

# UNIVERSIDAD DE GUADALAJARA

CENTRO UNIVERSITARIO DE CIENCIAS EXACTAS E INGENIERÍA

DEPARTAMENTO DE CIENCIAS COMPUTACIONALES

Administración de Bases de Datos

## Concurrencia y bloqueos en Oracle



**Alumno:** Citlalli Ruiz Puga

**Código:** 216241881

**Sección:** D01

**Profesor:** Sergio Javier Uribe Nava

**Carrera:** Ingeniería Informática

**Fecha:** 24/10/2023

## A) INVESTIGACION

Introducción a la simultaneidad y coherencia de datos en un entorno multiusuario

En una base de datos de un solo usuario, el usuario puede modificar datos en la base de datos sin preocuparse de que otros usuarios modifiquen los mismos datos al mismo tiempo. Sin embargo, en una base de datos multiusuario, las declaraciones dentro de múltiples transacciones simultáneas pueden actualizar los mismos datos. Las transacciones que se ejecutan al mismo tiempo deben producir resultados significativos y consistentes. Por lo tanto, el control de la concurrencia y la coherencia de los datos es vital en una base de datos multiusuario.

- La simultaneidad de datos significa que muchos usuarios pueden acceder a los datos al mismo tiempo.
- La coherencia de los datos significa que cada usuario ve una vista coherente de los datos, incluidos los cambios visibles realizados por las transacciones del propio usuario y las transacciones de otros usuarios.

Para describir el comportamiento consistente de las transacciones cuando las transacciones se ejecutan al mismo tiempo, los investigadores de bases de datos han definido un modelo de aislamiento de transacciones llamado serializabilidad. El modo serializable de comportamiento de transacciones intenta garantizar que las transacciones se ejecuten de tal manera que parecieran ejecutarse una a la vez, o en serie, en lugar de simultáneamente.

Si bien este grado de aislamiento entre transacciones es generalmente deseable, ejecutar muchas aplicaciones en este modo puede comprometer seriamente el rendimiento de la aplicación. El aislamiento completo de transacciones que se ejecutan simultáneamente podría significar que una transacción no puede realizar una inserción en una tabla que está siendo consultada por otra transacción. En resumen, las consideraciones del mundo real suelen requerir un compromiso entre el perfecto aislamiento de las transacciones y el rendimiento.

Fenómenos prevenibles y niveles de aislamiento de transacciones.  
Los tres fenómenos prevenibles son:

- **Lecturas sucias:** Una transacción lee datos escritos por otra transacción que aún no se ha confirmado.
- **Lectura no repetibles (difusas):** Una transacción vuelve a leer los datos que leyó anteriormente y descubre que otra transacción confirmada ha modificado o eliminado los datos.
- **Lecturas fantasmas (o fantasmas):** Una transacción vuelve a ejecutar una consulta que devuelve un conjunto de filas que satisface una condición de búsqueda y descubre que otra transacción confirmada ha insertado filas adicionales que satisfacen la condición.

Nivel de aislamiento	Lectura sucia	Lectura no repetible	Lectura fantasma
Leer no comprometido	Posible	Posible	Posible
Leer comprometido	Imposible	Posible	Posible
Lectura repetible	Imposible	Imposible	Posible
Serializable	Imposible	Imposible	Imposible

### Descripción general de los mecanismos de bloqueo

En general, la base de datos multiusuario utilizan alguna forma de bloqueo de datos para resolver los problemas asociados con la simultaneidad, coherencia e integridad de los datos para resolver los problemas. Los bloqueos son mecanismos que evitan la interacción destructiva entre transacciones que acceden al mismo tiempo recurso.

Los recursos incluyen dos tipos generales de objetos:

- Objetos de usuario, como tablas y filas (estructuras y datos)
- Objetos del sistema no visibles para los usuarios, como estructuras de datos compartidos en la memoria y filas de diccionario de datos.

