Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería en Ciencias y Sistemas

SISTEMAS DE BASES DE DATOS DE ALTA DISPONIBILIDAD

Gabriel Fernando Castillo Contreras

Asesorado por el Ing. Otto Amilcar Rodríguez Acosta

Guatemala, abril de 2006

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

SISTEMAS DE BASES DE DATOS DE ALTA DISPONIBILIDAD

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA

FACULTAD DE INGENIERÍA

POR

GABRIEL FERNANDO CASTILLO CONTRERAS

ASESORADO POR EL ING. OTTO AMILCAR RODRIGUEZ ACOSTA

AL CONFERÍRSELE EL TÍTULO DE

INGENIERO EN CIENCIAS Y SISTEMAS

GUATEMALA, ABRIL DE 2006

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

NÓMINA DE JUNTA DIRECTIVA

DECANO	Ing. Murphy Olympo Paiz Recinos
VOCAL I	
VOCAL II	Lic. Amahán Sánchez Álvarez
VOCAL III	Ing. Julio David Galicia Celada
VOCAL IV	Br. Kenneth Issur Estrada Ruiz
VOCAL V	Br. Elisa Yazminda Vides Leiva
SECRETARIA	Inga. Marcia Ivonne Véliz Vargas

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANO	Ing. Herbert René Miranda Barrios
EXAMINADOR	Inga. Marlene Morales Massella
EXAMINADOR	Ing. César Augusto Fernández Cáceres
EXAMINADOR	Ing. Carlos Alfredo Azurdia Morales
SECRETARIA	Inga. Gilda Marina Castellanos de Illescas

HONORABLE TRIBUNAL EXAMINADOR

Cumpliendo con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

SISTEMAS DE BASES DE DATOS DE ALTA DISPONIBILIDAD

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería en Ciencias y Sistemas, en enero de 2005.

Gabriel Fernando Castillo Contreras

DEDICATORIA A:

Dios Todopoderoso por ser la imagen espiritual de poder supremo, enseñado por la formación cristiana de mis padres y a la virgen María por constituir mi guía de confianza.

A mis padres Lisbeth y Carlos Roberto que con su amor lograron darme vida y que con su formación ejemplar brindaron en mí la confianza que hoy día se consolida en el final de esta meta, gracias los amo.

A mis hermanos Que me han acompañado a lo largo de mi carrera y con su apoyo incondicional me han dado fuerza para continuar.

A mi esposa Que con su paciencia y apoyo incondicional me motiva a continuar mi formación profesional.

A mi hija Que con su corta edad ya aspira lograr grandes metas por eso lucha por tus ideales y lograrás la felicidad, en mí encontrarás un apoyo incondicional. Te quiero mucho.

Todas las personas que directa o indirectamente contribuyeron para el logro de esta meta, muchas gracias.

ÍNDICE GENERAL

2. BASE DE DATOS EN ESPERA

3. REPLICACIÓN DE DATOS COMO MÉTODO DE ALTA DISPONIBILIDAD

ÍNDICE ILUSTRACIONES

FIGURAS

TABLAS

GLOSARIO

Base de datos Colección de datos relacionados, almacenados de tal manera que su contenido pueda ser fácilmente accedido, administrado y actualizado.

Bitácora Estructura utilizada por el DBMS para grabar las modificaciones de la base de datos, ella contiene información de la transacción y de los datos modificados, información necesaria para realizar recuperación de fallos de sistema.

Bit Elemento discreto de información por definición un dígito binario.

Bugs Fallas en programas de computación debido a errores humanos.

Diccionario de datos Conjunto de tablas utilizadas por un DBMS que contiene meta datos sobre la estructura de la base de datos.

DBMS Programa de computadora que se encarga de administrar los datos almacenados en una base de datos que se encuentra en una computadora.

Disco hot plug Disco que se puede instalar o remover a un arreglo de discos mientras que la computadora que lo posee esta encendida y en funcionamiento.

Cluster Arquitectura formada por dos o más computadoras y un dispositivo de almacenamiento de discos protegidos, cada computadora es provista con todos los componentes del sistema duplicados, si un componente falla este puede ser reemplazado de manera automática o manual por su duplicado.

Failover Proceso por el cual los servicios que están corriendo en una computadora que fallo son migrados a una computadora en operación permitiendo la continuidad del sistema.

FDDI Interfaz de datos distribuida en fibras es una tecnología LAN de muy alta velocidad (100 millones de bits por segundo o más).

Gigabyte 1,073,741,824 *bytes* también se referencia como Gb.

LAN: Red de área local que conecta entre sí computadoras empleado medios de comunicación de alta velocidad optimizados para áreas geográficas pequeñas.

Megabyte 1,048,576 *bytes,* también se referencia como Mb.

Petabyte 1024 *terabytes,* también se referencia como Pb.

RAID	Un conjunto de estándares para la organización de diferentes maneras de discos que permiten una rápida recuperación en el momento de la falla de un disco.
SMP	Configuración de acceso uniforme de memoria, en la cual un sistema de más de un procesador puede acceder a la memoria principal a la misma velocidad.
SCSI	Interfase utilizada para la conexión de discos y otros dispositivos en una computadora.
SQL	Lenguaje que se utiliza para comunicarse con un DBMS relacional.
Tabla	Organización de datos en forma de filas y columnas.
Terabyte	1024 *gigabytes*, también se referencia como Tb.
TMEF	Tiempo medio entre fallas. La cantidad de tiempo promedio que un componente puede tener entre diferentes fallas.
TMRF	Tiempo medio de reparación de fallas. El tiempo promedio que toma reparar las fallas en un sistema para nuevamente colocarlo en operación.

RESUMEN

El intervalo de tiempo entre caídas del sistema y la reparación de la falla que provocó esa caída son variables que determinan la disponibilidad de un sistema, para ello es necesario definir mas claramente un métrico indispensable para determinar el nivel de disponibilidad de un sistema.

Para cualquier sistema de base de datos es importante lograr un nivel de alta disponibilidad, es decir estar disponible el 99.98% del tiempo de operación. Para lograr esto se toma ventaja de mecanismos y características de los diversos manejadores de bases de datos, tales como: base de datos en espera, replicación y base de datos en paralelo.

Base de datos en espera es una réplica creada desde un respaldo de una base de datos primaria, esto aplicando la información de bitácora de la base de datos primaria sobre la base de datos en espera, estas dos bases de datos pueden estar sincronizadas. En el momento de una falla se inicia un proceso de *failover* para convertirla en base de datos primaria.

La replicación para manejar o proveer alta disponibilidad, no es más que replicar las transacciones de una base de datos primaria donde se encuentran conectados los usuarios en el sistema hacia una base de datos secundaria.

Una base de datos en paralelo, esta conformada por un *cluster* y varios nodos a en los cuales están conectados los usuarios, en el momento de una falla la base de datos sigue operando en el cluster y los usuarios pueden seguir trabajando conectados en otro nodo.

OBJETIVOS

- **General**

Realizar un estudio de las arquitecturas de sistemas de base de datos de alta disponibilidad, para proveer un amplio criterio de selección basado en los requerimientos de disponibilidad.

- **Específicos**

1. Definir el concepto de disponibilidad y los diferentes niveles.

2. Definir que es un sistema de base de datos de alta disponibilidad.

3. Explicar cada una de las arquitecturas de sistemas de base de datos de alta disponibilidad: base de datos en espera, replicación, base de datos en paralelo.

4. Explicar los beneficios de cada una de las arquitecturas de sistemas de base de datos de alta disponibilidad.

INTRODUCCIÓN

Las bases de datos han evolucionado de decenas de *megabytes* a *terabytes* y aún hasta *petabytes*. Agregando complejidad a la administración de estas enormes bases de datos, la restricción de largos períodos de operación que incluyan requerimientos de 24 horas diarias durante 7 días a la semana. Es bastante difícil proveer una administración apropiada para este tipo de bases de datos, tareas de mantenimiento, afinación y respaldo se convierten en imposibles; si la base de datos no puede sacarse de servicio por un momento, incluso algunas de estas tareas pueden tomar varios días. Una posible solución para este enigma, es la adopción de una tecnología de alta disponibilidad para sistemas de base de datos.

En el capítulo uno se define la disponibilidad y como medirla, qué son los tiempos de caída cuáles son sus causas y los costos en los que repercuten, se examinarán los diversos niveles de disponibilidad; en la última sección del capítulo, se profundiza en la relación alta disponibilidad y sistema de base de datos, definiendo qué es un sistema de base de datos de alta disponibilidad.

En los siguientes capítulos se examinan las soluciones para sistemas de base de datos de alta disponibilidad tales como base de datos en espera, utilización de la replicación y base de datos en paralelo.

En el capítulo dos, se define qué es una base de datos en espera, sus modos de configuración, la respuesta a fallos, el ciclo de vida, las ventajas y desventajas y consideraciones en cuanto a las consecuencias en la administración.

En el capítulo tres, se evalúa el uso de la replicación como un método de alta disponibilidad, los diferentes tipos de replicación y como pueden ser utilizados, así como las ventajas y desventajas de la replicación como método de alta disponibilidad.

Por último en el capítulo cuatro, se estudia lo que se denomina base de datos en paralelo, arquitectura que posee los niveles mas altos de disponibilidad, su complejidad reside en la elección del manejador de bases de datos y componentes de hardware para su implementación; ha sido diseñada específicamente para cubrir requerimientos de alta disponibilidad de una organización.

La elección de un sistema de base de datos de alta disponibilidad, esta en función de los requerimientos de funcionamiento de ese sistema en el tiempo, y de los recursos con los que se cuente para invertir en la compra de un DBMS con características para lograr un nivel de alta disponibilidad, este trabajo de tesis examina estas características y mecanismos para el logro de un sistema de base de datos de alta disponibilidad.

Este trabajo requiere del lector cierta familiaridad con los conceptos de bases de datos, SQL y arquitectura de computadores.

1. DISPONIBILIDAD

1.1 Medición de disponibilidad

Cuando se discuten los requerimientos de disponibilidad con un usuario o jefe de proyectos, invariablemente solicitan el 100 por ciento de disponibilidad: “Nuestro proyecto es tan importante que no podemos tener ningún tiempo de caída en el sistema”. Pero este requerimiento cambia cuando el jefe de proyectos descubre cual es el costo de ese 100 por ciento de disponibilidad. Entonces se convierte en un asunto de dinero, y como parte de un proceso de negociación.

En la Tabla 1, se observa que para muchas aplicaciones, el 99 por ciento de tiempo de operación es adecuado. Si el sistema tiene un promedio de una hora y media de tiempo de caída por semana, lo que puede ser satisfactorio. Por su puesto, esto depende de cuando suceda esa hora y media de tiempo de caída. Si esto sucede entre las 2:00 a.m. y 3:30 a.m. de un día domingo es más tolerable que si ocurre un día martes entre 11:00 a.m. y 12:30 p.m., o que esa hora y media este distribuida entre los días de la semana por períodos de 15 a 20 minutos.

Tabla I. Medición de la disponibilidad

Porcentaje de tiempo operación	Porcentaje de tiempo no operación o caídas	Tiempo de no operación por año	Tiempo de no operación por semana
98%	2%	7.3 días	3 horas, 22 minutos
99%	1%	3.65 días	1 hora, 41 minutos
99.8%	0.2%	17 horas, 30 minutos	20 minutos, 10 segundos
99.9%	0.1%	8 horas, 45 minutos	10 minutos, 5 segundos
99.99%	0.01%	52.5 minutos	1 minuto
99.999%	0.001%	5.25 minutos	6 segundos
99.9999%	0.0001%	31.5 segundos	0.6 segundos

Un punto de negociación, es el tiempo que se requiere un 100 por ciento de operación. Si esto únicamente es necesario por unas pocas horas al día, entonces la meta es completamente alcanzable. Por ejemplo, puede requerir el 100 por ciento de tiempo de operación durante las horas de producción, pero no durante el resto del día. Por otro lado, si se requiere un 100 por ciento de operación 7 días a la semana las 24 horas del día los 365 días del año, los costos se convierten en prohibitivos. Estos costos únicamente las aplicaciones más rentables y las empresas o corporaciones grandes pueden considerar.

Conforme se mueve progresivamente a niveles más altos de disponibilidad, los costos se incrementan muy rápidamente. Por ejemplo consideremos un servidor al que llamaremos “Alfa” del cual no se han tomado medidas especiales, ha excepción de que posee discos en espejo y respaldo, con un 99% de disponibilidad.

Si juntamos este servidor con otro configurado de manera idéntica al que llamaremos “Beta”, con la característica de que puede tomar el control del servidor “Alfa” cuando falle, y este servidor también ofrece el 99% de disponibilidad, entonces teóricamente, se puede lograr un porcentaje combinado de disponibilidad del 99.99%.

Matemáticamente, se multiplica el tiempo de caída de “Alfa” (1 %) por el tiempo de operación de “Beta” (99%); “Beta” únicamente es utilizado durante el 1 % de tiempo de caída de “Alfa”. El resultado es 0.99%, agregamos este porcentaje al original 99% y obtenemos el valor teórico de disponibilidad para este par combinado de 99.99%.

1.1.1 Definición de tiempo de caída

El definir tiempo de caída puede variar de una manera simple a compleja. Una definición simple, esta dada en términos del fallo de componentes del sistema, tales como el servidor, discos, la red, el sistema operativo, aplicaciones o la base de datos.

