DATABASE

Microsoft SQL Server :

1. Microsoft SQL Server – 2019 –(server)
2. Microsoft SQL SERVER MANAGEMENT STUDIO (SSMS) – 2018 (software with GUI)

MYSQL, ORACLE, MSACCESS, MSSQL : Techpile@0987654

--------------------------------------------------------------------------

DBMS : DBMS stands for database management system. It provides concept to store and manage data for future use. In DBMS data can be managed in many format like file format, table format, json format, key value format etc.

RDBMS : RDBMS stands for relational database management system. Here data is stored in form of table(Rows and columns).

Benefits to use RDBMS over file system is it is :

Easy to use

Easy to access

Easy to update , easy to search any particular data in large , Redundancy free data etc.

Many applications of RDBMS is present in market where you can store and manipulate the data , like MSACCESS, MYSQL, MSSQL, ORACLE, etc.

In RDBMS data is kept in form of Tables where tables are the collection of logically related records.

-----------------------------------------

To perform any action in any RDBMS software the language that is used is SQL(Structured query language).

**Terminology of RDBMS :**

Schema : Structure of Table

Fields : Columns

Records : Rows

Tuples : Rows

----------------------------------

SQL : SQL stands for Structured Query Language. It is a language that is used to perform any operation in RDBMS. It provides pre-defined syntax of queries that is used for transactions in Database.

So SQL queries are used to store, manipulate and retrieve any part of data in RDBMS.

-------------------------------------------------

According to the working SQL queries are divided into 4 parts :

SQL :

1. DDL(Data Definition Language)
   1. Create
   2. Alter
   3. Drop
   4. Truncate
2. DML (Data Manipulation Language)
   1. Insert
   2. Delete
   3. Update
   4. Select
3. DCL(Data Control Language)
   1. Grant
   2. Revoke
   3. rename
4. TCL(Transaction Control Language)
   1. Commit
   2. Rollback
   3. Savepoint

RDBMS :

Database : Database is the collection of Tables where tables are the collection of rows and columns. Table contains logically related records.

Database

…>

Tables

...>

Records

create database techpile\_db

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

Table = Schema(Structure of the table) + Records

DDL DML

Create : Create is a DDL command. It is used to create database, tables, views, functions, stored procedures, triggers etc.

**Syntax to create DB :**

Create database database\_name;

**Syntax to create tables :**

Create table table\_name

(

Column\_name data\_type(size),

.

.

);

Data Types : Data types defines type of value that a column may hold.

Smallint

Int : 4 byte : 2^16

Bigint

Float

Date

Datetime

Char(20) -fixed length char sequence

Varchar() – variable length char sequence

Text

Nchar()

Nvarchar()

Money

bit

ntext

-----------------------------------------------------

Create a table to store students records with following columns :

Rollno

Name

Fee

Course

Mobno

Emailed

-------------------------------------------------------------------

Difference between SQL and NoSQL

Difference between DBMS and RDBMS

What is server

What is SQL

What is Schema, Fields, Attributes, Tuples, Records, Data

Insert : Insert is a DML command. It is used to add a new row in pre-created schema.

By using insert, you can not add any value to the pre-existing rows, each time insert will add a new record in the table.

Syntax :

Insert into table\_name values(value\_1, value\_2, ………)

**Syntax to add values in specific columns :**

Insert into table\_name (column\_name\_1, column\_name\_2 ,………) values(value\_1, value\_2, …….)

**insert multiple record at once :**

insert into table\_name values(),(),();

insert into table\_name (column\_name\_1,column\_name\_2,……,column\_name\_n)

values(), (), ()

---------------------------------------------------------------

**Constraints : Constraints** are some rules to the columns. Constraints are some keywords that applies rules and regulations on the value that is inserted in the table.

Constraints defines some restrictions on the value of columns.

Constraints are applied on the columns of table.

It can add by using create command or alter command.

**Type of Constraint :**

**NOT NULL :** A not null type column does not accept NULL values. Each time you have to pass a value in the Not NULL type column.

