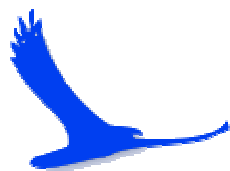


Documento generado por



Consultor Oracle  
**LIGARIUS.COM**

Para el sitio



La consigna es ayudar  
**ORACLE Y YO .COM**

## Índice

<b>1.</b>	Explicación del uso de RESULT CACHE en Oracle11g.....	2
1.1.	QUERY RESULT CACHE : .....	2
1.1.1.	<i>Ejemplos de uso</i> : .....	3
1.2.	PL/SQL FUNCTION RESULT CACHE.....	7
1.2.1.	<i>Ejemplos de uso</i> : .....	7
<b>2.</b>	¿Cómo podemos ver la información que se encuentra en memoria?, ¿específicamente en el Buffer Result Cache? .....	8
<b>3.</b>	Para ver un reporte del Buffer Result Cache .....	9
<b>4.</b>	Tamaños para el Buffer Result Cache. ....	9
<b>5.</b>	¿Qué pasa cuando se modifican los datos de origen del Result Cache? ¿Se invalidan los resultados de memoria? .....	10
<b>6.</b>	Parámetros para el Result Cache. ....	11
<b>7.</b>	Vistas para el Result Cache. ....	12
<b>8.</b>	Restricciones del uso del Result Cache .....	12
<b>9.</b>	Vaciando el Buffer Result Cache .....	13
<b>10.</b>	Referencias .....	13

---

## 1. Explicación del uso de RESULT CACHE en Oracle11g

EL Result cache es una nueva opción disponible en Oracle11g, la cual crea una porción de memoria destinada a guardar aquellos datos que son mas o menos estáticos dentro de Oracle, *¿la finalidad?*, pues poder consultar el valor final sin necesidad de pasar nuevamente por cálculos o por grandes planes de ejecución para resolver una sentencia SQL.

Cabe mencionar que RESULT CACHE sólo esta disponible para ediciones ENTERPRISE de Oracle11g.

Además de lo anterior, como el Buffer RESULT CACHE se almacena en la zona de la Shared Pool , y la Shared Pool se encuentra dentro de la SGA, para la arquitectura en RAC, cada nodo puede almacenar sus propios datos y no necesariamente estos datos viajan a través del Interconnect, a pesar de que las SGAs se encuentren dentro del Cache Fusion.

Lo que no quiere decir que cuando se invalida la información en un nodo, no se invalide en todo el RAC, esto si sucede en todo el Cache Fusion, por ende , nos da la tranquilidad de que el dato que consultamos a través del Cache Fusion con el RESULT CACHE es el válido.

Una explicación en detalle de las dos grandes opciones de RESULT CACHE, orientado a consultas y orientado a los resultados de las funciones PI/Sql

### 1.1. Query Result Cache :

Esta opción esta basada en una nueva porción de memoria, dispuesta por Oracle dentro de la SGA, esta porción de memoria se denomina Buffer Result Cache , en este buffer se almacenarán los datos para ser consultados posteriormente.

Para habilitar el result cache de forma automática, se debe setear el parámetro **RESULT\_CACHE\_MODE**, ese parámetro puede tener dos valores :

- **MANUAL** : Si se setea de esta forma el parámetro **RESULT\_CACHE\_MODE** , cada vez que se requiera utilizar y dejar en memoria el valor de alguna consulta, se deberá utilizar el hint **/\*+ result\_cache \*/**

Ejemplo : 

```
select /*+ result_cache */ count(*)
from tabla_ejemplo;
```

- **FORCE** : Seteado el parámetro **RESULT\_CACHE\_MODE** en **FORCE**, implica que siempre las consultas y ciertas funciones PI/Sql usarán el nuevo buffer de memoria destinado a almacenar los resultados.

Para no utilizarlo, se debe utilizar el hint **/\*+ no\_result\_cache \*/**

Ejemplo : 

```
select /*+ no_result_cache */ count(*)
from tabla_ejemplo;
```

---

### 1.1.1. Ejemplos de uso :

Creamos una tabla que posea una cantidad significativa de registros.

