School of Information Technology & Computer Science
University of Wollongong

Session: **3, July 2021**
Lecturer: **Janusz R. Getta**

CSCI317 Database Performance Tuning
Singapore 2020-3
Assignment 4

Published on 14 August 2021

Scope

This assignment includes the tasks related to improving performance through application of advanced data manipulation statements of SQL, improving performance through transformation of `SELECT` statements improving performance of JDBC application, improving performance through changing the sizes of data buffer caches and re-allocating the relational tables to data buffer caches, and improving performance through using in-memory column store.

This assignment is due by **Saturday, 28 August 2020, 9.00 pm (sharp) Singaporean Time.**

Please read very carefully information listed below.

This assignment contributes to 15% of the total evaluation in the subject.

A submission procedure is explained at the end of specification.

This assignment consists of 5 tasks and specification of each task starts from a new page.

It is recommended to solve the problems before attending a laboratory class in order to efficiently use supervised laboratory time.

A submission marked by Moodle as "late" is treated as a late submission no matter how many seconds it is late.

A policy regarding late submissions is included in the subject outline.

A submission of compressed files (zipped, gzipped, rared, tared, 7-zipped, lhzed, … etc) is not allowed. The compressed files will not be evaluated.

All files left on Moodle in a state "`Draft(not submitted)`" will not be evaluated.

It is expected that all tasks included within **Assignment 4** will be solved **individually without any cooperation** with the other students. If you have any doubts, questions, etc. please consult your lecturer or tutor during lab classes or office hours. Plagiarism will result in a **FAIL** grade being recorded for the assessment task.

Please read very carefully information included in Prologue section below about software environment to be used in the subject.

Prologue

In this subject we use Oracle 19c database server running under Oracle Linux 7.4 operating system on a virtual machine hosted by VirtualBox. To start Oracle database server you have to start VirtualBox first. If you have not installed VirtualBox on your system yet then it is explained in Cookbook for CSIT115 Recipe 1.1, Step 1 "How to use VirtualBox ?" (`https://www.uow.edu.au/~jrg/115/cookbook/e1-1-frame.html`) how to install and how to start VirtualBox.

When VirtualBox is started, import an appliance included in a file `OracleLinux7.4-64bits-Oracle19c-22-JAN-2020.ova`. You can download `ova` image of the appliance using the links published on Moodle.

When ready, power on a virtual machine `OracleLinux7.4-64bits-Oracle19c-22-JAN-2020`.

A password to a Linux user `ORACLE` is `oracle` and a password to Oracle users `SYSTEM` and `SYS` (database administrators) is also `oracle`. Generally, whenever you are asked about a password then it is always `oracle`, unless you change it.

When logged as a Linux user, you can access Oracle database server either through a command line interface (CLI) `SQLcl` or through Graphical User Interface (GUI) `SQL Developer`.

You can find in Cookbook for CSCI317, Recipe 1, How to access Oracle 19c database server, how to use SQL Developer, how to use basic SQL and SQLcl, and how to create a sample database ? (`https://documents.uow.edu.au/~jrg/317sim/cookbook/e1-2-frame.html`) more information on how to use `SQLcl` and `SQL Developer`.

Tasks

Task 1 (3 marks)

Improving performance through advanced DML.

In this task you must operate on the original state of a sample benchmark `TPC-HR` database. It is explained at the end of **Prologue** section how to return to the original state of the database.

Perform the following steps.

(1) Read and analyse SQL statements included a script file `task1.sql`.

(2) Use a script `showstat.sql` to find the costs of processing of a script `task1.sql`. The script should be used in the following way.

  (2.1) Connect as a user `system`.

  (2.2) Use `cd` command to move to a folder where the scripts `task1.sql` and `showstat.sql` are located.

  (2.3) Process a script to `showstat.sql` to get the performance statistics from processing of a script `task1.sql`. When prompted, enter a name of a script `task1.sql` to be tested.

