Blokady i obciążenie

Poziom: podstawy **Dotyczy: Oracle Database 19c** **Czas: ~20 min**

Zapytanie, które nagle wisi bez żadnego oczywistego powodu, prawie zawsze czeka na blokadę trzymaną przez inną sesję — a sesja zjadająca cały procesor rzadko zgłasza się sama, więc obie rzeczy sprawdzamy zapytaniem, nie domysłem.

Skróty użyte na tej stronie (3)

Skrót	Co znaczy
CPU	Central Processing Unit: procesor
RAC	Real Application Clusters: klaster Oracle, kilka serwerów na jedną bazę
WMS	Warehouse Management System: system zarządzania magazynem

Fiszka

TYLKO ODCZYT

Sprawdzenie blokujących sesji przez blocking_session

```
SELECT sid, serial#, username, blocking_session, status
FROM v$session
WHERE blocking_session IS NOT NULL;
```

Kolumna `blocking_session` w `v$session` to najszybsza droga do znalezienia, kto na kogo czeka — Oracle wylicza ją sam, bez potrzeby ręcznego łączenia `v$lock` z samą sobą. Prawidłowy wynik w zdrowej bazie to `no rows selected`.

Przykład:

```
       SID    SERIAL# USERNAME             BLOCKING_SESSION STATUS
---------- ---------- -------------------- ---------------- --------
       201       8832 REPORTS_USER                      142 ACTIVE
```

Zdarza się, że to zapytanie nie zwraca żadnych blokad, mimo że problem z blokowaniem faktycznie występuje — dzieje się tak, gdy sesja blokująca należy do innej instancji w konfiguracji RAC, albo gdy blokada jest typu, którego `blocking_session` nie wylicza wprost. W takich przypadkach sięgamy po pełniejszy, klasyczny łańcuch blokad niżej.

RAC = Real Application Clusters: klaster Oracle, kilka serwerów na jedną bazę

ZMIENIA SYSTEM

Łańcuch blokad przez v$lock

```
SELECT s1.username || ' (SID=' || s1.sid || ') is blocking ' ||
       s2.username || ' (SID=' || s2.sid || ')' AS blokada
FROM v$lock l1
JOIN v$session s1 ON s1.sid = l1.sid
JOIN v$lock l2 ON l1.id1 = l2.id1 AND l1.id2 = l2.id2 AND l2.request > 0
JOIN v$session s2 ON s2.sid = l2.sid
WHERE l1.block = 1;
```

Pojęcie: v$lock. Widok niskopoziomowy trzymający jeden wiersz na każdą blokadę trzymaną albo żądaną przez sesję — kolumna `block = 1` oznacza, że dana sesja blokuje kogoś innego, a `request > 0` oznacza, że dana sesja czeka na blokadę. Łączymy tabelę samą ze sobą po identyfikatorach zasobu (`id1`, `id2`), żeby sparować sesję trzymającą blokadę z sesją na nią czekającą.

Przykład:

```
BLOKADA
--------------------------------------------------------------------------------
APP_USER (SID=142) is blocking REPORTS_USER (SID=201)
```

SID = w Oracle: System Identifier, nazwa instancji; w Windows: Security Identifier, numer konta · **join** = połącz dwa pliki tekstowe po wspólnej kolumnie, jak złączenie tabel w bazie danych

TYLKO ODCZYT

Sprawdzenie obiektów zablokowanych przez daną sesję

```
SELECT b.sid, b.serial#, b.username, c.owner, c.object_name, c.object_type
FROM v$locked_object a, v$session b, dba_objects c
```

```
WHERE a.session_id = b.sid
  AND a.object_id = c.object_id
  AND a.session_id = NumerSID;
```

> **Pojęcie: v$locked_object.** Widok łączący blokadę z konkretnym obiektem bazy — tabelą, indeksem — zamiast z samym identyfikatorem zasobu jak `v$lock`. Odpowiadamy tym zapytaniem na pytanie „na jakich dokładnie tabelach ta sesja trzyma blokadę", nie tylko „czy kogoś blokuje".

Przykład:

```
       SID    SERIAL# USERNAME             OWNER      OBJECT_NAME          OBJECT_TYPE
---------- ---------- -------------------- ---------- -------------------- ------------
       142       5501 APP_USER             APP_OWNER  ZAMOWIENIA           TABLE
```

TYLKO ODCZYT

Sprawdzenie sesji zużywających najwięcej CPU

```
SELECT s.sid, s.serial#, s.username, s.program, s.status, ROUND(st.value/100) AS
cpu_usage_seconds
FROM v$session s, v$sesstat st, v$statname sn
WHERE s.sid = st.sid
  AND st.statistic# = sn.statistic#
  AND sn.name LIKE '%CPU used by this session%'
ORDER BY st.value DESC
FETCH FIRST 20 ROWS ONLY;
```

> **Pojęcie: v$sesstat i v$statname.** `v$sesstat` trzyma wartości liczników statystycznych dla każdej sesji, ale same liczniki są zakodowane numerycznie (`statistic#`) — `v$statname` tłumaczy ten numer na czytelną nazwę, tu akurat na `CPU used by this session`. Wartość jest wyrażona w setnych częściach sekundy, stąd dzielenie przez 100 w zapytaniu.

