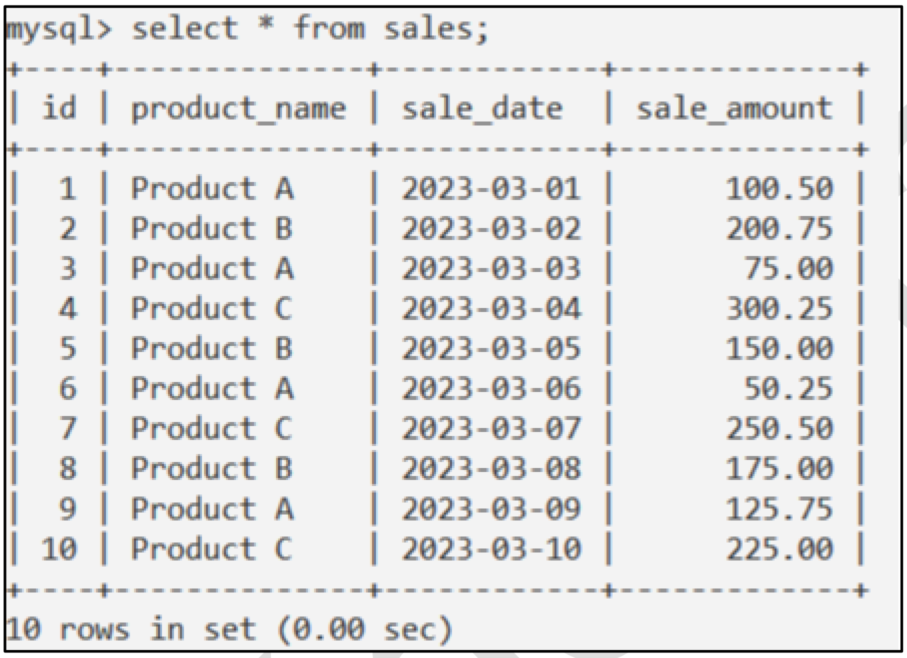
“Exp 10: Write down SQL by using Aggregate, Date & String functions.”

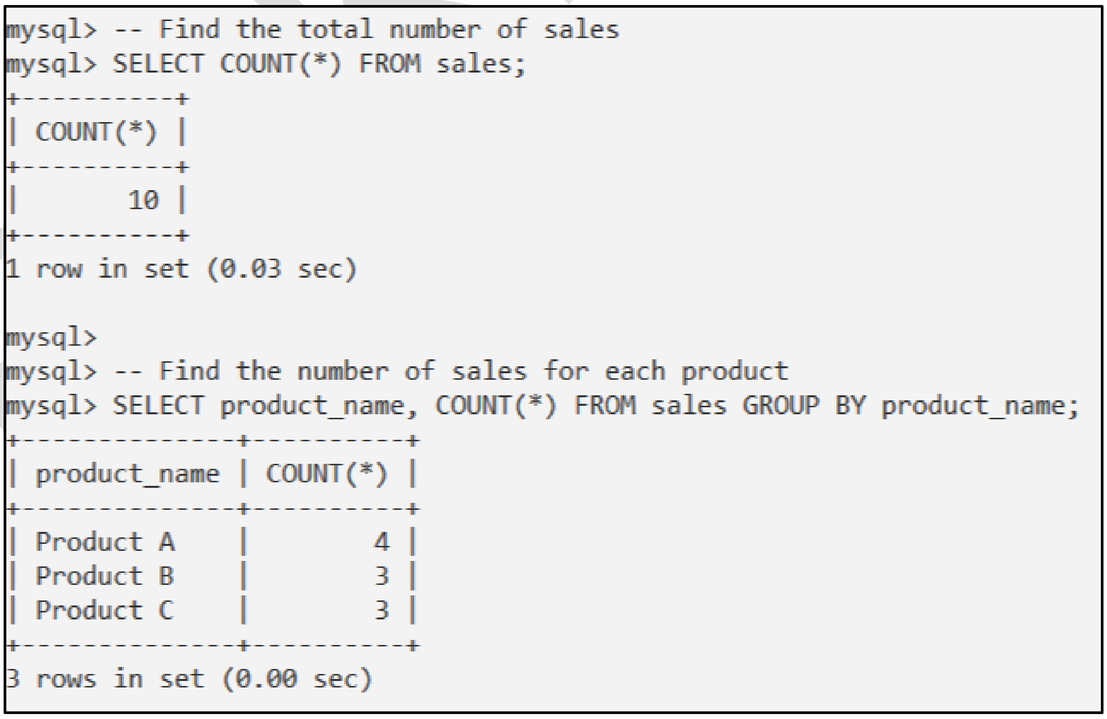
A] Aggregate functions.

1] Reference table : -



```
mysql> select * from sales;
+----+--------------+------------+-------------+
| id | product_name | sale_date  | sale_amount |
+----+--------------+------------+-------------+
|  1 | Product A    | 2023-03-01 |      100.50 |
|  2 | Product B    | 2023-03-02 |      200.75 |
|  3 | Product A    | 2023-03-03 |       75.00 |
|  4 | Product C    | 2023-03-04 |      300.25 |
|  5 | Product B    | 2023-03-05 |      150.00 |
|  6 | Product A    | 2023-03-06 |       50.25 |
|  7 | Product C    | 2023-03-07 |      250.50 |
|  8 | Product B    | 2023-03-08 |      175.00 |
|  9 | Product A    | 2023-03-09 |      125.75 |
| 10 | Product C    | 2023-03-10 |      225.00 |
+----+--------------+------------+-------------+
10 rows in set (0.00 sec)
```

