|  |  |  |  |
| --- | --- | --- | --- |
| **Group:** | **308** | **Lab User** | **fsdb308** |
| **Student:** | **Siro Brotón Gutierrez** | **NIA:** | **100496683** |
| **Student:** | **Yan Chen Zhou** | **NIA:** | **100496757** |
| **Student:** | **Ivan Ocheretianyi** | **NIA:** | **100487317** |

# Introduction (~more or less)

It will consist of an introductory paragraph, analyzing the problem to be solved.

Expose initial state: physic design, and prototypic workload (frequent processes) composition and cost. State the goal and summarize the steps you are about to follow for attaining it.

Our task is to implement a new physical design into the existing database so that it optimizes the speed of the retrieval and modification of data. However, our team will focus on optimizing a few selections of operations. An exhaustive analysis of the frequency of each query has already been carried out, indicating that the following **four queries**:

1. select \* from editions where pub\_place='…';
2. select \* from editions where publisher='…';
3. select \* from copies where condition='…';
4. select \* from editions;

are the most important operations performed in our database. The queries follow a proportion of 30%, 30%, 30% and 10% respectively.

To improve their performance, we will use the procedures provided by the package *PKG\_COSTES* and their statistics as the main metric to improve the physical design of our database.

# Analysis (~more or less)

Describe the setting: the current (initial) physical design and the prototypical workload (frequent processes). Add a screenshot with the (initial) average cost of the workload.

For each sentence in the workload (each query) find out, explain, and analyze the execution plan and basic statistics (focusing on consistent gets and timing). Point out the weaknesses and strengths of the initial physical design according to the needs of your specific problem (the workload). Incorporate screenshots of performance as you deem appropriate.

Propose improvements to the physical design based on the analysis of each instruction run individually, and comment on the expected benefits and the drawbacks that it could bring about on the global system (if any).

The initial physical design of all the tables defined in the database are the default settings: a serial non-consecutive using indexes on PKs with BS of 8KB, pctfree of 10% and pctused of 60%. To evaluate the performance of this design, we will simulate the four most frequent queries using the procedure PKG\_COSTES.PR\_WORKLOAD. This procedure consists of a sequence of 10 distinct queries performed on the database that follows proportions found from the previous section. We performed a test run of the workload by iterating 25 times and averaging out the results using PKG\_COSTES.RUN\_TEST, returning the following results.

|  |
| --- |
| SQL> EXEC PKG\_COSTES.RUN\_TEST(25);  Iteration 1  Iteration 2  …  Iteration 24  Iteration 25  RESULTS AT 01/05/2025 21:10:03  TIME CONSUMPTION (run): 1089.24 milliseconds.  CONSISTENT GETS (workload):60473 acc  CONSISTENT GETS (weighted average):6047.3 acc |

EXPLICATION OF RESULTS

We can see that the average response time of the SQL server is 1 second and the total number of accesses to secondary storage ???? is 60473.

ANALYZE POSSIBLE IMPROVEMENTS ARMENIAN BOY AND SIRIAN GIRL CAN DO

All the queries are selections performed on either the *copies* table or the *editions* table. Queries 1, 2, 3 fetches a selection of rows following a certain criterion and Query 4 selects all the rows of a table. Since Query 4 is the least frequent query among the four, we should focus on improving the efficiency of the first three queries.

To further understand the structure of the two tables, we have gathered their statistics with GATHER\_TABLE\_STATS and obtained.

|  |
| --- |
| SQL> EXEC DBMS\_STATS.GATHER\_TABLE\_STATS('SCHEMA\_NAME', 'EDITIONS');  …  SQL> EXEC DBMS\_STATS.GATHER\_TABLE\_STATS('SCHEMA\_NAME', 'COPIES');  …  SQL> select table\_name, avg\_row\_len, num\_rows, blocks from user\_tables;  TABLE\_NAME AVG\_ROW\_LEN NUM\_ROWS BLOCKS  ------------------ ----------- ---------- ----------  COPIES 24 241572 1000  EDITIONS 219 240632 7552 |

……..

