






















Important MySQL Short Notes Pdf For Beginners 2021

Learning SQL (Structured Query Language) is so easy for beginners if they use short note pdf. In this post, we will learn some important MySQL short notes with pdf.

Some Important MySQL Topic For Interview.

In this MySQL tutorial, we will learn some most important topics for interviews. For beginners the basic topic of MySQL is important. That's why we will focus on the basic topic.

-  MySQL Create
-  MySQL Insert
-  MySQL Select
-  MySQL Where
-  MySQL Update
-  MySQL Delete
-  MySQL Limit
-  MySQL AND, OR, NOT
-  MySQL Aggregate Function
-  MySQL Like
-  MySQL IN
-  MySQL BETWEEN
-  MySQL Order By
-  MySQL NULL Value
-  MySQL Group By
-  MySQL Having
-  MySQL EXISTS
-  MySQL Aliases
-  MySQL Case
-  MySQL Join
-  MySQL Union
-  MySQL INSERT INTO SELECT
-  MySQL Comment

👉 Is MySQL Keywords Are Case Sensitive?

📖 No, MySQL keywords are not case-sensitive. *insert* and *INSERT* are the same.

📖 But writing Upper-case is best practice.

```
INSERT INTO students(id,name,age,phone) VALUES(6,'Noman',16,'016349823xx')
```

```
insert into students(id,name,age,phone) values(6,'Noman',16,'016349823xx')
```

Both queries are the same and valid.

👉 Is Semicolon Mandatory After The MySQL Statement?

📖 The semicolon is not mandatory after the MySQL statement.

📖 But to run multiple queries at the same time we need to use semicolons to separate each of them.

```
INSERT INTO students(id,name,age,phone) VALUES(6,'Noman',16,'016349823xx')
```

No need semicolon(;) because we run only a single query.

```
INSERT INTO students(id,name,age,phone) VALUES(6,'Noman',16,'016348823xx');
```

```
INSERT INTO students(id,name,age,phone) VALUES(7,'Arafat',20,'016339823xx');
```

```
INSERT INTO students(id,name,age,phone) VALUES(8,'Kamrul',19,'016345823xx');
```

```
INSERT INTO students(id,name,age,phone) VALUES(9,'Monju',17,'014345823xx')
```

We Need a semicolon(;) because we run multiple queries at the same time.

👉 MySQL CREATE Statement.

📖 SQL CREATE Statement is used to create a Database.

📖 SQL CREATE Statement is used to create a Table.

```
CREATE DATABASE student_db
```

This query will create a database(name: student_db).

```
CREATE TABLE students (  
id INT(6) UNSIGNED AUTO INCREMENT PRIMARY KEY,  
name VARCHAR(30) NOT NULL,  
age INT(6),  
phone VARCHAR(30) NOT NULL  
)
```

This query will create a table(name: students).

🧐 Learn MoreThe Best way of creating a table in MySQL.

👉 MySQL INSERT Statement.

📌 The SQL INSERT statement is used to insert data into the database.

📌 We don't need to use column name when we INSERT data, if Table columns order and given data order are the same and We provide all columns value.

📌 Insert query return true or false. If data INSERT success it will return true, Else it will return false.

Suppose we have a 'students' table.

ID	Name	Age	Phone
1	Nasir	20	01637019XX
2	MIM	17	01637069XX
3	SAJIB	19	01633019XX
4	Imam	20	01437019XX
5	Arafat	18	01237019XX

```
INSERT INTO students(id,name,age,phone) VALUES (6,'Noman',16,'016349823xx');
```

This query will insert a new student.

```
INSERT INTO students VALUES(6,'Noman',16,'016349823xx');
```

This query will insert a new student.

👉 MySQL SELECT Statement.

📌 SELECT Statement is used to fetch data from the database.

📌 To Fetch all columns we need to use * sign. (* = All).

📌 We can also specify which column we want to fetch.

```
SELECT * FROM students
```

This query will fetch all columns from the 'students' table.

```
SELECT name,email,phone FROM students
```

This query will fetch only name, email, and phone columns from the 'students' table.

👉 MySQL WHERE Statement.

- 📖 Where clause is used to define the condition in SQL statement.
- 📖 We can fetch data Conditionally.
- 📖 It works like the "if" condition.

```
SELECT * FROM students WHERE age > 18
```

This query will fetch all students whose age is greater than 18.

👉 MySQL UPDATE Statement.

- 📖 UPDATE Statement is used to update record.
- 📖 When we use update statements we need to use conditions with where clause.
- 📖 If we don't use where clause properly all records will update automatically.

Don't do like this

```
UPDATE students SET age = 19
```

This query will update all student's records.

Do like this

```
UPDATE students SET age = 19 WHERE id = 5
```

This query will update only the 5th id's student.

👉 MySQL DELETE Statement.

- 📖 DELETE Statement is used to Delete record.
- 📖 When we use delete statements we need to use conditions with where clause.
- 📖 If we don't use the where clause properly all records will delete automatically.

Don't do like this

```
DELETE FROM students;
```

This query will Delete all student's records.

Do like this

```
DELETE FROM students WHERE id = 5
```

This query will delete only the 5th id's student.

👉 MySQL LIMIT Statement.

📖 We can use the LIMIT statement to specify how many records we want to fetch from the table.

📖 Also, we can specify FROM which record fetching will start.

```
SELECT * FROM students LIMIT 5;
```

This query return 5 records.

```
SELECT * FROM students LIMIT 3, 8;
```

This query will fetch 8 records FROM the 4th row to the 11th row.

👉 MySQL AND, OR, NOT Statement.

📖 we can use MySQL AND, OR, NOT conditions with WHERE clause like this.

```
SELECT * FROM students WHERE age > 18 AND age < 21
```

This query returns all students whose age is greater than 18 and less than 21.

```
SELECT * FROM students WHERE age > 18 OR size > 6
```

This query returns all students whose age is greater than 18 and size greater than 6.

```
SELECT * FROM students WHERE NOT class = 10
```

This query returns all students whose class not 10.

👉 MySQL MySQL Aggregate Functions.

📖 MySQL Aggregate functions are so important topic.

📖 In general, we have five Aggregate functions.

- 📌 1. COUNT()
↑ This function will count all rows.
- 📌 2. MAX()
↑ This function will return the maximum value from all rows.
- 📌 3. MIN()
↑ This function will return the minimum value from all rows.
- 📌 4. AVG()
↑ This function will return the average value from all rows.
- 📌 5. SUM()
↑ This function will return the summation value from all rows.

💖 Learn MySQL Aggregate functions for example.

👉 MySQL LIKE Statement.

- 📌 To create search features we can use MySQL LIKE statement.
- 📌 For pattern matching, we need to use MySQL LIKE Statement.
- 📌 The SQL LIKE statements is so important topic in the database. you can do so many different things using this LIKE statement.
- 📌 SQL LIKE statement is so important for Interview and EXAM.

```
SELECT * FROM students WHERE name LIKE "%keyword%"
```

This query will return all records if in the name our keyword is present at any place.

😍 Learn MySQL LIKE statement with a real-life example.

👉 MySQL Wildcard Characters.

- 📌 We use Wildcard Characters with LIKE statements.
- 📌 MySQL has 2 wildcard Characters percentages and underscores (% and _).
- 📌 Percentage(%) represents zero or more characters.
- 📌 Underscores(_) represents a single character.

😍 Learn Use of MySQL wildcard characters with an example.

👉 MySQL IN Statement.

- 📌 Using MySQL IN operator we can use multiple values in where close condition. It works like this OR operator.
- 📌 MySQL IN operator works like PHP in_array() function.

```
SELECT * FROM students WHERE age IN (17,18,19,20);
```

This query will return all records if the age is 17,18,19, or 20.

👉 MySQL BETWEEN Statement.

📌 TO select data FROM a range we can use MySQL BETWEEN operator.

📌 MySQL BETWEEN operator works like \geq and \leq . That means it includes the condition values.

```
SELECT * FROM students WHERE age BETWEEN 18 AND 20
```

This query will return all records if the age is 18,19, or 20.

👉 MySQL ORDER BY Statement.

📌 TO sort data we can use ORDER BY and ASC(ascending), DSC (descending) keyword.

📌 Using this ORDER BY keyword you can nth maximum or nth minimum records easily.

```
SELECT * FROM students ORDER BY age ASC
```

This query will return all records in ascending order according to age.

```
SELECT * FROM students ORDER BY age DSC
```

This query will return all records in descending order according to age.

👉 MySQL NULL Value

📌 A NULL value means no value.

📌 NULL value and zero(0) value are not the same.

📌 We can not use conditional operator($<$, $<=$, $!$) to check NULL value.

📌 TO check NULL value we need to use IS NULL or IS NOT a NULL statement.

```
SELECT * FROM students where name IS NOT NULL
```

This query will return true or false.

👉 MySQL GROUP BY Statement.

📌 Using MySQL Group BY statement we can fetch row BY Group-wise.

📌 Most of the time we use MySQL Group Statement with our Aggregate functions.

📌 We can use multiple column names in the MySQL Group BY Statement.

```
SELECT count(*) AS total, name, age  
FROM students  
GROUP BY name, age;
```

This query will return students name and ages Group-wise.

👉 MySQL HAVING Statement.

📖 Technically the MySQL WHERE and HAVING is the same but We can not use WHERE statements with the Aggregate function that's why we need to use HAVING close.

```
SELECT COUNT(age) AS Total, Name  
FROM Students  
GROUP BY age  
HAVING COUNT(age) > 17;
```

This query will return total students age name Group-wise age where total age greater than 17

👉 MySQL EXISTS Statement.

- 📖 MySQL EXISTS Operator used with a subquery.
- 📖 It returns true or false, if data exists it returns true or not return false.
- 📖 MySQL EXISTS is so powerful when we need to run complex queries.

👉 MySQL Aliases Statement.

- 📖 Using MySQL Aliases we can create a temporary name of our table and columns.
- 📖 To create Aliase we need to use MySQL AS Statement.
- 📖 To run a complex query we need to use Aliases.

```
SELECT SUM(age) AS Total_AGE FROM students;
```

This query will return the total age as the Total_Age column name.

👉 Learn MySQL Aliases with an example.

👉 MySQL CASE Statement.

- 📖 To select data based on multiple conditions we can use MySQL CASE statement.
- 📖 MySQL CASE Statement work like an if-else condition.
- 📖 MySQL CASE Statement works like a switch case.
- 📖 MySQL CASE Statement is so useful topic for Interviews.

👉 Learn MySQL Case Statement with an Example.

👉 MySQL JOIN Query

- 📖 MySQL joins are used to fetch data FROM multiple tables.
- 📖 MySQL joins is one of the main topics in relational databases.
- 📖 MySQL CASE Statement works like a switch case.
- 📖 In general, there are four types of joins present in MySQL.

- 📌 INNER JOIN
- 📌 LEFT JOIN
- 📌 RIGHT JOIN
- 📌 CROSS JOIN

🧐 Must Learn How MySQL JOIN statements work.

👉 MySQL UNION Operator

- 📖 In general, the MySQL UNION Statement works like the union of mathematics.
- 📖 MySQL Union is one of the important topics for interviews and exams.

🧐 Must Learn How MySQL Union Statement work?

👉 MySQL INSERT INTO SELECT Statement

- 📖 To create a table based on another table we can use MySQL INSERT INTO SELECT statement.
- 📖 INSERT INTO SELECT is so important for Interview and EXAM.

🧐 Must Learn MySQL INSERT INTO SELECT statement with an example.

👉 MySQL Comments

- 📖 Like other programming languages, MySQL has also comment features.
- 📖 MySQL Comments is so useful and important for exam and interview.
- 📖 Single line comment -- this is a comment.
- 📖 Multi-line comment /* this is multi-line comment */.
- 📖 We can please comments inside the MySQL Query.

```
SELECT name, phone FROM students; --this is a single-line comment
```

This is an example of a single-line comment.

```
SELECT name, phone FROM students;
```

```
/*
```

```
This is
```

```
multi-line comment
```

```
*/
```

This is an example of a multi-line comment.

```
SELECT name, /*age,*/ phone FROM students;
```

This query will return only the name and phone from the students table.

💖💖 For any kind of issue, Feel Free to [Contact](#) Us.

💖💖 Check Our [Services](#).

Learn More:

- [FrontEnd Topic](#)
- [BackEnd Topic](#)



© ALL RIGHTS RESERVED. [INSIDETHEDIV](#).