Backbone
Una trayectoria de comunicación compartida que sirve a múltiples usuarios vía multiplexión de puntos de salto designados.

Benchmark
Es un método para medir el rendimiento de un sistema en un ambiente controlado, usando una metodología estandar. Los benchmark para medir el rendimiento en manejadores de bases de datos más usados son: el Benchmark de DeWitt y el TP1.

Bios
Parte del sistema operativo que efectúa la comunicación con el hardware del computador. La mayor parte del BIOS se encuentra en la ROM (memoria de solo lectura), el resto se carga del disco magnético.

Boot-Strap
Una rutina de entrada en la cual son usadas operaciones de computador simples para cargar instrucciones que al terminar provocan a más instrucciones que serán cargadas hasta que el programa de computadora este almacenado completamente.

Buffer

Dispositivo de almacenamiento usado para compensación de una diferencia en porcentaje de flujo de datos, ó tiempo de ocurrencia de eventos, cuando se estan transmitiendo datos de un dispositivo a otro.

Circuito Integrado

Un circuito integrado es un cristal pequeño semiconductor de silicio denominado una pastilla que contiene componentes electrónicos, tales como resistencias, transistores, diodos y condensadores. Los diversos componentes se interconectan dentro de la pastilla.

Conectividad

La habilidad de enlazar diferentes piezas de hardware (Macintoshes, Microcomputadoras, Minicomputadoras y Mainframes) y software en un ambiente de red, donde los recursos (aplicaciones, software, etc) son compartidos.

Coprocesador

Una unidad lógica central (cpu) adicional la cual ejecuta tareas específicas mientras la principal ejecuta tareas primarias, este procesador ejecuta tareas, tales como operaciones matemáticas.

Cursor

Un indicador de la posición empleado por el servidor de bases de datos para indicar la posición en la cual se encuentran los datos resultantes del Query.

Database server

(servidor de bases de datos) también llamado Back-end en la arquitectura cliente-servidor. Consiste en proporcionar acceso a archivos de bases de datos compartidas, en una red.

dBase

Es una serie de programas diseñados para el manejo y administración de archivos, estos archivos son manejados por estructuras tipo tabla. Estos programas estan comercializados por la compañia Ashton-Tate.

Disco virtual

Dispositivo de almacenamiento simulado y que no esta implementado físicamente en el cual las direcciones simuladas son mapeadas dentro de las direcciones reales.

En línea

Conectado a un computador, así que los datos pueden pasar para ó del computador con la intervención humana.

File server
Dispositivo de red que proporciona acceso a programas y archivos campartidos.

Frecuencia
Una expresión de como frecuentemente una forma de onda periodica ó señal se regenera asi misma a una amplitud dada.

Gateway
Ejecuta operaciones de conversión de protocolos para que se interconecten dos redes ó dispositivos incompatibles.

Hardware
El equipo usado sobre una red (tales como computadoras, impresoras, cable, tarjetas de red,etc).

Interrupción 21h
Es una señal al hardware que le indica al CPU que pare temporalmente mientras ejecuta funciones de entrada y salida, por ejemplo: desplegado, entrada y salida de caracteres, funciones de impresión, funciones de disco y funciones de manejo de archivos.

IEEE
"institute of Electrical and Electronics Engineer" Un grupo involucrado en recomendaciones estandar para el campo de la computación y comunicaciones.

Ipx
"Internetwork Packet Exchange" Es una implementación del protocolo XNS (xerox Networking Systems) que consiste en enviar los paquetes del origen al destino.

Spx
"Sequenced Packet Exchange" Debido a que Ipx solo envía los paquetes, SPX revisa que todos los paquetes estén completos y que llegen en orden.

Interface
Límite entre dos programas a través de la cual todas las señales que pasen son cuidadosamente definidas.

Hz
Unidad de frecuencias electromagnéticas igual a un ciclo por segundo.

Khz
Kilohertz.

Mainframe
Término empleado para referirse a computadores grandes que requieren de algún medio ambiente especial (aire acondicionado, etc). Su capacidad de conexión de terminales es de miles.

Medio de transmisión
Cualquier material que es, ó puede ser usado para la propagación de señales de un punto a otro.

