Name: **Praveen Kumar Sharma**

AU 2021 - SQL Concepts & Fundamentals (ASSIGNMENT)

Q1. Create Student Database

Student_Management

Limit to 1000 rows

```sql
-- Create Student Database
CREATE DATABASE Student;

-- Select Database
USE Student;
```

Output

Action Output

#	Time	Action	Message
1	12:58:57	CREATE DATABASE Student	1 row(s) affected
2	12:59:01	USE Student	0 row(s) affected

Q2. Create the following table under the Student Database:

StudentBasicInformation

StudentAdmissionPaymentDetails

StudentSubjectInformation

SubjectScholarshipInformation

```sql
-- Create StudentBasicInformation Table
CREATE TABLE StudentBasicInformation (
    StudentRollNo int NOT NULL,
    StudentName varchar(255) NOT NULL,
    StudentSurname varchar(255),
    StudentAddress varchar(255),
    StudentGender varchar(10),
    StudentDateOfBirth varchar(20),
    StudentContactDetail varchar(22),
    PRIMARY KEY (StudentRollNo)
);

-- Create StudentAdmissionPaymentDetails Table
CREATE TABLE StudentAdmissionPaymentDetails (
    StudentRollNo int NOT NULL,
    AmountPaid int NOT NULL,
    AmountBalance int NOT NULL,
    TotalAmount int NOT NULL,
    ModeOfPayment varchar(20),
    FineTillDate int NOT NULL,
    LastDateOfPayment varchar(20),
    PRIMARY KEY (StudentRollNo)
);
```

```sql
-- Create StudentSubjectInformation Table
CREATE TABLE StudentSubjectInformation (
    SubjectOpted varchar(20) NOT NULL,
    StudentRollNo int NOT NULL,
    SubjectTotalMarks int NOT NULL,
    SubjectObtainedMarks int NOT NULL,
    StudentMarksPercentage FLOAT(2) NOT NULL,
    StudentGrade varchar(20),
    PRIMARY KEY (StudentRollNo)
);

-- Create SubjectScholarshipInformation Table
CREATE TABLE SubjectScholarshipInformation (
    StudentRollNo int NOT NULL,
    ScholarshipName varchar(20),
    ScholarshipDescription varchar(255),
    ScholarshipAmount int NOT NULL,
    ScholarshipCategory varchar(20),
    ScholarshipBankDetail varchar(20),
    ScholarshipStatus varchar(20),
    PRIMARY KEY (StudentRollNo)
);
```