2] COUNT() : -

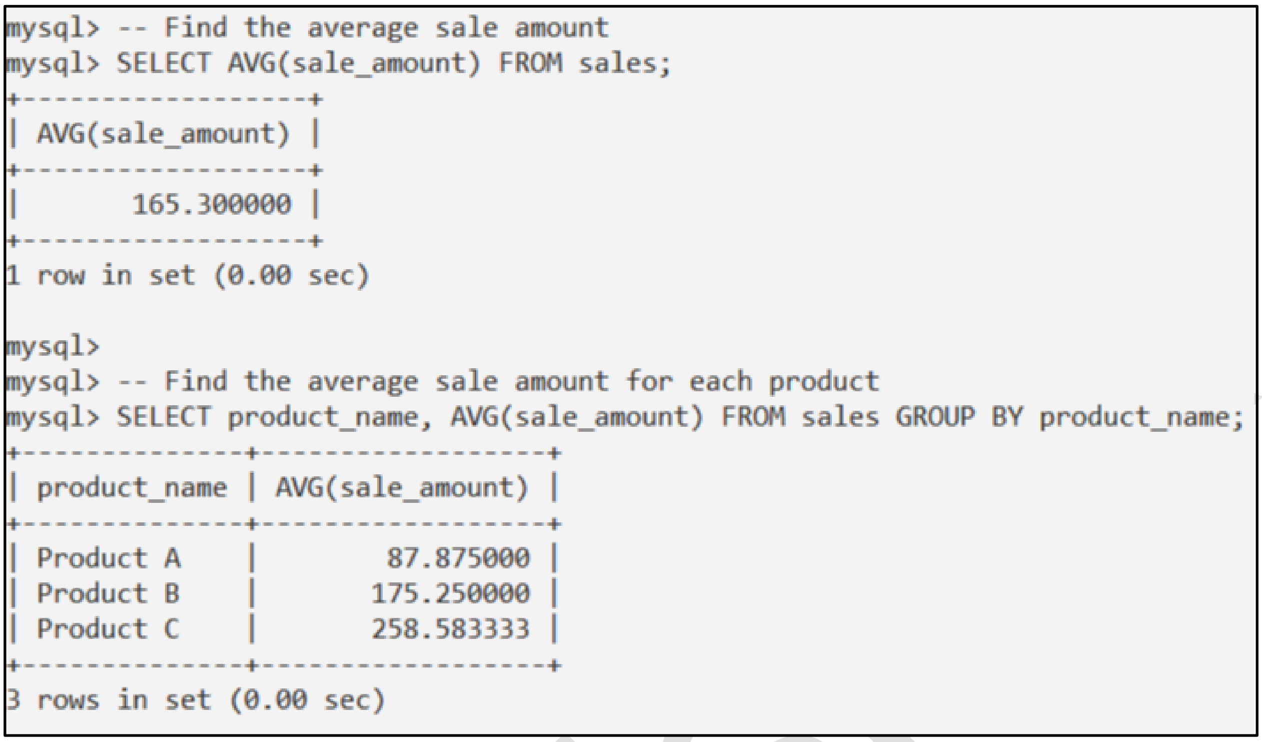


```
mysql> -- Find the total number of sales
mysql> SELECT COUNT(*) FROM sales;
+----------+
| COUNT(*) |
+----------+
|       10 |
+----------+
1 row in set (0.03 sec)

mysql>
mysql> -- Find the number of sales for each product
mysql> SELECT product_name, COUNT(*) FROM sales GROUP BY product_name;
+--------------+----------+
| product_name | COUNT(*) |
+--------------+----------+
| Product A    |        4 |
| Product B    |        3 |
| Product C    |        3 |
+--------------+----------+
3 rows in set (0.00 sec)
```

P.T.O →

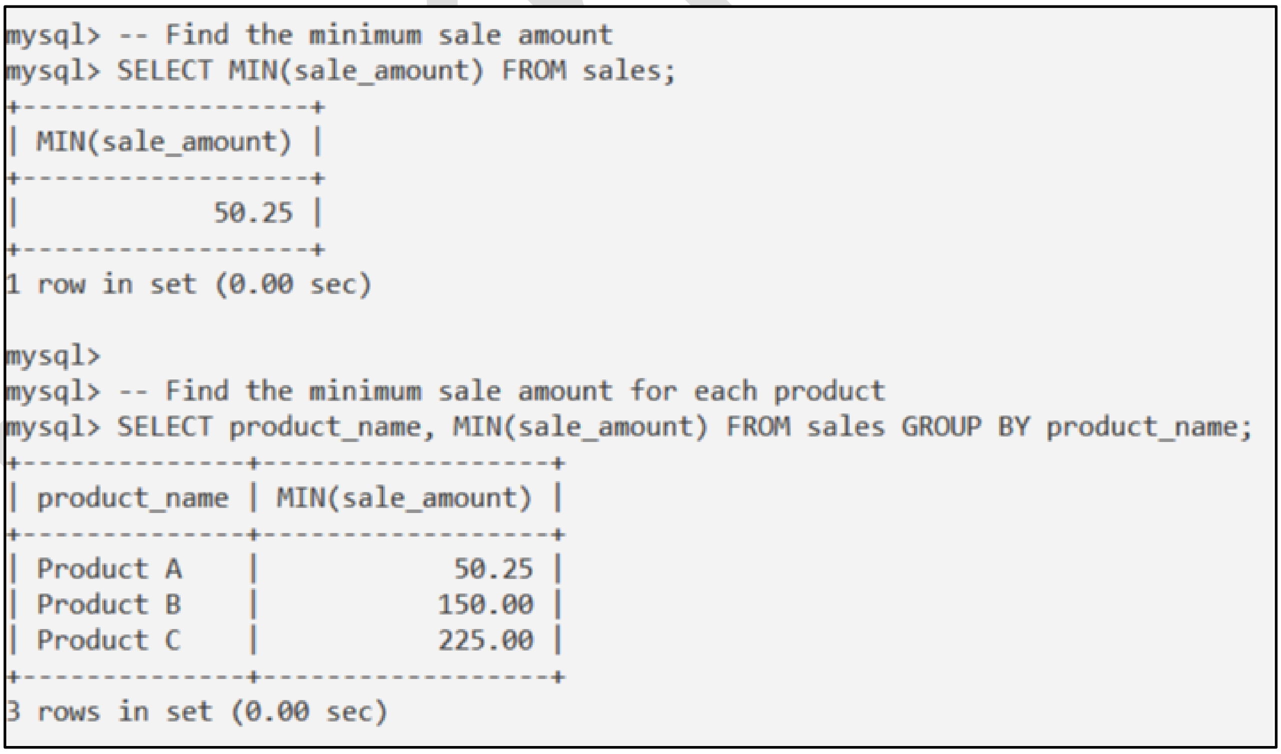
3] AVG() : -



```
mysql> -- Find the average sale amount
mysql> SELECT AVG(sale_amount) FROM sales;
+------------------+
| AVG(sale_amount) |
+------------------+
|       165.300000 |
+------------------+
1 row in set (0.00 sec)

mysql>
mysql> -- Find the average sale amount for each product
mysql> SELECT product_name, AVG(sale_amount) FROM sales GROUP BY product_name;
+--------------+------------------+
| product_name | AVG(sale_amount) |
+--------------+------------------+
| Product A    |        87.875000 |
| Product B    |       175.250000 |
| Product C    |       258.583333 |
+--------------+------------------+
3 rows in set (0.00 sec)
```

