Faculty of Science, Engineering and Technology

Fundamentals of Data Management

Pass Tasks 8.1: SQL - DML - Updates

Overview

In this tutorial, you will practise using SQL DML statements to add and manipulate data in a relational database.

Purpose

Learn to write SQL DML statements to add, remove and change data in a database.

Task

Solve the tasks given below.

Time

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

Resources

- Online module (from Canvas)
- Online resources, e.g.
    - Tutorialspoint:
      http://www.tutorialspoint.com/mysql/mysql-insert-query.htm
      http://www.tutorialspoint.com/mysql/mysql-update-query.htm
      http://www.tutorialspoint.com/mysql/mysql-delete-query.htm
    - MySQL reference:
      http://dev.mysql.com/doc/refman/5.7/en/insert.html
      http://dev.mysql.com/doc/refman/5.7/en/update.html
      http://dev.mysql.com/doc/refman/5.7/en/delete.html

Feedback

Discuss your solutions with the tutorial instructor.

Next

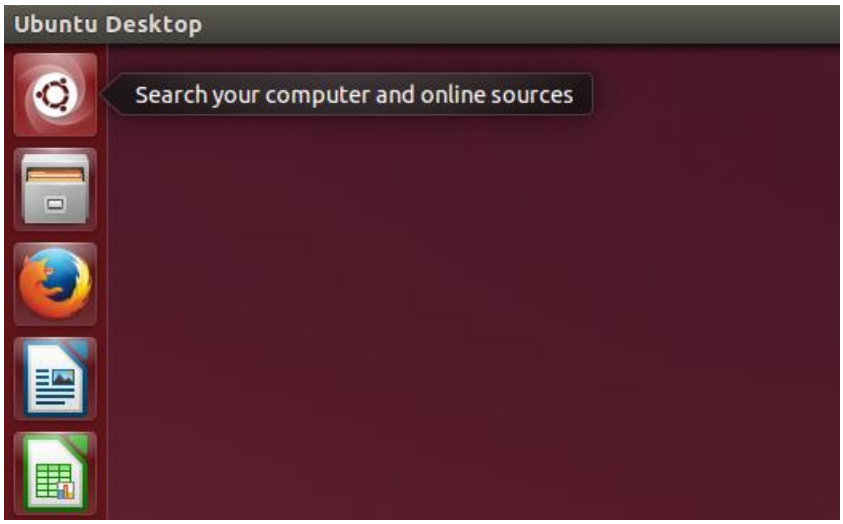
Get started on module 9.

Pass Tasks 8.1 — Submission Details and Assessment Criteria

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

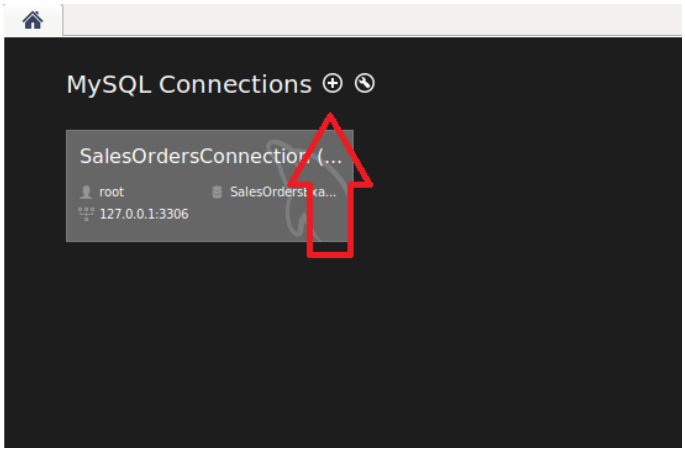
Getting Started

Open the VMWare Player and start the virtual machine. The password for fdm is admin. Click on the application icon (the uppermost icon on the left).