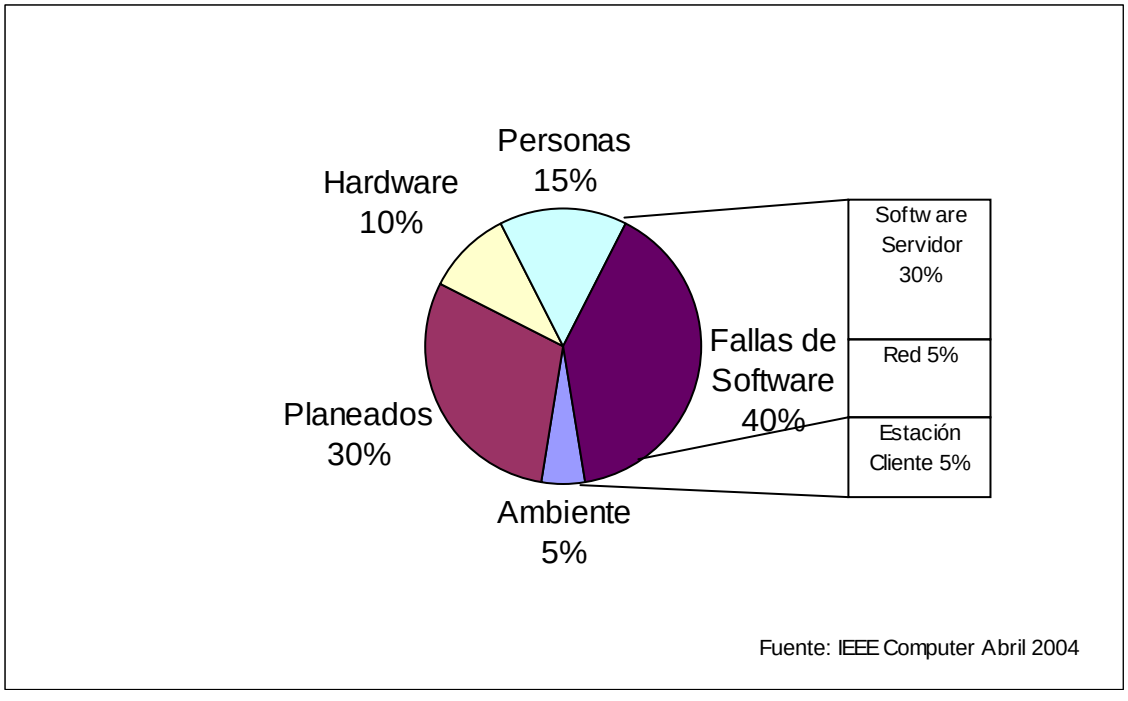
Definiciones complejas, pueden incluir bajo rendimiento en el servidor o la red, la imposibilidad de restaurar respaldos, o simplemente la no accesibilidad a los datos. El sistema provee un servicio a los usuarios para permitirles realizar un trabajo en una manera eficiente y de manera oportuna. Cuando las circunstancias impiden al usuario de realizar este trabajo, sin tener en cuenta la razón, el sistema esta abajo. Por lo que se dice que el sistema sufre de un tiempo de caída, entonces tiempo de caída se define como: el tiempo que los usuarios tienen que esperar para que el sistema se encuentre funcionando nuevamente.

1.1.2 Causas del tiempo de caída

En la figura 1, se examinan varias causas de los tiempos de caída. Una de las regiones más grandes de la gráfica es para los tiempos planeados de caída. Este es también uno de los segmentos que mas fácil se puede reducir.

Los tiempos planeados de caída, son eventos programados, usualmente en la tarde o noche, cuando los administradores de sistema agregan hardware a sus sistemas, actualizan sistemas operativos u otro software crítico como el manejador de base de datos, o realizan una tarea administrativa de mantenimiento sobre el sistema. Algunas veces estos tiempos de caída planeados consisten únicamente en realizar un reinicio preventivo, para limpiar bitácoras, directorios temporales, y memoria.

Figura 1. Causas de tiempo de caída



Hoy en día, la mayoría de estos eventos son llevados a cabo con el sistema arriba, discos pueden ser agregados en arreglos de disco *Hot Plug* sin necesidad de interrumpir algún servicio. Muchas aplicaciones críticas pueden ser actualizadas sin interrumpir el servicio.

En un ambiente de *failover*, el cual consta de un *cluster*, dos servidores uno de los cuales se encuentra en operación y el segundo en espera de fallos, se puede actualizar uno de los servidores mientras el otro esta operando, cuando se ocurre una falla con el primero se realiza un intercambio, y se convierte en servidor primario el servidor recién actualizado, mientras se actualiza el segundo. La única interrupción de servicio se da durante el intercambio de servicios, al cual se le llama *failover*.

El factor humano es otra de las mayores causas de los tiempos de caída. Las personas causan tiempos de caída por dos razones íntimamente relacionadas. Como primera razón es que ellos algunas veces cometen errores por descuido o torpeza.

La segunda razón es que las personas causan tiempos de caída debido a que no siempre comprenden completamente la manera en que el sistema opera. La mejor manera para combatir los tiempos de caída ocasionadas por personas es a través de una combinación de educación y diseño simple del sistema. Enviando al personal a capacitación que los mantenga actualizados en las tecnologías, y teniendo una sólida documentación al día, puede reducir la cantidad de caídas de sistema por esta causa.

Posiblemente la región de la gráfica que más sorpresa causa es el hardware. El hardware causa solamente el 10 por ciento de las caídas de sistema. Esto quiere decir que el mejor arreglo RAID de discos en el mundo y las mejores redes redundantes, lo previenen únicamente de este 10 por ciento de tiempos de caída.

De hecho, además de fallas de discos y red, los problemas de hardware también incluyen a la unidad central de procesamiento y fallas de memoria, pérdida de fuentes de poder, y sistemas internos de enfriamiento.

La causa más común del tiempo de caída de un sistema es probablemente las fallas de software. En total, el software es responsable del 40 por ciento de los tiempos de caída de un sistema. Los *bugs* de software son probablemente las fallas más difíciles de corregir. Como el hardware se convierte más confiable, y existen métodos para reducir los tiempos planeados de caída, sus porcentajes decrecen, mientras el porcentaje de problemas atribuidos a causas de software incrementan. Esto debido a que el software se convierte más complejo, y sus problemas se convierten más frecuentes. Por supuesto, el desarrollo de nuevas técnicas de depuración ha tenido avances, por lo que los problemas de software son menos prevalecientes.

1.1.3 Qué es disponibilidad

En su nivel mas simple, disponibilidad ya sea alta, baja, o en un término medio, es una medida de tiempo que el sistema esta funcionando normalmente. De lo cual se puede tener una ecuación para calcular la disponibilidad:

$$D = \frac{TMEF}{TMEF + TMRF}$$

Donde D es el grado de disponibilidad expresado como porcentaje, $TMEF$ es el tiempo medio entre fallas, y $TMRF$ es el máximo tiempo para reparar o resolver un problema particular.

De lo que se puede observar:

1. Cuando *TMRF* se aproxime a cero, *D* se incrementa y tiende a 100 por ciento.
2. Mientras *TMEF* sea mas largo, *TMRF* tiene menos impacto sobre *D*.

Por ejemplo, si un sistema tiene un *TMEF* de 100,000 horas, y un *TMRF* de 1 hora, se tiene una impresionante tasa de nivel de disponibilidad de 100,000/100,001, o 99.999 por ciento. Si reducimos el *TMRF* a un valor de 6 minutos, es decir 1/10 de hora, la disponibilidad adquiere un 9 extra, para alcanzar el 99.9999 por ciento. Pero para lograr este nivel de disponibilidad con 6 minutos de tiempo de caída, se necesita un componente con un tiempo medio entre fallas de uno cada 100,000 horas, es decir una falla cada 11.4 años.

Explicado de otra manera, para lograr el 99.9999 por ciento de disponibilidad están permitidos solamente 6 minutos de tiempo de caída en 11.4 años. Esto es 6 minutos en 11.4 años sobre todo el sistema, no solamente en un componente. Dada la tecnología de hoy, esto es inalcanzable en términos prácticos, ya que se convierte en una meta irreal. Tiempos de Caída de menos de 10 minutos por año (cerca de 99.998%) son probablemente alcanzables. Además de los sistemas bien diseñados, un grado significativo de suerte seguramente puede ser requerido, y el factor suerte no se puede planear. Buena suerte es cuando su mejor programador esta trabajando hasta tarde en la noche y la aplicación mas crítica de su sistema se viene abajo, y el soluciona el problema lo mas rápido posible. Mala suerte, es cuando alguien entra a centro de cómputo y se tropieza con el cordón de poder de su sistema de producción. O también es mala suerte cuando un vehículo choca en el costado del edificio donde se encuentra el surtido de energía eléctrica. Por ser un factor impredecible es muy difícil su consideración.

1.1.4 Métrico para la disponibilidad

El término clave *TMEF*, es una media de tiempo. Un tiempo medio entre fallas es una media estadística. Un promedio, si un disco tiene un *TMEF* de 200,000 horas (alrededor de 23 años), esto no significa que todos los discos ensamblados en esa línea están garantizados para trabajar en exactamente 23 años, y que luego colapsen.

La media es una tendencia. Si se observan los discos de una línea de ensamble durante un tiempo, el promedio esperado de vida de un disco, antes de que falle es de cerca de 23 años. Esto significa, sin embargo que algunos discos pueden fallar el primer día, y otros pueden hacerlo al llegar a los 40 años. Otro número clave es la desviación estándar. Esta nos dice que tan lejos se encuentran los miembros de una población de la media.

Cuando se buscan componentes de hardware, se desean aquellos cuyo *TMEF*, posean una pequeña desviación estándar. Algo que nos indica acerca de la calidad del proveedor de hardware.

Las mismas normas que aplican al *TMEF* se siguen en el *TMRF*. Si al administrador del sistema le toma 15 minutos para recuperarse de un problema particular, esto no necesariamente significa que siempre le tomarán 15 minutos. Algunas complicaciones se pueden dar durante el tiempo de reparación. El reinicio del sistema se puede incrementar, debido a que el sistema posee mayores componentes que son evaluados durante el proceso de reinicio. Muchos aspectos del *TMRF* pueden salirse de control, por ejemplo si algún componente se daña y el proveedor lo ha descontinuado; o bien que tome varios días su entrega.

1.1 Determinar los requerimientos de disponibilidad

El primer paso, en el diseño de alta disponibilidad, es el descubrir los verdaderos requerimientos para disponibilidad de los usuarios y los servicios que provee el sistema en general. Esto requiere un acercamiento con la mayor cantidad de usuarios posible, cubriendo al menos todos los usuarios de aplicaciones críticas. La respuesta inicial de la mayoría de usuarios será que el sistema debe estar disponible todo el tiempo, por lo que se necesita explicar el costo elevado de proveer una alta disponibilidad.

1.2.1 Definición de un acuerdo de nivel de servicio

Estas consultas con los usuarios forman la base de un acuerdo de nivel de servicio entre el proveedor de servicios de sistemas de informática y los usuarios. Se puede elegir limitar a un acuerdo simple que cubra solamente la disponibilidad del sistema, o bien expandirlo a que incluya:

1. Los tiempos de respuesta.
2. Nuevas características.
3. Rendimiento
4. y otras características de calidad.

Se recomienda iniciar únicamente con las características de disponibilidad del sistema, y conforme el sistema se estabilice se puede incluir otros puntos en el acuerdo de servicio de disponibilidad. Esta propuesta tiene muchos beneficios:

1. Las expectativas de los usuarios están delimitadas por el acuerdo de servicio.
2. Permite ganar tiempo para que los departamentos de informática mejoren sus servicios.
3. Permite administrar la demanda de servicios, ya que todo se basa en un acuerdo inicial.

Una vez se demuestra que se ha logrado el acuerdo inicial de disponibilidad, se puede expandir a nuevas áreas.

1.2.2 Identificar requerimientos de disponibilidad

Las primeras preguntas que se debe preguntar a los usuarios son:

¿Cuál es su agenda de actividades en el sistema? ¿En que horas del día y días de la semana espera utilizar el sistema o aplicación?

Las respuestas ayudarán a identificar en que momentos el sistema o aplicación debe de estar disponible. Normalmente, las respuestas coinciden con las horas regulares de trabajo de los empleados. Por ejemplo, usuarios que principalmente trabajan en una aplicación de las 8:00 a.m. a 5:00 p.m. de lunes a viernes requerirán estos períodos de tiempo. Sin embargo, algunos usuarios desean utilizar el sistema en horario extraordinario, dependiendo del número de usuarios que lo utilicen durante estas horas, se puede incluir estos tiempos en el rango de horas de disponibilidad del sistema.

Cuando existen usuarios externos o bien clientes que utilicen el sistema, las horas de operación son a menudo extendidas más allá de las horas normales de trabajo. Esto es muy común en servicios de banca en línea, servicios de *Internet, e-commerce* y otros servicios tales como la electricidad, agua y comunicaciones. Los usuarios de estos sistemas usualmente demandan disponibilidad de 24 horas al día, 7 días a la semana, o lo mas cercano posible.

Las siguientes preguntas a realizar a los usuarios son:

¿Cuánto tolera las caídas de sistema durante los momentos que esta utilizando la aplicación? ¿Qué opina de cuando se baja el sistema de manera programada?

La meta es comprender el impacto sobre los usuarios, si el sistema no esta disponible cuando se tiene programado que este disponible. Por ejemplo, un usuario puede decir que solamente puede soportar dos caídas del sistema al mes.

Las respuestas a estas preguntas permiten saber si se puede programar bajar el sistema durante el tiempo de operación normal, para alguna tarea administrativa. Por ejemplo, un sistema que debe de estar en línea 24 horas al día, 7 días, a la semana puede requerir que se baje de manera programada a la media noche para realizar un respaldo completo.

La pregunta final a realizar a los usuarios es:

¿Cuál es la cantidad máxima de tiempo que espera de una caída del sistema?

Esta pregunta ayuda a identificar, cuan largo los usuarios pueden esperar para la restauración de los servicios del sistema durante una caída, o cuanto la pueden tolerar. Por ejemplo algunos usuarios pueden tolerar hasta 3 horas de falta de sistema, o bien si esta se programó pueden esperar el tiempo que se haya indicado.

1.3 Costos y riesgos

La única manera de convencer a las personas quienes controlan el presupuesto, es demostrar el valor que tiene proteger la operación continua del sistema y detallarlos en términos monetarios.

1.3.1 El costo de los tiempos de caída.

Los costos más obvios de tiempos de caída, no son probablemente los más elevados. Los costos más elevados en los de tiempos de caída, se representan en la pérdida de productividad, y el costo actual es dependiente sobre que trabajo realizan los usuarios en los sistemas afectados.

Si se realiza el cálculo en base al costo de los usuarios internos y programadores de la empresa, entonces quizás el costo parezca ser nada mas el tiempo y el acarreo de costos de las personas que en ese momento no pueden trabajar.