Q3,4. Insert more than 10 records in each and every table created

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	13:27:05	CREATE DATABASE Student	1 row(s) affected	0.000 sec
2	13:27:05	USE Student	0 row(s) affected	0.000 sec
3	13:27:05	CREATE TABLE StudentBasicInformation (StudentRollNo int NOT NULL, StudentName varchar(255) NOT NULL, StudentSurname varchar(255), StudentAddress varchar(255), StudentGender varchar(10), StudentDateOfB...	0 row(s) affected	0.000 sec
4	13:27:05	CREATE TABLE StudentAdmissionPaymentDetails (StudentRollNo int NOT NULL, AmountPaid int NOT NULL, AmountBalance int NOT NULL, TotalAmount int NOT NULL, ModeOfPayment varchar(20), FineTillDate int NO...	0 row(s) affected	0.015 sec
5	13:27:05	CREATE TABLE StudentSubjectInformation (SubjectOpted varchar(20) NOT NULL, StudentRollNo int NOT NULL, SubjectTotalMarks int NOT NULL, SubjectObtainedMarks int NOT NULL, StudentMarksPercentage FLOAT(...	0 row(s) affected	0.016 sec
6	13:27:05	CREATE TABLE SubjectScholarshipInformation (StudentRollNo int NOT NULL, ScholarshipName varchar(20), ScholarshipDescription varchar(255), ScholarshipAmount int NOT NULL, ScholarshipCategory varchar(20), Sc...	0 row(s) affected	0.015 sec
7	13:27:05	ALTER TABLE StudentAdmissionPaymentDetails ADD FOREIGN KEY (StudentRollNo) REFERENCES StudentBasicInformation(StudentRollNo)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.032 sec
8	13:27:05	ALTER TABLE StudentSubjectInformation ADD FOREIGN KEY (StudentRollNo) REFERENCES StudentBasicInformation(StudentRollNo)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.031 sec
9	13:27:05	ALTER TABLE SubjectScholarshipInformation ADD FOREIGN KEY (StudentRollNo) REFERENCES StudentBasicInformation(StudentRollNo)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.015 sec
10	13:27:05	INSERT INTO StudentBasicInformation (StudentRollNo, StudentName, StudentSurname, StudentAddress, StudentGender, StudentDateOfBirth, StudentContactDetail) VALUES (1, 'Mohan','Sharma','RZ12 Badli Delhi', 'M', '20-12-1998', '...	12 row(s) affected Records: 12 Duplicates: 0 Warnings: 0	0.000 sec
11	13:27:05	INSERT INTO StudentAdmissionPaymentDetails (StudentRollNo, AmountPaid, AmountBalance, TotalAmount, ModeOfPayment, FineTillDate, LastDateOfPayment) VALUES (1, 5000,5000,10000, 'CASH', 0, '15-12-2020'), (2, 3000,700...	12 row(s) affected Records: 12 Duplicates: 0 Warnings: 0	0.015 sec
12	13:27:05	INSERT INTO StudentSubjectInformation (SubjectOpted, StudentRollNo, SubjectTotalMarks, SubjectObtainedMarks, StudentMarksPercentage, StudentGrade) VALUES ("English", 1,100,92, 92, 'A+'), ("English", 2,100,88, 88, 'A'), ("...	12 row(s) affected Records: 12 Duplicates: 0 Warnings: 0	0.000 sec
13	13:27:05	INSERT INTO SubjectScholarshipInformation (StudentRollNo, ScholarshipName, ScholarshipDescription, ScholarshipAmount, ScholarshipCategory, ScholarshipBankDetail, ScholarshipStatus) VALUES (1,'MERIT','MERIT SCHOLARSH...	11 row(s) affected Records: 11 Duplicates: 0 Warnings: 0	0.000 sec

```sql
SELECT * FROM StudentBasicInformation;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

StudentRollNo	StudentName	StudentSurname	StudentAddress	StudentGender	StudentDateOfBirth	StudentContactDetail
1	Mohan	Sharma	RZ12 Badli Delhi	M	20-12-1998	9876554353
2	Ramesh	Kumar	RZ35 Ranhola Delhi	M	12-11-1997	7834234653
3	Gautam	Singh	RZ75 VikasPuri Delhi	M	24-10-1998	2783972354
4	Santosh	Sharma	RZ23 Nangloi Delhi	M	13-12-1998	8753612983
5	Deepak	Singh	RZ98 Badli Delhi	M	3-08-1997	8975618231
6	Anukriti	Kumari	RZ43 Nangloi Delhi	F	20-12-1998	6743287398
7	Manoj	Patel	RZ55 Najafgarh Delhi	M	20-12-1998	7653829731
8	Pooja	Kumari	RZ23 Dwarka Delhi	F	3-7-1998	7232819371
9	Deepti	Kumari	RZ09 Badl Delhi	F	13-2-1996	9891235456
10	Rajesh	Sharma	RZ68 Uttam Nagar Delhi	M	15-9-1998	5634683711
11	Ashraf	Sharma	RZ422 Nangli Delhi	M	15-10-1998	5642342371
12	Manish	Sharma	RZ764 Uttam Nagar Delhi	M	25-4-1998	6545683711
NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
146 •  SELECT * FROM StudentAdmissionPaymentDetails;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

StudentRollNo	AmountPaid	AmountBalance	TotalAmount	ModeOfPayment	FineTillDate	LastDateOfPayment
1	5000	5000	10000	CASH	0	15-12-2020
2	3000	7000	10000	NET BANKING	0	15-12-2020
4	7000	3000	10000	CASH	60	15-10-2020
5	6000	4000	10000	NET BANKING	0	15-12-2020
6	5000	5000	10000	DEBIT CARD	0	15-12-2020
7	4500	5500	10000	CASH	0	15-12-2020
9	4000	6000	10000	CASH	0	15-12-2020
10	3500	6500	10000	DEBIT CARD	0	15-12-2020
11	3500	6500	10000	CASH	0	15-12-2020
12	3500	6500	10000	DEBIT CARD	0	15-12-2020
NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
147 •  SELECT * FROM StudentSubjectInformation;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

SubjectOpted	StudentRollNo	SubjectTotalMarks	SubjectObtainedMarks	StudentMarksPercentage	StudentGrade
English	1	100	98	0	-
English	2	100	88	0	-
English	3	100	74	0	-
English	4	100	95	0	-
English	5	100	56	0	-
English	6	100	56	0	-
English	8	100	32	0	-
English	9	100	98	0	-
English	10	100	77	0	-
English	11	100	76	0	-
English	12	100	91	0	-
NULL	NULL	NULL	NULL	NULL	NULL

```
148 •  SELECT * FROM SubjectScholarshipInformation;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

StudentRollNo	ScholarshipName	ScholarshipDescription	ScholarshipAmount	ScholarshipCategory	ScholarshipBankDetail	ScholarshipStatus
1	MERIT	MERIT SCHOLARSHIP	10000	-	45345353535	Accepted
2	MERIT	MERIT SCHOLARSHIP	7500	-	23424234234	Accepted
3	MERIT	MERIT SCHOLARSHIP	5000	-	576756756765	Rejected
4	MERIT	MERIT SCHOLARSHIP	8200	-	456456363663	Accepted
6	MERIT	MERIT SCHOLARSHIP	10000	-	6858757474	Rejected
7	MERIT	MERIT SCHOLARSHIP	3300	-	78657464564	Rejected
8	MERIT	MERIT SCHOLARSHIP	4500	-	2142345325	Accepted
9	MERIT	MERIT SCHOLARSHIP	10000	-	4352342141	Rejected
10	MERIT	MERIT SCHOLARSHIP	6300	-	54635252525	Accepted
11	MERIT	MERIT SCHOLARSHIP	2300	-	7645323543	Rejected
12	MERIT	MERIT SCHOLARSHIP	2500	-	7645323543	Rejected
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Q5,6. 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

