DROP DATABASE name;

Page Subhash

Date

* ~~Create database~~

~~MySQL > CREATE database~~

(i) To Create new Database :-

Syntax :- CREATE DATABASE database name;

eg :- CREATE DATABASE my_db;

Guidelines :- • Name should start with an alphabate.
• Blank space and Single quotes are not allowed
• Reserve words of that RDMS/DBMS can not be used and database

Output

```
Mysql > CREATE DATABASE my_db;
Query OK, 1 row affected (0.00 sec)
Mysql -
```

(ii) USE :-

USE - This is used to tell your RDBMS/DBMS that you want to use this database.

Syntax :- USE database name;

eg :- USE my_db;

```
mysql > CREATE DATABASE my_db;
Query OK, 1 row affected (0.00 sec)

mysql > USE my_db;
Database changed
mysql>
```

III. Create Table:-

Syntax:-

```
CREATE TABLE table_name
(
  column_name1 data_type (size) [constraints],
  column_name2 data_type (size) [constraints],
  column_name3 data_type (size) [constraints]
);
```

Example:-

```
CREATE TABLE my_tab
( id int auto_increment primary key not null,
  name varchar(30), NOT NULL,
  roll number (4), PRIMARY KEY,
  address varchar (10)
);
```

Guidelines for Creation of Table.

- Table name should start with an alphabet
- In table name, blank space are single
- quotes are not allowed
- Reserve word of RDBMS/DBMS cannot be used as table name.

10.09

Page Subhash
Date

- Proper data type and size should be specified.
- Unique column name should be specified.

Output

```
mysql > USE my_db;
Database changed
mysql > CREATE TABLE My_tab
    -> (
    -> name Varchar (50) not AUTO Null,
    -> roll Int (4) PRIMARY KEY,
    -> Address Varchar (100)
    -> );
Quary ok 0 rows affected (0.51sec)
mysql>
```

Jiya-7048

IV Describe Table :- (DESC)

This is use to describe your table. DESC only describes structure of the table not the information (rows) inside table. DESC is short for of describe.

Syntax :-

```
DESC table_name;
```

example :- DESC my_tab;

output

Page Subhash
Date

Output

mysql > DESC my_tab;

Field	type	Null	key	Default	Extra
name	varchar(50)	Yes		NULL	
Roll	int(40)	YES		NULL	

2 rows in set (0.03 Sec)

mysql >

V. Show DATABASES / TABLES

- SHOW DATABASES - This command is used to view all the databases name.

Syntax:- SHOW DATABASES;

- SHOW TABLES - This Command is to view tables of current database.

Syntax:- SHOW TABLES;

Output

mysql > SHOW DATABASES;

Database
List of database which is created in your system

3 rows in Set (0.00 sec)

mysql _

```
MySQL> SHOW TABLES;
+-----------------+
| Tables_in_my_db |
+-----------------+
| my_tab          |
+-----------------+
```

VI. Insert Into:-

The INSERT INTO Statement is used to insert new records/row/tuple in a table.

- Syntax:-

INSERT INTO table_name (column1, column2, column3, column4, ...)

Example:-

```
INSERT INTO my_tab (stu_id, name, address, mobile_no)
VALUES (05, 'Anu', 'Delhi', 98112);
```

Rules:-

- Column and Value order should be same.
- Any value that goes into a VARCHAR, CHAR, DATE or TEXT Column has Single quotes around it. There are no need of quotes for numeric value. (INT, DEC)

Need Single quotes.	No need
CHAR	int or INTEGER
VARCHAR	DEC or DECIMAL
DATE	
DATE-TIME	
TIMESTAMP	
TIME	
BLOB	
TEXT	

Output

```
mysql> USE my_db;
Database changed
mysql> SHOW TABLES;
```

Tables_in_my_db
emp
my_tab
student

3 rows in set (0.01 sec)

```
mysql> DESC student; my_tab;
```

Field	type	Null	Key	Default	Extra
Name	Varchar(50)	Yes		NULL	
Roll	int(40)	Yes		NULL	

2 rows in set (0.00 sec)

```
mysql- INSERT INTO my_tab (Name, Roll)
    -> Values ('Raj', 123456);
Query OK, 1 row affected (0.14 sec)
```

To show what we write in table

```
mysql> SELECT * FROM my_tab;
```

Insert INTO

Without specifying Column name.

Syntax:-

INSERT INTO table_name
VALUES (value1, 'value2', 'value3', value4);

Ex:-

INSERT INTO My_tab
VALUES (05, 'Anu', 'Delhi', 982112);

Rules:-

- The values order should be same as column.
- We need to insert record for each column we can not leave and column.

Output

mysql > INSERT INTO student
-> VALUES (01, 'Sona', 'Delhi', 652345);
Query ok, 1 row affected (0.12 sec)

Mysql > SELECT * FROM student;

Stu_id	name	city	Pin
1	Sona	Delhi	652345

INSERT INTO

* Changing the order of column:

Syntax:-

INSERT INTO table_name (column2, column1, column4, column3)
VALUES ('value2', value1, value4, 'value3');

Ex:-

INSERT INTO my_tab (name, stu_id, ~~mobile_no~~ pin, address)
VALUES ('Anu', 05, 982112, 'Delhi');

Output

mysql > INSERT INTO student (name, stu_id, Pin, city)
-> VALUES ('Anu', 2, 879845, 'Kolkata');

Query OK, 1 row affected (0.12 sec)

mysql > SELECT * FROM student;

Stu_id	name	city	Pin
1	Sona	Delhi	652345
2	Anu	Kolkata	879845

2 rows in Set (0.00 sec)

Page ____ Subhash
Date ____

* Insert Data only in specified Columns

Syntax:-

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

Ex:-

```
INSERT INTO my_tab (stu-id, name, address)
VALUES (05, 'Anu', 'Delhi');
```

Output

```
mysql> INSERT INTO student(stu_id, name, Pin)
    -> VALUES (4, 'Sonu', 764839);
Query OK, 1 row affected (0.10 sec)

mysql> SELECT * FROM student;
```

stu_id	name	city	Pin
1	Sona	Delhi	123956
2	Anu	Kolkata	374845
3	Sonu	null	764839

3 rows in Set (0.00 sec)

Insert multiple record at one time.

