MySQL

MySQL Basic To Advance

A Practical Programming Guide

Sanket Mishrikotkar

MySQL
Basic To Advance

A Practical Programming Guide

HELLO,
I HAVE COMPILED THE DATA FROM VARIOUS RESOURCES AND WRITTEN THE MYSQL E-BOOK IN VERY SIMPLE & UNDERSTANDING LANGUAGE FOR THE BEGINNER.

Sanket Mishrikotkar

CONTACT:

sanketmishrikotkar

SanketMishrikotkar

msanket17

mishrikotkarsanket@gmail.com

Note: Links are attached to logo & text. Click on that & you will get my profile.

1. MySQL - Home:

SQL is a most popular proper and standard language used for storing, manipulating and retrieving data in databases. MySQL is the open source relational SQL database management system which is very popular and user friendly. MySQL RDBMS is used for developing various applications on web-based.

Database:

A database is an application where the well designed and well organised collection of records, data are stored. We can perform many operations on the database like creating, accessing, managing, searching, and replication of the data the database stores with the distinct API.

2. MySQL - Introduction:

MYSQL is an open source database software and it is supported by Oracle Company. MYSQL is a database management system which supports the relational database management system. It is very fast, smooth, reliable, scalable, and very simple & easy to use. MySQL was developed by Michael Widenius and David Axmark in 1994. It is developed by MySQL AB and MySQL is written in C, C++ programming languages. MySQL supports many operating systems like Windows, Linux, MacOS, etc. MySQL is available in English Language.

A MySQL is a Relational Database Management System (RDBMS) software provides facility of:

- It gives the platform to implement the database operation on tables, rows, columns, index, etc.
- This maintains the relationship of database in the form of tables, are known as relations.
- Table is collection of rows and columns.
- This confirms and shows the Referential Integrity between rows and tables.
- This interprets many SQL queries and combines meaningful information from many tables.
- It provides the facility to update the table.
- SQL stands for Structure Query Language.

Terms related to the RDBMS:

- **Database** − A database is a collection of records, tables, other data and metadata
- **Table** − A table is a collection of rows and columns which is represented in matrix form.
- **Column** − A columns is a vertical dataset in table.
- **Row** – A row is a horizontal dataset in table. It can be called tuple or entry or record.
- **Redundancy** – A redundancy is storing the same data at two different places.
- **Primary Key** – A primary is a unique property where the key value cannot be entered in a table twice.
- **Foreign Key** − A foreign key links the pin between two tables.
- **Compound Key** − A compound key (composite key) is a key that have multiple columns, because one column is not sufficiently unique.
- **Index** − An **index** is a copy of selected columns of data from a **table**
- **Referential Integrity** – It refers to the accuracy and consistency of data within a relationship.

Advantages of MySQL:

- It is an open source.
- It is standard form of well-organized and known SQL.
- It works on multiple operating system.
- It is very friendly with PHP.
- It is having large storage capacity.
- It is custumizable as per the user.

3. MySQL - Installation

Steps to install MySQL:

Step 1: download MySQL

Download MySQL from dev.mysql.com/downloads/.

Step 2: extract the files

Download and extract the folder. Then rename the folder as “mysql”. Install MySQL to C drive. You can install anywhere in any drive but prefer C drive for installation.

Step 3: move the data folder (optional)

I recommend placing the data folder on another drive or partition to make backups and re-installation easier. For the purposes of this example, we will create a folder called D:MySQLdata and move the contents of C:mysqldata into it.
You should now have two folders, D:MySQLdatamysql and D:MySQLdatatest. The original C:mysqldata folder can be removed.

Step 4: create a configuration file

MySQL provides several configuration methods but, in general, it is easiest to to create a my.ini file in the mysql folder. There are hundreds of options to tweak MySQL to your exact requirements, but the simplest my.ini file is:

```
[mysqld]
# installation directory
basedir="C:/mysql/"

# data directory
datadir="D:/MySQLdata/"
```

(Remember to change these folder locations if you have installed MySQL or the data folder elsewhere.)

Step 5: test your installation

Open a command box (Start > Run > cmd) and enter the following commands:

```
cd mysqlbin
mysqld
```

This will start the MySQL server which listens for requests on localhost port 3306. You can now start the MySQL command line tool and connect to the database. Open another command box and enter:

```
cd mysqlbin
mysql -u root
```

This will show a welcome message and the mysql> prompt. Enter “show databases;” to view a list of the pre-defined databases.

Step 6: change the root password

The MySQL root user is an all-powerful account that can create and destroy databases. If you are on a shared network, it is advisable to change the default (blank) password. From the mysql> prompt, enter:

```
UPDATE mysql.user SET password=PASSWORD("my-new-password") WHERE
User='root';
FLUSH PRIVILEGES;
```

You will be prompted for the password the next time you start the MySQL command line.

Enter “exit” at the mysql> prompt to stop the command line client. You should now shut down

MySQL with the following command:

```
mysqladmin.exe -u root shutdown
```

Step 7: Install MySQL as a Windows service

The easiest way to start MySQL is to add it as a Windows service. From a command prompt, enter:

```
cd mysqlbin
mysqld –install
```

Open the Control Panel, Administrative Tools, then Services and double-click MySQL. Set the Startup type to “Automatic” to ensure MySQL starts every time you boot your PC.

Alternatively, set the Startup type to “Manual” and launch MySQL whenever you choose using the command “net start mysql”.

4. MySQL - Administration

Administration of MySQL consists the total management part of database and tables. Admin of the database can access to each and every table.

Administrative MySQL Command:

- **USE Databasename** –This command will give give order to use the current database for work.
- **SHOW DATABASES** – This command will show the list of databases.
- **SHOW TABLES** – This command shows the tables in the database.
- **SHOW COLUMNS FROM *tablename:*** Shows the attributes, types of attributes, key information, whether NULL is permitted, defaults, and other information for a table.
- **SHOW INDEX FROM tablename** − Presents the details of all indexes on the table, including the PRIMARY KEY.
- **SHOW TABLE STATUS LIKE tablename\G** − Reports details of the MySQL DBMS performance and statistics.

5. MySQL – PHP Syntax

Out of these languages, PHP is the most popular one because of its web application development capabilities.
You would require to call the PHP functions in the same way you call any other PHP function.

Syntax for PHP Query is:
mysql_function(value,value,...);

The following example shows a generic syntax of PHP to call any MySQL function.

```
<html>
  <head>
    <title>PHP with MySQL</title>
  </head>
  <body>
    <?php
      $retval = mysql_function(value, [value,...]);
      if( !$retval ) {
        die ( "Error: a related error message" );
      }
    ?>
  </body>
</html>
```

6. MySQL - Connection

PHP provides **mysql_connect()** function to open a database connection. This function takes five parameters and returns a MySQL link identifier on success or FALSE on failure.

Syntax:

connection mysql_connect(server,user,passwd,new_link,client_flag);

Sr.No.	Parameter & Description
1	**server** Optional − The host name running the database server. If not specified, then the default value will be **localhost:3306**.
2	**user** Optional − The username accessing the database. If not specified, then the default will be the name of the user that owns the server process.
3	**passwd** Optional − The password of the user accessing the database. If not specified, then the default will be an empty password.
4	**new_link** Optional − If a second call is made to mysql_connect() with the same arguments, no new connection will be established; instead, the identifier of the already opened connection will be returned.
5	**client_flags** Optional − A combination of the following constants − • MYSQL_CLIENT_SSL − Use SSL encryption. • MYSQL_CLIENT_COMPRESS − Use compression protocol. • MYSQL_CLIENT_IGNORE_SPACE − Allow space after function names. • MYSQL_CLIENT_INTERACTIVE − Allow interactive timeout seconds of inactivity before closing the connection.

You can disconnect from the MySQL database anytime using another PHP function **mysql_close()**. This function takes a single parameter, which is a connection returned by the **mysql_connect()** function.

