**MYSQL**

Database: A database is a collection of data and also method for accessing and manipulation that data

Database management system is the interface between database and the application that access the database.

It is a structured set of computerized data with an accessible interface.

SQL: It is the language that we use to talk to the database i.e. manipulating the data.

MYSQL : it is the database management system ,that is used to interact with the database.

The difference between all the sql database management systems may be the security and other functionalities.

-Creating a database: syntax: CREATE DATABASE <name of the database>;

Eg create database library;

-Dropping or deleting the database : syntax : DROP DATABASE <name of the database>;

DROP DATABASE library;

-Use database : This command helps to use a specific database if there are more databases, it is used to switch between different databases;

Syntax: USE <database name>;

-SELECT database(); : This command is used in order to see which database is currently used.

TABLES : is a collection of related data held in a structured format within the database;(it is collection of data)

-Datatypes : There are number of datatypes.view the document for the number of datatypes.

-Creating a table in the database : syntax : CREATE TABLE <tablename> (

column\_name data\_type,

column\_name data\_type

);

Table name should be plurals

SHOW TABLES : displays the tables

-SHOW COLUMNS FROM <tablename> : This shows the table columns in that database;

-DESC <table name> : describe table name is also used to show the columns in the database;

-DROP TABLE <table name> : drops the tables or deletes the table;

-INSERT : insert command adds the data into the tables i.e into the database.

Syntax: INSERT INTO tablename ( column names)

VALUES( the corresponding column values);

-Select : this command helps to view the data that is loaded into the database.

Syntax; SELECT \* FROM database name

-multiline inseration : INSERT INTO tablename(column name)

VALUES (the corresponding column values)

, (next column value );

Not null : inserting not null after the declaration of the column ensures that the column is not empty. Not null inserted into the variable ensures that the value should be declared or it takes the default values.

Default : inserting default values after the declaration , helps the column to take a default value if nothing is inserted into that column.

AUTO\_INCREMENT inserting this after the declaration does the auto increment of the variable we no need have to keep the track of it.

-Select command may be more specific to get more specific data, i.e getting only name from the database eg: SELECT column name FROM table name ;

-including WHERE into the select statement helps in pulling the specific data with respect to age or name and may be used to update them.

-AS command helps to rename the column name( also known as aliasing function)

DELETE command helps to delete the specific data form the column.

DELETE FROM table name WHERE column name = to be deleted;

DELETE FROM tablename; deletes whole table

Eg; delete from users where name = ‘hari’

UPDATE DATABASE : this command updates the data base with the new details .

Syntax : UPDATE tablename SET to be updated WHERE new value that has to be added.

**String functions**

CONCAT : combine the data for cleaner output, concatenates multiple columns

Syntax: SELECT

CONCAT (column , anothercolumn name);

FROM tablename;

CONCAT\_WS: concat with seperator , to concatenate multiple fields with a separator

Syntax : SELECT

CONCAT\_WS (‘seperator’ , column name, another columnname)

FROM tablename;

Substring : This is used to select a part of the string .

SELECT SUBSTRING ( string to work with , from the start point , end point );

SELECT SUBSTRING( string on which to work with, value to be passed);

-if the value is small then the string count starts from first

-if the value is large then the string count starts from last

-if the value is negative the string count starts form last

SUBSTR is a short cut for SUBSTRING , we can replace with SUBSTR.

-Replace : replace function helps to replace the exiting string with new thing that it to be replaced with.

Syntax: SELECT REPLACE( ‘ the string’ , ‘word to be replaced’ , ‘ to be replaced with’);

Eg ; select replace(‘hello world’ , ‘world ‘ , ‘there’);

-Reverse : this function reverses the string

BASIC syntax: SELECT REVERSE(the string to be reversed);

-CHAR\_LENGTH: this function says how many characters are there in the string.

-UPPER() and LOWER() : this command is used to change the case of the characters;

Syntax: UPPER( string to be changed);

LOWER(string to be changed);

Refining the selections:

DISTINCT : this command helps to reduce the duplicates present in the database, and provide only the unique that are different from each other.

