STMIK
Amik Riau

MATAKULIAH
BASIS DATA TERDISTRIBUSI

Rahmaddeni, M.Kom
NIDN : 1007128301

Pertemuan 5 – Kontrol Transaksi Konkuren

STMIK
Amik Riau

#01

TRANSAKSI

DEFENISI TRANSAKSI

- Dalam istilah database, transaksi adalah aksi membaca dari atau menulis ke database.
- Transaksi dapat mencakup:
  - Pernyataan **SELECT** untuk menampilkan isi tabel
  - Pernyataan **UPDATE** untuk mengubah nilai atribut dalam tabel
  - Pernyataan **INSERT** untuk menambah baris ke 1 atau lebih tabel
  - Kombinasi dari pernyataan **SELECT, UPDATE, & INSERT**
- Transaksi database biasanya dibentuk oleh 2 atau lebih permintaan database berupa pernyataan SQL dalam program aplikasi.
- Jika salah satu permintaan dari transaksi gagal dieksekusi maka database akan berada dalam kondisi tidak konsisten & tidak dapat digunakan untuk transaksi berikutnya.

PROPERTI TRANSAKSI

- **Atomicity**: semua operasi (permintaan SQL) dari transaksi harus diselesaikan; jika tidak maka transaksi dibatalkan.
- **Consistency**: ketika transaksi diselesaikan, database harus dalam kondisi konsisten; jika ada bagian transaksi yang melanggar batasan integritas maka seluruh transaksi dibatalkan.
- **Isolation**: data yang digunakan selama eksekusi transaksi tidak dapat digunakan oleh transaksi kedua sampai transaksi pertama selesai. Berguna dalam **database multiuser** karena beberapa user dapat mengakses & mengupdate database pada waktu yang sama.
- **Durability**: memastikan bahwa ketika perubahan transaksi dilakukan & dicommit, maka tidak dapat dibatalkan atau dihilangkan, bahkan ketika terjadi kegagalan sistem.
- **Serializability**: memastikan bahwa jadwal untuk eksekusi konkurensi dari transaksi menghasilkan hasil yang konsisten. Penting dalam **database multiuser** & terdistribusi dimana terdapat beberapa transaksi yang dieksekusi sekaligus.

MANAJEMEN TRANSAKSI DENGAN SQL

STMIK Amik Riau

- ANSI (The American National Standards Institute) mendefinisikan standar yang mengawal transaksi database SQL.
- Standar tersebut menyebutkan ketika serangkaian transaksi diberikan oleh user atau program aplikasi, maka urutan tersebut harus dieksekusi terus sampai salah satu dari 4 kejadian berikut ini muncul:
  - Adanya pernyataan **COMMIT** yang mengakhiri transaksi SQL.
  - Adanya pernyataan **ROLLBACK** yang membatalkan semua perubahan & database dibalikkan ke kondisi konsisten sebelumnya.
  - **Akhir dari program tercapai** dimana semua perubahan tersimpan dalam database (sama dengan COMMIT).
  - **Program diberhentikan** dimana semua perubahan dibatalkan & database dibalikkan ke kondisi konsisten sebelumnya (sama dengan ROLLBACK).

STMIK
Amik Riau

#02

KONTROL KONKURENSI

DEFENISI

- **Kontrol konkurensi** adalah suatu system manajemen database (DBMS) konsep yang digunakan untuk mengatasi konflik dengan mengakses simultan atau mengubah data yang dapat terjadi dengan system multiuser

- Tujuan dari kontrol konkurensi adalah untuk memastikan serializability dari transaksi dalam lingkungan multiuser.

- Tiga masalah utama: **lost updates**, **uncommitted data**, & **inconsistent retrievals**.

1. LOST UPDATES (Hilangnya Data Pembaruan)

- Muncul ketika 2 transaksi konkuren, T1 & T2 mengupdate elemen data yang sama & salah satu update hilang (ditimpa oleh transaksi lain).

