SQL QUERIES

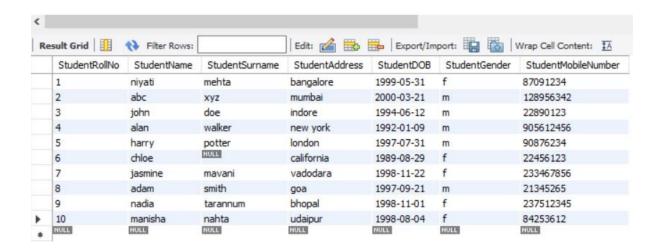
```
1. CREATE DATABASE Student;
   2. CREATE TABLE 'student'. 'studentbasicinformation' (
      `StudentRollNo` INT NOT NULL,
      `StudentName` VARCHAR(45),
      `StudentSurname` VARCHAR(45),
      `StudentAddress` VARCHAR(45),
       `StudentDOB` DATE,
      `StudentGender` VARCHAR(1),
       `StudentMobileNumber` INT,
      PRIMARY KEY ('StudentRollNo'));
      CREATE TABLE studentadmissionpaymentdetails (
      AdmissionID INT,
      StudentRollNo INT,
      AmountPaid INT,
      AmountBalance INT,
      DateOfPayment DATE,
       ModeOfPayment VARCHAR(45),
      PRIMARY KEY(AdmissionID),
      FOREIGN KEY (StudentRollNo) REFERENCES studentbasicinformation
(StudentRollNo));
      CREATE TABLE studentsubjectinformation (
      StudentInfoID INT NOT NULL,
       StudentRollNo INT,
      StudentOpted VARCHAR(45),
      SubjectTotalMarks INT,
      SubjectObtainedMarks INT, StudentMArksPercentage INT,
      PRIMARY KEY(StudentInfoID),
      FOREIGN KEY(StudentRollNo) REFERENCES
studentbasicinformation(StudentRollNO));
      CREATE TABLE subjects cholarship info (
      ScholarshipID INT,
       StudentRollNo INT,
      ScholarshipName VARCHAR(45),
      ScholarshipDescription VARCHAR(45),
      ScholarshipAmount INT,
      ScholarshipCategory INT,
```

ScholarshipObtainedDate DATE,

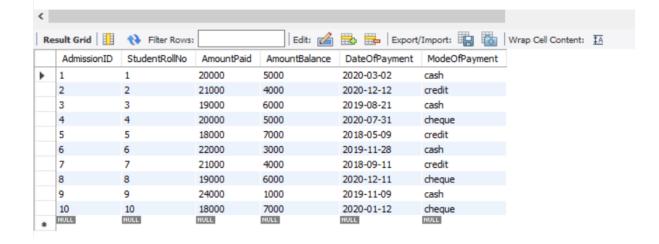
PRIMARY KEY (ScholarshipID), FOREIGN KEY (StudentRollNo) REFERENCES studentbasicinformation (StudentRollNo));

3 & 4

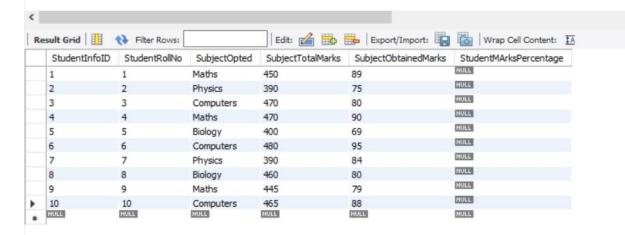
select * from studentbasicinformation;



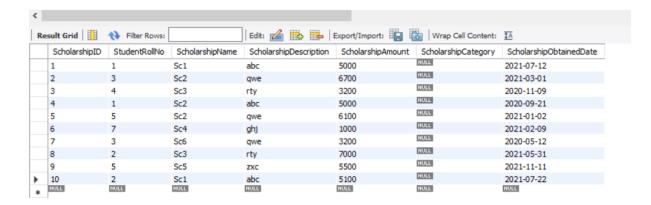
1 select * from studentadmissionpaymentdetails;



1 select * from studentsubjectinformation;



select * from subjectscholarshipinfo;

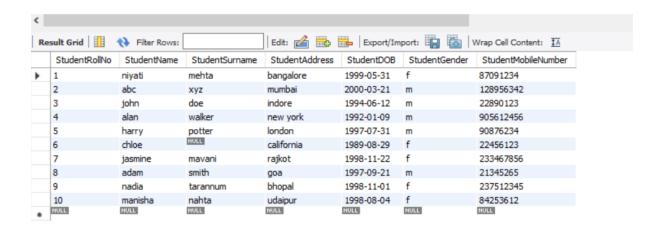


Updation List

5.& 6.

