Transacción

El concepto de transacción es muy importante en el entorno de base de datos. Se define como transacción a una operación atómica de modificación de información, es decir, no es posible descomponerla en partes. Si no se aplicaran todos los cambios realizados, la base de datos no quedaría íntegra.

Supongamos que tenemos una tabla donde registramos los pagos de nuestros clientes, y además hemos clasificado a nuestros clientes entre quienes nos deben y quienes no. En el momento de ingresar un pago, si determinado cliente ha saldado su deuda, también deberíamos actualizar la columna donde guardamos su estado de cuenta.

Es necesario que la base de datos **registre las dos operaciones** o, en caso de que ocurra un problema, no haga **ninguna** de las dos.

Oracle administra perfectamente ese concepto. Cuando efectuamos operaciones de modificación de datos (`insert`, `update` o `delete`), Oracle no confirma los mismos hasta que no le damos la instrucción explícita de hacerlo (*commit*). Si hay un problema entre las distintas operaciones, o al realizar el commit, el motor de la base de datos deshace la operación completa (*rollback*).

Si programamos de manera correcta, es imposible tener un problema de consistencia de información por esta causa, por más que una transacción comprenda la modificación de miles de registros.

En el caso de una falla, un error o por requerimiento de la aplicación, tenemos la posibilidad de utilizar marcadores (*savepoint*) para no deshacer la transacción completa, sino deshacer sólo los cambios posteriores a la etiqueta referida por la instrucción `ROLLBACK`.

SCN (System Change Number)

Cada vez que la base de datos realiza una confirmación (*commit*) de una transacción, el motor de la base de datos efectúa las siguientes tareas:

- Almacena un número de transacción único (SCN) en una tabla interna de transacciones del segmento de rollback.
- Por medio del proceso LGWR, aplica al archivo redo on line la información almacenada en los buffer de los archivos redo. En el archivo redo on line almacena también el SCN.
- Libera los bloqueos sobre registros.
- Marca la transacción como completa.

El identificador único de transacciones (SCN) se utiliza en los procesos de recuperación de una base de datos. El mismo identifica qué transacciones de la bitácora tenemos que aplicar. Dichas transacciones derivan de la diferencia entre el SCN de la base iniciada y los SCNs almacenados en la bitácora de transacciones.

Consistencia

Oracle garantiza que durante la ejecución de una consulta la información devuelta es consistente con un determinado período de tiempo. Es decir, cuando ejecutamos una instrucción de consulta, no importa cuánto demore Oracle en ejecutarla; todo el conjunto de datos devuelto será la información de las tablas al momento de iniciar la misma. Es como si el motor de la base de datos sacara una foto de las tablas involucradas en una consulta, entregándonos así una lectura consistente en el tiempo. Para brindar tal capacidad, Oracle mira el SCN (número de cambio del sistema) e ignora en la consulta aquellas transacciones con SCN mayor a dicho número. Oracle realiza esa tarea por defecto, sin necesidad de utilizar ninguna instrucción, y es válida para una instrucción SQL. Si quisiéramos que la base de datos nos devuelva una visión íntegra que involucre más de una sentencia SQL, deberíamos agregar alguna instrucción adicional a nuestro código.

Para llevar adelante esta tarea, Oracle no bloquea las tablas involucradas. En una consulta, la única manera de bloquear registros de una tabla es pedirlo explícitamente. En cualquier otro caso, Oracle no bloquea ninguna tabla, por más larga y compleja que sea la consulta.

Para implementar una lectura consistente, Oracle utiliza la información almacenada en los rollback segment. Es allí donde guarda, de manera transitoria, los valores anteriores de los registros modificados.

