A5. Transacciones.

Índice.

A5. Transacciones.

1. ¿Qué es una Transacción?

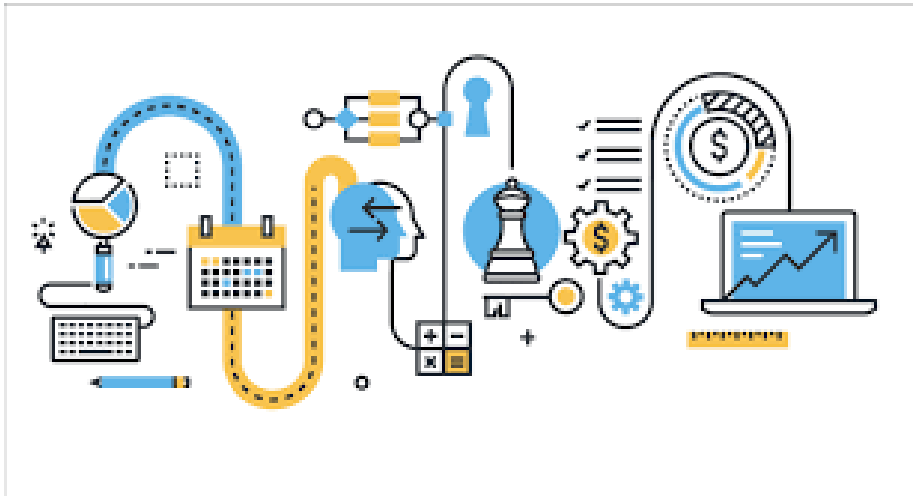
A5. Transacciones.

1. ¿Qué es una Transacción?

Las transacciones en SQL son unidades (o secuencias de trabajo) realizadas de forma ordenada y separada en una Base de Datos. Normalmente, representan cualquier cambio en la Base de Datos.

Las transacción en SQL tienen dos objetivos principales:

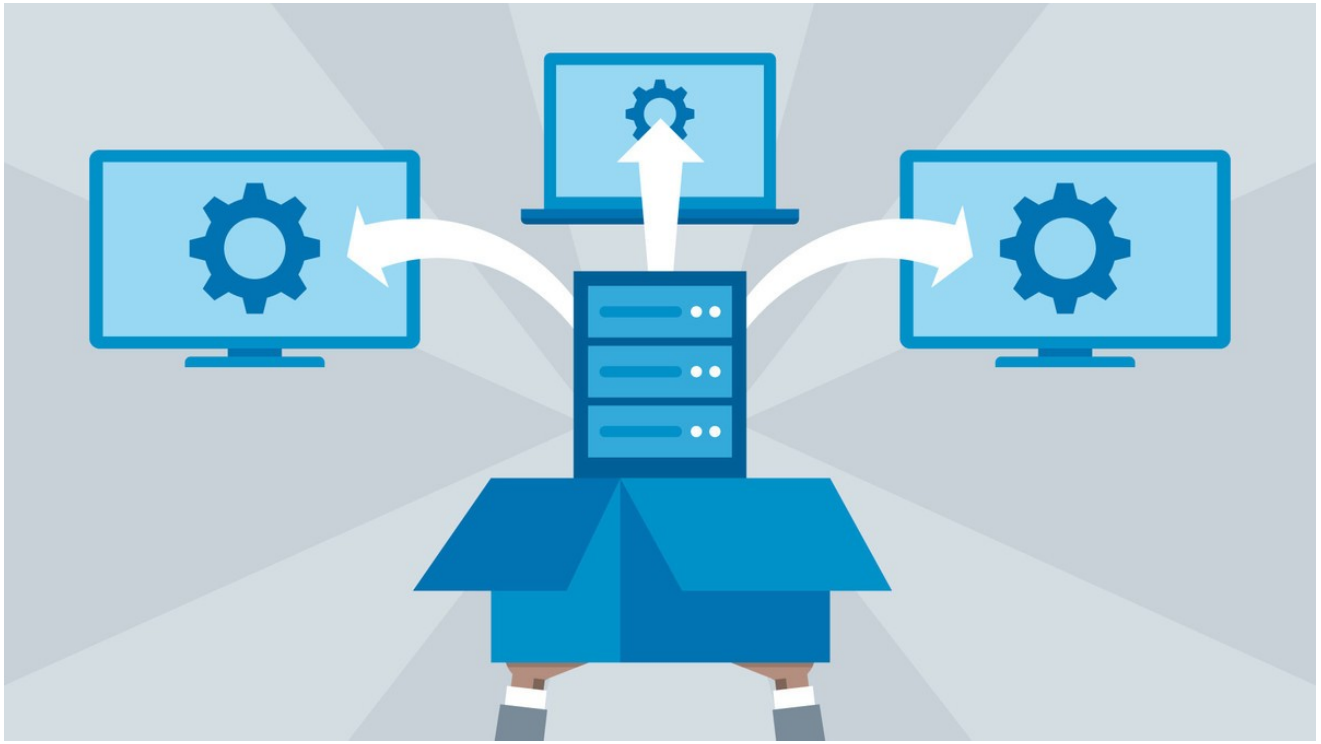
- Proporcionar secuencias de trabajo fiables que permitan poder recuperarse fácilmente ante errores y mantener la Base de Datos consistente, incluso frente a fallos del sistema.
- Proporcionar aislamiento entre programas, accediendo a la vez a la Base de Datos.



A5. Transacciones.

1. ¿Qué es una Transacción?

Una **transacción** es una propagación de uno o más cambios en la Base de Datos, bien al crear, modificar o eliminar registros. En la práctica, suele consistir en la agrupación de consultas SQL y en su ejecución como parte de una transacción.



A5. Transacciones.

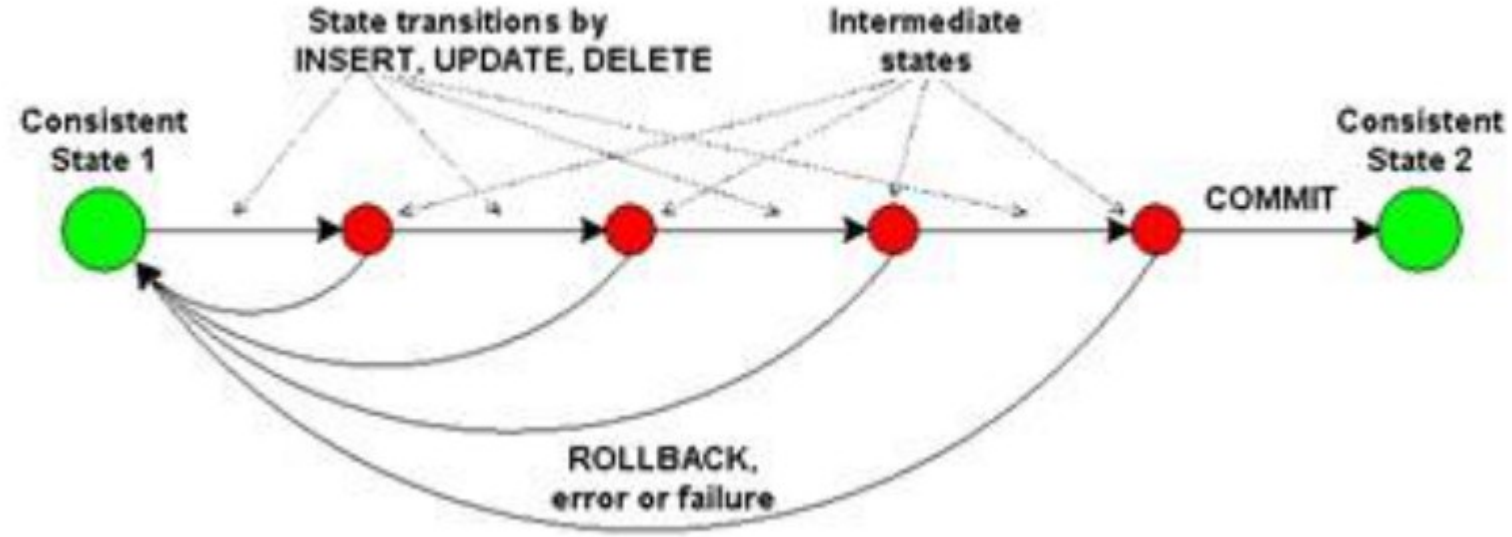
1. ¿Qué es una Transacción?

Una transacción SQL es un conjunto de sentencias de SQL que se ejecutan formando una unidad lógica de trabajo, es decir, en forma indivisible o atómica.

MySQL permite realizar transacciones entre tablas SÓLO si se emplea el motor de almacenamiento InnoDB.

La utilización de transacciones permite la realización de tareas de forma segura y la recuperación de datos si se produjese algún fallo en el Servidor durante la transacción.

Un gran inconveniente es el aumento de tiempo necesario para la ejecución de las instrucciones.



2. Propiedades ACID de las Transacciones.

ACID es el acrónimo de las propiedades que debe cumplir una Base de Datos:

- **Atomicity (atomicidad)** → asegura que TODAS las operaciones de una secuencia de trabajo se completan satisfactoriamente, SINO se abandona en el punto de error y la Base de Datos vuelve a su estado inicial.
- **Consistency (consistencia)**→ asegura que la Base de Datos estará SIEMPRE en un estado válido y consistente tras cada transacción satisfactoria.
- **Isolation (aislamiento)** → asegura que las operaciones son independientes (aisladas) y transparentes unas de otras.
- **Durability (durabilidad)** → asegura que el resultado de una transacción completada permanece en caso de error del sistema.

Las propiedades ACID garantizan la realización de Transacciones en una Base de Datos de forma SEGURA.

Un Sistema Gestor de Bases de Datos es **ACID Compliant** si permite la realización de transacciones.

3. Control de las Transacciones.

El control de las Transacciones se realiza a través de tres comandos básicos:

- **COMMIT** → guarda los cambios realizados en la Base de Datos.
- **ROLLBACK** → abandona la transacción y deshace los cambios efectuados, es una marcha atrás.
- **SAVEPOINT** → crea un punto intermedio de restauración de los cambios efectuados en la Base de Datos.

```
1  start transaction; -- Nueva transacción
2      insert into audita values('Hola');
3      insert into audita values('Saludos');
4      savepoint x; -- Punto de salvaguarda
5      update audita set mensaje = 'Saluditos' where mensaje like 'Saludos';
6      delete from audita where mensaje like 'Saluditos';
7      rollback to x; -- Se deshacen los cambios hasta el último savepoint
8      insert into audita values('Yipicayei');
9  commit; -- Grabo las operaciones y cierro la actual transacción
```

4. Pasos para realizar una Transacción.

Los pasos para realizar una transacción en SQL son los siguientes:

- Indicar el inicio de una Transacción con las sentencias → **start transaction**, **begin**, o, **begin work**.
- Realizar las operaciones de manipulación de datos sobre la base de datos.
- Si las operaciones se han realizado correctamente y queremos que los cambios se apliquen de forma permanente sobre la Base de Datos → **commit**.
- Si durante las operaciones ha ocurrido algún error y no queremos aplicar los cambios realizados, podemos deshacerlos con → **rollback**.
- Si trabajamos con tablas del tipo InnoDB en MySQL, también podemos hacer uso de estas otras sentencias:
  - **Savepoint** → establece un punto de recuperación dentro de la transacción. Ha de tener un identificador.
  - **Rollback to savepoint** → hace un rollback sólo de las instrucciones ejecutadas desde el **savepoint** especificado.
  - **Release savepoint** → elimina un **savepoint**.

5. Instrucciones para manejar Transacciones.

Las instrucciones de manejo de transacciones se pueden clasificar en:

- **Definición de variables.**
  - SET AUTOCOMMIT = [0 | 1]; → COMMIT explícito (0) y necesario en ejecución de TODAS las transacciones, o implícito (1)
  - SET TRANSACTION ISOLATION LEVEL [READ UNCOMMITTED | READ COMMITED | REPEATEABLE READ | SERIALIZABLE] → modo de aislamiento en el que trabaja la transacción
  - SET TRANSACTION [READ WRITE | READ ONLY] → modo de acceso de la transacción
  - …
- **Inicio y fin.**
  - START TRANSACTION | BEGIN [WORK]; → inicio de la transacción
  - COMMIT [WORK]; → confirmación de los cambios realizados en la transacción
  - ROLLBACK [WORK]; → anulación de los cambios realizados en la transacción
- **Puntos de restauración.**
  - SAVEPOINT <identificador>; → creación de un punto de restauración
  - RELEASE SAVEPOINT <identificador>; → confirmación de los cambios realizados
  - ROLLBACK [WORK] TO SAVEPOINT <identificador>; → anulación de los cambios realizados
- **Protección.**
  - LOCK TABLES <tabla> {READ | WRITE } [, <tabla> {read|write}]; → bloqueo de tablas
  - UNLOCK TABLES; → desbloqueo de tablas