Data Types:

1]Char: it has a fixed type of memory location

Eg: RAHUL: occupies 5 memory locations

2]VARCHAR:

The length of the memory length is variable.

Date and time:

1]yyyy-mm-dd

2]yyyy-mm-dd HH:MI:SS

3](‘1970-01-01- 00:00’UTC)

Note: Binary large data type is to store data types like picture audio vedio.

Miscellaneous Data Type

1] CLOB :

2]BLOB :

3]XML :

4]JSON – used to store objects in key value pairs

Note: while creating a table data type is mandatory to assign to column but constrains are not mandetry but they are highly recommended

Note : History of sql – IBM was 1st company to develop the DBMS which follow relational model and known as system-R

=>IBM developed a language to communicate with system –r which is called SEQUEL(Simple English query language)

=>Due to its simplicity it was very popular at that time in 1980 ANSI(American national standard institute) acquired SEQEL and after doing some modification it was converted as SQL and made a standard English language for RDBMS

Note: SQL is case in sensitive,

My SQL is a application

SQL is a language

SQL:

=> it is programming language which is used to store and managing data in database

=>it is firest commertial language introduces for EF cob relational model of database

=>using this u can add delet update create

=>MySql,Databases,Oracle etc uses SQL language

=>queries are other opertations are written in statements

5 types of SQl Statement catagories:

1]Data Defiition Language( Create ,alter, drop, truncate, rename)->we only deal with the structure of table but not data

2]Data Manupitation Language(select,insert,update,delet) //used to manipulate date

2]Data controle Language(revoke,create of revoke)

4]Transactional control lang(roll back,commit,save point)

5] data query language

ALTER IN DATA BASE CREATION:

To add=> Syntax: ALTER TABLE table\_name ADD column\_name column\_datatype

To drop => Syntax: ALTER TABLE table\_name DROP column\_name column\_datatype

DATA QUERY LANGUAGE:

=> The sql statement used to retrieve the data from the database is known as data query language

=>only one command to retrieve the data from database is

SELECT

* =>with the help of select we can result sheet
* \* used = selecting all the data from the table

@Projection: select column\_name from databe\_name.

* From=( takes the table from database) From is used to select the database and execute, we can pass table name as argument
* Select=( used to prepare the result sheet)it is used to select columns or an expresstion from the table which is under execution and is also reponsible to prepare the
* Order of execution is **from->select**

@Selection: if we want to select to select or retrieve the data selecting by rows (only the selected rows)

@Join: retrieval of data from multiple table at the same time we use join

CLAUSE : Sql statements has multiple clauses .Each clause accepts some arguments

@WHERE clause: used thwen we hav to filter some data according to the condition

Syntax: select column\_name from table\_name

Where condition;

>order of execution is **from->where->select.**

**>we can’t** write where condition for where clause

@ GROUP BY : used to group the data based on column name

@HAVING : used to filter the group by column

@ ORDER BY : Arranging the result set in ascending or desending order

TRUNCATE: used to delete entire table

DELETE : use to delete the particular row or record

DROP : to delete the data as well as the structure of data we go for drop

EXPRESSTION : statement which gives any result or output

Eg: c= a+b:

A and B =operands

+ =operator

Eg: select salary +5000 from employe\_info

@operands are of to types 1) literal =>

->number Eg:676

->character Eg:’bmb’( characters are case sensitive )

->date Eg:’2019-08-08’

2) column

ALIAS =>

to give temporary name to the table and column members.

Used to make names more readable

Syntax : select column\_name AS alias\_name FROM table\_name;

select column\_name AS alias\_name FROM AS alias\_name;

DISTINCT =>

Used to select unique values from the table.