Syntax: SELECT DISTINCT string to be selected.

Select distinct ‘name’ ‘age’ from users.

ODER BY : this function is used to order the data in the database or does gives sorted values.

Syntax: SELECT column name FROM table name ORDER BY column name.

By including DESC at the last in the above syntax it sorts in descending order

By default it is ascending but we can use ASC to order it in ascending order.

The syntax : eg : SELECT column name , column name FROM tablename ORDER BY 1

Here 1 indicated the first column name .

Eg: SELECT column name1 , column name2 FROM table name ORDER BY columnname ,column name

First the table is sorted by columnname1 and then it is sorted by column name2

LIMIT: limit helps to limit the number of outputs to be displayed.

Syntax: SELECT column name FROM table name LIMIT value;

LIKE: it allows to do better searching from the database.

Syntax: SELECT column name FROM tablename WHERE columnname LIKE ‘% to be searched%’;

% these are known as wild cards , the element to be searched is written between these braces. If we neglect the % then the element to be searched should be exactly the same as the element written inside the quotes.

There is another wild card i.e ‘\_’ where each underscore indicates a quantity so while searching we can use number of underscore to indicate quantity rather than numbers.

There is another character known as scape character ‘\’ where scape character is used to search characters which has wild cards in the name.

Aggregate functions:

They are known as aggregate function as they work on aggregate functions.

COUNT: this command helps to count the number of items in the database .

Syntax: SELECT COUNT( column name) FROM table name.

GROUP BY : group by aggregates or summarizes the identical data into a single row. When the command is executed it creates a virtual column to specific elements in the backgound.

Syntax: SELECT columnname FROM table name GROUP BY to be grouped with.

MIN and MAX: these command provides the min and max values of any element.

Syntax: SELECT MIN/MAX(column name) FROM table name;

SUBQuery : this means it allows one query tor run inside another query. In every situation subquery will run first and then the actual query.

Eg : SELECT title FROM books WHERE pages =(SELECT MAX(pages) FROM books);

The underlined is the subquery.

SUM : this command is used to find the sum of elements in a database.

Syntax : SELECT SUM(column name) FROM table name .

We can also use with group by .

AVG: this is used to find the average of the elements present in the database.

Syntax; SELECT AVG(column name) FROM table name.

We can also use this with group by

DATA types :

VCHAR and CHAR : the difference between char and vchar is that char has fixed length , the longest char is of 255 length long. If the data is more than the specified fixed length then it truncates the values or if it is less then it adds blank spaces.

VCHAR stores the data dynamically .

Decimal: syntax : DECIMAL(M,N);

Here M is the maximum number that is declared including the number after the decimal point.

And N is the no of digits that can be declared after the decimal point.

The decimal is of fixed type and the calculations are accurate.

Floating point and double : they are floating pointing point type and their calculation are approximate. Double is gives more storage and more numbers can be added into it.

The floating point number stores a different number when the number initialized is more than 7 digits. And double stores a different number when the number is more than 15 digits. The float will be accurate till 7 digits and double will be accurate till 15 digits.

Dates and time:

Date: this type stores only date and not the time and the format it stores is YYYY-MM-DD

-Time: this type stores the time and not the date HH:MM:SS

-Date and time : this stores both date and time and the format is YYYY-MM-DD HH:MM:SS

The dates and time functions

CURDATE - current date;

CURTIME – current time;

NOW- now the date and the time

The above functions can be used with insert statement.

Date and time functions

Date formats :

DAY(), DAYNAME(), DAYOFYEAR() , DAYOFWEEK (), within the brackets we have to enter the column name to get the relative functions these similar can be used to even month and as well as year.check the documentation for further more.

Syntax: SELECT dateformat() FROM table name.

Specifier: These are similar to the date formats but can be written in better way.

Eg : %d for day and %m for month and %y for year.Please check the documentation for further more.

Syntax: SELECT DATE\_FORMAT(column name, ‘specifier’) FROM table name.

