WER150018

WIA2001 Database

Tutorial 8

- Deadlock prevention
- Deadlock detection
- Deadlock avoidance

Q transaction isolation is when data used during transaction cannot be used by second transaction until the first is completed.

It is important because in a multiuser database, several users can access and update database at the same time.

A transaction is a logical unit of work that must be entirely completed of aborted; no intermediate states are accepted. In other words, a transaction, composed of several database requests, is treated by the DBMS as a unit of work in which all transaction steps must be fully completed if the transaction is to be accepted by the DBMS.

Example:

A credit card requires a minimum of three database operations: 1) an invoice is created for the sold product.
2) The product's inventory quantity on hand is reduced
3) The customer accounts payable balance is increased by the amount listed on the invoice

Leaving one procedure operation and the database will be left in an inconsistent state. Unless all parts (1, 2, 3) are completed, the entire sales transaction is canceled.

4. Error in inputting the request/operation does not affect the database as long the request is valid, database needed is exist and database is still in consistent state.

Example: entering the wrong code of product to remove a product. the DBMS will execute the transaction, the database is technically consistent, but in a real-word inconsistent state, because the wrong product is updated.

5. Atomicity: -all operations of a transaction must be completed
-if not, the transaction is aborted

Consistency: permanence of database's consistent state

Isolation: data used during transaction cannot be used by second transaction until the first is completed.

Durability: ensures that once transactions are committed, they cannot be undone or lost

Serializability: ensures that the schedule for the concurrent execution of several transactions should yield consistent results.