Using Basic Functions in MySQL - II

Session 14

Objectives

- *Describe the use of Date functions in MySQL*
- *Describe the use of String functions in MySQL*
- *Describe the use of System Information functions in MySQL*

ADDDATE Function

- The ADDDATE function adds two date expressions

- The syntax for adding two date expressions is:

```
SELECT ADDDATE(expr1,expr2);
```

where,

ADDDATE – calculates the date

expr1 – specifies a date

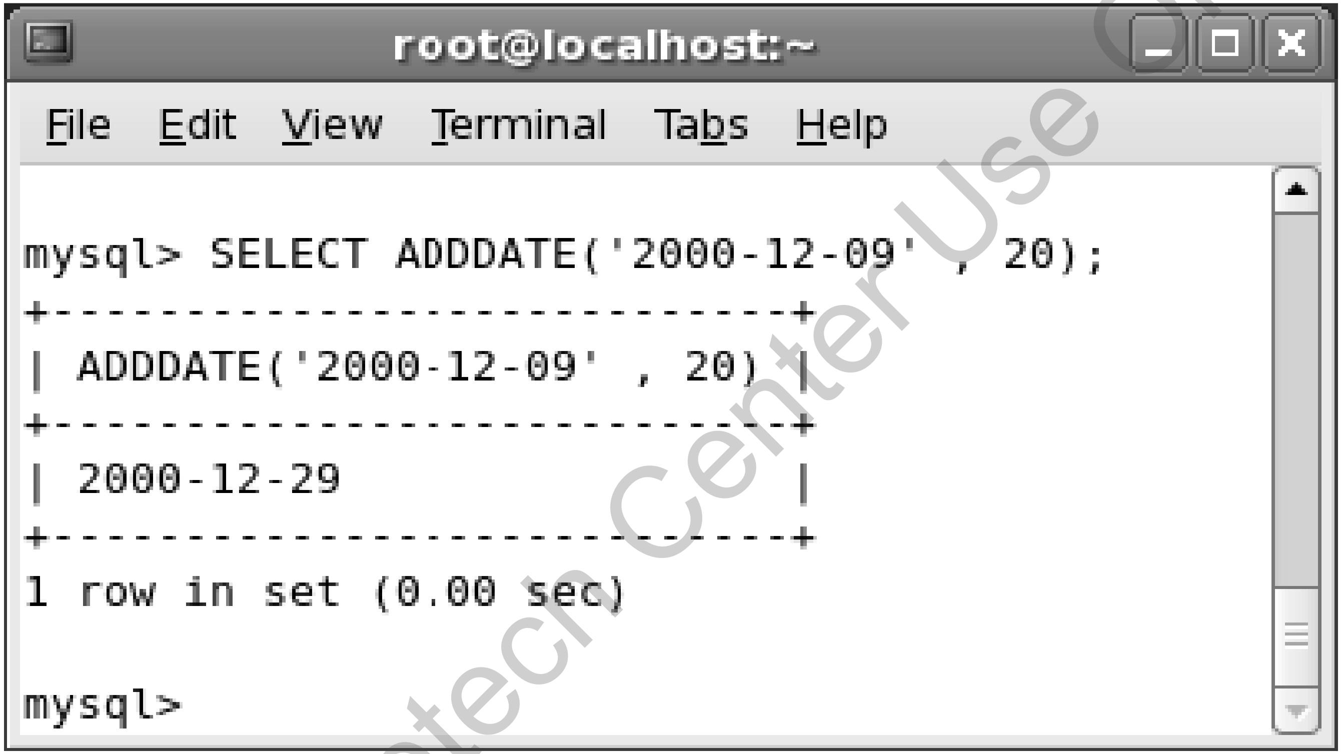
expr2 – defines the date to be added

- To add 20 days to the date 2000-12-09, enter the following command at the command prompt:

```
SELECT ADDDATE(‘2000-12-09’, 20);
```

ADDDATE Function

Figure displays the output of the command



```
mysql> SELECT ADDDATE('2000-12-09' , 20);
+-----------------------------+
| ADDDATE('2000-12-09' , 20) |
+-----------------------------+
| 2000-12-29                  |
+-----------------------------+
1 row in set (0.00 sec)

mysql>
```

ADDTIME Function 1-2

- The ADDTIME function is used to add two time expressions.
- The syntax for using this function is:

```
SELECT ADDTIME(expr1, expr2);
```

where,

ADDTIME – calculates the time

expr1 – defines a time or datetime expression

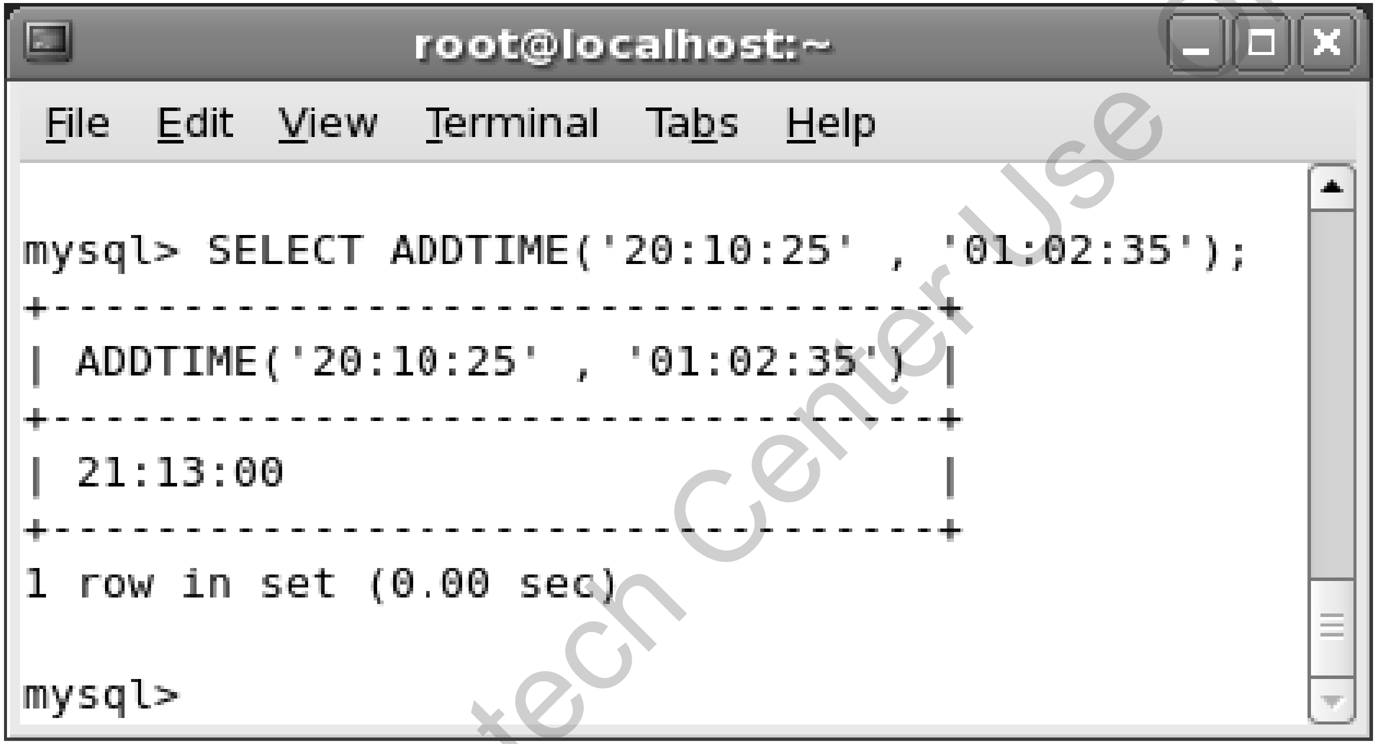
expr2 – specifies a time expression

- To calculate the addition of 20:10:25 and 01:02:35, enter the following command at the command prompt:

```
SELECT ADDTIME(‘20:10:25’,’01:02:35’);
```

ADDTIME Function

Figure displays the output of the command



```
mysql> SELECT ADDTIME('20:10:25' , '01:02:35');
+-----------------------------------+
| ADDTIME('20:10:25' , '01:02:35') |
+-----------------------------------+
| 21:13:00                          |
+-----------------------------------+
1 row in set (0.00 sec)

mysql>
```

CURDATE Function

- The CURDATE function returns the current date in the YYYY-MM-DD or YYYYMMDD format.
- The syntax to retrieve the current date is:

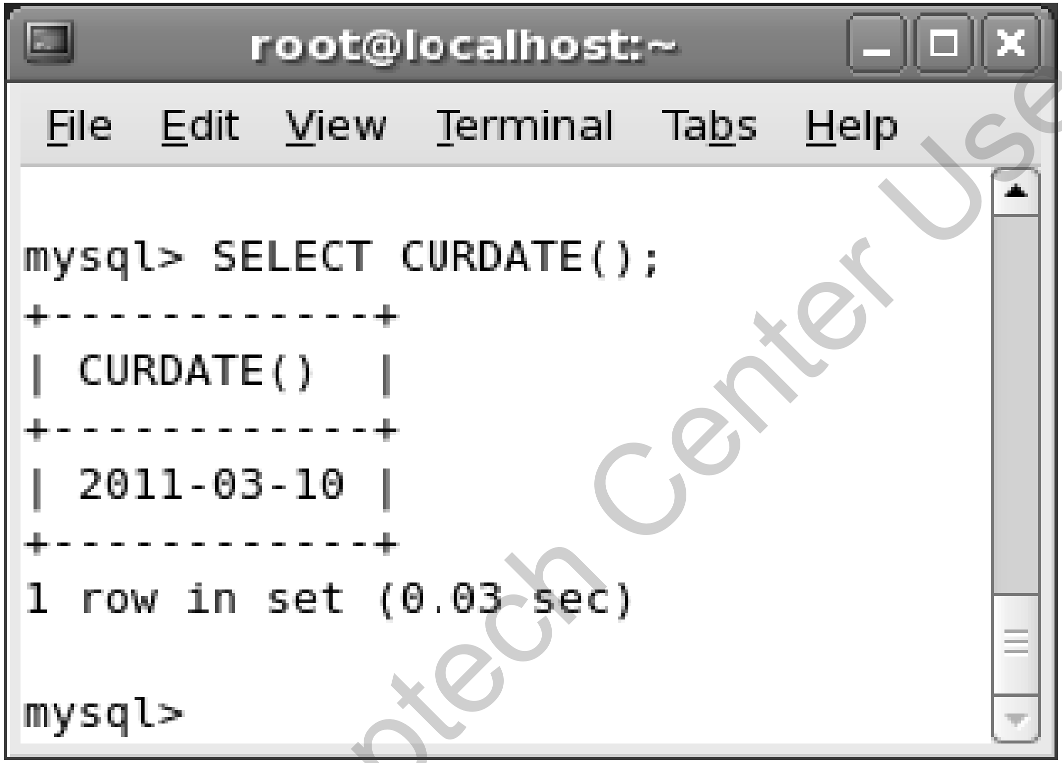
```
SELECT CURDATE();
```

- For example, to view the current date, enter the following command at the command prompt:

```
SELECT CURDATE();
```

CURDATE Function

Figure displays the output of the command



```
mysql> SELECT CURDATE();
+------------+
| CURDATE()  |
+------------+
| 2011-03-10 |
+------------+
1 row in set (0.03 sec)

mysql>
```

CURTIME Function

- The CURTIME function displays the current time
- The syntax to view the current time is:

```
SELECT CURTIME();
```

- For example, to obtain the current time, enter the following command at the command prompt:

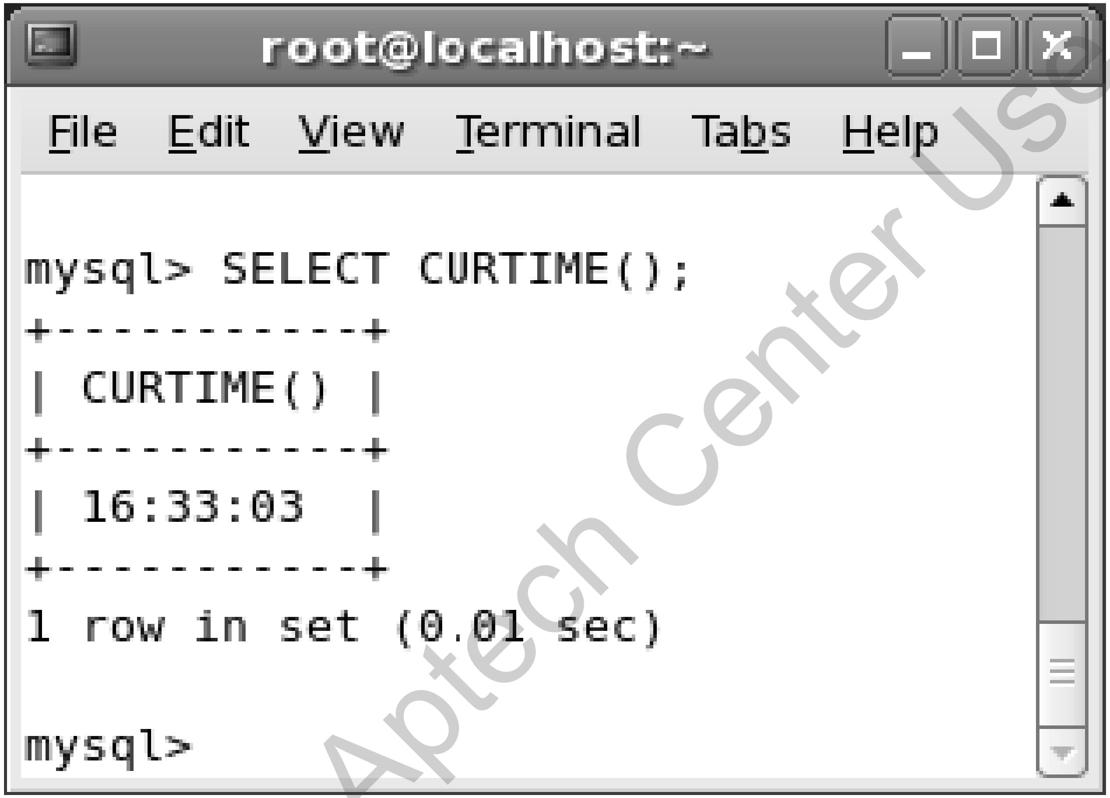
```
SELECT CURTIME();
```

- The alternative syntax for viewing the current time is:

```
SELECT CURRENT_TIME();
```

CURTIME Function

Figure displays the output of the command



```
mysql> SELECT CURTIME();
+------------+
| CURTIME()  |
+------------+
| 16:33:03   |
+------------+
1 row in set (0.01 sec)

mysql>
```

DATE_ADD Function

- The `DATE_ADD` function appends a specified time interval to the given date

- The syntax for this function is:

```
SELECT DATE_ADD(date, INTERVAL expr unit);
```

where,

`DATE_ADD` – alters the date with the specified interval

`date` – specifies the starting date or datetime value

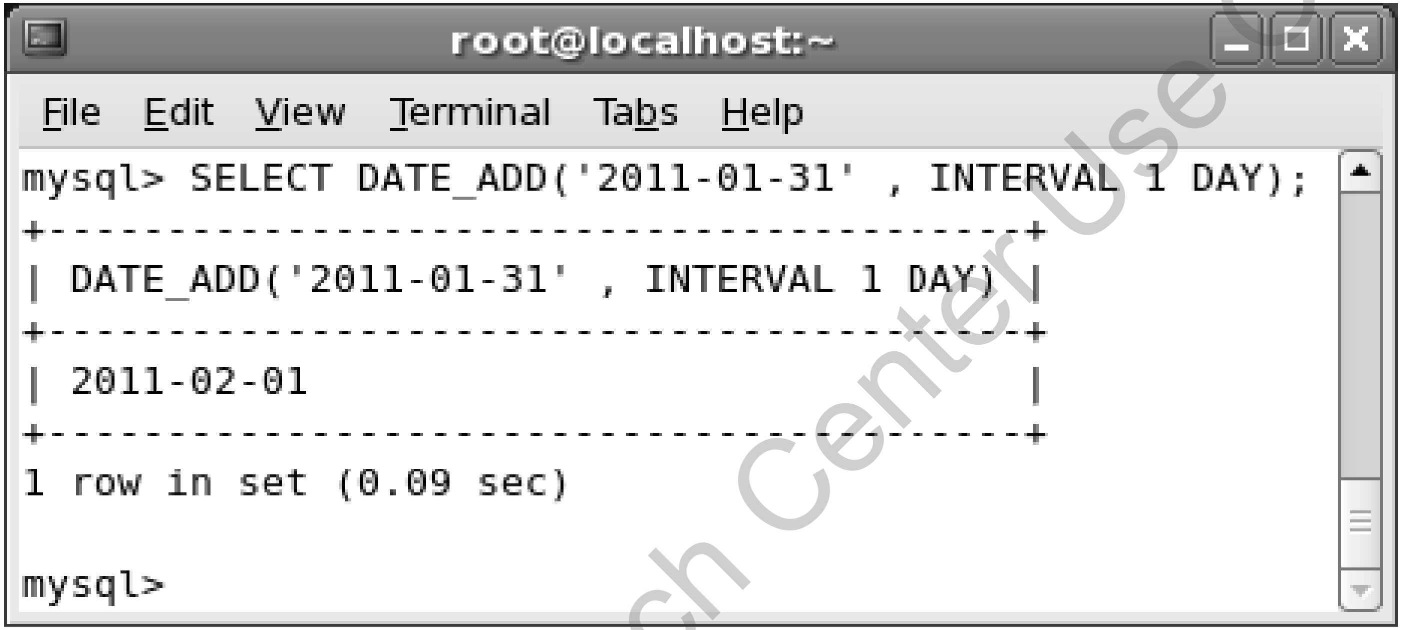
`INTERVAL expr unit` – specifies the interval to be added to the given date

- For example, to add `1` day to the date `2011-01-31`, enter the following command at the command prompt:

```
SELECT DATE_ADD('2011-01-31', INTERVAL 1 DAY);
```

DATE_ADD Function

Figure displays the output of the command



```
mysql> SELECT DATE_ADD('2011-01-31' , INTERVAL 1 DAY);
+---------------------------------------+
| DATE_ADD('2011-01-31' , INTERVAL 1 DAY) |
+---------------------------------------+
| 2011-02-01                            |
+---------------------------------------+
1 row in set (0.09 sec)

mysql>
```

DATE function

- The DATE function displays the date part of the specified date or timestamp
- The syntax for this function is:

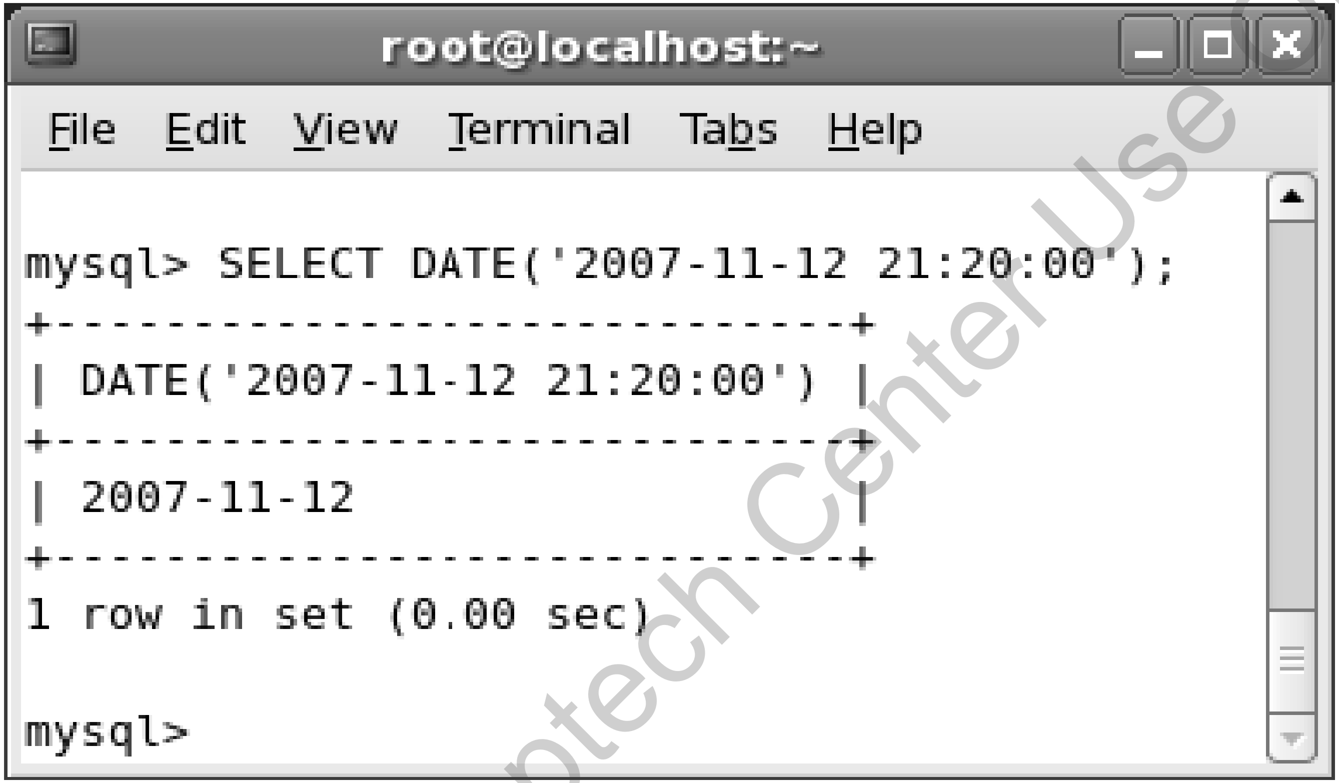
```
SELECT DATE(expression);
```

- For example, to view only the date part from 2007-11-12 21:20:00, enter the following command at the command prompt:

```
SELECT DATE('2007-11-12 21:20:00');
```

DATE function

Figure displays the output of the command



```
mysql> SELECT DATE('2007-11-12 21:20:00');
+-----------------------------+
| DATE('2007-11-12 21:20:00') |
+-----------------------------+
| 2007-11-12                  |
+-----------------------------+
1 row in set (0.00 sec)

mysql>
```