  (2.4) Save the performance statistics (logical reads, physical reads, and the others) in a text file.

  (2.5) Next, use `ROLLBACK` statement to reverse the database modifications performed by a script `task1.sql`.

(3) Next, improve the performance of the script through implementation of another sequence of SQL statements that do the same job as SQL statements in the original script `task1.sql`. The main objective of the optimization is to minimize the total number of times a relational table `PART` is accessed. Save your solution in a script `solution1.sql`.

When ready, process a script `solution1.sql` and save a report from processing of script file `solution1.sql` in a file `solution1.lst`.

(4) Next, use `ROLLBACK` statement to reverse the database modifications performed by a script `solution1.sql`.

(5) Get the performance statistics from processing of a script `solution1.sql` in the same way as you did it for a script `task1.sql` in a step (2) above.

(6) Include the performance statistics from processing of the scripts `task1.sql` and `solution1.sql` at the end of a report from processing `solution1.lst`.

When processing an improved SQL script `solution1.sql` you must put the following `SQLcl` statements

```
SPOOL solution1
SET ECHO ON
SET FEEDBACK ON
SET LINESIZE 300
SET PAGESIZE 300
```

at the beginning of each SQL script implemented and the following statement at the end of the script

```
SPOOL OFF
```

A report from processing of the script must have NO syntax errors !

The script must be processed with `SQLcl` options `ECHO` and `FEEDBACK` set to `ON` such that all SQL statements processed are included in the report !

A report from processing of the script must have NO syntax errors !

Deliverables
A file `solution1.lst` that contains a report from the processing of a script `solution1.sql`, and the performance statistics obtained from the testing of the scripts `task1.sql` and `solution1.sql`.

Task 2 (3 marks)
Improving performance through transformation of SELECT statements

In this task you must operate on the original state of a sample benchmark TPC-HR database. It is explained at the end of **Prologue** section how to return to the original state of the database.

Consider SELECT statements included in a file task2.sql.

Your task is to find the more efficient implementations for each one of SELECT statements included in a file task2.sql.

Implement SQL script file solution2.sql that performs the following actions.

(1) First, the script finds the query processing plans of SELECT statement included in a file task2.sql. A simple method is to copy the contents of a file task2.sql into a file solution5.sql, add EXPLAIN PLAN clauses and use showplan.sql script to list the processing plans.

(2) Then, in the same way the script finds the query processing plans of the improved SELECT statements.

Hint.
To find the improved versions of SELECT statement you have to understand what information is retrieved by SELECT statement and then try to implement the same query in a simpler way.

Make sure that the improved versions of SELECT statements implement exactly the same retrieval tasks as the original SELECT statements.

When ready process a file solution2.sql and create a report from processing solution2.lst.

To get a report from processing of a file solution2.sql use SQLcl and set the options ECHO and FEEDBACK set to ON such that all SQL statements processed are included in the report !

You must put the following SQLcl statements

```
SPOOL solution2
SET ECHO ON
SET FEEDBACK ON
SET LINESIZE 300
SET PAGESIZE 300
```

at the beginning of each SQL script implemented and the following statement at the end of the script

```
SPOOL OFF
```

A report from processing of the script must have NO syntax errors !

Deliverables
A file `solution2.lst` that contains a report from the processing of a script `solution2.sql`.

Task 3 (3 marks)
Improving performance of JDBC application

In this task you must operate on the original state of a sample benchmark `TPC-HR` database. It is explained at the end of **Prologue** section how to return to the original state of the database.

In this task you must operate on the original state of a sample benchmark `TPC-HR` database. It is explained at the end of **Prologue** section how to return to the original state of the database.

Consider implementation of JDBC application in a file `task3.java`.

Your task is to improve performance of the application.

Implement JDBC application with the same functionality as an application included in a file `task3.java` and save it in a file `solution3.java`.

