11G 性能优化新技术: SQL QUERY RESULT CACHE

BY SHOUG. 罗敏

How to Find SHOUG?

11g 性能优化新技术：SQL Query Result Cache

第一次听说 SQL Query Result Cache 这个 11g 新技术，是在 2007 年 11 月的 Oracle 公司内部 11g 培训课程中。可能是因为 11g 新特性太多，而且那次连续几天的培训，被老师轰炸得审美疲劳了，对这个新技术并没有留下太深印象。当时只是感觉 Oracle 又多了个什么 Cache，也没完全弄明白其原理，也不知道与传统 Buffer Cache 有什么区别，更不知道这个技术适合于什么场景。

待日后有精力再深入研究该技术，特别是结合客户相关实际问题，才猛然发现 Oracle 这个新技术太牛了！Oracle 公司也太富有创新能力了，居然在几十年难得一动的 SGA 和 Shared_Pool 内存架构中增加了 Result Cache 结构，并提供了新的 SQL Query Result Cache 技术，解决了很多重复查询语句导致资源开销过大的典型问题。

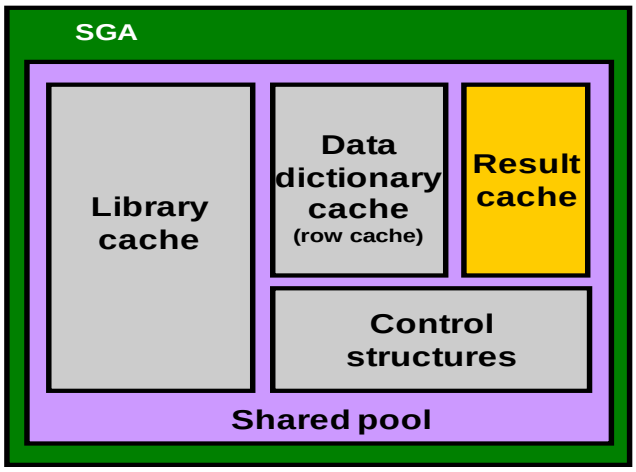
Result Cache 到底是什么新技术？如何使用？适合于什么应用场景？… …。希望本章的内容能回答这些问题，更希望您别像我当年一样，仅仅是走马观花地了解一下而已，而是能立马激动起来，并能运用到您的实际应用开发工作中去。呵呵。

7.1 Result Cache 原理

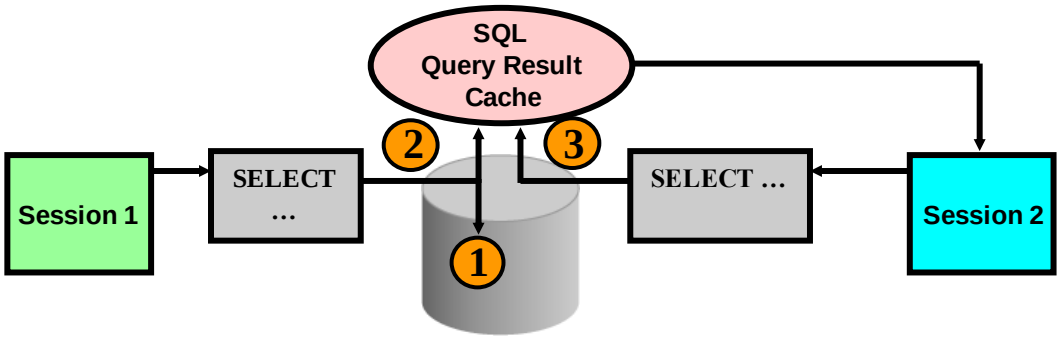
基本原理

11g 出了个 SQL Query Result Cache 的新特性，简称 Result Cache 技术吧。其基本原理就是将 SQL 语句查询结果数据直接存储在内存中，这样后续相同查询语句就直接从该内存中读取了，这将极大地提高该类语句重复查询性能。

为此，Oracle 在 SGA 的 Shared Pool 内存中专辟了一个 Result Cache 内存，如下图所示：



以下就是 Result Cache 的工作原理图：



即：

1. 当第一个会话进行第一次查询时，将首先从硬盘读取数据①。
2. 第一个会话同时将该数据存储在 SQL Query Result Cache 区域②。
3. 第二个会话进行相同语句查询时，Oracle 直接从 SQL Query Result Cache 区域读取数据③，将极大地提升查询效率。