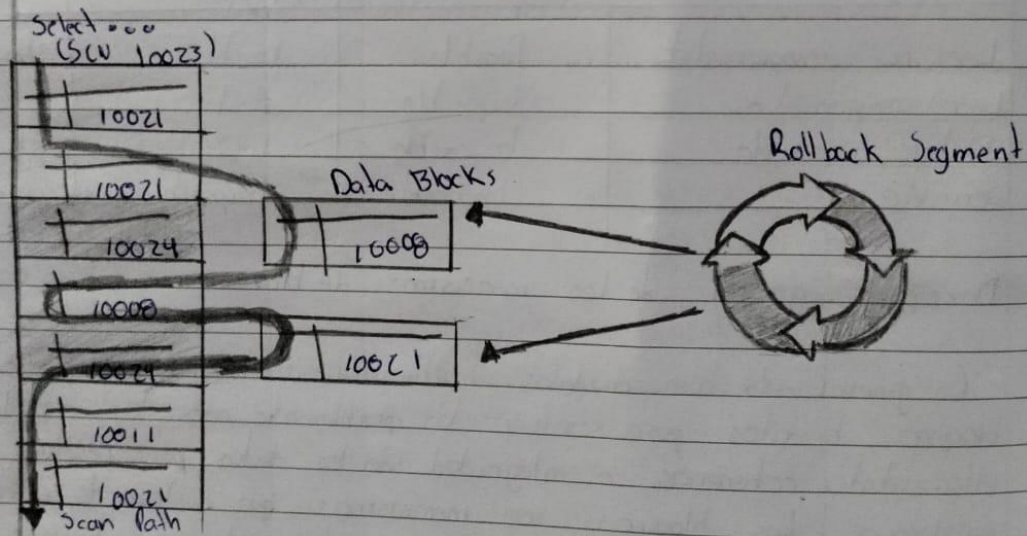


## Como gestiona Oracle la simultaneidad y la coherencia de datos

### Control de concurrencia multiversión

Oracle proporciona automáticamente coherencia de lectura a una consulta para que todos los datos que ve la consulta provengan de un único punto en tiempo (consistencia de lectura a nivel de declaración). Oracle también puede proporcionar coherencia de lectura a todas las consultas de una transacción (consistencia de lectura a nivel de transacción).

Oracle utiliza la información mantenida en sus segmentos de reversión para proporcionar estas vistas consistentes. Los segmentos de reversión contienen los valores antiguos de datos que han sido modificados por transacciones no confirmadas o confirmadas recientemente.



Cuando una consulta ingresa a la etapa de ejecución, se determina el número de cambio del sistema (SCN) actual. Como los bloques de datos se leen en nombre de la consulta, solo se utilizan los bloques escritos con el SCN observado. Los bloques con datos modificados (SCN más reciente) se reconstruyen a partir de datos en los segmentos de reversión y los datos reconstruidos se devuelven para la consulta. Por lo tanto, cada consulta devuelve todos los datos confirmados con respecto al SCN registrado.

### Coherencia de lectura a nivel de declaración

Oracle siempre hace cumplir coherencia de lectura a nivel de declaración. Esta organizada para garantizar que todos los datos devueltos por una única consulta provienen de un único momento: el momento en que comenzó la consulta. Por lo tanto, una consulta nunca ve datos sucios ni ninguno de los cambios realizados por las transacciones que se confirman durante la ejecución de la consulta. A medida que avanza la ejecución de la consulta, solo los datos confirmados antes de que comenzara la consulta son visibles por la consulta. La consulta no ve cambios confirmados después de que comienza la ejecución de la declaración.

Se proporciona un conjunto de resultados coherente para cada consulta, lo que garantiza la coherencia de los datos, sin que el usuario tenga que realizar ninguna acción. Las declaraciones SQL `SELECT`, `INSERT` con una subconsulta, `UPDATE` y `DELETE` todos los datos de la consulta, ya sea explícita o implícitamente, devuelven datos consistentes. Cada una de estas declaraciones utiliza una consulta para determinar a qué datos afectará (`SELECT`, `INSERT`, `UPDATE` o `DELETE`, respectivamente).

Una `SELECT` declaración es una consulta explícita y puede tener consultas anidadas o una operación de unión. Una `INSERT` declaración puede utilizar consultas anidadas. `UPDATE` y `DELETE` las declaraciones pueden usar `WHERE` cláusulas o subconsultas para afectar solo a algunas filas de una tabla en lugar de a todas las filas.

