**Assignment –1**

**Creating sample tables and inserting values.**

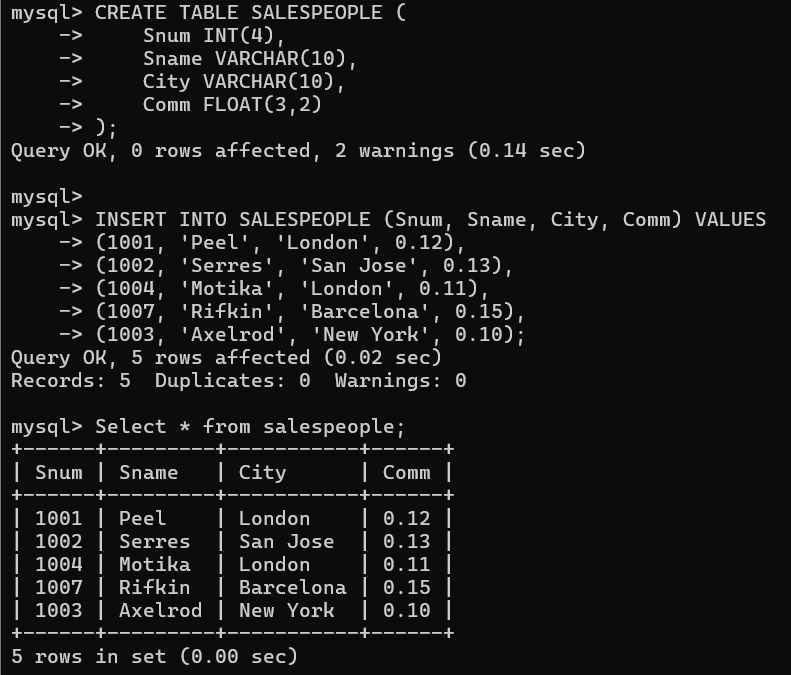
**Create the following tables with the given structures and insert sample data as specified: - A) SALESPEOPLE**

**Snum int(4)**

**Sname varchar(10)**

**City varchar(10)**

**Comm float(3,2)**

****

**B) CUSTOMERS**

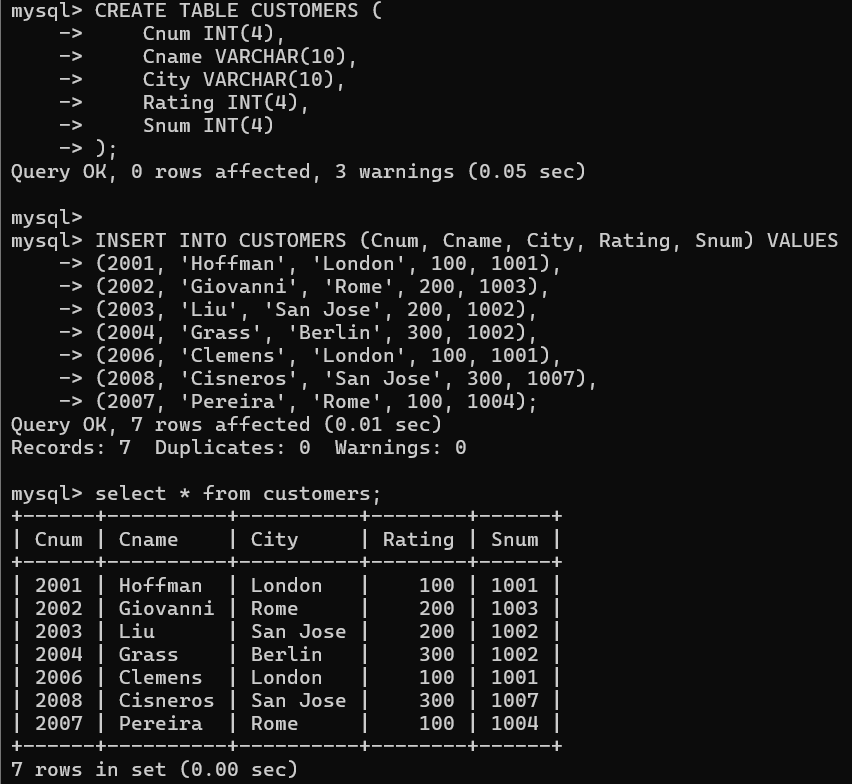
**Cnum int(4)**

**Cname varchar(10)**

**City varchar(10)**

**Rating int(4)**

**Snum int(4)**

****

**C) ORDERS**

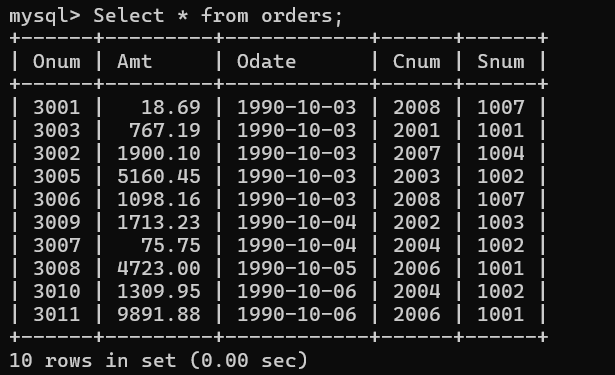
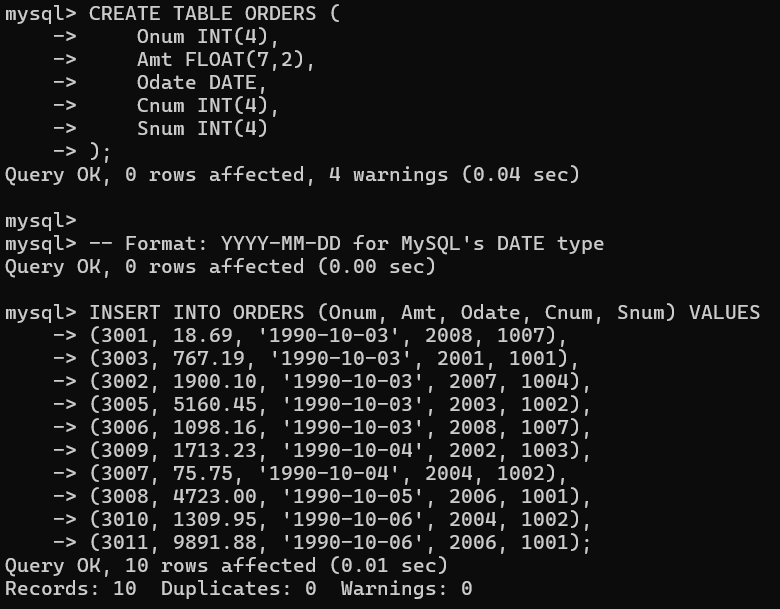
**Onum int(4)**

**Amt float(7,2)**

**Odate date**

**Cnum int(4)**

**Snum int(4)**

****

**Q. '1995-01-15' or '95-01-15'**

**Why is 1970 the cut-off year?**

The year **1970** is often used as a "cut-off" point in **computing systems** because it is the **start of the Unix epoch**. Here's why:

**\* Unix Epoch (January 1, 1970)**: The Unix operating system, along with many other systems that adopt similar time-keeping standards, uses **January 1, 1970** as the base point (epoch) for time calculations. This is the starting point where time is measured in **seconds** since that date.

**Date Representation**: Dates and times are often stored as the number of seconds that have passed since the Unix epoch.

**Range of Representable Years**: The Unix timestamp system originally allowed for the representation of dates from **January 1, 1970** onwards (with a **32-bit system**). This means that the system was able to represent **dates from 1970 onward** and represented them as positive integers.

**Why 1970?** The choice of 1970 as the starting point was somewhat arbitrary but has become standard due to the widespread adoption of Unix and similar systems. It marks a significant point in computing history and serves as a convenient starting point for time calculations.

\*For older systems or systems that do not use Unix-style timekeeping, you might encounter different "cut-off" dates, but **1970** remains a critical reference point in computing and databases.

**Assignment –2**

**Introducing Relational Databases.**

**1. Which field of the Customers table is the primary key?**

In the CUSTOMERS table, the **primary key** is the **Cnum** field. This is because Cnum represents a unique identifier for each customer. Usually, primary keys are set on fields that contain unique, non-null values for each record.

**2.What is the 4th column of the Customers table?**

The 4th column in the CUSTOMERS table is **Rating**.

**3.What is another word for row? For column?**

 **Row**: Another word for "row" in a database table is **"record"** or **"tuple"**.

 **Column**: Another word for "column" is **"field"** or **"attribute"**.

**4.Why isn’t it possible to see the first five rows of a table?**

There could be a few reasons why you might not be able to see the first five rows of a table:

 **Data Query**: If you’re trying to **select the first five rows**, make sure your query is correct. For example, in MySQL, you can limit the number of rows returned by using LIMIT:

**SELECT \* FROM CUSTOMERS LIMIT 5;**

 **No Rows**: If the table doesn't have **at least five rows**, it wouldn't be possible to view them.

 **Ordering**: If there's no **order** specified (like sorting by a specific column), the "first five rows" could vary depending on how the database engine stores them.

 **Permissions**: You might not have the **necessary permissions** to view all rows, depending on your user’s access rights.

**Assignment –3**

**Overview of SQL.**

**1. Does ANSI recognize the data type DATE?**

Yes, **ANSI SQL** (American National Standards Institute) recognizes the **DATE** data type. In ANSI SQL, the DATE type is used to store date values in the format YYYY-MM-DD.

