# --CLASS - 1

--SQL - Structure Query Language

--PL/SQL - Programming language / Structure Query language

--Data

--Collection of meaningful information

--Database

--It is collection of data in any file format where we able to store data.

--for ex: excel,word,textfile etc.

--Disadvantages of database

--1.It stores less amount of data

--2.data extraction process is very slow

--3.no relationship between two file formats

--RDBMS(Database) - Relational Database Management System

--It is collection of table related information

--It stores huge amount of data, whatever the data we are going to store in RDBMS, to extract that data we have simple language called SQL.

--Their is relationship between two or more tables.

--table

--it is collection of rows and columns

--Diffrent flavours

--1.SQL Server

--2.SQL developer

--3.MySql

--4.Teradata

--5.Postgres SQL

--6.TOAD

--7.DB2 etc

--Two types of databases

--1.System Defined Database

--2.User Defined Database (Database Snapshots)

--Four System defined databses

--1.master - By default DB of SQL Server

--2.model

--3.msdb

--4.tempdb

--SQL is not case sensitive Language

--For ex: menaing of DELHI is same as delhi

--Create database

--synatx: create database User\_Defined\_DB\_Name

CREATE DATABASE JOBHUNT;

--To execute SQL statements

--Use F5 key from keyboard

--use Execute tab for top bar

--To navigate user defined database we can use below syantax

--synatx: USE User\_Defined\_DB\_Name

USE JOBHUNT

--comments

--Two types of comments

--1.single line comment (--)

--2.Multiple Line Comments (/\* statements \*/)

/\* We have Started SQL

and it will take around 20 to 25 sessions

and we cover some of PL/SQL part as well \*/

--Data Type

--1.Numeric Data Type

--2.String data type

--3.Approximate data type

--4.Date and time data type

--1.Numeric Data Type

--1.Tinyint - 1 byte

--It will store number ranging from 0 to 255.(128 64 32 16 8 4 2 1 = 255)

declare @Student\_ID tinyint

SET @Student\_ID = 255

print(@Student\_ID)

--2.smallint - 2byte

--It will store value ranging from -32768 to 32767.

declare @ID smallint

set @ID = 256

print(@ID)

--CLASS-2

--3.int - 4byte

--It will store value ranging from numbers between -2,147,483,648 and 2,147,483,647.

declare @No int

set @No = 2147483647

print(@no)

--int and integer both are same

--4.bigint - 8byte

--It will store value ranging from numbers between -9,223,372,036,854,775,808 and 9,223,372,036,854,775,807.

declare @No1 bigint

set @No1 = 21474836478

print(@no1)

--5.DECIMAL(p, s)

--An exact fixed point numnber.

--Exact numerical, precision p, scale s.

--A decimal number, that is a number that can have a decimal point in it.

--The size argument has two parts : precision and scale.

--The scale can not exceed the precision.

--Precision comes first, and a comma must separate from the scale argument.

--234567.234

Declare @deci decimal

set @deci =2345.67323456789

print(@deci)

--Floating Point Numbers

--1.float(p) - 4byte to 8byte

--Approximate numerical, Precision is greater than or equal to 1 and

--the maximum precision depends on the DBMS.

--It will store value ranging from number -1.79E + 308 to 1.79E + 308.

declare @f1 float

set @f1 = 1.234545678 --1.23455e+009

print(@f1)

declare @f2 float

set @f2 = 922720368547758071234567891234567878 --9.22337e+018

print(@f2)

--2.real - 4 byte

--Same as FLOAT type except that the DBMS defines the precision.

--Floating precision number data from -3.40E + 38 to 3.40E + 38

declare @f3 real

set @f3 = 1234545678 --1.23455e+009

print(@f3)

declare @f4 real

set @f4 = 92233720368547758071234567891234567878 --9.22337e+018

print(@f4)

--3.String or character data type

--string data alaways enclosed in single quotes('') in sql.

--1.char

--A-Z,a-z,0-9 and all the special charecters

--Static memeory allocation and it can store 8000 characters.

--Fixed width

--Student\_Name char(50) - Kiran - 5

declare @Student\_Name char(50)

set @Student\_Name = 'KIRAN'

print(@Student\_Name)

print datalength((@Student\_Name)) -- it defines datatype size

print len((@Student\_Name)) -- it will give string length

--2.varchar

--A-Z,a-z,0-9 and all the special charecters

--Dynamic memeory allocation and it can store 8000 characters.

--Student\_Name char(50) - Kiran - 5

declare @Student\_Name1 varchar(50)

set @Student\_Name1 = 'KIRAN'

print(@Student\_Name1)

print datalength((@Student\_Name1)) -- it defines datatype size

print len((@Student\_Name1)) -- it will give string length

--A BC = 3 block

--ಕ ಡ = 2 blocks

--ನ್ನ= 2 blocks

--ಡ = 2 blocks

--3.nchar

--A-Z,a-z,0-9 and all the special charecters

--Static memeory allocation and it can store 4000 characters.

--Fixed width

declare @lan nchar(50)

set @lan = 'ಕನ್ನಡ'

print(@lan)

print datalength((@lan))

print len((@lan))

--3.nvarchar

--A-Z,a-z,0-9 and all the special charecters

--Dynamic memeory allocation and it can store 4000 characters.

declare @lan1 nvarchar(50)

set @lan1 = 'K'

print(@lan1)

print datalength((@lan1))

print len((@lan1))

--4.date and time data type

--getdate() - server current date and time

SELECT GETDATE()

--1.Date

--if we want to store only date in a cloumn or variable then we can use date data type.

--format - 2023-04-16

declare @current\_date date

set @current\_date = GETDATE()

print (@current\_date)

--2.time

--if we want to store only time in a cloumn or variable then we can use time data type.

--format - 09:43:16.383 (HH:MM:SS.MIS)

declare @current\_time time

set @current\_time = GETDATE() --09:44:20.6266667

print (@current\_time)

--3.datetime

--if we want to store date as well as time in a cloumn or variable then we can use datetime data type.

--format - Apr 16 2023 9:47AM

declare @current\_DT datetime

set @current\_DT = '2023-04-18 10:48:13.033'

print (@current\_DT)

--Create table

--synatx:

--CREATE TABLE TBALE\_NAME(Col1 data\_type,col2 data\_type.....coln dataType)

--inf123

--5678

CREATE table employee(EMP\_ID int,

EMP\_NAME varchar(50),

CITY varchar(50),

Salary int,

DOJ datetime)

--how to select table

--synatx:

--SELECT \* from Table\_Name

-- \* - start will indicate all the columns from table

select \* from employee

--suppose if we want only employee name and his salaray then

select EMP\_NAME,salary from employee

--How to insert data into table

--METHOD- I

--It will allow you to insert values into a table as per sequence defined while creating table.

--we have to supply values as per table number of table columns.

--synatx:

--INSERT INTO TABLE\_NAME values (clo1,col2,clo3...coln) -- if we have n number of columns in atable

INSERT INTO employee values (1,'Suresh','Latur',4500,'2022-04-01 10:00:00.000')

INSERT INTO employee values (2,'Varun','Yavatmal',5500) --'2022-04-01 10:00:00.000'

--above statement throgh an below exception

--exception

--Column name or number of supplied values does not match table definition.

INSERT INTO employee values (2,'Varun','Yavatmal',5500,'2021-01-01 11:00:00') --'2022-04-01 10:00:00.000'

select \* from employee

INSERT INTO employee values ('','','','','')

--If we supply blank/empty value then it will store by default value for respective datatype

--int - default value is zero(0)

--varchar - default value is blank

--datetime - default value is 1900-01-01 00:00:00.000

--METHOD - II

--It will allow you to insert values as per your choice but condition is that you have to follow your own defined sequence.

--Synatx:

--INSERT INTO TABLE\_NAME (clo2,clo4,col1) values (clo2,clo4,col1)

--NOTE: whatever the column we have not included in insertion syntax for those columns by default it will insert 'NULL' value.

insert into employee (EMP\_ID,EMP\_NAME,DOJ) values (3,'Shital',GETDATE()-180)

insert into employee (EMP\_ID,EMP\_NAME,DOJ) values (3,'Shital')

--There are more columns in the INSERT statement than values specified in the VALUES clause.

