Grau en Enginyeria Informàtica

Sistemes de Gestió de Bases de Dades

Pràctica 4 Undo & Redo

Lluis Barca Pons
lluis.barca1@estudiant.uib.es

8 de gener de 2024

1. *Introducció*

En l'àmbit de la gestió de bases de dades, un aspecte crucial és la capacitat de recuperar-se de diverses fallades, incloent-hi fallades operacionals, de programari, i físiques. Aquest treball se centra en la comprensió dels mecanismes de recuperació de transaccions, específicament els diaris *Undo&Redo*. S'analitzaran cinc transaccions específiques, amb els seus valors inicials i els canvis realitzats durant la seva execució, aquest treball proporciona una visió detallada del funcionament intern dels diaris *Undo&Redo*. Aquest anàlisi no només il·lumina les operacions a nivell de transacció, sinó que també revela com els sistemes de bases de dades gestionen situacions d'error, com una fallada del sistema, garantint així la robustesa i la fiabilitat de la gestió de dades.

1.1 Transacció analitzada a la Primera Part

```
Begin transaction;
A := A * C;
B := D / 2;
C := A / D;
D := C
commit;
```

Valors inicials:

- A = 5
- B = 2
- C = 8
- D = 10

1.2 Transacció analitzada a la Segona Part

```
1.  Begin Transaction; A := A * 2; B := B * 2; Commit;
2.  Checkpoint
3.  Begin Transaction; A := A * B; B := B * 2; Commit;
4.  Begin Transaction; A := A * 2; B := B * 2; Rollback;
5.  Begin Transaction; A := A * 4; B := B * 4;
6.  Checkpoint;
7.  Begin Transaction; A := A * B; B := B * 2; Rollback;
8.  Begin Transaction; A := A * 2; B := B * 2; Commit;
9.  Commit; -- De la transacció que faltava.
```

Valors inicials:

- A = 2
- B = 4

Suposem que s'executen totes les passes exceptuant el darrer *commit*, ja que el sistema cau. El tipus de diari és *Redo&Undo log*.

2. *Primera part*

Per a aquesta primera part, s'ha realitzat corresponent taula de l'aplicació d'un diari *Redo log*, explicades les passes que s'han seguit i mencionat els processos i memòria implicats.

Valors Inicials: A = 5, B = 2, C = 8, D = 10

Pas	**Acció**	**Variables**		**Memòria Cau**				**Disc**				**Diari**
		t	**z**	**A**	**B**	**C**	**D**	**A**	**B**	**C**	**D**	
1								5	2	8	10	
2	READ(A, t)	5		5				5	2	8	10	Inici(T)
3	READ(C, z)	5	8	5		8		5	2	8	10	
4	t := t * z	40	8	5		8		5	2	8	10	
5	WRITE(A, t)	40	8	40		8		5	2	8	10	Update(T, A, 40)
6	READ(B, t)	2	8	40	2	8		5	2	8	10	
7	READ(D, z)	2	10	40	2	8	10	5	2	8	10	
8	t := z / 2	5	10	40	2	8	10	5	2	8	10	
9	WRITE(B, t)	5	10	40	5	8	10	5	2	8	10	Update(T, B, 5)
10	READ(A, t)	40	10	40	5	8	10	5	2	8	10	
11	t := t / z	4	10	40	5	8	10	5	2	8	10	
12	WRITE(C, t)	4	10	40	5	4	10	5	2	8	10	Update(T, C, 4)
13	WRITE(D, t)	4	10	40	5	4	4	5	2	8	10	Update(T, D, 4)
14												Commit(T)
15	Escriure log											
16	OUTPUT(A)	4	10	40	5	4	4	40	2	8	10	
17	OUTPUT(B)	4	10	40	5	4	4	40	5	8	10	
18	OUTPUT(C)	4	10	40	5	4	4	40	5	4	10	
19	OUTPUT(D)	4	10	40	5	4	4	40	5	4	4	

2.1 Explicació de Passes

1. En aquest pas no succeeix res realment, únicament podem observar l'estat actual del sistema i els valors pertinents al disc.
2. El DBW passa el valor d'A de disc a memòria cau (SGA) i s'emmagatzema en la variable `t`.
3. El DBW passa el valor de C de disc a memòria cau (SGA) i s'emmagatzema en la variable `z`.
4. Es realitza l'operació `t := t * z`, que equivaldria a l'operació `A = A * C`, emmagatzemant el resultat en la variable `t`.
5. El procés servidor passa el valor emmagatzemat en `t`, que es troba a la PGA, a la memòria cau (SGA). També afegeix un registre al diari indicant la modificació del bloc A amb el seu nou valor.
6. El DBW passa el valor de B de disc a memòria cau (SGA) i s'emmagatzema en la variable `t`.
7. El DBW passa el valor de D de disc a memòria cau (SGA) i s'emmagatzema en la variable `z`.
8. Es realitza l'operació `t := z / 2`, que equivaldria a l'operació `B = D / 2`, emmagatzemant el resultat en la variable `t`.
9. El procés servidor passa el valor emmagatzemat en `t`, que es troba a la PGA, a la memòria cau (SGA). També afegeix un registre al diari indicant la modificació del bloc B amb el seu nou valor.
10. El DBW passa el valor d'A de disc a memòria cau (SGA) i s'emmagatzema en la variable `t`.
11. Es realitza l'operació `t := t / z`, que equivaldria a l'operació `A = A / D`, emmagatzemant el resultat en la variable `t`.
12. El procés servidor passa el valor emmagatzemat en `t`, que es troba a la PGA, a la memòria cau (SGA). També afegeix un registre al diari indicant la modificació del bloc C amb el seu nou valor.
13. El procés servidor passa el valor emmagatzemat en `t`, que es troba a la PGA, a la memòria cau (SGA). També afegeix un registre al diari indicant la modificació del bloc D amb el seu nou valor.
14. Es registra el *commit* al diari, la qual cosa indica que podem registrar els canvis a disc.
15. Escrivim al diari tots els canvis que han estat realitzats en la memòria cau, però encara no s'han escrit en el disc.
16. El DBW escriu el valor actual d'A de la memòria cau al disc.
17. El DBW escriu el valor actual d'B de la memòria cau al disc.
18. El DBW escriu el valor actual d'C de la memòria cau al disc.
19. El DBW escriu el valor actual d'D de la memòria cau al disc.

