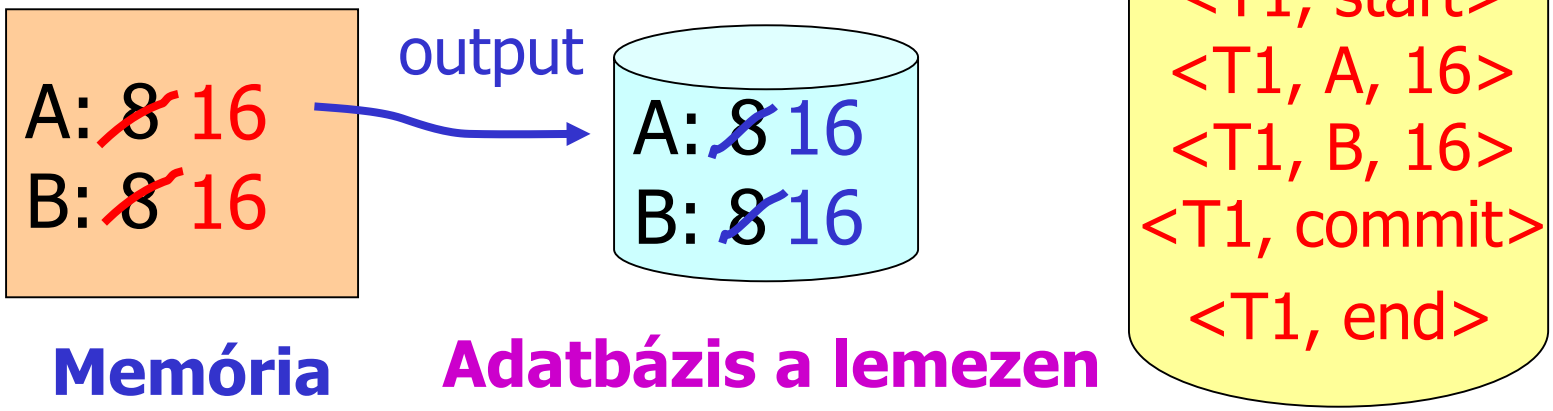
Redo logging (Helyrehozó naplózás)

T1: Read(A,t); t← t×2; write (A,t);
Read(B,t); t ←t×2; write (B,t);
Output(A); Output(B)



- **Az új értéket naplózzuk!**
- **Késleltetett kiírás!**

A helyrehozó naplózás szabálya

R1. Mielőtt az adatbázis bármely **X** elemét a lemezen módosítanánk, az **X** módosítására vonatkozó összes naplóbejegyzésnek, azaz **<T,X,v>**-nek és **<T, COMMIT>**-nak a lemezre kell kerülnie.

A helyrehozó naplózás esetén a lemezre írás sorrendje

(1) Ha egy T tranzakció v-re módosítja egy X adatbáziselem értékét, akkor egy <T,X,v> bejegyzést kell a naplóba írni.

(2) Az adatbáziselemek módosítását leíró naplóbejegyzések lemezre írása.

(3) A COMMIT naplóbejegyzés lemezre írása. (2. és 3. egy lépésben történik.)

(4) Az adatbáziselemek értékének cseréje a lemezen.

(5) A <T,end>-t bejegyezzük a naplóba, majd kiírjuk lemezre a naplót.

A helyrehozó naplózás szabályai

Lépés	*Tevékenység*	*t*	*M-A*	*M-B*	*D-A*	*D-B*	*Napló*
1)							<T,START>
2)	READ(A,t)	8	8		8	8	
3)	t := t*2	16	8		8	8	
4)	WRITE(A,t)	16	16		8	8	<T,A,16>
5)	READ(B,t)	8	16	8	8	8	
6)	t := t*2	16	16	8	8	8	
7)	WRITE(B,t)	16	16	16	8	8	<T,B,16>
8)							<T,COMMIT>
9)	FLUSH LOG						
10)	OUTPUT(A)	16	16	16	16	8	
11)	OUTPUT(B)	16	16	16	16	16	
12)							<T,END>
13)	FLUSH LOG						

Helyreállítás a REDO naplóból

- For every Ti with <Ti, commit> in log:
  - For all <Ti, X, v> in log:
    - Write(X, v)
    - Output(X)

Jó ez így?