--The number of values in the VALUES clause must match the number of columns specified in the INSERT statement.

insert into employee (EMP\_ID,EMP\_NAME,DOJ) values (4,'Meena','')

select \* from employee

--Multiple insert records

insert into employee values (5,'Mohan','Mumbai',5000,GETDATE()-180),

(6,'Rohan','Nagpur',6000,GETDATE()-300)

--CLASS-3

--SQL Clauses

--1.WHERE

--2.ORDER BY

--3.GROUP BY

--4.Having

--1.WHERE

--WHERE Clause is used to filter data by providing condition

--WHERE Clause is used for filtering purpose.

--WHERE Clause is used along with SQL operator.

--SQL OPerator

--Types

--1.Comparision

--2.Logical

--3.Arithamatic

--4.IN and NOT IN

--5.Between and Not Between

--6.LIKE

--1.Comparision

--It is used to compare the condition provided into where clause

-- = - equal to

-- > - greater than

-- < - less than

-- >= - greater than euql to

-- <= - less than equal to

-- <> or != - not equal to

select \* from employee where EMP\_ID =5

select \* from employee where EMP\_ID >2

select \* from employee where EMP\_ID <4

select \* from employee where EMP\_ID >=2

select \* from employee where EMP\_ID <=4

select \* from employee where EMP\_ID <>2

select \* from employee where EMP\_ID != 2

--2.Logical Operator

--It is used to compare two inputs logically based upon operation specified into where clause.

--1.AND

--2.OR

--3.NOT

--1.AND

--If both the condition are TRUE then output is also TRUE

--If any of the condition fail or both the conditions are fail then ouput is FALSE.

--It is type of multiplication operation.

--AND oprtaion

--input1 input2 output

--TRUE TRUE TRUE

--TRUE FALSE FALSE (NO OUTPUT or Balnk result)

--FALSE TRUE FALSE

--FALSE FALSE FALSE

select \* from employee where EMP\_NAME = 'Gita' and CITY ='Latur'

select \* from employee where EMP\_NAME = 'MOHAN' and CITY ='Mumbai'

--2.OR

--If both the condition are TRUE oe either of the condition is TRUE then output is also TRUE

--If both the conditions are fail then ouput is FALSE.

--It is type of Addition operation.

--AND oprtaion

--input1 input2 output

--TRUE TRUE TRUE

--TRUE FALSE TRUE

--FALSE TRUE TRUE

--FALSE FALSE FALSE (NO OUTPUT or Balnk result)

select \* from employee where EMP\_NAME = 'Gita' OR CITY ='Latur'

select \* from employee where EMP\_NAME = 'MOHAN' OR CITY ='Mumbai'

select \* from employee where EMP\_NAME = 'SURESH' OR CITY ='Mumbai'

--3.NOT

--IT will perform negation

--If provided condition is TRUE in output it will give FALSE

--It will perform opposite opeartion.

--NOT Oprtaion

--INPUT OUTPUT

--TRUE FALSE

--FALSE TRUE

select \* from employee where not EMP\_ID =3

select \* from employee where not EMP\_NAME ='Mohan'

INSERT INTO employee values (7,'Shikhar','Wardha',6500,'2020-10-01 10:30:00'),

(8,'Veena','Warangal',7500,GETDATE()-750)

--Alias

--By using alias in SQL we can specify user defined names to nay column.

--synatx:

--1.

--column\_Name as userDefinedName

--2.

--column\_Name userDefinedName

--3.

-- userDefinedName = column\_Name

--3.arithmatic oprator

--It is used to perform arithamatic oprator.

--we have multiple operator in SQL i.e +,-,\*,/ and %

select \* from employee

--salary increment

select \* ,EMP\_INCREMENT\_SAL = salary + 1000 from employee

--perday salary

select \* ,PER\_DAY\_SAL = Salary/30 from employee

--increse salary of each employee by 10% and add in its base salary

select \*,Salary\_INC = salary + Salary\*10/100,bonus = Salary \*0.1 from employee

--display even records

select \* from employee where EMP\_ID %2 =0

--display odd records

select \* from employee where EMP\_ID %2 =1

--4.IN and NOT IN

--IN :This operator allow you to navigate or point out values specified into the condition.

--NOT IN : It will perform reverse operation or opposite operation of IN operator

select \* from employee where EMP\_ID in (1,5,7)

select \* from employee where CITY in ('Sanagli','Latur','Mumbai')

--out of 1000 need to display only 201 to 220 = 20

--out of 1000 need to display apart from 201 to 220 = output = 980

--5.Between and not between

--BETWEEN: This operator will you to display values or records between range you have specified

--NOT BETWEEN : This operator will you to display values or records by skkiping the rane mentioned or it will perform opposite operation of BETWEEN.

--These opeartor will work with Logical operator

select \* from employee where EMP\_ID between 3 and 5

select \* from employee where EMP\_ID not between 3 and 5

select \* from employee where EMP\_NAME between 'a' and 'Mo'

--6.LIKE

--LIKE operator will allow you to search pattern from given string or number

--LIKE operator used with WHERE clause

--LIKE is used most often with character and we can also use integer.

--LIKE operator used with following wildcards for searching pattern

--1. % : It represents one or multiple charatcters

--2. '\_': represents one or single character/ substitute exactly for one character.

--3.[charlist]% : any single character from the lsit from starting

-- %[charlist] : any single character from the lsit from ending

-- %[charlist]% : any single character from the lsit from start and end.

--4.[^charlist] or [!charlist] : anay single character not in list

-- A% - it will display all the names which start with A

-- %A - it will display all the names which end with A

-- %A% - anaywhere inside the name.

select \* from employee where EMP\_NAME like 'M%'

select \* from employee where EMP\_NAME like '%M'

select \* from employee where EMP\_NAME like '%M%'

select \* from employee where EMP\_NAME like '%a%'

select \* from employee where EMP\_NAME like '[M-S]%'

--Display names which start with C,S,R

select \* from employee where EMP\_NAME like '[CRS]%'

--Display names which does not start with A,M,R

select \* from employee where EMP\_NAME like '[!AMR]%'

select \* from employee where EMP\_NAME like '[^AMR]%'

select \* from employee

INSERT INTO employee values (9,'Amit,patil','Chandrapur',7500,'2021-10-01 10:30:00'),

(10,'Suhas\_patil','Jalna',7500,'2021-10-01 10:30:00')

select \* from employee where EMP\_NAME like '%[,\_]%'

select \* from employee where EMP\_NAME like '%[\_]%'

select \*,EMP\_NAME EMPLOYEENAME from employee

--CLASS-4

--2.Order by

--It is used to display the content of column either in ascending or dececnding order.

--by default it will ASC always if you not mentioned anaything

--for ascending (ASC) and for Descending (DESC)

select \* from employee order by Salary -- by default it is ascending

select \* from employee order by Salary desc

select \* from employee order by EMP\_NAME

--SQL aggregate

--1.AVG()

--2.SUM()

--3.MIN()

--4.MAX()

--5.Count()

--1.AVG()

--This function is used to find avarege value of particular column from table.

select \* from employee

select AVG(Salary) as Avg\_SAL from employee

--2.sum()

--it will add the values for particular column mentioned in sum function.

select \* from employee

select sum(Salary) as Avg\_SAL from employee

--3.MIN()

--This function find minimum value from column in a table.

select min(Salary) as Avg\_SAL from employee

select min(EMP\_NAME) as Avg\_SAL from employee

--4.MAX()

--This function find Maximum value from column in a table.

select max(Salary) as Avg\_SAL from employee

select max(EMP\_NAME) as Avg\_SAL from employee

--5.count()

--This function returns number of rows from table or column in table.