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	13:53:02	UPDATE StudentBasicInformation SET StudentAddress = 'RZ23 vikaspuri Delhi' WHERE StudentRollNo = 4	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
2	13:53:02	UPDATE SubjectScholarshipInformation SET ScholarshipBankDetail = '9898989898' WHERE StudentRollNo = 1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
3	13:53:02	UPDATE StudentSubjectInformation SET SubjectObtainedMarks = 82 WHERE StudentRollNo = 2	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.016 sec
4	13:53:02	UPDATE StudentAdmissionPaymentDetails SET ModeOfPayment = 'CASH' WHERE StudentRollNo = 2	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
5	13:53:02	UPDATE SubjectScholarshipInformation SET ScholarshipStatus = 'Accepeted' WHERE StudentRollNo = 6	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec

```
SELECT * FROM StudentBasicInformation;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

StudentRollNo	StudentName	StudentSurname	StudentAddress	StudentGender	StudentDateOfBirth	StudentContactDetail
1	Mohan	Sharma	RZ12 Badli Delhi	M	20-12-1998	9876554353
2	Ramesh	Kumar	RZ35 Ranhola Delhi	M	12-11-1997	7834234653
3	Gautam	Singh	RZ75 VikasPuri Delhi	M	24-10-1998	2783972354
4	Santosh	Sharma	RZ23 vikaspuri Delhi	M	13-12-1998	8753612983
5	Deepak	Singh	RZ98 Badli Delhi	M	3-08-1997	8975618231
6	Anukriti	Kumari	RZ43 Nangloi Delhi	F	20-12-1998	6743287398
7	Manoj	Patel	RZ55 Najafgarh Delhi	M	20-12-1998	7653829731
8	Pooja	Kumari	RZ23 Dwarka Delhi	F	3-7-1998	7232819371
9	Deepti	Kumari	RZ09 Badl Delhi	F	13-2-1996	9891235456
10	Rajesh	Sharma	RZ68 Uttam Nagar Delhi	M	15-9-1998	5634683711
11	Ashraf	Sharma	RZ422 Nangli Delhi	M	15-10-1998	5642342371
12	Manish	Sharma	RZ764 Uttam Nagar Delhi	M	25-4-1998	6545683711
NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
SELECT * FROM StudentAdmissionPaymentDetails;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

StudentRollNo	AmountPaid	AmountBalance	TotalAmount	ModeOfPayment	FineTillDate	LastDateOfPayment
1	5000	5000	10000	CASH	0	15-12-2020
2	3000	7000	10000	CASH	0	15-12-2020
4	7000	3000	10000	CASH	60	15-10-2020
5	6000	4000	10000	NET BANKING	0	15-12-2020
6	5000	5000	10000	DEBIT CARD	0	15-12-2020
7	4500	5500	10000	CASH	0	15-12-2020
9	4000	6000	10000	CASH	0	15-12-2020
10	3500	6500	10000	DEBIT CARD	0	15-12-2020
11	3500	6500	10000	CASH	0	15-12-2020
12	3500	6500	10000	DEBIT CARD	0	15-12-2020
NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
178 •   SELECT * FROM StudentSubjectInformation;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

SubjectOpted	StudentRollNo	SubjectTotalMarks	SubjectObtainedMarks	StudentMarksPercentage	StudentGrade
English	1	100	98	0	-
English	2	100	82	0	-
English	3	100	74	0	-
English	4	100	95	0	-
English	5	100	56	0	-
English	6	100	56	0	-
English	8	100	32	0	-
English	9	100	98	0	-
English	10	100	77	0	-
English	11	100	76	0	-
English	12	100	91	0	-
NULL	NULL	NULL	NULL	NULL	NULL

```
179 •   SELECT * FROM SubjectScholarshipInformation;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

StudentRollNo	ScholarshipName	ScholarshipDescription	ScholarshipAmount	ScholarshipCategory	ScholarshipBankDetail	ScholarshipStatus
1	MERIT	MERIT SCHOLARSHIP	10000	-	9898989898	Accepted
2	MERIT	MERIT SCHOLARSHIP	7500	-	23424234234	Accepted
3	MERIT	MERIT SCHOLARSHIP	5000	-	576756756765	Rejected
4	MERIT	MERIT SCHOLARSHIP	8200	-	456456363663	Accepted
6	MERIT	MERIT SCHOLARSHIP	10000	-	6858757474	Accepeted
7	MERIT	MERIT SCHOLARSHIP	3300	-	78657464564	Rejected
8	MERIT	MERIT SCHOLARSHIP	4500	-	2142345325	Accepted
9	MERIT	MERIT SCHOLARSHIP	10000	-	4352342141	Rejected
10	MERIT	MERIT SCHOLARSHIP	6300	-	54635252525	Accepted
11	MERIT	MERIT SCHOLARSHIP	2300	-	7645323543	Rejected
12	MERIT	MERIT SCHOLARSHIP	2500	-	7645323543	Rejected
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Q7. Select the student details records who has received the scholarship more than 5000Rs/-

```
-- Select the student details records who has received the scholarship more than 5000Rs/-
SELECT * FROM StudentBasicInformation
WHERE StudentRollNo IN
    (SELECT StudentRollNo FROM SubjectScholarshipInformation
        WHERE SubjectScholarshipInformation.ScholarshipAmount > 5000);
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

StudentRollNo	StudentName	StudentSurname	StudentAddress	StudentGender	StudentDateOfBirth	StudentContactDetail
1	Mohan	Sharma	RZ12 Badli Delhi	M	20-12-1998	9876554353
2	Ramesh	Kumar	RZ35 Ranhola Delhi	M	12-11-1997	7834234653
4	Santosh	Sharma	RZ23 vikaspuri Delhi	M	13-12-1998	8753612983
6	Anukriti	Kumari	RZ43 Nangloi Delhi	F	20-12-1998	6743287398
9	Deepti	Kumari	RZ09 Badl Delhi	F	13-2-1996	9891235456
10	Rajesh	Sharma	RZ68 Uttam Nagar Delhi	M	15-9-1998	5634683711
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Q8. Select the students who opted for scholarship but has not got the scholarship