- Contoh ilustrasi pada tabel PRODUCT: Asumsikan nilai produk (PROD_QOH) adalah 35. Terdapat 2 transaksi, T1 & T2, muncul & mengupdate nilai PROD_QOH. T1 menambah 100 unit sementara T2 mengurangi 30 unit ke nilai PROD_QOH.

1. LOST UPDATES (Hilangnya Data Pembaruan)

STMIK Amik Riau

- Contoh ilustrasi pada tabel PRODUCT: Asumsikan nilai produk (PROD_QOH) adalah 35. Terdapat 2 transaksi, T1 & T2, muncul & mengupdate nilai PROD_QOH. T1 menambah 100 unit sementara T2 mengurangi 30 unit ke nilai PROD_QOH.

Eksekusi Berurutan dari 2 Transaksi

TIME	TRANSACTION	STEP	STORED VALUE
1	T1	Read PROD_QOH	35
2	T1	PROD_QOH = 35 + 100	
3	T1	Write PROD_QOH	135
4	T2	Read PROD_QOH	135
5	T2	PROD_QOH = 135 – 30	
6	T2	Write PROD_QOH	105

Masalah Lost Updates

TIME	TRANSACTION	STEP	STORED VALUE
1	T1	Read PROD_QOH	35
2	T2	Read PROD_QOH	35
3	T1	PROD_QOH = 35 + 100	
4	T2	PROD_QOH = 35 – 30	
5	T1	Write PROD_QOH (Lost update)	135
6	T2	Write PROD_QOH	5

1. LOST UPDATES (Hilangnya Data Pembaruan)

Misalkan ada tabel PRODUCT, dimana dalam tabel tsb ada field bernama Jumlah_Produk yang isinya 35.

1. Kemudian ada dua transaksi

   T1 = Membaca isi field Jumlah_Produk yang isinya 35

   T2 = Ikut juga Membaca

2. Kemudian
   T1 = Melakukan update dengan menambah isi dengan 100

   T2 = Melakukan operasi hasil yang sama setelah membaca dengan Jumlah_Produk - 30

3. Terjadilah masalah

   TI = Setelah menjumlahkan, maka T1 akan mengupdate nilai yang ada di database menjadi 100 + 35 = 135

   T2 = Juga ditulis

4. Problem

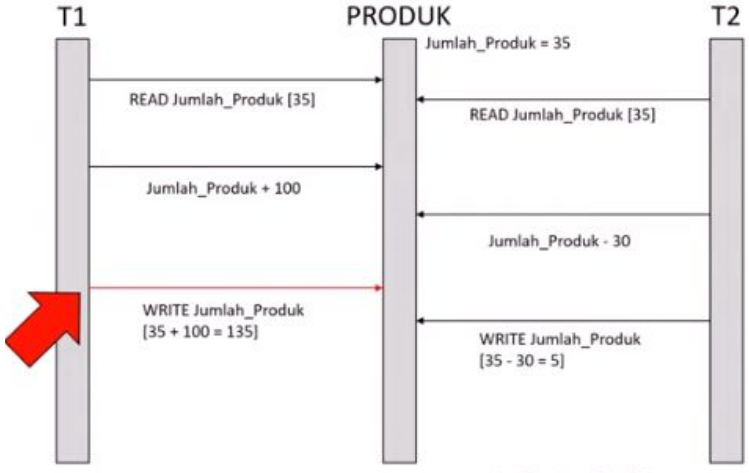
   Hasil akhir didatabase = 5 sementara pada proses update seharusnya 135 – 30

   Sehigga kita melihat problem :

   T1 tidak diisolasi dgn baik / tdk menyelesaikan transaksi sampai selesai, setelah WRITE harusnya commit baru hasil di T1 diaca oleh T2

Problem sebetulnya terjadi saat proses update Jumlah_Produk di T1 tambah 100, T2 dikurangi 30, Hasilnya akan berbeda

➔ Sehingga filed Jumlah_Produk tidak konsisten, pembaruan yang dilakukan oleh T1 Hilang karena ditimpa oleh T2