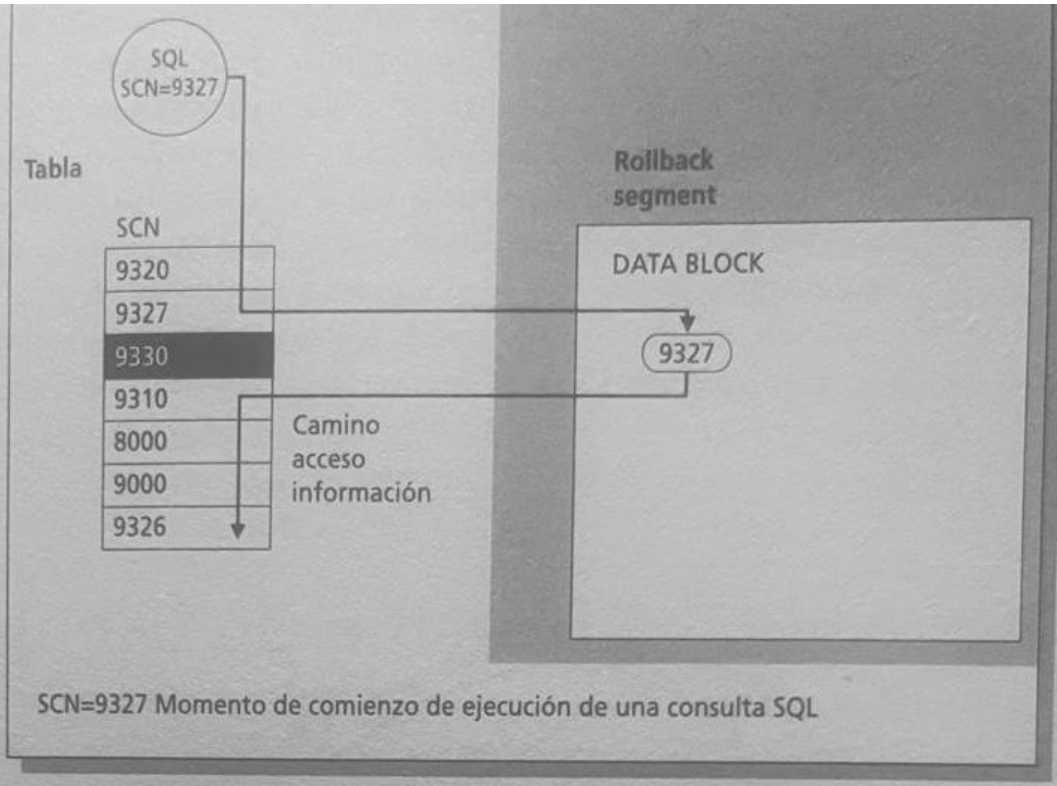

Figura 13. *Para que Oracle resuelva consultas complejas en forma íntegra, será necesario que los segmentos de rollback tengan un tamaño considerable.*

Control de concurrencia

En un ambiente multiusuario es necesaria la función de bloqueos de recursos para asegurar que dos usuarios no modifiquen los mismos registros al mismo tiempo.

Existen dos niveles de bloqueo en Oracle: al nivel de registro y al nivel de tabla. Hay instrucciones para bloquear uno o más registros utilizando una sentencia de consulta; existe también una instrucción para bloquear una tabla. No obstante, la base de datos está pensada desde su concepción para administrar una múltiple concurrencia de usuarios, por lo que no es necesario realizar ningún bloqueo manual para asegurar la consistencia de la información.

A diferencia de otras bases de datos relacionales, Oracle solamente efectúa el bloqueo de los registros realmente necesarios. Algunos productos bloquean la página completa donde está el registro, cuando sólo necesitan realizar un bloqueo de un registro. Tal forma de trabajo anula recursos innecesarios e incrementa la probabilidad de demoras por contención en la resolución de sentencias SQL.

Cuando actualizamos un registro, Oracle lo bloquea hasta que se confirme la transacción (*commit*). Si otro usuario quiere modificar el mismo

registro, Oracle lo mantendrá así hasta que termine la primera transacción. El motor de base de datos no permitirá eliminar o modificar estructura de tablas si hay en ellas transacciones pendientes de confirmación. Estos mecanismos son automáticos y nativos de la base de datos.

El motor de la base de datos soluciona también los problemas de punto muerto. Este tipo de problemas se soluciona cuando un recurso A modificó el registro X y luego quiere modificar el registro Y, el cual fue modificado por un recurso B que quiere, a su vez, cambiar el registro X. En la **Figura 14** se ve un ejemplo de punto muerto.

