Materia: ADMINISTRACIÓN DE BASES DE DATOS
Maestro: Gerardo Rodríguez Rojano
Fecha: 31 de Mayo 2019
T A R E A NO. 7

Fecha de Entrega: 31 de mayo 2019;

Modalidad: En equipos de 2 a 3 integrantes.
Forma de entregar: subir este archivo con las respuestas en cada pregunta en formato PDF al campus virtual del curso.

Datos de Integrantes:

Expediente	Nombre Completo (nombre y apellidos)
258849	**Michael Brandon Serrato Guerrero**

INSTRUCCIONES:
Leer los apuntes de la Unidad 5 del curso de Administración de Bases de Datos y contestar el siguiente cuestionario.

Fuente de estudio: Administracion deTransacciones - Unidad 5 - 2019

1. Considere una base de datos con las tablas:

Artículos(artclave(Pk), artnombre, artprecio, artstock, proveedorclave);

Proveedores(proveedorclave(Pk), proveedornombre);

Ventas (vtanumero(Pk), vtafecha, vtatotal);

ventasDetalle (vtanumero(Pk), artclave(Pk), vtadetprecio, vtadetcantidad);

nota: los atributos llave primaria están subrayados y con la palabra Pk.

Crear la tabla con código DDL y agregar las siguientes restricciones:

- Los campos subrayados e indicados *pk* son la llave primaria.
- Validar en tabla artículos que un precio sea por defecto 10 y no menor de 10.
- Validar en tabla artículos que el stock no sea negativo.
- Validar en tabla ventasdetalle que un precio sea mayor o igual a 10.
- Validar en tabla ventasdetalle que una cantidad sea mayor a cero.
- Especificar la relación de integridad entre artículos y proveedores.
- Especificar la relación de integridad entre ventas y ventasdetalle.
- Especificar la relación de integridad entre artículos y ventasdetalle
- Decidir los tipos de datos de cada atributo donde ninguno vaya en contra de las restricciones impuestas al diseño de las tablas.

CÓDIGO:

```
CREATE DATABASE tarea_07_db ENCODING 'utf8';

-- Proveedores(proveedorclave(Pk), proveedornombre);
CREATE TABLE Proveedores (
    proveedorclave SERIAL PRIMARY KEY,
    proveedornombre VARCHAR(45)
);

-- Artículos(artclave(Pk), artnombre, artprecio, artstock, proveedorclave);
CREATE TABLE Articulos (
    artclave SERIAL PRIMARY KEY,
    artnombre VARCHAR(45),
    artprecio DECIMAL(10, 2) DEFAULT 10.00 CHECK(artprecio >= 10.00),
    artstock INT DEFAULT 0 CHECK(artstock >= 0),
    proveedorclave INT NOT NULL
);
```

```sql
ALTER TABLE Articulos
   ADD CONSTRAINT fk_proveedores_articulos
   FOREIGN KEY (proveedorclave)
   REFERENCES Proveedores (proveedorclave);

-- Ventas (vtanumero(Pk), vtafecha, vtatotal);
CREATE TABLE Ventas (
    vtanumero SERIAL PRIMARY KEY,
    vtafecha TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    vtatotal DECIMAL(10, 2)
);

-- ventasDetalle (vtanumero(Fk), artclave(Fk), vtadetprecio, vtadetcantidad);
CREATE TABLE VentasDetalle (
    id SERIAL PRIMARY KEY,
    vtanumero  INT NOT NULL,
    artclave INT NOT NULL,
    vtadetprecio DECIMAL(10, 2) CHECK(vtadetprecio >= 10.00),
    vtadetcantidad INT CHECK(vtadetcantidad > 0)
);

ALTER TABLE VentasDetalle
   ADD CONSTRAINT fk_ventas_ventasdetalle
   FOREIGN KEY (vtanumero)
   REFERENCES Ventas (vtanumero);

ALTER TABLE VentasDetalle
   ADD CONSTRAINT fk_articulos_ventasdetalle
   FOREIGN KEY (artclave)
   REFERENCES Articulos (artclave);
```

2. ¿Por qué una transacción es la unidad de recuperación?

R: Una transacción es una unidad de recuperación puesto que ya sea confirmada o fallida en ambos casos se debe mantener la consistencia de la base de datos. Específicamente hablando en caso de falla todo lo que se modifica durante la transacción sobre las tablas debe ser restaurado a su estado inmediatamente anterior (consistente) basándose en el redo log file. Ahora bien, nos referimos a "consistencia" como aquel estado en el que la base de datos obedece a todas las restricciones de integridad definidas sobre ella.

3. Las características ACID dicen: cada transacción debe preservar la integridad de la base de datos. ¿Qué relación existe con la descripción completa y detallada de los casos de uso?

