ORACLE入門講座
待機イベントを知ろう

アリエル・ネットワーク
鈴木 健介

これは何？

ORACLEの内部状態を示す情報の一つである、待機イベントについて解説します。

待機イベントを知ることで、一歩進んだパフォーマンスチューニングが出来ます。
また、待機イベントという切り口を通して、ORACLEのアーキテクチャに対する
理解を深めていきます。

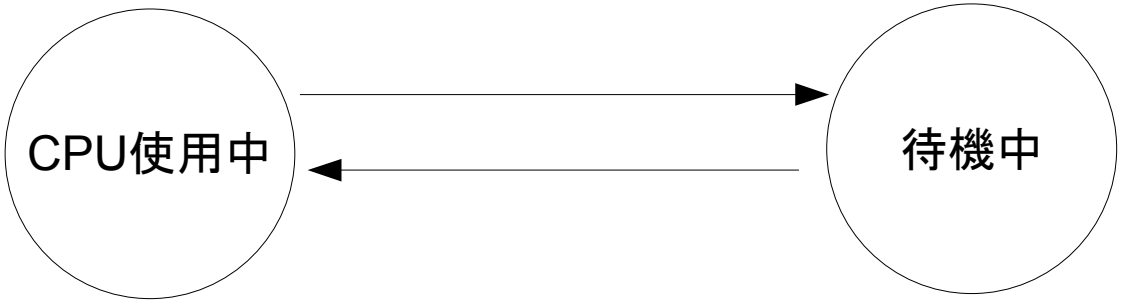
なお、ORACLEのバージョンは10.2を想定しています。

待機イベントってなんね？

ORACLEでのセッションが、ある特定の待機状態にあることを表したもの

待機とは？
->CPU使用中以外すべての状態

ORACLEセッションの状態



SQL解析処理
検索条件による絞込み処理
ソート処理

行ロック待ち
ディスクI/O完了待ち
他のプロセスの処理完了待ち
ユーザからの要求待ち

待機イベントを知ると何がうれしい？

・なんとなくORACLEが遅い、でもその原因が分からない。
　そんなとき、何が起きてて遅いのかが分かる。

・ORACLEの内部動作を透かしてみることが出来る

まずは見てみよう

今接続しているセッションの待機イベントを見てみる。
v$session(9i以前ではv$session_wait)で現在接続しているセッションの
待機イベントを調べることが出来る。

```
$ sqlplus  "/as sysdba"
SQL> select sid, username, event, state from v$session;

       SID USERNAME        EVENT                          STATE
---------- --------------- ------------------------------ -------------------
        20 HANA_TEST       SQL*Net message from client    WAITING
        22 LEOTEST         SQL*Net message from client    WAITING
        24 HANA_TEST       SQL*Net message from client    WAITING
        28 TOZAWA          SQL*Net message from client    WAITING
        29 V160            SQL*Net message from client    WAITING
        30 LEOTEST         SQL*Net message from client    WAITING
        32 V160            SQL*Net message from client    WAITING
        37 V160            SQL*Net message from client    WAITING
       126 SYS             SQL*Net message to client      WAITED SHORT TIME
以下略。
103 rows selected.
SQL>
```

なにが起きているか？
＝＞何も起きてません。

なお、STATEが"WAITING"でない場合は、CPU使用中の状態。
EVENTには前回待機したイベントが表示される。

待機イベントを見る方法

待機イベントを確認する方法としては、見たい尺度によって以下の方法がある。
採り方、見方については、必要に応じて後で述べる。

1. 今、発生している待機イベント
  v$session, v$session_wait

2. システム起動時からの累計
  v$system_event

3. セッションごとの累計
  v$session_event

4. 任意のセッションで発生した全ての待機イベント
  SQLトレース

5. ある期間内で発生した待機イベントの統計
  AWR, Statspack, bstat/estat
  (内部では、期間開始点と終了点でのv$system_eventの差分をとっている)

ここまで読んで、”分かった。あとは自分で調べる。”という人のために

もうひとつ重要な情報
待機イベントは最大3つのパラメータを持っていて、イベントの詳細情報を示している。
パラメータはそれぞれ”P1”, “P2”, “P3”という列名でv$sessionから確認できる。

各パラメータの意味は、待機イベントによって異なる。

例えば、”db file scattered read”なら、

P1：読み込むファイル番号
P2：読み込みを開始するブロック番号
P3：読み込むブロック数

となる。

その他、各待機イベントに関する説明は、リファレンスマニュアルの付録に書いてある。

http://otndnld.oracle.co.jp/document/products/oracle10g/102/doc_cd/server.102/B19228-03/waitevents.htm#11902

