C A P I T O L U L 15

CONCURENTA SI BLOCAREA : O VEDERE GENERALA

Scopul acestui capitol este acela de a arata cum ORACLE gestioneaza blocarea.

Ariile acoperite sunt :

- Tipuri si nivele de blocari
- Blocarea normala
- Blocari implicite si explicite
- DEADLOCK

Blocari

ORACLE utilizeaza blocarile pentru a controla accesul concurent la date. Blocarile sunt mecanisme pentru a preveni interactiunea destructiva intre utilizatorii care acceseaza aceleasi date.

Blocarile sunt utilizate pentru a atinge doua scopuri importante:

CONCURENTA DATELOR

- Utilizatorii care doresc sa citeasca adeleasi date sa nu astepte utilizatorii care scriu aceleasi blocuri de date.
- Utilizatorii care doresc sa scrie datr in baza de date sa nu astepte utilizatorii care citesc aceleasi date.
- Utilizatorii care vor sa scrie date asteapta doar ulilizatorii care modifica aceleasi linii.

CONSISTENTA LA CITIRE

- Fiecare bloc de instructiuni executat are o vedere consistenta asupra datelor din baza de date.
- Fiecare bloc nu este afectat de catre nici o schimbare efectuata de catre alt utilizator in timpul executiei acestiua.

Pentru a asigura concurenta datelor, ORACLE utilizeaza blocari si tranzactii. Blocarile sunt mecanisme utilizate pentru a preveni interactiunea destructiva intre utilizatorii care acceseaza aceleasi resurse.

Blocarile in ORACLE intra in urmatoarele categorii :

Blocari de date (DML locks)

Aceste blocari protejeaza datele. Blocarile de tabele blocheaza intreaga tebela, in timp ce blocarile de coloane blocheaza doar coloanele selectate.

Blocari de Dictionar (Dictionary Locks)

Aceste blocari protejeaza structura de obiecte a bazei de date; de exemplu ele protejeaza definitiile tebelelor si vederile (views). Blocarile de dictionar sunt de doua tipuri : blocari sintactice (parser) si blocari DDL.

In acest curs ne vom concentra doar asupra blocarilor de date.

Nivele de Blocare

Blocarile pot fi aplicate la nivelul unie tabele sau al unei linii.

Nivel de Linie
Afecteaza liniile individual. Doar blocarile exclusive (x) se aplica la nivel de linie.

Nivel de Tabela
Afecteaza utilizarea intregii tabele.

Tipuri de Blocari

Mod blocare Prescurtare	Actiuni interzise	Actiuni permise	Obtinut prin comenzile SQL
Share -S-	Modificari RX,SRX,X	Blocare linii, RS,S,cereri	LOCK TABLE tabela IN SHARE MODE
Exclusive -X-	Toate exceptand cereri,RS,RX,S, SRX,X si majoritatea blocurilor DDL	Cereri	LOCK TABLE tabela IN EXCLUSIVE MODE
Row share -RS-	Acces exclusiv, X	Modificari, RS,RX,S,SRX, cereri	SELECT...FOR UPDATE, LOCK TABLE tabela IN ROW SHARE MODE
Row exclusive -RX-	Acces exclusiv pentru citire/ scriere,S,SRX,X	modificari RX,RS, cereri	UPDATE,INSERT, DELETE, LOCK TABLE tabela IN ROW EXCLUSIVE MODE

```
Share row       Citirea tabelei    Blocare linii
LOCK TABLE tabela
exclusive       in intregime,RX,   RS,cereri
IN SHARE ROW
-SRX-           S,SRX,X
EXCLUSIVE MODE
------------------------------------------------------------
-----------------
```

Sumarul Blocarilor de Tabela (TM)

Blocurile care modifica linii explicite dintr-o tabela obtin mereu EXCLUSIVE ROW LOCKS (blocari exclusive de linie) si o blocare de tabela corespunzatoare.

EXCLUSIVE (X)
Blocarile exclusive permit cererile de interogare in resursa blocata dar interzic orice alta acti- vitate in tabela.

SHARE (S)
Blocarile de acest tip permit interogarile dar interzic modificarile intr-o tabela.

ROW SHARE (RS)
Permite accesul concurent la o tabela. Interzice ca alti utilizatori sa blocheze tabela pentru acces exclusiv.

ROW EXCLUSIVE (RX))
Sunt asemanatoare cu ROW SHARE dar interzic blocarea in mod SHARE. Aceste blocari sunt generate la modi- ficare,inserare sau stergere.

SHARE ROW
Sunt folosite pentru a vizualiza o tabela pentru a

EXCLUSIVE (SRX)
executa modificari selectiv si pentru a permite altor utilizatori sa vizualizeze tabela fara a o bloca in modul SHARE sau a modifica linii. Acest mod nu prea este utilizat.

Abreviatiile din paranteze sunt cele utilizate in Lock Monitor al SQL*DBA.