Las consultas utilizadas en las declaraciones `INSERT`, `UPDATE` y `DELETE` tiene garantizado un conjunto consistente de resultados. Sin embargo, no ven los cambios realizados por la propia declaración DML. En otras palabras, la consulta en estas operaciones ve los datos tal como existían antes de la operación comenzara a realizar cambios.



## Coherencia de lectura a nivel de transacción

Oracle también ofrece la opción de imponer coherencia de lectura a nivel de transacción. Cuando una transacción se ejecuta en modo serializable, todos los accesos a los datos reflejan el estado de la base de datos en un momento en que comenzó la transacción. Esto significa que los datos vistos por todas las consultas dentro de la misma transacción son consistentes con respecto a un único momento, excepto que los cambios realizados por la propia transacción. La coherencia de lectura a nivel de transacción produce lecturas repetibles y no expone una consulta a fantasmas.

## Niveles de aislamiento de Oracle

Nivel de aislamiento	Descripción
Leer comprometido	Este es un nivel de aislamiento de transacciones predeterminado. Cada consulta ejecutada por una transacción ve solo los datos que se confirmaron antes de que comenzara la consulta (no la transacción). Una consulta de Oracle nunca lee datos sucios (no confirmados). Debido a que Oracle no impide que otras transacciones modifiquen los datos leídos por una consulta, esos datos pueden ser cambiados por otras transacciones entre dos ejecuciones de la consulta.
Serializable	Las transacciones serializables ven solo los cambios que se confirmaron en el momento en que comenzó la transacción, más los cambios realizados por la propia transacción a través de declaraciones INSERT, UPDATE y DELETE. Las transacciones serializables no experimentan lecturas no repetibles ni fantasmas.
Solo lectura	Las transacciones de solo lectura ven solo los cambios que se confirmaron en el momento en que comenzó la transacción y no permiten declaraciones INSERT, UPDATE y DELETE.

### Leer aislamiento comprometido

El nivel de aislamiento predeterminado para Oracle es lectura comprometida. Este grado de aislamiento es apropiado para entornos donde es probable que pocas transacciones entren en conflicto. Oracle hace que cada consulta se ejecute con respecto a su propio tiempo de visualización materializada, lo que permite lecturas no repetibles y fantasmas para múltiples ejecuciones de una consulta, pero proporciona mayor rendimiento potencial. El aislamiento de lectura comprometida es el nivel adecuado de aislamiento para entornos donde es probable que pocas transacciones entren en conflicto.

### Aislamiento serializable

El aislamiento serializable es adecuado para entornos:

- Con grandes bases de datos y transacciones cortas que actualizan solo unas pocas filas.
- Donde la posibilidad de que dos transacciones simultáneas modifiquen las mismas filas es relativamente baja.
- Cuando las transacciones de relativamente larga duración son principalmente de solo lectura.

El aislamiento serializable permite que las transacciones simultáneas realicen solo aquellos cambios en la base de datos que podrían haber realizado si las transacciones hubieran sido programadas para ejecutarse una tras otra. Específicamente, Oracle permite que una transacción serializable modifique una fila de datos solo si se puede determinar que los cambios anteriores en la fila fueron realizados por transacciones que se habían comprometido cuando comenzó la transacción serializable.

Para determinar de manera eficiente, Oracle utiliza la información de control almacenada en el bloque de datos que indica que filas del bloque contienen cambios confirmados y no confirmados. En cierto sentido, el bloque contiene un historial reciente de transacciones que afectaron a cada fila del bloque. La cantidad de historial que se retiene está controlada por el `INITRANS`



Repeated query sees the same data, even if it was changed by another concurrent user

SET TRANSACTION ISOLATION  
LEVEL SERIALIZABLE

SELECT

SELECT

Fails attempting to update a row changed and committed by another transaction since this transaction began