In SQL standards:

* **DATE** stores only the date (year, month, day).
* There are other types like **TIME** (to store time) and **DATETIME** (to store both date and time), but DATE is part of the ANSI SQL standard.

**2.Which subdivision of SQL is used to insert values in tables?**

The subdivision of SQL used to insert values into tables is the **Data Manipulation Language (DML)**.

* **DML** includes SQL statements that manage the data stored within the database.
* The **INSERT** statement is part of DML and is used to insert new rows (records) into a table.

For example:

INSERT INTO CUSTOMERS (Cnum, Cname, City, Rating, Snum)

VALUES (2001, 'Hoffman', 'London', 100, 1001);

Other DML commands include:

* **SELECT**: to retrieve data.
* **UPDATE**: to modify existing data.
* **DELETE**: to remove data.

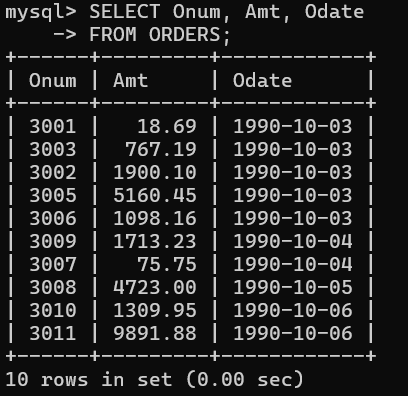
**Assignment –4**

**Retrieving Information from Tables.**

**1) Write a select command that produces the order number, amount, and date for all rows in the Orders table.**

SELECT Onum, Amt, Odate

FROM ORDERS;

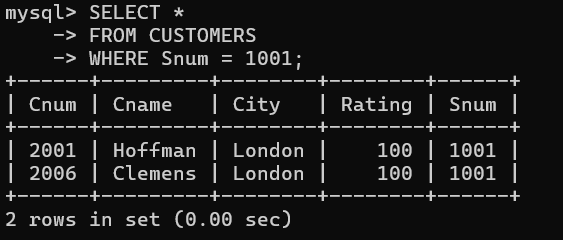


**2) Write a query that produces all rows from the Customers table for which the salesperson’s number is 1001.**

SELECT \*

FROM CUSTOMERS

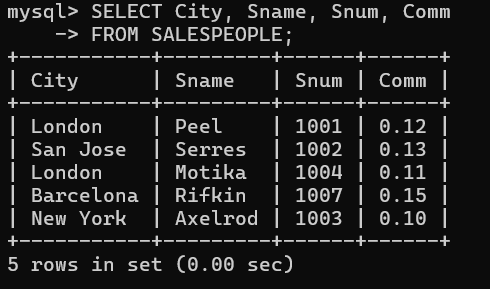
WHERE Snum = 1001;



**3) Write a query that displays the Salespeople table with the columns in the following order: city, sname, snum, comm.**

SELECT City, Sname, Snum, Comm

FROM SALESPEOPLE;

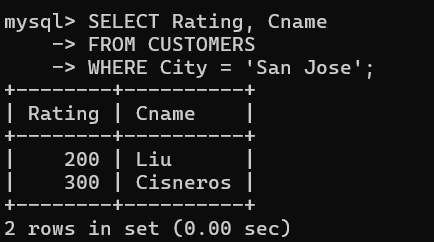


**4) Write a select command that produces the rating followed by the name of each customer in San Jose.**

SELECT Rating, Cname

FROM CUSTOMERS

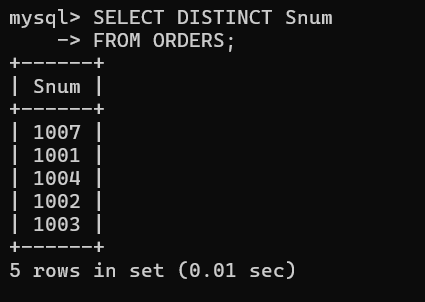
WHERE City = 'San Jose';

****

**5) Write a query that will produce the snum values of all salespeople (suppress the duplicates) with orders in the Orders table.**

SELECT DISTINCT Snum

FROM ORDERS;



**Assignment –5**

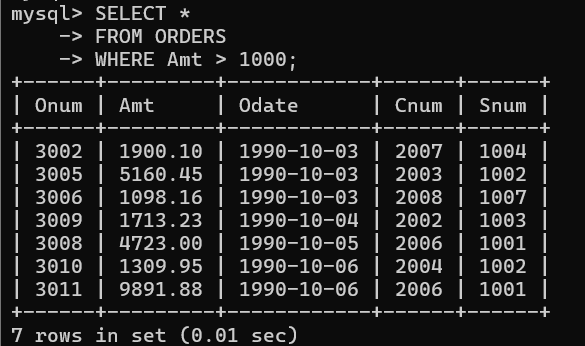
**Relational and Logical Operators.**

**1) Write a query that will give you all orders for more than Rs. 1,000.**

SELECT \*

FROM ORDERS

WHERE Amt > 1000;

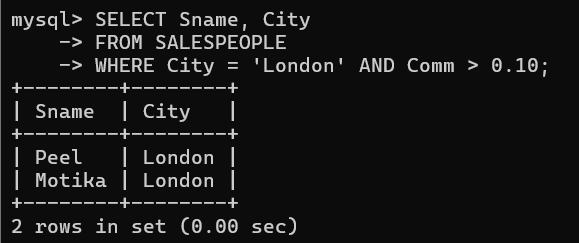


**2) Write a query that will give you the names and cities of all salespeople in London with a commission above .10.**

SELECT Sname, City

FROM SALESPEOPLE

WHERE City = 'London' AND Comm > 0.10;

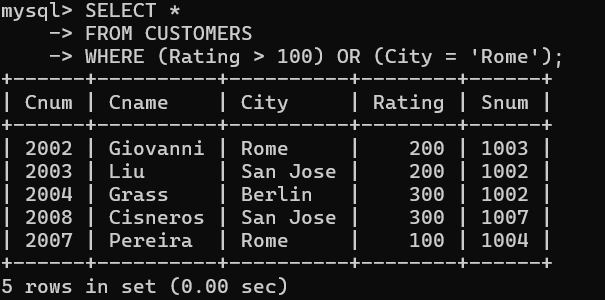


**3) Write a query on the Customers table whose output will exclude all customers with a rating <= 100, unless they are located in Rome.**

SELECT \*

FROM CUSTOMERS

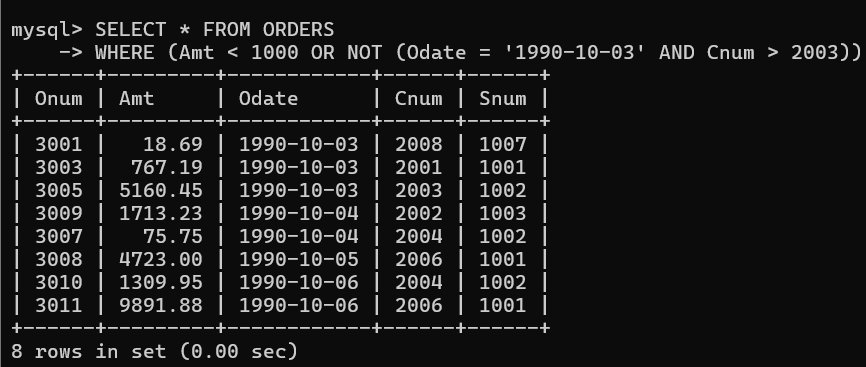
WHERE (Rating > 100) OR (City = 'Rome');



**4) What will be the output from the following query?**

**Select \* from Orders**

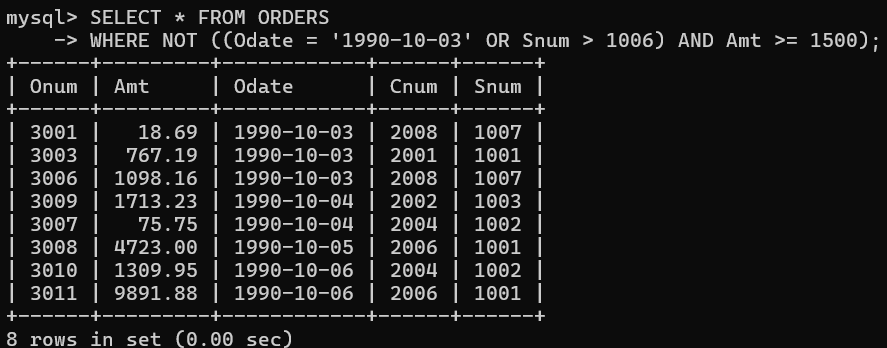
**where (amt < 1000 OR NOT (odate = ‘1990-10-03’ AND cnum > 2003));**