3. *Segona part*

3.1 Contingut del diari

Transacció	Diari
1	Inici(T1)
1	Update(T, A, 2, 4)
1	Update(T, B, 4, 8)
1	Commit(T1)
	Checkpoint
2	Inici(T2)
2	Update(T, A, 4, 32)
2	Update(T, B, 8, 16)
2	Commit(T2)
3	Inici(T3)
3	Update(T, A, 32, 64)
3	Update(T, B, 16, 32)
3	Rollback
4	Inici(T4)
4	Update(T, A, 32, 128)
4	Update(T, B, 16, 64)
	Checkpoint
5	Inici(T5)
5	Update(T, A, 128, 8192)
5	Update(T, B, 64, 128)
5	Rollback
6	Inici(T6)
6	Update(T, A, 128, 256)
6	Update(T, B, 64, 128)
6	Commit(T6)
4	Commit(T4)

3.2 Valors d'A i B

Recordem que els valors **inicials** d'A i B son els següents, abans d'executar cap transició:

- **A = 2**
- **B = 4**

I després d'executar totes les transicions (a la taula esquerra podem observar el contingut del diari *Undo&Redo log* corresponent) els valors d'A i B, en el **moment de la caiguda**, són els següents:

- **A = 128**
- **B = 64**

3.3 Procés de Recuperació del Sistema

3.3.1 Protcol ARIES

El protocol ARIES[1], quan el sistema es recupera després d'una caiguda, segueix aquests passos per a assegurar que la base de dades retorna a un estat consistent:

- **Analysis Phase:**
  1. Establim el punt més antic en què va començar una transacció activa en el moment de l'error.
  2. Establim el punt de la modificació més antic no escrit en disc.
- **Redo Phase:** Repeteix totes les modificacions des del punt 2 fins al final.
- **Undo Phase:** Torna endarrere les accions que no han realitzat el *commit* fins a arribar a 1.

3.3.2 Aplicació al nostre cas

Valors abans de la caiguda:

Últim estat conegut d'A i B després de la transacció 4: A = 128, B = 64.
Les transaccions 5 i 6 no s'han completat.

Fases d'ARIES:

- **Analysis:**
  1. Identifiquem les transaccions actives i les transaccions completades des del darrer *checkpoint*.
  2. Transacció 6 i la part pendent de la Transacció 4 són actives.
- **Redo:**
  1. Tot el diari es reprodueix des del darrer *checkpoint*, fins i tot les operacions de les transaccions que finalment es desfaran.
  2. Redo de T4: A = 128 * 4, B = 64 * 4; però ja que aquesta transacció no es va completar, no es guarden els canvis a disc.
  3. Redo de T6: A = 128 * 2, B = 64 * 2; els canvis es reprodueixen, però no es guarden a disc perquè la transacció no s'ha confirmat.

[1] L'Algorisme per a la Recuperació i la Integració d'Estructures d'Emmagatzematge, és un enfocament sofisticat per a la recuperació de bases de dades que utilitza tres components principals: *redo*, *undo*, i *analysis*.

- **Undo:**

    1. Desfem les transaccions incompletes en ordre invers.
    2. Undo de T6: Reverteix els canvis a A i B al seu estat pretransacció.
    3. Undo de la part pendent de T4: Com que no hi ha un *rollback* explícit, però no hi ha hagut un *commit* final, els canvis no es guarden.

Després de la recuperació, els valors a disc serien els mateixos que després de la transacció 4, que és l'últim estat consistent conegut: A = 128, B = 64.

Això assegura que tots els canvis realitzats per transaccions completades des de l'últim checkpoint fins al moment de la fallada es mantenen, mentre que els canvis de transaccions no completades són descartats, retornant la base de dades a un estat consistent.

4. *Conclusió*

L'anàlisi detallada de les transaccions i la seva gestió a través dels diaris *Undo&Redo* ens ha permès comprendre millor els mecanismes subtils que operen dins dels sistemes de gestió de bases de dades. Hem vist que aquests diaris juguen un paper molt important en el manteniment de la consistència de la base de dades, especialment en situacions d'error i fallada del sistema. El procés de *Redo* assegura que les transaccions confirmades es reflecteixin de manera fiable, mentre que el procés d'*Undo* proporciona un mitjà per revertir les transaccions inacabades, mantenint així la integritat de les dades.

Aquest treball demostra la importància d'una planificació acurada i d'una gestió eficaç de les transaccions per assegurar operacions fiables i eficients en qualsevol sistema de bases de dades. Amb aquest coneixement, els desenvolupadors i administradors de bases de dades poden dissenyar millor els seus sistemes per afrontar amenaces i imprevistos, garantint al mateix temps una gestió de dades robusta i eficient.