```
-- Select the students who opted for scholarship but has not got the scholarship.
SELECT * FROM StudentBasicInformation
WHERE StudentRollNo IN
    (SELECT StudentRollNo FROM SubjectScholarshipInformation
        WHERE SubjectScholarshipInformation.ScholarshipStatus='Rejected');
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

StudentRollNo	StudentName	StudentSurname	StudentAddress	StudentGender	StudentDateOfBirth	StudentContactDetail
3	Gautam	Singh	RZ75 VikasPuri Delhi	M	24-10-1998	2783972354
7	Manoj	Patel	RZ55 Najafgarh Delhi	M	20-12-1998	7653829731
9	Deepti	Kumari	RZ09 Badl Delhi	F	13-2-1996	9891235456
11	Ashraf	Sharma	RZ422 Nangli Delhi	M	15-10-1998	5642342371
12	Manish	Sharma	RZ764 Uttam Nagar Delhi	M	25-4-1998	6545683711
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Q9. Fill in data for the percentage column i.e. StudentMarksPercentage in the table StudentSubjectInformation by creating and using the stored procedure created

```
SELECT * FROM StudentSubjectInformation;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

SubjectOpted	StudentRollNo	SubjectTotalMarks	SubjectObtainedMarks	StudentMarksPercentage	StudentGrade
English	1	100	98	0	-
English	2	100	82	0	-
English	3	100	74	0	-
English	4	100	95	0	-
English	5	100	56	0	-
English	6	100	56	0	-
English	8	100	32	0	-
English	9	100	98	0	-
English	10	100	77	0	-
English	11	100	76	0	-
English	12	100	91	0	-
NULL	NULL	NULL	NULL	NULL	NULL

```
delimiter $$
DROP PROCEDURE IF EXISTS Percentage_Cal;
CREATE procedure Percentage_Cal ()
DETERMINISTIC
BEGIN
   UPDATE StudentSubjectInformation AS sb
    SET sb.StudentMarksPercentage=sb.SubjectObtainedMarks/sb.SubjectTotalMarks*100
        WHERE sb.StudentRollNo!=0;

   UPDATE StudentSubjectInformation AS sb
    SET sb.StudentGrade=CASE
        WHEN sb.StudentMarksPercentage>90 THEN '0'
        WHEN sb.StudentMarksPercentage>80 THEN 'A+'
        WHEN sb.StudentMarksPercentage>70 THEN 'A'
        WHEN sb.StudentMarksPercentage>60 THEN 'B+'
        WHEN sb.StudentMarksPercentage>50 THEN 'B'
        WHEN sb.StudentMarksPercentage>40 THEN 'C'
        WHEN sb.StudentMarksPercentage>33 THEN 'P'
        ELSE 'F'
    END
    WHERE sb.StudentRollNo!=0;

END $$
delimiter ;

#========================

CALL Percentage_Cal;
```

```
228 •   SELECT * FROM StudentSubjectInformation;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

SubjectOpted	StudentRollNo	SubjectTotalMarks	SubjectObtainedMarks	StudentMarksPercentage	StudentGrade
English	1	100	98	98	0
English	2	100	82	82	A+
English	3	100	74	74	A
English	4	100	95	95	0
English	5	100	56	56	B
English	6	100	56	56	B
English	8	100	32	32	F
English	9	100	98	98	0
English	10	100	77	77	A
English	11	100	76	76	A
English	12	100	91	91	0
NULL	NULL	NULL	NULL	NULL	NULL

<u>Q10. 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</u>

```
230   ##########################################################################################################
231   -- Decide the category of the scholarship depending upon the marks/percentage
232   -- obtained by the student and likewise update the ScholarshipCategory column,
233   -- create a stored procedure in order to handle this operation
234 •  SELECT * FROM SubjectScholarshipInformation;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

StudentRollNo	ScholarshipName	ScholarshipDescription	ScholarshipAmount	ScholarshipCategory	ScholarshipBankDetail	ScholarshipStatus
1	MERIT	MERIT SCHOLARSHIP	10000	-	9898989898	Accepted
2	MERIT	MERIT SCHOLARSHIP	7500	-	23424234234	Accepted
3	MERIT	MERIT SCHOLARSHIP	5000	-	576756756765	Rejected
4	MERIT	MERIT SCHOLARSHIP	8200	-	456456363663	Accepted
6	MERIT	MERIT SCHOLARSHIP	10000	-	6858757474	Accepeted
7	MERIT	MERIT SCHOLARSHIP	3300	-	78657464564	Rejected
8	MERIT	MERIT SCHOLARSHIP	4500	-	2142345325	Accepted
9	MERIT	MERIT SCHOLARSHIP	10000	-	4352342141	Rejected
10	MERIT	MERIT SCHOLARSHIP	6300	-	54635252525	Accepted
11	MERIT	MERIT SCHOLARSHIP	2300	-	7645323543	Rejected
12	MERIT	MERIT SCHOLARSHIP	2500	-	7645323543	Rejected
NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
delimiter $$
DROP PROCEDURE IF EXISTS Category_Cal;
CREATE procedure Category_Cal ()
DETERMINISTIC
BEGIN
   UPDATE SubjectScholarshipInformation AS si
    SET si.ScholarshipCategory=CASE
        WHEN ((SELECT sp.StudentMarksPercentage FROM StudentSubjectInformation AS sp WHERE sp.StudentRollNo=si.StudentRollNo)>80) THEN 'CAT_1'
        WHEN ((SELECT sp.StudentMarksPercentage FROM StudentSubjectInformation AS sp WHERE sp.StudentRollNo=si.StudentRollNo)>60) THEN 'CAT_2'
        WHEN ((SELECT sp.StudentMarksPercentage FROM StudentSubjectInformation AS sp WHERE sp.StudentRollNo=si.StudentRollNo)>40) THEN 'CAT_3'
        WHEN ((SELECT sp.StudentMarksPercentage FROM StudentSubjectInformation AS sp WHERE sp.StudentRollNo=si.StudentRollNo)>20) THEN 'CAT_4'
        ELSE 'CAT_4'
    END
    WHERE si.StudentRollNo!=0;

END $$
delimiter ;

#=======================

CALL Category_Cal;
```

```
SELECT * FROM SubjectScholarshipInformation;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

StudentRollNo	ScholarshipName	ScholarshipDescription	ScholarshipAmount	ScholarshipCategory	ScholarshipBankDetail	ScholarshipStatus
1	MERIT	MERIT SCHOLARSHIP	10000	CAT_1	9898989898	Accepted
2	MERIT	MERIT SCHOLARSHIP	7500	CAT_1	23424234234	Accepted
3	MERIT	MERIT SCHOLARSHIP	5000	CAT_2	576756756765	Rejected
4	MERIT	MERIT SCHOLARSHIP	8200	CAT_1	456456363663	Accepted
6	MERIT	MERIT SCHOLARSHIP	10000	CAT_3	6858757474	Accepeted
7	MERIT	MERIT SCHOLARSHIP	3300	CAT_4	78657464564	Rejected
8	MERIT	MERIT SCHOLARSHIP	4500	CAT_4	2142345325	Accepted
9	MERIT	MERIT SCHOLARSHIP	10000	CAT_1	4352342141	Rejected
10	MERIT	MERIT SCHOLARSHIP	6300	CAT_2	54635252525	Accepted
11	MERIT	MERIT SCHOLARSHIP	2300	CAT_2	7645323543	Rejected
12	MERIT	MERIT SCHOLARSHIP	2500	CAT_1	7645323543	Rejected
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Q11. Create the View which shows balance amount to be paid by the student along with the student detailed information (use join)

```
-- Create the View which shows balance amount to be paid
-- by the student along with the student detailed information (use join)
CREATE VIEW BalancePay
AS SELECT a.* , b.AmountBalance
FROM StudentBasicInformation a INNER JOIN StudentAdmissionPaymentDetails b
ON a.StudentRollNo=b.StudentRollNo;
```

```
SELECT * FROM BalancePay;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

StudentRollNo	StudentName	StudentSurname	StudentAddress	StudentGender	StudentDateOfBirth	StudentContactDetail	AmountBalance
1	Mohan	Sharma	RZ12 Badli Delhi	M	20-12-1998	9876554353	5000
2	Ramesh	Kumar	RZ35 Ranhola Delhi	M	12-11-1997	7834234653	7000
4	Santosh	Sharma	RZ23 vikaspuri Delhi	M	13-12-1998	8753612983	3000
5	Deepak	Singh	RZ98 Badli Delhi	M	3-08-1997	8975618231	4000
6	Anukriti	Kumari	RZ43 Nangloi Delhi	F	20-12-1998	6743287398	5000
7	Manoj	Patel	RZ55 Najafgarh Delhi	M	20-12-1998	7653829731	5500
9	Deepti	Kumari	RZ09 Badl Delhi	F	13-2-1996	9891235456	6000
10	Rajesh	Sharma	RZ68 Uttam Nagar Delhi	M	15-9-1998	5634683711	6500
11	Ashraf	Sharma	RZ422 Nangli Delhi	M	15-10-1998	5642342371	6500
12	Manish	Sharma	RZ764 Uttam Nagar Delhi	M	25-4-1998	6545683711	6500