Another possible optimization is to move the tables to another tablespace, in our database, we have defined 3 different tablespaces

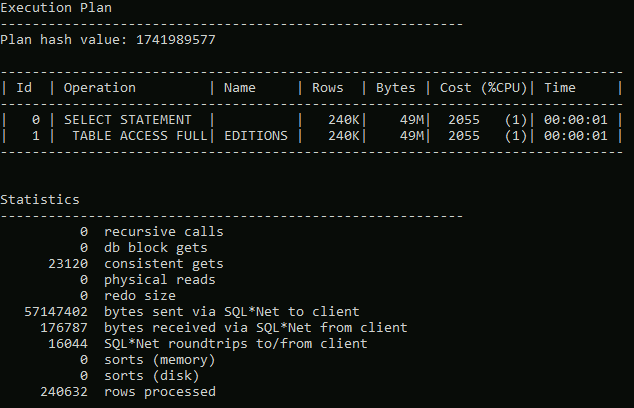
1. CREATE TABLESPACE TAB\_2k DATAFILE 'TAB\_2k.dbf' BLOCKSIZE 2048;
2. CREATE TABLESPACE TAB\_8k DATAFILE 'TAB\_8k.dbf' BLOCKSIZE 8192;
3. CREATE TABLESPACE TAB\_16k DATAFILE 'TAB\_16k.dbf' BLOCKSIZE 16384;

where the second tablespace is the default one.

Modifying pctfree and pctused of *editions* and *copies* depends on the frequency of select, insert and delete operations. This is unknown for now, so no changes will be performed to these two parameters.

Our proposition after the analysis is to index by pub\_place and publisher in *editions* and condition in *copies* and create a cluster for each.

STATS ON ORIGINAL EDITION TABLE



# Physical Design (x)

Following the analysis done in the previous section, settle a complete physical design of the whole DB (at least one, yet you can propose several alternatives). Improvements may include changes in base organizations (hashing, clustering), auxiliary organizations (indexes), redundancies (immediately refreshed materialized views), DB block (bucket) parameters, etc.

Notice that some improvements aimed at improving a given part of the workload could worsen other operations. Justify every design decision.

## Indexing editions and copies

It massively improves the accesses made on editions, from both pub\_place and publisher.

However, it seems that creating an index to the condition attribute of copies worsens the performance (the smaller the tablespace of the index, the worse the performance).

But if we apply an indexed cluster to *condition* in copies, we see performance in consistent gets and CPU run time.

Overall, the results are the following

|  |
| --- |
| SQL> EXEC PKG\_COSTES.RUN\_TEST(10);  Iteration 1  Iteration 2  …  Iteration 9  Iteration 10  RESULTS AT 02/05/2025 15:30:26  TIME CONSUMPTION (run): 944.7 milliseconds.  CONSISTENT GETS (workload):22898 acc  CONSISTENT GETS (weighted average):2289.8 acc |

CHANGING THE TABLESPACE OF AN EDITIONS’ INDEX GIVE US ALMOST NEGLIGABLE OVERALL RESULTS. EVALUATE THEM INDIVIDUALLY

# Evaluation (x)

You have measured the performance of the initial physical design and stated it. After implementing your improved physical design and measured the new performance, compare both and analyze the results obtained (comment divergences with expected results). Add screenshots for backing your evaluation.

SCREENSHOT OF NEW RESULTS FROM NEW PHYSICAL DESIGN

EXPLICATION OF NEW RESULTS

COMPARISON OF RESULTS

# Concluding Remarks (x)

Firstly, make conclusions on the work and the results obtained. Reflex on (defend or criticize) the achieved result (if you think it is good, explain why).

After stating your results, comment on your achievement through this *labwork*, and all assignments in general: required effort, knowledge gain, progress, etc. You can also propose improvements for further editions (focus, size of the problem, requested items, deadlines, supporting materials, etc.). Finally, you can add comments on the whole course (lacks in the syllabus, issues you would like to study more deeply, non-useful issues, etc.).