SQL –Structured Query Language

Data - Information

Data can be anything (Name/age/Location)

Database-where we keep large volume of information.

DBMS :Sybase

Informix

Microsoft

Oracle

Relational DBMS : Connection of software/Language with and Data base to perform operations

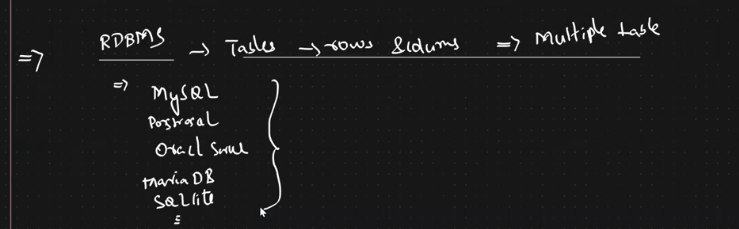
DBMS: We can store data in rows and columns.

Each row we call it as an record.

When data becomes large it became difficult store all data in the single table.

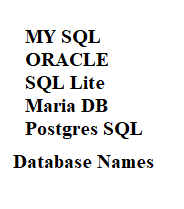
Then the concept of relational DBMS.

Store the data in multiple tables rather than single table and creating connection b/w those tables.



Relational Databases (Above all are the Database names)

SQL-Language



These are all the Database names.

NO SQL: In No SQL data will not be stored in tables. It will be stored in Key-value pair/Documents/graphs.

Mongo DB

Cassandra

For each No SQL Database there will be different language associated with it.

Usually we will use Oracle/MYSQL to integrate with Java.

SQL is case insensitive.

Creation of Database:

Create Database D1

Create Database D2

Create Database D3

From above we could see that we have 3 databases.

In order to specify in which Database we need to store the data .We will use the use command use Database Name

For Example for creating table T1 in Database D2:

Use D2;

Create table T1(……………);

Numeric -🡪int

String -🡪VARCHAR

Max Characters allowed –256

Id int(5) —Means the integer length cannot cross 5 digits.

Name VARCHAR(50) –Length of the name cannot be more than 50 characters.

Show databases;

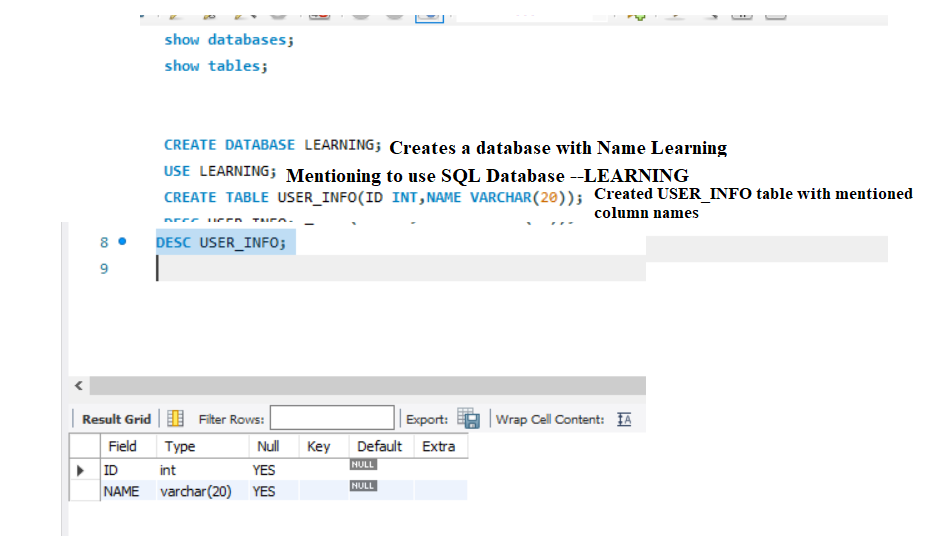
Show Tables

Desc table/Describe table.

My SQL Sever: place where the data will be stored.

Work Bench : Place that we use to perform queries.

In MYSQL both the database and Schema are same.



Creating Database

Using that Database

Creating the table in that Database

Description of that table

As per standard Pre-defined Keywords name should be in Capital(Not Mandatory).Both the lower case and Upper case alphabets will work without any issue.



DESCRIBE USER\_INFO –Returns the description of table USER\_INFO



When we have multiple databases rather than using USE DATABASENAME. we can also specify in the above way in which database we need to create the table.

As you could see from the above we are trying to create USER\_INFO\_NEW table in LEARNING Database.

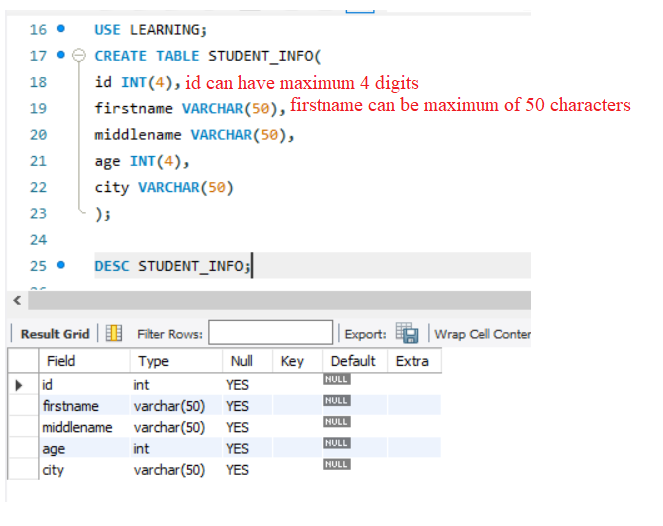
CRUD Operation: --They will come into picture only after creation of tables.

C-insert statement

R – select statement

U – update statement

D - Delete Statement.



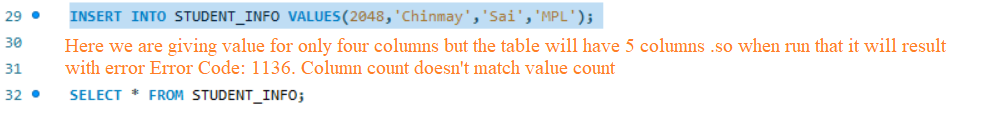


This will insert 1 row in STUDENT\_INFO with corresponding details.

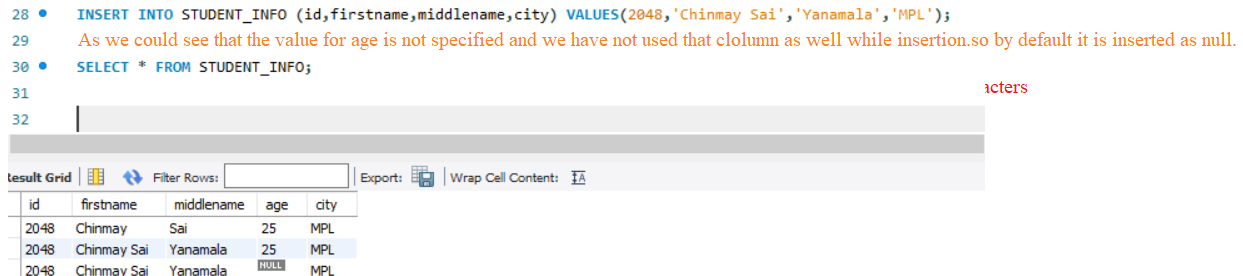
A screenshot of a computer

Description automatically generated

Print the data



To resolve that we can write the insert statement in the below format.



In real world users will be providing the values, so we will have place holders for getting these inputs.

insert into student\_info(id,firstname,middlename,age,city) values(?,?,?,?,?);

Each row will be considered as record.

CREATE TABLE EMPLOYYE\_INFO

( ID int(5) NOT NULL,--NOT NULL HERE INDICATES THAT ID COLUMN CANNOT HAVE NULL VALUE

firstname varchar(20) NOT NULL,

middlename varchar(20),

lastname varchar(20) NOT NULL,

salary int(7) NOT NULL

);

If a column has been defined as NOT NULL then we can’t insert NULL values to that column.