Minicomputadora
Computadora de tamaño mediano; Las minicomputadoras se encuentran en un punto intermedio entre las microcomputadoras y mainframes, y su capacidad de conexión de terminales es entre unas cuantas hasta varios cientos.

Modem
Una contracción de modulación y demodulación. Dispositivo de conversión instalado en parejas en cada terminación de la línea de comunicación.

Modulación
Mezcla de una señal con la portadora. La modulación es el proceso de entremezclar una señal de voz o una serie de datos con una portadora, para su transmisión a través de la red.

Named-Pipes
Es un protocolo de más alto nivel que el Netbios, una función del Named-pipes es equivalente a una o varias llamadas de Netbios, por medio de Named-pipes se maneja el establecimiento y terminación de una sesión entre dos usuarios de la red o entre el usuario y el servidor.

Net3
Emulador del netbios desarrollado por novell, es el que se encarga de las administración de las comunicaciones, establece y termina la sesión de comunicación entre dos usuarios de la red, ó entre un usuario y el file server.

Netbios
"Network Basic Input Output system" Fue desarrollado por IBM, originalmente el netbios fue proporcionado sobre la tarjeta de red en si misma. Netbios se encarga de iniciar y terminar la sesión entre dos usuarios de la red, ó entre un usuario y el servidor de la red.

Os/2
Es un sistema operativo para microcomputadoras que permite tomar todas las ventajas del procesador 80286, además permite ejecutar más de una aplicación al mismo tiempo.

Paquete

Un paquete es la unidad discreta más pequeña, o pedazo de datos que un protocolo puede manejar. Los mensajes que se van a enviar a través de una red son divididos en paquetes. Parte de cada paquete contiene los datos a ser enviados y parte contiene información con respecto al mismo paquete, incluyendo la dirección destino del paquete.

Pc

"Personal Computer" Computadora de tamaño pequeño (en comparación con mainframes y minicomputadoras) o de escritorio, estos computadores son de uso generalizado.

Plataforma

Referencia al sistema operativo que funciona en una máquina computadora tal como OS/2, DOS, Unix, etc.

Performance

Referencia al desempeño de algún sistema.

Protocolo

Son los lenguajes de la red. Son un conjunto de reglas por medio de las cuales se establece, mantiene y controla la comunicación

Lenguaje de alto nivel

Lenguaje de programación orientado hacia la resolución de problemas, Las instrucciones se dan a la computadora usando letras convenientes, símbolos o texto parecido al inglés, en lugar de usar el código de entrada/salida que entiende la computadora.

Lenguaje de tercera generación

Lenguaje orientado a procedimientos, es decir el usuario indica cuales datos quiere y como debe obtenerlos.

Lenguaje de cuarta generación

Lenguaje no procedural, es decir el usuario indica que datos desea sin decir como obtenerlos.

RAM

Una técnica de almacenamiento en la cual el tiempo requerido para obtener datos es independiente de la localización, este tipo de memoria es volatil.

Repetidor

En transmisión analógica, equipo que recibe una serie de señales, las amplifica y reenvía a otro dispositivo. En transmisión digital, equipo que recibe una serie de señales, las reconstruye y entonces amplifica y reenvía.

Rollback

Instrucción de bases de datos que deshace todos los cambios a la base de datos efectuados por una transaccion que aún no ha sido terminada.

Ruteador
Los ruteadores pueden enviar paquetes de datos sobre diferentes trayectorias en una red.

Requestor
El Netware requestor OS/2 permite a estaciones con sistema operativo OS/2 iniciar una sesión con el file serve, así mismo hacer uso de utilerias que corren bajo el sistema operativo OS/2.

Sistema abierto
Capacidad del dispositivo o computadora de poder comunicarse con cualquier otro dispositivo para el intercambio de información.

Software
Instrucciones de computador que ejecutan funciones comunes para todos los usuarios, como también aplicaciones específicas para necesidades de usuarios particulares.

Terminal tonta
Dispositivo capaz de enviar y/o recibir información sobre un canal de comunicación.

Topología
El arreglo físico ó lógico de estaciones de la red en relación de una con otra.

Tarjeta
Dispositivo que va instalado dentro de una microcomputadora y según su especificación, cada tarjeta determina la forma de conexión de la red.