****

**5) What will be the output of the following query?**

**Select \* from Orders**

**where NOT ((odate = ‘1990-10-03’ OR snum >1006) AND amt >= 1500);**

****

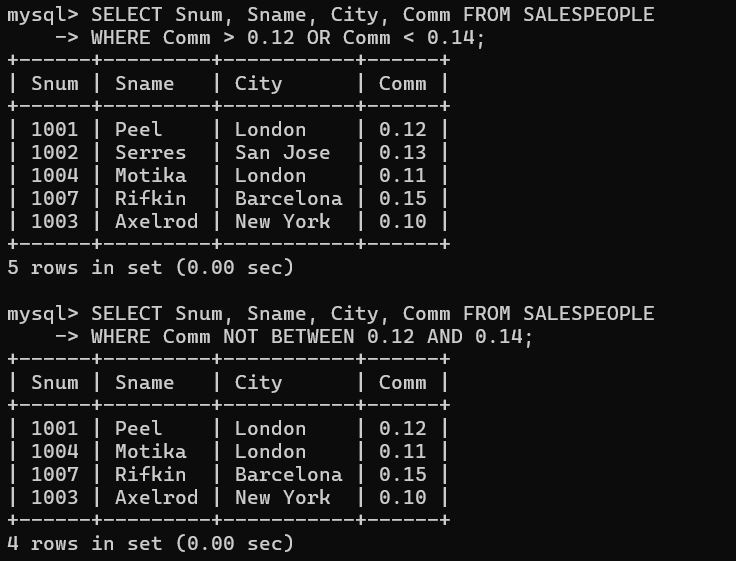
**6) What is a simpler way to write this query?**

**Select snum, sname, city, comm From Salespeople**

**where (comm > .12 OR comm <.14);**

SELECT Snum, Sname, City, Comm FROM SALESPEOPLE

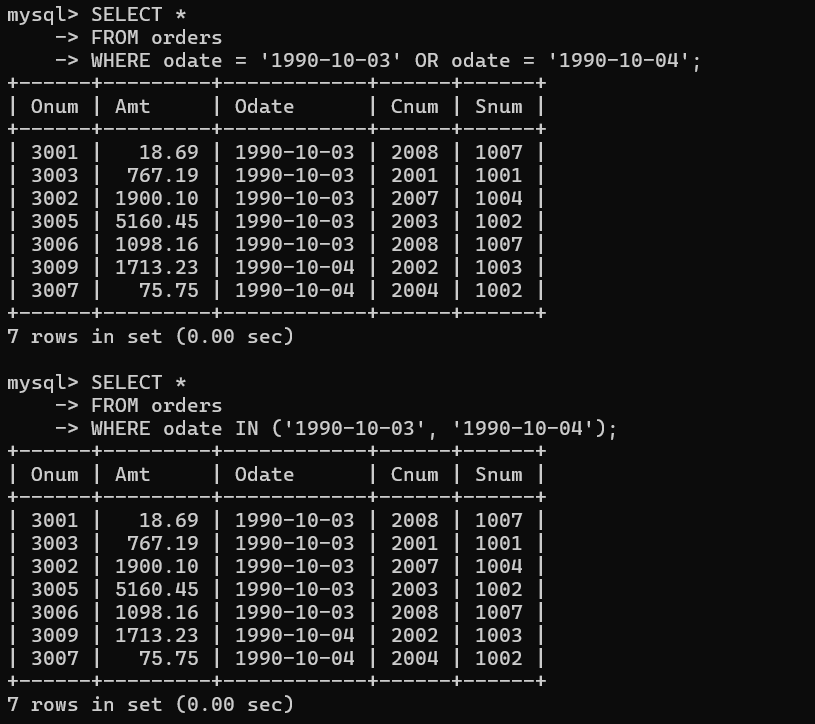
WHERE Comm NOT BETWEEN 0.12 AND 0.14;



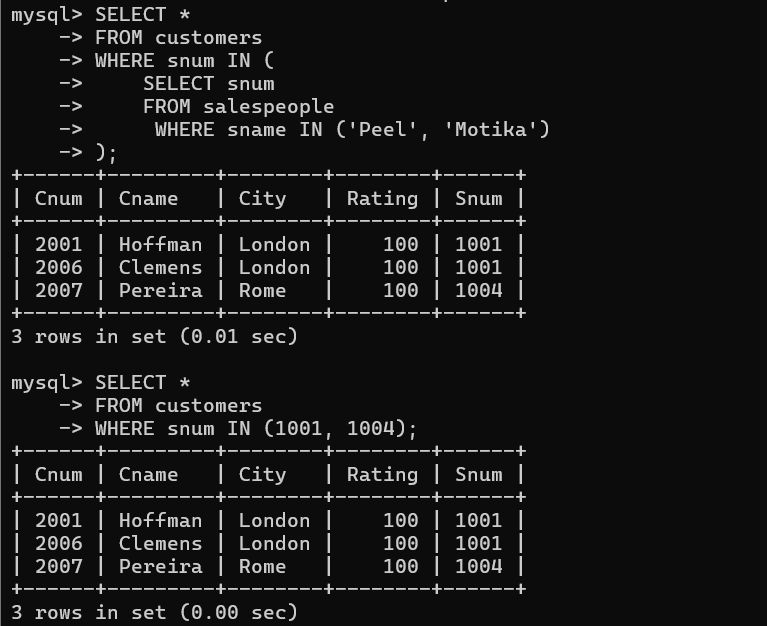
**Assignment –6**

**Using Special Operators in Conditions.**

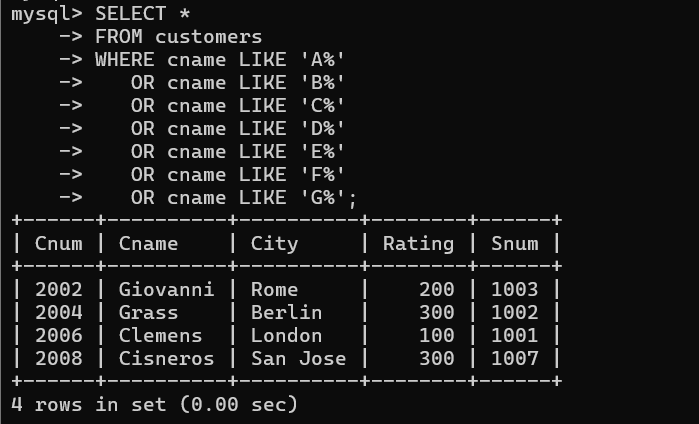
**1) Write two different queries that would produce all orders taken on October 3rd or 4th, 1990.**

****

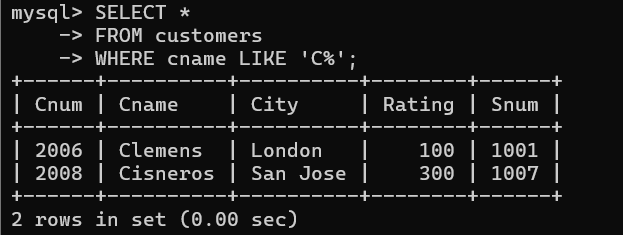
**2) Write a query that selects all of the customers serviced by Peel or Motika. (Hint: the snum field relates the two tables to one another).**

****

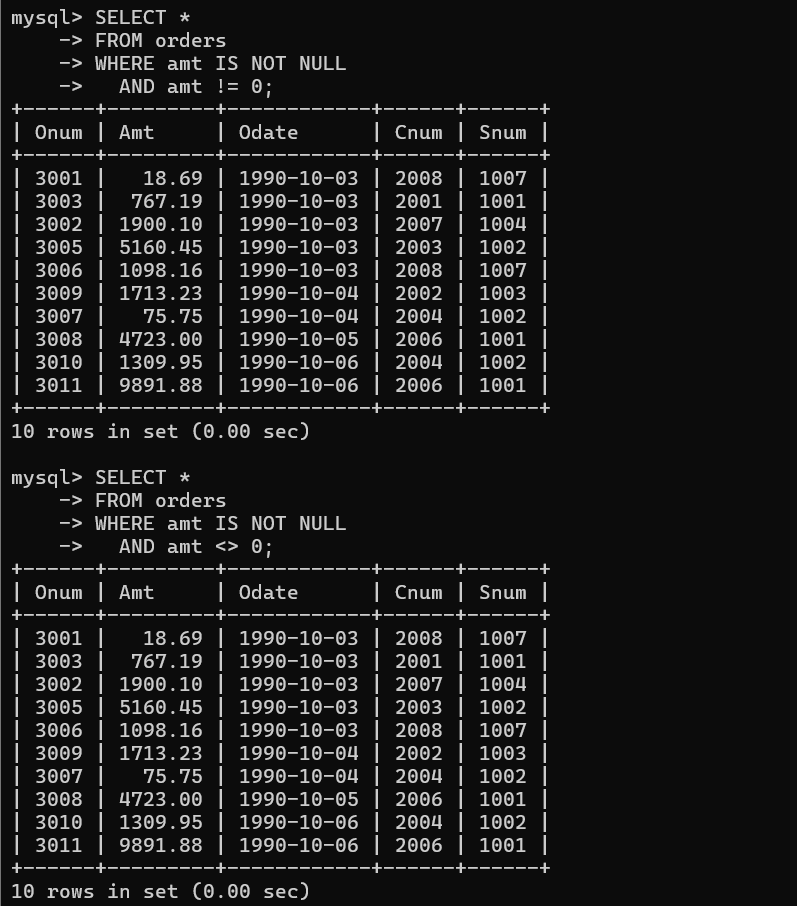
**3) Write a query that will produce all the customers whose names begin with a letter from ‘A’ to ‘G’.**

****

**4) Write a query that selects all customers whose names begin with the letter ‘C’.**

****

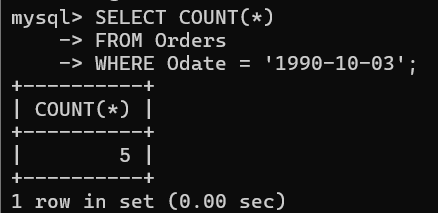
**5) Write a query that selects all orders except those with zeroes or NULLs in the amt field.**

