Contents

1 Literal

1.1 Transacción

Es una sucesión o secuencia de acciones (u operaciones) llevadas a cabo como una unidad lógica de trabajo simple.

Sus propiedades se conocen como ACID (en inglés):

- Atomicidad - Atomicity Debe ser una unidad que ya no se pueda fraccionar, por lo que se ejecuta completamente o no se ejecuta.
- Consistencia - Consistency Debe llevar a la base de datos de un estado consiste a otro (los programas deben ser correctos).
- Aislamiento - Isolation Las operaciones de una transacción deben aislarse de las llevadas a cabo por otras transacciones -concurrentes-.
- Durabilidad - Durability Una vez terminada la transacción con éxito, sus efectos deben hacerse permanentes en la base de datos.

Comienza con la ejecución satisfactoria de una intrucción BEGIN TRANSACTION. Termina con la ejecución satisfactoria de una instrucción COMMIT o ROLLBACK.

- COMMIT: es un punto de confirmación, corresponde a una unidad de trabajo lógica o sea un punto en el que la base es consistente.
- ROLLBACK: regresa al estado que estaba antes del BEGIN TRANSACTION.

1.2 Concurrencia

Se refiere al hecho de que un SMBD permita que más de una transacción puedan acceder a una misma base de datos a la vez. Sub beneficios son:

- Aumentar la productividad, con más transacciones ejecutadas por minuto.
- Aumentar la utilización de la CPU y el control del disco.
- Reducir el tiempo medio de respuesta de transacciones.

¿Por qué es necesario el control de la concurrencia? Porque pueden surgir problemas si las transacciones concurrentes se ejecutan de manera no administrada. Su objetivo es conservar la integridad de los datos.

1.2.1 Puntos de vista

- Pesimista: Obliga a una transacción a esperar a que se resuelva el conflicto que pueda o ponga en riesgo la concurrencia para dejarle continuar cuando el conflicto haya sido resuelto.
- Optimista: Permite la ejecución de la transacción como si no ocurriera ningún conflicto y resuelve éste al final del commit. Generalmente se emplean sellos de tiempo y copias de los elementos de la transacción.
- Mixto: Combina diferentes controles de concurrencia a diferentes objetos y tipos de datos en una misma transacción.
- Semi-optimista: Es una variante del modo mixto que no detiene a la transacción hasta que esta termina.

1.2.2 Actualización perdida

También conocida como “Lost Update”. T1 y T2 acceden a los mismos datos, tienen sus operaciones intercaladas de modo que actuaizan de manera incorrecta el valor de algún dato.

```
T1.[ SELECT X ];
T1.[ X= X - N ];
   T2.[ SELECT X ];
   T2.[ X= X + M ];
T1.[ UPDATE X ];
T1.[ SELECT Y ];
   T2.[ UPDATE X ];
T1.[ Y= Y + N ];
T1.[ UPDATE Y ];
```

1.2.3 Temporal Update

También conocida como “Temporal Update”. T1 actualiza un elemento X de la BD y luego falla, pero antes de que se restaure el valor original de X, T2 tiene acceso al «valor temporal» de X.

```
T1.[ SELECT X ];
T1.[ X= X - N ];
T1.[ UPDATE X ];
   T2.[ SELECT X ];
   T2.[ X= X + M ];
   T2.[ UPDATE X ];
T1.[ SELECT Y ];
T1.[ ERROR ];
T1.[ ROLLBACK ];
```

1.2.4 Lectura no adecuada

También conocida como “Dirty read”. Una transacción T2 calcula una operación de resumen sobre varios registros (suma, por ejemplo), mientras otra transacción T1 actualiza registros involucrados.

```
   T2.[ SET S = 0 ];
   T2.[ SELECT X ];
   T2.[ S = S + X ];
T1.[ SELECT X ];
T1.[ X= X - N ];
T1.[ UPDATE X ];
   T2.[ SELECT Y ];
```

```
   T2.[ S = S + Y ];
T1.[ SELECT Y ];
T1.[ Y= Y + N ];
T1.[ UPDATE Y ];
```

1.2.5 Lectura no repetible

También conocida como “No repeatable read”. T2 lee un elemento X dos veces y la transacción T1 modifica dicho valor X entre las dos lecturas. T2 recibe diferentes valores para el mismo elemento.

```
T1.[ SELECT X ];
T1.[ X= X - N ];
   T2.[ SELECT X ];
T1.[ UPDATE X ];
T1.[ SELECT Y ];
   T2.[ SELECT X ];
T1.[ Y= Y + N ];
T1.[ UPDATE Y ];
```

1.3 Control concurrencia

Los principales mecanismos de control (bloqueos) de concurrencia son tres:

- Semáforos: Candados que prohíben accesos que puedan provocar conflictos de acceso.
- Sellos de tiempo: Impiden acciones sobre los datos.
- Multiversión: Guardar múltiples versiones de los objetos de datos.

Los bloqueos, se pueden clasificar también como:

- Bloqueo compartido: Posibilidad de leer elementos de la BD, pero no actualizarlos. Varias transacciones pueden mantener simultáneamente bloqueos compartidos.
- Bloqueo exclusivo: Posibilidad de leer y actualizar elementos. Sólo una transacción puede tenerlo a la vez.
- Interbloqueo : Se puede definir como el bloqueo permanente de acceso a recursos, generado por un conjunto de procesos que compiten por el acceso. No existe una solución eficiente para el caso general. Ante la existencia de un caso como este, -normalmente- se elige un proceso “víctima” que es terminado forzadamente (kill).

1.4 Recuperación

Dado algún error de software o hardware y se haya dañado la BD, se puede regresar a como se encontraba antes del error. Supone el restablecimiento de la BD a un estado correcto anterior al fallo. Se debe garantizar la reconstrucción a partir de otros datos almacenados redundantemente en algún otro lugar del sistema. Los daños pueden ser por un error humano, ataques, fallos del equipo o incluso catástrofes naturales.