Eksekusi Berurutan dari 2 Transaksi

TIME	TRANSACTION	STEP	STORED VALUE
1	T1	Read PROD_QOH	35
2	T1	PROD_QOH = 35 + 100	
3	T1	Write PROD_QOH	135
4	T2	Read PROD_QOH	135
5	T2	PROD_QOH = 135 – 30	
6	T2	Write PROD_QOH	105

Masalah Lost Updates

TIME	TRANSACTION	STEP	STORED VALUE
1	T1	Read PROD_QOH	35
2	T2	Read PROD_QOH	35
3	T1	PROD_QOH = 35 + 100	
4	T2	PROD_QOH = 35 – 30	
5	T1	Write PROD_QOH (Lost update)	135
6	T2	Write PROD_QOH	5

2. UNCOMMITTED DATA

Muncul ketika 2 transaksi, T1 & T2, dieksekusi secara bersamaan & transaksi pertama (T1) dibalikkan setelah transaksi kedua (T2) mengakses data yang tidak dicommit – sehingga melanggar properti isolation dari transaksi.

Eksekusi yang Benar dari 2 Transaksi

TIME	TRANSACTION	STEP	STORED VALUE
1	T1	Read PROD_QOH	35
2	T1	PROD_QOH = 35 + 100	
3	T1	Write PROD_QOH	135
4	T1	*****ROLLBACK *****	35
5	T2	Read PROD_QOH	35
6	T2	PROD_QOH = 35 – 30	
7	T2	Write PROD_QOH	5

Masalah Uncommitted Data

TIME	TRANSACTION	STEP	STORED VALUE
1	T1	Read PROD_QOH	35
2	T1	PROD_QOH = 35 + 100	
3	T1	Write PROD_QOH	135
4	T2	Read PROD_QOH (Read uncommitted data)	135
5	T2	PROD_QOH = 135 – 30	
6	T1	***** ROLLBACK *****	35
7	T2	Write PROD_QOH	105

2. UNCOMMITTED DATA

STMIK Amik Riau

1. TI melakukan transaksi membaca Jumlah_Produk (35)
   Lalu T1 menambah Jumlah_Produk (35) dengan nilai baru 100 ➔35 + 100

   Ditulis menjadi 135

   Sampai disini tdk ada masalah. Sudah ditulis.

2. Problem muncul : Hasil dari T1 = 135 dibaca oleh T2, setelah itu 135 setelah dibaca – 30 oleh T2.

3. Problem muncul pada saat T1 MEMBATALKAN/ROLLBACK (Dikembalikan ke nilai awal). WRITE yang 135 sementara ini sudah dibaca oleh T2.

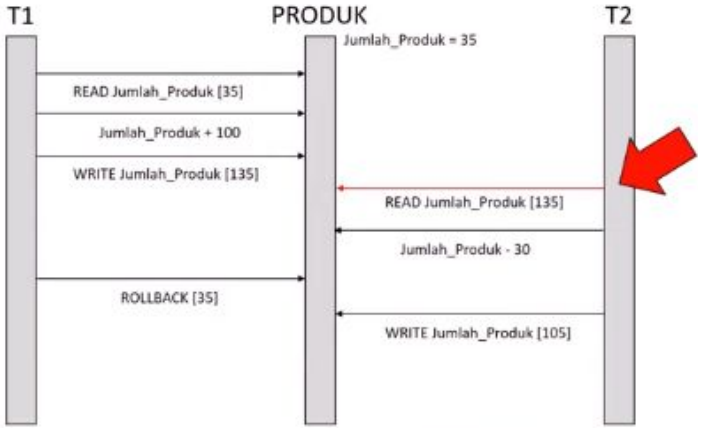
   Seakan akan Jumlah_Produk + 100 tidak jadi dilaksanakan.

4. T2 = Pada saat WRITE sudah salah. Seharusnya hasilnya adalah 35 – 30 = 5 bukan 105
   Tidak Konsisten

Betul data sudah ditulis tapi belum commit.. belum benar2 ditulis menjadi database yang konsisten baru boleh dibaca oleh transaksi berikutnya (T2).