4] MIN() : -

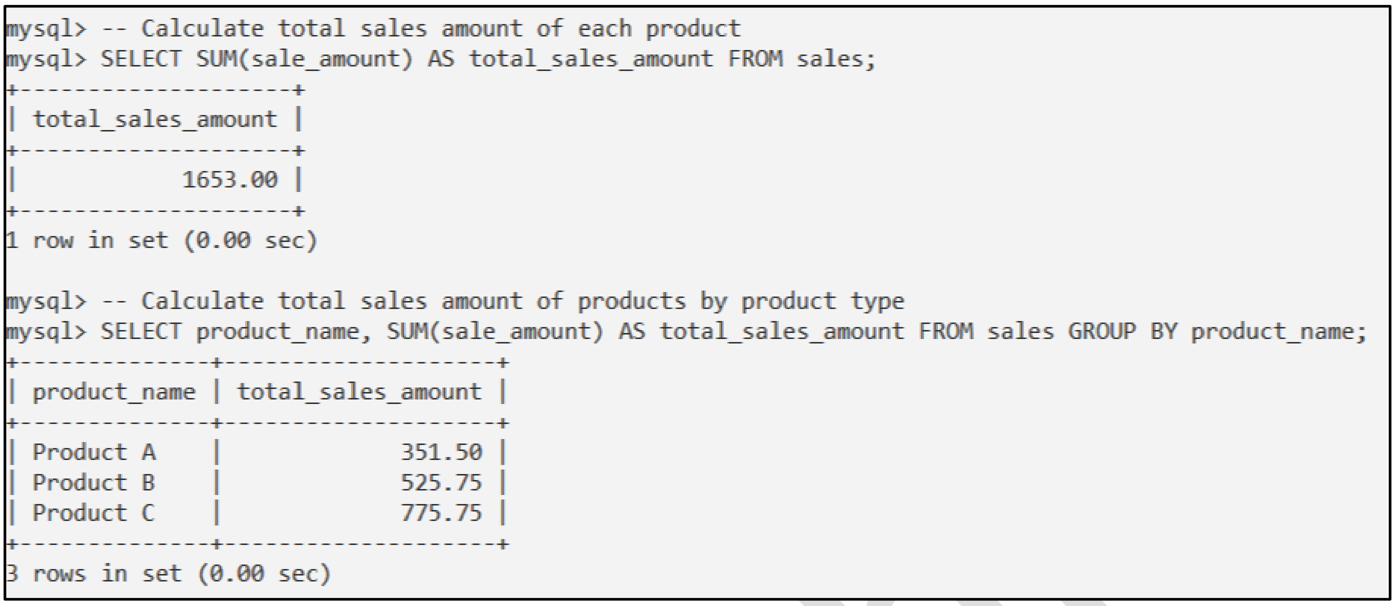


```
mysql> -- Find the minimum sale amount
mysql> SELECT MIN(sale_amount) FROM sales;
+------------------+
| MIN(sale_amount) |
+------------------+
|            50.25 |
+------------------+
1 row in set (0.00 sec)

mysql>
mysql> -- Find the minimum sale amount for each product
mysql> SELECT product_name, MIN(sale_amount) FROM sales GROUP BY product_name;
+--------------+------------------+
| product_name | MIN(sale_amount) |
+--------------+------------------+
| Product A    |            50.25 |
| Product B    |           150.00 |
| Product C    |           225.00 |
+--------------+------------------+
3 rows in set (0.00 sec)
```

P.T.O →

5] SUM() : -



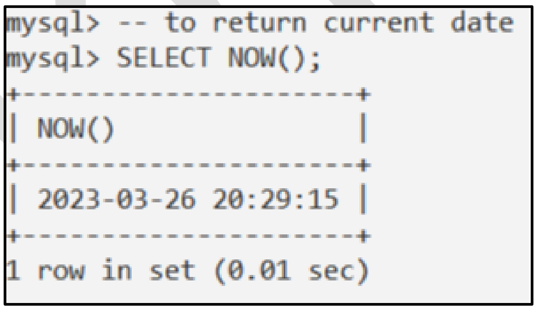
```
mysql> -- Calculate total sales amount of each product
mysql> SELECT SUM(sale_amount) AS total_sales_amount FROM sales;
+--------------------+
| total_sales_amount |
+--------------------+
|            1653.00 |
+--------------------+
1 row in set (0.00 sec)

mysql> -- Calculate total sales amount of products by product type
mysql> SELECT product_name, SUM(sale_amount) AS total_sales_amount FROM sales GROUP BY product_name;
+--------------+--------------------+
| product_name | total_sales_amount |
+--------------+--------------------+
| Product A    |             351.50 |
| Product B    |             525.75 |
| Product C    |             775.75 |
+--------------+--------------------+
3 rows in set (0.00 sec)
```

B] Date functions.

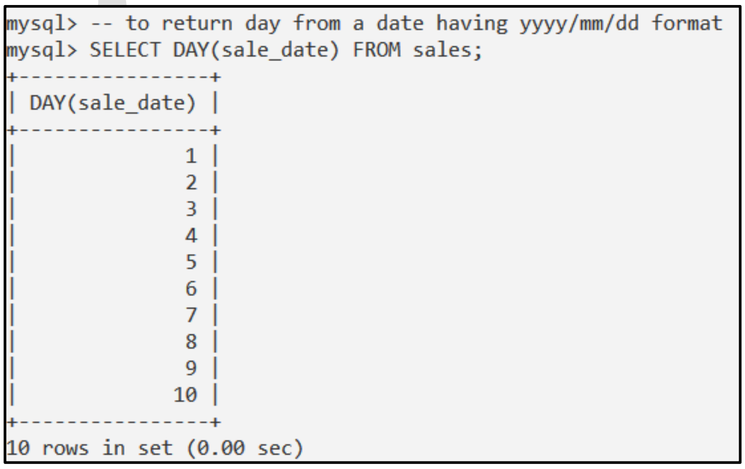
1] Reference table – same as for aggregate functions.