Y verificamos cuanto se demora la consulta de sus datos, de hecho la ejecutamos 2 veces para calcular el tiempo con la sentencia cargada en la Shared Pool.

```
SQL> SQL> set tim on
22:01:21 SQL> set timin on
22:01:22 SQL>
22:01:23 SQL> select count(*) from tabla_grande;

COUNT(*)
-----
3808310

Elapsed: 00:00:13.56
22:01:41 SQL> r
1* select count(*) from tabla_grande

COUNT(*)
-----
3808310

Elapsed: 00:00:13.47
```

Cada consulta demoro mas o menos 13 segundos, imagínense una tabla realmente grande de unos 800 millones de registros.

Verificamos el plan de ejecución de esta sentencia.

```
22:03:29 SQL> set autotrace trace exp stat
22:03:36 SQL> select count(*) from tabla_grande;

Elapsed: 00:00:13.19

Execution Plan
-----
Plan hash value: 751939757

-----
| Id | Operation          | Name          | Rows | Cost (%CPU)| Time     |
-----
| 0  | SELECT STATEMENT  |               |      | 15212 (1)| 00:03:03 |
| 1  | SORT AGGREGATE    |               |      | 1         |          |
| 2  | TABLE ACCESS FULL| TABLA_GRANDE | 3558K | 15212 (1)| 00:03:03 |
-----

Note
-----
- dynamic sampling used for this statement

Statistics
-----
      0 recursive calls
      0 db block gets
  55940 consistent gets
  55935 physical reads
      0 redo size
    421 bytes sent via SQL*Net to client
    420 bytes received via SQL*Net from client
      2 SQL*Net roundtrips to/from client
      0 sorts (memory)
      0 sorts (disk)
      1 rows processed

22:04:00 SQL> 22:04:00 SQL> █
```

Aparece claramente la cantidad de bloques leídos desde disco (55935)

Esta misma instrucción la ejecutamos dejando en memoria el resultado y midiendo sus tiempos de respuesta

```
22:06:48 SQL> select /*+ result_cache */ count(*)  
22:06:57 2 from tabla_grande;
```

Elapsed: 00:00:12.33

Execution Plan

Plan hash value: 751939757

Id	Operation	Name	Rows	Cost (%CPU)	Time
0	SELECT STATEMENT		1	15212 (1)	00:03:03
1	SORT AGGREGATE		1		
2	TABLE ACCESS FULL	TABLA_GRANDE	3558K	15212 (1)	00:03:03

Note

- dynamic sampling used for this statement

Statistics

```
0 recursive calls  
0 db block gets  
55940 consistent gets  
55935 physical reads  
0 redo size  
421 bytes sent via SQL*Net to client  
420 bytes received via SQL*Net from client  
2 SQL*Net roundtrips to/from client  
0 sorts (memory)  
0 sorts (disk)  
1 rows processed
```

```
22:07:13 SQL> 22:07:13 SQL>
```

No se aprecia ninguna mejoría , de hecho seguimos leyendo la misma cantidad de bloques desde disco.

Pues bien, si ejecutamos nuevamente la sentencia , veremos una mejora trascendental y sustancial.

```

08:57:23 SQL> r
1* select /*+ result_cache */ count(*) from tabla_grande
COUNT(*)
-----
3808310
Elapsed: 00:00:00.12
Execution Plan
-----
Plan hash value: 751939757

-----
| Id | Operation          | Name                | Rows | Cost (%CPU)| Time     |
-----
| 0  | SELECT STATEMENT  |                     |      |              |          |
| 1  | RESULT CACHE      | gra6bjzsay9aqlds3v95tf3cx0 |      |              |          |
| 2  | SORT AGGREGATE    |                     |      |              |          |
| 3  | TABLE ACCESS FULL| TABLA_GRANDE        | 3808K | 15316 (1)   | 00:03:04 |
-----

Result Cache Information (identified by operation id):
-----

1 - column-count=1; dependencies=(TEST.TABLA_GRANDE); attributes=(single-row); name="select /*+ result_cache */ count(*) from tabla_grande"

Statistics
-----
177 recursive calls
0 db block gets
18 consistent gets
1 physical reads
0 redo size
421 bytes sent via SQL*Net to client
420 bytes received via SQL*Net from client
2 SQL*Net roundtrips to/from client
5 sorts (memory)
0 sorts (disk)
1 rows processed
08:57:26 SQL> █

```

Impresionante!!

Ya que ahora la consulta demora 12 centésimas de segundo en leer un poco menos de 4 millones de registros y la cantidad de bloques leídos, disminuyo a 1 ☺.

