ELTE | IK

INFORMATIKAI KAR

Adatbázisok 2.

9. gyakorlat

Tranzakciók

- Tranzakciónak tekintjük egy logikai egységbe tartozó adatbázis utasítások halmazát.
- A tranzakcióknak rendelkezniük kell az ACID tulajdonságokkal.
- A naplózás a tartósságot (durability) biztosítja (ACID).

Tranzakciók

- A tranzakciók és az adatbázis kölcsönhatásának 3 helyszíne van:
  - Az adatbázis elemeit tartalmazó lemezblokkok
  - A pufferkezelő által használt memóriaterület
  - A tranzakció memóriaterülete

Tranzakciók alapműveletei

- INPUT(X): Az *X* adatbáziselemet tartalmazó lemezblokk másolása a memória pufferbe.
- READ(X,t): Az *X* adatbáziselem bemásolása a tranzakció *t* lokális változójába. Ezután kapja meg a *t* lokális változó az *X* értékét.
- WRITE(X,t): A *t* lokális változó tartalma az *X* adatbáziselem memóriapufferbeli tartalmába másolódik. Ezután másolódik át a *t* lokális változó értéke a pufferbeli *X*-be.
- OUTPUT(X): Az *X* adatbáziselemet tartalmazó puffer kimásolása lemezre.
- FLUSH LOG – naplóblokkok lemezre írása

Tranzakció: A és B értékének a megduplázása

Tevékenység	t	Mem A	Mem B	Lemez A	Lemez B
READ(A, t)	5	5		5	8
t = t*2	10	5		5	8
WRITE(A, t)	10	10		5	8
READ(B, t)	8	10	8	5	8
t = t*2	16	10	8	5	8
WRITE(B, t)	16	10	16	5	8
OUTPUT(A)	16	10	16	10	8
OUTPUT(B)	16	10	16	10	16

Naplózás

- A napló nem más, mint naplóbejegyzések sorozata.
- Ha hiba keletkezik megnézhetjük mi történt az adatbázisban a hiba fellépéséig.
- A naplót a naplókezelő írja és kizárólag írásra van megnyitva.
- A naplóblokkok elsődlegesen a memóriában vannak tárolva, de amint lehet fizikai tárolóra írja őket a rendszer.

UNDO naplózás lemezre írási sorrendje

1. A módosításra vonatkozó naplóbejegyzések kiírása.
2. A módosított adatbáziselemek kiírása.
3. A COMMIT naplóbejegyzés kiírása.

UNDO naplózás példa

Tevékenység	t	Mem A	Mem B	Lemez A	Lemez B	Napló
						<START T>
READ(A, t)	5	5		5	8	
t = t*2	10	5		5	8	
WRITE(A, t)	10	10		5	8	<T, A, 5>
READ(B, t)	8	10	8	5	8	
t = t*2	16	10	8	5	8	
WRITE(B, t)	16	10	16	5	8	<T, B, 8>
OUTPUT(A)	16	10	16	10	8	
OUTPUT(B)	16	10	16	10	16	
						<COMMIT T>
FLUSH LOG						

UNDO helyreállítás

- A naplót a végétől visszafelé vizsgáljuk.
- Ha találunk egy módosító bejegyzést két eset lehetséges:
  - A tranzakcióhoz már volt COMMIT – semmit nem kell tenni.
  - A tranzakció befejezetlen – az értéket vissza kell állítani.
- Ha elfogytak a naplóbejegyzések, a be nem fejezett tranzakcióhoz <ABORT T> naplóbejegyzést írunk.
- Ezután FLUSH LOG művelettel kiváltjuk a napló lemezre írását.

UNDO feladatok

- Adjuk meg a helyreállítás-kezelő tevékenységeit, ha az utolsó lemezre került naplóbejegyzés:
  1. <start U>
  2. <U, D, 40>
  3. <COMMIT U>
  4. <T, E, 50>
  5. <COMMIT T>

< START T>

<T, A, 10>

<START U>

<U, B, 20>

<T, C, 30>

<U, D, 40>

<T, A, 11>

<U, B, 21>

<COMMIT U>

<T, E, 50>

<COMMIT T>

UNDO egyszerű ellenőrzőpont képzés

- Új tranzakció indítási kérések leállítása.
- Várakozás a még aktív tranzakciók befejezésére.
- A napló lemezre írása.
- <CKPT> naplóbejegyzés írása a naplóba és a lemezre.
- Tranzakció indítási kérések fogadása.

UNDO működés közbeni ellenőrzőpont képzés

- <START CKPT(T1,…,Tk)> naplóbejegyzés készítése és a naplóbejegyzések lemezre írása (FLUSH)
- Várakozás a T1..Tk tranzakciók befejezésére, közben nem tiltva új tranzakciók indítását.
- Ha T1…Tk mindegyike befejeződött, akkor <END CKPT> naplóbejegyzés készítése és lemezre írása (FLUSH).

UNDO helyreállítás ellenőrzőpont esetén

- Ha visszafelé haladva <END CKPT> bejegyzéssel találkozunk, akkor elég visszamenni az előtte lévő <START CKPT> bejegyzésig.
- Ha visszafelé haladva <START CKPT(T1…Tk)> bejegyzéssel találkozunk elég visszamenni T1…Tk közül a legkorábbi kezdetéhez.

