National University of Singapore
School of Computing

CS5226: Database Tuning
Semester 2, 2012/13

Lab 3: Lock Contention

Due: 11:59pm April 15, 2013 (Monday)

1 Instructions

Lab 3 is on lock contention. The readings for this lab are available at http://www.comp.nus.edu.sg/~cs5226/lab3-reading.html. A brief summary of basic SQL*PLUS commands are available at http://www.comp.nus.edu.sg/~cs5226/sqlplus.html.

Submit your lab report by following these instructions:

1. Upload your lab report (in pdf format) to CS5226 IVLE's *Lab 3 Submission* workbin by the deadline (April 15). Late submissions will not be graded.
2. Your lab report should be named using your matriculation number as follows: **lab3-matriculationNumber.pdf**. E.g., lab3-a1234567x.pdf or lab3-u123456x.pdf.
3. Your report should include the following:
   - Your name & matriculation number
   - Answers to each of the underlined parts in the questions.

2 Lab Setup

1. Copy the following files over to your dbtune/dbtune2 account:

```
$> cd $HOME/lab
$> cp ~cs5226/lab/lab3.tar .
$> tar xvfp lab3.tar
$> cd lab3
$> sh rename.sh
```

2. Start up the SQL client program as follows:

```
$> sqlplus / as sysdba
```

3. Start up the database with `init.ora` and run `setup.sql`.

```
SQL> startup pfile=init.ora
SQL> @setup
```

3 Question 1

1. Run `q1setup.sql` to create and populate two tables `employee` and `department`.

   ```
   SQL> @q1setup
   ```

2. Run `q1select.sql` to examine the contents of these two tables.

   ```
   SQL> @q1select
   ```

   Note that `employee.dept_id` is a foreign key that references `department.dept_id`.

3. Run `q1emp.sql` to delete the employee record with `emp_id` = 2.

   ```
   SQL> @q1emp
   ```

4. Start a second concurrent SQL client session. For example, invoke another ssh client to login to dbtune/dbtune2 server and do the following in the second login session.

   ```
   $> cd $HOME/lab/lab3
   $> sqlplus / as sysdba
   ```

5. Run `q1dept.sql` in the second session to delete the department record with `dept_id` = 20.

   ```
   SQL> @q1dept
   ```

6. Note that the deleted employee record in the first session has `dept_id` = 10 which is not related at all to the department record that you are trying to delete in the second session. However, your deletion statement in the second session is blocked.

7. Do the following in the first session to unblock the second session.

   ```
   SQL> rollback;
   ```

   Notice that the deletion in the second session is now unblocked and executed.

8. Undo the deletion in the second session by issuing the following in the second session.

   ```
   SQL> rollback;
   ```

9. Run an appropriate DDL statement (in either one of the sessions) such that when steps 3 & 5 are repeated, both deletions will execute without any blocking.
   Write down the DDL statement you used in your report.

4 Question 2

1. Run `q2setup.sql` to create and populate the table `dept`.

```
SQL> @q2setup
```

2. Run `q2s.sql` to examine the contents of the created table.

```
SQL> @q2s
```

3. In the following, we consider the interleaved execution of two transactions.

Transaction A	Transaction B
increment the budget of dept 10 by $100;	
	delete dept with budget of $200;
increment the budget of dept 20 by $100;	
	commit;
commit;	

`Transaction A` updates the budget of each dept by $100, while `transaction B` deletes dept with a budget of $200. The transactions are interleaved such that `transaction B` is executed after dept 10 has been updated but before the update of dept 20. We simulate the above interleaved execution by steps 4 to 9.

4. Run `q2u10.sql` to update the budget of dept 10 and run `q2s.sql` to examine the contents.

```
SQL> @q2u10
SQL> @q2s
```

Record the contents of `dept` table in your report.

5. Start a second concurrent SQL client session. For example, invoke another ssh client to login to dbtune/dbtune2 server and do the following in the second login session.

```
$> cd $HOME/lab/lab3
$> sqlplus / as sysdba
```

6. Run `q2d.sql` in the second session to delete dept with a budget of $200 and run `q2s.sql` to examine the contents.

```
SQL> @q2d
SQL> @q2s
```

Record the contents of `dept` table in your report.

7. Run `q2u20.sql` in the first session to update the budget of dept 20.

```
SQL> @q2u20
```

8. Commit `transaction B` in the second session and run `q2s.sql` to examine the contents.

```
SQL> commit;
SQL> @q2s
```

Record the contents of `dept` table in your report.

9. Commit `transaction A` in the first session and run `q2s.sql` to examine the contents.

```
SQL> commit;
SQL> @q2s
```

Record the contents of `dept` table in your report. Is the output equivalent to that produced by some *serial execution* of the transactions?
If so, write down the order of the equivalent serial execution (i.e. either (A,B) or (B,A)).

10. Terminate the second session.

```
SQL> quit;
```

11. Now, consider a second interleaved execution of the two transactions, where `transaction A` updates dept 20 before dept 10.

Transaction A	Transaction B
increment the budget of dept 20 by $100;	
	delete dept with budget of $200;
increment the budget of dept 10 by $100;	
commit;	
	commit;

Following a similar procedure (given by steps 4 to 9) used to simulate the first interleaved execution, run appropriate queries/commands in two concurrent sessions to simulate the second interleaved execution.

In your report, record the contents of `dept` table at the end of the following commands:

(a) After the first update by `transaction A`,

(b) After the second update by `transaction A`,

(c) After the commit by `transaction A`, and

(d) After the commit by `transaction B`.

Is the final output equivalent to that produced by some serial execution of the transactions? If so, write down the order of the equivalent serial execution (i.e. either (A,B) or (B,A)).

5 Shutdown Oracle Instance (VERY IMPORTANT)

Shutdown your oracle instance before leaving the lab:

```
SQL> shutdown immediate
SQL> quit
```