Syntax:-

INSERT INTO table_name
(Column1, Column2, Column3, Column4)
VALUES (Value1, 'value2', 'Value3', Value4);
(value1, 'value2', 'val3', Value4);

Example:-
INSERT INTO My_tbl
(Stu_id, Name, address, Mobile_no)
VALUES (01, 'ANU', 'Delhi', 98211)
(02, 'Rohan', Mumbai', 541425);

Output

mysql> INSERT, INTO Student (stu_id, name, city, Pin)
→ Values (5, 'Shubham', 'HYD', 542367),
→ (6, 'Amit', 'Ranchi', 123456);
Query OK, 2 rows affected (0.06sec)
Records 2 Duplicates: 0 Warning: 0

mysql > Select * From student;

↓ Data table

6 Rows inset (0.00sec)

Page ____ Subhash
Date ____

SELECT

The SELECT statement is used to select data from a database and retrieve the information.

1. Select all columns From the table
Syntax :- Select * ~~From~~ FROM table_name;
Ex :- SELECT * FROM My tab;

2. Select Particular Columns From the table
Syntax :- SELECT Column-name1, Column-name2, ...
FROM table_name;

Ex :- SELECT name, Mobile
FROM my_tab;

Output

Mysql > SELECT Stu_id, name FROM Student;

Select with limit.

Stu_id	name
1	Raj
2	Sona
3	Anu
4	Sonu
5	S Radha
6	Amit

Select * From my_tab
→ limit 5.;

6 rows in Set (0.00 sec)
My Sql >_

☆ Single quotes Problem

Syntax:-

```
INSERT INTO my_tab (t_id, C_name, address)
VALUES (142, 'K.K's Company', 'Delhi');
```

Their are two way to solve this problem

- Use backslash
  Ex :- 'K.K\'s Company'
- Use two time single quotes
  Ex :- 'K.K''s Company'

Output

```
mysql> INSERT INTO Student( stud_id , name, City, Pin )
VALUES (7, 'kk's Company', 'Delhi', 345345);
'> (error)
```

To Remove this error we give (';)

① mysql> INSERT INTO student(stu_id, name, City, Pin)
VALUES (7, 'kk\'s Company', 'Delhi', 345345);

② or VALUES (8, 'kk''s Company', 'Delhi', 345345);

Query ok, 1 row affected (0.04 sec)

Stu_id	name	city	Pin
7	Kk's Company	Delhi	345345
8	Kk's Company	Delhi	345345

* WHERE Claws:-

WHERE is use to search for specific data

Syntax:-

1) Specific data from all Column.

Syntax:-

```
Select * FROM table_name
WHERE Column_name operater 'value';
```

2. Specific data from specific Column

Syntax:-

```
SELECT Column_name FROM table_name
WHERE Column_name operater 'Value';
```

Note:- Value Can be text or numeric. If it is text then we have to put single quotes.

operater	Description
=	Equal
<> or !=	Not Equal
>	Greater then
<	less then
>=	Greater then or equal
<=	less then or equal
BETWEEN	Between an inclusive range
LIKE	Search for a Pattern.
IN	To specify multiple possible values for a Column.

NULL Value

NULL value represent missing unknown data.
NULL ≠ 0

IS NULL - This is used to select only the records with NULL value in the Column.

Syntax:-

```
SELECT Column_name FROM table_name
WHERE Column_name IS NULL;
```

Output

```
mysql > Select * from student
    -> WHERE City IS NULL;
```

IS NOT NULL :- This is use to select only the records with no NULL values in the column.

Syntax:-

```
SELECT emp_id, emp_name, City
FROM emp
WHERE City IS NOT NULL;
```

output

```
mysql > Selelt * From student
    -> WHERE City IS NOT NULL;
```

✱ AND Operator

The AND operator displays a record if both the first condition AND the second ~~Condi~~ condition are true.

Syntax:-

```
SELECT * FROM table_name
WHERE Column_name = 'Value'
AND Column = 'Value';
```

Output

```
MySql > Selet * FROM Student
      -> WHERE name = 'Sona'
      -> AND stu_id = 2;
```

Stu_id	name	city	pin
2	Sona	Delhi	652395

✱ OR Operator

The OR operator display a record if either the first condition OR the ~~Second~~ second condition is true.

Syntax:-

```
SELECT * FROM table_name
WHERE Column_name = 'Value'
OR Column = 'value';
```

✱ IN

The IN operator allows you to specify multiple values in a WHERE clause.

Syntax:-

```
SELECT * FROM table_name
WHERE Column_name IN ('value1', 'value2', );
```

Output

```
MySql > Select * from emp
      → WHERE City IN ('Delhi', 'Ranch');
```

emp_id	name	dept	city
001	Ram	IT	Delhi
002	Sajan	Account	Delhi
003	Sita	HR	Ranchi

✱ BETWEEN

The BETWEEN Operator selects values within a range. The values can be number, text, or dates.

① Between number

Syntax:-

```
SELECT * FROM table_name
where Column_name BETWEEN Value1 and vale2;
```

ex.

```
SELECT * FROM new_tab
where stu_id BETWEEN 6 AND 8
```

Select *

2. Between Text

Syntax:-

```
SELECT * FROM table_name
Where Column_name
BETWEEN 'Value1' AND 'Value2';
```

ex:-

```
SELECT * FROM new_tab
WHERE name BETWEEN 'B' AND 'J';
```

3. Between Date

Syntax

```
SELECT * FROM table-name
WHERE Column_name
BETWEEN 'YYYY/MM/dd' AND 'YYYY/MM/dd';
```

example:-

```
SELECT * FROM new_tab
WHERE Date BETWEEN '2000/01/04' and '2001/02/
```

* NOT BETWEEN

To display the data which is not in range.

Syntax:-

```
SELECT * FROM table_name
WHERE Column_name
NOT BETWEEN Value1 and Value2
```

ex

```
SELECT * FROM Product
where stu_id
NOT BETWEEN 3 AND 7;
```

Page ____ Subhash
Date ____

★ BETWEEN with IN

Syntax:-

```
SELECT * FROM table_name
WHERE (column_name BETWEEN value1 AND value2)
    AND Column-name IN (value1, value2);
```

Ex :-

```
SELECT * FROM emp
WHERE (name BETWEEN 'B' AND 'J');
AND Department IN ('HR', 'Manager')
```

★ BETWEEN with NOT IN

Ex :-

```
SELECT * FROM emp
WHERE (Salary BETWEEN 2500 AND 5000)
AND NOT dept IN ('HR', 'Manager');
```

★ LIKE

The LIKE operater is used to search for a specified Pattern in a Column.