****

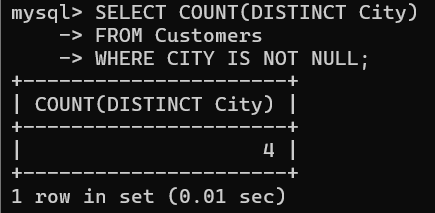
**Assignment –7**

**Summarizing Data with Aggregate Functions.**

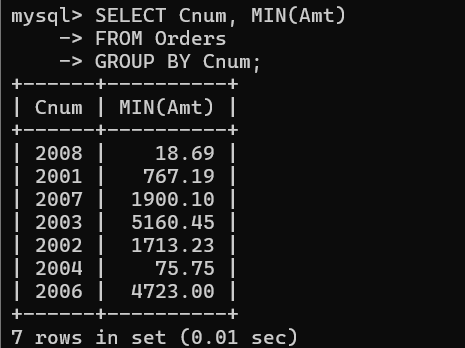
**1) Write a query that counts all orders for October 3.**

****

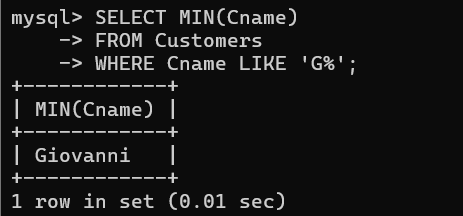
**2) Write a query that counts the number of different non-NULL city values in the Customers table.**

****

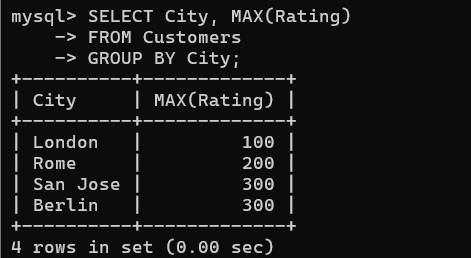
**3) Write a query that selects each customer’s smallest order.**

****

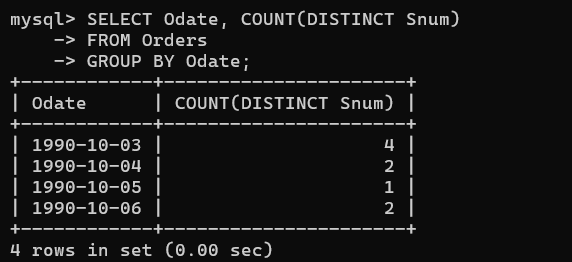
**4) Write a query that selects the first customer, in alphabetical order, whose name begins with G.**

****

**5) Write a query that selects the highest rating in each city.**

****

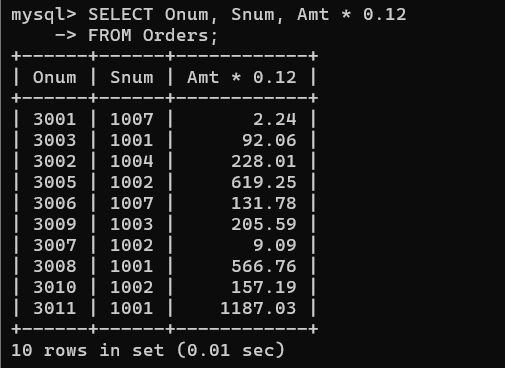
**6) Write a query that counts the number of salespeople registering orders for each day. (If a salesperson has more than one order on a given day, he or she should be counted only once.).**

****

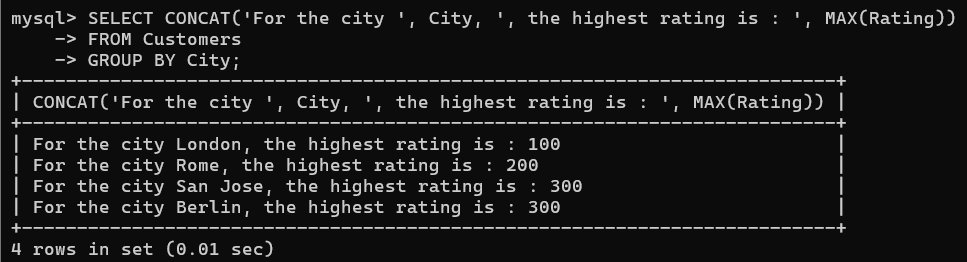
**Assignment –8**

**Formatting Query output.**

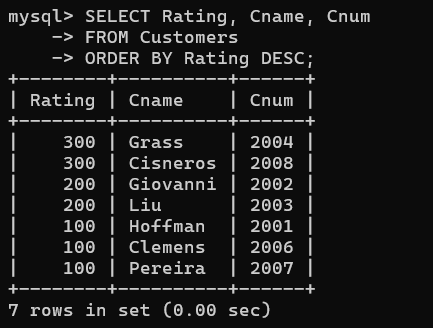
**1) Assume each salesperson has a 12% commission. Write a query on the orders table that will produce the order number, the salesperson number, and the amount of the salesperson’s commission for that order.**

****

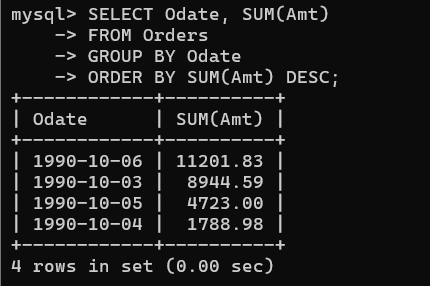
**2) Write a query on the Customers table that will find the highest rating in each city. Put the output in this form: For the city (city), the highest rating is : (rating).**

****

**3) Write a query that lists customers in descending order of rating. Output the rating field first, followed by the customer’s name and number.**

****

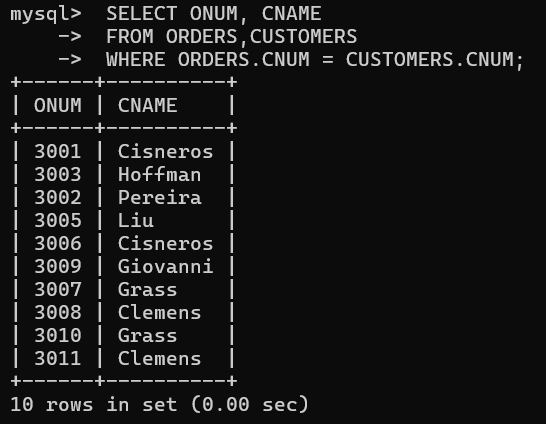
**4) Write a query that totals the orders for each day and places the results in descending order.**

****

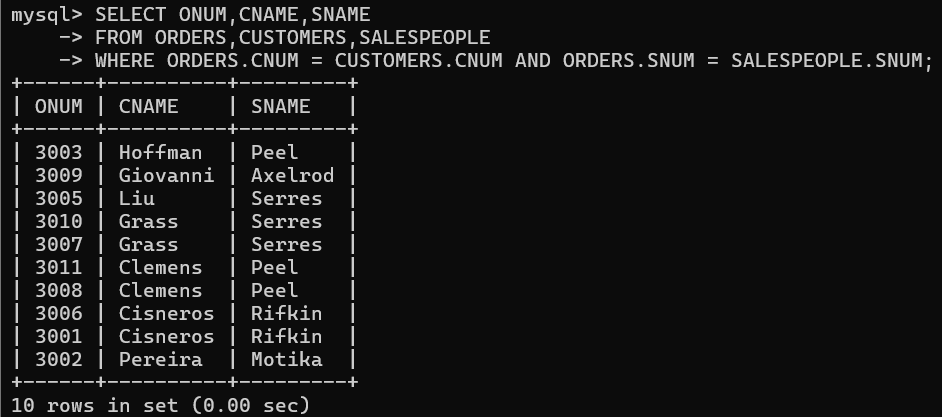
**Assignment – 9**

**Querying Multiple Tables at Once.**

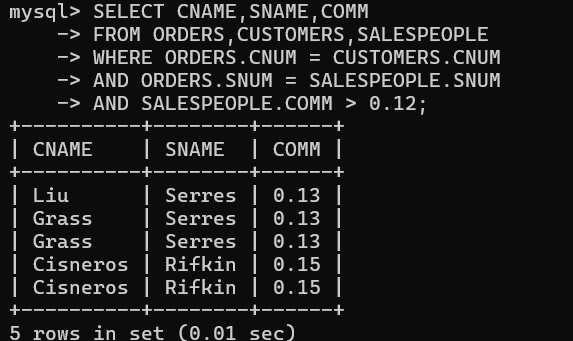
**1) Write a query that lists each order number followed by the name of the customer who made the order.**

