vahidusefzadeh@gmail.com

enq: TX - row lock contention

این نوع wait معمولا زمانی که چند نفر بخواهند به صورت همزمان یک رکورد را تغییر دهند، اتفاق می افتد. و برای کم رنگ کردن آن باید برنامه را طوری تغییر داد تا کاربر زودتر مجبور به rollback یا commit شود.

این wait در مود Share(mode=4) یا بعبارتی دیگر ستون p1 ویوی v$active_session_history

برابر با 1415053316 می باشد) و یا مود Exclusive(mode=6 و p1=1415053318) ممکن است اتفاق بیفتد.

```
select distinct event,p1, mod(p1,16)  as "mode" from v$active_session_history where event like 'enq:%' and mod(p1,16) in(4,6);
```

معمولا در مود ۶(Exclusive) این اتفاق بخاطر delete و یا update یک رکورد رخ می دهد ولی در مود ۴، دلیل این اتفاق، به سه مورد زیر بر می گردد:

1. Unique key contention
2. Foreign Key contention
3. Bitmap index contention

MODE=6

برای مثال در نظر بگیرید فردی دستوری را اجرا می کند تا بتواند عدد ۱۵ موجود در ستون col1 را به ۱۶۵ تغییر دهد منتهی بعد از اجرای این دستور، مشغول کار دیگری می شود و این دستور را commit یا rollback نمی کند در همین حال فرد دیگری قصد دارد اطلاعات همان رکورد را ویرایش کند:

session1	session2
update usef_test set col1=165 where col1='15';	
	update usef_ test set col1=168 where col1='15'; **wait(**enq: TX - row lock contention**– mode=6)….**
update usef_test2 set name='usef' where name='vahid';	

در این صورت فرد دوم باید منتظر بماند تا نفر اول، تکلیف دستورش را با commit یا rollback مشخص کند. این انتظار نفر دوم باعث ایجاد رویداد انتظار enq: TX - row lock contention می شود. دو دستور زیر این نکته را به خوبی نشان می دهند:

```
select sid, sql_text from v$session s, v$sql q where sid in (select sid from v$session where state in ('WAITING') and wait_class != 'Idle' and event='enq: TX - row lock contention' and (q.sql_id = s.sql_id  or q.sql_id = s.prev_sql_id));
```

SID SQL_TEXT

```
--------        ----------------------------------------------------------------
164             begin :id := sys.dbms_transaction.local_transaction_id; end;
164             update usef_test  set col1=168 where col1='15'
```

select blocking_session, sid, serial#, wait_class, seconds_in_wait from v$session where blocking_session is not NULL order by blocking_session;

BLOCKING_SESSION	SID	SERIAL#	WAIT_CLASS	SECONDS_IN_WAIT
70	164	1024	Application	825

بعد از این اتفاق، دو گزارش از وضعیت قبل و بعد اجرای دستور update، از بانک گرفتیم که می تواند به ما کمک بیشتری کند. قسمت Top 5 Timed Foreground Event دوگزارش AWR را در این قسمت می بینید:

وضعیت بانک قبل از اجرای دستورات

Event	Waits	Time(s)	Avg wait (ms)	% DB time	Wait Class
DB CPU		1		130.38	
log file sync	17	0	4	12.08	Commit
Disk file operations I/O	100	0	0	3.74	User I/O
RFS attach	10	0	2	3.32	Other
RFS dispatch	10	0	1	2.31	Other

وضعیت بانک بعد از اجرای دستورات

Event	Waits	Time(s)	Avg wait (ms)	% DB time	Wait Class
enq: TX - row lock contention	1	1,437	1437112	98.04	Application
DB CPU		20		1.37	
db file scattered read	73	2	26	0.13	User I/O
db file sequential read	109	0	3	0.02	User I/O
SQL*Net break/reset to client	134	0	1	0.01	Application

MODE=4

همانطور که قبلا اشاره کردیم، در مود Share هم ممکن است این نوع از wait اتفاق بیفتد که مثال هر کدام از حالتهای ممکن را در این قسمت آوردیم.

۱.unique index :

```
create table usef_wait(a number(10));

create unique index usef_indx1 on usef_wait(a);
```

vahidusefzadeh@gmail.com

session1	session2
insert into usef_wait values(18);	
	insert into usef_wait values(18); **wait(**enq: TX - row lock contention **– mode=4)….**
…..	

۲. foreign key:

```
create table usef_parent(code number primary key);
create table usef_child (code number references usef_parent,book varchar(15));
```

session1	session2
insert into usef_parent values(1);	
	insert into usef_child values (1,'marefate_nafs'); **wait(**enq: TX - row lock contention **– mode=4)….**
…..	

