

# MySQL and NetBeans Practice: Database Manipulation Language (DML) 1

## Contents:

### **MySQL SELECT PRACTICE Some Review Questions**

1. What are the two major components of SQL and what function do they serve?
2. What are the advantages and disadvantages of SQL?
3. Explain the function of each of the clauses in the SELECT statement. What restrictions are imposed on these clauses?
4. What restrictions apply to the use of the aggregate functions within the SELECT statement? How do nulls affect the aggregate functions?
5. Explain how the GROUP BY clause works. What is the difference between the WHERE and HAVING clauses?
6. What is the difference between a subquery and a join? Under what circumstances would you not be able to use a subquery?

### **SQL Queries Exercise Using NetBeans 6.x.x – Create and Insert Executing SQL Queries Exercises**

1. List full details of all hotels.
2. List full details of all hotels in London.
3. List the names and addresses of all guests in London, alphabetically ordered by name.
4. List all double or family rooms with a price below £40.00 per night, in ascending order of price.
5. List the bookings for which no date\_to has been specified.

### **Aggregate Functions**

How many hotels are there?

What is the average price of a room?  
What is the total revenue per night from all double rooms?  
How many different guests have made bookings for August?

### **Subqueries and Joins**

1. List the price and type of all rooms at the Grosvenor Hotel.
2. List all guests currently staying at the Grosvenor Hotel.
3. List the details of all rooms at the Grosvenor Hotel, including the name of the guest staying in the room, if the room is occupied.
4. What is the total income from bookings for the Grosvenor Hotel today?
5. List the rooms that are currently unoccupied at the Grosvenor Hotel.
6. What is the lost income from unoccupied rooms at the Grosvenor Hotel?

### **Grouping**

1. List the number of rooms in each hotel.
2. List the number of rooms in each hotel in London.
3. What is the average number of bookings for each hotel in April?
4. What is the most commonly booked room type for each hotel in London?
5. What is the lost income from unoccupied rooms at each hotel today?

### **Creating and Populating Tables With Data**

1. Using the CREATE TABLE statement, create the Hotel, Room, Booking and Guest tables.
2. Insert records into each of these tables.
3. Update the price of all rooms by 5%.
4. Create a separate table with the same structure as the Booking table to hold archive records. Using the INSERT statement, copy the records from the Booking table to the archive table relating to bookings before 1st January 2008. Delete all bookings before 1st January 2008 from the Booking table.

In this exercise we will be introduced to some theory on SQL and then proceed to creating database, tables and inserting sample data. Then we will start executing the SQL queries against the database that we have created, exploring the SQL DML language. The pre requirements for this tutorial are [MySQL 5.x.x](#) and [NetBeans 6.x.x](#) (or other SQL editor such as MySQL Query Browser etc.)

### **MySQL SELECT PRACTICE**

**Note:** MySQL is not a case sensitive. If you want to retain the case, use double quotes ("" ) for the string identifier. You will find that your uppercase in MySQL script will be converted to a lowercase when executed.

### **Some Review Questions**

#### **1. What are the two major components of SQL and what function do they serve?**

A data definition language (DDL) for defining the database structure. A data manipulation language (DML) for retrieving and updating data.

#### **2. What are the advantages and disadvantages of SQL?**

Advantages:

1. Satisfies ideals for database language
2. (Relatively) Easy to learn
3. Portability
4. SQL standard exists
5. Both interactive and embedded access
6. Can be used by specialist and non-specialist.

Disadvantages:

1. Impedance mismatch - mixing programming paradigms with embedded access
2. Lack of orthogonality - many different ways to express some queries
3. Language is becoming enormous (SQL-92 is 6 times larger than predecessor)
4. Handling of nulls in aggregate functions
5. Result tables are not strictly relational - can contain duplicate tuples, imposes an ordering on both columns and rows.

#### **3. Explain the function of each of the clauses in the SELECT statement. What restrictions are imposed on these clauses?**

- FROM - Specifies the table or tables to be used.
- WHERE - Filters the rows subject to some condition.
- GROUP BY - Forms groups of rows with the same column value.
- HAVING - Filters the groups subject to some condition.

- SELECT - Specifies which columns are to appear in the output.
- ORDER BY - Specifies the order of the output.

If the SELECT list includes an aggregate function and no GROUP BY clause is being used to group data together, then no item in the SELECT list can include any reference to a column unless that column is the argument to an aggregate function.

When GROUP BY is used, each item in the SELECT list must be single-valued per group.

Further, the SELECT clause may only contain:

- Column names.
- Aggregate functions.
- Constants.
- An expression involving combinations of the above.

All column names in the SELECT list must appear in the GROUP BY clause unless the name is used only in an aggregate function.

**4. What restrictions apply to the use of the aggregate functions within the SELECT statement? How do nulls affect the aggregate functions?**

An aggregate function can be used only in the SELECT list and in the HAVING clause. Apart from COUNT(\*), each function eliminates nulls first and operates only on the remaining non-null values. COUNT(\*) counts all the rows of a table, regardless of whether nulls or duplicate values occur.

**5. Explain how the GROUP BY clause works. What is the difference between the WHERE and HAVING clauses?**

SQL first applies the WHERE clause. Then it conceptually arranges the table based on the grouping column(s). Next, applies the HAVING clause and finally orders the result according to the ORDER BY clause. WHERE filters rows subject to some condition; HAVING filters groups subject to some condition.

**6. What is the difference between a subquery and a join? Under what circumstances would you not be able to use a subquery?**

With a subquery, the columns specified in the SELECT list are restricted to one table. Thus, cannot use a subquery if the SELECT list contains columns from more than one table.

**SQL Queries Exercise Using NetBeans 6.x.x**

The following tables form part of a database held in a relational DBMS:

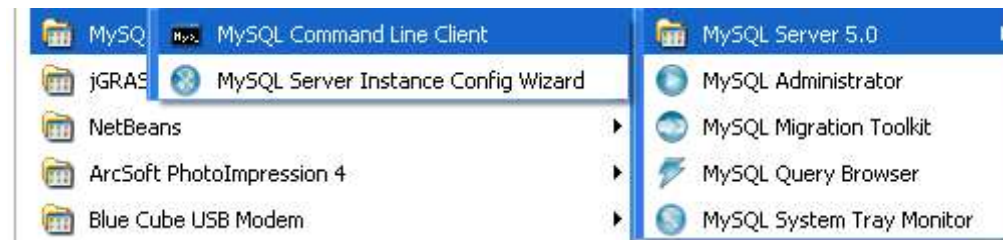
- Hotel (HotelNo, Name, City)
- Room (RoomNo, HotelNo, Type, Price)
- Booking (HotelNo, GuestNo, DateFrom, DateTo, RoomNo)

- Guest (GuestNo, GuestName, GuestAddress)
- where Hotel contains hotel details and HotelNo is the primary key
- Room contains room details for each hotel and (HotelNo, RoomNo) forms the primary key
- Booking contains details of the bookings and the primary key comprises (HotelNo, GuestNo and DateFrom)
- Guest contains guest details and GuestNo is the primary key.

Let create a database and tables. Then populate those tables with some sample data. Firstly, we use MySQL Command Line Client.

