



Manajemen Transaksi

Sistem Basis Data

Gentisya Tri Mardiani, S.Kom

Schedule (Penjadwalan)



- Urutan instruksi yang menspesifikasikan urutan kronologi instruksi dari transaksi yang dieksekusi.
- Sebuah jadwal harus menjaga urutan instruksi yang muncul di setiap transaksi.

Schedule 1



- A= \$100, B= \$100
- T1 transfer \$50 dari A ke B
- T2 transfer 10% dari A ke B
- Schedule serial dimana T1 diikuti T2:

T1	T2
Read (A) $A \leftarrow A - 50$ Write(A) Read(B) $B \leftarrow B + 50$ Write(B)	
	Read (A) $Temp \leftarrow 0.1 * A$ $A \leftarrow A - temp$ Write(A) Read(B) $B \leftarrow B + temp$ Write(B)

Schedule 2



- Penjadwalan tidak serial, tetapi ekuivalen dengan schedule 1

T1	T2
Read(A) $A \leftarrow A - 50$ Write(A)	
	Read (A) $Temp \leftarrow 0.1 * A$ $A \leftarrow A - temp$ Write(A)
Read(B) $B \leftarrow B + 50$ Write(B)	
	Read(B) $B \leftarrow B + temp$ Write(B)

Schedule 3



- Penjadwalan tidak serial, dan hasilnya tidak konsisten.

T1	T2
Read(A) $A \leftarrow A - 50$	
	Read (A) $Temp \leftarrow 0.1 * A$ $A \leftarrow A - temp$ Write(A) Read(B)
Write(A) Read(B) $B \leftarrow B + 50$ Write(B)	
	$B \leftarrow B + temp$ Write(B)

Serializability



- Setiap transaksi harus tetap menjaga konsistensi database.
- Sistem basis data harus dapat mengontrol eksekusi konkurensi dari suatu transaksi untuk memastikan database tetap terjaga konsistensinya.
- Penjadwalan serializable merupakan penjadwalan secara serial.

Serializability



- Setiap transaksi dengan sepenuhnya terisolasi sedemikian rupa sehingga transaksi bertindak seolah-olah mereka telah mengeksekusi berturutan, satu demi satu; berturut-turut.
- Dalam mencapai hal ini, DBMS akan secara khusus mengunci setiap baris yang dibaca, maka sesi lain tidak boleh memodifikasi data itu sampai transaksi telah selesai. Kunci dilepaskan ketika transaksi *commit* atau *rollback*.

Teknik Pengontrolan Konkurensi



- Metode locking
- Metode timestamp
- Metode locking dan timestamp dapat menyebabkan penundaan transaksi jika terjadi konflik dengan transaksi lainnya pada waktu yang sama.

Locking



- Metode locking merupakan pendekatan yang paling banyak digunakan untuk memastikan serializability.
- Apabila suatu transaksi mengakses suatu data maka suatu lock (kunci) dapat mencegah pengaksesan oleh transaksi lain.

Locking



- Secara umum, transaksi harus menegaskan penguncian (*lock*) *shared (read)* atau *exclusive (write)* terhadap data item sebelum pembacaan (*read*) atau penulisan (*write*).
- Aturan dasar penguncian (locking):
- ***Shared Lock***, maka transaksi dapat melakukan pembacaan tetapi tidak melakukan perubahan.
- ***Exclusive Lock***, maka transaksi dapat melakukan pembacaan dan perubahan terhadap data item tersebut.

Locking



Cara kerja dari kunci :

- Kita asumsikan terdapat 2 (dua) macam kunci :
- **Kunci X** (kunci eksklusif) dan **kunci S** (kunci yang digunakan bersama-sama/ shared)
- Jika **transaksi A** menggunakan **kunci X** pada record R, maka permintaan dari **transaksi B harus menunggu** sampai nanti transaksi A melepaskan kunci
- Jika **transaksi A** menggunakan **kunci S** pada record R, maka :
 - Bila **transaksi B** ingin menggunakan **kunci X**, maka B harus **menunggu** sampai A melepaskan kunci tersebut.
 - Bila **transaksi B** ingin menggunakan **kunci S**, maka B bisa **menggunakan kunci S bersama A**

Locking



- Kunci X dan kunci S akan dilepaskan pada saat ***Synchpoint*** (*synchronization point*).
- Bila synchpoint ditetapkan maka:
 - semua modifikasi program menjalankan operasi COMMIT atau ROLLBACK
 - semua kunci dari record dilepaskan

Matriks Locking



A \ B	X	S	-
X	N	N	Y
S	N	Y	Y
-	Y	Y	Y

- Untuk menjamin serializability, membutuhkan protokol tambahan mengenai posisi dari operasi penguncian dan pelepasan kunci dalam setiap transaksi.