UNDO ellenőrzőpont feladat

- Adjuk meg a helyreállítás-kezelő tevékenységeit, ha az utolsó lemezre került naplóbejegyzés:
  1. <COMMIT U>
  2. <R, E, 60>

< START T>

<T, A, 10>

< START U>

<T, A, 11>

<U, B, 21>

<START CKPT(T, U)>

<START R>

<COMMIT U>

<R, B, 31>

<T, E, 50>

<COMMIT T>

<END CKPT>

<R, E, 60>

REDO naplózás

- A régi értékek helyett az új értékeket kell a naplóban megőriznünk.
- Lemezre írás sorrendje:
  - A módosításokat leíró naplóbejegyzések kiírása.
  - A COMMIT naplóbejegyzés kiírása.
  - Az adatbáziselemek cseréje a lemezen.

REDO naplózás példa

Tevékenység	t	Mem A	Mem B	Lemez A	Lemez B	Napló
						<START T>
READ(A, t)	5	5		5	8	
t = t*2	10	5		5	8	
WRITE(A, t)	10	10		5	8	<T, A, 10>
READ(B, t)	8	10	8	5	8	
t = t*2	16	10	8	5	8	
WRITE(B, t)	16	10	16	5	8	<T, B, 16>
						<COMMIT T>
FLUSH LOG						
OUTPUT(A)	16	10	16	10	8	
OUTPUT(B)	16	10	16	10	16	
						<END T>
FLUSH LOG						

REDO helyreállítás

- A naplót az elejétől a vége felé vizsgáljuk.
- Ha találunk egy módosító bejegyzést három eset lehetséges:
  - A tranzakcióhoz nincs COMMIT – nem kell tenni semmit.
  - A tranzakcióhoz van END – nem kell tenni semmit.
  - A tranzakcióhoz van COMMIT, de nincs END – a módosítást hajtsuk végre.
- Ha elfogytak a naplóbejegyzések, a befejezett tranzakciókhoz írjunk <END T> bejegyzést.
- Ezután FLUSH LOG művelettel kiváltjuk a napló lemezre írását.

Módosított REDO napló: befejezett tranzakciókhoz nincs <END T>, helyette a be nem fejezettekhez írunk <ABORT T> bejegyzést.

REDO feladatok

- Adjuk meg a helyreállítás-kezelő tevékenységeit, ha az utolsó lemezre került naplóbejegyzés:
  1. <start U>
  2. <COMMIT U>
  3. <T, E, 50>
  4. <COMMIT T>
  5. <END U>

<start T>

<T, A, 10>

<start U>

<U, B, 20>

<T, C, 30>

<U, D, 40>

<T, A, 11>

<U, B, 21>

<COMMIT U>

<T, E, 50>

<COMMIT T>

<END U>

<END T>

REDO működés közbeni ellenőrzőpont képzés

- <START CKPT(T1,...,Tk)> naplóbejegyzés készítése és a naplóbejegyzések lemezre írása (FLUSH)
- Azon adatbáziselemek lemezre írása, amelyek az ellenőrzőpont kezdetekor már befejeződtek, de a módosításaik még nincsenek a lemezen.
- <END CKPT> naplóbejegyzés készítése és lemezre írása (FLUSH).

- Helyreállítás:
  - A helyreállításhoz elég a legutóbbi <END CKPT> előtti <START CKPT(Ti)> tranzakciói közül a legkorábbinak a kezdetéig -> <START Ti>-ig visszamenni

UNDO/REDO naplózás

- A módosító bejegyzés két komponensű: <T, X, u, v>
- Tároljuk a régi és az új értéket is.
- A módosított elemek lemezre írása a COMMIT előtt és után is történhez.

UNDO/REDO helyreállítás

- A legkorábbitól kezdve állítsuk helyre minden befejezett (COMMIT) tranzakció hatásait.
- A legutolsótól kezdve tegyük semmissé minden be nem fejezett tranzakció módosításait.

UNDO/REDO naplózás példa

Tevékenység	t	Mem A	Mem B	Lemez A	Lemez B	Napló
						<START T>
READ(A, t)	5	5		5	8	
t = t*2	10	5		5	8	
WRITE(A, t)	10	10		5	8	<T, A, 5, 10>
READ(B, t)	8	10	8	5	8	
t = t*2	16	10	8	5	8	
WRITE(B, t)	16	10	16	5	8	<T, B, 8, 16>
OUTPUT(A)	16	10	16	10	8	
						<COMMIT T>
FLUSH LOG						
OUTPUT(B)	16	10	16	10	16	

UNDO/REDO működés közbeni ellenőrzőpont

- <START CKPT(T1,…,Tk)> naplóbejegyzés készítése és a naplóbejegyzések lemezre írása (FLUSH)
- Írjuk a lemezre az összes adatbáziselemet, amelyek még nem kerültek oda.
- <END CKPT> naplóbejegyzés készítése és lemezre írása (FLUSH).

UNDO/REDO feladatok

- Adjuk meg a helyreállítás-kezelő tevékenységeit, ha az utolsó lemezre került naplóbejegyzés:
  1. <start U>
  2. <COMMIT U>
  3. <T, E, 50, 51>
  4. <COMMIT T>
  5. <END U>

<start T>

<T, A, 10, 11>

<start U>

<U, B, 20, 21>

<T, C, 30, 31>

<U, D, 40, 41>

<COMMIT U>

<T, E, 50, 51>

<COMMIT T>

<END U>

<END T>