UPDATE

IF "Can't Serialize Access"  
THEN ROLLBACK;  
LOOP and retry

	Leer comprometido	Serializable
Escritura Suces	Imposible	Imposible
Lectura Suces	Imposible	Imposible
Lectura no repetible	Posible	Imposible
Fantomas	Posible	Imposible
Cumple con ANSI/ISO SQL 92	Si	Si
Leer tiempo de visualización materializado	Declaración	Transacción
Coherencia del conjunto de transacciones	Nivel de declaración	Nivel de transacción
Bloque a nivel de fila	Si	Si
Los lectores bloquean a los escritores	No	No
Los lectores bloquean a los lectores	No	No
Los escritores de la misma fila bloquean a los escritores	No	No
Espere a que se bloquee la transacción	Si	Si
Subject cannot serialize access	No	Si



## ¿Cómo Oracle bloquea los datos?

Los bloqueos son mecanismos que evitan la interacción destructiva entre transacciones que acceden al mismo recurso, ya que sean objetos de usuarios, como tablas y filas, u objetos del sistema que no son visibles para los usuarios, como estructuras de datos compartidas en la memoria y filas de directorio de datos.

En todos los casos, Oracle obtiene automáticamente los bloqueos necesarios al ejecutar sentencias SQL, por lo que los usuarios no necesitan preocuparse por esos detalles. Oracle utiliza automáticamente el nivel más bajo de restricción aplicable para proporcionar el mayor grado de simultaneidad de datos y al mismo tiempo proporcionar integridad de datos a pruebas de fallos. Oracle también permite al usuario bloquear datos manualmente.

## Transacciones y simultaneidad de datos

Oracle proporciona simultaneidad e integridad de datos entre transacciones utilizando sus mecanismos de bloque. Debido a que los mecanismos de bloque de Oracle están estrechamente vinculados al control de transacciones, los diseñadores de aplicaciones sólo necesitan definir las transacciones correctamente y Oracle gestiona automáticamente el bloqueo.

Tener en cuenta que el bloqueo de Oracle es completamente automático y no requiere ninguna acción de usuario. El bloqueo implícito ocurre para todas las declaraciones SQL, de modo que el usuarios de la base de datos nunca necesitan bloquear ningún recurso explícitamente. Los mecanismos de bloqueo predeterminados de Oracle bloquean los datos en el nivel más bajo de restricción para garantizar la integridad de los datos y al mismo tiempo permitir el mayor grado de simultaneidad de datos.

## Modos de bloqueo

Oracle utiliza dos modos de bloqueo en una base de datos multiusuario.

- El modo de bloqueo exclusivo evita que se comparta el recurso asociado. Este modo de bloqueo se obtiene para modificar datos. La primera transacción que bloquea un recurso de forma exclusiva es la única transacción que puede modificar el recurso hasta que se libere el bloqueo exclusivo.

- El modo de bloqueo compartido permite compartir el recurso asociado según las operaciones involucradas. Varios usuarios que leen datos pueden compartirlos, manteniendo bloqueos compartidos para evitar el acceso simultáneo por parte de un escritor (que necesita un bloqueo exclusivo). Varios transacciones pueden adquirir bloqueos compartidos en el mismo recurso.

## Duración del bloqueo

Todos los bloqueos adquiridos por declaraciones dentro de una transacción se mantienen durante la duración de la transacción, lo que evita interferencias destructivas, incluidas lecturas sucias, actualizaciones perdidas y operaciones DDL destructivas de transacciones concurrentes. Los cambios realizados por la sentencias SQL de una transacción se vuelven visibles solo para otras transacciones que comienzan después de que se confirma la primera transacción.

Oracle libera todos los bloqueos adquiridos por las declaraciones dentro de una transacción cuando usted confirma o deshace la transacción. Oracle también libera los bloqueos adquiridos después de un punto de recuperación al retroceder al punto de recuperación. Sin embargo, solo las transacciones que no se esperan los recursos previamente bloqueados pueden adquirir bloqueos en los recursos ahora disponibles. Las transacciones en espera continuarán esperando hasta que la transacción original se confirme o se revierte por completo.

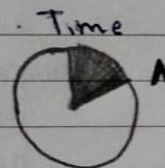


## Puntos Muertos

Puede ocurrir un punto muerto cuando dos o más usuarios están esperando datos bloqueados entre sí. Los interbloqueos impiden que algunas transacciones sigan funcionando.