Syntax :select distinct col1,col2…. From table\_nam;

“If more than argument is passed to distinct it will remove the combination of column duplicate”

OPERATORS:

An operatior is a reserved keyword or a character used in sql primarily

Types : arithmetic

Comparesion

Logical

special

COMPARISTION :

A=b;

A<b;

a>b;

a<=b;

a>=b;

a<>b;

a!=b;

LOGICAL OPERATORS:

AND : the AND operator allows the exiatence of multiple conditions in an SQL statement where clause

|  |  |  |
| --- | --- | --- |
| A | B | OUTPUT |
| True | True | true |
| true | false | false |

Case1: if both the conditions are false AND operator returns false without evaluating the second condition

Case2: if the first condition is true result depends on the second condition

Case3: if both the conditions are true then output is true

OR OPERATOR: If any one of the results is true then we will get the output

Syntax : select \* from Table\_name where column\_name=value OR column\_name;

CASE1 :

if the first condition is true the result of OR operator is true without evaluating the second condition

CASE2:

if the first condition is false then the result will depend on the second condition ,if it is true then the result will be true

NOT OPERATOR:

Syntax: select \* from Table\_nmae where NOT column\_name = value;

ORDER OF EXECUTION OF OPERATORS:

I] NOT

2]AND

3]OR

SPECIAL OPERATOR:

@IN = if there is to apply multiple comdition for the same column or to deal with multiple values

It is an alternate for the OR operator

Syntax: Select \* from Table\_name where Column\_name IN(values1,values2…)

@NOT IN :If we want to execute multiple not operation on the same column then we use NOTIN

Syntax: Select \* from Table\_name where Column\_nameNOT IN(values1,values2…)

@BETWEEN= it I to filter the value in the range it takes 3 arguments

Syntax: Selct \* from Table\_name where column\_name btewen lower\_limit and upper\_limit

@NOT BETWEEN = to reject the range of data from the table

Syntax: Selcte \* from Table\_name where column\_name NOT BETWEEN lower\_limit and upper\_limit

@IS = this is used to check whether any value is empty,if record is not null then it gives true

Syntax : from table\_name where column\_name is null;

@NULL = used to check aany value is null

@NOT NULL = used o check weather the column is not null

@LIKE =used to find similarity between values{ CASE SENSITIVE}

Used to perform patter matching

Syntax: select \* from table\_name where column\_name LIKE ‘ % or \_ ’;

‘ ’= it’s a pattern

Note: character are of to types =ordinary and special character[Wild Card Character(eg:\*)]

%(percentage) = used to match only the first character of the word[0-n].

\_ (underscore) = used to match the the exact number of characters[one character]

@NOT LIKE : used for excluding the given pattern

FUNCTIONS(complete case study ):

It is a named block of codes which performs specific tasks;

=>It should have 3 arguments :

1] input argument

2] Function name

3] return type

=>it has pre-defined functions: 1] Scalar function(Single Row Function )

2] Agrigate Function(Multi Row Fuction)

AGRIGATE-FUNCTION :

We will pass n-number of input and get only one output.

More than one i/p

Function()

One result

METHODS : 1] AVG() – returns average

Syntax: select avg(column\_name) from table\_name

2] COUNT() – counts no of rows

3] MAX() – returns max value of the column

4] MIN() - returns min value of the selected column

5] SUM() – returns sum of all the values of the selected columns

SCALAR-FUNCTION :

=>for every single input we get particular ouput

In1

Function()

3 results

In2

In3

=>case manupilation single row function

@ UPPER() = convering to upper case

@LOWER() = convertion to lower case

@INITCAP() = initial letter should be capital // not supported in My sql

* Character manupilation

@LENGTH() = to calculate the length

@CONCAT() = to join strings ,Syntax : cancat(arg1,arg2) from table\_name

@REVERSE() =displaying the given data in reverse order

@SUBSTR = it is used to extract one string from another string ,it takes 3 argumeents

Syntax: substr ( string, starting position,length)

String = string from we get a new string

Starting position = it is a number which specify the position in the string to extract the

Substring

Length = which specify the length of new string

@INSTR() = it is used to check weather there is any substring

Syntax: instr( string,sunstr) table\_name;

String = it is a string in which we have to find the sub string

Substr = substring to find

@REPLACE() = It is used to replace the character or a string in the given string

* + - Syntax : replace( string.substring1,substring2);

GROUP BY

* It is used to give the result in group used with aggregate functions to group the result –set

Syntax: select column\_name(s) From table\_name

where condition GROUP BY column\_name(s)

|  |  |  |
| --- | --- | --- |
| Dept no | Name | Deptid |
| 1 | Rahul | 10 |
| 2 | Sneha | 20 |
| 3 | Manu | 10 |
| 4 | Sai | 30 |
| 5 | guru | 20 |

Note :

* after the execution of group by record are grouped therefore all the clause will execute after the execution of grp ny caluse
* Order of execution FROM,WHERE, GROUP BY then SELECT
* GROUP BY function is often used with arrigate function