Helyreállítás a REDO naplóból

A helyreállítás során fontos a módosítások sorrendje!

(1) Let S = set of transactions with <Ti, commit> (and no <Ti, end>) in log

(2) For each <Ti, X, v> in log, **in forward order** (earliest → latest) do:

- if Ti ∈ S then
  - Write(X, v)
  - Output(X)

(3) For each Ti ∈ S, write <Ti, end>

Ez miért lesz jó?

Helyreállítás a módosított REDO naplóból

Nem használunk <Ti,end> bejegyzést a befejezett tranzakciókra, helyette a be nem fejezetteket jelöljük meg <Ti,abort>-tal. (Módosított REDO napló)

1. Meghatározzuk a befejezett tranzakciókat (COMMIT).
2. Elemezzük a naplót az elejétől kezdve. Minden <T,X,v> naplóbejegyzés esetén:

a) Ha T be nem fejezett tranzakció, akkor nem kell tenni semmit.

b) Ha T befejezett tranzakció, akkor v értéket kell írni az X adatbáziselembe.

3. Minden T be nem fejezett tranzakcióra vonatkozóan <T, ABORT> naplóbejegyzést kell a naplóba írni, és a naplót ki kell írni lemezre (FLUSH LOG).

Helyreállítás

Lépés	Tevékenység	t	M-A	M-B	D-A	D-B	Napló
1)							<T,START>
2)	READ(A,t)	8	8		8	8	
3)	t := t*2	16	8		8	8	
4)	WRITE(A,t)	16	16		8	8	<T,A,16>
5)	READ(B,t)	8	16	8	8	8	
6)	t := t*2	16	16	8	8	8	
7)	WRITE(B,t)	16	16	16	8	8	<T,B,16>
8)							<T,COMMIT>
9)	FLUSH LOG						
10)	OUTPUT(A)	16	16	16	16	8	
11)	OUTPUT(B)	16	16	16	16	16	

1. **Ha a katasztrófa a 9) lépés után következik be:**

- A <T,COMMIT> bejegyzés már lemezen van. A helyreállító rendszer T-t befejezett tranzakcióként azonosítja.
- Amikor a naplót az elejétől kezdve elemzi, a <T,A,16> és a <T,B,16> bejegyzések hatására a helyreállítás-kezelő az A és B adatbáziselemekbe a 16 értéket írja.

Helyreállítás

Lépés	Tevékenység	t	M-A	M-B	D-A	D-B	Napló
1)							<T,START>
2)	READ(A,t)	8	8		8	8	
3)	t := t*2	16	8		8	8	
4)	WRITE(A,t)	16	16		8	8	<T,A,16>
5)	READ(B,t)	8	16	8	8	8	
6)	t := t*2	16	16	8	8	8	
7)	WRITE(B,t)	16	16	16	8	8	<T,B,16>
8)							<T,COMMIT>
9)	FLUSH LOG						
10)	OUTPUT(A)	16	16	16	16	8	
11)	OUTPUT(B)	16	16	16	16	16	

2. **Ha a hiba a 8) és 9) lépések között jelentkezik:**

- **A <T,COMMIT> bejegyzés már a naplóba került, de nem biztos, hogy lemezre íródott.**
- **Ha lemezre került, akkor a helyreállítási eljárás az 1. esetnek megfelelően történik, ha nem, akkor pedig a 3. esetnek megfelelően.**

Helyreállítás

Lépés	Tevékenység	t	M-A	M-B	D-A	D-B	Napló
1)							<T,START>
2)	READ(A,t)	8	8		8	8	
3)	t := t*2	16	8		8	8	
4)	WRITE(A,t)	16	16		8	8	<T,A,16>
5)	READ(B,t)	8	16	8	8	8	
6)	t := t*2	16	16	8	8	8	
7)	WRITE(B,t)	16	16	16	8	8	<T,B,16>
8)							<T,COMMIT>
9)	FLUSH LOG						
10)	OUTPUT(A)	16	16	16	16	8	
11)	OUTPUT(B)	16	16	16	16	16	

3. **Ha a katasztrófa a 8) lépést megelőzően keletkezik:**

- **Akkor <T,COMMIT> naplóbejegyzés még biztosan nem került lemezre, így T be nem fejezett tranzakciónak tekintendő.**
- **Ennek megfelelően A és B értékeit a lemezen még nem változtatta meg a T tranzakció, tehát nincs mit helyreállítani. Végül egy <ABORT T> bejegyzést írunk a naplóba.**