--Count function always accepts only one arugument.

select COUNT(\*) AS noofrecords from employee

select COUNT(EMP\_NAME) from employee

select COUNT('sdfghjk') -- O/P = 1

select COUNT('521345678909876543213456') + COUNT('7') --O/P = 2

select COUNT('523456789') + COUNT('asdfghj') --O/P = 2

select COUNT() --The COUNT function requires 1 argument(s).

--Operand data type NULL is invalid for count operator.

select COUNT(NULL) + COUNT(123) --Operand data type NULL is invalid for count operator.

select COUNT(123) --O/P = 1

select COUNT('123abncj') --O/P = 1

select COUNT(CITY) from employee --it will not count null values

INSERT INTO employee values (11,'Suman',' ',8500,'2021-10-01 10:30:00')

select COUNT(CITY) from employee --it will not count null values

select \* from employee

--TOP

--It will allow to select TOP records from table.

--We can display interms of number sor percent.

--When we have large numer of records in atble then we can use top keyword to display limited numer of records

--synatx:

--select top number / 50% percent \* from table

select top 50 PERCENT \* from employee

select top 2 \* from employee order by Salary desc

--bottom 2 records in order

select \* from employee where EMP\_ID > (select COUNT(\*) from employee) -4

--second maximum salary

select Min(salary) as maxsal from employee --where IN (8500,7500,7500)

where Salary in (select top 3 Salary from employee order by Salary desc)

--second minumum salary

select Min(salary) as minsal from employee

where Salary in (select top 3 Salary from employee order by Salary asc)

select MAX(salary) from employee where Salary in

(select top 7 salary from employee order by Salary)

--class-5

--distinct

--It will allow you to find distinct or diffrent values from column.

--synatx:

--select distinct(column\_name) from Table\_Name

select count(distinct(Salary)) from employee

--How will avoid duplicate records from table?

--How to find distinct values from table?

--NULL values

--NULL is used to indicate absence of any data value.

--results will be unknown

--NULL = 0

--NULL <> 0

--NULL > 0

--NULL = NULL

--Null value in SQL treated as diffrent value and we need a specil function to validate NULL values in SQL.

--NULL value is used as placeholder for unknown or inapplicable value in SQL.

select \* from employee where CITY = NULL

--To test null values from table we have two SQL functions

--1. IS NULL

--2. IS NOT NULL

select \* from employee where CITY is NULL

select \* from employee where CITY is not NULL

select \* from employee where CITY is NULL and Salary is null

select \* from employee where CITY = ''

--SQL Constraints

--1.Primary key

--2.Foreign key

--3.NOT NULL key

--4.Unique key

--5.Default key

--6.check key

--1.primary key(PK)

-- NOT NULL + UNIQUE

-- PK constraint uniquely identifies each record in database table.

create table orders (OID int primary key ,Order\_Name varchar(30),PIN int)

--if you want check whether constraints are defined or structure of table on particular table then we can use

exec sp\_help orders

insert into orders values (1,'TV',411057)

--Violation of PRIMARY KEY constraint 'PK\_\_orders\_\_CB394B391A2FA025'.

--Cannot insert duplicate key in object 'dbo.orders'. The duplicate key value is (1).

insert into orders values (2,'TV',411057)

insert into orders values ('','TV',411057)

insert into orders values (NULL,'TV',411057)

--Cannot insert the value NULL into column 'OID', table 'JOBHUNT.dbo.orders';

--column does not allow nulls. INSERT fails.

select \* from orders

--2.Foreign Key(FK)

--FK can point from one table to PK of another table.

--It will ensure that all the values in columns are diffrent

--FK column will accept NULL values

create table Department (DEPT\_ID int primary key , DEPT\_NAME varchar(50))

insert into Department values (1,'Testing'),(2,'Dev'),(3,'DS'),(4,'ML')

create table Student(S\_ID int,S\_Name varchar(20),City varchar(30),

DID int foreign key references Department(DEPT\_ID))

--create table SPORTS\_ACTIVITY(SP\_ID,SPORT\_NAME, DID FK)

insert into Student values (1,'Praveen','PUNE',4),(2,'Manish','Mumbai',1)

select \* from Department

select \* from Student

insert into Student values (3,'Praveen','PUNE',5)

--The INSERT statement conflicted with the FOREIGN KEY constraint "FK\_\_Student\_\_DID\_\_2B3F6F97".

--The conflict occurred in database "JOBHUNT", table "dbo.Department", column 'DEPT\_ID'.

insert into Student values (3,'Praveen','PUNE',NULL)

insert into Student values (4,'Amit','Ranchi',NULL)

insert into Student values (5,'Sumit','INDORE',4)

insert into Student values (6,'Rohan','DELHI','')

--The INSERT statement conflicted with the FOREIGN KEY constraint "FK\_\_Student\_\_DID\_\_2B3F6F97".

--The conflict occurred in database "JOBHUNT", table "dbo.Department", column 'DEPT\_ID'.

--NOTE:

--if FK point to particulr table column and which is not defined with PK then it will through below exception

--Exception: There are no primary or candidate keys in the referenced table 'Department' that match

--the referencing column list in the foreign key 'FK\_\_Student\_\_DID\_\_276EDEB3'.

--CLASS-6

--NOT NULL constraint restricts NULL value on particular column where we have defined NOT NULL constraint.

create table NOT\_NULL\_TEST(NID int primary key , NAME varchar(20) NOT NULL, city varchar(20) NOT NULL)

insert into NOT\_NULL\_TEST values (1,'','PUNE')

insert into NOT\_NULL\_TEST (NID,NAME) values (1,'Praveen')

--Cannot insert the value NULL into column 'city', table 'JOBHUNT.dbo.NOT\_NULL\_TEST';

--column does not allow nulls. INSERT fails.

select \* from NOT\_NULL\_TEST

insert into NOT\_NULL\_TEST values ('','','')

insert into NOT\_NULL\_TEST values (NULL,NULL,NULL)

--Cannot insert the value NULL into column 'city', table 'JOBHUNT.dbo.NOT\_NULL\_TEST';

--column does not allow nulls. INSERT fails.

--4.UNIQUE

--UNIQUE constraint uniquely identifies each record in column.

--One NULL Value inserted into unique constraint defined column.

create table unique\_test(UID int primary key , UID\_NAME varchar(50) unique , city varchar(50) NOT NULL)

insert into unique\_test values (1,'amit','INDORE')

insert into unique\_test values (2,'','PUNE')

insert into unique\_test values ('','','')

--Violation of UNIQUE KEY constraint 'UQ\_\_unique\_t\_\_83C6AFCE7E148F54'.

--Cannot insert duplicate key in object 'dbo.unique\_test'. The duplicate key value is ().

insert into unique\_test values (4,NULL,'')

--Violation of UNIQUE KEY constraint 'UQ\_\_unique\_t\_\_83C6AFCE7E148F54'.

--Cannot insert duplicate key in object 'dbo.unique\_test'. The duplicate key value is (<NULL>).

insert into unique\_test values (4,NULL,'')

select \* from unique\_test

--5.check

--Check constraint is used to restrict or limit the value on particular column.

Create table GAME\_APP(GID int primary key,

GAMENAME varchar(50) unique NOT NULL check (GAMENAME != '\s'), -- PUBG

Age int check(age >=18))

insert into GAME\_APP values (1,'WAR ROOM', 17)

--The INSERT statement conflicted with the CHECK constraint "CK\_\_GAME\_APP\_\_GAMENA\_\_36B12243".

--The conflict occurred in database "JOBHUNT", table "dbo.GAME\_APP", column 'GAMENAME'.

insert into GAME\_APP values (1,'WARROOM', 17)

--The INSERT statement conflicted with the CHECK constraint "CK\_\_GAME\_APP\_\_Age\_\_3C69FB99".

--The conflict occurred in database "JOBHUNT", table "dbo.GAME\_APP", column 'Age'.

insert into GAME\_APP values (1,'WARROOM', 18)

select \* from GAME\_APP

--6.Default

--DEFAULT Constraint is used to insert default value into column.

--Whenever user forget to insert any record and that record has by default value then it will automatically insert by default value.

Create table STUDY\_CIRCLE(SID int primary key,

TOPICNAME varchar(50) unique NOT NULL check (TOPICNAME != '\s'), -- PUBG

Age int check(age >=10) default 10 ,

CITY varchar(50) default 'PUNE',

PIN int default 411057)

insert into STUDY\_CIRCLE values (1,'VEDICMATH',9,'INDORE',567890)

--The INSERT statement conflicted with the CHECK constraint "CK\_\_STUDY\_CIRCL\_\_Age\_\_412EB0B6".

--The conflict occurred in database "JOBHUNT", table "dbo.STUDY\_CIRCLE", column 'Age'.

insert into STUDY\_CIRCLE values (1,'VEDICMATH',10,'INDORE',567890)

--Method-I

insert into STUDY\_CIRCLE values (2,'BASICMATH',Default,Default,Default)

select \* from STUDY\_CIRCLE

--METHOD-II

insert into STUDY\_CIRCLE (SID,TOPICNAME) values (3,'ENGLISH')

Create table STUDY\_CIRCLE1(SID int primary key, --unique + NOT NULL

TOPICNAME varchar(50) unique not null) --=PK

--AUTO-INCRREMENT

--It is used to create unique values inside column where we define auto increment.

--While creating an unique values we need provide the diffrence in uniqueness.

--synatx: Column\_Name datatype IDENTITY(Start,Diffrence of next value)

Create table SOFTWARE\_CIRCLE(SID int identity,

COURSENAME varchar(50) unique NOT NULL check (COURSENAME != '\s'),

CITY varchar(50) default 'PUNE',

PIN int default 411057)

insert into SOFTWARE\_CIRCLE values (1,'Testing','INDORE',567890)

--An explicit value for the identity column in table 'SOFTWARE\_CIRCLE' can only be specified

--when a column list is used and IDENTITY\_INSERT is ON.

insert into SOFTWARE\_CIRCLE values ('Testing','INDORE',567890)

select \* from SOFTWARE\_CIRCLE

insert into SOFTWARE\_CIRCLE values ('DS','DELHI',111001)

Create table BANK\_ACCOUNT(ACCOUNT\_NUMBER int identity(1111788510,1),

BANK\_NAME varchar(50) unique NOT NULL,

CITY varchar(50) default 'MUMBAI',

PIN int default 400001)

ExEC SP\_HELP BANK\_ACCOUNT

select \* from BANK\_ACCOUNT

insert into BANK\_ACCOUNT values ('BOI','PUNE',411027)

--CLASS-7

select \* from BANK\_ACCOUNT

insert into BANK\_ACCOUNT values ('SBI','PUNE',411057)

insert into BANK\_ACCOUNT values (1111788511,'HDFC',Default,400001)

--Inorder to insert missing value into identity defined table column we need to set identity\_insert on

SET IDENTITY\_INSERT BANK\_ACCOUNT ON

insert into BANK\_ACCOUNT (ACCOUNT\_NUMBER,BANK\_NAME,CITY,PIN)values (1111788511,'HDFC',Default,400001)

--3.GROUP BY

--GROUP BY Clause is used to group the results which have same values.

--Group By is used along with aggregate functions to group by result set by one or more values.

--synatx:

--select <col1>,<col2>,....<coln>,aggregate\_function

--From Table\_Name

--Where col\_name = Condition

--GROUP BY <col1>,<col2>,....<coln>

select \* from employee

create table emp(eid int identity,EMP\_NAME varchar(30),Salary int, DEPT varchar(20))

insert into emp values ('Praveen',3000,'TESTING'),('Amit',4000,'DS'),('Sumit',3500,'TESTING'),

('Vinit',3800,'ML'),('Rohan',6000,'AI'),('Kisan',4500,'TESTING'),

('Kiran',7600,'DS'),('Naman',7800,'ML'),('Jay',9000,'AI'),

('Varun',8000,'ML')

select \* from emp

--count number of employees in dept

select DEPT,COUNT(\*) as EMPCOUNT from emp

group by DEPT

--sumof salary deptwise

select DEPT,SUM(Salary) DEPTWISESALARY from emp

group by DEPT

--emp and dept wise minimumn salary

select DEPT,MIN(Salary) from emp

group by DEPT

select distinct(DEPT) from emp

select DEPT,COUNT(\*) from emp

group by DEPT

select DEPT,MIN(Salary) from emp

group by DEPT

select DEPT,max(Salary) from emp

group by DEPT

select DEPT,AVG(Salary) from emp

group by DEPT

select DEPT,COUNT(\*) from emp

where count(DEPT) >2 --can not used along with aggregate function.

group by DEPT

--error :An aggregate may not appear in the WHERE clause unless it is in a subquery contained in a HAVING clause

--or a select list, and the column being aggregated is an outer reference.

--4.HAVING Clause

--Having Clause is added in SQL because we can not use WHERE clause along with aggregate functions.

--synatx:

--select <col1>,<col2>,....<coln>,aggregate\_function

--From Table\_Name

--Where col\_name = Condition

--GROUP BY <col1>,<col2>,....<coln>

--HAVING Aggreagte\_Function(column\_name) opeartor Value/ condition.

--Diffrence between WHERE clause and Having Clause?

select DEPT,COUNT(\*) as CountOfEMP from emp

group by DEPT

having COUNT(\*) > 2

--List out department whose sum of salary is greater than 15000

select DEPT,SUM(Salary) as SUMOFSAL from emp

group by DEPT

having SUM(Salary) > 15000

--Find duplicate records are present in a table

--Duplicate records are those records which is exactly same as existing records in table.

create table emp1(eid int,EMP\_NAME varchar(30),Salary int, DEPT varchar(20))

insert into emp1 values (1,'Praveen',3000,'TESTING'),(2,'Amit',4000,'DS'),(3,'Sumit',3500,'TESTING'),

(4,'Vinit',3800,'ML'),(5,'Rohan',6000,'AI')

select \* from emp1

--methods to identify distinct records

select distinct eid,EMP\_NAME,Salary,DEPT from emp1

select eid,EMP\_NAME,Salary,DEPT from emp1

group by eid,EMP\_NAME,Salary,DEPT

--identifying duplicate records present in table

select eid,EMP\_NAME,Salary,DEPT,COUNT(\*) as Duplicate from emp1

group by eid,EMP\_NAME,Salary,DEPT

having COUNT(\*) >1

--SQL Statements

--1.DML(Data Manipulation Language) - S\_UID

--SELECT ,UPDATE, INSERT DELETE

--2.DDL (Data Defination Language) --DR.CAT

--DROP,RENAME,CREATE,ALTER,TRUNCATE

--3.DCL (Data Control Language)

--GRANT,REVOKE

--4.TCL(Transaction Control Language)

-- COMMIT,ROLLBACK, TRAN, COMMITTRAN

--CLASS-8

--1.DML(Data Manipulation Language) - S\_UID

--SELECT ,UPDATE, INSERT DELETE

--DML operation are used to play with table data stored inside the table.

--SELECT

--This tatement is used to select all the records from table

--By using select we can select any string or value or any sql functions

select 88888

select \* from employee

select 5678 from employee

select COUNT(12345)

--UPDATE

--UPDATE statement is used to update existing records from tbale.

--While updating column if you have not specified condtion then it will modify/update the complete column.

--synatx:

--UPDATE Table\_Name SET ColUmn\_Name = 'Value' where column\_Name = Condition

select \* from employee

UPDATE employee SET CITY = 'DELHI' where EMP\_ID =3

UPDATE employee SET Salary = 5200 where Salary is null

UPDATE employee SET CITY = 'KOTA' where CITY is null OR CITY = ''

UPDATE employee SET CITY = 'KOTA', EMP\_NAME = 'MANISH' where EMP\_ID =10

select \* from Student

UPDATE Student set DID =2

--INSERT

--This statement is used to insert the value into table

--Synatx:

--INSERT INTO TABLE\_NAME VALUES (Col1,col2,.....coln)

--or

--INSERT INTO TABLE\_NAME (Col1,col2,.....coln) VALUES (Col1,col2,.....coln)

--DELETE

--THIS statement is used to delete table data based on condition.

--If we have not mentioned condition then it will delete complete data.

--DELETE operation will be perfomed on ROW-BY-ROW basis.

--Synatx:

--DELETE FROM TABLE\_NAME WHERE Column\_NAME = CONDITION

select \* from emp1

delete from emp1 --it will delete all the records

select \* from employee

delete from employee where EMP\_ID =11

--2.DDL (Data Defination Language) --DR.CAT

--DROP,RENAME,CREATE,ALTER,TRUNCATE

--DDL stamenets are basically used to play with table structure

--DDL statement used along with TABLE Keyword

exec sp\_help employee

--DROP

--DROP statement is used to DELETE table struture as well table data.

--DROP stement is used to delete complete user defined database

--synatx:

--DROP TABLE TBALE\_NAME

--DROP DATABASE database\_Name

--drop table

select \* from SOFTWARE\_CIRCLE

DROP table SOFTWARE\_CIRCLE

drop table employee

CREATE DATABASE TEST

--DROP DATABASE

DROP DATABASE Test

DROP TABLE EMP1

select \* from emp1

--Diffrence between DELETE and DROP

--TRUNCATE

--TRUNCATE Statement will allow you to DELETE complete table data at once.

--IT will not allow you to provide condition

--syantx:

--truncate table TABLE\_NAME

select \* from BANK\_ACCOUNT

DELETE FROM BANK\_ACCOUNT --it is slow compared to truncate because it will delete ROW-BY-ROW

Truncate Table BANK\_ACCOUNT --IT is fast because it will delete complete records within fraction of time

--Diffrence between DELETE ,DROP and Truncate

--CLASS-9

--ALTER

--ALTER statement used to play with or manipulate table structure.

--by using ALTER statement , we can perform multiple operations

--1.We can add one or more columns at a time into table.

--2.We can Drop or delete one or more columns at a time into table.

--3.We can Increase or decrease size of columns into table.

--4.We can change data type of columns.

--5.We can define constraints to column into table.

--6.We can drop constraints from column into table.

--1.We can add one or more columns at a time into table.

--synatx:

--ALTER TABLE Table\_Name ADD COLUMN\_NAME DATA\_TYPE

CREATE table emp(EMP\_ID int,

EMP\_NAME varchar(50),

CITY varchar(50))

select \* from emp

insert into emp values (1,'Rohit','PUNE')

--ADD single column

alter table emp add Salary int

--Multiple columns

alter table emp add DOJ datetime , PIN int

--2.We can Drop or delete one or more columns at a time into table.

--Syntax:

--ALTER TABLE TABLE\_NAME DROP COLUMN COLUMN\_NAME

select \* from emp

--DROP SINGLE COLUMN FROM TABLE

ALTER TABLE emp DROP COLUMN PIN

--DROP MULTIPLE COLUMN FROM TABLE

ALTER TABLE emp DROP COLUMN Salary,DOJ

--3.We can Increase or decrease size of columns into table.

EXEC SP\_HELP EMP

--synatx:

--ALTER TABLE Table\_Name ALTER COLUMN Column\_Name Datatype

insert into emp values (2,'TABLE Table\_Name ALTER COLUMN Column\_Name DatatypeTABLE Table\_Name ALTER COLUMN Column\_Name Datatype','PUNE')

--Error: String or binary data would be truncated in table 'JOBHUNT.dbo.emp', column 'EMP\_NAME'. Truncated value: 'TABLE Table\_Name ALTER COLUMN Column\_Name Datatype'.

alter table emp alter column EMP\_NAME varchar(200)

select \* from emp

alter table emp alter column EMP\_NAME varchar(100)

--4.We can change data type of columns.

--synatx:

--ALTER TABLE Table\_Name ALTER COLUMN COlumn\_Name Datatype

EXEC sp\_help emp

select \* from emp

alter table emp alter column emp\_name int

alter table emp alter column EMP\_ID varchar(20)

truncate table emp

--5.We can define constraints to column into table.

--synatx:

--ALTER TABLE Table\_Name ADD CONSTRAINT Constraint\_Name Constraint (Column\_Name );

create table ALTER\_CONSTRAINT\_CHECK(

CID int,

First\_Name varchar(20),

Last\_Name varchar(20),

City varchar(20),

Age int)

insert into ALTER\_CONSTRAINT\_CHECK values (1,'Amit','Patil','Pune',20)

select \* from ALTER\_CONSTRAINT\_CHECK;

--If we want to define PK on existing table column then,

--1.If table does not have data or records or empty table then we define primary key but PK defining column must have NOT NULL constraint.

--2.If data is already inserted in table then it should be unique and column should have NOT NULL defined constraint.

ALTER TABLE ALTER\_CONSTRAINT\_CHECK ADD CONSTRAINT praveen primary key (CID )

exec sp\_help ALTER\_CONSTRAINT\_CHECK

truncate table ALTER\_CONSTRAINT\_CHECK

select \* from ALTER\_CONSTRAINT\_CHECK

--NULL CONSTRAINT

alter table ALTER\_CONSTRAINT\_CHECK alter column CID int NOT NULL

select \* from INFORMATION\_SCHEMA.TABLES

--6.We can drop constraints from column into table.

--synatx:

--alter table table\_Name DROP Constraint COnstraint\_Name

alter table ALTER\_CONSTRAINT\_CHECK DROP Constraint praveen

--default

--synatx:

--alter table table\_name ADD constraint constraint\_name default Default\_Vale FOR Column\_Name

select \* from ALTER\_CONSTRAINT\_CHECK

exec sp\_help ALTER\_CONSTRAINT\_CHECK

alter table ALTER\_CONSTRAINT\_CHECK ADD constraint Default\_C default 'PUNE' FOR CITY

--unique

--synatx:

--alter table table\_Name add constraint Constraint\_Name unique (col1,col2)

alter table ALTER\_CONSTRAINT\_CHECK add constraint Unique1 unique (First\_Name,Last\_Name)

--Foreign Key

--alter table table\_Name add constraint Constraint\_Name Foreign Key (column\_name) refrences (PK\_of\_another table)

--CLASS-10

--RENAME

--In SQL we have built in store procedure to rename column or table.

--We dont have any such keyword like RENAME in SQL server.

--To rename table column

--synatx:

--EXEC SP\_RENAME 'TABLE\_NAME.COLUMN\_NAME (old)', 'COLUMN\_NAME(NEW)'

select \* from Student

exec sp\_rename 'Student.S\_NAME','STUDENT\_NAME'

--whenever we are changing any column name then it will give below warning

--Caution: Changing any part of an object name could break scripts and stored procedures.

--change table name

--synatx:

--EXEC SP\_RENAME 'DATABASE\_NAME.DBO.TABLE\_NAME (old)', 'TABLE\_NAME(NEW)'

EXEC sp\_rename 'JOBHUNT.[dbo].[NOT\_NULL\_TEST]','PUNE'

--TABLE BACKUP

--If we want to take table back up then we can use below synatx

--synatx:

--SELECT \* INTO TABLE\_NAME\_BKP (new Table) from TABLE\_NAME(Existing)

select \* from [dbo].[employee]

select \* into employee\_bkp\_20230426 from employee

select \* from employee\_bkp\_20230426

--If we want to copy only structure as well as data of table

select \* INTO employee\_structure from employee where 1=1

----If we want to copy only structure

select \* INTO employee\_str from employee where 1=2

select \* from employee\_str

--Copy table data

--If we want to copy some of the records from one table into another then we can copy it

--but condition is that structure should be same for both the tables

--Synatx:

--insert into TABLE\_NAME(existing table ) select \* from TABLE\_NAME(existing table )

select \* from employee

select \* into emp\_sal from employee where 1=2

--create same employee copy table whose salary is <=5000

insert into emp\_sal select \* from employee where salary <= 5000

select \* from emp\_sal

--INFORMATION\_SCHEMA

--By using INFORMATION\_SCHEMA, we will identify how amny tables are their in particular database.

