CENTRE ESPECÍFIC
D´EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

UNIDAD 5

MODELO FÍSICO DML

BASES DE DATOS (BD) 22/23
CFGS DAW

PARTE 2. TRANSACCIONES E ÍNDICES

Revisado por:
Sergio Badal, Abelardo Martínez y Pau Miñana
Autores:
Paco Aldarias
Fecha: 08/12/22

ÍNDICE DE CONTENIDO

UD05.2. MODELO FÍSICO DML. TRANSACCIONES E ÍNDICES

> Importante
>
> Veremos en este tema las particularidades de Oracle, pero recordad que en esta primera parte del curso solo se os pide conocer y usar la sintaxis de MySQL.

1. TRANSACCIONES

Una transacción está formada por una serie de instrucciones DML (INSERT, DELETE, UPDATE).

Una transacción comienza con la primera instrucción DML (INSERT, DELETE, UPDATE) que se ejecute y **finaliza** con alguna de estas circunstancias**:**

- **Una operación COMMIT o ROLLBACK**
- **Una instrucción DDL (como ALTER TABLE por ejemplo)**
- **Una instrucción DCL (como GRANT)**
- **El usuario abandona la sesión**
- **Caída del sistema**

Hay que tener en cuenta que cualquier instrucción DDL o DCL da lugar a un COMMIT implícito (automático). Es decir, todas las instrucciones DML ejecutadas hasta ese instante pasan a ser **definitivas**.

1.1 COMMIT

La instrucción COMMIT hace que los cambios realizados por la transacción sean definitivos, irrevocables. Sólo se debe utilizar si estamos de acuerdo con los cambios, conviene asegurarse mucho antes de realizar el COMMIT ya que las instrucciones ejecutadas pueden afectar a miles de registros.

El cierre correcto de la sesión da lugar a un COMMIT, aunque siempre conviene ejecutar explícitamente esta instrucción a fin de asegurarnos de lo que hacemos.

1.2 ROLLBACK

Esta instrucción regresa a la instrucción anterior al inicio de la transacción, normalmente el último COMMIT, la última instrucción DDL o DCL o al inicio de sesión.

Anula definitivamente los cambios, por lo que conviene también asegurarse de esta operación.

Un abandono de sesión incorrecto o un problema de comunicación o una caída del sistema dan lugar a un ROLLBACK implícito (automático).

1.3 Estado de los datos durante la transacción

Recordemos que una transacción se inicia con un comando DML (INSERT, DELETE, UPDATE).

Cuando esto sucede, hay que tener en cuenta que:

- Se puede volver a la instrucción anterior al inicio de la transacción cuando se desee.
- Las instrucciones de consulta (DQL o SELECT) realizadas por el usuario que inició la transacción muestran los datos ya modificados por las instrucciones DML (INSERT, DELETE, UPDATE).
- El resto de usuarios ven los datos tal cual estaban antes de la transacción, de hecho los registros afectados por la transacción aparecen bloqueados hasta que la transacción finalice. Esos usuarios no podrán modificar los valores de dichos registros.
- Tras la transacción, todos los usuarios ven los datos tal cual quedan tras el fin de transacción. Los bloqueos son liberados y los puntos de ruptura borrados.

1.4 COMMIT y ROLLBACK en MySQL

Existe un modo llamado autocommit que hace un COMMIT automático con cada ejecución de una sentencia DML.