主だった待機イベントについては、パフォーマンスチューニングガイドに解説あり。
http://otndnld.oracle.co.jp/document/products/oracle10g/102/doc_cd/server.102/B19207-02/instance_tune.htm#169276

とはいえ、書いてある内容を理解するには、ORACLEのアーキテクチャに対する理解が
まぁそこそこそれなりに必須。なので、次からで軽くおさらいする。

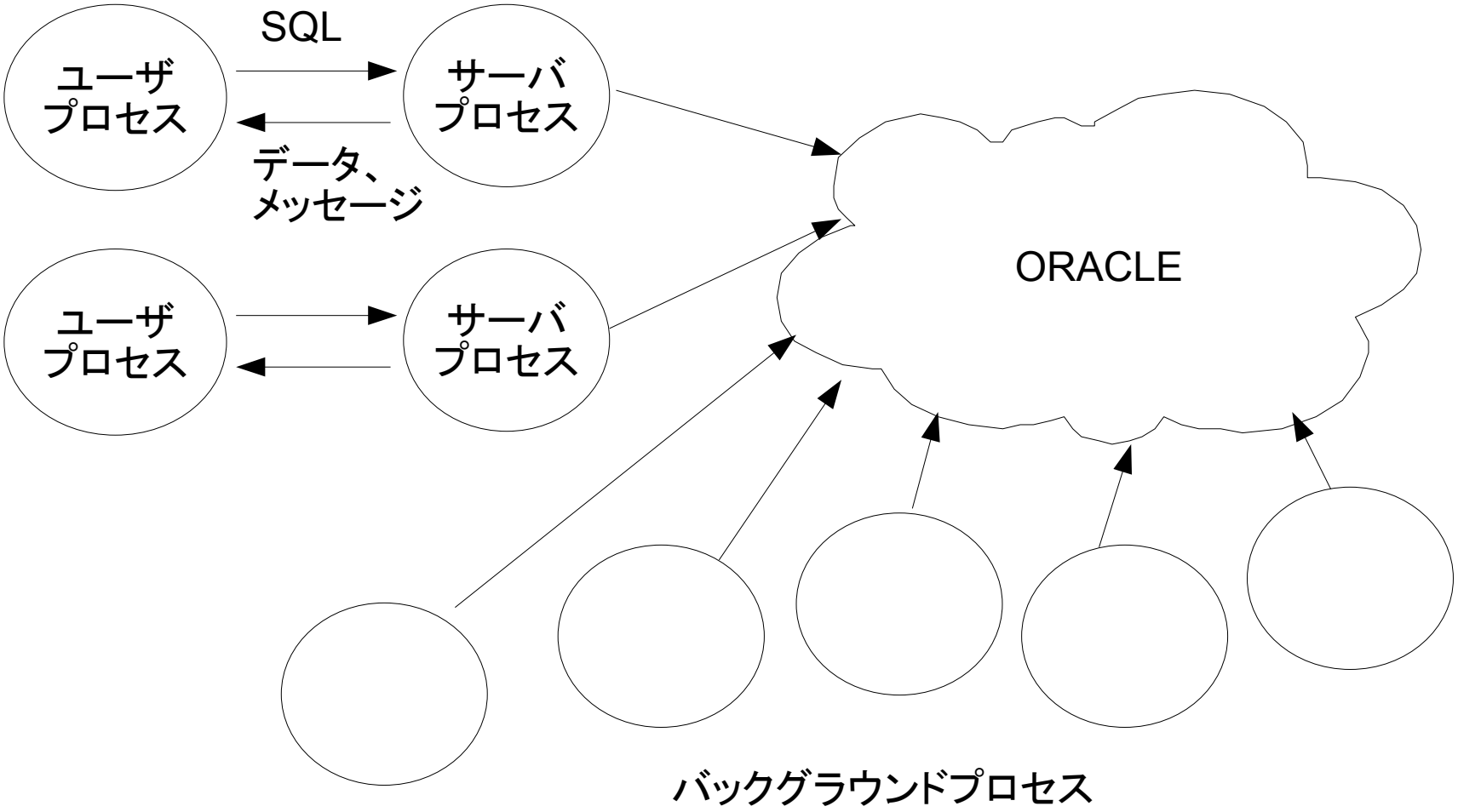
ORACLEの登場人物その1：プロセス

ユーザプロセス：ORACLEへ要求を発行するプロセス。SQL*Plusとかユーザ作成のDBアプリとか。

サーバプロセス：ユーザプロセスの要求を受け取って、データの処理を行うプロセス。

バックグラウンドプロセス：ORACLEのメンテナンス処理を担当。
基本的にはユーザの要求とは非同期に活動する。

待機イベントでは、サーバプロセスとバックグラウンドプロセスの状態を確認できる。



ORACLEの登場人物その2：SGA

システムグローバルエリアの略。

サーバプロセスとバックグラウンドプロセス達が、共通して参照できるメモリ領域
UNIX系では共有メモリ、WINDOWSではスレッド間で共用できるメモリ領域として表現される。