Además , aparece una nueva estructura en el plan de ejecución .

## 1.2. PL/Sql Function Result Cache

Existe otra forma en que se puede ocupar el Buffer Result Cache, es mediante las funciones , las cuales se pueden dejar “marcadas” para que su valor de retorno quede en memoria, de acuerdo al nombre, tipo de parámetros (por ende hace la diferenciación si se utiliza OVERLOADED) y valor de los parámetros.

Con todo lo anterior les quiero decir , que si la consulta realiza un calculo enorme, en la segunda oportunidad donde se ejecute la función con los mismos parámetros, y los mismos valores, el resultado será obtenido de forma instantánea.

Para activar esta cualidad, se debe crear la función con los siguientes comandos.

**RESULT\_CACHE** : Con esta cláusula se le indica a Oracle que deje el resultado de la función en el Buffer Result Cache, con el nombre de la función, el tipo de datos de los parámetros y el valor de los parámetros.

**RELIES\_ON** :Esta cláusula es opcional (pero se recomienda) y se le indica a Oracle que deje registrada las tablas que utiliza la función para obtener el resultado (objetos dependientes)

### 1.2.1. Ejemplos de uso :

Imaginemos ejecutamos una pequeña función que retorna el valor de una consulta, puede ser un count(\*) o cualquier calculo intrincado.

Si validamos los tiempos de ejecución, entre dejarla o no en el Buffer Result Cache, vamos a ver los resultados en el siguiente ejemplo

Creemos la función con la cláusula Result Cache

```
16:42:17 SQL> create or replace function calculo_total (num1 number) return number
16:57:29 2 result_cache relies_on(tabla_grande)
16:57:49 3 is
16:57:50 4   var_retorno number;
16:58:09 5 begin
16:58:10 6   select count(*)
16:58:14 7     into var_retorno
16:58:19 8     from tabla_grande a , tabla_grande b
16:58:28 9     where a.object_id = b.object_id
16:58:35 10    ;
16:58:36 11 return var_retorno;
16:58:42 12 end;
16:58:47 13 /

Function created.

Elapsed: 00:00:03.84
```

Ejecutamos la consulta para inscribir la sentencia en memoria

```
16:58:51 SQL> select calculo_total(1) from dual;

CALCULO_TOTAL(1)
-----
209457161

Elapsed: 00:00:51.57
```

Si vemos, son cerca de 51 segundos de ejecución, pero en la segunda ejecución de la sentencia vemos lo impresionante que resultarán los tiempos de respuesta

```
17:12:20 SQL> r
1* select calculo_total(1) from dual

CALCULO_TOTAL(1)
-----
209457161

Elapsed: 00:00:00.04
```

Impresionante!!! Menos de 1 segundo..

Si analizamos los objetos que están en memoria, podremos ver las funciones y tablas dependientes

```
17:21:32 SQL> r
1 select type , status , name
2* from v$result_cache_objects

TYPE          STATUS      NAME
-----
Dependency    Published  TEST.CALCULO_TOTAL
Dependency    Published  TEST.TABLA_GRANDE
Result        Published  "TEST"."CALCULO_TOTAL"::8."CALCULO_TOTAL"#fac892c7867b54c6 #
1

Elapsed: 00:00:00.02
```

Se aprecia claramente que están las dependencias, tabla y función, además del resultado de la operación

## 2. ¿Cómo podemos ver la información que se encuentra en memoria?, ¿específicamente en el Buffer Result Cache?

Esto se puede llevar a cabo mediante los siguientes sentencias y/o comandos

```
Select name,status,row_count,creation_timestamp
From v$result_cache_objects
Where cache_id='valor de hash que aparece en el plan de ejecución'
```

Para nuestro caso sería algo así la sentencia

```
09:06:04 SQL> r
1 select name , status , row_count , creation_timestamp
2 from v$result_cache_objects
3* where cache_id = 'gra6bjzsay9aq1ds3v95tf3cx0'

NAME                                                                 STATUS      ROW_COUNT CREATION
-----
select /*+ result_cache */ count(*) from tabla_grande              Published          1 03/05/09

Elapsed: 00:00:00.01
09:06:04 SQL> █
```

### 3. Para ver un reporte del Buffer Result Cache

Para poder obtener información del uso real de memoria, se puede ejecutar el package DBMS\_RESULT\_CACHE.MEMORY\_REPORT.

