Ejercicios de Transacciones con Bloqueos

====== **EJ-TB1** ========== ==========

Queremos hacer una aplicación con transacciones concurrentes para comprar entradas de cine con esta BD:

Butaca(codSala,codButaca,. . .)
Sala(codSala, totButacas. . .)
Entrada(dni,codSala,codButaca, fechaSesion,horaSesion,. . .)
Usuario(eni,nom,ape,. . .)

Escoger entre estas opciones para tus transacciones:
- Read commited
- Serializable
- Select for update
- Lock table

Tener en cuenta dos situaciones de terminación:

a) El usuario termine su compra
b) El usuario se arrepienta y cancele la operación

Se pide:

a) - Definir los pasos (textualmente) a dar para que la compra de una entrada sea correcta en un entorno de transacciones concurrentes
b) – Escribir las instrucciones PL/SQL adecuadas para esos pasos.

========**EJ-TB2** ==========

En la BDejemplo ha llegado una compra nueva que queremos procesar con los siguientes valores de sus atributos en el orden de creación de la tabla:

('00000003', '30000002',111, 0501,'sesion 1',1)

Los pasos que queremos seguir son:

1.- Comprobar que tiene saldo (en tabla TieneT) la tarjeta con la que se va a pagar. Mientras se hace la comprobación no queremos que nadie pueda consultar ni actualizar esa tarjeta, así que la bloqueamos

2.- Si el saldo de la tarjeta es mayor o igual a la compra, entonces:
- Creamos una fila en la tabla Compras con los valores de la compra
- Actualizamos el saldo de la tarjeta, restando al saldo actual el importe de la compra.

3.- Desbloqueamos la tarjeta y la tabla compras.

Se pide: Escribir las instrucciones en PL/SQL para ejecutar dichos pasos

SOLUCION:

En un procedimiento declaramos saldovar antes

```
-- 1.- bloqueo el saldo de la cuenta
    select saldo INTO saldovar
    from TieneT
    where dni = '00000003' and numt ='30000002'
    for update;
```

```
if  saldovar  >  1 THEN
     INSERT INTO Compras VALUES ('00000003', '30000002',111, 0501,'sesion 1',1);
      update TieneT
            set saldo = saldo - 1
     where dni = '00000003' and numt ='30000002';
      commit;
End if;
```

====== EJ-TB3 ==

Queremos comprar billetes de avión en esta BD:

vuelo(num-V, num-Avion, orig-V, dest-V, diaSemana, hora-Sale, duración)
asiento(numAvion, numAsiento, fila-A, letra-A, ocupado)
reserva(numRes, num-V, fila-A, letra-A, fecha, NumPasaporte)
cliente(NumPasaporte, nombre, apellido1, apellido2)

- Un viaje tiene varias escalas:
  - cada tramo es un vuelo independiente, que corresponde a una reserva independiente
  - Cada reserva tiene el mismo numRes para todos los tramos del viaje
  - La reserva consta de tres tramos: de C1 a C2, de C2 a C3 y de C3 a C4
  - Todos los tramos se hacen el mismo día DX

- Vamos pidiendo datos el usuario. Asumimos que tenemos cada dato en una variable **v_nombre_NN** donde nombre es el nombre del atributo en la tabla correspondiente y NN para cuando se necesiten varias variables del mismo atributo. Se deben comprobar y bloquear lo necesario para que mientras se realiza el proceso no interfiera otra transacción, sin por ello bloquear toda una tabla.

Se pide:

a) Escribe en texto cada paso a dar para realizar la compra.
b) Escribe las instrucciones PL/SQL para realizar la compra. Incluye el nivel de aislamiento

==== EJ-TB4 ==

Queremos aplicar descuentos (dto) a una tabla de usuarios que se usa para facturar. No queremos que mientras se hace el proceso se ejecute ninguna factura para ningún usuario, puesto que actualiza totFacturado.
Usuario(Nom-U, app-U. dni-U, dto , totalFacturado)

Si el totalFacturado es menor que 100 el dto será cero; si está entre 100 y 1000 el dto es 10%;
Si sobrepasa los 1000, el dto es 15%

Se pide:

a) Escribe en texto cada paso a dar para realizar la compra.
b) Escribe las instrucciones PL/SQL para realizar la compra. Incluye el nivel de aislamiento

==== EJ-TB5 ==========

a) Estudia la secuencia de operaciones de los escenarios de la tabla para ver cómo reacciona Oracle en los distintas situaciones. Ejecútalas. Hay algún error? (Recordar usar **set autocommit off**)

b) Crea otros escenarios con situaciones distintas

SESION 1	SESION 2
-- ESCENARIO: dos sesiones serializables: **-- sesión 1 bloquea SOLO una fila con select . . TieneT . . . for update** **-- sesion 2 bloquea toda la tabla al hacer INSERT INTO Compras** **-- paso 1 -- session 1** set transaction isolation level serializable; -- debe responder: transaction ISOLATION correcto. set autocommit off;	
	-- paso 2 -- session 2 commit; set transaction isolation level serializable; set autocommit off; INSERT INTO Compras VALUES ('00000003', '30000002',200, 0501,'sesion2:una',1); select saldo from TieneT where dni = '00000003' and numt ='30000002'; -- da el inicial = 30, permite hacerla INSERT INTO TieneT VALUES ('00000004', '30000300', 0901, 40300);
-- paso 3 -- session 1 select saldo from TieneT where dni = '00000003' and numt ='30000002' for update; -- solo bloquea esa fila para actualizar -- da el inicial = 30	
	-- paso 4 -- session 2 select saldo from TieneT where dni = '00000003' and numt ='30000002'; -- devuelve el valor inicial : 30 -- permite consultarla (no deja si es UPDATE) update TieneT set saldo = saldo - 1 where dni = '00000004' and numt = '30000300';

	-- permite: esta fila no está bloqueada select saldo from TieneT where dni = '00000004' and numt = '30000300'; -- devuelve el valor actualizado: 40299
-- paso 5 -- session 1 update TieneT set saldo = saldo - 1 where dni = '00000003' and numt ='30000002'; -- permite: la bloqueó esta sesión select saldo from TieneT where dni = '00000003' and numt ='30000002'; -- devuelva el valor actualizado por ella: 29	
	-- paso 6 -- session 2 select * from compras where dni= '00000003' and numt = '30000002'; -- filas: la inicial y la añadida por esta sesión 00000003 30000002 1 501 tienda7 3 00000003 30000002 200 501 sesion2:una 1
-- paso 7 -- session 1 INSERT INTO Compras VALUES ('00000003', '30000002',111, 0501,'sesion1:??',1); -- Error SQL: ORA-08177: no se puede serializar el acceso para esta transacción -- porque está "serializable" y eso bloquea la tabla compras entera debido a que -- sesion 2 actualizó antes COMPRAS y así la bloqueó select * from compras where dni= '00000003' and numt = '30000002'; -- responde , pero no ve la fila nueva de la sesion 2 00000003 30000002 1 501 tienda7 3	
	---- paso 8 -- sesion 2 update TieneT set saldo = saldo - 1 where dni = '00000005' and numt ='50000030'; -- fila que existía antes de empezar las dos sesiones 00000005 50000030 901 500 -- da ORA-08177: no se puede serializar el acceso para esta transacción CAUSA: por se serializable (no por bloqueada)

	Commit **Borrar y crear BDEjemplo** - da error de tabla Cliente (y otras) ORA-00054: recurso ocupado y obtenido con NOWAIT especificado o timeout vencido →la otra trans. tiene bloqueada tabla con claves ajenas en Cliente
commit	
	Borrar y crear BDEjemplo - lo ejecuta bien
-- ESCENARIO: dos sesiones serializables: **-- sesión 1 bloquea SOLO una fila con select . . TieneT . . . for update** **-- sesion 2 bloquea SOLO otra fila select . . TieneT . . . for update** set transaction isolation level serializable;	
	Commit **set transaction isolation level serializable;**
- paso 1 -- session 1 select saldo from TieneT where dni = '00000003' and numt ='30000002' for update; -- solo bloquea esa fila = 30	
	- paso 2 -- session 2 select saldo from TieneT where dni = '00000003' and numt ='30000002' for update; -- se queda esperando (cancelar a mano) select saldo from TieneT where dni = '00000005' and numt = '50000030' for update; -- da 500 -- solo está bloqueada la otra fila en sesion1 update TieneT set saldo = saldo - 1 where dni = '00000005' and numt ='50000030'; -- queda saldo= 499 -- deja actualizar porque está bloqueada otra fila en la sesión 1 -- 1a fila selccionada con select . . . for update
update TieneT set saldo = saldo - 1	

