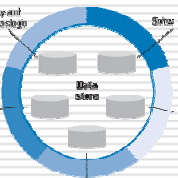
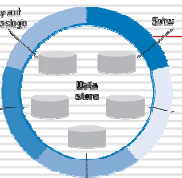
Base de Datos II

2da Parte



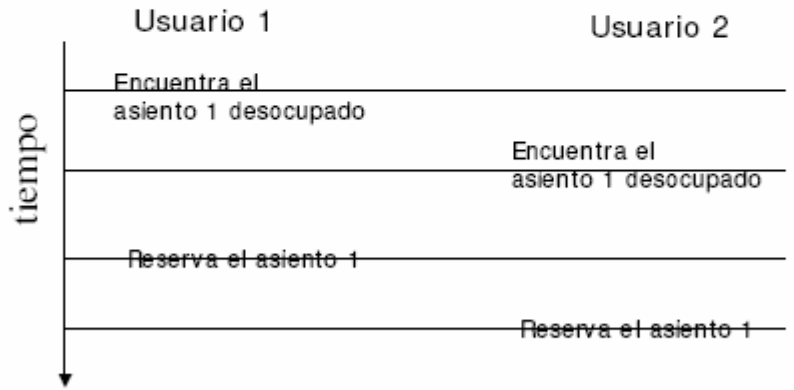
Transacciones en SQL

- Una transacción es un conjunto de operaciones que deben ser ejecutadas con criterio de atomicidad.
- Solución 1: Serialización.
  - Consiste en colocar en una cola todas las operaciones que se desean ejecutar en la base de datos.
  - Esto es imposible de pensar para grandes aplicaciones.
- Esto nos crea un nuevo problema ... La concurrencia.



Transacciones

- Pensemos en un sistema de reserva de asientos para un avion.

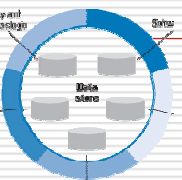


Lo más probable es que esto termine con un pasajero muy molesto.

Propiedad ACID

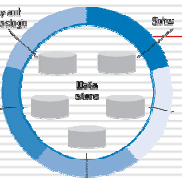
- La propiedad ACIDa es una carácterística de un DBMS para poder compartir datos en forma segura.

- **A : Atomicity**
- **C: Consistency**
- **I : Isolation**
- **D: Durability**



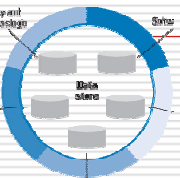
Transacciones: Atomicidad

- Sabiendo que la serialización es impensable, pensemos que pueden ocurrir los siguientes problemas:
  - 2 o más operaciones sean ejecutadas sobre la BD.
  - Una operación deje la BD en un estado inconsistente si un problema de software o hardware ocurre durante la ejecución.



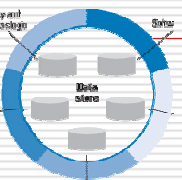
Atomicidad

- “Todo o nada” es la mejor forma de explicar atomicidad.
- Cuando ocurre una actualización en la base de datos, garantizamos que se hace en forma completa o no se hace.
  - COMMIT
  - ROLLBACK
- a transaction is a unit of operation either all the transaction's actions are completed or none are
- atomicity is maintained in the presence of deadlocks
- atomicity is maintained in the presence of database software failures
- atomicity is maintained in the presence of application software failures
- atomicity is maintained in the presence of CPU failures
- atomicity is maintained in the presence of disk failures
- atomicity can be turned off at the system level
- atomicity can be turned off at the session level