1. Create a database named **hotel\_db**. Then switch to NetBeans.

```
CREATE DATABASE hotel_db;
```

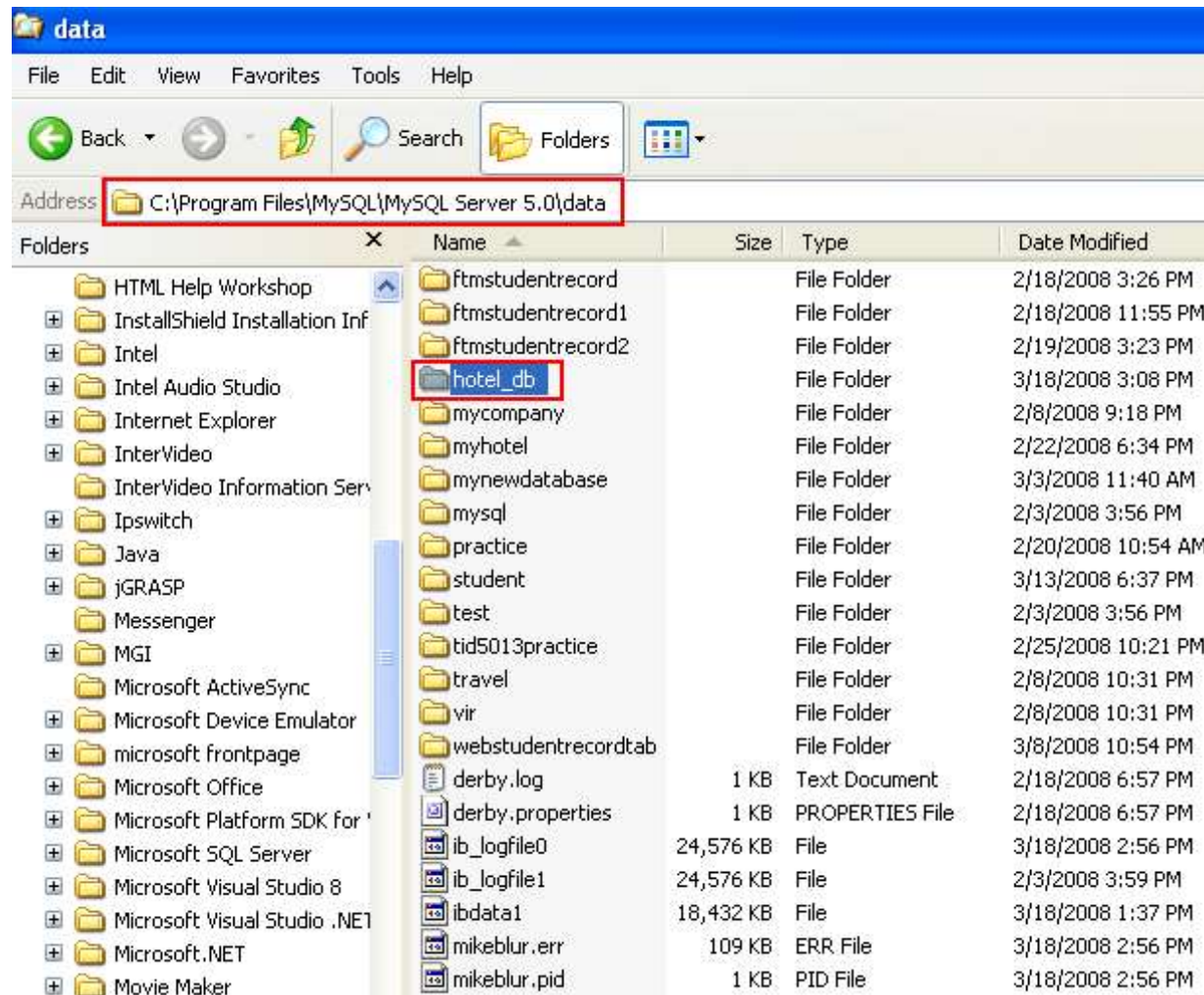


```
mysql> CREATE DATABASE hotel_db;
Query OK, 1 row affected (0.01 sec)

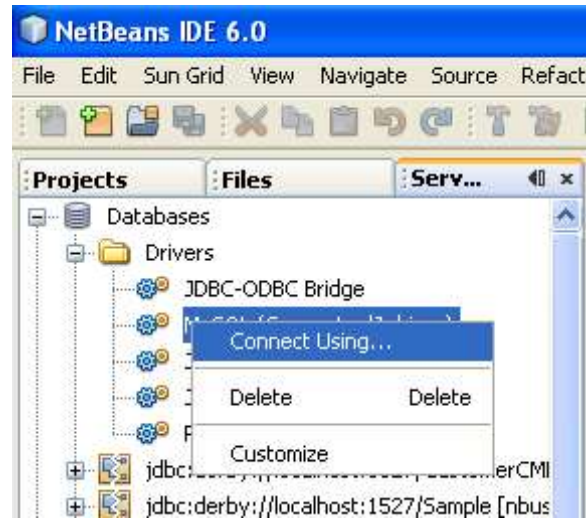
mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| ftmstudentrecord |
| ftmstudentrecord1 |
| ftmstudentrecord2 |
| hotel_db |
| mycompany |
| myhotel |
| mynewdatabase |
| mysql |
| practice |
| student |
| test |
| tid5013practice |
| travel |
| vir |
| webstudentrecordtab |
+-----+
16 rows in set (0.00 sec)

mysql> exit
```

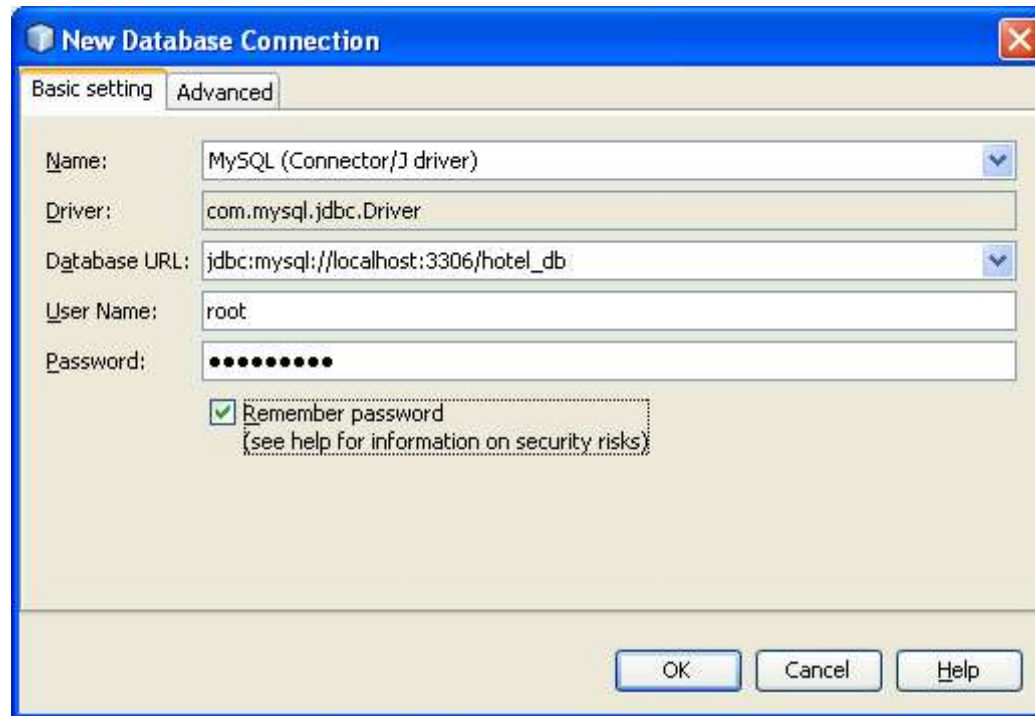
2. If you want to see the database files, it is under the **data** folder of MySQL path.



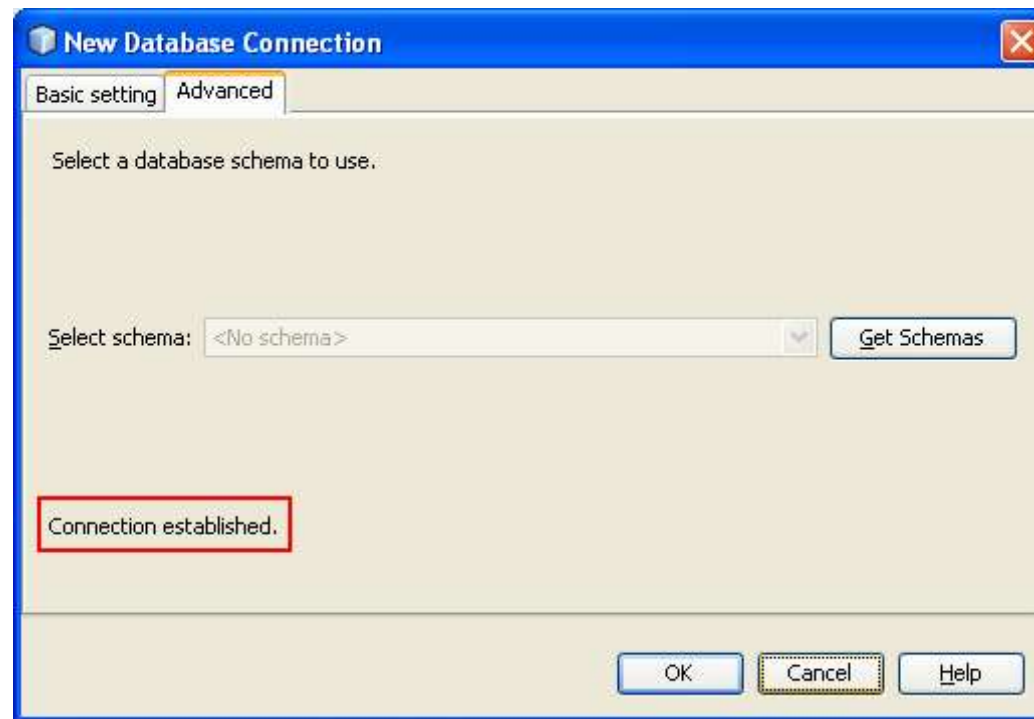
3. You can continue using the MySQL Command Line Client Tool for the following database manipulation exercise, however we we will use NetBeans. Launch NetBeans and connect to the just created database as shown in the following Figures. Select Database > expand Driver > select MySQL (Connector / J driver) > right click mouse > select Connect Using.