SGAの中身としては、以下の領域がある。

データベースバッファキャッシュ：データをキャッシュする領域。
　ORACLEでは一部の例外を除いて、データのやり取りはバッファキャッシュを通して行われる

共有プール：SQLの解析結果をキャッシュしておく領域。
　足りなくなるとSQLがエラーになる場合がある

REDOログバッファ：REDOログ情報(ORACLEの更新ログ)をバッファリングする領域
　LGWR(ログライター)バックグラウンドプロセスで、逐次REDOログファイルに書き出される

その他：ラージプール、JAVAプールなど

ORACLEの登場人物その3：ファイル群

データファイル：表や索引などデータベースオブジェクトのデータを格納する領域

UNDOデータファイル：正確にはデータファイルの一種。データの更新時に更新前のイメージを格納

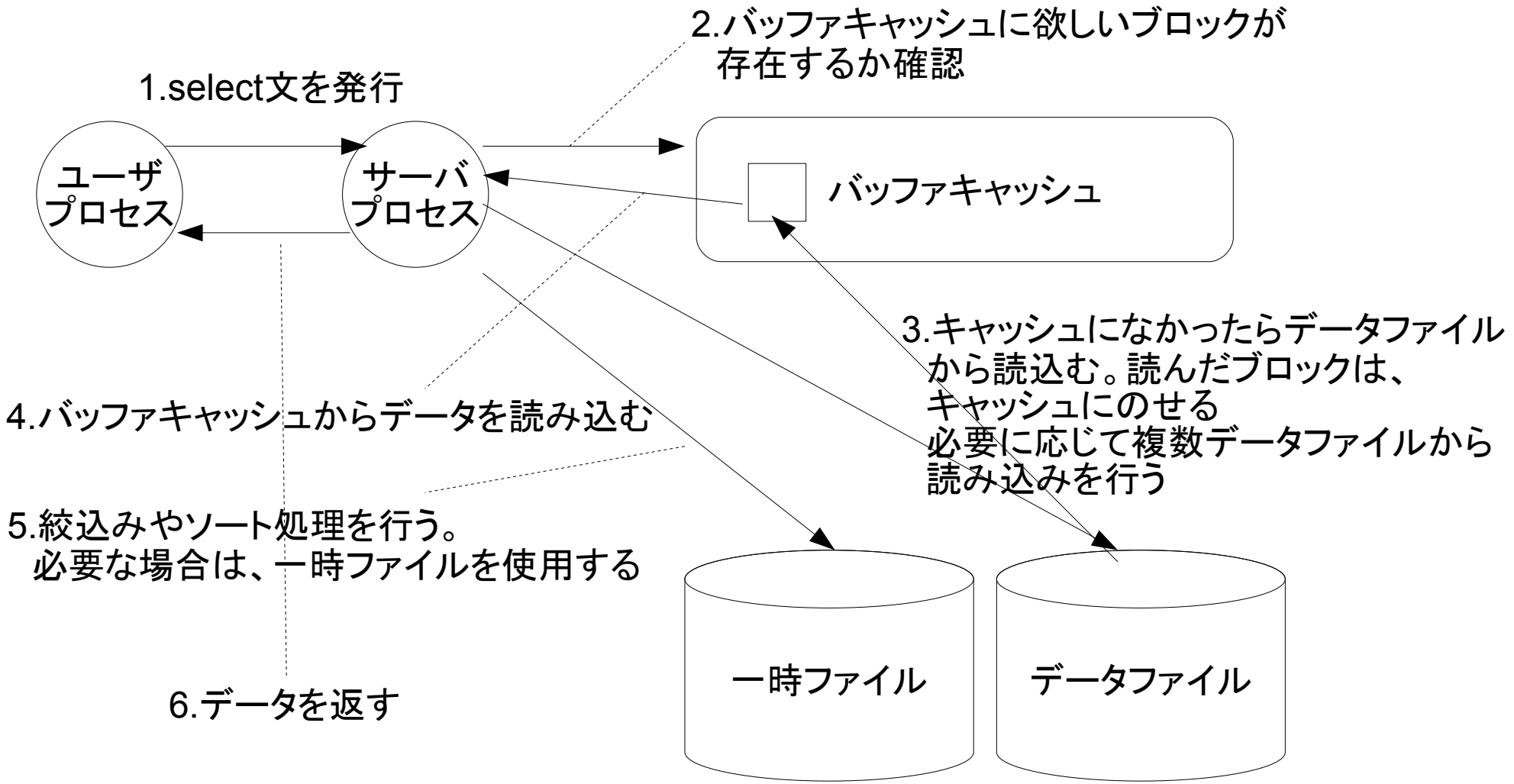
一時ファイル：ソート用の作業領域などに使用

REDOログファイル：REDOログを格納する領域

制御ファイル：データファイルの位置など管理情報を格納

以上を踏まえて、まずはSQLを発行したときにどんな待機イベントが発生するか見ていく。

SELECT実行時の動作



SELECT実行時の動作 - 発生する待機イベント

[よく出る]