Trigger
Son procedimientos almacenados que son ejecutados automáticamente cuando se hace un intento de modificación a los datos que ellos protegen.

Tupla
Referencia a un renglón de una tabla (relación) definida en una base de datos con modelo relacional.

Unix
Un sistema operativo multiusuario desarrollado por los laboratorios Bell.

BIBLIOGRAFIA.

FACULTAD DE INGENIERIA U.N.A.M.
introduccion a redes locales de micro computadoras.
División de Educación Continua.
1989.

ALABAU MUÑOS ANTONIO
teleinformática y redes de computadoras.
Editorial: Publicaciones Macombo.
2da Edición México.
1987.

CARDENAS ALFONSO F.
sistemas de administración de bancos de datos.
Editorial:LIMUSA.
1a Edicion México.
1983.

DATE C.J
introducción a los sistemas de bases de datos.
Editorial:SITESA.
3a Edicion México.
1989.

GONZALEZ SAINZ NESTOR.
comunicaciones y redes de procesamiento de datos.
Editorial: Mc Grawhill
1_a edicion Colombia.
1987.

HENRY F. KORTH & ABRAHAM SILBERSCHATZ.
fundamentos de bases de datos.
Editorial McGrawhill.
1a edicion México.
1987.

CHAPA VERGARA SERGIO.
apuntes materia: Bases de Datos.
9no semestre.
U.N.A.M.
1989.

DIAZ LLANO EMILIANO
apuntes materia: teleprocesos.
8vo semestre.
U.N.A.M.
1988.
MILLER MARK A.
LAN troubleshooting hanbook.
Editorial: M&T Books.
1a edicion Estados Unidos.
1989.

Oracle Server
Manuales de instalación y de usuario.
Oracle Corporation.
1989.

SqlBase
Manuales de instalación y de usuario.
Gupta Technologies
1989.

Netware Sql.
Manuales de instalación y de usuario.
Novell Inc.
Novell Development products Division
1989.

Sql Server.
Manuales de instalación y de usuario.
Microsoft Corporation,Ashton-Tate,Sybase, Inc.
1989.

REFERENCIAS.

KYLE HOMPHRIES.
working with XQL.
Revista: Netware Technical Journal.
Enero 1989.

NOVELLCO DE MEXICO.
seminario en redes.
Septiembre 1989.

NOVELLCO DE MÉXICO.
redes locales de México.
seminarios en redes.
Agosto 1989.

COMPER S.A.
curso TRANSACT-SQL.
Febrero 1990.

MADRON THOMAS W.
local area networks the second generation.
Editorial: Thon Wiley & Sons,Inc.
1a edicion 1988.
Impreso en Estados Unidos.

CHOUINARD PAUL.
first normal form: don't overdo it.
Revista: Database Programming & Design.
Febrero 1989.

PASCAL FABIAN.
a brave new world.?
Revista: Byte
Septiembre 1989.

WAI GILBERT
take your pick
Revista: Byte
Julio 1989.

BRUCE J.WALKER and GERALD JOPOPEK.
a transparent environment.
Revista: Byte.
Julio 1989.

DEVIS RALPH.
sharing the wealth.
Revista: Byte.
Septiembre 1989.

FINKELSTEIN RICHARD
Moving toward distributed database.
Revista: DBMS.
Marzo 1989.

ASSOCIATES MARKETING
3+ open Lan Manager applications.
Febrero 24,1989.

FINKELSTEIN RICHARD
In front of the server.
Revista: Pc Tech Journal.
Marzo, 1989.

DOWGIALLO ED.
The database server games begin.
Revista : Lan Technology.
Noviembre 1989.

DOWGIALLO ED.
The database server games.
Revista : Lan Technology.
Diciembre 1989.

STREHLO KEVIIN.
Why not TPI.
Revista: DBMS.
Noviembre 1989.

FINKELSTEIN RICHARD Y PASCAL FABIAN.
SQL database management systems.
Revista : Byte.
Enero 1986.

EDELSTEIN HERB
Auditing the auditors.
Revista : DBMS.
Marzo 1990.

VINZANT DAVID R.
Novell is providing named pipes support and other aids for running SQL Server on Netware.
reporte tecnico.
Septiembre/Octubre 1989.

HORWITZ MIKE and GOLDBERG CHERYL J.
Back seat driver.
Revista : DBMS.
Junio 1989.

