



**DALHOUSIE  
UNIVERSITY**

**CSCI 5408**

**Data Management, Warehousing and  
Analytics**

**Assignment 2**

**Problem - 1**

**Name : Sagarkumar Pankajbhai Vaghasia**

**CSID : vaghasia**

**Banner ID : #B00878629**

GitLab Link :

[https://git.cs.dal.ca/vaghasia/csci5408\\_s22\\_sagarkumar\\_vaghasia\\_b00878629](https://git.cs.dal.ca/vaghasia/csci5408_s22_sagarkumar_vaghasia_b00878629)

## Problem #1

1) I have created all the tables from the previous EERD of Parks Nova Scotia. There were total 12 entities I created previously. So, I have created 12 tables according to the entities and inserted 4 to 5 rows of data in each tables. The screenshot for the list of tables is given below and the values which I inserted are submitted in dump.



2) I have created one local database and named it a2\_locsg682034 where I have put the above-mentioned 12 tables and their values. Then, I wrote a transaction block having 1 SELECT and 1 UPDATE statement.

```
1 • set autocommit = false;
2 • set profiling = 1;
3 • START TRANSACTION;
4 • select * from park;
5 • select * from accommodation where acc_id = 1;
6 • update accommodation set acc_name = "airbnb" where acc_id = 1;
7 • commit;
8 • show profiles;
```

I have recorded the execution time. The transaction having 1 SELECT AND 1 UPDATE and its output along with the recorded time is given below:

The screenshot shows the MySQL Workbench interface with a transaction named 'A2 P1-2-1' containing 8 steps. The SQL editor displays the following queries:

1. `set autocommit = false;`
2. `set profiling = 1;`
3. `START TRANSACTION;`
4. `select * from park;`
5. `select * from accommodation where acc_id = 1;`
6. `update accommodation set acc_name = "airbnb" where acc_id = 1;`
7. `commit;`
8. `show profiles;`

The Result Grid shows the execution details for these queries:

Query_ID	Duration	Query
49	0.00041025	SELECT * FROM a2_jccag682034.accommodatio...
50	0.00044150	SELECT * FROM a2_jccag682034.accommodatio...
51	0.00028875	set autocommit=0
52	0.00109075	UPDATE 'a2_jccag682034', 'accommodation' ...
53	0.00472775	commit
54	0.00024500	SELECT * FROM a2_jccag682034.accommodatio...
55	0.00031500	set autocommit=1
56	0.00027550	set autocommit = false
57	0.00018200	set profiling = 1
58	0.00012075	SHOW WARNINGS
59	0.00019900	START TRANSACTION
60	0.00040075	select * from park LIMIT 0, 1000
61	0.00057575	select * from accommodation where acc_id = 1 ...

The screenshot shows the MySQL Workbench interface with a transaction named 'A2 P1-2-1' containing 8 steps. The SQL editor displays the following queries:

1. `set autocommit = false;`
2. `set profiling = 1;`
3. `START TRANSACTION;`
4. `select * from park;`
5. `select * from accommodation where acc_id = 1;`
6. `update accommodation set acc_name = "airbnb" where acc_id = 1;`
7. `commit;`
8. `show profiles;`