Syntax:

bool mysql_close (resource $link_identifier);

If a resource is not specified, then the last opened database is closed. This function returns true if it closes the connection successfully otherwise it returns false.

Example:

Try the following example to connect to a MySQL server –

```
<html>
  <head>
    <title>Connecting MySQL Server</title>
  </head>
  <body>
    <?php
      $dbhost = 'localhost:3306';
      $dbuser = 'guest';
      $dbpass = 'guest123';
      $conn = mysql_connect($dbhost, $dbuser, $dbpass);

      if(! $conn ) {
        die('Could not connect: ' . mysql_error());
      }
      echo 'Connected successfully';
      mysql_close($conn);
    ?>
  </body>
</html>
```

7. MySQL - Create Database:

In MySQL, we create the database to save the data, information, records etc. To create the database, CREATE DATABASE command is used.

Syntax:

```
CREATE DATABASE databasename;
```

Example:

```
create database Info;
```

8. MySQL - Drop Database:

Here drop means delete database. To drop the existing database, DROP DATABASE command is used.

Syntax:

```
DROP DATABASE databasename;
```

Example:

```
drop database Info;
```

9. MySQL - Select Database:

Selecting database means here we select or choose the database to use. To use the database we have created, USE DATABASE command is used.

Syntax:

```
USE databasename;
```

Example:

```
Use Info;
```

10. MySQL - Data Types

Numeric Data Type:

Data Type Syntax	Description
INT	A normal-sized integer that can be signed or unsigned. Range If Signed: -2147483648 to 2147483647 If unsigned: 0 to 4294967295 You can specify a width of up to 11 digits.
TINYINT	A very small integer that can be signed or unsigned. Range If Signed: -128 to 127 If unsigned: 0 to 255 You can specify a width of up to 4 digits.
SMALLINT	A small integer that can be signed or unsigned. Range If Signed: -32768 to 32767 If unsigned: 0 to 65535 You can specify a width of up to 5 digits.
MEDIUMINT	A medium-sized integer that can be signed or unsigned. Range If Signed: -8388608 to 8388607 If unsigned: 0 to 16777215 You can specify a width of up to 9 digits.
BIGINT	A large integer that can be signed or unsigned. Range If Signed: -9223372036854775808 to 9223372036854775807 If unsigned: 0 to 18446744073709551615 You can specify a width of up to 20 digits.
FLOAT(m,d)	A floating-point number that cannot be unsigned. You can define the display length (m) and the number of decimals (d). Decimal precision can go to 24 places for a float.
DOUBLE(m,d)	A double precision floating-point number that cannot be unsigned. You can define the display length (m) and the number of decimals (d). Decimal precision can go to 53 places for a double. Real is a synonym for double.
DECIMAL(m,d)	An unpacked floating-point number that cannot be unsigned. In unpacked decimals, each decimal corresponds to one byte. Defining the display length (m) and the number of decimals (d) is required. Numeric is a synonym for decimal.

Date and Time Data Type:

Data Type Syntax	Maximum Size	Explanation
DATE	Values range from '1000-01-01' to '9999-12-31'.	Displayed as 'yyyy-mm-dd'.
DATETIME	Values range from '1000-01-01 00:00:00' to '9999-12-31 23:59:59'.	Displayed as 'yyyy-mm-dd hh:mm:ss'.
TIMESTAMP(m)	Values range from '1970-01-01 00:00:01' UTC to '2038-01-19 03:14:07' TC.	Displayed as 'YYYY-MM-DD HH:MM:SS'.
TIME	Values range from '-838:59:59' to '838:59:59'.	Displayed as 'HH:MM:SS'.
YEAR[(2|4)]	Year value as 2 digits or 4 digits.	Default is 4 digits.

String Data Types:

Data Type Syntax	Maximum Size	Explanation
CHAR(size)	Maximum size of 255 characters.	Where size is the number of characters to store. Fixed-length strings. Space padded on right to equal size characters.
VARCHAR(size)	Maximum size of 255 characters.	Where size is the number of characters to store. Variable-length string.
TINYTEXT(size)	Maximum size of 255 characters.	Where size is the number of characters to store.
TEXT(size)	Maximum size of 65,535 characters.	Where size is the number of characters to store.
MEDIUMTEXT(size)	Maximum size of 16,777,215 characters.	Where size is the number of characters to store.
LONGTEXT(size)	Maximum size of 4GB or 4,294,967,295 characters.	Where size is the number of characters to store.
BINARY(size)	Maximum size of 255 characters.	Where size is the number of binary characters to store. Fixed-length strings. Space padded on right to equal size characters.
VARBINARY(size)	Maximum size of 255 characters.	Where size is the number of characters to store. Variable-length string.

Binary Large Object Data Types (BLOB):

Data Type Syntax	Maximum Size
TINYBLOB	Maximum size of 255 bytes.
BLOB(size)	Maximum size of 65,535 bytes.
MEDIUMBLOB	Maximum size of 16,777,215 bytes.
LONGTEXT	Maximum size of 4gb or 4,294,967,295 characters

11. MySQL - Create Table:

The MySQL CREATE TABLE command is used to create a new table into the database. Creation of table command requires three things:

- Name of the table
- Names of fields
- Definitions for each field

Syntax:

```
CREATE TABLE table_name (
   column1 datatype,
   column2 datatype,
   column3 datatype,
  ....
);
```

Example:

```
create table Student(Rno int, Name varchar(50), Address varchar(60));
```

Output:

```
Table created.

0.47 seconds
```

Create Table Using Another Table

A copy of an existing table can also be created using CREATE TABLE.

Syntax:

```
CREATE TABLE new_table_name AS
   SELECT column1, column2,...
   FROM existing_table_name
   WHERE ....;
```

Example:

```
create table info as select Rno, Name from Student;
```

Output:

```
Table created.

0.01 seconds
```

12. MySQL - Drop Table

The DROP TABLE statement is used to drop (delete) an existing table in a database.
Syntax:

```
DROP TABLE table_name;
```

Example:

```
drop table info;
```

Output:

```
Table dropped.

0.56 seconds
```

13. MySQL - Insert Query:

The INSERT query is used to insert the new record in the table.

Syntax:

```
INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);
```

Example:

```
insert into Student(Rno, Name, Address) values(1, 'Sanket', 'Aurangabad');
```

Output:

```
1 row(s) inserted.

0.01 seconds
```

Syntax for all fields:

```
INSERT INTO table_name
VALUES (value1, value2, value3, ...);
```

Example:

```
insert into Student values(2, 'John', 'Aurangabad');
```

Output:

```
1 row(s) inserted.

0.01 seconds
```

14.MySQL - Select Query:

The SELECT query is used to select the data from the database. This query gives the output for the data which we have selected.

Syntax:
To display whole table
SELECT * FROM table_name;

Example:
Select * from Student;

Output:

RNO	NAME	ADDRESS
1	Sanket	Aurangabad
2	John	Aurangabad

2 rows returned in 0.00 seconds

For perticulaar column:
SELECT column1, column2, ...
FROM table_name;

Example:
Select Rno, Name from Student;

Output:

RNO	NAME
1	Sanket
2	John

2 rows returned in 0.00 seconds

15. MySQL - Where Clause:

The WHERE clause is used to filter records from the table.
The WHERE clause is used to fetch only those records that fulfill a specified condition given in a query.

Syntax:

```
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

Example:

```
Select Name, Address from Student where Rno=1;
```

Output:

NAME	ADDRESS
Sanket	Aurangabad

1 rows returned in 0.00
seconds

16. MySQL - Update Query:

The UPDATE query is used to modify or to change the values in the existing record.

Syntax:

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

Example:

```
Update Student SET Address='Pune' where Rno=2;
```

Output:

```
1 row(s) updated.