DATEDIFF Function 1-3

- The `DATEDIFF` function returns the number of days between a start date and an end date
- These dates are entered as arguments
- The syntax to calculate the date difference is:

```
SELECT DATEDIFF(expr1,expr2);
```

where,

`DATEDIFF` – calculates the number of days

`expr1` – defines the start date

`expr2` – defines the end date

DATEDIFF Function

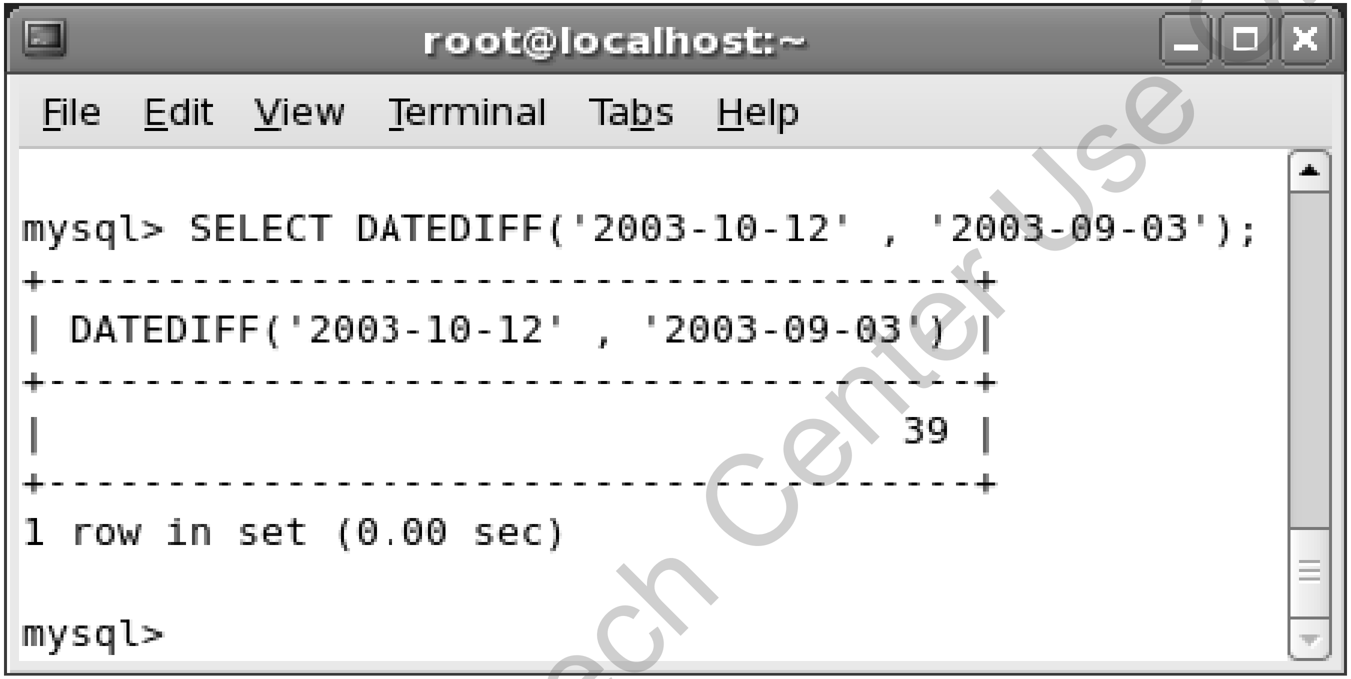
- For example, to calculate the difference between the dates 2003-10-12 and 2003-09-03, enter the following command at the command prompt:

```
SELECT DATEDIFF(‘2003-10-12’,’2003-09-03’);
```

DATEDIFF Function

3-3

Figure displays the output of the command



```
mysql> SELECT DATEDIFF('2003-10-12' , '2003-09-03');
+-----------------------------------------+
| DATEDIFF('2003-10-12' , '2003-09-03') |
+-----------------------------------------+
|                                      39 |
+-----------------------------------------+
1 row in set (0.00 sec)

mysql>
```

DATE_FORMAT Function 1-3

- The DATE_FORMAT function displays the specified date in a particular format

- The syntax to use this function is:

```
SELECT DATE_FORMAT(date, format);
```

where,

date – defines a valid date

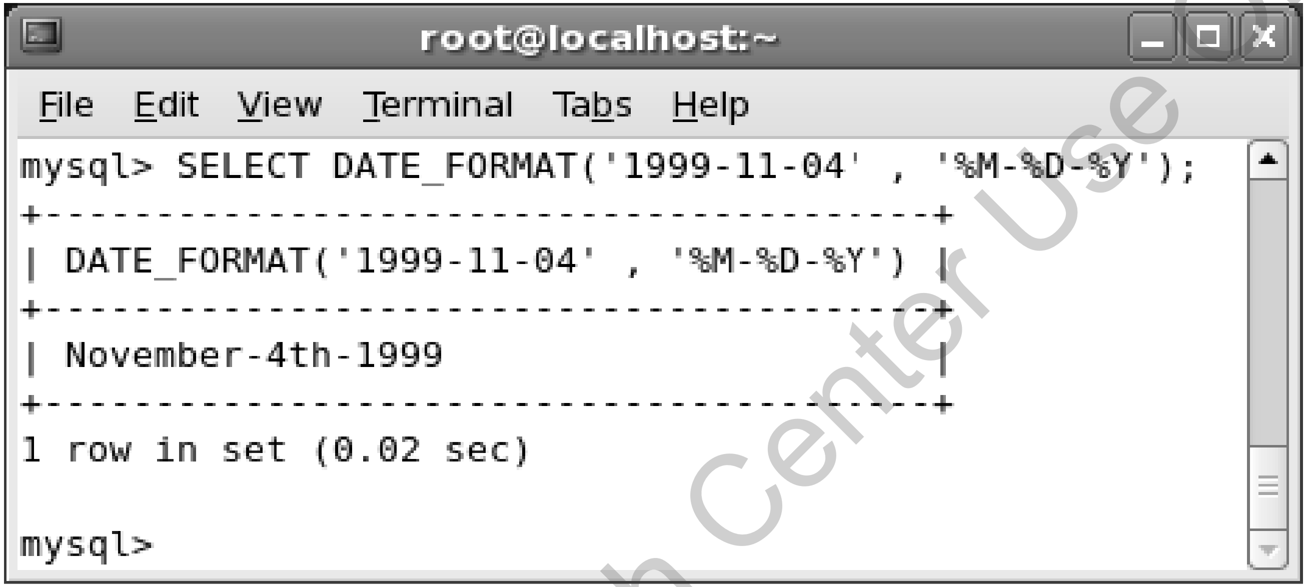
format – defines the output format to display the date

- To display a particular date in the MM–DD–YYYY format, enter the following command at the command prompt:

```
SELECT DATE_FORMAT('1999-11-04','%M-%D-%Y');
```

DATE_FORMAT Function

Figure displays the output of the command



```
mysql> SELECT DATE_FORMAT('1999-11-04' , '%M-%D-%Y');
+----------------------------------------+
| DATE_FORMAT('1999-11-04' , '%M-%D-%Y') |
+----------------------------------------+
| November-4th-1999                      |
+----------------------------------------+
1 row in set (0.02 sec)

mysql>
```

DATE_FORMAT Function

3-3

Table lists some of the format specifiers

Specifier	Description
%a	Abbreviated weekday name
%f	Microseconds
%H	Hour (00 – 23)
%h	Hour (00 -12)
%M	Month name (January – December)
%m	Month, numeric (01 – 12)
%p	A.M. or P.M.
%s	Seconds
%D	Date
%Y	Year, numeric, four digits

DAY Function

- The DAY function returns the day of the month for the specified date
- The range is from 1 to 31
- The syntax to display the day of the month is:

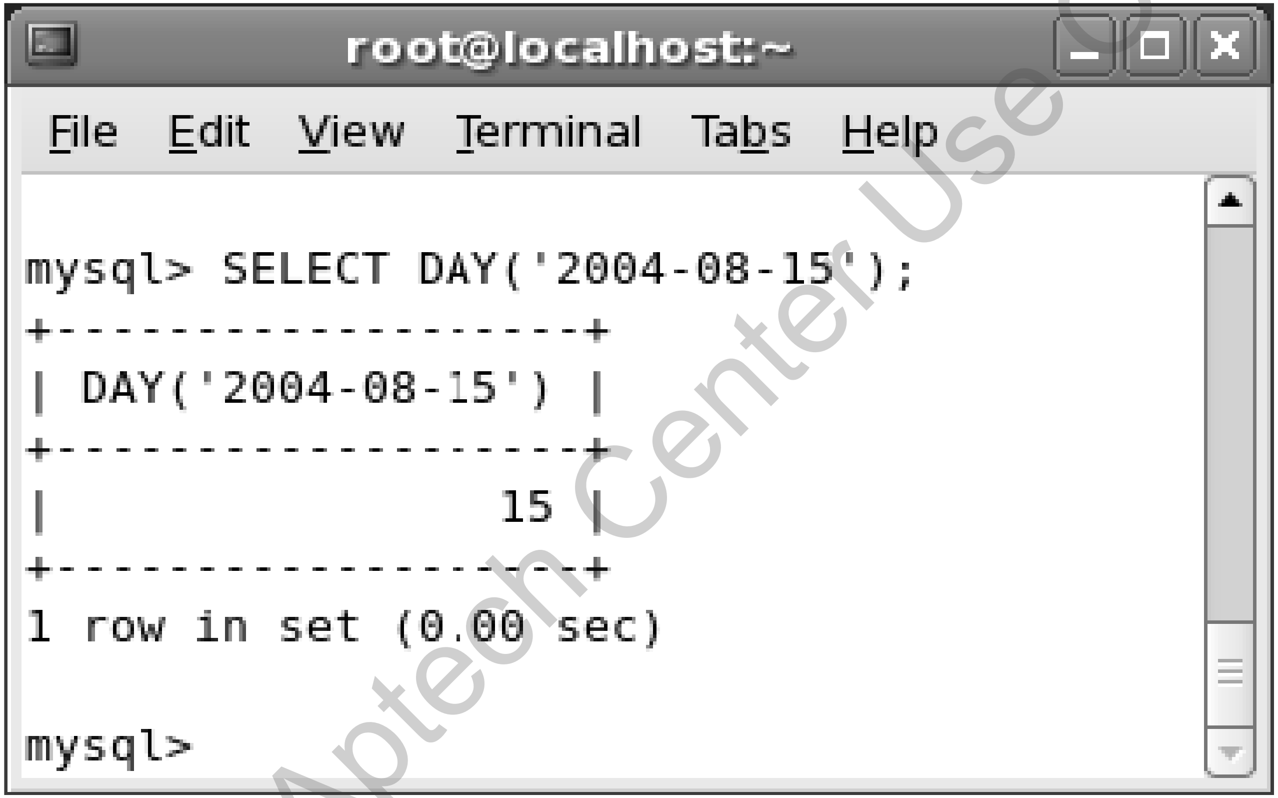
```
SELECT DAY(date);
```

- For example, to obtain the day for the date 2004-08-15, enter the following command at the command prompt:

```
SELECT DAY(‘2004-08-15’);
```

DAY Function

Figure displays the output of the command



```
mysql> SELECT DAY('2004-08-15');
+-------------------+
| DAY('2004-08-15') |
+-------------------+
|                15 |
+-------------------+
1 row in set (0.00 sec)

mysql>
```

DAYNAME Function 1-2

- The DAYNAME function returns the name of the weekday for a date entered as an argument

- The syntax to display the name of the weekday is:

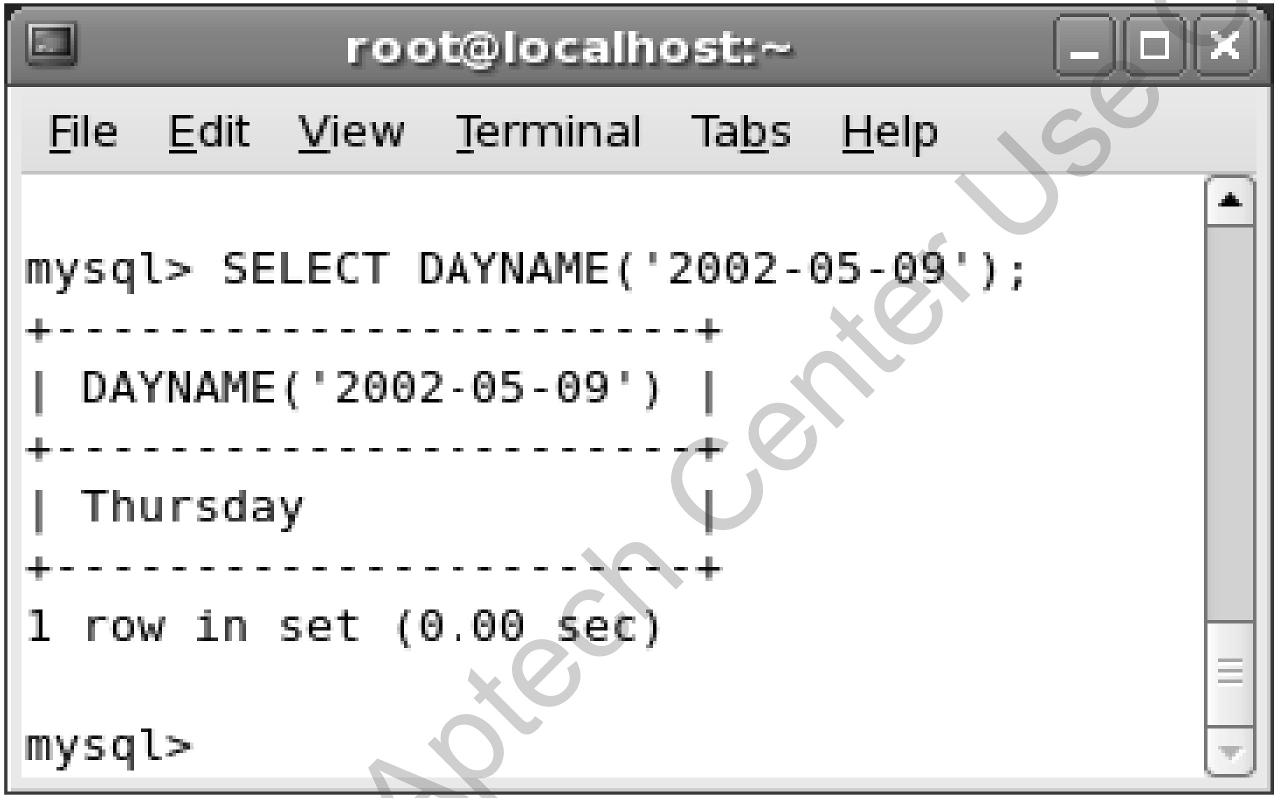
```
SELECT DAYNAME(date);
```

- For example, to obtain the name of the day for the date 2002-05-09, enter the following command at the command prompt:

```
SELECT DAYNAME('2002-05-09');
```

DAYNAME Function

Figure displays the output of the command



```
mysql> SELECT DAYNAME('2002-05-09');
+-----------------------+
| DAYNAME('2002-05-09') |
+-----------------------+
| Thursday              |
+-----------------------+
1 row in set (0.00 sec)

mysql>
```

HOUR Function

- The HOUR function returns the hour of the time specified as an argument
- The syntax for this function is:

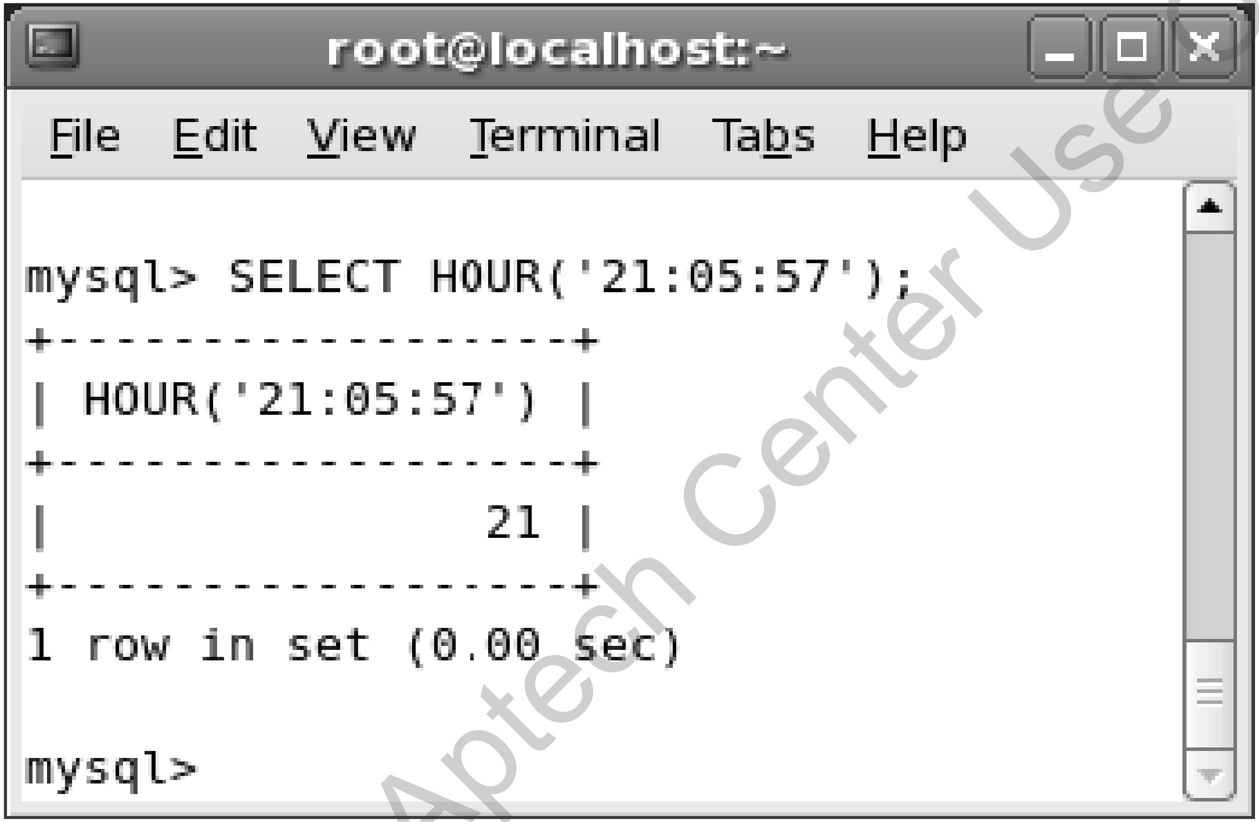
```
SELECT HOUR(expression);
```

- For example, to retrieve the hour from the expression 21:05:57, enter the following command at the command prompt:

```
SELECT HOUR('21:05:57');
```

HOUR Function

Figure displays the output of the command



```
mysql> SELECT HOUR('21:05:57');
+-------------------+
| HOUR('21:05:57')  |
+-------------------+
|                21 |
+-------------------+
1 row in set (0.00 sec)

mysql>
```

MINUTE Function 1-2

- The MINUTE function extracts the minutes from the specified time argument

- The output of this function is a numeric value ranging from 0 to 59

- The syntax for this function is:

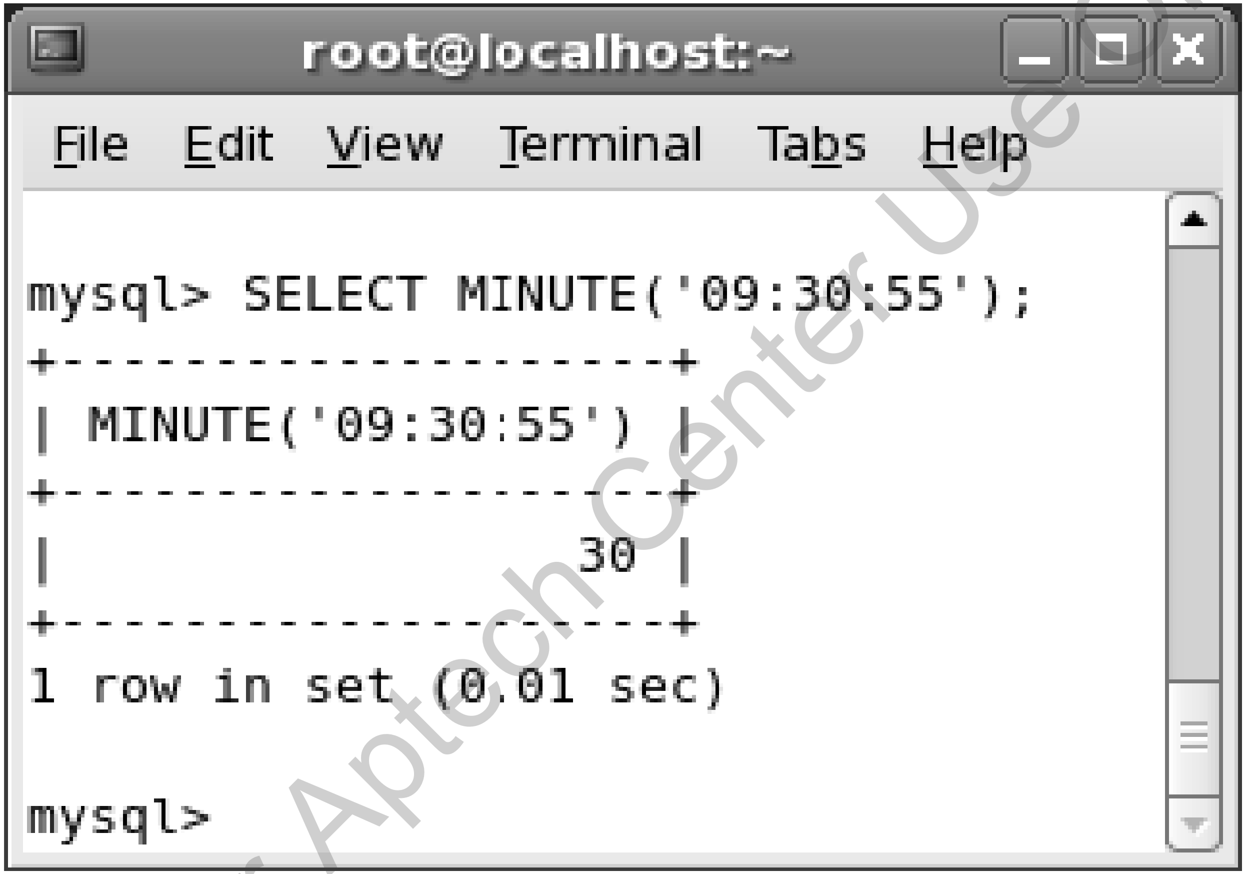
```
SELECT MINUTE(expression);
```

- For example, to display the minutes from a datetime expression, enter the following command at the command prompt:

```
SELECT MINUTE('09:30:55');
```

MINUTE Function

Figure displays the output of the command



```
mysql> SELECT MINUTE('09:30:55');
+----------------------+
| MINUTE('09:30:55') |
+----------------------+
|                 30 |
+----------------------+
1 row in set (0.01 sec)

mysql>
```

MONTH Function

- The MONTH function extracts the month from the specified argument

- The output of this function is a numeric value ranging from 1 to 12

- The syntax for this function is:

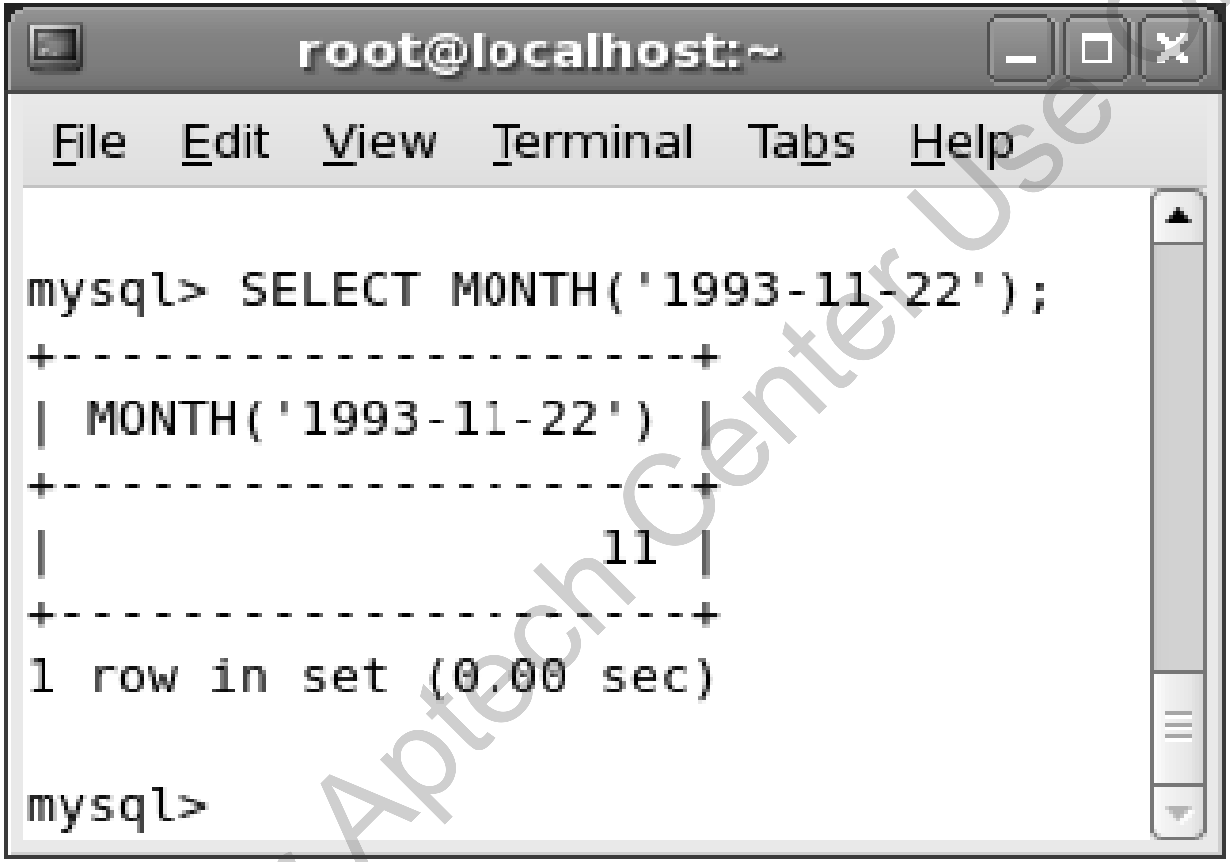
```
SELECT MONTH(expression);
```

- For example, to display the month from the datetime expression 1993-11-22, enter the following command at the command prompt:

```
SELECT MONTH('1993-11-22');
```

MONTH Function

Figure displays the output of the command



```
mysql> SELECT MONTH('1993-11-22');
+---------------------+
| MONTH('1993-11-22') |
+---------------------+
|                  11 |
+---------------------+
1 row in set (0.00 sec)

mysql>
```

MONTHNAME Function 1-2

- The `MONTHNAME` function returns the name of the month for the date passed as an argument

- The syntax for displaying the name of the month is:

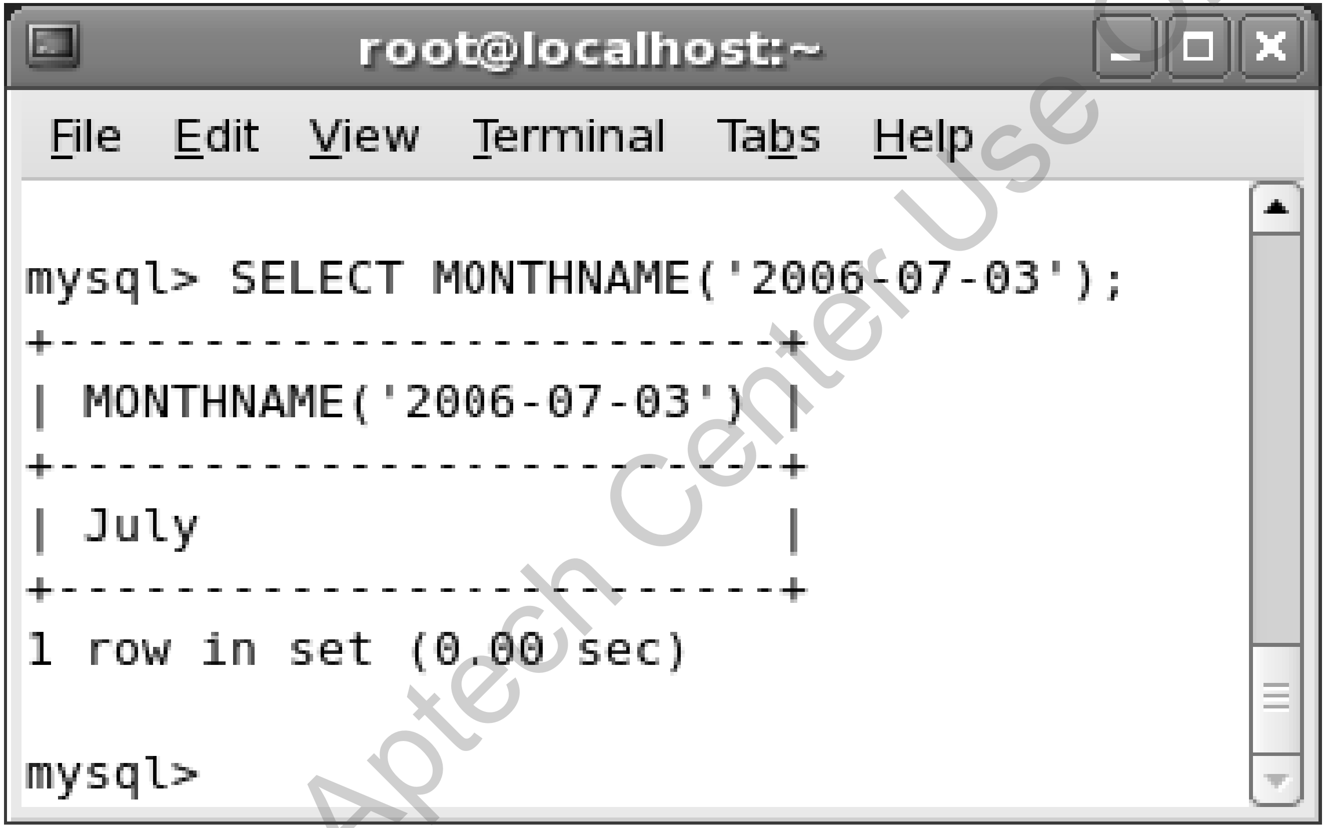
```
SELECT MONTHNAME(date);
```

- For example, to display the name of the month for the date `2006-07-03`, enter the following command at the command prompt:

```
SELECT MONTHNAME('2006-07-03');
```

MONTHNAME Function

Figure displays the output of the command



```
mysql> SELECT MONTHNAME('2006-07-03');
+-------------------------+
| MONTHNAME('2006-07-03') |
+-------------------------+
| July                    |
+-------------------------+
1 row in set (0.00 sec)

mysql>
```

NOW Function

- The NOW function returns the current date and time
- The output of this function is in the YYYY-MM-DD HH:MM:SS or YYYYMMDDHHMMSS.uuuuuu format, and the value is expressed in the current time zone
- The syntax to view the current date and time value is:

```
SELECT NOW();
```

- For example, to view the current timestamp, enter the following command at the command prompt:

```
SELECT NOW();
```

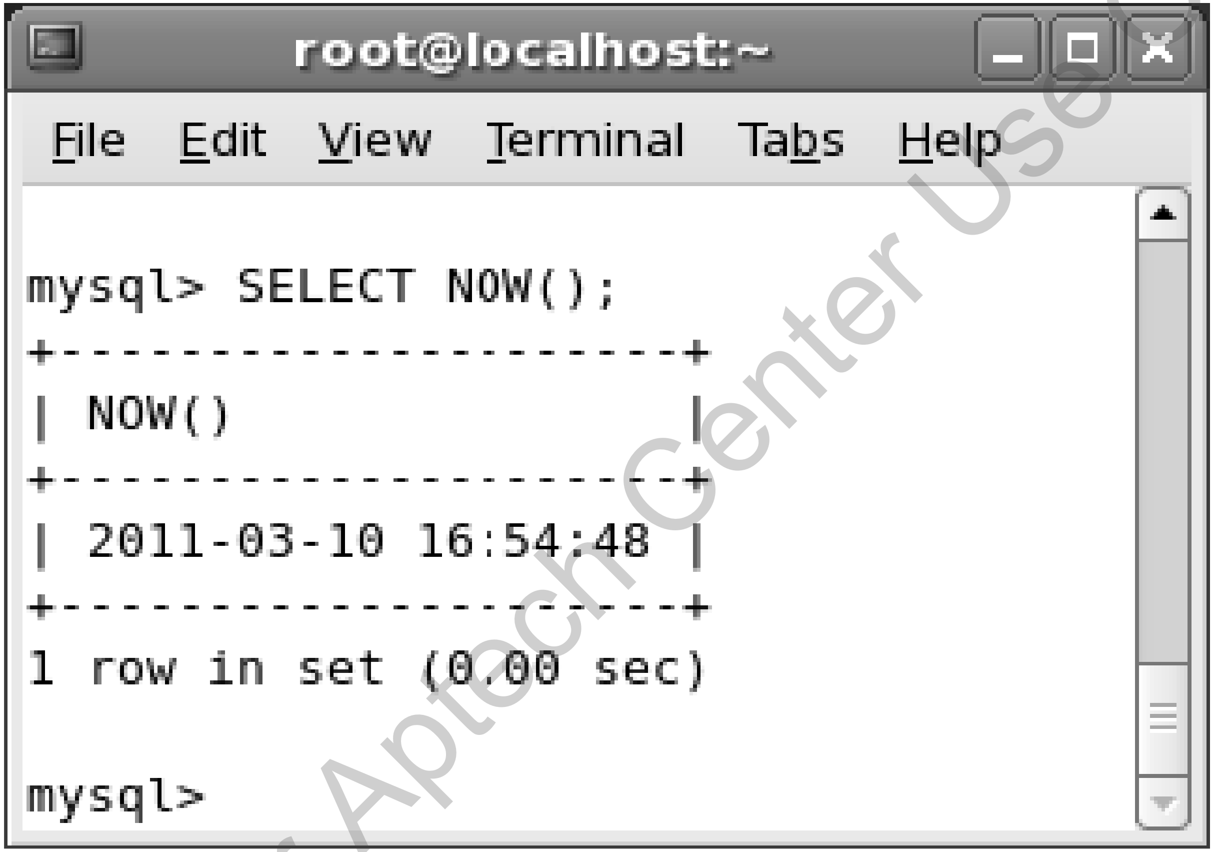
NOW Function

- The alternative commands to display the current time and date are:

```
SELECT CURRENT_TIMESTAMP();
SELECT LOCALTIME();
SELECT LOCALTIMESTAMP();
```

NOW Function

Figure displays the output of the command



```
mysql> SELECT NOW();
+---------------------+
| NOW()               |
+---------------------+
| 2011-03-10 16:54:48 |
+---------------------+
1 row in set (0.00 sec)

mysql>
```

NOW Function 3-3

- The `SECOND` function returns the number of seconds specified in the argument
- The output of this function is a numeric value ranging from 1 to 59
- The syntax for this function is:

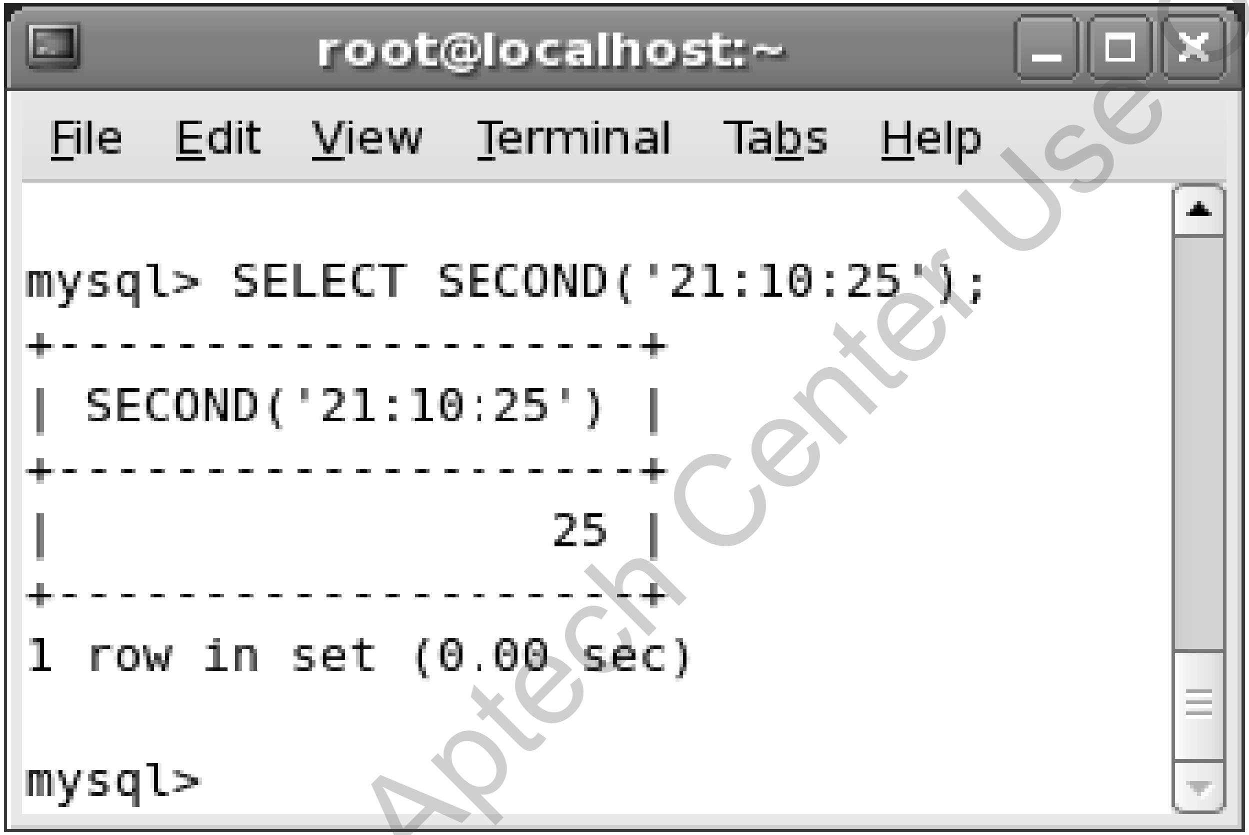
```
SELECT SECOND(expression);
```

- For example, to calculate the number of seconds for `21:10:25`, enter the following command at the command prompt:

```
SELECT SECOND(‘21:10:25’);
```

SECOND Function

Figure displays the output of the command



```
mysql> SELECT SECOND('21:10:25');
+--------------------+
| SECOND('21:10:25') |
+--------------------+
|                 25 |
+--------------------+
1 row in set (0.00 sec)

mysql>
```

SEC_TO_TIME Function

- The SEC_TO_TIME function converts the number of seconds specified as an argument to HH:MM:SS format
- The syntax for this function is:

```
SELECT SEC_TO_TIME(expression);
```

where,

SEC_TO_TIME – converts seconds to HH:MM:SS format

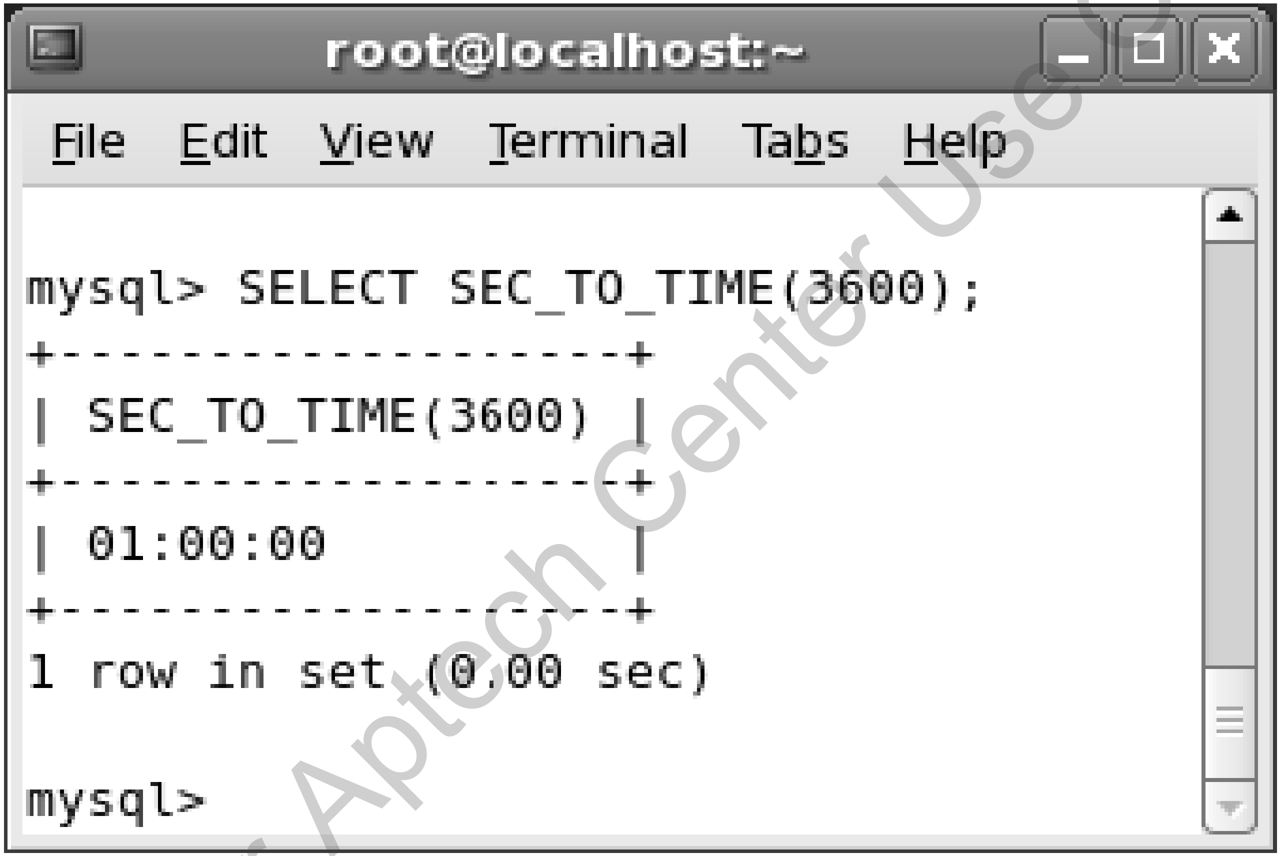
expression – defines a numeric value specifying the number of seconds

- To convert 3600 to the HH:MM:SS format, enter the following command at the command prompt:

```
SELECT SEC_TO_TIME(3600);
```

SEC_TO_TIME Function

Figure displays the output of the command



```
mysql> SELECT SEC_TO_TIME(3600);
+-------------------+
| SEC_TO_TIME(3600) |
+-------------------+
| 01:00:00          |
+-------------------+
1 row in set (0.00 sec)

mysql>
```

SYSDATE Function

- The SYSDATE function returns the current date and time value after execution

- SYSDATE function is different from the NOW function

- The SYSDATE function returns this unique time required to execute a function

- The NOW function will display the same execution time for adding all the 50,000 records to the table

- The syntax for this function is:

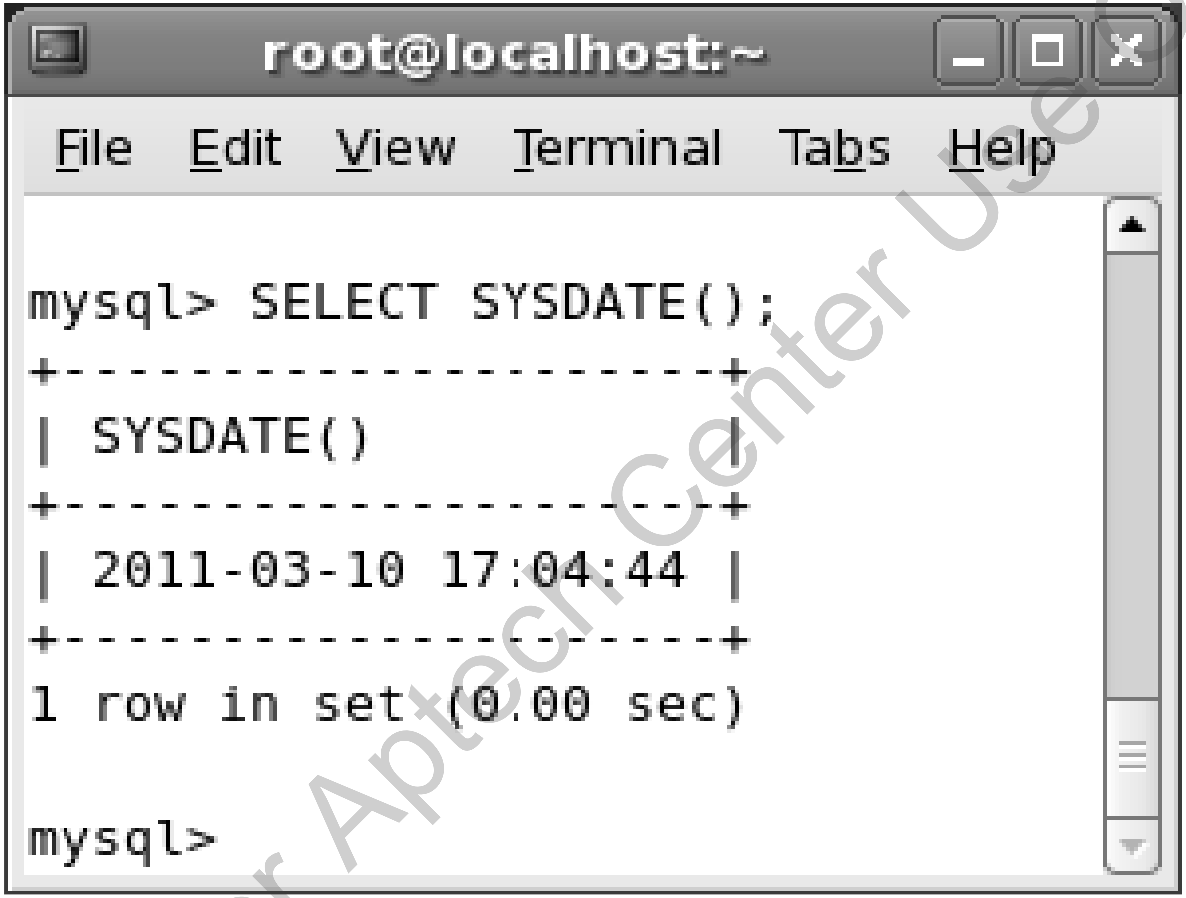
```
SELECT SYSDATE(expression);
```

- For example, to display the current date and time value, enter the following command at the command prompt:

```
SELECT SYSDATE();
```

SYSDATE Function

Figure displays the output of the command



```
mysql> SELECT SYSDATE();
+---------------------+
| SYSDATE()           |
+---------------------+
| 2011-03-10 17:04:44 |
+---------------------+
1 row in set (0.00 sec)

mysql>
```

TIME Function

- The `TIME` function displays only the time part of the specified date or timestamp

- The syntax for this function is:

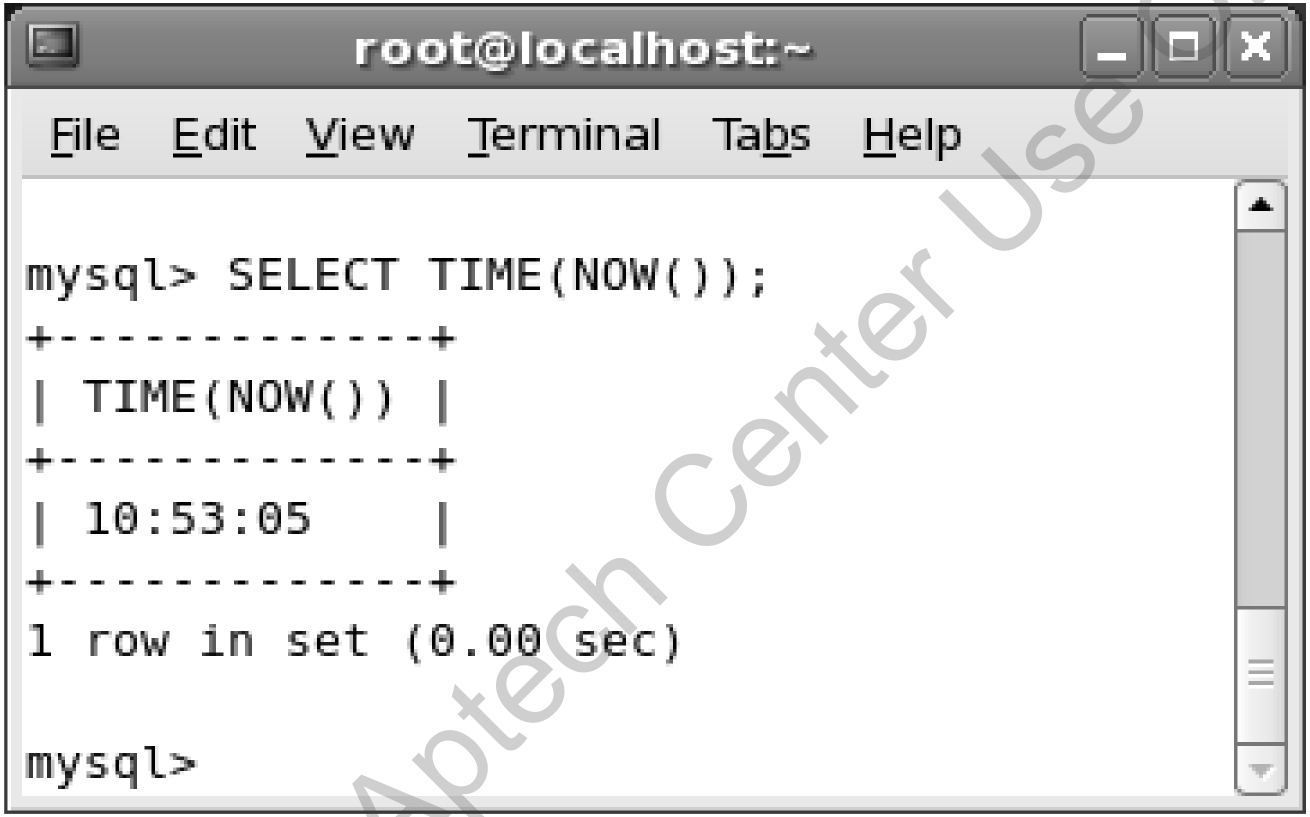
```
SELECT TIME(expression);
```

- For example, to display only the time from `2007-09-12 20:20:00`, enter the following command at the command prompt:

```
SELECT TIME(NOW());
```

TIME Function 2-2

Figure displays the output of the command



```
mysql> SELECT TIME(NOW());
+-------------+
| TIME(NOW()) |
+-------------+
| 10:53:05    |
+-------------+
1 row in set (0.00 sec)

mysql>
```

TIMEDIFF Function

- The `TIMEDIFF` function returns time interval between two time arguments
- The syntax to calculate the time difference is:

```
SELECT TIMEDIFF(time1, time2);
```

where,

`TIMEDIFF` – calculates the time difference

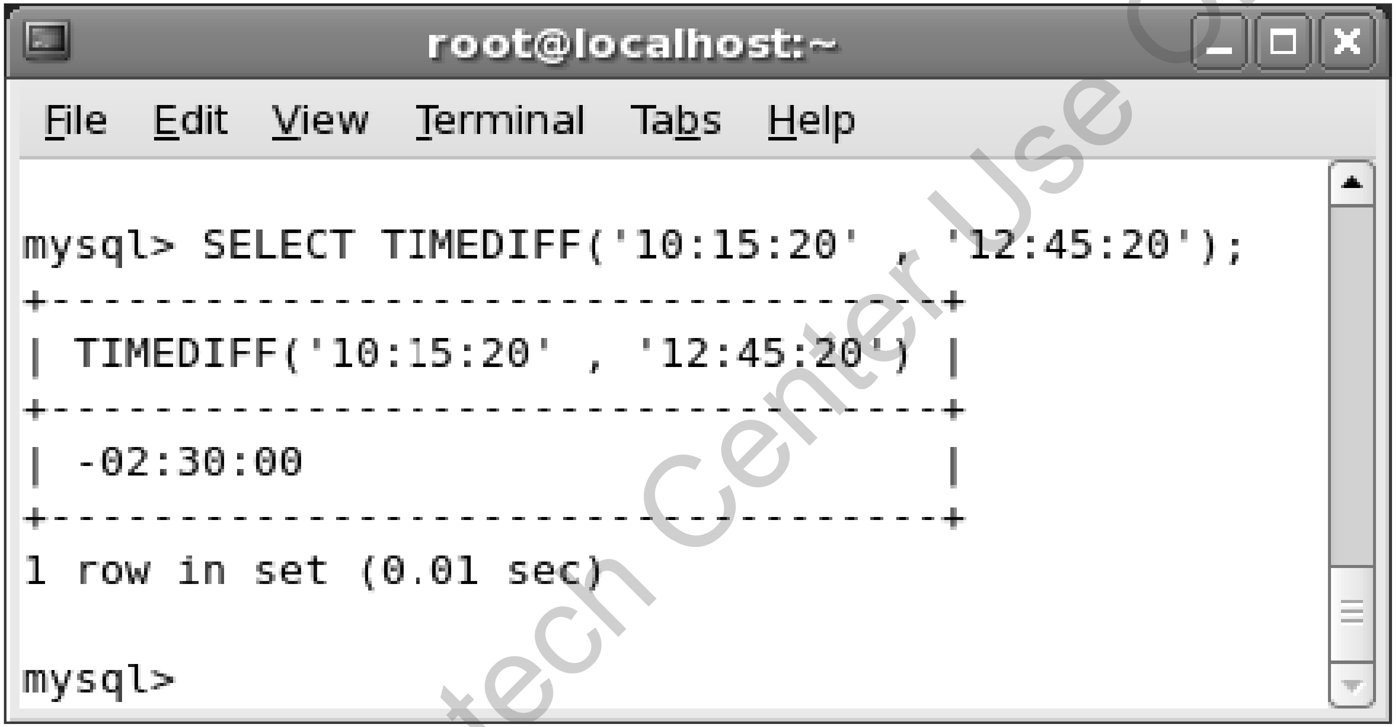
`time1, time2` – specifies the time expressions

- For example, to calculate the difference between `10:15:20` and `12:45:20`, enter the following command at the command prompt:

```
SELECT TIMEDIFF('10:15:20','12:45:20');
```

TIMEDIFF Function 2-2

Figure displays the output of the command



```
mysql> SELECT TIMEDIFF('10:15:20' , '12:45:20');
+-------------------------------------+
| TIMEDIFF('10:15:20' , '12:45:20')   |
+-------------------------------------+
| -02:30:00                           |
+-------------------------------------+
1 row in set (0.01 sec)

mysql>
```

WEEK Function

- The WEEK function returns the week number for the value specified in the argument
- The syntax for this function is:

```
SELECT WEEK(date, mode);
```

where,

WEEK – displays the week number

date – specifies the date

mode – defines the return value. The return value can be within a range of 1 to 53 or 0 to 53 depending on whether the week starts on Monday or Sunday

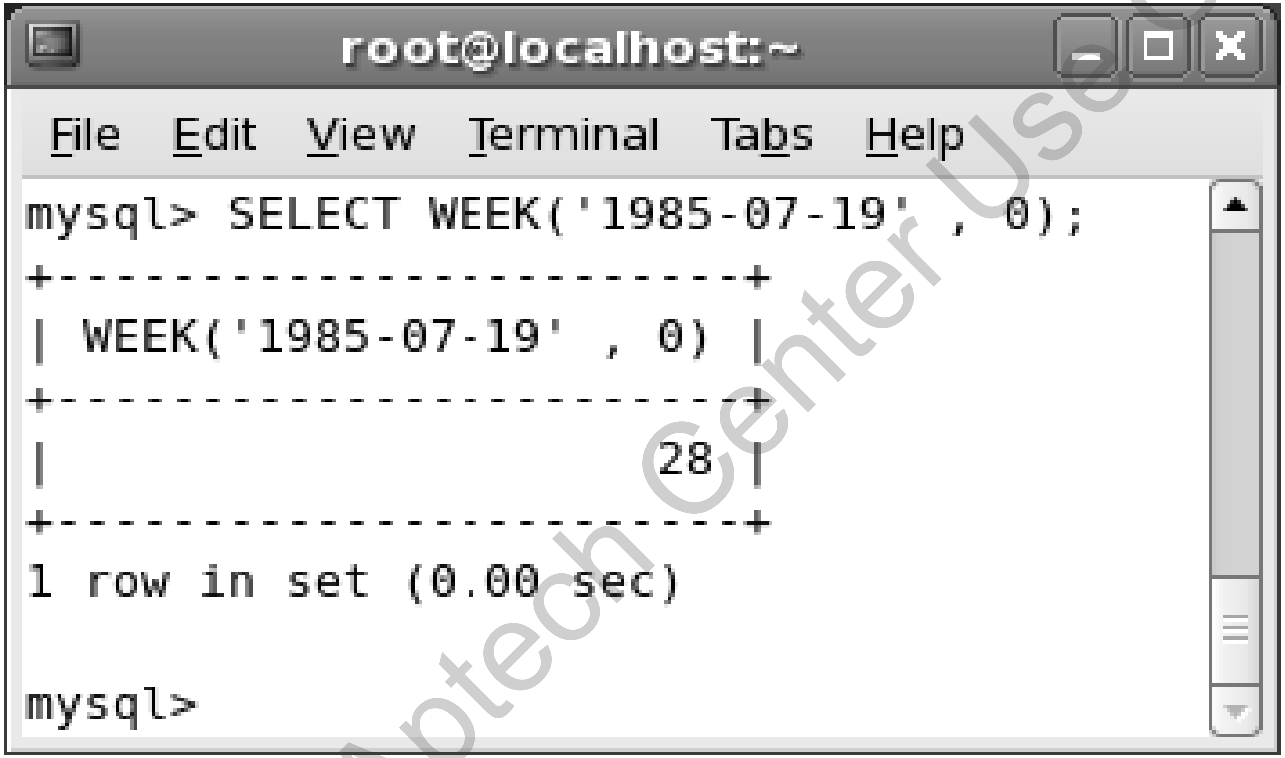
WEEK Function

- For example, to retrieve the week number for `1985-07-19`, enter the following command at the command prompt:

```
SELECT WEEK(‘1985-07-19’, 0);
```

WEEK Function

Figure displays the output of the command



```
mysql> SELECT WEEK('1985-07-19' , 0);
+--------------------------+
| WEEK('1985-07-19' , 0) |
+--------------------------+
|                       28 |
+--------------------------+
1 row in set (0.00 sec)

mysql>
```

WEEKOFYEAR Function

- The WEEKOFYEAR function returns the week number for the specified date
- The output of this function is a numeric value ranging between 1 to 53. The syntax for this function is:

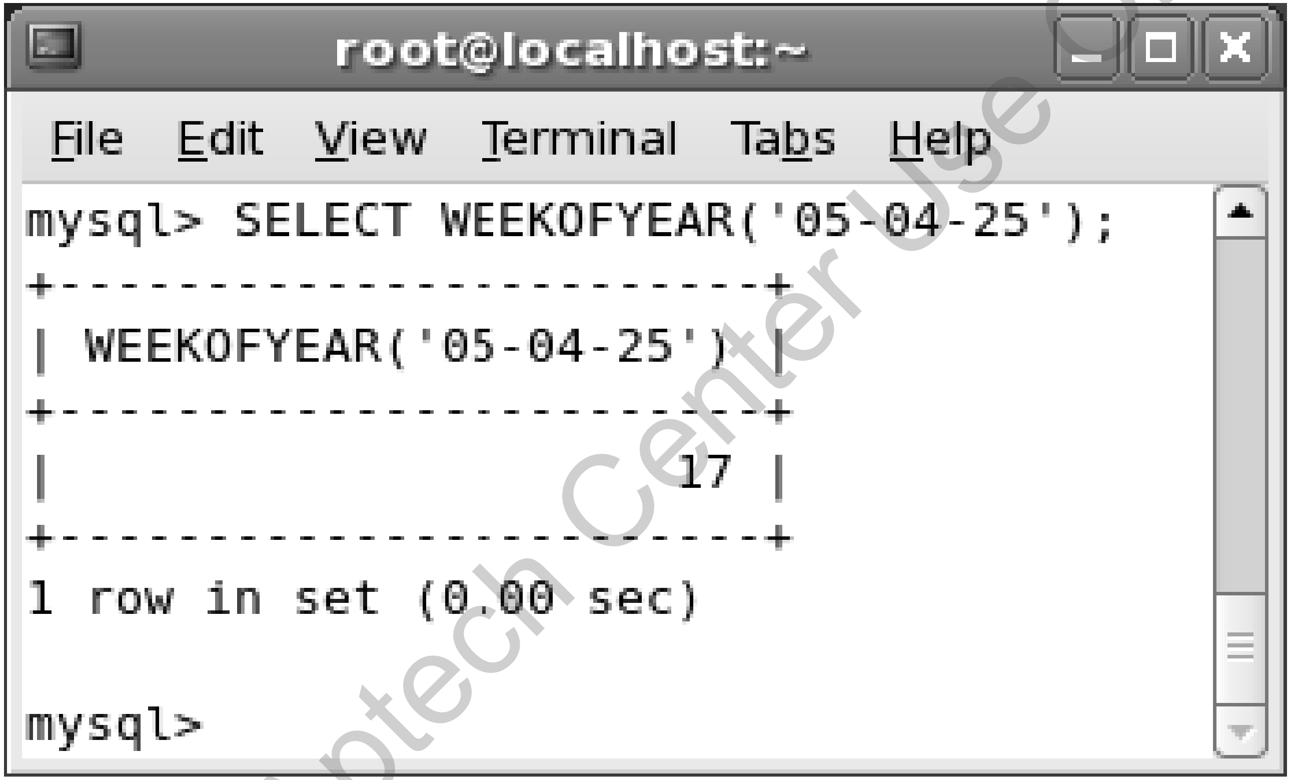
```
SELECT WEEKOFYEAR(expression);
```

- For example, to calculate the week number for the date 2005-04-25, enter the following command at the command prompt:

```
SELECT WEEKOFYEAR('2005-04-25');
```

WEEKOFYEAR Function 2-2

Figure displays the output of the command



```
mysql> SELECT WEEKOFYEAR('05-04-25');
+------------------------+
| WEEKOFYEAR('05-04-25') |
+------------------------+
|                     17 |
+------------------------+
1 row in set (0.00 sec)

mysql>
```

YEAR Function

- The YEAR function returns the year for the date specified in the argument

- The syntax for obtaining the year from a date argument is:

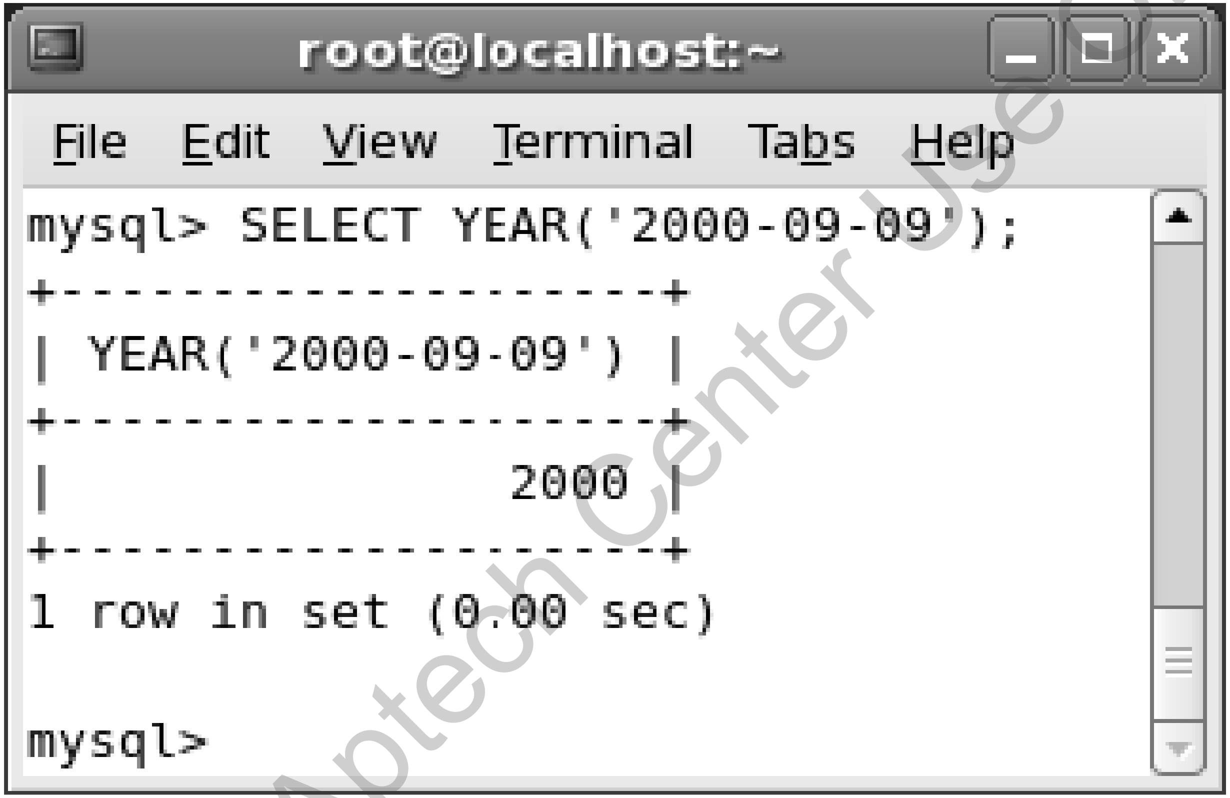
```
SELECT YEAR(date);
```

- For example, to display the year from the date 2000-09-09, enter the following command at the command prompt:

```
SELECT YEAR('2000-09-09');
```

YEAR Function

Figure displays the output of the command



```
mysql> SELECT YEAR('2000-09-09');
+--------------------+
| YEAR('2000-09-09') |
+--------------------+
|               2000 |
+--------------------+
1 row in set (0.00 sec)

mysql>
```

YEAR Function

3-15

- Table describes the additional Date functions in MySQL

Name	Description	Example
CONVERT_TZ	The CONVERT_TZ function changes the time zone of an argument. The syntax for using this function is: SELECT CONVERT_TZ(dt, from_tz, to_tz); where, CONVERT – edits the time zone dt – specifies the date in the current time zone from_tz – specifies the current time zone of the argument to_tz – specifies the new time zone	To convert the time zone for the date 2004-01-01, enter the following command at the command prompt: SELECT CONVERT_TZ(‘2004-01-01 12:00:00’, ‘+00:00’, ‘+10:00’); The output of this function is: 2004-01-01 22:00:00

YEAR Function

Name	Description	Example
CURRENT_ TIMESTAM P	The CURRENT_TIMESTAMP function displays the current date and time as a value in YYYY-MM-DD HH:MM:SS or YYYYMMDDHHMMSS.uuuuuu format. The value is expressed in the current time zone. The syntax to view the current timestamp is: `SELECT CURRENT_TIMESTAMP();`	For example, to view the current timestamp, enter the following command at the command prompt: `SELECT CURRENT_TIMESTAMP();` The output of this function will be the current date and time.
DATE_SUB	The DATE_SUB function subtracts a specified time interval from the given date. The syntax for this function is: SELECT DATE_SUB(date, INTERVAL expr unit); Where, DATE_SUB – subtracts the time date – defines the starting date and datetime value INTERVAL expr- specifies the interval to be subtracted from the given date	To subtract 1 day from the current date, enter the following command at the command prompt: SELECT DATE_SUB(CURDATE(), INTERVAL 1 DAY); The output of this function will be yesterday's date

YEAR Function

Name	Description	Example

DAYOFMONTH	The DAYOFMONTH function returns the day of the month for a date. The output of this function is a numeric value between 1 and 31. The syntax for obtaining the day of the month is:
	`SELECT DAYOFMONTH(date);`
		For example, to extract the day of the month for the date 2005-03-18, enter the following command at the command prompt:
		`SELECT DAYOFMONTH(‘2005-03-18’);`
		The output of this function is:
		`18`

DAYOFWEEK	The DAYOFWEEK function returns the day of the week for a date. The output for this function is a numeric value between 1 and 7, where 1 is Sunday and 7 is Saturday. The syntax to display the day of the week is:
	`SELECT DAYOFWEEK(date);`
		For example, to display the day of the week for a particular date, enter the following command at the command prompt:
		`SELECT DAYOFWEEK(‘1995-06-07’);`
		The output of this function is:
		`4`

YEAR Function

Name	Description	Example
DAYOFYEAR	The DAYOFYEAR function returns the day of the year for a date. The output of this function is a numeric value between 1 and 366. The syntax for obtaining the day of the month is: `SELECT DAYOFYEAR(date);`	For example, to display the day of the year for a particular date, enter the following command at the command prompt: `SELECT DAYOFYEAR('2001-10-21');` The output of this function is: 294
EXTRACT	The EXTRACT function extracts a particular part from a given date. The syntax for this function is: `SELECT EXTRACT(unit from date);` where, date – defines a valid date expression unit – specifies any valid unit, such as year, month, second, minute, or day	For example, to display only the year from a particular date, enter the following command at the command prompt: `SELECT EXTRACT(YEAR FROM '2008-04-06');` The output of this function is: 2008

YEAR Function 7-15

Name	Description	Example

`FROM_DAYS`	The `FROM_DAYS` function returns the date of the number specified as an argument. This function works only for dates after the advent of the Gregorian calendar (1582). The syntax for this function is:
	`SELECT FROM_DAYS(argument);`
		For example, to display the date for `5,00,000`, enter the following command at the command prompt:
		`SELECT FROM_DAYS(500000);`
		The output of this function is:
		`1368-12-14`

`LAST_DAY`	The `LAST_DAY` function returns the last date of the month for the argument. The output is a numeric value ranging from 1 to 31, if the argument is valid. However, if the argument is invalid, the output is `NULL`. The syntax for using this function is:
	`SELECT LAST_DAY(expression);`
		For example, enter the following command at the command prompt:
		`SELECT LAST_DAY('2008-02-05');`
		The output of this function is:
		`2008-02-29`

YEAR Function

Name	Description	Example
MICROSECOND	The MICROSECOND function returns the number of microseconds in the argument specified as a datetime value. The output of this function is a numeric value ranging from 0 to 999999. The syntax for this function is: `SELECT MICROSECOND(expression);`	For example, to display the microseconds from the datetime expression 11:45:45.000123, enter the following command at the command prompt: `SELECT MICROSECOND (‘11:45:45.000123’);` The output of this function is: 123
PERIOD_ADD	The PERIOD_ADD function adds a specified number of months to a given period expressed either as YYMM or YYYYMM. The final output is displayed in the form of YYYYMM. The syntax for this function is: `SELECT PERIOD_ADD(P, N);`	For example, to add two months to the period 200705, enter the following command at the command prompt: `SELECT PERIOD_ADD(‘200705’, 02);` The output of this function is: 200707

YEAR Function

Name	Description	Example
PERIOD_DIFF	The PERIOD_DIFF function calculates the difference in number of months between two periods. The final output is a numeric value. The syntax for this function is: `SELECT PERIOD_DIFF(P1, P2);` where, PERIOD_DIFF – calculates the number of months between two periods P1 - stands for Period1 P2 - stands for Period2	For example, to calculate the number of months between 199608 and 199503, enter the following command at the command prompt: `SELECT PERIOD_DIFF('199608', '199503');` The output of this function is: 17
QUARTER	The QUARTER function returns the quarter number for the argument specified as a date value. The output of this function is a numeric value ranging from 1 - 4 The syntax for this function is: `SELECT QUARTER(expression);`	For example, to calculate the quarter for the date 1987-12-24, enter the following command at the command prompt: `SELECT QUARTER('1987-12-24');` The output of this function is: 4

YEAR Function

Name	Description	Example
STR_TO_DATE	The STR_TO_DATE function converts a string into a specified format. The output of this function can be a date, time, or a datetime value depending upon the format. The syntax of this function is: SELECT STR_TO_DATE(String, format);	To convert the string 01,05,2013 into the YYYY-MM-DD format, enter the following command at the command prompt: SELECT STR_TO_DATE(‘01,05,2013’,’%m, %d, %Y’); The output of this function is: 2013-01-05
SUBTIME	The SUBTIME function subtracts a specified time interval from the given datetime or time value. The syntax for this function is: SELECT SUBTIME(expr1, expr2); where, SUBTIME – subtracts a specified time interval expr1 – defines the starting datetime or time value expr2 – specifies the time value to be subtracted	To subtract 45 minutes from the current date, enter the following command at the command prompt: SELECT SUBTIME(NOW(),’00:45:00’); The output of this function will be a timestamp of 45 minutes prior to the current time.

YEAR Function

11-15

Name	Description	Example
TIME_FORMAT	The TIME_FORMAT function displays the time in a specific format. The syntax for this function is: `SELECT TIME_FORMAT(time, format);` where, time – defines a valid time format – defines the output to display the time	To display the current time in the HH-MM-SS format, enter the following command at the command prompt: `SELECT TIME_FORMAT(NOW(), '%H-%I-%S');` The output of this function will be the current time in HH-MM-SS format.
TIME_TO_SEC	The TIME_TO_SEC function converts a time argument into seconds. The output of this function is a numeric value and the syntax for this function is: `SELECT TIME_TO_SEC(expression)`	To convert the time expression 23:45:02 into seconds, enter the following command at the command prompt: `SELECT TIME_TO_SEC('23:45:02');` The output of this function is: 85502

YEAR Function

Name	Description	Example
TIMESTAMP	The TIMESTAMP function returns the datetime expression of the specified argument. When you specify only a single argument, the output is a datetime value of only that argument. If you specify two arguments, the output would be the sum of the two expressions The syntax for this function is: `SELECT TIMESTAMP(expression);`	For example, to display the current timestamp value, enter the following command at the command prompt: `SELECT TIMESTAMP(NOW());` The output of this function will be the current date and time.

TO_DAYS	The TO_DAYS function converts a datetime value into a numeric value, which is the day number. It starts counting the number of days after the advent of the Gregorian calendar (1582). The syntax for this function is:
	`SELECT TO_DAYS(expression);`
		For example, to calculate the day number for the date 1982-06-24, enter the following command at the command prompt:
		`SELECT TO_DAYS(‘1982-06-24’);`
		The output of this function will be:
		`724085`

YEAR Function

13-15

Name	Description	Example
`TO_SECONDS`	The `TO_SECONDS` function converts the argument into seconds. The argument is a datetime value and the output is a numeric value. The syntax for this function is: `SELECT TO_SECONDS(expression);`	To convert `1975-02-25` into seconds, enter the following command at the command prompt: `SELECT TO_SECONDS('1975-02-25');` The output of this function is: `62329737600`
`UTC_DATE`	The `UTC_DATE` function returns the current UTC date as a value in `YYYY-MM-DD` or `YYYYMMDD` format, depending on whether the function is used in a string or numeric context. The syntax for this function is: `SELECT UTC_DATE(expression);`	For example, to retrieve the current `UTC` date, enter the following command at the command prompt: `SELECT UTC_DATE();` The output of this function will be the current `UTC` date

YEAR Function

Name	Description	Example
UTC_TIME	The UTC_TIME function returns the current UTC time as a value in HH:MM:SS or HHMMSS.uuuuuu format, depending on whether the function is used in a string or numeric context. The syntax for this function is: SELECT UTC_TIME(expression);	For example, to retrieve the current UTC time, enter the following command at the command prompt: SELECT UTC_TIME(); The output of this function will be the current UTC time
UTC_TIMESTAMP	The UTC_TIMESTAMP function returns the current UTC date and time as a value in YYYY-MM-DD HH:MM:SS or YYYYMMDDHHMMSS.uuuuuu format, depending on whether the function is used in a string or numeric context. The syntax for this function is: SELECT UTC_TIMESTAMP(expression);	For example, to retrieve the current UTC date and time, enter the following command at the command prompt: SELECT UTC_TIMESTAMP(); The output of this function will be the current date and time

YEAR Function

Name	Description	Example
`WEEKDAY`	The `WEEKDAY` function returns the weekday index for the argument. The output of this function is a numeric value between `0 to 6` where, 0 represents Monday and 6 represents Sunday The syntax for this function is : `SELECT WEEKDAY(expression);`	The output of this function will be a numerical value corresponding to the weekday
`YEARWEEK`	The `YEARWEEK` function returns the year and the week number of the date specified in the argument. The syntax for this function is: `SELECT YEARWEEK(expression);`	For example, to retrieve the year and week for the date `1982-12-29`, enter the following command at the command prompt: `SELECT YEARWEEK('1982-12-29');` The output of this function is:`198252`

CHAR Function 1-2

- The CHAR function interprets the values specified in the argument as integers
- This function returns a string that contain characters returned by the code value of the integers
- If the arguments contain NULL values, then they are skipped
- The syntax to retrieve the string of characters is:

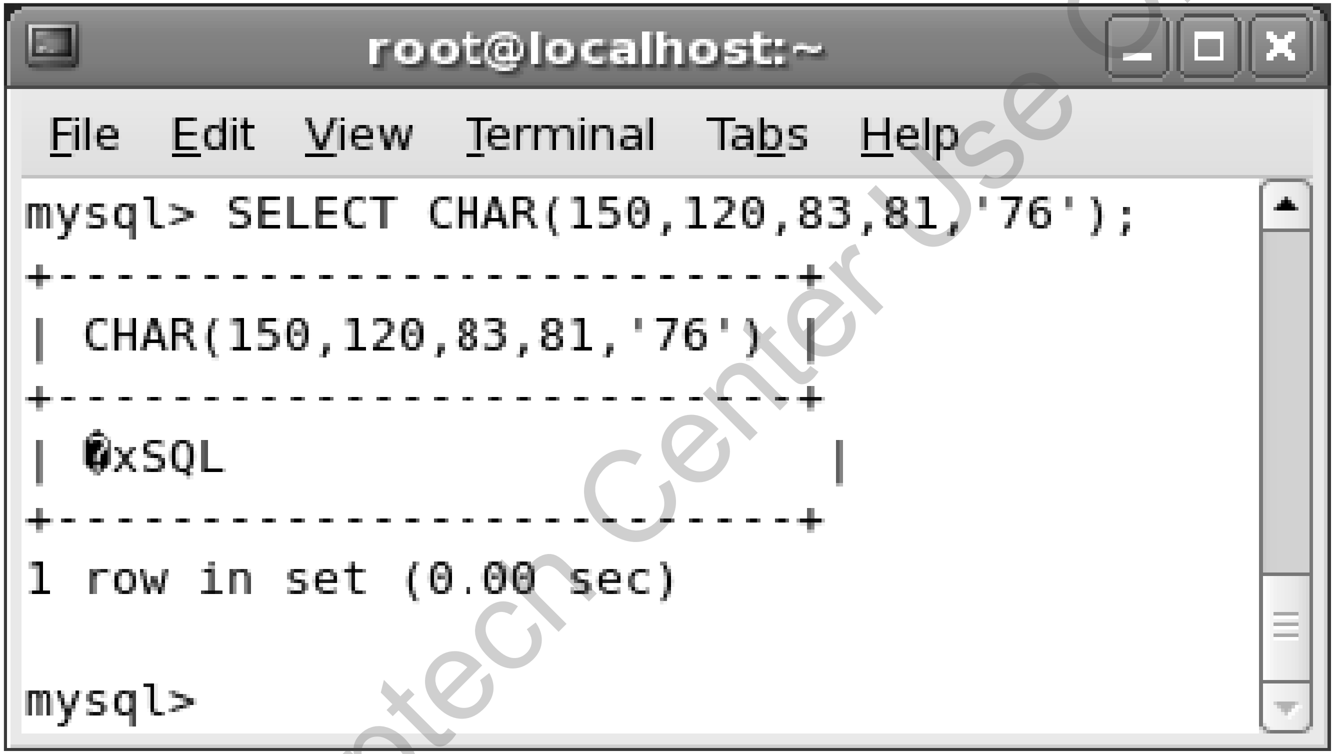
```
SELECT CHAR(N,...);
```

- For example, to retrieve the code value for the integers 150, 120, 83, 81, ‘76’, enter the following command at the command prompt:

```
SELECT CHAR(150, 120, 83, 81, ‘76’);
```

CHAR Function

Figure displays the output of the command



```
mysql> SELECT CHAR(150,120,83,81,'76');
+--------------------------+
| CHAR(150,120,83,81,'76') |
+--------------------------+
| �xSQL                     |
+--------------------------+
1 row in set (0.00 sec)

mysql>
```

CHARACTER_LENGTH Function

- The CHARACTER_LENGTH function also known as CHAR_LENGTH function returns the number of characters present in the string

- This function counts a multibyte character as a single character

- The syntax for this function is:

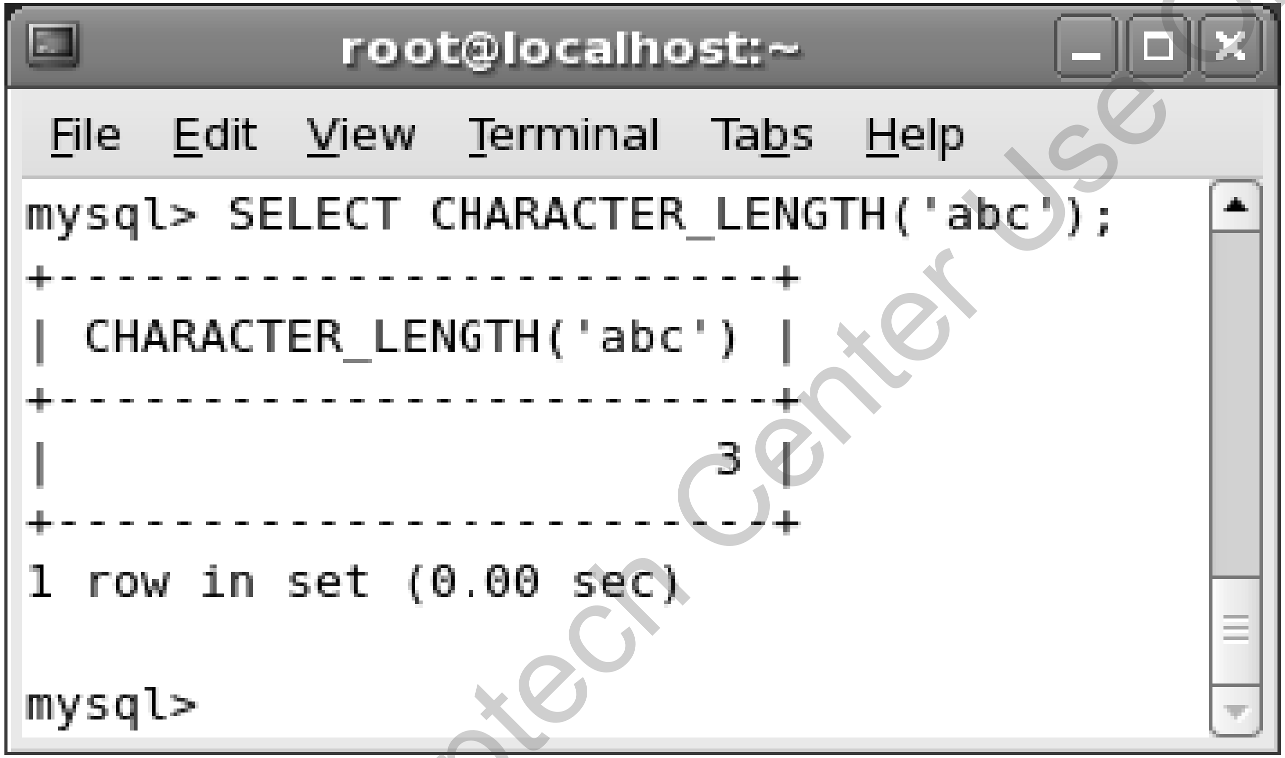
```
SELECT CHARACTER_LENGTH(expression);
```

- For example, to calculate the number of characters present in the string abc, enter the following command at the command prompt:

```
SELECT CHARACTER_LENGTH('abc');
```

CHARACTER_LENGTH Function 2-2

Figure displays the output of the command



```
mysql> SELECT CHARACTER_LENGTH('abc');
+-------------------------+
| CHARACTER_LENGTH('abc') |
+-------------------------+
|                       3 |
+-------------------------+
1 row in set (0.00 sec)

mysql>
```

CONCAT Function

- The CONCAT function returns a string after joining the specified arguments
- This function returns a NULL value if NULL arguments are specified
- The syntax for this function is:

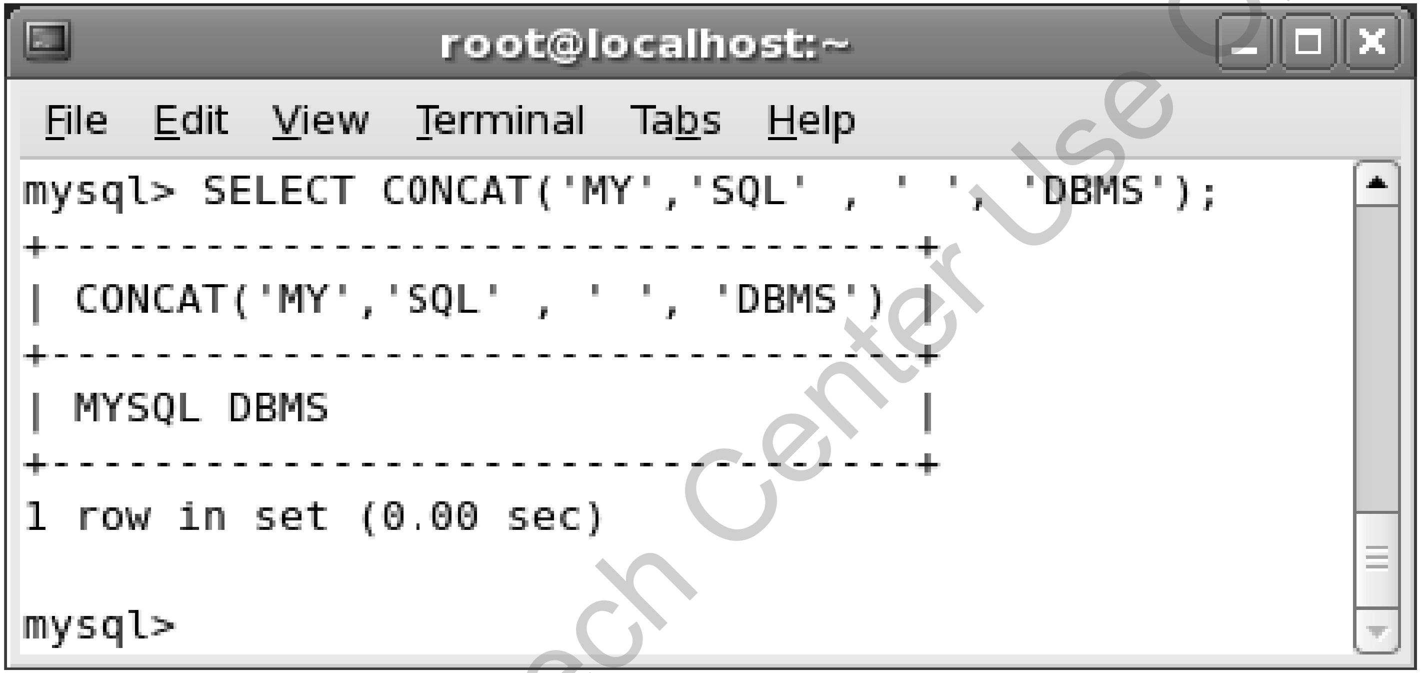
```
SELECT CONCAT(STR1,STR2,...);
```

- To concatenate the strings My, SQL, DBMS, enter the following command at the command prompt:

```
SELECT CONCAT('My','SQL',' ','DBMS');
```

CONCAT Function

Figure displays the output of the command



```
mysql> SELECT CONCAT('MY','SQL' , ' ', 'DBMS');
+---------------------------------+
| CONCAT('MY','SQL' , ' ', 'DBMS') |
+---------------------------------+
| MYSQL DBMS                      |
+---------------------------------+
1 row in set (0.00 sec)

mysql>
```

INSERT Function

- The INSERT function adds a new string to the existing string at a specified position
- The output of the function is also a string
- The syntax for the INSERT function is:

```
INSERT (str, pos, len, newstr);
```

where,

INSERT – adds a new string

str – specifies the existing string

pos – defines the starting position of the new string

len – specifies the length of the new string

newstr – specifies the new string to be inserted

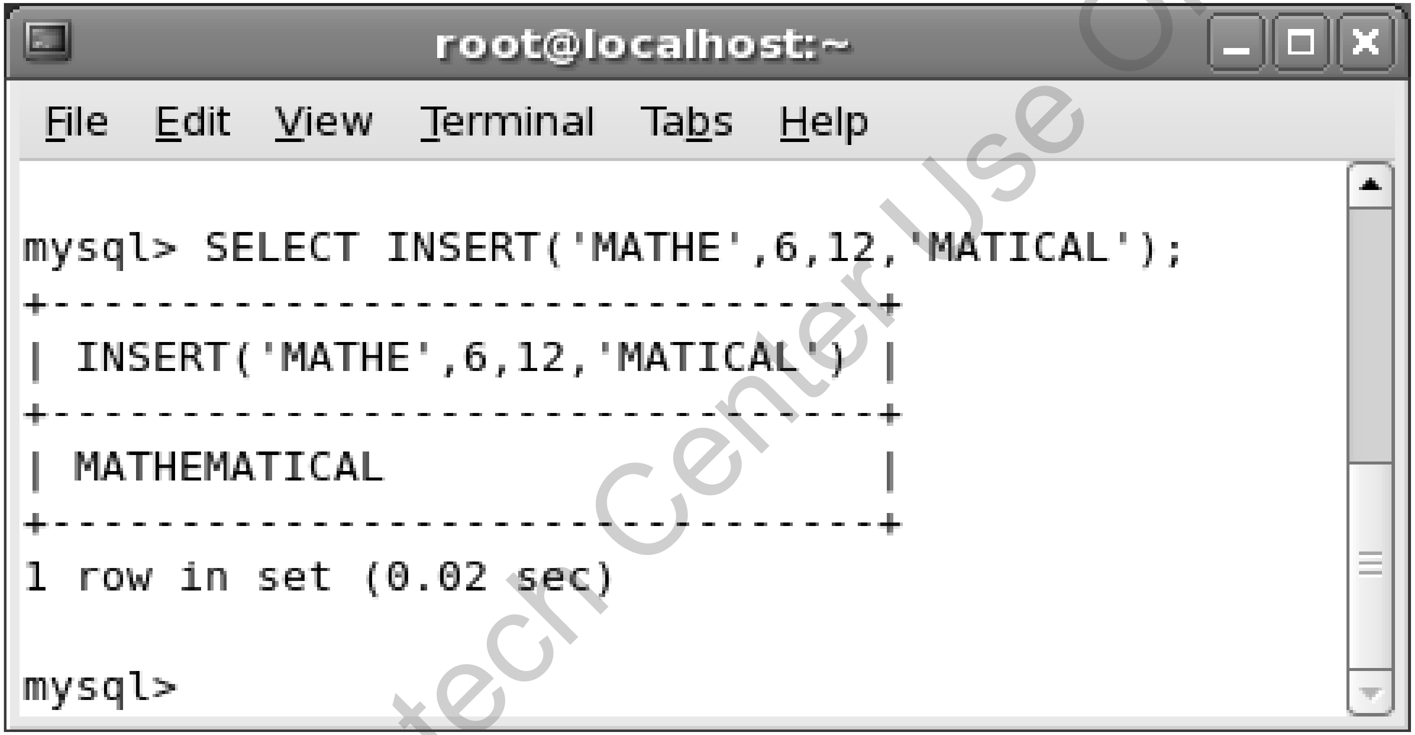
INSERT Function

- To insert MATHEMATICAL at position 6 of the string MATHE, enter the following command at the command prompt:

```
SELECT INSERT('MATHE', 6, 12, 'MATICAL');
```

INSERT Function

Figure displays the output of the command



```
mysql> SELECT INSERT('MATHE',6,12,'MATICAL');
+--------------------------------+
| INSERT('MATHE',6,12,'MATICAL') |
+--------------------------------+
| MATHEMATICAL                   |
+--------------------------------+
1 row in set (0.02 sec)

mysql>
```

INSTR Function

- The `INSTR` function works with two arguments, such as string and substring
- It returns the position of the first instance of a substring in a string
- The output of this function is a numeric value ranging from 0 to the length of the string
- The syntax for this function is:

