

# **DATABASE**

## **SQL (continues)**

## SOME MORE USEFUL COMMANDS

- At first, we will learn the commands to do the followings-
  - How to load data from a file?
  - How to save result in files?
  - How to run a script?
  - Some math and date functions

# HOW TO LOAD DATA FROM A FILE?

This can be done using **infile** option

Let's consider the following data.txt file

```
Kiran, gandhi rd, delhi  
John, park st, delhi  
Ena, 24th cross road, delhi
```

Now we want to load these values into customer table

Customer_name	Customer_street	Customer_city
Alice	DU street	Delhi
Bob	Park road	Delhi

```
mysql> LOAD DATA LOCAL INFILE 'data.txt' INTO TABLE  
customer FIELDS TERMINATED BY ',' LINES TERMINATED  
BY '\n'
```

# HOW TO SAVE QUERY RESULT IN A FILE?

This can be done using **outfile** option

Let's consider the following customer table

Customer_name	Customer_street	Customer_city
Alice	DU road	Delhi
Bob	Park road	Delhi

Now we want to save the rows into a text file name data.txt

```
mysql> SELECT *  
      INTO OUTFILE 'data.txt'  
      FIELDS TERMINATED BY ',' LINES TERMINATED BY '\n'  
      FROM customer
```

If you don't have the permission to write to your current folder then use the following command to check which folder you may use

```
mysql> SELECT @@GLOBAL.secure_file_priv;
```

## TO RUN MULTIPLE COMMANDS

- Source command can be used within mysql

Let's consider the following commands  
saved in a file `query1.sql`

```
SELECT * FROM emp;  
SELECT * FROM customer;
```

To execute them from a file use the following

```
mysql> source query1.sql
```

# SOME MATH FUNCTIONS

- ABS(n): returns the absolute value of a number

- Example: select abs(-1.72)



1.72

- CEIL(n): returns the smallest integer value not less than n

- Example: select ceil(2.73)



3

- FLOOR(n): returns the largest integer value not greater than n

- Example: select floor(2.73)



2

- CONV(n, from\_base, to\_base): converts a number from one base to another

- Example: select conv(1111,2,10)



15

- DIV operator is used to perform integer division

- Example: select 102 div 5



20

- '/' operator is also used to perform division

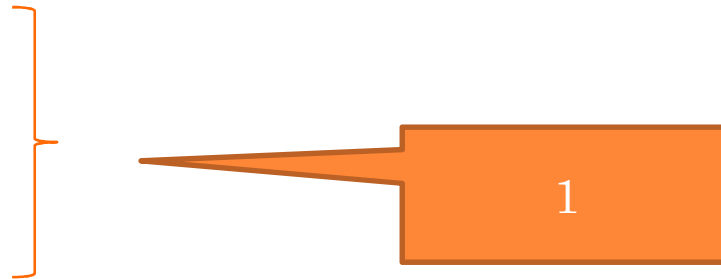
- Example: select 102 / 5



20.4000

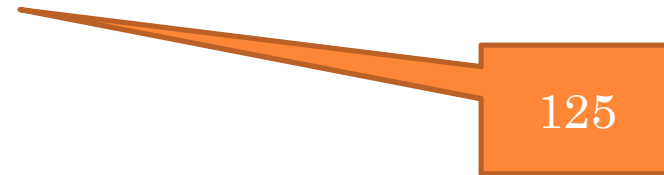


- MOD(): returns the remainder of a number divided by another number
- MOD(n,m) or  $n \% m$  or  $n \text{ MOD } m$ 
  - Example:
  - select mod(10,3)
  - select 10%3
  - select 10 mod 3



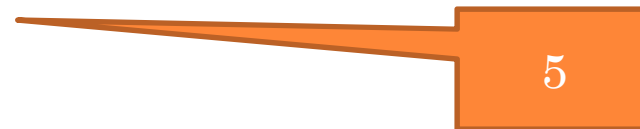
- POW(): returns the value of a number raised to the power of another number

- Example: `select pow(5,3)`



- SQRT(): returns the square root of a non-negative number

- Example: `select sqrt(25)`



- ROUND(): rounds a number specified as an argument up to a number specified as another argument
- ROUND(n,[d]), here n is the number which will be rounded upto d decimal places
  - select round(17.78)
  - select round(17.78,1)



18

A diagram consisting of an orange arrow pointing from the text 'select round(17.78)' to an orange rectangular box containing the number '18'.



17.8

A diagram consisting of an orange arrow pointing from the text 'select round(17.78,1)' to an orange rectangular box containing the number '17.8'.

