

Primera Prueba Solemne y Pauta de Corrección
Base de Datos

Profesor: **Álvaro Garrido**

TIEMPO: 90 Minutos

I.- Alternativas (20 pts)

1. **Un tipo de problema de concurrencia es la actualización temporal, esta consiste en:**
 - a. T1 y T2 transacciones, acceden al mismo dato y tienen sus operaciones intercaladas.
 - b. T1 y T2 transacciones, T1 utiliza una función de agregado sobre varios registros mientras otras transacciones actualizan dichos registros mientras T1 todavía no termina.
 - c. T1 y T2 transacciones, T1 lee un elemento "x" dos veces y T2 modifica un elemento entre las dos lecturas.
 - d. **T1 y T2 transacciones, T1 actualiza un dato "x" pero falla, y antes de que se restaure el valor original T2 tiene acceso al valor temporal.**
2. El aislamiento consiste en:
 - a. Asegurar que, una vez que la transacción se ha comprometido, las actualizaciones hechas por la transacción no se pierden incluso si hay un fallo del sistema.
 - b. Asegurar que si la base de datos es consistente inicialmente, la ejecución de la transacción (debido a la misma) deja la base de datos en un estado consistente.
 - c. **Asegurar la ejecución de las transacciones concurrentes, dando la impresión que ninguna otra transacción se ejecuta concurrentemente con otra.**
 - d. Asegurar que toda transacción se realiza en un 100% o simplemente no se realiza.
3. Uno de los problemas que se entrega en el protocolo de bloqueo de dos fases es:
 - a. Permite demasiada disponibilidad de los datos.
 - b. **Permite la generación de cuellos de botella.**
 - c. Permite la generación de bloqueos de datos.
 - d. Permite que la escritura de un dato "x" requiera de candados de todas las copias de "x".
4. Una granularidad mayor, como por ejemplo en las tablas, es costoso en términos de:
 - a. **Simultaneidad, debido a que restringe los accesos a las demás transacciones.**
 - b. Inserción de datos.
 - c. Bloqueo de datos, debido a una sobrecarga en el sistema.
 - d. Deadlock.
5. Una restricción de la vista es:
 - a. No se puede incluir la cláusula UNION.
 - b. **No se puede incluir la cláusula ORDER BY.**
 - c. No se puede incluir la cláusula ALTER VIEW.
 - d. Si se actualiza no conserva los permisos asignados.
6. El comando REVOKE, permite:
 - a. Dar acceso a una tabla.
 - b. Quitar solo el privilegio de GRANT
 - c. Quitar solo el privilegio de DENY
 - d. **Quitar el privilegio de GRANT y DENY.**
7. Uno de los procedimientos recomendados para el uso de vistas y procedimientos almacenados es:
 - a. Hacer las labores de administración del motor más fáciles.
 - b. **Simplificar la seguridad.**
 - c. Para no hacer tantos JOINS.
 - d. Para no recargar el motor de base de datos.
8. Es posible garantizar la integridad de los datos mediante la implementación de varios conceptos claves, estos son:
 - a. Uso solamente de procedimientos almacenados y vistas.
 - b. Uso de roles de base de datos, definición de reglas, uso de joins integrados.

- c. Normalización, definición de reglas de la empresa, integridad referencial, Validación.
 - d. Mediante creación de JOINS complejos.
9. La integridad del dominio, se refiere a que:
- a. La base de datos no debe contener valores de llaves ajenas sin concordancia.
 - b. Ningún componente de la llave primaria de una relación puede aceptar valores nulos.
 - c. La combinación de datos en la llave primaria.
 - d. Las llaves foráneas no deben aceptar NULL.
10. La integridad de datos declarativa, se refiere a:
- a. Los criterios se definen en la definición del objeto
 - b. Los criterios se definen en una secuencia de comandos
 - c. Los criterios se definen en la definición del objeto y a través de secuencia de datos.
 - d. Aplicación de criterios mediante procedimientos almacenados y triggers.