db file sequential read

データファイルからの読み込み待ち。1イベントで1ブロック読み込み
索引経由の表アクセスなどで発生する

db file scattered read

データファイルからの読み込み待ち。1イベントで複数ブロック読み込み
全件検索での表アクセスで発生する

[時々出る]

direct path read temp

direct path write temp

一時ファイルに対するread/write
ディスクソートが行われたときに発生する

[まれに出る]

buffer busy waits

バッファ上の競合による待ち。
SELECT同士では、同時にデータファイルからの読み込みが必要となった場合に発生

free buffer waits

バッファに乗せようとしたら、空きバッファが足りなかった場合発生する。
変更済みのブロックがファイルに書き出されて、空きができるまで待機する。

実際に見てみよう - SQLトレースを用いて(1)

```
$ sqlplus /nolog

# 下準備。ディスクI/Oの発生を見たいので、あえてバッファキャッシュをクリアする。
SQL> conn /as sysdba
SQL> alter system flush buffer_cache;

System altered.

SQL> conn suzu/suzu

# SQLトレースの開始。waits=>trueで待機イベントを採取できる
SQL> exec dbms_monitor.session_trace_enable(waits=>true, binds=>true);

PL/SQL procedure successfully completed.

# トレースしたいSQLの実行。ここでは索引を使った表アクセス。
SQL> select * from suzutab where col1 = '128';

COL1                           COL2
------------------------------ ------------------------------
128                            128

# SQLトレースの停止。これをしないと、トレースが増え続けるので注意
SQL> exec dbms_monitor.session_trace_disable();

PL/SQL procedure successfully completed.
```

実際に見てみよう - SQLトレースを用いて(2)

トレースファイルがuser_dump_destに出力されていることを確認

```
$ ls -lrt | tail -n
-rwxrwx---+ 1 Administrators SYSTEM  32275 Aug 29 17:05 arieldb_ora_528.trc

$ tkprof arieldb_ora_528.trc tkp1.txt

[tkp1.txtの中身]
select *
from
 suzutab where col1 = '128'

(略) 実行計画などの情報

Elapsed times include waiting on following events:
  Event waited on                             Times   Max. Wait  Total Waited
  ----------------------------------------   Waited  ----------  ------------
  db file sequential read                         4        0.01          0.01
  SQL*Net message to client                       2        0.00          0.00
  SQL*Net message from client                     2        0.00          0.00
```

そのSQLで発生した待機イベントが統計情報として出力される

実際に見てみよう - SQLトレースを用いて(3)

トレースファイルの中身を見てみる

```
PARSING IN CURSOR #2 len=40 dep=0 uid=68 oct=3 lid=68 tim=21940409819 hv=1582318214 ad='1d863ec8'
select * from suzutab where col1 = '128'
END OF STMT
PARSE #2:c=62500,e=248963,p=12,cr=30,cu=0,mis=1,r=0,dep=0,og=1,tim=21940409810
BINDS #2:
EXEC #2:c=0,e=74,p=0,cr=0,cu=0,mis=0,r=0,dep=0,og=1,tim=21940409979
WAIT #2: nam='SQL*Net message to client' ela= 5 driver id=1111838976 #bytes=1 p3=0 obj#=-1
 tim=21940410049
WAIT #2: nam='db file sequential read' ela= 172 file#=4 block#=160196 blocks=1 obj#=59239
 tim=21940410312
WAIT #2: nam='db file sequential read' ela= 154 file#=4 block#=160221 blocks=1 obj#=59239
 tim=21940410539
WAIT #2: nam='db file sequential read' ela= 150 file#=4 block#=160191 blocks=1 obj#=59238
 tim=21940410803
FETCH #2:c=0,e=777,p=3,cr=4,cu=0,mis=0,r=1,dep=0,og=1,tim=21940410872
WAIT #2: nam='SQL*Net message from client' ela= 296 driver id=1111838976 #bytes=1 p3=0 obj#=59238
 tim=21940411614
FETCH #2:c=0,e=3,p=0,cr=0,cu=0,mis=0,r=0,dep=0,og=0,tim=21940411688
WAIT #2: nam='SQL*Net message to client' ela= 3 driver id=1111838976 #bytes=1 p3=0 obj#=59238
 tim=21940411737
WAIT #2: nam='SQL*Net message from client' ela= 324 driver id=1111838976 #bytes=1 p3=0 obj#=59238
 tim=21940412098
STAT #2 id=1 cnt=1 pid=0 pos=1 obj=59238 op='TABLE ACCESS BY INDEX ROWID SUZUTAB (cr=4 pr=3 pw=0 time=768 us)'
STAT #2 id=2 cnt=1 pid=1 pos=1 obj=59239 op='INDEX UNIQUE SCAN SYS_C0010241 (cr=3 pr=2 pw=0 time=532 us)'
WAIT #0: nam='SQL*Net message to client' ela= 3 driver id=1111838976 #bytes=1 p3=0 obj#=59238 tim=21940412320
*** 2008-08-29 17:04:03.968
WAIT #0: nam='SQL*Net message from client' ela= 19325062 driver id=1111838976 #bytes=1 p3=0 obj#=59238 tim=21959737425
=====================
```