Sebetulnya probemnya sama dengan yg Lost Updates, yaitu Isolasi.

T1 belum selesai pekerjaan, kalau perintahnya sampai commit tdk ada masalah, namun karena dia rollback (dikembalikan nilai awal) ini jadi masalah. Hasilnya adalah database yang tidak konsisten, seharusnya hasilnya 5 ini menjadi 105.



Eksekusi yang Benar dari 2 Transaksi

TIME	TRANSACTION	STEP	STORED VALUE
1	T1	Read PROD_QOH	35
2	T1	PROD_QOH = 35 + 100	
3	T1	Write PROD_QOH	135
4	T1	*****ROLLBACK *****	35
5	T2	Read PROD_QOH	35
6	T2	PROD_QOH = 35 – 30	
7	T2	Write PROD_QOH	5

Masalah Uncommitted Data

TIME	TRANSACTION	STEP	STORED VALUE
1	T1	Read PROD_QOH	35
2	T1	PROD_QOH = 35 + 100	
3	T1	Write PROD_QOH	135
4	T2	Read PROD_QOH (Read uncommitted data)	135
5	T2	PROD_QOH = 135 – 30	
6	T1	***** ROLLBACK *****	35
7	T2	Write PROD_QOH	105

3. INCONSISTENT RETRIEVALS

- Muncul ketika transaksi mengakses data sebelum & sesudah 1 atau lebih transaksi lain yang selesai menggunakan data tersebut.

- Contoh jika transaksi TI menghitung fungsi summary (aggregate) pada serangkaian data sementara transaksi lain (T2) mengupdate data yang sama. Masalahnya transaksi mungkin membaca beberapa data sebelum diubah & beberapa data setelah diubah, sehingga menghasilkan hasil yang salah.

3. INCONSISTENT RETRIEVALS

STMIK Amik Riau

Pengambilan Data selama Update

```
TRANSACTION T1              TRANSACTION T2
SELECT SUM(PROD_QOH)        UPDATE PRODUCT
FROM PRODUCT                SET PROD_QOH = PROD_QOH + 10
                            WHERE PROD_CODE = 1546-QQ2
                            UPDATE PRODUCT
                            SET PROD_QOH = PROD_QOH – 10
                            WHERE PROD_CODE = 1558-QW1
                            COMMIT;
```

Masalah Inconsistent Retrievals

TIME	TRANSACTION	ACTION	VALUE	TOTAL
1	T1	Read PROD_QOH for PROD_CODE = '11QER/31'	8	8
2	T1	Read PROD_QOH for PROD_CODE = '13-Q2/P2'	32	40
3	T2	Read PROD_QOH for PROD_CODE = '1546-QQ2'	15	
4	T2	PROD_QOH = 15 + 10		
5	T2	Write PROD_QOH for PROD_CODE = '1546-QQ2'	25	
6	T1	Read PROD_QOH for PROD_CODE = '1546-QQ2'	25	(After) 65
7	T1	Read PROD_QOH for PROD_CODE = '1558-QW1'	23	(Before) 88
8	T2	Read PROD_QOH for PROD_CODE = '1558-QW1'	23	
9	T2	PROD_QOH = 23 – 10		
10	T2	Write PROD_QOH for PROD_CODE = '1558-QW1'	13	
11	T2	***** COMMIT *****		
12	T1	Read PROD_QOH for PROD_CODE = '2232-QTY'	8	96
13	T1	Read PROD_QOH for PROD_CODE = '2232-QWE'	6	102

3. INCONSISTENT RETRIEVALS

T1 ➔ Membaca (READ) Jumlah_Produk untuk Produk tertentu, misalnya : P01 jumlahnya (8) lalu
T1 ➔ Kembali membaca (READ) Jumlah_Produk utk Kode_Produk = P02 (32)
Jadi Total yang sudah dibaca T1 = 8 + 32 = 40 tanpa memperdulikan Kode_Produk hanya Memperdulikan Jumlah_Produk saja, hasilnya = 40

Ini belum COMMIT.