--INFORMATION\_SCHEMA is SQL server in-built schema.

--all the tables from database

select \* from INFORMATION\_SCHEMA.TABLES

--all the columns from all the tables existing in database

select \* from INFORMATION\_SCHEMA.COLUMNS

--if we want to know the columns from particular table

select \* from INFORMATION\_SCHEMA.COLUMNS where TABLE\_NAME = 'employee'

--if we want to search a particular column then we can use like operator

select \* from INFORMATION\_SCHEMA.TABLES where TABLE\_NAME like 'e%'

--JOIN

--JOIN is used to return value from two or more than two tables, which should have common column in both the tables.

--JOIN keyword is used to join two or more tables and extract data from it.

--JOIN = Cross Product + Condition

--Q.What are the diffrent types of joins in SQL?

--Types of joins

--1. JOIN / INNER JOIN

--2. OUTER JOIN

--A. LEFT JOIN / LEFT OUTER JOIN

--B. RIGHT JOIN / RIGHT OUTER JOIN

--C. FULL JOIN / FULL OUTER JOIN

--Q.What is the diffrence between Join and Inner join? - Both are same.

--Q.What is the diffrence between left Join and Left outer join?

--Q.What is the diffrence between Right Join and Right outer join?

--Q.What is the diffrence between Full Join and Full outer join?

--Without using join key word

--3.SELF JOIN

--4.EQUI-JOIN

--Q.What is the diffrence between JOIN or INNER join and EQUI-JOIN?

--5.CROSS JOIN

--Types

--1.JOIN or INNER JOIN

--INNER JOIN returns only matching records from both the tables based on condition.

--JOIN = Cross Product + condition

--synatx:

--Select \*/Column\_Name(s) from Table\_Name1 T1

--JOIN / INNER JOIN Table\_Name2 T2

--ON T1.Column\_NAME = T2.Column\_NAME

--Ex:

--students S1 Dept D1

--S\_ID S\_NAME City D\_ID DEPT\_NAME S\_ID

--ON S1.S\_ID = D1.S\_ID

create table students (S\_ID int, S\_NAME varchar(20), City varchar(20))

insert into students values(1,'Amar','Ranchi'),(2,'Akash','Nagpur'),(3,'Rohit','Nanded'),

(4,'Priya','Wardha'),(5,'Suyog','Yavatmal')

create table Dept (D\_ID int, DEPT\_NAME varchar(20), S\_ID int)

insert into Dept values (1,'Electronis',3),(2,'Civil',4),(3,'Mech',5),(4,'Chemical',6),(5,'CSE',7)

select \* from students;

select \* from Dept;

select \* from students s1 join Dept d1 ON s1.S\_ID = d1.S\_ID

--join

select \* from students s1 join Dept d1 ON s1.S\_ID = d1.S\_ID

--need to extract S\_ID,S\_NAME and DEP\_NAME form student and department

select s1.S\_ID,S\_NAME,DEPT\_NAME from students s1 join Dept d1 ON s1.S\_ID = d1.S\_ID

--CLASS-11

--2.Outer Join

--A.Left Join / Left Outer JOIN

--All the records from left table and matching records from right table and

--it will show NULL values for no-matching records left table.

--synatx:

--Select \*/Column\_Name(s) from Table\_Name1 T1

--LEFT JOIN / LEFT OUTER JOIN Table\_Name2 T2

--ON T1.Column\_NAME = T2.Column\_NAME

select \* from students

select \* from Dept

select \* from students s1 left join Dept d1 on s1.S\_ID = d1.S\_ID

--A =1000 b = 2000 full join 500 + 500 + 1500 = 2500

--B.RIGHT Join / RIGHT Outer JOIN

--All the records from Right table and matching records from Left table and

--it will show NULL values for no-matching records in right table.

--synatx:

--Select \*/Column\_Name(s) from Table\_Name1 T1

--RIGHT JOIN / RIGHT OUTER JOIN Table\_Name2 T2

--ON T1.Column\_NAME = T2.Column\_NAME

select \* from students s1 right join Dept d1 on s1.S\_ID = d1.S\_ID

--C.FULL JOIN or FULL OUTER JOIN

--It will display all the records or complete records from both the tables.

--If records are not matching then, for respective tables it will display NULL values.

--synatx:

--Select \*/Column\_Name(s) from Table\_Name1 T1

--FULL JOIN / FULL OUTER JOIN Table\_Name2 T2

--ON T1.Column\_NAME = T2.Column\_NAME

select \* from students s1 FUll join Dept d1 on s1.S\_ID = d1.S\_ID

select \* from students s1 FUll outer join Dept d1 on s1.S\_ID = d1.S\_ID

--Q. can we convert left join to inner join?

--Q. can we convert Right join to inner join?

truncate table students

--4.Equi-Join

--If we are joining two tables without using JOIN keyword then that will be considered it as Equi\_Join.

select \* from students S1 , Dept d1 where S1.S\_ID = d1.S\_ID --equi-join

select \* from students join Dept ON students.S\_ID = Dept.S\_ID --join/innerjoin

/\* Q. Create below tables students(S\_ID,S\_NAME,CITY) , dept(D\_ID,DEPT\_NAME,S\_ID) ,

college\_sport\_Team(CST\_ID,TEAM\_NAME,D\_ID) find out those students name who is part of college sport team.\*/

--5.Cross JOIN

--It is cartesian product.

--For ex: If table A having 4 records and Table B having 3 records then it will display (4\*3=12) 12 records.

select \* from students cross join Dept where students.S\_ID = Dept.S\_ID -- inner join

--CLASS-12

--SELF JOIN

**OPERATORS :**

select count(\*) from results

select \* from results

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

Operators-

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

1. Arithmetic operators--- + , - , /, \*

select (PHYSICS\_MARKS + CHEMISTRY\_MARKS) from results

select (PHYSICS\_MARKS - CHEMISTRY\_MARKS) from results

select (PHYSICS\_MARKS \* CHEMISTRY\_MARKS) from results

select (PHYSICS\_MARKS / CHEMISTRY\_MARKS) from results

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

2. COMPARISON OPERATORS- = , >, <, >=, <=, != , IN, NOT IN, BETWEEN, NOT BETWEEN

SELECT \* FROM RESULTS WHERE PHYSICS\_MARKS = 99

SELECT \* FROM RESULTS WHERE PHYSICS\_MARKS > 50

SELECT \* FROM RESULTS WHERE PHYSICS\_MARKS < 99

SELECT \* FROM RESULTS WHERE PHYSICS\_MARKS >= 99

SELECT \* FROM RESULTS WHERE PHYSICS\_MARKS <= 99

SELECT \* FROM RESULTS WHERE PHYSICS\_MARKS != 99

SELECT \* FROM RESULTS WHERE PHYSICS\_MARKS IN (99,85,60)

SELECT \* FROM RESULTS WHERE PHYSICS\_MARKS NOT IN (99,85,60)

SELECT \* FROM RESULTS WHERE PHYSICS\_MARKS BETWEEN 50 AND 99

SELECT \* FROM RESULTS WHERE PHYSICS\_MARKS NOT BETWEEN 50 AND 99

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

3. CONCATINATION OPERATORS

SELECT (SNAME || SCITY) FROM RESULTS

SELECT (SNAME ||' '|| SCITY) FROM RESULTS

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

4. LOGICAL\_OPERATORS ------ AND, OR , NOT

SELECT \* FROM RESULTS WHERE PHYSICS\_MARKS > 55 AND CHEMISTRY\_MARKS > 55

SELECT \* FROM RESULTS WHERE PHYSICS\_MARKS > 55 OR CHEMISTRY\_MARKS > 55

SELECT \* FROM RESULTS WHERE NOT(STUDENT\_NO =12)

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

5. LIKE\_OPERATORS

SELECT \* FROM RESULTS WHERE SNAME LIKE 'c%'