Q12. Get the details of the students who haven't got any scholarship (use joins/subqueries)

```
-- Get the details of the students who haven't got any scholarship (use joins/subqueries)
SELECT * FROM StudentBasicInformation
WHERE StudentRollNo NOT IN
    (SELECT StudentRollNo FROM SubjectScholarshipInformation
        WHERE SubjectScholarshipInformation.ScholarshipStatus!='Rejected');
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

StudentRollNo	StudentName	StudentSurname	StudentAddress	StudentGender	StudentDateOfBirth	StudentContactDetail
3	Gautam	Singh	RZ75 VikasPuri Delhi	M	24-10-1998	2783972354
5	Deepak	Singh	RZ98 Badli Delhi	M	3-08-1997	8975618231
7	Manoj	Patel	RZ55 Najafgarh Delhi	M	20-12-1998	7653829731
9	Deepti	Kumari	RZ09 Badl Delhi	F	13-2-1996	9891235456
11	Ashraf	Sharma	RZ422 Nangli Delhi	M	15-10-1998	5642342371
12	Manish	Sharma	RZ764 Uttam Nagar Delhi	M	25-4-1998	6545683711
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Q13. 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

```
-- 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
delimiter $$
DROP PROCEDURE IF EXISTS `student`.`Category_Cal`;
CREATE procedure Category_Cal (IN rollno INT, OUT amount INT)
DETERMINISTIC
BEGIN
    SELECT AmountBalance
    INTO amount
    FROM StudentAdmissionPaymentDetails
    WHERE StudentRollNo = rollno;
END $$
delimiter ;

#=======================
CALL Category_Cal(2,@balance);
SELECT @balance AS Balance_Pay;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Balance_Pay
7000

Q14. Retrieve the top five student details as per the StudentMarksPercentage values (use subqueries)

```
--  Retrieve the top five student details as per the StudentMarksPercentage values (use subqueries)
SELECT * FROM StudentBasicInformation
    WHERE StudentRollNo IN
        ( SELECT T2.StudentRollNo FROM
            (SELECT * FROM StudentSubjectInformation
                ORDER BY StudentSubjectInformation.SubjectObtainedMarks DESC LIMIT 5)
                    AS T2);
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

StudentRollNo	StudentName	StudentSurname	StudentAddress	StudentGender	StudentDateOfBirth	StudentContactDetail
1	Mohan	Sharma	RZ12 Badli Delhi	M	20-12-1998	9876554353
2	Ramesh	Kumar	RZ35 Ranhola Delhi	M	12-11-1997	7834234653
4	Santosh	Sharma	RZ23 vikaspuri Delhi	M	13-12-1998	8753612983
9	Deepti	Kumari	RZ09 Badl Delhi	F	13-2-1996	9891235456
12	Manish	Sharma	RZ764 Uttam Nagar Delhi	M	25-4-1998	6545683711
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Q15. 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)

1. **INNER JOIN**

   Selects records that have matching values in both tables.

   Ex. List the details all students along with admission payment Details:

```
SELECT * FROM StudentBasicInformation
    AS si
    INNER JOIN StudentAdmissionPaymentDetails
        AS sap
            ON si.StudentRollNo=sap.StudentRollNo;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

StudentRollNo	StudentName	StudentSurname	StudentAddress	StudentGender	StudentDateOfBirth	StudentContactDetail	StudentRollNo	AmountPaid	AmountBalance	TotalAmount	ModeOfPayment	FineTillDate	LastDateOfPayment
1	Mohan	Sharma	RZ12 Badli Delhi	M	20-12-1998	9876554353	1	5000	5000	10000	CASH	0	15-12-2020
2	Ramesh	Kumar	RZ35 Ranhola Delhi	M	12-11-1997	7834234653	2	3000	7000	10000	CASH	0	15-12-2020
4	Santosh	Sharma	RZ23 vikaspuri Delhi	M	13-12-1998	8753612983	4	7000	3000	10000	CASH	60	15-10-2020
5	Deepak	Singh	RZ98 Badli Delhi	M	3-08-1997	8975618231	5	6000	4000	10000	NET BANKING	0	15-12-2020
6	Anukriti	Kumari	RZ43 Nangloi Delhi	F	20-12-1998	6743287398	6	5000	5000	10000	DEBIT CARD	0	15-12-2020
7	Manoj	Patel	RZ55 Najafgarh Delhi	M	20-12-1998	7653829731	7	4500	5500	10000	CASH	0	15-12-2020
9	Deepti	Kumari	RZ09 Badl Delhi	F	13-2-1996	9891235456	9	4000	6000	10000	CASH	0	15-12-2020
10	Rajesh	Sharma	RZ68 Uttam Nagar Delhi	M	15-9-1998	5634683711	10	3500	6500	10000	DEBIT CARD	0	15-12-2020
11	Ashraf	Sharma	RZ422 Nangli Delhi	M	15-10-1998	5642342371	11	3500	6500	10000	CASH	0	15-12-2020
12	Manish	Sharma	RZ764 Uttam Nagar Delhi	M	25-4-1998	6545683711	12	3500	6500	10000	DEBIT CARD	0	15-12-2020

2. **LEFT JOIN**

   Returns all records from the left table (table1), and the matched records from the right table. The result is NULL from the right side, if there is no match.

