QUERYING
AND
SQL FUNCTION

SQL Functions

A function is a predefined set of commands that performs specific task or operation on data returns a value.

SQL functions are also set of command which are applied on relations to perform a operation.
The MySql functions broadly divided into two categories.

Single Row functions (Scalar functions) Multiple Row functions (group or aggregate functions)

Single Row Functions

- The single row functions work with each row at a time and return one result per row or you can say that a relation have n number of rows than it will return n number of values.

  e.g.

- Maths functions
- String functions
- Date & Time functions

Multiple Row Functions

The Multiple row functions work with more than one row at a time and return one result or you can say that a relation have n number of rows than it will return aggregate values.

e.g.

- aggregate functions (Min(), Max(), Avg(), Count(), Sum())

Maths functions

Mathematical functions are also called number functions that take numeric value as a input and return numeric value as a output.

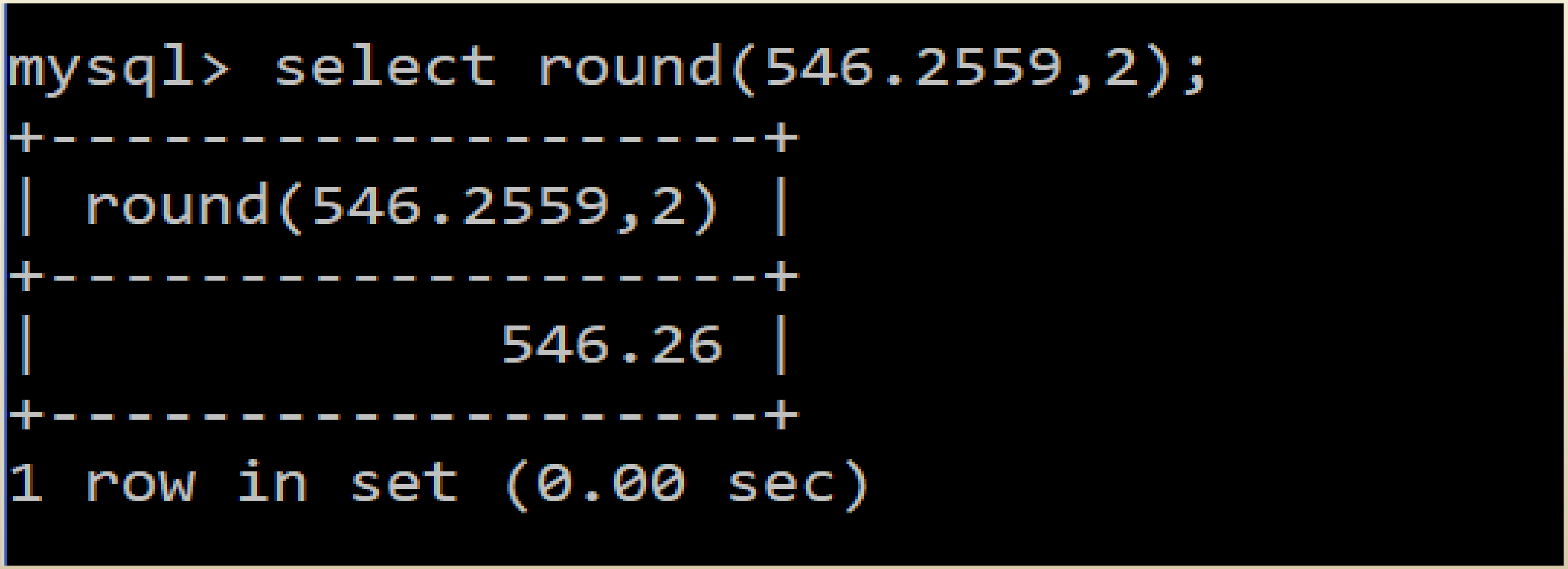
Mathematical functions are…

- Round()
- Trunc()
- Mod()
- Power()

Round()

The ROUND() function rounds a number to a specified number of decimal places.

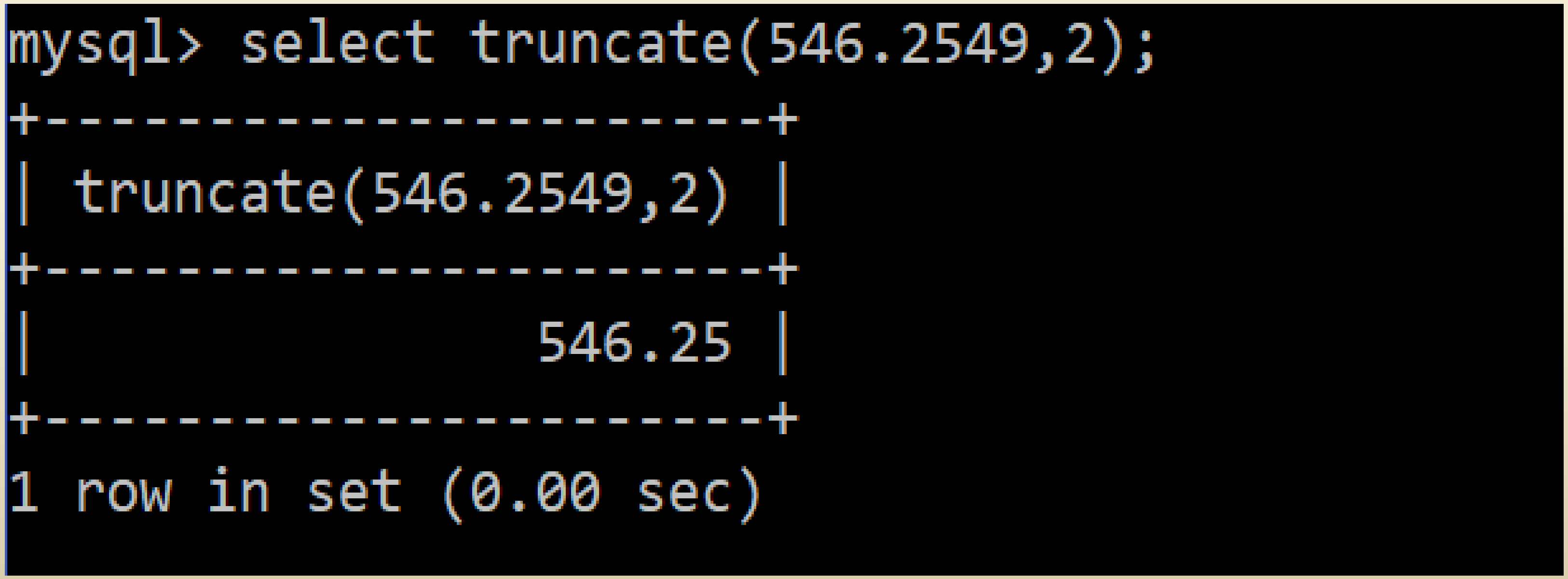
- ROUND(*number, decimals*)



```
mysql> select round(546.2559,2);
+--------------------+
| round(546.2559,2) |
+--------------------+
|             546.26 |
+--------------------+
1 row in set (0.00 sec)
```

Truncate()

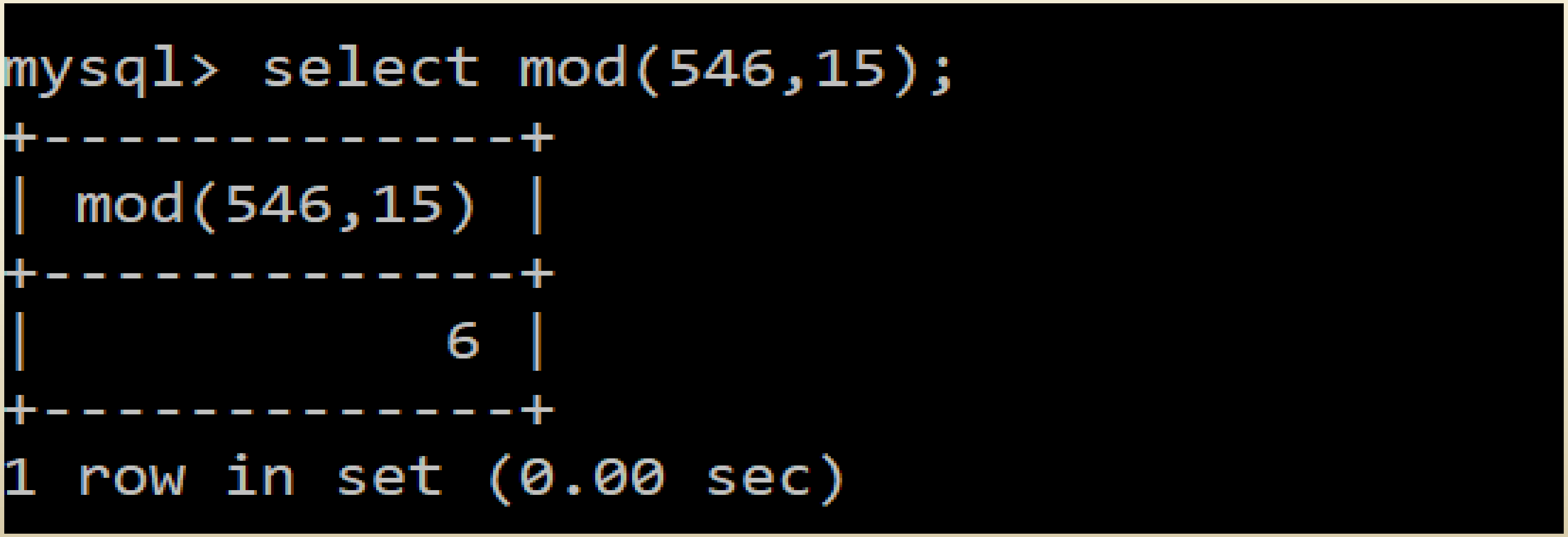
MySQL TRUNCATE() returns a number after truncated to certain decimal places. The number and the number of decimal places are specified as arguments of the **TRUNCATE** function**.**



```
mysql> select truncate(546.2549,2);
+----------------------+
| truncate(546.2549,2) |
+----------------------+
|               546.25 |
+----------------------+
1 row in set (0.00 sec)
```

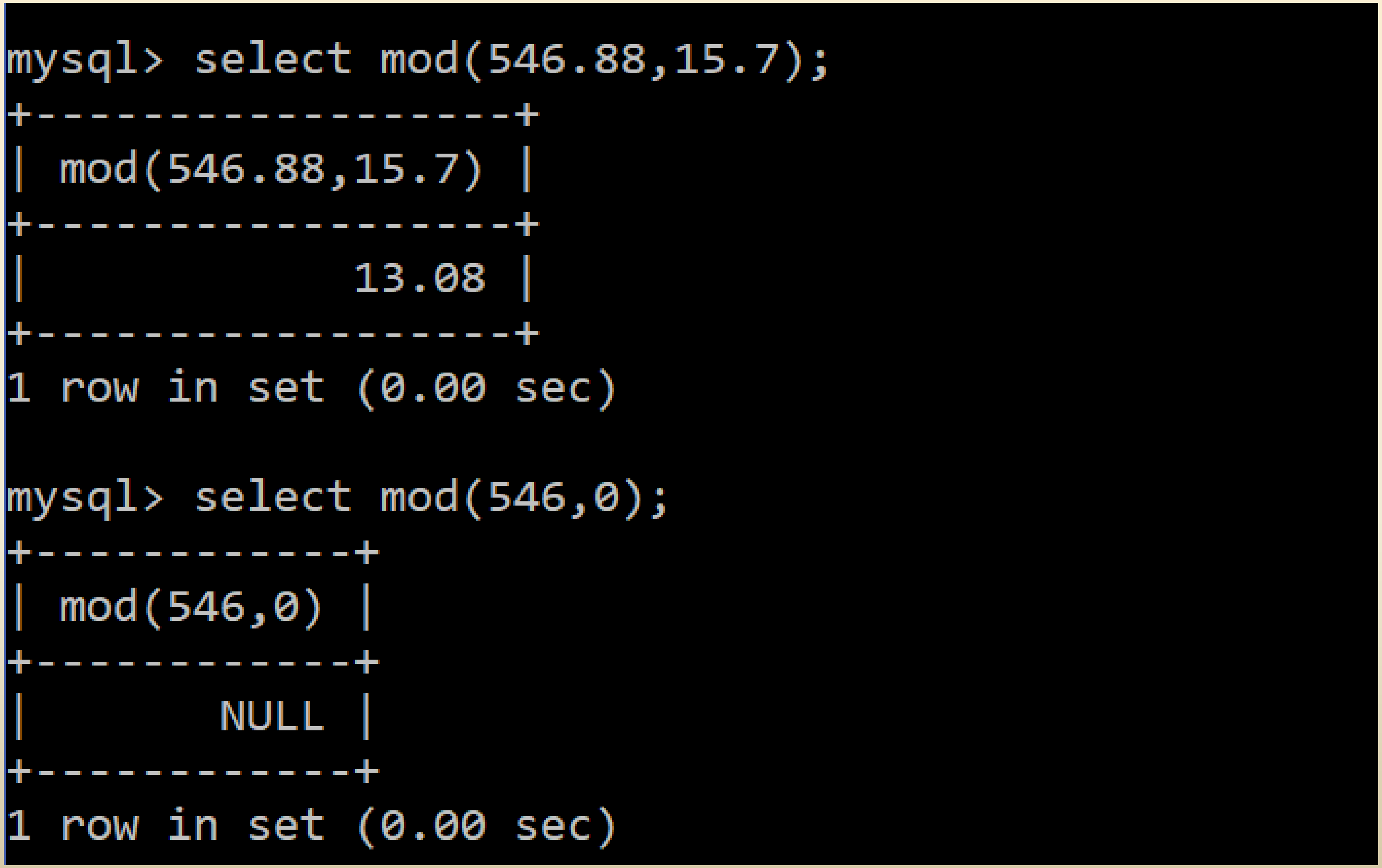
Mod()

MySQL MOD() returns the **remainder** of a number divided by another number. This **function** also works on fractional values and returns the exact **remainder**. The **function** returns NULL when the value of divisor is 0.



```
mysql> select mod(546,15);
+--------------+
| mod(546,15) |
+--------------+
|            6 |
+--------------+
1 row in set (0.00 sec)
```

Example :



```
mysql> select mod(546.88,15.7);
+------------------+
| mod(546.88,15.7) |
+------------------+
|            13.08 |
+------------------+
1 row in set (0.00 sec)

mysql> select mod(546,0);
+------------+
| mod(546,0) |
+------------+
|       NULL |
+------------+
1 row in set (0.00 sec)
```

Power()

MySQL POWER() returns the **value** of a number raised to the power of another number.

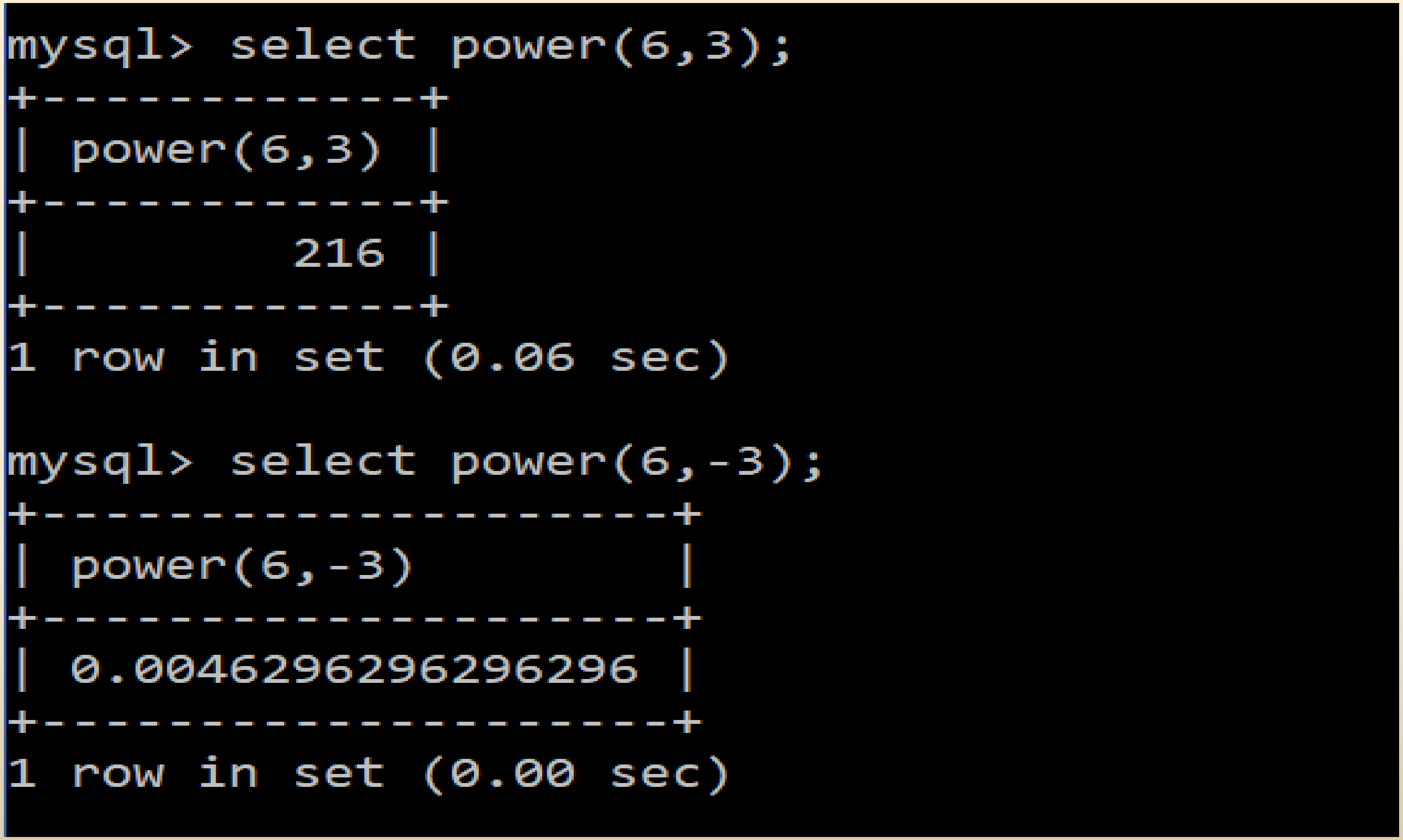
The synonym of POWER() is POW()

POWER(m, n)

M Numeric value. It is the base used in the calculation.