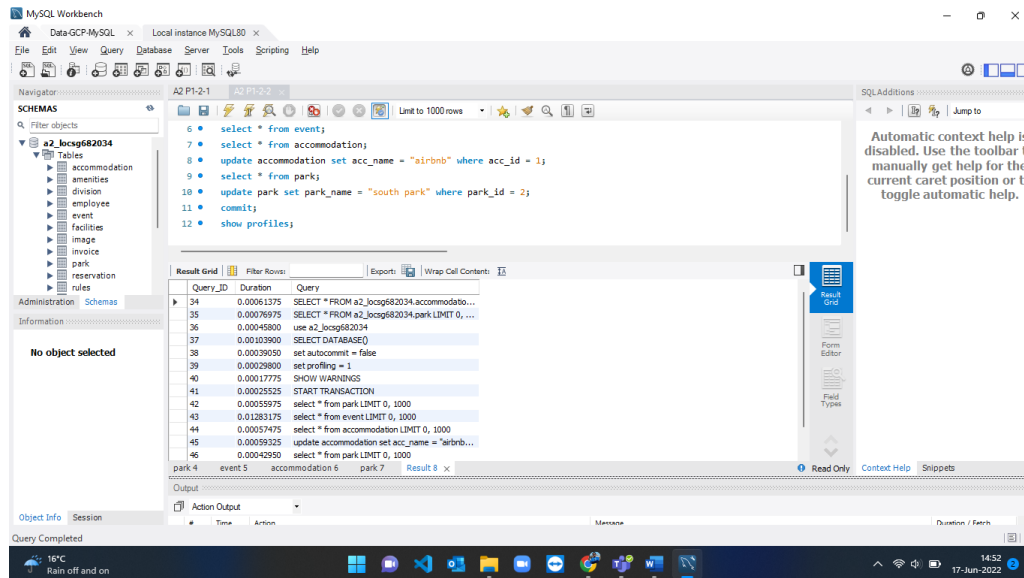
The Result Grid shows the execution details for these queries:

Query_ID	Duration	Query
52	0.00109075	UPDATE 'a2_jccag682034', 'accommodation' ...
53	0.00472775	commit
54	0.00024500	SELECT * FROM a2_jccag682034.accommodatio...
55	0.00031500	set autocommit=1
56	0.00027550	set autocommit = false
57	0.00018200	set profiling = 1
58	0.00012075	SHOW WARNINGS
59	0.00019900	START TRANSACTION
60	0.00040075	select * from park LIMIT 0, 1000
61	0.00057575	select * from accommodation where acc_id = 1 ...
62	0.00086125	update accommodation set acc_name = "airbnb...
63	0.00240650	commit

After That, I added one more SELECT and UPDATE statement in the transaction.

```
1 • set autocommit = false;
2 • set profiling = 1;
3 • START TRANSACTION;
4 • select * from park;
5 • select * from event;
6 • select * from accommodation;
7 • update accommodation set acc_name = "airbnb" where acc_id = 1;
8 • select * from park;
9 • update park set park_name = "south park" where park_id = 2;
10 • commit;
11 • show profiles;
```

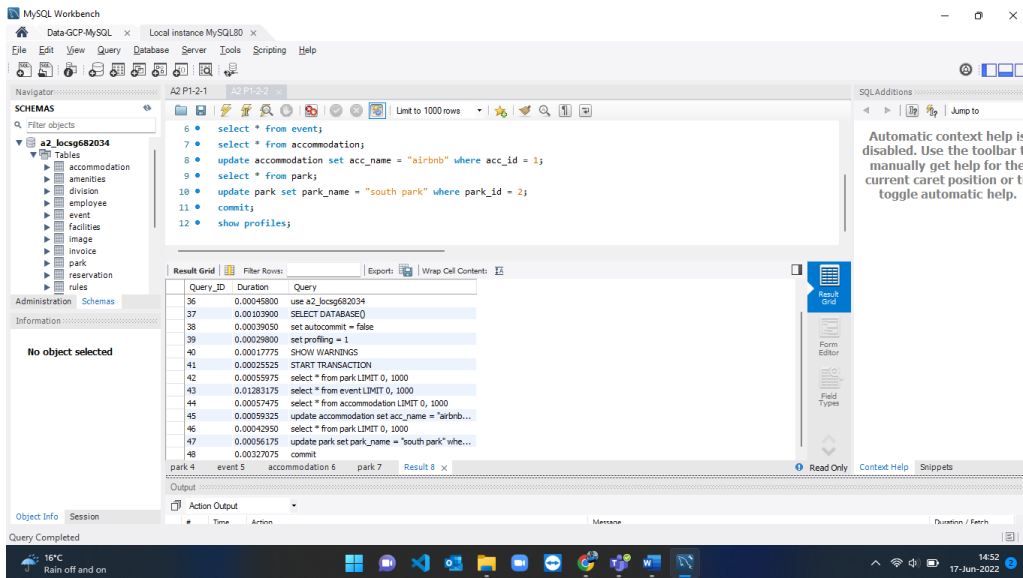
I have recorded the execution time. The transaction having 2 SELECT AND 2 UPDATE and its output along with the recorded time is given below:



The screenshot shows the MySQL Workbench interface with a transaction execution. The 'Query ID' column shows durations for each query. The 'Query' column shows the SQL statements. The 'Result Grid' shows the output of the queries.