where dni = '00000003' and numt ='30000002'; -- ejecuta: 1 filas actualizadas -> da 29 select saldo from TieneT where dni = '00000005' and numt = '50000030'; -- ejecuta, pero da 500 (valor inicial) update TieneT set saldo = saldo - 1 where dni = '00000005' and numt ='50000030'; -- se queda esperando, está bloqueada en sesion2	
	commit
Error SQL: ORA-08177: no se puede serializar el acceso para esta transacción: -- por ser serializable no permite actualizar, Aunque ha liberado la fila la sesion2 -- no aborta la trans select saldo from TieneT where dni = '00000005' and numt = '50000030'; -- sigue viendo valor inicial 500 commit	
-- ESCENARIO: dos sesiones read committed: **-- sesión 1 bloquea SOLO una fila con select . . TieneT . . . for update** **-- sesion 2 bloquea SOLO otra fila select . . TieneT . . . for update** commit; set transaction isolation level read committed;	
	Borrar y crear BDEjemplo - lo ejecuta bien commit **set transaction isolation level read committed;**
- paso 1 -- session 1 select saldo from TieneT where dni = '00000003' and numt ='30000002' for update; -- solo bloquea esa fila = 30	
	- paso 2 -- session 2 select saldo from TieneT where dni = '00000005' and numt = '50000030' for update; -- da 500 -- deja actualizar porque está bloqueada otra fila

	en la sesión 1 -- 1a fila selccionada con select . . . for update update TieneT set saldo = saldo - 1 where dni = '00000005' and numt = '50000030'; -- da 499
select saldo from TieneT where dni = '00000005' and numt = '50000030'; -- da 500	
	commit
select saldo from TieneT where dni = '00000005' and numt = '50000030'; -- da 499 sin salir de su trans, por ser read commited y la sesion2 ha hecho commit	
	select saldo from TieneT where dni = '00000003' and numt ='30000002' for update; -- espera (fila bloqueada en session1)
commit	
	-- ejecuta y da el resultado= 30 Commit
-- **ESCENARIO: dos sesiones read committed:** -- **sesión 1 bloquea tabla exclusive** Lock table Tienet in exclusive mode **select . . TieneT . . . for update** -- **sesion 2 bloquea SOLO otra fila** **select . . TieneT . . . for update** set transaction isolation level read committed;	
	set transaction isolation level read committed;
Lock table Tienet in exclusive mode;	
	update TieneT set saldo = saldo - 1 where dni = '00000005' and numt = '50000030'; -- se queda esperando (cancelo a mano) Lock table Tienet in share mode; -- se queda esperando (cancelo a mano)

update TieneT set saldo = saldo - 1 where dni = '00000003' and numt ='30000002'; -- actualizado Commit; -- **termina transacción** -- empieza otra	
	update TieneT set saldo = saldo - 1 where dni = '00000005' and numt = '50000030'; -- ejecuta : actualiza: 499 Lock table Tienet in share mode;
Lock table Tienet in share mode; -- Se queda esperando (cancelo a mano) porque sesion2 hizo un update que bloque la tabla	
	update TieneT set saldo = saldo - 1 where dni = '00000003' and numt ='30000002'; -- ejecuta : 28 Commit -- **termina transacción** -- empieza otra Lock table Tienet in share mode;
Lock table Tienet in share mode; -- ejecuta el lock Lock table Tienet in exclusive mode; -- se queda esperando	