Faculty of Science, Engineering and Technology

Fundamentals of Data Management

Credit Tasks 9.2: Transactions and Concurrency

Overview

You'll learn how to implement transactions in practice and how to investigate the concurrency behaviour of your database.

Purpose

Gain practical experience of the effects of concurrency in different isolation levels using MySQL. You are free to use any other relational database management system.

Task

Download the Ubuntu virtual machine from Canvas and open it in the VMWare Player. Follow the instructions to open two connections to the MySQL server from the MySQL Workbench. Work through the tasks below.

Time

This task should be completed in your lab class and submitted for feedback in lab 9 or at the beginning of lab 10.

Resources

- Online module (from Canvas)
- Book Chapters, e.g.
  - Database Systems, Connolly & Begg (http://goo.gl/cQ9vJr), chapter 22
  - Fundamentals of Database Systems, Elmasri & Navathe, chapters 21, 22
- MySQL (on FDM virtual machine) and MySQL Workbench (or other RDBMS and suitable client).

Feedback

Discuss your solutions with the tutorial instructor.

Next

Get started on module 10.

Credit Tasks 9.2 — Submission Details and Assessment Criteria

Document your solutions to the tasks using a Word processor. Upload the Pass level work to Doubtfire. The tutors will discuss them with you in the lab.

Before attempting the task, check that autocommit is set to false.

```
show variables like ‘autocommit’;
```

If it is not, set it to false.

Transactions for this task:

Transaction T1	Transaction T2
`SELECT * FROM Products WHERE ProductNumber=1; SELECT * FROM Orders WHERE OrderNumber=945; SELECT * FROM Order_Details WHERE OrderNumber=945;`	`UPDATE Products SET QuantityOnHand= QuantityOnHand-2 WHERE ProductNumber=1;` `INSERT INTO Orders (OrderNumber, OrderDate, ShipDate, CustomerID, EmployeeID) VALUES (945, '2015-09-04', '2015-09-05', 1004, 701);` `INSERT INTO Order_Details (OrderNumber, ProductNumber, QuotedPrice, QuantityOrdered) VALUES (945, 1, 1200.00, 2);`

Subtask 9.2.2

In both Workbenches, run the statement:

set session transaction isolation level read committed;

Note: Ensure you end your current transaction by issuing a commit or rollback before you start a new exercise. Remember a workbench starts a transaction whenever you issue a DML statement. It is important to note when the first statement was issued – it decides what results of another transactions this transaction can see.

Scenario
Run the **first** statement of T2 again in your right MySQL Workbench instance. Run **all** statements of T1 in your left Workbench instance. What do you see? Run the **rest of T2** in the right MySQL Workbench. Check again what you can see in your left Workbench. Commit T2 in your right Workbench. Re-run T1 in your left instance. What do you see? Commit T1 in your left Workbench. Re-run T1 again. What do you see?

How did the query results differ from the ones in subtask 2.1? How can this difference lead to a lost update? Explain the difference in your report and list the necessary SQL statements to produce a lost update at read committed isolation level.