Query ID	Duration	Query
34	0.00061375	SELECT * FROM a2_jocsp682034.accommodatio...
35	0.00078975	SELECT * FROM a2_jocsp682034.park LIMIT 0, ...
36	0.00046800	use a2_jocsp682034
37	0.00103900	SELECT DATABASE()
38	0.00039050	set autocommit = false
39	0.00029800	set profiling = 1
40	0.00017775	SHOW WARNINGS
41	0.00025525	START TRANSACTION
42	0.00055975	select * from park LIMIT 0, 1000
43	0.01283175	select * from event LIMIT 0, 1000
44	0.00057475	select * from accommodation LIMIT 0, 1000
45	0.00059325	update accommodation set acc_name = "airbnb..."
46	0.00042950	select * from park LIMIT 0, 1000

The 'Result Grid' shows the output of the queries. The first query returns 6 rows. The second query returns 4 rows. The third query returns 1 row. The fourth query returns 1 row. The fifth query returns 1 row. The sixth query returns 1 row. The seventh query returns 1 row. The eighth query returns 1 row. The ninth query returns 1 row. The tenth query returns 1 row. The eleventh query returns 1 row. The twelfth query returns 1 row.



Sr No.	Transaction	Time (seconds)
1	1 SELECT 1 UPDATE	0.012541
2	2 SELECT 2 UPDATE	0.022824

3) I have created one remote database by using GCP and named it a2\_remsg682034 where I have put the 12 tables and their values. Then, I wrote the same transaction block as in 2) having 1 SELECT and 1 UPDATE statement and I recorded the execution time. The remote transaction and its output along with the recorded time is given below :

The image displays two screenshots of the MySQL Workbench interface, showing a transaction block of SQL queries and their execution results.

**Top Screenshot:**

- Query 1:**

```

1 use a2_remsg682034;
2 set autocommit = false;
3 set profiling = 1;
4 START TRANSACTION;
5 select * from accommodation;
6 update accommodation set acc_name = "airbnb" where acc_id = 1;
7 commit;
8 show profiles;

```
- Result Grid:**

Query_ID	Duration	Query
9	0.00012100	select * from accommoda...
10	0.00018775	set autocommit = false
11	0.00015350	set profiling = 1
12	0.00014300	SHOW WARNINGS
13	0.00013525	START TRANSACTION
14	0.00012450	select * from accommoda...
15	0.00015700	use a2_remsg682034
16	0.00022075	SELECT DATABASE()
17	0.00018000	set autocommit = false
18	0.00018900	set profiling = 1
19	0.00015325	SHOW WARNINGS