可见，Result Cache 具有跨语句、跨会话能力，同时当数据发生变更时，Result Cache 中相关数据将变为非法（INVALID）状态，Oracle 会自动重新从硬盘读取变更数据，并再次存储在 Result Cache 之中。

Result Cache 与 Buffer Cache 的区别

当 2007 年 11 月在参加 11g 新特性培训中，第一次听说这个技术时，一时没有反应过来："Oracle 不是将查询数据已经存储在 Buffer Cache 中了吗？怎么又来一个 Result Cache 呢？"后来，稍加琢磨，反应过来了：Buffer Cache 中存储的是需要访问的数据，其作用只是将需要到硬盘访问的数据变为内存访问了，而 Result Cache 是访问数据的结果数据。因此，Result Cache 的作用比 Buffer Cache 有效多了。

例如：

```
SELECT department_id, AVG(salary)
FROM emp
GROUP BY department_id;
… …
统计信息
----------------------------------------------------
--------
          0  recursive calls
          0  db block gets
      16045  consistent gets
          0  physical reads
          0  redo size
        726  bytes sent via SQL*Net to client
        415  bytes received via SQL*Net from
client
          2  SQL*Net roundtrips to/from client
```

```
          0  sorts (memory)
          0  sorts (disk)
         12  rows processed
```

上述语句如果使用 Buffer Cache 技术，Oracle 只是将 employees 表所有记录存储在 Buffer Cache 中，后续语句仍然要从 Buffer Cache 中访问 employees 表所有记录，并进行按 department_id 的部门平均工资统计，例如上述语句依然有 16045 次 consistent gets 访问。而如果使用了 Result Cache 技术，Oracle 则将该语句如下查询结果存储到 Result Cache 中：

```
DEPARTMENT_ID AVG(SALARY)
------------- -----------
          100  8601.33333
           30        4150
                     7000
           20        9500
           70       10000
           90  19333.3333
          110       10154
           50  3475.55556
           40        6500
           80  8955.88235
           10        4400
           60        5760

已选择 12 行。
```

后续语句直接从 Result Cache 中访问这些结果数据，显然结果数据比明细数据少得多。再看该语句的资源消耗情况：

```
 SELECT /*+ RESULT_CACHE */ department_id,
AVG(salary)
 FROM emp
 GROUP BY department_id;
… …
统计信息
----------------------------------------------------
--------
          0  recursive calls
          0  db block gets
          0  consistent gets
          0  physical reads
          0  redo size
        726  bytes sent via SQL*Net to client
        415  bytes received via SQL*Net from
client
          2  SQL*Net roundtrips to/from client
          0  sorts (memory)
          0  sorts (disk)
         12  rows processed
```

可见，此时 Oracle 不仅没有访问硬盘（physical reads = 0），而且也根本没有访问 Buffer Cache（consistent gets=0）！

因此，Result Cache 技术非常适合于满足如下条件的场景：

- 查询的记录数很多，但返回结果数据较少的应用
- 重复查询频度比较高
- 数据相对静态，变化量不大

例如，数据仓库系统的各种统计运算就是比较典型的应用场景。

7.2 Result Cache 使用过程

初始化参数的设置

欲使用 Result Cache，首先需要考虑初始化参数的设置。以下是几个典型参数的含义和配置建议：

- RESULT_CACHE_MODE

该参数表示是否需要采用 Result Cache 技术，取值如下：

  - MANUAL：表示当在语句中增加相关 HINT（RESULT_CACHE）时，才使用 Result Cache 技术。该值为缺省值。
  - FORCE：表示只要有可能，所有查询语句都将使用 Result Cache 技术。

无论该参数如何设置，Oracle 将优先考虑 RESULT_CACHE 和 NO_RESULT_CACHE 的 hint。

- RESULT_CACHE_MAX_SIZE

该参数设置 Result Cache 的最大容量。如果设置为 0，则将关闭 Result Cache 功能。该参数的缺省值，依赖于内存管理模式和相关参数配置。例如：

  - 当只设置 memory_target 参数时，RESULT_CACHE_MAX_SIZE = memory_target*0.25%。
  - 当设置 sga_target 参数时，RESULT_CACHE_MAX_SIZE = sga_target*0.5%。
  - 当设置 shared_pool_size 参数时，RESULT_CACHE_MAX_SIZE = shared_pool_size*1%。

