MySQL – Continent Exercise

I INF 202 : Introduction to Data & Databases

Log on Shared Server

* Use either instructions for Mac or Windows to logon to Shared Server.

* Once you have successfully logged on to the server, type

1. pwd (verify that your current working directory and you are on the Ubuntu Linux server)
2. mysql –u<username> -p

Check next slide for logon information

Start MySQL

- Type “mysql -u user[#] -p”. Replace “user[#]” as below:
  - Group 1: user1
  - Group 2: user2
  - Group 3 : user3
  - Group 4: user4

- When queried for your password, type “user[#]pass”. Again, replace “user[#]” with above.
  - Password for “user1” is “user1pass” (do not include quotes)

- Should see prompt that looks like “mysql>”

MySQL Command Line

```
ubuntu@ip-10-182-189-221:~/mysql$ mysql -u user1 -p
Enter password:
```

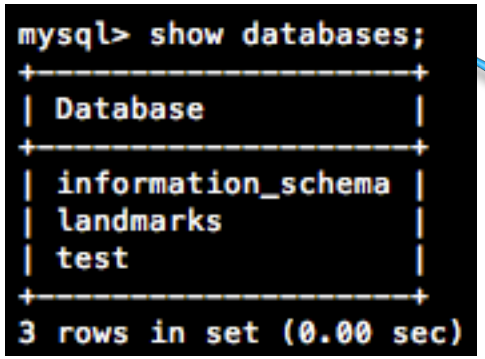
Enter password for “user1”

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 142
Server version: 5.5.35-0ubuntu0.12.04.2 (Ubuntu)

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

After successful logon, you should see “mysql” prompt

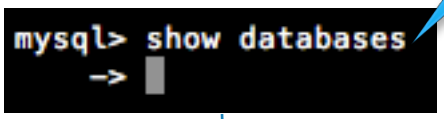
Find Databases

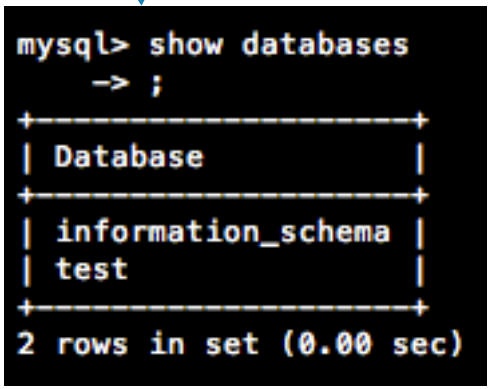
* Type "**Show databases;**" (do not type the quotes)
* All mysql commands end with a **semi-colon** and then press enter
* If you miss the semi-colon and try to type the "enter" key, the command line will expect that your commands are incomplete



With semi-colon

Without semi-colon

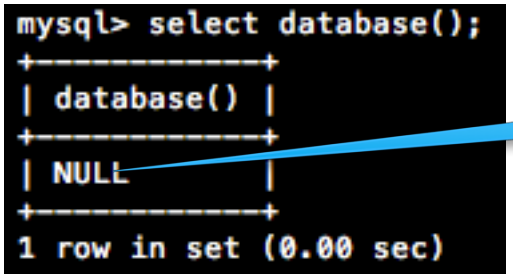




Let's start exploring

Type "select database();"

mysql> select database();



No database is in use or selected

We need to select a database to work on. Type "Use <databasename>". Replace <databasename> with landmarks, because this is the database we want to work with.

Select the “Landmarks” database

Type “use landmarks;” to select the database

```
mysql> use landmarks;
Database changed
mysql>
```

Type “select database();”

```
mysql> select database();
+------------+
| database() |
+------------+
| landmarks  |
+------------+
1 row in set (0.00 sec)
```

Now all subsequent commands would be with respect to the “landmarks” database

Find tables in database

* Type “show tables;” to find all tables in the landmarks database

  mysql> show tables;

* Type “describe continents” to show details about the table

  mysql> describe continents;

* Type “select * from continents”

  mysql> select * from continents;

* Note that Asia = 1, Europe= 2, Ocenia =3 South America =4, North America =5, Africa = 6 .

Review -- Database Functions

- All database have these functions:
  - Create: Insert data.
  - Read: Query data.
  - Update: Modify data.
  - Delete: Remove data.
- Creates great hacker acronym:

CRUD

Create Table

- Task: Create your own personal table using template below. DO NOT TYPE TEMPLATE (see next slide).

