Nombre: Jorge Manuel Oyoqui Aguilera
Matrícula: A01711783
Fecha: 18/05/2025
Materia: Construcción de Software y Toma de Decisiones
Grupo: 501
Profesores: Enrique Alfonso Calderón Balderas,
Denisse L. Maldonado Flores, Alejandro Fernández Vilchis

Tecnológico de Monterrey

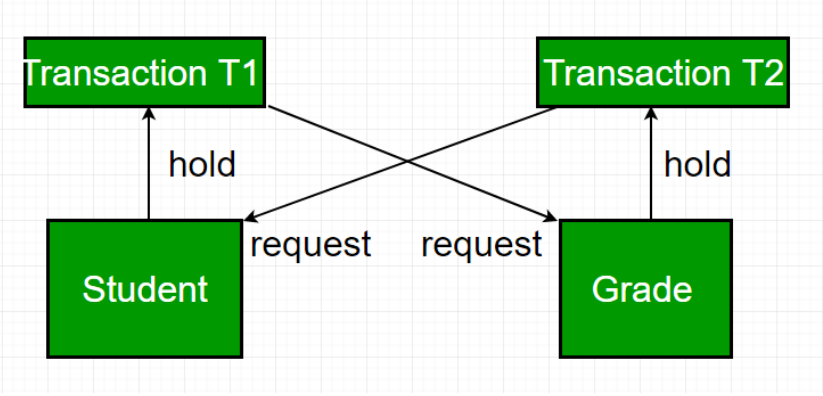
Laboratorio 25. Transacciones

Cuando hablamos de bases de datos, un **deadlock** es una situación en la que se produce un bloqueo cuando dos o más transacciones no pueden continuar porque cada una espera a que la otra libere los bloqueos de recursos. Esta situación crea un ciclo de dependencias donde ninguna transacción puede continuar, lo que provoca la paralización del sistema. Igualmente, los bloqueos pueden afectar gravemente el rendimiento y la fiabilidad de un SGBD. (Geeks for geeks, 2024).

Las condiciones que se deben presentar para que se dé un deadlock son:

- Que se las transacciones se excluyan mutuamente, donde un recurso sólo puede ser usando por una transacción a la vez.
- Que haya una retención y espera donde una transacción mantiene recursos en lo que espera otros más.
- No puede haber expropiación, o sea, no se les puede quitar a las transacciones sus recursos.
- Hay una cadena circular de transacciones donde cada una espera un recurso que la siguiente tiene.
- Las transacciones se bloquean indefinidamente, esperando a que los recursos que necesitan estén disponibles.
- Que haya datos inconsistentes también provoca que las transacciones no se puedan provocar, generando así bloqueos.

Un ejemplo de cómo ocurre un deadlock sería el siguiente que obtuve de Geeks for Geeks:



En este caso, la transacción T1 mantiene un bloqueo en algunas filas de la tabla Estudiantes y necesita actualizar algunas filas de la tabla Calificaciones. Simultáneamente, la transacción T2 mantiene bloqueos en esas mismas filas (que T1 necesita actualizar) de la tabla Calificaciones, pero necesita actualizar las filas de la tabla Estudiantes que contiene la transacción T1. (Geeks for geeks, 2024).

Ahora, es mejor prevenir que sucedan estos casos en la base de datos a tener que reiniciarla o eliminarla. En el caso de las bases de datos grandes, lo que más conviene es usar el **esquema de prevención de interbloqueos**, donde el mecanismo propone dos esquemas: o utiliza el esquema **espera-muerte**, donde si una transacción requiere un recurso bloqueado por otra transacción entonces espera hasta que el recurso esté disponible pero cancela la transacción posterior; y el **esquema de espera de interrupción**, donde si la transacción anterior solicita un recurso que posee una transacción posterior , entonces el esquema obliga a la anterior a cancelarse u a liberar el recurso.

Unos ejemplos de las aplicaciones a las que se les puede dar a estos esquemas de prevención son:

1. **Transacciones retrasadas:** Los interbloqueos pueden causar retrasos en las transacciones, ya que los recursos necesarios están retenidos por otras transacciones, generando así tiempos de respuesta más lentos y tiempos de espera más largos para los usuarios. (Geeks for geeks, 2024).
2. **Transacciones perdidas:** pues en algunos casos, los bloqueos pueden provocar que se pierdan o cancelen transacciones, generando inconsistencias en los datos u otros problemas. (Geeks for geeks, 2024).
3. **Concurrencia reducida:** Los interbloqueos pueden reducir el nivel de concurrencia en el sistema, ya que las transacciones se bloquean a la espera de que los recursos estén disponibles, provocando un procesamiento de transacciones más lento y una reducción del rendimiento general. (Geeks for geeks, 2024).
4. **Mayor uso de recursos:** Los interbloqueos pueden provocar un mayor uso de recursos, ya que las transacciones bloqueadas a la espera de disponibilidad de recursos continúan consumiendo recursos del sistema, provocando una degradación del rendimiento y una mayor contención de recursos. (Geeks for geeks, 2024).
5. **Menor satisfacción del usuario:** Los bloqueos pueden generar una percepción de bajo rendimiento del sistema y reducir la satisfacción del usuario con la aplicación, impactando negativamente la adopción y retención de usuarios. (Geeks for geeks, 2024).

Referencias Bibliográficas

Geeks for Geeks (2024). “Deadlock in DBMS”. Recuerado de: https://www.geeksforgeeks.org/deadlock-in-dbms/

Geeks for Geeks (2025). “Introduction to TimeStamp and Deadlock Prevention Schemes in DBMS”. Recuperado de: https://www.geeksforgeeks.org/introduction-to-timestamp-and-deadlock-prevention-schemes-in-dbms/