0.01 seconds
```

RNO	NAME	ADDRESS
1	Sanket	Aurangabad
2	John	Pune
12	Rohit	Jalna

```
3 rows returned in 0.00
seconds
```

17. MySQL - Delete Query:

The delete statement is used to delete the existing record in the table.

Syntax:

```
DELETE FROM table_name WHERE condition;
```

Example:

```
Delete from Student where Rno=12;
```

Output:

```
1 row(s) deleted.

0.00 seconds
```

RNO	NAME	ADDRESS
1	Sanket	Aurangabad
2	John	Pune

2 rows returned in 0.00 seconds

To delete all records:

```
DELETE FROM table_name;
```

Example:

```
Delete from Student;
```

Output:

```
2 row(s) deleted.

0.00 seconds

no data found
```

18. MySQL - Like Clause:

The LIKE operator is used in a WHERE clause to search for a specified pattern in a column.

Syntax:

```
SELECT column1, column2, ...
FROM table_name
WHERE columnN LIKE pattern;
```

There are two wildcards often used in conjunction with the LIKE operator:

% - The percent sign represents zero, one, or multiple characters

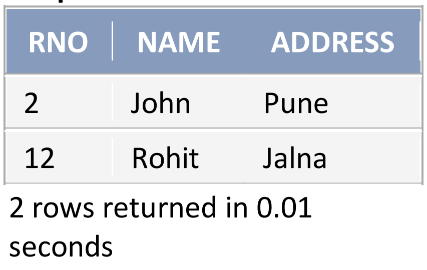
_ - The underscore represents a single character

LIKE Operator	Description
WHERE .... LIKE 'a%'	Finds any values that start with "a"
WHERE .... LIKE '%a'	Finds any values that end with "a"
WHERE .... LIKE '%or%'	Finds any values that have "or" in any position
WHERE .... LIKE '_r%'	Finds any values that have "r" in the second position
WHERE .... LIKE 'a_%'	Finds any values that start with "a" and are at least 2 characters in length
WHERE .... LIKE 'a__%'	Finds any values that start with "a" and are at least 3 characters in length
WHERE .... LIKE 'a%o'	Finds any values that start with "a" and ends with "o"

Example:

```
select * from Student where Rno LIKE '%2';
```

Output:



RNO	NAME	ADDRESS
2	John	Pune
12	Rohit	Jalna

2 rows returned in 0.01 seconds

19. MySQL - Sorting Results:

Sorting results in MySQL is one of the important part in database. To sort the data in ascending or descending order, the ORDER BY keyword is used. This sorts the records in the ascending order by default. To sort the records in descending order, use the DESC keyword.

Syntax:

```
SELECT column1, column2, ...
FROM table_name
ORDER BY column1, column2, ... ASC|DESC;
```

Example:

```
select * from Student ORDER BY Rno DESC;
```

Output:

RNO	NAME	ADDRESS
12	Rohit	Jalna
2	John	Pune
1	Sanket	Aurangabad

3 rows returned in 0.01 seconds

20. MySQL - Using Join:

A JOIN clause is used to combine rows from two or more tables, based on a related column between them. MySQL JOINS are used with SELECT statement. It is used to retrieve data from multiple tables. It is performed whenever you need to fetch records from two or more tables.

Following are the types of MySQL joins:

- MySQL INNER JOIN (or sometimes called simple join)
- MySQL LEFT OUTER JOIN (or sometimes called LEFT JOIN)
- MySQL RIGHT OUTER JOIN (or sometimes called RIGHT JOIN)
- FULL OUTER JOIN
- Self – Join
- Cartesian Product or Cross Join

Example:

```
create table Student(Rno int, Name varchar(50));

insert into Student values(1, 'Sanket');
insert into Student values(2, 'Tejas');
insert into Student values(4, 'Rohit');

create table Info(Rno int, Address varchar(50));

Insert into Info values(1, 'Aurangabad');
Insert into Info values(2, 'Kannd');
Insert into Info values(3, 'Jalna');

select * from Student;

Select * from Info;
```

Output:

RNO	NAME
1	Sanket
2	Tejas
4	Rohit
3	Rushi

4 rows returned in 0.00 seconds

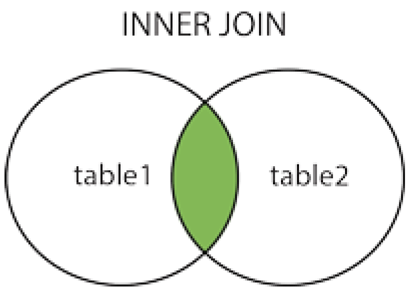
RNO	ADDRESS
1	Aurangabad
2	Kannd
3	Jalna

3 rows returned in 0.00 seconds

MySQL INNER JOIN:

The INNER JOIN keyword extracts the records that have same or exact values in both tables. It returns all the rows from both tables where condition is satisfied. The INNER JOIN is also called as SIMPLE INNER JOIN. It is the most common type of join.

Syntax:
SELECT column_name(s)
FROM table1
INNER JOIN table2
ON table1.column_name = table2.column_name;



Example:
select * from Student INNER JOIN Info on Student.Rno = Info.Rno;

Output:

RNO	NAME	RNO	ADDRESS
1	Sanket	1	Aurangabad
2	Tejas	2	Kannd
3	Rushi	3	Jalna

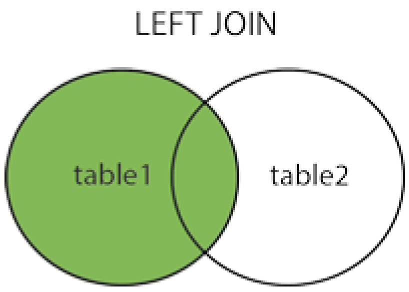
3 rows returned in 0.02 seconds

MySQL LEFT OUTER JOIN:

The LEFT OUTER JOIN returns all the rows of the left hand side table and only the rows from right hand side table where the condition is satisfied. If the condition is not satisfied, then the NULL is the result. It is also called as LEFT JOIN.

Syntax:

```
SELECT columns
FROM table1
LEFT [OUTER] JOIN table2
ON table1.column = table2.column;
```



Example:

```
Select * FROM Student LEFT OUTER JOIN Info ON (Student.Rno = Info.Rno);
```

Output:

RNO	NAME	RNO	ADDRESS
1	Sanket	1	Aurangabad
2	Tejas	2	Kannd
3	Rushi	3	Jalna
4	Rohit	-	-

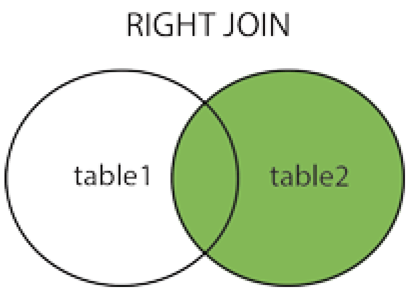
4 rows returned in 0.01 seconds

MySQL RIGHT OUTER JOIN:

The RIGHT OUTER JOIN returns all the rows of the right hand side table and only the rows from left hand side table where the condition is satisfied. If the condition is not satisfied, then the NULL is the result. It is also called as RIGHT JOIN.

Syntax:

```
SELECT columns
FROM table1
RIGHT [OUTER] JOIN table2
ON table1.column = table2.column;
```



Example:
Select * FROM Student RIGHT OUTER JOIN Info ON (Student.Rno = Info.Rno);

Output:

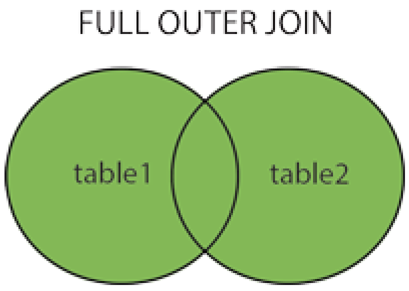
RNO	NAME	RNO	ADDRESS
1	Sanket	1	Aurangabad
2	Tejas	2	Kannd
3	Rushi	3	Jalna

3 rows returned in 0.00 seconds

FULL OUTER JOIN:

The FULL OUTER JOIN keyword returns all the records where the condition is satisfied in left or right table. FULL OUTER JOIN is also called as FULL JOIN.

Syntax:
SELECT column_name(s)
FROM table1
FULL OUTER JOIN table2
ON table1.column_name = table2.column_name
WHERE condition;



Example:
Select * FROM Student FULL OUTER JOIN Info ON (Student.Rno = Info.Rno);

Output:

RNO	NAME	RNO	ADDRESS
1	Sanket	1	Aurangabad
2	Tejas	2	Kannd
4	Rohit	-	-
3	Rushi	3	Jalna

4 rows returned in 0.00
seconds

Self join:

The SELF JOIN is used to join a table to itself as if the table were two tables

Syntax:
SELECT column_name(s)
FROM table1 T1, table1 T2
WHERE condition;

Example:
SELECT a.Rno, b.Name FROM Student a, Student b WHERE a.Rno < b.Rno;

Output:

RNO	NAME
1	Tejas
1	Rushi
1	Rohit
2	Rushi
2	Rohit
3	Rohit

6 rows returned in 0.01
seconds

Cartesian product or Cross Join:

This type of JOIN returns the cartesian product of rows from the tables in Join. It will return a table which consists of records which combines each row from the first table with each row of the second table.

Syntax:

```
SELECT column-name-list
FROM
table-name1 CROSS JOIN table-name2;
```

Example:

```
SELECT * FROM Student CROSS JOIN Info;
```

Output:

RNO	NAME	RNO	ADDRESS
1	Sanket	1	Aurangabad
2	Tejas	1	Aurangabad
4	Rohit	1	Aurangabad
3	Rushi	1	Aurangabad
1	Sanket	2	Kannd
2	Tejas	2	Kannd
4	Rohit	2	Kannd
3	Rushi	2	Kannd
1	Sanket	3	Jalna
2	Tejas	3	Jalna
More than 10 rows available. Increase rows selector to view more rows.			

10 rows returned in 0.03 seconds

21. MySQL - NULL Values:

NULL means nothing. Zero is a value. NULL doesn't mean ZERO. A field with a NULL value is a field with no value. NULL condition is used to check if there is any NULL value is there in table or not. It is used with SELECT, INSERT, UPDATE and DELETE statement (query).
We can test NULL values with comparison operators, such as =, <, or <>.

Syntax for IS NULL:

a) expression IS NULL
b) SELECT column_names
   FROM table_name
   WHERE column_name IS NULL;

Example:
Select * from Student WHERE Name IS NULL;

Output:

RNO	NAME
5	-
6	-

Syntax for IS NOT NULL:

a) expression IS NOT NULL
b) SELECT column_names
   FROM table_name
   WHERE column_name IS NOT NULL;

Example:
Select * from Student WHERE Name IS NOT NULL;

Output:

RNO	NAME
1	Sanket
2	Tejas
4	Rohit
3	Rushi

4 rows returned in 0.01
seconds

22. MySQL - Regexps:

REGEXP is an operator in MySQL which is based on the pattern matching operation.
Properties:

- Powerful and flexible pattern matching
- RLIKE is the synonym.
- Supports metacharacters and control pattern matching.
- The backslash is used as an escape character. It's only considered in the pattern match if double backslashes have used.
- It is not case sensitive.

Following is the table of pattern, which can be used along with the REGEXP operator.

Pattern	What the pattern matches
^	Beginning of string
$	End of string
.	Any single character
[...]	Any character listed between the square brackets
[^...]	Any character not listed between the square brackets
p1|p2|p3	Alternation; matches any of the patterns p1, p2, or p3
*	Zero or more instances of preceding element
+	One or more instances of preceding element
{n}	n instances of preceding element
{m,n}	m through n instances of preceding element
[A-Z]	match any upper case letter.
[a-z]	match any lower case letter
[0-9]	match any digit from 0 through to 9.
[[:<:]]	matches the beginning of words.
[[:>:]]	matches the end of words.
[:class:]	matches a character class i.e. [:alpha:] to match letters, [:space:] to match white space, [:punct:] is match punctuations and [:upper:] for upper class letters.
p1|p2|p3	Alternation; matches any of the patterns p1, p2, or p3
{n}	n instances of preceding element
{m,n}	m through n instances of preceding element

Example:
SELECT Name FROM Student WHERE Name REGEXP '^R';

Output:

RNO	NAME
4	Rohit
3	Rushi

23. MySQL - Transactions:

A transaction is a logical unit of work which contains multiple SQL statements like sequential group of statements, queries, or operation like select, update, insert, delete, etc that can be committed or rolled back. If any operation fails due to some reason then we can say that the whole transaction is failed. If the transaction performs multiple operations into the database, then two possibilities are as following:

- Transaction is committed when the modifications are done successfully.
- Transaction is rolled back when the all modifications are undone.

In simple language, any transaction cannot be successful without completing each and every operation.

Properties of Transaction:
Transaction consist of 4 main properties. These properties are called as ACID properties are as following:

1. Atomicity
2. Consistency
3. Isolation
4. Durability

Atomicity: This property ensures that the all the operations or work should be done successfully. If at any point the transactions failed due to some reason then the operations done that point will be rolled back to the previous state of the transactions.
Consistency: This property ensures the database can change the state if the transaction is committed successfully.
Isolation: This property ensures that the transaction can operate independently by their own and they can be transparent to each other.
Durability: This property ensures that the result will be permanently stored in the database if the system crashes or fails.

In MySQL, the transactions starts with statement **BEGIN WORK** and end with either a **COMMIT** or a **ROLLBACK** statement.

COMMIT and ROLLBACK

Commit and Rollback are two keywords used for Transactions.
When a transaction is completed successfully, then the COMMIT command should get executed.

If any failure occurs, then a ROLLBACK command get executed.

You can control the behaviour of a transaction by setting session variable called **AUTOCOMMIT**. If AUTOCOMMIT is set to 1 (the default), then each SQL statement (within a transaction or not) is considered a complete transaction and committed by default when it finishes.
When AUTOCOMMIT is set to 0, by issuing the **SET AUTOCOMMIT = 0** command, the subsequent series of statements acts like a transaction and no activities are committed until an explicit COMMIT statement is issued.
You can execute these SQL commands in PHP by using the **mysql_query()** function.

Syntax of START TRANSACTION, COMMIT, and ROLLBACK:

```
START TRANSACTION
  transaction_characteristic [, transaction_characteristic] ...]

transaction_characteristic:
   WITH CONSISTENT SNAPSHOT
   | READ WRITE
   | READ ONLY

BEGIN [WORK]
COMMIT [WORK] [AND [NO] CHAIN] [[NO] RELEASE]
ROLLBACK [WORK] [AND [NO] CHAIN] [[NO] RELEASE]
SET autocommit = {0 | 1}
```

24. MySQL - Alter Commands:

Alter command is used to change the structure of the table. This command is used when we need to add or delete, create or destroy indexes, to change the type of columns, rename columns or table itself. We can also change the comment.
The ALTER statement is always used with "ADD", "DROP" and "MODIFY" commands.

ADD Column:
It is used to add the column in table.

Syntax:
```
ALTER TABLE table_name
ADD column_name datatype;
```

Syntax to add multiple columns:
```
 ALTER TABLE table_name
 ADD new_column_name column_definition
 [ FIRST | AFTER column_name ],
ADD new_column_name column_definition
[ FIRST | AFTER column_name ],
  ...
;
```

Example:
```
alter table Student ADD Address varchar(100);
```

Output:
```
Table altered.
0.18 seconds
```

RNO	NAME	ADDRESS
1	Sanket	-
2	Tejas	-
4	Rohit	-
3	Rushi	-
6	-	-

5 rows returned in 0.01 seconds

DROP COLUMN

It is used to drop the column in table.

Syntax:

ALTER TABLE table_name
DROP COLUMN column_name;

Example:

Alter table Student DROP COLUMN Address;

Output:

Table altered.
0.18 seconds

RNO	NAME
1	Sanket
2	Tejas
4	Rohit
3	Rushi
6	-

5 rows returned in 0.01
seconds

ALTER/MODIFY Column:

It is used to change the data type of column in table.

Syntax:

SQL Server / MS Access:
ALTER TABLE table_name
ALTER COLUMN column_name datatype;

My SQL / Oracle (prior version 10G):

ALTER TABLE table_name
MODIFY COLUMN column_name datatype;

Oracle 10G and later:

ALTER TABLE table_name
MODIFY column_name datatype;

Example:

```
Alter table Student MODIFY Name char(10);
```

Output:

```
Table altered.

0.11 seconds
```

RENAME table:

Here we are able to change the table name.

Syntax:

```
ALTER TABLE table_name
RENAME TO new_table_name;
```

Example:

```
Alter table Student RENAME to Stud_info;
```

Output:

```
Table altered.
0.18 seconds
```

25. MySQL - Indexes:

An index is a data structure that improves the speed of operation, allows us to add indexes in existing table, and enables us to improve the faster retrieval of records from database tables. This creates index for each and every column in table. We use it to quickly find the record without searching each row in a database table whenever the table is accessed. We can create an index by using one or more columns of the table for efficient access to the records.
When a table is created with a primary key or unique key, it automatically creates a special index named PRIMARY. We called this index as a clustered index. All indexes other than PRIMARY indexes are known as a non-clustered index or secondary index.

Simple and Unique Index

We can create the unique index on each and every table. When two or more rows cannot have the same index value then it is called as unique index. As name it is unique. We can use one or more columns to create an index in the table.

Syntax:
CREATE UNIQUE INDEX index_name ON table_name (column1, column2,...);

For example, we can create an index on info using Name.

```
CREATE UNIQUE INDEX DATA_INDEX ON info (Name)
```

We can easily create the simple or unique index on a table. Just use UNIQUE keyword in the query to create a simple index. A simple index always allows the duplication of values in the table.
If you want the index values in descending order in a column, you can use the keyword DESC after the column name in the query.

```
mysql> CREATE UNIQUE INDEX DATA_INDEX ON info (Name DESC)
```

ALTER command to add and drop INDEX:
There are four types of statements for adding indexes to a table −

- **ALTER TABLE tbl_name ADD PRIMARY KEY (column_list)** – This command adds a PRIMARY KEY to the column.
- **ALTER TABLE tbl_name ADD UNIQUE index_name (column_list)** – This command creates the index where the values must be unique.
- **ALTER TABLE tbl_name ADD INDEX index_name (column_list)** – This command adds an index to the column in table.

- **ALTER TABLE tbl_name ADD FULLTEXT index_name (column_list)** – This command creates a special FULLTEXT index which is used

Example:
To add index in an existing table.
CREATE INDEX id_index ON Stud_info(Rno);

Drop index :
DROP INDEX 'id_index' ON 'Stud_info';

Add and drop the PRIMARY KEY

We can add primary key but the values should be NOT NULL.

Example:
ALTER TABLE Stud_info MODIFY Rno INT NOT NULL;
ALTER TABLE Stud_info ADD PRIMARY KEY (Rno);

Output:
Table altered.
0.18 seconds

You can use the ALTER command to drop a primary key as follows −

Example:
ALTER TABLE Stud_info DROP PRIMARY KEY;

Output:
Table altered.

0.11 seconds

Displaying INDEX Information
SHOW INDEX command is used to display all the index information associated with the table.

Syntax:
mysql>SHOW INDEX FROM table_name\G

Example:
SHOW INDEX FROM Stud_info\G

26. MySQL - Temporary Tables:

A temporary table is a special table which is used to store the temporary data in the table where we can reuse the temporary data multiple times in a single session.
A temporary table is very useful and flexible to perform the complex operations that requires the single SELECT statement with JOIN clauses.

Features of temporary table:

- CREATE TEMPORARY TABLE statement or command is used to create a temporary table in database. It is as same as normal table creation. The keyword TEMPORARY is added between the CREATE and TABLE keywords.
- MySQL temporary table automatically gets deleted when the user closes or ends the session or terminates the connection manually.
- Temporaary table is accessible and available to the clients who created that table where differnent clients can use the temporary tables with same name without conflicting each other because the only client have created that table. But the user cannot create the two temporary table having the same name in same session.
- A temporary table can have the same name as normal table in the database.

Create temporary table

Syntax:

```
CREATE TEMPORARY TABLE table_name(
  column_1_definition,
  column_2_definition,
  ...,
  table_constraints
);
```

Example:

```
CREATE GLOBAL TEMPORARY TABLE Students(Rno int NOT NULL, Name
VARCHAR(40) NOT NULL);
```

Output:

```
Table altered.
0.18 seconds
```

Drop temporary table

Syntax:

```
DROP TEMPORARY TABLE table_name;
```

Example:

```
DROP TABLE Students;
```

Output:

```
Table dropped.

0.32 seconds
```

27. MySQL - Clone Tables:

If there is any situation that we need an exact copy of the existing table but we do not want to edit the the existing table then we can create the copy of the table. It is nothing but cloning of the table.

Step 1: Creating an Normal Empty Table
First use the following statement to create an empty table based on the definition of original table. It also includes the column attributes and indexes defined in the original table:

```
CREATE TABLE new_table LIKE original_table;
```

Step 2: Inserting Data into Table
Now, use the following statement to populate the empty table with data from original table:

```
INSERT INTO new_table SELECT * FROM original_table;
```

Let's make a clone of the table using the MySQL command-line tool.

```
mysql> CREATE TABLE employees_clone LIKE employees;
```

Now, execute another SQL statement which inserts all the records from employees table into employees_clone table. After executing this statement you'll get the employees_clone table which is an exact copy or duplicate of the employees table

```
mysql> INSERT INTO employees_clone SELECT * FROM employees;
```

Simple Cloning

Example:

```
CREATE TABLE new_table SELECT * FROM original_table;
```

The following command creates a simple copy of the employees table.

```
mysql> CREATE TABLE employees_dummy SELECT * FROM employees;
```

28. MySQL - Database Info:

Data about data is called as metadata.
Here we can get three types of information from the table or database:

- Information about result of queries: Here we get the information about the number of records affected by the SQL commands.
- Information about tables and databases: Here we get the information about the structure of the tables and databases.
- Information about the MySQL server: Here weget the information about the database server, version, etc.

We can get metadata of server:

Sr. No.	Command	Description
1	SELECT VERSION( )	Server version string
2	SELECT DATABASE( )	Current database name (empty if none)
3	SELECT USER( )	Current username
4	SHOW STATUS	Server status indicators
5	SHOW VARIABLES	Server configuration variables

29. MySQL - Using Sequences:

Sequence is a set of integers 1, 2, 3, … that are generated and supported by some database systems to produce unique values on demand.

- A sequence is a user defined schema bound object that generates a sequence of numeric values.
- Sequences are frequently used in many databases because many applications require each row in a table to contain a unique value and sequences provides an easy way to generate them.
- The sequence of numeric values is generated in an a**scending or descending order** at defined intervals and can be configured to restart when exceeds max_value.

Syntax:

```
CREATE SEQUENCE sequence_name
START WITH initial_value
INCREMENT BY increment_value
MINVALUE minimum value
MAXVALUE maximum value
CYCLE|NOCYCLE ;
```

sequence_name: Name of the sequence.
initial_value: starting value from where the sequence starts. Initial_value should be greater than or equal to minimum value and less than equal to maximum value.
increment_value: Value by which sequence will increment itself. Increment_value can be positive or negative.
minimum_value: Minimum value of the sequence.
maximum_value: Maximum value of the sequence.
cycle: When sequence reaches its set limit it starts from beginning.
nocycle: An exception will be thrown if sequence exceeds its max_value.

Following is the sequence query creating sequence in ascending order.

Example 1:

```
CREATE SEQUENCE sequence_1
start with 1
increment by 1
minvalue 0
maxvalue 100
cycle;
```

Above query will create a sequence named sequence_1.Sequence will start from 1 and will be incremented by 1 having maximum value 100. Sequence will repeat itself from start value after exceeding 100.

Example 2:
Following is the sequence query creating sequence in descending order.