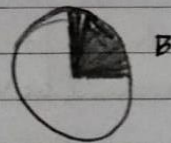
En la figura, no existe ningún problema en el momento A, ya que cada transacción tiene un bloqueo de fila en fila que intenta actualizar. Cada transacción continúa sin ser terminada. Sin embargo, cada una intenta a continuación actualizar la fila en la que está y tiene la otra transacción por lo tanto, se produce un punto muerto en el momento B, porque ninguna transacción puede obtener el recurso que necesita para continuar o terminar. Es un punto muerto porque no importa cuánto tiempo espere cada transacción, los bloqueos en conflicto se mantienen.

Transaction 1 (T1)  
UPDATE emp  
SET sal = sal\*1.1  
WHERE empno = 1000;



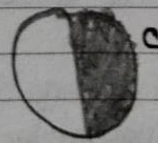
Transaction 2 (T2)  
UPDATE emp  
SET mgr = 1342  
WHERE empno = 2000;

UPDATE emp  
SET sal = sal\*1.1  
WHERE empno = 2000;



UPDATE emp  
SET mgr = 1342  
WHERE empno = 1000;

ORA-00060:  
deadlock detected while  
waiting for resource



## Detección de interbloqueo

Oracle detecta automáticamente situaciones de interbloqueo y las resuelve deshaciendo una de las declaraciones involucradas en el interbloqueo, liberando así un conjunto de bloqueos de fila en conflicto. También se devuelve el mensaje correspondiente a la transacción que se somete a una reversión a nivel de declaración. La declaración revertida es la que pertenece a la transacción que detecta el punto muerto.

## Tipos de cerraduras

Oracle utiliza automáticamente diferentes tipos de bloqueos para controlar el acceso simultáneo a los datos y evitar interacciones destructivas entre usuarios. Oracle bloquea automáticamente un recurso en nombre de una transacción para evitar que otras transacciones hagan algo que también requiera acceso exclusivo al mismo recurso. El bloqueo se libera automáticamente cuando ocurre algún evento para que la transacción ya no requiera el recurso.

Los bloqueos de Oracle se clasifican en una de tres categorías generales:

Cerrar	Descripción
Bloqueos DML (bloqueos de datos)	Los bloqueos DML protegen los datos. Por ejemplo, los bloqueos de tablas enteras, los bloqueos de filas seleccionadas.
Bloqueos DDL (bloqueos de diccionario)	Los bloqueos DDL protegen la estructura de los objetos del esquema, por ejemplo, las definiciones de tablas y vistas.
Cerraduras y pestillos internos	Los bloqueos y pestillos internos protegen las estructuras de bases de datos internas, como los archivos de datos. Las cerraduras y pestillos internos son totalmente automáticos.

## Descripción general de la consulta Flashback de Oracle

Oracle Flashback Query le permite ver y reparar datos históricos. Puede realizar consultas en la base de datos a partir de una determinada hora del reloj o número de cambio del sistema (SCN) especificado por el usuario.

Flashback query utiliza las capacidades de coherencia de lectura multiversión de Oracle para restaurar datos aplicados deshacer indica la cantidad de tiempo que debe transcurrir antes de que se puede sobrescribir la información de deshacer anterior (es decir, la información de deshacer de transacciones confirmadas). La base de datos recopila estadísticas de uso y ajusta el periodo de retención de los datos.



en función de estas estadísticas y del tamaño del espacio de tabla de deshacer.

Con Flashback Query, puede consultar la base de datos tal como existía esta mañana, ayer o la semana pasada. La velocidad de esta operación depende únicamente de la cantidad de datos que se consultan y de la cantidad de cambios en los datos que se deben deshacer.

Puedes consultar el suyo historial de una fila o transacción determinada, al utilizar los datos de deshacer almacenados en la base de datos, puede ser todas las versiones de una fila y volver a una versión anterior de esa fila. El historial de consultar de transacciones le permite examinar los cambios en la base de datos a nivel transacción.