  - CREATE TABLE tablename (
    - Fieldname1 INT NOT NULL AUTO_INCREMENT,
    - Fieldname2 [INT OR VARCHAR(200)] [NOT NULL],
    - Fieldname3 [INT OR VARCHAR(200)] [NOT NULL],
    - Fieldname4 [INT OR VARCHAR(200)] [NOT NULL],
    - PRIMARY KEY (Fieldname),
    - FOREIGN KEY (Fieldname) REFERENCES tablename(Fieldname)
  -);

Column Definition -101

cityId INT UNSIGNED NOT NULL AUTO_INCREMENT, PRIMARY KEY (member_id)

- cityId→ name of the column
- INT → signifies that the column holds integers (numeric values with no fractional part)
- UNSIGNED → prohibits negative values
- NOT NULL → requires that the column value must be filled in. This prevents members from being created with no ID number
- AUTO_INCREMENT → special attribute in MySQL, It indicates that the column holds sequence numbers. If you provide no value when you create a row, then MySQL automatically generates the next sequence number and assigns it to the column.
- PRIMARY KEY (member_id) → Indicates that member_id column is the primary key that ensures that each value in the column is unique.

Tip:

- Any PRIMARY KEY column must also be NOT NULL. So, if we missed it, MySQL database will enforce it automatically.
- PRIMARY KEY clause indicates that the column is indexed for fast lookups while retrieving data from the database.

Create Table

Wherever you see [lastname] insert your last name. DO NOT TYPE "[lastname]".

- **CREATE TABLE student[lastname] (**
  - **[lastname]ID INT NOT NULL ...TO_INCREMENT,**
  - **[lastname]URL VARCHAR...**
  - **[lastname]Name VARCHA...**
  - **cityID INT UNSIGNED NO...**
  - **PRIMARY KEY ([lastname...**
  - **FOREIGN KEY (cityID) REF...**

);

This is the name of the table you are creating. Make sure you use the prefix "student" so when these tables are deleted, we don't lose core database tables.

Create Table

Wherever you see [lastname] insert your last name. DO NOT TYPE "[lastname]".

- **CREATE TABLE student[lastname] (**
  - **[lastname]ID INT NOT NULL AUTO_INCREMENT,**
  - **[lastname]URL VARCHAR(2[illegible]) NOT NULL,**
  - **[lastname]Name VARCH[illegible]**
  - **cityID INT UNSIGNED N[illegible]**
  - **PRIMARY KEY ([lastnam[illegible]**
  - **FOREIGN KEY (cityID) RE[illegible]**

);

This line creates a unique identifier for each entry of the table. It can't be null and each entry auto-increments the value. We'll get back to this field.

Create Table

Wherever you see [lastname] insert your last name. DO NOT TYPE "[lastname]".

- CREATE TABLE student[lastname] (
  - [lastname]ID INT NOT NULL AUTO_INCREMENT,
  - [lastname]URL VARCHAR(200) NOT NULL,
  - [lastname]N me VARCHAR(200) NOT NULL
  - cityID INT UNSIGNED
  - PRIMARY KEY ([lastname
  - FOREIGN KEY (cityID) REF

);

This line requires a URL for any table entry. It cannot be null.

Create Table

Wherever you see [lastname] insert your last name. DO NOT TYPE "[lastname]".

- CREATE TABLE student[last
  - [lastname]ID INT NO
  - [lastnam
  - [lastname]Name VARCHAR(200) NOT NULL,
  - cityID INT UNSIGNED NOT NULL,
  - PRIMARY KEY ([lastname]ID),
  - FOREIGN KEY (cityID) REFERENCES city(cityId)

);

This line requires a name for any table entry. It cannot be null.

Create Table

Wherever you see [lastname] [illegible] NOT TYPE "[lastname]".

- CREATE TABLE student[last[illegible]
  - [lastname]ID INT NOT N[illegible]
  - [lastname]URL VARCHA[illegible]
  - [lastname]Name VARCHAR[illegible]) NOT NULL,
  - cityID INT UNSIGNED NOT [illegible]ULL,
  - PRIMARY KEY ([lastname]ID),
  - FOREIGN KEY (cityID) REFERENCES city(cityId)

);

This line provides a place for the primary key of the city table. It provides a means to establish a relationship between your table and the city table.