**Bottom Screenshot:**

- Query 1:**

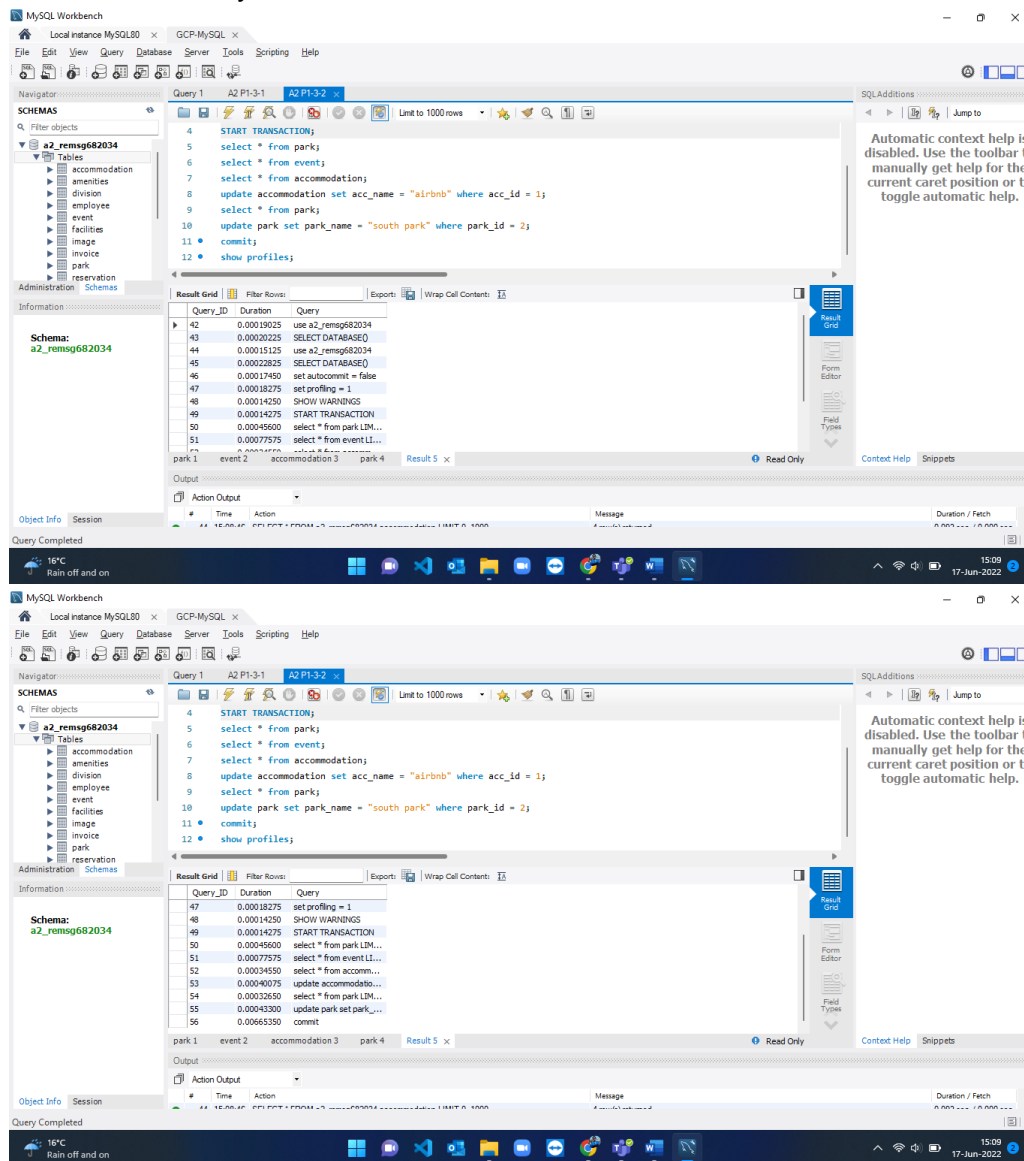
```

1 use a2_remsg682034;
2 set autocommit = false;
3 set profiling = 1;
4 START TRANSACTION;
5 select * from accommodation;
6 update accommodation set acc_name = "airbnb" where acc_id = 1;
7 commit;
8 show profiles;

```
- Result Grid:**

Query_ID	Duration	Query
13	0.00013525	START TRANSACTION
14	0.00012450	select * from accommoda...
15	0.00015700	use a2_remsg682034
16	0.00022075	SELECT DATABASE()
17	0.00018000	set autocommit = false
18	0.00018900	set profiling = 1
19	0.00015325	SHOW WARNINGS
20	0.00013700	START TRANSACTION
21	0.00037750	select * from accommoda...
22	0.00051175	update accommodation set ...
23	0.00563000	commit

Then, I added one SELECT and UPDATE statement in the transaction and recorded the time taken by it.