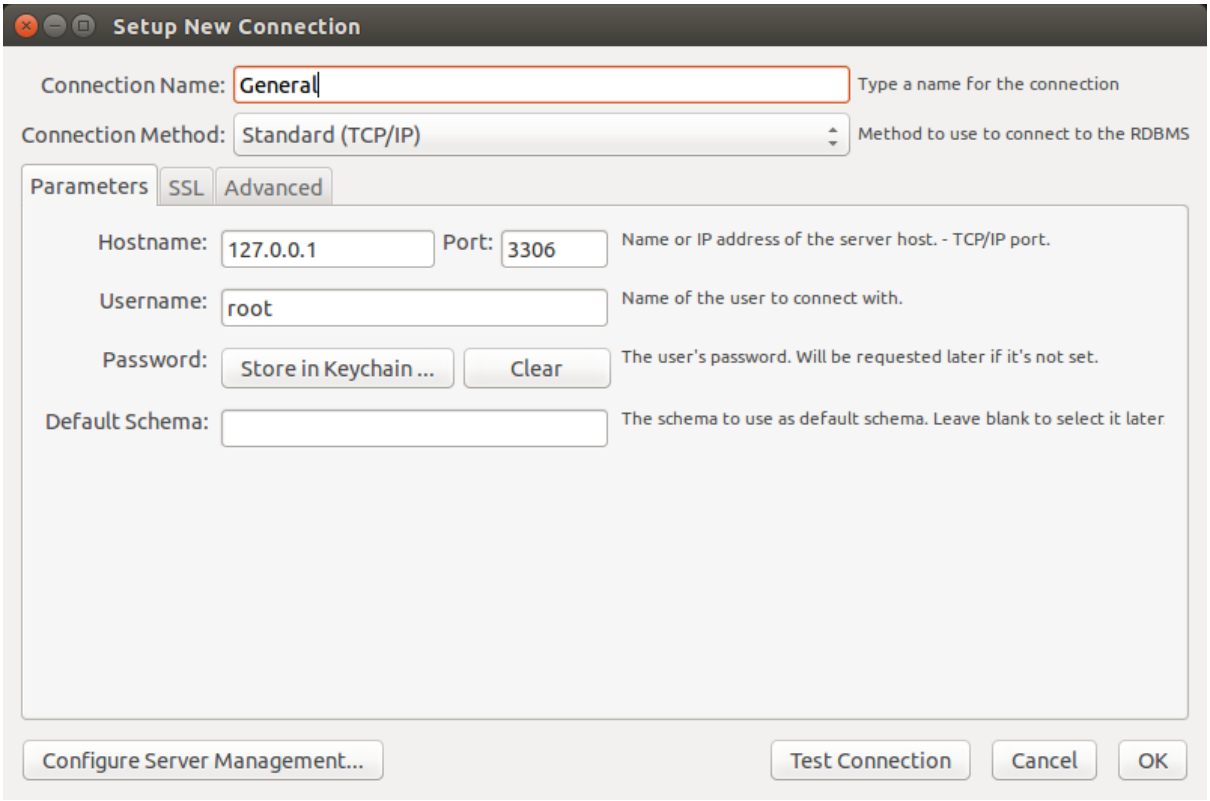


If you can't find MySQL Workbench, type it into the search field:

Open MySQL Workbench. If you need a new connection, click on the plus sign next to MySQL Connections.



Make a new connection (you can name it any way you like) by leaving all the defaults. Don't specify a default schema:



Open the new connection. If you haven't made your own database yet, type

```
CREATE SCHEMA test;
```

OR

```
CREATE DATABASE test;
```
(or any other name for your database)

Type

```
USE test
```

This sets 'test' as your working schema (environment).

First, create two tables to work with:

```
CREATE TABLE Purchase(
purchaseID int unsigned not null auto_increment,
custName VARCHAR (30) not null,
orderedDate DATE not null,
shipDate DATE,
PRIMARY KEY (purchaseID));

CREATE TABLE PurchasedItem(
purchaseID int unsigned not null,
itemNo int unsigned not null,
productName VARCHAR(30) not null,
orderedQty TINYINT unsigned not null,
quotedPrice DECIMAL(5, 2) not null,
PRIMARY KEY (purchaseID, itemNo),
FOREIGN KEY (purchaseID) REFERENCES Purchase(purchaseID));
```

Second, turn off autocommit:

```
SET AUTOCOMMIT = false;
```

Subtask 8.1.1

Run the following statement:

```
INSERT INTO PurchasedItem (purchaseID, itemNo, productName, orderedQty,
quotedPrice)
VALUES (111, 1, 'Lawn mower', 3, 105.30);
```

What happens? What do you have to do to make this INSERT possible? At the end, commit your changes.

Document and submit your solution.