Create table student

(

Name varchar(50) not null

);

**Default :** Default constraint is used to apply a default value for column. When user left it NULL then except NULL external default value is passed to the column.

Create table student

(

Course varchar(50) default ‘B.tech(C.S)’

)

**Check :** check constraint is used to apply condition before any record insertion in column. If check constraint is applied to the column then any value will be added only if it satisfies the condition.

Column\_name data\_type(size) check (condition)

Ex:

Create table student

(

Age int check (age>=18),

Fee int check(fee>=0)

)

**Primary key : Primary** key is used to uniquely identify each record of table. A primary key type column never accept redundant value, it does not accept NULL value to.

A table may have only a single primary key type column. Each table must should have a primary key type column.

Ex:

Create table student

(

Email\_id varchar(100) primary key

)

**Unique key : A** unique key type column is also used to uniquely identify each record of table.

Unique key= Primary key + NULL value

Ex :

Create table student

(

Rollno int unique

)

**Foreign key : Foreign** key is used to create connection within two or more tables. A primary key type column of a table may perform as a foreign key in another table.

Foreign key is mostly used in joining to select records from two or more tables.

---------------------------------------------------------------------

Auto Increment Column : Auto increment column is a int type column which value is automatic inserted for each new record. In MSSQL auto increment is known as identity . So identity is property of column which inserts unique value automatic.

In MSSQL any external value for identity column is not acceptable.

Syntax :

Column\_name data\_type(size) identity(start\_value, increment\_value);

Create a table named tbl\_student with following columns rollno , name , age , course , mobileno, email\_id

Rollno should be identity

Name and mobileno is not null type

Default value of course is Diploma

Email\_id is a primary key type column

Age of student should be greater than 18

--------------------------

Create table tbl\_student

(

Rollno int identity(101,1),

Name varchar(50) not null,

Age int check(age>=18) not null,

Course varchar(50) default ‘Diploma’,

Mobile\_no bigint not null,

Email\_id varchar(100) primary key

)

---------------------------------------------------------------------

Select : select is a DML command, it is used to retrieve any part of data from large amount of data from database.

Select is the maximum used query of SQL.

**Syntax :**

Select column\_name(s) from table\_name;

**Select specific rows with condition :**

Select column\_name(s) from table\_name where <condition>

**Operators in Condition:**

= :

>

<

>=

<=

<>

And

Or

In

Not in

Is null

Is not null

Between

Like

select name,email,mobno from ststudent

select \* from ststudent

select \* from ststudent where course='b.tech(C.S)'

select \* from ststudent where regid>=10000

select \* from ststudent where tfee<4000

select \* from ststudent where name<>'dv'

select \* from ststudent where name<>'dv' and

course='B.Tech(c.s)'

select \* from ststudent where course='Diploma(C.S)' or course='Diploma(I.T)'

Select \* from ststudent where (course='Diploma(c.s)'

or course='Diploma(i.t)') and year='First'

select \* from ststudent where course in ('Diploma(c.s)'

,'diploma(i.t)','bca') and year in('First','Second')

select \* from ststudent where course not in('Diploma(c.s)',

'Diploma(I.t)')

select \* from ststudent where course='Diploma(C.S)'

and (regid>5000 and regid<10000)

select \* from ststudent where regid between 5000 and 10000

select \* from ststudent where uname='amanst203339'

select \* from ststudent where picture is null and fathername is not null

----------------------------------------

Create a new Database :

Create database database\_name

1. Create a new table :

Create table table\_name

(

Column\_name data\_type(size),

Column\_name data\_type(size),

.

.

.

);

1. Data Types :

Int, smallint, bigint, float, decimal , char(), varchar(), text, money, date, datetime, smalldatetime, bit, nchar(), nvarchar(), ntext

1. Constraints on columns:

Not null

Default constraint

Check constraint

Primary key

Unique

Foreign key

1. Indentity() , Auto increment
2. Insert :

Insert into table\_name values(value\_1, value\_2,……,value\_n)

Insert into table\_name (column\_name\_1,column\_name\_2,…., column\_name\_n) values(value\_1,value\_2,…….,value\_n)

Insert into table\_name values(), (), ()

1. Select :

Select column\_name(s) from table\_name

Select \* from table\_name

Operators : = , > , < , >= , <= , <> , and , or , in , not in , between, is null, is not null, like

Like : Like operator is used when we have partial information of a column. Like operator is used to select record based on the partial known value of a column.