4. Fill in the root's password and click OK.

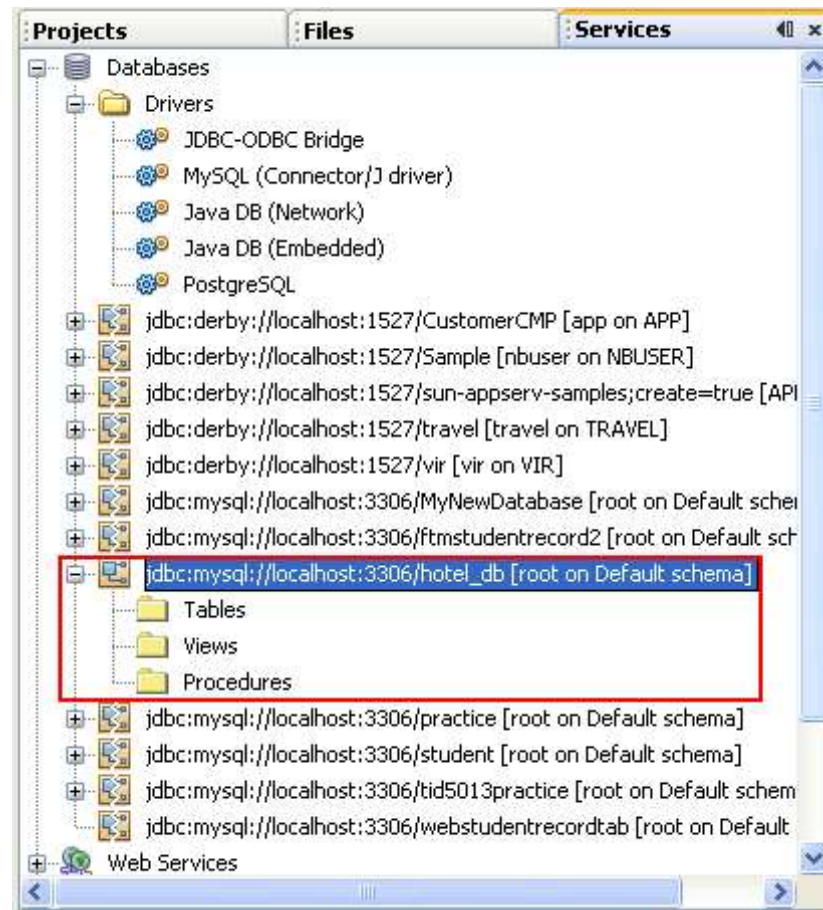


5. The following Figure shows that the connection was established.

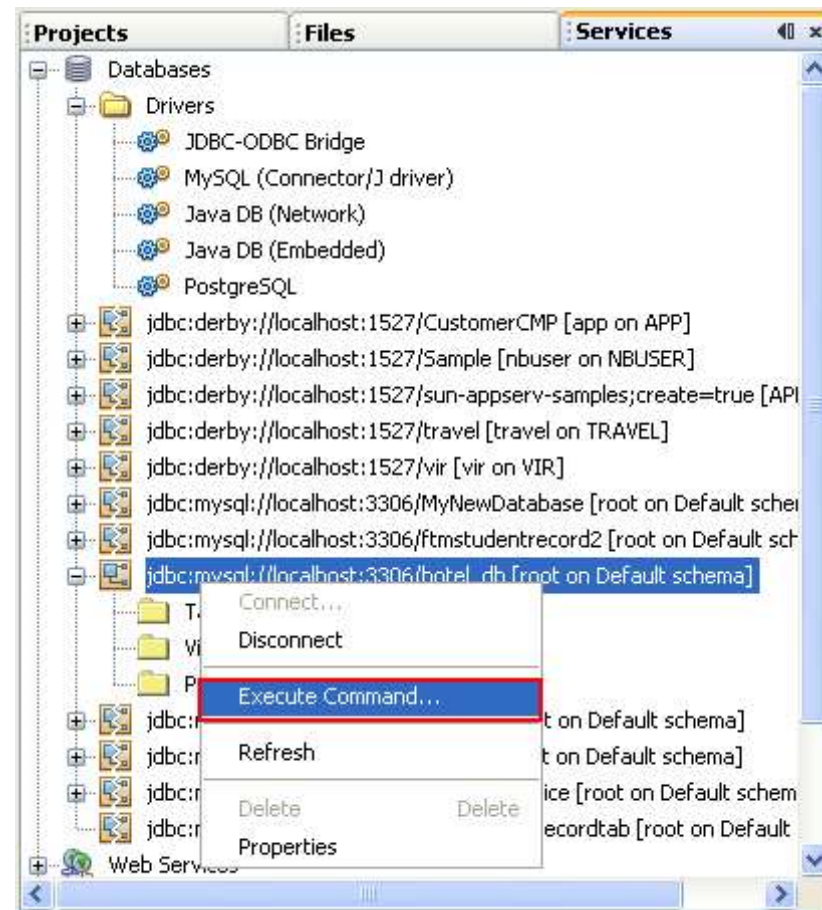



6. The following Figure shows that the connection to the **hotel\_db** database has been established.





7. Let execute SQL queries. Select the hotel\_db connection > right click mouse > select Execute Command. This will launch query editor on the right window.

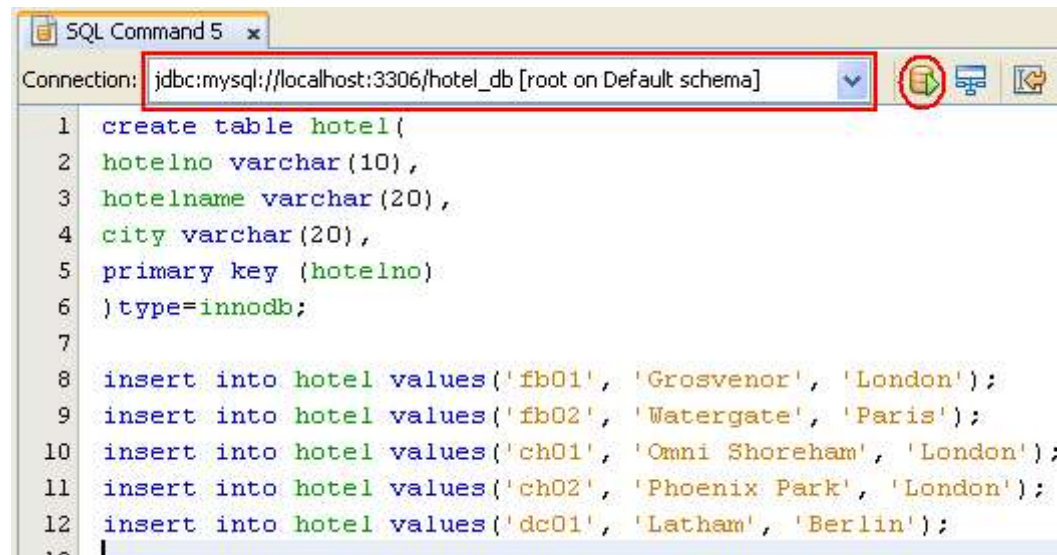


8. Next, create tables and populate them with sample data. Copy and paste the following SQL script into the query editor and click the Run button (  ).

```
-- Script for MySQL 5.x.x exercises
-- Revised 3/24/2008 by yummy
```

```
-- create a table named hotel
create table hotel(
hotelno varchar(10),
hotelname varchar(20),
city varchar(20),
primary key (hotelno)
)type=innodb;
```

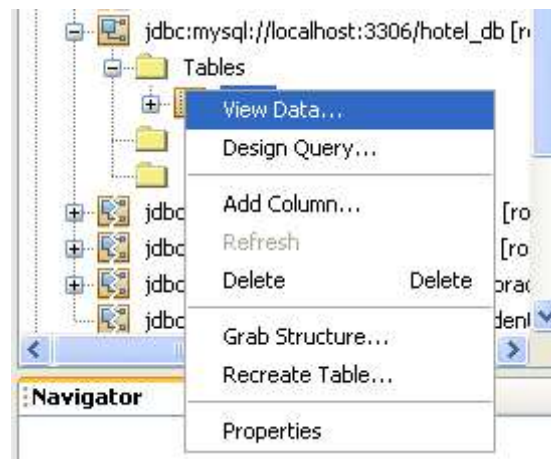
```
-- insert some sample data
insert into hotel values('fb01', 'Grosvenor', 'London');
insert into hotel values('fb02', 'Watergate', 'Paris');
insert into hotel values('ch01', 'Omni Shoreham', 'London');
insert into hotel values('ch02', 'Phoenix Park', 'London');
insert into hotel values('dc01', 'Latham', 'Berlin');
```



