Reflexión Individual - Diego Patzan - 23525

1. ¿Por qué eligieron este sistema de gestión de bases de datos (DBMS)? ¿Qué ventajas y desventajas

Optamos por SQL Server primero porque es de gratis y su estructura robusta garantiza la integridad y reduce redundancias en los datos, algo crítico para un sistema de flujos de aprobación, comparado con opciones como SQL server o MySQL, Postgres ofrece mejor soporte para transacciones complejas y reporting integrado, aunque su licencia puede ser costosa en proyectos grandes.

2. ¿Qué estándares o criterios usaron para diseñar su base de datos?

Seguimos la normalización hasta la tercera forma normal (3FN) para eliminar redundancias, separamos GrupoAprobacion de Usuario para no repetir datos de grupos en cada usuario, también priorizamos la auditoría: tablas como Solicitud o FlujoActivo registran quién creó el registro y cuándo (creado_por, fecha_creacion).

3. Entidades más importantes

- Solicitud: El núcleo del sistema, donde se inicia cada proceso.
- FlujoActivo y PasoSolicitud: Gestionan la ejecución en tiempo real de los flujos, incluyendo bifurcaciones y uniones.
- Usuario y GrupoAprobacion: Definen quiénes interactúan con el sistema y cómo toman decisiones.

4. Normalización y problemas evitados

Aplicamos normalización iterativa: en lugar de guardar los nombres de los grupos de aprobación en cada paso, usamos GrupoAprobacion_id, esto evitó inconsistencias al editar un nombre, también separamos Inputs en su propia tabla para reutilizarlos en múltiples pasos o solicitudes, eliminando duplicación de lógica.

5. Restricciones e integridad de datos

- PK autoincrementales: Como id_flujo en FlujoAprobacion, evitando IDs manuales.

- FK estrictas: Por ejemplo, departamento_id en Usuario garantiza que no existan usuarios en departamentos fantasmas.
- Valores por defecto: fecha_creacion se llena automáticamente, asegurando trazabilidad.
- CHECK en columnas: Como tipo_flujo solo permite 'normal', 'bifurcacion' o 'union', evitando pasos inválidos.

6. Manejo de cambios en la estructura

Usamos GitHub para versionar scripts (DDL) y migraciones, esto nos permitió llevar mejor control sobre las versiones de la base de datos

7. Datos de prueba y validación

- Usuarios con roles contradictorios: Un "solicitante" en el grupo de aprobadores, para probar restricciones.
- Flujos con bifurcaciones extremas: Como un paso de unión sin pasos previos, para validar que el sistema detecte errores.
- Solicitudes en estados límite: Aprobadas antes de iniciar, para forzar validaciones de fecha (fecha_inicio < fecha_fin).

8. Mi contribución y organización del equipo

Yo con otro compañero nos enfocamos principalmente en implementar los cambios

sugeridos durante la primera presentación de la base de datos

9. Equidad y mejoras posibles

Considero que para esta entrega el nivel de trabajo no fue igual para todos, pudimos mejorar,

pero al final considero que el aporte de cada persona fue clave para lograr obtener el trabajo que cumpliría con los requisitos de la entrega.