N Numeric value. It is the exponent used in the calculation.

Example :



```
mysql> select power(6,3);
+------------+
| power(6,3) |
+------------+
|        216 |
+------------+
1 row in set (0.06 sec)

mysql> select power(6,-3);
+----------------------+
| power(6,-3)          |
+----------------------+
| 0.004629629629629629 |
+----------------------+
1 row in set (0.00 sec)
```

String functions

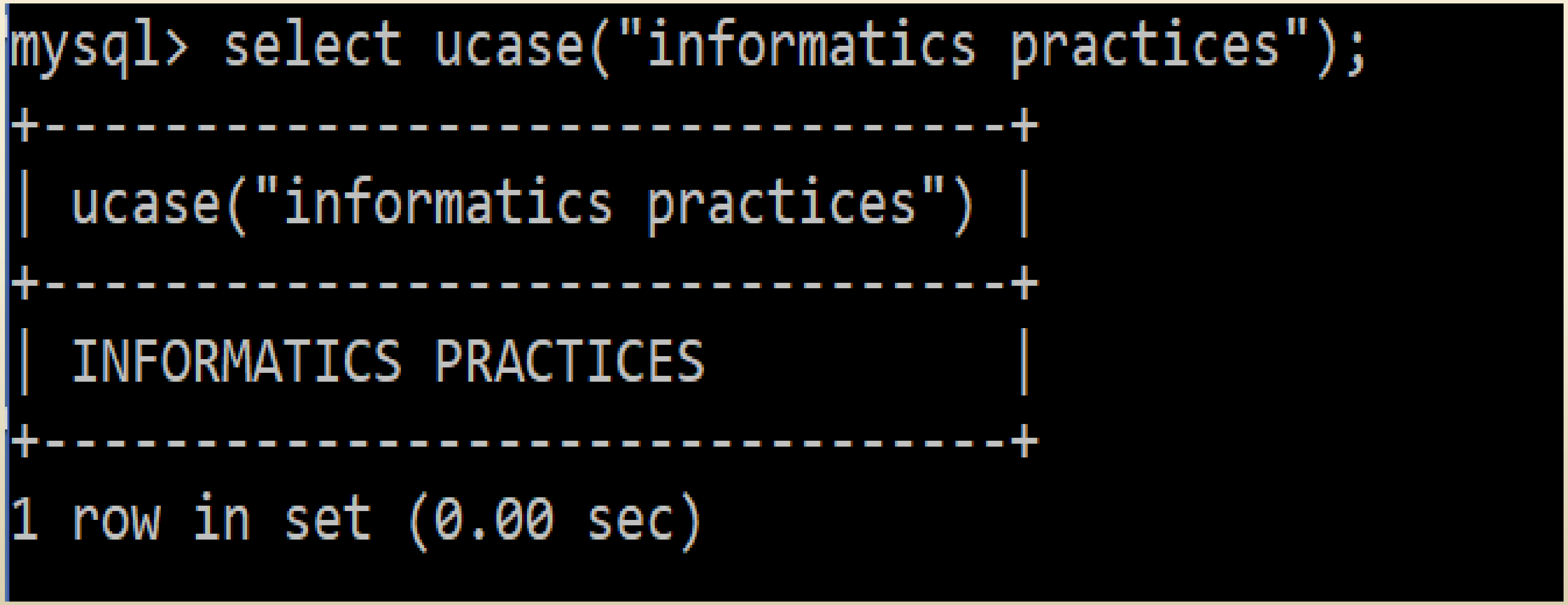
String functions that take string as a input and return String/Numeric value as a output depend on function.

Functions are…

- UCASE ()/UPPER (),
- LCASE ()/LOWER (),
- MID ()/SUBSTRING ()/SUBSTR (),
- LENGTH (),
- LEFT (),
- RIGHT (),
- INSTR (),
- LTRIM (),
- RTRIM (),
- TRIM ().

UCASE ()/UPPER ()

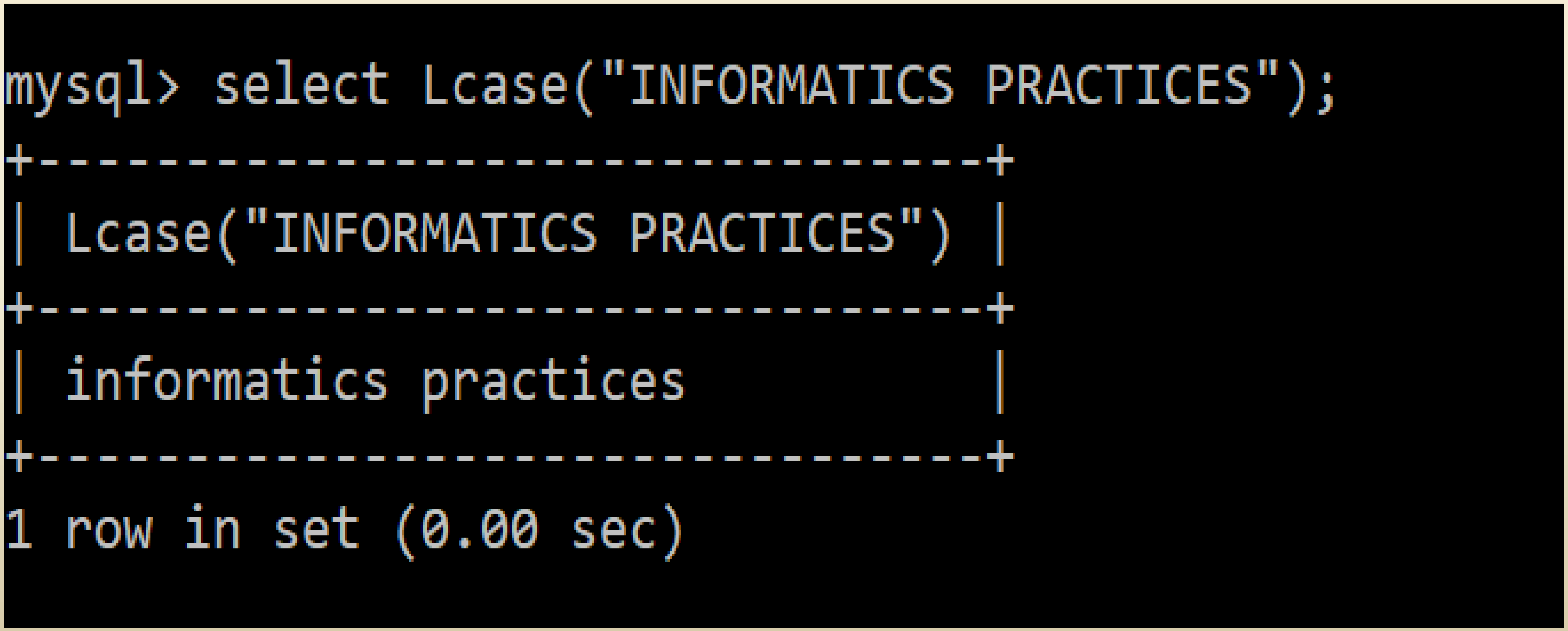
Returns the string with all characters changed to uppercase according to the current character set mapping.



```
mysql> select ucase("informatics practices");
+--------------------------------+
| ucase("informatics practices") |
+--------------------------------+
| INFORMATICS PRACTICES          |
+--------------------------------+
1 row in set (0.00 sec)
```

LCASE ()/LOWER ()

Returns the string with all characters changed to lowercase according to the current character set mapping.

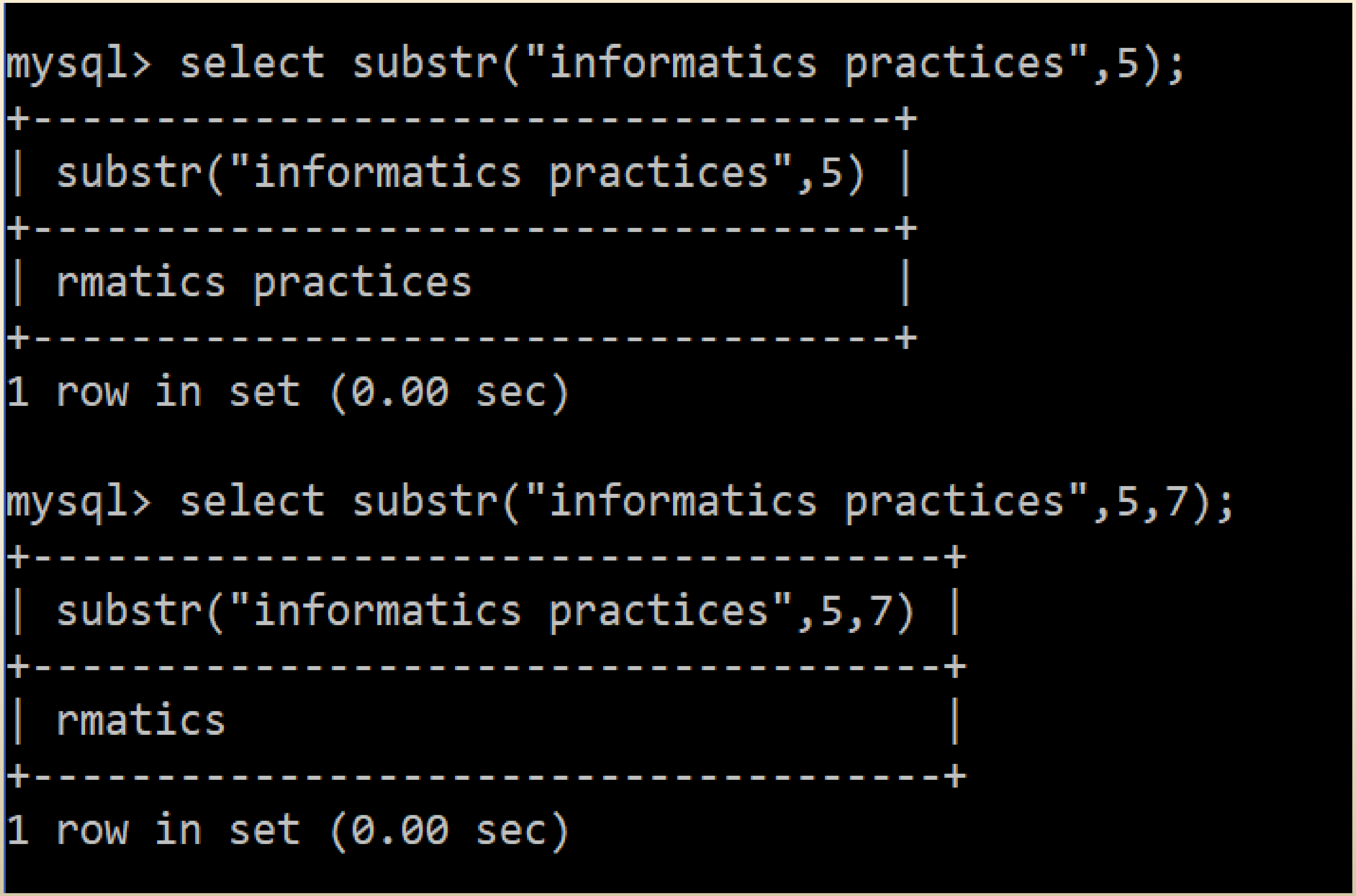


```
mysql> select Lcase("INFORMATICS PRACTICES");
+--------------------------------+
| Lcase("INFORMATICS PRACTICES") |
+--------------------------------+
| informatics practices          |
+--------------------------------+
1 row in set (0.00 sec)
```

MID ()/SUBSTRING ()/SUBSTR ()

- The forms of function (without a len argument) return a substring from string str starting at position pos upto last character of string str.
- The forms of function (with a len argument) return a substring of length len characters long from string str, starting at position pos.
- (It is also possible to use a negative value for pos. In this case, the beginning of the substring is pos characters from the end of the string, rather than the beginning. A negative value may be used for pos in any of the forms of this function).
- SUBSTRING(str,pos) [SUBSTRING(str FROM pos)]
- SUBSTRING(str,pos,len)[SUBSTRING(str FROM pos FOR len)]

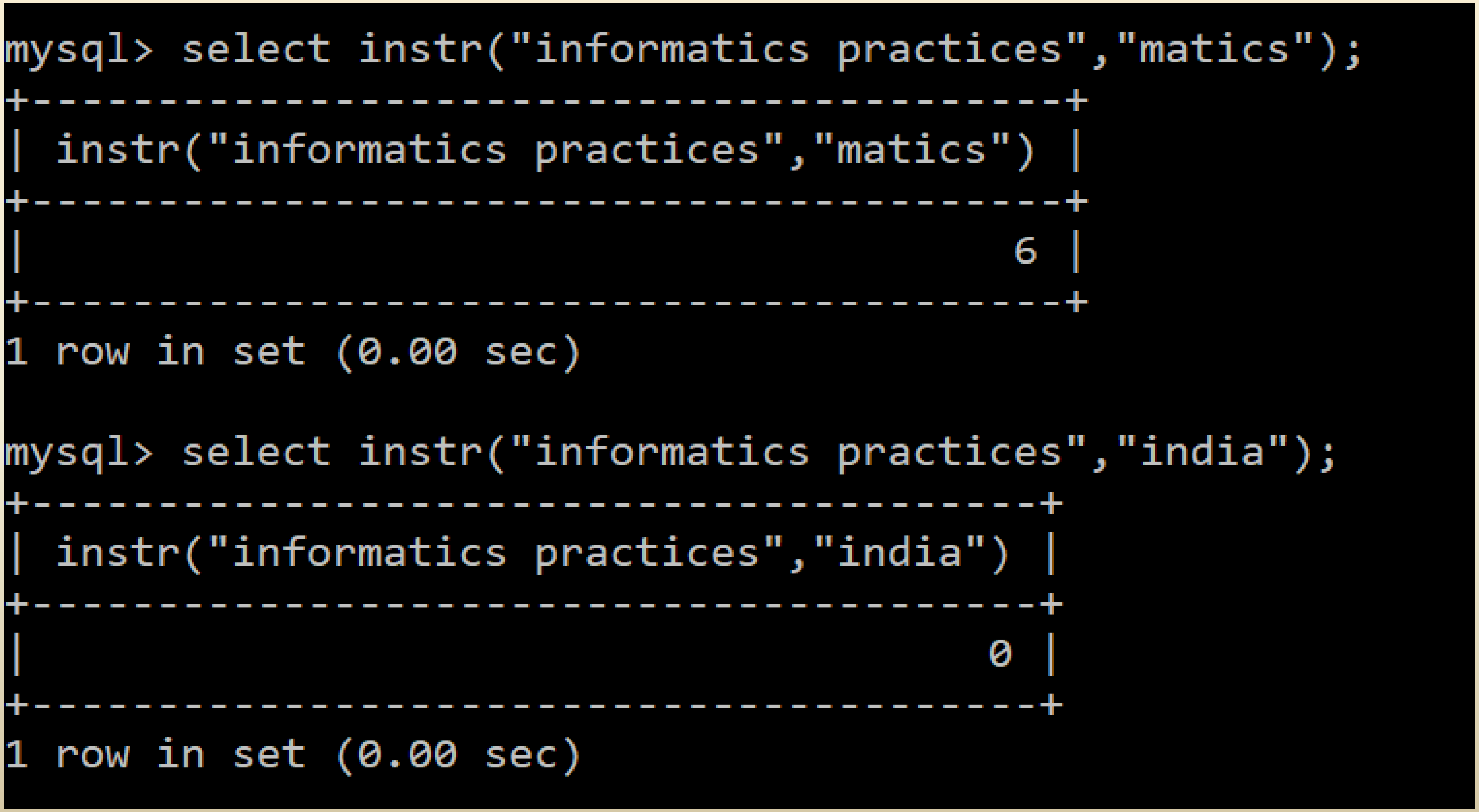
Example:



```
mysql> select substr("informatics practices",5);
+-----------------------------------+
| substr("informatics practices",5) |
+-----------------------------------+
| rmatics practices                 |
+-----------------------------------+
1 row in set (0.00 sec)

mysql> select substr("informatics practices",5,7);
+-------------------------------------+
| substr("informatics practices",5,7) |
+-------------------------------------+
| rmatics                             |
+-------------------------------------+
1 row in set (0.00 sec)
```

INSTR ()

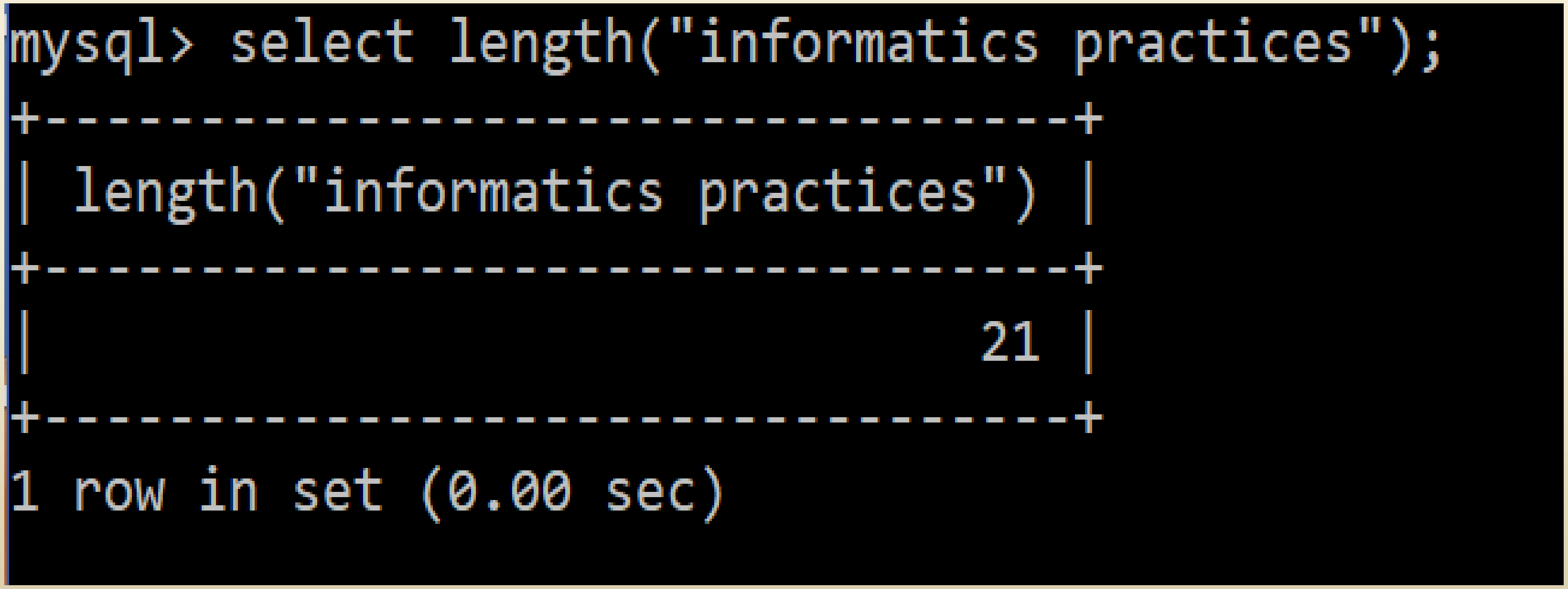
Returns the position of the first occurrence(position) of substring substr in string str and if substring is not a part of string than return 0.



```
mysql> select instr("informatics practices","matics");
+---------------------------------------+
| instr("informatics practices","matics") |
+---------------------------------------+
|                                     6 |
+---------------------------------------+
1 row in set (0.00 sec)

mysql> select instr("informatics practices","india");
+--------------------------------------+
| instr("informatics practices","india") |
+--------------------------------------+
|                                    0 |
+--------------------------------------+
1 row in set (0.00 sec)
```