Syntax:-

```
SELECT * FROM table_name
WHERE Column_name LIKE 'Pattern';
```

Ex:-

```
SELECT * FROM new_tab
WHERE name LIKE '%u';
```

Wild Cards

Wild cards are used to search for data within a table. These characters are used with the LIKE operator.

Wild Card	Description
%	Zero or more characters
_	one single character
[charlist]	Set and ranges of characters to match
[^charlist] or [!charlist]	matches only a character NOT specified within the brackets

→ 1 - 5

1. % - Zero or more characters

eg:-

'Geek%' - All starting with geek eg. Geek Show

'%Shows' - All ending with Shows. eg - Geeks shows

'%Sh%' - All Containing with Sh. eg - Greek Shows

2. _ :- One single character

'Show_' :- Starting with Show then any character. eg.- Shows.

'eek' :- Any character then eek eg:- geek

'G-ek' :- G then any character. then any character. then k Ex :- Geek.

Page ____ Subhash
Date ____

NOT LIKE

Syntax:-

Select * FROM table-name
WHERE Column-name NOT LIKE 'Pattern';

eg

Mysql > SELECT * FROM emp
→ WHERE emp-name NOT LIKE '%nu';

(It will show all the column and Rows expect the employ name end with nu;

ORDER BY

This is used to sort the records.

ASC - It sorts in ascending order (by default)

DESC - It sorts in descending order.

1. Sort in descending order

Syntax:-
SELECT * from table-name
ORDER BY Column-name DESC;

eg:-
Mysql > SELECT *FROM EMP
→ ORDER BY name DESC;

2. Sort in ascending order

Syntax:-

SELECT * FROM table-name
ORDERED BY Column-name ASC; (ASC is optional).

eg:.

mysql > SELECT * FROM Student
→ ORDER BY Stu-id;

NOT NULL

By default, a table's Column can hold NULL value.
The NOT NULL Constraint enforces a field to always Contain a value. This means that you cannot insert a new record, or update a record without adding a value to this field.

eg:-

```
CREATE TABLE Student
(
Name Varchar (30),
Roll integer (5),
mobile-no integer (10) NOT NULL
);
```

UNIQUE KEY (Does not Contain Same Value).

The UNIQUE Constraint uniquely identifies each record in a database table. There can be many UNIQUE Constraints per table. A Unique key Column can Contain NULL Values

eg:-

```
MySQL > CREATE TABLE U-tab
    -> (
    -> Stu-id int (5) UNIQUE KEY,
    -> Name varchar (30),
    -> roll int (5) UNIQUE KEY,
    -> City varchar (40)
       );
```

MySQL > DESC U-tab;

Field	TYPE	NULL	Key	Default	Extra
Stu-id	int(5)	Yes	UNI	NULL	
name	varchar(30)	Yes		NULL	
roll	int(5)	Yes	UNI	NULL	
city	varchar	Yes		NULL	

PRIMARY KEY

The PRIMARY KEY Constraint uniquely identifies each record in a database table. Primary keys must contain UNIQUE values. A primary key column cannot contain NULL values. Most tables should have a primary key, and each table can have only ONE primary key.

Example

```
MySQL > CREATE TABLE Student
-> (
-> Name Varchar(30)
-> Roll integer (5) NOT NULL PRIMARY KEY,
-> Mobile_no integer (10)
-> );
```

or

```
(
Name Varchar(30)
Roll integer (5) NOT NULL,
mobile_no integer (10),
PRIMARY KEY (Roll)
);
```

```
MySQL > DESC student;
```

Field	type	NULL	Key	Default	Extra
Name	varchar(30)	Yes		NULL	
Roll	int(5)	NO	PRI	NULL	
mobile	int(10)	Yes		NULL	

Page ____ Subhash
Date ____

AUTO INCREMENT

Auto increment is used to generate an unique number, when a new record is inserted into a table.

Syntax :-

```
CREATE TABLE table_name,
(
Column_name int NOT NULL AUTO_INCREMENT,
Column_name1 Varchar (50) NOT NULL,
Column_name2 Varchar (50),
PRIMARY KEY (column_name)
);
```

Syntax :- (insert)

```
INSERT INTO emp ( emp_name, city)
VALUES
('Subham', 'Delhi'),
('Ankit', 'Mumbai');
```

To start Auto increment from particular end then we use (ALTER TABLE)

eg

```
MySql > ALTER TABLE K_tab AUTO_INCREMENT=10;
```

Date

ALIASES

Aliases are used to temporarily rename a table name or a column name.

For Table :-

Syntax :- SELECT Column_name FROM table_name AS alias_name;

eg:- SELECT name FROM Student AS widyanthi;

For Column :-

syntax:- SELECT Column_name AS alias_name FROM table_name;

eg:- SELECT name AS Student_name from Student;

SELECT name Student_name FROM Student;

SELECT name AS "Student name" FROM Student;

SELECT name AS [Student name] FROM Student;

Page Subhash
Date

Arithmetic Operation

* / + -

Syntax :- SELECT Column_name, Column_name Operator Value FROM table_name;

Ex :- SELECT name., Cost, Cost+100 FROM item-tab;
SELECT name, Cost+100 AS "New Cost" From items-tab;

example

mysql > SELECT Stid, name, mark, mark+100 FROM student;

Stid	name	marks	mark+100
1	Ram	20	30
2	Shayam	30	40
3	Mahi	40	50

My SQL > SELECT Stid, name mark, mark+100 AS "newmark" FROM student;

Stid	name	marks	new marks
1	Ram	20	30
2	Shayam	30	40
3	mahi	40	50

MySQL> SELECT Stid, name, mark, mark + Stid AS "New mark" FROM Student;

Stid	name	marks	new marks
1	Ram	20	21
2	Shayam	30	32
3	Mahi	40	43

Page ____ Subhash

Date ____

SELECT DISTINCT

The SELECT DISTINCT statement is used to display only distinct (different) values.