DATE MATH:

-DATEDIFF: it takes two dates and subtracts them.

Syntax : SELECT DATEDIFF(date1,date2) FROM tablename

-DATE\_ADD/DATE\_SUB: this adds the corresponding elements.

Syntax: SELECT DATE\_ADD(column name, interval ) FROM tablename.

-we can use operator to add and subtract,

Syntax: SELECT column name +/- interval FROM tablename.

We can have multiple interval while adding and subtracting.

TIMESTAMP and DATETIME: the date time has large range when compared to timestamp where timestamp is from 1970 to 2038.

Logical operators: in the background it is doing Boolean true or false.

Not equal : != is opposite of equal operator.

Not like : Not like is opposite of like

Syntax : SELECT column name FROM table name WHERE column name NOT LIKE = “%not like type%’;

Greater than and less than : Greater than here in DB is ,it is used to select the column values after a certain period. In less than it is used to select the element values before a certain period.

SELECT column name FROM table name WHERE column name > />=/</<= certain period.

Logical AND/OR operator: synatax : SELECT column name FROM tablename WHERE column name = condition AND/OR column name = condition.

We can have more than two operands or column operands while doing the logical AND/OR

Logical AND = && and Logical OR= ||

We can use operators symbols rather than writing OR and AND

Between operator: it is to get the values within a certain range.

Syntax : SELECT column name FROM table name WHERE columname BETWEEN intital value AND final value;

Not between : it is neglect the values within a certain range.

Syntax :SELECT column name FROM table name WHERE columnname NOT BETWEEN initial value AND final value;

Casting: this is used to convert value from one data type to another.

Syntax : SELECT CAST(value AS datatype);

IN : this is used to select elements of specified more than one elements in the column.

Syntax : SELECT column name FROM table name WHERE column name IN ( element1, element2);

Internally it is doing or operation between the elements.

NOT IN : this does the opposite of IN , here internally it is using and operation

Syntax : SELECT column name FROM table name WHERE column name Not IN ( element1, element2);

CASE STATEMENT: case statement , we can go through the value and can return some value.

Syntax : SELECT column name

CASE

WHEN

ELSE

END AS new column name

FROM table name

Relationship and joins:

Three types of relationship

One to many relationship

Many to one relationship

Many to many relationship

Primary key: primary key is used to identify every entry in the database or table separately.it is unique identifier.

CREATE TABLE table name( name of the variable

PRIMARY KEY (name of the variable that should be a primary key);

Foreign key: foreign key is used to reference one table to other.

Syntax : FOREIGN KEY (current column name) REFERENCES other tablename(corresponding column name);

JOIN:

CROSS JOIN : cross join connects one table with other table i.e it connects each row of first column with every element of other column and it does similar to multiplication, for eg : if there are 5 rows in one table and 5 rows in other table , we will get 25 rows when we use cross join .

SELECT \* FROM firstcolumn name , second column name;

INNER JOIN (implicit) :this join is used you connect the two table by comparing their primary key with the foreign key. It is the intersection of the multiple tables

Implicit join

Syntax: SELECT \* FROM first colunname, second column name WHERE column name.id of that column = secondcolumn.foreign key;

Explicit join

Syntax: SELECT \* FROM first column name

JOIN second columnname

ON join condition.

LEFT JOIN: left join is the combination of intersection of the table and with the complete left table

Syntax: SELECT \* FROM firstcolumnname

LEFT JOIN second column name

ON join condition.

Right JOIN: right is opposite of left join

Syntax : SELECT \* FROM firstcolumnname

RIGHT JOIN second column name

ON join condition

IFNULL: this function checks whether the element is null or not

IFNULL(to be compared , to be replaced);

ROUND : this command does the rounding of the decimal to the specified factors ie it may be rounded to two or three decimal places

Syntax : ROUND( to be rounded, place to be rounded to;

IF statement: this command is used to check the condition if the condition is only for two arguments

Syntax: IF(condition , true then action, false condition) aliasing if necessary.