Para una empresa de desarrollo con decenas de desarrolladores, estos costos pueden ser verdaderamente significativos. Un desarrollador puede ganar entre $500 y $1000 al día, esto puede variar dependiendo varios factores. Es razonable pensar, que mantener a un grupo de 50 desarrolladores sin hacer nada durante una semana cuesta $250,000.

Pero aún así los $250,000 son únicamente el costo directo de la planilla de programadores. No se toma en cuenta el costo adicional requerido para recuperar el tiempo perdido, y asegurar que se entreguen a tiempo los proyectos de los cuales están encargados. Si los programadores son consultores, u otro tipo de empleado al que se le paga por hora, entonces ese tiempo extra puede exceder fácilmente los $375,000.

Para los usuarios finales, el costo obviamente varía dependiendo de la línea de trabajo a la que pertenezcan y los servidores afectados. Por ejemplo para una empresa de bolsa de valores en *Wall Street*, el valor de 20 minutos sin sistema equivale a 2 millones de dólares. Sin tomar en cuenta todas aquellas oportunidades de inversión que se pudieron haber concretado.

1.3.2 Problemas de administración

Una vez se haya comprendido los costos directos e indirectos debidos a los tiempos de caída, y asumiendo que no tenemos la decisión de compra, se necesita explicar a la alta gerencia que se debe de realizar para cerrar esos agujeros en los sistemas:

1. Primero, explicar los de niveles de disponibilidad a la administración, entonces asignar costos a cada uno de ellos. Observar todo el panorama, planear todos los escenarios posibles. Considerar los problemas que pueden ocurrir, y observarlos desde una amplia perspectiva.

2. Luego, decidir sobre los riesgos que vale la pena tomar. Esto significa gastar de manera inteligente, en cosas que tienen los efectos más benéficos y que son los que tienen mayor probabilidad de suceder.

3. Finalmente, realizar y justificar recomendaciones. Al menos que la administración tenga opiniones específicas, y asumiendo que las recomendaciones tienen buen sentido financiero, ellos tomaran muy en serio estas recomendaciones.

Uno de los mayores desafíos en explicar los riesgos y costos a la administración, es convencer de no esperar en aprender de una mala experiencia. Explicar a detalle lo que puede suceder. Asegurarse que el balance de estas horribles predicciones contra los costos para la empresa si estos eventos ocurren es elevado y sirva para tomar una decisión para la disponibilidad de los sistemas. Muchas compañías no planean apropiadamente que harán con una falla hasta que esta ocurre, es entonces cuando la alta gerencia comienza en entender el costo real de esas fallas. Esto se convierte en un proceso iterativo: un desastre ocurre; la compañía planea por si puede ocurrir este desastre nuevamente. Un evento de caída de sistema no planeado ocurre y la compañía una vez más elabora una respuesta. No se planea sobre eventos que no hayan ocurrido.

La mayoría de estas compañías operan de una manera reactiva, y trabajan sobre el escenario perder-perder. Si se planea el que hacer o prevenir estos eventos, estos no afectarán al sistema.

1.4 Niveles de disponibilidad

Es importante remarcar que los niveles de disponibilidad que se discuten a continuación no tienen una naturaleza discreta. Hay una cantidad enorme de pasos incrementales y combinación de tecnologías, que pueden agregar disponibilidad al sistema en general.

1.4.1 Nivel 1. Disponibilidad regular: No realizar nada especial.

Este es el nivel mas básico de protección es esencialmente no protegerse en nada. No se toman medidas especiales para proteger a los sistemas durante las caídas de sistema. El nivel 1 puede incluir *backups*, pero nada más. Se enfrenta con la caída de sistema, sin tener ningún plan en especial. Este nivel es suficiente para muchas aplicaciones, pero el resultado para recuperarse de una falla puede tomar días. Se puede perder también datos. Si se sufre de una falla de discos cerca del final del día, antes de que inicien los procesos del *backup*, se perderá un día entero de trabajo.

1.4.2 Nivel 2. Disponibilidad incrementada

El nivel 2 no difiere significativamente del nivel 1, a excepción que este incluye algunos mecanismos de protección de datos. Esto significa emplear la tecnología RAID; esto no es más que arreglos de disco en espejo, o bien llegar a niveles más altos de protección como RAID-5. La falla en un disco, no resultará en la pérdida de la información, debido a que los datos son almacenados en más de un disco físico.

La política de backups sigue siendo importante, para proteger la información causada por errores del usuario, o fallas totales del equipo. Y estas últimas u otros componentes pueden tomar varios días en restaurar el sistema.

1.4.3 Nivel 3. Alta disponibilidad: Protección del sistema.

El nivel 3 es lo que comúnmente se le denomina alta disponibilidad o HA (*High Availability*). En una configuración de HA, se toman dos servidores separados y se unen parar formar un *cluster,* combinados con un arreglo de discos protegidos, el sistema tiene todos sus componentes duplicados. Si un componente falla, este puede ser automáticamente o manualmente reemplazado con su duplicado.

Algún tiempo de caída ocurre en las configuraciones HA, pero en la mayoría de casos esta tiene un tiempo limitado de duración. Tiempos de operación que excedan el 99.98 por ciento pueden ser consideradas en este nivel. Para el diseño de sistemas configurados en HA requieren diseño de red, consideraciones de soporte de sistema, administración de sistema y red, y mecanismos de auditoría que aseguren que se tienen cubiertos todos los escenarios.

Una desventaja en la implementación de este nivel de disponibilidad es que existe un cierto grado de complejidad que se agrega al ambiente, y la complejidad provoca que sea más difícil su administración. La otra desventaja es el costo. Usted esta agregando redundancia, y esto requiere dinero para los sistemas, redes, tiempo de los ingenieros, soporte y configuración.

1.4.4 Nivel 4. Recuperación de desastres: Proteger a la organización.

Recuperación de desastres es el nivel más alto y el más caro en la protección de sistemas. Cuando se implementa este nivel, se protege contra la pérdida total del edificio o el sitio de operación de la organización, teniendo un lugar alterno ubicado a cierta distancia donde se replica la información. Se debe poseer el hardware necesario para operar un sistema en reserva en el sitio de respaldo, teniendo políticas y procedimientos de mantenimiento de este sitio, y teniendo la infraestructura para que las aplicaciones puedan ser operacionales lo mas rápido posible después de que haya ocurrido esta falla.

1.5 ¿Qué es un sistema de base de datos de alta disponibilidad?

Un sistema de base de datos de alta disponibilidad, no es más que la implementación de las características que un manejador de base de datos para lograr un nivel disponibilidad en la clasificación de alta disponibilidad. Estas características a las que pueden ser definidas por el fabricante para alta disponibilidad, o bien la utilización de características existentes para otro fin. Un componente crítico de una solución para la empresa de hoy es un sistema de base de datos de alta disponibilidad. Que no es más que la habilidad de mantener en operación continúa el manejador de base de datos, sin fallas, corrupción de datos y sin tiempos de caída. Dentro de los mecanismos que un manejador de base de datos posee para lograr estos requerimientos se tiene:

1. Replicación
2. Base de datos en espera
3. Base de datos en paralelo

Cada una de estas implementaciones tiene sus ventajas y desventajas, en los siguientes capítulos se examinará cada una de estas implementaciones para el logro de un sistema de base de datos de alta disponibilidad.

2. BASE DE DATOS EN ESPERA

2.1 Qué es una base de datos en espera

En todo sistema de base de datos se tiene una base de datos funcional y en producción atendiendo a un número de usuarios conectados, a esta base de datos se le llama base de datos primaria. Una base de datos en espera es una réplica creada desde un respaldo de la base de datos primaria. Esto aplicando la información de bitácora desde la base de datos primaria a la base de datos en espera, estas dos bases de datos pueden estar sincronizadas.

Una base de datos en espera tiene los siguientes propósitos:

1. Proveer alta disponibilidad al sistema.
2. Protección contra corrupción de información.
3. Base de datos de consultas.

Si la base de datos primaria es destruida o se corrompe la información, se puede realizar un *failover* a la base de datos en espera, en tal caso la base de datos en espera se convierte en una nueva base de datos primaria. La base de datos en espera puede estar abierta en modo de lectura, de tal modo que puede funcionar como base de datos de consultas.

2.1.1 Opciones de configuración

Se puede configurar una base de datos en espera de diferentes maneras, dependiendo del método de:

- Transferencia de información de bitácora al sitio en espera.
- Aplicación automática de la información de bitácora sobre la base de datos en espera.

La mayoría de de manejadores de bases de datos utilizan dos mecanismos para la implementación de un ambiente de base de datos en espera que se pueden clasificar de la siguiente manera:

Ambiente administrado en espera: En este ambiente la información de bitácora de la base de datos primaria es enviada a la base de datos en espera y se aplica cuando esta última es iniciada.

Ambiente administrado en modo recuperación: En este ambiente la base de datos en espera aplica automáticamente la información de bitácora una vez es recibida desde la base de datos primaria. Algunos manejadores de bases de datos permiten aplicar la información de bitácora manualmente. En cualquier instante la base de datos en espera se puede abrir en modo de solo lectura, para propósitos de consulta.

2.1.2 Ventajas y desventajas

Una base de datos en espera es una alternativa productiva y poderosa para lograr un sistema de bases de datos de alta disponibilidad, como alternativa a prevención de desastres y proveer una base de datos de consultas. Por ejemplo se puede tener las siguientes arquitecturas:

- Administrar una base de datos en espera en un lugar geográficamente remoto de la base de datos primaria, o mantener varias bases de datos en espera en diversos lugares geográficamente.

- Administrar una base de datos en espera en diferentes discos en el mismo servidor, de tal manera que si los discos de la base de datos primaria fallan, se puede activar la base de datos en espera y continuar la operación normal del sistema.

En un ambiente administrado en modo de recuperación, la información de bitácora es automáticamente aplicada sobre la base de datos en espera desde la base de datos primaria, por lo que el tiempo de activar la base de datos en espera es mínimo y se logra un mejor porcentaje de disponibilidad del sistema.

Esta configuración provee protección contra errores de usuario, o corrupción de datos por aplicaciones que estén corriendo en la base de datos primaria.

Los requerimientos de hardware para la implementación de este sistema de bases de datos de alta disponibilidad, requiere como mínimo un servidor con características similares al servidor de la base de datos primaria, y una conexión de red fiable entre la base de datos primaria y la base de datos en espera.

En cuanto a desventajas se tiene el grado de complejidad a nivel de infraestructura de telecomunicaciones y equipo de hardware para mantener y dar operación a la base de datos en espera en el momento de una falla.

2.1.3 Conceptos y terminología

Algunos de los términos básicos importantes para la comprensión de este capitulo son los siguientes:

Failover: Es la operación de transformar una base de datos en espera a una base de datos primaria funcional. Esta operación también es llamada *activación* de la base de datos en espera. Luego de realizar el *failover*, no se puede convertir nuevamente esta base de datos primaria funcional en una base de datos en espera.

Secuencia de intervalo: Una secuencia de archivos de bitácora que deben ser manualmente aplicados a la base de datos en espera, antes de colocar en modo administrado de recuperación.

Modo administrado de recuperación: Cuando una base de datos corre en modo administrado de recuperación, automáticamente aplica todos los archivos de bitácora recibidos desde la base de datos primaria.

Ambiente administrado de espera: Una configuración en la cual una base de datos primaria automáticamente administra los archivos de bitácora del sitio primario, sobre el sitio en espera. Si la base de datos en espera esta en modo administrado de recuperación aplica los archivos de bitácora recibidos de manera automática desde la base datos primaria a la base de datos en espera. De lo contrario, el sitio en espera continúa recibiendo y archivando los archivos de bitácora en espera de su conversión a base de datos primaria, en el momento del *failover*.

Modo manual de recuperación: Este modo permite realizar la recuperación o *failover* de una base de datos en espera de manera manual, es decir aplicar la información de bitácora archivada en el sitio de espera con una secuencia de comandos.

Ambiente en espera no administrado: Cualquier ambiente en el cual la base de datos primaria no envía, ni archiva la información de bitácora al sitio en espera. En este ambiente, de manera manual se deben de transferir los archivos de bitácora al sitio en espera y luego aplicarlos.

Base de datos primaria: Una base de datos, a partir de la cual se crea una base de datos en espera. Toda base de datos en espera esta asociada con una y solamente una base de datos primaria. Una sola base de datos primaria, puede soportar más de una base de datos en espera. Esta es la base de datos a la cual todos los usuarios permanecen conectados desde sus aplicaciones, realizando sus labores diarias.

Sito primario: Es la ubicación de la base de datos primaria. Notar que los sitios primarios y sitio de espera pueden operar en servidores separados o en el mismo servidor.

Modo sólo lectura: Este modo permite realizar consultas en la base de datos en espera, pero no permite realizar cambios sobre los datos.

Base de datos en espera: Una base de datos réplica, creada utilizando un respaldo de la base de datos primaria. Una base de datos en espera tiene su propia configuración en cuanto a recursos de memoria a utilizar.

Ambiente de base de datos en espera: La configuración física de la base de datos primaria y sus bases de datos en espera. El ambiente depende de algunos factores, incluyendo:

1. El número de bases de datos en espera asociadas con la base de datos primaria.
2. El número de servidores utilizados para las bases de datos.
3. Las estructuras físicas de directorio utilizadas para las bases de datos.
4. La configuración de red.

Sitio en espera: La ubicación de la base de datos en espera. El sitio en espera puede ser el mismo servidor donde reside a base de datos primaria o bien un servidor separado.

