APARTADO 1

3-SET AUTOCOMMIT OFF

1-CREATE TABLE cuentas (

numero number primary key,

saldo number not null

);

INSERT INTO cuentas VALUES (123, 400);

INSERT INTO cuentas VALUES (456, 300);

COMMIT;

2- Se abre una instancia T2

4- UPDATE cuentas SET saldo=saldo + 100 WHERE numero=123;COMMIT;

5-SALDO desde T2: 400

7-SALDO desde T2: 500

APARTADO 2

1-UPDATE cuentas SET saldo=saldo + 100 WHERE numero=123;COMMIT;

2- No se puede, está en continuo funcionamiento

3-Al hacer el commit en T1 resulta que T2 terminaba y ejecutaba el update.

4- select saldo from cuentas WHERE numero=123;

SALDO desde T1: 600

6- select saldo from cuentas WHERE numero=123;

SALDO desde T1: 800

APARTADO 3

En T1

1-UPDATE cuentas SET saldo=saldo + 100 WHERE numero=123;

3-UPDATE cuentas SET saldo=saldo + 300 WHERE numero=456;

COMMIT;

En T2

2-UPDATE cuentas SET saldo=saldo + 200 WHERE numero=456;

4-UPDATE cuentas SET saldo=saldo + 400 WHERE numero=123;

COMMIT;

Igual que en la primera parte del apartado dos no se ejecutan las instrucciones 3 y 4 puesto que las instrucciones 1 y 2 han actualizado las mismas cuentas y todavía no se ha hecho commit.

Se produce un interbloqueo en la instrucción 3 al hacer el commit en T1; puesto que este commit permite la realización de la instrucción 4.

La tabla acaba con los datos:

Cuenta 123----- > 1300

Cuenta 456----- > 500

APARTADO 4

1-

ALTER SESSION SET ISOLATION_LEVEL = SERIALIZABLE;

Session alterado.

2-

SELECT SUM(saldo) FROM cuentas;

SELECT SUM(saldo) FROM cuentas; -------> 1800

3-

UPDATE cuentas SET saldo=saldo+100; COMMIT

2 filas actualizadas. Confirmación terminada.

4-

SELECT SUM(saldo) FROM cuentas;

SUM(SALDO)----- >1800

La suma del saldo permanece invariante, aunque a primera vista hubiéramos pensado que tomaría el valor 2000; pero al aislar T1 del resto, no le afectan los cambios confirmados de otras transacciones.

5-

ALTER SESSION SET ISOLATION_LEVEL = READ COMMITTED;

Session alterado.

6-

SELECT SUM(saldo) FROM cuentas;

SUM(SALDO)----- >2000 (se consideran ya las actualizaciones de T2)

7-

```
UPDATE cuentas SET saldo=saldo +100; COMMIT;
```

2 filas actualizadas. Confirmación terminada.

8-

```
-- 4
SELECT SUM(saldo) FROM cuentas;
```

SUM(SALDO)----- >2200

APARTADO 5

1-

```
CREATE TABLE butacas(id number(8) primary key,
 evento varchar(30),
fila varchar(10),
columna varchar(10)) ;
CREATE TABLE reservas(id number(8) primary key,
 evento varchar(30),
 fila varchar(10),
 columna varchar(10)) ;
CREATE SEQUENCE Seq_Butacas INCREMENT BY 1 START WITH 1
NOMAXVALUE;
CREATE SEQUENCE Seq_Reservas INCREMENT BY 1 START WITH 1
NOMAXVALUE;
```

Table BUTACAS creado. Table RESERVAS creado.

Sequence SEQ_BUTACAS creado. Sequence SEQ_RESERVAS creado.

2-

```
INSERT INTO butacas VALUES (Seq_Butacas.NEXTVAL,'Circo','1','1');
INSERT INTO butacas VALUES (Seq_Butacas.NEXTVAL,'Circo','1','2');
INSERT INTO butacas VALUES (Seq_Butacas.NEXTVAL,'Circo','1','3');
COMMIT;
```

3 filas insertadas. Confirmación terminada.

3-@ ‘C:\hlocal\script.sql’

INFO: Se intenta reservar.

Procedimiento PL/SQL terminado correctamente.

SCRIPT_COL

"C:\hlocal\MIC\preguntar.sql"

V_ERROR

---

false

'¿Confirmar la reserva?'

s

INFO: Localidad reservada.

Procedimiento PL/SQL terminado correctamente.

Confirmación terminada.

4- @ ‘C:\hlocal\script.sql’

ERROR: La localidad ya está reservada.

Procedimiento PL/SQL terminado correctamente.

SCRIPT_COL

"C:\hlocal\MIC\preguntar.sql"

V_ERROR

---

true

'¿Confirmar la reserva?'

s

INFO: No se ha reservado la localidad.

Procedimiento PL/SQL terminado correctamente.

Confirmación terminada.

5- @ ‘C:\hlocal\script.sql’

ERROR: No existe esa localidad.

Procedimiento PL/SQL terminado correctamente.

SCRIPT_COL

"C:\hlocal\MIC\preguntar.sql"

V_ERROR

---

true

'¿Confirmar la reserva?'

s

INFO: No se ha reservado la localidad.

Procedimiento PL/SQL terminado correctamente.

Confirmación terminada.

6- @ 'C:\hlocal\script.sql'

7- Se realiza la reserva planteada en el punto 7 antes de confirmar la del punto 6

@ 'C:\hlocal\script.sql'

INFO: Se intenta reservar.

Procedimiento PL/SQL terminado correctamente.

SCRIPT_COL

"C:\hlocal\MIC\preguntar.sql"

V_ERROR

---

false

'¿Confirmar la reserva?'

s

INFO: Localidad reservada.

Procedimiento PL/SQL terminado correctamente.

Confirmación terminada.

8- Me permite confirmar la reserva del punto 6.

INFO: Se intenta reservar.

Procedimiento PL/SQL terminado correctamente.

SCRIPT_COL

"C:\hlocal\MlC\preguntar.sql"

V_ERROR

---

false

'¿Confirmar la reserva?'

s

INFO: Localidad reservada.

Procedimiento PL/SQL terminado correctamente.

Confirmación terminada.

9- SET TRANSACTION ISOLATION LEVEL = SERIALIZABLE;

@ 'C:\hlocal\script.sql'

Me permite hacer la reseva propuesta en el punto 7

Transaction ISOLATION correcto.

INFO: Se intenta reservar.

Procedimiento PL/SQL terminado correctamente.

SCRIPT_COL

"C:\hlocal\MlC\preguntar.sql"

V_ERROR

---

false

'¿Confirmar la reserva?'

s

INFO: Localidad reservada.

Procedimiento PL/SQL terminado correctamente.

Confirmación terminada.

Al confirmar la reserva del punto 6 salta el error:

begin

```
 if '&v_confirmar'='s' and :v_error='false' then
   insert into reservas values (Seq_Reservas.NEXTVAL,'&v_evento','&v_fila','&v_columna');
   dbms_output.put_line('INFO: Localidad reservada.');
 else
   dbms_output.put_line('INFO: No se ha reservado la localidad.');
 end if;
end;
```

```
Informe de error -
ORA-08177: no se puede serializar el acceso para esta transacción
ORA-06512: en línea 3
08177. 00000 -  "can't serialize access for this transaction"
*Cause:    Encountered data changed by an operation that occurred after
           the start of this serializable transaction.
*Action:   In read/write transactions, retry the intended operation or
           transaction.
Confirmación terminada
```