[ORACLE会话连接进程三者总结](http://www.cnblogs.com/kerrycode/p/4100865.html)

2014-11-16 00:15 by 潇湘隐者, 11768 阅读, 0 评论, [收藏](http://www.cnblogs.com/kerrycode/p/4100865.html), [编辑](https://i.cnblogs.com/EditPosts.aspx?postid=4100865)

http://www.cnblogs.com/kerrycode/p/4100865.html

**概念介绍**

     通俗来讲，会话(Session) 是通信双方从开始通信到通信结束期间的一个上下文（Context）。这个上下文是一段位于服务器端的内存：记录了本次连接的客户端机器、通过哪个应用程序、哪个用户登录等信息.

     连接（Connection）：连接是从客户端到ORACLE实例的一条物理路径。连接可以在网络上建立，或者在本机通过IPC机制建立。通常会在客户端进程与一个专用服务器或一个调度器之间建立连接。

     会话(Session) 是和连接(Connection)是同时建立的，两者是对同一件事情不同层次的描述。简单讲，连接(Connection)是物理上的客户端同服务器的通信链路，会话(Session)是逻辑上的用户同服务器的通信交互。

     ORACLE中一个用户登录ORACLE服务器的前提，就是该用户具有ORACLE的 “CREATE SESSION”权限。ORACE允许同一个用户在同一个客户机上建立多个同服务器的会话，每个SESSION都代表了用户与服务器的一个交互。就像你用IE浏览器打开博客园网站，然后你再打开一个IE窗口，又打开一个博客园网站。两个IE窗口就相当于两个SESSION， 而物理链路就相当于连接（Connection）。后台进程PMON会每隔一段时间，就会检测用户连接状况，如果连接已断开，PMON会清理现场，释放相关的资源。

     在一条连接上可以建立0个、一个或多个会话。各个会话是单独而且独立的，即使它们共享同一条数据库物理连接也是如此。一个会话中的提交不会影响该连接上的任何其他会话。实际上，还可以有连接而无相应的会话。另外，一个会话可以有连接也可以没有连接。使用高级Oracle Net特性（如连接池）时，客户可以删除一条物理连接，而会话依然保留（但是会话会空闲）。客户在这个会话上执行某个操作时，它会重新建立物理连接。

    在专用服务器中，一个会话对应一个服务器进程（Process),如果数据库运行在共享服务器方式，一个服务器进程可以为多个会话服务。

下面是一段关于连接（connection）、会话、进程的英文描述

A connection is a physical circuit between you and the database. A connection might be one of many types -- most popular begin DEDICATED server and SHARED server. Zero, one or more sessions may be established over a given connection to the database as show above with sqlplus. A process will be used by a session to execute statements. Sometimes there is a one to one relationship between CONNECTION->SESSION->PROCESS (eg: a normal dedicated server connection). Sometimes there is a one to many from connection to sessions (eg: like autotrace, one connection, two sessions, one process).

A process does not have to be dedicated to a specific connection or session however, for example when using shared server (MTS), your SESSION will grab a process from a pool of processes in order to execute a statement. When the call is over, that process is released back to the pool of processes.

**会话&链接**

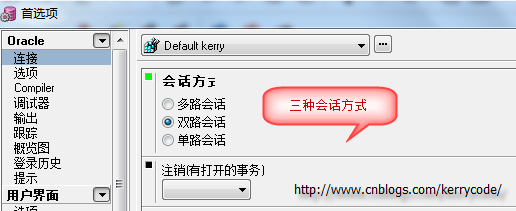
在具体的应用场景中连接(connction) 和 会话(session) 有很多情况：

1. SQL\*PLUS 登录 ORACLE

这种场景比较容易理解，一个连接对应一个Session。

2. PL/SQL Developer工具登录ORACLE

PL/SQL Developer工具——>首选项——>连接下，你可以设置会话方式，如下图所示：

[](http://images.cnitblog.com/blog/73542/201411/160029403907308.png)

如果设置选项选择多路会话，：PL/SQL Developer 登录ORACLE，每打开一个窗口，将创建一个新的会话，而设置选项选择单路会话，则新打开的窗口会共用一个会话。具体你可以参考PLSQL Developer8.0用户指南：