2] NOW() : -



```
mysql> -- to return current date
mysql> SELECT NOW();
+---------------------+
| NOW()               |
+---------------------+
| 2023-03-26 20:29:15 |
+---------------------+
1 row in set (0.01 sec)
```

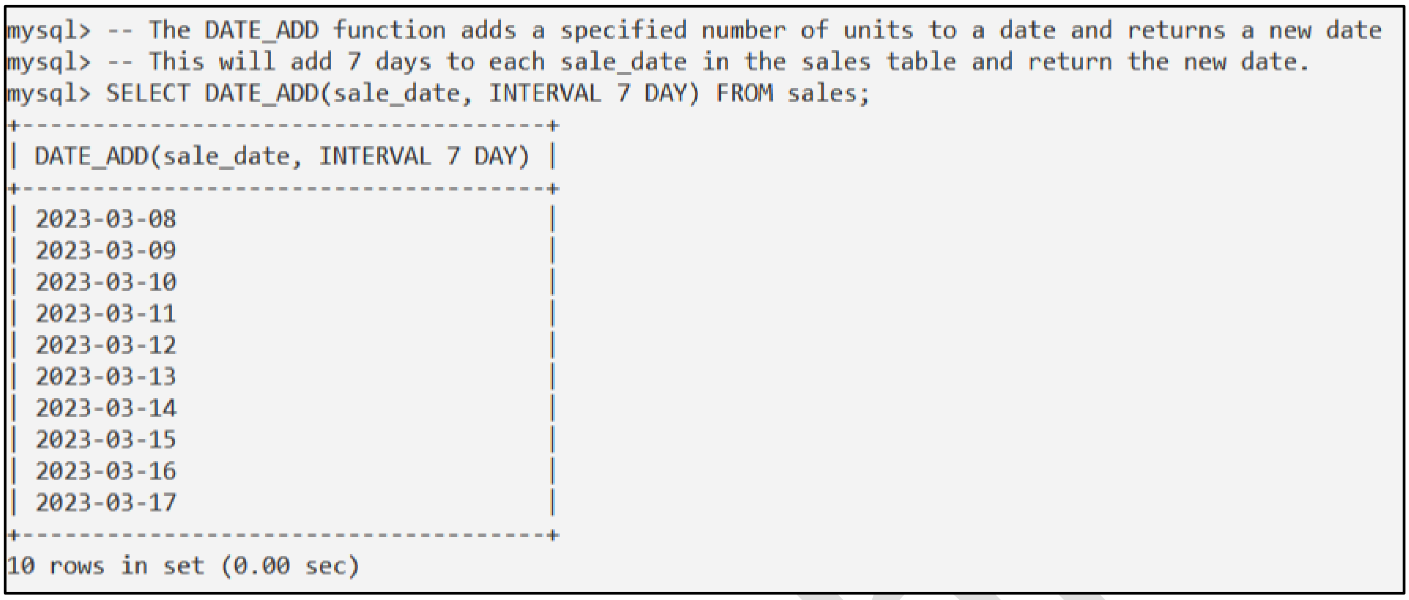
3] DAY() : -



```
mysql> -- to return day from a date having yyyy/mm/dd format
mysql> SELECT DAY(sale_date) FROM sales;
+----------------+
| DAY(sale_date) |
+----------------+
|              1 |
|              2 |
|              3 |
|              4 |
|              5 |
|              6 |
|              7 |
|              8 |
|              9 |
|             10 |
+----------------+
10 rows in set (0.00 sec)
```

