Database Creation and Management

Concurrency control

Sasmita Kumari Nayak
CSE Department

CONCURRENCY CONTROL

- The process of managing simultaneous operations on the database without having them interferes with one another.
- Many businesses such as banks or airlines have many users accessing the databases concurrently.
- Multiple users accessing the database simultaneously cannot be permitted to interfere with one another.
- The objective of concurrency control is to maximize transaction throughput while preventing interference among multiple users.

Cont...

- Transaction throughput is the number of transactions processed per unit of time, is a measure of the amount of work performed by the DBMS.
- Concurrent transactions manipulating common data may cause interference problems for example on the seat-remaining column of a flight table.
- Many users may want to book on the number of seats remaining in a particular flight. It is crucial that the DBMS control concurrent access to the seat-remaining column of a flight table while users are booking simultaneously.

Advantages

- Advantages of Concurrent Execution of Transaction
  - Improved Throughput: Average no. of transactions completed in a given time (CPU Waiting).
  - Reduced Waiting Time: Short transaction can complete quickly.

Interference Problems

- The three examples of interference problems that can be caused by concurrent access are the:
  - Lost update problem
  - Uncommitted dependency problem
  - Inconsistent analysis problem

Lost Update Problem

- This type of problem occurs when one user's update overwrites another user's update.
- This is a serious problem as updates to the database are lost forever.
- We show an example of this problem by using a bank account transaction.
- Consider the following transactions shown in figure 7.4, TransactionA and TransactionB in which TransactionA is executing concurrently with TransactionB. TransactionA is withdrawing Rs.100 from an account with BalanceX (BX), initially RM500,

Time	TransactionA	TransactionB	BX
T_1		Begin_transaction	500
T_2	Begin_transaction	Read (BX)	500
T_3	Read (BX)	BX=BX + 100	500
T_4	BX=BX - 100	Write (BX)	600
T_5	Write (BX)	Commit	400
T_6	Commit		400

Cont...

and TransactionB is depositing Rs.100 into the same account.

- TransactionA and TransactionB both read the balance as Rs.500. TransactionB increases the Balance to Rs.600 and writes it to the database. TransactionA reduces the balance by Rs.100 and writes the value of Rs.400 as the balance. TransactionA has therefore overwritten the value of TransactionB, it has lost Rs.100 in the process. The correct value of BX should therefore be Rs.600.

Uncommitted dependency problem

- An uncommitted dependency problem occurs when one transaction reads data written by another transaction before the other transaction commits.
- This type of interference problem is also known as the dirty read because it is caused by the transaction reading dirty (uncommitted) data.
- From the example shown in figure 7.5, TransactionB updates BalanceX(BX) to RM600, but aborts the transaction. This could be due to an error or maybe

Time	TransactionA	TransactionB	Balance
T_1		Begin_transaction	500
T_2		Read (BX)	500
T_3		BX=BX + 100	500
T_4	Begin_transaction	Write (BX)	600
T_5	Read (BX)	.....	600
T_6	BX=BX -100	Rollback	500
T_7	Write (BX)		500
T_8	Commit		500

- Because it was updating the wrong account. It aborts the transaction by issuing a ROLLBACK which causes the value in BX to revert back to its original value of Rs.500.
- But TransactionA has incorrectly read the value in BX as Rs.600 at time T_5. It then decrement it by RM100 giving BX an incorrect value of Rs.500 and goes on to commit it. As you would have figured the correct value in BX is Rs.400.

Inconsistent analysis problem

- This problem occurs when a transaction calculating a summary function reads some values before another transaction changes the values but reads other values after another transaction changes the values.
- In the example shown in figure 7.6, TransactionA is updating the balances of BalanceX(BX) and BalanceY(BY). Transaction B is summarizing BalanceX (BX) and BalanceY (BY).

Time	TransactionA	TransactionB	BX	BY	SUM
T_1	Begin_transaction		500	500	0
T_2	Read (BX)		500	500	0
T_3	BX=BX-100		500	500	0
T_4	Write (BX)	Begin_transaction	400	500	0
T_5		Read (BX)	400	500	0
T_6		Sum= Sum + BX	400	500	400
T_7		Read (BY)	400	500	500
T_8		Sum = Sum + BY	400	500	900
T_9	Read (BY)	Write Sum	400	500	900
T_{10}	BY=BY+100	Commit	400	500	900
T_{11}	Write BY		400	600	900
T_{12}	Commit		400	600	900

Figure 7.6: The Inconsistent analysis Problem

- Transaction B has read the value of BX correctly which is Rs.400 but it reads the value of BY as Rs.500 before TransactionA has incremented it to RM100.
- This has resulted in an incorrect value in Sum at time T_9. TransactionB should therefore wait for TransactionA to update BY to read the new value of Rs.600 in BY. The correct value written to the database should be Rs.1000.