多路会话：每个测试窗口、SQL窗口和命令窗口都将有它自己的会话，另外的一个会话将被用于编译。这是最灵活的设置，明显地会导致最大数量的数据库会话。另外可能的缺点是，在更新被提交之后，它们只在X窗口中可以看到，而在Y窗口看不到。

双路会话：测试窗口、SQL窗口和命令窗口将共享一个会话，另外一个会话将被用于编译。这个模式的缺点是每次只有一个窗口可以运行程序。

单路会话：所有的窗口和所有的编译都使用同一个会话，这使事务管理变得很困难。在这个模式中调试器被禁用使用。如果你被限制只能使用一个数据库会话，那么你只能使用这个设置了。

**会话&进程**

在Oracle中如何查看参数sessiones或processes的值呢？一般使用show parameter命令查看。

SQL> show parameter processes;

NAME                                 TYPE                             VALUE

------------------------------------ -------------------------------- ---------------

aq\_tm\_processes                      integer                          0

db\_writer\_processes                  integer                          1

gcs\_server\_processes                 integer                          0

job\_queue\_processes                  integer                          10

log\_archive\_max\_processes            integer                          2

processes                            integer                          850

SQL> show parameter session;

NAME                                 TYPE                             VALUE

------------------------------------ -------------------------------- -----------

java\_max\_sessionspace\_size           integer                          0

java\_soft\_sessionspace\_limit         integer                          0

license\_max\_sessions                 integer                          0

license\_sessions\_warning             integer                          0

logmnr\_max\_persistent\_sessions       integer                          1

session\_cached\_cursors               integer                          20

session\_max\_open\_files               integer                          10

sessions                             integer                          940

shared\_server\_sessions               integer

SQL>

方法2：查询v$parameter

select name, type, value ,display\_value, isses\_modifiable, issys\_modifiable

from v$parameter

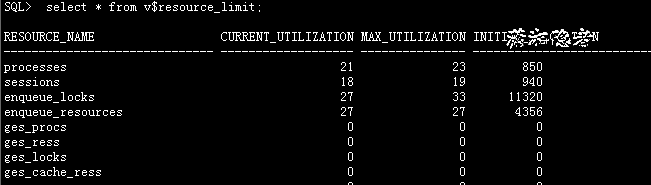
 where name='sessions';

select name, type, value ,display\_value, isses\_modifiable, issys\_modifiable

from v$parameter

 where name='processes';

方法3：查询v$resiyrce\_limit;

[](http://images.cnitblog.com/blog/73542/201411/160029412658407.png)

Oracle的sessions和processes的数量关系是：

**Oracle 11g R1以及之前版本**

             sessions=(1.1 \* processes) + 5

**Oracle 11g R2**

             sessions=(1.5 \* processes) + 22

如下例子所示，在Oracle 10g 版本中，processes与sessions的关系如下所示：

SQL> select \* from v$version;

