mysql > help

mysql > use database-name; #you can use database which you want

mysql > show database;

mysql > show tables;

mysql > create database database-name;

mysql > select \* from table-name;

mysql > quit

mysql > desc table;

0. CREATE DATABASE *dbname*;

0.1. CREATE TABLE *table\_name*

(

*id data\_type*(*size*) UNIQUE PRIMARY KEY AUTO\_INCREMENT,

*column\_name2 varchar*(*225*) NOT NULL,

*column\_name3 data\_type*(*size*),

....

);

0.2. INSERT INTO *table\_name*

(*column1-id*, *column2-name, column3*)

VALUES (*value1*,’*value2’,’value3’*,...);

1.SELECT \* FROM *table\_name*; // All rows will display with all columns

2.SELECT *column1\_name*,*column2\_name // column1* ,*column2 data will display*

FROM *table\_name*;

3.SELECT DISTINCT *salary*, *govt // DISTINCT will filter*

FROM *table\_name*;

4.SELECT \* FROM Customers // Exact matching records will display

WHERE sal = 20000;

5.SELECT \* FROM Customers // Exact matching records will display with two conditions

WHERE sal = 10000

AND depart= 'computer';

5.1. SELECT *name* // Exact matching records will display with s letter

FROM *table\_name*

WHERE *name* LIKE *‘%s%’*;

5.2. SELECT \* FROM Customers // Exact matching records will display with starting s letter

WHERE City LIKE 's%';

5.3. SELECT \* FROM Customers // Exact matching records will display without sri name

WHERE Country NOT LIKE '%sri%';

5.4. SELECT \* FROM Customers // Exact matching records will display with sri name

WHERE Country LIKE '%sri%';

6.select id, name from details order by id desc; // Exact matching records will display desc order

7. INSERT INTO *table\_name*

(*column1-id*, *column2-name,* ,*column3*)

VALUES (*value1*,’*value2’,’value3’*,...);

8. UPDATE *table\_name* // update any value in table

SET *column1*=’*value1’*,*column2*=’*value2’*,...

WHERE *some\_column*=’*some\_value’*;

9.DELETE FROM *table\_name //delete any row*

WHERE *some\_column*=*some\_value*;

10. SELECT *column\_name(s)* // Exact matching records will display in range

FROM *table\_name*

WHERE *column\_name* IN (*value1*,*value2*,...);

11.SELECT *column\_name(s)*

FROM *table\_name*

WHERE *column\_name* BETWEEN *value1* AND *value2;*

12. CREATE TABLE *table\_name*

(

*id data\_type*(*size*) UNIQUE, PRIMARY KEY AUTO\_INCREMENT,

*name data\_type*(*size*) NOT NULL,

*column\_name3 data\_type*(*size*),

....

);

13. DROP TABLE table\_name;

14. DROP DATABASE database\_name;

15. ALTER TABLE table\_name ADD column\_name datatype;

15.0. ALTER TABLE Orders ADD FOREIGN KEY (P\_Id) REFERENCES Persons(P\_Id);

15.1. ALTER TABLE table\_name DROP COLUMN column\_name;

15.2. RENAME TABLE *table-Name* TO*[new-Table-Name](http://docs.oracle.com/javadb/10.8.3.0/ref/rrefnewtablename.html#rrefnewtablename)*

15.3. ALTER TABLE table\_name DROP COLUMN column\_name;

16. SELECT NOW(),CURDATE(),CURTIME();

17. ALTER TABLE TABLE CHANGE OLD\_COLUMN NEW\_COLUMN VARCHAR(255) NOT NULL; #rename column

17.0) UPDATE table\_name

SET column1=value, column2=value2,...

WHERE some\_column=some\_value;

## 17. SQL Aggregate Functions

* AVG() - Returns the average value
* COUNT() - Returns the number of rows
* FIRST() - Returns the first value
* LAST() - Returns the last value
* MAX() - Returns the largest value
* MIN() - Returns the smallest value
* SUM() - Returns the sum
* GROUP BY - Returns the smallest value
* HAVING - Returns the sum
* LIMIT 10 - it shows first 10 record
* ORDER BY - ASC |DESC

18. SELECT LastName,FirstName,Address FROM Persons WHERE Address IS NOT NULL;

## 19. SQL INNER JOIN Keyword

The INNER JOIN keyword selects all rows from both tables as long as there is a match between the columns in both tables.

### SQL INNER JOIN Syntax

SELECT *column\_name(s)*

FROM *table1*

INNER JOIN *table2*

ON *table1.column\_name*=*table2.column\_name*;

or:

SELECT *column\_name(s)*

FROM *table1*

JOIN *table2*

ON *table1.column\_name*=*table2.column\_name*;

**PS!** INNER JOIN is the same as JOIN.



## 20. SQL LEFT JOIN Keyword

The LEFT JOIN keyword returns all rows from the left table (table1), with the matching rows in the right table (table2). The result is NULL in the right side when there is no match.

### SQL LEFT JOIN Syntax

SELECT *column\_name(s)*

FROM *table1*

LEFT JOIN *table2*

ON *table1.column\_name*=*table2.column\_name*;

or:

SELECT *column\_name(s)*

FROM *table1*

LEFT OUTER JOIN *table2*

ON *table1.column\_name*=*table2.column\_name*;

**PS!** In some databases LEFT JOIN is called LEFT OUTER JOIN.



## 21.SQL RIGHT JOIN Keyword

The RIGHT JOIN keyword returns all rows from the right table (table2), with the matching rows in the left table (table1). The result is NULL in the left side when there is no match.

### SQL RIGHT JOIN Syntax

SELECT *column\_name(s)*

FROM *table1*

RIGHT JOIN *table2*

ON *table1.column\_name*=*table2.column\_name*;

or:

SELECT *column\_name(s)*

FROM *table1*

RIGHT OUTER JOIN *table2*

ON *table1.column\_name*=*table2.column\_name*;

**PS!** In some databases RIGHT JOIN is called RIGHT OUTER JOIN.



## 22. SQL FULL OUTER JOIN Keyword

The FULL OUTER JOIN keyword returns all rows from the left table (table1) and from the right table (table2).

The FULL OUTER JOIN keyword combines the result of both LEFT and RIGHT joins.

### SQL FULL OUTER JOIN Syntax

SELECT *column\_name(s)*

FROM *table1*

FULL OUTER JOIN *table2*

ON *table1.column\_name*=*table2.column\_name*;



## 23. SQL Aliases

SQL aliases are used to give a database table, or a column in a table, a temporary name.

Basically aliases are created to make column names more readable.

### SQL Alias Syntax for Columns

SELECT *column\_name* AS *alias\_name*

FROM *table\_name;*

### SQL Alias Syntax for Tables

SELECT *column\_name(s)*

FROM *table\_name* AS *alias\_name;*

Simplest way to fetch second max salary & nth salary

select DISTINCT(salary) from employee order by salary desc limit 1,1

Note:

limit 0,1 - Top max salary

limit 1,1 - Second max salary

limit 2,1 - Third max salary

limit 3,1 - Fourth max salary

**Powered By: SRS**

**Srinivas**