UNIVERSIDAD DEL VALLE DE GUATEMALA

CC3088 – Base de Datos 1

Sección 20

Bacilio Bolaños

Laboratorio 2

Diego Patzán - 23525

Ihan Marroquin - 23108

GUATEMALA, 20 de mayo de 2025

1. **¿Qué ventajas encontraron al encapsular lógica en funciones en lugar de repetir consultas SQL?**
    a. Al poner la lógica en funciones evitamos duplicar fragmentos de SQL en distintas partes del mismo SQL, esto facilita el mantenimiento (si cambia la regla de negocio, sólo ajustamos la función) y reduce errores humanos.

2. **¿Qué criterios usaron para decidir cuándo implementar una función y cuándo una vista?**
    a. Usamos funciones cuando necesitábamos lógica dinámica o parámetros de entrada.
    b. Usamos vistas para consultas estáticas o agregaciones frecuentes que se exponen como tablas virtuales.

3. **¿Qué limitaciones encontraron al trabajar con procedimientos almacenados en comparación con funciones?**
    a. Los procedimientos no devuelven valores directamente, sólo pueden usar variables OUT o notificaciones; son menos flexibles para integración con SELECT, ademas, requieren una llamada explícita y no encajan en subconsultas, a diferencia de las funciones.

4. **¿Creen que el trigger que implementaron garantiza la integridad de los datos en todos los escenarios posibles? Justifiquen su respuesta.**
    a. El trigger BEFORE en reserva evita reservas que desborden el cupo actual, pero no controla todas las ramas: si hay cambios de estado fuera de INSERT/UPDATE directo, podría violarse la capacidad, pero para cubrir esos casos, haríamos un trigger también en actualizaciones de estado o una restricción a nivel de aplicación.

5. **¿Cómo adaptarían su solución para que escale en una base de datos con millones de registros?**
    a. Crear índices sobre columnas de filtro frecuentes.
    b. Particionar tablas muy grandes.
    c. Usar materialized views refrescables para agregados pesados, y refrescarlos en lotes.

6. **¿Qué escenarios podrían romper su lógica actual si no existiera el trigger?**
    a. Sin el trigger de capacidad, un usuario podría insertar reservas en paralelo que sumen más de la capacidad, o modificar directamente el estado en la tabla y sobredimensionar el cupo.

7. **¿Qué dificultades enfrentaron al definir funciones que devuelven conjuntos de resultados?**
    a. Definir firmas de funciones con RETURNS TABLE exige que cada columna devuelta coincida en tipo y orden, y concatenar cadenas obliga a usar TEXT en lugar de VARCHAR, además, agrupar y filtrar en la misma consulta obligó a pensar bien el GROUP BY para evitar errores.

8. **¿Consideran que su diseño sería compatible con una arquitectura de microservicios? ¿Por qué sí o por qué no?**
    a. Nuestro diseño encapsula la lógica en la base de datos, lo que facilita a microservicios ligeros invocar funciones y vistas con un contrato claro.

9. **¿Cómo reutilizarían las vistas que definieron en reportes o en otros sistemas?**
    a. Las vistas pueden consumirse directamente desde herramientas de reporting o exportarse por APIs REST de un microservicio, por ejemplo, v_plazas_disponibles sirve para mostrar en tiempo real cuántos cupos quedan en el portal web sin reescribir la lógica.

10. **¿Qué aprendieron sobre la separación entre la lógica de negocio y la lógica de persistencia al hacer este laboratorio?**
    a. Este laboratorio mostró la importancia de aislar la lógica compleja en funciones/procedimientos, mientras que las vistas y el DDL se centran en la estructura de datos.