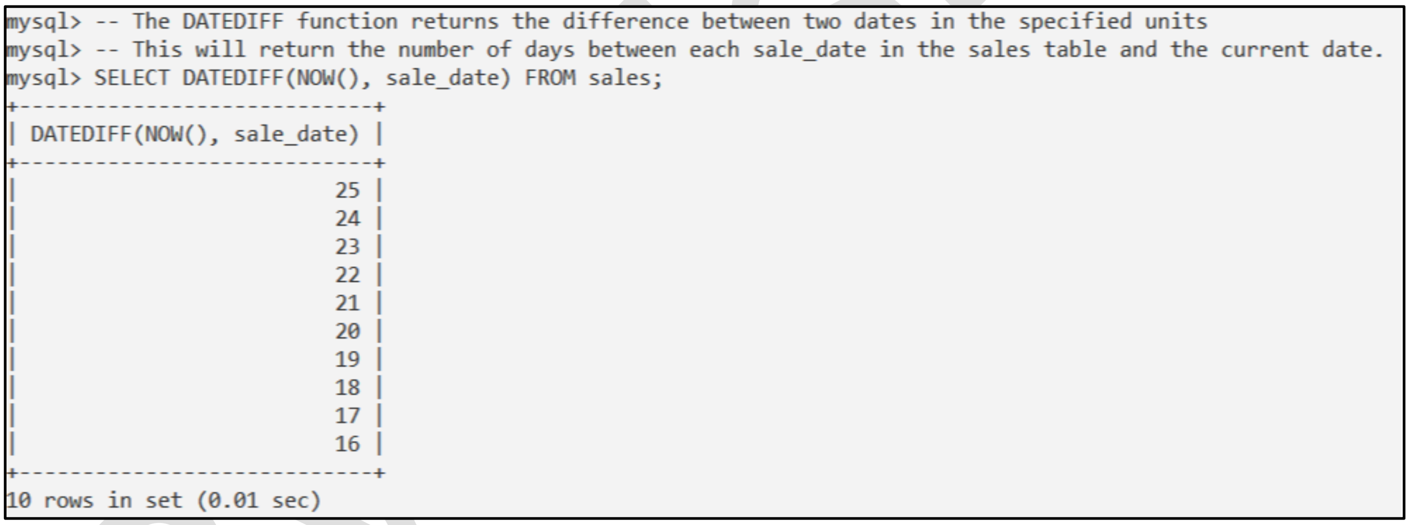
P.T.O →

4] DATE_ADD() : -



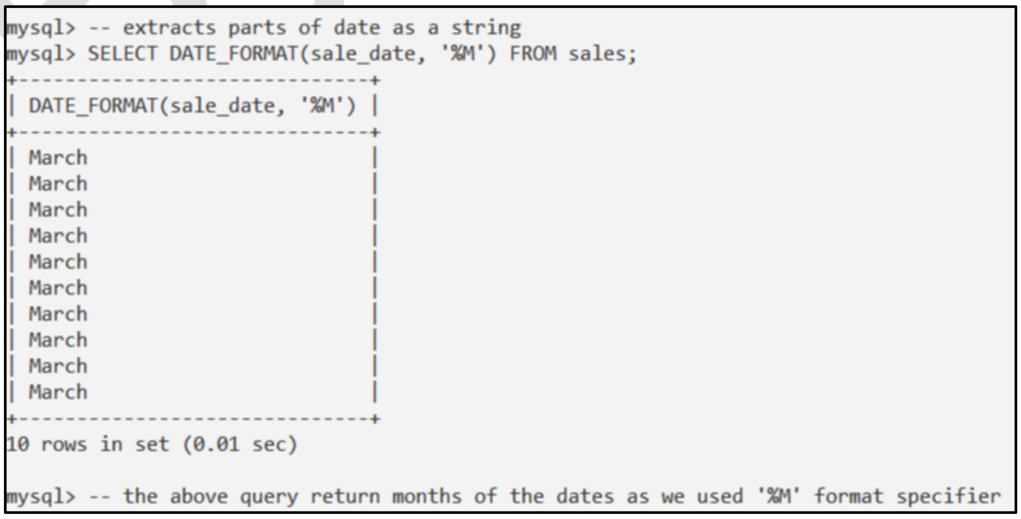
```
mysql> -- The DATE_ADD function adds a specified number of units to a date and returns a new date
mysql> -- This will add 7 days to each sale_date in the sales table and return the new date.
mysql> SELECT DATE_ADD(sale_date, INTERVAL 7 DAY) FROM sales;
+-----------------------------------+
| DATE_ADD(sale_date, INTERVAL 7 DAY) |
+-----------------------------------+
| 2023-03-08                        |
| 2023-03-09                        |
| 2023-03-10                        |
| 2023-03-11                        |
| 2023-03-12                        |
| 2023-03-13                        |
| 2023-03-14                        |
| 2023-03-15                        |
| 2023-03-16                        |
| 2023-03-17                        |
+-----------------------------------+
10 rows in set (0.00 sec)
```

5] DATEDIFF() : -



```
mysql> -- The DATEDIFF function returns the difference between two dates in the specified units
mysql> -- This will return the number of days between each sale_date in the sales table and the current date.
mysql> SELECT DATEDIFF(NOW(), sale_date) FROM sales;
+----------------------------+
| DATEDIFF(NOW(), sale_date) |
+----------------------------+
|                         25 |
|                         24 |
|                         23 |
|                         22 |
|                         21 |
|                         20 |
|                         19 |
|                         18 |
|                         17 |
|                         16 |
+----------------------------+
10 rows in set (0.01 sec)
```

6] DATE_FORMAT() : -



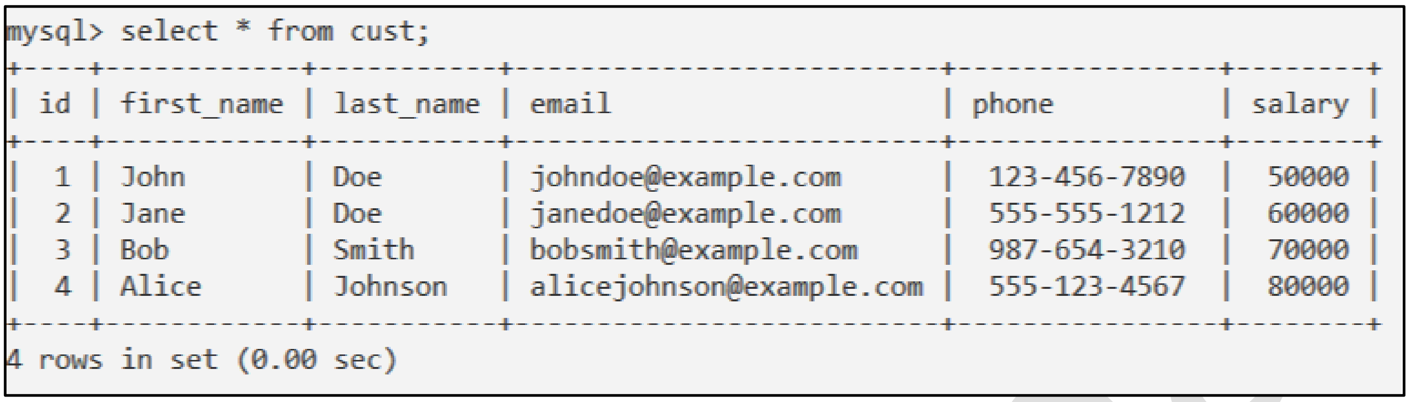
```
mysql> -- extracts parts of date as a string
mysql> SELECT DATE_FORMAT(sale_date, '%M') FROM sales;
+------------------------------+
| DATE_FORMAT(sale_date, '%M') |
+------------------------------+
| March                        |
| March                        |
| March                        |
| March                        |
| March                        |
| March                        |
| March                        |
| March                        |
| March                        |
| March                        |
+------------------------------+
10 rows in set (0.01 sec)

mysql> -- the above query return months of the dates as we used '%M' format specifier
```

P.T.O →

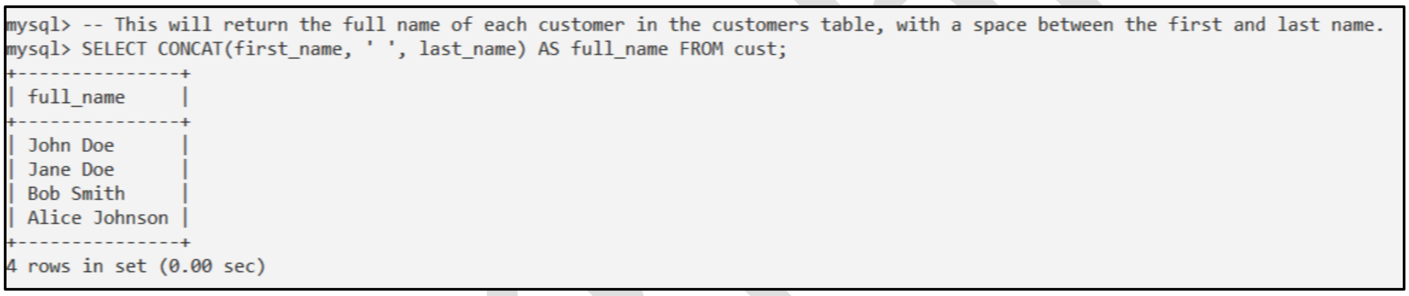
C] String Functions.

1] Reference table : -



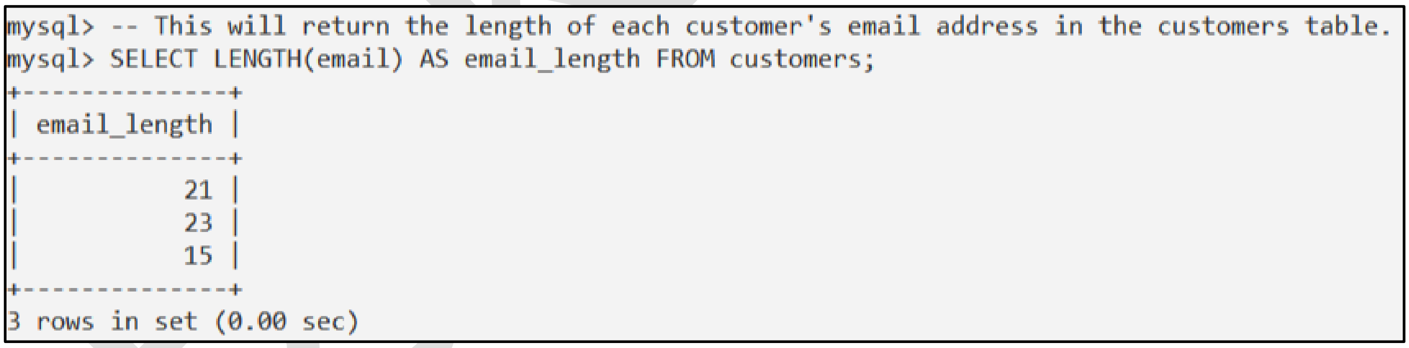
```
mysql> select * from cust;
+----+------------+-----------+-------------------------+--------------+--------+
| id | first_name | last_name | email                   | phone        | salary |
+----+------------+-----------+-------------------------+--------------+--------+
|  1 | John       | Doe       | johndoe@example.com     | 123-456-7890 |  50000 |
|  2 | Jane       | Doe       | janedoe@example.com     | 555-555-1212 |  60000 |
|  3 | Bob        | Smith     | bobsmith@example.com    | 987-654-3210 |  70000 |
|  4 | Alice      | Johnson   | alicejohnson@example.com | 555-123-4567 |  80000 |
+----+------------+-----------+-------------------------+--------------+--------+
4 rows in set (0.00 sec)
```

2] CONCAT() : -



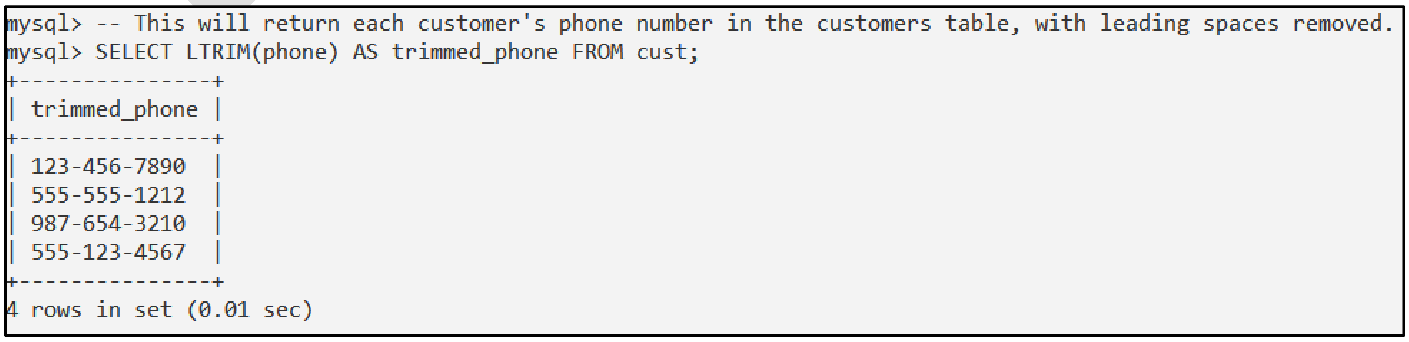
```
mysql> -- This will return the full name of each customer in the customers table, with a space between the first and last name.
mysql> SELECT CONCAT(first_name, ' ', last_name) AS full_name FROM cust;
+---------------+
| full_name     |
+---------------+
| John Doe      |
| Jane Doe      |
| Bob Smith     |
| Alice Johnson |
+---------------+
4 rows in set (0.00 sec)
```

3] LEN() : -



```
mysql> -- This will return the length of each customer's email address in the customers table.
mysql> SELECT LENGTH(email) AS email_length FROM customers;
+--------------+
| email_length |
+--------------+
|           21 |
|           23 |
|           15 |
+--------------+
3 rows in set (0.00 sec)
```

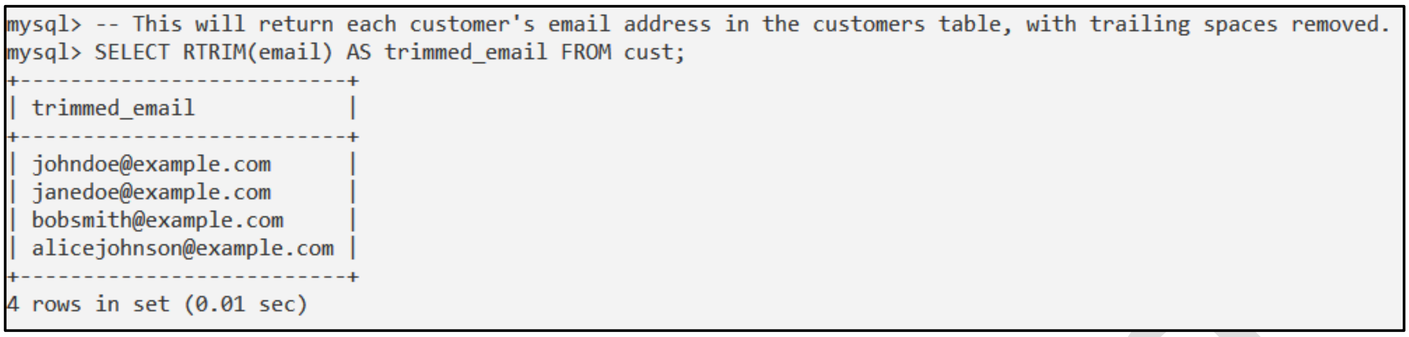
4] LTRIM() : -



```
mysql> -- This will return each customer's phone number in the customers table, with leading spaces removed.
mysql> SELECT LTRIM(phone) AS trimmed_phone FROM cust;
+---------------+
| trimmed_phone |
+---------------+
| 123-456-7890  |
| 555-555-1212  |
| 987-654-3210  |
| 555-123-4567  |
+---------------+
4 rows in set (0.01 sec)
```

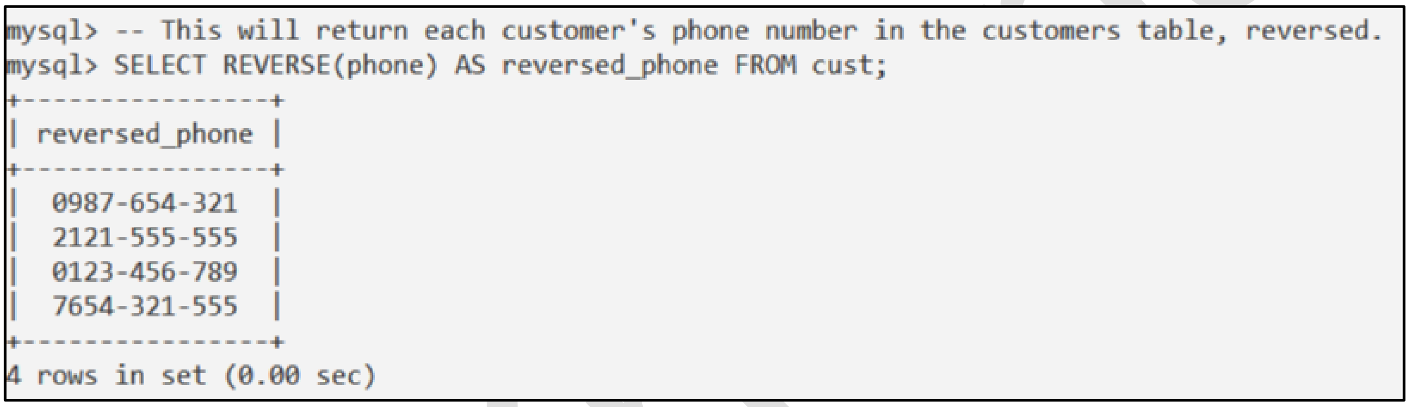
P.T.O →

5] RTRIM() : -



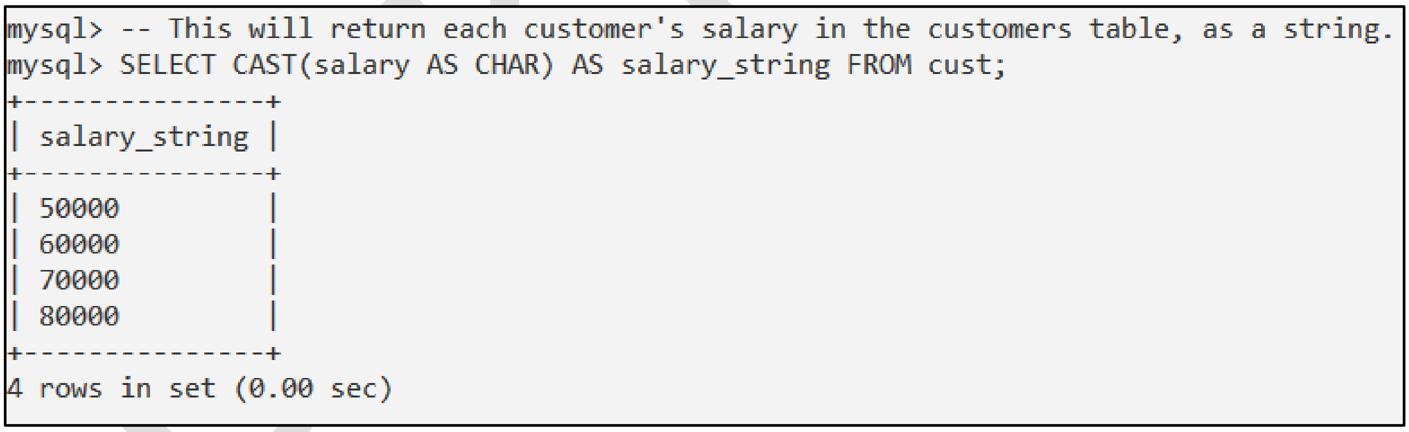
```
mysql> -- This will return each customer's email address in the customers table, with trailing spaces removed.
mysql> SELECT RTRIM(email) AS trimmed_email FROM cust;
+-------------------------+
| trimmed_email           |
+-------------------------+
| johndoe@example.com     |
| janedoe@example.com     |
| bobsmith@example.com    |
| alicejohnson@example.com |
+-------------------------+
4 rows in set (0.01 sec)
```

