【共有プールにキャッシュされている実行計画の出力】
v$SQL_PLAN ビューからの「SQL 実行計画」の取得方法

v$SQL_PLAN ビューには、現在ライブラリキャッシュにある SQL の実行計画が保持されている

よって、v$SQL_PLAN を検索することによって、実行計画と実行統計情報を取り出すことができる。

Ⓐ ライブラリキャッシュ内にある SQL 実行計画を出力する方法

v$SQL_PLAN ビューの中から、sql_id 列が上記で調査した SQL__ID 値と一致するレコードを検索する

sql>　@d:¥temp¥list07_2.sql　＜SQL__ID 値＞　；

↑

この値が、list07_2.sql 中の&1 に代入される

例）@ d:¥temp¥list07_2.sql　4cqayrarbsfd ；

↓

次ページにソース・コードを掲載

※ ソース・コードを SQL*Plus にコピー・ペーストで使用する場合は、&1 パラメータに 2 回入力を求めてくるので、同一の SQL_ID 値を入力すること

また、単一 SQL 文なので、コピー時に↵（リターン・コード）を余分にコピーしてしまうと、1 回目の SQL_ID 値に↵（リターン・コード）のみが誤って入ってしまうので要注意

※ SQL__ID 値から検索を行った場合の実行例を書きに示す
しかし、1 つの SQL 文に複数の実行計画が存在する場合には、複数の実行計画が同時（Plan_hash_value 順）に出力される~~正しい調査結果が得られない~~
この場合には、SQL 文に対する SQL_ID 値と CHILD_NUMBER 値を求めて、この 2 つの値を Select 文の Where 条件句に指定して実行すること

実行結果（実行計画の表示）

```
-----------------------------------------------------------------------------------------------
|  Operation                    | PHV/Object Name    | Rows      | Bytes  | Cost    |
-----------------------------------------------------------------------------------------------
| UPDATE STATEMENT              | ----- 618325093 ----- |          |        |       1 |
| UPDATE                        |WRI$_ADV_PARAMETERS |          |        |         |
|    INDEX UNIQUE SCAN          |WRI$_ADV_PARAMETERS_| 1        |   36   |       0 |
-----------------------------------------------------------------------------------------------
```

list07_2.sql スクリプトのソース・コード　　&1：スクリプト呼出し時の、パラメータ
==========　list07_2.sql　　（ 調査したいSQL文のSQL_ID値 ）

```
set heading off ver off
select '-----------------------------------------------------------------------' from dual
union all
select '| Operation                          | PHV/Object Name       |   Rows | Bytes|
        Cost |'   as "Optimizer Plan:" from dual
union all
select '-----------------------------------------------------------------------' from dual
union all
select * from (
        select
        rpad('|'||substr(lpad(' ',1*(depth-1))||operation||
              decode(options, null,',''||options), 1, 32), 33, ' ')||'|'||
        rpad(decode(id, 0, '----- '||to_char(plan_hash_value)||' -----'
                        ,  substr(decode(substr(object_name,  1,  7),  'SYS_LE_',  null,
                                    object_name)
                          ||' ',1, 20)), 21, ' ')||'|'||
        lpad(decode(cardinality,null,'   ',
                  decode(sign(cardinality-1000), -1, cardinality||' ',
                  decode(sign(cardinality-1000000), -1, trunc(cardinality/1000)||'K',
                  decode(sign(cardinality-1000000000),                                  -1,
trunc(cardinality/1000000)||'M',
                        trunc(cardinality/1000000000)||'G')))), 7, ' ') || '|' ||
        lpad(decode(bytes,null,' ',
                  decode(sign(bytes-1024), -1, bytes||' ',
                  decode(sign(bytes-1048576), -1, trunc(bytes/1024)||'K',
                  decode(sign(bytes-1073741824), -1, trunc(bytes/1048576)||'M',
                        trunc(bytes/1073741824)||'G')))), 6, ' ') || '|' ||
        lpad(decode(cost,null,' ',   decode(sign(cost-10000000), -1, cost||' ',
                  decode(sign(cost-1000000000), -1, trunc(cost/1000000)||'M',
                        trunc(cost/1000000000)||'G'))), 8, ' ') || '|' as "Explain plan"
          from   v$sql_plan
         where sql_id   =   '&1'
           and plan_hash_value in (select distinct plan_hash_value   from v$sql
                                     where sql_id   =   '&1' )
          order by   plan_hash_value,   id
)
union all
select '-----------------------------------------------------------------------' from dual   ;
```