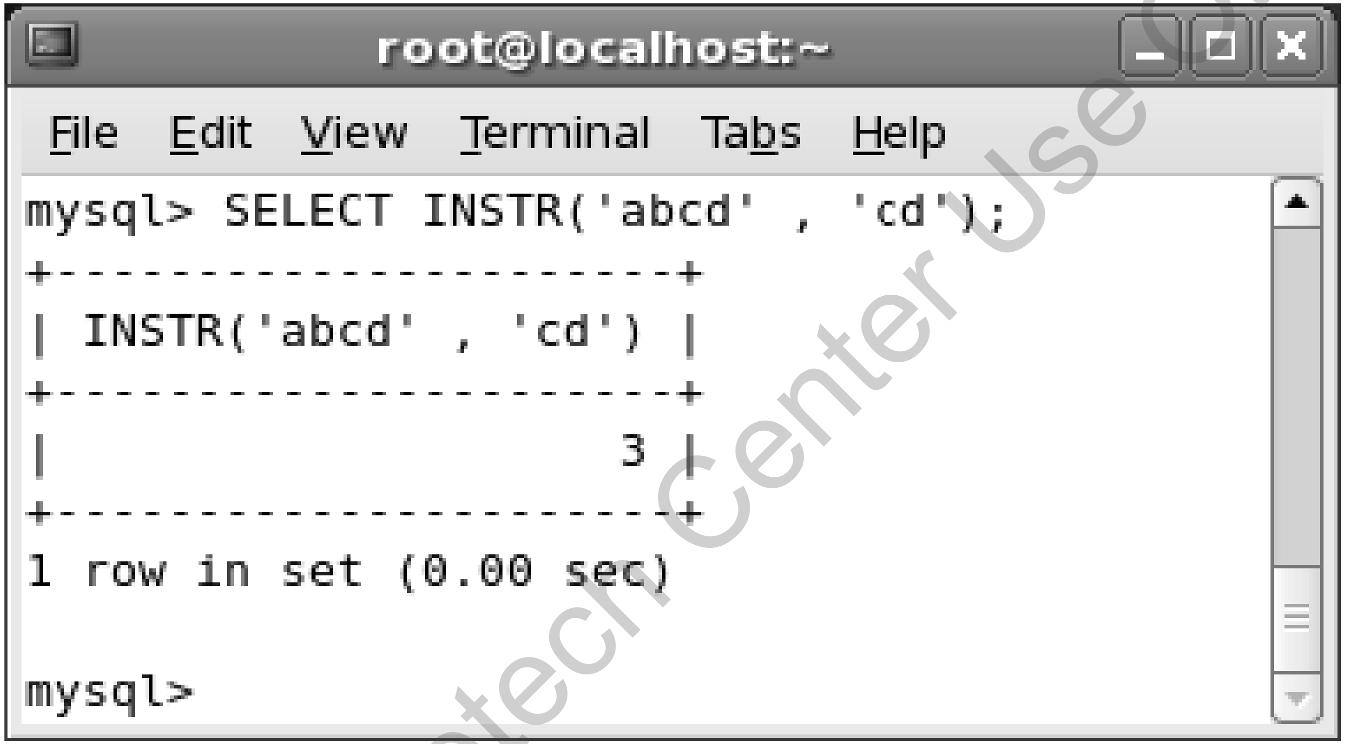
```
SELECT INSTR(string, substring);
```

- For example, to retrieve the position of the string cd from the string `abcd`, enter the following command at the command prompt:

```
SELECT INSTR('abcd', 'cd');
```

INSTR Function

Figure displays the output of the command



```
mysql> SELECT INSTR('abcd' , 'cd');
+----------------------+
| INSTR('abcd' , 'cd') |
+----------------------+
|                    3 |
+----------------------+
1 row in set (0.00 sec)

mysql>
```

LCASE Function

- The LCASE function changes all the characters of the values specified in the argument to lowercase
- This function does not work with binary strings
- The syntax for this function is:

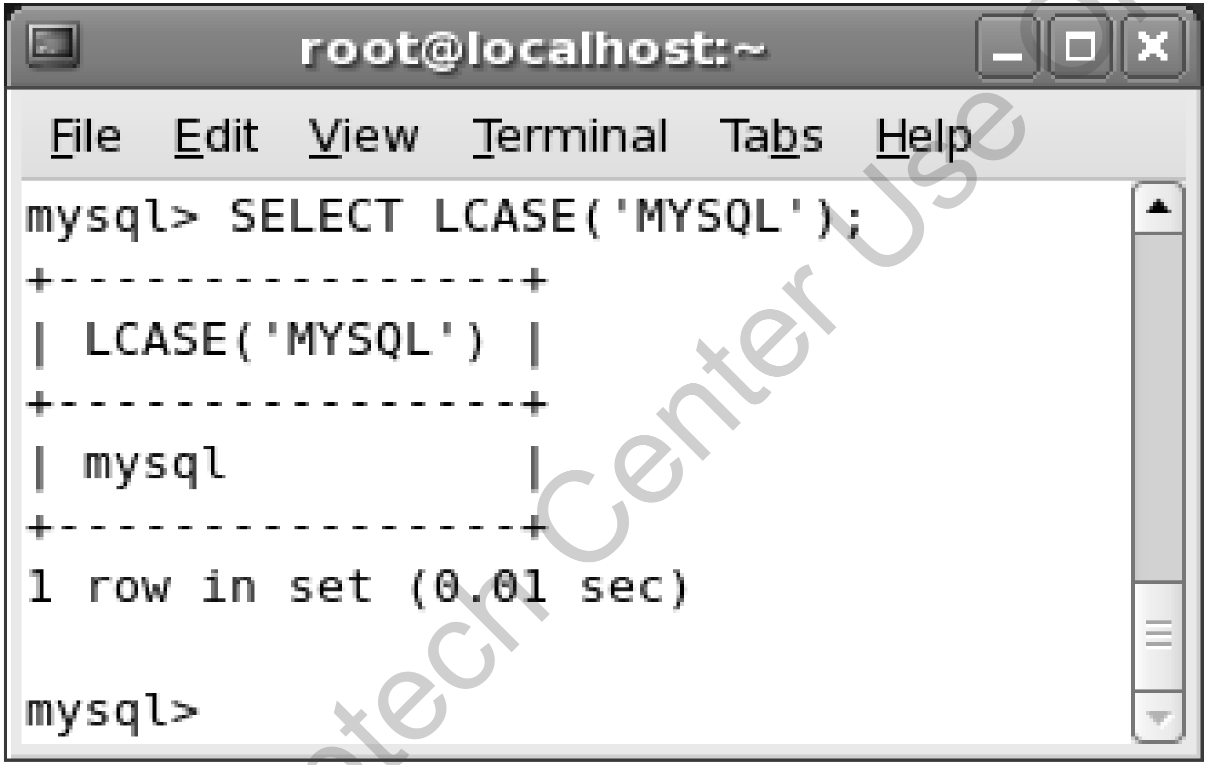
```
SELECT LCASE(expression);
```

- To convert MYSQL to lowercase, enter the following command at the command prompt:

```
SELECT LCASE(‘MYSQL’);
```

LCASE Function

Figure displays the output of the command



```
mysql> SELECT LCASE('MYSQL');
+----------------+
| LCASE('MYSQL') |
+----------------+
| mysql          |
+----------------+
1 row in set (0.01 sec)

mysql>
```

LEFT Function 1-2

- The LEFT function returns the leftmost characters of the values specified in the argument
- The syntax for this function is:

```
SELECT LEFT('str', len);
```

where,

LEFT – returns the leftmost character

str – contains the original string

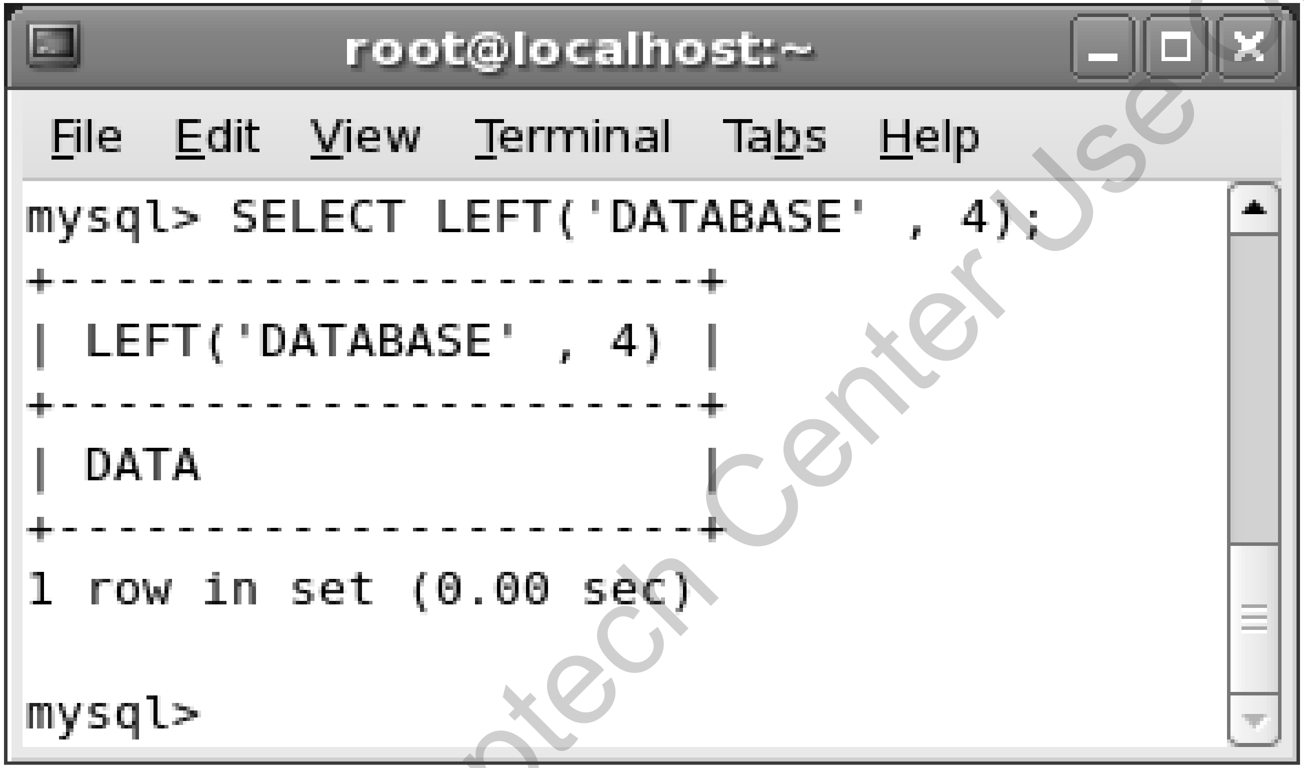
len – specifies the number of characters to be extracted from the leftmost end

- For example, to obtain the first four character of the string DATABASE, enter the following command at the command prompt:

```
SELECT LEFT('DATABASE' , 4);
```

LEFT Function

Figure displays the output of the command



```
mysql> SELECT LEFT('DATABASE' , 4);
+-----------------------+
| LEFT('DATABASE' , 4) |
+-----------------------+
| DATA                  |
+-----------------------+
1 row in set (0.00 sec)

mysql>
```

LENGTH Function

- The LENGTH function returns the length of a string
- The syntax to calculate the length of a string is:

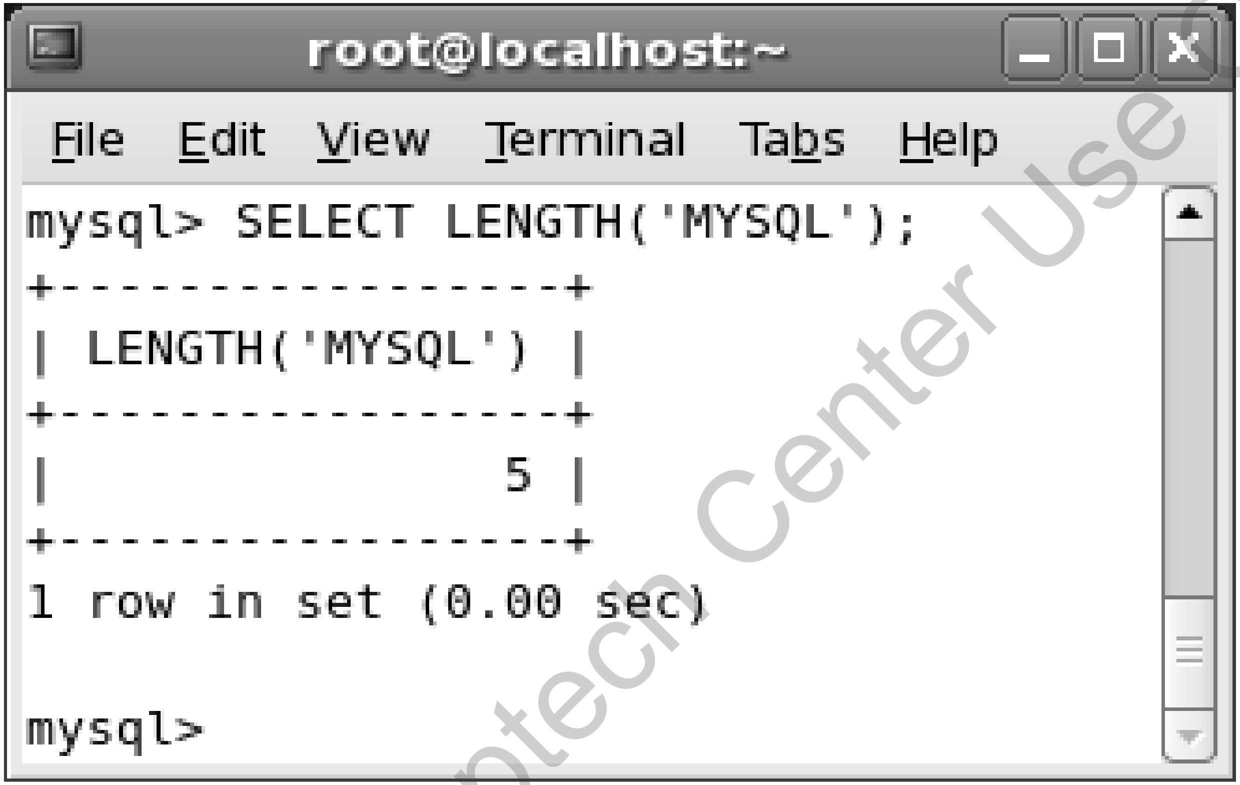
```
SELECT LENGTH(str);
```

- For example, to calculate the length of the string MYSQL, enter the following command at the command prompt:

```
SELECT LENGTH('MYSQL');
```

LENGTH Function

Figure displays the output of the command



```
mysql> SELECT LENGTH('MYSQL');
+-------------------+
| LENGTH('MYSQL') |
+-------------------+
|                 5 |
+-------------------+
1 row in set (0.00 sec)

mysql>
```

LOCATE Function 1-2

- The LOCATE function returns the position of the first instance of a substring in a string
- The syntax to retrieve the position of the first instance of a substring in a string is:

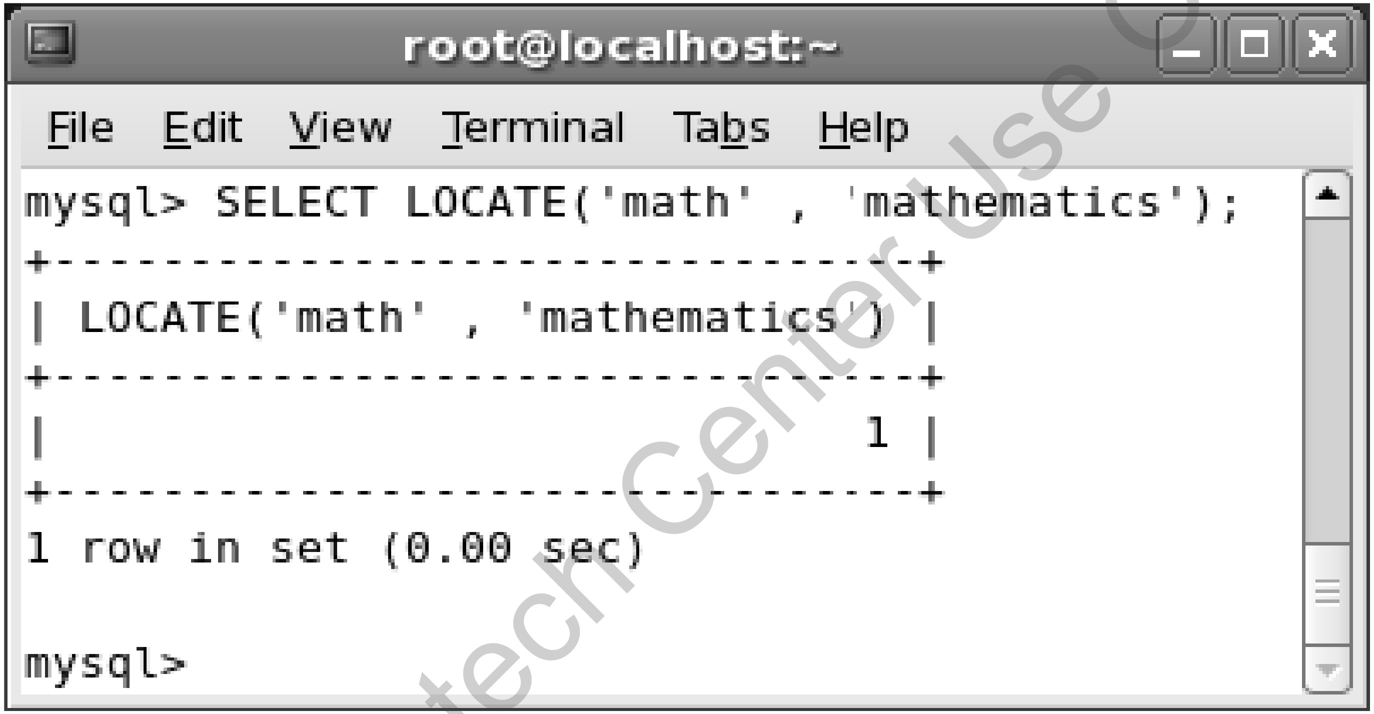
```
SELECT LOCATE(substr, str);
```

- For example, to identify the position of the substring math in the string mathematics, enter the following command at the command prompt:

```
SELECT LOCATE(‘math’, ‘mathematics’);
```

LOCATE Function

Figure displays the output of the command



```
mysql> SELECT LOCATE('math' , 'mathematics');
+--------------------------------+
| LOCATE('math' , 'mathematics') |
+--------------------------------+
|                              1 |
+--------------------------------+
1 row in set (0.00 sec)

mysql>
```

REPEAT Function

- The `REPEAT` function copies the given string N number of times
- If N is less than 1, the output is an empty string
- The syntax for this function is:

```
SELECT REPEAT(String, count);
```

where,

`REPEAT` – copies the string

`String` – specifies the string to copy

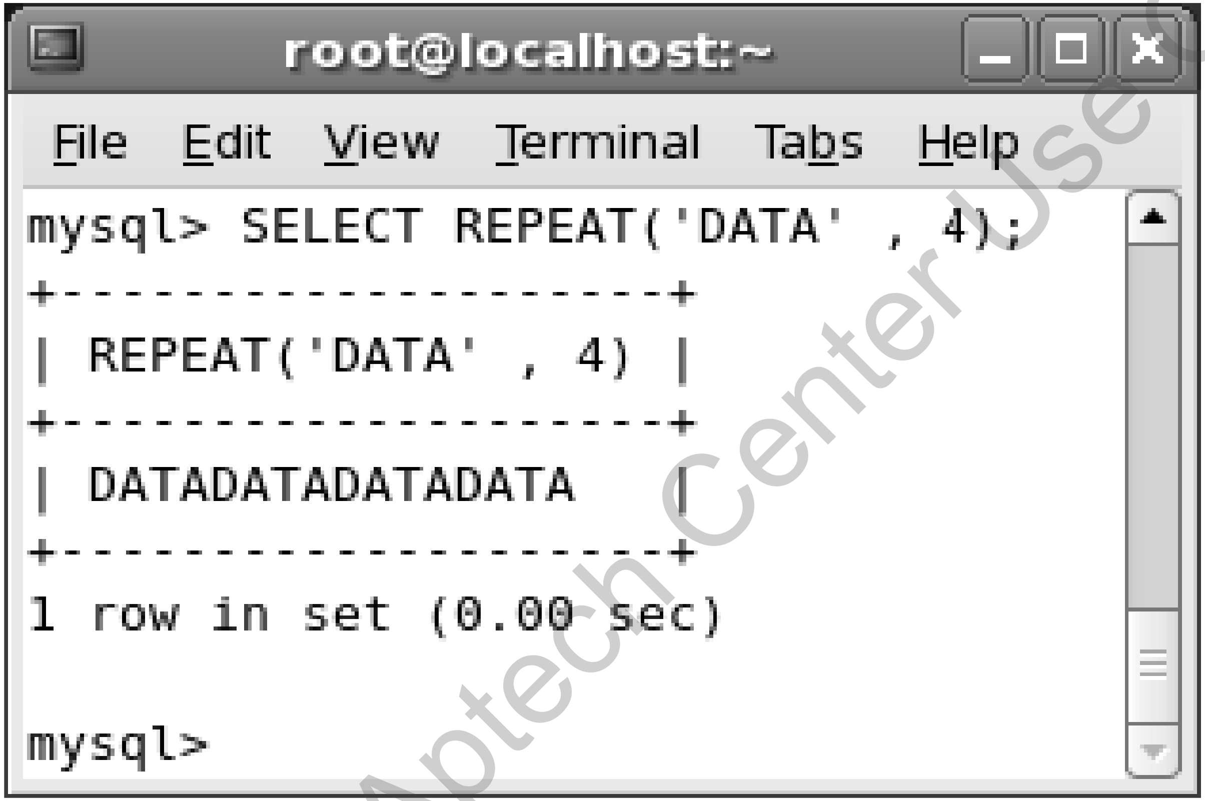
`count` – specifies the number of times to copy the string

- To replicate the string DATA four times, enter the following command at the command prompt:

```
SELECT REPEAT('DATA', 4);
```

REPEAT Function

Figure displays the output of the command



```
mysql> SELECT REPEAT('DATA' , 4);
+---------------------+
| REPEAT('DATA' , 4) |
+---------------------+
| DATADATADATADATA    |
+---------------------+
1 row in set (0.00 sec)

mysql>
```

REPLACE Function 1-2

- The REPLACE function modifies a part or whole of the string with the new string

- The syntax for this function is:

```
SELECT REPLACE(str, from_str, to_str);
```

where,

REPLACE – modifies the string

str – specifies the original string

from_str – defines the string to be replaced

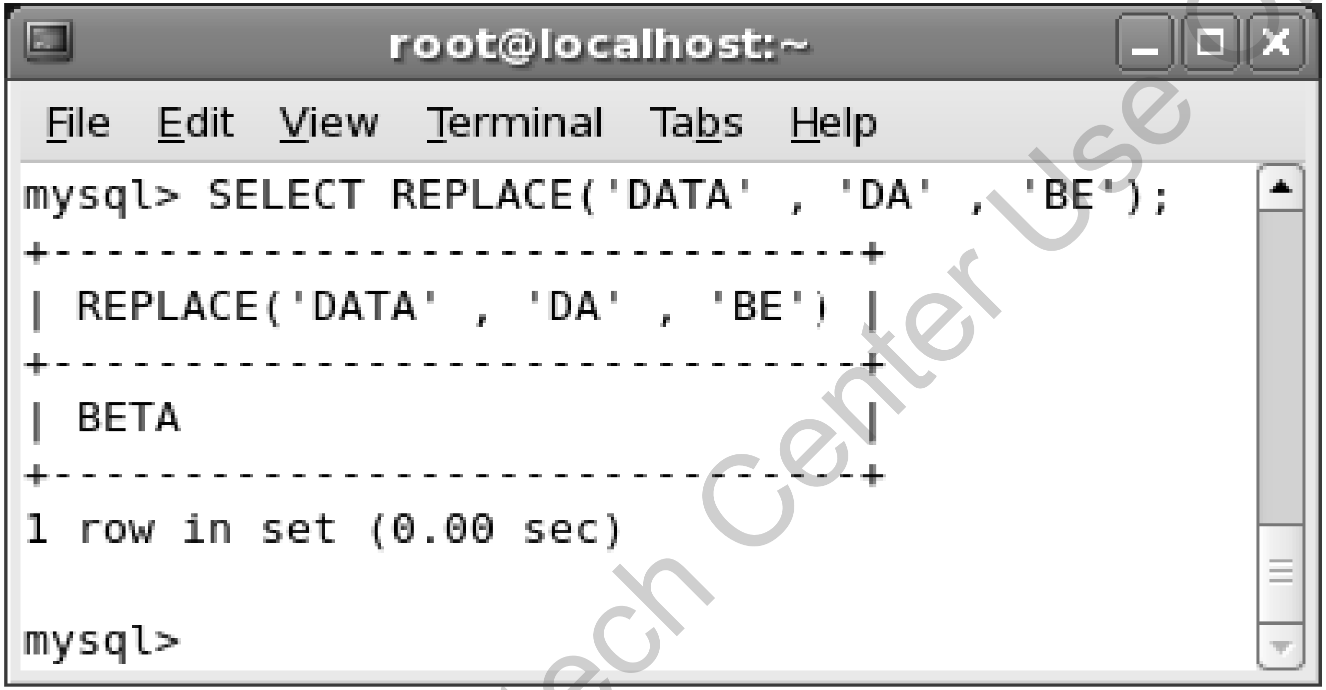
to_str – defines the new string to be inserted