1.5 Prevenimms

Existen opciones variadas en la prevención de perdida de datos:

- De Software: Respaldos, ya sean manuales o automáticos. Zonas alternas (internas) de respaldo. Antivirus y Antimalware. Detectores y prevención de intrusos y ataques.
- De Procedimiento: Extracción de los respaldos a un lugar distinto, seguro. Pruebas de recuperación.
- De Infraestructura: Suministro de energía ininterrumpido. Prevención de incendios.

1.6 Fragmentación

Se refiere al particionamiento de la información, distribuyéndola a diferentes entidades. Es un fragmento de las relaciones. Su finalidad ha de ser siempre la búsqueda de un mejor rendimiento. Esto favorece a la ejecución concurrente de transacciones.

> No. La f. interna se refiere a recursos asignados inútiles y la f. externa se refiere a recursos sin libres que no se pueden asignar por su tamaño -pequeño-.
>
> Ejemplo: dado un archivo binario, la f. interna serían registros "borrados" lógicamente, que no se pueden reutilizar en el archivo. La f. externa serían espacio en el disco duro entre asrchivos, que no se pueden asignar a otro archivo.

1.7 Replicación

- Definición.- La replicación es una copia de una base de datos a otra con la que se sincroniza y mantiene coherencia. Supone que la copia se lleva a un servidor (físico) en otro lugar, por seguridad.
- Copia de seguridad.- Este método puede utilizarse para hacer un backup de la base de datos y tenerla así asegurada.
- Alta disponibilidad y Escalabilidad.- Es muy usado por empresas que tienen locaciones en diversos lugres y desean tener la misma base de datos en todas sus locaciones, en vez de un sola instancia central, para que sea más eficiente el acceso.
- Beneficios:
  - Disponibilidad
  - Fiabilidad
  - Rendimiento
  - Reducción de carga

El archivo de **log** es gestionado por el servidor de base de datos en el que se registran todas las sentencias SQL de modificación de datos o estructura.

1.8 Reparación

Las bases se pueden dañar por: fallas de hardware, apagones inesperados del servidor y transacciones interrumpidas incorrectamente.

Las herramientas de reparación en una base de datos intentan arreglar algunos tipos de daños, es decir: Se orientan a tablas, consultas e índices. No reparan formularios, informes, macros ni módulos dañados; dejando a estos intactos en la base de datos ya reparada. Antes de realizar cualquier procedimiento de reparación es conveniente realizar un respaldo de la base de datos dañada -si se puede- o hacer una copia física de los archivos.

1.9 Compactación

Es el reacomodo de los datos para evitar espacios vacíos. El espacio ocupado por elementos ya eliminados se marca como disponible para agregar en ella nuevos objetos o registros. Vuelve a generar las estadísticas de las tablas utilizadas en el proceso de optimización de la consulta.

¿Qué sucede? Se crea una nueva base de datos y nuevos índices. Las páginas de las tablas se reorganizan. Los datos se juntan en regiones, no dejando espacio libre.

Limitaciones:

- Espacio necesario, en el disco para la base de datos actual y la comprimida.
- Debe la base de datos debe de deshabilitarse durante este periodo.

- Ocultar

Beneficios:
- Las consultas se eficientan.
- La fragmentación se hace más fácil para llevarse acabo.
- Se actualizan los registros ya creados.

Limitaciones:
- Se requiere espacio en el disco para alojar la base de datos actual y la compactada.
- La base de datos no se encuntra accesible durante el período de compactación.

> Dado que la compactación presenta varios beneficios, ¿se debe realizar MUY a menudo?
>
> Retroalimentación
>
> Qué tan frecuentemente se debe hacer depende de muchos aspectos, la frecuencia de operaciones en la base cambia de una a otra, pues las frecuencias no son las mismas, y por ello, la fragmentación -de espacios de memoria- no se complica a la misma velocidad.
>
> Además, si compactar implica ELIMINAR de forma definitiva datos, y si no se cuenta con un respaldo con la misma frecuencia o superior, se corre el riesgo de peder datos.
>
> Por lo tanto, no necesarimente a menudo es algo bueno.

2 Ejercicios

1. Paises: Escriba un query que liste todos los países -sin repetir- de la tabla customers:

   ```
   SELECT DISTINCT country FROM customers ORDER BY country;
   ```

2. Estados: Dado el país de Estados Unidos, liste -sin repetir- todos los estados de la tabla de customers.

   ```
   SELECT DISTINCT state FROM customers WHERE  (country='USA') ORDER BY state;
   ```

3. F horizontal: Dado el país Estados Unidos y el estado California, liste a todos los campos de la tabla customers. Realícelo también para el estado de Massachusetts.

   ```
   SELECT * FROM customers WHERE (country='USA') AND (state='CA');
   SELECT * FROM customers WHERE (country='USA') AND (state='MA');
   ```

4. F vertical: Dado el país de Estados Unidos y el estado de California, genere una vista que solamente muestre los campos customerNumber, customerName, salesRepEmployeeNumber, que se llame vUSA_CA_Emp.

   Realice algo similar, pero ahora liste customerNumber, customerName, creditLimit, y que la vista sea nombrada como vUSA_CA_Lim.

   ```
   CREATE VIEW vUSA_CA_Emp AS SELECT customerNumber, customerName,
   salesRepEmployeeNumber FROM customers WHERE  (country='USA')  AND (state='CA');

   CREATE VIEW vUSA_CA_Lim AS SELECT customerNumber, customerName, creditLimit FROM
   customers WHERE  (country='USA')  AND (state='CA');
   ```