  该参数最大不能超过 shared_pool_size 的 75%。

- RESULT_CACHE_MAX_RESULT

该参数为单个 SQL 查询语句设置可使用的最大 Result Cache 容量，缺省为 RESULT_CACHE_MAX_SIZE 的 5%。

- RESULT_CACHE_REMOTE_EXPIRATION

该参数表示当 SQL 语句访问远程数据库对象时，允许远程对象数据发生变化的过期时间。缺省值为 0，表示一旦远程对象数据发生变化，相关查询的 Result Cache 数据变为 INVALID。

Result Cache 的使用

当上述初始化参数设置好之后，我们就可以考虑 Result Cache 技术的使用了。通常，我们不建议通过采取 RESULT_CACHE_MODE 设置为 FORCE 而强制使用 Result Cache 的策略。因为，这将导致系统将所有查询操作结果都考虑进行缓存，反而会增加系统不必要的开销。

如果将 RESULT_CACHE_MODE 设置为 MANUAL，则语句中应增加/*+ RESULT_CACHE */的 HINT。如下例所示：

```
SELECT /*+ RESULT_CACHE */ department_id,
AVG(salary)
 FROM emp
 GROUP BY department_id;
```

执行计划如下：

Description	Object owner	Object name	Cost	Cardinality	Bytes
SELECT STATEMENT, GOAL = ALL_ROWS			4509	11	77
RESULT CACHE		8k66uas70mk859zxgaq75kfq9u			
HASH GROUP BY			4509	11	77
TABLE ACCESS FULL	HR	EMP	4427	1753088	12271616

可见，执行计划中增加了 RESULT CACHE 操作，所访问的 Object Name “8k66uas70mk859zxgaq75kfq9u”是该语句结果集在 Result Cache 中的编号（Cache_ID）。

以下语句就是强制不使用 Result Cache 技术，即便 RESULT_CACHE_MODE 设置为 FORCE。

```
SELECT /*+ NO_RESULT_CACHE */ department_id,
AVG(salary)
FROM emp
GROUP BY department_id;
```

再看如下语句：

```
SELECT prod_subcategory, revenue
FROM (SELECT /*+ RESULT_CACHE */ p.prod_category,

p.prod_subcategory,

sum(s.amount_sold) revenue
      FROM products p, sales s
      WHERE s.prod_id = p.prod_id and
            s.time_id BETWEEN to_date('01-JAN-
2006','dd-MON-yyyy')
                              and
```

```
                                    to_date('31-DEC-
2006','dd-MON-yyyy')
      GROUP BY ROLLUP(p.prod_category,
p.prod_subcategory))
WHERE prod_category = 'Women';
```

如果在上述 in-line 视图中使用了 Result Cache 技术，Oracle 将不会再进行视图的合并（Merge）等变更操作，而是直接将视图结果存储在 Result Cache 中，后续查询包括外围条件发生变化，例如 prod_category = 'Men'，都将使用 Result Cache 技术。

7.3 Result Cache 的管理

DBMS_RESULT_CACHE 包的使用

通过 11g 的 DBMS_RESULT_CACHE 包，可以对 Result Cache 进行相关管理工作：

- Result Cache 状态

如下语句可查询 Result Cache 状态：

```
SQL> SELECT DBMS_RESULT_CACHE.STATUS FROM DUAL;
STATUS
--------
ENABLED
```

- Result Cache 的使用情况

如下语句可查询 Result Cache 的使用情况：

```
SQL> set serveroutput on;
SQL> exec DBMS_RESULT_CACHE.MEMORY_REPORT;
R e s u l t   C a c h e   M e m o r y   R e p o r
t
[Parameters]
Block Size          = 1K bytes
Maximum Cache Size  = 3168K bytes (3168 blocks)
Maximum Result Size = 158K bytes (158 blocks)
[Memory]
Total Memory = 107812 bytes [0.044% of the Shared
Pool]
... Fixed Memory = 9460 bytes [0.004% of the
Shared Pool]
... Dynamic Memory = 98352 bytes [0.040% of the
Shared Pool]
....... Overhead = 65584 bytes
....... Cache Memory = 32K bytes (32 blocks)
........... Unused Memory = 29 blocks
........... Used Memory = 3 blocks
............... Dependencies = 1 blocks (1 count)
............... Results = 2 blocks
```

```
.................. SQL      = 2 blocks (2 count)

PL/SQL 过程已成功完成。
```

