

Contents

Using MySQL	2
Starting the database	2
Starting a MySQL interactive session	2
Creating a Database	2
Connecting to a Database	2
Entering SQL Commands	3
Display a list of tables in a database.	3
Display a list of fields in a table	3
Shutting down the database	3
Terminating MySQL interactive session	3
Executing SQL commands stored in an external file	3
Database Schemas	4
MySQL Lab 01	6
Preparation	6
Connect to your html cloud9 workspace. Download and install the necessary files for this lab	6
Connect to the sandbox database	6
Review the database schema for this lab	6
Creating the sandbox database	8
Create the tables necessary for this lab	8
Adding comments to sql files	8
Inserting (adding) data to the customer and orders tables	8
Inserting (adding) data to the orderItems tables	9
Execute each of these sample queries	10

Using MySQL

MySQL is relational database software that supports structured query language (SQL) for creating and managing, and supporting databases. It can run from the Linux command line and can be used as a back-end for web-based front-ends. It is a good choice for developing prototypes as well as production systems,

Starting the database

In a cloud9 workspace that has access to mysql, a mysql server must be running in order to create or access a database. From the Linux command line, enter:

```
mysqlctl start
```

Starting a MySQL interactive session

To enter into an interactive command-line sessions with the mysql server, where you can enter SQL commands. you use the following command from the Linux command line:

```
mysqlctl cli
```

Creating a Database

A database is a collection of tables, before tables can be created a database must be created. With MySQL you may create multiple databases. It is a good idea to create one database for development, where you care testing and making changes, and a separate database for a client to review which would be more static and consistent. The mysql online gradebook database last year was created using the command

```
CREATE DATABASE olgb;
```

Additional database parameters can be specified using the CREATE DATABASE command, but for our project no additional parameters were required.

Connecting to a Database

Because mysql can support multiple databases, before altering a database you need to connect to the database. This tells mysql which database you will be using. When you are running mysql interactively you will see a prompt such as:

```
mysql>
```

From this command prompt you can issue the `mysql> show databases;` command to see a list of the databases that are available. After you create a database it should be listed with the output of this command.

```
mysql> show databases;
```

To connect to a specific database, use the connect command with the database name. Note SQL commands are terminated with a semicolon;

```
mysql> connect olgb;
```

Entering SQL Commands

SQL commands can be entered interactively, one line at a time. But since mistakes are bound to occur it is a good idea to store sql commands in an external file, with a .sql extension, and tell mysql to execute the commands that are in the external file. When you are developing a database system it is likely that you will want to make change to the table structures once in a while, if you drop (delete) tables you will be able to quickly recreated them using external sql commands. You can generate test data and execute test queries as well using external sql files.

Display a list of tables in a database.

Once you are connect to a database you can view a list of all of the tables in the database using the command :

```
mysql> show tables;
```

Display a list of fields in a table

To see a list of fields that are in a table use the `describe` command (`desc` for short). This will show you the field name as well as the field's data type. The following example was used to display the list of fields from a database table named `School`.

```
mysql> describe School;
```

You may abbreviate the word describe:

```
mysql> desc School;
```

Shutting down the database

When you are finished using the mysql database, prior to terminating your cloud9 session, shut down the database using the following command from the Linux command line:

```
mysql-ctl stop
```

Terminating MySQL interactive session

To exit the interactive command-line sessions with the mysql server and return to the command prompt, use the `exit` command:

```
mysql> exit;
```

Executing SQL commands stored in an external file

SQL commands that may be issued more than once, or commands that span multiple lines and may need editing and refining, should be stored in an external source file with a .sql extension. Using external files will simplify development where tables may need to be dropped and recreated or data inserted and deleted. To tell mysql to execute the commands in a source file use the `source` command:

```
mysql> source source-file.sql;
```

MySQL Lab 01

Student _____

Date: _____

Database Schemas

What is a database schema? At its simplest, a database schema is a collection of definitions of the tables that make up a database, and can be seen as a collection of CREATE TABLE statements. In a relational database system it includes how the tables can – or must – relate to each other. They can identify fields within a table that must be present so they can act as a PRIMARY KEY, and they can specified fields that act as a FOREIGN KEY. FOREIGN KEYS are used to prevent data being inserted into one table (such as a class table) before data is entered into another table (such as a student table). You wouldn't want to have a database that contains class records for students that do not exist. Other items that can be defined as part of a database schema can include which fields are indexed for searching efficiency, how much physical space should be allocated to hold a table or database, and many other parameters.

MySQL Lab 01

Student _____

Date: _____

MySQL Lab 01

Preparation

Prior to this lab you should have completed the following for homework:

1. Practice starting the mysql server
2. Connecting to the mysql server
3. Creating a database name sandbox
4. Verifying the sandbox database was created using the **show databases** command.

If you have not completed these tasks, then complete them now – and start taking homework seriously.

Connect to your html cloud9 workspace. Download and install the necessary files for this lab

1. Open a terminal window
2. Issue the following wget command to download the needed files for this lab:
`wget hwmath.net/IBCS/files/sandbox.tar`
3. Issue the following tar command to extract the files:

```
tar -xvf binary_tree.tar
```

This command extracts all of the files like unzip

4. Change directories to the sanbox subdirecty and verify that you have the necessary sql files.

Connect to the sandbox database

1. Start the mysql server (if it is not started already)
2. Start a mysql interactive session
3. Connect to the sandbox database.

Review the database schema for this lab

For this lab there will be three tables that are part of an online store database. The **customer** table will be used to contain some basic customer information. The primary key for this table will be sequential number that will automatically be generated when a new customer record is inserted into the customer table. All fields in the customer table will have a cust_ prefix. The second table is named **orders**, which contains a generated order_id which implements the relation between an order placed by a customer and the list of items that make up that order. The third table is the **orderItem** table which will contain one row for every item in an order.

MySQL Lab 01

The sandbox database schema:

```
CREATE TABLE customer (
    cust_ID          SERIAL,
    cust_last_name   VARCHAR(50),
    cust_first_name  VARCHAR(40),
    cust_street      VARCHAR(50),
    cust_city        VARCHAR(50),
    cust_state       CHAR(2),
    cust_zip         CHAR(10),
    cust_email       VARCHAR(50),
    cust_phone       CHAR(15),
    PRIMARY KEY(cust_ID)
);

CREATE TABLE orders (
    order_ID         SERIAL,
    order_cust_ID    BIGINT,
    order_Date       DATETIME,
    PRIMARY KEY (order_ID)
);

CREATE TABLE orderItem (
    oi_catalogID    VARCHAR(15),
    oi_order_ID     BIGINT,
    oi_color        VARCHAR(10),
    oi_size         VARCHAR(10),
    oi_price        DECIMAL(9,2)
);
```

MySQL Lab 01

Creating the sandbox database

Create the tables necessary for this lab

In the mysql session, which you have started from the sandbox subdirectory, enter the following command to create all three tables needed for this lab:

```
mysql> source create_tables.sql;
```

Adding comments to sql files

If you browse the create_tables.sql source file, you will see at the top of the file that three lines are commented out which will drop each of these tables. All text following a # will be treated as a comment. Multiple line comments begin with /* and end with */. A sql drop command will delete the table from the database. In a development environment it could be necessary to drop and recreate tables multiple times. Prior to executing a create table statement you could also use the sql command:

```
DROP TABLE IF EXISTS `table_name`;
```

Inserting (adding) data to the customer and orders tables

Data for one customer and one orders record have been defined in the insert_customer.sql file. Add one record to each table using the following command:

```
mysql> source insert_customer.sql;
```

The two sql insert statements in this file are shown below. Notice that they both have the same format of INSERT INTO TABLENAMES (list of field names that are provided) (field values provided).

```
insert into customer (cust_last_name,  cust_first_name,
                        cust_street,    cust_city,
                        cust_state,     cust_zip,
                        cust_email,     cust_phone)
values ("Smith", "John",
       "76 Main Street", "Salem",
       "MA", "1970",
       "JSmith@samlem.ma.us", "555-555-1212"
       );
insert into orders (order_cust_ID, order_Date)
values ("1", now());
```

Other items to note about the insert statements:

- Generate fields such as the order_id are not included in these statements
- A field that is required to not be null should be identified in the create statement for the table/field
- Notice that the order_date field is provided using a function call to the now() function which will initialize the fields with the current date and time that the record is inserted into the orders tables.

MySQL Lab 01

- **ORDER** is a sql keyword, which is why the orders table was named with an "s".
- Fields that are not required will be initialized to NULL values.

Inserting (adding) data to the orderItems tables

In the sample database we will start with one customer record and one orders record. For the one order that is placed there will be two **orderItems records**, which means that one customer has placed one order that contains two line items (a sweater and a hat). Add data to the orderItems table using the following command:

```
mysql> source insert_oi.sql;
```

The insert_oi.sql file contains these two insert statements:

```
insert into orderItem (oi_catalogID, oi_order_id,
                      oi_color,      oi_size,
                      oi_price)
values ("sweater90", "1",
       "red", "M",
       "16.99");
```

```
insert into orderItem (oi_catalogID, oi_order_id,
                      oi_color,      oi_size,
                      oi_price)
values ("SkiHat421", "1",
       "red", "6 1/2",
       "11.95");
```

Notes about the insert_oi.sql file:

- Like the other insert statements there are two field names listed per line, and two value fields listed per line. This is not a requirement, but debugging time can be reduced by having the fieldnames and field values easily matched up. Don't go crazy trying to get as much information on one line as possible – not only won't you be able to easily follow your code but neither will anyone else.

MySQL Lab 01

Execute each of these sample queries

Directions:

Access GoogleClassroom to open a copy of this part of the lab. Execute each of these queries in mysql and copy the results into the corresponding text boxes. Do not errors – if you encounter any errors correct them and show only correct output.

1. Selecting (viewing) all fields for all records in a table.

```
SELECT * FROM customer;
```

Copy the output from the correctly executed command above into this textbox:

Query 1 output goes here:

2. Selecting data from multiple tables – joining the fields on a common field:

```
SELECT * FROM orders,orderItem  
WHERE orders.order_ID = orderItem.oi_order_ID;
```

Copy the output from the correctly executed command above into this textbox:

Query2 output goes here:

MySQL Lab 01

3. Displaying a calculated value from multiple rows of data, trace this command so you understand what is displayed:

```
SELECT sum(oi_price)
FROM orderItem
WHERE oi_order_id=1;
```

Copy the output from the correctly executed command above into this textbox:

Query 3 output goes here:

4. Displaying a combination of three tables: this should be in line with what you would expect from a order placed with multiple items by the same customer in the same order:

```
SELECT cust_last_name, order_cust_ID, orders.order_ID,
orders.order_Date, sum(orderItem.oi_price)
FROM customer, orders,orderItem
WHERE orders.order_ID = orderItem.oi_order_ID and
cust_ID = order_cust_ID;
```

Copy the output from the correctly executed command above into this textbox:

Query 4 output goes here: