Oracle 12c Threaded Execution - Ressourcen sparen und Performance gewinnen - zum Nulltarif?

Markus Flechtner
Trivadis GmbH
Düsseldorf

Schlüsselworte
Oracle 12c, Prozessmodell, Thread, THREADED_EXECUTION

Einleitung

In der Dedicated-Server-Architektur von Oracle unter Unix oder Linux hat jede Datenbank-Session ihren eigenen Server-Prozess, der Server-Ressourcen (CPU und Hauptspeicher) benötigt. Mit der schon seit vielen Jahren vorhandenen Shared-Server-Architektur oder mit Connection Pools kann man dem entgegenwirken und die Serverlast reduzieren. Mit dem "Multithreaded Oracle Process Model" für Unix und Linux-Systeme bietet Oracle Database 12c eine neue Möglichkeit, um Datenbank-Sessions auf Betriebssystem-Ebene zusammenzufassen und Server-Ressourcen zu sparen. Datenban-Sessions werden als Threads innerhalb eines Betriebssystem-Prozesses ausgeführt.
Für Windows-Systeme ist "Threaded Execution" nicht relevant, denn bekanntermassen sind dort die Datenbank-Prozesse Threads innerhalb des "oracle.exe".

Konfiguration

Die Konfiguration erfolgt über den statischen Instanz-Parameter THREADED_EXECUTION.

```
ALTER SYSTEM SET THREADED_EXECUTION=TRUE SCOPE=SPFILE;
```

Nach einem Neustart der Instanz läuft die Instanz dann im "threaded Modus". In dieser Konfiguration werden allerdings nur Datenbank-Sessions, die direkt auf dem Server gestartet werden, als Threads ausgeführt. Damit auch Client-Prozesse als Threads ausgeführt werden, muss noch ein Listener Parameter gesetzt werden:

```
DEDICATED_THROUGH_BROKER_<LISTENER_NAME>=ON
```

Der Listener muss nach dieser Änderung durchgestartet werden. Der Parameter sorgt dafür, dass der Listener bei einer Verbindungsanfrage keinen neuen Server-Prozess für eine dedizierte Datenbank-Session startet, sondern dass er die Verbindungsanfrage an einen Broker-Prozess der Datenbank-Instanz (genauer: an einen Broker-Thread namens N000) weiterleitet, der die Anfrage dann bearbeitet.

Architektur

Nach dem Start präsentiert sich eine Instanz im "threaded Modus" mit einer deutlich reduzierten Prozess-Liste. Im Gegensatz zum "nonthreaded Modus" mit mehr als 40 Prozessen in der Normalkonfiguration sind es jetzt nur noch sechs Hintergrundprozesse:

```
oracle@si:~/ [SE12C] ps -ef |grep SE12C
oracle    5704     1  0 22:38 ?        00:00:00 ora_pmon_SE12C
oracle    5706     1  0 22:38 ?        00:00:00 ora_psp0_SE12C
oracle    5708     1  4 22:38 ?        00:00:01 ora_vktm_SE12C
oracle    5712     1  2 22:38 ?        00:00:00 ora_u004_SE12C
```

```
oracle    5718     1 27 22:38 ?        00:00:05 ora_u005_SE12C
oracle    5724     1  0 22:38 ?        00:00:00 ora_dbw0_SE12C
```

Wir sehen, dass nur noch wenige der bekannten Datenbank-Hintergrundprozesse als Prozesse ausgeführt werden, nämlich PMON, PSP0, VKTM und DBW0. Alle anderen Hintergrundprozesse sind Threads innerhalb der uNNN-Prozesse:

```
oracle@si:~/ [SE12C] pidstat -t -p 5712
Linux 3.8.13-35.3.4.el6uek.x86_64 (doagdemo)  08/25/2014  _x86_64_  (2 CPU)
 TGID      TID    %usr %system  %guest    %CPU   CPU  Command
 5712        -    0.06    0.14    0.00    0.20     0  ora_scmn_se12c
   -      5712    0.00    0.00    0.00    0.00     0  |__ora_scmn_se12c
   -      5713    0.00    0.00    0.00    0.00     0  |__oracle
   -      5714    0.00    0.00    0.00    0.00     1  |__ora_gen0_se12c
   -      5715    0.02    0.00    0.00    0.02     1  |__ora_mman_se12c
   -      5721    0.01    0.00    0.00    0.01     0  |__ora_dbrm_se12c
   -      5725    0.00    0.01    0.00    0.02     0  |__ora_lgwr_se12c
   -      5726    0.00    0.01    0.00    0.01     0  |__ora_ckpt_se12c
   -      5727    0.00    0.04    0.00    0.04     1  |__ora_lg00_se12c
   -      5728    0.00    0.00    0.00    0.00     0  |__ora_lg01_se12c
   -      5729    0.00    0.00    0.00    0.00     0  |__ora_smon_se12c
   -      5731    0.00    0.00    0.00    0.00     0  |__ora_lreg_se12c
```