Syntax:- SELECT DISTINCT column_name
FROM table_name;

Example:- SELECT DISTINCT name FROM student;

Hint

id Remove the duplicate value in the column.

ie: अगर एक column में एक से ज्यादा same value होगा तो भी sirf एक ही show करेगा।

Page ____ Subhash

Date ____

ALTER TABLE

This Command is used to Add / change / Modify / Drop existing structure of the table.

- ADD Column
- Enable / Disable constraints
- Change Column
- Modify Column
- Drop Column.

* ADD Column :- when a new Column is to be added to the table structure. without constraints.

Syntax :- ALTER TABLE table_name.
ADD COLUMN Column_name datatype (size.);

example

mysql > ALTER TABLE Student
-> ADD COLUMN m_no int (11);

Query ok,

mysql > Select * from student;

Stid	name	city	marks	m_no
1	AB	X	20	999
2	CD	Y	30	888
3	EF	Z	50	777

Alter TABLE for adding Multiple Columns without Constraints :-

Syntax:-

```
ALTER TABLE table_name
ADD COLUMN Column_name datatype(size),
ADD COLUMN Column_name datatype(size);
```

eg:-

```
MYSQL> ALTER TABLE my_tab
    -> ADD COLUMN City Varchar(50),
    -> ADD COLUMN stu_id integer(5);
```

Alter table for Adding Column by position without Constraints :-

ADD COLUMN by position :-

- Last (By default)
- First
- After

Syntax:- (First)

```
ALTER TABLE table_name
ADD COLUMN Column_name datatype(size) FIRST;
```

example

```
MYSQL > ALTER TABLE my_tab
    -> ADD COLUMN Roll varchar(10) FIRST;
```

Syntax:- (AFTER)

```
ALTER TABLE table_name
ADD COLUMN Column_name datatype(size) AFTER
                                    column_name;
```

Example :-

```
mysql > ALTER TABLE my_tab
   -> ADD COLUMN L_name Varchar(50) AFTER
                                    name;
```

ADD Column - When a new column is to be added to the table structure with Constraints.

Syntax:-

```
ALTER TABLE table_name
ADD COLUMN Column_name datatype(size)
                                    Constraint_name,
    ADD Constraint_name Column_nam;
```

example:-

```
mysql > ALTER TABLE my tab
-> ADD COLUMN Roll int(10) NOT NULL,
   ADD PRIMARY KEY (roll);
```

Alter TABLE For Adding Column by Position with Constraints

```
mysql > ALTER TABLE items
    -> ADD COLUMN item_no int NOT NULL
       AUTO_INCRMENT FIRST,
    -> ADD PRIMARY KEY (item_no);
```

```
MySql > SELECT * FROM items;
```

item-no	name
1	Mobile
2	Laptop
3	T.V
4	PC

```
MySql > ALTER TABLE items
    -> ADD COLUMN Seller_id int UNIDUE KEY
       AFTER item_no;
```

```
MySql > SELECT * FROM items;
```

item-no	Seller_id	name
1	NULL	Mobile
2	NULL	Laptop
3	NULL	T.V
4	NULL	PC

ALTER TABLE For Adding Constraint in Column.

* When integrity Constraints have to be included.

Syntax:-

```
ALTER TABLE table_name
ADD CONSTRAINT Constraint_name Column-name;
```

Example:-

```
mysql> ALTER TABLE items
    -> ADD CONSTRAINT UNIQUE KEY(seller_id);
```

Change Column Name and Its Data Type without Constraints:-

Change C

Change Column:- This is used to change name and data type of an existing column without constraints.

Syntax:-

```
ALTER TABLE table_name
CHANGE COLUMN old_column_name new_column_name
new_data_type (size);
```

Example:-

```
mysql> ALTER TABLE items
    -> CHANGE COLUMN name P_Name varchar(50);
Query OK, 0 rows affected (0.12 sec)
```

Change more than one Column Name and its data type without Constraints.

Change more than one column.

Syntax:-

```
ALTER TABLE table_name
CHANGE COLUMN old_column_name new_column_name
    new_data_type (size),
CHANGE COLUMN old_column_name new_column_name
    new_data_type (size);
```

Example:-

```
mysql> ALTER TABLE items
    -> CHANGE COLUMN seller_id sale_id int(8),
    -> CHANGE COLUMN P_Name product_name
                        Varchar(50);
```

Query ok, 0 rows affected (0.11 sec)

Change Column name and its Data Type with Constraint.

Change Column with Constraints:-

Syntax:-

```
ALTER TABLE table_name
CHANGE COLUMN old_column_name
    new_column_name new_data_type (size)
    constraint_name;
ADD constraint_name (column_name);
```

Example :-

```
mysql > ALTER TABLE test
  -> CHANGE COLUMN id stu_id int(10) NOT NULL,
     ADD PRIMARY KEY (stu_id);
Query OK, 0 rows affected (0.46 sec)
```

MODIFY COLUMN Data Type and Its Size with or without Constraint

MODIY COLUMN – This is used to modify size of the data type or the data type itself of an existing column without changing column name.

Syntax :-

```
ALTER TABLE table_name
MODIFY COLUMN column_name datatype(size);
```

Example :-

```
mysql > ALTER TABLE test
  -> MODIFY COLUMN name char(40);

Query OK, 0 rows affected (0.63 sec)
```

With constraint

```
mysql > ALTER TABLE test
  -> MODIFY COLUMN name char(40) NOT NULL;

Query OK, 0 rows affected (0.63 sec)
```

Page ____ Subhash
Date ____

DROP COLUMN with or without Constraint.

When a Column in a table need to delete.

Syntax:- (~~without~~

```
ALTER TABLE table_name
DROP COLUMN Column_name;
```

Example:-

```
mysql > ALTER TABLE test
    -> DROP COLUMN name;
```

Query OK, 0 rows affected (0.75sec)

When removing Constraint from a Column.

Syntax:-

```
ALTER TABLE table_name
DROP Constraint_name Column_name;
```

Example :-

```
mysql > ALTER TABLE test
  -> DROP
```

★ Delete Data of TABLE using TRUNCATE TABLE :-

When we only want to delete the data inside the table, and not the table itself.

Syntax:-

```
TRUNCATE TABLE table_name;
```

Example:-

```
MySql > TRUNCATE TABLE my-tab;
```

★ Rename Table Name using RENAME TABLE

This Command is used to rename one or more table.

