Nombre: Anthony Sagbay

Materia: herramientas informaticas para el despliegue de diagramas

Curso: 3ero de Desarrollo de Software

Tema: Consultas, Operaciones Lógicas y Transacciones en un Sistema de Gestión Universitaria

Fecha: 14 de enero de 2026

Docente: Jostin Vacacela

Cuenca

https://github.com/ANTHONYSAGBAY/sgu-sagbay.git

 # Análisis de Principios ACID - Sistema de Gestión Universitaria

Este documento explica cómo se aplican los principios ACID en el proceso de matriculación de estudiantes.

1. Atomicidad (Atomicity)

La ****Atomicidad**** garantiza que el proceso de matriculación sea una operación de "todo o nada". En nuestra implementación, utilizamos `$transaction` de Prisma.

- ****Caso de éxito:**** Se verifica que el estudiante esté activo, que haya cupo, se crea el registro en `StudentSubject` y se descuenta el cupo en `Subject`. Todas estas operaciones se confirman (commit) juntas.

- ****Caso de error:**** Si el estudiante no está activo o si la actualización del cupo falla (por ejemplo, por una restricción de base de datos), toda la operación se revierte (rollback). No queda un estudiante matriculado sin que se haya descontado el cupo, ni viceversa.

2. Consistencia (Consistency)

La ****Consistencia**** asegura que la base de datos pase de un estado válido a otro estado válido.

- En nuestro sistema, la consistencia se garantiza mediante reglas de negocio validadas dentro de la transacción:

  - No se permite matricular a un estudiante en una asignatura que no tiene cupos (`capacity > 0`).

  - Solo estudiantes con estado `active` pueden matricularse.

- Además, las restricciones de integridad referencial (llaves foráneas) en la base de datos aseguran que no se puedan crear matrículas para estudiantes o asignaturas inexistentes.

3. Aislamiento (Isolation)

El ****Aislamiento**** determina cómo los cambios realizados por una operación se vuelven visibles para otras.

- Cuando varios estudiantes intentan matricularse en la misma asignatura simultáneamente, Prisma (y PostgreSQL) manejan el aislamiento mediante niveles de transacción.

- Al actualizar el campo `capacity` dentro de una transacción, se bloquea la fila de la asignatura correspondiente. Esto evita el fenómeno de "lectura sucia" o "actualización perdida",

asegurando que si dos estudiantes intentan tomar el último cupo, uno de ellos fallará debido a que el primero ya habrá actualizado la capacidad a 0.

4. Durabilidad (Durability)

La ****Durabilidad**** garantiza que una vez que la transacción se ha completado con éxito, los cambios persistan incluso en caso de un fallo del sistema (como un corte de energía).

- Prisma, al trabajar sobre PostgreSQL, confía en el registro de transacciones (Write-Ahead Logging - WAL) del motor de base de datos. Una vez que el servidor responde que la matrícula fue exitosa, los datos están grabados en disco de forma permanente.

- Esto es crítico en un sistema universitario para evitar que un estudiante pierda su cupo o su registro de notas debido a un fallo técnico tras haber completado su trámite.