The screenshot shows a window titled "SQL Command 5" with a connection string "jdbc:mysql://localhost:3306/hotel\_db [root on Default schema]". The SQL code is as follows:

```
1 create table hotel(  
2 hotelno varchar(10),  
3 hotelname varchar(20),  
4 city varchar(20),  
5 primary key (hotelno)  
6 )type=innodb;  
7  
8 insert into hotel values('fb01', 'Grosvenor', 'London');  
9 insert into hotel values('fb02', 'Watergate', 'Paris');  
10 insert into hotel values('ch01', 'Omni Shoreham', 'London');  
11 insert into hotel values('ch02', 'Phoenix Park', 'London');  
12 insert into hotel values('dc01', 'Latham', 'Berlin');
```

9. If there is no error, view the data. Select hotel table > right click mouse > Select View Data.



10. The hotel table and its data shown in the following Figure.

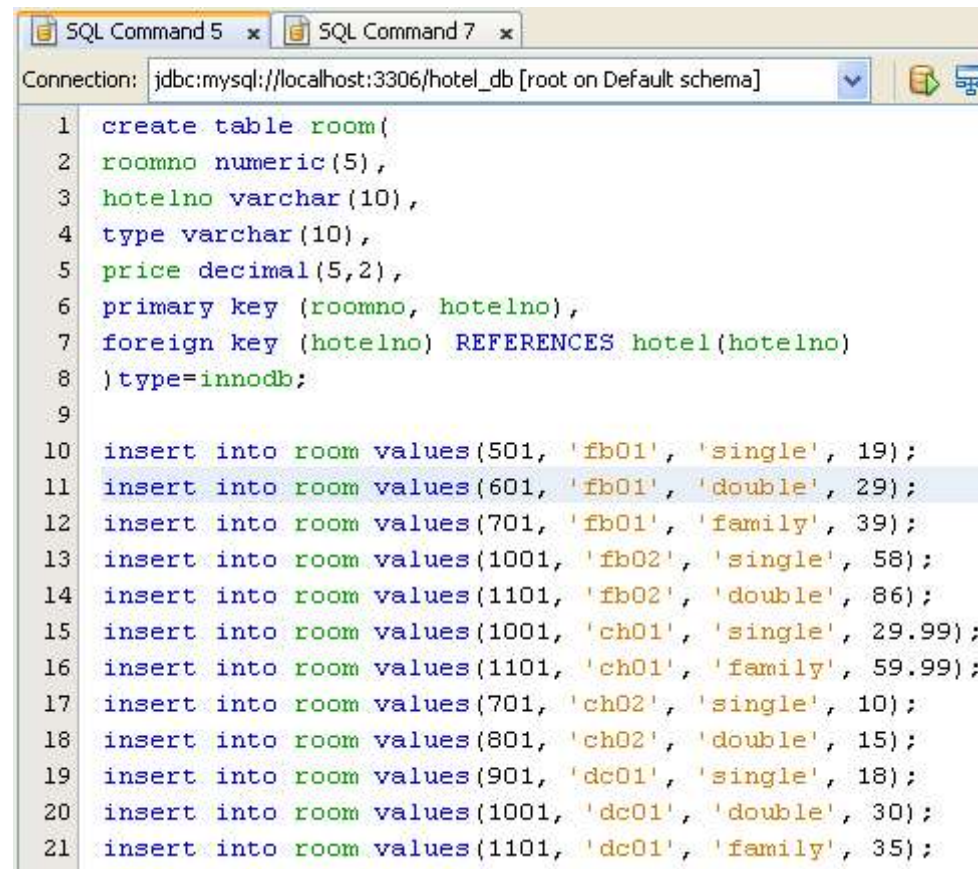
The screenshot shows a SQL command window with two tabs: 'SQL Command 5' and 'SQL Command 7'. The connection string is 'jdbc:mysql://localhost:3306/hotel\_db [root on Default schema]'. The command entered is 'select \* from hotel'. The result is displayed in a table with 3 columns: 'hotelno', 'hotelname', and 'city'. There are 5 rows of data.

hotelno	hotelname	city
ch01	Omni Shoreham	London
ch02	Phoenix Park	London
dc01	Latham	Berlin
fb01	Grosvenor	London
fb02	Watergate	Paris

11.Next, create more tables and populate sample data. Follow the same steps as previously done.

```
create table room(
roomno numeric(5),
hotelno varchar(10),
type varchar(10),
price decimal(5,2),
primary key (roomno, hotelno),
foreign key (hotelno) REFERENCES hotel(hotelno)
)type=innodb;

insert into room values(501, 'fb01', 'single', 19);
insert into room values(601, 'fb01', 'double', 29);
insert into room values(701, 'fb01', 'family', 39);
insert into room values(1001, 'fb02', 'single', 58);
insert into room values(1101, 'fb02', 'double', 86);
insert into room values(1001, 'ch01', 'single', 29.99);
insert into room values(1101, 'ch01', 'family', 59.99);
insert into room values(701, 'ch02', 'single', 10);
insert into room values(801, 'ch02', 'double', 15);
insert into room values(901, 'dc01', 'single', 18);
insert into room values(1001, 'dc01', 'double', 30);
insert into room values(1101, 'dc01', 'family', 35);
```



The screenshot shows a SQL command window with two tabs: 'SQL Command 5' and 'SQL Command 7'. The connection string is 'jdbc:mysql://localhost:3306/hotel\_db [root on Default schema]'. The SQL code is as follows:

```
1 create table room(  
2 roomno numeric(5),  
3 hotelno varchar(10),  
4 type varchar(10),  
5 price decimal(5,2),  
6 primary key (roomno, hotelno),  
7 foreign key (hotelno) REFERENCES hotel(hotelno)  
8 )type=innodb;  
9  
10 insert into room values(501, 'fb01', 'single', 19);  
11 insert into room values(601, 'fb01', 'double', 29);  
12 insert into room values(701, 'fb01', 'family', 39);  
13 insert into room values(1001, 'fb02', 'single', 58);  
14 insert into room values(1101, 'fb02', 'double', 86);  
15 insert into room values(1001, 'ch01', 'single', 29.99);  
16 insert into room values(1101, 'ch01', 'family', 59.99);  
17 insert into room values(701, 'ch02', 'single', 10);  
18 insert into room values(801, 'ch02', 'double', 15);  
19 insert into room values(901, 'dc01', 'single', 18);  
20 insert into room values(1001, 'dc01', 'double', 30);  
21 insert into room values(1101, 'dc01', 'family', 35);
```

12. Then verify the process.



SQL Command 5 x SQL Command 7 x SQL Command 8 x				
Connection: jdbc:mysql://localhost:3306/hotel_db [root on Default schema]				
1	select * from room			
1:1	INS			
roomno	hotelno	type	price	
501	fb01	single	19.00	
601	fb01	double	29.00	
701	ch02	single	10.00	
701	fb01	family	39.00	
801	ch02	double	15.00	
901	dc01	single	18.00	
1001	ch01	single	29.99	
1001	dc01	double	30.00	
1001	fb02	single	58.00	
1101	ch01	family	59.99	
1101	dc01	family	35.00	
1101	fb02	double	86.00	

```
create table guest(
  guestno numeric(5),
  guestname varchar(20),
  guestaddress varchar(50),
  primary key (guestno)
)type=innodb;
```

```
insert into guest values(10001, 'John Kay', '56 High St, London');
insert into guest values(10002, 'Mike Ritchie', '18 Tain St, London');
insert into guest values(10003, 'Mary Tregear', '5 Tarbot Rd, Aberdeen');
insert into guest values(10004, 'Joe Keogh', '2 Fergus Dr, Aberdeen');
insert into guest values(10005, 'Carol Farrel', '6 Achray St, Glasgow');
insert into guest values(10006, 'Tina Murphy', '63 Well St, Glasgow');
insert into guest values(10007, 'Tony Shaw', '12 Park Pl, Glasgow');
```

```
SQL Command 5 x SQL Command 7 x SQL Command 8 x
Connection: jdbc:mysql://localhost:3306/hotel_db [root on Default schema]
1 create table guest (
2 guestno numeric(5),
3 guestname varchar(20),
4 guestaddress varchar(50),
5 primary key (guestno)
6 ) type=innodb;
7
8 insert into guest values(10001, 'John Kay', '56 High St, London');
9 insert into guest values(10002, 'Mike Ritchie', '18 Tain St, London');
10 insert into guest values(10003, 'Mary Tregear', '5 Tarbot Rd, Aberdeen');
11 insert into guest values(10004, 'Joe Keogh', '2 Fergus Dr, Aberdeen');
12 insert into guest values(10005, 'Carol Farrel', '6 Achray St, Glasgow');
13 insert into guest values(10006, 'Tina Murphy', '63 Well St, Glasgow');
14 insert into guest values(10007, 'Tony Shaw', '12 Park Pl, Glasgow');
```