Ejemplo :

```
09:08:46 SQL> set serveroutput on size 1000000
09:08:51 SQL> exec dbms_result_cache.memory_report;
Result Cache Memory Report
[Parameters]
Block Size           = 1K bytes
Maximum Cache Size  = 832K bytes (832 blocks)
Maximum Result Size = 41K bytes (41 blocks)
[Memory]
Total Memory = 103528 bytes [0.051% of the Shared Pool]
... Fixed Memory = 5132 bytes [0.003% of the Shared Pool]
... Dynamic Memory = 98396 bytes [0.049% of the Shared Pool]
..... Overhead = 65628 bytes
..... Cache Memory = 32K bytes (32 blocks)
..... Unused Memory = 30 blocks
..... Used Memory = 2 blocks
..... Dependencies = 1 blocks (1 count)
..... Results = 1 blocks
..... SQL      = 1 blocks (1 count)

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.12
09:08:58 SQL> █
```

Mas información sobre este package, puede ser obtenidos desde [tahiti.oracle.com](http://tahiti.oracle.com)

Como se puede apreciar, se utiliza la Shared Pool para almacenar los datos del Buffer Result Cache, mejor dicho, dentro de la Shared Pool , existe ahora otra porción de memoria, a parte del Dictionary Cache y Library Cache, está nueva porción se llama Report Cache.

### 4. Tamaños para el Buffer Result Cache.

Con respecto al tamaño que se le asigna al result cache, esto depende de las variables de memoria definida, por ejemplo, si tenemos seteado.

**MEMORY\_TARGET** : El Buffer Result Cache , va a estar seteado por Oracle en 0.25% del valor del MEMORY\_TARGET.

**SGA\_TARGET** : El Buffer Result Cache, va a estar seteado por Oracle en 0,5% de la SGA\_TARGET.

**SHARED\_POOL\_SIZE** : El Buffer Result Cache , va a estar seteado en un 1% del valor de SHARED\_POOL\_SIZE.

Nunca Oracle va a asignar de forma automática más de un 75% de la SGA al Buffer Result Cache.

## 5. ¿Qué pasa cuando se modifican los datos de origen del Result Cache? ¿Se invalidan los resultados de memoria?

Cada vez que se realiza una operación DML o DDL sobre la tabla que este en memoria, los resultados de esta en el Result Cache se invalidan.

Por lo tanto se tienen que volver a calcular, ¿pero como puedo determinar que mis resultados se encuentran inválidos?

Pues, se puede ejecutar la siguiente consulta :

```
Select * from v$result_cache_statistics
```

Con ello , puede aparecer información, como la que sigue.

a) Al ejecutar el Result Cache por primera vez

```
01:50:54 SQL> r
1* select * from v$result_cache_statistics

  ID NAME                                VALUE
-----
  1 Block Size (Bytes)                    1024
  2 Block Count Maximum                   832
  3 Block Count Current                     32
  4 Result Size Maximum (Blocks)          41
  5 Create Count Success                     0
  6 Create Count Failure                     0
  7 Find Count                               0
  8 Invalidation Count                       0
  9 Delete Count Invalid                     0
 10 Delete Count Valid                       0

10 rows selected.

Elapsed: 00:00:00.04
```

Se crean los datos en el Result Cache

b) Al ejecutar el Result Cache por segunda vez

```
01:52:21 SQL> r
1* select * from v$result_cache_statistics

  ID NAME                                VALUE
-----
  1 Block Size (Bytes)                    1024
  2 Block Count Maximum                   832
  3 Block Count Current                     32
  4 Result Size Maximum (Blocks)          41
  5 Create Count Success                     1
  6 Create Count Failure                     0
  7 Find Count                               0
  8 Invalidation Count                       0
  9 Delete Count Invalid                     0
 10 Delete Count Valid                       0

10 rows selected.

Elapsed: 00:00:00.03
```

Aparece un contador de que el resultado fue encontrado de forma exitosa en el Buffer Result Cache.

