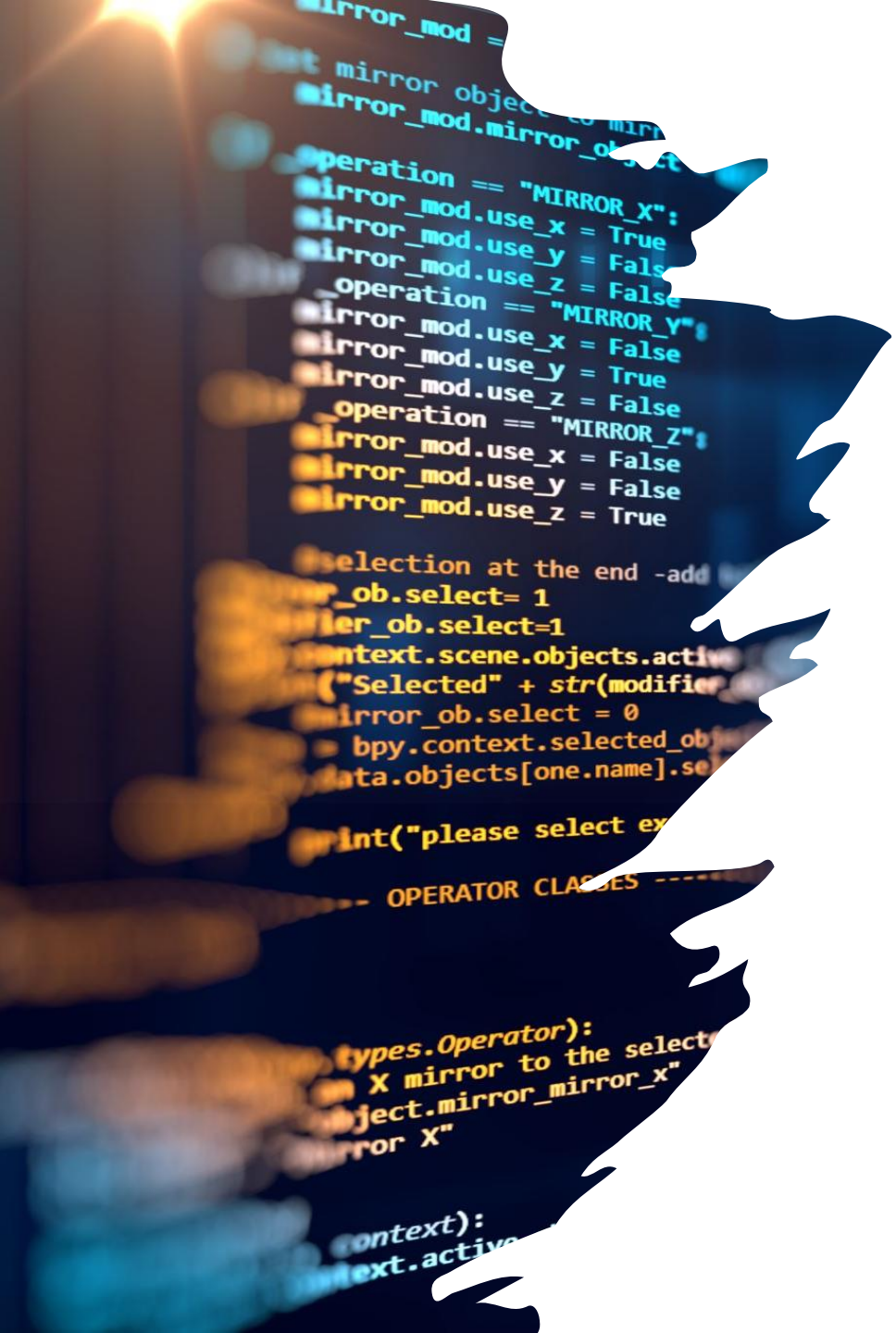
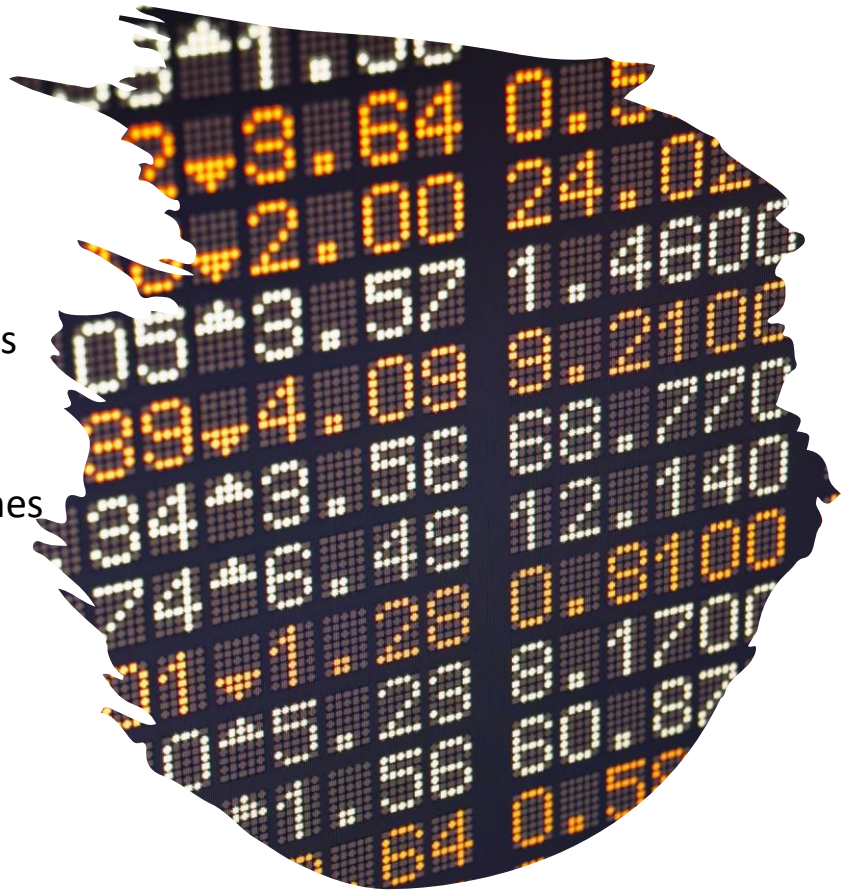


Bases de Datos

- TRANSACCIONES

TRANSACCIONES

Una transacción es un conjunto de operaciones o instrucciones SQL, generalmente de actualización de datos, que forman un proceso conjunto.



- El conjunto de instrucciones de una transacción no queda realizado a medias, o se realizan todas las operaciones o no se realiza ninguna de las operaciones de la transacción.

- Durante la ejecución de las instrucciones que forman una transacción, podemos anular todas las instrucciones o confirmarlas.

- Por defecto, MySQL se comporta de forma que toda instrucción es una transacción que se confirma automáticamente en el momento de ejecutarla. Por tanto, no se puede anular su ejecución una vez realizada. SE DICE QUE POR DEFECTO TRABAJA EN ESTADO NO TRANSACCIONAL.

- En MySQL se pueden usar transacciones con tablas InnoDB (las tablas que se crean por defecto). En algunos otros tipos de tablas no se pueden usar como, por ejemplo, en tablas MyISAM.

TRANSACCIONES

Las cuatro propiedades de las transacciones (ACID)

- *Atomicidad: Significa que es una unidad indivisible. Es la propiedad que asegura que la operación se ha realizado o no, y por lo tanto ante un fallo del sistema no puede quedar a medias.*
- *Consistencia: Indica que después de ejecutarse una transacción, la BD debe quedar en estado correcto.*
- *Isolation (Aislamiento): Indica que el comportamiento de una transacción no se ve afectada por el hecho de que otras transacciones sean ejecutadas al mismo tiempo.*
- *Durabilidad: Cuando se completa una transacción con éxito los cambios se vuelven permanentes.*

TRANSACCIONES

- Ejemplo de transacción en base de datos alquileres
- Al finalizar un contrato, se deben realizar varias operaciones de actualización:
  - Establecer la fecha final del contrato
  - Establecer los kilómetros finales del contrato
  - Establecer el importe del contrato
  - Marcar el automóvil como no alquilado o disponibles
  - Establecer en kilómetros del automóvil los kilómetros que tenía el automóvil al finalizar el contrato.