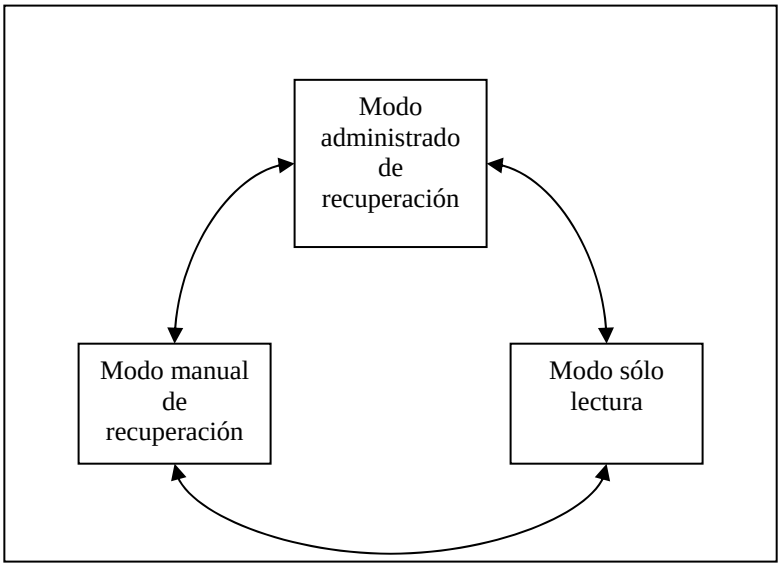
2.2 Modos y configuración de bases de datos en espera

Dependiendo del nivel de disponibilidad que se quiere lograr implementando una arquitectura de alta disponibilidad podemos realizar las siguientes operaciones:

1. Administrar la base de datos en modo manual de recuperación.
2. Administrar la base de datos en modo administrado de recuperación.
3. Abrir la base de datos en modo sólo lectura, para realizar operaciones de consulta.

Sin embargo, no se puede tener una base de datos operando en mas de uno de estos modos al mismo tiempo, lo que es posible es pasar de un modo a otro. Por ejemplo, se puede estar operando en un ambiente administrado de recuperación, luego abrir la base de datos en modo solo lectura, y como ultimo paso colocarla en modo manual de recuperación, como se muestra en la siguiente figura:

Figura 2. Transición de modos en una base de datos en espera



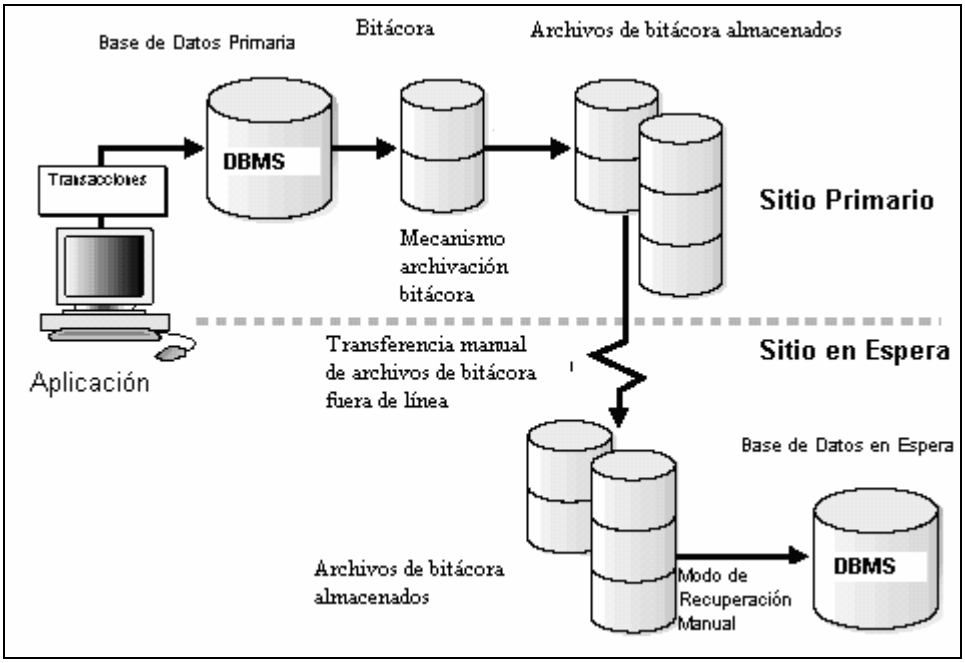
2.2.1 Modo manual de recuperación

En esta opción se coloca la base de datos en modo manual de recuperación, en tal caso de manera manual y continua se deben transferir los archivos de bitácora a la base de datos en espera para sincronizarla con la base de datos primaria.

El manejador de base de datos debe disponer de un comando en modo de operación manual. La figura 3 muestra un ejemplo de la base de datos en modo manual de recuperación.

El modo manual de recuperación, es útil en ambientes en los cuales no se desea establecer una conexión directa de la base de datos primaria a la base de en espera, a través del mecanismo nativo de conectividad del manejador de base de datos. También, si por alguna razón la base de datos primaria no puede transferir los archivos de bitácora archivados a la base de datos en espera en un ambiente administrado de recuperación, es necesario ejecutar la recuperación manual de la base de datos en espera para mantener sincronizados ambos sitios.

Figura 3. Base de datos en espera en modo manual de recuperación

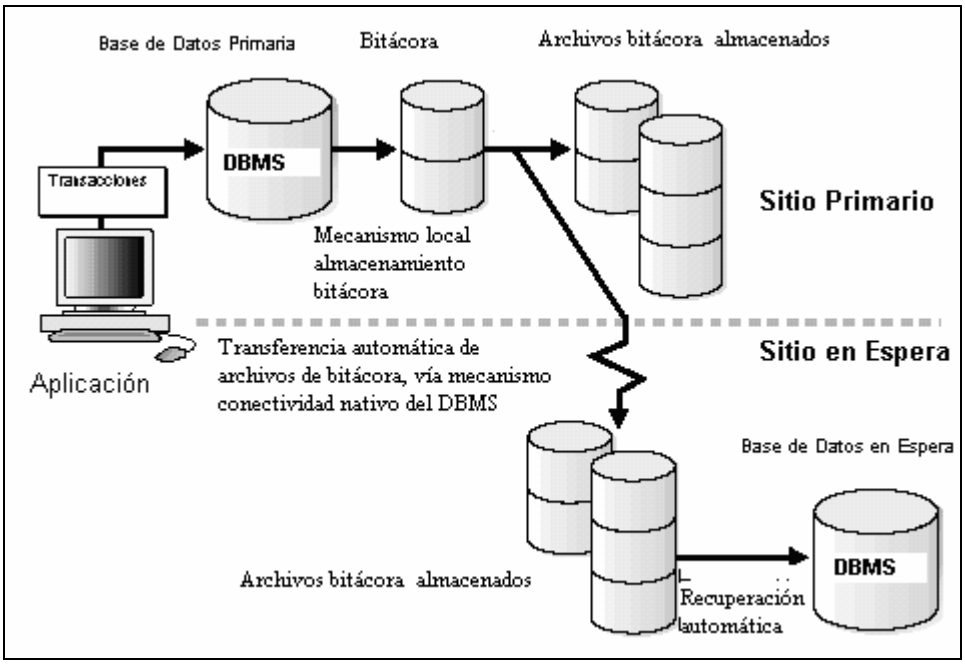


2.2.2 Modo administrado de recuperación

Se puede colocar la base de datos en espera en modo administrado de recuperación, en este caso se aplican sobre la base de datos en espera los archivos de bitácora conforme son recibidos desde la base de datos primaria.

La principal ventaja de operar una base de datos en modo administrado de recuperación es que no se tiene que transferir o aplicar los archivos de bitácora de manera manual: el manejador de base de datos automatiza este procedimiento

Figura 4. Modo administrado de recuperación



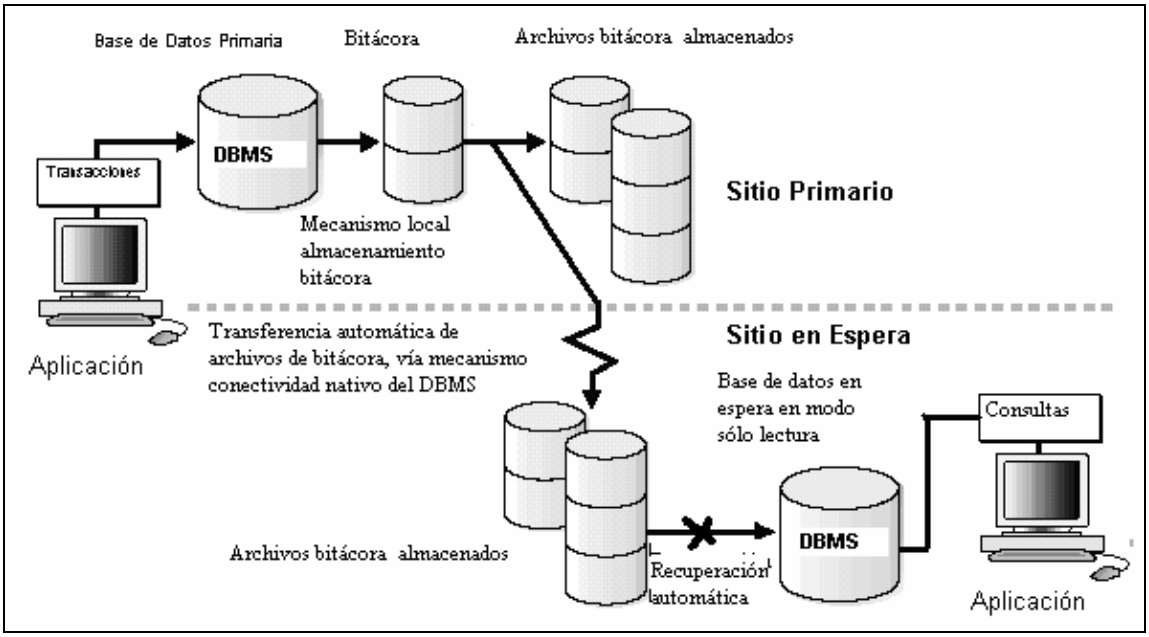
2.2.3 Modo sólo lectura

También se puede operar una base de datos en espera en modo sólo lectura, después de realizar una recuperación manual o administrada. Por lo que se pueden realizar consultas a la base de datos en espera al momento de la recuperación. Luego se puede colocar nuevamente la base de datos en modo manual de recuperación o administrado, según sean las necesidades. La figura 5 muestra como opera una base de datos en modo solo lectura.

En un ambiente administrado en espera, el sitio en espera continúa recibiendo archivos de bitácora desde la base de datos primaria, cuando la base de datos en espera esta operando en modo sólo lectura, no se realiza ninguna recuperación por lo cual estos archivos de bitácora se archivan hasta que el administrador de base de datos decida reiniciar en alguno de los modos de recuperación de bases de datos en espera.

Una base de datos de espera operando en modo lectura es útil cuando, se desea reducir el número de consultas a la base de datos primaria.

Figura 5. Base de datos en espera modo solo lectura

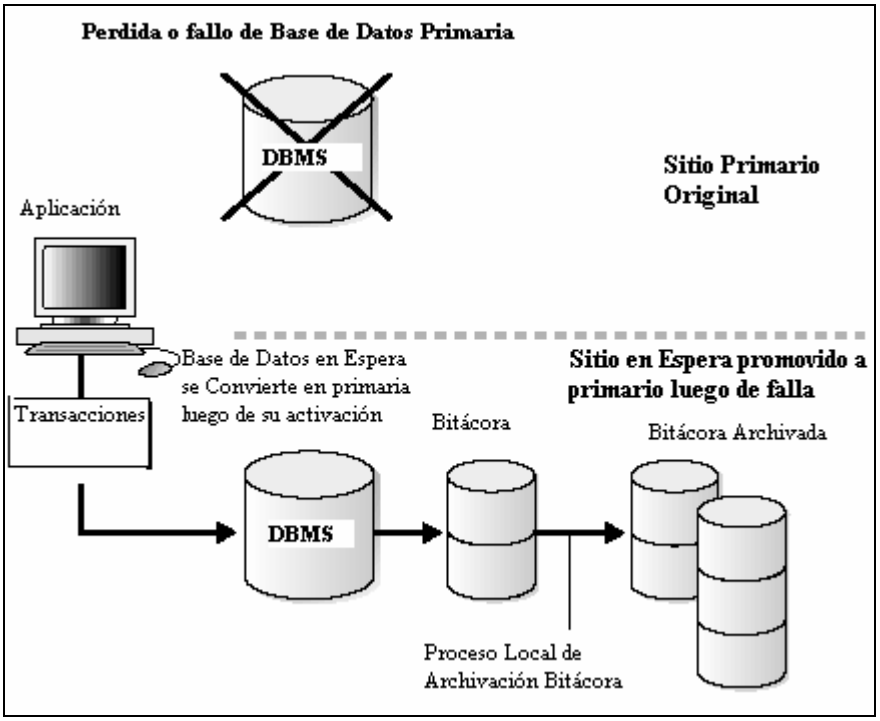


2.3 Respuesta a fallos en una base de datos en espera

Al procedimiento de *failover* a una base de datos en espera, también es conocido como activación, esto ocurre cuando se le indica a la base de datos en espera, que se active y se convierta en una base de datos primaria.

La figura 6 muestra el proceso de activación de un sitio primario hacia su respectivo sitio en espera, que es administrado en modo administrado de recuperación.

Figura 6. Activación de base de datos en espera



Después de activar la base de datos en espera, esta cesa de ser una base de datos en espera y se convierte en una base de datos primaria y funcional. En este punto esta base de datos es abierta en modo de lectura y escritura, y todos los usuarios del sistema se reconectan a la base de datos en el sitio en espera.

El tiempo de activación de una base de datos en espera, depende del modo de recuperación en el cual este, el mas efectivo es el modo administrado de recuperación ya que constantemente mantiene sincronizada la base de datos en espera, el tiempo de caída no rebasará mas allá de 15 minutos. Mientras que en el modo manual, este dependerá de la proporción de información de bitácora que se debe aplicar a la base de datos en espera, que se tiene de la ultima sincronización manual. Por lo que depende de las políticas implementadas de sincronización manual. La información en la base de datos en espera debe estar sincronizada, a todas aquellas transacciones cometidas hasta un instante antes de la falla de la base de datos primaria.