Összehasonlítás

- **Különbség a az UNDO protokollhoz képest:**
- **Az adat változás utáni értékét jegyezzük fel a naplóba**
- **Máshová rakjuk a COMMIT-ot, a kiírás elé => megtelhet a puffer**
- **Az UNDO protokoll esetleg túl gyakran akar írni => itt el lehet halasztani az írást**

Helyrehozó naplózás ellenőrzőpont-képzés használatával

- **Új probléma: a befejeződött tranzakciók módosításainak lemezre írása a befejeződés után sokkal később is történhet.**
- **Következmény: ugyanazon pillanatban aktív tranzakciók számát nincs értelme korlátozni, tehát nincs értelme az egyszerű ellenőrzőpont-képzésnek.**
- **A kulcsfeladat – amit meg kell tennünk az ellenőrzőpont-készítés kezdete és befejezése közötti időben – az összes olyan adatbáziselem lemezre való kiírása, melyeket befejezett tranzakciók módosítottak, és még nem voltak lemezre kiírva.**
- **Ennek megvalósításához a pufferkezelőnek nyilván kell tartania a piszkos puffereket (dirty buffers), melyekben már végrehajtott, de lemezre még ki nem írt módosításokat tárol. Azt is tudnunk kell, hogy mely tranzakciók mely puffereket módosították.**

Helyrehozó naplózás ellenőrzőpont-képzés használatával

- **Másrészről viszont be tudjuk fejezni az ellenőrzőpont-képzést az aktív tranzakciók (normális vagy abnormális) befejezésének kivárása nélkül, mert ők ekkor még amúgy sem engedélyezik lapjaik lemezre írását.**

- **A helyrehozó naplózásban a működés közbeni ellenőrzőpont-képzés a következőkből áll:**

1. **<START CKPT(T1,…,Tk)> naplóbejegyzés elkészítése és lemezre írása, ahol T1,…,Tk az összes éppen aktív tranzakció.**
2. **Az összes olyan adatbáziselem kiírása lemezre, melyeket olyan tranzakciók írtak pufferekbe, melyek a START CKPT naplóba írásakor már befejeződtek, de puffereik lemezre még nem kerültek.**
3. **<END CKPT> bejegyzés naplóba írása, és a napló lemezre írása.**

Helyreállítás ellenőrzőpont esetén

1. <T1, START>
2. <T1,A,5>
3. <T2, START>
4. <T1, COMMIT>
5. <T2,B,10>
6. <START CKPT(T2)>
7. <T2,C,15>
8. <T3, START>
9. <T3,D,20>
10. <END CKPT>
11. <T2, COMMIT>
12. <T3, COMMIT>

- Amikor az ellenőrzőpont-képzés elkezdődött, csak T2 volt aktív, de a T1 által A-ba írt érték még nem biztos, hogy lemezre került. Ha még nem, akkor A-t lemezre kell másolnunk, mielőtt az ellenőrzőpont-képzést befejezhetnénk.

Helyreállítás ellenőrzőpont 1. eset

1. <T1, START>
2. <T1,A,5>
3. <T2, START>
4. <T1, COMMIT>
5. <T2,B,10>
6. <START CKPT(T2)>
7. <T2,C,15>
8. <T3, START>
9. <T3,D,20>
10. <END CKPT>
11. <T2, COMMIT>
12. <T3, COMMIT>

RENDSZERHIBA

1. Ha a hiba előtt a naplóba feljegyzett utolsó ellenőrzőpont-bejegyzés <END CKPT>.

- Az olyan értékek, melyeket olyan tranzakciók írtak, melyek a <START CKPT(T1,…,Tk)> naplóbejegyzés megtétele előtt befejeződtek, már biztosan lemezre kerültek, így nem kell velük foglalkoznunk.

Helyreállítás ellenőrzőpont 1. eset

1. <T1, START>
2. <T1,A,5>
3. <T2, START>
4. <T1, COMMIT>
5. <T2,B,10>
6. <START CKPT(T2)>
7. <T2,C,15>
8. <T3, START>
9. <T3,D,20>
10. <END CKPT>
11. <T2, COMMIT>
12. <T3, COMMIT>

RENDSZERHIBA

<T2,COMMIT> és <T3,COMMIT> miatt T2 és T3 befejezett tranzakció.