SELECT \* FROM RESULTS WHERE SNAME LIKE '%m'

SELECT \* FROM RESULTS WHERE SNAME LIKE '%m\_'

SELECT \* FROM RESULTS WHERE SNAME LIKE '%a%'

SELECT \* FROM RESULTS WHERE SNAME LIKE '%hin%'

**---\*\*\*DATETIME FUNCTIONS\*\*\*---**

--add months

--months\_between

--next day

--Last\_Day

--sysdate

--current\_date

--Systimestamp

--Current\_timestamp

--TZ\_OFFSET

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

------add months----------------------------------------------------------------

Select birth\_date, Add\_Months (birth\_date, '6') From Employee;

select add\_months ('05-Jul-23', 6) From dual;

------months\_between------------------------------------------------------------

Select months\_between (Birth\_date, Hire\_date) From Employee;

select months\_between ('05-Jul-23', '01-Jul-21') From dual;

------next day------------------------------------------------------------------

select Birth\_date, next\_day (Birth\_date, 'Sat') From Employee;

select next\_day ('01-Jun-23', 'Fri') From dual;

------last day------------------------------------------------------------------

select Hire\_date, Last\_Day (Hire\_date) From Employee;

Select Last\_Day ('01-Jun-23') From Dual;

------System date----------------To see the current day date--------------------

select sysdate From dual;

------Current date----------------To see the current day date-------------------

select current\_date from dual;

------System time stamp-------It displays present date, time(in detailed manner -Hr/ min/sec/ milliseconds etc)-------

Select Systimestamp From Dual;

------Current time stamp-------It displays present date, time(in detailed manner -Hr/ min/sec/ milliseconds), TIMEZONE details etc-------

Select CURRENT\_timestamp From Dual;

---comparison-------------------------------------------------------------------

Select Systimestamp, CURRENT\_timestamp From Dual;

------TZ\_OFFSET-------------It will give you the offset of timezone

select TZ\_OFFSET('ASIA/CALCUTTA') from dual;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

------\*\*\*Important-------------------------------------------------------------------------------

----------Capgemini/Cognizant/ Infosys / Accenture ----working experience question—-----------------------------------------

Q. How will you find out the current experience of working candidates if their Date Of Joining is given?

Select Months\_Between (Sysdate, Hire\_date) From Employee;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**SELECT \* FROM RESULTS**

**---------------------RANK-------------------------**

Pseudo Columns- These are the virtual columns which will be applied on table externally.

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

Select RANK ( ) over (order by TOTAL\_PERCENTAGE desc) as Rank from RESULTS;

Select TOTAL\_PERCENTAGE, RANK ( ) over (order by TOTAL\_PERCENTAGE desc) as Rank from RESULTS;

Select RESULTS .\*,rank ( ) over (order by TOTAL\_PERCENTAGE desc) as Rank from RESULTS;

----------------------limitation---------------------------------------------------------------------

Mostly Rank queries are not flexible as these queries consider duplicate records also-

It is a drawback of rank query.

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

Note- In ranks-we must use order by clause

---------------------DENSE RANK-------------------------

Select DENSE\_RANK ( ) over (order by TOTAL\_PERCENTAGE desc) as Rank from RESULTS;

Select TOTAL\_PERCENTAGE, DENSE\_RANK ( ) over (order by TOTAL\_PERCENTAGE desc) as Rank from RESULTS;

Select RESULTS .\*,DENSE\_RANK ( ) over (order by TOTAL\_PERCENTAGE desc) as Rank from RESULTS;

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

Select \* from

(Select RESULTS .\*,DENSE\_RANK ( ) over (order by TOTAL\_PERCENTAGE desc) as Rank from RESULTS )

Where Rank = 1;

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

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Q. INTERVIEW QUESTIONS

FIND THE MAX SALARY/ MAX\_PERCENTAGE OF THE DATA?

----

Select \* from

(Select RESULTS .\*,DENSE\_RANK ( ) over (order by TOTAL\_PERCENTAGE desc) as Rank from RESULTS )

Where Rank = 1;

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

Q. INTERVIEW QUESTIONS

FIND SECOND MAX SALARY/ SECOND MAX\_PERCENTAGE OF THE DATA?

----

Select \* from

(Select RESULTS .\*,DENSE\_RANK ( ) over (order by TOTAL\_PERCENTAGE desc) as Rank from RESULTS )

Where Rank = 2;

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

Q. INTERVIEW QUESTIONS

FIND 10TH MAX SALARY/ 10TH MAX\_PERCENTAGE OF THE DATA?

FIND 15TH MAX SALARY/ 15TH MAX\_PERCENTAGE OF THE DATA?

----

Select \* from

(Select RESULTS .\*,DENSE\_RANK ( ) over (order by TOTAL\_PERCENTAGE desc) as Rank from RESULTS )

Where Rank = 10;

Select \* from

(Select RESULTS .\*,DENSE\_RANK ( ) over (order by TOTAL\_PERCENTAGE desc) as Rank from RESULTS )

Where Rank = 15;

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

Q. INTERVIEW QUESTIONS

FIND TOP 5 MAX SALARY/ TOP 5 MAX\_PERCENTAGE OF THE DATA?

-----

Select \* from

(Select RESULTS .\*,DENSE\_RANK ( ) over (order by TOTAL\_PERCENTAGE desc) as Rank from RESULTS )

Where Rank <= 5;

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

Q. INTERVIEW QUESTIONS

FIND 6TH TO 10TH MAX SALARY/ TOP 6TH TO 10TH MAX\_PERCENTAGE OF THE DATA?

-----

Select \* from

(Select RESULTS .\*,DENSE\_RANK ( ) over (order by TOTAL\_PERCENTAGE desc) as Rank from RESULTS )

Where Rank BETWEEN 6 AND 10;

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

Q. INTERVIEW QUESTIONS

FIND 6TH HIGHEST, 10TH HIGHEST, 15TH HIGHEST SALARY/6TH HIGHEST, 10TH HIGHEST, 15TH HIGHEST MAX\_PERCENTAGE OF THE DATA?

-----

Select \* from

(Select RESULTS .\*,DENSE\_RANK ( ) over (order by TOTAL\_PERCENTAGE desc) as Rank from RESULTS )

Where Rank IN (6,10,15);

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

Q. INTERVIEW QUESTIONS

FIND MAX SALARIES OTHER THAN- 6TH HIGHEST, 10TH HIGHEST, 15TH HIGHEST SALARY/

HIGHEST MAX\_PERCENTAGE OTHER THAN 6TH HIGHEST, 10TH HIGHEST, 15TH HIGHEST OF THE DATA?

-----

Select \* from

(Select RESULTS .\*,DENSE\_RANK ( ) over (order by TOTAL\_PERCENTAGE desc) as Rank from RESULTS )

Where Rank NOT IN (6,10,15);

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

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Note- In RANK AND DENSE RANKS-we must use order by clause

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

Note- For Minimum salaries-------Use ascending function only at the place of desc

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

create table

Results

(Student\_no int,

Sname varchar(25),

Srollno int primary key,

Physics\_marks int,

Chemistry\_marks int,

Maths\_marks int,

Biology\_marks int,

total\_marks int,

Total\_Percentage float(4),

Sresult varchar(5),

Scity Char(15))

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

Insert Into Results Values (001, 'chinmay', 111, 99, 88, 88, 89, Null, Null,Null, 'Pune');

Insert Into Results Values (002, 'shobha', 112, 83, 88, 88, 80, Null, Null,Null, 'Mumbai');

Insert Into Results Values (003, 'aparna', 113, 85, 88, 88, 55, Null, Null,Null, 'Pune');

Insert Into Results Values (004, 'mohit', 114, 81, 88, 88, 32, Null, Null,Null, 'Pune');

Insert Into Results Values (005, 'kashish', 115, 85, 88, 88, 80, Null, Null,Null, 'Pune');

Insert Into Results Values (006, 'jhon', 116, 88, 40, 88, 85, Null, Null,Null, 'Pune');