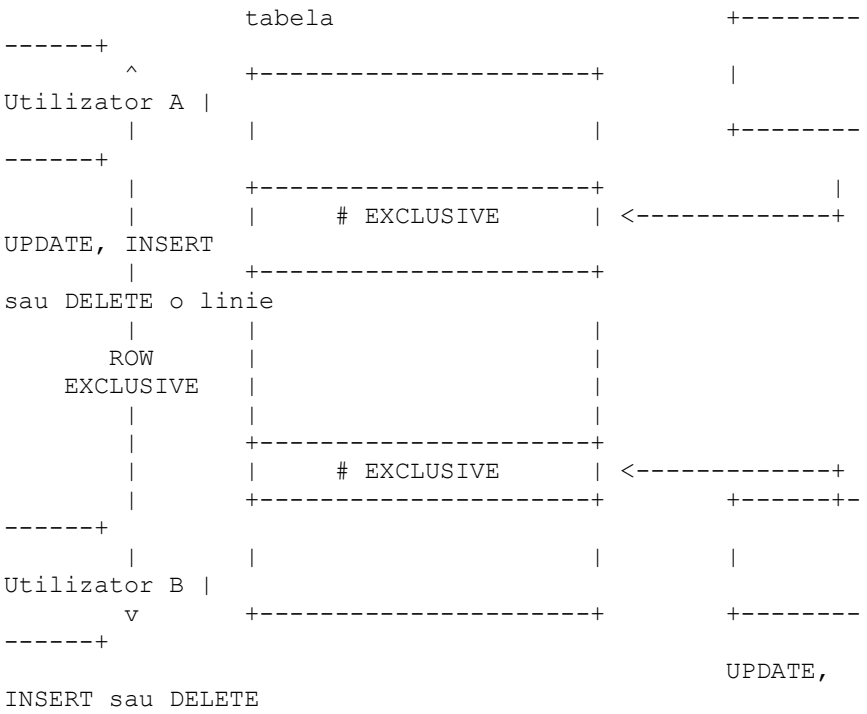
Blocarile manuale pot fi cerute de optiunea NOWAIT atunci cand utilizatorul nu doreste sa astepte ca resursa blocata sa fie eliberata.

Durata Blocarilor

- Toate blocarile acumulate de-a lungul unei tranzactii sunt eliberate cand tranzactia se termina.
- Toate blocarile acumulate de-a lungul unei tranzactii sunt eliberate cand tranzactia se intoarce .
- Toate blocarile acumulate dupa un punct de salvare sunt eliberate cand tranzactia se intoarce la punctul de salvare.

Blocarea implicita

Blocarile care apar in timpul unui DML normal, fara o cerere specifica de blocare.



Blocarile Datelor

Blocarea manuala poate fi preferata daca :

- O vizualizare consistenta asupra mai multor tabele este ceruta.
- Nu doriti ca altcineva sa modifice datele saupra carora opereaza tranzactia dumneavoastra.
- Un bloc de instructiuni nu trebuie sa astepte o blocare.
- O cerere este de citire repetata dintr-o tabela fara a schimba datele.

Blocurile de instructiuni care modifica blocarile implicite sunt :

- Selectiile pentru modificari
- Blocarile de tabele.

Blocarile manuale pot fi preferate blocarilor implicite in multe cazuri. Acestea pot include vizualizarea consistenta a unor tabele. Un exemplu poate fi un master si cateva tabele detaliate.

O tranzactie poate dori schimbarea unor date bazata pe alte inregistrari, dar inregistrarea respectiva nu trebiue modificata pana cand nu se termina toata tranzactia.

Un bloc nu trebuie sa astepte dupa nici un alt bloc ca sa se termine evitandu-se astfel conflictele de blocare.

- Un utilizator poate dori sa vada oun set de date in aceeasi stare in cereri multiple: principiul 'citirii repetate'.

Comanda LOCK TABLE

Comanda LOCK TABLE este utilizata pentru a bloca la nivel de tabela explicit :

```
LOCK TABLE tabela, tabela, ...
IN mod_de_blocare [NOWAIT]
```

Instructiunea SELECT FOR UPDATE

- Modifica blocarea implicita
- Blocheaza liniile cand acestea sunt selectate
- Blocheaza o linie fara a o modifica

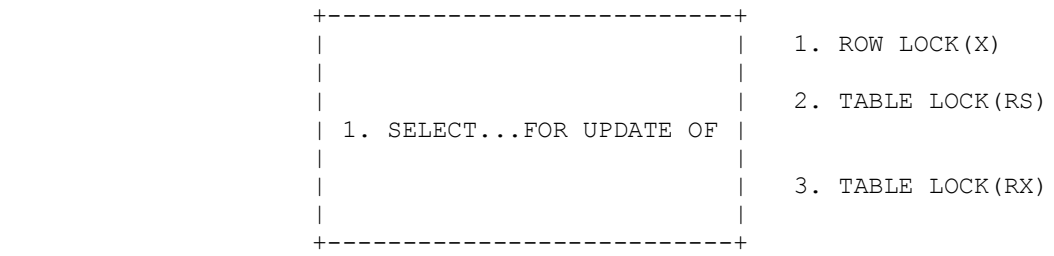
SELECT FOR UPDATE va suprascrie mecanismul implicit de blocare. Este utilizata inaintea executiei unei modificari (update). Diferenta dintre SELECT FOR UPDATE si UPDATE este aceea ca SELECT FOR UPDATE blocheaza lini ile mai devreme in tranzactie. Mai intai se executa cererea de identificare a liniilor ce vor fi modificate si dupa aceea blocheaza setul de linii, per mitand ca schimbarile sa aiba loc mai tarziu.

```
Sintaxa :

SELECT ...
FOT UPDATE [ OF coloana, coloana, ... ] [NOWAIT]
```