### Beneficios de la consulta Flashback

- **Transparencia de la aplicación:** Las aplicaciones empaquetadas, como las herramientas de generación de informes que solo realizan consultas, se pueden ejecutar en modo Flashback Query mediante activadores de inicio de sesión. Las aplicaciones pueden ejecutarse de forma transparente sin necesidad de cambios en el código.
- **Rendimiento de la aplicación:** Si una aplicación requiere acciones de recuperación, puede hacerlo guardando los SCN y volviendo a esos SCN. Esto es mucho más fácil y rápido que guardar conjuntos de datos y restaurarlos más tarde, lo que sería necesario si la aplicación realizara un control de versiones explícito. Al utilizar Flashback Query, no hay costos de registro en los que se incurriría mediante el control de versiones explícito.

## B) PRACTICA

### Tipo de bloqueo 1

1.- Este tipo de bloqueo hace que no podamos hacer algún movimiento a la tabla

The screenshot shows two SQL Plus windows. The left window displays a successful update query: `SQL> UPDATE test`, `2 SET DAT01 = 'M'`, `3 WHERE DAT01 = 'x';`. The right window shows the command `SQL> LOCK TABLE citlallim.test IN EXCLUSIVE MODE;` resulting in `Table(s) Locked.` Below this, a query `SQL> SELECT * FROM V$LOCKED_OBJECT;` returns a table with columns: XIDUSN, XIDSLOT, XIDSQN, OBJECT\_ID, SESSION\_ID, ORACLE\_USERNAME, OS\_USER\_NAME, PROCESS, and LOCKED\_MODE. The table shows two rows: one for SYSTEM (6316:16064) and one for CITLALLIM (13944:13940). The CITLALLIM row is highlighted with a red box, indicating it is the process that is blocked.

2.- Por ejemplo, vamos a modificar un ejemplo de la tabla bloqueada y como vemos no puede hacer nuestra petición y se queda "trabada".

3.- Con esta sentencia podemos observar que esta bloqueado el proceso y no puede seguir el proceso.

The screenshot shows two SQL Plus windows. The left window displays the update query `SQL> UPDATE test`, `2 SET DAT01 = 'M'`, `3 WHERE DAT01 = 'x';` followed by `3 rows updated.` and `SQL>`. The right window shows the command `SQL> LOCK TABLE citlallim.test IN EXCLUSIVE MODE;` resulting in `Table(s) Locked.` Below this, a query `SQL> SELECT * FROM V$LOCKED_OBJECT;` returns the same table as before. The CITLALLIM row is highlighted with a red box, indicating it is the process that is blocked.

5.- Ya una vez que dimos commit ya se desbloquea el proceso.

4.- Para poder desbloquear el proceso tenemos que dar commit.



## Tipo de bloqueo 2

The screenshot shows two SQL Plus windows. The left window contains the following commands: `SQL> select * from test;`, `SQL> UPDATE test`, `2 SET DATO1 = 'x'`, `3 WHERE DATO1 = 'M';`, and `SQL> conn citlallis`. The right window shows the password prompt, connection confirmation, and the command `SQL> LOCK TABLE test IN SHARE MODE;` which results in `Table(s) Locked.`

```
SQL> select * from test;
DATO1
-----
M

SQL> UPDATE test
2 SET DATO1 = 'x'
3 WHERE DATO1 = 'M';
1 row updated.

SQL> conn citlallis
Enter password:
connected.

SQL> UPDATE citlallis.test
2 SET DATO1 = 'M'
3 WHERE DATO1 = 'x'
```

```
Enter password:
Connected to:
Oracle Database 11g Release 11.2.0.1.0 - 64bit Production

SQL> LOCK TABLE test IN SHARE MODE;
Table(s) Locked.
```

1.- Este tipo de bloque hace que no podamos modificar la tabla si no eres el usuario que la creó

2.- Como podemos observar estamos en sesión de citlallis que es la que creo la tabla entonces debemos de cambiarnos a citlallis que otro usuario que tenemos para poder probar el bloqueo.