Length()

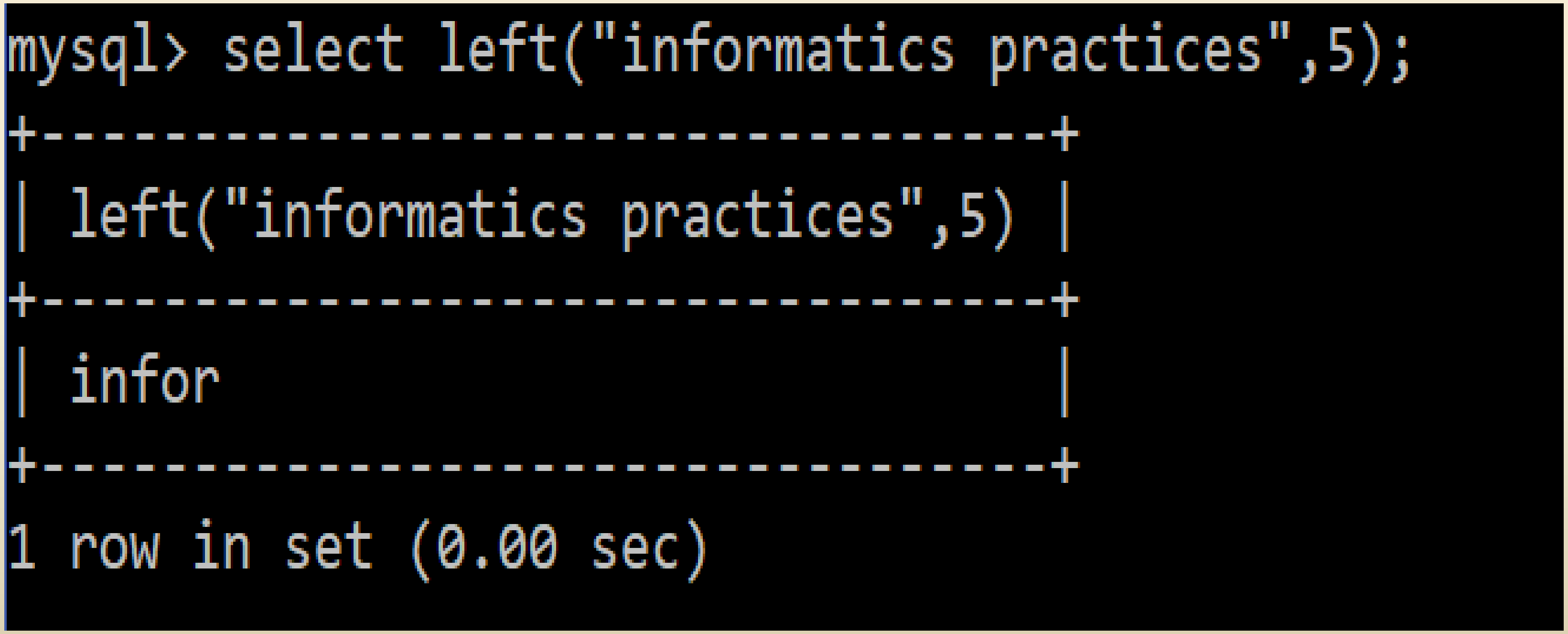
Returns the length of the string str, measured in bytes. A multi-byte character counts as multiple bytes.



```
mysql> select length("informatics practices");
+--------------------------------+
| length("informatics practices") |
+--------------------------------+
|                             21 |
+--------------------------------+
1 row in set (0.00 sec)
```

Left()

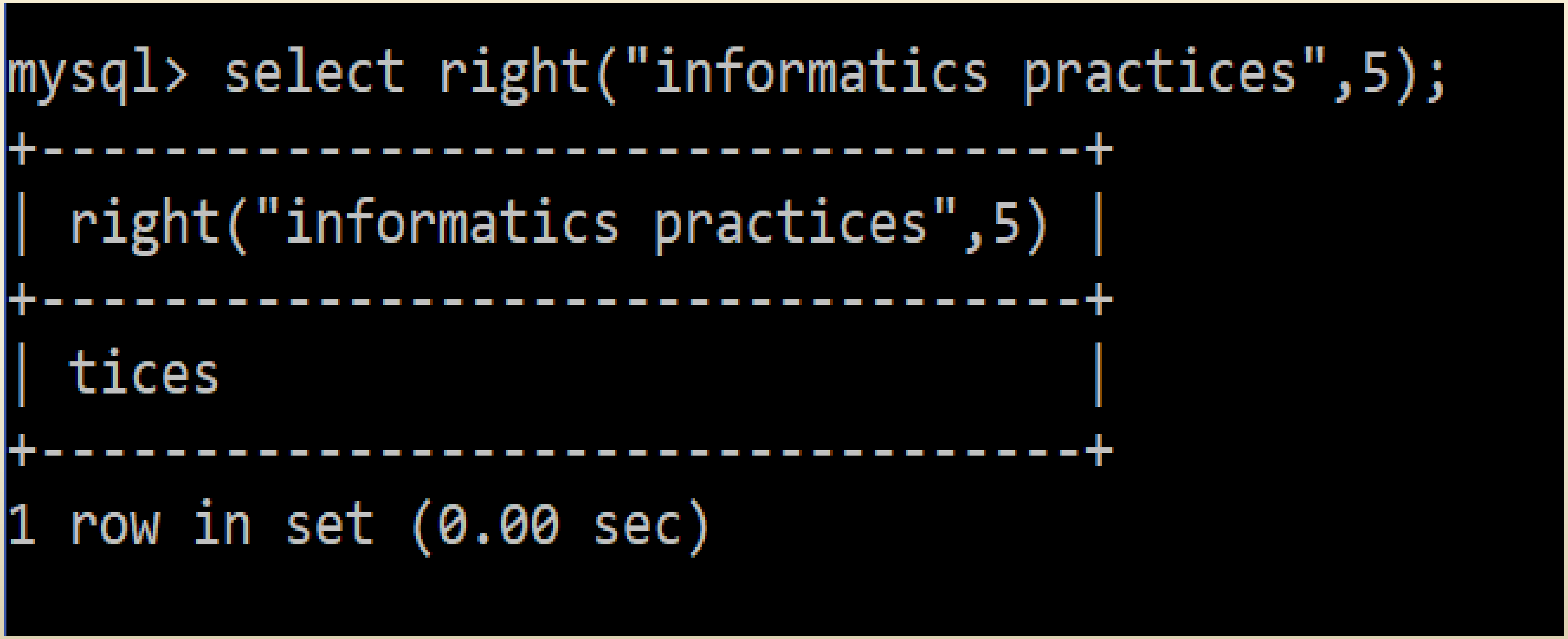
Returns the leftmost len characters from the string str, or NULL if any argument is NULL.



```
mysql> select left("informatics practices",5);
+--------------------------------+
| left("informatics practices",5) |
+--------------------------------+
| infor                           |
+--------------------------------+
1 row in set (0.00 sec)
```

Right()

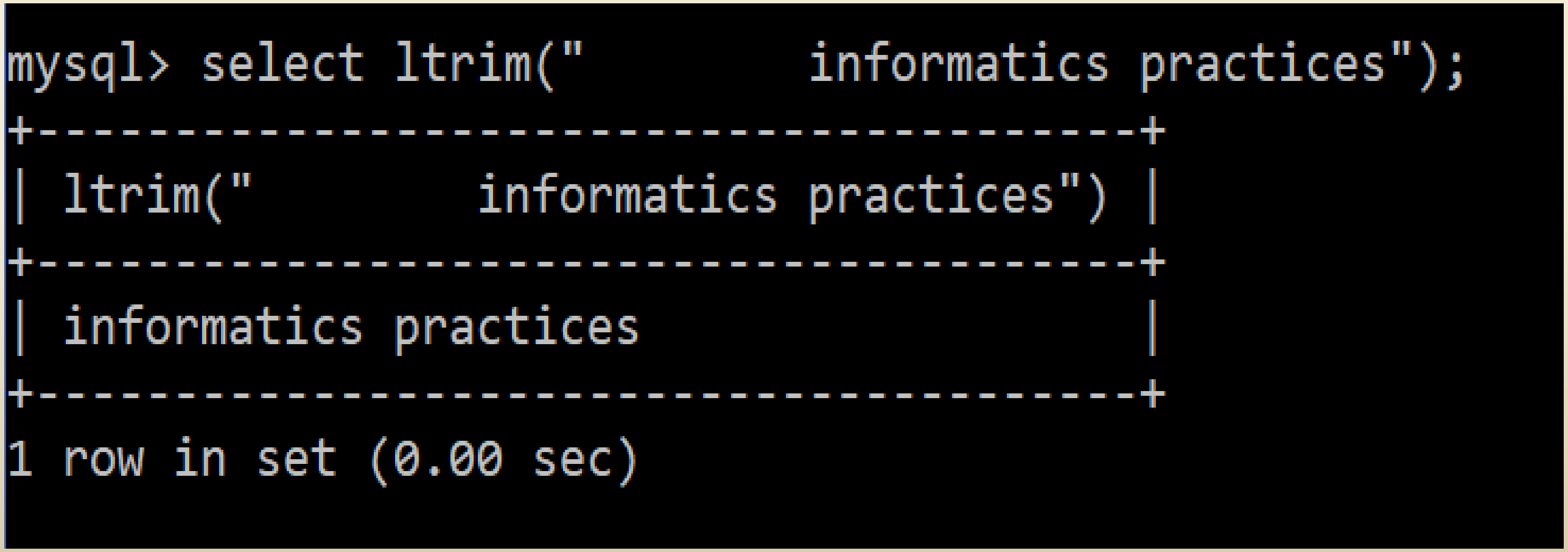
Returns the rightmost len characters from the string str, or NULL if any argument is NULL.



```
mysql> select right("informatics practices",5);
+----------------------------------+
| right("informatics practices",5) |
+----------------------------------+
| tices                            |
+----------------------------------+
1 row in set (0.00 sec)
```

Ltrim()

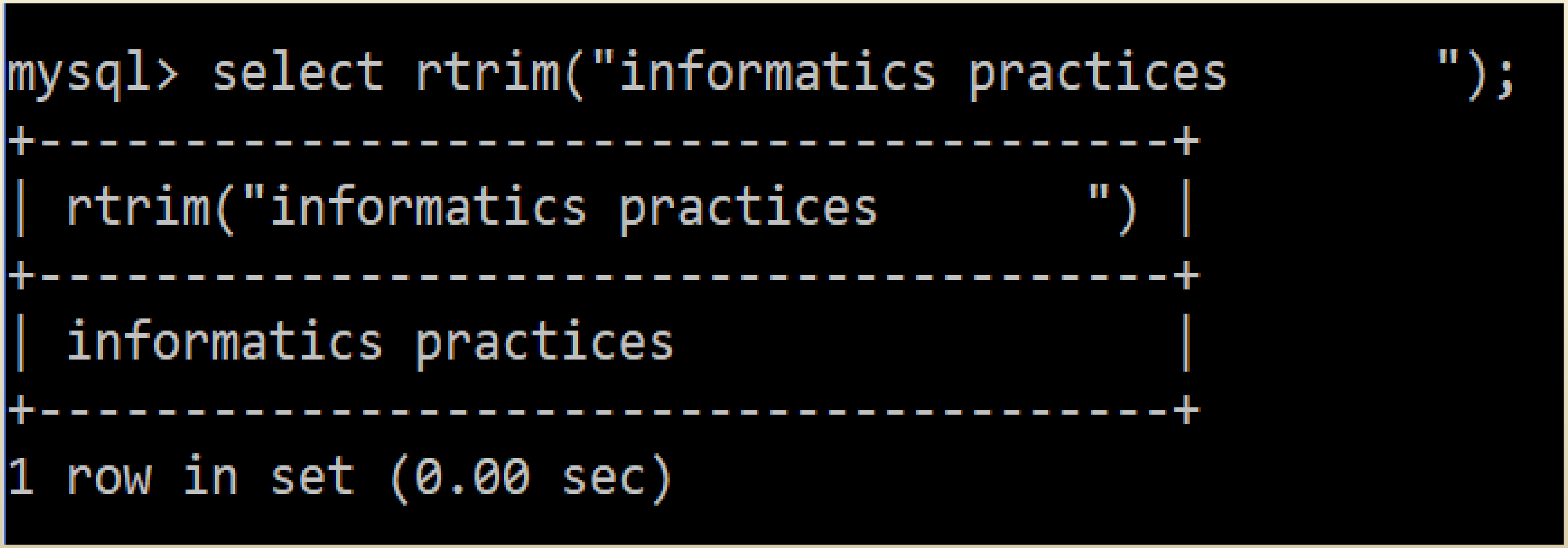
LTRIM() removes the leading space characters of a **string** passed as an argument. A **string** whose leading space characters are to be removed.



```
mysql> select ltrim("        informatics practices");
+-----------------------------------------+
| ltrim("        informatics practices") |
+-----------------------------------------+
| informatics practices                   |
+-----------------------------------------+
1 row in set (0.00 sec)
```

Rtrim()

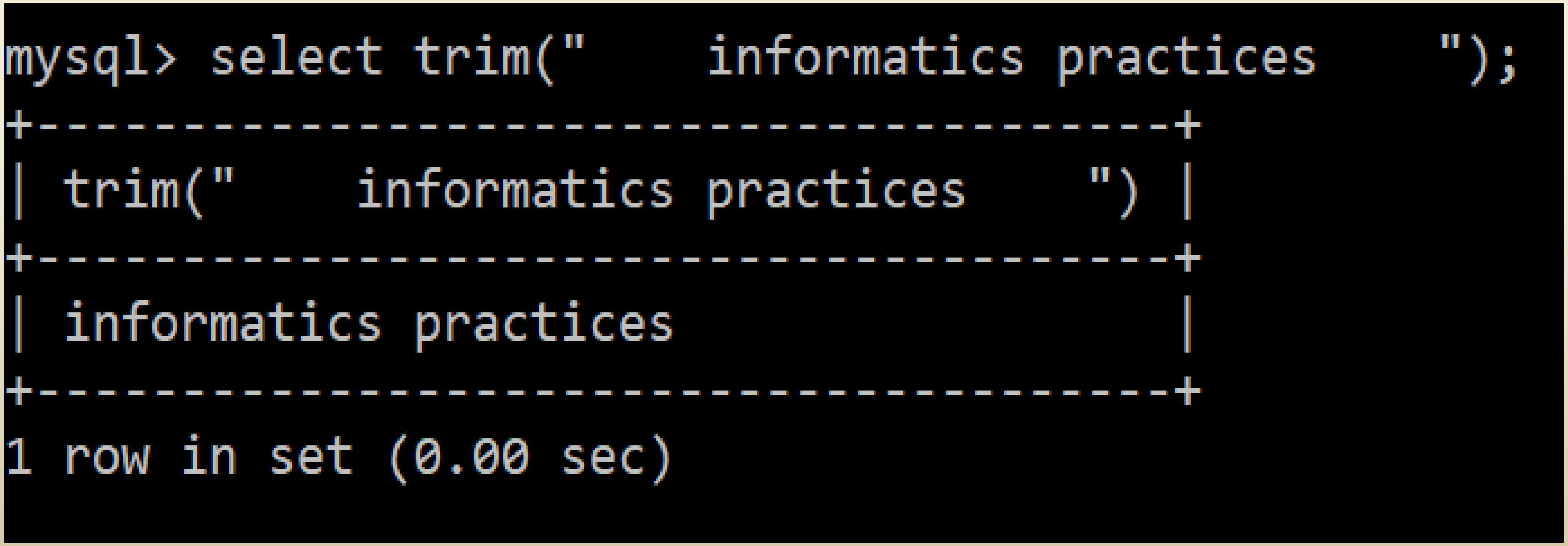
RTRIM() removes the trailing space characters of a string passed as an argument. A string whose trailing space characters are to be removed.



```
mysql> select rtrim("informatics practices      ");
+------------------------------------------+
| rtrim("informatics practices      ") |
+------------------------------------------+
| informatics practices                    |
+------------------------------------------+
1 row in set (0.00 sec)
```

Trim()

Returns the string str with all remstr prefixes or suffixes removed. If none of the specifies BOTH, LEADING, or TRAILING is given, BOTH is assumed.

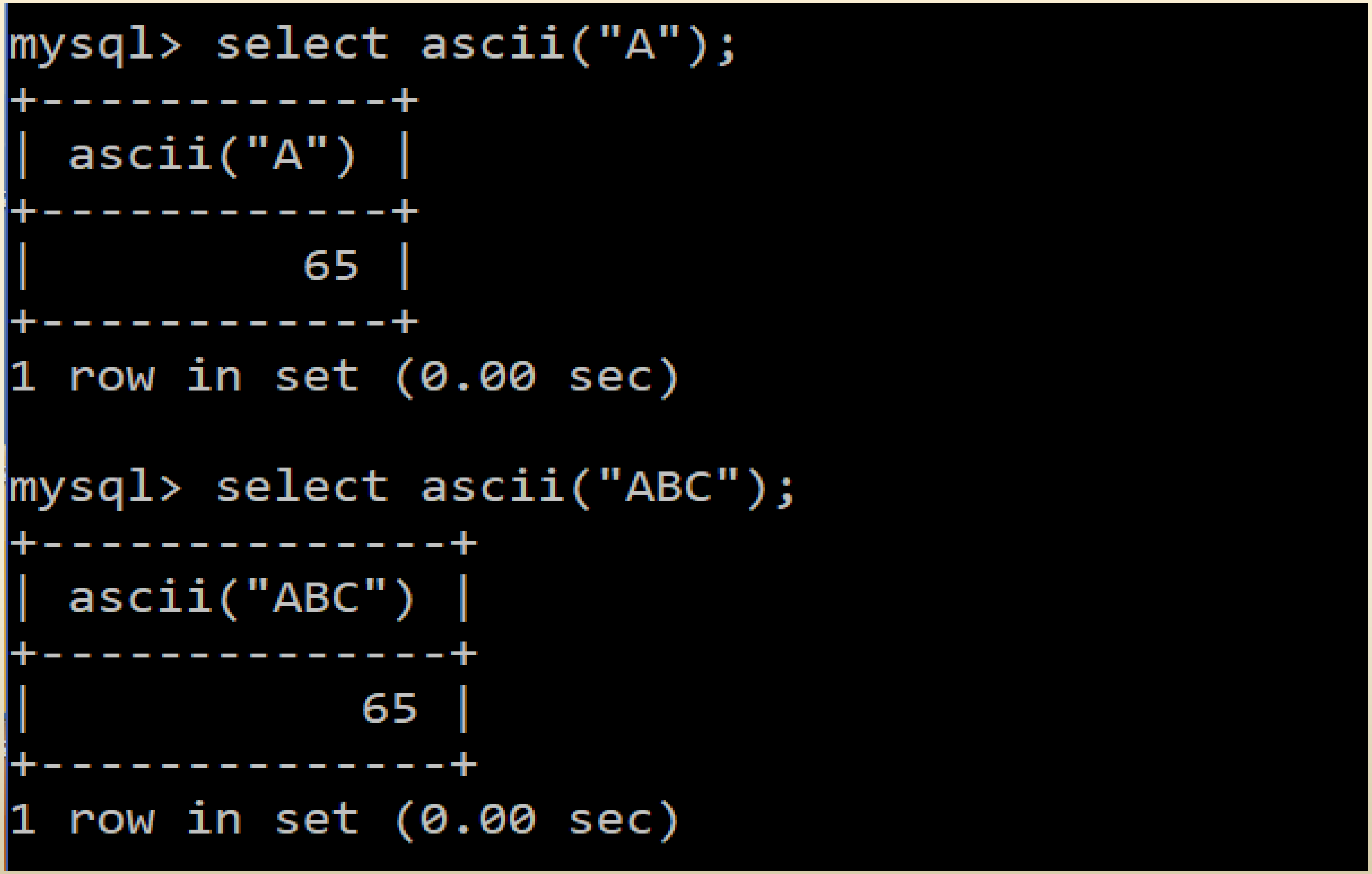


```
mysql> select trim("    informatics practices    ");
+---------------------------------------+
| trim("    informatics practices    ") |
+---------------------------------------+
| informatics practices                 |
+---------------------------------------+
1 row in set (0.00 sec)
```

Ascii()

Returns the numeric value of the leftmost character of the string str. Returns 0 if str is the empty string. Returns NULL if str is NULL. ASCII() works for characters with numeric values from 0 to 255.

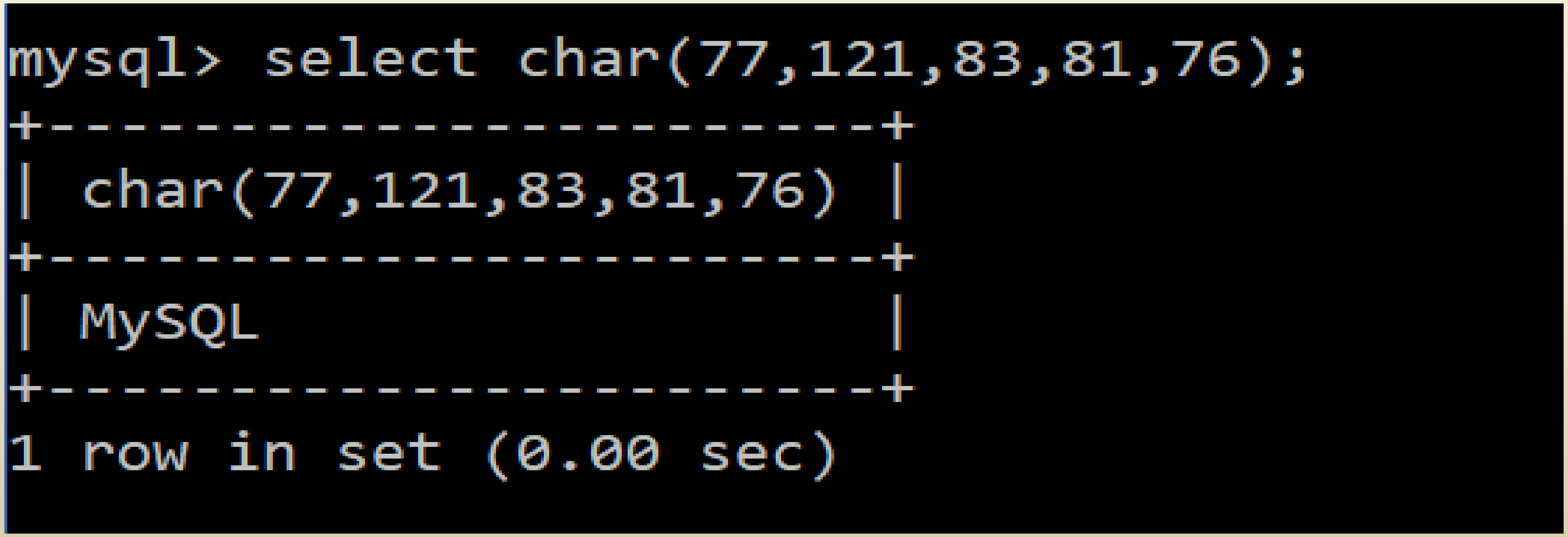
Example:



```
mysql> select ascii("A");
+------------+
| ascii("A") |
+------------+
|         65 |
+------------+
1 row in set (0.00 sec)

mysql> select ascii("ABC");
+--------------+
| ascii("ABC") |
+--------------+
|           65 |
+--------------+
1 row in set (0.00 sec)
```

CHAR()

CHAR() interprets each argument N as an integer and returns a string consisting of the characters given by the code values of those integers. NULL values are skipped.



```
mysql> select char(77,121,83,81,76);
+-----------------------+
| char(77,121,83,81,76) |
+-----------------------+
| MySQL                 |
+-----------------------+
1 row in set (0.00 sec)
```

Date and Time Functions

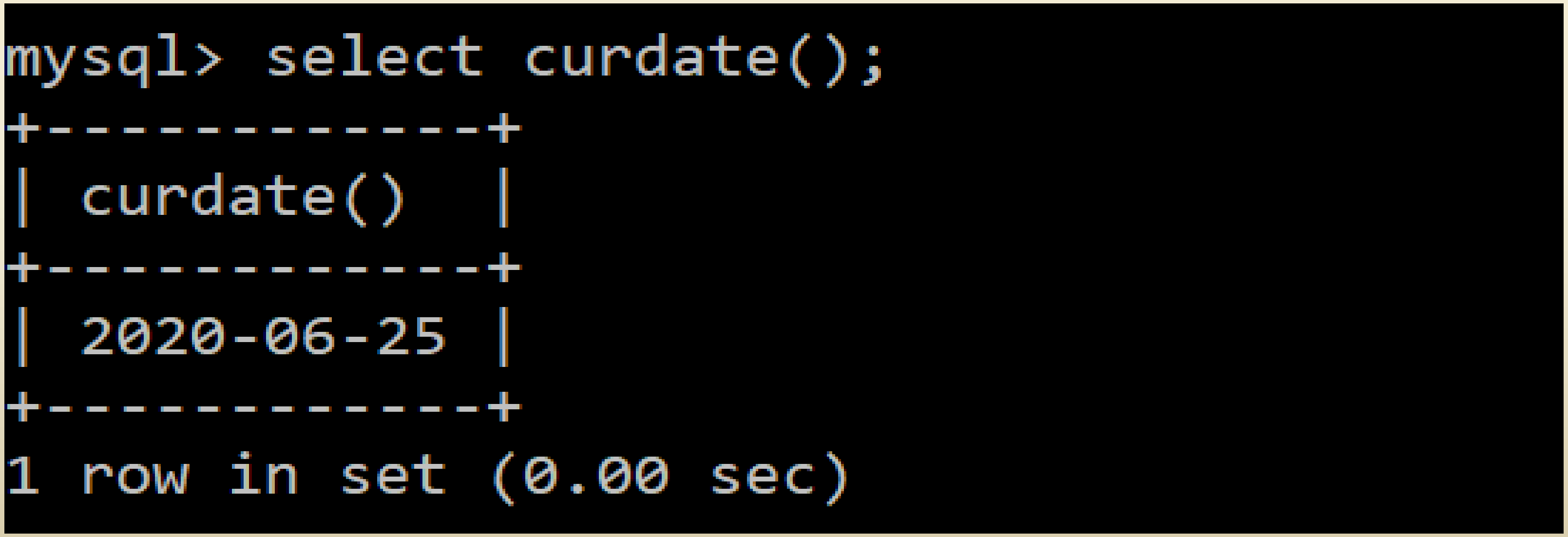
Date functions operate on values of the DATE and TIME data type:

Function are….

- Curdate()
- Day()
- Date()
- Month()
- Year()
- Dayname()
- Monthname()
- Dayofweek()
- Dayofmonth()
- Dayofyear()
- Sysdate()
- Now()

Curdate()

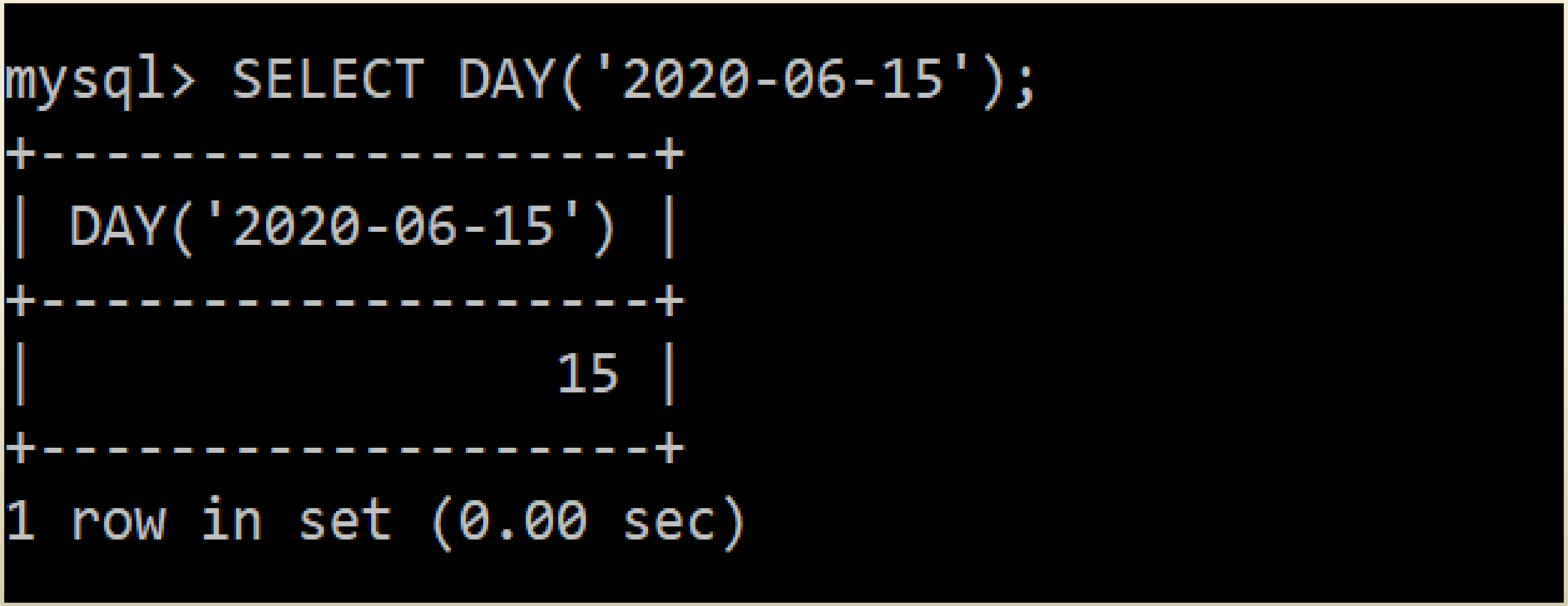
In MySQL the CURDATE() returns the current date in 'YYYY-MM-DD' format or 'YYYYMMDD' format depending on whether numeric or string is used in the function.



```
mysql> select curdate();
+------------+
| curdate()  |
+------------+
| 2020-06-25 |
+------------+
1 row in set (0.00 sec)
```

Day()

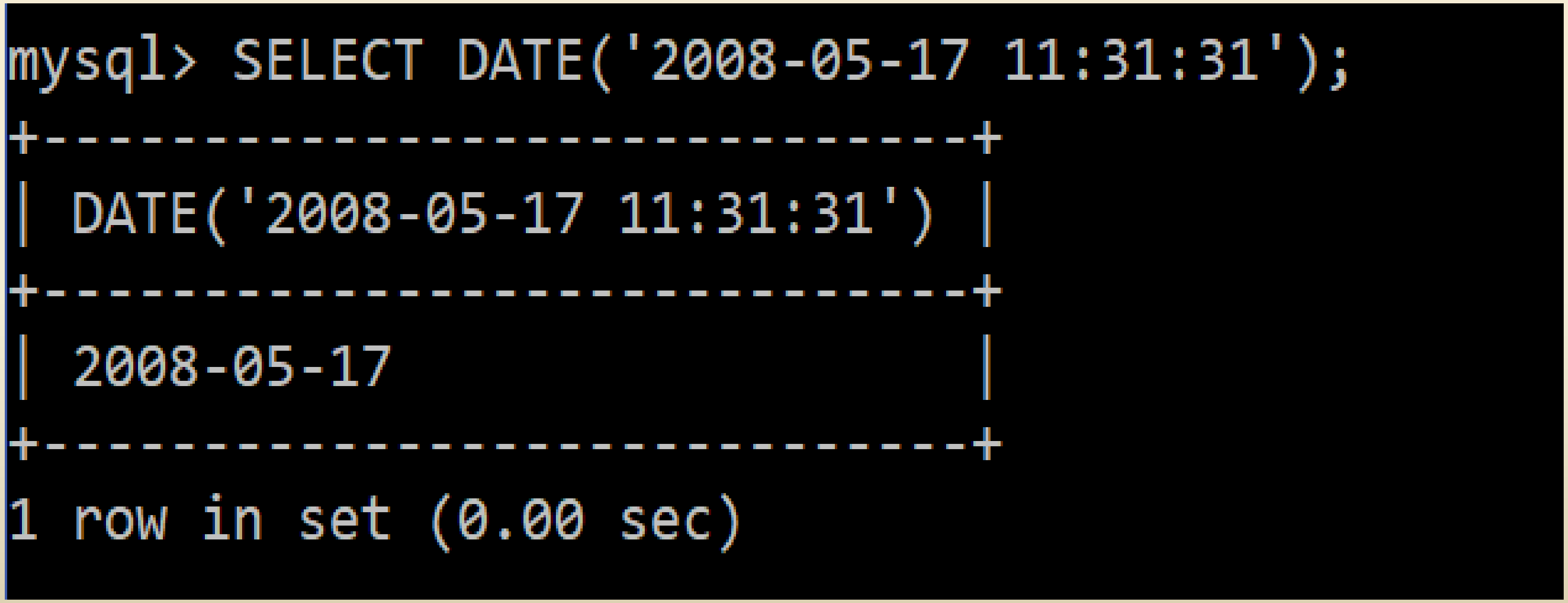
MySQL DAY() returns the day of the month for a specified date. The day returned will be within the range of 1 to 31.



```
mysql> SELECT DAY('2020-06-15');
+-------------------+
| DAY('2020-06-15') |
+-------------------+
|                15 |
+-------------------+
1 row in set (0.00 sec)
```

Date()

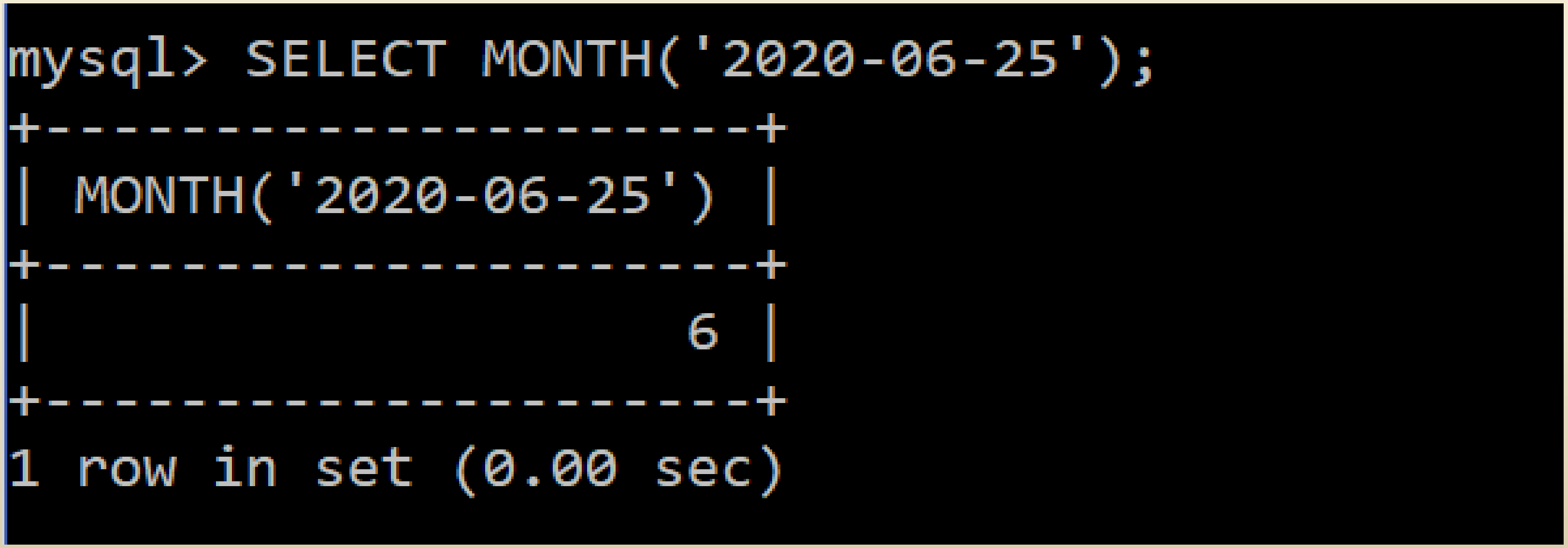
MySQL DATE() returns the DATE part out from a datetime expression.



```
mysql> SELECT DATE('2008-05-17 11:31:31');
+-----------------------------+
| DATE('2008-05-17 11:31:31') |
+-----------------------------+
| 2008-05-17                  |
+-----------------------------+
1 row in set (0.00 sec)
```

Month()

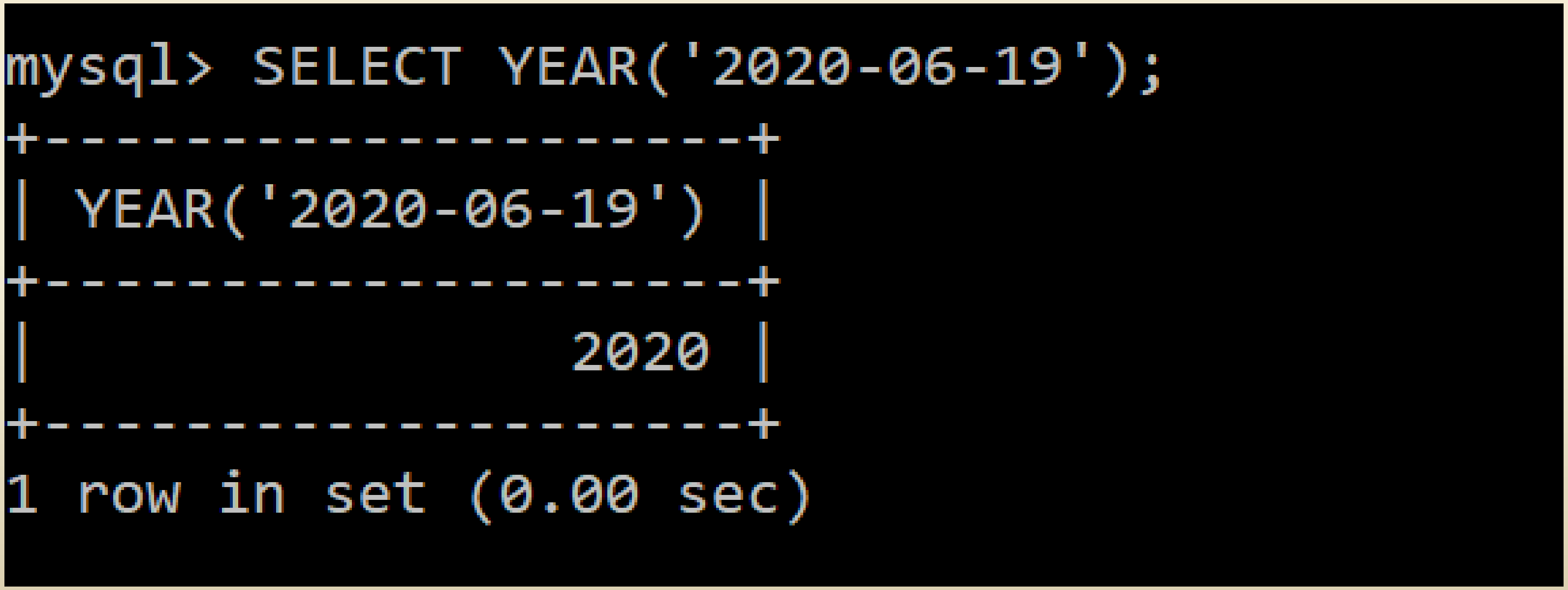
MySQL MONTH() returns the MONTH for the date within a range of 1 to 12 (January to December). It Returns 0 when MONTH part for the date is 0.



```
mysql> SELECT MONTH('2020-06-25');
+---------------------+
| MONTH('2020-06-25') |
+---------------------+
|                   6 |
+---------------------+
1 row in set (0.00 sec)
```

YEAR()

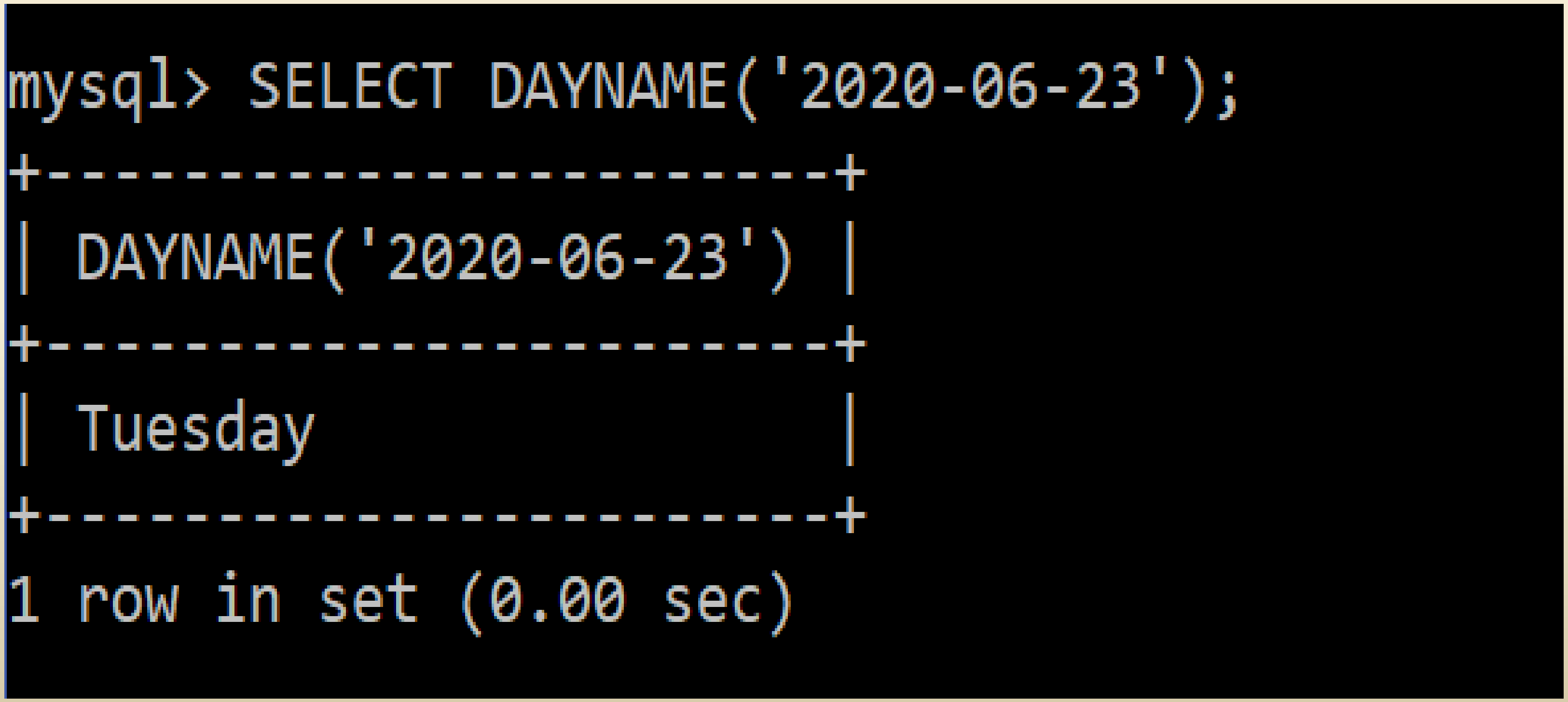
MySQL YEAR() returns the year for a given date. The return value is in the range of 1000 to 9999 or 0 for 'zero' date.



```
mysql> SELECT YEAR('2020-06-19');
+--------------------+
| YEAR('2020-06-19') |
+--------------------+
|               2020 |
+--------------------+
1 row in set (0.00 sec)
```

Dayname()

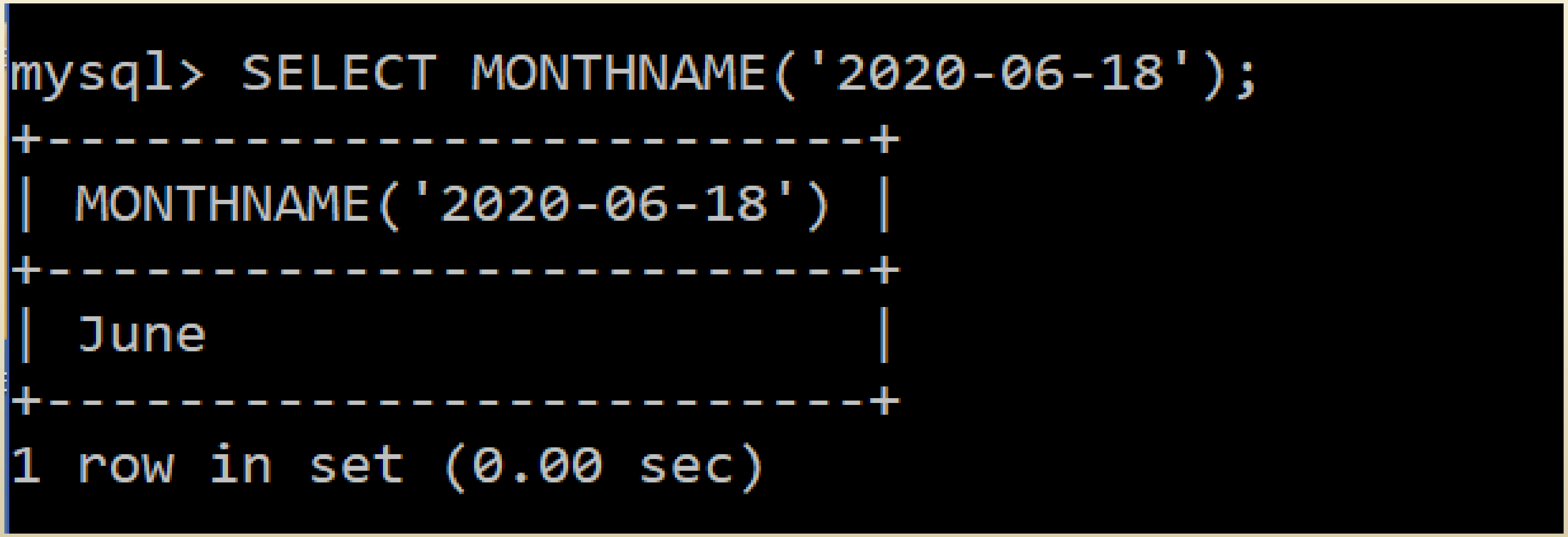
MySQL DAYNAME() returns the name of the week day of a date specified in the argument.



```
mysql> SELECT DAYNAME('2020-06-23');
+-----------------------+
| DAYNAME('2020-06-23') |
+-----------------------+
| Tuesday               |
+-----------------------+
1 row in set (0.00 sec)
```

Monthname()

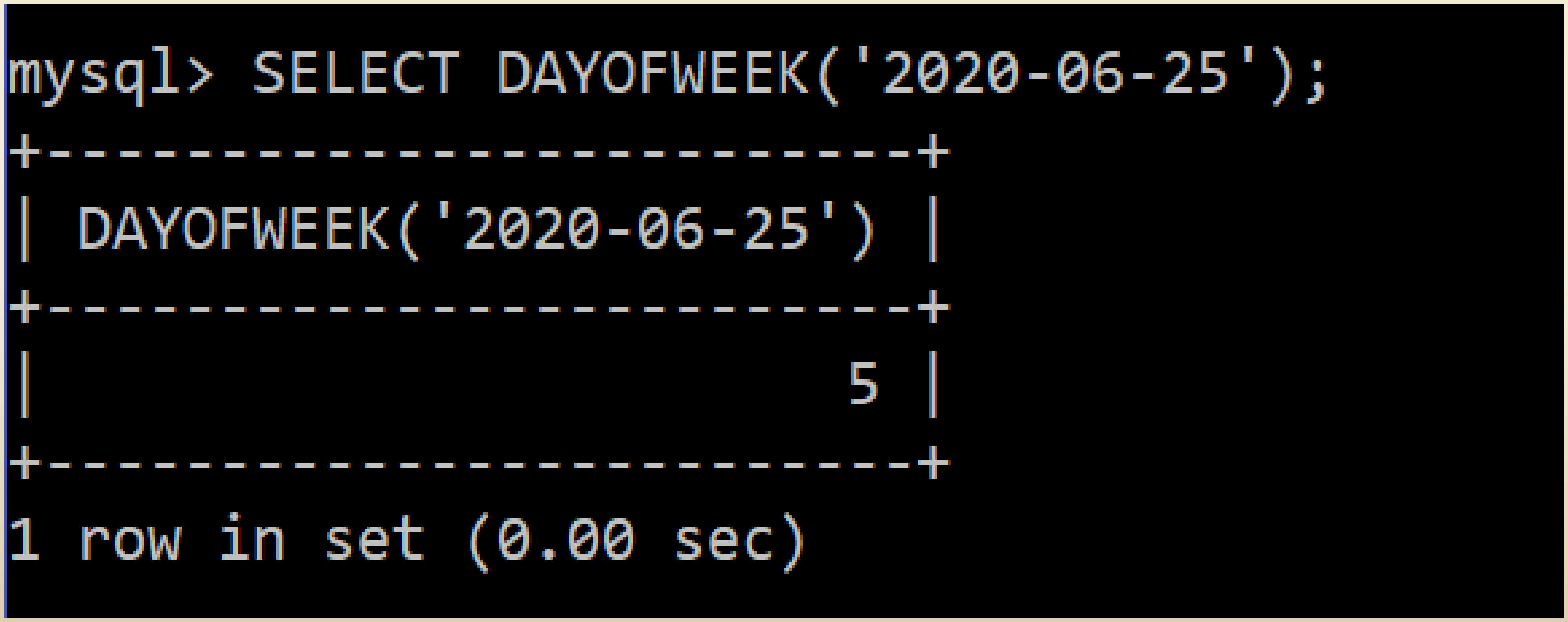
MySQL MONTHNAME() returns the full name of the month for a given date. The return value is within the range of 1 to 12 (January to December). It Returns NULL when month part for the date is 0 or more than 12.



```
mysql> SELECT MONTHNAME('2020-06-18');
+-------------------------+
| MONTHNAME('2020-06-18') |
+-------------------------+
| June                    |
+-------------------------+
1 row in set (0.00 sec)
```

Dayofweek()

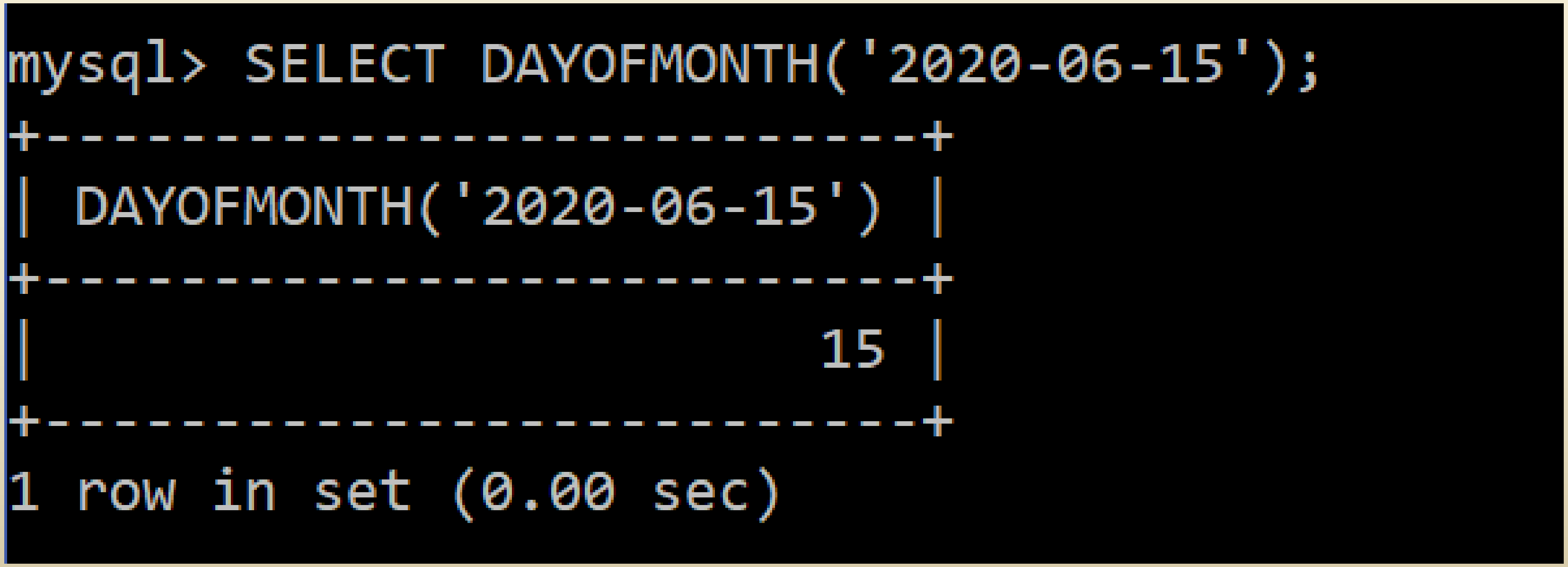
MySQL DAYOFWEEK() returns the week day number (1 for Sunday,2 for Monday …… 7 for Saturday) for a date specified as argument.



```
mysql> SELECT DAYOFWEEK('2020-06-25');
+-------------------------+
| DAYOFWEEK('2020-06-25') |
+-------------------------+
|                       5 |
+-------------------------+
1 row in set (0.00 sec)
```

Dayofmonth()

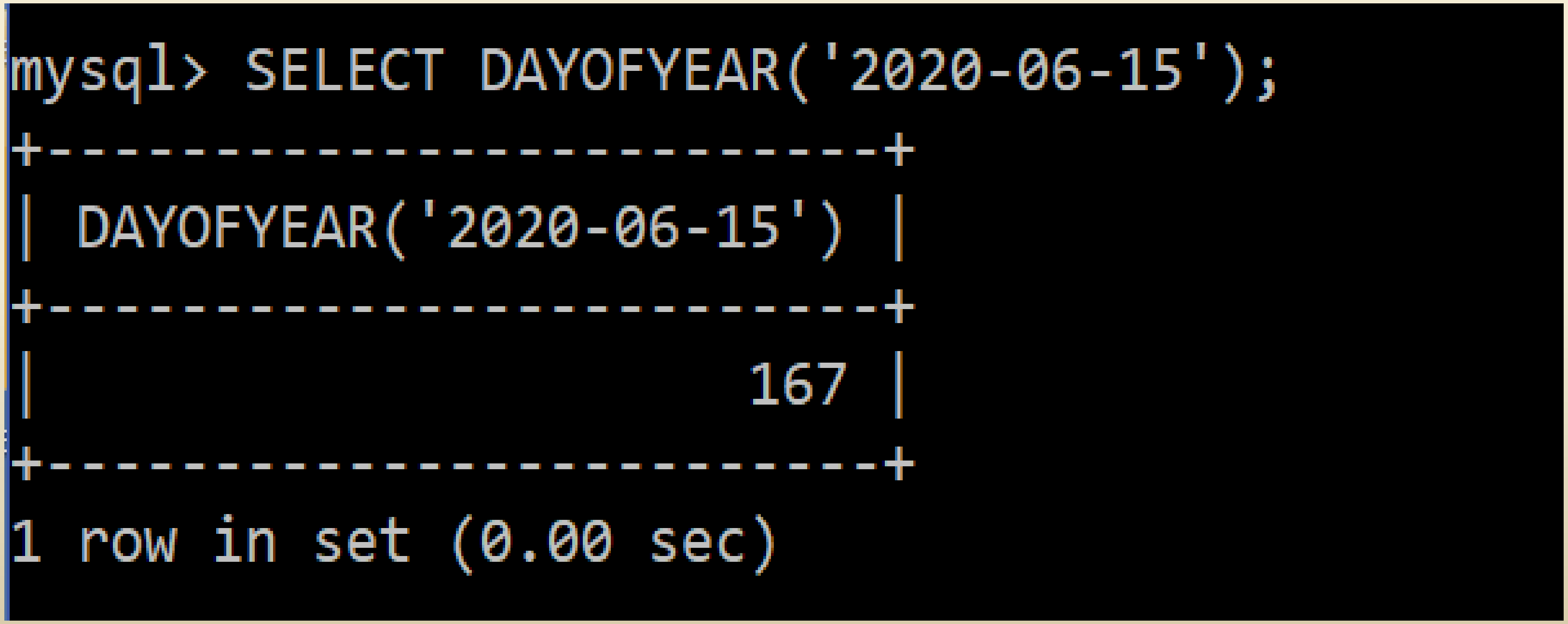
MySQL DAYOFMONTH() returns the day of the month for a given date. The day returned will be within the range of 1 to 31. If the date is ‘0000-00-00’, the function will return 0. The DAY() is the synonym of DAYOFMONTH().



```
mysql> SELECT DAYOFMONTH('2020-06-15');
+--------------------------+
| DAYOFMONTH('2020-06-15') |
+--------------------------+
|                       15 |
+--------------------------+
1 row in set (0.00 sec)
```

Dayofyear()

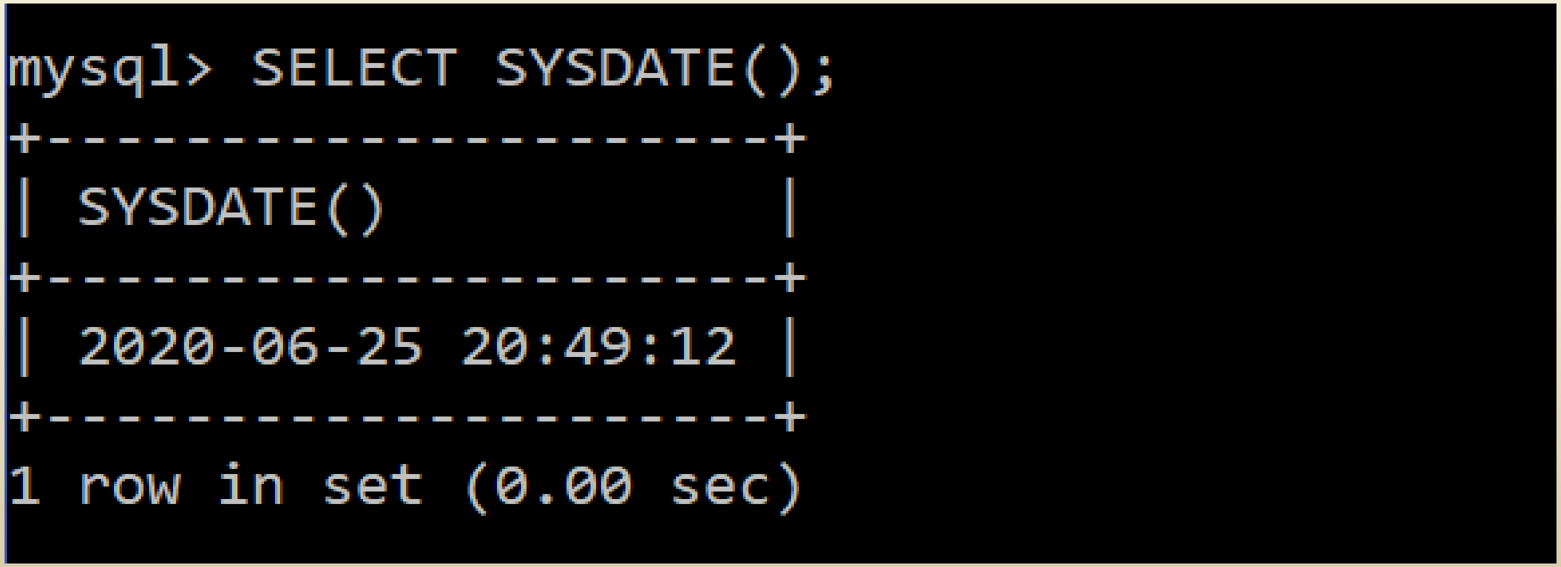
MySQL DAYOFYEAR() returns day of the year for a date. The return value is within the range of 1 to 366.



```
mysql> SELECT DAYOFYEAR('2020-06-15');
+-------------------------+
| DAYOFYEAR('2020-06-15') |
+-------------------------+
|                     167 |
+-------------------------+
1 row in set (0.00 sec)
```

Sysdate()

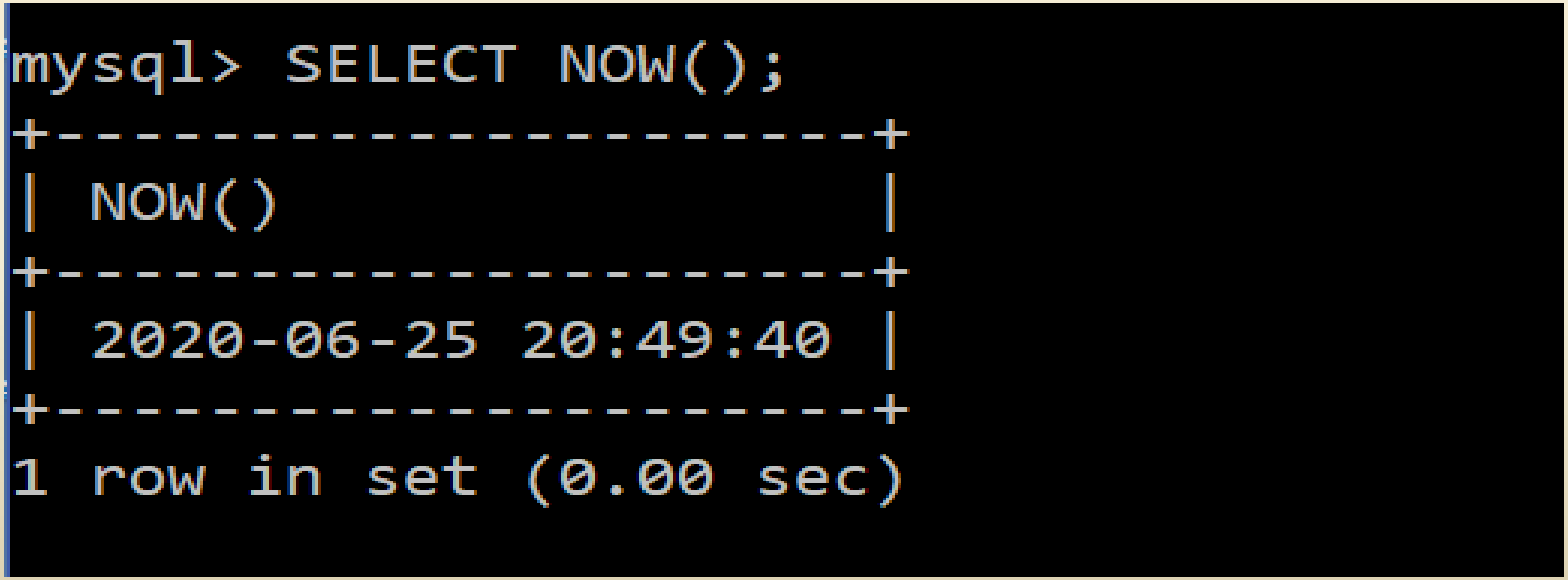
MySQL SYSDATE() returns the current date and time in YYYY-MM-DD HH:MM:SS or YYYYMMDDHHMMSS. format depending on the context of the function.



```
mysql> SELECT SYSDATE();
+---------------------+
| SYSDATE()           |
+---------------------+
| 2020-06-25 20:49:12 |
+---------------------+
1 row in set (0.00 sec)
```

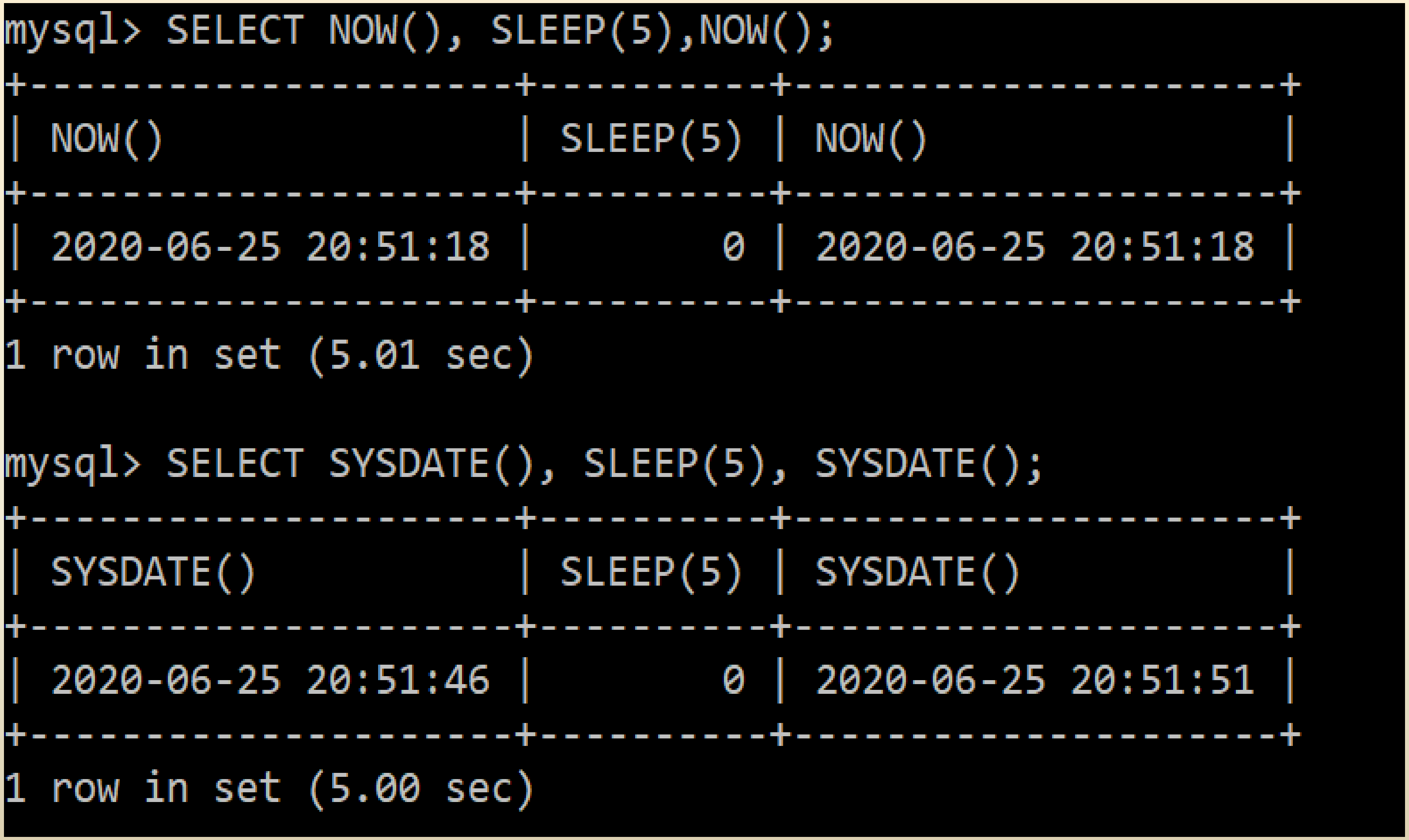
Now()

MySQL NOW() returns the value of current date and time in 'YYYY-MM-DD HH:MM:SS' format or YYYYMMDDHHMMSS. format depending on the context (numeric or string) of the function.



```
mysql> SELECT NOW();
+---------------------+
| NOW()               |
+---------------------+
| 2020-06-25 20:49:40 |
+---------------------+
1 row in set (0.00 sec)
```

Difference b/w Now() & Sysdate()



```
mysql> SELECT NOW(), SLEEP(5),NOW();
+---------------------+----------+---------------------+
| NOW()               | SLEEP(5) | NOW()               |
+---------------------+----------+---------------------+
| 2020-06-25 20:51:18 |        0 | 2020-06-25 20:51:18 |
+---------------------+----------+---------------------+
1 row in set (5.01 sec)

mysql> SELECT SYSDATE(), SLEEP(5), SYSDATE();
+---------------------+----------+---------------------+
| SYSDATE()           | SLEEP(5) | SYSDATE()           |
+---------------------+----------+---------------------+
| 2020-06-25 20:51:46 |        0 | 2020-06-25 20:51:51 |
+---------------------+----------+---------------------+
1 row in set (5.00 sec)
```

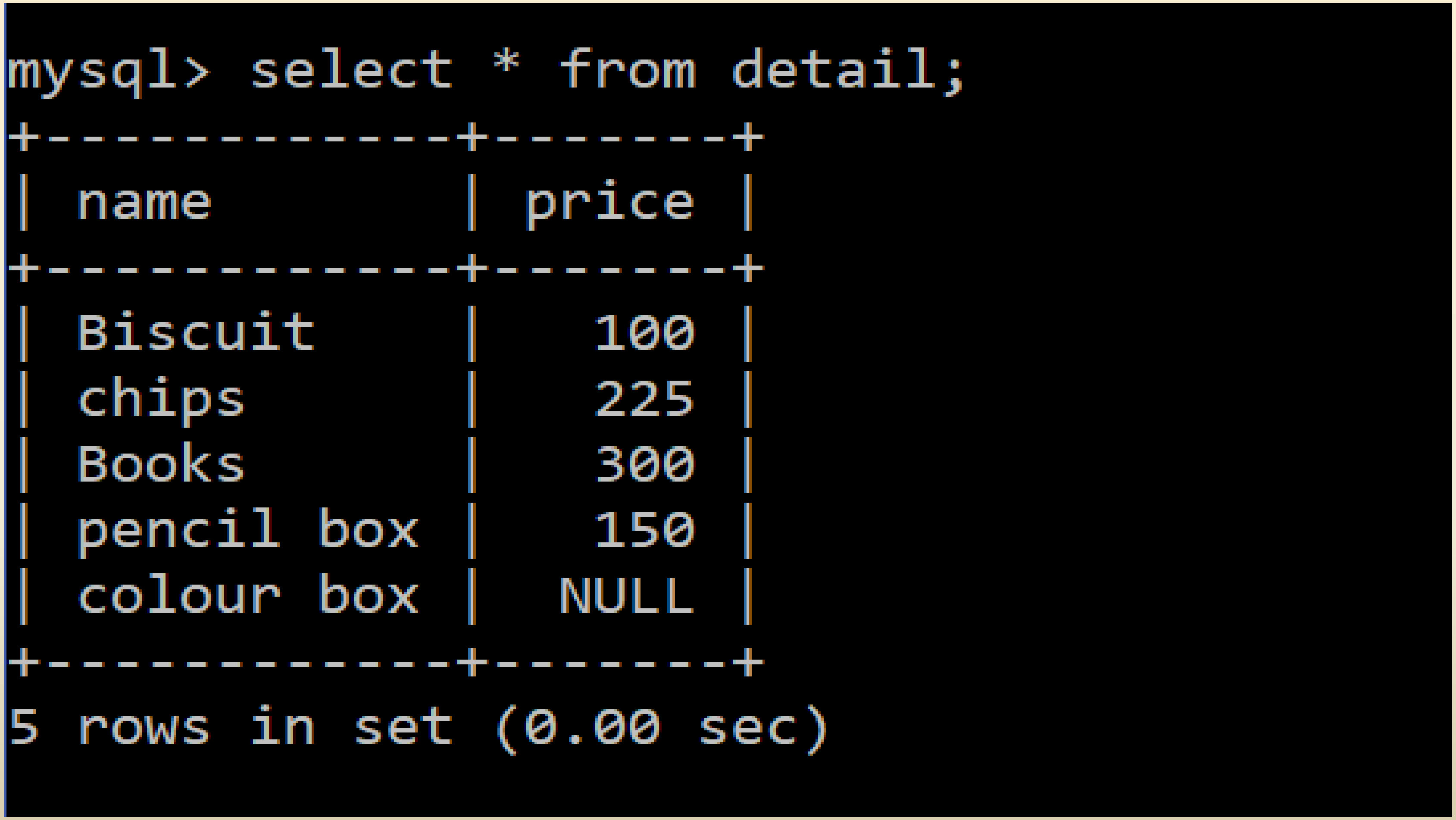
Aggregate functions

An aggregate function performs a calculation on multiple values and returns a single value.

Functions are…

- Min()
- Max()
- Avg()
- Count()
- Sum()

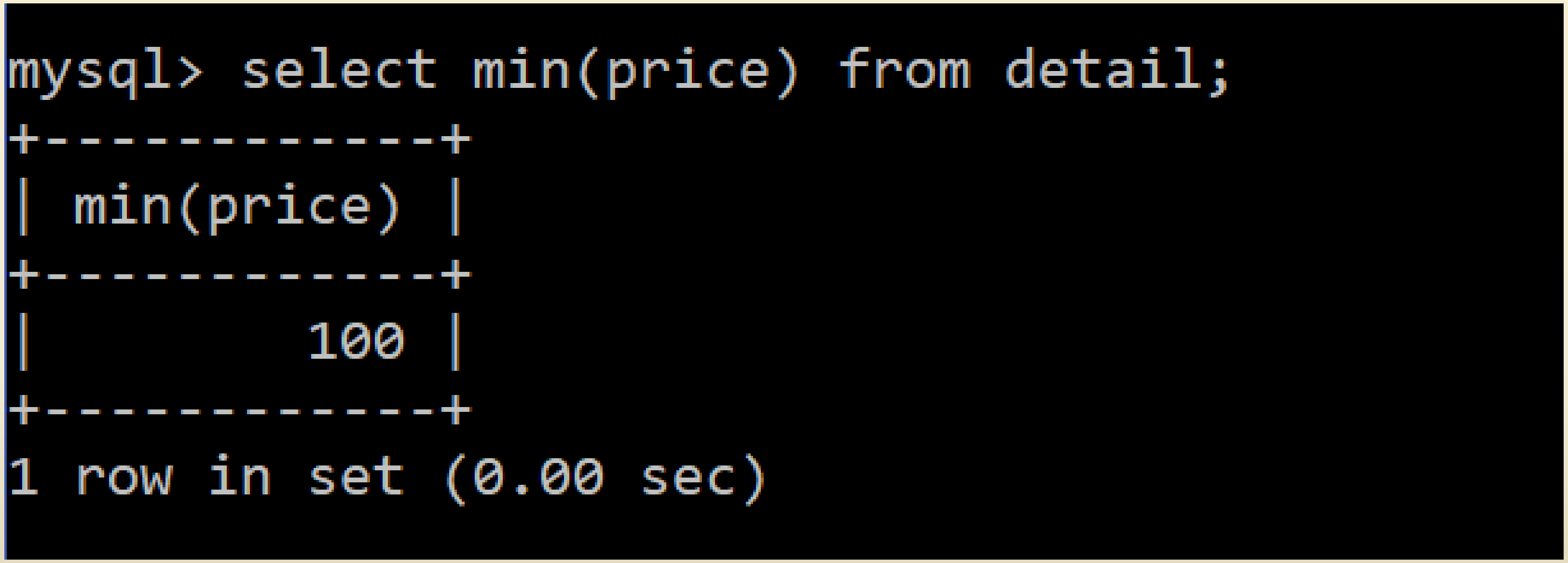
Table used for Aggregate function



```
mysql> select * from detail;
+-------------+--------+
| name        | price  |
+-------------+--------+
| Biscuit     |    100 |
| chips       |    225 |
| Books       |    300 |
| pencil box  |    150 |
| colour box  |   NULL |
+-------------+--------+
5 rows in set (0.00 sec)
```

Min()

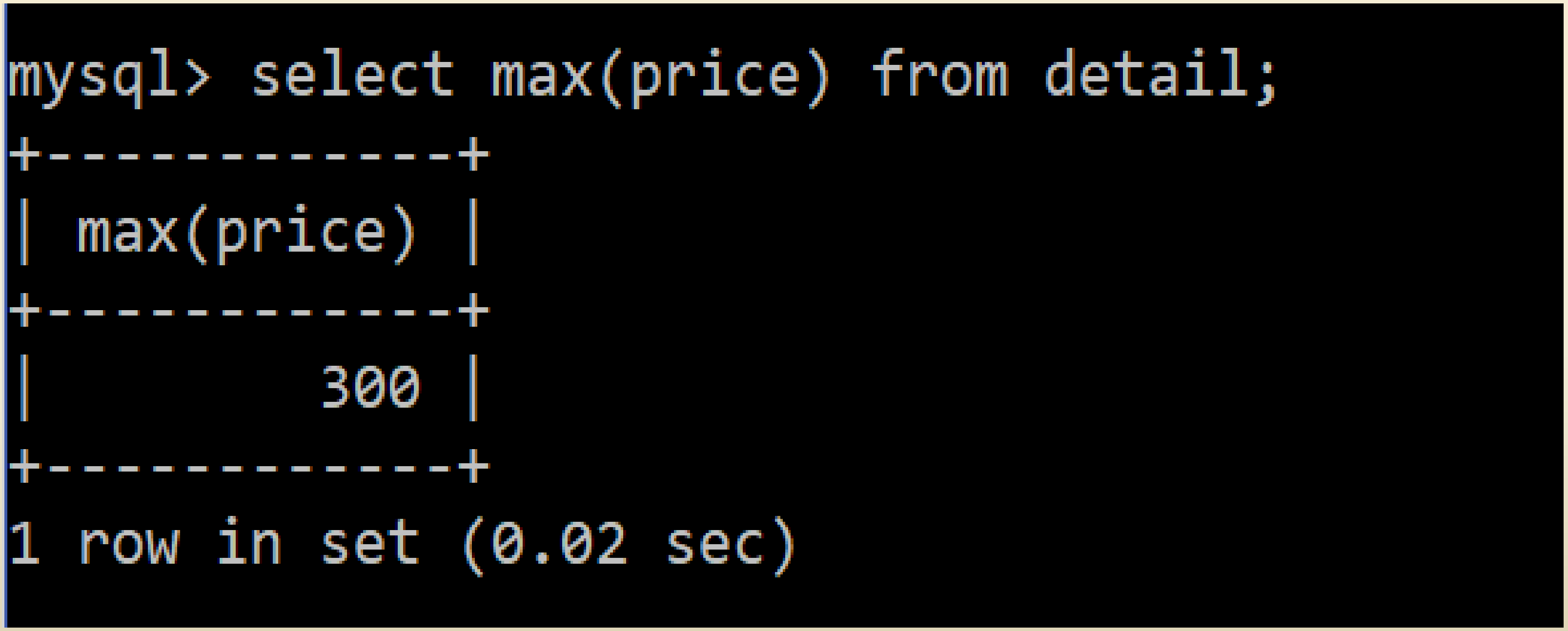
Return the lowest value (minimum) in a set of non-NULL values.



```
mysql> select min(price) from detail;
+-------------+
| min(price) |
+-------------+
|         100 |
+-------------+
1 row in set (0.00 sec)
```

Max()

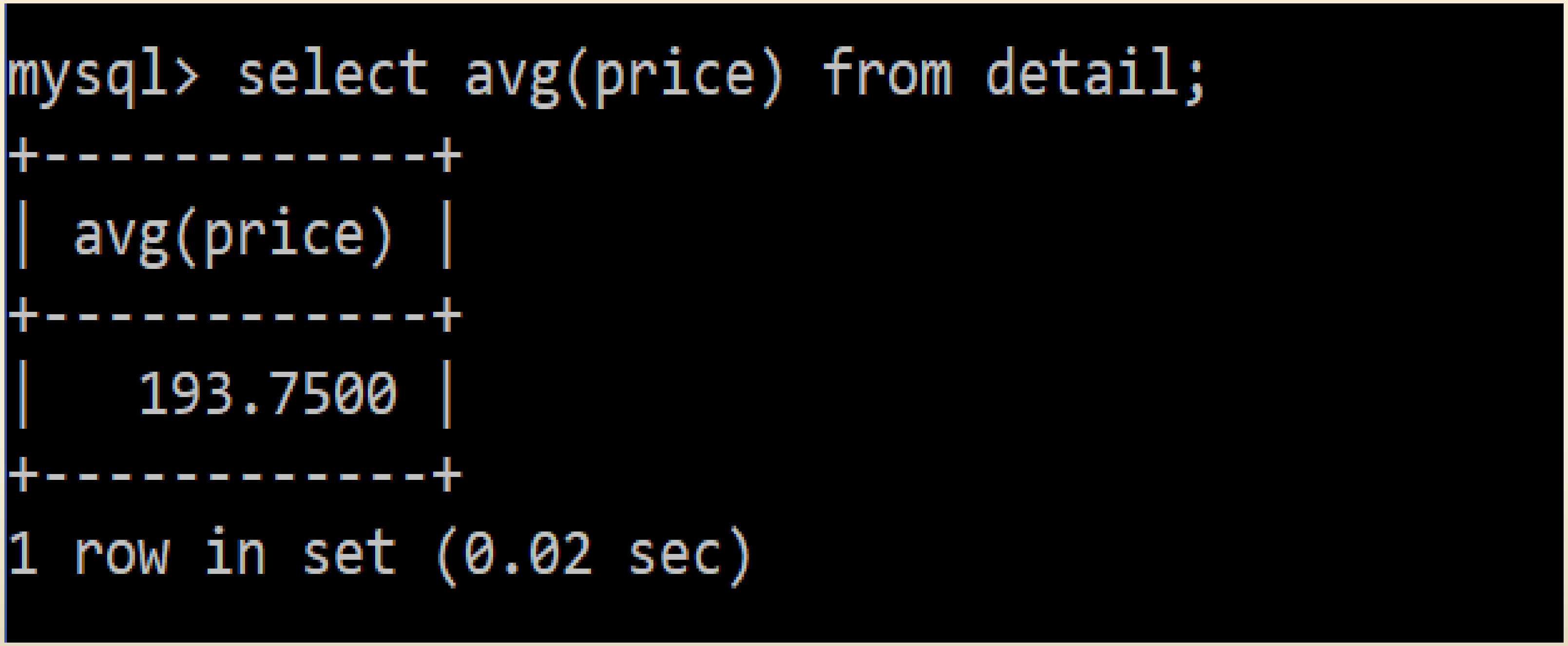
Return the highest value (maximum) in a set of non-NULL values.



```
mysql> select max(price) from detail;
+------------+
| max(price) |
+------------+
|        300 |
+------------+
1 row in set (0.02 sec)
```

Avg()

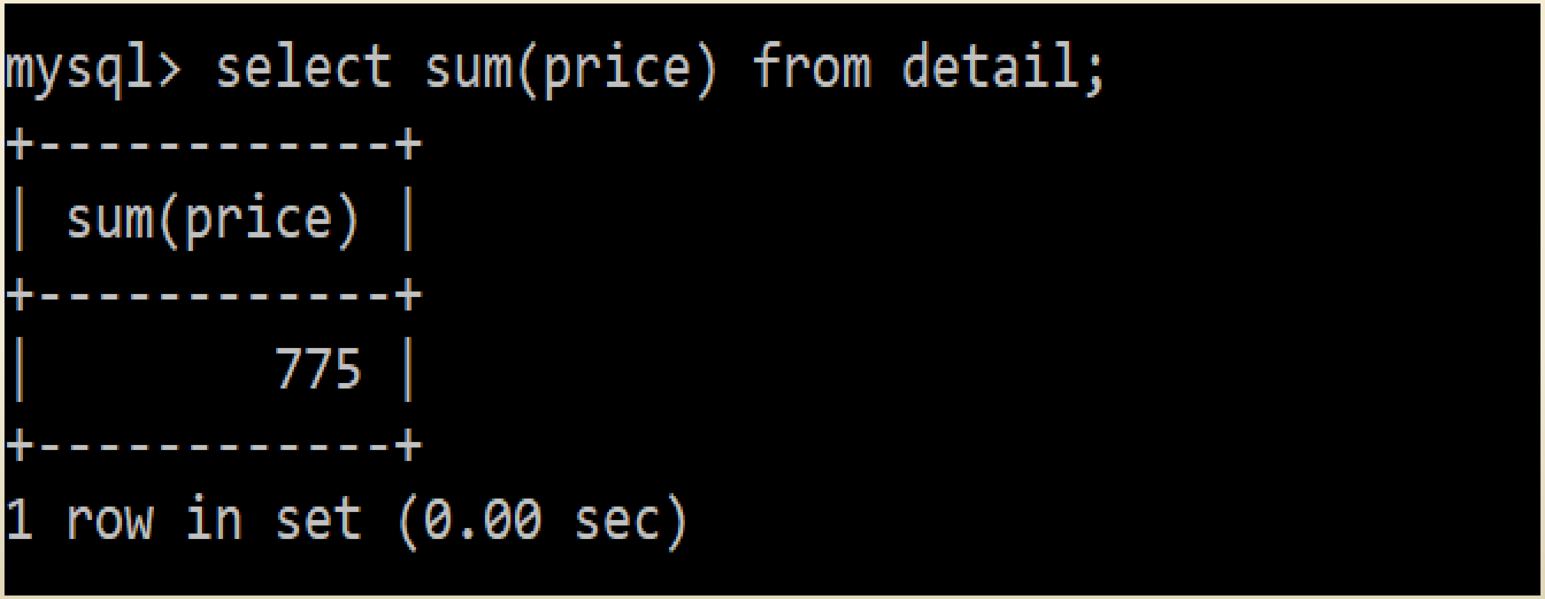
Return the average of non-NULL values.



```
mysql> select avg(price) from detail;
+------------+
| avg(price) |
+------------+
|   193.7500 |
+------------+
1 row in set (0.02 sec)
```

Sum()

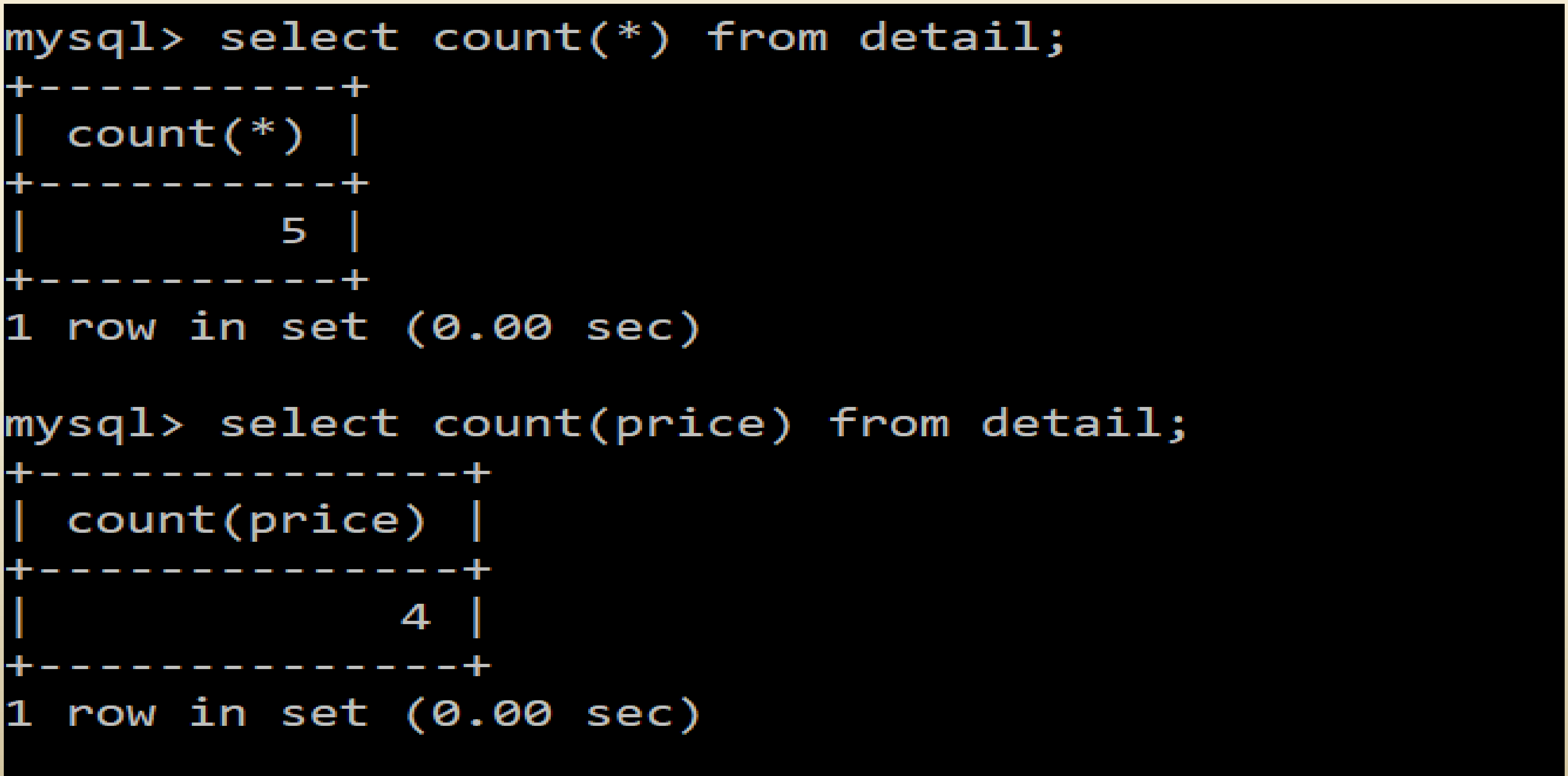
Return the summation of all non-NULL values a set.



```
mysql> select sum(price) from detail;
+------------+
| sum(price) |
+------------+
|        775 |
+------------+
1 row in set (0.00 sec)
```

Count()

Return the number of rows in a group, including rows with NULL values. But when we apply on single value it will not consider null values.