SQL Command 9 x SQL Command 5 x SQL Command 7 x

Connection: jdbc:mysql://localhost:3306/hotel\_db [root on Default schema]

1 select \* from guest

1:1 INS

guestno	guestname	guestaddress
10001	John Kay	56 High St, London
10002	Mike Ritchie	18 Tain St, London
10003	Mary Tregear	5 Tarbot Rd, Aberdeen
10004	Joe Keogh	2 Fergus Dr, Aberdeen
10005	Carol Farrel	6 Achray St, Glasgow
10006	Tina Murphy	63 Well St, Glasgow
10007	Tony Shaw	12 Park Pl, Glasgow

```
create table booking(
hotelno varchar(10),
guestno numeric(5),
datefrom date,
dateto date,
```



```
roomno numeric(5),
primary key (hotelno, guestno, datefrom),
foreign key (roomno, hotelno) REFERENCES room(roomno, hotelno),
foreign key (guestno) REFERENCES guest(guestno)
)type=innodb;
```

```
insert into booking values('fb01', 10001, '04-04-01', '04-04-08', 501);
insert into booking values('fb01', 10004, '04-04-15', '04-05-15', 601);
insert into booking values('fb01', 10005, '04-05-02', '04-05-07', 501);
insert into booking values('fb01', 10002, '16-05-04', '04-05-29', 601);
insert into booking values('fb01', 10001, '04-05-01', null, 701);
insert into booking values('fb02', 10003, '04-04-05', '10-04-04', 1001);
insert into booking values('fb02', 10005, '04-05-12', '30-05-04', 1101);
insert into booking values('ch01', 10006, '04-04-21', null, 1101);
insert into booking values('ch02', 10002, '04-04-25', '04-05-06', 801);
insert into booking values('dc01', 10007, '04-05-13', '04-05-15', 1001);
insert into booking values('dc01', 10003, '04-05-20', null, 1001);
```

```
SQL Command 5 x
Connection: jdbc:mysql://localhost:3306/hotel_db [root on Default schema]

1 create table booking(
2 hotelno varchar(10),
3 guestno numeric(5),
4 datefrom date,
5 dateto date,
6 roomno numeric(5),
7 primary key (hotelno, guestno, datefrom),
8 foreign key (roomno, hotelno) REFERENCES room(roomno, hotelno),
9 foreign key (guestno) REFERENCES guest(guestno)
10 )type=innodb;
11
12 insert into booking values('fb01', 10001, '04-04-01', '04-04-08', 501);
13 insert into booking values('fb01', 10004, '04-04-15', '04-05-15', 601);
14 insert into booking values('fb01', 10005, '04-05-02', '04-05-07', 501);
15 insert into booking values('fb01', 10002, '16-05-04', '04-05-29', 601);
16 insert into booking values('fb01', 10001, '04-05-01', null, 701);
17 insert into booking values('fb02', 10003, '04-04-05', '10-04-04', 1001);
18 insert into booking values('fb02', 10005, '04-05-12', '30-05-04', 1101);
19 insert into booking values('ch01', 10006, '04-04-21', null, 1101);
20 insert into booking values('ch02', 10002, '04-04-25', '04-05-06', 801);
21 insert into booking values('dc01', 10007, '04-05-13', '04-05-15', 1001);
22 insert into booking values('dc01', 10003, '04-05-20', null, 1001);
```

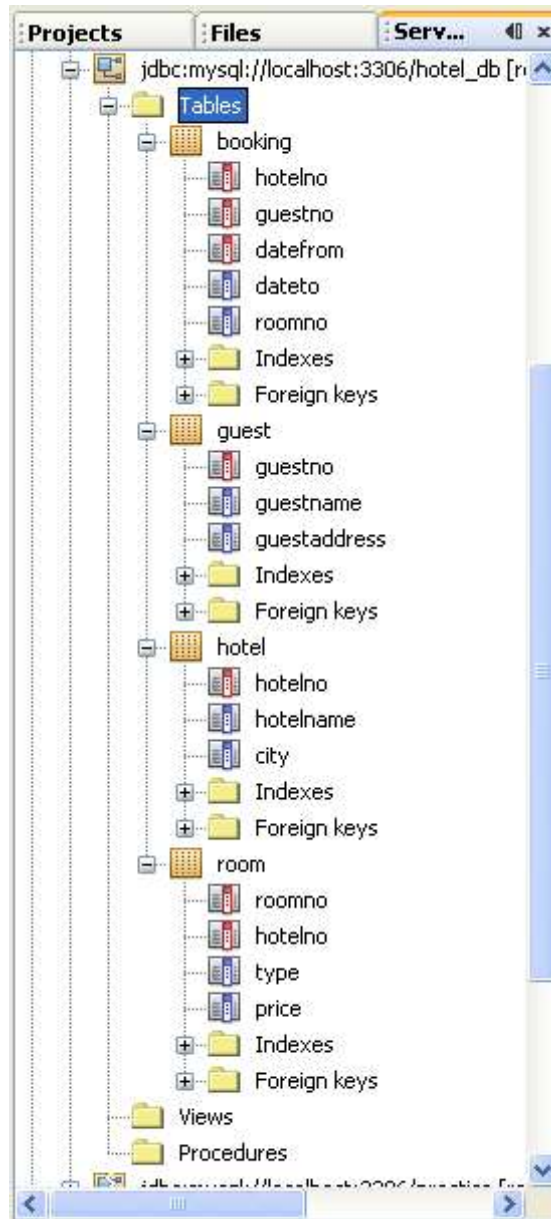
SQL Command 5 x SQL Command 7 x

Connection: jdbc:mysql://localhost:3306/hotel\_db [root on Default schema]

1 select \* from booking

1:1 INS

hotelno	guestno	datefrom	dateto	roomno
ch01	10006	Apr 21, 2004	NULL	1101
ch02	10002	Apr 25, 2004	May 6, 2004	801
dc01	10003	May 20, 2004	NULL	1001
dc01	10007	May 13, 2004	May 15, 2004	1001
fb01	10001	Apr 1, 2004	Apr 8, 2004	501
fb01	10001	May 1, 2004	NULL	701
fb01	10002	May 4, 2016	May 29, 2004	601
fb01	10004	Apr 15, 2004	May 15, 2004	601
fb01	10005	May 2, 2004	May 7, 2004	501
fb02	10003	Apr 5, 2004	Apr 4, 2010	1001
fb02	10005	May 12, 2004	May 4, 2030	1101

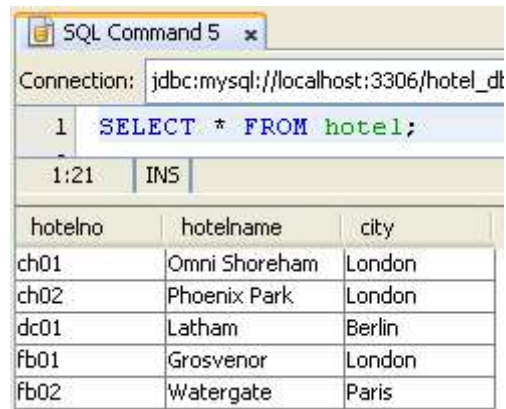


## Executing SQL Queries Exercises

Next, let execute SQL queries.