Sr No.	Transaction	Time (seconds)
1	1 SELECT 1 UPDATE	0.008441
2	2 SELECT 2 UPDATE	0.010806

Transaction	Time taken in local Database	Time taken in remote database (GCP)
1 (1 SELECT & 1 UPDATE)	0.012541	0.008441
2 (2 SELECT & 2 UPDATE)	0.022824	0.010806

### **Observation and Analysis :**

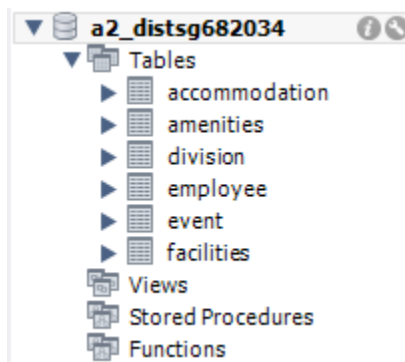
I have observed that in the local database the transaction having 1 SELECT and 1 UPDATE took less time compared to the transaction having 2 SELECT and 2 UPDATE.

For the remote database (GCP), I have executed the same transactions as of the local database. Similar trend is observed where the transaction having 1 SELECT and 1 UPDATE took less time than the transaction having 2 SELECT and 2 UPDATE.

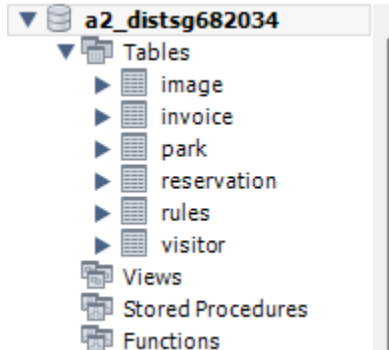


4) I have created 2 databases having the same name in local and remote(GCP) : a2\_distsg682034. I have a total of 12 tables. So, I put 6 tables from that in the local database and 6 in the remote database. Then, I wrote java code to connect both the databases and I wrote a transaction block having 1 SELECT and 1 UPDATE executed on a local database and 1 SELECT and 1 UPDATE executed on a remote database.

Tables in a2\_distsg682034 (local database) :



Tables in a2\_distsg682034 (remote database) :



I have used HashMap to create a Global Data Catalog. Here, I have passed two arguments in the HashMap one is the table name and second is the connection. By using this way we can identify which table is located in which partition.

```

2 usages
public class GDC
{
    14 usages
    public HashMap<String, Connection> dataDictionary;

    1 usage
    public GDC(Connection local, Connection remote)
    {
        this.dataDictionary = new HashMap<String, Connection>();

        this.dataDictionary.put("accommodation", local);
        this.dataDictionary.put("amenities", local);
        this.dataDictionary.put("division", local);
        this.dataDictionary.put("employee", local);
        this.dataDictionary.put("event", local);
        this.dataDictionary.put("facilities", local);

        this.dataDictionary.put("image", remote);
        this.dataDictionary.put("invoice", remote);
        this.dataDictionary.put("park", remote);
        this.dataDictionary.put("reservation", remote);
        this.dataDictionary.put("rules", remote);
        this.dataDictionary.put("visitor", remote);
    }
}