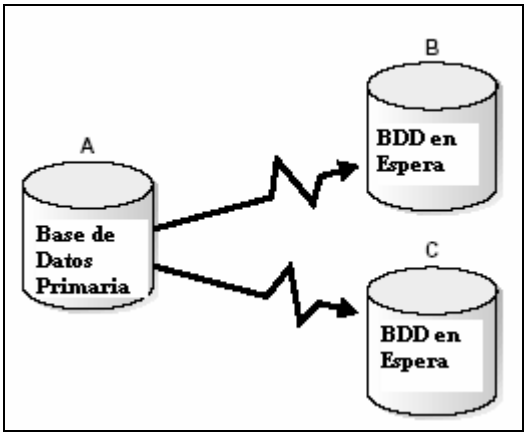
Esto representa una gran ventaja de esta arquitectura que es implementada por la mayoría de manejadores de bases de datos.

2.3.1 Consecuencias de la activación de una base de datos en espera

La activación de una base de datos en espera, la convierte de manera permanente en una base de datos primaria. Debido a que la activación es una operación unidireccional, no es posible retornar nuevamente la base de datos primaria a cualquier modo de base de datos en espera. En otras palabras, no se puede deshacer la operación de activación.

Otra consecuencia de la activación es que si existe más de una base de datos en espera que haya estado soportando a la base de datos primaria original, pasan a convertirse en bases de datos inválidas, ya que solamente una de ellas fue activada a base de datos primaria. Por ejemplo, asuma que la base de datos primaria A tiene como base de datos en espera B y C, como se ilustra en la figura 7.

Figura 7. Base de datos primaria con múltiples bases de datos en espera



Si en el momento de una falla de la base datos primaria A, se activa la base de datos B, entonces la base de datos C no puede ser configurada como una base de datos en espera de B, dado que la bitácora de B fue reiniciada, y por lo tanto la información de bitácora de B no puede ser aplicada a C.

En algunas situaciones, es conveniente administrar múltiples bases de datos en espera, incluso el ubicarlas en lugares geográficos distantes para un nivel de disponibilidad de recuperación de desastres; en el escenario de esta configuración podemos tener los siguientes escenarios:

1. El servidor A sufre una falla de Hardware.
2. Se activa la base de datos en espera B. Los usuarios ahora se conectan a B como base de datos primaria.
3. Inmediatamente se reparan los problemas de hardware del servidor A.
4. Se apaga y cierra la base de datos B, y se inicia y abre la base de datos A.
5. Los usuarios se conectan ahora a la base de datos A como la base de datos primaria. C continua funcionando como una base de datos en espera para A, mientras B se ha invalidado como base en espera.

Una consecuencia de este escenario es que cualquier cambio que la base de datos B sufra durante el breve tiempo que sirva como base de datos primaria no puede ser aplicado por bitácora a la base de datos A. Algunos manejadores de base de datos disponen de utilitarios para examinar las sentencias de SQL entre dos períodos de tiempo y son estas las que se deberían aplicar a la base de datos A.

2.3.2 Verificación de la base de datos en espera sin realizar activación

Debido a que la activación destruye la funcionalidad de una base de datos en espera, es recomendable realizar esta operación única y absolutamente cuando sea necesario.

Una manera de verificar que el procedimiento de base de datos en espera sea el correcto, es abrir la base de datos en espera en modo sólo lectura. De esta manera se puede consultar la base de datos para verificar que los archivos de la base de datos se estén actualizando de la manera correcta, según la información recibida de la bitácora de la base de datos primaria.

2.3.3 Recreación de la base de datos primaria original después de activación.

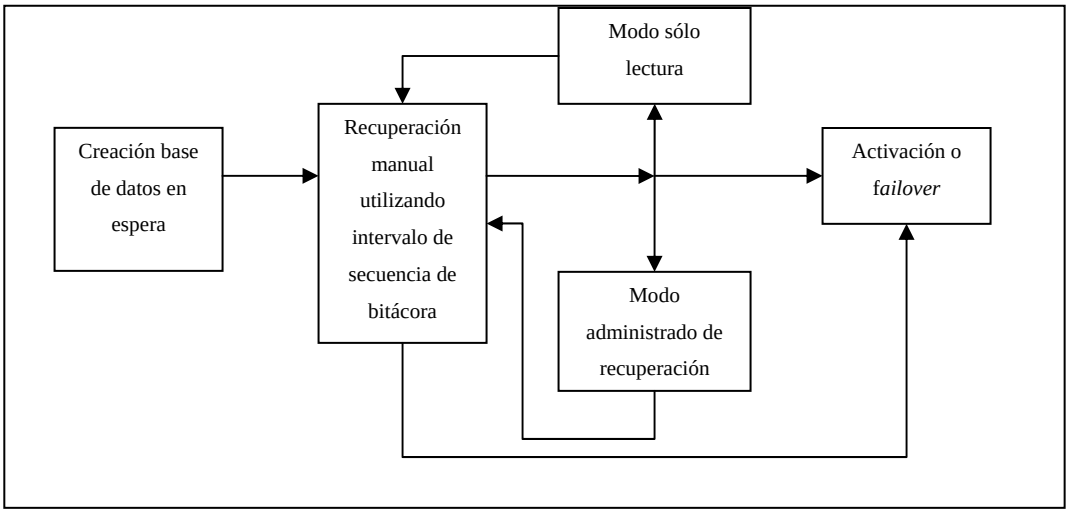
Al activar la base de datos en espera y resolver el problema en sitio primario que requirió la operación de activación, se tiene la opción de recrear la base de datos primaria en el original sitio primario. Para ello son necesarios los siguientes pasos, asumiendo que el sitio original primario esta en el servidor A y se activo la base de datos del sitio en espera del servidor B:

1. Realizar un respaldo de la base de datos activada del servidor B
2. Restaurar el respaldo creado del servidor B sobre el servidor A.
3. Cerrar la base de datos en el servidor B.
4. Abrir la base de datos A restaurada. Esta es ahora la base de datos primaria.
5. Realizar un respaldo de la base de datos del servidor A.
6. Utilizar el respaldo de A para recrear una base de datos en espera sobre el nodo B.

2.4 Ciclo de vida de una base de datos en espera

La mayoría de implementaciones de bases de datos en espera utilizan el modo administrado de recuperación. El ciclo de vida de una base de datos en espera es el siguiente:

Figura 8. Ciclo de vida de una base de datos en espera



Las cuatro etapas ilustradas son:

1. Creación de la base de datos en espera.
2. Recuperación manual utilizando intervalo de secuencia de bitácora.
3. Modo solo lectura o modo administrado de recuperación.
4. Activación o *failover*.

2.5 Mantenimiento de una base de datos en espera

Si bien un ambiente administrado de base de datos en espera esta automatizado. Es necesario llevar algunas tareas para verificar el estado de la base de datos en espera. Las tareas más comunes de mantenimiento son:

1. Revisión de la información de bitácora.
2. Respaldo de la base de datos en Espera
3. Consideraciones en tareas de administración sobre cambios físicos sobre la base de datos primaria.

2.5.1 Revisión de la información de bitácora:

El manejador de base de datos debe proveer un mecanismo, que nos indique que la base de datos esta funcionando de manera apropiada, por lo que es necesario saber si los archivos de bitácora han sido entregados al sitio en espera y cuales han sido aplicados. El diccionario del manejador debe proveer una vista donde se encuentre esta información.

2.5.2 Respaldo de una base de datos en espera

Si es necesario, se debe realizar respaldo de la base de datos en espera, pero no mientras la base de datos este en modo manual o administrado de recuperación. Por lo que es necesario colocar la base de datos fuera de estos modos, y realizar el respaldo. El mecanismo empleado en todos los manejadores de base de datos es cerrar la base de datos y realizar el respaldo o bien mientras este en modo sólo lectura. El respaldo no es más que una copia de todos los archivos que componen la base de datos en espera.

2.5.3 Consideraciones en tareas de administración

Cambios en las estructuras físicas de almacenamiento en la base de datos primaria obviamente afectan a la base de datos en espera, estas tareas en algunos casos son llevadas a cabo por la información almacenada bitácora. En algunos casos sin embargo, es necesario, llevar a cabo tareas de mantenimiento sobre la base de datos en espera, algunos de los ejemplos más comunes vistos en los manejadores de base de datos son:

1. Agregar archivos de datos a la base de datos primaria.
2. Eliminación de archivos de la base de datos primaria.

Las tareas de mantenimiento sobre la base de datos en espera que se deben de ejecutar son las mismas que en la base de datos primaria, algunos manejadores restringen que la base de datos este modo manual de recuperación.

3. REPLICACIÓN DE DATOS COMO MÉTODO DE ALTA DISPONIBILIDAD

Utilizar los métodos de replicación para manejar o proveer alta disponibilidad, no es más que replicar las transacciones de una base de datos primaria, donde se encuentran conectados los usuarios en el sistema, hacia una base de datos secundaria. Esta base de datos secundaria recibe los cambios propagados por la base de datos primaria a través de un canal de comunicación. Si el servidor donde reside la base de datos primaria falla, todas las fuentes de datos y las aplicaciones, se reconectan al servidor donde reside la base de datos secundaria. La replicación no provee ningún mecanismo automatizado de respuesta a fallos, dada que su naturaleza reside en un ambiente de bases de datos distribuidas, en el cual se requiere una reconexión.

La utilización de replicación para el logro de alta disponibilidad, requiere un análisis de las tablas empleadas por los sistemas para poder construir las réplicas en la base de datos secundaria, agregando complejidad a su administración. Técnicas como base de datos en espera y base de datos en paralelo pueden ser más adecuadas y menos complejas.

3.1 Introducción a la replicación

Replicación es el proceso de copiar y administrar objetos de base de datos, tales como tablas, hacia múltiples bases de datos en localidades remotas que son parte de un sistema de bases de datos distribuido. Los cambios ejecutados en una localidad son capturados y guardados localmente antes de ser aplicados a las localidades remotas.

Los términos sistemas de bases de datos distribuidas y replicación de bases de datos, están relacionados, pero no son equivalentes. En un sistema puro de bases de datos distribuidas se maneja o administra una sola copia de todos los objetos de la base de datos y sus datos, es decir que existe de manera única la ocurrencia de un objeto de base de datos en todas las localidades, es decir la información se encuentra particionada de manera horizontal entre todas las localidades. Las aplicaciones en una base de datos distribuida utilizan transacciones distribuidas para acceder y modificar tanto los datos locales como remotos.

El término replicación se refiere a la operación de copiar y administrar objetos de base de datos en múltiples bases de datos a lo largo de un sistema distribuido, en este caso, existen varias copias del mismo objeto en diferentes localidades. Dado que la replicación depende de una tecnológica de base de datos distribuida, la replicación ofrece beneficios en las aplicaciones, que no son posibles en un ambiente puro de base de datos distribuida, tal como la disponibilidad y rendimiento.

La replicación es utilizada para mejorar el rendimiento de la base de datos local y brindar alta disponibilidad en el sistema; ya que existen localidades alternas con copias o réplicas de los objetos de base de datos. Por ejemplo, una aplicación puede normalmente acceder a la base de datos local, en lugar de un servidor remoto para minimizar el tráfico de red y lograr un máximo rendimiento.

La replicación mejora la disponibilidad de las aplicaciones debido a que provee localidades alternas para el acceso de los datos. Si una de las localidades no esta disponible, los usuarios pueden continuar consultando o modificando la información, en cualquiera de las localidades restantes. En otras palabras la replicación provee una excelente alternativa como respuesta a fallos a un tiempo de caída en una localidad particular.

3.2 Componentes básicos

3.2.1 Objeto réplica

Un objeto réplica es un objeto de la base de datos que esta definido en varios servidores en un sistema de base de datos distribuida. En un ambiente de réplica, cualquier actualización hecha en el objeto réplica en una localidad es propagada y aplicada en todas las copias existentes en cada localidad. Algunos de los objetos de base de datos de los cuales el manejador de base de datos debería de poder replicar tenemos:

1. Tablas.
2. Índices
3. Vistas.
4. Sinónimos.
5. Procedimientos y funciones.

3.2.2 Esquema

Se refiere el espacio nombrado que contiene y es propietario de los objetos de base de datos sobre los cuales se ejecutan las aplicaciones. Objetos tales como: Tablas, vistas, sinónimos, procedimientos, etc.

3.2.3 Grupos de réplica

En un ambiente de replicación, el manejador de base de datos administra los objetos de réplica, utilizando grupos de réplica. Un grupo de réplica, es un conjunto de objetos replica que están lógicamente relacionados. Por lo que tanto, grupo de réplica y objeto réplica son administrados juntos.

Típicamente, se crea y utilizan los grupos de réplica para organizar los objetos de un esquema de la base de datos; necesarios para el funcionamiento de determinada aplicación. Sin embargo, los grupos de réplica y esquemas no necesitan corresponder uno con otro. Un grupo de réplica puede contener objetos de varios esquemas, y un sólo esquema puede tener sus objetos en diferentes grupos de réplica. Sin embargo, un objeto de réplica puede ser miembro de solamente uno y sólo un grupo de réplica.

3.2.4 Localidades de réplica

Un grupo de réplica puede existir en múltiples lugares, a estos lugares se les denomina localidades de réplica. Un ambiente de replicación soporta dos tipos básicos de localidad: localidad maestra y localidad subordinada.

Las diferencias entre localidad maestra y localidad subordinada son:

Un grupo de réplica en la localidad maestra es referido como grupo maestro. Un grupo de réplica en sitio subordinado es llamado grupo subordinado. Además, cada grupo tiene exactamente y únicamente una definición de localidad maestra. Un definición de localidad maestra es el sitio principal que sirve de centro de control para los grupos de réplica y los objetos en el grupo.

Una localidad maestra mantiene una copia completa de todos los objetos en grupo de réplica, mientras las localidades subordinadas pueden contener todos o subconjuntos de los datos de las tablas dentro de un grupo maestro.

Todas las localidades maestras en un ambiente de replicación multi-maestro se comunican directamente una con otra de manera continua esto para propagar los cambios que se den el grupo de réplica. Un sitio subordinado contiene una imagen, o instantánea, de los datos de las tablas con respecto a algún punto en el tiempo. Típicamente, un sitio subordinado es refrescado de manera periódica para sincronizar la información con el sitio maestro.