Create Table

Wherever you see [lastname] … NOT TYPE "[lastname]".

- CREATE TABLE student[last…
  - [lastname]ID INT NOT N…
  - [lastname]URL VARCHA…) NOT NULL,
  - [lastname]Name VA…CHAR(200) NOT NULL,
  - cityID INT UN…GNED NOT NULL,
  - PRIMARY KEY ([lastname]ID),
  - FOREIGN KEY (cityID) REFERENCES city(cityId)

);

This line sets the unique identifier that we created four slides ago as the primary key of the table.

Create Table

Wherever you see [lastname] [illegible] NOT TYPE "[lastname]".

- CREATE TABLE student[last[illegible]
  - [lastname]ID INT NOT N[illegible]
  - [lastname]URL VARCHA[illegible]) NOT NULL,
  - [lastname]Name VA[illegible]HAR(200) NOT NULL,
  - cityID INT UNS[illegible]NED NOT NULL,
  - PRIMARY KEY ([lastname]ID),
  - FOREIGN KEY (cityID) REFERENCES city(cityId)

);

This line identifies the cityID field we created two slides ago as a foreign key. It creates the relationship between your table and the city table.

Create Entry in Country Table

- The remainder of this activity is based on the country-city work you did for the Command Line exercise in section 1.7.
- SELECT * FROM country;
- Check to see if your country is on the list.
- If so, you are done here. Move on to Create Entry in City Table (slide 21).

Create Entry in Country Table

- If your country is not on the list, create the proper entry.
- INSERT INTO country (countryName, continentID) VALUES ('[the name of your country]', [the number of your team's continent]);
  - Asia = 1
  - Europe = 2
  - Ocenia =3
  - South America = 4
  - North America =5
  - Africa =6

This value ties your entry to the continent table -- creating a relationship with an entry (your team's continent) of that table.

Check Your Work

- **Find your country's Name:**
  - **SELECT * FROM country;**

- **You'll need this name later in this activity to establish the relationship between your city and your country.**

Create Entry in City Table

- Check for your city:
  - SELECT * FROM city;
- INSERT INTO city (cityName, countryName) VALUES (‘[the name of your city]’, [the countryName of your country]);
- Check your work:
  - SELECT * FROM city;
- Note down your city’s ID number.

Create Entry in Person Table

- Check for your name:
  - SELECT * FROM person;
- INSERT INTO person (firstName, lastName, cityID) VALUES (‘[your first name]’, ‘[your last name]’, [your cityID]);
- Check your work:
  - SELECT * FROM person;

Create Entry in Landmark Table

- See what's what:
  - SELECT * FROM landmark;
- INSERT INTO landmark (landmarkURL, landmarkName, cityID) VALUES ('[a URL for a landmark in your city]', '[the name of the landmark]', [the cityID of the city in which the landmark is located]);
- Check your work:
  - SELECT * FROM landmark;

Read (Query) Country

- SELECT * FROM continents;

```
SELECT country.countryName, continents.continentName
FROM
        country, continents
WHERE
continents.continentID=country.continentID
ORDER BY country.countryName;
```

Read (Query) City

```
SELECT cityName, country.countryName, continentName
FROM
city, country, continents
WHERE
continents.continentID=country.continentID
AND
country.countryId=city.countryId
ORDER BY
continentName, country.countryName;
```

Read (Query) Landmark

```
SELECT landmarkName, cityName, countryName,
continentName
FROM
landmark, city, country, continents
WHERE
continents.continentID=country.continentID
AND
country.countryId=city.countryId
AND
city.cityID=landmark.cityID;
```

Recommend Your Landmark

- Design a query so that you can recommend a landmark to visit in your city.
- Use a join structure to link your person entry to your landmark entry.
- Test the query.

Recommend Landmark

- Pick a classmate.
- Design a query so that you can recommend a landmark to visit in that classmate's city.
- Use a subquery structure.
- Test the query.

Submit Assignment

- Using the SQL queries you designed for the last two slides, submit assignment 2.5 MySQL in section 2.5 MySQL on Blackboard.

References

- **If you get stuck or need help, consider the following sources:**
- http://www.w3schools.com/sql/default.asp
- http://dev.mysql.com/doc/refman/5.5/en/index.html