Two-Phase Locking (2PL)



- Suatu transaksi menggunakan protokol 2PL jika seluruh operasi penguncian (locking) mendahului operasi pelepasan kunci (unlock) dalam transaksi.
- Terdapat dua fase untuk transaksi, yaitu :
- *Growing phase* – mendapatkan seluruh kunci tetapi tidak dapat melepaskan kunci.
- *Shrinking phase* – melepaskan kunci tetapi tidak mendapatkan kunci baru.

Two-Phase Locking (2PL)



- Intinya, suatu transaksi jangan pernah melepaskan kunci sebelum operasi selesai, dengan aturan:
- Satu transaksi harus meminta/ menetapkan sebuah kunci sebelum melaksanakan operasi pada transaksi tersebut. Kunci yang diminta dapat berupa *write lock (exclusive)* maupun *read lock (shared)* , sesuai kebutuhan.
- Sekali transaksi melepaskan kunci, maka transaksi tersebut tidak dapat meminta kunci yang baru.

Penyelesaian masalah dengan Locking



- Lost update problem

Time	T_1	T_2	bal_x
t_1		begin_transaction	100
t_2	begin_transaction	write_lock(bal_x)	100
t_3	write_lock(bal_x)	read(bal_x)	100
t_4	WAIT	$bal_x = bal_x + 100$	100
t_5	WAIT	write(bal_x)	200
t_6	WAIT	commit/unlock(bal_x)	200
t_7	read(bal_x)		200
t_8	$bal_x = bal_x - 10$		200
t_9	write(bal_x)		190
t_{10}	commit/unlock(bal_x)		190

Penyelesaian masalah dengan Locking



- Uncommitted Dependency Problem

Time	T_3	T_4	bal_x
t_1		begin_transaction	100
t_2		write_lock(bal_x)	100
t_3		read(bal_x)	100
t_4	begin_transaction	$bal_x = bal_x + 100$	100
t_5	write_lock(bal_x)	write(bal_x)	200
t_6	WAIT	rollback/unlock(bal_x)	100
t_7	read(bal_x)		100
t_8	$bal_x = bal_x - 10$		100
t_9	write(bal_x)		90
t_{10}	commit/unlock(bal_x)		90

Penyelesaian masalah dengan Locking



- Tugas!
- Inconsistent Analysis Problem

Nilai 1 = 40

Nilai 2 = 50

Nilai 3 = 30

Transaksi A menjumlahkan
nilai 1, 2 dan 3

Transaksi B nilai3 dikurangi 10
dan nilai1 ditambah 10

Transaksi A	Waktu	Transaksi B
-	↓	-
Baca nilai 1(40) juml = 40	t1	-
-	↓	-
Baca nilai 2 (50) juml = 90	t2	-
-	↓	-
-	t3	Baca nilai 3 (30)
-	↓	-
-	t4	Modifikasi nilai 3 30 → 20
-	↓	-
-	t5	Baca nilai 1 (40)
-	↓	-
-	t6	Modifikasi nilai 1 40 → 50
-	↓	-
-	t7	Commit
-	↓	-
Baca nilai 3 (20) juml = 110 (bukan 120)	t8	-
-	↓	-

Deadlock



- Deadlock merupakan kebuntuan (impasse) yang mungkin dihasilkan ketika dua atau lebih transaksi saling menunggu kunci yang disimpan oleh transaksi lain agar dilepaskan.

Deadlock



Time	T ₁₇	T ₁₈
t ₁	begin_transaction	
t ₂	write_lock(bal_x)	begin_transaction
t ₃	read(bal_x)	write_lock(bal_y)
t ₄	bal_x = bal_x - 10	read(bal_y)
t ₅	write(bal_x)	bal_y = bal_y + 100
t ₆	write_lock(bal_y)	write(bal_y)
t ₇	WAIT	write_lock(bal_x)
t ₈	WAIT	WAIT
t ₉	WAIT	WAIT
t ₁₀	:	WAIT
t ₁₁	:	:



- Teknik untuk mengatasi deadlock...