Syntax:-

```
RENAME TABLE old_table_name to new_table_name;
```

Example:-

```
mysql > RENAME TABLE my_tab to your_tab;

Query OK, 0 rows affected (0.17 Sec)
```

★ ALTER DATA BASE

ALTER DATABASE enables you to change the overall characteristics of a database. These characteristics are stored in the db.opt file in the database directory. To use ALTER DATABASE, you need the ALTER privilege on the database.

ALTER SCHEMA is a synonym for ALTER-DATABASE. The database name can be omitted from the syntax, in which case the statement applies to the default database.

Syntax:- ALTER DATABASE database-name;

DROP DATABASE

The DROP DATABASE statement is used to delete a database.

Syntax:- DROP DATABASE database_name;

Example:-

mysql > DROP DATABASE db;

SHOW COLUMNS

It shows all the columns of table and their data type along with any other column specific details. It is just like DESC tablename.

Syntax:-

SHOW COLUMNS FROM table_name;

Example:-

SHOW COLUMNS FROM emp;

SHOW CREATE DATABASE

It shows Commands which you have written while creating your Database.

Syntax:-

SHOW CREATE DATABASE database_name;

Exercise:-

mysql > SHOW CREATE DATABASE my_db;

Database	Create Database
my_db	create database 'my_db'

SHOW CREATE TABLE

It shows Commands which you have written while creating your table.

Syntax :-

```
SHOW CREATE TABLE table_name;
```

example :-

```
SHOW CREATE TABLE emp;
```

★ UPDATE Records in TABLE (Part-1)

UPDATE

The UPDATE Statement is used to update existing records in a table.

Syntax :-

```
UPDATE table_name
SET Column1 = Value1, Column2 = Value2, ...
WHERE Some_Column = Some_Value;
```

Example :-

```
UPDATE emp
SET emp_name = 'Arjun', Salary = 35000
WHERE emp_id = 101;
```

NOTE :- WHERE is necessary otherwise all records will be replaced with given value.

mysql> Select * from item;

itemno	Seller_id	name
1	null	mobile
2	null	Laptop
3	null	T.V
4	null	P.C.

mysql > UPDATE student
-> SET seller_id = 2201
-> WHERE itemno = 1;

Query ok, 1 row affected (0.14 sec)

mysql > Select * from item;

item no	seller_id	name.
1	2201	mobile
2	null	Laptop
3	null	T.V
4	null	P.C

Part 2 (UPDATE with CASE)

Syntax:==

```
UPDATE table_name
SET new_Column =
 CASE
      WHEN Column name1 = Some value1 THEN newvalue1
 WHEN Column name2 = Some value2 THEN new_value2
 ELSE new_value3
END ;
```

Example :-

```
UPDATE student
SET Result =
CASE
    WHEN mark >= 300 THEN 'First'
    WHEN mark < 300 AND mark >= 250 THEN 'Second'
    WHEN mark < 250 AND mark >= 150 THEN 'Third'
    ELSE 'Fail'
END;
```

Ans

```
mysql > Select * From student;
```

Stuid	Name	Marks
21	ABC	300
22	DEF	400
25	GHI	200
26	JKL	140

```
mysql > ALTER TABLE Student
    -> ADD COLUMN result varchar(50);
Query OK, 0 rows affected.
```

mysql > Select * From Student;

Stid	name	mark	Result
21	ABC	300	null
22	DEF	400	null
25	GHI	200	null
26	JKL	190	null

mysql> UPDATE Student
-> SET result =
-> CASE
-> WHEN Mark >= 300 THEN 'FIRST'
-> WHEN Mark < 300 AND Mark >= 250 THEN 'SECOND'
-> WHEN Mark < 250 AND Mark >= 150 THEN 'THIRD'
-> ELSE 'FAIL'
-> END;

Query ok, 6 rows affected (0.10 sec)

mysql > Select * from student;

Stid	name	~~city~~	mark	result
21	ABC		300	FIRST
22	DEF		400	FIRST
25	GHI		200	THIRD
26	JKL		190	FAIL

6 rows in set (0.00 sec)

mysql> UPDATE student
SET result = 'FIRST'
WHERE stid = 26;

* DELETE Record in Table:-

DELETE

The DELETE statement is used to delete records in a table.

1. Delete a Specific Record

Syntax:-

```
DELETE FROM table_name
WHERE Some_column = Some_value;
```

Example:-

```
DELETE FROM emp
WHERE emp_name = 'Sonam' AND EMP_id = 105
```

Note:- WHERE ~~Some~~ is neccessary otherwise all record will be delete.

2. Delete All Record:-

Syntax:-

- DELETE FROM Table_nam;
- DELETE * FROM Table_name;

Example:-

```
DELETE FROM emp;
```

or

```
DELETE * FROM emp;
```

NOTE:- We Can not Undo this!

Page ____ Subhash
Date ____

★ Copy old Table to new table within same Database :-

Syntax :-

CREATE TABLE new_table LIKE old_table;
INSERT new_table SELECT * FROM old_table;

Example :-

mysql> ~~Select~~ DESC ~~Std~~ Student;

ok, queries

Field	type	Null	key	Default	extra
Stu_id	int(10)	No	Pry	null ~~Yes~~	
Name	varchar(20)	Yes		null ~~Yes~~	
marks	int	Yes		null ~~Yes~~	

mysql> CREATE TABLE Teacher like student;

Query ok, 0 row affected

mysql> DESC Teacher;

Field	type	Null	key	Default	extra
Stu_id	int(10)	NO	PRY	Null	
Name	varchar(20)	Yes		Null	
marks	int	Yes		Null	

mysql> INSERT Teacher SELECT * FROM emp;

Query ok, 14 rows affected (0.13 sec)

mysql> SELECT * From Teacher;

Stu-id	name	marks
21	ABC	300
22	DEF	400
25	GHI	200
26	JKL	140

★ Copy old Table to new Table within different Database;-

within different Database;

use the database where you want to copy the old table.

Syntax:-

CREATE TABLE new_table LIKE ~~old_db~~
old_db.old_table;

INSERT new_table SELECT * FROM
old_db.old_table;

Example :-

mysql> CREATE TABLE teacher LIKE my-db.student ;

mysql> INSERT teacher SELECT * FROM my-db.studant;

★ MIN and MAX Function in SQL :-

- MIN (column-name) - Smallest value of the selected column.
- MAX (column-name) - Largest value of the selected column.
- Sum (column-name) - The total sum of a numeric column.
- Avg (column-name) - The average value of a numeric column.
- SQRT (column-name) - The square root of a numeric column.
- ROUND (column-name, decimal) - Function is used to round a numeric field to the number of decimals specified.