- Todas las instrucciones que realizan estas operaciones deben quedar realizadas o bien no quedar realizada ninguna.
- Si se realizan algunas de ellas y otras no, la base de datos quedaría en un estado incongruente.
- Por ejemplo, si no se realiza la última operación, ocurrirá que un automóvil tendrá menos kilómetros que los que tiene registrados en su último contrato.

TRANSACCIONES

- **Ejemplo de transacción en base de datos alquileres:** Escribir las instrucciones que forman la transacción para hacer todas las operaciones correspondientes a que el contrato número 21 finaliza hoy con 73256 kilómetros del automóvil al finalizar el contrato.

TRANSACCIONES

- **Ejemplo de transacción en base de datos alquileres:** Escribir las instrucciones que forman la transacción para hacer todas las operaciones correspondientes a que el contrato número 21 finaliza hoy con 73256 kilómetros del automóvil al finalizar el contrato.

```
UPDATE contratos
SET ffin=curdate(),kfin=73256
WHERE numcontrato=21;

UPDATE contratos INNER JOIN automoviles
ON contratos.matricula=automoviles.matricula
SET importe=precio*datediff(ffin,fini)
WHERE numcontrato=21;

UPDATE contratos INNER JOIN automoviles
ON contratos.matricula=automoviles.matricula
SET alquilado=false,kilometros=73256
WHERE numcontrato=21;
```

TRANSACCIONES

- **Para ejecutarlo como transacción habría que añadir**

```
START TRANSACTION;

UPDATE contratos
SET ffin=curdate(),kfin=73256
WHERE numcontrato=21;

UPDATE contratos INNER JOIN automoviles
ON contratos.matricula=automoviles.matricula
SET importe=precio*datediff(ffin,fini)
WHERE numcontrato=21;

UPDATE contratos INNER JOIN automoviles
ON contratos.matricula=automoviles.matricula
SET alquilado=false,kilometros=73256
WHERE numcontrato=21;
```

Y si todo ha ido bien, ejecutaríamos al final la instrucción para que se confirme la transacción:

```
COMMIT;
```

TRANSACCIONES

En MySQL podemos usar dos estados de gestión de transacciones.

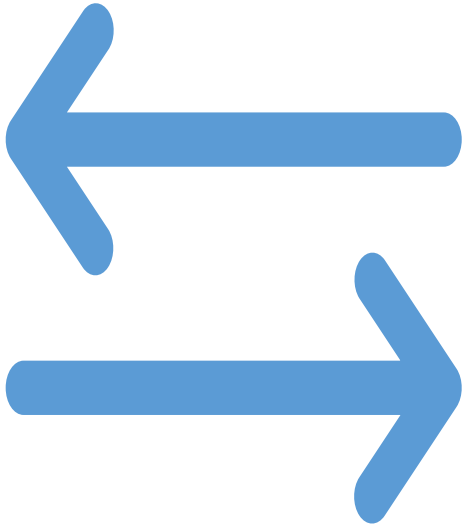
En su configuración por defecto, tiene establecido el **estado no transaccional.**

- Si ejecutamos una instrucción de actualización de datos, ésta queda realmente realizada, no hay vuelta atrás.
- Podemos realizar transacciones con varias instrucciones iniciando una transacción con START TRANSACTION.
- Podemos confirmar todos lo realizado en la transacción con COMMIT o anularlo con ROLLBACK.

El otro es el **estado transaccional**. En este estado:

- No hay que indicar que se inicia una transacción.
- Una transacción comienza cuando otra finaliza.
- Una transacción finaliza cuando se confirma su realización o cuando se anula su realización.

TRANSACCIONES



- Cambio de estado de gestión de transacciones.
- Cada sesión cliente MySQL trabaja en un estado (transaccional o no transaccional).
- Puedes cambiar el estado para tu sesión mediante la instrucción SET AUTOCOMMIT.
  - SET AUTOCOMMIT=0; /*Establece el estado transaccional*/
  - SET AUTOCOMMIT=1; /*Establece el estado NO transaccional*/
- Por defecto, toda sesión se inicia en estado no transaccional (toda instrucción es una transacción que se autoconfirma al ejecutarla).
- También podemos leer el estado de esta variable del sistema con la instrucción:
  - SHOW VARIABLES WHERE Variable_name='autocommit';