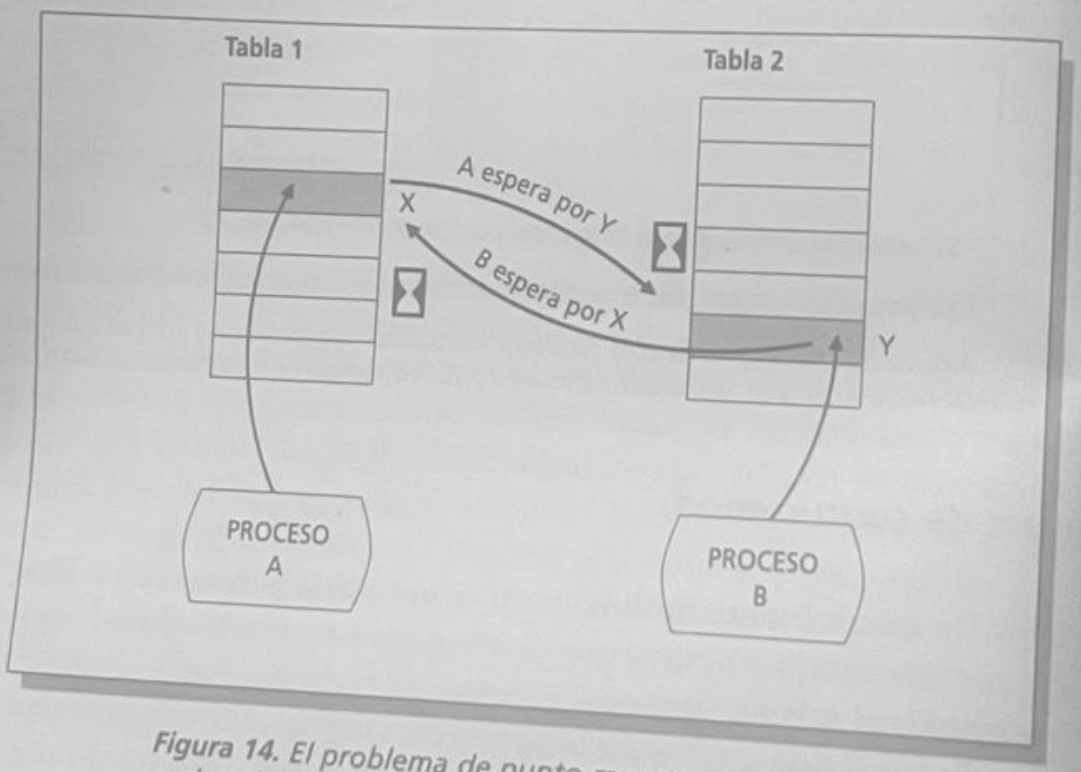

Figura 14. El problema de punto muerto es evitable si se define un orden de bloqueo de las tablas.

Una vez detectado el inconveniente, Oracle deshace (*rollback*) una de las transacciones y devuelve un error sólo para esa transacción.

Diccionario de datos

Todas las definiciones de una base de datos y su estructura lógica de tablas, columnas, procedimientos, vistas, índices, etc. se almacenan en tablas creadas en el momento en que se instala un servidor Oracle. Es posible acceder a esta información almacenada en formato relacional, tablas y columnas, para consultar lo que sea necesario. Q

registra en tablas temporales la información de los procesos que se están ejecutando en memoria. La forma de acceder a esa información es con la utilización de lenguaje SQL.

Cada vez que la base de datos recibe un SQL para ejecutar, valida en estas tablas que la sentencia esté escrita correctamente, que todas las tablas, columnas y funciones existan en la base de datos, y que el usuario tenga acceso a ellas. Esta información es mantenida en la SGA, ya que es accedida permanentemente.

Cuando se crea una base de datos, todas esas tablas residen en un espacio de tablas llamado SYSTEM. Se recomienda no crear en dicho espacio objetos que no pertenezcan al diccionario de datos.

Integridad de la base de datos

Existen restricciones provistas por la base de datos para asegurar que la información almacenada cumpla con condiciones definidas a priori. Estas herramientas nos garantizan la calidad, integridad y coherencia de la información almacenada. Actúan además como control cruzado de las aplicaciones, ya que si alguna aplicación quiere insertar datos incoherentes (por ejemplo, un descuento mayor al 100%), es la base de datos la que le impedirá la transacción.

Clave primaria

El concepto de clave primaria es básico para el funcionamiento de una base de datos relacional. Por medio de la clave primaria es posible identificar un registro unívocamente. Todas las tablas deben tener una clave primaria. Si no es así, sería indicación de que hay un error grosero en el diseño del modelo de datos. Oracle nos permite identificar la clave primaria de una tabla y controla que no se repita la misma combinación de valores para todas las columnas que formen parte de la clave primaria. RDBMS crea un índice para asegurar el cumplimiento de esa restricción.

Clave alternativa

Esta clave es otra posible combinación de columnas que identifican un registro de forma única. Por ejemplo, si hacemos un sistema que

registre los automóviles de determinada región, la clave primaria será su número de patente. No obstante, la combinación de marca del automóvil y número de serie nos devolverá también a lo sumo una fila por lo que esta combinación de columnas será una clave alternativa.