BANNER

----------------------------------------------------------------

Oracle Database 10g Release 10.2.0.4.0 - 64bit Production

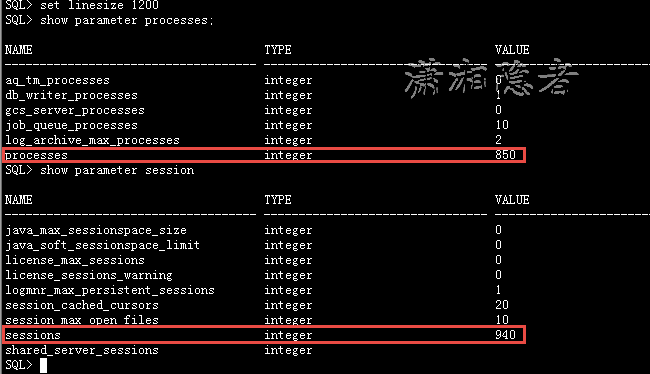
PL/SQL Release 10.2.0.4.0 - Production

CORE    10.2.0.4.0      Production

TNS for Linux: Version 10.2.0.4.0 - Production

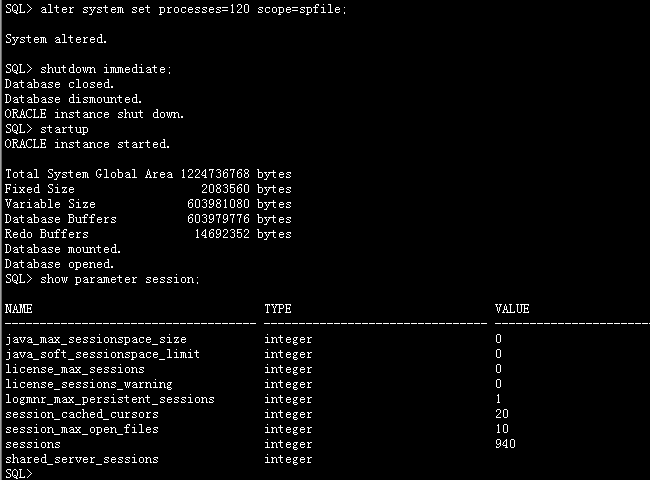
NLSRTL Version 10.2.0.4.0 - Production

SELECT (1.1 \*850)+ 5 FROM DUAL; 其值刚好为940

[](http://images.cnitblog.com/blog/73542/201411/160029423439977.png)

一般修改参数processes后，sessions参数也会随之变化，但是有一个奇怪的现象时，如下所示，我将processes从

850改为120后，重启数据库实例，发现sessions的值并没有随之变化。这个现象一般发生在改小processes参数。为什么这样呢？

[](http://images.cnitblog.com/blog/73542/201411/160029435628505.png)

[](http://images.cnitblog.com/blog/73542/201411/160029449535748.png)

共享服务器模式，一个会话可能由多个服务进程轮流为之服务，一个进程可能为多个会话服务。简单地说，进程和会话之间有一种多对多的关系。

**会话管理**

1：查看当前所有用户的会话(SESSION)：

SELECT \* FROM V$SESSION

WHERE USERNAME IS NOT NULL

ORDER BY LOGON\_TIME , SID;

其中Oracle内部进程的USERNAME为空

2：查看当前用户的所有SESSION：

SELECT \* FROM V$SESSION

WHERE USERNAME = USER

ORDER BY LOGON\_TIME, SID;

3：查看当前窗口/当前用户的会话信息

SELECT SID, SERIAL#, STATUS FROM V$SESSION WHERE AUDSID=USERENV('SESSIONID');

4：查看所有ACTIVE会话(活动会话)

SELECT \* FROM V$SESSION

WHERE USERNAME IS NOT NULL AND STATUS='ACTIVE'

ORDER BY LOGON\_TIME, SID;

5：查看当前会话的ID可以通过如下脚本:

SELECT \* FROM V$MYSTAT WHERE ROWNUM =1

查看当前用户的SPID

SELECT P.SPID, S.SID, S.SERIAL#

FROM V$PROCESS P

INNER JOIN V$SESSION S ON P.ADDR = S.PADDR

WHERE S.AUDSID=USERENV('SESSIONID');

6：查看数据库允许最大会话数

SQL> SHOW PARAMETER SESSIONS;

NAME                                 TYPE        VALUE

------------------------------------ ----------- ------------------------------

java\_max\_sessionspace\_size           integer     0

java\_soft\_sessionspace\_limit         integer     0

license\_max\_sessions                 integer     0

license\_sessions\_warning             integer     0

logmnr\_max\_persistent\_sessions       integer     1

sessions                             integer     225

shared\_server\_sessions               integer

SQL> SELECT NAME, TYPE, VALUE FROM V$PARAMETER WHERE NAME LIKE 'session%';

NAME                       TYPE                 VALUE

---------                 ----------             -----------

sessions                          3                225

session\_cached\_cursors            3                20

session\_max\_open\_files            3                10

7：查看曾经的最大会话数：

SQL>

SQL> SELECT SESSIONS\_MAX,SESSIONS\_WARNING,SESSIONS\_CURRENT,SESSIONS\_HIGHWATER

  2  FROM v$license;

SESSIONS\_MAX SESSIONS\_WARNING SESSIONS\_CURRENT SESSIONS\_HIGHWATER

------------ ---------------- ---------------- ------------------

           0                0              512                553

SESSIONS\_HIGHWATER表示曾经的最大会话数512

8：查询那些应用的连接数此时是多少

SELECT  B.PROGRAM , COUNT(1)