Example :- (MIN AND MAX)

mysql > ~~Select~~ * ~~From~~ ~~emp~~;

Mysql > SELECT MIN (Salary) FROM emp;

MIN (Salary)
10000

1 row in set (0.00 sec.)

My sql > SELECT MAX (Salary) From emp;

MAX (Salary)
50000

My sql > SELECT MIN (Salary) AS NewSalary
FROM emp;

NewSalary
10000

Page ____ Subhash

Date ____

Example :- (SUM And AVG)

```
mysql> SELECT SUM(salary) from emp;
```

SUM (salary)
250000

```
mysql> SELECT AVG(salary) FROM emp;
```

AVG (salary)
50000

SORT Function in SQL :-

```
mysql> SELECT. Salary, SORT(salary) FROM emp;
```

Salary	SORT(Salary)
50000	223.606
10000	212.132
35000	122.474

★ DECIMAL Data Type in SQL :—

Syntax :-

① Column-name DECIMAL (T, D);

where

T = Total digits. Range 1-65

D = digits after decimal. Range - 0-30 and ~~not~~ must not be more than T.

Ex :-

Price DECIMAL (7,2)

1 8 9 9 9 . 8 1 ← D

T

in the Place of DECIMAL we can use :-

DEC ;
NUMERIC ;
FIXED ;

② Column-name DECIMAL (T) ;

it can Read as :- Column-name DECIMAL (T, 0);

③ Column-name DECIMAL ;

it can Read Default value :- T = 10

No of Digits	No of Bytes
0	0
1-2	1
3-4	2
5-6	3
7-9	4

eg:- 16999.81
5 2

Before Decimal - 5 Digit - no of bytes - 3
After ——— - 2 - ——— - 1

Total no of byte - 3+1 = 4 byte.

eg:- 1456 4345 66499 . 81
1→9 2-4
13 2

13 are not present in Decimal Digits

Before Decimal - 13 Digit ⇒ ① 9 → 4 byte
② 4 → 2 byte

After Decimal - 2 Digit ⇒ 1 byte

Total no of byte - 4+2+1 ⇒ 7 byte.

example:-

```
MYSQL> CREATE TABLE Product
    -> (
    -> id INT AUTO_INCREMENT PRIMARY KEY,
    -> Pname VARCHAR (40),
    -> Price DECIMAL (7,2) NOT NULL
    );

Query OK, 0 rows affected;
```

```
mysql> INSERT INTO Product (Pname, Price)
    -> VALUES ("Mobile", 69999.55),
    -> ("Computer", 16000.56)
    -> ("Laptop", 25000.00),
    -> ("Printer", 10000.23);
```

Query OK, 4 rows affected

```
MYSQL> Select * From Product;
```

id	Pname	Price
1	Mobile	69999.55
2	Computer	16000.56
3	Laptop	25000.00
4	Printer	10000.23

4 rows in set (0.00 sec)

Zero Fill

```
mysql> CREATE TABLE Products(
    -> (
    -> id INT AUTO_INCREMENT PRIMARY KEY,
    -> Pname VARCHAR (40),
    -> Price DECIMAL (12, 3) ZEROFILL
    );
```

Query OK, 0 rows affected

```
MYSQL> INSERT INTO items (Pname, Price)
-> VALUES ("Mobile", 69999.55),
->        ("Computer", 16000.56),
->        ("Laptop", 25000.00),
->        ("Printer", 10000.23);

Query OK 4 rows affected
```

```
MYSQL> SELECT * FROM items;
```

id	Pname	Price
1	Mobile	69999.55
2	Computer	16000.56
3	Laptop	25000.00
4	Printer	10000.23

```
MYSQL> SELECT *, ROUND(Price,1) from Product;
```

ID	Name	Price	ROUND(Price,1)
1	Mobile	69999.55	69999.6
2	Computer	16000.56	16000.6
3	Laptop	25000.00	25000.0
4	Printer	10000.23	10000.2

★ Count Function :-

- The COUNT (column_name) Function return the ~~no~~ number of value (NULL values will not be counted) of the specified column.

SELECT COUNT (column_name) FROM table_name;
(Total no of records in column)

- COUNT (*) function returns the ~~name~~ number of records in a table.

SELECT COUNT (*) FROM table_name;
(Total no of record in table)

- The COUNT (DISTINCT column_name) function returns the number of distinct values of the specified column.

SELECT COUNT (DISTINCT column_name) FROM table_name;
(Same value वाला ~~को~~ count नहीं होता है।)

* Same value का Record कितना है उसको find करने के लिए

SELECT COUNT (column_name) From table_name
→ where Record = 101;

UPPER And LOWER Function in SQL :-

- UPPER (column_name) or UCASE (column_name) - Converts the value of a field to uppercase.
- LOWER (column_name) or LCASE (column_name) - Converts the value of a field to lowercase.
- MID (column_name, Start, length) or SUBSTRING (column_name, start, length) - Function is used to extract characters from a text field.
- LENGTH (column_name) - the length of the value in a text field.

Example :-

① mysql > SELECT UPPER (emp_name) from emp;
↳ UCASE

② mysql > SELECT UPPER (emp_name) AS Name, City FROM emp;

③ mysql > SELECT LOWER (emp_name) from emp;
↳ LCASE

MID and LENGTH Function :-

Example :-

① SELECT MID(City, 1, 3) AS Short_City From emp;
↳ SUBSTRING

② SELECT City, LENGTH(city) From emp;

CONCAT Function :-

- CONCAT (Column_name1, column_nam2, ---) - It joints two column.
- REVERSE (column_name) - It reverse the order of letter in string.
- NOW() - Function returns the current system date and time.
- FORMAT (Column_name, format_type) - Function is used to format how a field is to be displayed.