### 1. List full details of all hotels.

```
SELECT * FROM hotel;
```



SQL Command 5

Connection: jdbc:mysql://localhost:3306/hotel\_db

1 SELECT \* FROM hotel;

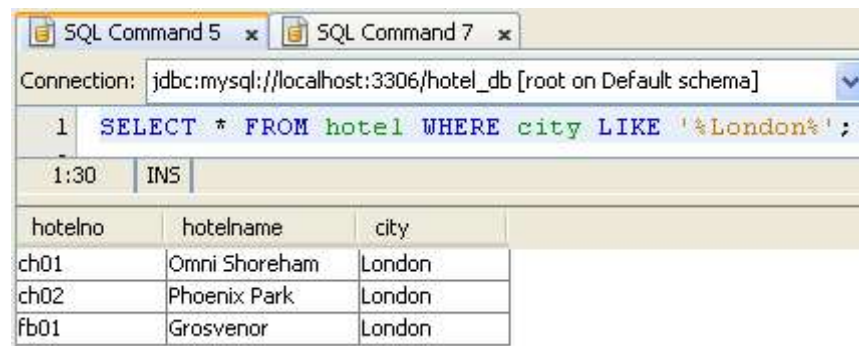
1:21 INS

hotelno	hotelname	city
ch01	Omni Shoreham	London
ch02	Phoenix Park	London
dc01	Latham	Berlin
fb01	Grosvenor	London
fb02	Watergate	Paris

### 2. List full details of all hotels in London.

```
SELECT * FROM hotel WHERE city LIKE '%London%';
```

Strictly speaking, this would also find rows with an address like: '10 London Avenue, New York'.



SQL Command 5 x SQL Command 7 x

Connection: jdbc:mysql://localhost:3306/hotel\_db [root on Default schema]

1 SELECT \* FROM hotel WHERE city LIKE '%London%';

1:30 INS

hotelno	hotelname	city
ch01	Omni Shoreham	London
ch02	Phoenix Park	London
fb01	Grosvenor	London

### 3. List the names and addresses of all guests in London, alphabetically ordered by name.

```
SELECT guestname, guestaddress  
FROM guest  
WHERE guestaddress LIKE '%London%'  
ORDER BY guestname;
```

SQL Command 5 x SQL Command 7 x SQL Command 8 x

Connection: jdbc:mysql://localhost:3306/hotel\_db [root on Default schema]

```
1 SELECT guestname, guestaddress
2 FROM guest
3 WHERE guestaddress LIKE '%London%'
4 ORDER BY guestname;
5
```

4:15 INS

guestname	guestaddress
John Kay	56 High St, London
Mike Ritchie	18 Tain St, London

4. List all double or family rooms with a price below £40.00 per night, in ascending order of price.

```
SELECT * FROM room
WHERE price < 40 AND type IN ('Double', 'Family')
ORDER BY price;
```

(Note, ASC is the default setting).

SQL Command 9 x SQL Command 5 x SQL Command 7 x SQL Com

Connection: jdbc:mysql://localhost:3306/hotel\_db [root on Default schema]

```
1 SELECT * FROM room
2 WHERE price < 40 AND type IN ('Double', 'Family')
3 ORDER BY price;
```

1:19 INS

roomno	hotelno	type	price
801	ch02	double	15.00
601	fb01	double	29.00
1001	dc01	double	30.00
1101	dc01	family	35.00
701	fb01	family	39.00

5. List the bookings for which no date\_to has been specified.

```
SELECT * FROM booking WHERE dateto IS NULL;
```

SQL Command 9 x SQL Command 5 x SQL Command 7 x					
Connection: jdbc:mysql://localhost:3306/hotel_db [root on Default schema]					
1	SELECT * FROM booking WHERE dateto IS NULL;				
1:33	INS				
hotelno	guestno	datefrom	dateto	roomno	
ch01	10006	Apr 21, 2004	NULL	1101	
dc01	10003	May 20, 2004	NULL	1001	
fb01	10001	May 1, 2004	NULL	701	

## Aggregate Functions

### 1. How many hotels are there?

```
SELECT COUNT(*) FROM hotel;
```

SQL Command 9 x SQL Command 5 x SQL Command 7 x	
Connection: jdbc:mysql://localhost:3306/hotel_db [root on Default schema]	
1	SELECT COUNT(*) FROM hotel;
1:28	INS
COUNT(*)	
	5

### 2. What is the average price of a room?

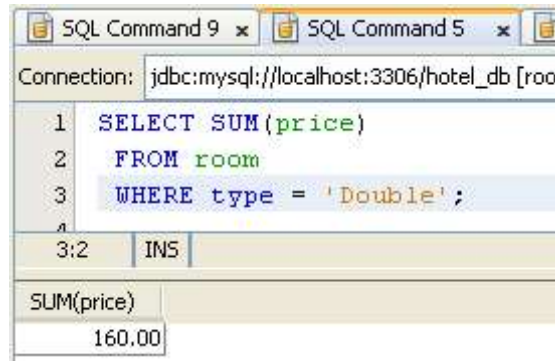
```
SELECT AVG(price) FROM room;
```

SQL Command 9 x SQL Command 5 x SQL Command 7 x	
Connection: jdbc:mysql://localhost:3306/hotel_db [root on Default schema]	
1	SELECT AVG(price) FROM room;
2	
1:29	INS
AVG(price)	
	35.748333



### 3. What is the total revenue per night from all double rooms?

```
SELECT SUM(price) FROM room WHERE type = 'Double';
```

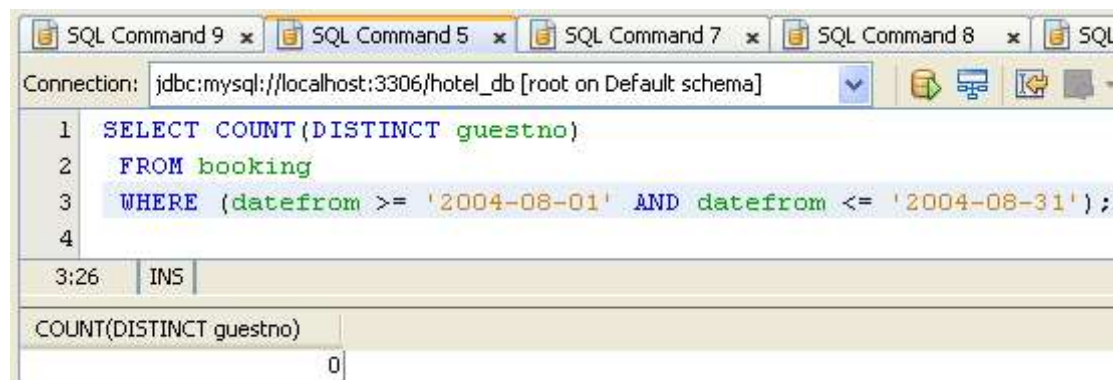


The screenshot shows a SQL Command window with the following content:

SQL Command 9 x SQL Command 5 x	
Connection: jdbc:mysql://localhost:3306/hotel_db [root	
1	SELECT SUM(price)
2	FROM room
3	WHERE type = 'Double';
3:2	INS
SUM(price)	
	160.00

### 4. How many different guests have made bookings for August?

```
SELECT COUNT(DISTINCT guestno)
FROM booking
WHERE (datefrom >= '2004-08-01' AND datefrom <= '2004-08-31');
```



The screenshot shows a SQL Command window with the following content:

SQL Command 9 x SQL Command 5 x SQL Command 7 x SQL Command 8 x SQL	
Connection: jdbc:mysql://localhost:3306/hotel_db [root on Default schema]	
1	SELECT COUNT(DISTINCT guestno)
2	FROM booking
3	WHERE (datefrom >= '2004-08-01' AND datefrom <= '2004-08-31');
4	
3:26	INS
COUNT(DISTINCT guestno)	
	0

What about for May?

```
SELECT COUNT(DISTINCT guestno)
FROM booking
WHERE (datefrom >= '2004-05-01' AND datefrom <= '2004-05-31');
```



SQL Command 9 x		SQL Command 5 x		SQL Command 7 x		SQL Command 8 x		SQL	
Connection: jdbc:mysql://localhost:3306/hotel_db [root on Default schema]									
1	SELECT COUNT(DISTINCT guestno)								
2	FROM booking								
3	WHERE (datefrom >= '2004-05-01' AND datefrom <= '2004-05-31');								
4									
3:58	INS								
COUNT(DISTINCT guestno)									
4									

## Subqueries and Joins

### 1. List the price and type of all rooms at the Grosvenor Hotel.

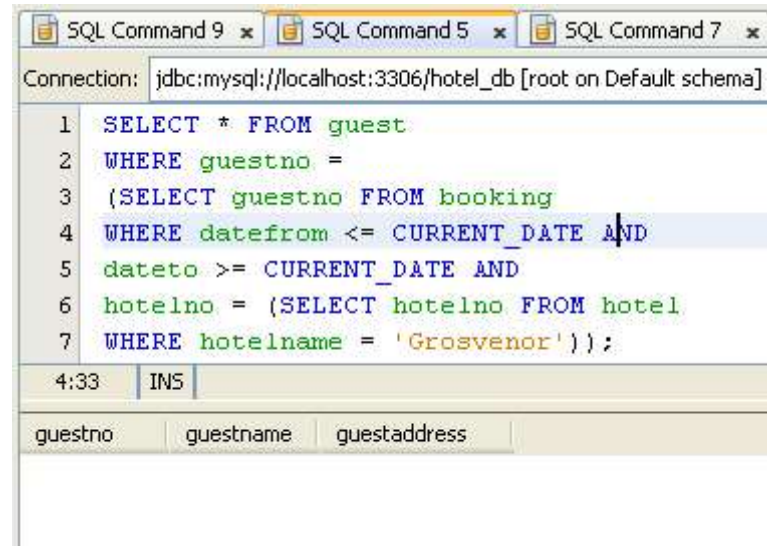
```
SELECT price, type
FROM room
WHERE hotelno = (SELECT hotelno FROM hotel
WHERE hotelname = 'Grosvenor');
```

