SQL – The basics

Before you begin please do the following:

1. Read the link that helps you install MySQL and MySQL Workbench (or SquirrelSQL). Make sure you have it working properly.
2. Watch the video lecture posted on how to use MySQL with MySQL Workbench/SquirrelSQL. Many of the commands on the videos for this course may not be done with the same IDE you use, but you should be able to type in SQL code and obtain the same results.
3. You can also look at the SQLCheatsheet in the "Resources" page for many of the common SQL commands.
4. The below exercises corresponds to the lecture titled SQL Basics (2 lectures on D2L, long and short).

Submission:
You must create a SQL file with all your answers: The SQL file should have your name in it as a comment (using #) on the first line. The following line should have your "use" command preceded by a "#" comment. The next line will contain the date and number that results from: **`SELECT @d:=NOW(),SHA(@d);`** separated by a space or TAB. This line should also be preceded by a "#" comment, followed by a blank line.
Then, for every answer, first write a "#" followed by the question number and in the following lines write your answer. Then, skip a blank line and repeat the process for the next question. For example, a file from John Doe with 2 questions, where the first question has 2 items, may look like this:

```
# John Doe
#use jdoe_db;
#2017-01-05 17:38:27  41860934238f8b3a1fdd51dee11e2d4219c92068
#1.1
Some SQL code goes
Here as the answer to
The question. SELECTs, CREATEs, whatever.
#1.2
More SQL code answering
Question number 1.2 (or I.2)
The answer can take up as many lines as needed in SQL
-- non sql line to explain a result (if asked)
-- should be commented with '--'
#2.1
Yet more SQL commands
That answer
The first part of the second question
```

If you want to add additional comments, use the -- notation. You should be able to load your entire SQL file into your IDE, uncomment the second line and run your code without **syntax** errors, at least, before submitting.
You will need to submit the SQL file. **Other formats will not be graded**. The SQL file

ends in **.sql** you can choose whatever name you like WITHOUT spaces. A good idea is your last and first name and homework number. For example: DoeJohnHW2.sql

Premise:
In order to make some extra cash to pay for college you decide to sublet your basement. You take advantage of this course by using a **database** to help you manage it.

I. Creating Tables (2pt)

Please write the correct statements to create the following tables (try it in a text file first and save it, you will need to submit all of the statements):

1. Guest – The Guest table will help you keep track of the guests who call you. You will enter their information into the Guest Table.
    i. *guestNo (integer – unique number for each guest number)*
    ii. *lastName (characters – you decide what length you think you will need, hint: Use varchar)*
    iii. *firstName (varchar – you decide the length)*
    iv. phoneNo (varchar – assume user will enter something like 999-999-9999 format)
2. Booking with the following fields:
    i. *guestNo (integer)*
    ii. *dateArrive (date)*
    iii. *dateDepart(date)*
    iv. *price (decimal representing how much you are charging)*

Help: If this step is done properly, each of the following SQL commands should return "1" You can run these commands yourself. Do not submit them in your homework.

```
SELECT count(*)=2
FROM information_schema.tables
WHERE table_name in ('Booking','Guest');

SELECT count(*)=8 AND count(distinct(column_name))=7
FROM information_schema.COLUMNS
WHERE table_name in ('Booking','Guest') and
        column_name in ('guestNo','firstName','lastName',
                        'phoneNo','guestNo','dateArrive','dateDepart','price');
```

II Inserting Rows (5pt)

Write the statements to do the below. **Be consistent.**

1. Insert at least **5** rows into the table **Guest**, use IDs 101-105. Guest 101 should be George Washington, and 102 Jane Smith. You decide the rest.

2. Insert 5 rows into the table **Booking**.
    You can charge $500 a month rent to each of them.
    Hint: Format for date is '2014-07-01'
    a) Guest 101 would like to use your space from June 1, 2014 – June 30, 2014 and August 1, 2014-August 31, 2014.

b) Guest 102 would like to come July 1, 2014- July 31, 2014 and July 1, 2015-July 31, 2015.
c) Guest 103 would like to come September 1, 2014 – September 30, 2014.

III Selecting Rows

Please write the SQL statements to do the following:

1. Write the statement to list all the details for all rows in the table Booking.
2. Write the statement to list all the details for all rows in the table Guest **ordered by the guests' last names.**
3. Write the statement to list **ONLY** the guestNo from the Booking Table.
4. Write the statement to select all **distinct** guestNos from the Bookings table.
5. Write the statement to select the first name from the Guest Table **where** the guest's last name is Washington.
6. Oh no! You need to look at a guest's record, but you can't quite remember his last name, was it Washing or Wash or Washington or Washings or perhaps Thompson-Wahsings. Write the statement to select his first name given that his last name is something **LIKE** Wash, with, perhaps some characters before Wash and some after.

IV Deleting Rows

Guest 103 decides to cancel his/her reservation.

1. Write the statement to delete only that row from the Booking table.
2. Please verify that s/he is no longer in the table by running the necessary select statement. Write the statement.

V Updating Rows

Congratulations! Jane Smith got married and decided to change her last name to Johnson.

1. Write the statement to update that information.
2. Please write a statement to view that your changes took place correctly.

Have fun, and welcome to SQL!