```
CREATE SEQUENCE sequence_2
start with 100
increment by -1
minvalue 1
maxvalue 100
cycle;
```

Above query will create a sequence named sequence_2.Sequence will start from 100 and should be less than or equal to maximum value and will be incremented by -1 having minimum value 1.

Example to use sequence : create a table named students with columns as id and name.

```
CREATE TABLE students
(
ID number(10),
NAME char(20)
);
```

Now insert values into table

```
INSERT into students VALUES(sequence_1.nextval,'Ramesh');
INSERT into students VALUES(sequence_1.nextval,'Suresh');
```

where sequence_1.nextval will insert id’s in id column in a sequence as defined in sequence_1.
Output:

```
 ________________________
| ID |     NAME     |
------------------------
| 1 |    Ramesh    |
| 2 |    Suresh    |
 ----------------------
```

30. MySQL - Handling Duplicates

The table may contain the duplicate record sometimes it can be allowed but

sometimes it need to verify.

For preventing it we can use Primary key or unique key.

Counting and Identifying Duplicates

Following is the query to count duplicate records with first_name and last_name in a table.

```
mysql> SELECT COUNT(*) as repetitions, last_name, first_name
  -> FROM person_tbl
  -> GROUP BY last_name, first_name
  -> HAVING repetitions > 1;
```

This query will return a list of all the duplicate records in the person_tbl table. In general, to identify sets of values that are duplicated, follow the steps given below.
Determine which columns contain the values that may be duplicated.
List those columns in the column selection list, along with the COUNT (*).
List the columns in the GROUP BY clause as well.
Add a HAVING clause that eliminates the unique values by requiring the group counts to be greater than one.

Eliminating Duplicates from a Query Result

```
mysql> SELECT DISTINCT last_name, first_name
   -> FROM person_tbl
    -> ORDER BY last_name;
```

Removing Duplicates Using Table Replacement

```
mysql> CREATE TABLE tmp SELECT last_name, first_name, sex
  -> FROM person_tbl;
  -> GROUP BY (last_name, first_name);
mysql> DROP TABLE person_tbl;
mysql> ALTER TABLE tmp RENAME TO person_tbl;
```

31. MySQL - SQL Injection:

SQL injection is a code injection technique that might destroy your database.
It is one of the most common web hacking techniques.
SQL injection is the placement of malicious code in SQL statements, via web page input.
Never trust the data provided by a user, process this data only after validation; as a rule, this is done by pattern matching.

If you use MySQL, the mysql_query () function does not permit query stacking or executing multiple queries in a single function call. If you try to stack queries, the call fails. However, other PHP database extensions, such as SQLite and PostgreSQL, happily perform stacked queries, executing all the queries provided in one string and creating a serious security problem.

Prevention of SQL Injection
It can be handled in Scripting languages like Pearl and PHP
Php by using function mysql real escape string()

32. MySQL - Database Export:

Exporting the table data to text file is done by Select… Into Outfile statement
The syntax for this statement contains a regular SELECT command with INTO OUTFILE filename at the end.
Syntax for Database Export is:

```
mysql> SELECT * FROM Table_name
        -> INTO OUTFILE 'xyz.txt';
```

Exporting Tables as Raw Data

The mysqldump program is used to copy or back up tables and databases. It can write the table output either as a Raw Datafile or as a set of INSERT statements that recreate the records in the table.

Exporting Table Contents or Definitions in SQL Format

To export a table in SQL format to a file, use the command shown below.

```
mysqldump -u root -p TUTORIALS tutorials_tbl > dump.txt

password ----
```

33. MySQL - Database Import:

There are two ways of importing data:

1. Importing Data with LOAD DATA
2. Importing Data with mysqlimport

Importing Data with LOAD DATA:

Load data statement acts as a bulk data loader which is provided by MySQL.

It is possible to import data from client to a remote MySQL database server using the LOAD DATA INFILE statement. When you use the LOCAL option in the LOAD DATA INFILE, the client program reads the file on the client and sends it to the MySQL server.

For example:

```
mysql> LOAD DATA LOCAL INFILE 'dump.txt' INTO TABLE mytbl;
```

If Local Keyword is not present then, MySQL reads the file from the given location by datafile on the server host which fully satisfy the location of file

To specify a file format explicitly, use a FIELDS clause to describe the characteristics of fields within a line, and a LINES clause to specify the line-ending sequence.

```
mysql> LOAD DATA LOCAL INFILE 'dump.txt' INTO TABLE mytbl
        -> FIELDS TERMINATED BY ':
        -> LINES TERMINATED BY '\r\n';
```

We can load the file as:

```
mysql> LOAD DATA LOCAL INFILE 'dump.txt'
        -> INTO TABLE mytbl (b, c, a);
```

Importing Data with mysqlimport

mysqlimport acts as a wrapper around LOAD DATA, it can load the input files directly from the command line.

34. MySQL Functions:

MySQL String Functions:

Function	Description
ASCII	Returns the ASCII value for the specific character
CHAR_LENGTH	Returns the length of a string (in characters)
CHARACTER_LENGTH	Returns the length of a string (in characters)
CONCAT	Adds two or more expressions together
CONCAT_WS	Adds two or more expressions together with a separator
FIELD	Returns the index position of a value in a list of values
FIND_IN_SET	Returns the position of a string within a list of strings
FORMAT	Formats a number to a format like "#,###,###.##", rounded to a specified number of decimal places
INSERT	Inserts a string within a string at the specified position and for a certain number of characters
INSTR	Returns the position of the first occurrence of a string in another string
LCASE	Converts a string to lower-case
LEFT	Extracts a number of characters from a string (starting from left)
LENGTH	Returns the length of a string (in bytes)
LOCATE	Returns the position of the first occurrence of a substring in a string
LOWER	Converts a string to lower-case
LPAD	Left-pads a string with another string, to a certain length
LTRIM	Removes leading spaces from a string
MID	Extracts a substring from a string (starting at any position)
POSITION	Returns the position of the first occurrence of a substring in a string
REPEAT	Repeats a string as many times as specified
REPLACE	Replaces all occurrences of a substring within a string, with a new substring
REVERSE	Reverses a string and returns the result
RIGHT	Extracts a number of characters from a string (starting from right)
RPAD	Right-pads a string with another string, to a certain length
RTRIM	Removes trailing spaces from a string
SPACE	Returns a string of the specified number of space characters
STRCMP	Compares two strings

SUBSTR	Extracts a substring from a string (starting at any position)
SUBSTRING	Extracts a substring from a string (starting at any position)
SUBSTRING_INDEX	Returns a substring of a string before a specified number of delimiter occurs
TRIM	Removes leading and trailing spaces from a string
UCASE	Converts a string to upper-case
UPPER	Converts a string to upper-case

MySQL Numeric Functions

Function	**Description**
ABS	Returns the absolute value of a number
ACOS	Returns the arc cosine of a number
ASIN	Returns the arc sine of a number
ATAN	Returns the arc tangent of one or two numbers
ATAN2	Returns the arc tangent of two numbers
AVG	Returns the average value of an expression
CEIL	Returns the smallest integer value that is >= to a number
CEILING	Returns the smallest integer value that is >= to a number
COS	Returns the cosine of a number
COT	Returns the cotangent of a number
COUNT	Returns the number of records returned by a select query
DEGREES	Converts a value in radians to degrees
DIV	Used for integer division
EXP	Returns e raised to the power of a specified number
FLOOR	Returns the largest integer value that is <= to a number
GREATEST	Returns the greatest value of the list of arguments
LEAST	Returns the smallest value of the list of arguments
LN	Returns the natural logarithm of a number
LOG	Returns the natural logarithm of a number, or the logarithm of a number to a specified base
LOG10	Returns the natural logarithm of a number to base 10
LOG2	Returns the natural logarithm of a number to base 2
MAX	Returns the maximum value in a set of values
MIN	Returns the minimum value in a set of values
MOD	Returns the remainder of a number divided by another number
PI	Returns the value of PI
POW	Returns the value of a number raised to the power of another number

POWER	Returns the value of a number raised to the power of another number
RADIANS	Converts a degree value into radians
RAND	Returns a random number
ROUND	Rounds a number to a specified number of decimal places
SIGN	Returns the sign of a number
SIN	Returns the sine of a number
SQRT	Returns the square root of a number
SUM	Calculates the sum of a set of values
TAN	Returns the tangent of a number
TRUNCATE	Truncates a number to the specified number of decimal places

MySQL Date Functions

Function	Description
ADDDATE	Adds a time/date interval to a date and then returns the date
ADDTIME	Adds a time interval to a time/datetime and then returns the time/datetime
CURDATE	Returns the current date
CURRENT_DATE	Returns the current date
CURRENT_TIME	Returns the current time
CURRENT_TIMESTAMP	Returns the current date and time
CURTIME	Returns the current time
DATE	Extracts the date part from a datetime expression
DATEDIFF	Returns the number of days between two date values
DATE_ADD	Adds a time/date interval to a date and then returns the date
DATE_FORMAT	Formats a date
DATE_SUB	Subtracts a time/date interval from a date and then returns the date
DAY	Returns the day of the month for a given date
DAYNAME	Returns the weekday name for a given date
DAYOFMONTH	Returns the day of the month for a given date
DAYOFWEEK	Returns the weekday index for a given date
DAYOFYEAR	Returns the day of the year for a given date
EXTRACT	Extracts a part from a given date
FROM_DAYS	Returns a date from a numeric datevalue
HOUR	Returns the hour part for a given date
LAST_DAY	Extracts the last day of the month for a given date
LOCALTIME	Returns the current date and time
LOCALTIMESTAMP	Returns the current date and time

MAKEDATE	Creates and returns a date based on a year and a number of days value
MAKETIME	Creates and returns a time based on an hour, minute, and second value
MICROSECOND	Returns the microsecond part of a time/datetime
MINUTE	Returns the minute part of a time/datetime
MONTH	Returns the month part for a given date
MONTHNAME	Returns the name of the month for a given date
NOW	Returns the current date and time
PERIOD_ADD	Adds a specified number of months to a period
PERIOD_DIFF	Returns the difference between two periods
QUARTER	Returns the quarter of the year for a given date value
SECOND	Returns the seconds part of a time/datetime
SEC_TO_TIME	Returns a time value based on the specified seconds
STR_TO_DATE	Returns a date based on a string and a format
SUBDATE	Subtracts a time/date interval from a date and then returns the date
SUBTIME	Subtracts a time interval from a datetime and then returns the time/datetime
SYSDATE	Returns the current date and time
TIME	Extracts the time part from a given time/datetime
TIME_FORMAT	Formats a time by a specified format
TIME_TO_SEC	Converts a time value into seconds
TIMEDIFF	Returns the difference between two time/datetime expressions
TIMESTAMP	Returns a datetime value based on a date or datetime value
TO_DAYS	Returns the number of days between a date and date "0000-00-00"
WEEK	Returns the week number for a given date
WEEKDAY	Returns the weekday number for a given date
WEEKOFYEAR	Returns the week number for a given date
YEAR	Returns the year part for a given date
YEARWEEK	Returns the year and week number for a given date

MySQL Advanced Functions

Function	**Description**
BIN	Returns a binary representation of a number
BINARY	Converts a value to a binary string
CASE	Goes through conditions and return a value when the first condition is met
CAST	Converts a value (of any type) into a specified datatype
COALESCE	Returns the first non-null value in a list

CONNECTION_ID	Returns the unique connection ID for the current connection
CONV	Converts a number from one numeric base system to another
CONVERT	Converts a value into the specified datatype or character set
CURRENT_USER	Returns the user name and host name for the MySQL account that the server used to authenticate the current client
DATABASE	Returns the name of the current database
IF	Returns a value if a condition is TRUE, or another value if a condition is FALSE
IFNULL	Return a specified value if the expression is NULL, otherwise return the expression
ISNULL	Returns 1 or 0 depending on whether an expression is NULL
LAST_INSERT_ID	Returns the AUTO_INCREMENT id of the last row that has been inserted or updated in a table
NULLIF	Compares two expressions and returns NULL if they are equal. Otherwise, the first expression is returned
SESSION_USER	Returns the current MySQL user name and host name
SYSTEM_USER	Returns the current MySQL user name and host name
USER	Returns the current MySQL user name and host name
VERSION	Returns the current version of the MySQL database

35. MySQL - interview question and answer

1. What is MySQL?
MySQL is an open source DBMS which is built, supported and distributed by MySQL AB (now acquired by Oracle)

2. What are the technical features of MySQL?
MySQL database software is a client or server system which includes
Multithreaded SQL server supporting various client programs and libraries
Different backend
Wide range of application programming interfaces and
Administrative tools.

3. Why MySQL is used?
MySQL database server is reliable, fast and very easy to use. This software can be downloaded as freeware and can be downloaded from the internet.

4. What are Heap tables?
HEAP tables are present in memory and they are used for high speed storage on temporary basis.

• BLOB or TEXT fields are not allowed
• Only comparison operators can be used =, <,>, = >,=<
• AUTO_INCREMENT is not supported by HEAP tables
• Indexes should be NOT NULL

5. What is the default port for MySQL Server?
The default port for MySQL server is 3306.
MySQL.svg

6. What are the advantages of MySQL when compared with Oracle?
MySQL is open source software which is available at any time and has no cost involved.
MySQL is portable
GUI with command prompt.