With like some symbols are used to select records:

% , \_ , []

% : Any value with any number of characters may be present at the place of % in pattern.

\_ : only one any value may be present at the place of \_

[] : [] defines range in pattern

Q : Select all student record where course contains diploma anywhere

select \* from ststudent where course like '%diploma%'

Q:select all student record where college column contains lucknow anywhere. Lucknow sequence must should be present in column at the place of % any value may present.

select \* from ststudent where college like '%lucknow%'

Q: Select all student record where first character of name is either a or e or i.

select \* from ststudent where name like '[aei]%'

Q : Select all student record where first character of name is within a to e

select \* from ststudent where name like '[a-e]%'

Q : select all students record where third character of name is e.

select \* from ststudent where name like '\_\_e%'

Select with order by clause :

Order by is used arrange selected records based on a column in ascending or descending order.

**Syntax :**

Select column\_name(s) from table\_name where <condition>

Order by column\_name asc/desc

Example:

select \* from ststudent order by name desc

select \* from ststudent order by course , year asc

select \* from ststudent order by 4

------------------------------------------------------

Select with CASE : To select column with condition case keyword is used. Case is used to select different values in a single column based on given condition.

------------------------------------------------------

Syntax :

1. Case when <condition> then <statement> end
2. Case when <condition> then <statement>

Else <statement> end

1. Case when <condition> then <statement>

When <condition> then <statement>

.

Else <statement> end

**Example :**

select name,mobno,tfee, case when tfee=0 then 'free'

when tfee<4000 then 'less paid' else 'full paid' end as feetype from ststudent

**order by with CASE :**

select \* from ststudent order by (case when mode='online' then 1

when mode='offline' then 2 else 3 end) desc

-----------------------------------------------

Aggregate Function : Aggregate functions are some pre-defined functions of SQL that is used to combined selected data and return a single value.

When we use aggregate functions with select then only a single record will return.

Aggregate functions are used only after select command.

1. Sum()
2. Min()
3. Max()
4. Avg()
5. Count()

Sum() :sum() is used only on numeric type columns. It is used to add all selected values of particular column.

**Syntax :**

Select sum(column\_name) from table\_name

Min() : It is used to find minimum value of selected records.

Syntax :

Select min(column\_name) from table\_name where <condition>

Max() : it is used to find maximum value from all selected records.

Syntax :

Select max(column\_name) from table\_name

Avg() : this function is used only on numeric type columns. It is used calculate round of average value of selected records.

Select avg(column\_name) from table\_name

Count() : it is used to count number of selected records. Count() ignores null values.

Syntax :

Select count(column\_name) from table\_name

Select count(\*) from table\_name

select count(course) as totalcs,sum(regid),

count(\*) from ststudent

where course='diploma(c.s)'

------------------------------------------------------

select \* from ststudent where

course like '%diploma%'

select \* from ststudent where college

like '%lucknow%'

select \* from ststudent where name like '[aei]%'

select \* from ststudent order by name desc

select \* from ststudent order by course , year asc

select \* from ststudent order by 4

select \* from ststudent where name like '\_\_e%'

select distinct mode from ststudent

select name,mobno,tfee, case when tfee=0 then 'free'

when tfee<4000 then 'less paid'

else 'full paid' end as feetype from ststudent

select \* from ststudent order by (case when mode='online' then 1

when mode='offline' then 2 else 3 end) desc

select \* from ststudent where tfee=600

select min(tfee) from ststudent where course like '%diploma%'

select sum(regid) from ststudent

select count(course) as totalcs,sum(regid),

count(\*) from ststudent

where course='diploma(c.s)'

row\_number()

Group By Clause : Group by clause is used to concatenate rows that has similar values in one particular column.