3.3 Tipos de replicación

Existen tres tipos de ambiente de replicación:

1. Replicación maestra.
2. Replicación de instantánea.
3. Configuración híbrida de instantánea y multi-maestro.

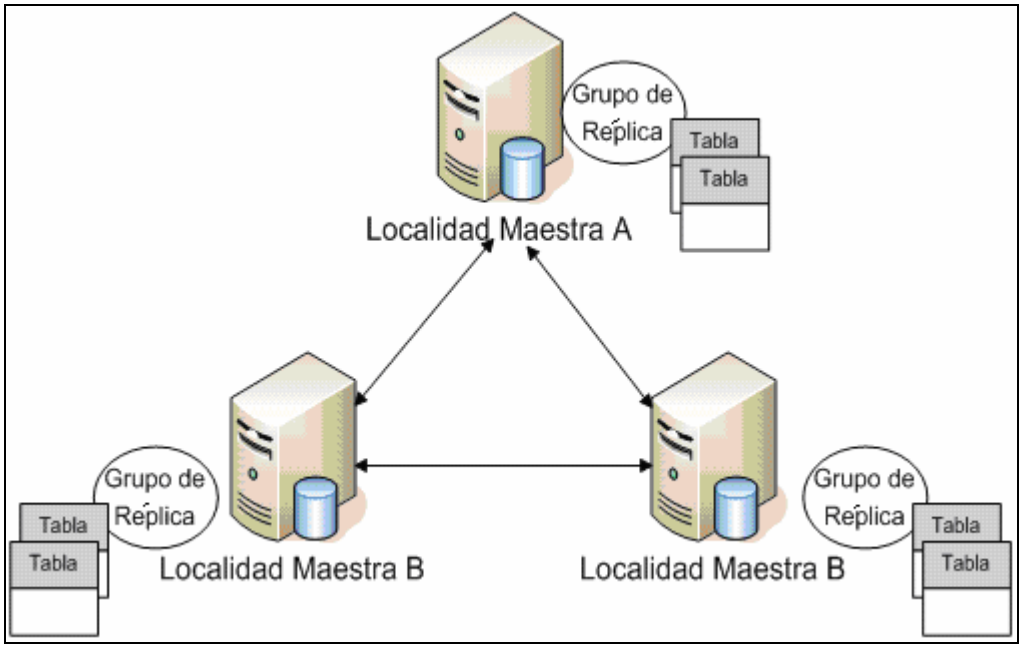
3.3.1 Replicación maestra

Existen dos formas de replicación maestra: replicación de una localidad maestra y replicación con múltiples localidades maestras o replicación multi-maestro.

La replicación de una localidad maestra, es aquella en la cual existe una sola localidad maestra y uno o más sitios de réplica. Todo cambio en las tablas se realiza sobre el sitio maestro, por lo que la detección y resolución de conflictos son llevadas a cabo en el sitio maestro.

La replicación con múltiples localidades maestras también denominada replicación multi-maestro o replicación de par a par o replicación de n-vías, permite que múltiples localidades, actúen como pares iguales para administrar grupos de objetos de base de datos replicados. Las aplicaciones pueden operar sobre las tablas replicadas en cualquiera de las localidades en el ambiente. El manejador de base de datos opera sobre las localidades maestras para que los datos converjan sobre todas las réplicas de las tablas para asegurar la consistencia e integridad de las transacciones.

Figura 9. Replicación multi-maestro



En la figura 9 se muestra un ambiente de replicación multi-maestro en el cual se tienen las localidades maestras A, B y C con su respectivo grupo de réplica que incluye un conjunto de tablas, en este esquema las aplicaciones pueden estar operando sobre cada una de las bases de datos localidades y replicar la información hacia las localidades adyacentes.

3.3.2 Replicación de instantánea

Una instantánea contiene una copia completa o parcial de una tabla en un punto del tiempo, basada en una instrucción SELECT. Una instantánea puede ser definida para sólo lectura o actualizable.

Todas las instantáneas proveen los siguientes beneficios:

1. Permiten acceso local, lo que mejora el tiempo de respuesta y disponibilidad.
2. Reducir la carga sobre la localidad maestra, debido a que los usuarios pueden realizar consultas sobre las instantáneas
3. Aumentan la seguridad sobre los datos, restringiendo el contenido de la instantánea en ciertos datos.

3.3.3 Configuración híbrida de instantánea y multi-maestro

La replicación multi-maestro y de instantánea pueden ser combinadas para crear una configuración híbrida esto para cumplir los requerimientos de las aplicaciones. Estas configuraciones híbridas tienen un número de localidades maestras, y múltiples localidades de instantáneas por cada una de localidad maestra.

3.4 Utilización de replicación para el logro de alta disponibilidad

Se han definido los tres tipos de replicación en la sección anterior, se debe de elegir el tipo adecuado para el logro de alta disponibilidad. Desde un punto de vista básico, la replicación deberá ser utilizada para asegurar que la información (datos) esté disponible cuando sea necesaria, en la localidad donde este replicada la información sin ninguna dependencia a la localidad original o maestra. De esta premisa, se determina que la replicación maestra cumple con este requerimiento y siendo más puntal la replicación de una localidad maestra.

3.4.1 *Failover*

La replicación de una localidad maestra, puede ser utilizada para proteger la disponibilidad de bases de datos de misión crítica. Por ejemplo, un ambiente de replicación de una localidad maestra se replica todos los datos de la base de datos primaria a una localidad subordinada (*failover),* la base de datos en la localidad subordinada disponible para consultas.

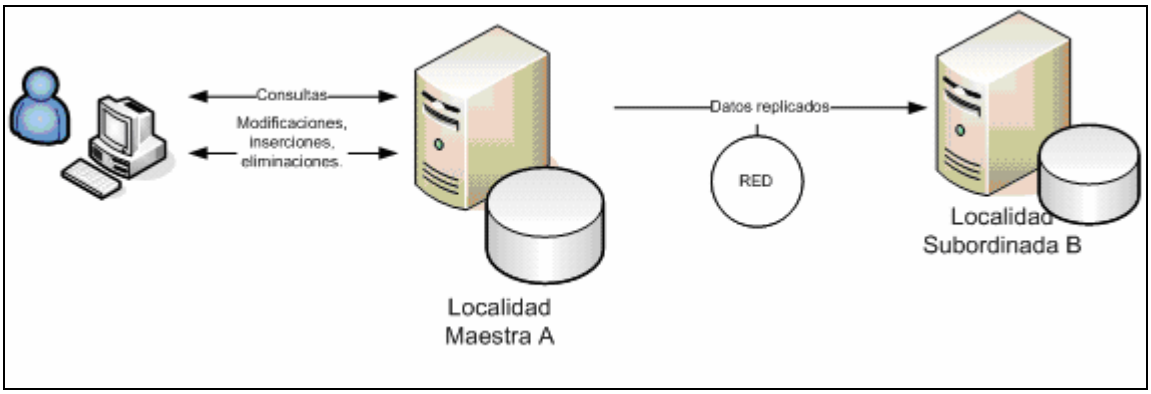
La localidad subordinada se convierte en una primaria cuando la base de datos en la localidad maestra no este disponible, debido a una caída del sistema o comunicaciones.

Al utilizar la replicación como un mecanismo para la alta disponibilidad del sistema, se tendrá una localidad maestra y una o más localidades subordinadas, y estas localidades solo están disponibles para *failover* .

Es importante subrayar que en las localidades subordinadas no se estarán efectuando operaciones de actualización, es decir las aplicaciones no se encuentran conectadas a ellas para efectuar cambios, ya que están en espera para el procedimiento de *failover*.

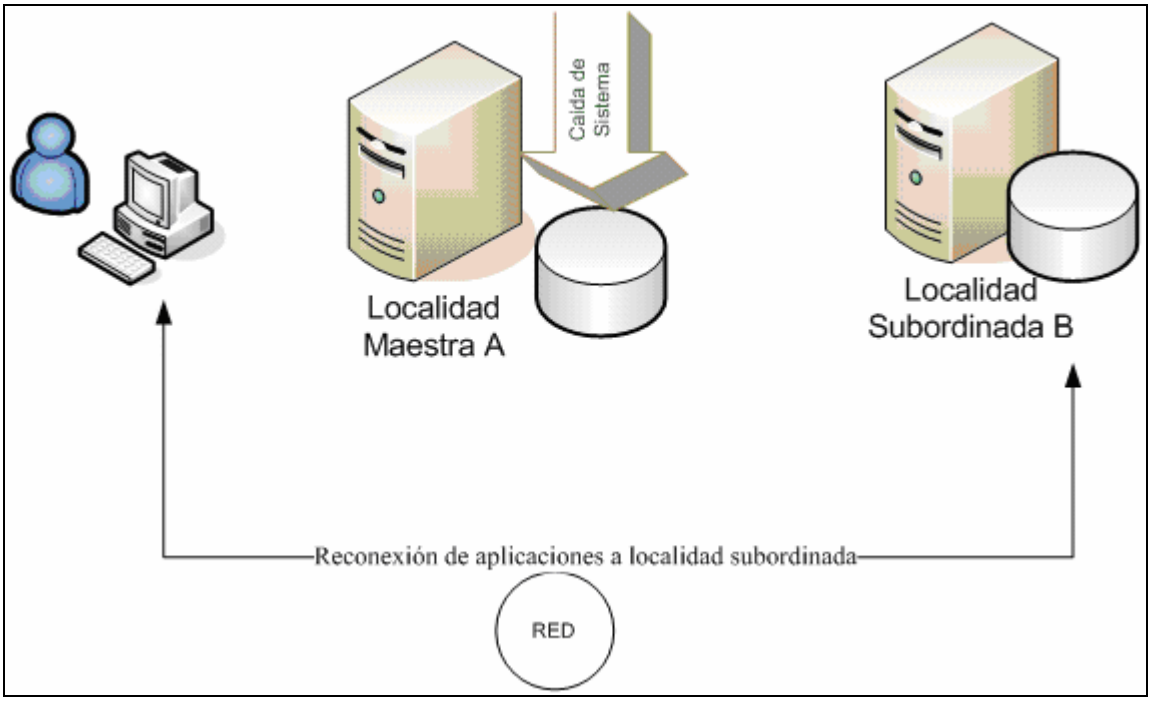
El objetivo en el caso de lograr alta disponibilidad es definir una localidad maestra y al menos una localidad subordinada, esta última esta disponible, en el momento de una falla, la arquitectura se muestra en la figura 10.

Figura 10. Replicación como mecanismo para alta disponibilidad



Todas las operaciones realizadas por las aplicaciones son efectuadas en la localidad maestra A, la cual replica los cambios efectuados en su base de datos hacia la localidad subordinada B, estos cambios son enviados en una sola vía, lo que constituye la localidad subordinada B en una localidad *failover*, en la figura 11 se muestra el momento que entra en operación la localidad subordinada.

Figura 11. Proceso de *failover* en un ambiente de replicación



En el momento que ocurre una falla en la localidad maestra A, los usuarios de aplicaciones conectadas a estas deben abrir una conexión a la localidad subordinada B, ya que en esta se encuentra una los objetos del grupo de replica de la localidad A.

3.4.2 Proceso de replicación de la información

Para el proceso de la replicación de la información entre la localidad maestra y la localidad subordinada se disponen de los siguientes tipos: asíncrona y síncrona. La replicación asíncrona, es también conocida como almacenar y continuar, captura cualquier cambio en la localidad maestra, la almacena en una cola, y en intervalos regulares, la propaga y aplica en la localidad subordinada; con esta forma de replicación, hay un periodo de tiempo antes de que todos los datos en las localidades converjan.

La replicación síncrona, también es conocida como replicación de tiempo real, aplica cualquier cambio sobre todas las localidades participantes en el ambiente de replicación como una sola transacción. Si los cambios fallan en alguna localidad, se deshace la transacción. Esta replicación asegura la consistencia de los datos en todas las localidades en tiempo real.

En el caso de lograr alta disponibilidad, requerimos que la información este disponible lo más cercano a la realidad y esto se logra únicamente con la replicación síncrona.

3.4.3 Ausencia de conflictos por resolver

Cuando se tienen un ambiente de replicación en el cual los usuarios realizan cambios a través de sus aplicaciones de manera directa sobre las localidades maestras y subordinadas, y estos son sobre la misma información se menciona que se produce un conflicto, estos conflictos son delegados a la localidad maestra para su resolución, pero para el ambiente de replicación propuesto, estos conflictos no ocurren dado que la localidad subordinada únicamente esta configurada para *failover*.

3.5 Arquitectura de replicación propuesta para el logro de alta disponibilidad

En la arquitectura que se muestra en la figura 10 cumple el objetivo de reducir el tiempo de reparación de fallos *TMRF*, ella consta de la localidad maestra en la cual las aplicaciones del sistema están modificando y consultando la información de la base de datos. En el momento de la caída de la localidad maestra, los usuarios se reconectan a la localidad subordinada y continúan con su operación normal. Esto determina lo que se conoce como sobrevivencia del sistema.

La propagación de cambios de la localidad maestra hacia la localidad subordinada es de manera síncrona,

3.6 Ventajas y desventajas

La utilización de replicación un sistema de bases de datos de alta disponibilidad presenta algunas ventajas y desventajas puede ser utilizada como alternativa a prevención de desastres.

Dentro las ventajas se tienen:

1. Administrar una o más localidades subordinadas en lugares geográficamente remotos de la base de datos en la localidad maestra.

2. En la localidad maestra, limitar el número de objetos en el grupo de réplica en aquellos que son exclusivos a las aplicaciones que se requiere dar alta disponibilidad.

3. La complejidad de la administración se reduce dado que hay ausencia de conflictos dada la alternativa que se presenta, en la cual en la localidad subordinada no se realiza ningún cambio.

4. El tiempo medio de reparación de fallos se da de manera inmediata y se reduce al tiempo de reconexión por parte de los usuarios a la localidad subordinada.

Dentro de las posibles desventajas se tiene:

1. Se requiere conocer todos los esquemas, que son utilizados por las aplicaciones, para definir de manera adecuada los grupos de réplica en cada una de las localidades.

2. Al elegir la replicación síncrona, se requiere una comunicación confiable entre la localidad maestra y la subordinada, ya que un fallo en la infraestructura de comunicaciones provoca un desfase en la convergencia de datos entre las dos localidades.