SQL Command 9 x		SQL Command 5 x	SQL Command 7 x	SQL
Connection: jdbc:mysql://localhost:3306/hotel_db [root on Default schema]				
1	SELECT price, type			
2	FROM room			
3	WHERE hotelno = (SELECT hotelno FROM hotel			
4	WHERE hotelname = 'Grosvenor');			
4:1	INS			
price	type			
19.00	single			
29.00	double			
39.00	family			

### 2. List all guests currently staying at the Grosvenor Hotel.

```
SELECT * FROM guest
WHERE guestno =
(SELECT guestno FROM booking
```

```
WHERE datefrom <= CURRENT_DATE AND dateto >= CURRENT_DATE AND  
hotelno = (SELECT hotelno FROM hotel  
WHERE hotelname = 'Grosvenor'));
```



**3. List the details of all rooms at the Grosvenor Hotel, including the name of the guest staying in the room, if the room is occupied.**

```
SELECT r.* FROM room r LEFT JOIN  
(SELECT g.guestname, h.hotelno, b.roomno FROM Guest g, Booking b, Hotel h  
WHERE g.guestno = b.guestno AND b.hotelno = h.hotelno AND  
h.hotelname= 'Grosvenor' AND  
b.datefrom <= CURRENT_DATE AND b.dateto >= CURRENT_DATE) AS XXX  
ON r.hotelno = XXX.hotelno AND r.roomno = XXX.roomno;
```

-----

-----

SQL Command 9 x SQL Command 5 x SQL Command 7 x SQL Command 8 x SQL Command 10 x				
Connection: jdbc:mysql://localhost:3306/hotel_db [root on Default schema]				
1	SELECT r.* FROM room r LEFT JOIN			
2	(SELECT g.guestname, h.hotelno, b.roomno FROM Guest g, Booking b, Hotel h			
3	WHERE g.guestno = b.guestno AND b.hotelno = h.hotelno AND			
4	h.hotelname= 'Grosvenor' AND			
5	b.datefrom <= CURRENT_DATE AND b.dateto >= CURRENT_DATE) AS XXX			
6	ON r.hotelno = XXX.hotelno AND r.roomno = XXX.roomno;			
7				
4:24	INS			
roomno	hotelno	type	price	
501	fb01	single	19.00	
601	fb01	double	29.00	
701	ch02	single	10.00	
701	fb01	family	39.00	
801	ch02	double	15.00	
901	dc01	single	18.00	
1001	ch01	single	29.99	
1001	dc01	double	30.00	
1001	fb02	single	58.00	
1101	ch01	family	59.99	
1101	dc01	family	35.00	
1101	fb02	double	86.00	

#### 4. What is the total income from bookings for the Grosvenor Hotel today?

```
SELECT SUM(price) FROM booking b, room r, hotel h
WHERE (b.datefrom <= CURRENT_DATE AND
b.dateto >= CURRENT_DATE) AND
r.hotelno = h.hotelno AND r.roomno = b.roomno;
```

SQL Command 9 x SQL Command 5 x SQL Command 7 x SQL Com		
Connection: jdbc:mysql://localhost:3306/hotel_db [root on Default schema]		
1	SELECT SUM(price) FROM booking b, room r, hotel h	
2	WHERE (b.datefrom <= CURRENT_DATE AND	
3	b.dateto >= CURRENT_DATE) AND	
4	r.hotelno = h.hotelno AND r.roomno = b.roomno;	
5		
4:26	INS	
SUM(price)		
	298,98	

**5. List the rooms that are currently unoccupied at the Grosvenor Hotel.**

```
SELECT * FROM room r
WHERE roomno NOT IN
(SELECT roomno FROM booking b, hotel h
WHERE (datefrom <= CURRENT_DATE AND
dateto >= CURRENT_DATE) AND
b.hotelno = h.hotelno AND hotelname = 'Grosvenor');
```

SQL Command 9 x SQL Command 5 x SQL Command 7 x SQL Command 8 x				
Connection: jdbc:mysql://localhost:3306/hotel_db [root on Default schema]				
1	SELECT * FROM room r			
2	WHERE roomno NOT IN			
3	(SELECT roomno FROM booking b, hotel h			
4	WHERE (datefrom <= CURRENT_DATE AND			
5	dateto >= CURRENT_DATE) AND			
6	b.hotelno = h.hotelno AND hotelname = 'Grosvenor');			
6:32	INS			
roomno	hotelno	type	price	
501	fb01	single	19.00	
601	fb01	double	29.00	
701	ch02	single	10.00	
701	fb01	family	39.00	
801	ch02	double	15.00	
901	dc01	single	18.00	
1001	ch01	single	29.99	
1001	dc01	double	30.00	
1001	fb02	single	58.00	
1101	ch01	family	59.99	
1101	dc01	family	35.00	
1101	fb02	double	86.00	

## 6. What is the lost income from unoccupied rooms at the Grosvenor Hotel?

```
SELECT SUM(price) FROM room r
WHERE roomno NOT IN
(SELECT roomno FROM booking b, hotel h
WHERE (datefrom <= CURRENT_DATE AND
dateto >= CURRENT_DATE) AND
b.hotelno = h.hotelno AND hotelname = 'Grosvenor');
```

SQL Command 9 x SQL Command 5 x SQL Command 7 x SQL Command 8 x		
Connection: jdbc:mysql://localhost:3306/hotel_db [root on Default schema]		
1	SELECT SUM(price) FROM room r	
2	WHERE roomno NOT IN	
3	(SELECT roomno FROM booking b, hotel h	
4	WHERE (datefrom <= CURRENT_DATE AND	
5	dateto >= CURRENT_DATE) AND	
6	b.hotelno = h.hotelno AND hotelname = 'Grosvenor');	
7		
6:49	INS	
SUM(price)		
		428.98

## Grouping

### 1. List the number of rooms in each hotel.

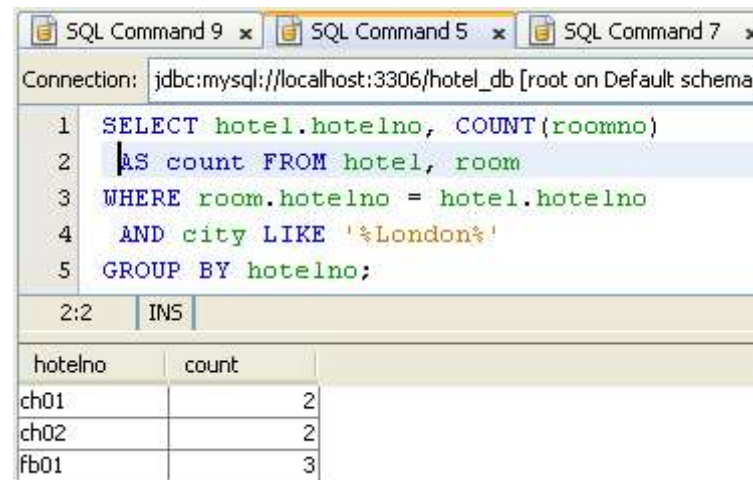
```
SELECT hotelno, COUNT(roomno) AS count FROM room
GROUP BY hotelno;
```

SQL Command 9 x SQL Command 5 x SQL Command 7 x SQL Command 8 x		
Connection: jdbc:mysql://localhost:3306/hotel_db [root on Default schema]		
1	SELECT hotelno, COUNT(roomno) AS count FROM room	
2	GROUP BY hotelno;	
1:27	INS	
hotelno	count	
ch01		2
ch02		2
dc01		3
fb01		3
fb02		2

### 2. List the number of rooms in each hotel in London.

```
SELECT hotel.hotelno, COUNT(roomno)
AS count FROM hotel, room
WHERE room.hotelno = hotel.hotelno
```

```
AND city LIKE '%London%'
GROUP BY hotelno;
```