- RAND(): returns a random floating point value between the range 0 and 1
  - Example: select rand(), rand();

Two random number are generated

- RAND(seed): returns a repeatable random floating point value between the range 0 and 1
  - Example: select rand(2), rand(2);

Same random number  
generated twice

# SOME DATE FUNCTIONS

## 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
- CURRENT\_DATE and CURRENT\_DATE functions are same as CURDATE()
  - `mysql> SELECT curdate();`
  - `mysql> SELECT current_date();`
  - `mysql> SELECT current_date;`

## SYSDATE()

- SYSDATE() returns the current date and time in YYYY-MM-DD HH:MM:SS or YYYYMMDDHHMMSS.uuuuuu format depending on the context of the function.
  - `mysql> SELECT sysdate();`

## EXTRACT()

- EXTRACTs a part of a given date. This function does not perform date arithmetic. The unit specifiers of DATE\_ADD() and DATE\_SUB() work with this function also.
  - Syntax: `extract(unit from date1)`
  - `mysql> SELECT EXTRACT(year from '2018-09-24 20:34:45')`
  - Like year, one can extract month, day, hour, minute, seconds, etc





## ADDDATE()

- MySQL ADDDATE() adds a time value with a date.
- The DATE\_ADD() is the synonym of ADDDATE()
  - Syntax: ADDDATE(date, INTERVAL expr unit),  
ADDDATE(expr,days)
  - `mysql>SELECT ADDDATE('2018-05-15', INTERVAL 10 DAY) as required_date;`

## ADDTIME()

- In MySQL the ADDTIME() returns a time or datetime after adding a time value with a time or datetime.
  - Syntax: ADDTIME(expr1,expr2)
  - `mysql>SELECT ADDTIME('2018-05-15 13:20:32.50','2 1:39:27.50') as required_datetime;`

# DATE\_FORMAT

- DATE\_FORMAT(date, format): it formats the date value according to the format string
- In the *format* string, specifier character is used along with the %symbol
- Example:
  - `mysql> SELECT DATE_FORMAT ( '1998-10-18', '%D %b %Y' )`  
  
18<sup>th</sup> Oct 1998
  - `mysql> SELECT DATE_FORMAT ( '1998-10-18', '%d %c %y' )`  
  
18 10 98

## SPECIFIES TABLE

Specifier	Description
%a	Abbr. weekday name (like Sun..Sat)
%b	Abbr. month name (like Jan..Dec)
%c	Month numeric (0..12)
%D	Day of the month with English suffix (0 <sup>th</sup> , 1 <sup>st</sup> , 2 <sup>nd</sup> , 3 <sup>rd</sup> , ...)
%d	Day of the month, numeric (00..31)
%M	Month name (January,..., December)
%m	Month, numeric (00,...,12)
%Y	Year numeric (4 digits)
%H	Hour (00..23)
%h	Hour (01..12)
%i	Minutes, numeric (0..59)
%s	Seconds (00,..59)
%p	AM or PM

## EXAMPLE OF SOME FORMAT STRINGS

date_format String	example
'%a %D %b %Y'	Mon 24th Sep 2018
'%a %D %b %Y %H:%i'	Mon 24th Sep 2018 12:30
%a %D %b %Y %T'	Mon 24th Sep 2018 12:30:10
%a %b %e %Y'	Mon Sep 24 2018
'%W %D %M %Y'	Monday 24th September 2018
'%M %e, %Y'	September 24, 2018

## DATE\_SUB()

- MySQL DATE\_SUB() function subtract a time value (as interval) from a date.
  - Syntax: DATE\_SUB(date, INTERVAL expr unit)
  - `mysql> SELECT DATE_SUB('2019-08-29', INTERVAL 10 DAY);`

## DATEDIFF()

- DATEDIFF() returns the number of days between two dates or datetimes. This function only calculates the date portion from each expression.
  - Syntax DATEDIFF(expr1,expr2);
  - `mysql> SELECT DATEDIFF('2019-08-29 11:31:31','2019-08-15');`

## DAYNAME()

- DAYNAME() returns the name of the week day of a date specified in the argument.
  - Syntax: DAYNAME(date1)
  - `mysql> SELECT DAYNAME('2019-09-15');`



## DAYOFWEEK

- DAYOFWEEK() returns the week day number (1 for Sunday, 2 for Monday ..... 7 for Saturday ) for a date specified as argument.
  - Syntax: DAYOFWEEK(date)
  - `mysql>SELECT DAYOFWEEK('2018-09-15');`

## LAST\_DAY()

- LAST\_DAY() returns the last day of the corresponding month for a date or datetime value. If the date or datetime value is invalid, the function returns NULL.
  - Syntax: LAST\_DAY(date1)
  - `mysql> SELECT LAST_DAY('2019-08-18');`

## DAYOFYEAR

- MySQL DAYOFYEAR() returns day of the year for a date. The return value is within the range of 1 to 366.
  - Syntax: DAYOFYEAR(date1)
  - `mysql> SELECT DAYOFYEAR('2019-08-15');`

## TO\_DAYS()

- MySQL TO\_DAYS() returns a number of days between a given date and year 0
  - Syntax: TO\_DAYS(date);
  - `mysql> SELECT TO_DAYS('2019-08-15');`