**Prior Instructions**

* **Please do read all the questions before performing any operations in the database**
* **Once you have fully gone through the questions then likewise decide the contents and table columns and follow the below instructions**

1. Create Student Database
2. Create the following table under the Student Database:
   1. StudentBasicInformation
      1. Columns
         1. StudentName
         2. StudentSurname
         3. StudentRollNo
         4. StudentAddress
         5. Add more three basic columns of the name of your own
   2. StudentAdmissionPaymentDetails
      1. Columns
         1. StudentRollNo
         2. AmountPaid
         3. AmountBalance
         4. Add more four basic columns of the name of your own
   3. StudentSubjectInformation
      1. Columns
         1. SubjectOpted
         2. StudentRollNo
         3. SubjectTotalMarks
         4. SubjectObtainedMarks
         5. StudentMarksPercentage
         6. Add more one columns of the name of your own
   4. SubjectScholarshipInformation
      1. Columns
         1. StudentRollNo
         2. ScholarshipName
         3. ScholarshipDescription
         4. ScholarshipAmount
         5. ScholarshipCategory
         6. Add more two columns of the name of your own
3. Insert more than 10 records in each and every table created
4. Snap of the all the tables once the insertion is completed
5. Update any 5 records of your choice in any table like update the StudentAddress with some other address content and likewise so on with any records of any table of your choice
6. Snap of the all the tables post updation
7. Select the student details records who has received the scholarship more than 5000Rs/-
8. Select the students who opted for scholarship but has not got the scholarship
9. Fill in data for the percentage column i.e. StudentMarksPercentage in the table StudentSubjectInformation by creating and using the stored procedure created
10. Decide the category of the scholarship depending upon the marks/percentage obtained by the student and likewise update the ScholarshipCategory column, create a stored procedure in order to handle this operation
11. Create the View which shows balance amount to be paid by the student along with the student detailed information (use join)
12. Get the details of the students who haven’t got any scholarship (use joins/subqueries)
13. Create Stored Procedure which will be return the amount balance to be paid by the student as per the student roll number passed through the stored procedure as the input
14. Retrieve the top five student details as per the StudentMarksPercentage values (use subqueries)
15. Try to use all the three types of join learned today in a relevant way, and explain the same why you thought of using that particular join for your selected scenarios (try to cover relevant and real time scenarios for all the three studied joins)
16. Mention the differences between the delete, drop and truncate commands
17. Get the count of the Scholarship category which is highly been availed by the students, i.e. get the count of the total number of students corresponding to the each scholarships category
18. Along with the assignment no. 17 try to retrieve the maximum used scholarship category
19. Retrieve the percentage of the students along with students detailed information who has scored the highest percentage along with availing the maximum scholarship amount
20. Difference between the Triggers, Stored Procedures, Views and Functions

**ANSWERS**

CREATE TABLE StudentBasicInformation

(

StudentName VARCHAR(20) PRIMARY KEY,

StudentSurName VARCHAR(20),

StudentRollNo INT,

StudentDOB DATE,

StudentAddress VARCHAR(30),

Gender CHAR(1),

PhoneNo NUMBER(10) NOT NULL,

)

insert into StudentBasicInformation values('Nitish', 'Awasthi', 101, '1998-01-10', 'Uttarakhand', 'M', '868467352');

insert into StudentBasicInformation values('Raman', 'Deep', 102, '1998-01-23', 'Delhi', 'F', '8362534118');

insert into StudentBasicInformation values('Mohit', 'Sharma', 103, '1997-05-14', 'Uttarakhand', 'M', '98876765656');

insert into StudentBasicInformation values('Shikhar', 'Sharma', 104, '1999-02-23', 'Uttarakhand', 'M', '9345768129');

insert into StudentBasicInformation values('Deepak', 'Verma', 105, '1994-07-20', 'Uttar Pradesh', 'F', '8764543322');

insert into StudentBasicInformation values('Rishabh', 'Saxena', 106, '1999-07-17', 'Karnataka', 'M', '8765433212');

insert into StudentBasicInformation values('Mohan', 'Madan', 107, '1999-02-27', 'Sikkim', 'F', '8798767898');

insert into StudentBasicInformation values('Saksham', 'Maurya', 108, '1999-05-12', 'Sikkim', 'F', '8776565643');

insert into StudentBasicInformation values('Darshan', 'Singh', 100, '1999-04-24', 'Sikkim', 'F', '6243787887');

insert into StudentBasicInformation values('Saksham', 'Maurya', 110, '1999-03-06', 'Sikkim', 'F', '3242342342');

SELECT \* FROM StudentBasicInformation;

CREATE TABLE StudentAdmissionPaymentDetails

(

StudentRollNo INT PRIMARY KEY,

AmountPaid DECIMAL(10,2),

AmountBalance DECIMAL(10,2),

StudentDepartment VARCHAR(40),

PaymentMode VARCHAR(10),

StudentAdmissionDate DATE,

StudentCategory VARCHAR(10),

FOREIGN KEY(StudentRollNo) references StudentBasicInformation(StudentRollNo)

);

insert into StudentAdmissionPaymentDetails values(1, 23000.00, 5000.00, 'CS DEPT', 'Online', '2019-11-15', 'General');

insert into StudentAdmissionPaymentDetails values(2, 22000.00, 0.00, 'ME DEPT', 'Offine', '2020-11-28', 'OBC');

insert into StudentAdmissionPaymentDetails values(3, 23000.00, 2000.00, 'CS DEPT', 'Online', '2020-12-10', 'OBC');

insert into StudentAdmissionPaymentDetails values(4, 25000.00, 0.00, 'CIV DEPT', 'Offine', '2019-11-15', 'General');

insert into StudentAdmissionPaymentDetails values(5, 23000.00, 0.00, 'POLITICAL SCIENCE', 'Online', '2019-11-15', 'SC');

insert into StudentAdmissionPaymentDetails values(6, 23000.00, 3000.00, 'ME DEPT', 'Offine', '2020-12-25', 'ST');

insert into StudentAdmissionPaymentDetails values(7, 23000.00, 5000.00, 'CS DEPT', 'Offine', '2020-11-15', 'OBC');

insert into StudentAdmissionPaymentDetails values(8, 18000.00, 2000.00, 'CS DEPT', 'Online', '2020-11-18', 'PWD');

insert into StudentAdmissionPaymentDetails values(9, 10000.00, 10000.00, 'CIV DEPT', 'Online', '2020-12-18', 'SC');

insert into StudentAdmissionPaymentDetails values(10, 23000.00, 2000.00, 'ME DEPT', 'Online', '2020-11-20', 'General');

SELECT \* FROM StudentAdmissionPaymentDetails;

CREATE TABLE StudentSubjectInformation

(

StudentRollNo INT,

SubjectOpted VARCHAR(30),

SubjectTotalMarks INT,

SubjectObtainedMarks INT,

StudentMarksPercentage FLOAT,

SubjectTeacherName VARCHAR(30),

PRIMARY KEY(StudentRollNo, SubjectOpted),

FOREIGN KEY(StudentRollNo) references StudentBasicInformation(StudentRollNo)

);

insert into StudentSubjectInformation values(1, 'hindi', 100, 95, 95.0, 'mr. mohit');

insert into StudentSubjectInformation values(1, 'astronomy', 50, 45, 90.0, 'mr. saksham');

insert into StudentSubjectInformation values(1, 'ME DEPT', 50, 43, 86.0, 'Dr. Rochna');

insert into StudentSubjectInformation values(2, 'hindi', 100, 86, 86.0, 'mr. mohit');

insert into StudentSubjectInformation values(2, 'astronomy', 50, 45, 90.0, 'mr. saksham');

insert into StudentSubjectInformation values(3, 'hindi', 100, 91, 91.0, 'mr. mohit');

insert into StudentSubjectInformation values(3, 'astronomy', 50, 47, 94.0, 'mr. saksham');

insert into StudentSubjectInformation values(4, 'hindi', 100, 90, 90.0, 'mr. mohit');

insert into StudentSubjectInformation values(4, 'astronomy', 50, 45, 90.0, 'mr. saksham');

insert into StudentSubjectInformation values(5, 'hindi', 100, 82, 82.0, 'mr. mohit');

insert into StudentSubjectInformation values(5, 'astronomy', 50, 23, 46.0, 'mr. saksham');

insert into StudentSubjectInformation values(6, 'hindi', 100, 88, 88.0, 'mr. mohit');

insert into StudentSubjectInformation values(6, 'astronomy', 50, 30, 60.0, 'mr. saksham');

insert into StudentSubjectInformation values(7, 'hindi', 100, 88, 88.0, 'mr. mohit');

insert into StudentSubjectInformation values(7, 'astronomy', 50, 7, 14.0, 'mr. saksham');

insert into StudentSubjectInformation values(8, 'hindi', 100, 74, 73.0, 'mr. mohit');

insert into StudentSubjectInformation values(8, 'astronomy', 50, 7, 14.0, 'mr. saksham');

insert into StudentSubjectInformation values(9, 'hindi', 100, 75, 75.0, 'mr. mohit');

insert into StudentSubjectInformation values(9, 'astronomy', 50, 39, 78.0, 'mr. saksham');

insert into StudentSubjectInformation values(10, 'hindi', 100, 43, 43.0, 'mr. mohit');

insert into StudentSubjectInformation values(10, 'astronomy', 50, 17, 34.0, 'mr. saksham');

select \* from StudentSubjectInformation;

create table SubjectScholarshipInformation

(

StudentRollNo INT PRIMARY KEY,

ScholarshipName VARCHAR(20),

ScholarshipAmount INT,

ScholarshipCategory VARCHAR(30),

ScholarshipBeginDate DATE,

ScholarshipEndDate DATE,

ScholarshipStatus VARCHAR(10),

FOREIGN KEY(StudentRollNo) references StudentBasicInformation(StudentRollNo)

);

insert into SubjectScholarshipInformation values(1, 'UP Scholar', 10000, 'Medium', '2019-11-15', '2021-01-10', 'pending');

insert into SubjectScholarshipInformation values(2, 'SC/ST Scholarship', 10000, 'Medium', '2019-11-15', '2021-01-10', 'approved');

insert into SubjectScholarshipInformation values(3, 'NIELT SCHOLAR',12000, 'HIGH', '2019-11-15', '2021-01-10', 'approved');

insert into SubjectScholarshipInformation values(4, 'UP Scholar', 10000, 'Medium', '2019-11-15', '2021-01-10', 'approved');

insert into SubjectScholarshipInformation values(5, 'NIELT SCHOLAR',12000, 'HIGH', '2019-11-15', '2021-01-10', 'pending');

insert into SubjectScholarshipInformation values(6, 'UP Scholar', 5000, 'LOW', '2019-11-15', '2021-01-10', 'approved');

insert into SubjectScholarshipInformation values(7, 'NIELT SCHOLAR', 5000, 'LOW', '2019-11-15', '2021-01-10', 'approved');

insert into SubjectScholarshipInformation values(8, 'UP Scholar', 5000, 'LOW', '2019-11-15', '2021-01-10', 'pending');

insert into SubjectScholarshipInformation values(9, 'SC/ST Scholarship', 5000, 'LOW', '2019-11-15', '2021-01-10', 'approved');

insert into SubjectScholarshipInformation values(10, 'SC/ST Scholarship', 12000, 'HIGH', '2019-11-15', '2021-01-10', 'pending');

select \* from SubjectScholarshipInformation;

update SubjectScholarshipInformation

set ScholarshipStatus = 'approved'

where StudentRollNo = 1;

update SubjectScholarshipInformation

set ScholarshipEndDate = '2022-11-30'

where StudentRollNo = 3;

update SubjectScholarshipInformation

set ScholarshipName = 'Utkarsh Scholarship'

where ScholarshipDescription = 'NIELT SCHOLAR';

update SubjectScholarshipInformation

set ScholarshipEndDate = '2021-07-23'

where ScholarshipAmount <= 9000;

update SubjectScholarshipInformation

set ScholarshipAmount = 1000

where ScholarshipCategory = 'High';

select \* from SubjectScholarshipInformation;

7.

select \* from StudentBasicInformation

where StudentRollNo in

(select StudentRollNo from SubjectScholarshipInformation where ScholarshipAmount > 5000);

8.

select StudentName,StudentSurname

from StudentBasicInformation A join SubjectScholarshipInformation B

on A.StudentRollNo = B.StudentRollNo

where ScholarshipStatus = 'pending';

9.

create or replace procedure fillPercentage()

language plpgsql

as $$

begin

update StudentSubjectInformation

set StudentMarksPercentage= cast(SubjectObtainedMarks as float) /cast(SubjectTotalMarks as float) \* 100.0;

end;$$

call fillPercentage();

select \* from StudentSubjectInformation;

10.

create or replace procedure setScholarshipCategory(

rollno int

)

language plpgsql

as $$

declare

percentage float;

begin

select StudentMarksPercentage into percentage

from StudentSubjectInformation where StudentRollNo = rollno;

if(percentage > 80 and percentage <= 90) then

update SubjectScholarshipInformation

set ScholarshipCategory = 'Low', ScholarshipAmount = 5000

where StudentRollNo = rollno;

end if;

if (percentage > 90 and percentage <= 95) then

update SubjectScholarshipInformation

set ScholarshipCategory = 'Medium', ScholarshipAmount = 10000

where StudentRollNo = rollno;

end if;

if (percentage > 95) then

update SubjectScholarshipInformation

set ScholarshipCategory = 'High', ScholarshipAmount = 15000

where StudentRollNo = rollno;

end if;

commit;

end;$$

call setScholarshipCategory(6);

select \* from SubjectScholarshipInformation;