****

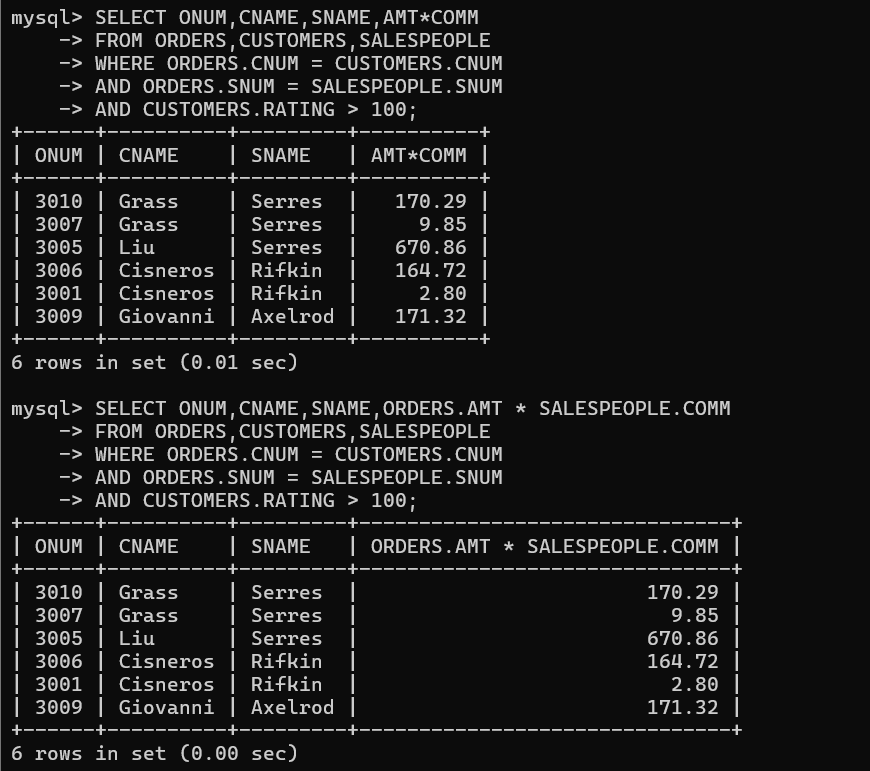
**2) Write a query that gives the names of both the salesperson and the customer for each order along with the order number.**

****

**3) Write a query that produces all customers serviced by salespeople with a commission above 12%. Output the customer’s name, the salesperson’s name, and the salesperson’s rate of commission.**

****

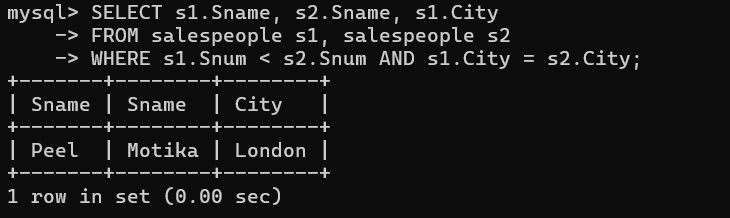
**4) Write a query that calculates the amount of the salesperson’s commission on each order by a customer with a rating above 100.**

****

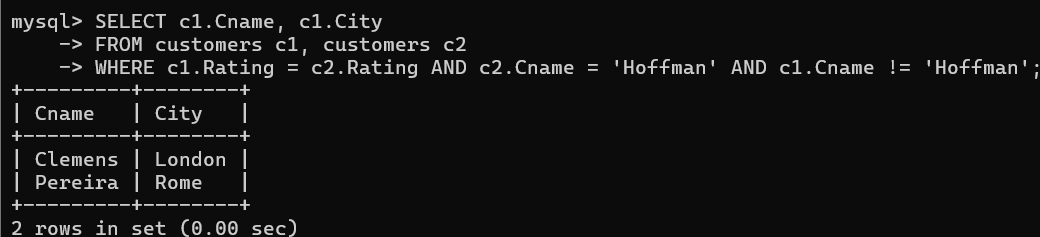
**Assignment – 10**

**Joining a Table to Itself.**

**1) Write a query that produces all pairs of salespeople who are living in the same city. Exclude combinations of salespeople with themselves as well as duplicate rows with the order reversed.**

****

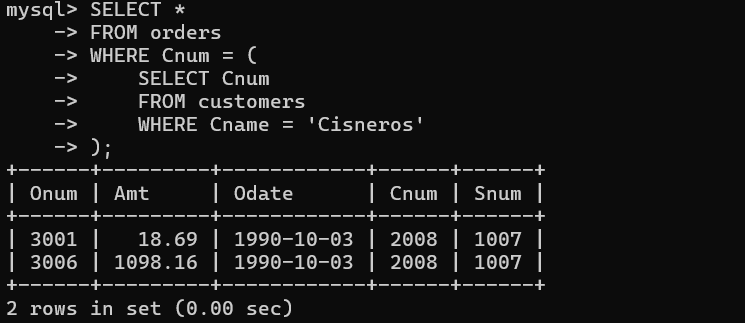
**2) Write a query that produces the names and cities of all customers with the same rating as Hoffman.**

****

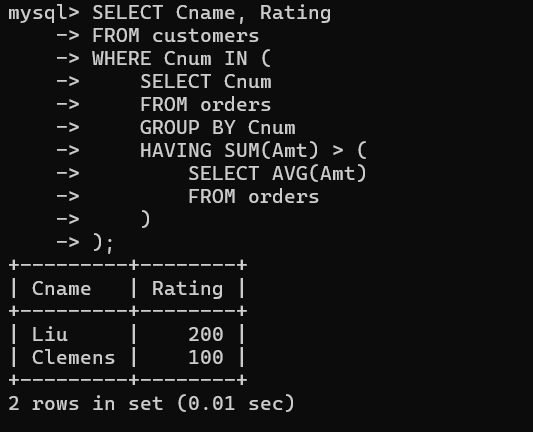
**Assignment – 11**

**Subqueries.**

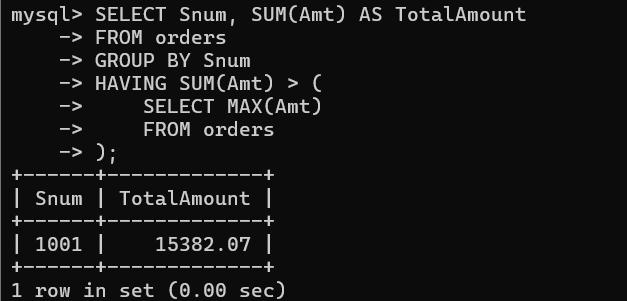
**1) Write a query that uses a subquery to obtain all orders for the customer named Cisneros. Assume you do not know his customer number (cnum).**

****

**2) Write a query that produces the names and ratings of all customers who have above-average orders.**

****

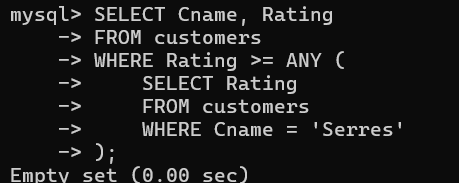
**3) Write a query that selects the total amount in orders for each salesperson for whom this total is greater than the amount of the largest order in the table.**

****

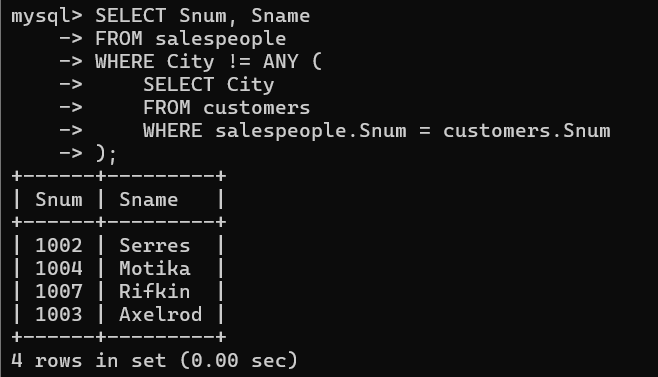
**Assignment – 12**

**Using the operators IN, ANY, and ALL.**

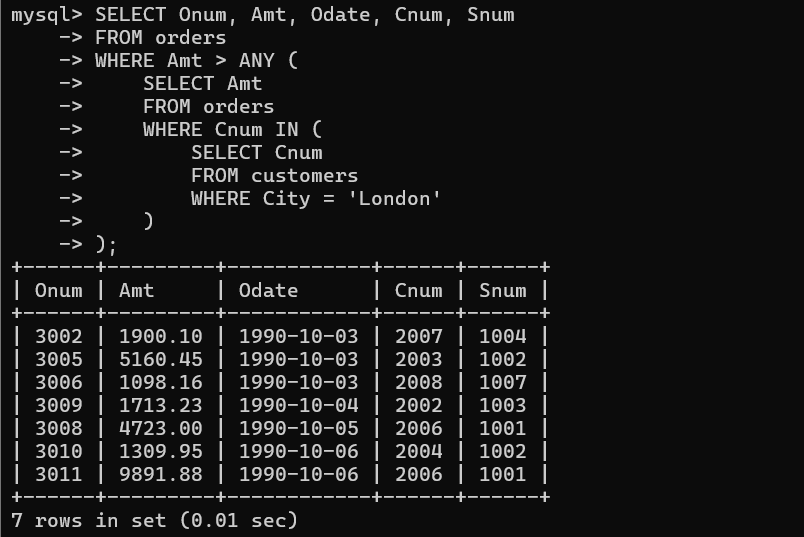
**1) Write a query that selects all customers whose ratings are equal to or greater than ANY of Serres’.**

****

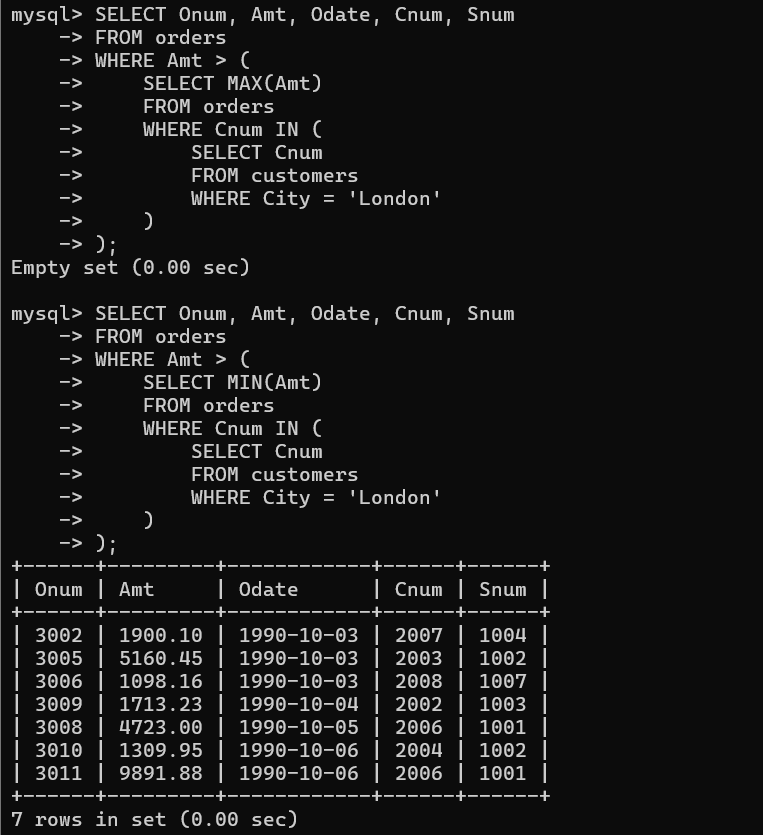
**2) Write a query using ANY or ALL that will find all salespeople who have no customers located in their city.**

****

**3) Write a query that selects all orders for amounts greater than any for the customers in London.**

****

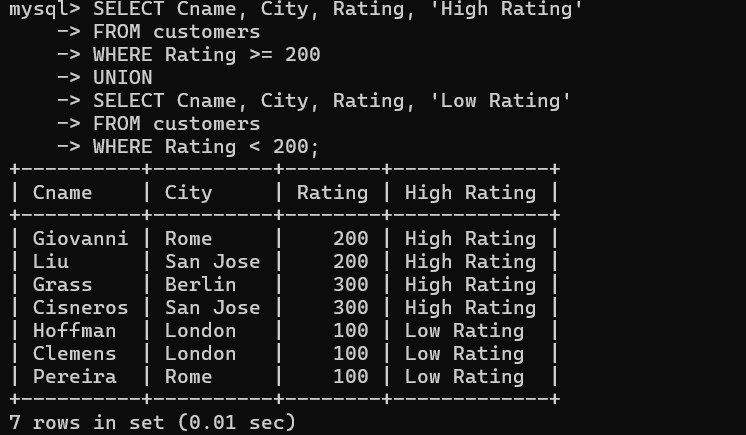
**4) Write the above query using MIN or MAX.**

****

**Assignment – 13**

**Using the UNION clause.**

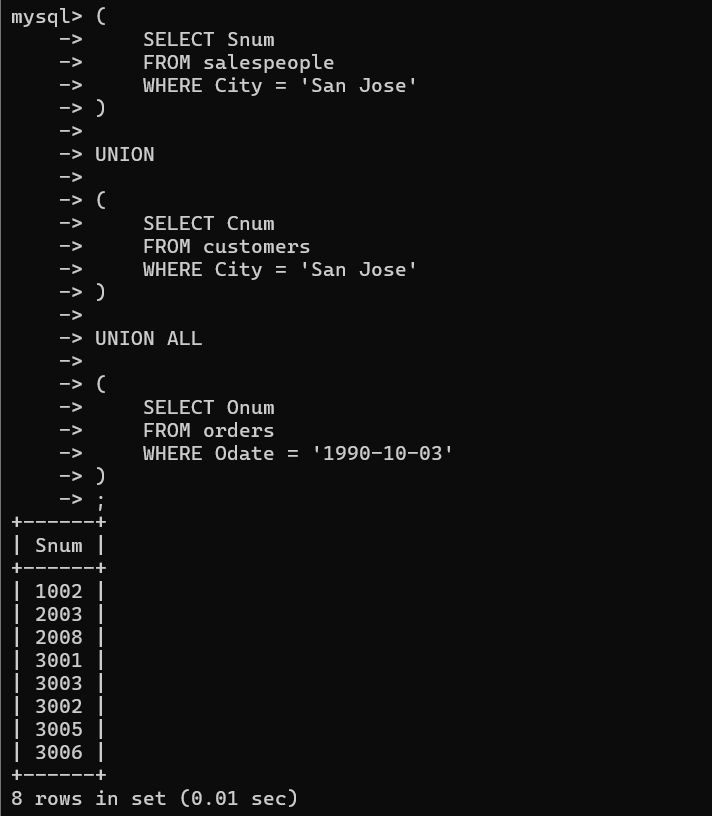
**1) Create a union of two queries that shows the names, cities, and ratings of all customers. Those with rating of 200 or greater will also have the words “High Rating”, while the others will have the words “Low Rating”.**

****

**2) Write a command that produces the name and number of each salesperson and each customer with more than one current order. Put the results in alphabetical order.**

****

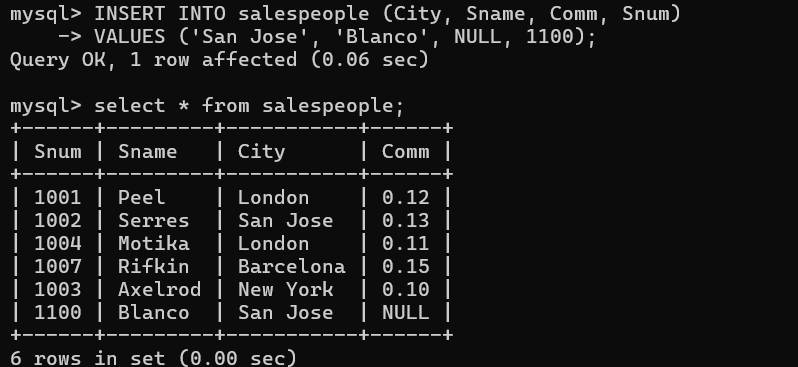
**3) Form a union of three queries. Have the first select the snums of all salespeople in San Jose; the second, the cnums of all customers in San Jose; and the third the onums of all orders on October 3. Retain duplicates between the last two queries but eliminate any redundancies between either of them and the first. (Note: in the sample tables as given, there would be no such redundancy. This is besides the point.)**

****

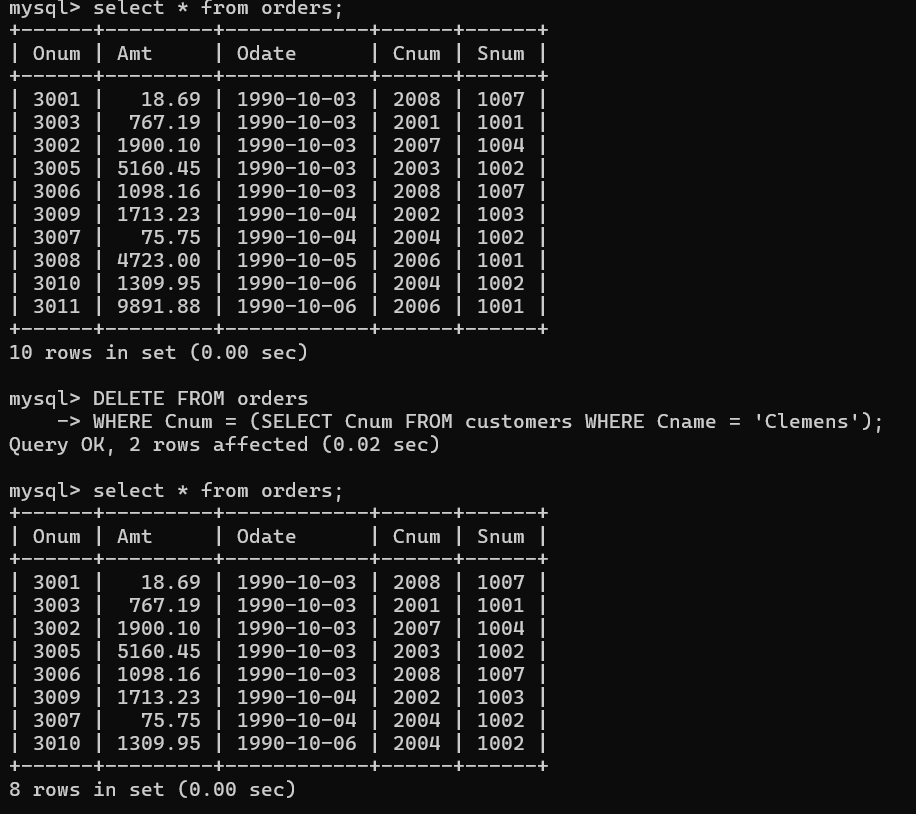
**Assignment – 14**

**Entering, Deleting, and Changing Field Values.**

**1) Write a command that puts the following values, in their given order, into the salespeople table: city – San Jose, name – Blanco, comm – NULL, cnum – 1100.**