Group by clause is applied on column of a tables. If we apply group by clause on a column of a table, if creates different group for each distinct value present in column.

On that group, created by group by clause any aggregate function may be used.

Syntax :

Select column(s) from table\_name where <condition> group by column\_name;

Select statement may select only those columns which is present in group by clause or any column with aggregate function.

------------------------------------------------------------------------------

Having : Having is used to apply condition on records selected by group by clause. Having always comes after group by.

Syntax : select column(s) from table\_name where <condition>

Group by column\_name having <condition>;

Ex:

select sum(amount) as Total\_Paid,count(amount) as count,userid

from Fee\_Payment group by userid

select \* from fee\_payment

select mode,sum(amount) as paid from Fee\_Payment group by mode

select date,sum(amount) from Fee\_Payment group by date

select \* from Fee\_Payment

select date,sum(amount) from Fee\_Payment group by date

having date between '2022-10-20' and '2022-10-25'

select date,sum(amount) from Fee\_Payment where mode='cash'

group by date

having sum(amount)>5000 and date

between '2022-10-20' and '2022-10-25' order by date

----------------------------------------------------------------------------

Row\_number() : row\_number() is used to provide a dynamic generated serial number to every records of table. Row\_number() function adds a column to table with a unique serial number for every records. Row\_number() is used with over keyword. That means numbering is always applied based on the descending/ascending order of values of any column of tables.

Example :

select ROW\_NUMBER() over (order by Student.name desc) as sr,

Student.name,Student.Mobno,Fee\_Payment.Amount,

Fee\_Payment.Date, Fee\_Payment.Mode from Fee\_Payment left join Student

on Fee\_Payment.Userid=Student.email\_id

select ROW\_NUMBER() over (order by sum(amount) desc) as sn,

userid,sum(amount) as total from Fee\_Payment group by userid

CTE : A Common Table Expression, also called as CTE in short form, is a temporary named result set that you can reference within a SELECT, INSERT, UPDATE, or DELETE statement. The CTE can also be used in a View.

with fee(sr,total) as

(select ROW\_NUMBER() over (order by total\_fee desc) as sr,

total\_fee as total from student group by Total\_Fee )

select \* from student where Total\_Fee in (select total from fee where

sr=2 or sr=3)

with paidfee(userid,paid) as(

select userid,sum(amount) as paid from Fee\_Payment group by userid)

select Student.name ,Student.Mobno ,Student.email\_id, Student.Total\_Fee,

case when paidfee.paid is null then 0 else paidfee.paid end as paid

,Student.Total\_Fee-paidfee.paid as remaining

from Student left join paidfee on Student.email\_id=

paidfee.userid

-------------------------------------------------------------------------------------------

Update : Update command is used to change in pre-existing records. Update command is a DML command, which can be applied only on pre-existing rows of table.

Syntax : **(Update command to update a column value of every row)**

Update table\_name set column\_name=value ;

**(Update command to change any column value of selected row):**

Update table\_name set column\_name=value where <condition>

**(Update command to update multiple column value of rows :):**

Update table\_name set column\_name=value, column\_name=value, …….,column\_name\_n=value where <condition>

update student set Mobno=7007237006 where

Mobno is null

update Student set Total\_Fee=Total\_Fee+1000

select \* from Student

update Student set Total\_Fee=(select max(Total\_Fee) from

student) where sr=2

update Student set Total\_Fee=(select max(Total\_Fee) from

Student),age=20 where sr=3

Delete : Delete command is used to remove pre-existing rows from table. By using delete command you can not remove any column of row but the whole row will be deleted.

Records deleted by delete command, does not deleted permanently it can be restore by using rollback command.

Delete without condition, deletes all existing rows of table. To delete any specific row of table, conditions are applied with delete.

Syntax : **(deletes all record)**

Delete from table\_name;

**(deletes any specific record where condition is true )**

Delete from table\_name where <condition>;

7411090119