۳. bitmap index:

همانطور که می دانیم این نوع از ایندکس معمولا برای محیط OLTP که دستورات DML زیادی دارند، مناسب نیست و بیشتر کاربرد آن در محیط DW می باشد.

```
create table usef_bitmap(id  number(10),code  number(10));
create bitmap index usef_bitindx1 on usef_bitmap(code);
insert into usef_bitmap values(1,2);
```

session1	session2
update usef_bitmap set code=6 where id=1	
	update usef_bitmap set code=6 where id=1 **wait(**enq: TX - row lock contention **– mode=4)….**
…..	

دستور زیر مشخص می کند که کدام دستور سبب ایجاد این رویداد انتظار شده است:

```
select    event, sql_id,CURRENT_OBJ# || ' ' || name obj,CURRENT_FILE# file#,CURRENT_BLOCK#  block# from
v$active_session_history ash,obj$ o where event  like 'enq: TX%' and o.obj# (+)= ash.current_obj#  order by
sample_time;
```

گزارش ASH بانک بعد از اجرای دستورات بالا:

Top SQL with Top Events

SQL ID	Planhash	Sampled # of Executions	% Activity	Event	% Event	Top Row Source	% RwSrc	SQL Text

3fdhbypzgtxhj	1287388552	1	44.92	enq: TX - row lock contention	44.92	UPDATE	44.92	update usef_bitmap set code=6 ...
8amd0nbjnm0vg		1	25.07	enq: TX - row lock contention	25.07	** Row Source Not Available **	25.07	insert into usef_wait values(1...
cgct9hhfuxfpv		1	17.90	enq: TX - row lock contention	17.90	** Row Source Not Available **	17.90	insert into usef_child values ...
3ju2t8005c6wv	1287388552	3	3.83	enq: TX - row lock contention	3.83	UPDATE	3.83	update usef_bitmap set code=4 ...

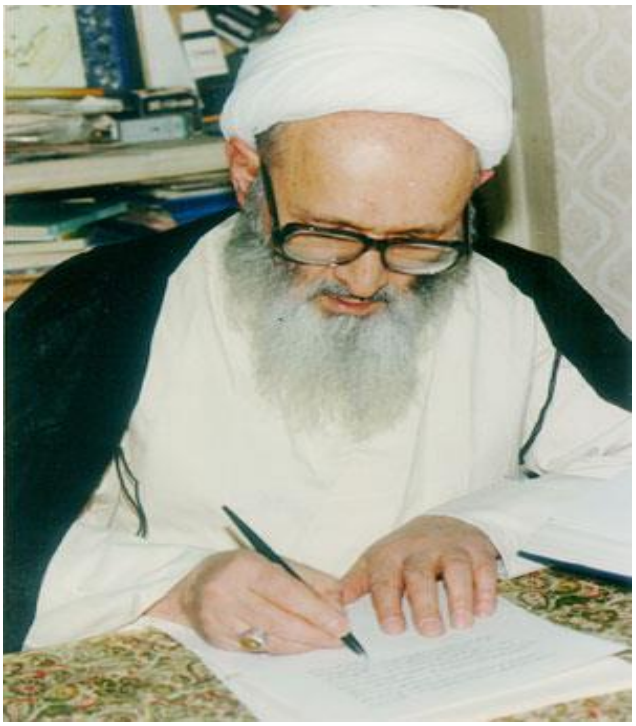
یک راه برای کاهش این رویداد انتظار:

همانطور که دیدیم دلیل اصلی این رویداد انتظار ، دادن فرصت زیاد به کاربر برای تصمیم گیری در مورد اینکه اطلاعات را ثبت نهایی کند یا خیر!؛ می باشد. بدون شک کاربر تمایل دارد تا فرصت زیادی را برای گرفتن این تصمیم در اختیار داشته باشد اما این فرصت می تواند برای بقیه ایجاد زحمت کند به همین دلیل باید چاره ای اندیشید.

یک راه برای حل این مسئله، تغییر ساختار کد به شکلی که کاربر بتواند از اطلاعات درون undo tablespace استفاده کند ولی مجبور باشد زود commit یا rollback کند یعنی این امکان را هم با استفاده از اطلاعات درون undo داشته باشد تا بتواند به اطلاعات قدیمی رکورد دسترسی داشته باشد برای این کار نیاز است علاوه بر اصلاح کد، اندازه undo tablespace را به اندازه نیازمان بزرگ کنیم و به مدت زمانی که لازم است اطلاعات قدیمی رکورد نگه داشته شوند، پارامتر undo_retention را تنظیم کنیم.

- ✓ البته می تونیم ایندکسهایی که wait ایجاد کردند را شناسایی و حذف کنیم و یا اینکه بعد از مدت زمانی مشخص، کاربرانی که commit یا rollback نمی کنند و دیگران راblock کردند را بکشیم هر چند که مَن يَقْتُلْ مُؤْمِنًا مُّتَعَمِّدًا فَجَزَآؤُهُ جَهَنَّمُ!

vahidusefzadeh@gmail.com



علامه حسن حسن زاده آملی:

الهی از من برهان توحید خواهند و من دلیل تکثیر.