CONCAT :-

① mysql > SELECT emp_name, CONCAT (city, ' ', Pin)
FROM emp;

② mysql > SELECT emp_name, CONCAT (Pin, ' , ',
(SELECT MID(City, 1, 3))) From emp;

REVERSE :-

Example :-

mysql > SELECT REVERSE(city) FROM emp;

NOW :-

Example :-

mysql > SELECT emp_name, Salary, NOW() AS DateTime FROM emp;

GROUP BY :=

इसमे दो value को आपस मे Compair करता है और जिसका value उसका एक Group बना देता है।

example :- 1

My sql > SELECT emp_name, min(Salary) FROM emp GROUP BY emp_name;

इस Command से employ के salary को compair करेगा और जिसका salary ~~same~~ कम होगा उसको display कर देगा।

Example :- 2.

My sql > SELECT CustID, Count(*) FROM order GROUP BY CustID;

इस मे सारे CustID मे उनके कीए गये order को count करता है। और उसको display कराता है।

HAVING Clause :-

HAVING को use करने के लिए GROUP BY का use करना जरूरी है। बीना group by के use करे HAVING को use नहीं कर पाएंगे।

Example :-

```
MySql> SELECT emp_name, MIN(Salary) FROM emp
       GROUP BY emp_name HAVING MIN
       (Salary) > 25000;
```

इस Command से employ की एक group बन जाएगा जीसमे की उनका Salary 25000 से ज्यादा होगा औ GROUP BY Command को fallow करेगा।

Why do we need multi Table in SQL

ex:-

Id	S-name	City	Pin	Course
1	ABC	X	xxx	MS, Paint, Linux
2	DEF	Y	xxx	Linux, P.E
3	GHI	Z	xxx	Photo build

जब कोई ऐसा Problum आता है जीसमे की एक से ज्यादा value enter करना पड़ता है तो ये Problum create करता है और Redondency increase हो जाता है। तब ईस Problum को कम करने के लीए हम उस column का Table ही अलग कर देते है जीसमे multiple value enter करना होता है और फीर उस Table को main table के कीसी एक colum से Connect कर देते है। ताकी उसको Identify करने मे HELP मील सके।

Page ____ Subhash
Date ____

To Study by own :-

① Relationship b/w table.
(i) one to one
(ii) one to many
(iii) many to one
(iv) many to many

② what is Normalization :-

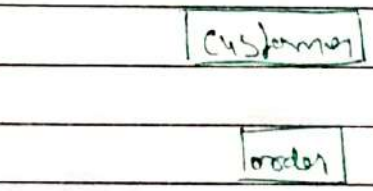
Normalization :-

Duplicate data को Remove करने के लिए Normalization का use करते हैं।

* Relationship : (B/w Table)

one to one
one to many
many to many.

Eg:



→ 1NF → • हर एक Record में एक value होगा
• Repeat group नहीं हो सकता

2 NF • Non key field depend on Primary

3 NF • Non key field not depend on Non key field.

• Must be 1NF and 2NF.

Page ____ Subhash
Date ____

FOREIGN KEY

- A FOREIGN KEY in one table Point to a PRIMARY KEY in another table.

- A foreign key can have different name then the primary key it comes from.

- The primary key used by a forgin key is also known as a Parent key. The table where the primary key is from is known as Parent key.

Syntax

```
CREATE TABLE table.table_name (Department
(
Did int NOT NULL AUTO_INCREMENT
        PRIMARY KEY,

D_Name Varchar (40),

FOREIGN KEY (E_id) REFERENCES employee
                              (E_id)
);
```

Tablename

Parent table

Example

```
MySql > CREATE TABLE employee
-> ( e.id INT NOT NULL AUTO-INCREMENT
     PRIMARY KEY,
-> ename VARCHAR (40),
-> address VARCHAR (40),
-> );
```

~~MySql~~

```
MYSql > CREATE TABLE department
   -> (
   -> did INT NOT NULL AUTO-INCREMENT
         PRIMARY KEY,
   -> dname VARCHAR (40),
   -> empid INT NOT NULL,
   -> FOREIGN KEY (empid) REFERENCES
         employee (eid)
   -> );
```

Page ____ Subhash

Date ____

How to Find CONSTRAINT Name.

MSql > Select * From INFORMATION_SCHEMA. TABLE - CONSTRAINTS

→ WHERE TABLE-NAME = 'department';

#. Drop FOREIGN KEY ~~emp~~

MySql > ALTER TABLE department

→ DROP FOREIGN KEY employee_eid-fak; (constraint name.)

AC

Page | AC | Subhash

Date

On DELETE

- ON DELETE CASCADE (जब table create करते Time. इस keyword का use करते हैं तो easily Parent table की Data को Delete कर सकते हैं।)
- ON DELETE SET NULL
- ON DELETE NO ACTION
- ON DELETE RESTRICT

1. ON DELETE CASCADE

```
CREATE TABLE department
(
did int not null auto-increment primary
                                key,
dname varchar(40),
empid int not null,
CONSTRAINT employee-eid-fk
FOREIGN KEY (empid) REFERENCES
            employee (eid)
ON DELETE CASCADE
);
```

11. ON DELETE SET NULL :— इसका use सिर्फ record को Delete करने के लिए use करते हैं। इससे Delete record me null value set हो जाएगा।

```
CREATE TABLE department
-> (
-> did int not null auto_increment primary key,
-> dname varchar (40),
-> constraint employee_eid_fk
-> foreign key (emid) references employee (eid)
-> ON DELETE SET NULL
-> );
```

```
mysql > INSERT INTO department (dname, empid)
  -> VALUES ('IT', 1),
            ('HR', 1),
            ('Admin', 2),
            ('Exe', 3);
```

```
mysql > DELETE FROM employee
        where eid = 1;
```

Page Subhash
Date

III. ON DELETE NO ACTION or
ON DELETE RESTRICT

• यह Default में होता है इसके पश्चात Pairent table के data को Delete or Update नहीं किया जा सकता।

ON UPDATE

I. ON UPDATE CASCADE

II. ON UPDATE SET NULL

III. ON UPDATE NO ACTION

IV. ON UPDATE RESTRICT

I. ON UPDATE CASCADE → अगर हम जो भी change Pairent table में करते हैं तो उसका effect child table में भी होता है।

II. ON UPDATE SET NULL :- अगर हम किसी भी Data को update करते हैं Pairent table में तो Child table में उसका value Null हो जाएगा.

iii. On update NO Action or
ON update Restrict.

```
MySql > CREATE TABLE department
    → (
    → did int not null auto-increment
                primary key,
    → dname varchar (40),
    → empid int,
    → Constraint employee_eid_FK
    → foreign key (empid) reference
                employee (eid)
    → ON UPDATE RESTRICT
    → );
```

```
mysql > Insert into INTO department
                (dname, empid)
→ value VALUES ('IT', 1),
    ('HR', 1),
    ('Admin', 1),
    ('IT', 2),
    ('Exe', 3),
    ('IT', 4),
    ('HR', 5);
```

Composite key :-

इसमे 2 Column को मिला कर एक Primary key create करते है। ~~जीसमे~~ जीससे record को uniqually Identify कर सकते है।

```
MySQL > CREATE TABLE Cource (
   → Coursecode varchar (40),
   → Date date,
   → Cname varchar (40),
   → Seat int,
   → Remain int,
   → Room int,
   → Recapa int,
   → PRIMARY KEY (Coursecode, date)
   );
```

Shorthand Notation :-

अगर 2 table मे Same column name है और अगर हमे ~~Retr~~ Data Retrive करना है तो हम Shorthand Notation का use करते है।

~~इसमे~~ इसको हम (.) की Help से call करते है।

eg :- table_name . column_name

How to Join Table :-

Table को Join करने के लिए उनके बीच Relation create करना पड़ता है।

एक Parent table होता है और एक या ज्यादा child table।

Parent table का Primary key होता है और child table में foreign key होता है।

CROSS Join / Cartesian Join / Cartesian Product / Cross product.

The Cross Join returns every Row from one table crossed with every row from the second.

```
SELECT * FROM Employee CROSS JOIN
                        Department;

SELECT * Name, Depname FROM Employee
        CROSS JOIN Department;

SELECT * FROM Employee, Department;
```

MYSQL

SPP DATE: __/__/____
PAGE: _____

INNER JOIN :-

An INNER JOIN is a CROSS JOIN with some result rows removed by a Condition in the query.

- EQUIJOIN
- NON-EQUIJOIN
- Natural JOIN

1. EQUIJOIN :-

```
SELECT Column_name FROM table1
INNER JOIN table2
ON Column_name = Column_name;
```

Syntax

```
SELECT Column_name From table1
INNER JOIN table2
ON Column_name = Column_name;
```

Good Write

DATE: __/__/____
PAGE: _____

Table - 1

```
mysql > CREATE TABLE emp (
    -> empid int auto_increment primary key,
    -> name Varchar (40),
    -> City Varchar (40)
    -> );
```

```
mysql > INSERT INTO emp (name, City)
    -> Values ('Rahul', 'Delhi'),
    -> ('krish', 'Kol'),
    -> ('Jay', 'mum');
```

Table - 2

```
MYSQL > CREATE TABLE dep (
    -> did int Primary key,
    -> depname Varchar (40),
    -> empid int,
    -> CONSTRAINT employee_id_fk
    -> FOREIGN KEY (empid) REFERENCES
                emp(empid)
    );
```

```
mysql > INSERT INTO dep (did, depname, empid)
    -> Value VALUES (101, 'IT', 3),
    -> (102, 'HR', 1),
       (103, 'Admin', 2);
```

Good Write

DATE: __/__/____
PAGE: _____

MySql > SELECT emp.name, dep.depname From emp

→ INNER JOIN dep ↑↓ 2nd run

→ ON emp.empid = dep.empid ;

3rd run

II. NON - EQUIJOIN <>

Syntax:

```
SELECT Column_name FROM table1
INNER JOIN table2
ON Column_name <> Column_name;
```

इसमें match न करने वाला का एक column आएगा

MySql > SELECT emp.name, dep.depname From dep.

→ INNER JOIN emp.

→ ON emp.empid <> dep.empid ;

Good Write

DATE: __/__/____
PAGE: _____

III. NATURAL JOIN

Syntax

SELECT Column_name FROM table1

Natural JOIN table2 ;

- इसमे Primarykey or foriegnkey दोनों का Column name Same होना चाहिए।

MySQL> SELECT empname, dep.depname FROM dep
--> NATURAL JOIN emp;

IV. Outer Join

An outer joins return all rows from one tables, along with matching information from another table.

- LEFT OUTER JOIN / LEFT JOIN
- RIGHT OUTER JOIN / RIGHT JOIN
- FULL OUTER JOIN / FULL JOIN

Good Write

DATE: __/__/____
PAGE: _____

LEFT OUTER JOIN :-

- The LEFT JOIN keyword return all rows from the left table, with the matching rows in the right table. The result is NULL in the right side when there is no match.

- In a LEFT OUTER JOIN the table that comes before the join is the left table, and the table that comes after the join is the right table.

Syntax :-

```
SELECT  Table_name1 . Column_name FROM
                    Left table ⟶ table1
LEFT JOIN  table2   ⟵ Table Right
ON Column table_name1 . column_name =
        table_name2 . column_name;
```

Example :- → SELECT emp.name, dep.depname From emp
→ LEFT JOIN dep
→ ON emp.empid = dep.empid ;

II. SELECT emp.name, dep.depname from dep
→ LEFT JOIN emp
→ ON emp.empid = dep.empid;

Good Write

DATE: __/__/____
PAGE: _____

RIGHT OUTER JOIN / RIGHT JOIN

→ The Right Join keyword return all rows from the right table, with the matching rows in the left table. The result is NULL in the left side when their is no match.

→ In a RIGHT OUTER JOIN the table that comes before the Join in the Right table, and the table that comes after the Join is the left table.

Syntax:-

```
SELECT table1.columns FROM table1 ← Right
RIGHT JOIN table2 ← left
ON table1.column = table2.column;
```

Example:-

```
mysql > SELECT emp.name, dep.depname FROM emp
→ RIGHT JOIN dep
→ ON emp.empid = dep.~~depempid~~ empid;
```

Good Write

DATE: __/__/____
PAGE: _____

Full Outer Join

- The FULL OUTER JOIN returns all rows from the left table and from the right table.
- The FULL OUTER JOIN combines the result of both LEFT and RIGHT joins.

Syntax:-

```
SELECT Column-name FROM table1
FULL JOIN table2
ON Column-name = Column-name;
```

What is self join :-

Self join is a table joined to itself.

why do we need self join

Syntax :-

```
SELECT e.name Name, m.name Manager
FROM empman e
INNER JOIN empman m
ON e.manid = m.empid;
```

Good Write