- c) Al ejecutar el una DML sobre la tabla , y verificar las estadísticas, nos encontramos con una invalidación, nótese que por ejemplo, estamos modificando no la cantidad de registros, sino que modificando el contenido de una de las celdas, a pesar de ello, se realiza la invalidación de todo el resultado en la Buffer Result Cache.  
 Esta invalidación solamente se hace efectiva al momento de realizar un commit.

```
01:55:39 SQL> r
1* select * from v$result_cache_statistics

```

ID	NAME	VALUE
1	Block Size (Bytes)	1024
2	Block Count Maximum	832
3	Block Count Current	32
4	Result Size Maximum (Blocks)	41
5	Create Count Success	1
6	Create Count Failure	0
7	Find Count	1
8	Invalidation Count	1
9	Delete Count Invalid	0
10	Delete Count Valid	0

```
10 rows selected.
Elapsed: 00:00:00.07
02:03:36 SQL>
```

Invalidación de los resultados en memoria

## 6. Parámetros para el Result Cache.

**RESULT\_CACHE\_MAX\_RESULT** : Especifica el porcentaje del RESULT\_CACHE\_MAX\_SIZE que una sola consulta puede utilizar.

**RESULT\_CACHE\_MODE** : Es el modo en que va a funcionar el Result Cache, si es por defecto automático para todas las sentencias (FORCE) o solamente de forma manual (MANUAL).

**RESULT\_CACHE\_MAX\_SIZE** : Indica el máximo tamaño del buffer para el result cache, si este parámetro esta seteado en 0 está deshabilitado el Result Cache, y cuando la instancia parte, si este valor no es seteado (null) , Oracle lo reserva de forma automática desde la Shared Pool.

**RESULT\_CACHE\_REMOTE\_EXPIRATION** : Indica la cantidad de minutos que un resultado que se encuentre en memoria y que este utilizando una tabla en memoria, estará cargado como válido en el Buffer Result Cache.

Desde Sql\*Plus escribir

**SQL> show parameter result**

## 7. Vistas para el Result Cache.

**V\$RESULT\_CACHE\_STATISTICS** : Provee estadísticas del uso de memoria y de las estadísticas de uso de la Buffer Result Cache.

**V\$RESULT\_CACHE\_MEMORY** : Provee un listado de todos los bloques en memoria (Buffer Result Cache) y de las estadísticas asociadas.

**V\$RESULT\_CACHE\_OBJECTS** : Provee un listado completo de todos los objetos (y dependencias) que se encuentren en el Buffer Result Cache.

**V\$RESULT\_CACHE\_DEPENDENCY** : Provee un listado de todos los objetos dependientes de los objetos que se encuentren en la Buffer Result Cache.

Cabe mencionar que el Buffer Result Cache, no posee un advisor como por ejemplo la Shared Pool , por ende su tuning (Buffer Result Cache) , se debe realizar por intermedio de la Shared Pool .

## 8. Restricciones del uso del Result Cache

- Result cache no esta soportado para el diccionario de datos .
- Result cache no esta soportado para tablas temporales.
- Seudo columnas de secuencias , no está soportado (nextval y curval).
- Funciones que utilicen CURRENT\_DATE, CURRENT\_TIMESTAMP, LOCAL\_TIMESTAMP, USERENV, CONTEXT, SYSDATE y SYS\_TIMESTAMP.
- Funciones PL/Sql no determinísticas

Para lo que es PL/Sql Function Result Cache, existen restricciones extras

- No se pueden utilizar modo OUT en los parámetros de la función
- No se pueden utilizar modo IN con campos BLOB en los parámetros de la función
- No se pueden utilizar campos REF\_CURSOR en los parámetros de la función

## 9. Vaciando el Buffer Result Cache

Para llevar a cabo el vaciado del Buffer Result Cache, se puede realizar cualquiera de estas actividades :

- Bajar la base de datos, sería contraproducente ¿no creen?
- O utilizar el package

### **DBMS\_RESULT\_CACHE.FLUSH**

El cual sólo vacía el Buffer Result Cache

- O realizar un vaciado de la Shared Pool completa, lo cual se puede realizar mediante la ejecución del package

### **DBMS\_SHARED\_POOL.FLUSH**

Mas información sobre el package [DBMS\\_SHARED\\_POOL](#)

## 10. Referencias

[http://www.oracle.com/technology/obe/11gr1\\_db/perform/rescache/res\\_cache.htm](http://www.oracle.com/technology/obe/11gr1_db/perform/rescache/res_cache.htm)

**11g New Feature PL/SQL Function Result Cache 430887.1**

**11g New Feature : SQL Query Result Cache 453567.1**