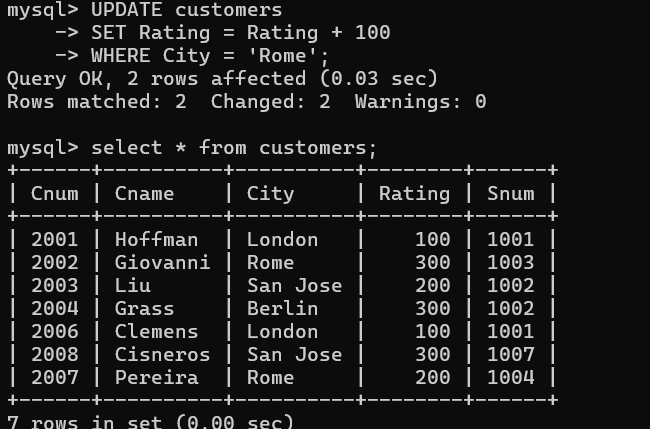
****

**2) Write a command that removes all orders from customer Clemens from the Orders table.**

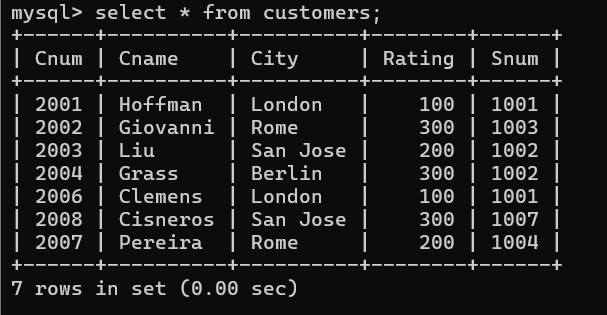
****

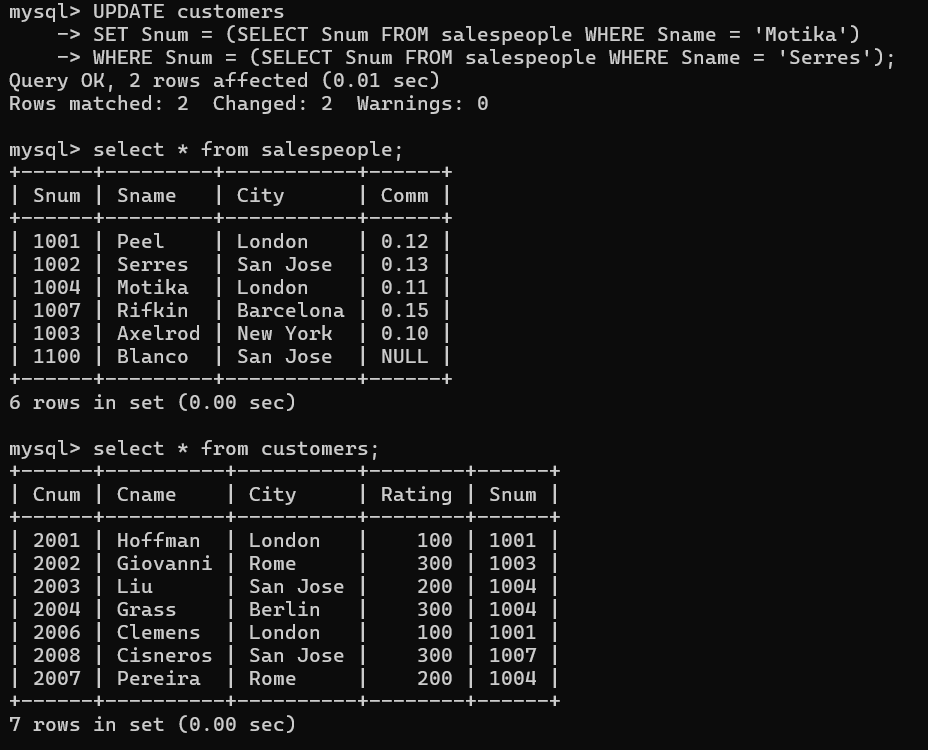
**3) Write a command that increases the rating of all customers in Rome by 100.**

****

****

**4) Salesperson Serres has left the company. Assign her customers to Motika.**

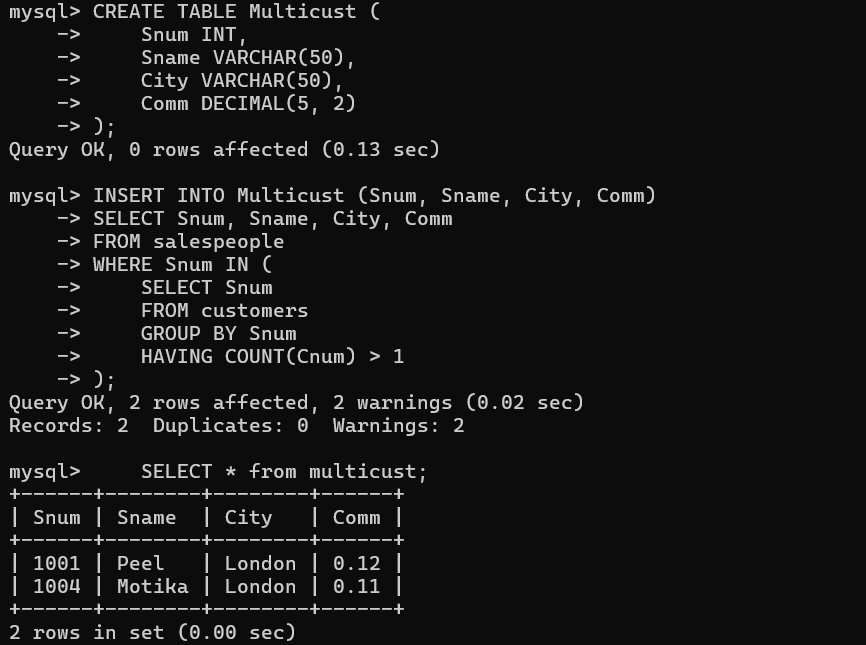


****

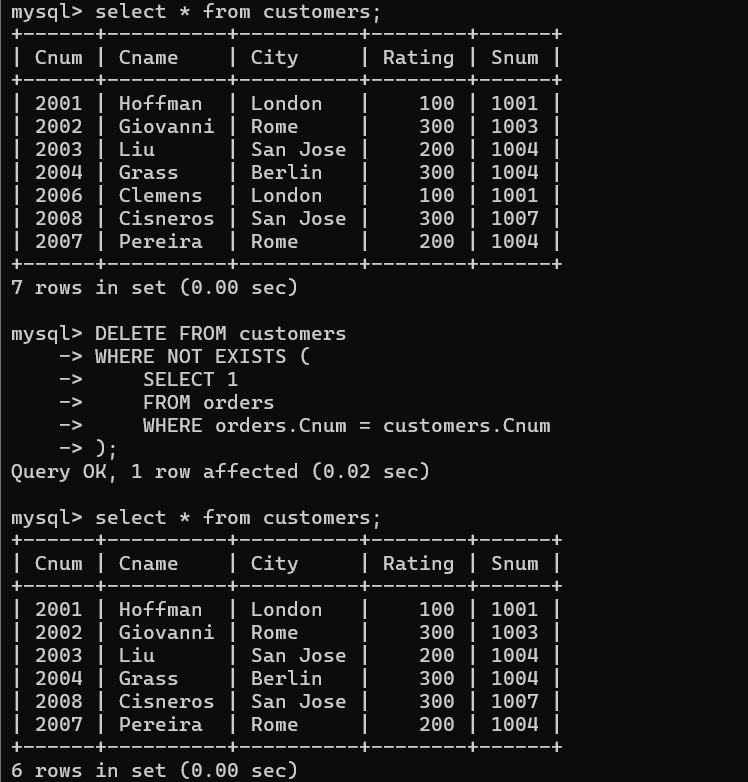
**Assignment – 15**

**Using Subqueries with DML Commands.**

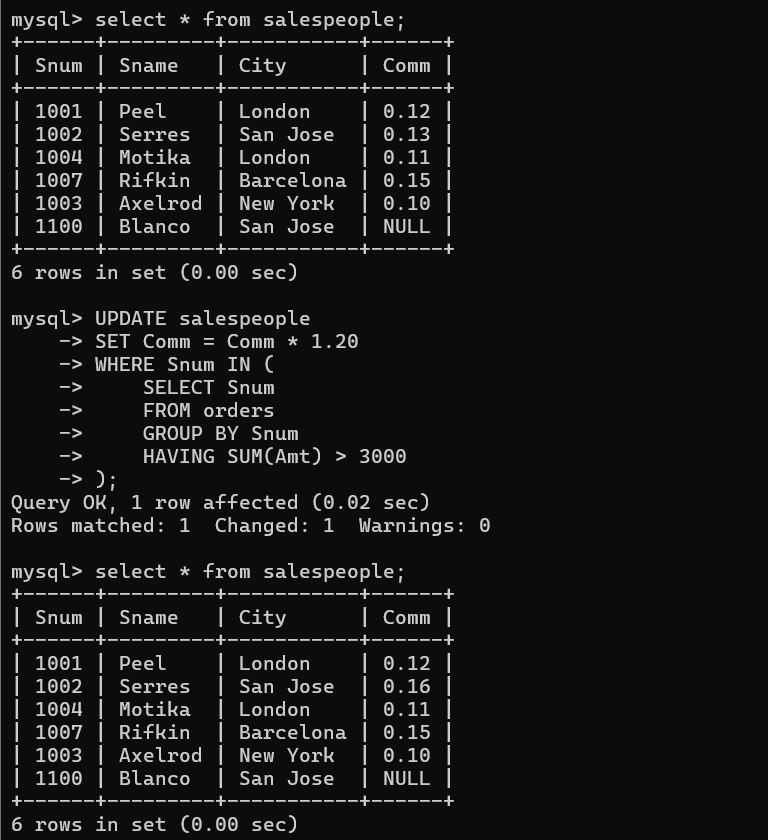
1. **Assume there is a table called Multicust, with all of the same column definitions as Salespeople. Write a command that inserts all salespeople with more than one customer into this table.**



