**MYSQL ALL QUERIES**

**CRUD OPERATION**

1. **Listing All Existed Databases:-**

**SHOW DATABASES;**

1. **Create Database:-**

**CREATE DATABASE <DB\_NAME>;**

1. **Shift Database to Current Create Database:-**

**USE <DB\_NAME>;**

1. **Clear All Cmd:-**

**system cls;**

1. **Cmd To Check Which Database you Are Using:-**

**SELECT DATABASE();**

1. **DELETING A DATABASE:-**

**DROP DATABASE<DB\_NAME>;**

1. **CREATE A COLLECTION IN DATABASES:-**

**CREATE TABLE <TB\_NAME>(<COL1> <DATATYPE>,<COL2> <DATATYPE>);**

1. **Check Table is Created Or Not:-**

**SHOW TABLES;**

1. **DELETING TABLE FROM DATABASE:-**

**DROP TABLE <TB\_NAME>;**

1. **Cmd for Table Information:-**

**DESC <TB\_NAME>;**

1. **INSERT VALUE INTO TABLE:-**

**INSERT INTO <TB\_NAME> (<COL1>,<COL2>,<COL3>) VALUES**

**(<VAL1>,<VAL1>,<VA L1>),(<VAL2>,<VAL2>,<VAL2>);**

**Note:- If U are insert the value in same collection sequence no need to write collection name while insert the data  
  
INSERT INTO <TB\_NAME> VALUES (<VAL1>,<VAL2>,<VAL3>);**

1. **READING DATA FROM TABLES:-**

**SELECT \* FROM <TB\_NAME>;**

**All Data are show**

1. **READING DATA BASE ON COLLECTION:-**

**SELECT <COL\_NAME> FROM <TB\_NAME>;**

**Show that specific Collection Value;**

1. **READING DATA BASE ON CONDITION:-**

**SELECT <COL\_NAME> FROM <TB\_NAME> WHERE <CONDITION>;**

**SELECT \* FROM <TB\_NAME> WHERE <CONDITION>;**

**Base on the Conditions Display Data**

1. **Insert New Collection into Existed Table:-**

**ALTER TABLE <TB\_NAME> ADD COLUMN <COL\_NAME> <DATATYPE> AFTER <LOCATION>;**

**INSERT INTO <TB\_NAME> (<Col>) VALUES (<Val>);**

**New Collection insert With Value**

1. **Add Multiple attribute with one cmd:-**

**ALTER TABLE <TB\_NAME> ADD COLUMN (<COL1><DATATYPES>), ADD COLUMN (<COL2><DATATYPES>), ADD COLUMN (<COL3> <DATATYPES>);**

1. **Delete Attribute from existed Table:-**

**ALTER TABLE <TB\_NAME> DROP COLUMN <COL>;**

1. **Modify Columns DataType in Existed Table:-**

**ALTER TABLE <TB\_NAME> MODIFY COLUMN <COL\_NAME> <COL\_NEW\_DATATYPE>;**

1. **Modify/Update data from a Table:-**

**UPDATE <TB\_NAME> SET <COL\_NAME>=<VAL>**

**WHERE <Condition>;**

1. **Delete Value From Table:-**

**DELETE FROM SQL\_P**

**WHERE ID IS NULL AND NAME IS NULL AND CITY IS NULL;**

**DELETE FROM <TB\_NAME> WHERE <CONDITION>;**

1. **Default Value Change in Table:-**

**ALTER TABLE <TB\_NAME> MODIFY <COL> <DATATYPES> DEFAULT ‘<Acc to u’;**

1. **Update Multiple Entity Corresponding to single Attribute:-**

**UPDATE CUSTOMERS**

**SET DIVISION = CASE**

**WHEN ID = 101 THEN '8A11'**

**WHEN ID = 102 THEN '8B12'**

**WHEN ID = 103 THEN '8C13'**

**WHEN ID = 104 THEN '8A6'**

**ELSE DIVISION -- Keeps the value unchanged for other rows**

**END**

**WHERE ID IN (101, 102, 103, 104);**

1. **Rename Table Name Or Attribute Name:-;**

**ALTER TABLE <TB\_NAME> RENAME TO <NEW\_TABLE\_NAME>  
  
ALTER TABLE <TB\_NAME> RENAME COLUMN <OLD\_COL\_NAME> TO <NEW\_COL\_NAME>;**

1. **Check If Any Primary Key Exist:-**

**SHOW INDEX FROM <TB\_NAME>;**

1. **DROP PRIMARY KEY FROM TABLE:-**

**ALTER TABLE <TB\_NAME> DROP PRIMARY KEY;**

1. **Modify the Column and Add New Primary Key with auto increment:-**

**ALTER TABLE <TB\_NAME> MODIFY COLUMN <COL\_NAME> INT PRIMARY KEY AUTO\_INCREMENT;**

1. **Display The Data In decrease order:-**

**SELECT \* FROM <tb\_name>**

**ORDER BY <COL\_PRIMARY\_KEY) DESC;**

**ALIAS FUNCTION**

1. **Alias Function To Change Attribute Name While During Display Data:-**

**SELECT <COL> AS <AS\_ACC\_TO\_COL\_NAME> FROM <TB\_NAME>;**

**STRING FUNCTION**

**CONCAT STRING:-**

1. **MERGE TWO STRING/ATTRIBUTE VALUE USING CONCAT FUNCTION USING SQL CMD:-**

**SELECT CONCAT(<’VAL1’>,<’VAL2’>);**

**SELECT CONCAT(<COL1>,<COL2>) AS <WhatEver\_Attribute\_Name\_You\_Want> FROM <TB\_NAME>;**

**Note:- Display Only Fname & Lname in Concat**

**SELECT \*, CONCAT(<COL1>,<COL2>) FROM <TB\_NAME>;**

**Note:- Display All Data**

**SELECT CONCAT(<FNAME>,<’VAL’>) FROM EMPLOYEES;**

**Note:- Display Fname with Val**

**CONCAT\_WS(CONCAT WITH SEPARATOR):-**

1. **MERGE TWO STRING/ATTRIBUTE VALUES WITH SEPARATOR USING CONCAT\_WS FUNCTION USING SQL CMD;**

**SELECT CONCAT\_WS(‘-’,<’VAL1’>,<’VAL2’>);**

**Note:- Display Value with separator**

**SELECT CONCAT\_WS(‘-’,<COL1>,<COL2>) FROM EMPLOYEES;**

**Note:- Display Concat Attribute Value with separator;**

**SELECT \*, CONCAT\_WS(‘-’,<COL1>,<COL2>) FROM EMPLOYEES;**

**Note:-Display All Data With Concat\_WS Function**

**SELECT CONCAT\_WS(‘-’,<COL1>,<’VAL1’>) FROM EMPLOYEES;**

**Note:-Display Attribute or value With Separator**

**SUBSTR STRING FUNCTIONS:-(Display Char from Start to end);**

1. **By using SUBSTRING FUNCTION DISPLAY CHAR TO MENTION START AND END POINT:-**

**SELECT SUBSTRING(<’VAL’>,<START\_POSITION>,<END\_POSITION>);**

**Note:- display the data with substring function**

**SELECT SUBSTRING(<COL>,<POSITION>) FROM EMPLOYEES;**

**SELECT \* , SUBSTRING(<COL>,<POSITION>) FROM EMPLOYEES;**

**CONCAT\_WS(CONCAT WITH SEPARATOR) & SUBSTR STRING FUNCTIONS:-(Display Char from Start to end) COMBINE OPERATION:-**

**SELECT \*, SUBSTRING(CONCAT\_WS(<’-’>,<COL>,<COL>),<POSITION>) AS <ATTRIBUTE\_NAME> FROM <TB\_NAME>;**

**SELECT \*,SUBSTRING(CONCAT(<COL>,<COL>),<POSITION> AS <ATTRIBUTE\_NAME> FROM <TB\_NAMR>;**

**REPLACE STRING FUNCTION(REPLACE THE VALUE WITH ACTUAL VALUE):-**

1. **Replace The Value With Actual Value using REPLACE FUNCTION:-**

**SELECT REPLACE(<STRING>,<CURRENT\_VALUE>,<REPLACE\_VALUE>);**

**SELECT REPLACE(<COL>,<CURRENT\_VALUE>,<REPLACE\_VALUE>) FROM <TB\_NAME>**

**SELECT \*, REPLACE(<COL>,<CURRENT\_VALUE>,<REPLACE\_VALUE>) FROM <TB\_NAME>;**

**REVERSE STRING FUNCTION(REVERSE THE VALUE):-**

1. **Reverse The Value Using Reverse Function**

**SELECT REVERSE(<’VAL’>);**

**SELECT \* , REVERSE(<COL>),REVERSE(<COL>) FROM <TB\_NAME>;**

**UPPER & LOWER FUNCTION In Table Data:-**

1. **Data In Upper/Lower Case:-**

**SELECT \*, UPPER(<COL>) FROM <TB\_NAME>;**

**SELECT \*, LOWER(<COL>) FROM <TB\_NAME>;**

**SELECT \*, UCASE(<COL>) FROM <TB\_NAME>;**

**SELECT \*, LCASE(<COL>) FROM <TB\_NAME>;**

**CHAR\_LENGTH FUNCTION IN STRING FUNCTION(FOR DISPLAY LENGTH):-**

1. **DISPLAY CHAR LENGTH USING CHAR\_LENGTH FUNCTION:-**

**SELECT CHAR\_LENGTH(<’VAL’>);**

**SELECT \*, CHAR\_LENGTH(<COL>) FROM EMPLOYEES;**

**SELECT \*, LENGTH(<COL>) FROM EMPLOYEES**

**LEFT, RIGHT , REPEAT, TRIM INSERT FUNCTION:-**

1. **Insert New Value Between Existed Value:-**

**SELECT INSERT(<VAL>,<Start>,<end>,<Insert\_Value>);**

1. **Display Char from Left And Right side using Left and Right function:-**

**SELECT LEFT(<’VAL’>,<Number\_Of\_Char>);**

**SELECT RIGHT(<’VAL’>,<NUMBER\_OF\_CHAR>);**

1. **Repeat Function to repeat the Value:-**

**SELECT REPEAT(<’CHAR’>,<How\_much\_Time>);**

1. **Trim Function Use to Remove Whitespace:-**

**SELECT TRIM(<’VAL’>);**

**DISTINCT KEYWORDS**

1. **Using Distinct Display Distinct Data :-**

**Select Distinct<Col> FROM <TB\_NAME>;**

**ORDER BY KEYWORDS**

1. **Order By Use for Sorting the data order by attribute :-**

**SELECT \* FROM <TB\_NAME> ORDER BY <COL>; (Increasing Order)  
  
SELECT \* FROM <TB\_NAME> ORDER BY <COL> DESC;(Decreasing Order)**

**SELECT \* FROM <TB\_NAME> ORDER BY <COL> ,<COL> ; (Increasing Order Sorting in Col then entity value base sorting in col2 is called subsorting)**

**LIKE KEYWORDS**

1. **LIKE operator is used in a WHERE clause to search for a specified pattern in a column :-**

**SELECT \* FROM <TB\_NAME> WHERE <COL> LIKE “<%<Pattern>%>”;**

**SELECT \* FROM <TB\_NAME> WHERE <COL> LIKE “<%<PATTERN>>”;**

**SELECT \* FROM <TB\_NAME> WHERE <COL> LIKE “<PATTERN>%>”;**

**SELECT \* FROM <TB\_NAME> WHERE <COL> LIKE “\_\_\_\_\_”;**

**LIMIT KEYWORDS**

1. **Display Result Base on Limit:-**

**SELECT \* FROM <TB\_NAME> LIMIT <VAL>;**

**SELECT \* FROM <TB\_NAME> LIMIT <VAL>,<VAL>;**

**COUNT KEYWORDS**

1. **Number of Records Present in Data Count Use:-**

**SELECT COUNT(\*) FROM <TB\_NAME>;**

**GROUP BY KEYWORDS**

1. **Use for Grouping the Data in Database:-**

**SELECT <COL1> FROM <TB\_NAME> GROUP BY <COL1>;**

**SELECT <COL1>, COUNT<COL> FROM <TB\_NAME> GROUP\_BY <COL1>;**

1. **Change Attribute Position in existed Table Use Change Keyword:-**

**ALTER TABLE <TB\_NAME> CHANGE <COL> <COL> <DATATYPES> AFTER <COL>;**

**MAX AND MIX KEYWORD**

1. **MAX AND MIN Used to find Maxi and mini Value from Data Sets:-**

**SELECT MAX(<COL>) FROM<TB\_NAME>**

**SELECT MIN(<COL>) FROM <TB\_NAME>**

**SUB-QUERIES FUNCTIONS**

1. **Its Use to Multiple Queries Run At a Time:-**

**SELECT <COL> <COL> <COL> FROM <TB\_MAME> WHERE <COL> = (SELECT MAX(<COL>) FROM <TB\_NAME>);**

**SUM AND AVG FUNCTION**

1. **Sum And Avg Function Use to find Sum and Avg Vakue in data:-**

**SELECT SUM(<COL>) FROM <TB\_NAME>;**

**SELECT AVG(<COL>) FROM <TB\_NAME>;**

**DECIMAL DATA TYPES**

1. **Decimal Datatypes use for inserting decimal values in data:-**

**DECIMAL(TOTAL\_DIGIT , DIGIT\_AFTER\_DECIMAL);**

**Note:-Maximum Number of decimal digit allow including decimal value is 65**

**FLOAT AND DOUBLE DATA TYPES**

1. **Float And Double After Decimal Print Value**

**f FlOAT  
d DOUBLE**

**Note:- Float - up-to ~7 digits, takes 4 bytes of memory  
 Double - up-to ~ 15 digits, takes 8 bytes of memory**

**DATE , TIME AND DATETIME DATATYPES**

1. **Date,Time,DateTime Data Types use to insert date and time in data set:-**

**DATE :- ‘YYYY-MM-DD’ FORMAT**

**TIME :- ‘HH:MM:SS’ FORMAT**

**DATETIME :- ‘YYYY-MM-DD HH:MM:SS’ FORMAT**

**CURDATE, CURTIME, NOW IN DATE & TIME FUNCTION**

1. **CURDATE,CURTIME & NOW FUNCTION USE TO STORE CURRENT VALUE IN DATA SET:-**

**CURDATE() :- YYYY-MM-DD**

**CURTIME() :- HH:MM:SS**

**NOW() :- YYYY-MM-DD HH:MM:SS**

**DAYNAME, DAYOFMONTH, DAYOFWEEK FUNCTION IN DATE AND TIME**

1. **DAYNAME, DAYOFMONTH AND DAYOFWEEK Function is use to find month week name and week :-**

**SELECT DAYNAME(‘VAL’);**

**SELECT DAYOFMONTH(‘VAL’);**

**SELECT DAYOFWEEK(‘VAL’);**

**SELECT MONTHNAME(‘VAL’);**

**SELECT HOUR(‘VAL’);**

**SELECT MINUTE(‘VAL’);**

**SELECT SECOND(‘VAL’);**

**SELECT YEAR(‘VAL’);**

**OPERATORS:-**

1. **Relational Operators:-**

**(< ) Less than**

**(>) Greater than  
(<=) Less than or equal to  
(>=) Greater than or equal to  
(=) Equal to  
(!=) Not Equal to**

1. **Logical Operators:-**

**( And ) When Both Conditions are true**

**(OR)  When one of Both Condition True**

**Functional Keywords:-**

1. **Use of “IN” & “NOT IN”:-**

**SELECT \* FROM <TB\_NAME> WHERE <COL>IN (<’VAL1’>,<’VAL2’>);**

**SELECT \* FROM <TB\_NAME> WHERE <COL> NOT IN (<’VAL1’>,<’VAL2’>);**

1. **Between Keyword:-**

**SELECT \* FROM <TB\_NAME> WHERE <COL> BETWEEN <VAL1> AND <VAL2>;**

1. **CASE Keyword:-**

**UPDATE <TB\_NAME> SET <COL> = CASE   
WHEN <COL>  < <VAL> THEN ‘VAL’**

**END  
WHERE <COL> BETWEEN (<COL\_VAL\_START> AND <COL\_VAL\_END>**

**SELECT <COL>,<COL> CASE**

**WHEN <COL> >= <VAL> THEN ‘VAL>**

**END AS <VAL>**

**FROM <TB\_NAME>;**

1. **IS NULL & NOT LIKE Keywords:-**

**IS NULL :- Is use to check is any null value is present in attribute or not**

**SELECT \* FROM <TB\_NAME> WHERE <COL> IS NULL;**

**NOT LIKE:- use to check the sentence is not start with ‘%<char>’);**

**SELECT \* FROM <TB\_NAME> WHERE <COL> NOT LIKE <’char’>%>;**

**MATHS FUNCTIONS:-**

1. **Convert INR TO DOLLAR:-**

**SELECT <COL>,<COL> , <COL>/80 FROM <TB\_NAME>;**