Homework 2

- These exercises use the hotel schemas shown with the Chapter 4 exercises.
- For each problem, formulate the appropriate mysql commands to solve the problem
- Put your answers into a file named **hw2.txt**. We will cut-and-paste your SQL answers into the mysql query processor on the departmental machines to test your answers.
- When you test your queries, you should add or delete additional sample data to the Hotel schemas as needed. For example, when I was testing my queries for guest and room overbooking, I added sample data that overbooked guests and rooms, and when I was creating an archival table for the booking table, I inserted older dated booking tuples into the Booking relation.

1. 6.18: Make sure that you print the details for all rooms, even unoccupied ones. If the room is occupied, then print the name of the guest staying in that room.

2. 7.11: mySql will not enforce the constraints, but it will check them syntactically. You can check to see whether or not your constraints might work in practice by running them independently of the create table command and seeing whether they would detect the prohibited condition. For example, you can run your overbooking query at the command line to check whether or not it would return overbooked rooms.

3. This problem requires that you write three commands:
    a. Write a command that creates a separate table called **BookingOld** with the same structure as the hotel Booking table to hold archive records.
    b. Using the INSERT statement, copy the records from the Booking table to the archive table relating to bookings before 1st January 2003. Use your judgment in choosing the options to use in creating the new table.
    c. Write a query that deletes all bookings before 1st January 2003 from the Booking table.

4. Create a view named **CurrentGuestCount** that contains a count of the number of guests currently staying at each hotel. Your view should contain columns for the hotelNo and the guest count.

5. Create a view named **HotelData** containing the names of all guests currently staying at one of our hotels and the names of the hotel at which they are staying. Order the results by hotel name.

6. Create a view named **CheckingOutToday** that contains the account information for each guest at the Grosvenor Hotel who is checking out today. The account information should include the guest number, guest name, guest address, room number being checked out of, number of days spent in the room, and the total cost of the stay. You will need to use the DATEDIFF function to calculate the number of dates spent in the room and you will need to do some arithmetic in the field that represents the total cost of the stay.

7. Consider the following view defined on the Hotel schema:

```
CREATE VIEW RoomBookingCount (hotelNo, roomNo, bookingCount)
AS    SELECT b.hotelNo, r.roomNo, COUNT(*)
        FROM Room r, Booking b
        WHERE r.roomNo = b.roomNo AND r.hotelNo = b.hotelNo
        GROUP BY b.hotelNo, r.roomNo;
```

For each of the following queries, state whether the query is valid and for the valid ones show how each of the queries would be mapped onto a query on the underlying base tables. Base your decision on whether a query is valid or invalid on the criteria presented in class and in the book, not on whether the query runs in mysql. All three queries shown below will run in mysql, but they may or may not be valid according to the SQL standard.

(a)
```
SELECT hotelNo, roomNo
FROM RoomBookingCount
WHERE hotelNo = 1;
```

(b)
```
SELECT hotelNo, SUM(bookingCount)
FROM RoomBookingCount
GROUP BY hotelNo;
```

(c)
```
SELECT *
FROM RoomBookingCount
ORDER BY bookingCount;
```

8. Required for graduate students, extra credit (15 points) for undergraduate students: Write an SQL query that prints each branch (i.e., the branchNo) whose staff count exceeds the average staff count for a branch, and print the amount by which each such branch exceeds the average staff count. Use the branch and staff relations from the DreamHome case study in the book.