Így <T2,B,10>, <T2,C,15> és <T3,D,20> alapján a lemezre újraírjuk a B, a C és a D tartalmát, megfelelően 10, 15 és 20 értékeket adva nekik.

- Elég azokat a tranzakciókat venni, melyek az utolsó <START CKPT(T1,…,Tk)> naplóbejegyzésben a Ti-k között szerepelnek, vagy ezen naplóbejegyzést követően indultak el. (T2,T3)
- A naplóban való keresés során a legkorábbi <Ti, START> naplóbejegyzésig kell visszamennünk, annál korábbra nem.
- Ezek a START naplóbejegyzések akárhány korábbi ellenőrzőpontnál előbb is felbukkanhatnak.

Helyreállítás ellenőrzőpont 1. eset

1. <T1, START>
2. <T1,A,5>
3. <T2, START>
4. <T1, COMMIT>
5. <T2,B,10>
6. <START CKPT(T2)>
7. <T2,C,15>
8. <T3, START>
9. <T3,D,20>
10. <END CKPT>
11. ~~<T2, COMMIT>~~ **RENDSZERHIBA**
12. <T3, COMMIT>

<T2,COMMIT> miatt csak T2 befejezett tranzakció.

Így <T2,B,10>, <T2,C,15> alapján a lemezre újraírjuk a B, a C tartalmát, megfelelően 10, 15 értékeket adva nekik.

- Elég azokat a tranzakciókat venni, melyek az utolsó <START CKPT(T1,…,Tk)> naplóbejegyzésben a Ti-k között szerepelnek, vagy ezen naplóbejegyzést követően indultak el. (T2,T3)
- T3 most be nem fejezett, így nem kell újragörgetni.
- **A helyreállítást követően egy <T3, ABORT> bejegyzést írunk a naplóba.**

Helyreállítás ellenőrzőpont 2. eset

1. <T1, START>
2. <T1,A,5>
3. <T2, START>
4. <T1, COMMIT>
5. <T2,B,10>
6. <START CKPT(T2)>
7. <T2,C,15>
8. <T3, START>
9. <T3,D,20>
10. <END CKPT>
11. <T2, COMMIT>
12. <T3, COMMIT>

RENDSZERHIBA

S...E...S

2. Ha a naplóba feljegyzett utolsó ellenőrzőpont-bejegyzés a <START CKPT(T1,…,Tk)>.

- Az előző <END CKPT> bejegyzéshez tartozó <START CKPT(S1,…,Sm)> bejegyzésig vissza kell keresnünk, és helyre kell állítanunk az olyan befejeződött tranzakciók tevékenységének eredményeit, melyek ez utóbbi <START CKPT(S1,…,Sm)> bejegyzés után indultak, vagy az Si-k közül valók.
- Ha nincs előző <END CKPT>, akkor a napló elejéig kell visszamenni.

Helyreállítás ellenőrzőpont 2. eset

1. <T1, START>
2. <T1,A,5>
3. <T2, START>
4. <T1, COMMIT>
5. <T2,B,10>
6. <START CKPT(T2)>
7. <T2,C,15>
8. <T3, START>
9. <T3,D,20>
10. <END CKPT>
11. <T2, COMMIT>
12. <T3, COMMIT>

RENDSZERHIBA

- Most nem találunk korábbi ellenőrzőpont-bejegyzést, így a napló elejére kell mennünk.
- Így esetünkben az egyedüli befejezett tranzakciónak T1-et fogjuk találni, ezért a <T1,A,5> tevékenységet helyreállítjuk.
- A helyreállítást követően <T2, ABORT> és <T3, ABORT> bejegyzést írunk a naplóba.

Redo naplózás összefoglalása

1. **Redo naplózás R1 szabálya, írás csak a Commit után**
2. **END nélküli naplózás**
3. **A helyreállítás algoritmusa (a befejezett tranzakciókat újra végrehajtjuk)**
4. **Ellenőrzőpont képzése (a start chkpoint előtt befejezett tranzakcióknál kikényszerítjük a változtatások lemezreírását.)**