**ACTIVIDAD DE CONCURRENCIA DE DATOS.**

1.- Abrir dos sesiones de MySQL.

2.- Abrir una sesión y verificar el tipo de aislamiento:

select @@tx\_isolation;

3.-Cambiar el tipo de aislamiento a read uncomitted.

Set session transaction isolation level read uncommitted;

4.- En una base de datos de prueba, crear una tabla “coches”

create table coches (nombre varchar(20));

insert into coches values (“toyota”),(“audi”),(“seat”);

**La lectura no repetible**:

Sesión 1:

Arranca una transacción:

start transaction;

Consulta la tabla coches:

mysql> select \* from coches;

+--------+

| nombre |

+--------+

| toyota |

| audi |

| seat |

La sesión 1 inició una transacción para hacer operaciones y segundos después una segunda sesión realiza cambios en la tabla mientras la transacción continúa en la sesión 1.

Sesión 2:

Modifica la tabla coches:

mysql> update coches set nombre="volvo" where nombre="audi";

Query OK, 1 row affected (0.02 sec)

Rows matched: 1 Changed: 1 Warnings: 0

La sesión 2 ha realizado un cambio en los datos, por tanto, la transacción de la primera sesión no será capaz de reproducir la situación anterior puesto que se ha producido una lectura no repetible.

Sesión 1:

mysql> select \* from coches;

+--------+

| nombre |

+--------+

| toyota |

| volvo |

| seat |

+--------+

3 rows in set (0.00 sec)

LA LECTURA FANTASMA:

Si a continuación, en la sesión 2 se inserta un nuevo registro en la tabla, la sesión 1 podrá inmediatamente ver los cambios. Esto es lo que se llama una lectura fantasma.

Sesión 2:

mysql> insert into coches values ("renault");

Query OK, 1 row affected (0.06 sec)

Sesión 1:

mysql> select \* from coches;

+---------+

| nombre |

+---------+

| toyota |

| volvo |

| seat |

| renault |

+---------+

4 rows in set (0.00 sec)

La potencial situación problemática tanto de la lectura no repetible como de la lectura fantasma es que datos que participan en una transacción cambian o aparecen sin previo aviso, pudiendo provocar que la transacción genere información que no es del todo íntegra.

LA LECTURA SUCIA.

El último problem, y más grave, es el de la lectura sucia. A continuación, siguiendo con las dos sesiones de los ejemplos anteriores, se reproduce esta situación. En este caso, la sesión 2 inicia una transacción y realiza operaciones DML sobre los datos.

Sesión 2:

mysql> start transaction;

Query OK, 0 rows affected (0.00 sec)

mysql> insert into coches values ("alfa");

Query OK, 1 row affected (0.00 sec)

mysql> update coches set nombre="honda" where nombre="toyota";

Query OK, 1 row affected (0.00 sec)

Rows matched: 1 Changed: 1 Warnings: 0

Después la sesión 1 lee los datos que hay en la tabla:

Sesión 1:

mysql> select \* from coches;

+---------+

| nombre |

+---------+

| honda |

| volvo |

| seat |

| renault |

| alfa |

+---------+

5 rows in set (0.00 sec)

Se puede observar, que gracias a esa lectura la sesión 1 puede generar nueva información o realizar algún trabajo que implique trabajar con la información modificada por la sesión 2. Pero, supóngase que ahora, la sesión 2 realiza un rollback de sus operaciones:

Sesión 2:

mysql> rollback;

Query OK, 0 rows affected (0.03 sec)

Se puede observar que la sesión 1, aunque ya no puede acceder a la información que fue insertada por la sesión 2, puesto que se ha efectuado un rollback, puee haber estado trabajando con datos, que en realidad, nunca han estado en la base de datos. Esto es una lectura sucia:

Sesión 1:

mysql> select \* from coches;

+---------+

| nombre |

+---------+

| toyota |

| volvo |

| seat |

| renault |

+---------+

4 rows in set (0.00 sec)