Por defecto, **la conexión con el servidor MySQL comienza con el modo de autocommit habilitado y con Oracle deshabilitado**, el cual automáticamente confirma cada sentencia SQL mientras lo ejecuta. Este modo de operación puede ser desconcertante si se tiene experiencia con otros sistemas de bases de datos, donde es una práctica estándar emitir una secuencia de instrucciones DML y confirmarlas (COMMIT) o deshacerlas (ROLLBACK) todas juntas.

```
mysql> -- By default, in MySQL autocommit is turned ON
mysql> CREATE TABLE ESTUDIANTES (IDEST VARCHAR(2), NOMBRECOMP VARCHAR (20));
Query OK, 0 rows affected (0.00 sec)
mysql> INSERT INTO ESTUDIANTES VALUES (12, 'Heikki');
Query OK, 1 row affected (0.00 sec)
mysql> -- Do another transaction with autocommit turned OFF
mysql> SET autocommit=0;
Query OK, 0 rows affected (0.00 sec)
mysql> INSERT INTO ESTUDIANTES VALUES (15, 'John');
Query OK, 1 row affected (0.00 sec)
mysql> INSERT INTO ESTUDIANTES VALUES (20, 'Paul');
Query OK, 1 row affected (0.00 sec)
mysql> DELETE FROM ESTUDIANTES WHERE NOMBRECOMP = 'Heikki';
Query OK, 1 row affected (0.00 sec)
mysql> -- Now we undo those last 2 inserts and the delete.
mysql> ROLLBACK;
Query OK, 0 rows affected (0.00 sec)
mysql> SELECT * FROM ESTUDIANTES;
+------+--------+
| IDEST| NOMBRECOMP
|
+------+--------+
| 12 | Heikki |
+------+--------+
1 row in set (0.00 sec)
mysql>
```

- **Si queremos trabajar con autocommit desactivado (**modo Oracle) desactivaremos autocommit con la instrucción **SET autocommit = 0** y finalizaremos cada transacción con COMMIT o ROLLBACK según corresponda.
- **Si queremos trabajar con autocommit activado** (modo MySQL), comenzaremos cada transacción con START TRANSACTION y la finalizaremos con COMMIT o ROLLBACK.

El siguiente ejemplo en MySQL por defecto (autocommit desactivado) muestra dos transacciones, la primera finaliza con COMMIT y la segunda con ROLLBACK.

```
mysql> -- By default, in MySQL autocommit is turned ON
mysql> CREATE TABLE ESTUDIANTES (IDEST VARCHAR(2), NOMBRECOMP VARCHAR (20));
Query OK, 0 rows affected (0.00 sec)
mysql> -- To do a transaction with autocommit turned ON
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)
mysql> INSERT INTO ESTUDIANTES VALUES (12, 'Heikki');
Query OK, 1 row affected (0.00 sec)
mysql> COMMIT;
Query OK, 0 rows affected (0.00 sec)
mysql> -- Do another transaction with autocommit turned OFF
mysql> SET autocommit=0;
Query OK, 0 rows affected (0.00 sec)
mysql> INSERT INTO ESTUDIANTES VALUES (15, 'John');
Query OK, 1 row affected (0.00 sec)
mysql> INSERT INTO ESTUDIANTES VALUES (20, 'Paul');
Query OK, 1 row affected (0.00 sec)
mysql> DELETE FROM ESTUDIANTES WHERE NOMBRECOMP = 'Heikki';
Query OK, 1 row affected (0.00 sec)
mysql> -- Now we undo those last 2 inserts and the delete.
mysql> ROLLBACK;
Query OK, 0 rows affected (0.00 sec)
mysql> SELECT * FROM ESTUDIANTES;
+------+--------+
| IDEST| NOMBRECOMP
|
+------+--------+
| 12 | Heikki |
+------+--------+
1 row in set (0.00 sec)
mysql>
```

2. ÍNDICES

Los índices son un objeto más de las bases de datos, como pueden ser las tablas, que hacen que las bases de datos aceleren las operaciones de consulta y ordenación de los datos (DQL o SELECT).

De cada tabla, se puede crear uno o varios índices siendo cada uno una copia de esa tabla de ordenada en función de un campo o criterio concreto.

Por ejemplo, puedo tener una tabla de ESTUDIANTES con los campos DNI, nombre, apellidos y fecha de nacimiento con un índice sobre el campo fecha de nacimiento que me permita consultar rápidamente todos los estudiantes ordenados por ese campo.

Se almacenan aparte de la tabla a la que hace referencia, lo que permite crearlos y borrarlos en cualquier momento.