R: Las propiedades ACID se encuentran en todos los casos de uso que a su vez dependen de las transacciones. La atomicidad es la propiedad que consiste en ejecutar todas o ninguna de las instrucciones de las transacciones, por lo que se asegura que un caso de uso debe estar completo (ser ejecutado en su totalidad y no parcialmente). La consistencia es la propiedad que asegura que no se van a romper las reglas y/o directrices de integridad en la base de de datos, por lo tanto los casos de uso deberán ser cumplidos según sus restricciones. El aislamiento es aquella que asegura que una operación no puede afectar a otras (por lo que no es posible alterar el funcionamiento de los demás casos de uso). Por último la durabilidad que asegura que una vez realizada la operación, ésta persistirá y no se podrá deshacer aún en caso de fallo del sistema (logrando la persistencia de los datos en base a la definición de los casos de uso).

4. ¿En qué consiste el control de concurrencia en bases de datos?

R: El control de concurrencia se refiere al mecanismo en que los DBMS permiten que las transacciones accedan a una misma base de datos al mismo tiempo sin afectar el resultado final y su consistencia. El control de accesos concurrentes y específicamente de transacciones concurrentes es manejado por un módulo del DBMS llamado Scheduler.

5. ¿Cuándo se dice que dos o más transacciones son serializables?

R: Cuando las transacciones son concurrentes y las ejecuciones de dichas transacciones concurrentes son equivalentes a una ejecución en serie.

6. Explique: 1) Lectura sucia 2) Lectura no repetible 2) Lectura fantasma

R:

1. **Lectura sucia:** Sucede cuando una transacción lee un registro utilizado por una transacción previa, antes de que ésta última haga commit posteriormente falle. De tal manera que la información obtenida es incorrecta.
2. **Lectura no repetible:** Sucede cuando durante una transacción se espera el mismo resultado de una cierta lectura de datos en distintas ocasiones, pero estos son alterados por otra transacción generando resultados diferentes a la primera lectura realizada por la primera transacción.
3. **Lectura fantasma:** Sucede cuando una transacción vuelve a ejecutar una consulta que devuelve un conjunto de registros que satisfacen una condición de búsqueda y encuentra que el conjunto de registros que

cumplen dicha condición ha cambiado debido a otra transacción recientemente confirmada.

7. ¿En qué consiste el problema del análisis inconsistente? ¿por qué es inconsistente?

R: Consiste en la ejecución de transacciones cuya lectura de datos es interferida por la ejecución de otras transacciones de duración menor, alterando los resultados, y por lo tanto, provocando una inconsistencia en los registros de las tablas de la base de datos.

8. ¿Por qué utilizar el protocolo READ-COMMITTED (Read Committed Isolation Level) garantiza mayor concurrencia de transacciones que utilizar el protocolo SERIALIZABLE?

R: Porque el protocolo **SERIALIZABLE** es el nivel máximo de aislamiento y **es aquel que genera el nivel máximo de bloqueos** por lo que la habilidad de ejecutar múltiples transacciones (concurrencia) disminuye. A diferencia del protocolo READ COMMITED que únicamente genera bloqueos durante la ejecución de las lecturas y no de la transacción completa, permitiendo una mayor disponibilidad de los registros para las demás transacciones pero a su vez generando el problema de las lecturas no repetibles y lecturas fantasma.

9. Explique la situación de Dead Lock que se puede provocar entre transacciones concurrentes.

R: Es una situación que sucede cuando dos transacciones solicitan recursos que ambas poseen. Desde otro punto, un deadlock es el momento en que se genera un ciclo de grafo en la asignación de recursos. Para evitar que ambas entren en una espera indefinida, el DBMS elimina la transacción que generó dicho ciclo.

10. Explique la diferencia entre un SELECT y un SELECT FOR SHARE.

R: Un SELECT normal permite la lectura de los datos sin generar bloqueo alguno (más que durante la extracción de los datos en dicha consulta únicamente), mientras que un SELECT FOR SHARE genera un bloqueo compartido sobre un conjunto de registros para garantizar la lectura repetible (en consultas posteriores), de tal manera que ninguna otra transacción podrá modificar los registros seleccionados durante la ejecución de la transacción completa.

11. Explique la diferencia entre un SELECT y un SELECT FOR UPDATE.

R: El SELECT FOR UPDATE es un tipo de consulta más estricta, puesto que genera un bloqueo exclusivo sobre un conjunto de registros no sólo para **garantizar la lectura repetible** si no también para **evitar un análisis inconsistente**. Básicamente, ninguna transacción podrá hacer SELECT (**de cualquier tipo**) sobre el mismo conjunto de registros y mucho menos podrá modificarlos.

12. Explique la diferencia entre un SELECT FOR UPDATE y un SELECT FOR SHARE

R: Durante un **SELECT FOR SHARE** es posible realizar la consulta simultánea de un mismo conjunto de registros evitando siempre la modificación de los mismos, mientras que en un **SELECT FOR UPDATE** esto no es posible, ya que el bloqueo sobre el conjunto de registros no permite la lectura de otras transacciones poniéndolas así en estado de espera.

F I N