- 清空 Result Cache

如下语句可清空 Result Cache，包括所有已存在的结果集数据：

```
SQL> exec DBMS_RESULT_CACHE.FLUSH;
PL/SQL 过程已成功完成。
SQL> exec DBMS_RESULT_CACHE.MEMORY_REPORT;
R e s u l t   C a c h e   M e m o r y   R e p o r
t
[Parameters]
Block Size          = 1K bytes
Maximum Cache Size  = 3168K bytes (3168 blocks)
Maximum Result Size = 158K bytes (158 blocks)
[Memory]
Total Memory = 9460 bytes [0.004% of the Shared
Pool]
... Fixed Memory = 9460 bytes [0.004% of the
Shared Pool]
... Dynamic Memory = 0 bytes [0.000% of the Shared
Pool]

PL/SQL 过程已成功完成。
```

- 将指定表的 Result Cache 设置为 INVALID

如下语句将 HR 用户下的 EMP 表的 Result Cache 设置为 INVALID：

```
SQL> EXEC
DBMS_RESULT_CACHE.INVALIDATE('HR','EMP');
PL/SQL 过程已成功完成。
```

相关视图的使用

- (G)V$RESULT_CACHE_STATISTICS

该视图显示 Result Cache 设置和该内存使用情况的统计信息。例如：

ID	NAME	VALUE
1	Block Size (Bytes)	1024
2	Block Count Maximum	3168
3	Block Count Current	32
4	Result Size Maximum (Blocks)	158
5	Create Count Success	2
6	Create Count Failure	0
7	Find Count	1
8	Invalidation Count	1
9	Delete Count Invalid	0
10	Delete Count Valid	0
11	Hash Chain Length	1

- (G)V$RESULT_CACHE_MEMORY

该视图显示 Result Cache 所有内存块和相关统计信息。

- (G)V$RESULT_CACHE_OBJECTS

该视图显示 Result Cache 中被缓存的对象，包括结果集数据和依赖的表及相关属性数据。例如，以下是部分字段查询结果：

ID	TYPE	STATUS	BUCKET_NO	HASH	NAME
0	Dependency	Published	371	2.947E+09	HR.EMP
2	Result	Published	3135	1.132E+09	SELECT /*+ RESULT_CACHE */ department_id, AVG(salary) FROM emp GROUP BY department_id
1	Result	Invalid	3135	1.132E+09	SELECT /*+ RESULT_CACHE */ department_id, AVG(salary) FROM emp GROUP BY department_id

- (G)V$RESULT_CACHE_DEPENDENCY

该视图显示结果集数据和依赖表的关联关系，例如：

RESULT_ID	DEPEND_ID	OBJECT_NO
2	0	74569

查询结果显示结果集 2（V$RESULT_CACHE_OBJECTS 表的 ID =2 ）与依赖数据 0（V$RESULT_CACHE_OBJECTS 表的 ID = 0 ）有关联关系，该依赖数据的 OBJECT_NO 为 74569，即 HR.EMP 表。

7.4 Result Cache 相关技术点

Result Cache 与 RAC

RAC 支持 Result Cache 技术。RAC 环境中的每个实例都有自己的 Result Cache，每个实例的 Result Cache 不能共享，即保存在 Result Cache 中的数据只能被本实例的应用进行访问。但是，一旦保存在某个 Result Cache 中的数据变成 INVALID，则整个 RAC 环境中各 Result Cache 中的该数据都将变成 INVALID。Oracle 通过专门的 RCBG 进程，处理 RAC 环境下 Result Cache 之间的数据同步。

Result Cache 与并行处理

并行处理也支持 Result Cache 技术。在并行查询中，整个查询结果集将被保存在 Result Cache 中，也就是说，整个并行查询语句方可使用 Result Cache 中的查询结果，单个并行查询子进程无法访问 Result Cache。在 RAC 环境下，并行查询结果保存在查询协调进程（Query Coordinator）所在实例的 Result Cache 中。

Result Cache 的局限

Result Cache 技术存在如下一些限制：

- 系统临时表（Temporary Table）和数据字典表不支持 Result Cache 技术。
- 非确定的（Nodeterministic）PL/SQL 函数不支持 Result Cache 技术。
- 查询语句若出现序列的 CURRVAL、NEXTVAL，则不支持 Result Cache 技术。
- 查询语句若出现 current_date, sysdate, sys_guid 等函数，则不支持 Result Cache 技术。