HAVE-IN => used to filter the output gropup by data using HAVING clause

Syntax : select column\_name from table\_name

group by column\_name HAVING condition

ORDER OF EXECUTION : from,where,group by,having and then select

Note:

Having caluse executes after the execution of group by clause therefore it checks where condition after grouping

=>It executes after grouping having clause can be used only useses a column used by group by

ORDERED BY : to arrange the result set in ascending or desending order

* By default it is ascending order
* Syntax : select column\_name from Table\_name ORDER BY column\_name ASC |DESC;
* Order of execution from, where, group by, have, select and order by
* Should always be last clause to be written
* If the group by clause is used then order by clause we can be use arrigate function
* Only the column name is used in the group by clause

SUB-QUERY

* A query inside another query . It is also known as nested query or inner query
* Sub query is also another way to fetch data from multiple table
* Syntax : query WHERE (sub query)
* Order of execution = sub query will execute first ,output of inner query is input to the outer query final result will come from outer query
* When to use sub-query in where clause

1]Sinario = if the given condition unknown value then we can use subquery, inorder to obtain the unknown value

2] sinario = if the column to b selected and the condition to be filtered and given to the table we can use sub-query

TYPES OF SUB-QUERY

* Single row Sub-query

@ single row functions use relational operators while dealing with where condition and the sub-query always gives single output(aggregate functions)

And the query will be “single row sub-query”

Syntax : select \* from Table\_name where condition operator( relational)(slect \* from table\_name)

* Multi row Sub-query

@ Multi row function will use condition mostly as IN and their will be multiple sub – queries

Syntax : select \* from Table\_name where condition (select \* from Table\_name where condition)

JOINS

* merging of more than one table horizontally is known as joins
* minimum required condition for joining table is , at least number of tables should be two
* types of joints
  + Cartesian joins OR cross joins

= The record from one table is merged with each and every record of other table

= The record in left table will be merged with each and every record in second table

= Syntax column name-list

From table-name1 CROSS JOIN table-name-1;

= formula to calculate is mxn(where m is number of rows in 1st table and n is numbers of columns in second table)

* + INNER JOINS
    - Joining to tables with the help of join column is known as inner join
    - Syntax : from Table\_one INNER JOIN table2 on condition
    - Eg condition : dept\_info.deptid = employe\_info.dept\_id;
  + OUTER JOINS
    - Left, Full, Right
    - LEFT OUTER JOIN
      * These give inner join + a unmatched record of left table. The recode do not have pair in the opposite table with respect to the joins
      * Syntax : select \* from Table1 Left outer join Table2 on condition (ANSI,MY Sql)
      * Syntax of oracle : select \* from Table1,Table2 where tables.colimn.name = Table2.coloumn.name(+)
    - RIGHT OUTER JOINS
      * These give inner join + a unmatched record of right table. The recode do not have pair in the opposite table with respect to the joins
      * Syntax : select \* from Table1 Right outer join Table2 on condition (ANSI,MY Sql)
      * ( Or) select \* from Table1 Right join Table2 on condition
      * Syntax of oracle : select \* from Table1,Table2 where tables.colimn.name(+)= Table2.coloumn.name
    - FULL OUTER JOINS
      * Full outer join gives inner join as well as all the record from both the table which do not have any pair
      * This is performed by using the keyword as LEFT OJ-UNION and RIGHT OJ
* KEY ATTRIBUTES
* Key- RDBMS

It is used to find the attributes using which we can uniquely identify the records

* All the attributes accept the key attributes are known as non key attributes
* PRIME KEY ATTRIBUTES
  + A key attribute which is chosen to be a main attribute to determine the record uniquely in the table
* NON PRIMARY KET ATTRIBUTES
  + The key other than primary key is known as non primary key attributes
* COMPOSIT KEY ATRIBUTE
  + The combination of two or more record which determines more record is known as composite key(table with more than 2 primary key)
* FOREGIN KEY ATTRIBUTE
  + Foreign key is used to establish relation between to tables and it can accept null and duplicate values
* FUNCTIONAL DEPENDENCY
  + In a functional dependency a relation is established such that an attribute is dependent on other attribute uniquely
  + Eg : Consider a relation R={x, y}

X = Determine ;( Primary key)