6] REVERSE() : -



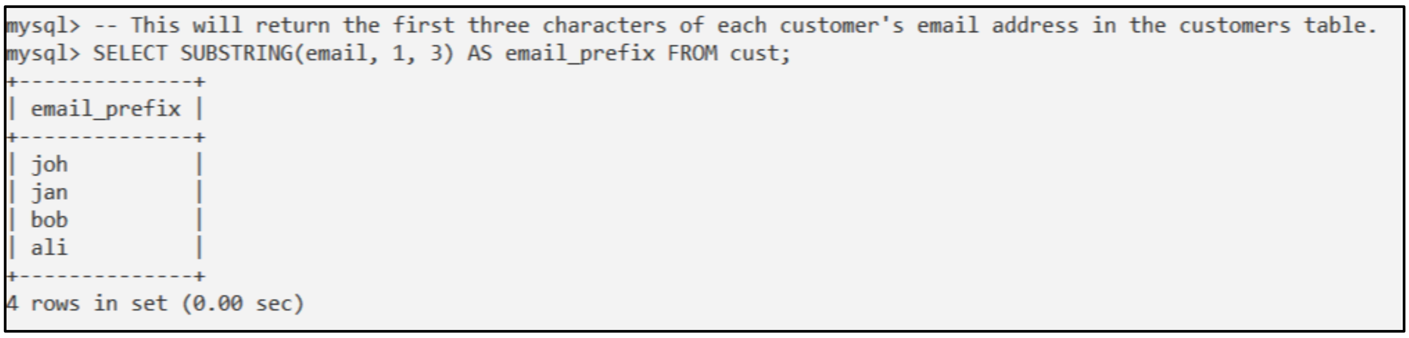
```
mysql> -- This will return each customer's phone number in the customers table, reversed.
mysql> SELECT REVERSE(phone) AS reversed_phone FROM cust;
+----------------+
| reversed_phone |
+----------------+
|  0987-654-321  |
|  2121-555-555  |
|  0123-456-789  |
|  7654-321-555  |
+----------------+
4 rows in set (0.00 sec)
```

7] CAST() : -



```
mysql> -- This will return each customer's salary in the customers table, as a string.
mysql> SELECT CAST(salary AS CHAR) AS salary_string FROM cust;
+---------------+
| salary_string |
+---------------+
| 50000         |
| 60000         |
| 70000         |
| 80000         |
+---------------+
4 rows in set (0.00 sec)
```

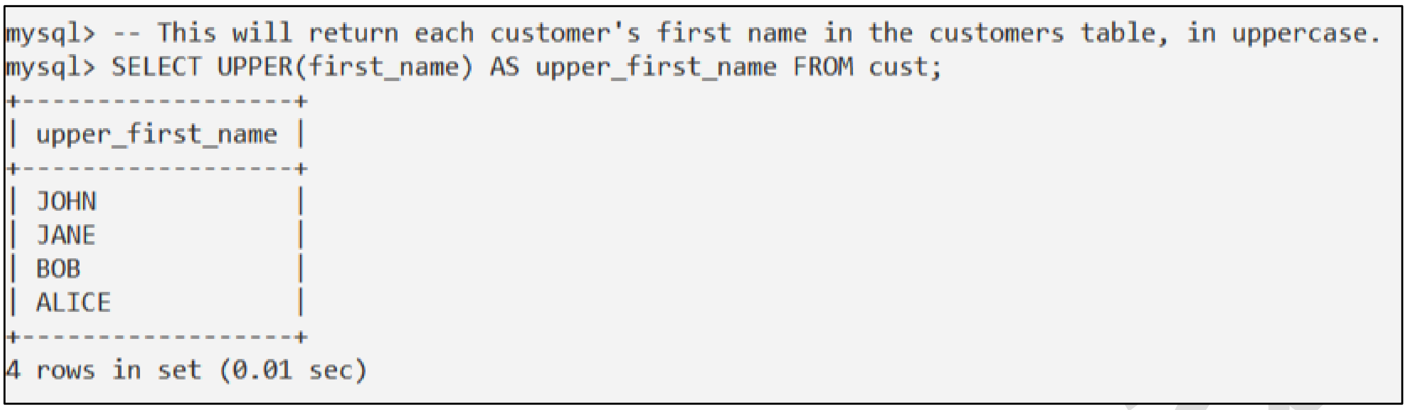
8] SUBSTRING.



```
mysql> -- This will return the first three characters of each customer's email address in the customers table.
mysql> SELECT SUBSTRING(email, 1, 3) AS email_prefix FROM cust;
+--------------+
| email_prefix |
+--------------+
| joh          |
| jan          |
| bob          |
| ali          |
+--------------+
4 rows in set (0.00 sec)
```

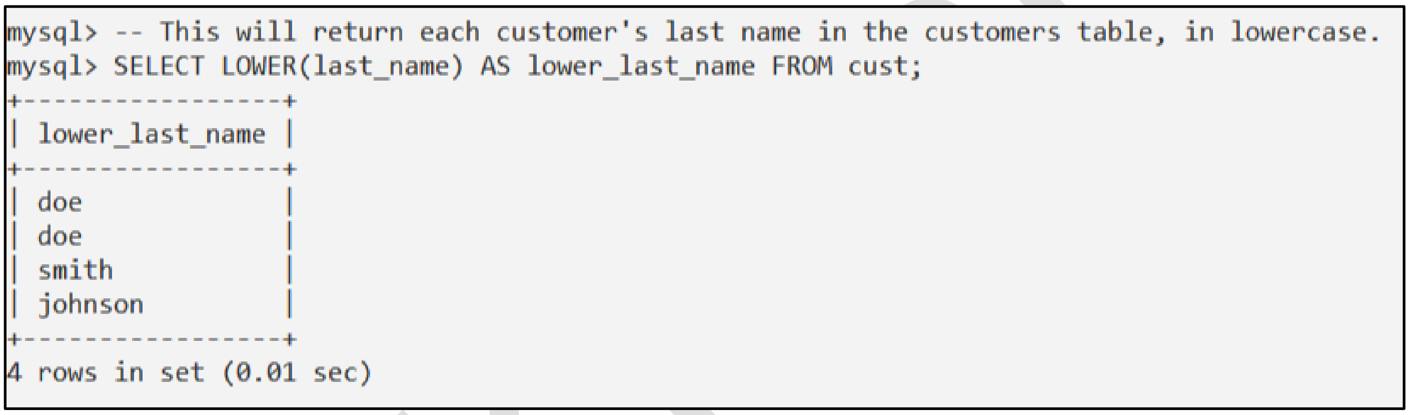
P.T.O →

9] UPPER() : -



```
mysql> -- This will return each customer's first name in the customers table, in uppercase.
mysql> SELECT UPPER(first_name) AS upper_first_name FROM cust;
+------------------+
| upper_first_name |
+------------------+
| JOHN             |
| JANE             |
| BOB              |
| ALICE            |
+------------------+
4 rows in set (0.01 sec)
```

10] LOWER() : -



```
mysql> -- This will return each customer's last name in the customers table, in lowercase.
mysql> SELECT LOWER(last_name) AS lower_last_name FROM cust;
+-----------------+
| lower_last_name |
+-----------------+
| doe             |
| doe             |
| smith           |
| johnson         |
+-----------------+
4 rows in set (0.01 sec)
```