3. Los requerimientos de hardware para su implementación, requiere como mínimo un servidor con características similares al servidor de la base de datos en la localidad maestra.

4. BASE DE DATOS EN PARALELO

4.1 ¿Qué es una base de datos en paralelo?

Una base de datos en paralelo reside en un ambiente en el que se combina el poder de procesamiento de múltiples servidores interconectados, operando sobre una base de datos compartida. El software del manejador de base de datos en paralelo y los componentes de hardware utilizados se les denomina *cluster,* este unifica el poder de procesamiento de cada componente para convertirse en un ambiente robusto de computación. Un *cluster* generalmente esta comprendido de uno mas servidores, o nodos. La base de datos sobre la que opera cada nodo en el cluster es denominada base de datos en paralelo o compartida.

En un ambiente de base de datos en paralelo, todos los nodos ejecutan transacciones de manera concurrente sobre la misma base de datos. El manejador de base de datos en paralelo coordina cada acceso por parte de los nodos a los datos compartidos con el objetivo de proveer consistencia e integridad.

El unificar el poder de varios nodos ofrece algunas ventajas. Si se divide un proceso largo en subprocesos y se distribuyen estos subprocesos entre múltiples nodos, el proceso original se completará más rápido que si es ejecutado en un solo nodo. Este tipo de procesamiento paralelo es claramente más eficiente que el procesamiento secuencial.

Una base de datos puede servir como un componente importante para una solución de alta disponibilidad. Una base de datos en paralelo puede tolerar fallas reduciendo al mínimo o eliminando las caídas de sistema.

4.2 Arquitectura de los componentes de hardware o cluster

Un *cluster* comprende uno más nodos que están enlazados por una interconexión. La interconexión sirve como un canal de comunicación entre los nodos en el *cluster*. Los nodos utilizan la interconexión para la comunicación requerida para sincronizar los cambios sobre los datos compartidos. Los datos compartidos que los nodos acceden residen en dispositivos de almacenamiento.

Un nodo tiene cuatro componentes principales:

1. UCP: La unidad principal de procesamiento de la computadora que lee y escribe en la memoria principal.
2. Memoria: El componente utilizado para situar datos y programas para su ejecución.
3. Almacenamiento: Un dispositivo que almacena datos. Usualmente un almacenamiento persistente que debe ser accedido por transacciones de lectura, y escritura que alteran su contenido.
4. Interconexión: Este es un enlace de comunicación entre los nodos.

Se pueden adquirir estos componentes en varias configuraciones. Su organización determina como cada nodo en un *cluster* utiliza la memoria y el almacenamiento. Todos los *clusters* utilizan las UCP más o menos de la misma manera. Sin embargo, los componentes restantes como memoria, almacenamiento e interconexión, pueden ser configurados de diferentes maneras.

4.3 Acceso de memoria

Típicamente múltiples UCP son configurados para compartir la memoria principal. Esto permite crear un sistema escalable en rendimiento. Este tipo de sistema también más barato que una unidad de UCP con el poder equivalente de procesamiento. Existen dos configuraciones de sistemas de memoria compartida:

1. Acceso uniforme de memoria (UMA).
2. Acceso no uniforme de memoria (NUMA).

Los sistemas de memoria compartida son conocidos como "sistemas firmemente acoplados.

4.3.1 Acceso uniforme de memoria

En una configuración de acceso uniforme de memoria o UMA, todos los procesadores puede acceder la memoria principal a la misma velocidad. En esta configuración, el acceso a la memoria es uniforme. Esta configuración es también conocida como un sistema de multiprocesamiento simétrico o SMP.

4.3.2 Acceso no uniforme de memoria

En una configuración de acceso no uniforme de memoria, o NUMA, todos los procesadores tienen acceso a todas las estructuras de memoria. Sin embargo, los accesos de memoria no son iguales. En otras palabras, el costo de acceso varía dependiendo sobre que partes de memoria cada procesador acceda.

En una configuración NUMA, el costo de acceso a una ubicación específica de memoria varía de manera diferente de un procesador a otro.

El rendimiento tanto en un sistema UMA y NUMA es limitado por el ancho de banda del bus de memoria. Esto significa que conforme se agreguen mas UCP al sistema se llegará a cierto punto, en el cual el rendimiento no se incrementará de manera lineal. El punto en el cual el agregar UCPS resulta en una mejora de rendimiento mínima varía de una arquitectura a otra.

4.4 Interconexión

La interconexión consiste de un canal de un ancho de banda grande y baja latencia, que conecta cada nodo con los otros en el cluster. El canal de comunicación de alta velocidad encamina mensajes y tráfico específico de procesamiento paralelo entre los nodos, para coordinar cada acceso de datos por parte de los nodos y los recursos dependientes de datos.

El manejador de base de datos en paralelo utiliza los denominados canales de intercomunicación de procesos CIP. Esto reduce considerablemente el consumo de UPC y reduce la latencia.

Se pueden utilizar interfaces ethernet, FDDI (*Fiber Distributed Data Interface*), o hardware propietario para la interconexión. Es necesario disponer de un canal redundante de intercomunicación en caso de falla del principal. Este canal de interconexión redundante permite alta disponibilidad y reduce la ocurrencia en que la interconexión se convierta en un punto único de falla.

4.5 Almacenamiento en un sistema de cluster

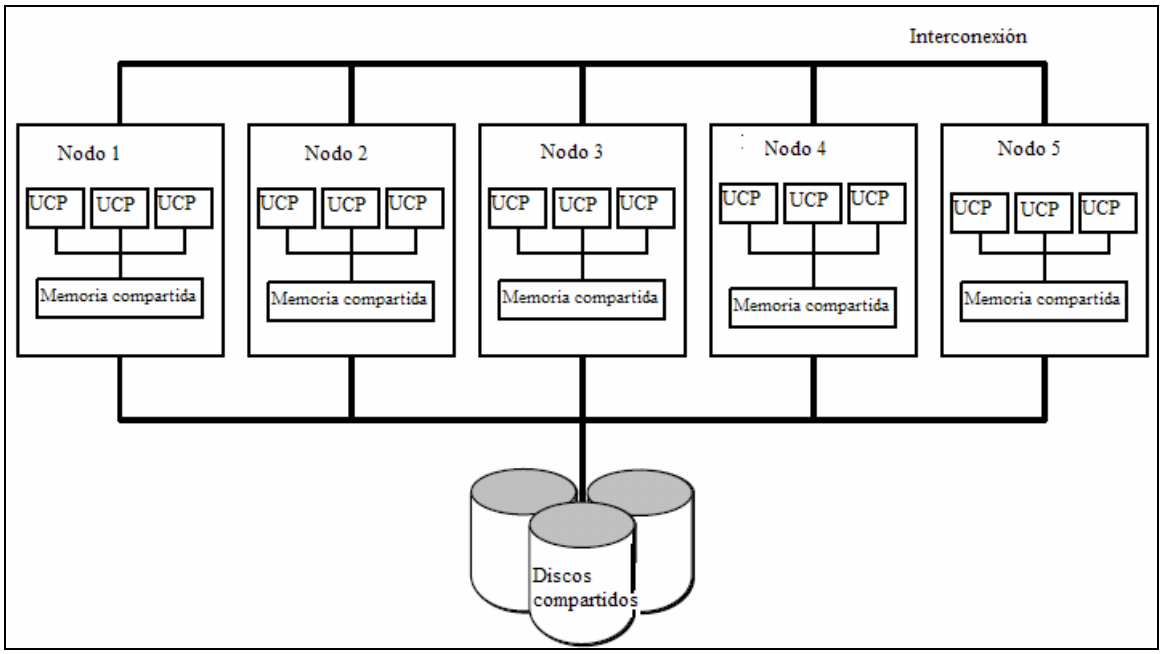
Los sistemas en *cluster* utilizan varias arquitecturas. Cada arquitectura utiliza un esquema particular de recursos compartidos que es el mejor dado un propósito particular.

Las arquitecturas de almacenamiento en sistema de *cluster* por su acceso a disco se divide en: uniforme y no uniforme. Estas arquitecturas de almacenamiento son independientes del tipo de acceso de memoria.

4.5.1 Acceso uniforme a disco

En un sistema de acceso uniforme a disco, o sistemas de discos compartido, como se muestra en la figura 12 el costo de acceso a disco es el mismo para todos los nodos.

Figura 12. Acceso uniforme a disco



El cluster en la figura 12 esta compuesto de múltiples nodos SMP. El subsistema de discos compartidos es comúnmente implementado utilizando una granja de discos interconectados utilizando interfaces SCSI de canal de fibra.

Las ventajas de utilizar procesamiento paralelo en sistemas de disco compartido son:

1. Los sistemas de discos compartidos permiten alta disponibilidad, es decir los datos están disponibles aun si un nodo falla.
2. Los sistemas de disco compartido permiten un crecimiento incremental.

4.5.2 Acceso no uniforme a disco

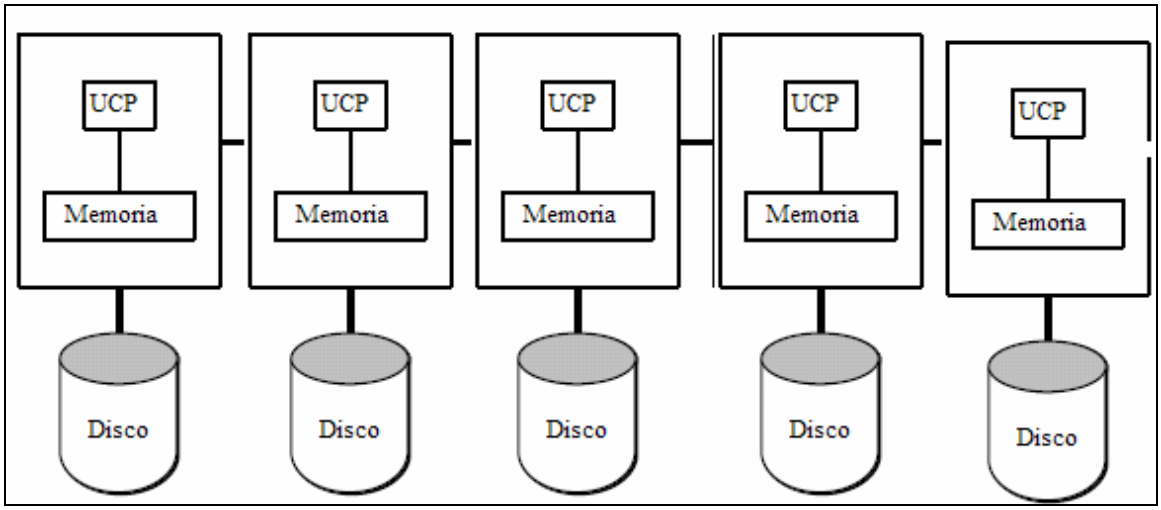
En algunos sistemas, el almacenamiento de discos esta instalado a únicamente un nodo. Para este nodo, el acceso es local. Para el resto de nodos, una solicitud de acceso a disco es enviada a una capa de software de disco virtual sobre el cual se interconecta al nodo donde el disco esta instalado. Esto significa que el costo de una lectura o escritura de disco varía significativamente dependiendo si es un acceso local o remoto. El costo asociado con leer o escribir los bloques desde un disco remoto, incluyendo la latencia de interconexión y la sobre carga de CIP, contribuyen al incremento de costo de este tipo de operación contra el costo de la misma operación utilizando una configuración de acceso uniforme a disco.

La configuración de acceso no uniforme a disco es comúnmente conocida como “sistemas de procesamiento paralelo masivo”. Para los requerimientos de alta disponibilidad, si un nodo falla, sus discos locales usualmente son reconfigurados para ser locales en otro nodo, la figura 13 ilustra este tipo de configuración.

Las ventajas de la utilización de procesamiento paralelo sobre un sistema de acceso no uniforme a disco son:

1. El número de nodos no esta limitado por las conexiones a discos físicos.
2. El espacio total de almacenamiento puede ser fácilmente escalable agregando más nodos.

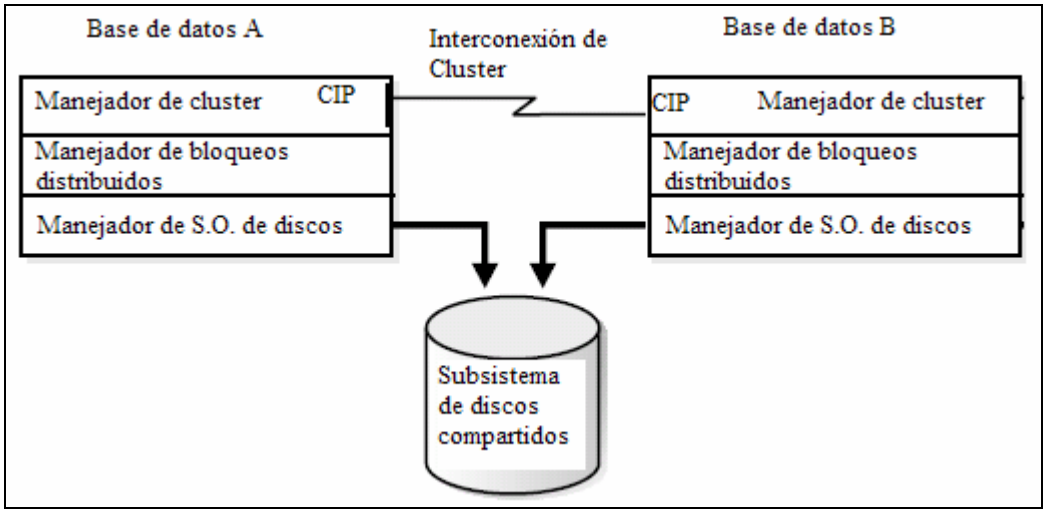
Figura 13. Acceso no uniforme a disco



4.6 Componentes del manejador de base de datos en paralelo

Basado en las arquitecturas descritas anteriormente, se necesita enumerar el software requerido para la implementación de un manejador de base de datos en paralelo. Cada proveedor de hardware implementa el procesamiento paralelo utilizando capas a nivel de sistema operativo. En la figura 14 se describen estos componentes.