The screenshot shows two SQL Plus windows. The left window contains the commands: `SQL> UPDATE test`, `2 SET DATO1 = 'x'`, `3 WHERE DATO1 = 'M';`, `SQL> conn citlallis`, and `SQL> UPDATE citlallis.test`. The right window shows the password prompt, connection confirmation, the command `SQL> LOCK TABLE test IN SHARE MODE;` resulting in `Table(s) Locked.`, and the command `SQL> SELECT * FROM V$LOCKED_OBJECT;` which returns a table of locked objects.

```
SQL> UPDATE test
2 SET DATO1 = 'x'
3 WHERE DATO1 = 'M';
1 row updated.

SQL> conn citlallis
Enter password:
connected.

SQL> UPDATE citlallis.test
2 SET DATO1 = 'M'
3 WHERE DATO1 = 'x'
```

```
Table(s) Locked.

SQL> commit;
Commit complete.

SQL> SELECT * FROM V$LOCKED_OBJECT;
XIDUSN XIDSLOT XIDSQN OBJECT_ID SESSION_ID
-----
ORACLE_USERNAME OS_USER_NAME
-----
PROCESS LOCKED_MODE
-----
SYSTEM 0 0 0 73641 8
9452:14536 citla 6
CITLALLIS 0 0 0 73641 191
13944:13940 citla 0
```

3.- Procedemos a querer modificar algo de la tabla y como observamos no procede por el bloqueo activado

4.- Para verificar el bloqueo ponemos esta sentencia y vemos que se bloqueó el proceso de citlallis .

```
SQL> UPDATE test
  2 SET DATO1 = 'x'
  3 WHERE DATO1 = 'M';
1 row updated.

SQL> conn citlallis
Enter password:
SQL> UPDATE citlallim.test
  2 SET DATO1 = 'M'
  3 WHERE DATO1 = 'x';
UPDATE citlallim.test
*
ERROR at line 1:
ORA-00028: your session has been killed
ORA-00028: your session has been killed

SQL>
```

```
SQL> rollback;
Rollback complete.

SQL> LOCK TABLE test IN SHARE MODE;
Table(s) Locked.

SQL> rollback;
Rollback complete.

SQL> SELECT * FROM V$LOCKED_OBJECT;
XIDUSN XIDSLOT XIDSQN OBJECT_ID SESSION_ID
-----
ORACLE_USERNAME OS_USER_NAME
-----
PROCESS LOCKED_MODE
-----
0 0 0 73641 8
SYSTEM
9452:14536 citla
6
0 0 0 73641 191
CITLALLIS
13944:13940 citla
0

XIDUSN XIDSLOT XIDSQN OBJECT_ID SESSION_ID
-----
ORACLE_USERNAME OS_USER_NAME
-----
PROCESS LOCKED_MODE
-----

SQL> SELECT SID, SERIAL#, STATUS
  2 FROM V$SESSION
  3 WHERE SID IN (SELECT SESSION_ID FROM V$LOCKED_OBJECT);
SID SERIAL# STATUS
-----
8 332 INACTIVE
191 86 ACTIVE

SQL> ALTER SYSTEM KILL SESSION '191,86';
System altered.

SQL>
```

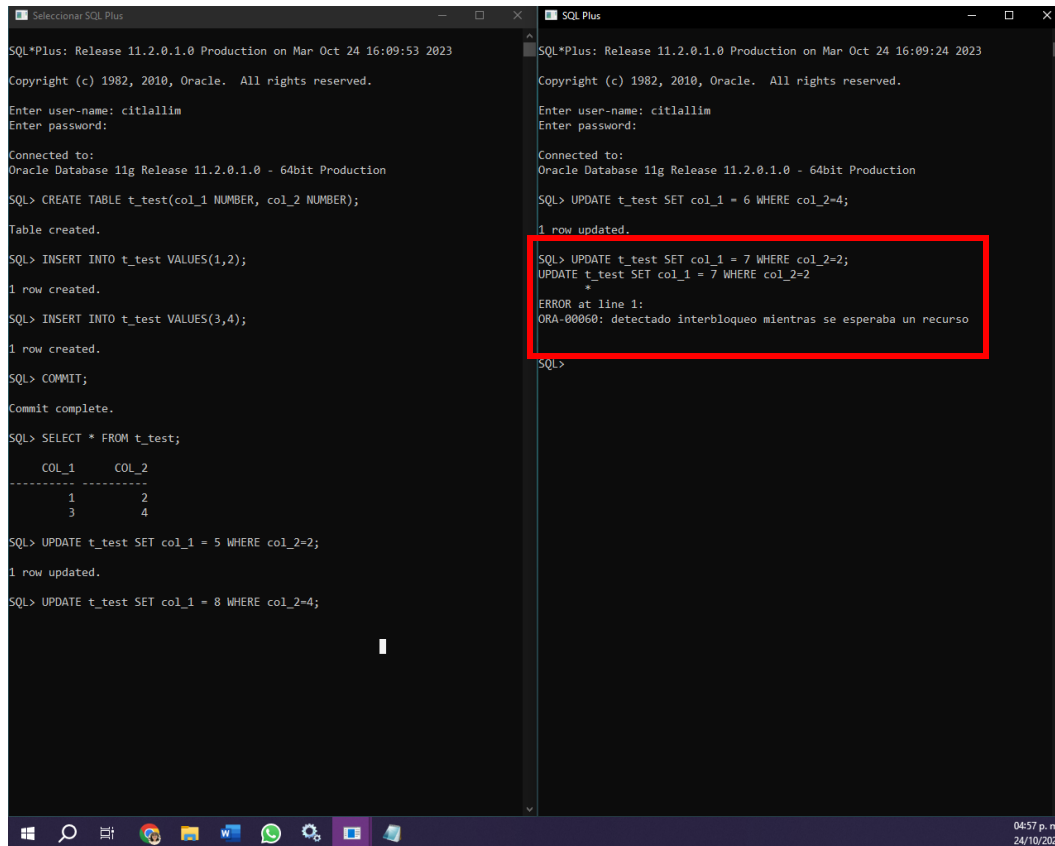
6.- Aquí ya se ve como se mato y se pudo desactivar.

5.- Como se muestra para desactivar el bloque se quiso usar el commit o el rollback, pero no se logro asi que se opto por matar el proceso y fue la única manera que pude hacer para desactivarlo



## C) DEADLOCK

Creamos una tabla y le insertamos unos datos para poder crear el deadlock vamos a actualizar datos de la tabla e intentamos actualizar en ambas sesiones la "col\_2" y vemos que hicimos un interbloqueo y aparece el error de que se detectó el deadlock



```
SQL*Plus: Release 11.2.0.1.0 Production on Mar Oct 24 16:09:53 2023
Copyright (c) 1982, 2010, Oracle. All rights reserved.
Enter user-name: citlallim
Enter password:
Connected to:
Oracle Database 11g Release 11.2.0.1.0 - 64bit Production
SQL> CREATE TABLE t_test(col_1 NUMBER, col_2 NUMBER);
Table created.
SQL> INSERT INTO t_test VALUES(1,2);
1 row created.
SQL> INSERT INTO t_test VALUES(3,4);
1 row created.
SQL> COMMIT;
Commit complete.
SQL> SELECT * FROM t_test;
   COL_1 COL_2
-----
      1      2
      3      4

SQL> UPDATE t_test SET col_1 = 5 WHERE col_2=2;
1 row updated.
SQL> UPDATE t_test SET col_1 = 8 WHERE col_2=4;
```

```
SQL*Plus: Release 11.2.0.1.0 Production on Mar Oct 24 16:09:24 2023
Copyright (c) 1982, 2010, Oracle. All rights reserved.
Enter user-name: citlallim
Enter password:
Connected to:
Oracle Database 11g Release 11.2.0.1.0 - 64bit Production
SQL> UPDATE t_test SET col_1 = 6 WHERE col_2=4;
1 row updated.
SQL> UPDATE t_test SET col_1 = 7 WHERE col_2=2;
UPDATE t_test SET col_1 = 7 WHERE col_2=2
*
ERROR at line 1:
ORA-00060: detectado interbloqueo mientras se esperaba un recurso
SQL>
```

## Bibliografía:

- Database concepts. (s. f.).  
[https://docs.oracle.com/cd/B19306\\_01/server.102/b14220/consist.html](https://docs.oracle.com/cd/B19306_01/server.102/b14220/consist.html)
- Deadlocks in Oracle. (s. f.). Stack Overflow.  
<https://stackoverflow.com/questions/28455231/deadlocks-in-oracle>