・調査対象の SQL 文の SQL_ID の調査方法

1）対象の SQL 文の実行

・SQL 文の実行（コメント句にてコメントを記述すると、後で検索しやすい）

例）select /* list07_1 */ ename from emp where empno = 2 ;

2）v$SQL_PLAN の調査

・system ユーザーで、SQL＊Plus へ接続する

・SQL 文に対する SQL_ID の検索

Oracle9 g の場合は、hash_value と address を検索

v$SQL ビューの中から、sql_text 列が SQL 文の文字列と一致するレコードを検索する

select sql_id from v$sql where sql_text like '文字列%' ;

例）select SQL_ID , hash_value, address from v$sql

where sql_text like 'select /* list07_1 */%' ;

SQL_ID	CHILD_NUMBER	HASH_VALUE	ADDRESS
9bumbu1kq5fsc	0	1700969228	B6A4D41C
9bumbu1kq5fsc	1	1700969228	B6A4D41C

もしくは、DBA_HIST_SQLTEXT ビュー

select sql_id from DBA_HIST_SQLTEXT

where sql_text like '文字列%' ;

※ SQL 文の文字列から検索を行った場合の SQL_ID 値の調査が上の例です

１つの SQL 文に複数の実行計画が存在する場合には、同一の SQL_ID が２レコード表示されます

この場合には、SQL 文に対する SQL_ID 値と CHILD_NUMBER 値を指定する必要があります

list07_2.sql スクリプトのソース･コードの Select 文の Where 条件句を修正して、この２つを指定して実行すること

Ⓑ ライブラリキャッシュ内にある SQL 実行計画を出力する方法

・ライブラリキャッシュ内にある SQL 実行計画の情報を出力する

※ SQL 文の文字列から検索を行った場合の実行例を書きに示す
しかし、1 つの SQL 文に複数の実行計画が存在する場合には、複数の実行計画が同時に出力されてしまい、調査することができない
この場合には、SQL 文に対する SQL_ID 値と CHILD_NUMBER 値　もしくは、HASH_VALUE 値と ADDRESS 値を先に求めてから、Select 文の Where 条件句にこれを指定して実行する

→ 次ページ

```
column   hash_value format 9999999999
column   id   format   999   newline
column   operation   format   a20
column   options   format   a15
column   object_name   format   a22   trunc
column   optimaizer   format   a3   trunc

select   VP.id ,   lpad ( '' , VP.depth ) || VP.operation   operation ,   VP.options ,
         VP.object_name ,   VP.optimizer ,   VP.cost

  from   v$sql_plan   VP ,   v$sqlarea   VA

  where   VA.hash_value   =   &hash_value
    and   VP.hash_value   =   VA.hash_value
    and   VP.address   =   VA.address

  start   with   VP.id   =   0
  connect   by
     (   prior   VP.id   =   VP.parent_id
           and   prior   VP.hash_value   =   VP.hash_value
           and   prior   VP.child_number   =   VP.child_number
     )
  order   siblings   by   VP.id ,   VP.position   ;
```

VA.sql_text like 'select /* list07_1 */%'の直接 SQL 文の条件指定では、Select 文が複数存在した時に、Oracle への負荷が重くなって全体のレスポンスに影響を与えた

実行結果（実行計画の表示）

ID	OPERATION	OPTION	OBJECT_NAME	OPTIMIZER ↓	COST
0	SELECT STATEMEN	(null)	(null)	ALL_ROWS	6
1	NESTED LOOPS	(null)	(null)	(null)	6
2	NESTED LOOPS	(null)	(null)	(null)	6
3	NESTED LOOPS	(null)	(null)	(null)	6
4	TABLE ACCESS	FULL	EMP	(null)	3
5	TABLE ACCESS	BY INDEX ROWID	DEPT	(null)	1
6	INDEX	UNIQUE SCAN	PK_DEPT	(null)	0
7	INDEXUNIQUE SCAN		PK_EMP	(null)	0
8	INDEX UNIQUE SCAN		PK_EMP	(null)	0

・調査対象の SQL 文の SQL_ID と CHILD_NUMBER の調査方法

調査した SQL_ID 値と CHILD_NUMBER 値　もしくは、HASH_VALUE 値と ADDRESS 値を検索条件にして、v$SQL_PLAN ビューの中から一致するレコードを検索する

```
/*    ステップ１    */
/*    調査対象の SQL 文に対する条件値となるキー値を求める                    */
select   substr(sql_text, 1, 40) ,   SQL_ID ,   CHILD_NUMBER ,   hash_value ,
         address
  from   v$sql
  where   sql_text   like   'select   /*   list07_1   */%'   ;
```

↑

/*　　調査したい対象の SQL 文を記述する　　*/

```
/*   注意   ステップ１の実行結果が１つになるまで、条件の SQL 文を記述する   */
/*          SQL 文のすべてを条件に記述しても、実行結果レコードが２件ある   */
/*          場合には、実行計画が２種類あることを意味している               */
/*              （業務途中で、実行計画が切り替わってしまっている）          */
/*          この場合は、次ページの SQL 文で SQL_ID 値と CHILD_NUMBER       */
/*          値（もしくは、HASH_VALUE 値と ADDRESS 値 9g 以前）を条件       */
/*          文に使い実行計画を出力する                                     */
```

・SQL_ID と CHILD_NUMBER からの SQL 文の調査方法

```
/*    ステップ２    */
/*   SQL_ID 値と CHILD_NUMBER 値（HASH_VALUE 値と ADDRESS 値              */
/*   9g 以前）を検索条件にして、v$SQL_PLAN ビューの                        */
/*   中から一致するレコードを検索するための SQL 文サンプル                 */

column   id   format   999   newline
column   operation   format   a20
column   options   format   a15
column   object_name   format   a22   trunc
column   optimaizer   format   a3   trunc

----   10 g 以降
select   id ,   lpad ( '' , depth ) || operation   operation ,   options ,   object_name ,
                optimizer ,   cost
       from   v$sql_plan
       where   SQL_ID   =   '&SQL_ID'
         and   CHILD_NUMBER   =   &CHILD_NUMBER
       start   with   id   =   0
       connect   by   (          prior   id   =   parent_id
                       and   prior   hash_value   =   hash_value
                       and   prior   child_number   =   child_number
                     )
       order   siblings   by   id ,   position   ;
```

```
----  9g 以前
  select  id ,  lpad ( '' , depth ) || operation  operation ,  options ,  object_name ,
                  optimizer ,  cost
        from  v$sql_plan
        where  hash_value  =  &hash_value  and  address  =  '&address'
        start  with  id  =  0
        connect  by  (        prior  id  =  parent_id
                        and  prior  hash_value  =  hash_value
                        and  prior  child_number  =  child_number
                      )
        order  siblings  by  id ,  position  ;
----
```

```
/*  実行すると、SQL_ID の変数の値を入力する                          */
/*  もしくは、SQL 文の&hash_value と&address の変数の値を入力する    */
```

```
enter  value  by  for  SQL_ID :              <SQL_ID 値>
enter  value  by  for  CHILD_NUMBER :        < CHILD_NUMBER 値>

enter  value  by  for  hash_value :          <ハッシュ・バリュー値>
enter  value  by  for  address :             <アドレス値>
```

求めた値を入力する

実行結果（実行計画の表示）

ID	OPERATION	OPTION	OBJECT_NAME	OPTIMIZER ↓	COST
0	SELECT STATEMEN	(null)	(null)	ALL_ROWS	6
1	NESTED LOOPS	(null)	(null)	(null)	6
2	NESTED LOOPS	(null)	(null)	(null)	6
3	NESTED LOOPS	(null)	(null)	(null)	6
4	TABLE ACCESS	FULL	EMP	(null)	3
5	TABLE ACCESS	BY INDEX ROWID	DEPT	(null)	1
6	INDEX	UNIQUE SCAN	PK_DEPT	(null)	0
7	INDEXUNIQUE SCAN		PK_EMP	(null)	0
8	INDEX UNIQUE SCAN		PK_EMP	(null)	0