Result Cache 更多技术限制，请参阅 11g 联机文档《Oracle® Database Performance Tuning Guide11g Release 2 (11.2)》第 7 章相关内容，以及其它 Oracle 文档。

Result Cache 的其它技术点

- Result Cache 支持 Flashback 查询。
- Result Cache 并不会自动释放其内存，Oracle 将一直使用该内存到其最大限额值（RESULT_CACHE_MAX_SIZE），然后通过 FIFO 技术，释放相关内存。另外，通过 DBMS_RESULT_CACHE.FLUSH 可清空 Result Cache 内存。
- Result Cache 支持绑定（BIND）变量的使用。针对绑定变量语句，Result Cache 不仅保存结果集，而且保存相关变量值。这样，只有相同变量值的查询语句，方可使用其在 Result Cache 中的结果集。

Result Cache 更多技术点，请参阅 11g 联机文档《Oracle® Database Performance Tuning Guide11g Release 2 (11.2)》第 7 章相关内容，以及其它 Oracle 文档。

7.5 客户端 Result Cache 技术

客户端也支持 Result Cache

前面介绍的 Result Cache 技术，主要针对服务器端。实际上，11g 在客户端也支持 Result Cache 技术，即 OCI Client Query Cache 技术。该技术特点如下：

- 将服务器端的 Result Cache 扩展到客户端，将充分利用客户端内存，降低服务器端内存开销，提高了整个系统的扩展性。
- 运用客户端的 Result Cache 技术，将降低网络往返传输开销（round-trips），从而显著提高系统整体性能。
- 当服务器端的数据发生变化时，Oracle 将自动刷新客户端的 Result Cache 数据。

- 客户端的 Result Cache 技术主要适合于重复查询、频度较高的 SQL 语句，而且结果数据较少、数据相对静态的应用。例如，大量代码表的查询等。

- 服务器端和客户端的 Result Cache 技术是相互独立的，可以分别进行设置。

- 通过 client_result_cache_stats$、v$client_result_cache_stats 视图，可监控客户端的 Result Cache 的使用。

客户端 Result Cache 的使用

- 数据库初始化参数

  - CLIENT_RESULT_CACHE_SIZE 参数

    该参数定义了单个客户端进程所使用的 Result Cache 最大值。如果设置为 0，则关闭了客户端 Result Cache 技术。该参数缺省值即为 0。

  - CLIENT_RESULT_CACHE_LAG 参数

    该参数定义客户端与服务端最后一次往返（round-trip）时间阀值，在该时间阀值内，客户端将向服务端查询客户端 Result Cache 中的数据是否发生变化，该参数缺省值为 3000 毫秒。即每隔 3 秒，Oracle 将客户端 Result Cache 中的数据与服务端进行一次同步。