Anmerkung: Ausgabe gekürzt

Interessanterweise ist auch der LogWriter-Prozess (LGWR) im Multithreaded Modell als Thread realisiert.
Innerhalb eines uNNN-Prozesses fungiert der SCMN-Thread als Koordinator der Threads.
Bei steigender Anzahl von Datenbank-Sessions werden uNNN-Prozesse für die entsprechenden Threads nachgestartet. Auch bei 800 Datenbank-Sessions waren dabei in den Tests nie mehr als sechs uNNN-Prozesse zu beobachten, d.h. mehr als 100 Sessions können von Oracle in einem Betriebssystem-Prozess zusammengefasst werden. Wenn ein uNNN-Prozess nicht mehr benötigt wird, dann wird er nach 30 Sekunden automatisch beendet.

In der View V$PROCESS gibt es zwei neue Spalten, STID (Thread-ID) und EXECUTION_TYPE, (Thread oder Process), die Auskunft darüber geben, wie der jeweilige Oracle-Prozess ausgeführt wird:

```
SQL> select spid,stid,execution_type,pname from v$process
  2  where execution_type='PROCESS' or
  3  pname in ('SMON','LGWR','CKPT','VKTM','PMON','DBW0','MMON','RECO')
  4  order by spid;
SPID            STID            EXECUTION_ PNAME
--------------- --------------- ---------- -----
5704            5704            PROCESS    PMON
5706            5706            PROCESS    PSP0
5708            5708            PROCESS    VKTM
5712            5725            THREAD     LGWR
5712            5729            THREAD     SMON
5712            5726            THREAD     CKPT
5718            5730            THREAD     RECO
5718            5732            THREAD     MMON
5724            5724            PROCESS    DBW0
```

Dabei ist zu beachten, dass alle Datenbank-Sessions "Dedicated Server"-Sessions sind und dass der Instanz-Parameter PROCESSES nicht heruntergesetzt werden darf. Aus Sicht der Datenbank-Instanz benötigt jede Session eine "Prozess-Slot". Ist der Parameter PROCESSES zu niedrig gesetzt, dann kann es bei einer zu großen Anzahl von Client-Sessions zur Fehlermeldung "TNS-12602 – Connection pooling Limit reached." kommen.

Analog den Shared-Server-Prozessen allokieren auch die Prozesse bzw. Threads im Multithreaded Modell ihre Process Global Area (PGA) im Shared Pool der Datenbank-Instanz.

Ressourcenverbrauch
Für die Datenbank-Sessions ergibt sich ein geringerer Overhead auf Betriebssystemebene, weil der Prozess-Overhead für die einzelne Datenbank-Session entfällt. In den Tests wurde ein bis zu 18% geringerer Hauptspeicherbedarf beobachtet.
Durch die reduzierte Anzahl von Betriebssystem-Prozessen muss das Betriebssystem weniger Context-Switche ausführen.

Performance
Auf der Oracle Open World 2012 (Vortrag "Maximizing Database Performance Using Database Replay") wurde von einer Performance-Verbesserung von etwa 6% bei der Verwendung von "Threaded Execution" berichtet. Thomas Bordeau (http://blog.arkzoyd.com/2014/01/17/oracle-multithreaded-does-it-worth-a-try) hat in einem Test mit dem bekannten Tool Swingbench (siehe http://www.dominicgiles.com) bessere Skalierbarkeit (aufgrund des geringeren Ressourcenverbrauchs) und eine bessere Performance bei einer großen Anzahl von Sessions ermittelt. Diese generellen Aussagen konnten wir in unseren Tests bestätigen: im Mittel ergab sich eine Performanceverbesserung im einstelligen Prozentbereich und eine bessere Skalierbarkeit.

