

Y

T5: Procesamiento de transacciones

Introducción

Transacción: conjunto de instrucciones SQL que se ejecutan como una unidad de manera atómica
 ↓
 (Éxito): todas las operaciones se guardan def. en la BBDD

Propiedades ACID:

- Atomicity → Transacciones (todas las operaciones o ninguna)
- Consistency → Integridad (solo permite 2º datos correctos)
- Isolation → Aislamiento (bloqueo) → Prever la coherencia
- Durability → durabilidad (Éxito + Cambios definitivos)

Transacciones MySQL

- solo InnoDB
- una transacción Puede terminar
 - Éxito (COMMIT): ejecutan operaciones y se hacen permanentes
 - Fallo (ROLLBACK): no se ejec. ninguna op

- PASOS Generales a realizar en una transacción:
 - 1) Iniciar transacción START TRANSACTION o BEGIN
 - 2) OP. update, Insert, delete autocommit=0
 - 3) ¿Quieres conservar cambios?



- transacciones automáticas → activar set autocommit 1
- desactivar set autocommit 0

- Comenzar una transacción provoca:
 - cualquier transacción pendiente sea confirmada/elec.
 - lock tables se eliminan (como si hubiesen hecho UNLOCK TABLES)

- Las transacciones no confirmadas (committed)
no registradas, Ficheros binarios de log

TRANSACCIONES EN JAVA

- Deshabilitar Autocommit:

```
conexion.setAutoCommit(false);
```

- Hacer Commit - ROLLBACK

```
conexion.commit();
conexion.rollback();
```

- Establecer, usar y borrar Savepoints

```
conexion.setSavepoint([name]);
conexion.rollback(Savepoint);
conexion.releaseSavepoint(Savepoint)
```

```

mysql> CREATE TABLE ventas
    > id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    > producto VARCHAR(20) NOT NULL,
    > cantidad TINYINT NOT NULL;
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO ventas(id, "producto", cantidad)
    > VALUES(1, "Pepeito Ternera", 3);
Query OK, 1 row affected (0.05 sec)
Rows matched: 1 Changed: 1 Warnings: 0

mysql> SELECT * FROM ventas;
+----+-----+-----+
| id | producto | cantidad |
+----+-----+-----+
| 1  | Pepeito Ternera | 3          |
+----+-----+-----+
1 row in set (0.00 sec)

mysql> ROLLBACK;
Query OK, 0 rows affected (0.10 sec)

mysql> SELECT * FROM ventas;
+----+-----+-----+
| id | producto | cantidad |
+----+-----+-----+
| 1  | Pepeito Ternera | 3          |
+----+-----+-----+
1 row in set (0.00 sec)

```

- Hay sentencias q no se pueden deshacer (rolled back)

↳ DDL (Data Definition Language)
 CREATE, DROP, ALTER

Transacciones deben diseñarse con cuidado

↳ Mejor solo DML (Data Management Language)
 SELECT, INSERT, UPDATE, DELETE

- Hay sentencias provocan commit implícito (automático)

↳ Confirman cualquier transacción pendiente
 ↳ Sentencias DDL: RENAME, TRUNCATE, DROP, ...
 ↳ sentencias q automáticamente usan e mod. tablas de mysql (BBDD)
 ALTER, CREATE, RENAME, DROP USER, GRANT, REVOKE, SET PASSWORD

↳ sentencias de bloqueo de tablas LOCK/UNLOCK TABLES
 ↳ sentencias control de transacciones BEGIN, SET AUTOCOMMIT, ...
 ↳ sentencias de larga de datos LOAD DATA INTO

- muchos al ejec → commit implícito (rara vez operaciones no pueden deshacer)

- Las transacciones NO se pueden anidar

- Motor InnoDB soporta sentencias SQL

```

SAVEPOINT identifier
ROLLBACK [WORK] TO [SAVEPOINT] identifier
RELEASE SAVEPOINT identifier

```

SAVEPOINT: Establece etiqueta mediante un id

ROLLBACK TO SAVEPOINT: ↳ como volver al commit del id

RELEASE SAVEPOINT: elimina la etiqueta indicada de la transacción actual

después de un commit o ROLLBACK → todos los savepoints son eliminados

```

mysql> BEGIN;
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO test VALUES(4);
Query OK, 1 row affected (0.04 sec)

mysql> SELECT * FROM test;
+----+
| 1  |
| 2  |
| 3  |
| 4  |
+----+
4 rows in set (0.00 sec)

mysql> COMMIT;
Query OK, 0 rows affected (0.07 sec)

mysql> SELECT * FROM test;
+----+
| 1  |
| 2  |
| 3  |
| 4  |
+----+
4 rows in set (0.00 sec)

mysql> exit;

```