- 客户端相关配置

通过在客户端的 sqlnet.ora 配置文件中设置如下参数，可以进行客户端 Result Cache 的配置。

  - OCI_RESULT_CACHE_MAX_SIZE

  - OCI_RESULT_CACHE_MAX_RSET_SIZE

  - OCI_RESULT_CACHE_MAX_RSET_ROWS

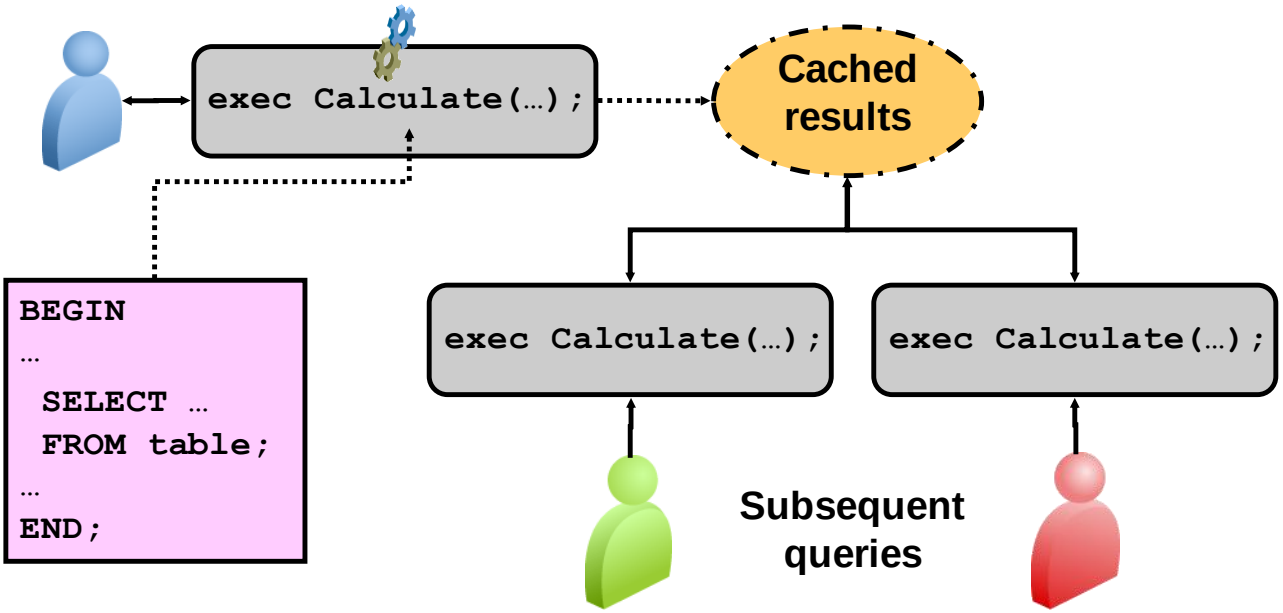
这些参数设置了单个客户端进程可使用的 Result Cache 最大值(Bytes)或最大记录数。客户端参数的设置将优先于服务器端的参数，例如 CLIENT_RESULT_CACHE_SIZE 参数。

欲使用客户端 Result Cache，应用程序应与 11.1 以上版本客户端库进行连接，并访问 11.1 以上数据库服务器。

PL/SQL 中 Result Cache 的使用

- 原理

PL/SQL 支持 Result Cache 技术的使用。以下就是原理示意图：



即当 PL/SQL 函数 Calculate 第一次执行时，Oracle 将按正常方式执行，并将执行结果存储在 Result Cache 中。而在后续的执行过程中，当该函数涉及的数据和调用参数没有发生变化时，Oracle 将避免重复计算，而是直接将 Result Cache 中保存的该函数结果返回客户。而当该函数涉及的数据和调用参数发生变化时，Oracle 将自动重新计算该函数，并将结果数据再次存储在 Result Cache 中。

在 PL/SQL 中运用 Result Cache 技术，特别适合于大批量复杂计算，并且数据相对静态的应用，而且对应用本身透明，更省去了在应用层面进行结果数据存储的开发和管理工作。

- 使用过程

以下就是 PL/SQL 中 Result Cache 的使用案例：

```
CREATE OR REPLACE FUNCTION productName
  (prod_id NUMBER, lang_id VARCHAR2)
  RETURN NVARCHAR2
  RESULT_CACHE RELIES_ON (product_descriptions)
IS
  result VARCHAR2(50);
BEGIN
  SELECT translated_name INTO result
    FROM product_descriptions
    WHERE product_id = prod_id AND language_id =
lang_id;
  RETURN result;
END;
```

即在函数声明部分增加 RESULT_CACHE 关键字，并且可增加 RELIES_ON 短语，表示该函数依赖于 product_descriptions 表。当 product_descriptions 表的数据发生变更时，该函数的结果集将变为 INVALID。

