Instance Caging

Por Francisco Riccio

Introducción

Instace Caging es una característica que Oracle Database 11g Release 2 nos ofrece y nos da la posibilidad de limitar el número de cores que serán utilizadas por una instancia en un servidor.

Su implementación solo está disponible en una edición Enterprise Edition.

Implementación

Paso 1: Determinar la cantidad de cores disponibles en el servidor.

select value from v$osstat where stat_name = 'NUM_CPUS';

```
SQL> select value from v$osstat where stat_name = 'NUM_CPUS';

     VALUE
----------
         4
```

Paso 2: Configurar Resource Manager en cada instancia de Base de Datos en los que se desea configurar Instance Caging. En mi escenario utilizaré el plan DEFAULT_PLAN ya creado por default en un base de datos Oracle.

```
SQL> alter system set resource_manager_plan='DEFAULT_PLAN';

System altered.
```

Debemos recordar que el plan DEFAULT_PLAN tiene las siguientes directivas:

PLAN: DEFAUL_PLAN	Consumo de CPU		
Grupos Consumidores	Nivel 1	Nivel 2	Nivel 3
SYS_GROUP	100%	0%	0%
OTHER_GROUPS	0%	90%	0%
ORA$AUTOTASK_SUB_PLAN	0%	5%	0%
ORA$DIAGNOSTICS	0%	5%	0%

Paso 3: Configurar el parámetro CPU_COUNT de cada instancia a la cantidad de cores que deseamos que se utilicen del servidor.

```
SQL> alter system set cpu_count=2;

System altered.
```

Aquí podemos tener dos diferentes escenarios

a) Partitioning approach: En está configuración la suma del valor del parámetro CPU_COUNT en todas las instancias no llega a ser un valor superior a la cantidad de cores que tiene asignado el servidor. La ventaja de esta configuración es que cada instancia tiene sus cores reservados de forma independiente y en caso una instancia estuviera demandando altos consumos de CPU no interfiere con los cores asignados de las otras instancias. La desventaja es que si una instancia está en IDLE estaríamos desaprovechando el uso de los asignados cores. Este tipo de configuración es ideal para ambientes productivos.

b) Over-provisioning approach: En está configuración la suma del valor del parámetro CPU_COUNT en todas las instancias llegan a ser un valor superior a la cantidad total de cores que tiene asignado el servidor. Si en caso todas las instancias necesitaran el uso del 100% de los cores asignados mediante el parámetro CPU_COUNT, cada una de ellas estarían compitiendo por el uso de cores del servidor al no estar asignados de forma independiente. Este tipo de configuración es ideal para ambientes de desarrollo donde normalmente no todas las instancias requieren usar al 100% los recursos de CPU asignados al mismo tiempo.

Paso 4: Validación

a) Validaremos el uso de Instance Caging:

select instance_caging from v$rsrc_plan where is_top_plan = 'TRUE';

```
SQL> select instance_caging from v$rsrc_plan where is_top_plan = 'TRUE';

INS
---
ON
```

Debe devolver el valor de ON.

b) Monitorearemos el uso de cores mediante Instance Caging:

select to_char(begin_time, 'HH24:MI') time, sum(avg_running_sessions) avg_running_sessions, sum(avg_waiting_sessions) avg_waiting_sessions from v$rsrcmgrmetric_history group by begin_time order by begin_time;

```
SQL> select to_char(begin_time, 'HH24:MI') time, sum(avg_running_sessions) avg_running_sessions,
sum(avg_waiting_sessions) avg_waiting_sessions
from v$rsrcmgrmetric_history
group by begin_time order by begin_time;

TIME  AVG_RUNNING_SESSIONS AVG_WAITING_SESSIONS
----- -------------------- --------------------
00:21           .000483333                    0
00:22           .000433333                    0
00:23           .000716667                    0
00:24           .000483333                    0
00:25                .0008                    0
00:26              .00065                    0
00:27              .00125                    0
00:28               .0005                    0
00:29           .001616667                    0
00:30           .000516667                    0
00:31           .000533333                    0
```

El campo avg_running_sessions es el promedio de sesiones por minuto utilizando cores, si su valor es mucho menor que el valor del parámetro CPU_COUNT significa que la instancia no está utilizando al 100% los cores asignados y podemos reducirlo.

El campo avg_waiting_sessions es el promedio de sesiones por minuto esperando por el uso de cores libres, si su valor es mayor a 0 podemos mejorar la performance incrementando el valor del parámetro CPU_COUNT.

Nota 1: CPU_COUNT es un parámetro dinámico por lo cual se puede cambiar en cualquier momento y su efecto toma en pocos segundos. Cabe mencionar que Oracle no recomienda el cambio de un valor muy bajo a un valor muy alto para el parámetro CPU_COUNT y asimismo no recomienda configurar con el valor de 1.

Nota 2: Instance Caging no influye en las reglas de licenciamiento con que cuenta el servidor.

Nota 3: Instance Caging y Resource Manager tienen algunos parches recomendados para evitar bugs reportados, por lo cual recomiendo la revisión de las siguientes documentaciones:

a) My Oracle Support (MOS) Nota: 1208064.1 (Recommended Patches for Instance Caging).

b) My Oracle Support (MOS) Nota: 1208133.1 (Recommended Patches for Managing Runaway Queries with Resource Manager).

c) My Oracle Support (MOS) Nota: 1339803.1 (Recommended Patches for CPU Resource Manager).

Escenario de validación

Un servidor que cuenta con 4 cores tiene una instancia de base de datos Oracle el cual tiene asignado los 4 cores.

Se ejecuta mediante 4 sesiones el siguiente código PLSQL:

```
set serveroutput on

declare

 n number:=0;

begin

 for i in 1..1000000000 loop

  n:=n+i;

  if (i mod 1000 = 0) then

   dbms_output.put_line(i);

  end if;

 end loop;

end;

/
```

Viendo algunas métricas del sistema operativo se puede apreciar lo siguiente:

```
[oracle@pcrac1 ~]$ vmstat 3 5
procs -----------memory---------- ---swap-- -----io---- --system-- -----cpu------
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 7  0      0 855896  83448 1146708    0    0   729    48   44  446 20  2 74  4  0
13  0      0 854284  83460 1146700    0    0    49   101  117 1805 98  2  0  0  0
10  0      0 854052  83468 1146716    0    0    64   112  125 1658 98  2  0  0  0
15  0      0 853528  83484 1146708    0    0    49    61  117 1554 98  2  0  0  0
 7  0      0 852412  83492 1146724    0    0    43    84  125 1684 98  2  0  0  0
```

Donde prácticamente el equipo está saturado por consumo de CPU. El comando vmstat muestra que el equipo tiene 100% de utilización de CPU y 0% de idle time.

Se puede apreciar a continuación que prácticamente no hay sesiones esperando por utilización de CPU.

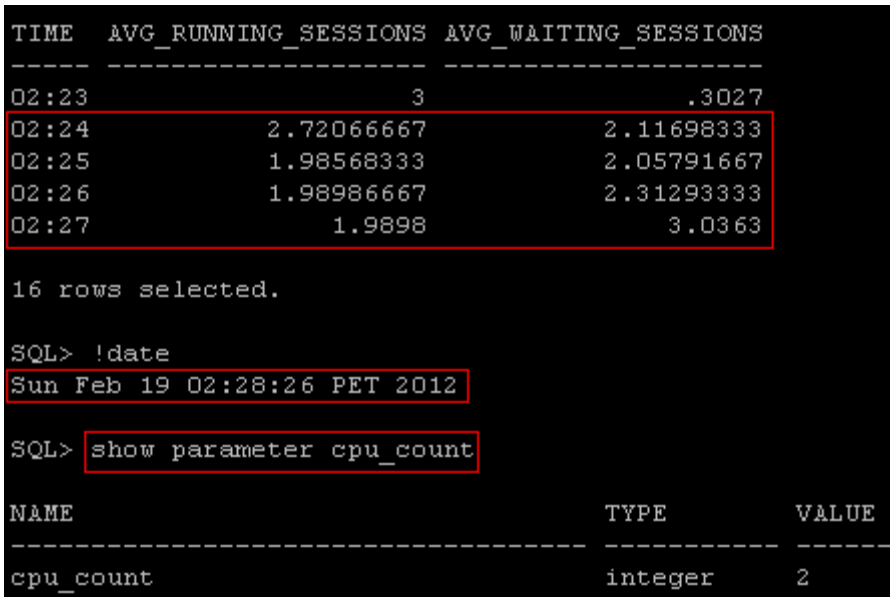
```
SQL> /

TIME  AVG_RUNNING_SESSIONS AVG_WAITING_SESSIONS
----- -------------------- --------------------
02:12           .081933333                .02805
02:13           .859783333           .022233333
02:14           .994083333           .043883333
02:15                .99605                .0001
02:16           1.01213333                .0001
02:17           .996533333           .000166667
02:18              1.01345           .000133333
02:19              3.35875               .00655
02:20           3.97851667           .009566667
02:21              4.04765           .258933333
02:22               3.9787                .08195

11 rows selected.

SQL> !date
Sun Feb 19 02:23:26 PET 2012

SQL> show parameter cpu_count

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------
cpu_count                            integer     4
```

Al cambiar el parámetro CPU_COUNT de 4 cores a 2 cores podemos ver las siguientes métricas.

La saturación de CPU se redujo a la mitad, se puede apreciar que el CPU está en un 50% de consumo y lo mismo en idle time.

```
[oracle@pcrac1 ~]$ vmstat 3 5
procs -----------memory---------- ---swap-- -----io---- --system-- -----cpu------
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 3  0      0 788684  84392 1147928    0    0   523    43   41  445 41  2 54  3  0
 3  0      0 788564  84400 1147944    0    0    48   100  119 1481 49  1 50  0  0
 5  0      0 787392  84408 1148096    0    0    64    63  123 1844 50  1 49  0  0
10  0      0 787144  84420 1148104    0    0    71   243  125 1851 49  1 50  0  0
 3  0      0 786864  84436 1148124    0    0    43    67  123 1670 49  1 50  0  0
```

Se puede apreciar también que ya existen sesiones de base de datos en promedio que están esperando por consumo de CPU al disminuir la cantidad de cores asignadas a la instancia.



```
TIME  AVG_RUNNING_SESSIONS AVG_WAITING_SESSIONS
----- ------------------- -------------------
02:23                   3                .3027
02:24          2.72066667          2.11698333
02:25          1.98568333          2.05791667
02:26          1.98986667          2.31293333
02:27              1.9898              3.0363

16 rows selected.

SQL> !date
Sun Feb 19 02:28:26 PET 2012

SQL> show parameter cpu_count

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------
cpu_count                            integer     2
```

Conclusión

En un servidor donde se alojan más de una instancia de base de datos Oracle podemos aprovechar la funcionalidad de Instance Caging para limitar el consumo de CPU que utilizará cada instancia de modo que cada instancia en un momento de carga no ocasione un overhead sobre las otras y así mantener una performance adecuada.

Publicado por Ing. Francisco Riccio. Es un IT Specialist en IBM Perú e instructor de cursos oficiales de certificación Oracle. Está reconocido por Oracle como un Oracle ACE y certificado en productos de Oracle Application & Base de Datos.

e-mail: francisco@friccio.com

web: www.friccio.com