発生した待機イベントが1つずつトレースされているので、より詳細な情報を得ることが出来る

実際に見てみよう－SQLトレースを用いて(4)

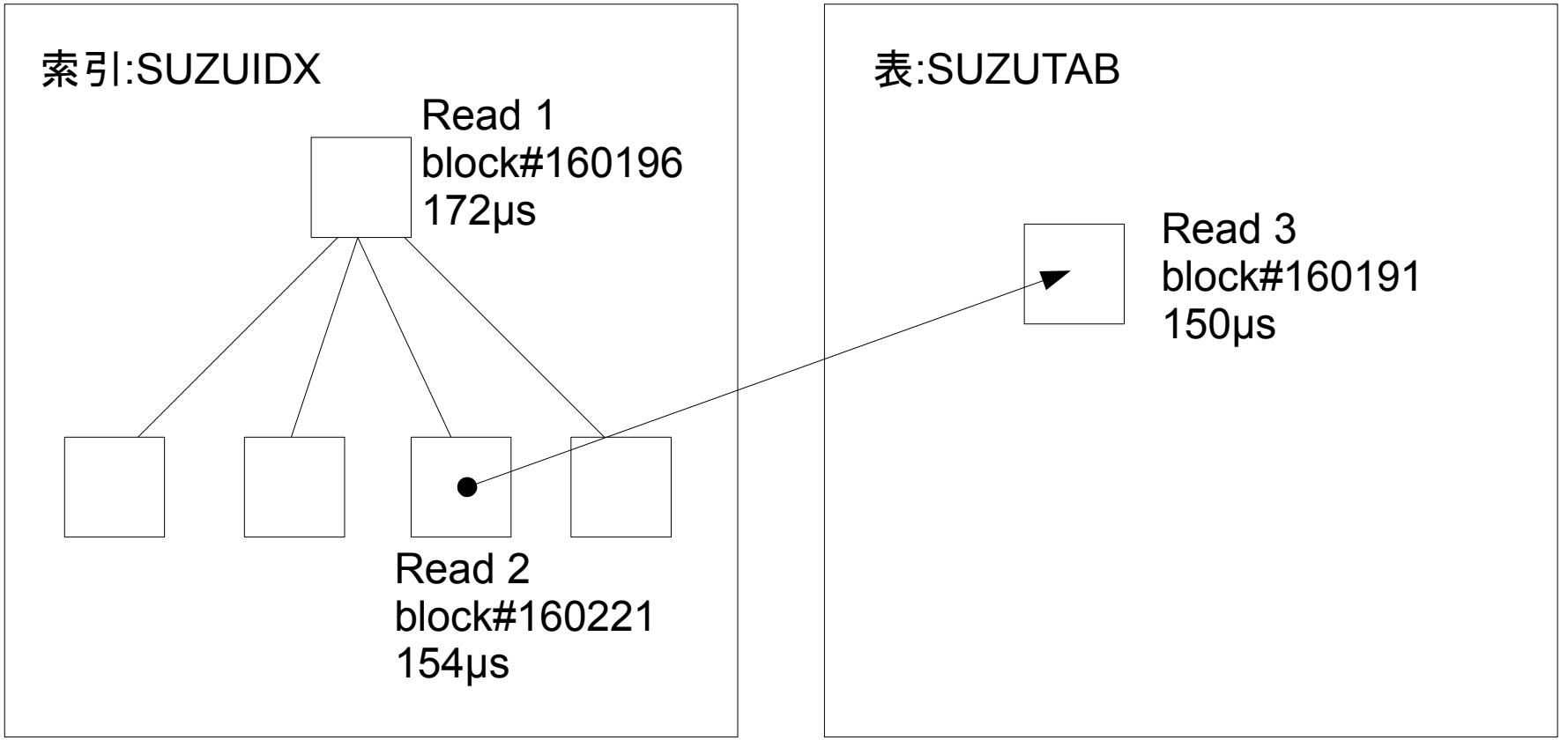
トレースファイルから、ファイル番号4のブロック番号160196, 160221, 160191を
読み込んでいることが分かった。これはどのオブジェクトだろう？