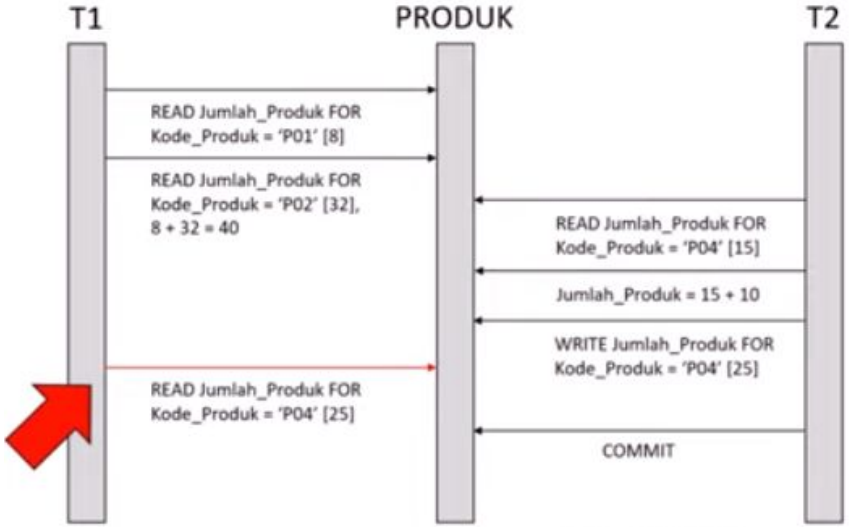
T2 ➔ Sudah melakukan transaksi membaca (READ), meskipun misalkan tidak masalah karena Kode_Produk nya berbeda hasilnya berbeda juga. Sebetulnya tdk apa apa dilakukan, misalkan T1, T2 atau T3 dst membaca tdk ada masalah.

Jika terjadi operasi pada T2, bisa mengakibatkan **inconsistent retrievals**, misalkan Jumlah_Produk = 15 + 10, lalu di tulis (WRITE). Sehingga hasilnya Kode_Produk = P04 (25)

Problem Muncul….

T1 ➔Karena sudah di tulis pada T2 dimasukkan kedalam Jumlah_Produk (25)
Maka nilai awal utk Kode_Produk apapun nilainya (25).
Masalah yang terjadi adalah yang membaca (READ) 2* pada T1 tertimpa oleh WRITE di T2.
Padahal seharusnya Kode_Produk P01 + P02 = 40 di T1 , tetapi karena di WRITE di T2 jumlahnya (25) yang dibaca jadinya 25, yg 40 hilang. TIDAK KONSISTEN.

T2 ➔ Commit. Tidak ada masalah sepertinya, di T1 yang terjadi inconsistent retrievals, sehingga data yang dihasilkan tdk konsisten.



STMIK
Amik Riau

#03

METODE KONTROL KONKURENSI

Tiga Metode Utk Mengatasi Kontrol Konkurensi pada DBMS

STMIK Amik Riau

1. OPTIMISTIC

- **Penundaan Pengecekan** apakah transaksi memenuhi isolasi dan aturan integritas lainnya (misalnya serializability dan recovery) sampai akhir, tanpa memblokir transaksi yang membaca atau menulis.

2. PESSIMISTIC

- **Melakukan Blok Operasi Transaksi** jika dapat menimbulkan pelanggaran aturan, sampai kemungkinan pelanggaran menghilang. Memblokir operasi biasanya akan berakibat pada pengurangan kinerja.

3. SEMI OPTIMISTIC

- Melakukan blok operasi dalam beberapa transaksi, jika transaksi tersebut dapat **menyebabkan pelanggaran** beberapa aturan. Tidak memblokir dalam situasi lain menggunakan aturan yang akan menunda pemeriksaan (jika diperlukan) sampai akhir transaksi ini, seperti yang dilakukan dengan mekanisme optimis.

STMIK Amik Riau

THANKS!

Ada Pertanyaan?
Boleh juga ke
WA
FB
IG