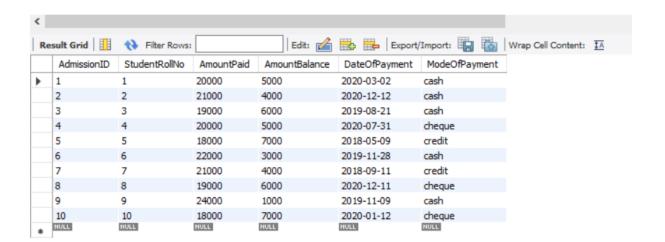
(i) update studentbasicinformation set StudentAddress="rajkot" where StudentRollNo=7;

```
update studentbasicinformation set StudentAddress="rajkot" where StudentRollNo=7;
select * from studentbasicinformation;
```

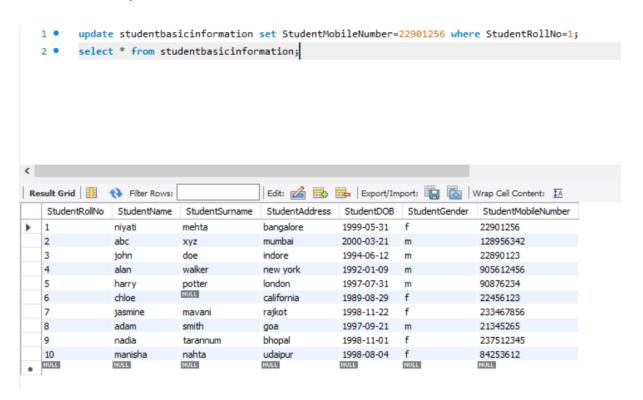


(ii) update studentadmissionpaymentdetails set ModeOfPayment="cash" where StudentRollNo=2;

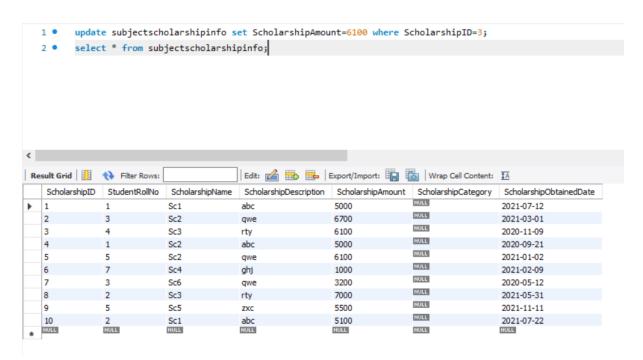
```
update studentadmissionpaymentdetails set ModeOfpayment="cash" where StudentRollNo=2;
select * from studentadmissionpaymentdetails;
```



(iii) studentbasicinformation set StudentMobileNumber=22901256 where StudentRollNo=1;

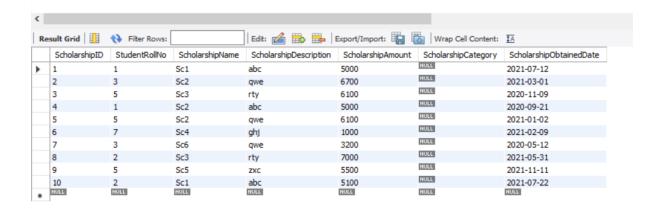


(iv) update subjectscholarshipinfo set ScholarshipAmount=6100 where ScholarshipID=3;



(v) update subjectscholarshipinfo set StudentRollNo=5 where ScholarshipID=3;

```
1 • update subjectscholarshipinfo set StudentRollNo=5 where ScholarshipID=3;
```



7. select sb.StudentRollNo,

sb.StudentName,sb.StudentSurname,ss.ScholarshipName,ss.ScholarshipAmount from studentbasicinformation as sb,subjectscholarshipinfo as ss where sb.StudentRollNo=ss.StudentRollNo AND ss.ScholarshipAmount>5000;