Figura 14. Componentes del manejador de base de datos en paralelo



4.6.1 Manejador del *cluster*

El manejador de *cluster*, provee una vista global del *cluster* y todos los nodos que participan en la arquitectura del sistema, también controla cuando se agrega un nodo al *cluster*. El manejador de cluster generalmente es un componente propietario específico al hardware y sistema operativo elegido.

Dentro de las funciones del manejador de *cluster*, esta el monitoreo de nodos para la detección de fallas, una desconexión en el manejador de cluster, puede ocurrir por tres razones: el cliente se desconecta voluntariamente, el proceso del cliente termina, o el nodo al cual el cliente esta conectado se apaga o sufre una falla. En este último caso, si uno o más nodos fallan, el manejador de *cluster* determina que nodo esta inactivo o no funciona apropiadamente, y termina todos los procesos sobre el nodo.

Si ocurrió una falla, la recuperación es transparente a los usuarios. El manejador de *cluster* reconfigura de manera automática el sistema aislado el nodo que falló y notifica al manejador de bloqueos distribuidos del estado. El manejador de base de datos en paralelo entonces recupera la base de datos a un estado válido.

4.6.2 Manejador de bloqueos distribuido

El manejador de bloqueos distribuido es un componente integrado al manejador de base de datos en paralelo, que coordina el acceso simultáneo a la base de datos compartida y a los recursos compartidos dentro de esa base de datos, con el objetivo de mantener consistencia e integridad de los datos.

El manejador de bloqueos distribuido provee transparencia en la coordinación de acceso a los recursos por parte de las aplicaciones, estas utilizan los mismos mecanismos de bloqueo que al estar en una base de datos convencional.

El manejador de bloqueos distribuido es tolerante a fallos, lo que permite un servicio continuo y mantiene la integridad de los bloqueos en la base datos aún si, varios nodos fallan. La base de datos compartida es accesible mientras exista un nodo activo.

4.6.3 Comunicación de procesos entre nodos

Los beneficios de un manejador de bases de datos en paralelo residen en la habilidad de correr sobre múltiples máquinas. El manejador confía en la comunicación inter-procesos (CIP) para lograrlo.

La comunicación inter-procesos define los protocolos e interfaces requeridas por el manejador para transferir mensajes entre cada segmento de memoria compartida, en cada nodo. Los mensajes constituyen la unidad fundamental de comunicación en esta interfaz. La principal funcionalidad de la comunicación inter-procesos es que está diseñada para el envío de mensajes de manera asíncrona, este mecanismo de comunicación está diseñado para el envío y recepción de mensajes tan rápido como el hardware lo permita.

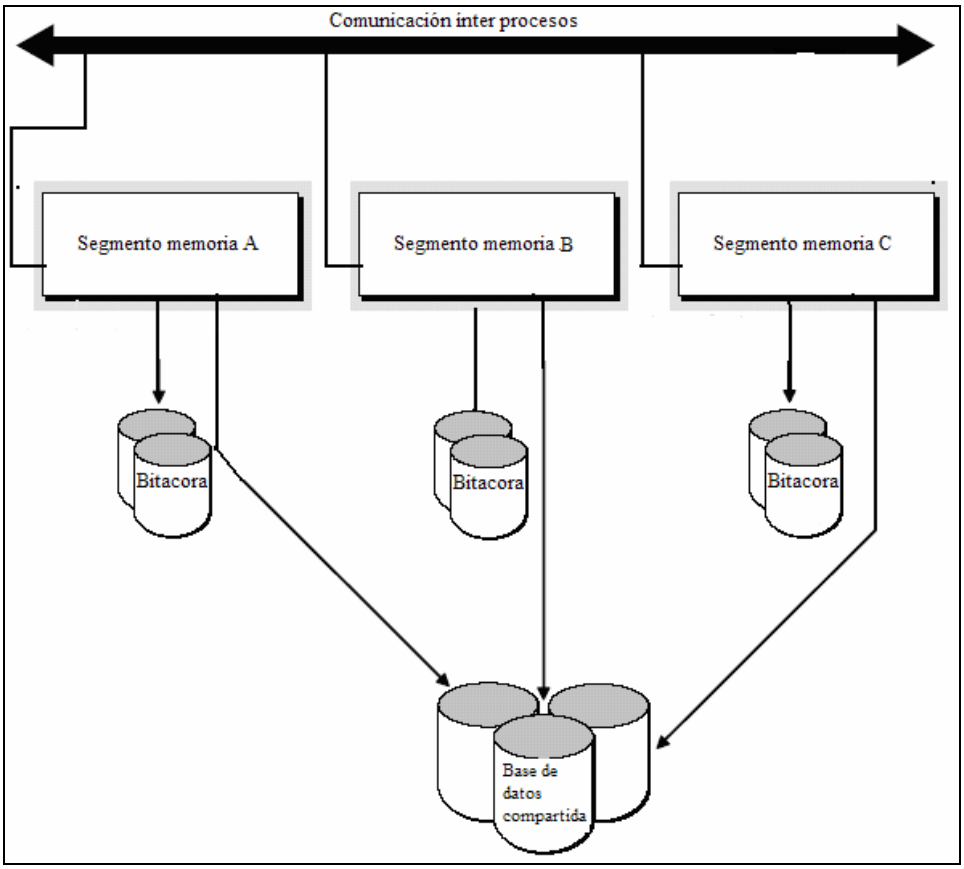
4.6.4 Subsistema de discos

Además de las capas dependientes de sistema operativo el manejador de base de datos requiere que todos los nodos tengan acceso simultáneo al almacenamiento. Esto permite tener múltiples nodos con acceso concurrente a la misma base de datos.

4.7 Arquitectura de la base de datos en paralelo

Un ambiente de base de datos en paralelo, requiere de los componentes mencionados en las secciones anteriores, en la figura 15 se muestra arquitectura de la base de datos en paralelo o compartida, en ella se puede observar que cada nodo posee un segmento de memoria compartida y una bitácora que reside en almacenamiento propio, y la base de datos reside en subsistema de discos compartidos.

Figura 15. Base de datos en paralelo o compartida



4.8 Configuraciones de alta disponibilidad utilizando base de datos en paralelo

Un ambiente de base de datos en paralelo, provee capacidades de alta disponibilidad, en este ambiente se puede variar la configuración de tres posibles maneras:

1. Configuración de n – nodos.
2. Configuración básica de alta disponibilidad.
3. Configuración compartida de alta disponibilidad.

4.8.1 Configuración de n – nodos

Esta es la configuración que se muestra en la figura 15, las transacciones de los clientes son procesadas en cualquier nodo del *cluster*, y las sesiones de los usuarios pueden estar balanceadas en nodos en particular. El tiempo de respuesta es optimizado por la disponibilidad de recursos de *cluster*, tales como UCP y memoria, distribuyendo la carga entre los nodos del *cluster*, se crea un ambiente de alta disponibilidad. Cuando ocurre un evento de falla en un nodo, otro nodo ejecuta las tareas de recuperación necesarias para la reconexión de los usuarios en los n-1 nodos restantes dentro del *cluster*.

4.8.2 Configuración básica de alta disponibilidad

La base de datos en paralelo en una configuración básica de alta disponibilidad, consiste en establecer un nodo primario el cual acepta todas las conexiones de los usuarios, mientras un nodo secundario únicamente aceptará las conexiones de usuarios cuando el nodo primario falle. Cuando esto sucede el nodo secundario se convierte en primario, solamente después que el manejador de cluster informa de la falla del nodo primario pero antes de que se de una configuración del manejador de bloqueos distribuidos, quien es el encargado de cualquier tarea de recuperación de transacciones y liberación de bloqueos. La redirección de las aplicaciones de un nodo a otro sucede de manera transparente.

En esta configuración, ambos nodos corren de manera concurrente, tal como un ambiente de n – nodos. Sin embargo, los usuarios de las aplicaciones de base de datos únicamente se conectan al nodo designado como primario.

4.8.3 Configuración compartida de alta disponibilidad

Al configurar la base de datos en paralelo en n-nodos, utiliza de manera óptima los recursos del *cluster*. Por otro lado, la inversión financiera puede ser prohibitiva en una configuración básica, al mantener en espera un nodo para *failover*. Estas situaciones permiten que la configuración compartida de alta disponibilidad sea una opción aceptable.

Este tipo de configuración, puede tener varios nodos en los cuales se corren módulos separados de aplicación o servicios de aplicación; que comparten una misma base de datos en paralelo. Se pueden separar los módulos, por nodo y diseñar el mecanismo de *failover* entre uno y otro nodo. Por lo que las aplicaciones son las responsables de la redirección de los usuarios en el momento de detectar una falla.

4.9 Ventajas y desventajas

Dentro de las ventajas en la implementación de una base de datos en paralelo, se tienen los niveles más altos de disponibilidad que las implementaciones de los capítulos anteriores. El operar uno o más nodos sobre una misma base de datos compartida permite reducir el tiempo medio de reparación de fallos TMRF considerablemente con efectos mínimos en el rendimiento de la aplicación. En un ambiente de *cluster* de n- nodos, se provee alta disponibilidad de hasta n-1 fallas en los nodos, sin perder operación del sistema, en otras palabras todos los usuarios tienen acceso a la información; mientras exista al menos un nodo disponible en el *cluster*.

Las desventajas que enfrenta la implementación de una base de datos en paralelo lo constituye la complejidad de selección de componentes de hardware y software, tanto a nivel técnico como económico, ya que constituye una inversión elevada.

CONCLUSIONES

1. La protección contra fallas tiene un costo monetario, al igual que las caídas de sistema, por lo que es importante encontrar el balance entre invertir en equipo y sistemas redundantes, y la pérdida financiera ocasionada por el tiempo de caída del sistema, por lo que medir la disponibilidad del sistema es obligatorio y clave, siendo la relación del tiempo promedio entre fallas y la suma de este tiempo y el tiempo promedio de reparación de fallas conocido como disponibilidad.

2. Un sistema de base de datos de alta disponibilidad, no es más que la implementación de características de un DBMS, para lograr un nivel de disponibilidad alta.

3. Un sistema de base de datos en espera es una alternativa para proveer alta disponibilidad, en el cual se tienen dos servidores uno con la base de datos primaria y el otro con la base de datos en espera.

4. Un ambiente de base de datos en espera administrado en modo recuperación posee la flexibilidad de promoverse en una base de datos de lectura, para ser consultada por los usuarios.

5. La activación de una base de datos en espera la convierte de manera permanente en una base de datos primaria, por lo que es necesario redefinir la arquitectura para proveer alta disponibilidad en el sistema.

6. La utilización de replicación para el logro de alta disponibilidad, requiere un análisis de las tablas empleadas por los sistemas para poder construir réplicas en la base de datos secundaria.

7. El arquitectura para un sistema de bases de datos de alta disponibilidad, utilizando replicación, esta formado por una localidad maestra y una o más localidades subordinadas las cuales solo están disponibles para *failover*.

8. Un sistema de base de datos en paralelo provee mecanismos de alta disponibilidad, desarrollados por el fabricante del DBMS para este fin. Por lo que esta arquitectura se constituye como la más eficiente.

9. Un sistema de base de datos en paralelo se combina dos o más servidores, el DBMS y arreglos de disco compartidos conociendo todos estos componentes como *cluster*.

10. En un sistema de base de datos en paralelo en configuración n-nodos, los usuarios del sistema pueden trabajar sobre cualquiera de los nodos en el sistema, sobre la misma base de datos.

11. En un sistema de base de datos en configuración básica de alta disponibilidad, se tiene un nodo primario el cual acepta todas las conexiones de los usuarios mientras un nodo secundario aceptará las conexiones únicamente cuando el primario falle.

RECOMENDACIONES

1. Para implementar alta disponibilidad en un sistema, es necesario comprender el nivel de disponibilidad que los usuarios requieren en las aplicaciones y su impacto financiero, por lo que se debe definir un acuerdo de nivel de servicio donde se debe realizar una revisión de las aplicaciones críticas y los horarios de operación, así como el costo de una eventual falla del sistema, tanto desde la perspectiva del negocio como a nivel de tecnología.

2. La revisión de los recursos tecnológicos actuales de una compañía es obligatoria para un director de tecnología, ante el desafío de elevar los niveles de disponibilidad de los sistemas.

3. Para la implementación de un sistema de base de datos de alta disponibilidad de bajo presupuesto, la replicación y la base de datos en espera, constituyen una excelente alternativa.

4. Para un sistema de base de datos en espera, la mejor opción en operación se constituye un modo administrado de recuperación.

5. Un sistema de base de datos en paralelo, se constituye como el ambiente ideal para el logro de alta disponibilidad, ya que ha sido diseñada por los fabricantes del DBMS para ese fin, por lo que se debe evaluar la inversión tanto en software como en hardware.

6. Para un sistema de base de datos en paralelo, la opción más productiva es una configuración de n-nodos.

BIBLIOGRAFÍA

1. Evan Marcus y Hal Stern. ***Blueprints for High Availability Design Resilient Distributed Systems 2a. ed.*** E.U.A.: John Wiley & sons, Inc., 1999.

2. Floyd Piedad y Michael Hawkins. ***High Availability Design Techniques and Processes by 1a. ed***. E.U.A.: Prentice Hall, 2001.

3. Steven H. Spewack. ***Enterprise Architecture Planning: Developing a Blueprint for Data, Applications, and Technology***. E.U.A.: John Wiley & sons, Inc., 1993.

4. Thomas M. Conolly y Carolyn E. Begg. ***Databases Systems: A Practical Approach to Design, Implementation, and Management 3a. ed.*** E.U.A.: Pearson Addison Wesley, 2001.

5. Mohammad Shahidehpour y Yaoyu Wang. ***Applications of Parallel and Distributed Processing***. E.U.A.: John Wiley & sons, Inc., June 2003.

6. Raghu Ramakrishnan y Johannes Gehrke. ***Database Management Systems 3a. ed***. E.U.A.: McGraw-Hill Companies, August 2002.

7. Date, C.J. **Introducción a los sistemas de bases de datos volumen 1 5a. ed.** Traducción: Roberto Escalona García. E.U.A.: addison/Wesley Iberoamericana, 1993.