Threaded Execution im Alltag des Datenbank-Administrators
Im DBA-Alltag ist "Threaded Execution" mit zwei Nachteilen verbunden:

1. Anmeldung mit Authentisierung über das Betriebssystem ("sqlplus / as sysdba") ist nicht mehr möglich.
2. Das Beenden einzelner Datenbank-Sessions über "kill -9 <server_prozess_id>" ist nicht mehr möglich.

Der zweite Punkt ist sicher schwerwiegender, denn das Killen einer hängenden Datenbank-Session, die sich auf andere Art und Weise nicht beenden lässt, auf Betriebssystemebene, ist in vielen Fällen die "ultima ratio". Im Threaded Execution-Modell bewirkt das Killen eines "uNNN"-Prozesses aber das Ende für viele Datenbank-Sessions und ggf. sogar einen Absturz der Datenbank-Instanz

Die fehlende Authentisierung über das Betriebssystem wird lt. Oracle 12.1.0.2-Readme Oracle-seitig als Bug eingestuft (unpublished bug 13877504). Ein Workaround ist es, bei der Anmeldung Benutzernamen und Password anzugeben ("sqlplus sys as sysdba"), d.h. eine Password-Datei ist erforderlich. Um sich ohne Password an der Datenbank anmelden zu können, kann das Password in einem Wallet gespeichert werden (Einzelheiten siehe MOS-Note 340559.1 "Using The Secure External Password Store"):

1. Wallet anlegen
```
mkstore -wrl <wallet_location> -create
```

2. Datenbank-Password eintragen

```
mkstore -wrl <wallet_location> -createCredential <db_connect_string>
<username> <password>
```

3. Datei sqlnet.ora ergänzen

```
WALLET_LOCATION =
   (SOURCE =
      (METHOD = FILE)
      (METHOD_DATA = (DIRECTORY = <wallet_location_directory>))
)
SQLNET.WALLET_OVERRIDE = TRUE
```

Zusätzlich ist zu beachten, dass der Database Configuration Assistant (dbca) und opatch (genauer: datapatch.pl) mit OS-Authentisierung arbeiten und dass daher vor Verwendung dieser Tools THREADED_EXECUTION deaktiviert werden muss. Auch müssen sicher die meisten Startup- und Shutdown-Skripte angepasst werden, da sie meist mit OS-Authentisierung arbeiten.

Für das SQL Tracing (ALTER SESSION SET SQL_TRACE=TRUE) gibt keine Änderungen: Die Thread-ID wird an den Dateinamen angefügt:

```
SQL> select spid,stid,tracefile from v$process where
execution_type='THREAD' order by spid,stid;
SPID   STID   TRACEFILE
------ ------ ------------------------------------------------------------
5712   5712
/u00/app/oracle/diag/rdbms/se12c/SE12C/trace/SE12C_scmn_5712_5712.trc
5712   5714
/u00/app/oracle/diag/rdbms/se12c/SE12C/trace/SE12C_gen0_5712_5714.trc
[..]
```

Fazit

Die fehlende OS-Authentisierung und die fehlende Möglichkeit einzelne Datenbank-Sessions auf Betriebssystem-Ebene sind der "Preis", den man für "Threaded Execution" zahlen muss. Wenn man dazu bereit ist, dann erhält man eine bessere Ausnutzung der System-Ressourcen und eine leicht bessere Performance. Und das zum "Nulltarif", denn "Threaded Execution" funktioniert auch bei der günstigsten Variante der Oracle Datenbank, der "Standard Edition One". Also: "give it a try"!

Kontaktadresse:
Markus Flechtner
Trivadis GmbH
Werdener Straße 4
D-40227 Düsseldorf

Telefon: +49 (0) 211 5866 6470
Fax: +49 (0) 211 5866 6471
E-Mail markus.flechtner@trivadis.com
Internet: www.trivadis.com