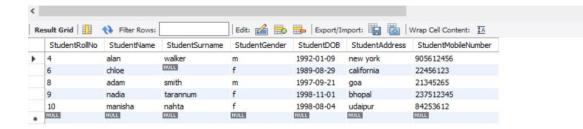
- select sb.StudentRollNo, StudentName,StudentSurname,ss.ScholarshipName,ScholarshipAmount
 from studentbasicinformation as sb,subjectscholarshipinfo as ss
 where sb.StudentRollNo=ss.StudentRollNo AND ss.ScholarshipAmount>5000;
- Result Grid Filter Rows: Export: Wrap Cell Content: IA ScholarshipAmount StudentRollNo StudentName StudentSurname ScholarshipName Sc2 6700 3 iohn doe Sc3 6100 harry potter 5 harry potter Sc2 6100 2 7000 abc Sc3 xyz 5 potter Sc5 5500 harry 2 Sc1 5100 xyz

8. Students who have Applied for scholarships but have not received it till "DATE"

select sb.StudentRollNo, sb.StudentName, sb.StudentMobileNumber, ss.ScholarshipAmount, ss.ScholarshipObtainedDate

from studentbasicinformation as sb, subjectscholarshipinfo as ss where sb.StudentRollNo=ss.StudentRollNo AND ss.ScholarshipObtainedDate>2021-02-14 order by sb.StudentRollNo;

```
select StudentRollNo,StudentName,StudentSurname,StudentGender,StudentDOB,StudentAddress,StudentMobileNumber
from studentbasicinformation where StudentRollNo not in
(select distinct StudentRollNo from subjectscholarshipinfo)
```



9. The percentage is taken as a calculation on the total marks which is assumed to be a summation of 5 subjects

drop procedure if exists calc_percentage; DELIMITER \$\$

create procedure calc percentage()

begir

update studentsubjectinformation set StudentMArksPercentage = (SubjectTotalMarks)/5; end\$\$

call calc_percentage();
select * from studentsubjectinformation;

```
1 •
         drop procedure if exists calc_percentage;
  2
         DELIMITER $$
  3 •
         create procedure calc_percentage()
  4
  5
         update studentsubjectinformation set StudentMArksPercentage = (SubjectTotalMarks)/5;
         end$$
  6
  7
  8 •
         call calc_percentage();
  9
         select * from studentsubjectinformation;
Result Grid | Filter Rows:
                                         Export: Wrap Cell Content: IA
   StudentInfoID
                 StudentRollNo
                               SubjectOpted SubjectTotalMarks
                                                             SubjectObtainedMarks
                                                                                  StudentMArksPercentage
                                                                                 90
  1
                1
                              Maths
                                                             89
  2
                2
                                            390
                                                             75
                                                                                 78
                              Physics
                              Computers
                                                             80
                                                                                 94
  3
                3
                                            470
                4
  4
                                           470
                                                             90
                                                                                 94
                              Maths
  5
                5
                              Biology
                6
                                            480
                                                             95
                                                                                 96
                              Computers
  7
                7
                              Physics
                                            390
                                                             84
                                                                                 78
                8
                                                             80
                                                                                 92
  8
                              Biology
                                            460
                                            445
  9
                9
                                                             79
                                                                                 89
                              Maths
  10
                10
                                            465
                                                             88
                                                                                 93
                              Computers
```

10. The category is chosen based on the percentage obtained by the student.

```
drop procedure if exists calc_category;

DELIMITER $$

create procedure calc_category()

begin

update subjectscholarshipinfo ss inner join studentsubjectinformation si on

ss.StudentRollNo=si.StudentRollNo

set ss.ScholarshipCategory = case

when si.StudentMArksPercentage>=94 then 1

when si.StudentMArksPercentage>=90 then 2

when si.StudentMArksPercentage>=80 then 3

when si.StudentMArksPercentage>70 then 4

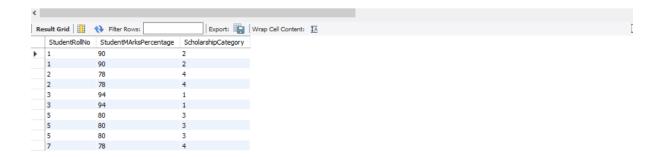
end;

END$$

call calc_category();
```

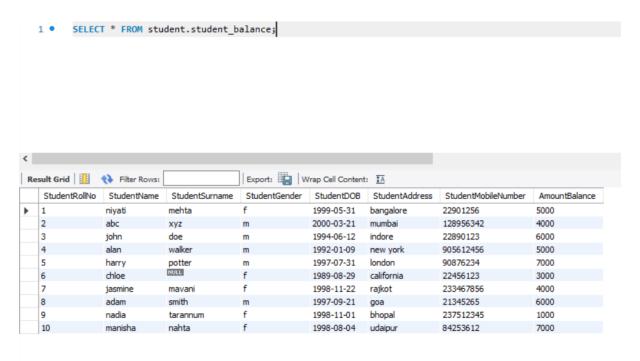
select StudentRollNo,StudentMArksPercentage,ScholarshipCategory from studentsubjectinformation natural join subjectscholarshipinfo;

1 • select StudentRollNo,StudentMArksPercentage,ScholarshipCategory from studentsubjectinformation natural join subjectscholarshipinfo;



11. create view Student_balance as select

sb.StudentRollNo,sb.StudentName,sb.StudentSurname,sb.StudentGender,sb.StudentDOB, sb.StudentAddress,sb.StudentMobileNumber,sa.AmountBalance from studentbasicinformation as sb natural join studentadmissionpaymentdetails as sa; SELECT * FROM student.student balance;



12. SUB-QUERY

select

StudentRollNo,StudentName,StudentSurname,StudentGender,StudentDOB,StudentAddress, StudentMobileNumber from studentbasicinformation where StudentRollNo not in

(select distinct StudentRollNo from subjectscholarshipinfo)

OR

JOIN

select distinct

sb. Student Roll No, Student Name, Student Surname, Student Gender, Student DOB, Student Address, Student Mobile Number

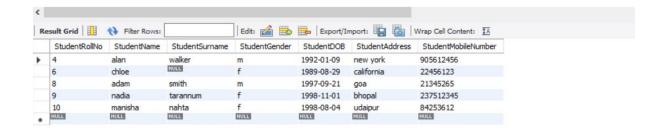
from studentbasicinformation as sb

left join subjectscholarshipinfo as ss

on sb.StudentRollNo=ss.StudentRollNo

where ss.StudentRollNo is NULL;

```
select StudentRollNo,StudentName,StudentGender,StudentDOB,StudentAddress,StudentMobileNumber
from studentbasicinformation where StudentRollNo not in
(select distinct StudentRollNo from subjectscholarshipinfo)
```



13. DELIMITER \$\$

create procedure student balance(IN rollno int)

begin

select

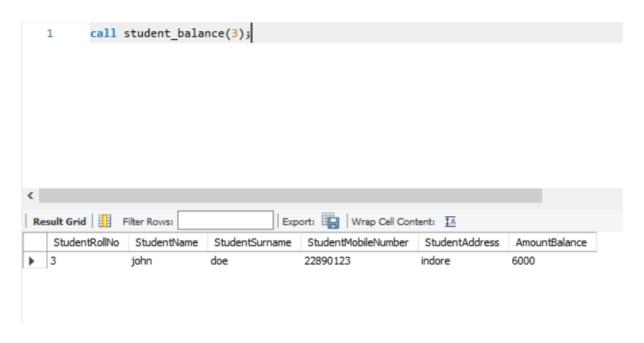
sb. Student Roll No, sb. Student Name, sb. Student Surname, sb. Student Mobile Number, sb. Student Address, sa. Amount Balance from student basic information as sb,

studentadmissionpaymentdetails as sa where sb.StudentRollNo=rollno and

sb.StudentRollNo=sa.StudentRollNo;

end \$\$

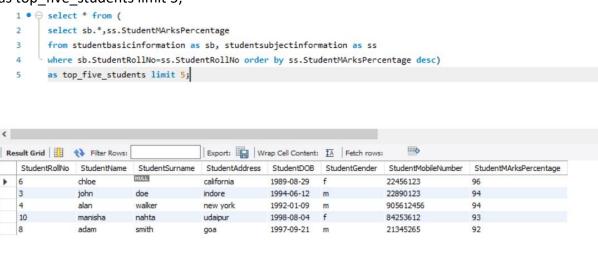
call student balance(3);



14. select * from (

select sb.*,ss.StudentMArksPercentage

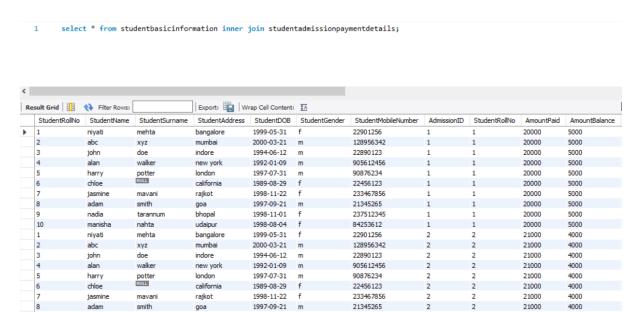
from studentbasicinformation as sb, studentsubjectinformation as ss where sb.StudentRollNo=ss.StudentRollNo order by ss.StudentMArksPercentage desc) as top_five_students limit 5;



15. Three Types of Joins

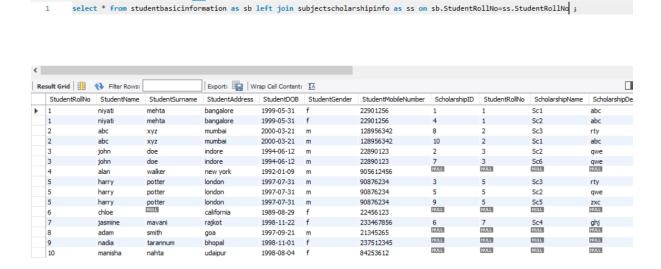
INNER JOIN

This join can be used for the basic info and the admission details so that we can see the student and the corresponding admission information about the student.



LEFT OUTER JOIN

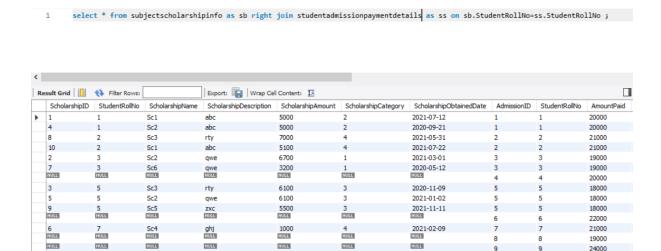
This join can be used on the basic info and scholarship info and this can be prepared so as to check which student has not applied for or got scholarships.



RIGHT OUTER JOIN

NULL

This join can be used with scholarship info and admission details to compare the scholarships given and the Fees paid so that scholarship amount can be redeemed in the fees.



NULL

18000

16. Difference between drop, delete and truncate

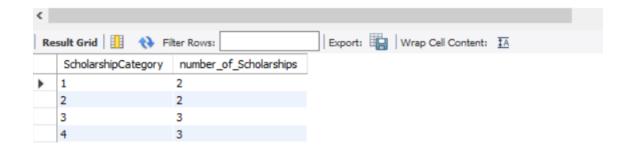
DROP – It is a DDL command where all the contents of the table are lost, i.e. the rows as well as the attributes(columns) are deleted and the space for the table is also freed.

DELETE – It is a DML command where a particular row is deleted based on a particular condition. If the condition is met that particular row is deleted.

TRUNCATE – It is DDL command where all the rows are deleted but the attributes (columns) in the table are not deleted.

17. select ScholarshipCategory,count(*) as number_of_Scholarships from subjectscholarshipinfo group by ScholarshipCategory order by ScholarshipCategory;

```
select ScholarshipCategory,count(*) as number_of_Scholarships
from subjectscholarshipinfo
group by ScholarshipCategory
order by ScholarshipCategory;
```



18. select all ScholarshipCategory, max(number_of_Scholarships) as max_number_scholarship from(
select ScholarshipCategory,count(*) as number_of_Scholarships
from subjectscholarshipinfo
group by ScholarshipCategory
order by ScholarshipCategory)
as bb:



19. select sb.*,s.StudentMArksPercentage from studentbasicinformation as sb natural join studentsubjectinformation as s where StudentRollNo in (select sch.StudentRollNo from subjectscholarshipinfo as sch where sch.ScholarshipAmount= (

 $select\ max (Scholar ship Amount)\ from\ subjects cholar ship info$

20.

TRIGGERS – It is a stored program which is invoked automatically when an event occurs such as an insertion, deletion or updation of a table.

STORED PROCEDURES – A stored procedure is a query in the form of a function which has to be invoked again and again. Thus, in this the stored procedure which is defined only once can be called again instead of writing the query again.

VIEWS – A view is a virtual table which is formed by joining 2 or more tables and has columns and rows. This is helpful when a particular joined table needs to be accessed again and again.

FUNCTIONS – The functions can be used for summarizing data in a table. Some of the most commonly functions provided by SQL are AVG(),COUNT(),MAX(),MIN().