Przykład:

```
       SID    SERIAL# USERNAME             PROGRAM                          STATUS
CPU_USAGE_SECONDS
---------- ---------- -------------------- -------------------------------- --------
-----------------
       142       5501 APP_USER             JDBC Thin Client
ACTIVE                   842
       305         12 ZABBIX               zbx_monitor
ACTIVE                     4
```

Ta lista jest naturalnym punktem wyjścia, gdy monitoring albo `top` na poziomie systemu operacyjnego pokazuje wysokie zużycie CPU przez procesy Oracle (`oracleNazwaSID`), ale nie mówi wprost, która sesja logiczna za tym stoi — łączymy `spid` z wyniku `ps` z `v$process`/`v$session`, tak jak opisano na stronie Sesje: aktywne, nieaktywne, najdłuższe, albo idziemy od razu przez to zapytanie, jeśli podejrzewamy problem na poziomie samej bazy.

CPU = Central Processing Unit: procesor

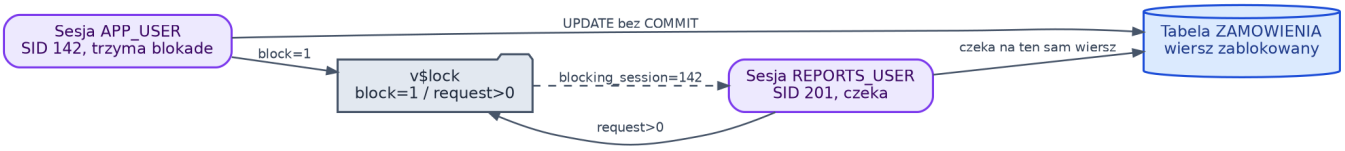
Z życia wzięte

Z ŻYCIA WZIĘTE

Kwadrans po starcie zmiany w Swarzędzu Marek pisze, że WMS zawiesza się przy zamykaniu zamówień. Tomek odpytuje `blocking_session` w `v$session` na WMSPRD i znajduje sesję REPORTS_USER czekającą na APP_USER, który od rana trzyma otwartą transakcję bez commitu. Po sprawdzeniu, że to nie jest legalna operacja wsadowa, zespół kończy sesję blokującą i kolejka zamówień odblokowuje się w kilka sekund.

Blokady w bazie

Pojęcie: blokada (lock). Mechanizm, przez który Oracle chroni spójność danych — sesja modyfikująca wiersz trzyma na nim blokadę wyłączną, dopóki nie wykona `COMMIT` albo `ROLLBACK`, a każda inna sesja próbująca zmodyfikować ten sam wiersz musi czekać. Ogólny mechanizm blokad, ich typy i eskalację opisuje strona Blokady: współdzielone, wyłączne, eskalacja — tu zajmujemy się tym, jak taką sytuację zdiagnozować w konkretnej instancji Oracle.



Sesja APP_USER trzyma blokadę na wierszu tabeli ZAMOWIENIA, sesja REPORTS_USER czeka na ten sam wiersz, a v$lock pozwala sparować obie strony blokady (kliknięcie otwiera pełny rozmiar)

Częste błędy

Poleganie wyłącznie na blocking_session. Kolumna działa dobrze dla zwykłych blokad wierszowych w pojedynczej instancji, ale nie pokrywa wszystkich przypadków (RAC, niektóre blokady systemowe) — jeśli podejrzewamy blokadę, a `blocking_session` milczy, sięgamy po pełne zapytanie na `v$lock`.

Zabicie sesji blokującej bez sprawdzenia, co robi. Sesja trzymająca blokadę często jest w środku długiej, ale legalnej transakcji wsadowej — `ALTER SYSTEM KILL SESSION` na niej cofnie całą pracę tej transakcji. Zanim zakończymy sesję blokującą, sprawdzamy jej zapytanie i czas trwania metodami ze strony Sesje: aktywne, nieaktywne, najdłuższe.

Traktowanie wysokiego CPU_USAGE_SECONDS jako wartości bieżącej. To licznik narastający od początku sesji, nie chwilowe zużycie — sesja otwarta od tygodnia z dużą wartością niekoniecznie zużywa dużo CPU teraz, tylko zdążyła nazbierać tę wartość w czasie.

Powiązane: Blokady: współdzielone, wyłączne, eskalacja · Sesje: aktywne, nieaktywne, najdłuższe · Zakleszczenie (deadlock): jak powstaje i jak je czytać z logu

Źródła: notatki własne, dokumentacja Oracle Database Reference 19c

---

Revision #42
Created 2026-09-24 23:34:18 UTC by Tomasz Konieczny
Updated 2026-10-26 14:08:49 UTC by Tomasz Konieczny