- To change the string DATA to BETA, enter the following command at the command prompt:

```
SELECT REPLACE('DATA', 'DA', 'BE');
```

REPLACE Function

Figure displays the output of the command



```
mysql> SELECT REPLACE('DATA' , 'DA' , 'BE');
+---------------------------------+
| REPLACE('DATA' , 'DA' , 'BE') |
+---------------------------------+
| BETA                            |
+---------------------------------+
1 row in set (0.00 sec)

mysql>
```

REVERSE Function 1-2

- The REVERSE function inverts the sequence of characters in an argument
- The syntax for this function is:

```
SELECT REVERSE(expression);
```

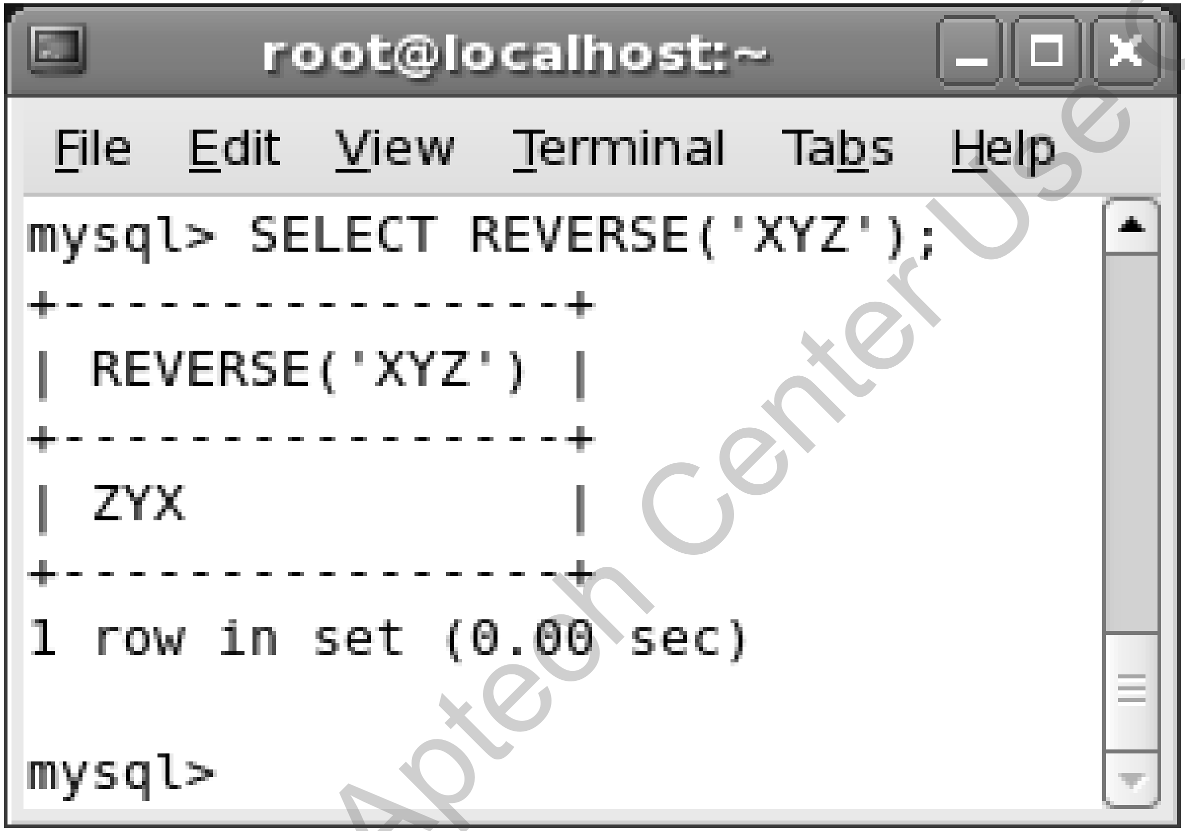
- To change the order of characters of the string XYZ, enter the following command at the command prompt:

```
SELECT REVERSE('XYZ');
```

REVERSE Function

2-2

Figure displays the output of the command



```
mysql> SELECT REVERSE('XYZ');
+-----------------+
| REVERSE('XYZ')  |
+-----------------+
| ZYX             |
+-----------------+
1 row in set (0.00 sec)

mysql>
```

RIGHT Function 1-2

- The `RIGHT` function accepts two arguments, such as string and length
- The function returns an output that contains string values starting from the right end of the input string
- The syntax for this function is:

```
SELECT RIGHT(string, length);
```

where,

`string` – defines the given string

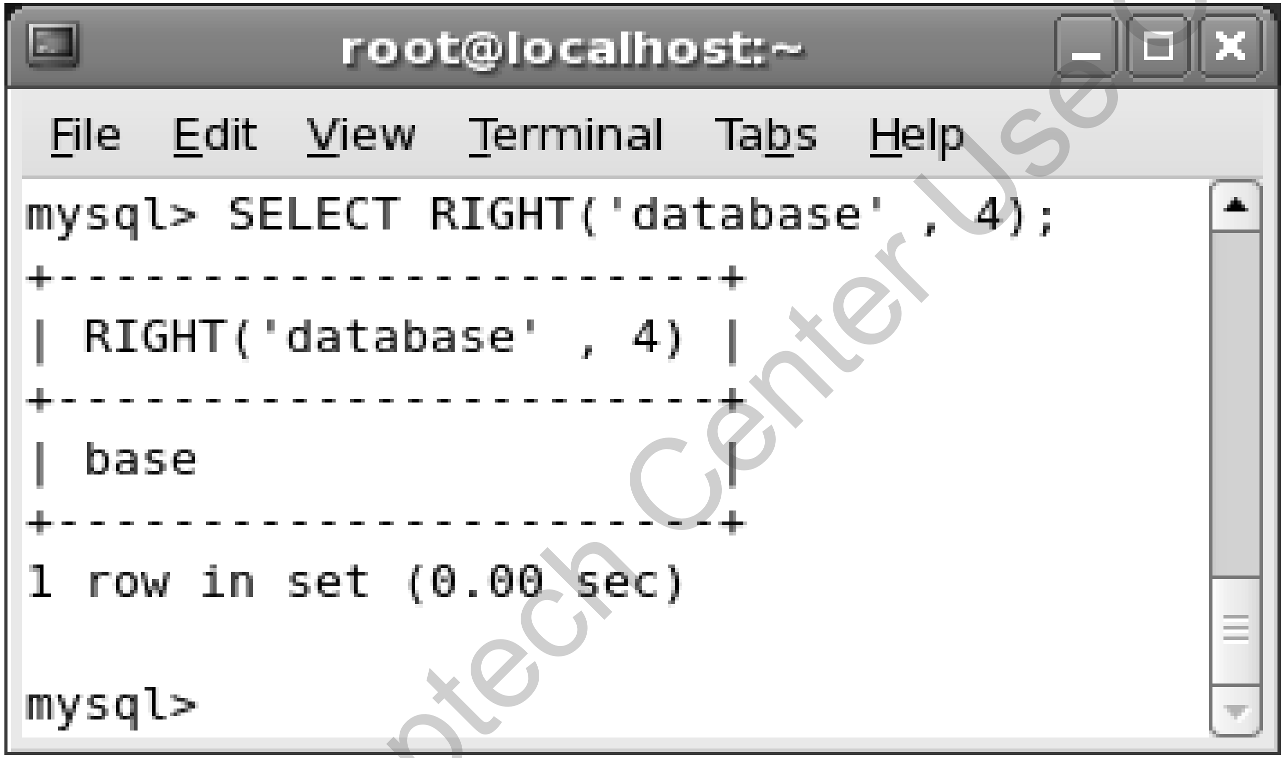
`length` – specifies the number of rightmost characters

- For example, to obtain the last four characters of the string database, enter the following command at the command prompt:

```
SELECT RIGHT('database', 4);
```

RIGHT Function

Figure displays the output of the command



```
mysql> SELECT RIGHT('database' , 4);
+------------------------+
| RIGHT('database' , 4) |
+------------------------+
| base                   |
+------------------------+
1 row in set (0.00 sec)

mysql>
```

STRCMP Function

- The `STRCMP` function accepts two strings as arguments
- It compares the strings and returns a numeric value ranging from -1 to 1 in the output
- The function returns 0 if the strings have same values, -1 when the first argument is smaller than the second and 1 when the first argument is greater than the second

STRCMP Function 2-3

- The syntax for this function is:

```
SELECT STRCMP(str1, str2);
```

where,

`str1, str2` – specify the strings to compare

If `str1 = str2`, the output is 0

If `str1 < str2`, the output is -1

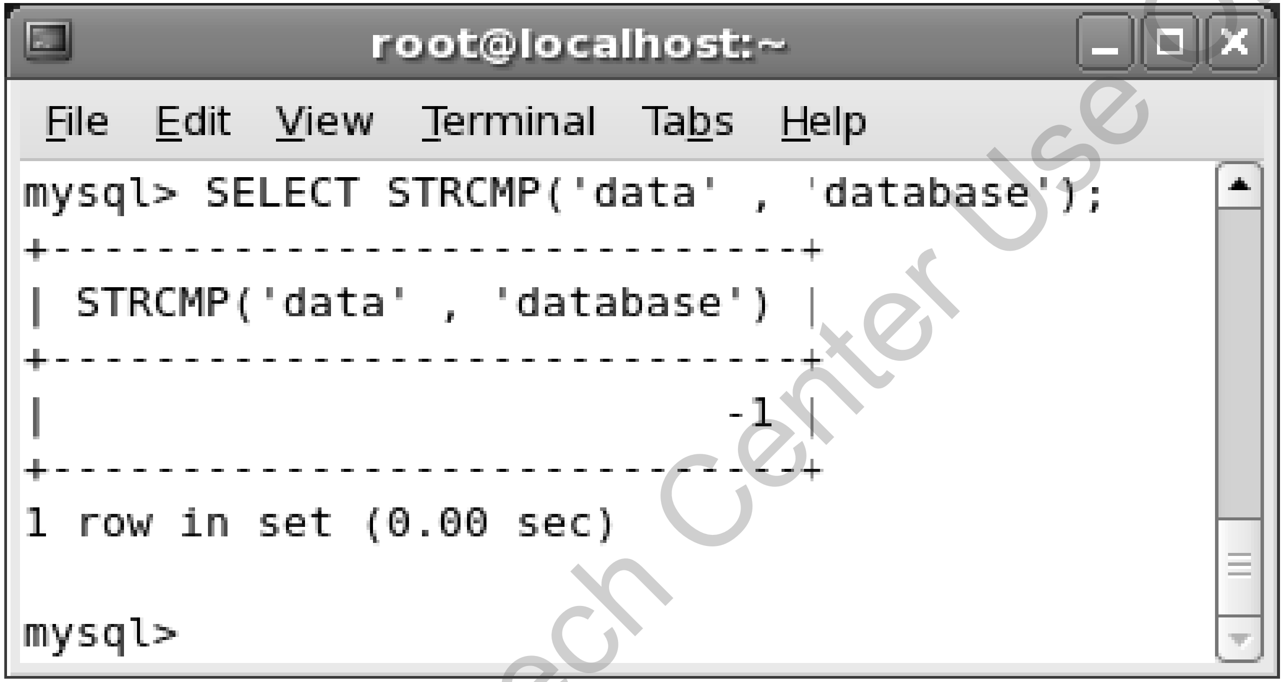
If `str1 > str2`, the output is 1

- To compare two strings data and database, enter the following command at the command prompt:

```
SELECT STRCMP('data', 'database');
```

STRCMP Function

Figure displays the output of the command



```
mysql> SELECT STRCMP('data' , 'database');
+------------------------------+
| STRCMP('data' , 'database') |
+------------------------------+
|                           -1|
+------------------------------+
1 row in set (0.00 sec)

mysql>
```

TRIM Function 1-2

- The TRIM function removes a specified prefix or suffix from a given string

- The syntax for this function is:

```
SELECT TRIM(location ‘remstring’ from ‘string’
```

where,

`location` – specifies the starting position for deletion

`remstring` – specifies the string to be removed

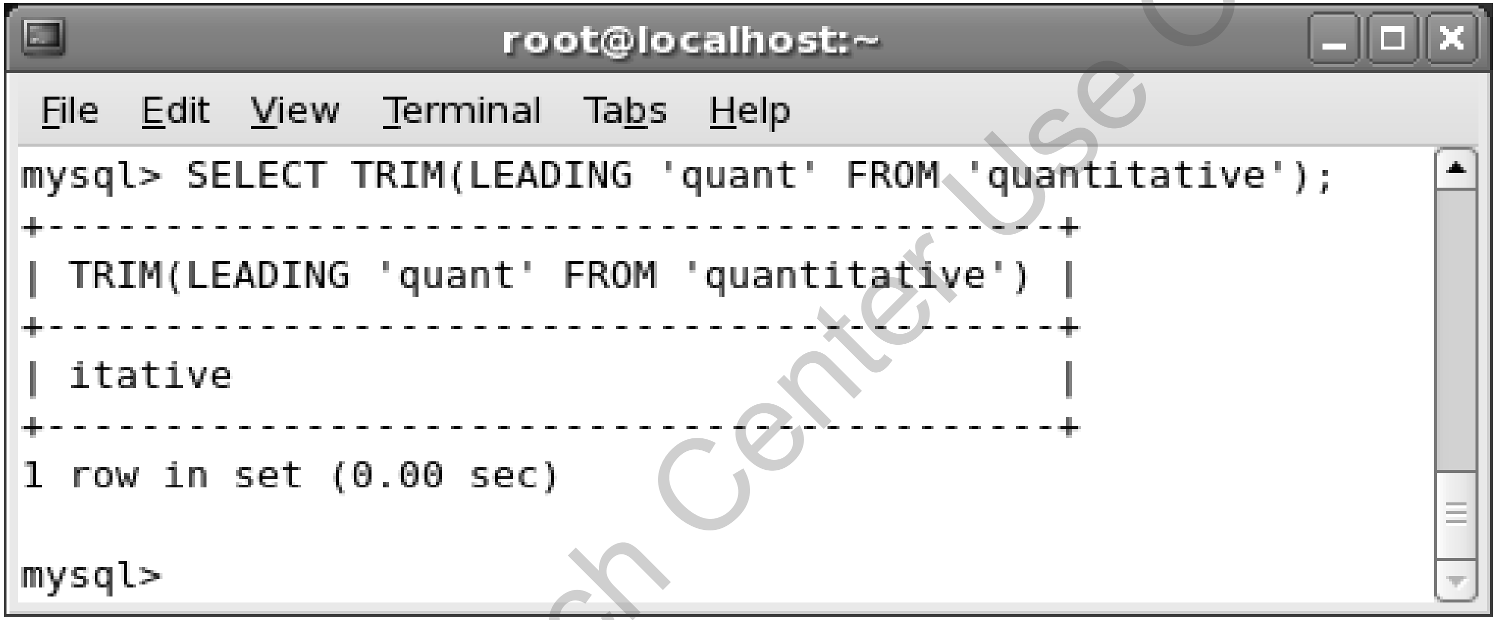
`string` – specifies the original string

- To remove the string quant from the string quantitative, enter the following command at the command prompt:

```
SELECT TRIM(LEADING ‘quant’ from ‘quantitative’);
```

TRIM Function

Figure displays the output of the command



```
mysql> SELECT TRIM(LEADING 'quant' FROM 'quantitative');
+--------------------------------------------+
| TRIM(LEADING 'quant' FROM 'quantitative') |
+--------------------------------------------+
| itative                                    |
+--------------------------------------------+
1 row in set (0.00 sec)

mysql>
```

UCASE Function

- The `UCASE` function changes all the characters of the argument into uppercase
- This function does not work with binary strings
- The syntax for this function is:

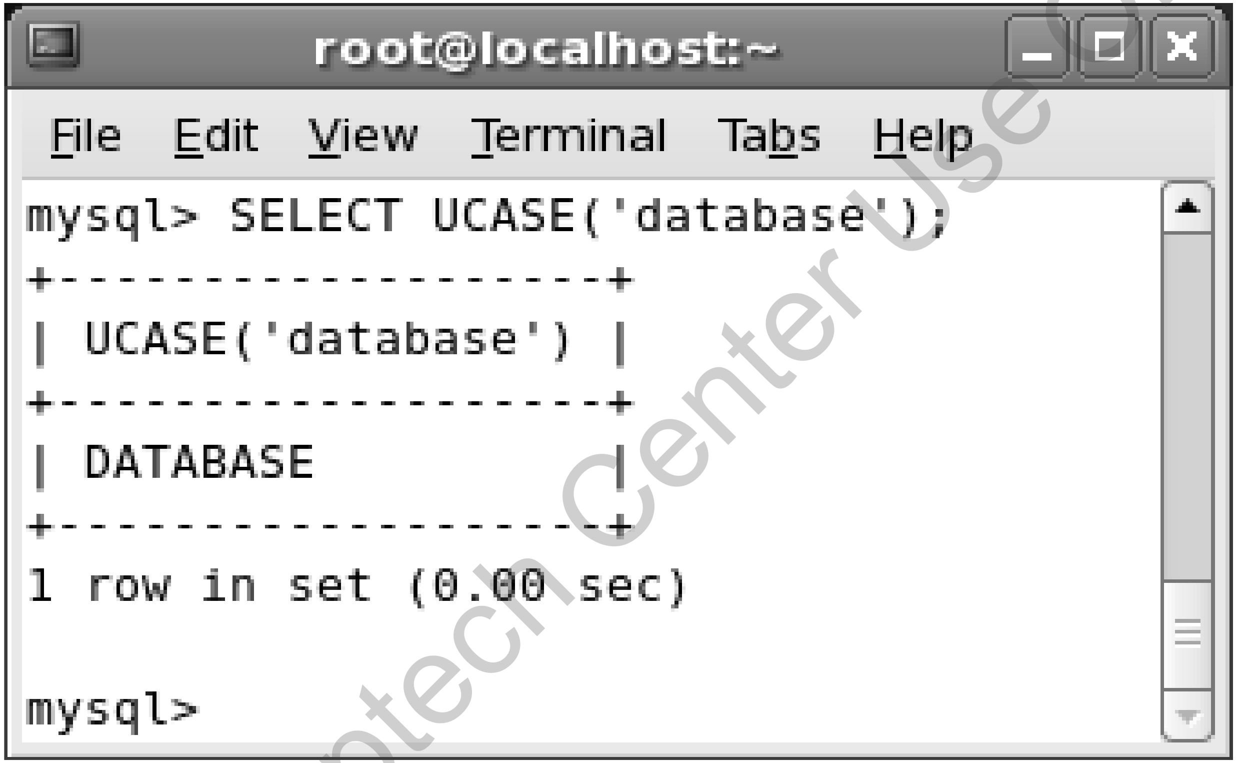
```
SELECT UCASE(expression);
```

- To convert database to uppercase, enter the following command at the command prompt:

```
SELECT UCASE(‘database’);
```

UCASE Function

Figure displays the output of the command



```
mysql> SELECT UCASE('database');
+-------------------+
| UCASE('database') |
+-------------------+
| DATABASE          |
+-------------------+
1 row in set (0.00 sec)

mysql>
```

UCASE Function 3-10

Table describes the additional String functions in MySQL

Name	Description	Example
ASCII	The ASCII function returns the numeric value of the leftmost character of the string, specified as the argument. This function returns a value of 0 if you enter an empty string. This function is applicable for characters whose numeric value ranges from 0 to 255. The syntax to retrieve the ASCII value is: `SELECT ASCII(string);`	For example, to retrieve the ASCII value of the string JOHN, enter the following command at the command prompt: `SELECT ASCII(‘JOHN’);` The output of this function is: 74
BIN	The BIN function returns a string representation of the binary value of N, where N contains values of BIGINT data type. The syntax to retrieve the string representation is: `SELECT BIN(expression);`	For example, to retrieve the binary value of 55, enter the following command at the command prompt: `SELECT BIN(55);` The output of this function is: 110111

UCASE Function

Name	Description	Example
BIT_LENGTH	The BIT_LENGTH function returns the length of the string specified in the argument. The syntax to calculate the length of the string in bits is: SELECT BIT_LENGTH(string);	For example, to calculate the length of the string MYSQL in bits, enter the following command at the command prompt: SELECT BIT_LENGTH('MYSQL'); The output of this function is: 40
COMPRESS	The COMPRESS function compresses a string. The syntax to compress a string is: SELECT COMPRESS(string) The compressed string outputs are stored as follows: ◆ Empty strings are stored as it is ◆ Non-empty strings are stored as a four-byte length of the uncompressed string (low byte first), followed by the compressed string. ◆ If the string contains a space at the end, the space is replaced by the '.' character to resolve trim errors while storing data in a CHAR or VARCHAR column.	To compress the string XYZ and obtain the length of the compressed string, enter the following command at the command prompt: SELECT LENGTH(COMPRESS('XYZ')); The output of this function is: 15

UCASE Function 5-10

Name	Description	Example
`CONCAT_WS`	The `CONCAT_WS` function is similar to the `CONCAT()` function. However, it allows you to add a separator between the arguments. The syntax for this function is: `CONCAT_WS(separator, str1, str2, ....);`	To join the strings My and `SQL` with a comma as the separator, enter the following command at the command prompt: `SELECT CONCAT_WS(‘,’ ,’My’,’SQL’);` The output of this function is: `MY,SQL`
`ELT`	The `ELT` function extracts the value of the string whose position is specified in the argument. The position is a numeric value. The syntax for using this function is: `SELECT ELT(N, str1, str2, ...);` where, `N` - specifies the numeric position of the value to be searched `str1, str2` - defines the strings passed as arguments	Enter the following command at the command prompt: `SELECT ELT(1, ‘a’, ‘b’, ‘c’);` The output of this function is: `a`

UCASE Function

Name	Description	Example
FIELD	The FIELD function returns the position of a particular string in a given set of strings specified in the argument. If the string is not found, this function returns a value of zero. The syntax for this function is: SELECT FIELD(string1,string2,string3....); where, string1 – represents the string to search for string2, string3 – list of strings to be searched from	To find the index of string C in the strings A, B, C, D, and E, enter the following command at the command prompt: SELECT FIELD(‘C’, ‘A’, ‘B’, ‘C’, ‘D’, ‘E’); The output of this function is: 3
FIND_IN_SET	The FIND_IN_SET function returns the position of the character present in the string list passed as an argument. A string list contains strings that are separated by a comma. The output of this function is a numeric value between 0 and N, where N is the number of substrings in the string list. If either of the argument is NULL, the final result is also NULL. The syntax for this function is: SELECT FIND_IN_SET(expression);	For example, to retrieve the index value of B in the string B,Y,E, enter the following command at the command prompt: SELECT FIND_IN_SET(‘B’, ‘B,Y,E’); The output of this function is: 1

UCASE Function 7-10

Name	Description	Example
`HEX`	The `HEX` function displays the values specified in the argument in its hexadecimal format. The syntax for this function is: `SELECT HEX(expression);`	For example, to retrieve the hexadecimal value of `MYSQL`, enter the following command at the command prompt: `SELECT HEX(‘MYSQL’);` The output of this function is:`4D5953514C`
`LOWER`	The `LOWER` function changes the given string into lowercase. The syntax to change the case for a string is: `SELECT LOWER(str);`	For example, to convert the string `PETER` to lowercase, enter the following command at the command prompt: `SELECT LOWER(‘PETER’);` The output of this function is: `peter`

UCASE Function

Name	Description	Example
LTRIM	The LTRIM function removes the whitespaces before the values specified in the argument. The syntax for this function is: SELECT LTRIM(expression);	To delete the whitespaces before ' xyz', enter the following command at the command prompt: SELECT LTRIM(' xyz'); The output of this function is:xyz
ORD	The ORD function returns the ASCII value of the first character for the string specified in the argument. The arguments should be a string. The syntax for this function is: SELECT ORD(expression);	To display the ASCII value of the first character from the string BAR, enter the following command at the command prompt: SELECT ORD('BAR'); The output of this function is: 66

UCASE Function

Name	Description	Example
RTRIM	The RTRIM function deletes the spaces after the values specified in the argument. The syntax for this function is: SELECT RTRIM(expression);	To remove the trailing space from the string ‘SPACE ’, enter the following command at the command prompt: SELECT RTRIM(‘SPACE ’); The output of this function is: SPACE
SPACE	The SPACE function returns a string that contains only space characters. You can specify the number of space characters as a numeric value in the argument. The syntax for this function is: SELECT SPACE(N); where, N – defines the number of spaces	For example, to obtain a string with seven spaces, enter the following command at the command prompt: SELECT SPACE(7); The output of this function will be a string of 7 spaces

UCASE Function

Name	Description	Example
UNHEX	The UNHEX function interprets each character of the string in the hexadecimal format and returns the character as the output. The syntax for this function is: SELECT UNHEX(expression);	For example, to obtain the characters for the hexadecimal string ‘AB’, enter the following command at the command prompt: SELECT UNHEX(‘AB’); The output of this function is: ?

