# [SQL Tuning Advisor使用实例](http://www.cnblogs.com/einyboy/archive/2012/08/14/2637659.html)

在oracle10g之前，想要优化一个sql语句是比较麻烦，但是在oracle10g这个版本推出的SQL Tuning Advisor这个工具，能大大减少sql调优的工作量，不过要想使用SQL Tuning Advisor，一定要保证你的优化器是CBO模式。

1.首先需要创建一个用于调优的用户bamboo,并授予advisor给创建的用户

SQL> create user bamboo identified by bamboo;

User created.

SQL> grant connect,resource to bamboo;

Grant succeeded.

SQL> grant advisor to bamboo;

Grant succeeded.

2.创建用户做测试的2张表,大表里面插入500万条数据，小表里面插入10万条数据，其创建方法如下

SQL> create table bigtable (id number(10),name varchar2(100));

Table created.

SQL> begin

  2  for i in 1..5000000 loop

  3  insert into bigtable values(i,'test'||i);

  4  end loop;

  5  end;

  6  /

PL/SQL procedure successfully completed.

SQL> commti;

SQL> create table smalltable (id number(10),name varchar2(100));

Table created.

SQL> begin

  2  for i in 1..100000 loop

  3  insert into smalltable values(i,'test'||i);

  4  end loop;

  5  end;

  6  /

PL/SQL procedure successfully completed.

SQL> commti;

3.然后对bigtable和smalltable做一个等连接查询，然后跟踪其执行计划

SQL> select a.id,a.name,b.id,b.name from bigtable a,smalltable b where a.id=b.id and a.id=40000;

        ID NAME                                             ID NAME

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

     40000 test40000                                     40000 test40000

Execution Plan

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

Plan hash value: 1703851322

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

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

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

|   0 | SELECT STATEMENT   |            |   839 |   106K|  3656   (5)| 00:00:44 |

|\*  1 |  HASH JOIN         |            |   839 |   106K|  3656   (5)| 00:00:44 |

|\*  2 |   TABLE ACCESS FULL| SMALLTABLE |     5 |   325 |    71   (3)| 00:00:01 |

|\*  3 |   TABLE ACCESS FULL| BIGTABLE   |   173 | 11245 |  3584   (5)| 00:00:44 |

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

Predicate Information (identified by operation id):

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

   1 - access("A"."ID"="B"."ID")

   2 - filter("B"."ID"=40000)

   3 - filter("A"."ID"=40000)

Note

-----

   - dynamic sampling used for this statement

Statistics

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

          9  recursive calls

          0  db block gets

      16151  consistent gets

      11469  physical reads

          0  redo size

        588  bytes sent via SQL\*Net to client

        385  bytes received via SQL\*Net from client

          2  SQL\*Net roundtrips to/from client

          2  sorts (memory)

          0  sorts (disk)

          1  rows processed

熟悉执行计划的就可以看出，这个sql执行是很慢的，2个表都做的是全表扫描，并且其物理读是11469，按照优化的经验，给2个表的id创建索引，减少查询时候的物理读,下面我们就看看通过优化器，oracle能我们什么样的建议呢？

4.下面就通过DBMS\_SQLTUNE包的CREATE\_TUNING\_TASK来创建一个优化任务，然后通过DBMS\_SQLTUNE.EXECUTE\_TUNING\_TASK来执行调优任务，生成调优建议

SQL> DECLARE

  2    my\_task\_name VARCHAR2(30);

  3    my\_sqltext CLOB;

  4  BEGIN

  5    my\_sqltext := 'select a.id,a.name,b.id,b.name from bigtable a,smalltable b where a.id=b.id and a.id=40000';

  6

  7    my\_task\_name := DBMS\_SQLTUNE.CREATE\_TUNING\_TASK(

  8                            sql\_text => my\_sqltext,

  9                            user\_name => 'SCOTT',

10                             scope => 'COMPREHENSIVE',

11                             time\_limit => 60,

12                             task\_name => 'test\_sql\_tuning\_task1',

13                             description => 'Task to tune a query');

14     DBMS\_SQLTUNE.EXECUTE\_TUNING\_TASK(task\_name => 'test\_sql\_tuning\_task1');

15  END;

16  /

5.执行的过程中，也可以通过user\_advisor\_tasks或者dba\_advisor\_tasks来查看调优任务执行的状况

SQL> select task\_name,ADVISOR\_NAME,STATUS from user\_advisor\_tasks;

TASK\_NAME                      ADVISOR\_NAME                             STATUS

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

test\_sql\_tuning\_task1          SQL Tuning Advisor                       COMPLETED

如果status是EXECUTING，则表示任务正在执行，如果为COMPLETED，则任务已经执行完毕

6.通过调用dbms\_sqltune.report\_tuning\_task可以查询调优的结果，不过在查询结果之前，得设置sqlplus的环境，如果不设置，则查询的结果出不来