- 相关限制

在如下情况下，PL/SQL 无法使用 Result Cache 技术。

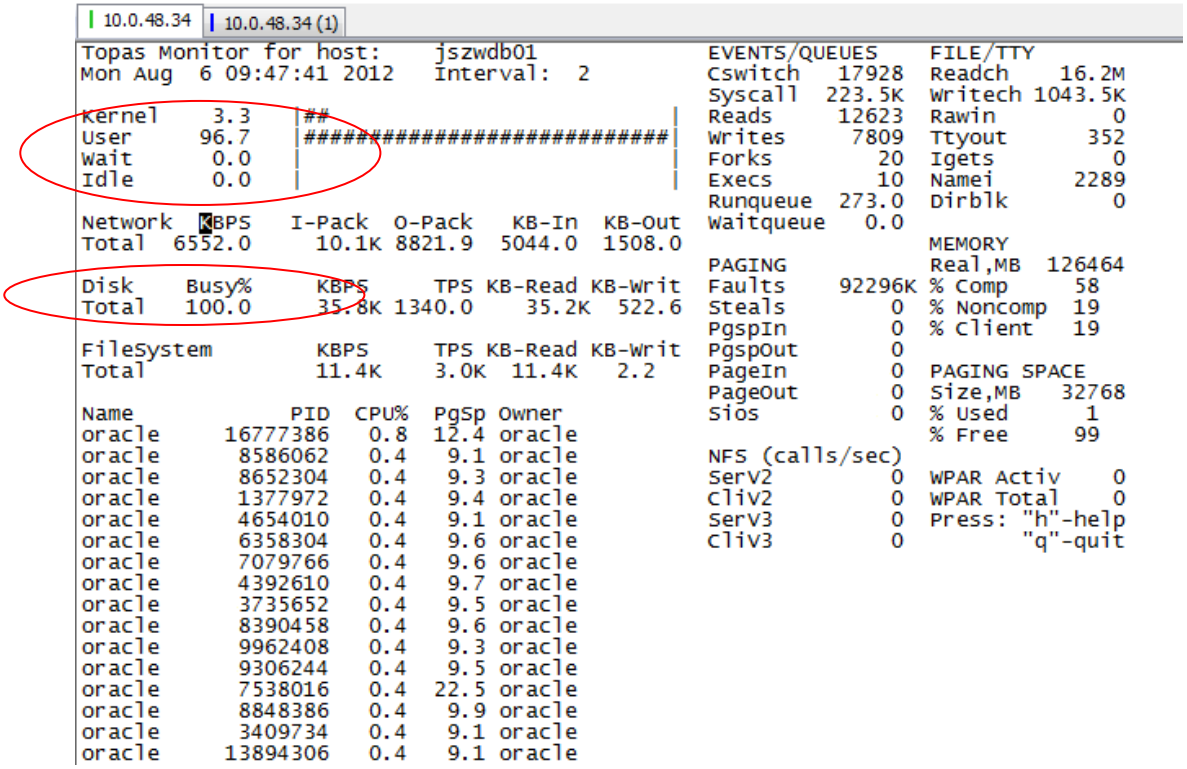
- 当函数定义在一个需要调用者权限的模块中，或者定义在一个匿名块中。
- 该函数是一个管道表（Pipelined Table）函数。
- 该函数有 OUT 或 IN OUT 参数。
- 该函数包括 BLOB、CLOB、NCLOB、REF CURSOR、collection、object、record 等类型参数。
- 该函数返回类型包括 BLOB, 、CLOB、NCLOB、REF CURSOR、object、record，以及包含上述不支持返回类型的 collection。

7.6 适合 Result Cache 的典型案例

相关背景

- 系统概述

某运营商总部的集中结算系统共有 2 套互为 HA 备份的单机系统，节点 1 主要承担联机交易，节点 2 主要承担报表、统计等批处理业务。两套系统主要问题都是运行状态不佳。例如以下是我们在现场调研时发现的节点 1 的运行状况：



可见节点 1 系统的 CPU、I/O 利用率都几乎达到 100%，并经常导致系统宕机和重启。

- 数据库层面资源消耗情况

以下是节点 1 数据库的 Load Profile：

	Per Second	Per Transaction
Redo size:	1,023,082.84	13,340.70
Logical reads:	1,838,583.03	23,974.58
Block changes:	5,770.79	75.25
Physical reads:	3,237.45	42.22
Physical writes:	774.41	10.10
User calls:	6,344.54	82.73
Parses:	3,289.28	42.89
Hard parses:	2.72	0.04
Sorts:	1,876.35	24.47
Logons:	42.58	0.56
Executes:	4,098.31	53.44
Transactions:	76.69	

可见内存和 I/O 开销都非常大。

以下是 Top 5 Timed Events

Event	Waits	Time(s)	Avg Wait(ms)	% Total Call Time	Wait Class
latch: cache buffers chains	639,561	35,422	55	28.1	Concurrency
CPU time		27,473		21.8	
SQL*Net message from dblink	85,888	1,848	22	1.5	Network
SQL*Net more data from dblink	3,937,144	1,587	0	1.3	Network
latch free	16,422	1,333	81	1.1	Other

最高等待事件是：latch: cache buffers chains，说明带系统存在比较严重的热块数据。

主要问题分析

- 最高等待事件分析

如上所述，最高等待事件 latch: cache buffers chains 说明带系统存在比较严重的热块数据。进一步分析如下 Segments by Logical Reads 数据：