Use Linux command `time` to measure time spend on processing of the application before and after the improvements in the following way.

```
time java task3
time java solution3
```

Explain in the comments attached at the end of a file `solution3.java` why the original application was slower than the improved one and include the results from testing with `time` command.

Deliverables
A file `solution3.java` with the improved application, with the explanations why the original application was slower than the improved one and with the results from testing with `time` command.

Task 4 (3 marks)
Tunning data buffer caches

In this task you must operate on the original state of a sample benchmark TPC-HR database. It is explained at the end of **Prologue** section how to return to the original state of the database.

Consider SQL script task4.sql that performs many frequent accesses to the relational tables of TPC-HR database.

Perform the following actions.

(1) Connect as a user SYSTEM and flush data buffer cache with a statement
ALTER SYSTEM FLUSH BUFFER_CACHE;

(2) Use a script showstat.sql to find the costs of processing of a script task4.sql. The script should be used in the following way.
(2.1) Connect as a user system.
(2.2) Use cd command to move to a folder where the scripts task4.sql and showstat.sql are located.
(2.3) Process a script to showstat.sql to get the performance statistics from processing of a script task4.sql. When prompted, enter a name of a script task4.sql to be tested.
(2.4) Save the performance statistics (logical reads, physical reads, and the others) in a text file.

There is no need to use ROLLBACK statement to reverse the database modifications this time because a script task4.sql contains only SELECT statements.

(3) Increase the size of System Global Area (SGA) by 200Mb. It is explained in Cookbook, Recipe 6.2 How to find and how to change the values of system initialization parameters ? step 4 How to change a system initialization parameter with ALTER SYSTEM statement and SCOPE option ? how to change the size of SGA.

The objective of this task is to use the increased capacity of SGA to increase the sizes of DEFAULT, KEEP, and RECYCLE data buffer caches and to assign the relational tables used in SQL statements of task4.sql to DEFAULT, KEEP, and RECYCLE data buffer caches in a way that minimizes the total number of physical operations performed by the script task4.sql.

Assume that you can invest only additional 200Mbytes into all three data buffer caches. Use ALTER SYSTEM statement to set the values of system initialisation parameters
db_cache_size,
db_keep_cache_size,

db_recycle_cache_size.

It is explained in Cookbook, Recipe 6.2 How to find and how to change the values of system initialization parameters ? step 4 How to change a system initialization parameter with ALTER SYSTEM statement and SCOPE option ? how to change the sizes of data buffer caches.

Assign the relational tables of a sample database to the appropriate data buffer caches. It is possible to it with ALTER TABLE statement in the following way.

```
ALTER TABLE ORDERS STORAGE (BUFFER_POOL KEEP);
ALTER TABLE LINEITEM STORAGE (BUFFER_POOL DEFAULT);
```

Implement a script solution4.sql that performs the following actions.

(1) First, the script connects as a user SYSTEM and flushes data buffer cache with a statement: ALTER SYSTEM FLUSH BUFFER_CACHE.

(2) Next, the script changes the size of DEFAULT, KEEP, and RECYCLE buffer caches and assigns the relational tables of TPC-HR database to the appropriate buffer caches.

When ready process a script solution4.sql to create a report solution4.lst.

To get a report from processing of a file solution4.sql use SQLcl and set the options ECHO and FEEDBACK set to ON such that all SQL statements processed are included in the report !

You must put the following SQLcl statements

```
SPOOL solution4
SET ECHO ON
SET FEEDBACK ON
SET LINESIZE 300
SET PAGESIZE 300
```

at the beginning of each SQL script implemented and the following statement at the end of the script

```
SPOOL OFF
```

A report from processing of the script must have NO syntax errors !

Next, repeat the performance testing of a script task4.sql after the modifications to data buffer caches and new assignments of relational tables to data buffer caches.