11.

create or replace view BalanceView as

select A.StudentRollNo, StudentName, StudentSurname, StudentDOB\_YYYY\_MM\_DD, StudentAddress , StudentGender, StudentPhoneNumber, AmountBalance

from StudentAdmissionPaymentDetails A join StudentBasicInformation B

on A.StudentRollNo = B.StudentRollNo;

select \* from BalanceView;

12.

select \* from StudentBasicInformation

where StudentRollNo not in

(select StudentRollNo from SubjectScholarshipInformation);

13.

create or replace function getAmountBalance(rollNo int)

returns float

language plpgsql

as $$

declare

balance integer;

begin

select AmountBalance into balance from StudentAdmissionPaymentDetails

where StudentRollNo = rollNo;

return balance;

end;$$

select getAmountBalance(7);

select getAmountBalance(9);

14.

select \* from StudentBasicInformation

where StudentRollNo in (

select StudentRollNo from StudentSubjectInformation

order by StudentMarksPercentage desc

limit 5

);

15.

select StudentName, StudentSurName from

StudentBasicInformation A left join SubjectScholarshipInformation B

on A.StudentRollNo = B.StudentRollNo

where ScholarshipAmount is null;

inner Join : Same as ques11

create or replace view BalanceView as

select A.StudentRollNo, StudentName, StudentSurname, StudentDOB\_YYYY\_MM\_DD, StudentAddress , StudentGender, StudentPhoneNumber, AmountBalance

from StudentAdmissionPaymentDetails A inner join StudentBasicInformation B

on A.StudentRollNo = B.StudentRollNo;

select \* from BalanceView;

Right Join : Selecting students who have got schlorship

select A.StudentRollNo, StudentSurName from SubjectScholarshipInformation A

right join StudentBasicInformation B

on A.StudentRollNo = B.StudentRollNo

where ScholarshipAmount is not null;

16.

**Delete :**

* The DELETE statement in SQL is a Data Manipulation Language (DML) Command.
* It is used to delete tuples from a table.
* The space for the tuples remains after deleting.
* The structure or schema of the table is preserved.

**Drop :**

* DROP statement is a Data Definition Language (DDL) Command.
* It is use to drop the whole table.
* The space for the tuples does not remain intact.
* The structure or schema of the table is not preserved.

**Truncate :**

* TRUNCATE command is a Data Definition Language (DDL)operation.
* It is use to delete all the rows of a relation (table) in one go but not the table itself.
* All spaces allocated for the tuples are removed.
* The structure or schema of the table is preserved.

17.

select ScholarshipCategory, count(StudentRollNo)

from SubjectScholarshipInformation

group by ScholarshipCategory

order by count(StudentRollNo) desc;

18.

create or replace view getScholarshipCategoryCount as

select ScholarshipCategory, count(StudentRollNo) as countOfCategory from SubjectScholarshipInformation

group by ScholarshipCategory

order by count(StudentRollNo) desc;

select ScholarshipCategory, countOfCategory from getScholarshipCategoryCount

where countOfCategory in

(select max(countOfCategory) from getScholarshipCategoryCount);

19.

select StudentName, StudentSurname, A.StudentRollNo, StudentMarksPercentage, ScholarshipAmount

from StudentBasicInformation A join StudentSubjectInformation B

on A.StudentRollNo = B.StudentRollNo

left join SubjectScholarshipInformation C

on A.StudentRollNo = C.StudentRollNo

order by StudentMarksPercentage desc,ScholarshipAmount desc

limit 1;

20.

* **Triggers :**
  + A trigger is a special type of stored procedure that automatically runs when an event occurs in the database server.
  + DML triggers run when a user tries to modify data through a data manipulation language (DML) event. DML events are INSERT, UPDATE, or DELETE statements on a table or view.
* **Stored Procedure :**
  + Stored procedure is a batch of statements grouped as a logical unit and stored in the database.
  + The stored procedure accepts the parameters and executes the T-SQL statements in the procedure, returns the result set if any.
  + It is a prepared SQL code that you can save, so the code can be reused over and over again.
* **View :**
  + In SQL, a view is a virtual table based on the result-set of an SQL statement.
  + A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real tables in the database.
  + You can add SQL functions, WHERE, and JOIN statements to a view and present the data as if the data were coming from one single table.
* **Functions :**
  + A function is a routine that accepts parameters, performs an action, such as a complex calculation, and returns the result of that action as a value.
  + The return value can either be a scalar (single) value or a table.