   Ex. List the student details along with subject whether he have opted or not if yes, the details of the subject which he have opted.

```
SELECT * FROM StudentBasicInformation
    AS si
    LEFT JOIN StudentSubjectInformation
        AS ssi
            ON si.StudentRollNo=ssi.StudentRollNo;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

StudentRollNo	StudentName	StudentSurname	StudentAddress	StudentGender	StudentDateOfBirth	StudentContactDetail	SubjectOpted	StudentRollNo	SubjectTotalMarks	SubjectObtainedMarks	StudentMarksPercentage	StudentGrade
1	Mohan	Sharma	RZ12 Badli Delhi	M	20-12-1998	9876554353	English	1	100	98	98	0
2	Ramesh	Kumar	RZ35 Ranhola Delhi	M	12-11-1997	7834234653	English	2	100	82	82	A+
3	Gautam	Singh	RZ75 VikasPuri Delhi	M	24-10-1998	2783972354	English	3	100	74	74	A
4	Santosh	Sharma	RZ23 vikaspuri Delhi	M	13-12-1998	8753612983	English	4	100	95	95	0
5	Deepak	Singh	RZ98 Badli Delhi	M	3-08-1997	8975618231	English	5	100	56	56	B
6	Anukriti	Kumari	RZ43 Nangloi Delhi	F	20-12-1998	6743287398	English	6	100	56	56	B
7	Manoj	Patel	RZ55 Najafgarh Delhi	M	20-12-1998	7653829731	NULL	NULL	NULL	NULL	NULL	NULL
8	Pooja	Kumari	RZ23 Dwarka Delhi	F	3-7-1998	7232819371	English	8	100	32	32	F
9	Deepti	Kumari	RZ09 Badl Delhi	F	13-2-1996	9891235456	English	9	100	98	98	0
10	Rajesh	Sharma	RZ68 Uttam Nagar Delhi	M	15-9-1998	5634683711	English	10	100	77	77	A
11	Ashraf	Sharma	RZ422 Nangli Delhi	M	15-10-1998	5642342371	English	11	100	76	76	A
12	Manish	Sharma	RZ764 Uttam Nagar Delhi	M	25-4-1998	6545683711	English	12	100	91	91	0

3. RIGHT JOIN

Returns all records from the right table (table2), and the matched records from the left table (table1). The result is NULL from the left side, when there is no match.

Ex. List the Students who have received the scholarship and status of the Admission payment details whether it is present or not and if present then Complete detail.

```
SELECT * FROM StudentAdmissionPaymentDetails
    AS si
    RIGHT JOIN SubjectScholarshipInformation
        AS ssi
            ON si.StudentRollNo=ssi.StudentRollNo;

##########################################################################################################################
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

StudentRollNo	AmountPaid	AmountBalance	TotalAmount	ModeOfPayment	FineTillDate	LastDateOfPayment	StudentRollNo	ScholarshipName	ScholarshipDescription	ScholarshipAmount	ScholarshipCategory	ScholarshipBankDetail	ScholarshipStatus
1	5000	5000	10000	CASH	0	15-12-2020	1	MERIT	MERIT SCHOLARSHIP	10000	CAT_1	9898989898	Accepted
2	3000	7000	10000	CASH	0	15-12-2020	2	MERIT	MERIT SCHOLARSHIP	7500	CAT_1	23424234234	Accepted
NULL	NULL	NULL	NULL	NULL	NULL	NULL	3	MERIT	MERIT SCHOLARSHIP	5000	CAT_2	576756756765	Rejected
4	7000	3000	10000	CASH	60	15-10-2020	4	MERIT	MERIT SCHOLARSHIP	8200	CAT_1	456456363663	Accepted
6	5000	5000	10000	DEBIT CARD	0	15-12-2020	6	MERIT	MERIT SCHOLARSHIP	10000	CAT_3	6858757474	Accepeted
7	4500	5500	10000	CASH	0	15-12-2020	7	MERIT	MERIT SCHOLARSHIP	3300	CAT_4	78657464564	Rejected
NULL	NULL	NULL	NULL	NULL	NULL	NULL	8	MERIT	MERIT SCHOLARSHIP	4500	CAT_4	2142345325	Accepted
9	4000	6000	10000	CASH	0	15-12-2020	9	MERIT	MERIT SCHOLARSHIP	10000	CAT_1	4352342141	Rejected
10	3500	6500	10000	DEBIT CARD	0	15-12-2020	10	MERIT	MERIT SCHOLARSHIP	6300	CAT_2	54635252525	Accepted
11	3500	6500	10000	CASH	0	15-12-2020	11	MERIT	MERIT SCHOLARSHIP	2300	CAT_2	7645323543	Rejected
12	3500	6500	10000	DEBIT CARD	0	15-12-2020	12	MERIT	MERIT SCHOLARSHIP	2500	CAT_1	7645323543	Rejected

Q16. Mention the differences between the delete, drop and truncate commands

- **DELETE :**

It is a Data Manipulation Language Command (DML).

