**有关Oracle 的执行计划说明，参考：Oracle Explain Plan 见 <http://www.linuxidc.com/Linux/2011-10/44226.htm>**

**v$process**

**v$session**

**v$sql**

**v$sqlare**

**v$dba\_objects**

**dba\_sql\_profiles**

如果要分析某条SQL的性能问题，通常我们要先看SQL的执行计划，看看SQL的每一步执行是否存在问题。 如果一条SQL平时执行的好好的，却有一天突然性能很差，如果排除了系统资源和阻塞的原因，那么基本可以断定是执行计划出了问题。

看懂执行计划也就成了SQL优化的先决条件。

**查看执行计划的方法**

1.Explain Plan For SQL

•丌实际执行SQL诧句，生成的计划未必是真实执行的计划

•必须要有plan\_table

SQL> EXPLAIN PLAN FOR SELECT \* FROM DAVE;

2.SQLPLUS AUTOTRACE

•除set autotrace traceonly explain外均实际执行SQL，但仍未必是真实计划

•必须要有plan\_table

set autotrace on;

3.awr中查看

select plan\_table\_output from table(dbms\_xplan.display\_awr('18r3zxbnn9b1d'));

or

?/rdbms/admin/awrsqrpt

?/rdbms/admin/sqtrpt  ---生成sql调整计划！！！！！！！

4 内存中

select plan\_table\_output from table(dbms\_xplan.display\_cursor('18r3zxbnn9b1d'));

select plan\_table\_output from table(dbms\_xplan.display\_cursor('2wwnx4vy33dy8'))--换成那个sql

5.SQL TRACE

•需要启用10046戒者SQL\_TRACE

•一般用tkprof看的更清楚些，当然10046里本身也有执行计划信息

6.V$SQL和V$SQL\_PLAN

•可以查询到多个子游标的计划信息了，但是看起来比较费劲

依据sqlid

7.Enterprise Manager

•可以图形化显示执行计划，但并非所有环境有EM可用

8.其他第三方工具

•注意 PL/SQL developer之类工具F5看到的执行计划未必是真实的

===============================================================

查看执行计划的方法:更靠谱的方法

最详细的执行计划信息收集

alter session set STATISTICS\_LEVEL = ALL; --不设置无法获得A-ROWS等信息

select d.dname,e.empno,e.ename,e.job from emp e,dept d where e.deptno=d.deptno;

**select \* from table(dbms\_xplan.DISPLAY\_CURSOR(null, null, 'ALLSTATS'));**

Execution Plan

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

Plan hash value: 844388907

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

| Id  | Operation                    | Name    | Rows  | Bytes | Cost (%CPU)| Time     |

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

|   0 | SELECT STATEMENT             |         |    14 |   476 |     6  (17)| 00:00:01 |

|   1 |  MERGE JOIN                  |         |    14 |   476 |     6  (17)| 00:00:01 |

|   2 |   TABLE ACCESS BY INDEX ROWID| DEPT    |     4 |    52 |     2   (0)| 00:00:01 |

|   3 |    INDEX FULL SCAN           | PK\_DEPT |     4 |       |     1   (0)| 00:00:01 |

|\*  4 |   SORT JOIN                  |         |    14 |   294 |     4  (25)| 00:00:01 |

|   5 |    TABLE ACCESS FULL         | EMP     |    14 |   294 |     3   (0)| 00:00:01 |

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

Predicate Information (identified by operation id):

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

   4 - access("E"."DEPTNO"="D"."DEPTNO")

       filter("E"."DEPTNO"="D"."DEPTNO")

Statistics

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

          1  recursive calls

          0  db block gets

         11  consistent gets

          0  physical reads

          0  redo size

       1108  bytes sent via SQL\*Net to client

        551  bytes received via SQL\*Net from client

          2  SQL\*Net roundtrips to/from client

          1  sorts (memory)

          0  sorts (disk)

         14  rows processed

SQL>

alter session set STATISTICS\_LEVEL = TYPICAL;

E-Rows 是优化器评估返回的行数

A-Rows 是实际执行

===========================================================================

sql执行计划说明

生成SQL的执行计划是Oracle在对SQL做硬解析时的一个非常重要的步骤，它制定出一个方案告诉Oracle在执行这条SQL时以什么样的方式访问数据：索引还是全表扫描，是Hash Join还是Nested loops Join等。 比如说某条SQL通过使用索引的方式访问数据是最节省资源的，结果CBO作出的执行计划是全表扫描，那么这条SQL的性能必然是比较差的。

       Oracle SQL的硬解析和软解析

<http://blog.csdn.net/tianlesoftware/archive/2010/04/08/5458896.aspx>

在Toad 里面，很清楚的显示了执行的顺序。 但是如果在SQLPLUS里面就不是那么直接。 但我们也可以判断：一般按缩进长度来判断，缩进最大的最先执行，如果有2行缩进一样，那么就先执行上面的。

1 执行计划中字段的解释

         plan hash value：3297093041

这一行是这一条语句的的hash值，我们知道ORACLE对每一条ORACLE语句产生的执行计划放在SHARE POOL里面，第一次要经过硬解析，产生hash值。下次再执行时比较hash值，如果相同就不会执行硬解析。

              Id: 执行序列，但不是执行的先后顺序。执行的先后根据Operation缩进来判断（采用最右最上最先执行的原则看层次关系，在同一级如果某个动作没有子ID就最先执行。一般按缩进长度来判断，缩进最大的最先执行，如果有2行缩进一样，那么就先执行上面的。）

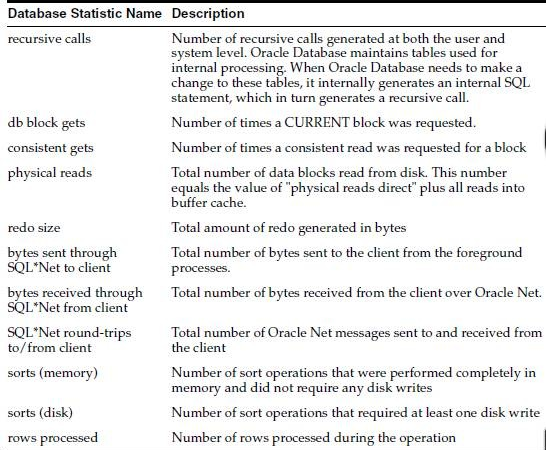
　　　　Operation：当前操作的内容。   
　　　　Name：操作对象   
　　　　Rows：也就是10g版本以前的Cardinality(基数)，Oracle估计当前操作的返回结果集行数。   
　　　　Bytes：表示执行该步骤后返回的字节数。   
　　　　Cost（CPU）：表示执行到该步骤的一个执行成本，用于说明SQL执行的代价。   
　　　　Time：Oracle 估计当前操作的时间。

2 谓词说明

Predicate Information (identified by operation id):

　　　　Access: 表示这个谓词条件的值将会影响数据的访问路劲（全表扫描还是索引）。   
　　　　Filter：表示谓词条件的值不会影响数据的访问路劲，只起过滤的作用。   
　　　　在谓词中主要注意access，要考虑谓词的条件，使用的访问路径是否正确。

3 统计信息



db block gets ： 从buffer cache中读取的block的数量

consistent gets： 从buffer cache中读取的undo数据的block的数量

physical reads： 从磁盘读取的block的数量

redo size： DML生成的redo的大小

sorts (memory) ：在内存执行的排序量

sorts (disk) ：在磁盘上执行的排序量

  Physical Reads通常是我们最关心的，如果这个值很高，说明要从磁盘请求大量的数据到Buffer Cache里，通常意味着系统里存在大量全表扫描的SQL语句，这会影响到数据库的性能，因此尽量避免语句做全表扫描，对于全表扫描的SQL语句，建议增 加相关的索引，优化SQL语句来解决。

4

**动态分析**

Note   
------------   
-dynamic sampling used for the statement

这提示用户CBO当前使用的技术，需要用户在分析计划时考虑到这些因素。 当出现这个提示，说明当前表使用了动态采样。我们从而推断这个表可能没有做过分析。

这里会出现两种情况：   
（1） 如果表没有做过分析，那么CBO可以通过动态采样的方式来获取分析数据，也可以或者正确的执行计划。   
（2） 如果表分析过，但是分析信息过旧，这时CBO就不会在使用动态采样，而是使用这些旧的分析数据，从而可能导致错误的执行计划。

**===============================================================**

**这个plan\_sql\_value是sql语句的执行计划，**

**sql\_hash\_value是对sql\_address做hash运算以后，得到的bucket的位置。  
 一个sql hash value所指示的bucket上可以有多个sql address。  
 也就是说，会出现相同的sql\_hash\_value对应多个不同的sql\_address**

**一个sql address 也会有多个不同的sql hash value**

**==========================================================**

<<Maclean Liu的Oracle性能优化讲座 第一回-真正读懂Oracle SQL执行计划Execution Plan .pdf>>

<<b45424758af2654de1cc8b7f5976c334.pdf>>