TRANSACCIONES

Ejemplo de ejecución de instrucciones en estado transaccional:

1.- INSTRUCCIÓN 1

2.- INSTRUCCIÓN 2

3.- INSTRUCCIÓN 3

4.- COMMIT; (quedan hechas realmente las instrucciones 1, 2 y 3)

5.- INSTRUCCIÓN 4

6.- INSTRUCCIÓN 5 (se ha producido algún problema por ser la instrucción incorrecta, por haber sido rechazada su ejecución, etc. y queremos anular la realizado)

7.- ROLLBACK; (se anulan las instrucciones 4 y 5, se vuelve al estado en el que estaba la base de datos en el punto 4)

8.- INSTRUCCIÓN 6

9.- ALTER TABLE …..; (produce un COMMIT por lo que queda hecha realmente la instrucción 6

10.- INSTRUCCIÓN 7

11.- INSTRUCCIÓN 8

12.- Terminamos la sesión cliente (No se ha confirmado la transacción y queda anulado lo realizado en las instrucciones 7 y 8).

TRANSACCIONES

Ejemplo de ejecución de instrucciones en estado NO transaccional:

1.- INSTRUCCIÓN 1 (queda realmente hecha la instrucción 1)

2.- START TRANSACTION (se inicia una transacción)

3.- INSTRUCCIÓN 2

4.- INSTRUCCIÓN 3

5.- COMMIT; (quedan hechas realmente las instrucciones 2 y 3)

6.- INSTRUCCIÓN 4 (queda realmente hecha la instrucción 4)

7.- INSTRUCCIÓN 5 (queda realmente hecha la instrucción 5)

8.- START TRANSACTION; (se inicia una transacción)

9.- INSTRUCCIÓN 6

10.- INSTRUCCIÓN 7 (ha habido algún problema)

11.- ROLLBACK; (quedan anuladas las instrucciones 6 y 7)

12.- INSTRUCCIÓN 8 (queda realmente hecha la instrucción 8)

TRANSACCIONES

En MySQL InnoDB las instrucciones de gestión de transacciones son:

START TRANSACTION o **BEGIN**: marca el inicio de una transacción en estado no transaccional.
ROLLBACK: Cierra la transacción en curso. Anula las instrucciones realizadas en ella.
COMMIT: Confirma el conjunto de operaciones ejecutadas tras el comienzo de la transacción.

SAVEPOINT etiqueta: Donde se ejecute, marca un punto de retorno o punto para anular instrucciones ejecutadas desde ahí en adelante. En etiqueta podemos poner el nombre que queramos. Dentro de una transacción podemos establecer varios puntos de retorno.
ROLLBACK TO SAVEPOINT etiqueta: Hace que se anulen las instrucciones ejecutadas desde el punto donde se ejecutó SAVEPOINT etiqueta. No confirma las instrucciones ejecutadas desde el comienzo de la transacción hasta el punto SAVEPOINT etiqueta.

SET AUTOCOMMIT=valor: Permite cambiar el estado transaccional de la sesión

TRANSACCIONES

- Hay muchas instrucciones que producen un **COMMIT IMPLÍCITO.**
- Es decir, que tras su ejecución es como si hubieses ejecutado también un commit.
- Algunas de ellas son:

- Las que **definen o modifican** los objetos de la base de datos:
- ALTER EVENT, ALTER FUNCTION, ALTER PROCEDURE, ALTER SERVER, ALTER TABLE, ALTER VIEW, CREATE DATABASE, CREATE EVENT, CREATE FUNCTION, CREATE INDEX, CREATE PROCEDURE, CREATE ROLE, CREATE SERVER, CREATE SPATIAL REFERENCE SYSTEM, CREATE TABLE, CREATE TRIGGER, CREATE VIEW, DROP DATABASE, DROP EVENT, DROP FUNCTION, DROP INDEX, DROP PROCEDURE, DROP ROLE, DROP SERVER, DROP SPATIAL REFERENCE SYSTEM, DROP TABLE, DROP TRIGGER, DROP VIEW...
- Aquellas que **modifican la base de datos** mysql:
- ALTER USER, CREATE USER, DROP USER, GRANT, RENAME USER, REVOKE, SET PASSWORD
- ...