Administration is supported using MySQL Query Browser

7. Differentiate between FLOAT and DOUBLE?
Following are differences for FLOAT and DOUBLE:

• Floating point numbers are stored in FLOAT with eight place accuracy and it has four bytes.
• Floating point numbers are stored in DOUBLE with accuracy of 18 places and it has eight bytes.

8. Differentiate CHAR_LENGTH and LENGTH?
CHAR_LENGTH is character count whereas the LENGTH is byte count. The numbers are same for Latin characters but they are different for Unicode and other encodings.

9. How to represent ENUMs and SETs internally?
ENUMs and SETs are used to represent powers of two because of storage optimizations.

10. What is the usage of ENUMs in MySQL?
ENUM is a string object used to specify set of predefined values and that can be used during table creation.
Create table size(name ENUM('Small', 'Medium','Large');

11. Define REGEXP?
REGEXP is a pattern match in which matches pattern anywhere in the search value.

12. Difference between CHAR and VARCHAR?
Following are the differences between CHAR and VARCHAR:
CHAR and VARCHAR types differ in storage and retrieval
CHAR column length is fixed to the length that is declared while creating table. The length value ranges from 1 and 255
When CHAR values are stored then they are right padded using spaces to specific length. Trailing spaces are removed when CHAR values are retrieved.

13. Give string types available for column?
The string types are:
SET
BLOB
ENUM
CHAR
TEXT
VARCHAR

14. How to get current MySQL version?
SELECT VERSION (); is used to get the current version of MySQL.

15. What storage engines are used in MySQL?
Storage engines are called table types and data is stored in files using various techniques.
Technique involves:
Storage mechanism
Locking levels
Indexing
Capabilities and functions.

16. What are the drivers in MySQL?
Following are the drivers available in MySQL:
PHP Driver
JDBC Driver
ODBC Driver
C WRAPPER
PYTHON Driver
PERL Driver
RUBY Driver
CAP11PHP Driver
Ado.net5.mxj

17. What does a TIMESTAMP do on UPDATE CURRENT_TIMESTAMP data type?
TIMESTAMP column is updated with Zero when the table is created. UPDATE CURRENT_TIMESTAMP modifier updates the timestamp field to current time whenever there is a change in other fields of the table.

18. What is the difference between primary key and candidate key?
Every row of a table is identified uniquely by primary key. There is only one primary key for a table.
Primary Key is also a candidate key. By common convention, candidate key can be designated as primary and which can be used for any foreign key references.

19. How do you login to MySql using Unix shell?
We can login through this command:
```
# [mysql dir]/bin/mysql -h hostname -u <UserName> -p <password>
```

20. What does myisamchk do?
It compress the MyISAM tables, which reduces their disk or memory usage.

21. How do you control the max size of a HEAP table?
Maximum size of Heal table can be controlled by MySQL config variable called max_heap_table_size.

22. What is the difference between MyISAM Static and MyISAM Dynamic?
In MyISAM static all the fields will have fixed width. The Dynamic MyISAM table will have fields like TEXT, BLOB, etc. to accommodate the data types with various lengths. MyISAM Static would be easier to restore in case of corruption.

23. What are federated tables?
Federated tables which allow access to the tables located on other databases on other servers.

24. What, if a table has one column defined as TIMESTAMP?
Timestamp field gets the current timestamp whenever the row gets altered.

25. What happens when the column is set to AUTO INCREMENT and if you reach maximum value in the table?
It stops incrementing. Any further inserts are going to produce an error, since the key has been used already.

26. How can we find out which auto increment was assigned on Last insert?
LAST_INSERT_ID will return the last value assigned by Auto_increment and it is not required to specify the table name.

27. How can you see all indexes defined for a table?
Indexes are defined for the table by:
SHOW INDEX FROM <tablename>;

28. What do you mean by % and _ in the LIKE statement?
% corresponds to 0 or more characters, _ is exactly one character in the LIKE statement.

29. How can we convert between Unix & MySQL timestamps?
UNIX_TIMESTAMP is the command which converts from MySQL timestamp to Unix timestamp
FROM_UNIXTIME is the command which converts from Unix timestamp to MySQL timestamp.

30. What are the column comparisons operators?
The = , <>, <=, <, >=, >,<<,>>, <=>, AND, OR, or LIKE operators are used in column comparisons in SELECT statements.

31. How can we get the number of rows affected by query?
Number of rows can be obtained by
SELECT COUNT (user_id) FROM users;

32. Is Mysql query is case sensitive?
No.
SELECT VERSION(), CURRENT_DATE;
SeLect version(), current_date;
seleCt vErSiOn(), current_DATE;
All these examples are same. It is not case sensitive.

33. What is the difference between the LIKE and REGEXP operators?
LIKE and REGEXP operators are used to express with ^ and %.
SELECT * FROM employee WHERE emp_name REGEXP "^b";
SELECT * FROM employee WHERE emp_name LIKE "%b";

34. What is the difference between BLOB AND TEXT?
A BLOB is a binary large object that can hold a variable amount of data. There are four types of BLOB –
TINYBLOB
BLOB
MEDIUMBLOB and
LONGBLOB
They all differ only in the maximum length of the values they can hold.
A TEXT is a case-insensitive BLOB. The four TEXT types
TINYTEXT
TEXT
MEDIUMTEXT and
LONGTEXT
They all correspond to the four BLOB types and have the same maximum lengths and storage requirements.
The only difference between BLOB and TEXT types is that sorting and comparison is performed in case-sensitive for BLOB values and case-insensitive for TEXT values.

35. What is the difference between mysql_fetch_array and mysql_fetch_object?
Following are the differences between mysql_fetch_array and mysql_fetch_object:
mysql_fetch_array() -Returns a result row as an associated array or a regular array from database.
mysql_fetch_object – Returns a result row as object from database.

36. How can we run batch mode in mysql?
Following commands are used to run in batch mode:
mysql ;
mysql mysql.out

37. Where MyISAM table will be stored and also give their formats of storage?
Each MyISAM table is stored on disk in three formats:
The '.frm' file stores the table definition
The data file has a '.MYD' (MYData) extension
The index file has a '.MYI' (MYIndex) extension

38. What are the different tables present in MySQL?
Total 5 types of tables are present:
MyISAM
Heap
Merge
INNO DB
ISAM
MyISAM is the default storage engine as of MySQL .

39. What is ISAM?

ISAM is abbreviated as Indexed Sequential Access Method.It was developed by IBM to store and retrieve data on secondary storage systems like tapes.

40. What is InnoDB?

lnnoDB is a transaction safe storage engine developed by Innobase Oy which is a Oracle Corporation now.

41. How MySQL Optimizes DISTINCT?

DISTINCT is converted to a GROUP BY on all columns and it will be combined with ORDER BY clause.
SELECT DISTINCT t1.a FROM t1,t2 where t1.a=t2.a;

42. How to enter Characters as HEX Numbers?

If you want to enter characters as HEX numbers, you can enter HEX numbers with single quotes and a prefix of (X), or just prefix HEX numbers with (Ox).
A HEX number string will be automatically converted into a character string, if the expression context is a string.

43. How to display top 50 rows?

In MySql, top 50 rows are displayed by using this following query:
SELECT * FROM
LIMIT 0,50;

44. How many columns can be used for creating Index?

Maximum of 16 indexed columns can be created for any standard table.

45. What is the different between NOW() and CURRENT_DATE()?

NOW () command is used to show current year,month,date with hours,minutes and seconds.
CURRENT_DATE() shows current year,month and date only.

46. What are the objects can be created using CREATE statement?

Following objects are created using CREATE statement:
DATABASE
EVENT
FUNCTION
INDEX
PROCEDURE
TABLE
TRIGGER
USER
VIEW

47. How many TRIGGERS are allowed in MySql table?
SIX triggers are allowed in MySql table. They are as follows:
BEFORE INSERT
AFTER INSERT
BEFORE UPDATE
AFTER UPDATE
BEFORE DELETE and
AFTER DELETE

48. What are the nonstandard string types?
Following are Non-Standard string types:
TINYTEXT
TEXT
MEDIUMTEXT
LONGTEXT

49. What are all the Common SQL Function?
CONCAT(A, B) – Concatenates two string values to create a single string output. Often used to combine two or more fields into one single field.
FORMAT(X, D) – Formats the number X to D significant digits.
CURRDATE(), CURRTIME() – Returns the current date or time.
NOW() – Returns the current date and time as one value.
MONTH(), DAY(), YEAR(), WEEK(), WEEKDAY() – Extracts the given data from a date value.
HOUR(), MINUTE(), SECOND() – Extracts the given data from a time value.
DATEDIFF(A, B) – Determines the difference between two dates and it is commonly used to calculate age
SUBTIMES(A, B) – Determines the difference between two times.
FROMDAYS(INT) – Converts an integer number of days into a date value.

50. Explain Access Control Lists.
An ACL (Access Control List) is a list of permissions that is associated with an object. This list is the basis for MySQL server's security model and it helps in troubleshooting problems like users not being able to connect.

MySQL keeps the ACLs (also called grant tables) cached in memory. When a user tries to authenticate or run a command, MySQL checks the authentication information and permissions against the ACLs, in a predetermined order.