**2) Write a command that deletes all customers with no current orders.**

****

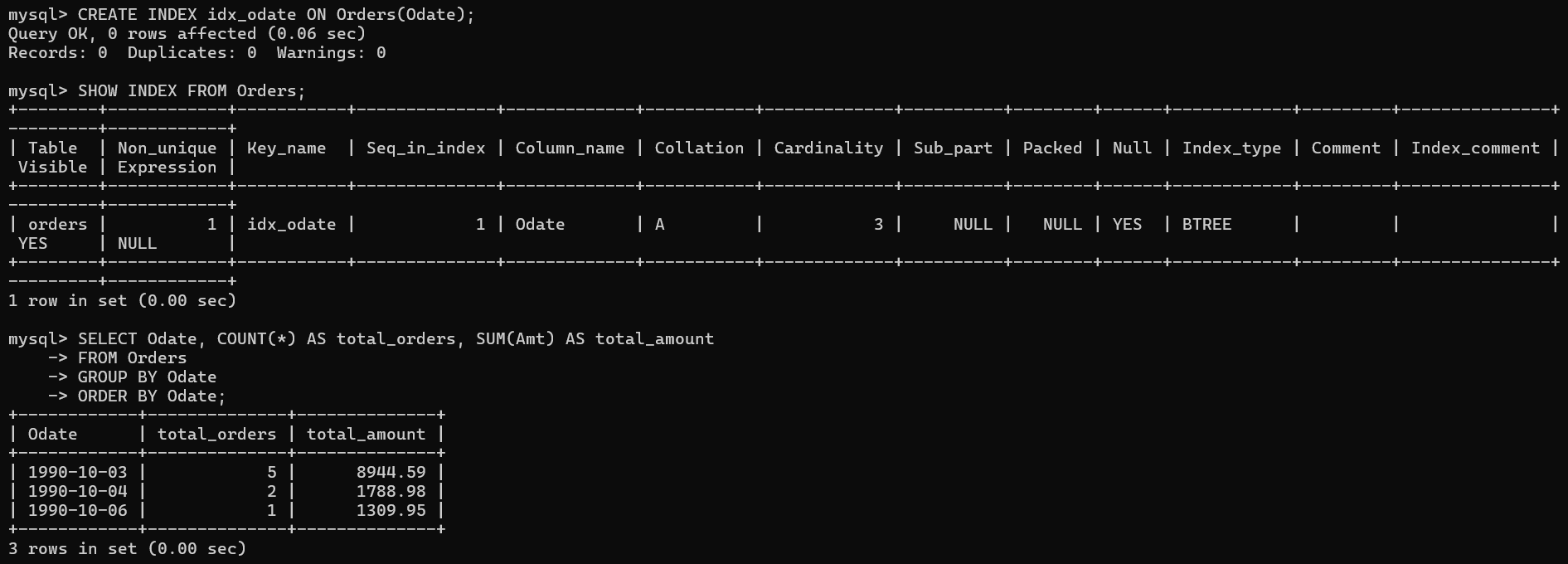
**3) Write a command that increases by twenty percent the commissions of all salespeople with total orders above Rs. 3,000.**

****

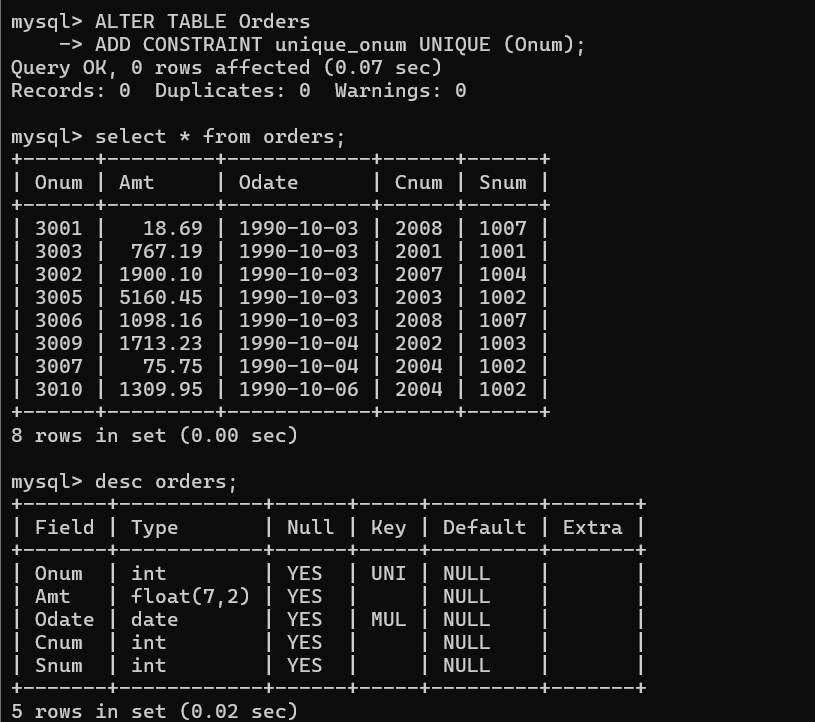
**Assignment – 16**

**Creating Tables and Indexes.**

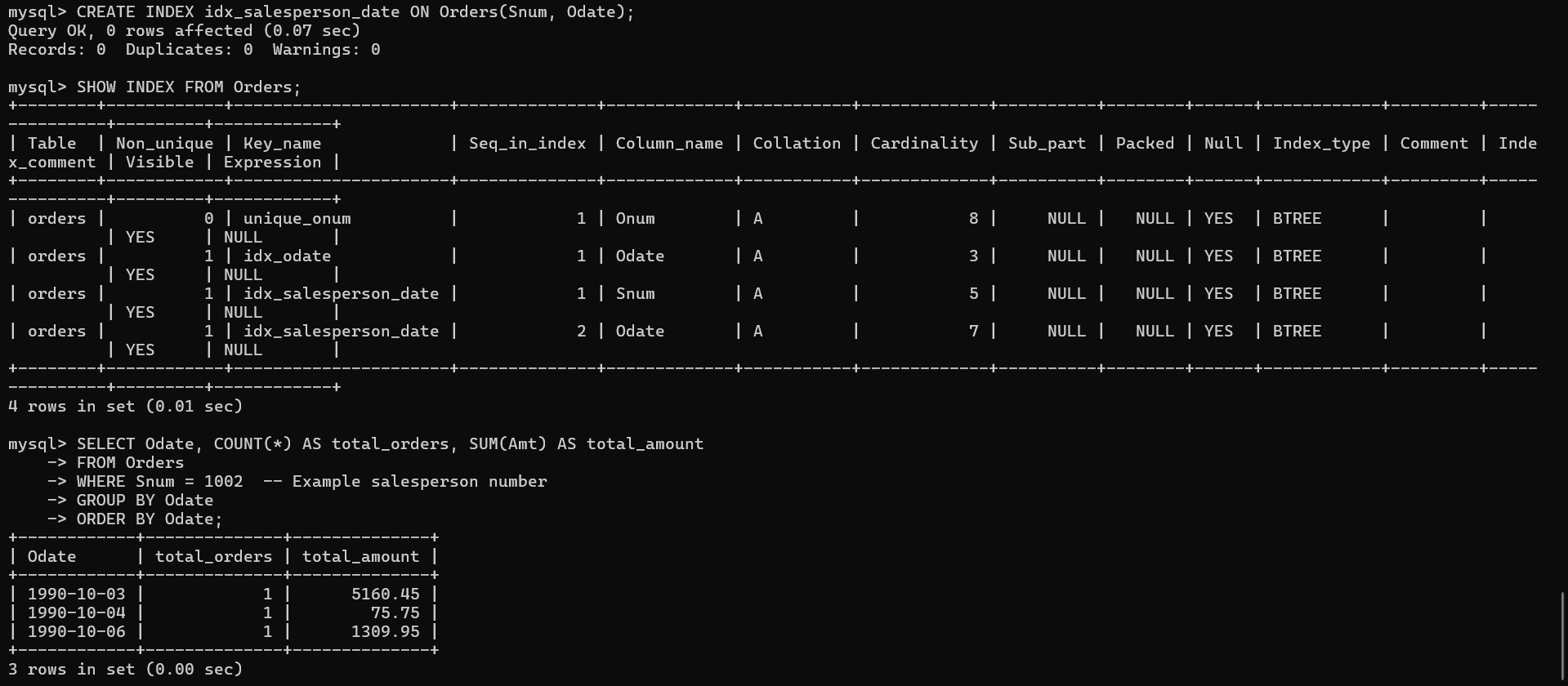
**1) Write a command that will enable a user to pull orders grouped by date out of the Orders table quickly.**

****

**2) If the Orders table has already been created, how can you force the onum field to be unique (assume all current values are unique)?**

****

**3) Create an index that would permit each salesperson to retrieve his or her orders grouped by date quickly.**

****