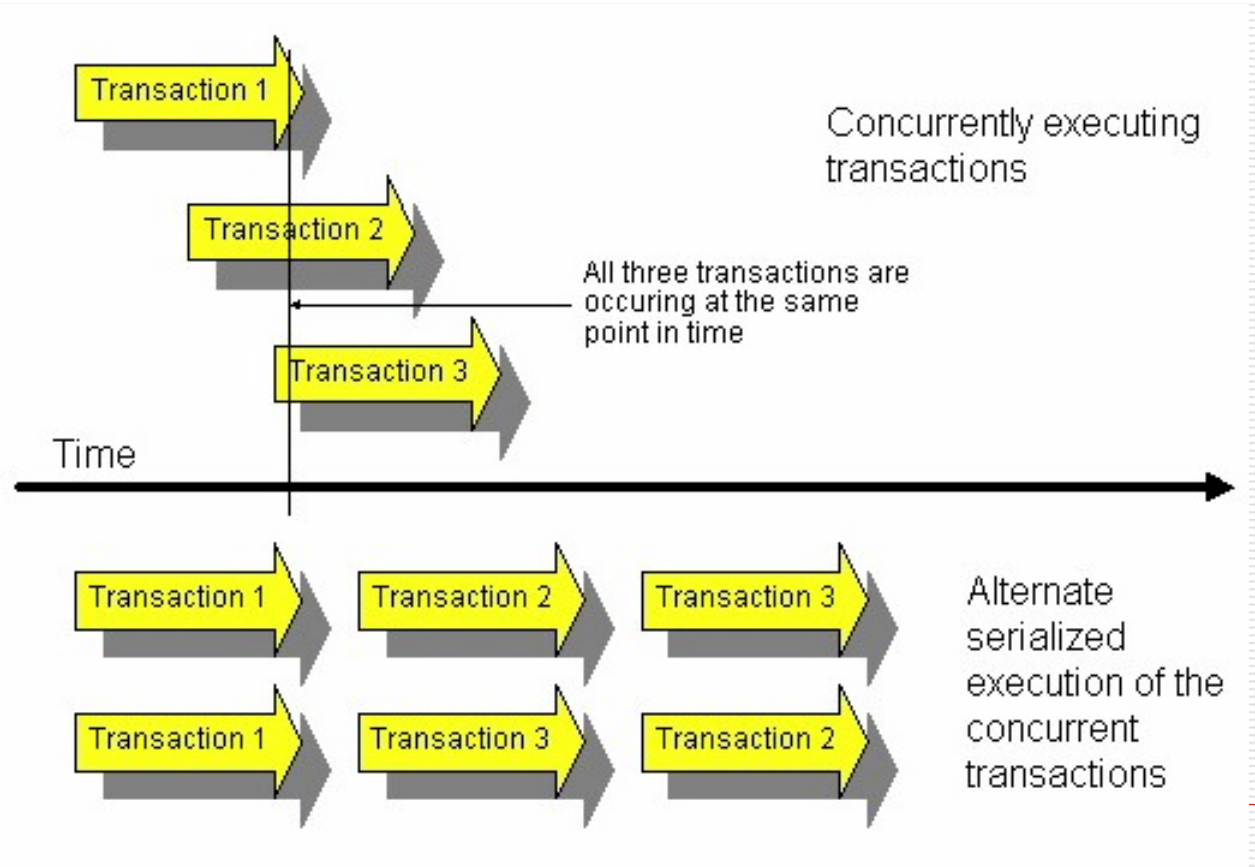
Consistency

- Garantiza que los cambios hechos en un instante son consistentes con los cambios hechos a otros datos en el mismo instante.



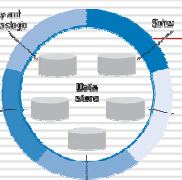
Isolation

- Necesaria cuando existe la concurrencia.



Isolation

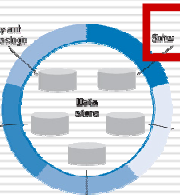
- Degrees of isolation:
  - degree 0 - a transaction does not overwrite data updated by another user or process ("dirty data") of other transactions
  - degree 1 - degree 0 plus a transaction does not commit any writes until it completes all its writes (until the end of transaction)
  - degree 2 - degree 1 plus a transaction does not read dirty data from other transactions
  - degree 3 - degree 2 plus other transactions do not dirty data read by a transaction before the transaction commits



Durability

- Los cambios que han sido “COMMIT” no se pueden perder!

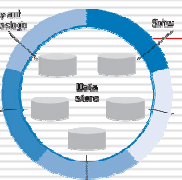
- recovery to the most recent successful commit after a database software failure
- recovery to the most recent successful commit after an application software failure
- recovery to the most recent successful commit after a CPU failure
- recovery to the most recent successful backup after a disk failure



Transacciones en Java

- Connection con;
- ...
- con.commit();
- con.rollback();

- Connection.setAutoCommit(boolean)
- Connection.getAutoCommit()



Propuestos

- Escribir programa en Java que usando al menos 2 threads ejecute operaciones sobre la base de datos.
- Elabore un plan para demostrar el funcionamiento de los deadlocks.
- Escriba un programa en Java que demuestre cómo opera el commit y el rollback.
- Elabore un plan para demostrar que PostgreSQL cumple con la propiedad ACID.