SELECT FOR UPDATE este recomandata pentru cazul cand se doreste blocarea unei linii fara modificare de exemplu, daca se doreste pornirea unei modificari a unor valori deja existente in linii, liniile nu trebuie sa poata fi modificare de catre altcineva inainte de suprascriere.

O cerere SELECT FOR UPDATE elibereaza blocarile la COMMIT sau ROLLBACK. Este indicat sa se utilizeze SELECT FOR UPDATE pentru cateva linii decat pentru un numar mare de linii.



Deadlock

Este foarte posibil intr-un mediu multi-utilizator, ca utilizatorii isi vor bloca unii altora resursele. Este de asemenea posibil ca doi utilizatori sa sfarseasca prin a-si bloca unul altuia resurse diferite. Aceasta situatie se numeste DEADLOCK deoarece fiecare utilizator asteapta sesursele detinute de celalalt utilizator.

De exemplu :

Tranzactia A	*Tranzactia B*
UPDATE EMP	*UPDATE EMP*
SET SAL=1200	*SET SAL=1000*
WHERE ENAME = 'SMITH';	*WHERE ENAME = 'JONES';*
UPDATE EMP	*UPDATE EMP*
SET SAL=2200	*SET SAL=1350*
WHERE ENAME = 'JONES';	*WHERE ENAME = 'SMITH';*

Cand primul bloc este executat, nu exista nici o problema. Oricum, cand ei vor incerca sa obtina blocarile pentru al doilea bloc, se va astepta in spatele celuilalt utilizator. Este deadlock deoarece nici macar prin asteptare problema nu va fi rezolvata.

Cand ORACLE detecteaza un deadlock, genereaza o eroare la unul dintre participantii tranzactiei, si intoarce blocul curent al tranzactiei respective. Aceasta rezolva deadlock-ul, astfel celalalt utilizator poate inca astepta pana cand resursa este disponibila. Utilizatorul semnalizat ar trebui sa execute explicit un ROLLBACK in tranzactie.

Deadlock-ul poate fi evitat daca utilizatorii care acceseaza aceleasi tabele blocheaza tabelele in aceeasi ordine ca ceilalti. Ar trebui predefinita o ordine de acces pentru toate tabelele in aplicatii si aplicatiile astfel construite incat sa urmeze ordinea specificata. Daca aceasta ordine este urmata in toate aplicatiile, probabilitatea de aparitie a dead-lock va fi minima.

Blocarile de date sunt eliberate de :

- COMMIT / ROLLBACK
- LOG OFF

```
Orice bloc explicit DDL.

ROWID (id. de linie) si Blocarea


COL ROWID VALOARE_NOUA IDENTIFICATOR_LINIE
NOPRINT

SELECT ENAME, JOB, HIREDATE SAL, ROWID
FROM EMP
WHERE WNAME = 'SCOTT'
FOR UPDATE


        +--------------------+
ROWID   | 00004C90.0001.0001 |
        +--------------------+

UPDATE EMP
SET JOB = 'SALESMAN,
        HIREDATE=SYSDATE,
        SAL=1.1*SAL
WHERE ROWID='&ROW_IDENT'
```

Este important sa se eexcute COMMIT cat mai cutand posibil pentru ca blocarile sa fue eliberate devreme.

Procesul COMMIT poate fi accelerat prin folosirea ROWID pentru a localiza liniile in tabele. ROWID este o pseudo-coloana care are cate o valoare pentru fiecare linie din tabela. ROWID ne spune adresa unei linii si prin aceasta este cea mai rapida cale de acces disponibila. Intoarce trei parti de informatie necesare pentru a localiza o linie :

```
ROWID           00004C90.0001.0001
```

- Care bloc in fisierul bazei de date (00004C90)
- Care linie din bloc (0001)
- In care fisier din baza de date este (0001)

Cand este creat ROWID ?

ROWID este creat automat de ORACLE cand o linie este modificata cu succes intr-o tabela de baze de date; cand o inserare este facuta permanenta.

De ce sa utilizam ROWID ?

ROWID are cateva utilizari importante :

- Sunt cele mai rapide mijloace de accesare a unei linii explicite
- Sunt identificatori unici pentru linii in tabele.

De notat ca se poate executa UPDATE / DELETE pe o linie cu ROWID doar daca linia respectiva a fost inainte blocata (adica SELECT FOR UPDATE).