Insert Into Results Values (007, 'danial', 117, 55, 88, 88, 80, Null, Null,Null, 'Pune');

Insert Into Results Values (008, 'chetan', 118, 52, 88, 88, 82, Null, Null,Null, 'Pune');

Insert Into Results Values (009, 'swapnil', 119, 56, 88, 88, 59, Null, Null,Null, 'Pune');

Insert Into Results Values (010, 'roman', 120, 56, 88, 88, 89, Null, Null,Null, 'Pune');

Insert Into Results Values (011, 'harry', 121, 55, 88, 88, 89, Null, Null,Null, 'Pune');

Insert Into Results Values (012, 'ronald', 122, 77, 88, 88, 89, Null, Null,Null, 'Pune');

Insert Into Results Values (013, 'yusuf', 123, 66, 88, 88, 80, Null, Null,Null, 'Mumbai');

Insert Into Results Values (014, 'reema', 124, 56, 88, 88, 82, Null, Null,Null, 'Pune');

Insert Into Results Values (015, 'abhishek', 125, 80, 88, 88, 86, Null, Null,Null, 'Satara');

Insert Into Results Values (016, 'seema', 126, 88, 88, 88, 90, Null, Null,Null, 'Satara');

Insert Into Results Values (017, 'sonali', 127, 83, 88, 88, 89, Null, Null,Null, 'Kolhapur');

Insert Into Results Values (018, 'amrut', 128, 86, 88, 88, 80, Null, Null,Null, 'Pune');

Insert Into Results Values (019, 'roopali', 129, 33, 88, 88, 49, Null, Null,Null, 'Pune');

Insert Into Results Values (020, 'manali', 130, 22, 88, 88, 69, Null, Null,Null, 'Pune');

Insert Into Results Values (021, 'aishwarya', 131, 22, 88, 88, 79, Null, Null,Null, 'Pune');

Insert Into Results Values (022, 'surekha', 132, 83, 88, 88, 99, Null, Null,Null, 'Hyderabad');

Insert Into Results Values (023, 'gaurav', 133, 86, 88, 88, 69, Null, Null,Null, 'Hyderabad');

Insert Into Results Values (024, 'kunal', 134, 89, 88, 88, 39, Null, Null,Null, 'Manali');

Insert Into Results Values (025, 'virat', 135, Null, 88, 88, 19, Null, Null,Null, 'Manali');

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

select \* from Results

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

update Results

set

total\_marks =

(Physics\_marks +

Chemistry\_marks +

Maths\_marks +

Biology\_marks )

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

update Results

set total\_percentage = (total\_marks/400)\*100

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

update Results

set

Sresult = 'DIST' where total\_percentage >=75

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

update Results

set

Sresult = 'FC' where total\_percentage between 60 AND 74

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

update Results

set

Sresult = 'Pass' where total\_percentage between 60 AND 35

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

update Results

set

Sresult = 'Fail' where total\_percentage <35

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

DELETE FROM Results WHERE STUDENT\_NO = 25

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

select \* from Results

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**SUBQUERY---**

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

SELECT MAX(PHYSICS\_MARKS) FROM RESULTS

q.INTERVIEW QUESTION-

DISPLAY ENTIRE DETAILS OF THE PERSON WITH MAX SALARY?

DISPLAY ENTIRE DETAILS OF THE PERSON WITH HIGHEST PHYSICS\_MARKS?

--->

SELECT \* FROM RESULTS

WHERE PHYSICS\_MARKS = (SELECT MAX(PHYSICS\_MARKS) FROM RESULTS)

\*\*\*\*\*\*\*OTHER FEW SCENARIOS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

SELECT \* FROM RESULTS

WHERE PHYSICS\_MARKS =

(SELECT MAX(PHYSICS\_MARKS) FROM RESULTS)

----------------USING IN ----------------------

SELECT \* FROM RESULTS

WHERE PHYSICS\_MARKS IN

(SELECT MAX(PHYSICS\_MARKS) FROM RESULTS)

------------------USING >, <-------------------

SELECT \* FROM RESULTS

WHERE PHYSICS\_MARKS >

(SELECT MAX(PHYSICS\_MARKS) FROM RESULTS)

SELECT \* FROM RESULTS

WHERE PHYSICS\_MARKS <

(SELECT MAX(PHYSICS\_MARKS) FROM RESULTS)

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

q.INTERVIEW QUESTION-

DISPLAY ENTIRE DETAILS OF THE PERSON WITH SECOND MAX SALARY?

DISPLAY ENTIRE DETAILS OF THE PERSON WITH SECOND HIGHEST PHYSICS\_MARKS?

--->

SELECT MAX(PHYSICS\_MARKS)

FROM RESULTS

WHERE PHYSICS\_MARKS

NOT IN

(SELECT MAX(PHYSICS\_MARKS) FROM RESULTS)

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

SELECT \* FROM RESULTS

WHERE PHYSICS\_MARKS IN

(SELECT MAX(PHYSICS\_MARKS)

FROM RESULTS

WHERE PHYSICS\_MARKS

NOT IN

(SELECT MAX(PHYSICS\_MARKS) FROM RESULTS))

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

SELECT MAX(PHYSICS\_MARKS)

FROM RESULTS

WHERE PHYSICS\_MARKS

<

(SELECT MAX(PHYSICS\_MARKS) FROM RESULTS)

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

SELECT \* FROM RESULTS

WHERE PHYSICS\_MARKS IN

(

SELECT MAX(PHYSICS\_MARKS)

FROM RESULTS

WHERE PHYSICS\_MARKS

<

(SELECT MAX(PHYSICS\_MARKS) FROM RESULTS)

)

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

q.INTERVIEW QUESTION-

DISPLAY ENTIRE DETAILS OF THE PERSON WITH THIRD MAX SALARY?

DISPLAY ENTIRE DETAILS OF THE PERSON WITH THIRD HIGHEST PHYSICS\_MARKS?

--->

SELECT \* FROM RESULTS

WHERE PHYSICS\_MARKS IN

(

SELECT MAX(PHYSICS\_MARKS)

FROM RESULTS WHERE PHYSICS\_MARKS

<

(SELECT MAX(PHYSICS\_MARKS) FROM RESULTS WHERE PHYSICS\_MARKS NOT IN (SELECT MAX(PHYSICS\_MARKS) FROM RESULTS)

)

)

**select \* from results**

**--------------------------------Rownum---------------------------------------**

--It will generate a temporary sequence number.

--To see the basic sequence-

select rownum from results;

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

--To see top 5 records from table-

select results .\*,rownum from results

where rownum<=5;

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

--To see Last 5 records from table-

Select \* from (Select results.\*,rownum from results order by rownum desc)

where rownum<=5;

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

----------------------Rowid-------------------------------------------------

---It is Unique id for each row from a table.

--To see the basic id -

Select rowid from results;

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

--To show all details + rowid-

Select results.\*, Rowid From results;

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

--Rowid is unique for every record.

--It can be anything, a sequence of alphanymeric values, characters, etc. given by database engine e.g-(fsdhgnbsd5865hAAA)

--Rowid will get disturbed if you delete some records.

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

• Important Question- How to find out duplicate records?

We can solve this by 2 methods-

1. Having clause.

2. Rowid

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

If used the having clause, the entire details of the record will not get displayed.

So it will be difficult to identify the original and duplicate records. So we often use rowid method.

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

• Important Question- How to delete duplicate records?

Delete from results where rowid not in

(select max(rowid) from results group by sname);

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

Max-Recent records will be considered as unique and remaining will be removed.

Min- Old records will be considered as unique and remaining will be removed.

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

• Important Question- How to display duplicate records?

Select emp\_info.\*, rowid from results where rowid not in

(select max(rowid) from results group by sname);

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

• Important Question- How to display Unique records?

There are 2 methods to display unique records.

1. Distinct-

Select distinct(sname) from results;

2. Rowid method-

Select results.\*, rowid from sname where rowid in

(select max(rowid) from sname group by sname);