Practical MySQL

Some Useful Usages of MySQL

What is MySQL

- Relational Database
- Variant of SQL (Structured Query Language)
- Free-ish
- Open Source
- Semi-Transactional (and Locking) (InnoDB only)

MySQL At DalFCS

- Host: db.cs.dal.ca
- User: CSID, username on Bluenose
- Password (default): Banner1D
- Port: 3306
Establish Connection – Command Line

- Bluenose$ mysql -u myusername -p -h db.cs.dal.ca [-D mydbname]
  - -u (Login to the database server): Same as logged into Bluenose with.
  - -p = Ask for a password, default is Banner ID
  - -h (database hostname): Here it is db.cs.dal.ca
  - -D (database name) (optional)

Connecting – MySQL Workbench

- Choose a connection name, set username, set hostname, leave port as 3306 (usually)
- Option to store password, and choose a default schema (usually same as username)
What if I am Off-Campus?

- SSH Forwarding can solve this.
- Windows: PuTTY
  - Session: Type Hostname in host field
  - Connection / SSH / Tunnels
  - Add new forwarded port:
    - Source: 3306
    - Destination: db.cs.dal.ca:3306
- www.cs.ualberta.ca/technical/services/ssh/putty/puttyfw.html

Change Your Password

- SET PASSWORD FOR `username`@`129.173.0.0/255.255.0.0`
  - PASSWORD ("mynewpassword");
Creating Your DB

- CREATE DATABASE name;
- CREATE TABLE (colname1 type, colname2 type);
- INSERT INTO table (colname1, colname2) VALUES (val1, val2);
- DROP TABLE table;
- GRANT select ON db.table to 'user'@'host';

Local (FCS) DB Creation

- $MYSQL -e "CREATE USER '$USER'@'129.173.0.0/255.255.0.0' IDENTIFIED BY 'PASSWORD';"
- $MYSQL -e "CREATE DATABASE $USER;"
- $MYSQL -e "GRANT ALTER, ALTER ROUTINE, CREATE, CREATE ROUTINE, CREATE TEMPORARY TABLES, CREATE VIEW, DELETE, DROP, EXECUTE, INDEX, \"
  INSERT, LOCK TABLES, SELECT, SHOW VIEW, UPDATE ON $USER.* TO \"
  '$USER'@'129.173.0.0/255.255.0.0';"
Local (FCS) Restrictions

- Can not create a database (can request more for group and special project work)
- Can not grant permissions to others (can be fixed)
- Can only connect from *.dal.ca

Why I Hate Using the SQL Command Line

- SELECT * FROM students s INNER JOIN attends r ON s.id = r.stu_id INNER JOIN courses c ON r.crs_id = c.id WHERE term = '201320';
- UPDATE students SET name = 'Jane Jones' WHERE name = 'Jane Smith';
- DELETE FROM students WHERE id = 'B00872729';

Connecting from Perl

#!/usr/bin/perl
use strict;
use warnings;
$school = "example.edu";
$dbname = "students";
$term = "201320";
$schema = "students";
$host = "localhost";
$dbname = "example";
$uname = "username";
$password = "password";

# Connect to database
$db = DBI->connect("dbi:mysql:$dbname:$host", $uname, $password, {RaiseError => 1})
  or die "Could not connect to database: $!";

# Use the database
$db->select_db($schema) or die "Could not select database: $!";

# Insert student into table
$sth = $db->prepare("INSERT INTO students (id, name) VALUES ('1', 'John Smith')");
$sth->execute or die "Could not execute query: $!";

# Update student name
$sth = $db->prepare("UPDATE students SET name = 'Jane Jones' WHERE id = '1'");
$sth->execute or die "Could not execute query: $!";

# Delete student
$sth = $db->prepare("DELETE FROM students WHERE id = '1'");
$sth->execute or die "Could not execute query: $!";

$db->disconnect;
$db = undef;

# End script
Connecting From PHP

```php
<?php
$connection = new mysqli('localhost', 'username', 'password', 'database');
if ($connection->connect_error) {
    die('Connection failed: ' . $connection->connect_error);
}
// Your PHP code here...
$connection->close();
```

MySQL Documentation

- https://www.cs.dal.ca/services/support/faq/connecting-mysql

Some Other DB Mgmt Systems

- MS Access (User I/F, Drop and Do Drop Query, Small)
- SQLite (A File)
- PostgreSQL (Enterprise: Transactions, etc.)
- Oracle (Expensive, Robust)
- MS SQLServer (Bigger than Access)