II.- SQL (15 pts)

Empleado (ID_Emp:integer, EmpNombre:varchar(50), EmpEdad:integer, EmpSueldo:float)
 Departamento(ID_Dep:integer, DepPresupuesto:float, ID_Administrador:integer)
 Trabajos(ID_Emp:integer, ID_Dep:integer, Tiempo:integer)

Nota: ID_Administrador se relaciona con ID_Emp

1. Listar el nombre de cada empleado cuyo sueldo excede el presupuesto de todos los departamentos en los cuales él o ella han trabajado. (5 pts)

```
SELECT E.EmpNombre
FROM Empleado E
WHERE E.EmpSueldo > ( SELECT D.DepPresupuesto
FROM Departamento D, Trabajos T
WHERE E.ID_Emp = T.ID_Emp AND
D.ID_Dep = T.ID_Dep)
```

2. Listar todos los ID de Administradores que han administrado solamente departamentos cuyo presupuesto sea mayor que 10.000.000 (5 pts)

```
SELECT distinct D.ID_administrador
FROM Departamento D
WHERE 10000000 < ( SELECT D2.DepPresupuesto
FROM Departamento D2
WHERE D2.ID_Administrador = D.ID_Administrador)
```

3. Listar todos los nombres de Administradores que han administrado departamentos con el presupuesto más alto. (5 pts)

```
Select E.EmpNombre
From Empleado E
WHERE E.ID_Emp IN (select D. ID_Administrador
From Departamento D
Where D.DepPresupuesto = (select MAX(D2.DepPresupuesto
From Departamento D2))
```

III.- SQL (25 pts)

Animal(ID_Animal, NombreAnimal, Categoria, PrecioLista)
 VentaAnimal(ID_Venta, ID_Animal, PrecioVenta, ID_cliente)
 Cliente(ID_cliente, Nombre, Apellido,Fono)
 Ventaltem(ID_item, ID_Venta, Cantidad, PrecioItem)

Nota: Ejemplo de Categoria -> "Gatos", "Perros"

4. Listar todos los clientes que compraron uno de los siguientes artículos (ítems) (1,39,29,34,40). **(7 ptos)**

```
Select cliente.nombre, cliente.apellido
From cliente INNER JOIN Venta ON cliente.ID_cliente = Venta.ID_cliente)
INNER JOIN Ventaltem ON Venta.ID_Venta = Ventaltem.ID_venta
Where (Ventaltem.ID_Item IN (1,39,29,34,40))
```

5. Listar todo los gatos que se vendieron por sobre su precio promedio. **(8 ptos)**
- ```
SELECT VentaAnimal.ID_Animal, Animal.Categoria, VentaAnimal.PrecioVenta
FROM Animal
INNER JOIN VentaAnimal ON Animal.ID_Animal=VentaAnimal.ID_Animal
WHERE ((Animal.Categoria='Gatos') AND (VentaAnimal.PrecioVenta >
(SELECT AVG(PrecioVenta)
FROM Animal
INNER JOIN VentaAnimal ON Animal.ID_Animal = VentaAnimal.ID_Animal
WHERE (Animal.Categoria='Gatos'))))
```

6. Listar todos los animales (nombre, precio venta y precio lista), cuyo precio de venta sea mayor o igual a la sumatoria del 10% del precio lista de todos los gatos vendidos. **(10 ptos)**

```
SELECT Animal.ID_Animal, Animal.Nombre, PrecioVenta, PrecioLista
FROM Animal
INNER JOIN VentaAnimal ON Animal.ID_Animal = VentaAnimal.ID_Animal
WHERE ((VentaAnimal.PrecioVenta >=
(SELECT SUM(0.10*Animal.PrecioLista)
FROM Animal
INNER JOIN VentaAnimal ON Animal.ID_Animal = VentaAnimal.ID_Animal
WHERE Animal.Categoria = 'Gatos'))
```