The screenshot shows a MySQL command window with three tabs: 'SQL Command 9', 'SQL Command 5', and 'SQL Command 7'. The active tab is 'SQL Command 9'. The connection string is 'jdbc:mysql://localhost:3306/hotel\_db [root on Default schema]'. The SQL query is as follows:

```
1 SELECT hotel.hotelno, COUNT(roomno)
2 AS count FROM hotel, room
3 WHERE room.hotelno = hotel.hotelno
4 AND city LIKE '%London%'
5 GROUP BY hotelno;
```

The query is executed, and the results are displayed in a table with two columns: 'hotelno' and 'count'. The results are:

hotelno	count
ch01	2
ch02	2
fb01	3

### 3. What is the average number of bookings for each hotel in April?

```
SELECT AVG(X) AS AveNumBook FROM
  (SELECT hotelno, COUNT(hotelno) AS X
   FROM booking b
   WHERE (b.datefrom >= DATE'2004-04-01' AND b.datefrom <= DATE'2004-04-31'))
GROUP BY hotelno) AS AnotherThing;
```

```
/*SELECT AVG(sum_column1) AS AvgSumCol1
   FROM (SELECT SUM(column1) AS sum_column1
         FROM t1 GROUP BY column1) AS t1;*/
```



SQL Command 9 x		SQL Command 5 x		SQL Command 7 x		SQL Command 8 x		SQL Command 10 x	
Connection: jdbc:mysql://localhost:3306/hotel_db [root on Default schema]									
1	SELECT AVG(X) AS AveNumBook FROM								
2	( SELECT hotelno, COUNT(hotelno) AS X								
3	FROM booking b								
4	WHERE (b.datefrom >= DATE'2004-04-01' AND b.datefrom <= DATE'2004-04-31')								
5	GROUP BY hotelno) AS AnotherThing;								
6									
7	/*SELECT AVG(sum_column1) AS AvgSumCol1								
8	FROM (SELECT SUM(column1) AS sum_column1								
9	FROM t1 GROUP BY column1) AS t1;*/								
5:30	INS								
AveNumBook									
1.2500									

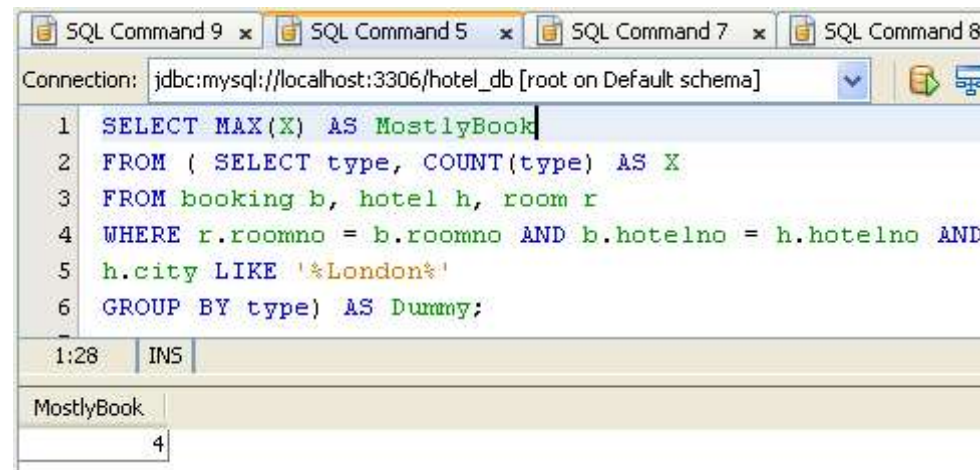
Syntax: SELECT ... FROM (subquery) [AS] name ...

The [AS] name clause is mandatory, because every table in a FROM clause must have a name.

#### 4. What is the **most** commonly booked room type for each hotel in London?

```
SELECT MAX(X) AS MostlyBook
FROM (SELECT type, COUNT(type) AS X
FROM booking b, hotel h, room r
WHERE r.roomno = b.roomno AND b.hotelno = h.hotelno AND
h.city LIKE '%London%'
GROUP BY type) AS Dummy;
```





The screenshot shows a SQL command window with four tabs: 'SQL Command 9', 'SQL Command 5', 'SQL Command 7', and 'SQL Command 8'. The active tab is 'SQL Command 9'. The connection string is 'jdbc:mysql://localhost:3306/hotel\_db [root on Default schema]'. The SQL query is as follows:

```
1 SELECT MAX(X) AS MostlyBook
2 FROM ( SELECT type, COUNT(type) AS X
3 FROM booking b, hotel h, room r
4 WHERE r.roomno = b.roomno AND b.hotelno = h.hotelno AND
5 h.city LIKE '%London%'
6 GROUP BY type) AS Dummy;
```

Below the query, the execution time is '1:28' and the status is 'INS'. The result set is shown in a table with one column, 'MostlyBook', and one row with the value '4'.

MostlyBook
4

Syntax: `SELECT ... FROM (subquery) [AS] name ...`

The `[AS]` name clause is mandatory, because every table in a `FROM` clause must have a name.

##### 5. What is the lost income from unoccupied rooms at each hotel today?

```
SELECT hotelno, SUM(price) FROM room r
WHERE roomno NOT IN
(SELECT roomno FROM booking b, hotel h
WHERE (datefrom <= CURRENT_DATE AND
dateto >= CURRENT_DATE) AND
b.hotelno = h.hotelno)
GROUP BY hotelno;
```

SQL Command 9 x SQL Command 5 x SQL Command 7 x SQL C		
Connection: jdbc:mysql://localhost:3306/hotel_db [root on Default schema]		
1	SELECT hotelno, SUM(price) FROM room r	
2	WHERE roomno NOT IN	
3	(SELECT roomno FROM booking b, hotel h	
4	WHERE (datefrom <= CURRENT_DATE AND	
5	dateto >= CURRENT_DATE) AND	
6	b.hotelno = h.hotelno)	
7	GROUP BY hotelno;	
7:15	INS	
hotelno	SUM(price)	
ch02	25.00	
dc01	18.00	
fb01	87.00	

## Creating and Populating Tables

### 1. Using the CREATE TABLE statement, create the Hotel, Room, Booking and Guest tables.

```
CREATE TABLE hotel(
    hotelno CHAR(4) NOT NULL,
    hotelname VARCHAR(20) NOT NULL,
    city VARCHAR(50) NOT NULL
);
```

```
CREATE TABLE room(
    roomno VARCHAR(4) NOT NULL,
    hotelno CHAR(4) NOT NULL,
    type CHAR(1) NOT NULL,
    price DECIMAL(5,2) NOT NULL
);
```

```
CREATE TABLE booking(
    hotelno CHAR(4) NOT NULL,
    guestno CHAR(4) NOT NULL,
    datefrom DATETIME NOT NULL,
    dateto DATETIME NULL,
    roomno CHAR(4) NOT NULL
```

```
);
```

```
CREATE TABLE guest(  
    guestno CHAR(4) NOT NULL,  
    guestname VARCHAR(20) NOT NULL,  
    guestaddress VARCHAR(50) NOT NULL  
);
```

## 2. Insert records into each of these tables.

```
INSERT INTO hotel VALUES ('H111', 'Grosvenor', 'London');  
INSERT INTO room VALUES ('1', 'H111', 'Single', 72.00);  
INSERT INTO guest VALUES ('G111', 'John Smith', 'London');  
INSERT INTO booking VALUES ('H111', 'G111', '2008-01-01', '2008-01-02', '1');
```

## 3. Update the price of all rooms by 5%.

```
UPDATE room SET price = price*1.05;
```

Note:  $5/100 = 0.05 + 1$  (original price) = 1.05

Can also be:

```
UPDATE room SET price = price + price*0.05;
```

## 4. Create a separate table with the same structure as the Booking table to hold archive records. Using the INSERT statement, copy the records from the Booking table to the archive table relating to bookings before 1st January 2008. Delete all bookings before 1st January 2008 from the Booking table.

```
CREATE TABLE booking_old(  
    hotel_no CHAR(4) NOT NULL,  
    guest_no CHAR(4) NOT NULL,  
    date_from DATETIME NOT NULL,  
    date_to DATETIME NULL,  
    room_no VARCHAR(4) NOT NULL  
);
```

```
INSERT INTO booking_old(  
SELECT * FROM booking  
WHERE date_to < DATE'2008-01-01');
```

```
DELETE FROM booking  
WHERE date_to < DATE'2008-01-01';
```

More practice on SQL queries using MySQL and NetBeans in next tutorial.

---

[<Use MySQL GUI Tool](#) | [Main Java & Gang](#) | [MySQL DML practice 2 Part 1](#) >