(1) Connect as a user SYSTEM and flush data buffer cache with a statement

```
ALTER SYSTEM FLUSH BUFFER_CACHE;
```

(2) Use a script showstat.sql to find the costs of processing of a script task4.sql. The script should be used in the following way.
   (2.1) Connect as a user system.
   (2.2) Use cd command to move to a folder where the scripts task4.sql and showstat.sql are located.
   (2.3) Process a script to showstat.sql to get the performance statistics from processing of a script task4.sql. When prompted, enter a name of a script task4.sql to be tested.
   (2.4) Save the performance statistics (logical reads, physical reads, and the others) in a text file.

There is no need to use ROLLBACK statement to reverse the database modifications this time because a script task4.sql contains only SELECT statements.

Finally, manually append at the end of a file solution4.lst the results from performance testing of a script task4.sql **before** changing the size of data buffer caches and allocations of the relational tables to buffer caches and **after** changing the size of data buffer caches and allocations of the relational tables to buffer caches.

Deliverables
A file solution4.lst with the results from processing of a script solution4.sql and with the results from performance testing of a script task4.sql **before and after** changing the size of data buffer caches and allocations of the relational tables to buffer caches.

Task 5 (3 marks)
Using In-Memory Column Store

In this task you must operate on the original state of a sample benchmark TPC-HR database. It is explained at the end of **Prologue** section how to return to the original state of the database.

An objective of this task is to speed up processing of SELECT statements included in a script task5.sql by assigning the relational tables to In-Memory Column Store.

Make sure that the size of In-Memory area is 400Mb. If it is necessary, increase the size of SGA by 400Mb and allocate it to In-Memory Column Store. It is explained in the lecture slides how to do it.

Implement SQL script solution5.sql that performs the following actions.

(1) First, the script finds the query processing plans for SELECT statements included in a script task5.sql.

(2) Next, the script populates In-Memory Column Store with relational tables and/or columns of relational tables used in a script task5.sql to improve performance in the best way.

(3) Finally, the script finds the query processing plans for SELECT statements included in a script task5.sql after populate In-Memory Column Store with relational tables.

When ready process a script solution5.sql to create a report solution5.lst.

To get a report from processing of a file solution5.sql use SQLcl and set the options ECHO and FEEDBACK set to ON such that all SQL statements processed are included in the report !

You must put the following SQLcl statements

```
SPOOL solution5
SET ECHO ON
SET FEEDBACK ON
SET LINESIZE 300
SET PAGESIZE 300
```

at the beginning of each SQL script implemented and the following statement at the end of the script

```
SPOOL OFF
```

A report from processing of the script must have NO syntax errors !

Deliverables

A file `solution5.lst` that contains a report from the processing of a script `solution5.sql`.

Submission

Note, that you have only one submission. So, make it absolutely sure that you submit the correct files with the correct contents. No other submission is possible!

Submit the files **`solution1.lst`**, **`solution2.lst`**, **`solution3.java`**, **`solution4.lst`**, and **`solution5.lst`** through Moodle in the following way:

(1) Access Moodle at **`http://moodle.uowplatform.edu.au/`**
(2) To login use a **`Login`** link located in the right upper corner the Web page or in the middle of the bottom of the Web page
(3) When logged select a site **`CSCI317 (SP321) Database Performance Tuning`**
(4) Scroll down to a section **`Submissions`**
(5) Click at a link **`In this place you can submit the outcomes of Assignment 4`**
(6) Click at a button **`Add Submission`**
(7) Move a file **`solution1.lst`** into an area **`You can drag and drop files here to add them`**. You can also use a link **`Add`**...
(8) Repeat step (7) for the files **`solution2.lst`**, **`solution3.java`**, **`solution4.lst`**, and **`solution5.lst`**.
(9) Click at a button **`Save changes`**
(10) Click at a button **`Submit assignment`**
(11) Click at the checkbox with a text attached: **`By checking this box, I confirm that this submission is my own work,`** ... in order to confirm the authorship of your submission.
(12) Click at a button **`Continue`**

End of specification