ASHTON-TATE/ MICROSOFT.
SQL SERVER the database engine for os/2 systems.
reporte técnico.
1989.

FINKELSTEIN RICHARD
moving toward distributed database.
Revista: DBMS.
Marzo 1989.

ASSOCIATES MARKETING
3+ open Lan Manager applications.
Febrero 24,1989.

FINKELSTEIN RICHARD
in front of the server.
Revista: Pc Tech Journal.
Marzo, 1989.

DOWGIALLO ED.
the database server games begin.
Revista : Lan Technology.
Noviembre 1989.

DOWGIALLO ED.
the database server games.
Revista : Lan Technology.
Diciembre 1989.

STREHLO KEVIIN.
why not TP1.
Revista: DBMS.
Noviembre 1989.

FINKELSTEIN RICHARD Y PASCAL FABIAN.
SQL database management systems.
Revista : Byte.
Enero 1988.

EDELSTEIN HERB
auditing the auditors.
Revista : DBMS.
Marzo 1990.

VINZANT DAVID R.
novell is providing named pipes suport and other aids for running SQL Server on Netware.
reporte tecnico.
Septiembre/Octubre 1989.

HORWIEZ MIKE and GOLDBERG CHERYL J.
back seat driver.
Revista : DBMS.
Junio 1989.

ASHTON-TATE/ MICROSOFT.
SQL SERVER the database engine for os/2 systems.
reporte técnico.
1989.

INMON WILLIAM H.
optimizing performance with denormalization.
Revista : Database Programming & design.
primera edicion 1987.

TUCKER MICHAEL JAY.
the inevitable merging of SQL.
Revista: Unix World.
Febrero 1990.

SCHNAIDT PATRICIA
client- server databases drive lan DBMSs home.
Revista: Lan Magazine
Mayo 1989.

LISKIN MIRIAM
the spotlight turns to database servers.
Revista : Personal Computing .
Noviembre 1989.

FINKELSTEIN RICHARD.
client/server computing - the best of two worlds.
Revista:Connect.
Verano 1989.

HINDIN ERIC M.
sharing the load: client/server computing.
Revista : Data Communications.
Marzo 21 1989.

VOLTI GARY R.
installing SQL SERVER 1.0 under Novell Netware.
consulting services.
Agosto 1989.

WFORKOWSKI GABRIELLE Y KULL DAVID.
the optimizer: invisible hand of the DBMS.
Revista: Database Programming & Desing.
Septiembre 1988.

WASIOLEK ERIC W.
distributed applications are the next step in lan connectivity.
Revista: Lan Magazine.
Septiembre 1988.

EDELSTEIN HERBERT A.
front-end tools.
revista: Pc Tech Journal.
Marzo 1989.

LIBERMAN DANIEL R.
codd's 12 rules: a method for dbms evaluation.
Revista:Database Programming & Design.
Diciembre 1988.

BUZZARD JAMES.
front-end friedndly.
Revista: Data Base Advisor.
Abril 1990.

FINKELSTEIN RICHARD.
the SQL.
Revista: Byte.
Mayo 1990.

THE COMMITTEE FOR ADVANCED DBMS FUNCTION.
third-generation data base system manifesto.
Editado: Electronics Research Laboratory. College of Engineering.
University of California, Berkelev,CA 94720.

ROSE CHARLES.
client-server computing for the sophisticated lan.
Revista: Lan Technology.
Diciembre 1989.

CORRIGAN PATRICK H. and GUY AISLING.
building local area networks with Novell's Netware.
editorial M&T Books.
1a Edicion 1989.

ATTHIAS JARKE and YANNIS VASSILION
framework for choosing a database query languaje
Revista: Computing survey,Vol.17, No. 3.
septiembre 1895.
SRULOVICZ PETER.
Ethernet Networks:fact and fiction
Reporte Tecnico presentado por LANsPlus, Canada Inc.
Noviembre, 1989.
FINKELSTEIN RICHARD.
lingua franca for databases
Revista: Pc Tech Journal
Diciembre 1987.

MARK L. VAN NAME and BILL CATCHINGS
SQL: a database language sequel to Dbase.
Revista: Byte Edicion Especial IBM
1989.