

```
CREATE TABLE cuentas (  
numero number primary key,  
saldo number not null  
);
```

--Table CUENTAS creado.

```
INSERT INTO cuentas VALUES (123, 400);
```

--1 fila insertadas.

```
INSERT INTO cuentas VALUES (456, 300);
```

--1 fila insertadas.

```
COMMIT;
```

--Confirmación terminada.

/\*

-- 1. Bloqueos(SELECT):

T1. SET AUTOCOMMIT OFF;

T2. SET AUTOCOMMIT OFF;

T1. UPDATE cuentas SET saldo = saldo + 100 WHERE numero = 123;

1 row updated.

T2. SELECT saldo FROM cuentas WHERE numero = 123;

SALDO

-----

400

T1. COMMIT;

Commit complete.

T2. SELECT saldo FROM cuentas WHERE numero = 123;

SALDO

-----

500

-- 2. Bloqueos(UPDATE):

T1. UPDATE cuentas SET saldo = saldo + 100 WHERE numero = 123;

1 row updated.

T2. UPDATE cuentas SET saldo = saldo + 200 WHERE numero = 123;

--Cuando tratamos de modificar los datos en T2, se queda bloqueada la pantalla de la consola de sql plus y,

--hasta que no hemos ejecutado en T1. COMMIT; Commit complete. la T2 no ha reaccionado. Esto ocurre para

--que las dos sesiones no accedan al mismo recurso a la vez. En cuanto hemos guardado los cambios

--en T1 ha ocurrido lo siguiente:

T2. 1 row updated.

T1. SELECT saldo FROM cuentas WHERE numero = 123;

SALDO

-----

600

--Es decir, solo se han sumado 100 al saldo, que antes su valor era 500. Esto implica que se ha ejecutado unicamente la

--sentencia de la sesión 1.

T2. COMMIT;

Commit complete.

T1. SELECT saldo FROM cuentas WHERE numero = 123;

SALDO

-----

800

--Es decir, se han sumado las 200 al saldo que faltaban de ejecutar el update de la sesión 2.

--3. Bloqueos(deadlock):

T1. UPDATE cuentas SET saldo = saldo + 100 WHERE numero = 123;

1 row updated.

T2. UPDATE cuentas SET saldo = saldo + 200 WHERE numero = 456;

1 row updated.

T1. UPDATE cuentas SET saldo = saldo + 300 WHERE numero = 456;

--T1 se ha quedado bloqueada y tiene que esperar a que se haga un commit en T2

T2. UPDATE cuentas SET saldo = saldo + 400 WHERE numero = 123;

--T2 se ha quedado bloqueada y tiene que esperar a que se haga un commit en T1.

Salta un error en T1:

UPDATE cuentas SET saldo = saldo + 300 WHERE numero = 456

\*

ERROR at line 1:

ORA-00060: deadlock detected while waiting for resource

--Este error salta porque ambas sesiones iban a quedar bloqueadas(se ha producido un deadlock) si no se hacía un COMMIT.

T1. COMMIT;

Commit complete.

T1. SELECT \* FROM cuentas;

NUMERO	SALDO
--------	-------

-----

123	900
-----	-----

456	300
-----	-----

T2. 1 row updated.

--Solo se ha ejecutado el primer update de la sesion 1.

T2. COMMIT;

Commit complete.

T1. SELECT \* FROM cuentas;

NUMERO	SALDO
--------	-------

-----

123    1300

456    500

--Se ejecutan ambos updates de la sesion 2.

--4. Niveles de aislamiento:

T1. ALTER SESSION SET ISOLATION\_LEVEL = SERIALIZABLE;

Session altered.

T1. SELECT SUM(saldo) FROM cuentas;

SUM(SALDO)

-----

1800

T2. UPDATE cuentas SET saldo = saldo + 100;

2 rows updated.

T2. COMMIT;

Commit complete.

T1. SELECT SUM(saldo) FROM cuentas;

SUM(SALDO)

-----

1800

--Como se ha cambiado el nivel de aislamiento a uno completo en la sesión 1,

--no ha permitido que la sesión 2 actualizase los datos que está utilizando su sesión.

--Hay consistencia de datos en el contexto de la sesión 1 y es por eso que la suma de

--salDOS no se ha visto modificada por la actualización de la sesión 2

T1. ALTER SESSION SET ISOLATION\_LEVEL = READ COMMITTED;

Session altered.

T1. SELECT SUM(saldo) FROM cuentas;

SUM(SALDO)

-----

2000

--Al cambiar la sesión 1 de serializable a read committed, los cambios que se

--comprometieron anteriormente en la sesión 2 ahora los ve la sesión 1 también

T2. UPDATE cuentas SET saldo = saldo + 100;

2 rows updated.

T2. COMMIT;

Commit complete.

T1. SELECT SUM(saldo) FROM cuentas;

SUM(SALDO)

-----

2200

--La sesión 1 ahora es read committed por lo que puede ver los cambios realizados

--en otras sesiones de los datos que está utilizando. Como se han actualizado los

--salos en la sesión 2 y se han comprometido, la sesión 1 los ve.

\*/

--5. Transacciones:

```
CREATE TABLE butacas(  
  id number(8) primary key,  
  evento varchar(30),  
  fila varchar(10),  
  columna varchar(10));
```

--Table BUTACAS creado.

```
CREATE TABLE reservas(  
  id number(8) primary key,  
  evento varchar(30),  
  fila varchar(10),  
  columna varchar(10));
```

--Table RESERVAS creado.

```
CREATE SEQUENCE Seq_Butacas INCREMENT BY 1 START WITH 1 NOMAXVALUE;
```

```
--Sequence SEQ_BUTACAS creado.
```

```
CREATE SEQUENCE Seq_Reservas INCREMENT BY 1 START WITH 1 NOMAXVALUE;
```

```
--Sequence SEQ_RESERVAS creado.
```

```
INSERT INTO butacas VALUES (Seq_Butacas.NEXTVAL,'Circo','1','1');
```

```
--1 fila insertadas.
```

```
INSERT INTO butacas VALUES (Seq_Butacas.NEXTVAL,'Circo','1','2');
```

```
--1 fila insertadas.
```

```
INSERT INTO butacas VALUES (Seq_Butacas.NEXTVAL,'Circo','1','3');
```

```
--1 fila insertadas.
```

```
COMMIT;
```

```
--Confirmación terminada.
```

```
/*
```

```
--3. Prueba el script desde SQL Developer reservando la fila 1, columna 1 para 'Circo'. Antes hay que modificar las rutas que hacen referencia a los otros scripts a la carpeta en donde se hayan depositado.
```

```
SQL> @'C:\Users\slato\Documents\UniUCM\BD\practica7\script.sql'
```

```
Commit complete.
```

INFO: Se intenta reservar.

PL/SQL procedure successfully completed.

SCRIPT\_COL

-----

"C:\Users\slato\Documents\UniUCM\BD\practica7\preguntar.sql"

V\_ERROR

-----

false

'?Confirmar la reserva?'

s

s

SCRIPT\_COL

-----

"C:\Users\slato\Documents\UniUCM\BD\practica7\preguntar.sql"

INFO: Localidad reservada.

PL/SQL procedure successfully completed.

Commit complete.

--4. Intenta reservar de nuevo la misma fila desde la misma consola SQL Developer y comprueba que no sea posible.

```
SQL> @'C:\Users\slato\Documents\UniUCM\BD\practica7\script.sql'
```

Commit complete.

ERROR: La localidad ya está reservada.

PL/SQL procedure successfully completed.

SCRIPT\_COL

-----

"C:\Users\slato\Documents\UniUCM\BD\practica7\no\_preguntar.sql"

V\_ERROR

-----

PL/SQL procedure suc

```
SQL> @'C:\Users\slato\Documents\UniUCM\BD\practica7\script.sql'
```

Commit complete.

cessfully completed.true

n

SCRIPT\_COL

-----



"C:\Users\slato\Documents\UniUCM\BD\practica7\no\_preguntar.sql"

INFO: No se ha reservado la localidad.

PL/SQL procedure successfully completed.

Commit complete.

--5. Realiza una nueva reserva desde la misma consola para la fila 1, columna 4 para la 'Circo' y comprueba que no es posible porque no existe esa butaca.

ERROR: No existe esa localidad.

SCRIPT\_COL

-----

"C:\Users\slato\Documents\UniUCM\BD\practica7\no\_preguntar.sql"

V\_ERROR

-----

true

n

SCRIPT\_COL

-----

"C:\Users\slato\Documents\UniUCM\BD\practica7\no\_preguntar.sql"

INFO: No se ha reservado la localidad.

PL/SQL procedure successfully completed.

Commit complete.

--6. Realiza una nueva reserva desde la misma consola para la fila 1, columna 2 para la 'Circo' pero sin realizar aún la confirmación.

T1: SQL> @'C:\Users\slato\Documents\UniUCM\BD\practica7\script.sql'

Commit complete.

INFO: Se intenta reservar.

PL/SQL procedure successfully completed.

SCRIPT\_COL

-----

"C:\Users\slato\Documents\UniUCM\BD\practica7\preguntar.sql"

V\_ERROR

-----

false

'?Confirmar la reserva?'

--7. Abre una nueva instancia de SQL Developer y realiza la misma reserva anterior desde esta instancia.

T2: SQL> @'C:\Users\slato\Documents\UniUCM\BD\practica7\script.sql'

SP2-0103: Nothing in SQL buffer to run.

INFO: Se intenta reservar.

PL/SQL procedure successfully completed.

SCRIPT\_COL

-----

"C:\Users\slato\Documents\UniUCM\BD\practica7\preguntar.sql"

V\_ERROR

-----

false

--8. Confirma la reserva del punto 7. ¿Qué sucede?

'?Confirmar la reserva?'

s

s

SCRIPT\_COL

-----

"C:\Users\slato\Documents\UniUCM\BD\practica7\preguntar.sql"

INFO: Localidad reservada.

PL/SQL procedure successfully completed.

Commit complete.

T1: s

s

SCRIPT\_COL

-----

"C:\Users\slato\Documents\UniUCM\BD\practica7\preguntar.sql"

INFO: Localidad reservada.

PL/SQL procedure successfully completed.

Commit complete.

SQL> select \* from reservas;

ID EVENTO	FILA	COLUMNA
-----		
4 Circo	1	2
2 Circo	1	1
3 Circo	1	2

El script está descrito de tal manera que acepta reservas repetidas cuando dos transacciones simultaneamente realizan la misma reserva.

--9. Modifica el script para resolver el punto anterior.

Se arregla añadiendo al principio del script la siguiente línea:

```
ALTER SESSION SET ISOLATION_LEVEL = SERIALIZABLE;
```

El resultado es:

T1: '?Confirmar la reserva?'

s

s

SCRIPT\_COL

-----

"C:\Users\slato\Documents\UniUCM\BD\practica7\preguntar.sql"

begin

\*

ERROR at line 1:

ORA-08177: can't serialize access for this transaction

ORA-06512: at line 3

Commit complete.