Y = Dependent; (Foreign key)

* TYPES OF FUNCTIONAL DEPENDENCY
  + Total functional dependency
    - All the attributes of a relation is determined by a key attribute which is known as totally functional dependency
    - Eg : R ={a, b, c, d} where a 🡪 b ,c ,d (b ,c ,d dependent on a)
  + Partial functional dependency
    - A relation is said to have partial functional dependency if it consist of composite key attributes and second is their exist a dependency where attribute is determined by another attribute which is a part of composite key
    - The table which is dependent on 2 tables which has primary key respectively
    - R = {a, b, c, d} where a & b are composite primary key , c & d(determined key) are the dependent key
  + Transitive functional dependency
    - Relation is said to be transitive if their exist an relation such that an attribute is determined by non-key attributes which intern is determined by non key attribute
    - R ={ a, b, c, d} where a🡪c; c🡪d; a🡪

* NORMALIZATION
  + Redundancy and repetition is the reason why we go for normalization
  + The side-effects that occurs while performing DMC operation is called Anomaly and to overcome this problem we go for normalization
  + The process of decomposing the table into smaller table in order to remove redundancy and anomaly (or) the process of reducing the table into normal form is known as normalization

NOTE : NORMAL FORM = A state of table without table redundancy and anomaly is knows as normal table

* TYPE OF NORMAL FORM
  + 1ST normal form
    - A table structure in which 1st normal form if they satisfy the following condition they are
      * Table should not have duplicate values
      * Every cell in the table should be single or atomic value
  + 2nd normal form
    - They are called second normal form if they follow these condition
      * It should be 1st normal form
      * The table should not have composite(partial functional dependency) key

NOTE : if the table contains partial functional dependency the attributes which are responsible are removed

* + 3rd normal form
    - They are called when the following conditions are satisfied they should following the condition
      * It should follow the 2nd nf
      * The table should have transitive functional dependency

NOTE : First , second and third normal form are based on functional , key attributes schema

* + BCNF form
  + 4th normal form
* ENTITY RELATION(ER)
  + This is used to describe the structure of data base with the help of diagram which is known as ER diagram
  + It is a blue print of data base that can later used to implement data base

COMPONENTS OF ER DIAGRAM

* + Attribute
  + Entity
  + Relationship

TO COONECT ENTITY

RELATION

ELIPS FOR ATTRIBUTES

ENTITY

Entity

RELATIONS:

* One to one(1:1)
* One to many(1:N)
* Many to many(N:N)

STUDING

STUDENT

(1:N)

COLLEGE

(N:1)

* VIEW
  + Create view🡪 VIEW `ust`.`EmpDesg`

AS

select \* from employe\_info where designation ='tester'; 🡪 copy everything and execute all query 🡪open new tab(ctrl+t)🡪 Select \* from EmpDesg🡪 later execute it

* + View are logical or virtual tables that can be created on the existing table
  + View do not occupy memory
  + Syntax : create view view\_name

AS

Select \* from tsble\_name;

NOTE: Any DML operation performed on view (table it will be reflected on the original table also) on the base driven it will be reflected

* TRANSACTION CONTROL LANGUAGE(TCL)
  + ROLL-BACK
  + commit
  + SAVE POINT
    - usually work with roll back
* Store procedure stored in data base dictionary which is called either from remote program stored program or the command line
  + To create our own functions we use store procedure
  + It is a collection of sql statements

Syntax : DELIMITTER $$

Create procedure<procedure Name>

As

Begin

<SQl Statements>;

End $$

DELIMITER;

* + Syntax to call the store procedure name “call store procedure name”
  + Create store procedure 🡪 press create 🡪 give query
  + DELIMITER $$
  + CREATE
  + /\*[DEFINER = { user | CURRENT\_USER }]\*/
  + PROCEDURE `ust`.`dept`()
  + /\*LANGUAGE SQL
  + | [NOT] DETERMINISTIC
  + | { CONTAINS SQL | NO SQL | READS SQL DATA | MODIFIES SQL DATA }
  + | SQL SECURITY { DEFINER | INVOKER }
  + | COMMENT 'string'\*/
  + BEGIN
  + select \* from dept\_info where dept\_id = 30;
  + END$$

DELIMITER ;🡪 open new tab and call the store procedure🡪 call dept;