SQL> set long 999999

SQL> set LONGCHUNKSIZE 999999

SQL> set serveroutput on size 999999

SQL> set linesize 200

SQL> select dbms\_sqltune.report\_tuning\_task('test\_sql\_tuning\_task1') from dual;

SQL> select dbms\_sqltune.report\_tuning\_task('test\_sql\_tuning\_task1') from dual;

DBMS\_SQLTUNE.REPORT\_TUNING\_TASK('TEST\_SQL\_TUNING\_TASK1')

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

GENERAL INFORMATION SECTION

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

Tuning Task Name                  : test\_sql\_tuning\_task1

Tuning Task Owner                 : BAMBOO

Scope                             : COMPREHENSIVE

Time Limit(seconds)               : 60

Completion Status                 : COMPLETED

Started at                        : 10/13/2011 05:07:53

Completed at                      : 10/13/2011 05:08:18

Number of Statistic Findings      : 2

Number of Index Findings          : 1

DBMS\_SQLTUNE.REPORT\_TUNING\_TASK('TEST\_SQL\_TUNING\_TASK1')

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

Schema Name: SCOTT

SQL ID     : 7arau1k5a3mv1

SQL Text   : select a.id,a.name,b.id,b.name from bigtable a,smalltable b

             where a.id=b.id and a.id=40000

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

FINDINGS SECTION (3 findings)

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

DBMS\_SQLTUNE.REPORT\_TUNING\_TASK('TEST\_SQL\_TUNING\_TASK1')

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

1- Statistics Finding

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

  Table "SCOTT"."SMALLTABLE" was not analyzed.

  Recommendation

  --------------

  - Consider collecting optimizer statistics for this table.

    execute dbms\_stats.gather\_table\_stats(ownname => 'SCOTT', tabname =>

            'SMALLTABLE', estimate\_percent => DBMS\_STATS.AUTO\_SAMPLE\_SIZE,

            method\_opt => 'FOR ALL COLUMNS SIZE AUTO');

DBMS\_SQLTUNE.REPORT\_TUNING\_TASK('TEST\_SQL\_TUNING\_TASK1')

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

  Rationale

  ---------

    The optimizer requires up-to-date statistics for the table in order to

    select a good execution plan.

2- Statistics Finding

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

  Table "SCOTT"."BIGTABLE" was not analyzed.

  Recommendation

  --------------

DBMS\_SQLTUNE.REPORT\_TUNING\_TASK('TEST\_SQL\_TUNING\_TASK1')

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

  - Consider collecting optimizer statistics for this table.

    execute dbms\_stats.gather\_table\_stats(ownname => 'SCOTT', tabname =>

            'BIGTABLE', estimate\_percent => DBMS\_STATS.AUTO\_SAMPLE\_SIZE,

            method\_opt => 'FOR ALL COLUMNS SIZE AUTO');

  Rationale

  ---------

    The optimizer requires up-to-date statistics for the table in order to

    select a good execution plan.

3- Index Finding (see explain plans section below)

DBMS\_SQLTUNE.REPORT\_TUNING\_TASK('TEST\_SQL\_TUNING\_TASK1')

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

  The execution plan of this statement can be improved by creating one or more

  indices.

  Recommendation (estimated benefit: 100%)

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

  - Consider running the Access Advisor to improve the physical schema design

    or creating the recommended index.

    create index SCOTT.IDX$$\_00790001 on SCOTT.SMALLTABLE('ID');

  - Consider running the Access Advisor to improve the physical schema design

DBMS\_SQLTUNE.REPORT\_TUNING\_TASK('TEST\_SQL\_TUNING\_TASK1')

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

    or creating the recommended index.

    create index SCOTT.IDX$$\_00790002 on SCOTT.BIGTABLE('ID');

  Rationale

  ---------

    Creating the recommended indices significantly improves the execution plan

    of this statement. However, it might be preferable to run "Access Advisor"

    using a representative SQL workload as opposed to a single statement. This

    will allow to get comprehensive index recommendations which takes into

    account index maintenance overhead and additional space consumption.

DBMS\_SQLTUNE.REPORT\_TUNING\_TASK('TEST\_SQL\_TUNING\_TASK1')

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

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

EXPLAIN PLANS SECTION

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

1- Original

-----------

Plan hash value: 1703851322

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

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

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

DBMS\_SQLTUNE.REPORT\_TUNING\_TASK('TEST\_SQL\_TUNING\_TASK1')

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

|   0 | SELECT STATEMENT   |            |   839 |   106K|  3656   (5)| 00:00:44 |

|\*  1 |  HASH JOIN         |            |   839 |   106K|  3656   (5)| 00:00:44 |

|\*  2 |   TABLE ACCESS FULL| SMALLTABLE |     5 |   325 |    71   (3)| 00:00:01 |

|\*  3 |   TABLE ACCESS FULL| BIGTABLE   |   173 | 11245 |  3584   (5)| 00:00:44 |

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

Predicate Information (identified by operation id):

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

   1 - access("A"."ID"="B"."ID")

   2 - filter("B"."ID"=40000)

DBMS\_SQLTUNE.REPORT\_TUNING\_TASK('TEST\_SQL\_TUNING\_TASK1')

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

   3 - filter("A"."ID"=40000)

2- Using New Indices

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

Plan hash value: 3720188830

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

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

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

|   0 | SELECT STATEMENT              |                |     1 |   130 |     5   (0)| 00:00:01 |

|   1 |  TABLE ACCESS BY INDEX ROWID  | BIGTABLE       |     1 |    65 |     3   (0)| 00:00:01 |

DBMS\_SQLTUNE.REPORT\_TUNING\_TASK('TEST\_SQL\_TUNING\_TASK1')

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

|   2 |   NESTED LOOPS                |                |     1 |   130 |     5   (0)| 00:00:01 |

|   3 |    TABLE ACCESS BY INDEX ROWID| SMALLTABLE     |     1 |    65 |     2   (0)| 00:00:01 |

|\*  4 |     INDEX RANGE SCAN          | IDX$$\_00790001 |     1 |       |     1   (0)| 00:00:01 |

|\*  5 |    INDEX RANGE SCAN           | IDX$$\_00790002 |     1 |       |     2   (0)| 00:00:01 |

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

Predicate Information (identified by operation id):

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

   4 - access("B"."ID"=40000)

   5 - access("A"."ID"=40000)

  从上面的结果可以看到oracle的调优顾问给我们3条建议：

（1）SCOTT.SMALLTABLE表没有做分析，需要做一下表结构的分析，并且给出一个分析的建议，如下所示

     execute dbms\_stats.gather\_table\_stats(ownname => 'SCOTT', tabname =>

            'SMALLTABLE', estimate\_percent => DBMS\_STATS.AUTO\_SAMPLE\_SIZE,

            method\_opt => 'FOR ALL COLUMNS SIZE AUTO');

（2）SCOTT.BIGTABLE表没有做分析，需要做一下表结构的分析，并且给出一个分析的建议，如下所示

     execute dbms\_stats.gather\_table\_stats(ownname => 'SCOTT', tabname =>

            'BIGTABLE', estimate\_percent => DBMS\_STATS.AUTO\_SAMPLE\_SIZE,

            method\_opt => 'FOR ALL COLUMNS SIZE AUTO');

（3）oracle建议我们在表SCOTT.SMALLTABLE，SCOTT.BIGTABLE的id列创建一个bitree索引，给的建议如下

      create index SCOTT.IDX$$\_00790002 on SCOTT.BIGTABLE('ID');

      create index SCOTT.IDX$$\_00790001 on SCOTT.SMALLTABLE('ID');

    当然创建索引的名字可以改成别的名字

    通过以上查看oracle的调优顾问给的建议，基本和我们在前面给出的调优方案是一致，因此当我们给一个大的ＳＱＬ做优化的时候，可以先使用oracle调优顾问，得到一些调优方案，然后根据实际情况做一些调整就可以。

 以下就是执行oracle调优顾问的建议，重新执行select a.id,a.name,b.id,b.name from bigtable a,smalltable b where a.id=b.id and a.id=40000这天语句得到的执行计划，可以看出查询时间和物理读大大减少

 SQL> select a.id,a.name,b.id,b.name from bigtable a,smalltable b where a.id=b.id and a.id=40000;

        ID NAME                                             ID NAME

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

     40000 test40000                                     40000 test40000

Execution Plan

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

Plan hash value: 777647921

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

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

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

|   0 | SELECT STATEMENT              |                 |     1 |    31 |     5   (0)| 00:00:01 |

|   1 |  TABLE ACCESS BY INDEX ROWID  | BIGTABLE        |     1 |    17 |     3   (0)| 00:00:01 |

|   2 |   NESTED LOOPS                |                 |     1 |    31 |     5   (0)| 00:00:01 |

|   3 |    TABLE ACCESS BY INDEX ROWID| SMALLTABLE      |     1 |    14 |     2   (0)| 00:00:01 |

|\*  4 |     INDEX RANGE SCAN          | I\_ID\_SAMLLTABLE |     1 |       |     1   (0)| 00:00:01 |

|\*  5 |    INDEX RANGE SCAN           | I\_ID\_BIGTABLE   |     1 |       |     2   (0)| 00:00:01 |

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

Predicate Information (identified by operation id):

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

   4 - access("B"."ID"=40000)

   5 - access("A"."ID"=40000)

Statistics

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

          0  recursive calls

          0  db block gets

          9  consistent gets

          0  physical reads

          0  redo size

        588  bytes sent via SQL\*Net to client

        385  bytes received via SQL\*Net from client

          2  SQL\*Net roundtrips to/from client

          0  sorts (memory)

          0  sorts (disk)

          1  rows processed