A screenshot of a computer

Description automatically generated

Lecture 81 : Thread session MCQ session is Pending

drop table EMPLOYYE\_INFO; -----To Drop the Table

A screenshot of a computer

Description automatically generated

Even if we have not specified the values as we have kept it as auto\_increment for id column the values are incremented automatically.

See above example screen shot .

The id column values are incremented automatically.

INSERT INTO EMPLOYYE\_INFO (firstname,middlename,lastname,salary) VALUES ("Chinmay","Sai","Yanamala",10000);

INSERT INTO EMPLOYYE\_INFO (firstname,lastname,salary) VALUES ("George","Yanamala",10000);

CREATE TABLE EMPLOYYE\_INFO

( ID int(5) auto\_increment PRIMARY KEY, -- NOT NULL HERE INDICATES THAT ID COLUMN CANNOT HAVE VALUE NULL

firstname varchar(20) NOT NULL,

middlename varchar(20),

lastname varchar(20) NOT NULL,

salary int(7) NOT NULL

);

Adding Multiple values/records at the Same time:

INSERT INTO NEW\_STUDENT (firstname,middlename,lastname,age) VALUES ("Chinmay","Sai","Yanamala",10000),("George","Sai","Yanamala",10000);

A screenshot of a computer

Description automatically generated

CRUD--CREATE READ UPDATE DELETE

A screenshot of a computer

Description automatically generated

Even though we have not specified id as NOT NULL we could see that id column NULL 🡪NO .Which means that Column will not accept NULL value.

As the id column is specified as primary Key it cannot have Duplicate values.

INSERT INTO NEW\_STUDENT (id,firstname,middlename,lastname,age) VALUES (1,"Chinmay","Sai","Yanamala",10000); ---1st insertion

INSERT INTO NEW\_STUDENT (id,firstname,middlename,lastname,age) VALUES (1,"Chinmay","Sai","Yanamala",10000); ----2nd insertion

This will lead to Error like Duplicate Entry as the id is Primary Key

Other way of Specifying the Primary Key

CREATE TABLE NEW\_STUDENT

( ID int(5) ,

firstname varchar(20) NOT NULL,

middlename varchar(20),

lastname varchar(20) NOT NULL,

age int(3) NOT NULL,

primary KEY(id)

);

UNIQUE KEY:

Values has to different.

Some databases allow one NULL values and other allows Multiple NULL values.

CREATE TABLE NEW\_STUDENT

( ID int(5) UNIQUE KEY,

firstname varchar(20) NOT NULL,

middlename varchar(20),

lastname varchar(20) NOT NULL,

age int(3) NOT NULL

);

INSERT INTO NEW\_STUDENT (id,firstname,middlename,lastname,age) VALUES (2,"George","Sai","New1",1000),(3,"Name1","Name11","Name111",100),(4,"Name2","Name22","Name222",10);

A screenshot of a computer

Description automatically generated

To retrieve Specific Columns

SELECT FIRSTNAME,MIDDLENAME,AGE FROM NEW\_STUDENT;

A screenshot of a computer

Description automatically generated

Where Clause:

SELECT FIRSTNAME,MIDDLENAME,AGE FROM NEW\_STUDENT where AGE>100;

SELECT FIRSTNAME,MIDDLENAME,AGE FROM NEW\_STUDENT where AGE>100 and FIRSTNAME='chinmay';

This will return the Result even though Chinmay is stored in Captial in DB

To Avoid that we can specify binary

SELECT FIRSTNAME,MIDDLENAME,AGE FROM NEW\_STUDENT where AGE>100 and binary FIRSTNAME='chinmay'; ---------------Here Case matching will also happen

Alias:

Other name for the Columns:

A screenshot of a computer

Description automatically generated

SELECT FIRSTNAME AS FNAME,MIDDLENAME AS MNAME FROM NEW\_STUDENT;

UPDATE:

A screenshot of a computer

Description automatically generated

UPDATE NEW\_STUDENT SET AGE=25 ;

If We don’t specify the Column name in the where all the Rows will get updated.