dba_extentsから確認

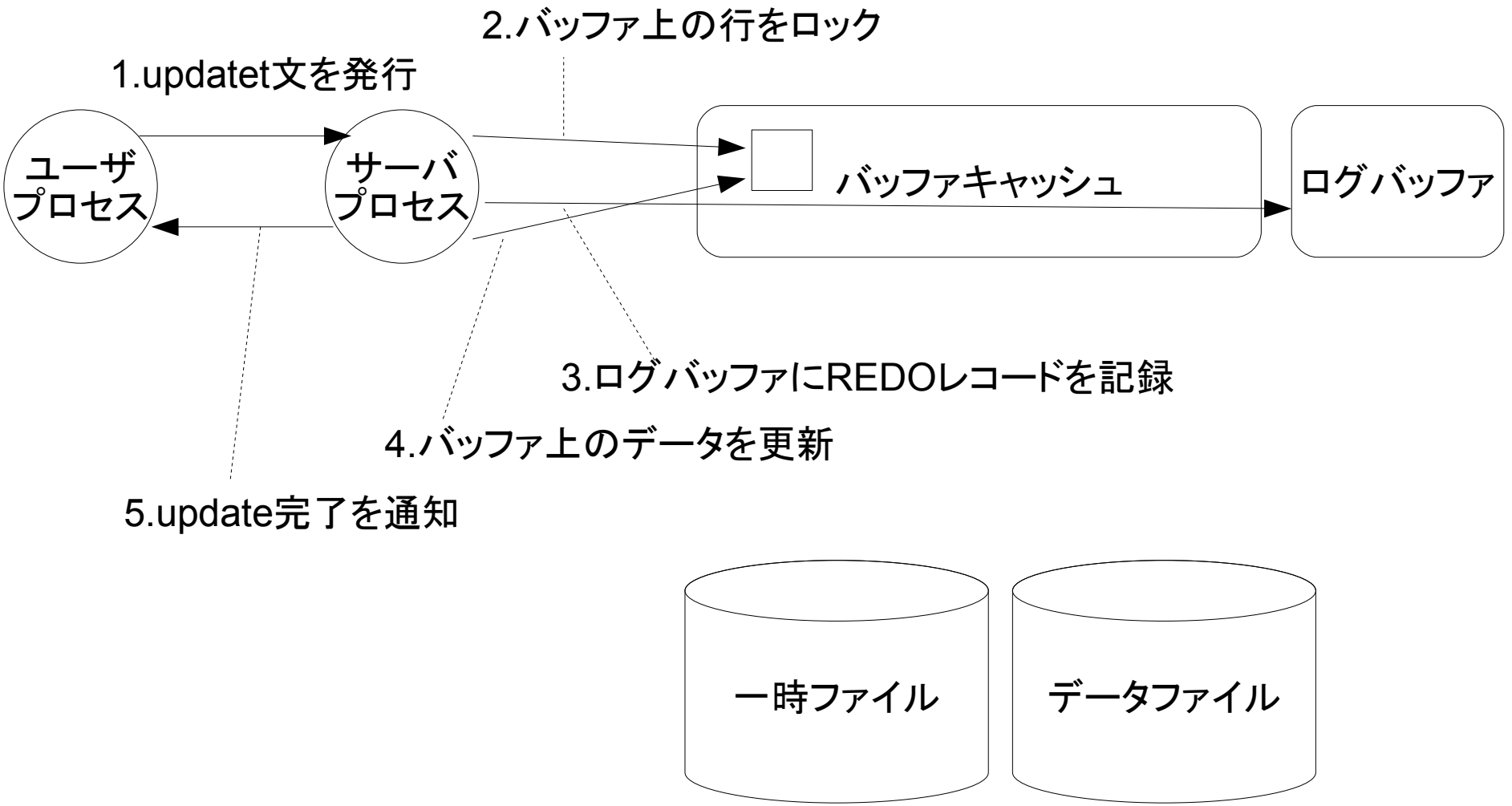
```
SQL> select segment_name, segment_type from dba_extents
  2  where file_id = 4 and 160196 between block_id and (block_id+blocks-1);

SEGMENT_NAME                   SEGMENT_TYPE
------------------------------ ------------------
SUZUIDX                        INDEX

SQL> select segment_name, segment_type from dba_extents
  2  where file_id = 4 and 160221 between block_id and (block_id+blocks-1);

SEGMENT_NAME                   SEGMENT_TYPE
------------------------------ ------------------
SUZUIDX                        INDEX

SQL> select segment_name, segment_type from dba_extents
  2  where file_id = 4 and 160191 between block_id and (block_id+blocks-1);

SEGMENT_NAME                   SEGMENT_TYPE
------------------------------ ------------------
SUZUTAB                        TABLE
```

索引経由の表アクセスをしていることが分かった

トレースを見て分かったこと(想像図)



UPDATE実行時の動作



バッファ上にデータがない場合は、SELECTと同じ動作でファイルから読み込む

UPDATE実行時の動作 - 発生する待機イベント

[よく出る]
enq: TX - row lock contention
　行ロック待ち。
　別のセッションと同一行を更新しようとしている場合がほとんど。
　レアなケースとしては、一意制約がついている列に重複した値を更新しようとしている
　ケースでも同じ待機イベントになる。

[時々出る]
buffer busy waits
　SELECT以外のDML同士では、同一ブロックに対するメモリ上の更新が
　同時に行われた場合に発生

[まれに出る]
log buffer space
　REDOログバッファに書き込もうとしたら、空きがなかった場合に発生する。
　LGWRプロセスがREDOログファイルに書き出しをして空きが出来るまで待機する。

enq: TXを見てみる。その1

●ケース１　通常の行競合

[セッション1]

```
SQL> update suzutab set col2=col2+1 where col1 = 1;

1 row updated.
```

[セッション2]

```
SQL> update suzutab set col2=col2+1 where col1 = 1;
```

=> 待機状態になる。

[セッション3]

```
SQL> select sid, username, event from v$session where username = 'SUZU';

       SID USERNAME                       EVENT
---------- ------------------------------ ------------------------------
       131 SUZU                           SQL*Net message from client
       142 SUZU                           enq: TX - row lock contention
```

確かに、enq: TXが発生する。

enq: TXを見てみる。その2

●ケース２　重複行の更新

[セッション1]

```
SQL> insert into suzutab (col1) values (10000);

1 row created.
```

(col1には一意制約がついている)

[セッション2]

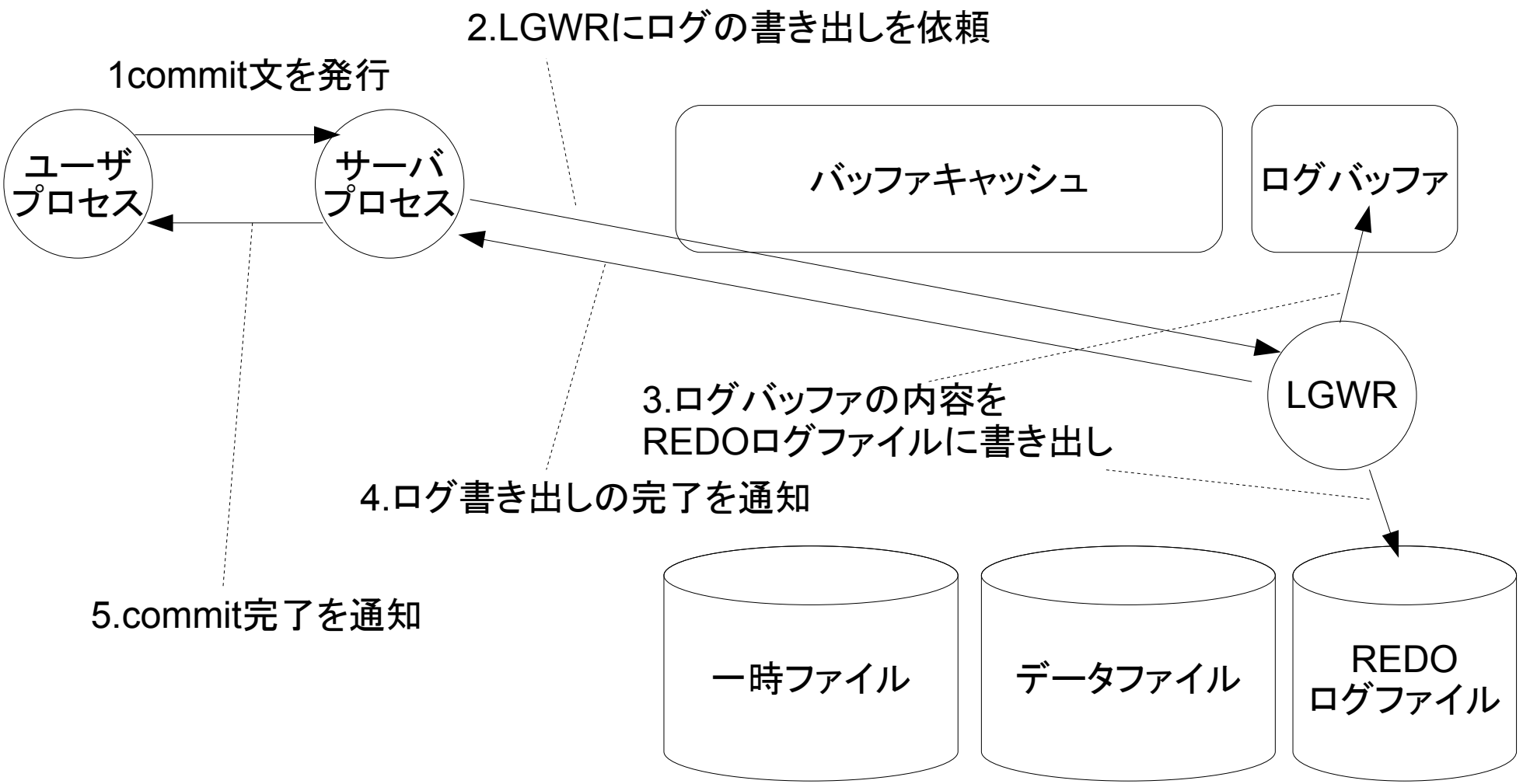
```
SQL> insert into suzutab (col1) values (10000);
```