CHARSET Function

- The `CHARSET` function returns the character set of the argument

- The argument entered must be a string argument

- The syntax for obtaining the character set of a string is:

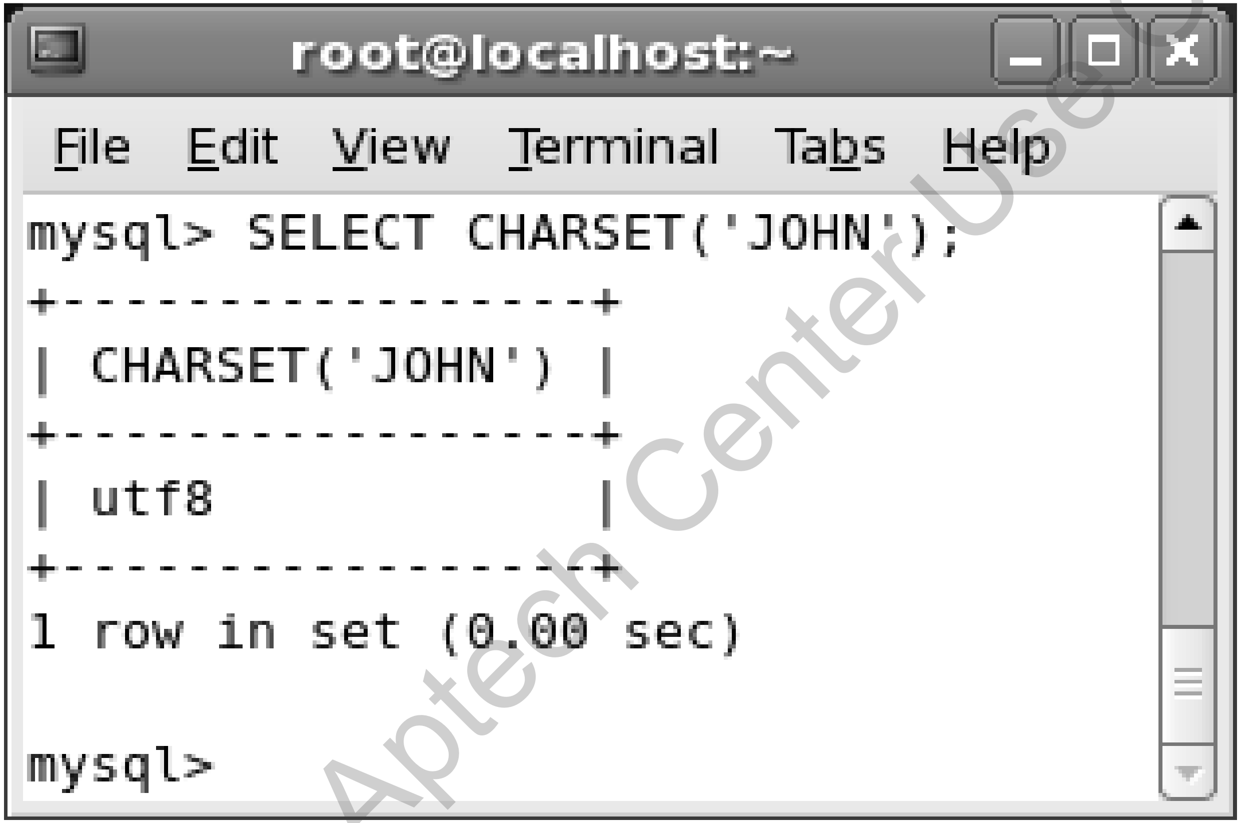
```
SELECT CHARSET(str);
```

- For example, to obtain the character set of ‘`John`’, enter the following command at the command prompt:

```
SELECT CHARSET(‘JOHN’);
```

CHARSET Function

Figure displays the output of the command



```
mysql> SELECT CHARSET('JOHN');
+-----------------+
| CHARSET('JOHN') |
+-----------------+
| utf8            |
+-----------------+
1 row in set (0.00 sec)

mysql>
```

CURRENT_USER Function

- The CURRENT_USER function returns the username and the hostname of the current session
- The syntax to view the hostname and the username is:

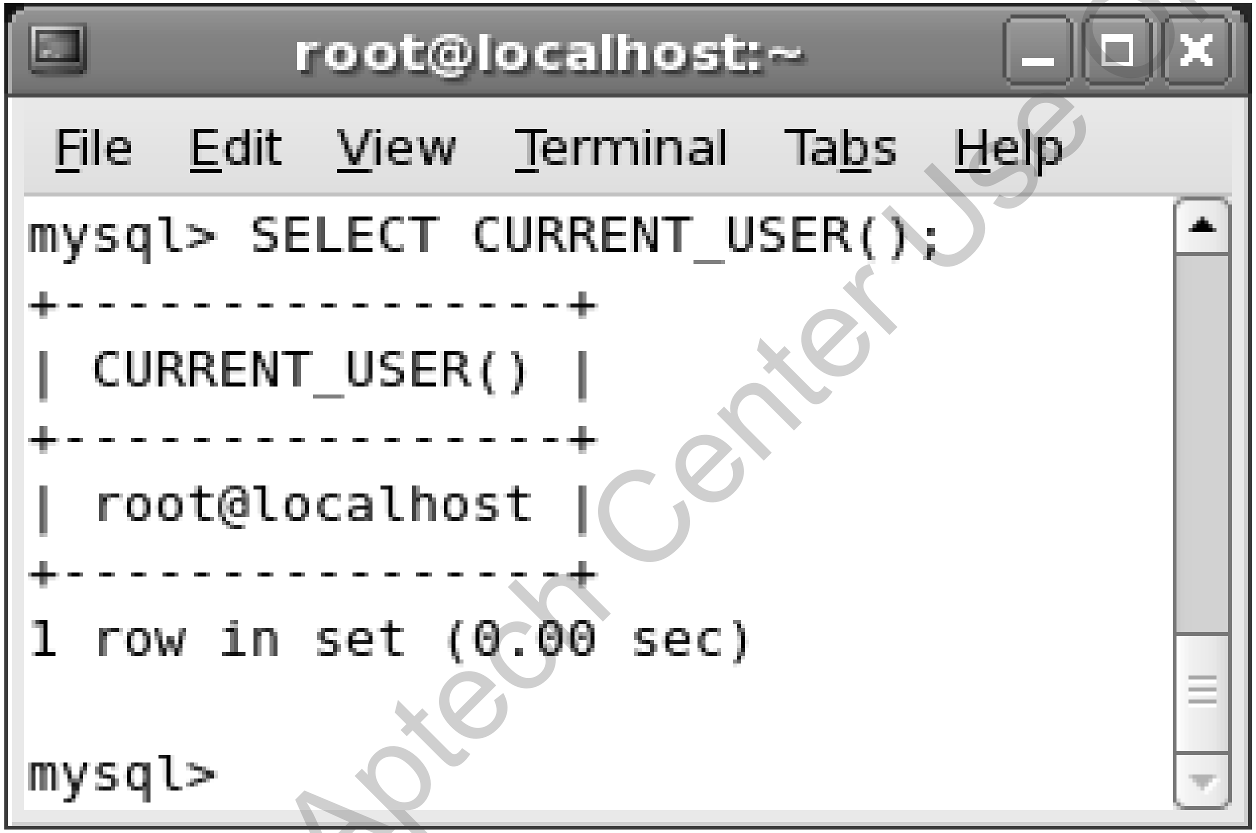
```
SELECT CURRENT_USER();
```

- For example, to obtain the username of the current account, enter the following command at the command prompt:

```
SELECT CURRENT_USER();
```

CURRENT_USER Function

Figure displays the output of the command



```
mysql> SELECT CURRENT_USER();
+----------------+
| CURRENT_USER() |
+----------------+
| root@localhost |
+----------------+
1 row in set (0.00 sec)

mysql>
```

DATABASE Function

1-2

- The DATABASE function returns the name of the current database
- If no database is activated, then this function returns a NULL value
- The syntax to view the name of the current database is:

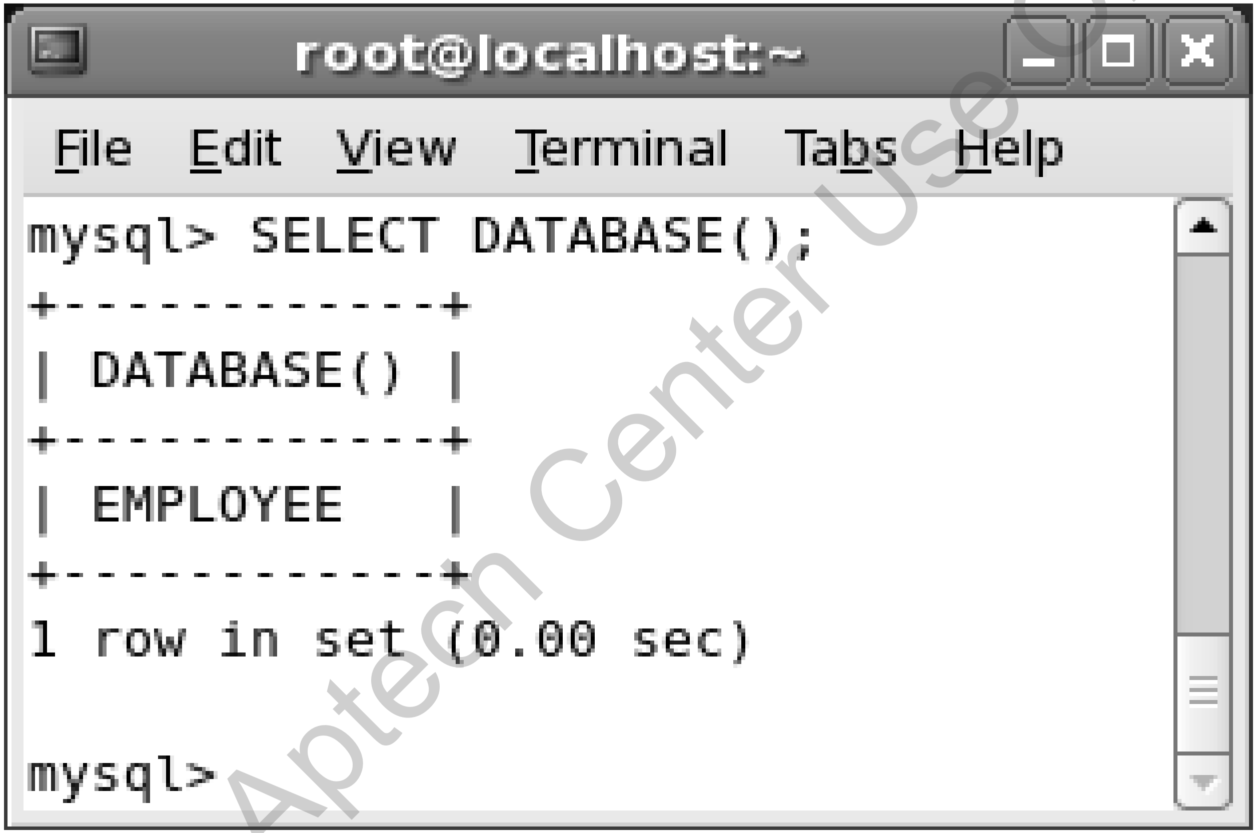
```
SELECT DATABASE();
```

- For example, to retrieve the name of the current database, enter the following command at the command prompt:

```
SELECT DATABASE();
```

DATABASE Function

Figure displays the output of the command



```
mysql> SELECT DATABASE();
+------------+
| DATABASE() |
+------------+
| EMPLOYEE   |
+------------+
1 row in set (0.00 sec)

mysql>
```

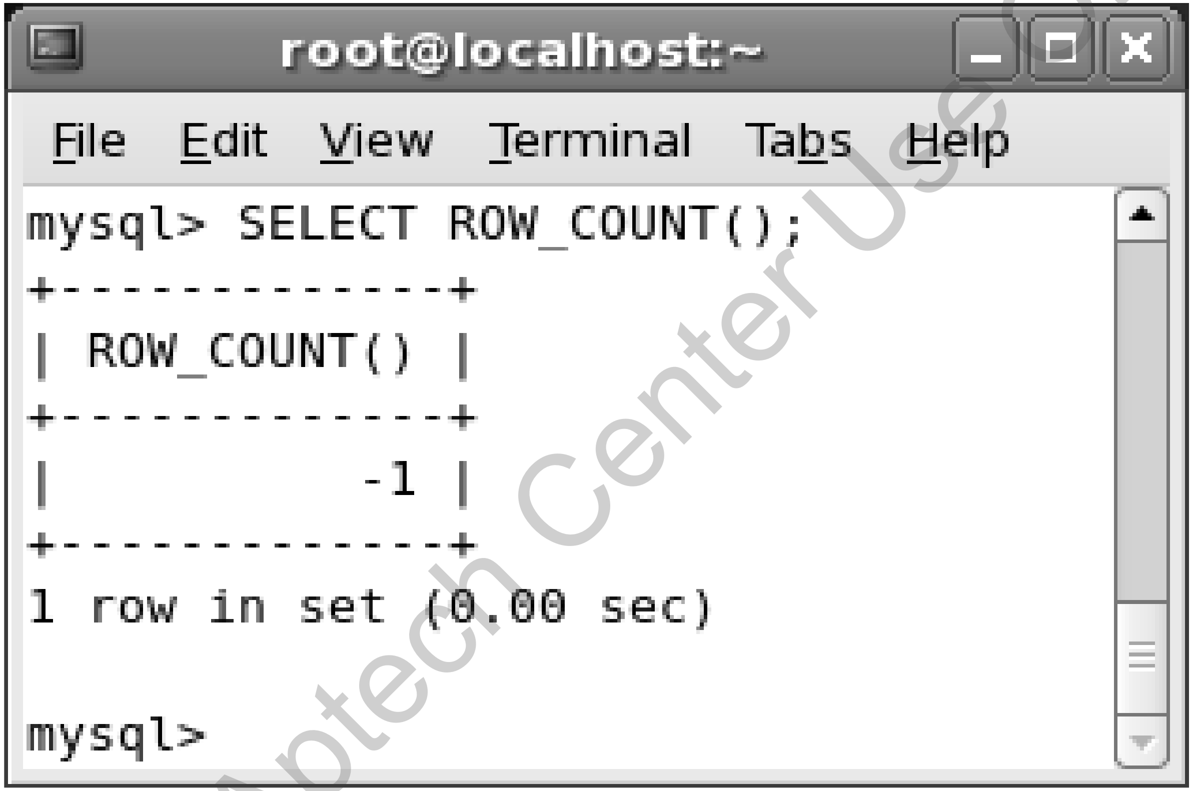
ROW_COUNT Function 1-2

- The ROW_COUNT function returns the number of rows inserted, updated, or deleted after the INSERT, UPDATE, or DELETE statements are executed
- The number of affected-rows returned in an UPDATE statement is the number of rows actually changed
- In REPLACE statement, the number of affected rows is two if the new row replaces an old row, because, the REPLACE statement inserts one row after deleting the duplicate row
- In case of an INSERT statement, the number of rows affected is one if a new row is inserted and the number of rows affected is two if an existing row is updated
- The syntax for this function is:

```
SELECT ROW_COUNT();
```

ROW_COUNT Function

Figure displays the output of the command



```
mysql> SELECT ROW_COUNT();
+-------------+
| ROW_COUNT() |
+-------------+
|          -1 |
+-------------+
1 row in set (0.00 sec)

mysql>
```

VERSION Function

- The `VERSION` function returns the version of MySQL
- The syntax to view the version of MySQL is:

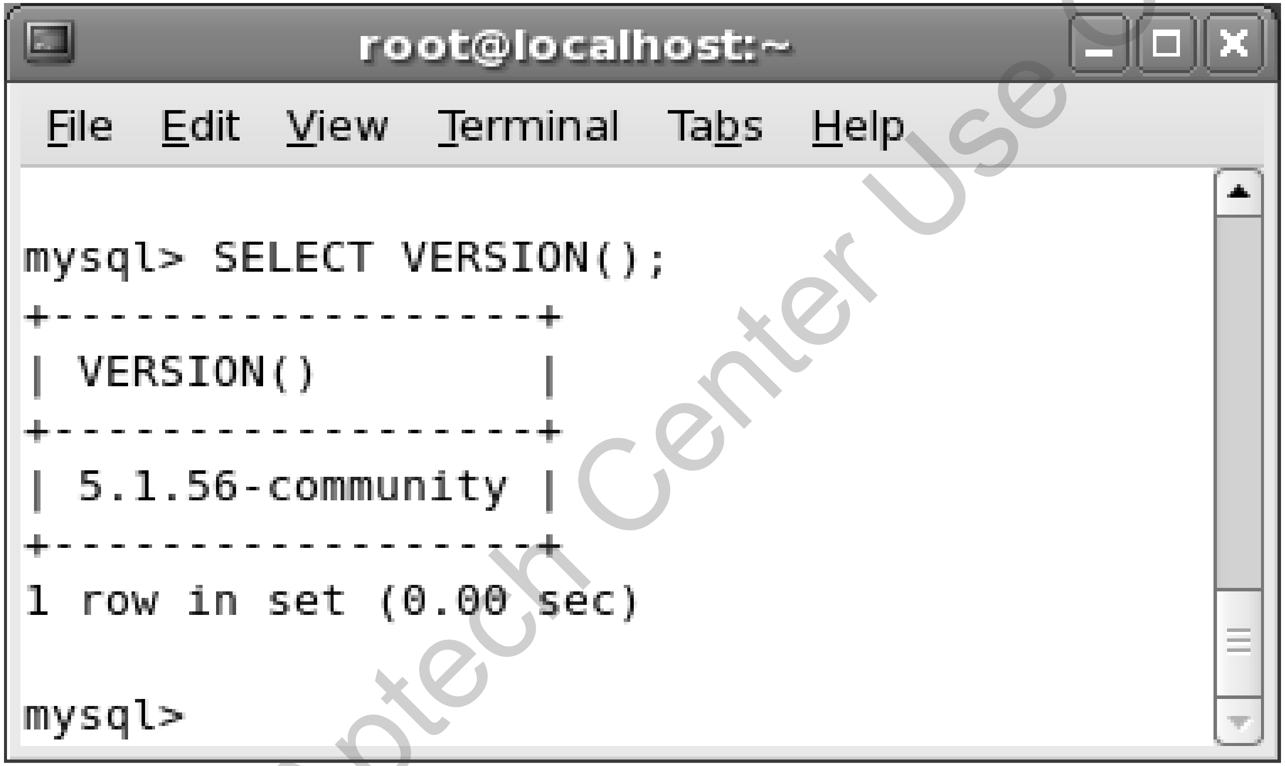
```
SELECT VERSION();
```

- For example, to view the version of MySQL, enter the following command at the command prompt:

```
SELECT VERSION();
```

VERSION Function 2-4

Figure displays the output of the command



```
mysql> SELECT VERSION();
+-------------------+
| VERSION()         |
+-------------------+
| 5.1.56-community |
+-------------------+
1 row in set (0.00 sec)

mysql>
```

VERSION Function

Table displays the description and use of other system information functions

Name	Description	Example
CONNECTION_ID	The CONNECTION_ID function returns the thread ID for the connection. MySQL assigns a unique ID for every thread or connection. The syntax to retrieve the thread ID is: SELECT CONNECTION_ID();	For example, to obtain the thread ID for the current connection, enter the following command at the command prompt: SELECT CONNECTION_ID(); The output of this function is: 2
ROW_COUNT	The ROW_COUNT function returns the number of rows inserted, updated, or deleted after the INSERT, UPDATE, or DELETE statements are executed. The syntax for this function is: SELECT ROW_COUNT();	Execute the following command and view the output SELECT ROW_COUNT; The output of this function is: -1

VERSION Function

Name	Description	Example

USER	The USER function returns the current username and hostname. The syntax to view the username is:
	`SELECT USER();`
		For example, to obtain the current username and hostname, enter the following command at the command prompt:
		`SELECT USER();`
		The output of this function is:
		`root@localhost.com`

Summary

- In SQL queries, functions can be used in place of the column name and in the WHERE clause
- Date functions can execute computations on date, time, and datetime data types. These functions can add and subtract the date, extract a part of the date, and convert a date from one format to another. Some examples of date functions are ADDDATE, YEAR, DATEDIFF, and CONVERT_TZ
- String functions operate on string data type. They can be used to retrieve the ASCII value of a string, calculate the number of bits in a string, join two or more strings with or without the separator, convert the given string to lowercase or uppercase, and calculate the hexadecimal value of string. Some examples of STRING functions are ASCII, CHARACTER_LENGTH, BIT_LENGTH, LCASE, and UCASE

Summary 2-2

- System information functions display system-related information. The version number of the database, name of the current user and host, and number of rows affected due to the preceding `INSERT`, `UPDATE`, or `DELETE` statement can be retrieved using system information functions. Some examples of system information functions are `BENCHMARK`, `CHARSET`, `USER`, and `VERSION`