CSCI317 – Report4 - Samuel Ian Black – SIB979 - 6025821

Table of Contents

CSCI317 – Report4 - Samuel Ian Black – SIB979 - 6025821

Note that before each run of a command time java task4 the below query is run.

```
CONNECT SYSTEM/oracle;
ALTER SYSTEM FLUSH SHARED_POOL;
```

Original time of query:

```
[oracle@localhost Exam]$ time java task4

Connected as tpchr user

Part key: 4471

Part key: 8526

Part key: 12224

Part key: 14988

Part key: 20143

Part key: 43902

Part key: 45138

Part key: 46172

Part key: 50506

Part key: 51788

Part key: 57573

Part key: 58525

Done.


real 0m1.307s

user 0m1.233s

sys  0m0.058s
```

Total time = 1.307 + 1.233 + 0.058 = **2.598s**

CSCI317 – Report4 - Samuel Ian Black – SIB979 - 6025821

Optimization #1 – Transformation of Select Statement

(1) Description of Improvement:

The results from the table `BRAND55` are every row from the table `PART` where the column `P_BRAND = 'Brand#55'`.
The results from the table NICKEL are every row from the table `PART` where the column `P_TYPE = 'ECONOMY BRUSHED NICKEL'`.
The results from the table `TEMP12` are every row from the table `BRAND55`, where `P_TYPE <> 'ECONOMY BRUSHED NICKEL'`.
The results from the original query are every row from the table `BRAND55` where `P_PARTKEY NOT IN` the table `TEMP12`.
This means that the query is returning every row from the table PART where P_BRAND = 'Brand#55' AND P_TYPE = 'ECONOMY BRUSHED NICKEL'.
Therefore, the query can be simplified to the below without the need for creation of tables.

```
Statement stmt = conn.createStatement();
ResultSet rset = stmt.executeQuery(
   "SELECT P_PARTKEY " +
   "FROM PART " +
   "WHERE P_BRAND = 'Brand#55' " +
   "AND P_TYPE = 'ECONOMY BRUSHED NICKEL' " +
   "ORDER BY P_PARTKEY" );
```

(2) Benefits of Improvement:

1. No longer using persistent storage to create multiple relational tables
2. Improved query is much more readable
3. The improved query will also improve performance because of improvement 1.

```
[oracle@localhost Exam]$ time java task4
Connected as tpchr user
Part key: 4471
Part key: 8526
Part key: 12224
Part key: 14988
Part key: 20143
```

```
Part key: 43902
Part key: 45138
Part key: 46172
Part key: 50506
Part key: 51788
Part key: 57573
Part key: 58525
Done.

real 0m0.839s
user 0m1.168s
sys  0m0.054s
```

real before improvement = **1.307s**
real improvement = 1.307 - 0.839 = **0.468s**

user before improvement = **1.233s**
user improvement = 1.233 - 1.168 = **0.065s**

sys before improvement = **0.058s**
sys improvement = 0.058 – 0.054 = **0.004s**

Total time = 0.839 + 1.168 + 0.054 = **2.061s**
Total time improvement = 2.598 - 2.061 = **0.537s**

(3) Costs of Improvement:
There is no cost increase associated with this improvement.

(4) Report from improvement:

```
import java.sql.*;

class task4
{
  public static void main (String args [])
       throws SQLException, ClassNotFoundException
  {
    // Load the Oracle JDBC driver
    Class.forName ("oracle.jdbc.driver.OracleDriver");
    Connection conn = DriverManager.getConnection
       ("jdbc:oracle:thin:@localhost:1521:db",  "tpchr", "oracle");
      System.out.println( "Connected as tpchr user");
  try{

        Statement stmt = conn.createStatement();
        ResultSet rset = stmt.executeQuery(
           "SELECT P_PARTKEY " +
           "FROM PART " +
           "WHERE P_BRAND = 'Brand#55' " +
           "AND P_TYPE = 'ECONOMY BRUSHED NICKEL' " +
           "ORDER BY P_PARTKEY" );

        while ( rset.next() )
          System.out.println("Part key: " + rset.getInt(1) );

        System.out.println( "Done." );
    }
   catch (SQLException e )
   {
     String errmsg = e.getMessage();
     System.out.println( errmsg );
```

```
    }
  }
}
```

CSCI317 – Report4 - Samuel Ian Black – SIB979 - 6025821

Optimization #2 – Creation of B*Tree Index

(1) Description of Improvement:

Creating an index `CREATE INDEX TASK4IDX1 ON PART(P_BRAND, P_TYPE, P_PARTKEY) COMPRESS 2;` allows for use of an index rather than accessing the entire `PART` table, compression also saves 2MB as opposed to not compressing and further decreases cost of the query since two prefix columns in the `TASK4IDX1` are repeated where as the primary key is unique and therefore, not.

(2) Benefits of Improvement:

The index allows for the query optimizer to traverse through the `TASK4IDX1` index vertically rather than accessing the entire `PART` table.

Time:

```
[oracle@localhost Exam]$ time java task4
Connected as tpchr user
Part key: 4471
Part key: 8526
Part key: 12224
Part key: 14988
Part key: 20143
Part key: 43902
Part key: 45138
Part key: 46172
```

```
Part key: 50506
Part key: 51788
Part key: 57573
Part key: 58525
Done.

real 0m0.778s
user 0m1.118s
sys  0m0.054s
```

real before improvement = **0.839s**
real improvement = 0.839 – 0.778 = **0.061s**

user before improvement = **1.168s**
user improvement = 1.168 - 1.118 = **0.05s**

sys before improvement = **0.054s**
sys improvement = 0.054 – 0.054 = **0s**

Total time = 0.778 + 1.118 + 0.054 = **1.95s**
Total time improvement = 2.061 – 1.95 = **0.111s**

Explain Plan Before Creation of Index

```
SQL> SET ECHO ON

SQL> SET FEEDBACK ON

SQL> SET LINESIZE 300

SQL> SET PAGESIZE 300

SQL>

SQL> EXPLAIN PLAN FOR
  2  SELECT P_PARTKEY
  3  FROM PART
  4  WHERE P_BRAND = 'Brand#55'
  5  AND P_TYPE = 'ECONOMY BRUSHED NICKEL'
  6  ORDER BY P_PARTKEY;


Explained.


SQL>

SQL> @showplan

SQL> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY);


PLAN_TABLE_OUTPUT
```

```
-------------------------------------------------------------------------------

Plan hash value: 2726178166


-------------------------------------------------------------------------------
| Id  | Operation          | Name | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |      |     6 |   234 |   402   (1)| 00:00:01 |
|   1 |  SORT ORDER BY     |      |     6 |   234 |   402   (1)| 00:00:01 |
|*  2 |   TABLE ACCESS FULL| PART |     6 |   234 |   401   (1)| 00:00:01 |
-------------------------------------------------------------------------------


Predicate Information (identified by operation id):
---------------------------------------------------


   2 - filter("P_TYPE"='ECONOMY BRUSHED NICKEL' AND
              "P_BRAND"='Brand#55')
```

Explain Plan After Creation of Index

```
SQL> SET FEEDBACK ON

SQL> SET LINESIZE 300

SQL> SET PAGESIZE 300

SQL>

SQL> CREATE INDEX TASK4IDX1 ON PART(P_BRAND, P_TYPE, P_PARTKEY) COMPRESS 2;


Index TASK4IDX1 created.


SQL>

SQL> EXPLAIN PLAN FOR

  2  SELECT P_PARTKEY

  3  FROM PART

  4  WHERE P_BRAND = 'Brand#55'

  5  AND P_TYPE = 'ECONOMY BRUSHED NICKEL'

  6  ORDER BY P_PARTKEY;


Explained.


SQL>
```

```
SQL> @showplan
SQL> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY);

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------
Plan hash value: 315755535

--------------------------------------------------------------------------------
| Id  | Operation         | Name       | Rows  | Bytes | Cost (%CPU)| Time     |
--------------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |            |    12 |   468 |     2   (0)| 00:00:01 |
|*  1 |  INDEX RANGE SCAN | TASK4IDX1  |    12 |   468 |     2   (0)| 00:00:01 |
--------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   1 - access("P_BRAND"='Brand#55' AND "P_TYPE"='ECONOMY BRUSHED
              NICKEL')
```

Total Cost before improvement = 402 + 402 + 401 = **1,205**

Total Cost after improvement = 2 + 2 = **4**

Total Cost improvement = 1,205 – 4 = **1201**

(3) Costs of Improvement:

The cost of creating the index is 1.5MB in persistent storage.

```
SQL> select sum(bytes)/1024/1024 as "Index Size (MB)" from dba_segments where
segment_name='&INDEX_NAME';

old:select sum(bytes)/1024/1024 as "Index Size (MB)" from dba_segments where
segment_name='&INDEX_NAME'

new:select sum(bytes)/1024/1024 as "Index Size (MB)" from dba_segments where
segment_name='TASK4IDX1'


Index Size (MB)

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

             1.5
```

(4) Report from improvement:

```
SQL> SET ECHO ON
SQL> SET FEEDBACK ON
SQL> SET LINESIZE 300
SQL> SET PAGESIZE 300
SQL>
SQL> CREATE INDEX TASK4IDX1 ON PART(P_BRAND, P_TYPE, P_PARTKEY) COMPRESS 2;

Index TASK4IDX1 created.

SQL>
SQL> SELECT P_PARTKEY
  2  FROM PART
  3  WHERE P_BRAND = 'Brand#55'
  4  AND P_TYPE = 'ECONOMY BRUSHED NICKEL'
  5  ORDER BY P_PARTKEY;

 P_PARTKEY
----------
```

```
 4471
 8526
12224
14988
20143
43902
45138
46172
50506
51788
57573
58525

12 rows selected.
```

CSCI317 – Report4 - Samuel Ian Black – SIB979 - 6025821

Total Costs

Persistent Storage: 87.75MB of 300MB

1. TASK1IDX1 = 6.5MB
2. TASK1IDX2 = 0.5MB
3. TASK2IDX1 = 1.75MB
4. INDEX_TS_32K = 64MB (Size of Tablespace used to calculate Persistent Storage)
    - TASK2IDX2 = 60MB
5. TASK3IDX1 = 13.5MB
6. TASK4IDX1 = 1.5MB

Transient Memory: 88MB of 100MB

1. db_32K_cache_size = 64M
2. db_cache_size = 208
    - Originally 184 + 24 from allocated 100MB Transient Storage expansion