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 in both workbenches.

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 9.2.1

Scenario
Run the **first** statement of T2 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. Copy **all** statements of **T1** into your right MySQL Workbench and run them. What do you see? 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?

Note: Increment the order number (initially 945) every time you restart to create distinct entries. Remember to update the number in both workbenches.

Write your conclusions into your report, ensuring that you answer the following questions:

What isolation level are you working at? (You can check using:
`show variables like 'tx_isolation';)`
When does a transaction see the changes made? Answer for both the transaction that makes the changes and a transaction that merely reads the changes.

Why can't T1 see the changes of T2 when T2 commits?

What do we mean by 'repeatable read' and do we have phantoms here in MySQL?

What does the SQL standard say about phantoms and Repeatable Read Isolation level?

This page can help:

https://dev.mysql.com/doc/refman/5.0/en/set-transaction.html