Clave foránea

En el modelo relacional, algunas columnas almacenan valores que sirven para relacionarse con otra tabla. Por ejemplo, si tenemos una tabla de facturas, la columna de código de cliente nos proporciona la clave de acceso para ir a la tabla de clientes y saber su nombre y dirección.

Es muy importante la definición explícita de las claves foráneas, ya que de esta manera la base de datos nos asegura que no podemos hacer referencia a una clave primaria inexistente en la tabla relacionada. En el ejemplo, no podríamos ingresar una factura con un código de cliente que no existiera en la tabla de clientes. Además, el motor de la base de datos no permitirá eliminar a un cliente de la tabla si existen facturas que hacen referencia a él. Para poder definir una clave foránea, deberemos tener declarada la clave primaria en la tabla a la que hacemos referencia.

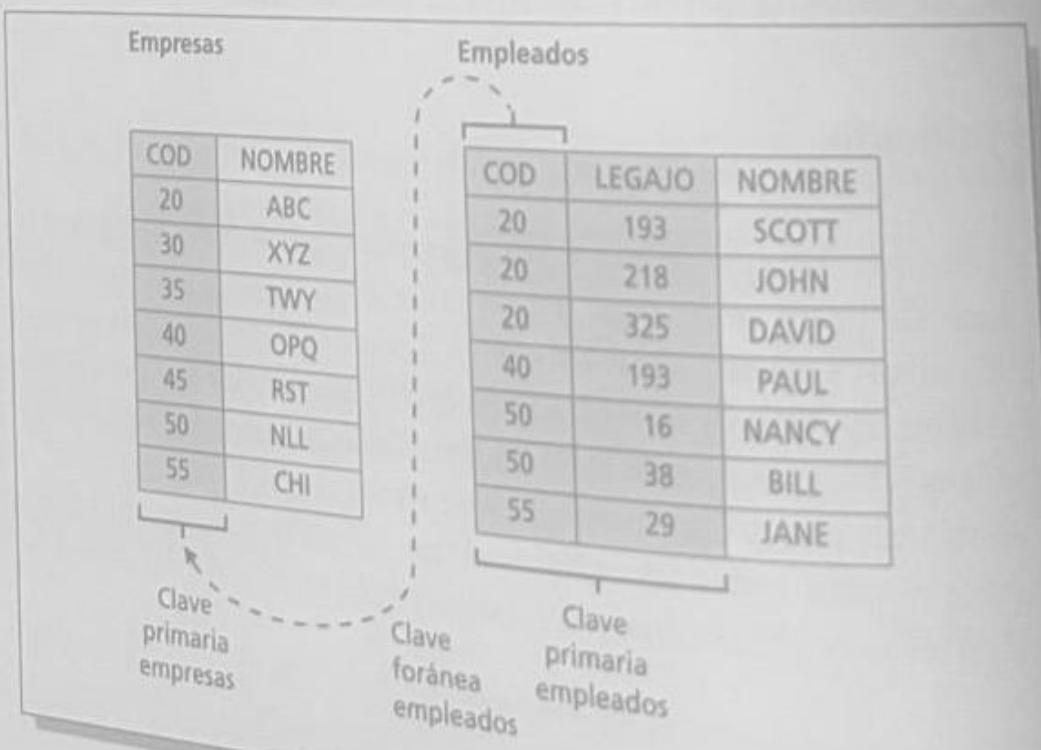

Figura 15. Ejemplo de clave primaria

Restricciones

La base de datos nos permite también el agregado de restricciones en los valores de algunas columnas. Por ejemplo, si tenemos una tabla donde registramos los pedidos, podríamos poner como restricción que la fecha de entrega sea igual o mayor que hoy, o para la lista de precios, que los precios sean mayor a 0 y que los descuentos tengan un valor máximo y mínimo.

Cualquier transacción que intente actualizar o insertar un valor no permitido será rechazada por la base de datos.

Disparadores

Para el control de integridad de información es también muy útil la utilización de disparadores (*triggers*). Por medio de ellos es posible agregar cualquier tipo de validación, por más compleja que sea, y así chequear en la base cualquier tipo de condiciones.

Utilice al máximo las posibilidades que brinda Oracle para asegurar la integridad de la información; de ese modo, su base de datos será mucho más segura y confiable. Además, es una manera de reducir la complejidad de las aplicaciones en su fase de desarrollo.