FROM V$PROCESS A, V$SESSION B

WHERE A.ADDR = B.PADDR

      AND  B.USERNAME IS NOT NULL

GROUP BY B.PROGRAM;

**会话状态：**

会话有ACTIVE、INACTIVE、KILLED、CACHED、SNIPED五个状态，一般比较常见的有ACTIVE、INACTIVE、KILLED三个状态。

ACTIVE   ：处于此状态的会话，表示正在执行，处于活动状态。

INACTIVE ：处于此状态的会话表示不是正在执行的

KILLED   ：处于此状态的会话，表示出现了错误或进程被杀掉，正在回滚，当然，这个状态的会话也占用系统资源的。还有一点就是，    KILLED的状态一般会持续较长时间，如果你想快速杀掉回话，可以参考我以前的一篇文章[ORACLE快速彻底Kill掉的会话](http://www.cnblogs.com/kerrycode/p/4034231.html" \t "_blank)

CACHED   ： Session temporarily cached for use by Oracle\*XA

SNIPED   ： Session inactive, waiting on the client。 标记为SNIPED的进程被释放有两种条件：

         1、相关的terminal再一次试图登录及执行sql

         2、手动的在操作系统后台kill掉相应的spid

关于会话信息

通过如下SQL你可以查询你的每个应用程序到底在等待什么，从而针对这些信息对数据库的性能进行调整。

COL USERNAME FOR A12;

COL PROGRAM  FOR A32;

COL EVENT    FOR A26;

SELECT S.USERNAME

      ,S.PROGRAM

      ,S.STATUS

      ,SE.EVENT

      ,SE.TOTAL\_WAITS

      ,SE.TOTAL\_TIMEOUTS

      ,SE.TIME\_WAITED

      ,SE.AVERAGE\_WAIT

FROM V$SESSION S, V$SESSION\_EVENT SE

WHERE S.SID=SE.SID AND SE.EVENT NOT LIKE 'SQL\*Net%'

  AND S.STATUS ='ACTIVE' AND S.USERNAME IS NOT NULL;

2.ORACLE中查询被锁的表并释放session

SELECT A.OWNER

  ,A.OBJECT\_NAME

  ,B.XIDUSN

  ,B.XIDSLOT

  ,B.XIDSQN

  ,B.SESSION\_ID

  ,B.ORACLE\_USERNAME

  ,B.OS\_USER\_NAME

  ,B.PROCESS

  ,B.LOCKED\_MODE

  ,C.MACHINE

  ,C.STATUS

  ,C.SERVER

  ,C.SID

  ,C.SERIAL#

  ,C.PROGRAM

FROM ALL\_OBJECTS A,V$LOCKED\_OBJECT B,SYS.GV\_$SESSION C

WHERE  A.OBJECT\_ID = B.OBJECT\_ID  AND B.PROCESS = C.PROCESS  ORDER BY 1,2;

3.查看占用系统IO较大的session

SELECT se.sid

      ,se.serial#

      ,pr.spid

      ,se.username

      ,se.status

      ,se.terminal

      ,se.program

      ,se.module

      ,se.sql\_address

      ,st.event

      ,st.p1text

      ,si.physical\_reads

      ,si.block\_changes

FROM v$session se,v$session\_wait st,v$sess\_io si,v$process pr

WHERE st.sid=se.sid  AND st.sid=si.sid

  AND se.paddr=pr.ADDR AND se.sid>6

  AND st.wait\_time=0 AND st.event NOT LIKE '%SQL%'

  ORDER BY physical\_reads DESC;

4.找出耗cpu较多的session

select a.sid

      ,spid

      ,status

      ,substr(a.program,1,40) prog

      ,a.terminal

      ,osuser

      ,value/60/100 value

from v$session a,v$process b,v$sesstat c

where c.statistic#=12 and c.sid=a.sid and a.paddr=b.addr

   order by value desc

**参考资料：**

<http://www.linuxidc.com/Linux/2011-06/37178.htm>

<http://shaharear.blogspot.com/2008/05/oracle-session-resource-profiler.html>

<http://www.linuxidc.com/Linux/2013-05/84924.htm>