```

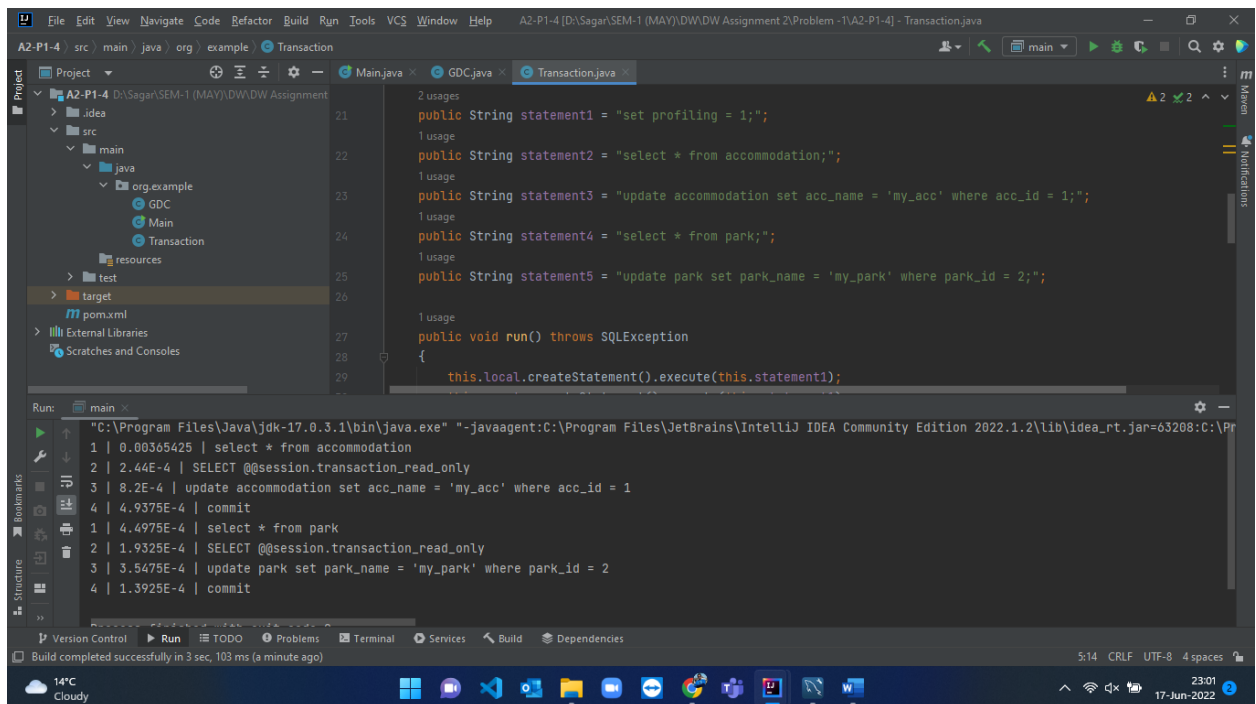
I have written a transaction block having 1 SELECT and 1 UPDATE statement for the table(accommodation) in the local database and 1 SELECT and 1 UPDATE statement for the table(park) in the remote database.

```

public String statement1 = "set profiling = 1;";
1 usage
public String statement2 = "select * from accommodation;";
1 usage
public String statement3 = "update accommodation set acc_name = 'my_acc' where acc_id = 1;";
1 usage
public String statement4 = "select * from park;";
1 usage
public String statement5 = "update park set park_name = 'my_park' where park_id = 2;";

```

After execution of the whole transaction block which performs operations on both local and remote database, I measured the time taken by both and the output screenshot for the time taken is attached below:



The screenshot displays the IntelliJ IDEA IDE with a project named 'A2-P1-4'. The 'Transaction.java' file is open, showing a class with five SQL statements and a 'run()' method. The Run console at the bottom shows the execution output, including timestamps and SQL queries.

```
public class Transaction {  
    21 public String statement1 = "set profiling = 1;";  
    22 public String statement2 = "select * from accommodation;";  
    23 public String statement3 = "update accommodation set acc_name = 'my_acc' where acc_id = 1;";  
    24 public String statement4 = "select * from park;";  
    25 public String statement5 = "update park set park_name = 'my_park' where park_id = 2;";  
    26  
    27 public void run() throws SQLException  
    28 {  
    29     this.local.createStatement().execute(this.statement1);  
}
```

Run: main

```
"C:\Program Files\Java\jdk-17.0.3.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.1.2\lib\idea_rt.jar=63208:C:\Pr  
1 | 0.00365425 | select * from accommodation  
2 | 2.44E-4 | SELECT @@session.transaction_read_only  
3 | 8.2E-4 | update accommodation set acc_name = 'my_acc' where acc_id = 1  
4 | 4.9375E-4 | commit  
1 | 4.4975E-4 | select * from park  
2 | 1.9325E-4 | SELECT @@session.transaction_read_only  
3 | 3.5475E-4 | update park set park_name = 'my_park' where park_id = 2  
4 | 1.3925E-4 | commit
```

Build completed successfully in 3 sec, 103 ms (a minute ago)