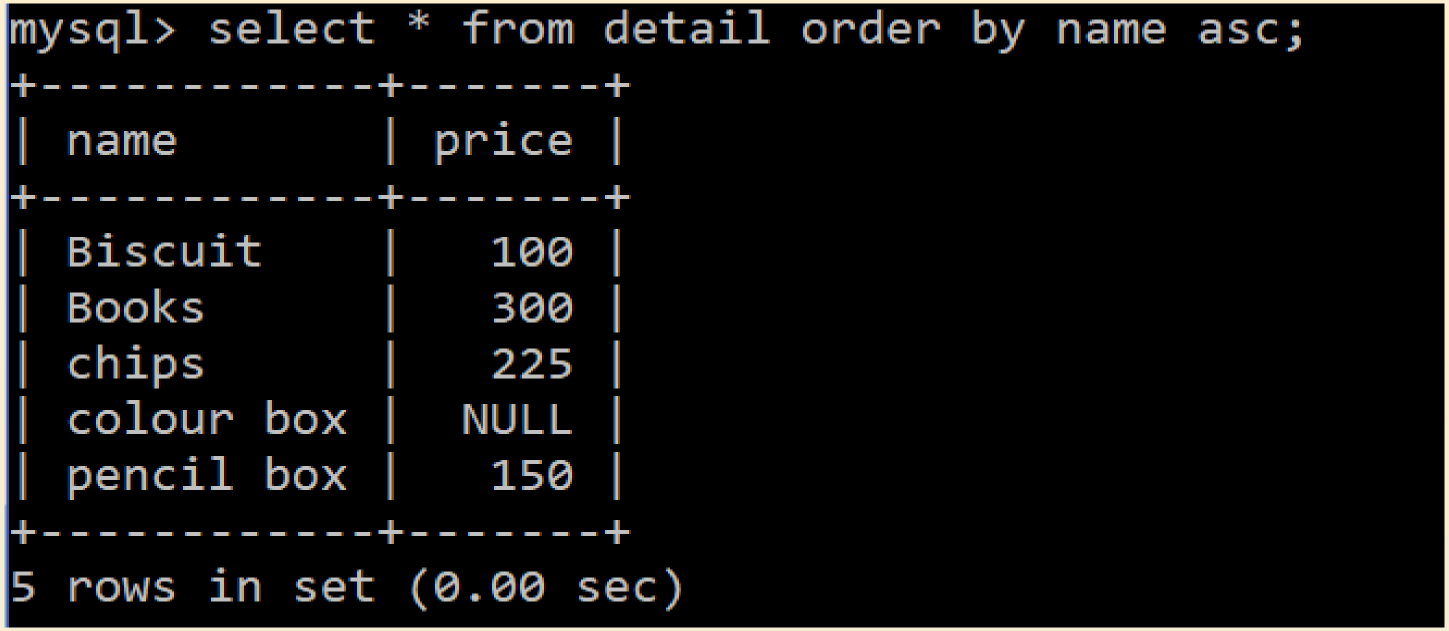
```
mysql> select count(*) from detail;
+----------+
| count(*) |
+----------+
|        5 |
+----------+
1 row in set (0.00 sec)

mysql> select count(price) from detail;
+--------------+
| count(price) |
+--------------+
|            4 |
+--------------+
1 row in set (0.00 sec)
```

ORDER BY Clause

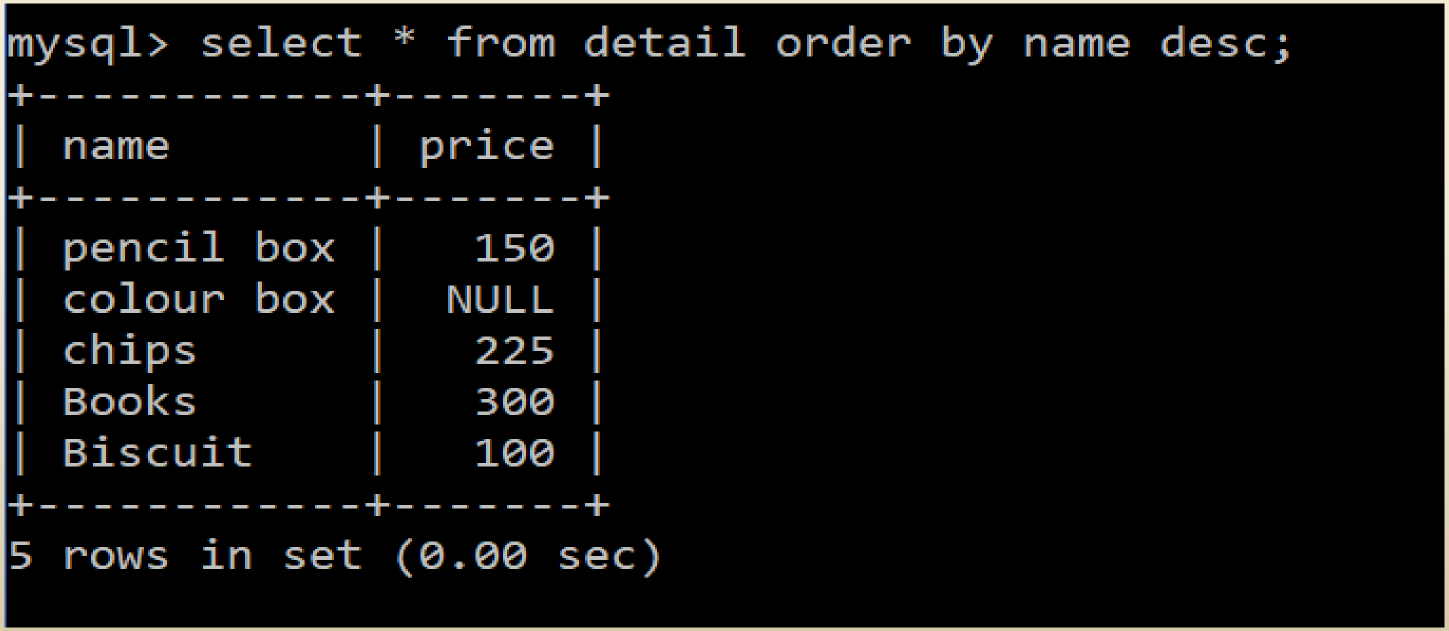
- **ORDER BY** clause is used to display the result of a query in a specific order(sorted order).

- The sorting can be done in ascending or in descending order. It should be kept in mind that the actual data in the database is not sorted but only the results of the query are displayed in sorted order.

- **Note:- If order is not specifies that by default the sorting will be performed in ascending order.**



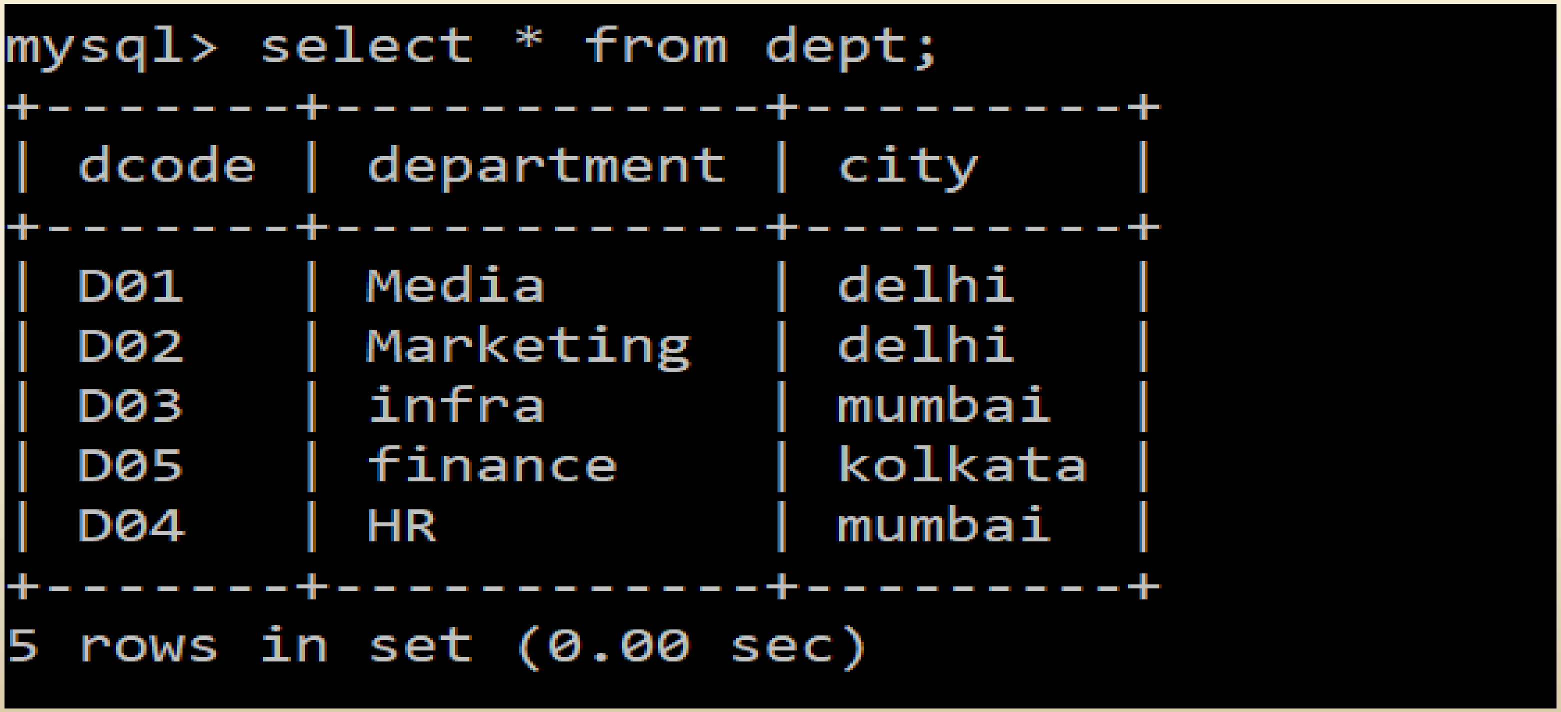
```
mysql> select * from detail order by name asc;
+------------+-------+
| name       | price |
+------------+-------+
| Biscuit    |   100 |
| Books      |   300 |
| chips      |   225 |
| colour box |  NULL |
| pencil box |   150 |
+------------+-------+
5 rows in set (0.00 sec)
```



```
mysql> select * from detail order by name desc;
+------------+-------+
| name       | price |
+------------+-------+
| pencil box |   150 |
| colour box |  NULL |
| chips      |   225 |
| Books      |   300 |
| Biscuit    |   100 |
+------------+-------+
5 rows in set (0.00 sec)
```

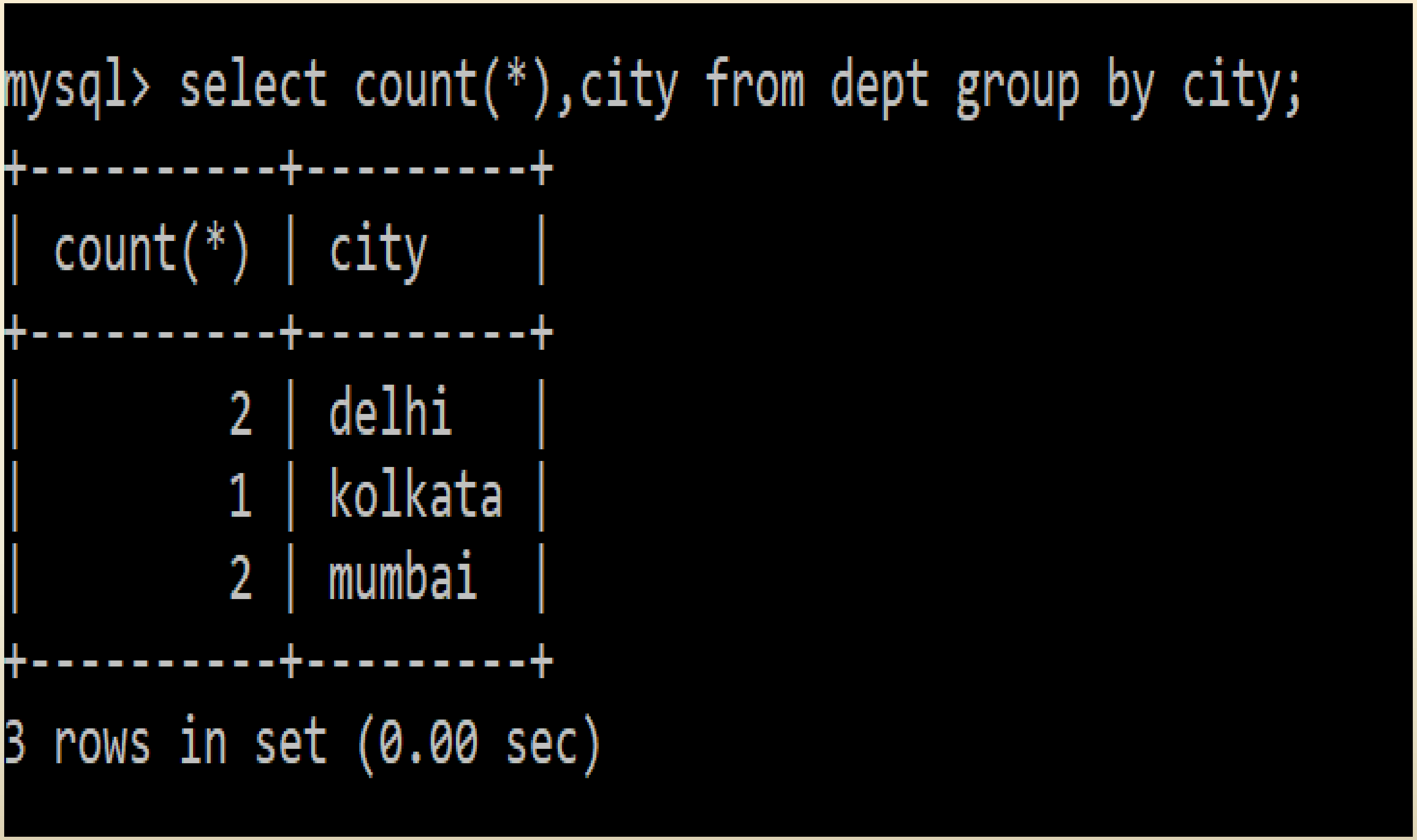
GROUP BY

- The GROUP BY clause can be used in a SELECT statement to collect data across multiple records and group the results by one or more columns.



```
mysql> select * from dept;
+-------+-------------+----------+
| dcode | department  | city     |
+-------+-------------+----------+
| D01   | Media       | delhi    |
| D02   | Marketing   | delhi    |
| D03   | infra       | mumbai   |
| D05   | finance     | kolkata  |
| D04   | HR          | mumbai   |
+-------+-------------+----------+
5 rows in set (0.00 sec)
```

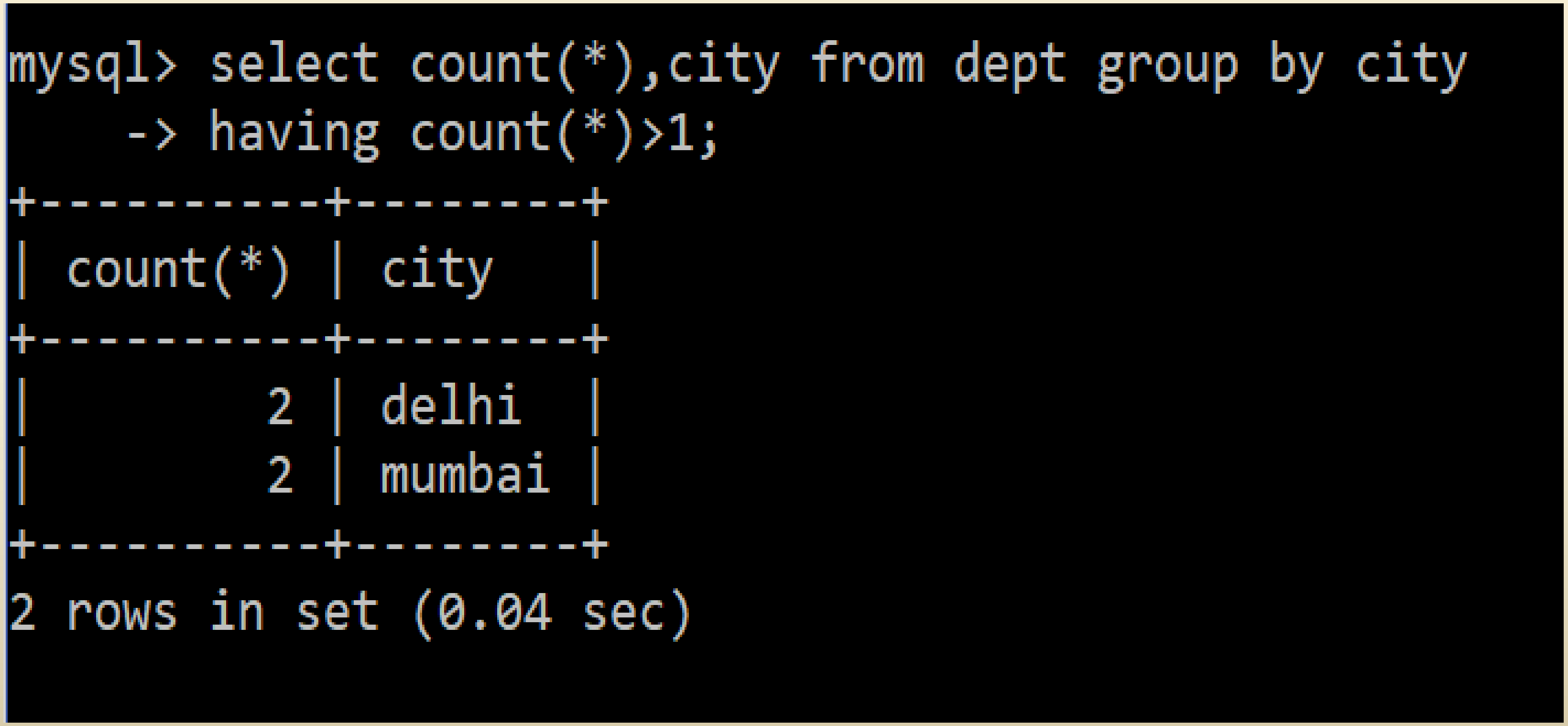
Example :



```
mysql> select count(*),city from dept group by city;
+----------+---------+
| count(*) | city    |
+----------+---------+
|        2 | delhi   |
|        1 | kolkata |
|        2 | mumbai  |
+----------+---------+
3 rows in set (0.00 sec)
```

HAVING Clause

- The **HAVING** clause is used in combination with the GROUP BY clause. It can be used in a **SELECT** statement to filter the records that a **GROUP BY** returns.



```
mysql> select count(*),city from dept group by city
    -> having count(*)>1;
+----------+--------+
| count(*) | city   |
+----------+--------+
|        2 | delhi  |
|        2 | mumbai |
+----------+--------+
2 rows in set (0.04 sec)
```

THANKS