Owner	Tablespace Name	Object Name	Subobject Name	Obj. Type	Logical Reads	%Total
STL_USR	COMMDATA	T_FILE_INFO_81		TABLE	1,125,308,256	33.98
SETTLE	STATDATA	T_FILE_INFO_18		TABLE	952,259,776	28.75
STL_USR	COMMDATA	T_FILE_INFO_13		TABLE	218,271,024	6.59
STL_USR	COMMDATA	T_FILE_INFO_30		TABLE	145,521,632	4.39
STL_USR	COMMDATA	T_FILE_INFO_50		TABLE	143,596,320	4.34

可见 T_FILE_INFO_81 和 T_FILE_INFO_18 是该系统被集中频繁访问的表。

- 典型应用分析

进一步分析，在 Top-SQL 语句中，访问上述两表的语句占了绝大多数。例如：

```
select count(*)
  from t_file_info_18
 where province_id = :"SYS_B_0"
   and operation_type_grade = :"SYS_B_1"
   and flow_id = :"SYS_B_2"
   and insert_ip = :"SYS_B_3"
   and (state = :"SYS_B_4" or state = :"SYS_B_5")
```

上述语句目前执行计划为全表扫描，该表目前已达到 500 多万记录，容量 1.6GB。

该表虽然已经建立了（PROVINCE_ID，OPERATION_TYPE_GRADE，FLOW_ID，STATE, INSERT_IP）的复合索引，但由于该索引的可选性不高，所以 Oracle 优化器选择了全表扫描。

- 解决方式建议

最佳解决方式是 11g SQL Query Result Cache 技术！经与应用和业务部门沟通，T_FILE_INFO_18 表数据的查询频度远高于 DML 操作频度，而且上述语句是全表扫描，访问结果仅仅是求和一条记录。即便基表 T_FILE_INFO_18 表数据发生变化，Oracle 也会自动将 Result Cache 中相关内容变为 INVALID，而再次访问变化数据，并将变化数据的结果集再次存储在 Result Cache 中，供后续相同查询语句访问。预计 Result Cache 技术的运用将极大地解决该系统突出的性能问题。

可惜现在该系统还无法使用 Result Cache 技术，为什么？因为该系统还运行在 10g 平台，暂时还没有计划升级到 11g。想唱戏的连个舞台都没有，呵呵。

7.7 本章参考资料及进一步读物

本章参考资料及进一步读物：

序号	资料类别	资料名称	资料概述
1.	Oracle 11g R2 联机文档	《Oracle® Database Performance Tuning Guide》	请大家重点看第 7 章第 7.6 节“Managing the Server and Client Result Caches”
2.	Oracle 11g R2 联机文档	《Oracle® Database Administrator's Guide》	请大家重点看第 6 章 “Specifying the Result Cache Maximum Size”小节
3.	Oracle 大学教材	《Oracle® Database 11g New Features》第 11 章	这是 Oracle 大学教材。欲看到图文并茂的该文档，只能报名参加该课程的培训了。
4.	My Oracle Support	《SQL Query Result Cache. Includes: [Video] (Doc ID 1108133.1)》	系统介绍 SQL Query Result Cache 的文档，还有视频哦，听听标准的美式英语吧。
5.	My Oracle Support	《11g New Feature : SQL Query Result Cache (Doc ID 453567.1)》	一篇介绍 SQL Query Result Cache 的简洁的文档，以及一个例子。

序号	资料类别	资料名称	资料概述
6.	My Oracle Support	《11g New Feature PL/SQL Function Result Cache (Doc ID 430887.1)》	一篇介绍 PL/SQL Function Result Cache 的简洁的文档，还有一个详细的例子。

作者个人简介

罗敏，Oracle ACS 高级顾问

80 年代毕业于武汉大学计算机科学系，国防科学技术大学计算机学院硕士学位。

在 Oracle 中国公司的 10 年时间里分别在顾问咨询部、技术服务部担任资深技术顾问。在银行、电信、政府等行业和部分参与了多个大型 IT 系统的建设，提供了体系架构设计、数据库设计、应用开发设计指导、性能优化等领域的咨询和技术支持服务。

已出版书籍《品悟 Oracle 性能优化》，罗敏的另一本新书即将出版。