La mayoría de los índices se crean de manera implícita (automática), como consecuencia de las restricciones PRIMARY KEY, UNIQUE y FOREIGN KEY. Estos son índices obligatorios, por los que los crea el propio SGBD.

Por ejemplo, si creamos la tabla ESTUDIANTES con el DNI como PK, el id de ciudad de residencia y el id de país de residencia como FKs y el numSS (número de la seguridad social) como UK estamos, implícitamente, creando estos objetos en la base de datos:

- La tabla ESTUDIANTES
- Una copia de esa tabla ordenada por la PK (DNI)
- Una copia de esa tabla ordenada por la FK id_ciudad
- Una copia de esa tabla ordenada por la FK id_pais
- Una copia de esa tabla ordenada por la UK numSS

De esta manera, cada vez que hagamos un DML (INSERT, UPDATE, DELETE) sobre esa tabla tendremos que reconstruir todos esos objetos por lo que **los índices son recomendados solo cuando las operaciones DML son infinitamente menores que las de consulta (DQL o SELECT).**

Los índices realizan una lista ordenada por la que el SGBD puede acceder para facilitar la búsqueda de los datos.

Con cada instrucción DML (INSERT, UPDATE, DELETE), los índices involucrados se actualizan a fin de que su información esté al día.

De ahí que cuantos más índices haya, más le costará al SGBD regenerarlos con cada DML (INSERT, UPDATE, DELETE), pero más rápidas se realizan las instrucciones de consulta (DQL o SELECT).

2.1 Creación de índices

Aparte de los índices implícitos (automáticos) comentados anteriormente, se pueden crear índices de forma explícita (manual). Éstos se crean para aquellos campos sobre los cuales se realizarán búsquedas e instrucciones de ordenación frecuente.

Veamos la sintaxis completa:

```
CREATE [UNIQUE|BITMAP] INDEX [esquema.]index_name
ON [esquema.]table_name [tbl_alias]
(col [ASC | DESC])
```

Y la sintaxis sencilla:

```
CREATE INDEX nombre
ON tabla (columna1 [,columna2...])
```

Ejemplo:

```
CREATE INDEX ix_nombre_completo
ON ESTUDIANTES (apellido1, apellido2, nombre);
```

El ejemplo crea un índice para los campos apellido1, apellido2 y nombre. Esto **no es lo mismo que crear un índice para cada campo**, puesto que este índice es efectivo solo cuando se buscan u ordenan clientes usando los tres campos (apellido1, apellido2 y nombre) a la vez.

Si realizamos una consulta sobre esa tabla ordenada por esos campos y en ese orden, la respuesta será de milisegundos aunque la tabla tenga millones de registros.

Se aconseja crear índices en campos que:

- Contengan una gran cantidad de valores.
- Contengan una gran cantidad de nulos.
- Sean parte habitual de cláusulas WHERE, GROUP BY u ORDER BY (que veremos más adelante).
- Sean parte de listados de consultas de grandes tablas sobre las que casi siempre se muestran como mucho un 4% de su contenido.

No se aconseja en campos que:

- Pertenezcan a tablas pequeñas.
- No se usen a menudo en las consultas.
- Pertenezcan a tablas cuyas consultas muestren menos de un 4% del total de registros.
- Pertenezcan a tablas que se actualicen frecuentemente (DML).

2.2 Lista de índices

Para ver la lista de índices en Oracle se utiliza la vista USER_INDEXES. Mientras que la vista USER_IND_COLUMNS muestra la lista de columnas que son utilizadas por índices.

En MySQL utilizamos el comando SHOW INDEX que nos devolverá la información de la tabla de índices. Su sintaxis es la siguiente:

```
SHOW {INDEX | INDEXES | KEYS}

{FROM | IN} tbl_name

[{FROM | IN} db_name]

[WHERE expr]
```

2.3 Borrar índices

La instrucción DROP INDEX seguida del nombre del índice permite eliminar el índice en cuestión.