=> セッション１がcommitされればエラーになるが、まだの場合待機状態になる。

[セッション3]

```
SQL> select sid, username, event from v$session where username = 'SUZU';

       SID USERNAME                       EVENT
---------- ------------------------------ ------------------------------
       131 SUZU                           SQL*Net message from client
       142 SUZU                           enq: TX - row lock contention
```

COMMIT実行時の動作



UPDATE実行時の動作 - 発生する待機イベント

[よく出る]
log file sync
  LGWRからの応答待ち
  ほとんどの場合、commitすると必ず発生する

その他の主な待機イベント

[アイドルイベント]
SQL*Net message from client
　ユーザプロセスからの応答待ち。要するに何もしていない。

[バックグラウンドプロセスのイベント]
db file parallel write
　DBWRによるデータファイルへの書き出し完了待ち
　基本的には、ユーザセッションと非同期に実施されるが、
　待機時間が長い場合には、影響を与える場合もある。

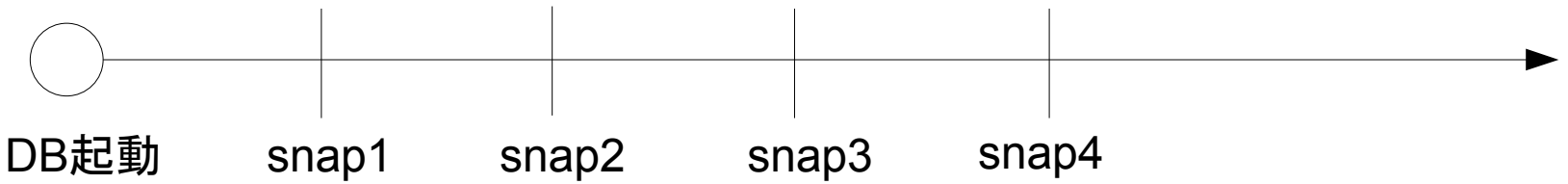
log file parallel write
　LGWRによるREDOログファイルへの書出し完了待ち
　待機時間が長いと、ユーザセッションでのlog file sync待ちが長くなる

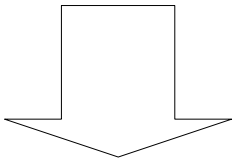
[SGA上の競合による待機]
library cache pin
　共有プール上のライブラリキャッシュに対するロック。
　SQLの解析が多い場合に発生する。

latch free
　SGA上の領域に対するロック解放待ち。
　小さい領域を短時間ロックしているのが普通。
　どの領域で競合しているかによって、原因が異なる。

AWR, Statspack, bstat/estatで見る待機イベント(1)

任意の２点で採取したスナップショット間の差分を整形して出力したもの
待機イベント以外にも様々な統計情報が出力される。





始点にsnap1, 終点にsnap3を指定

snap1からsnap3の期間のDB稼動状況のレポート出力

AWR, Statspack, bstat/estatで見る待機イベント(2)

Statspackレポートの待機イベントセクション

Top 5 Timed Events

Event	Waits	Time (s)	Avg wait (ms)	%Total Call Time
CPU time		51		86.5
db file sequential read	234	3	12	4.9
control file parallel write	1,206	2	1	2.8
control file sequential read	790	1	1	1.2
db file scattered read	59	1	9	0.9

終わりに

ORACLEの待機イベントについて説明しました。
パフォーマンスを解析するにあたって、待機イベントを知っておくと何かと便利です。