Nama : Lukas Sandy
NIM : 11814007
Kelas : MBD RD

1.) Aspek Atomicity ⇒ Setiap operasi dalam transaksi tersebut harus diselesaikan secara utuh / total dari awal sampai akhir. Jika terjadi kegagalan pada salah satu state ditengah proses transaksi, sistem harus memastikan bahwa state yang dieksekusi sebagian tidak mempengaruhi data di database.

Aspek consistency ⇒ Dampak dari eksekusi dari sebuah transaksi harus menjamin bahwa data pada database konsisten, kemungkinan saat transaksi dieksekusi, database menjadi tidak konsisten, tetapi setelah transaksi selesai dieksekusi, database harus konsisten.

Aspek Isolation ⇒ Jika dalam satu waktu banyak transaksi terjadi secara bersamaan, setiap transaksi tidak boleh mengakses transaksi lain yang sedang dieksekusi walaupun berhubungan. Jika hal tersebut dilanggar, maka akan membuat database tidak konsisten. Hasil dari transaksi yang sementara harus terjaga dan tidak boleh diakses oleh transaksi lain.

Aspek durability ⇒ Jika transaksi berhasil dilakukan, maka hal itu harus di update pada database walaupun terjadi kegagalan sistem.

2) Menurut saya, penjadwalan dari transaksi tersebut bukanlah serializability, karena ketika dilihat pada proses transaksi T_2 terdapat perintah R(B) dan W(B) dimana dilanjutkan dengan perintah R(A) yang menjadikannya bukan serializability. Jika dilihat dari urutan penjadwalannya, maka perintahnya : R(A) R(B) W(B) R(A) W(A) W(B).

3.) a. T_{11} =
```
Lock-S (X);
read (X);
~~release~~
unlock (X);
lock-S (Y);
read (Y);
unlock (Y);
if X=0 then Y:=Y+1;
lock-X (Y);
write (Y);
unlock (Y);
```

T_{12} =
```
Lock-S (Y);
read (Y);
unlock (Y);
Lock-S (X);
read (X);
unlock (X);
if Y=0 then X:=X+1;
lock-X (X);
write (X);
unlock (X);
```

b. Menurut saya kedua transaksi tersebut akan terjadi deadlock dimana transaksi T_{12} harus menunggu transaksi T_{11} melepaskan locknya untuk mengeksekusi, begitu juga sebaliknya sehingga kedua transaksi ini saling menunggu yang menyebabkan terjadinya deadlock.

contohnya ketika
```
if X=0 then Y:=Y+1;
        lock-X (Y);
        write (Y);
        unlock (Y);

if Y=0 then X:=X+1;
        lock-X (X);
        write (X);
        unlock (X);
```

dimana proses transaksi X harus menunggu proses transaksi Y selesai agar bisa dijalankan.

4.)

	T_7	T_8
1	read(D)	
2		read(D)
3	read(A)	
4		read(A)
5	display(A+D)	
6		B← B-100.000
7		write(D)
8		read(A)
9		A← A+100.000
10		write(A)
11		display(A+D)

5.)

a)

Log	Write	Output
<T0, start>		
<T0, X, 1000, 900>		
	X = 900	
<T0, Y, 1000, 50>		
	Y = 950	
<T1, commit>		
<T2, start>		
<T2, X, 900, 1100]		
	X = 1100	
<T3, start>		
<T3, Y, 950, 1550>		
	Y = 1550	
<T2, Y, 1550, 2550>		
	Y = 2550	
<T2, commit>		
<T3, X, 1100, 900>		
	X = 900	
<T3, commit>		

5.) Kondisi yang error yaitu transaksi tidak dapat dilakukan karena nilai X tidak dapat dibaca.

6.) Berdasarkan transaksi tersebut, kemungkinan error terjadi pada transaksi 2 dan 3 karena pd transaksi tersebut dilakukan secara bersamaan. Dapat diasumsikan bahwa ketika transaksi 2 membaca nilai Y dijalankan lebih dulu, maka nilai Y pada transaksi 3 akan berbeda.

7.) ~~a. nilai~~

DD	A	B	C	D
a.) inisial	0	0	0	0
b) crash	200	2500	1500	1500
c) recovery	200	0	0	1500

d) Pada saat terjadi crash setelah T_3, maka sistem melakukan recovery dengan melihat record sampai menemukan checkpoint. Setelah menemukan checkpoint maka dilakukan undo dan kembali dieksekusi pada T_3 dimana nilai A dieksekusi menjadi 200 dan nilai D menjadi 1500. Setelah itu sistem mencari checkpoint yang sudah melakukan start tetapi belum melakukan commit. setelah menemukannya yaitu T_1 dan T_2 maka sistem melakukan undo pada nilai B sehingga menjadi 0 dan C juga menjadi 0.

~~e)~~