A screenshot of a computer

Description automatically generated

UPDATE NEW\_STUDENT SET AGE=25 ;

Updating Specific Columns:

UPDATE NEW\_STUDENT SET AGE=26 WHERE AGE>=100;

Updating Multiple Columns in the table:

UPDATE NEW\_STUDENT SET AGE=26,Middlename="New" WHERE AGE>=25 and firstname="George" ;

Alter :

Used to modify the table structure

Deleting Column from Table:

ALTER TABLE NEW\_STUDENT DROP COLUMN LASTNAME;--This Removes the Column LASTNAME from the NEW\_STUDENT table

Adding Column to the Table:

ALTER TABLE NEW\_STUDENT ADD COLUMN LASTNAME VARCHAR(20);

A screenshot of a computer

Description automatically generated

For Adding Primary key to a table:

A screenshot of a computer

Description automatically generated

Student\_info table before adding primary key.

ALTER TABLE STUDNET\_INFO ADD primary key(ID);

A screenshot of a computer

Description automatically generated

After Adding Primary Key Table Structure.

DELETE :

DELETE FROM NEW\_STUDENT WHERE ID=1;

UNIQUE KEY--

CAN HAVE NULL.

Multiple Columns can be UNIQUE key

PRIMARY KEY--

CANNOT HAVE NULL

UNIQUE

Only One Column

FOREIGN KEY:

Multiple Columns in a table can be Foreign Key(Each key will be referring different table Primary Key)

NO NULL

Foreign Key in one table will be the Primary Key of another table.

CRUD--CREATE READ UPDATE DELETE

RDBMS:

Multiple tables with relations between them

CONSTRAINTS:

PRIMARY KEY:

NOT NULL

UNIQUE

FOREIGN KEY

UNIQUE KEY

DDL:STRUCTURE OF THE TABLE

DATA DEFINITION LANGUAGES

CREATE

ALTER

Drop

DML:DATA OF THE TABLE

DATA MANUPULATION LANGUAGES

INSERT

UPDATE

DELETE

TRUNCATE:

Will remove the contents of table.

Rollback is not possible.

TRUNCATE NEW\_STUDENT..

Even if we execute the Rollback data we cannot bring the data back.

Delete:

Will remove the contents of table

We can rollback the data if auto-commit is not set to true.

DELETE FROM NEW\_STUDENT ;

ROLLBACK.

Once the ROLLBACK is Executed data will be brought back to NEW\_STUDENT.

Foreign Key:

Individual data will be divided into Multiple tables

Foreign key in One table will be Primary Key in other table.

A black board with writing on it

Description automatically generated

A blackboard with white text

Description automatically generated

In the above image we could see that student\_basic has two foreign Keys Student\_id(sid) and Student\_city(scity).

So, these Foreign Keys will be Primary Keys of the Other tables.

FOREIGN KEY:

Multiple Columns in a table can be Foreign Key(Each key will be referring different table Primary Key)

NO NULL

Foreign Key in one table will be the Primary Key of another table.

BASIC FUNCTIONS

A screenshot of a computer

Description automatically generated

SUBSTRING:

A screenshot of a computer

Description automatically generated

SUBSTRING WITH ALIAS :

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Table Data:

BETWEEN Command:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

NOT BETWEEN:

A screenshot of a computer

Description automatically generated

Order by:

A screenshot of a computer

Description automatically generated

Ordering in Descending Order

A screenshot of a computer

Description automatically generated

By Default it will be Ordered in Ascending Order.

A screenshot of a computer

Description automatically generated

JOIN:

INNER JOIN :

Common data b/w two tables.

create table course

(cid int(3) PRIMARY KEY,

CourseNa varchar(20));

CREATE TABLE NEW\_STUDENT

( ID int(5) PRIMARY KEY,

firstname varchar(20) NOT NULL,

age int(3) NOT NULL,

COURSE INT(3),

foreign key(COURSE) REFERENCES course(cid)

);

insert into course(cid,CourseNa) values (1,'Java'),(2,'SQL'),(3,'Python');

INSERT INTO NEW\_STUDENT (id,firstname,AGE,Course) VALUES (2,"George",20,1),(3,"Name1",25,3),(4,"Name2",30,1);

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

INNER JOIN :

A screenshot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated