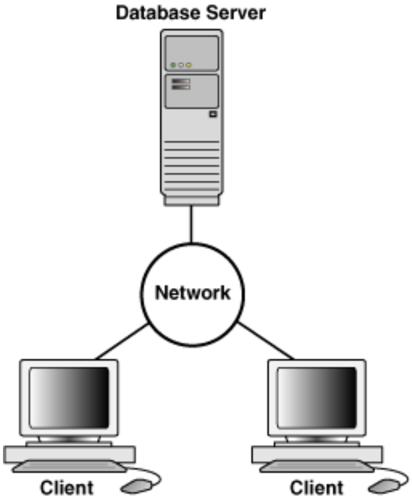
# *Introduction to MySQL*

We will be using the MySQL database throughout this module. MySQL is an open-source relational database management system (RDBMS). In this module we are going to suggest that you use the MySQL server that is provided by the Faculty of Natural & Mathematical Sciences (NMS) —which is version 5.7.15, but you can download MySQL and install it in your computer if you want.
In this session you will learn how to create and restore database backups. Note it is your responsibility to make backups of your database before and after each lab session.

## 1 Introduction to MySQL

The MySQL database system uses a client-server architecture as follows:



- The **server** is the program that actually manipulates databases.
- To tell the server what to do, use a **client** program that communicates your intent by means of statements written in Structured Query Language (SQL).

### 1.1 Getting Started with NMS Server

You should have received your database credentiasl in an email. If not, please contact the module lecturer. In that email, you should have received your userName and your password and your command line access. Your command line access should be similar to

```
mysql -u yourUserName -p -h nmsdvm999956.nms.kcl.ac.uk  yourDatabaseName
```

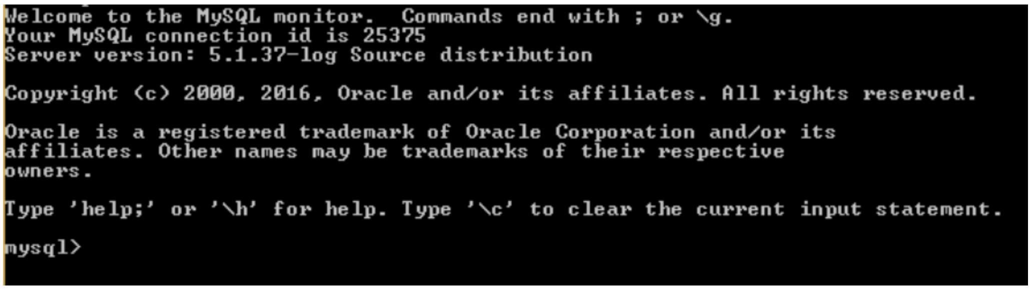
This line is a connection command that uses different parameters:

- `-u yourUserName` specifies the MySQL user name to use when connecting to the server.
- `-p` indicates that a password is needed when connecting to the server.
- `-h nmsdvm999956.nms.kcl.ac.uk` indicates the name of the host containing the MySQL server.

## 1.2 Using *mysql* Client Program

Client programs are written for diverse purposes, but each interacts with the server by connecting to it, sending SQL statements to have database operations performed, and receiving the results.
Clients are installed locally on the machine from which you want to access MySQL, but the server can be installed anywhere (in this case a machine maintained by NMS). The *mysql* program is one of the clients included in MySQL distributions. You can use the *mysql* program from Windows or Linux. For example, in Linux can use the console to run the *mysql* client program by following the next steps:

1. Launch the terminal.
2. Paste the connection command line into the command prompt and press `Enter`. You will be asked to enter your password
3. Type your password and press `Enter`.
4. You will be shown a brief introduction message and then be placed at the `mysql>` prompt.

   ```
   Welcome to the MySQL monitor.  Commands end with ; or \g.
   Your MySQL connection id is 25375
   Server version: 5.1.37-log Source distribution

   Copyright (c) 2000, 2016, 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>
   ```

   At the `mysql>` prompt, you can enter SQL commands.

5. The `quit` command terminates your mysql session. Don't type it now, as you will have to start the session again, just remember this command to close your session once you have finished.

# 2 Creating a Sample Table

1. Create a simple table by typing the following SQL statement and press `Enter`:

   ```
   mysql> CREATE TABLE limbs (thing VARCHAR(20), legs INT, arms INT);
   ```

   A **table** is a set of data elements (values) using a model of vertical columns (identifiable by name) and horizontal rows, the cell being the unit where a row and column intersect.

2. And populate it with a few rows:

   ```
   mysql> INSERT INTO limbs (thing,legs,arms) VALUES('human',2,2);
   mysql> INSERT INTO limbs (thing,legs,arms) VALUES('insect',6,0);
   mysql> INSERT INTO limbs (thing,legs,arms) VALUES('squid',0,10);
   mysql> INSERT INTO limbs (thing,legs,arms) VALUES('fish',0,0);
   mysql> INSERT INTO limbs (thing,legs,arms) VALUES('centipede',100,0);
   mysql> INSERT INTO limbs (thing,legs,arms) VALUES('table',4,0);
   mysql> INSERT INTO limbs (thing,legs,arms) VALUES('armchair',4,2);
   mysql> INSERT INTO limbs (thing,legs,arms) VALUES('phonograph',0,1);
   mysql> INSERT INTO limbs (thing,legs,arms) VALUES('tripod',3,0);
   mysql> INSERT INTO limbs (thing,legs,arms) VALUES('Peg Leg Pete',1,2);
   mysql> INSERT INTO limbs (thing,legs,arms) VALUES('space alien',NULL,NULL);
   ```

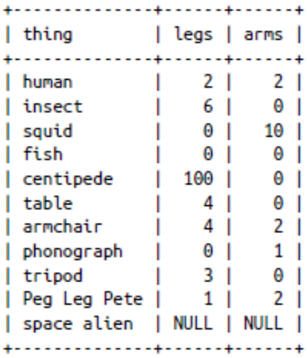
   Here's a tip for entering the `INSERT` statements more easily: after entering the first one, press the up arrow to recall it, press Backspace (or Delete) a few times to erase characters back to the last open parenthesis, then type the data values for the next statement.

   The table you just created is named limbs and contains three columns to record the number of legs and arms possessed by various life forms and objects. The physiology of the alien in the last row is such that the proper values for the arms and legs columns cannot be determined; NULL indicates "unknown value".

3. Verify that the rows were added to the limbs table by executing a SELECT statement:

```
mysql> SELECT * FROM limbs;
```

You should get the following output:



```
+--------------+------+------+
| thing        | legs | arms |
+--------------+------+------+
| human        |    2 |    2 |
| insect       |    6 |    0 |
| squid        |    0 |   10 |
| fish         |    0 |    0 |
| centipede    |  100 |    0 |
| table        |    4 |    0 |
| armchair     |    4 |    2 |
| phonograph   |    0 |    1 |
| tripod       |    3 |    0 |
| Peg Leg Pete |    1 |    2 |
| space alien  | NULL | NULL |
+--------------+------+------+
```

The SELECT statement is used to retrieve information from a database. The most basic SELECT statement has only 2 parts:

SELECT *columnName* (1)
FROM *tableName* (2)

where: (1) indicates what columns you want to return and (2) indicates what table(s) those columns come from. If we want to retrieve all of the information about all of the columns (as in the previous example) in a table, we could use the asterisk (*) as a shortcut for all of the columns.

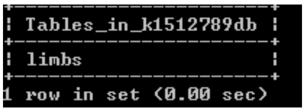
## 3 Getting Information about your Database

Let's start by taking a look on the database.

1. To know which tables are in your database type the command SHOW TABLES; and press Enter.

```
mysql> SHOW TABLES;
```

You should get the following output:



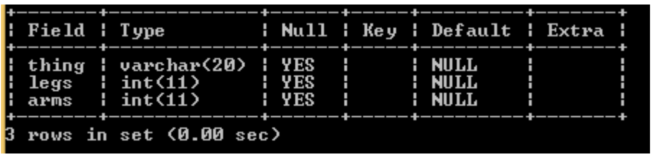
```
+---------------------+
| Tables_in_k1512789db |
+---------------------+
| limbs               |
+---------------------+
1 row in set (0.00 sec)
```

The SHOW TABLES command lists all the tables in a database. In this case, you only have the table named limbs that you have just created.

2. To see information about the type of information stored in each table (that is the different columns on the table) use the command DESCRIBE as follows:

```
mysql> DESCRIBE limbs;
```

This command gives you information about the columns in the specified table. In this case, we get the information about the columns on the weather table:



```
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| thing | varchar(20) | YES  |     | NULL    |       |
| legs  | int(11)     | YES  |     | NULL    |       |
| arms  | int(11)     | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

`Field` indicates the column name, `Type` is the data type for the column, `NULL` indicates whether the column can contain `NULL` values (which are empty or unknown values), `Key` indicates whether the column is indexed, and `Default` specifies the column's default value. `Extra` displays special information about columns.

3. Quit the mysql connection:

```
mysql> quit;
```

## 4 Database Backup

The `mysqldump` client utility performs logical backups, producing a set of SQL statements that can be executed to reproduce the original database object definitions and table data. It dumps one or more MySQL databases for backup or transfer to another SQL server. The `mysqldump` command can also generate output in CSV, other delimited text, or XML format.

1. Create a backup of your database as follows:

```
mysqldump -u yourUserName -p -h nmsdvm999956.nms.kcl.ac.uk yourDatabaseName >
    backup.sql
```

Note that `mysqldump` also needs connection parameters. This commands creates a backup that contains the contents of your database; i.e., the limbs table and its rows. In particular, the output of the mysqldump command is redirected to a file named `backup.sql`. If the notation > `fileName` is appended to any command (e.g., `mysqldump`) that normally writes its output to the terminal, the output of that command (e.g., the SQL commands for creating tables and inserting data) will be written to the file named `fileName` (e.g., `backup.sql`) instead of your terminal.

Now we are going to remove the rows of the limbs table and restore them using the backup file.

1. Connect to MySQL using your connection command line:

```
mysql -u yourUserName -p -h nmsdvm999956.nms.kcl.ac.uk yourDatabaseName
```

2. Delete all rows in the limbs table:

```
mysql> DELETE FROM limbs;
```

3. Verify that the rows were removed from the limbs table by executing a `SELECT` statement:

```
mysql> SELECT * FROM limbs;
```

You should get an empty set.

4. Quit the mysql connection:

```
mysql> quit;
```

To restore the contents of limbs we can redirect `mysql`'s input to the backup file. By default, the `mysql` program reads input interactively from the terminal, but you can feed it statements using other input sources such as a file or program.

1. To invoke mysql and redirect its input to read from the backup file execute the following command:

```
mysql -u yourUserName -p -h nmsdvm999956.nms.kcl.ac.uk yourDatabaseName < backup.
    sql
```

In this case, we are redirecting the input of the `mysql` command to the file `backup.sql`. In this example, the command `mysql` will read the contents of file `backup.sql` and use them as input. As the greater-than character > is used for output redirection, the less-than character < is used to redirect the input of a command.

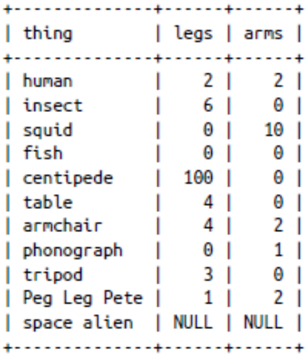
2. Connect to MySQL using your connection command line:

```
mysql -u yourUserName -p -h nmsdvm999956.nms.kcl.ac.uk yourDatabaseName
```

3. Verify that the rows have been restored by executing a SELECT statement:

```
mysql> SELECT * FROM limbs;
```

You should get the following output:



```
+--------------+------+------+
| thing        | legs | arms |
+--------------+------+------+
| human        |    2 |    2 |
| insect       |    6 |    0 |
| squid        |    0 |   10 |
| fish         |    0 |    0 |
| centipede    |  100 |    0 |
| table        |    4 |    0 |
| armchair     |    4 |    2 |
| phonograph   |    0 |    1 |
| tripod       |    3 |    0 |
| Peg Leg Pete |    1 |    2 |
| space alien  | NULL | NULL |
+--------------+------+------+
```

That is the rows that we have deleted have been restored from the backup file.

4. Quit the mysql connection:

```
mysql> quit;
```

Note that in the backup file has been given as a relative path name, and the name is interpreted relative to the directory in which the `mysql` program was started.

## 5 Creating a Directory for Backups

1. Create a directory named db:

```
mkdir db
```

2. Create a new backup of your database and save it into the db:

```
mysqldump -u yourUserName -p -h nmsdvm999956.nms.kcl.ac.uk yourDatabaseName > ./db/
    backup.sql
```

3. List the contents of the db directory to check that the backup has been created:

```
ls ./db
```

4. See the contents of the backup.sql file.

5. Restore the contents of the database using the file you have just created:

```
mysql -u yourUserName -p -h nmsdvm999956.nms.kcl.ac.uk yourDatabaseName < ./db/
    backup.sql
```

## 6 Accessing the NMS Server using Alternative Networks

If you plan on using your own laptop instead of a lab PC, note that wifi or networks will not allow direct access to the NMS MySQL server.
To connect to the MySQL Database from other networks you will first have to connect to the databases server using your kings credentials, not your database credentials. From the command prompt the SSH command to connect to the database server is (where k123456 is your username):

```
ssh yourUserName@nmsdvm999956.nms.kcl.ac.uk
```

Once on `nmsdvm999956.nms.kcl.ac.uk`, connect to the MySQL server with the usual command:

```
mysql -u yourUserName -p -h nmsdvm999956.nms.kcl.ac.uk yourDatabaseName
```

## 7 Drawing ER Models

Create a complete ERD in Crow's Foot notation that can be implemented in the relational model using the following description. Each region has a name and code. Each store has a code and an address. One region can be the location for many stores. Each store is located in only one region.
Use a diagramming program such as Microsoft Visio, OneNote, Powerpoint, Illustrator etc... for you ER diagram. You can also use web apps such as http://draw.io to help you sketch out your diagrams. Hand-drawn diagrams are usually unreadable!

## 8 Personal Study

If you finish these activities early then you can start this personal study in the lab, otherwise you should be working on these tasks in your own time.

1. Take a look into the documentation about the MySQL programs here:

   `https://dev.mysql.com/doc/refman/5.7/en/programs.html`