It is use to delete the one or more tuples of a table.

With the help of “DELETE” command we can either delete all the rows in one go or can delete row one by one. i.e., we can use it as per the requirement or the condition using Where clause.

It is comparatively slower than TRUNCATE cmd.

SYNTAX –

If we want to delete all the rows of the table:

DELETE from ;

SYNTAX –

If we want to delete the row of the table as per the condition then we use WHERE clause,

DELETE from WHERE ;

- **DROP:**

It is a Data Definition Language Command (DDL).

It is use to drop the whole table.

With the help of “DROP” command we can drop (delete) the whole structure in one go i.e. it removes the named elements of the schema.

By using this command the existence of the whole table is finished or say lost.

SYNTAX –
If we want to drop the table:
DROP table ;

- **TRUNCATE:**

It is also a Data Definition Language Command (DDL).

It is use to delete all the rows of a relation (table) in one go.

With the help of “TRUNCATE” command we can’t delete the single row as here WHERE clause is not used.

By using this command the existence of all the rows of the table is lost.

It is comparatively faster than delete command as it deletes all the rows fastly.

SYNTAX –
If we want to use truncate :
TRUNCATE ;

Q17. 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

```
-- 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
SELECT SubjectScholarshipInformation.ScholarshipCategory, COUNT(*) as Count
    FROM SubjectScholarshipInformation
        GROUP BY SubjectScholarshipInformation.ScholarshipCategory;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ScholarshipCategory	Count
CAT_1	5
CAT_2	3
CAT_3	1
CAT_4	2

Q18. Along with the assignment no. 17 try to retrieve the maximum used scholarship category

```
-- Try to retrieve the maximum used scholarship category
SELECT MAX(t2.CategoryCount) as Max FROM
    (SELECT SubjectScholarshipInformation.ScholarshipCategory, COUNT(*) CategoryCount FROM
        SubjectScholarshipInformation GROUP BY
            SubjectScholarshipInformation.ScholarshipCategory) AS t2;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Max
5

Q 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

```
-- Retrieve the percentage of the students along with students detailed information who has scored
-- the highest percentage along with availing the maximum scholarship amount
SELECT sbi.*,ssi.StudentMarksPercentage,si.ScholarshipAmount FROM StudentBasicInformation as sbi
    INNER JOIN StudentSubjectInformation ssi ON
        ssi.StudentRollNo=sbi.StudentRollNo
    INNER JOIN SubjectScholarshipInformation si ON
        si.StudentRollNo=sbi.StudentRollNo
    WHERE ssi.StudentMarksPercentage=
                (SELECT MAX(StudentMarksPercentage) as MaxPercent FROM
                                StudentSubjectInformation)
    AND  si.ScholarshipAmount=
                (SELECT MAX(ScholarshipAmount) as MaxScholar FROM
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

StudentRollNo	StudentName	StudentSurname	StudentAddress	StudentGender	StudentDateOfBirth	StudentContactDetail	StudentMarksPercentage	ScholarshipAmount
1	Mohan	Sharma	RZ12 Badli Delhi	M	20-12-1998	9876554353	98	10000
9	Deepti	Kumari	RZ09 Badl Delhi	F	13-2-1996	9891235456	98	10000

Q 20. Difference between the Triggers, Stored Procedures, Views and Functions

- **TRIGGERS**
  - Trigger can be executed automatically on specified action on a table like, update, delete, or update.
  - Trigger can't be called from Store Procedure or Function.
  - We can't pass a parameter to trigger.
  - Trigger never return value on execution.

    SYNTAX:

```
create trigger [trigger_name]
[before | after]
{insert | update | delete}
on [table_name]
[for each row]
[trigger_body]
```

- **STORED PROCEDURES**
  - We can execute the stored procedures when required.
  - Stored Procedures can't be called from a function because functions can be called from a select statement and Stored Procedures can't be called from. But you can call Store Procedure from Trigger.
  - Stored Procedures can accept any type of parameter. Stored Procedures also accept out parameter.
  - Stored Procedures may or may not return any values (Single or table) on execution.

    SYNTAX:

```
CREATE PROCEDURE procedure_name
```

```
AS
sql_statement
GO;

EXEC procedure_name;
```

- **VIEWS**
  - It can be used like a table in most situations, but unlike a table, it can encapsulate very complex calculations and commonly used joins.
  - Views are most useful when you always need to join the same set of tables say an Order with an Order
  - The view is a virtual table. It does not physically exist. Rather, it is created by query joining one or more tables.
  - The fields in a view are fields from one or more real tables in the database.

    SYNTAX:

```
CREATE VIEW view_name AS
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

- **FUNCTIONS**
  - We can call a function whenever required. Function can't be executed because a function is not in pre-compiled form.
  - Function can be called from Store Procedure or Trigger.
  - Function can accept any type of parameter. But function can’t accept out parameter.
  - Function must return any value.

    SYNTAX:

```
CREATE [OR REPLACE] FUNCTION function_name
[(parameter_name [IN | OUT | IN OUT] type [, ...])]
RETURN return_datatype
{IS | AS}
BEGIN
   < function_body >
END [function_name];
```