

SCHOOL MANAGEMENT SYSTEM

|  |  |  |  |
| --- | --- | --- | --- |
| **Academic Year** | **Module** | **Assessment Number** | **Assessment Type** |
| S20 | Introduction to Database Systems (DipIT07) | A1 | Individual Report |

Student Id : NP03A190093

Student Name : Chiranjivi Khadka

Section : DC7

Module Leader : Deepson Shrestha

Lecturer : Deepson Shrestha

Submitted on : <04-06-2020>

Acknowledgement

I would like to thanks my college for giving us this opportunity. This report is for the fulfillment of coursework of database. Special thanks to our module leader for this creative coursework. Similarly, I would like to thanks to Mr. Chiranjivi Khanal for his support.

Table of Contents

[1. Introduction 1](#_Toc42158144)

[Aim 1](#_Toc42158145)

[2. Description 1](#_Toc42158146)

[3. Table Description 1](#_Toc42158147)

[3.1. Students 1](#_Toc42158148)

[3.2 Teacher 1](#_Toc42158149)

[3.3 Student Registration 2](#_Toc42158150)

[3.4 Teacher Registration 2](#_Toc42158151)

[3.5 Department 2](#_Toc42158152)

[3.6 Subject Detail 2](#_Toc42158153)

[3.7 Salary 2](#_Toc42158154)

[3.8 Exam Result 2](#_Toc42158155)

[3.9 Guardian Detail 2](#_Toc42158156)

[3.10 Library 3](#_Toc42158157)

[3.11 Staff 3](#_Toc42158158)

[3.12 Fee Account 3](#_Toc42158159)

[4. Entity Relation diagram 4](#_Toc42158160)

[5. Create Database and Tables / Insert Data 5](#_Toc42158161)

[5.1 Database 5](#_Toc42158162)

[5.2 Tables 6](#_Toc42158163)

[5.2.1 Students 6](#_Toc42158164)

[5.2.2 Teacher 7](#_Toc42158165)

[5.2.3 Student Registration 8](#_Toc42158166)

[5.2.4 Teacher Registration 9](#_Toc42158167)

[5.2.5 Guardian Detail 9](#_Toc42158168)

[5.2.6 Fee Account 10](#_Toc42158169)

[5.2.7 Department 11](#_Toc42158170)

[5.2.8 Staff 12](#_Toc42158171)

[5.2.9 Salary 12](#_Toc42158172)

[5.2.10 Library 13](#_Toc42158173)

[5.2.11 Exam Report 14](#_Toc42158174)

[5.2.12 Subject Detail 15](#_Toc42158175)

[6. Select Statement Using Different Function 16](#_Toc42158176)

[7. Select statements Using Sub Query 24](#_Toc42158177)

[8. Select Statements using Count and Group Functions 25](#_Toc42158178)

[9. Select Statements Using Different Joins 26](#_Toc42158179)

[10. Insert Statements 28](#_Toc42158180)

[11. Update Statements 30](#_Toc42158181)

[12. Delete Statements 32](#_Toc42158182)

[13. Normalization 34](#_Toc42158183)

[1. First Normal form (1NF) 34](#_Toc42158184)

[2. Second Normal form (2NF) 35](#_Toc42158185)

[3. Third Normal form (3NF) 35](#_Toc42158186)

School Management System

# Introduction

## Aim

Khimti Project School is one of the best institutes of education situated outside the Kathmandu valley, Kirne - Dolakha. This school is providing the best education services since 2064. The teacher in this school is highly educated and well experienced. Currently, there are about 500 students in the school. There are other lots of creative activities and sports which are carried out in this school.

The purpose of this case study is to provide database of all the data happen in school activities which includes the detail of student, teacher, fees, registration, different department, salary etc. It records such data to implement in school management. Further, it can be update if required.

# Description

Student identities and details are required in school for various purposes. Likewise, teacher details are equally required along with other staff. Then the fee should be paid by the students. Staff and teacher get salary monthly on the basis of their work. Guardian of the student details is also required to school. It organizes different activities like quiz game, spelling contest, speaking contest and other sport activities. It has different departments like music club, dance club, sports club, ECA etc. It also have library for books materials.

# Table Description

There are different tables which are described below along with their Constraint:

## 3.1. Students

This table represents the details of students. It stores the data by their id, name, age, gender and class.

Students are provided by the unique id to help to relate and identify easily.

**Constraint**: Student id will be unique for each student.

## 3.2 Teacher

This table stores the details of Teachers working in the school. It stores their name, their related subject and phone number. Each teacher is provided with unique id.

**Constraint:** Teacher id is uniquely set and subject id must have a corresponding value in subject table.

## 3.3 Student Registration

This table stores the attendance of students. It has column of total days with present days of students and absent days.

**Constraint:** Student registration id must have unique. Student id must have a corresponding value in students table.

## 3.4 Teacher Registration

This table stores the attendance of teacher. It has column of total days with present days of students and absent days.

**Constraint:** Teacher registration id must have unique. Teacher id must have a corresponding value in teacher table.

## 3.5 Department

This table is for the record of the different department present in school.

**Constraint:** Department id is unique for each department.

## 3.6 Subject Detail

This table stores subject which are studied in school. It also gives the each subject a specific code.

**Constraint:** Subject detail must have unique id for each subject.

## 3.7 Salary

This table is for the salary distribution for teacher of the school. It records the monthly salary distributed to them and other bonuses as well.

**Constraint:** Salary id must be unique for each staff and it contain corresponding teacher id to distribute.

## 3.8 Exam Result

This table stores the result of students. It has subject id, exam id, classes of students, their GPA and remarks according to their performance.

**Constraint:** Exam report id must have unique id for each students report.

## 3.9 Guardian Detail

This table shows the details of parents. It has guardian id, student id respective from students table, name, address and phone number.

**Constraint:** Guardian id must have unique id.

## 3.10 Library

This table stores the data regarding books from library. It records the name of books, category of book, issued date, expire date and students id.

**Constraint:** Library id must be unique for each book issued be the student.

## 3.11 Staff

This table stores data regarding other staff except teacher of school. It stores their name and department id.

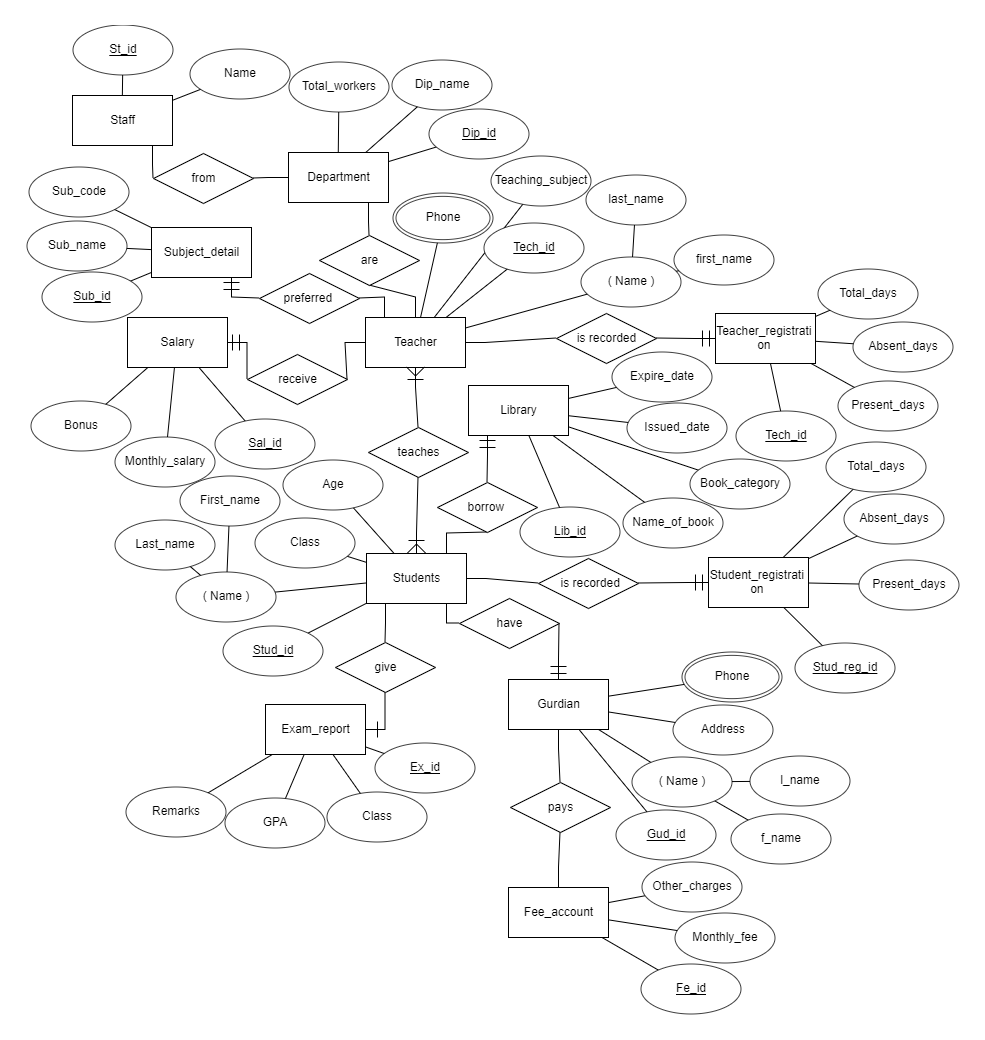
**Constraint:** Staff id must be unique for each staff.

## 3.12 Fee Account

This table is for the accounts for the students for their fees. It records the monthy fees of each student and other charges as well.

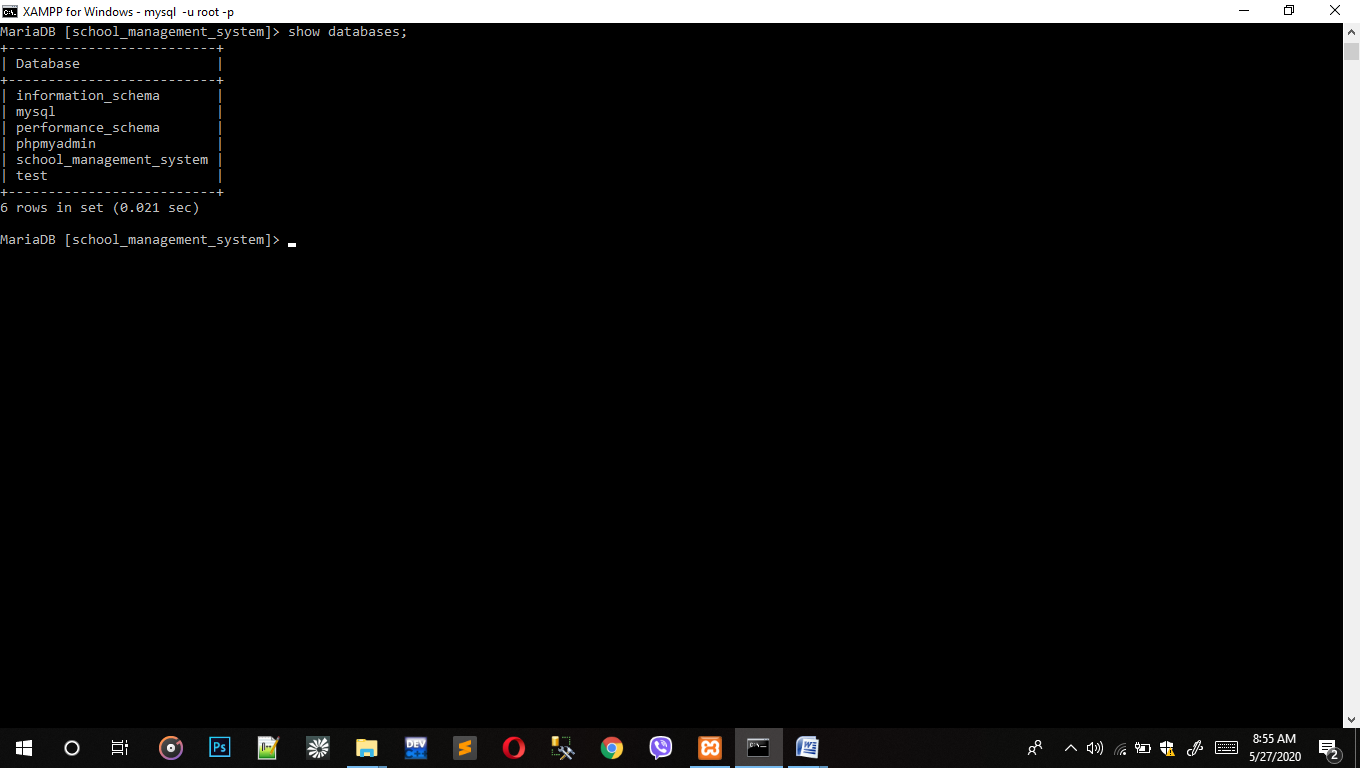
**Constraint:** Fee account must have unique id for each transaction.

# Entity Relation diagram

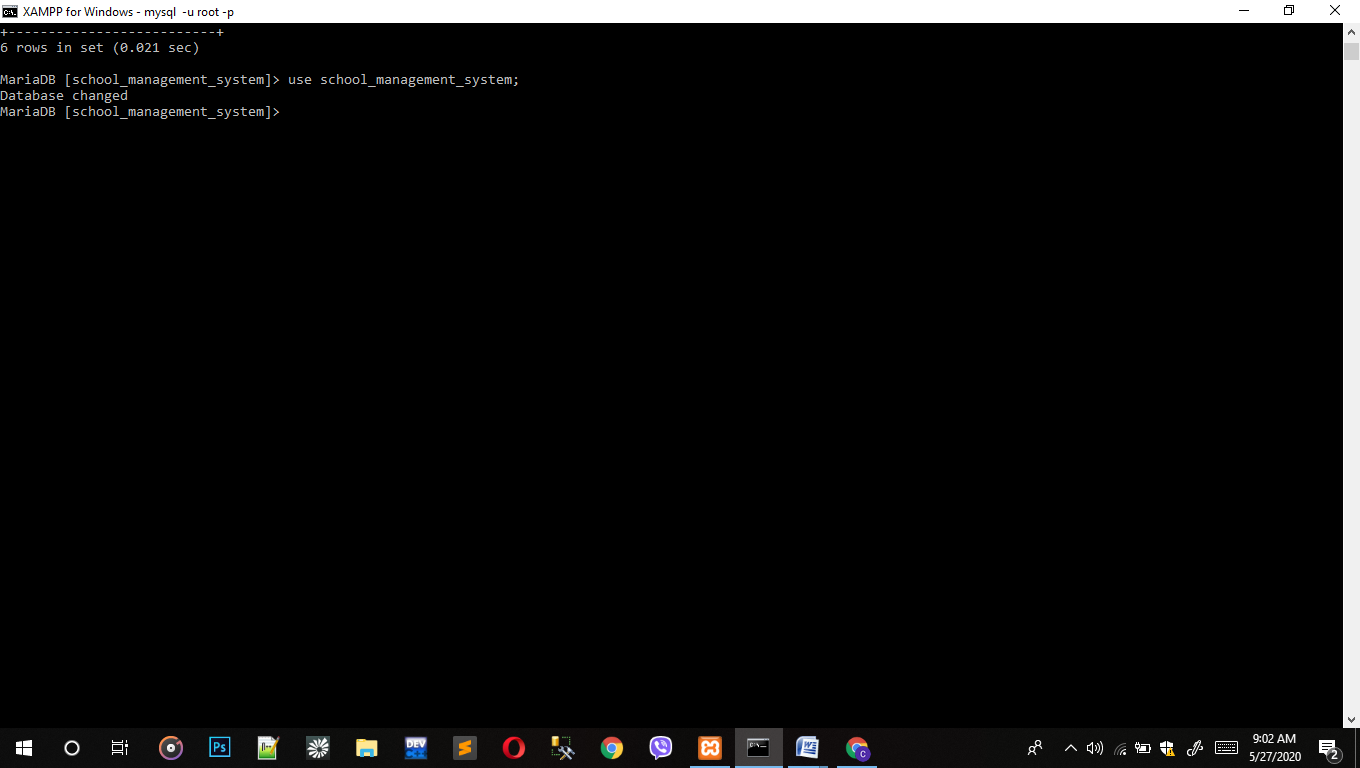


# 5. Create Database and Tables / Insert Data

## 5.1 Database



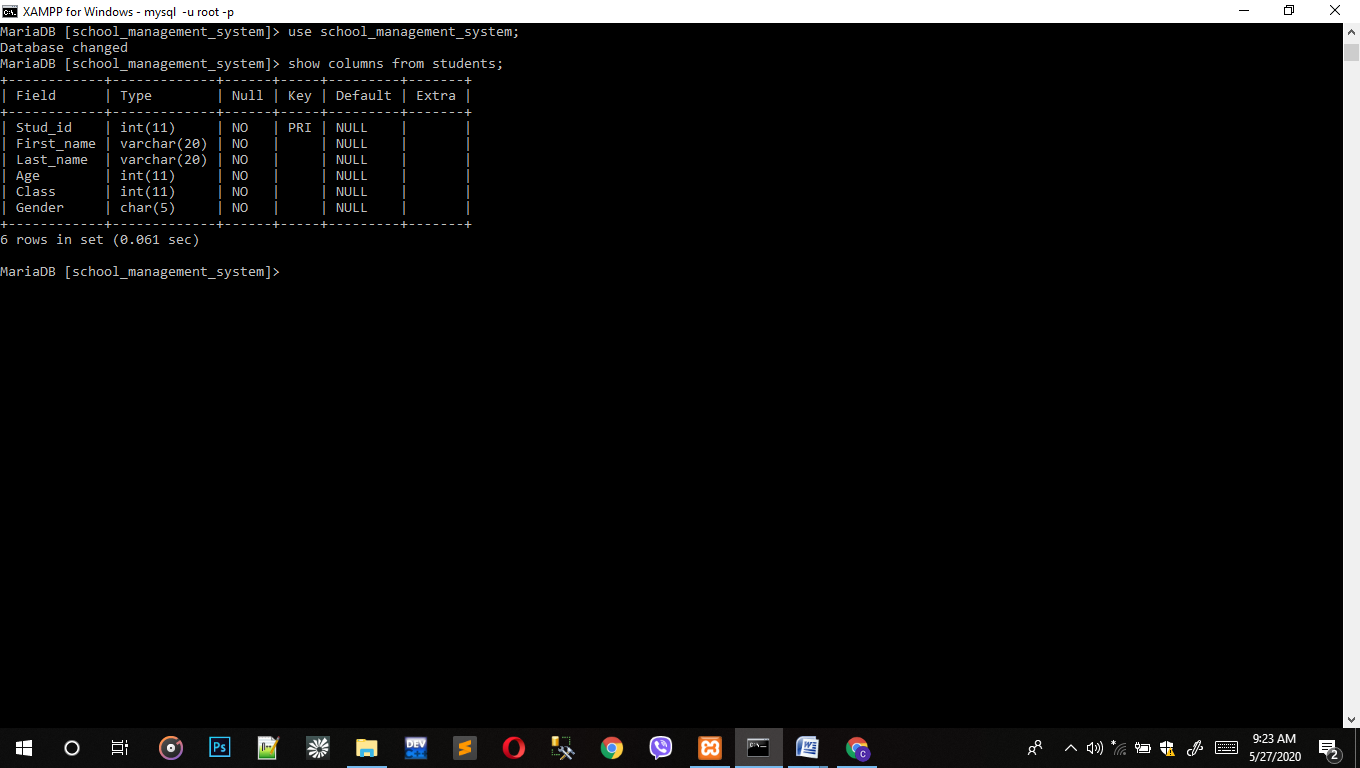
Syntax: Create database school\_management\_system;



Syntax: use school\_management\_system;

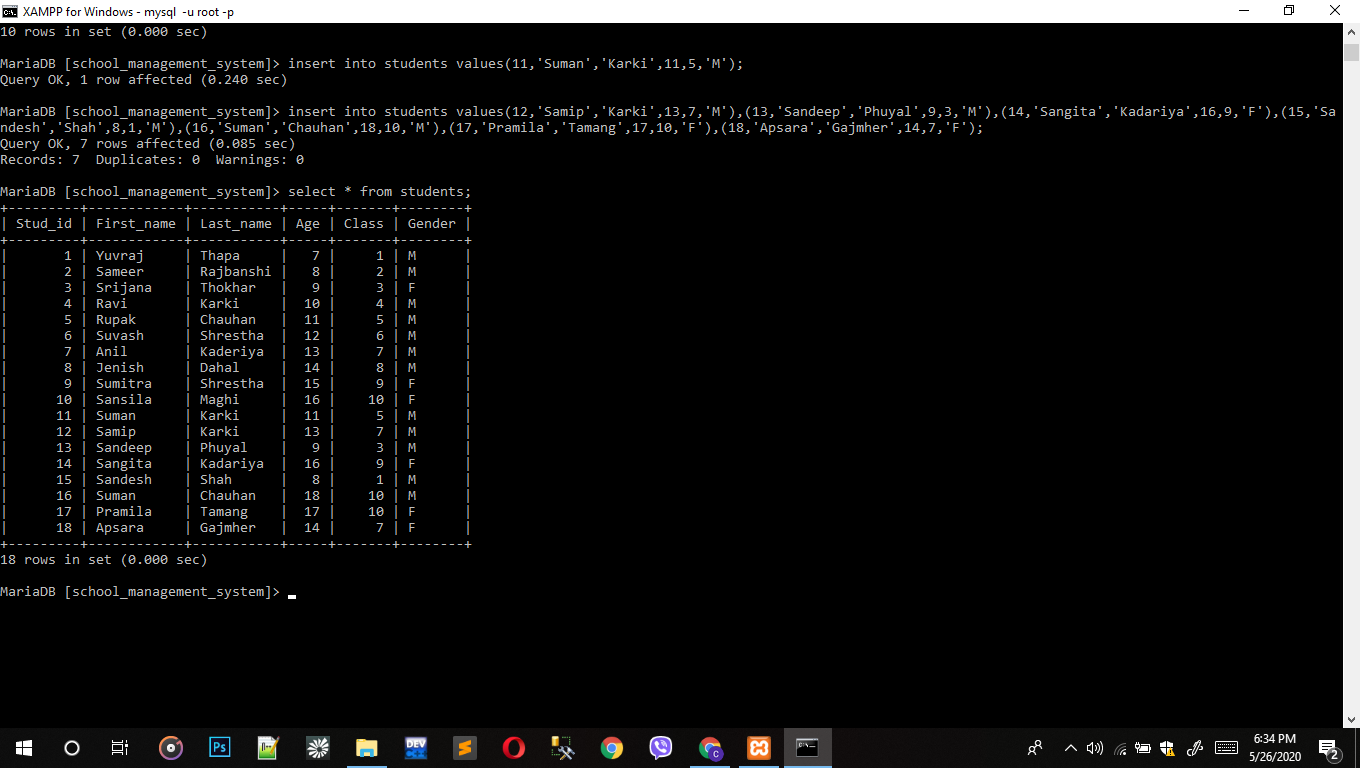
## 5.2 Tables

### 5.2.1 Students

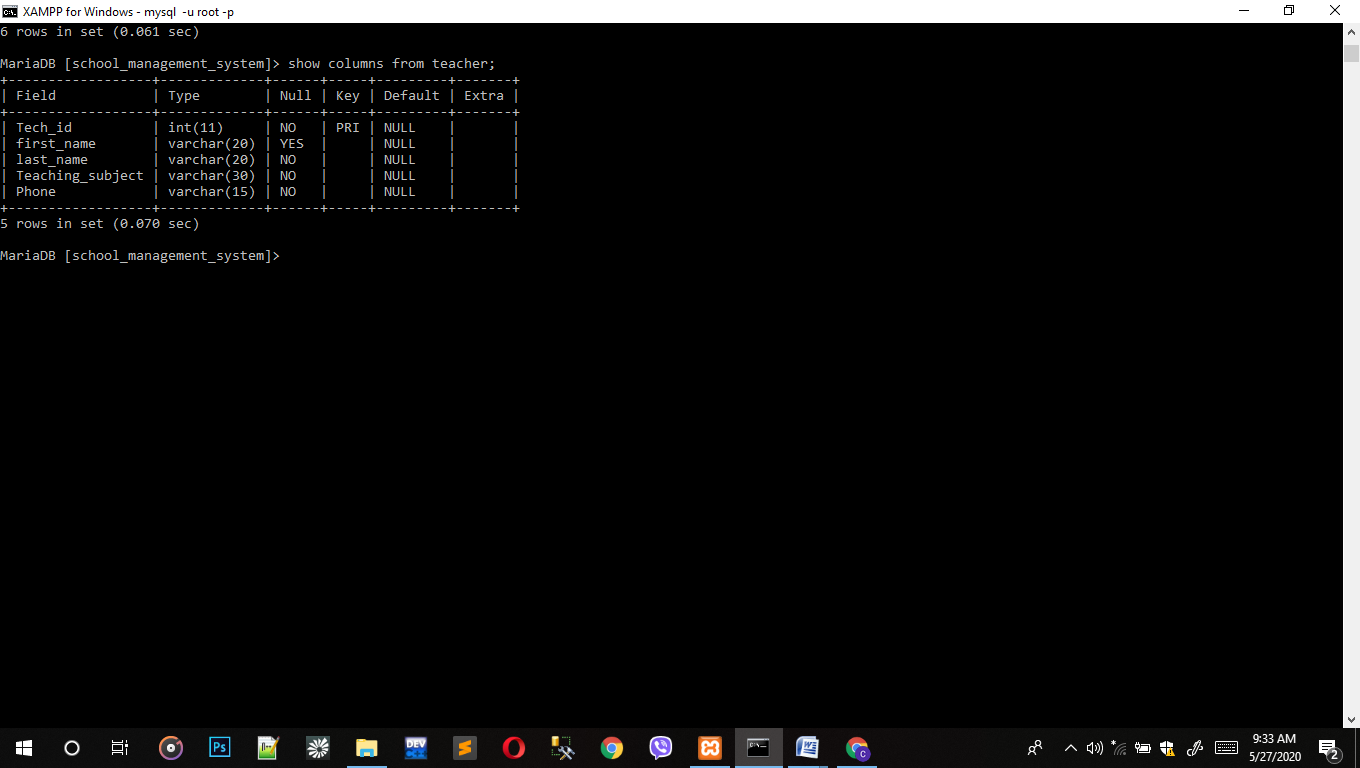


Syntax: Create table Students(Stud\_id int not null primary key, First\_name varchar(20) not null, Last\_name varchar(20) not null, Age int not null, Class int not null, Gender char(5) not null);

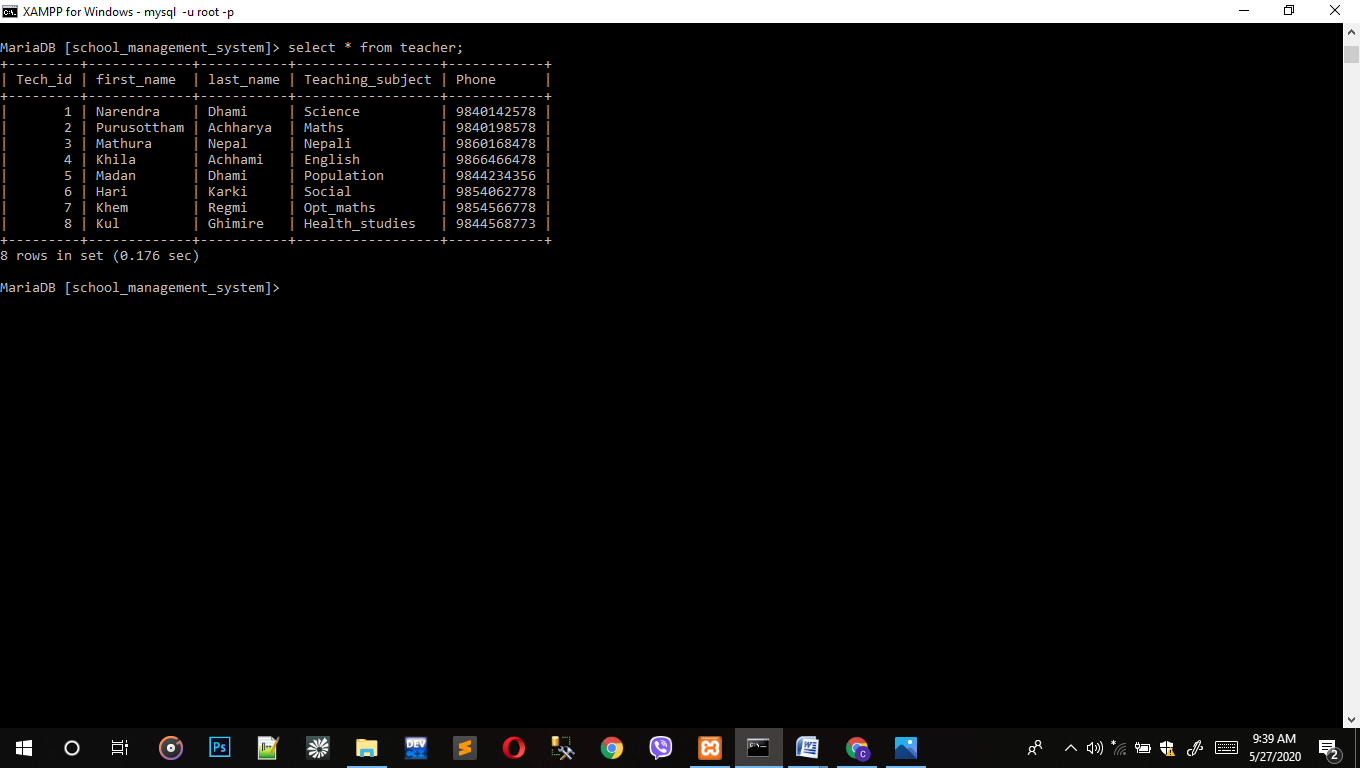
Syntax: insert into students values(1,'Yuvaraj','Thapa',7, 1,'M'), (2,'Sameer','Rajbanshi', 8,2,'M'), (3,'Srijana','Thokhar',9, 3,'F'), (4,'Ravi','Karki',10, 4,'M'), (5,'Rupak','Chauhan',11,5,'M'), (6,'Suvash','Shrestha',12, 6,'M'), (7,'Anil','Kadariya',13, 7,'M'), (8,'Jenish','Dahal',14, 8,'M'), (9,'Sumitra','Shrestha',15, 9,'F'), (10,'Sansila','Maghi',16, 10,'F'), (11,'Suman','Karki',11, 5,'M'), (12,'Samip','Karki',13, 7,'M'), (13,'Sandesh','Shah',8, 1,'M'), (14,'Sandeep','Phuyal',9, 3,'M'), (15,'Sangita','Kaderiya',16, 9,'F'), (16,'Suman','Chauhan',18, 10,'M'), (17,'Pramila','Tamag',17, 10,'F'),(18, 'Apsara','Gajmher',14,7,'F');



### 5.2.2 Teacher



Syntax: Create table Teacher(Tech\_id int not null primary key, first\_name varchar(20) not null, last\_name varchar(20) not null, Teaching\_subject varchar(30), not null, Phone varchar(15) not null);

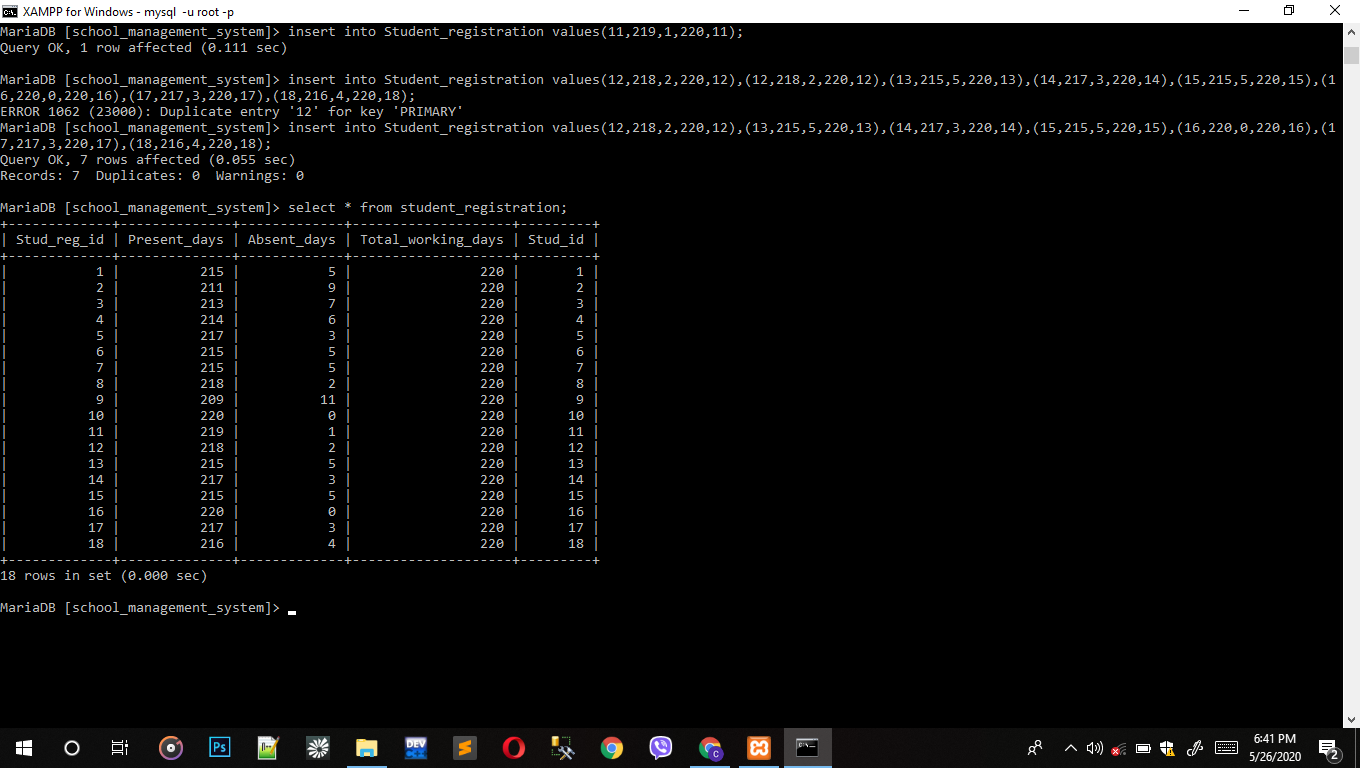


Syntax: insert into teacher values(1,'Narendra','Dhami','Science',9840142578), (2,'Purusottham','Achharya','Maths',9840198578), (3,'Mathura','Nepal','Nepali',9860168478), (4,'Khila','Achhami','English',9866466478), (5,'Madhan','Dhami','Population',9844234356), (6,'Hari','Karki','Social',9854062778), (7,'Khem','Regmi','Opt\_maths',9854566778), (8,'Khul',Ghimire','Health\_studies',984456873);

### 5.2.3 Student Registration

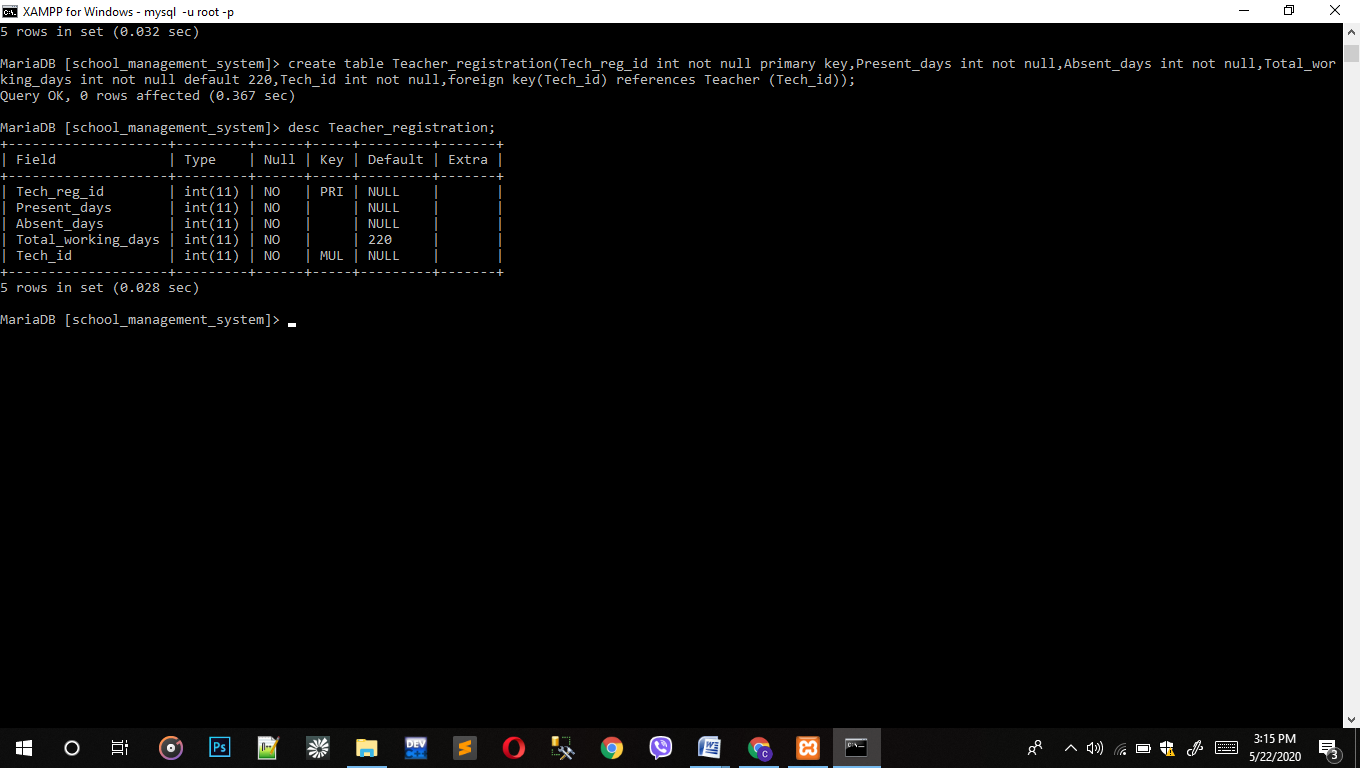


Syntax: create table Student\_registrstion(Stud\_reg\_id int not null primary key,Present\_days int not null, Absent\_days int not null,Total\_working\_days int not null,Stud\_id int not null, foreign key(Stud\_id) references Students(Stud\_id));

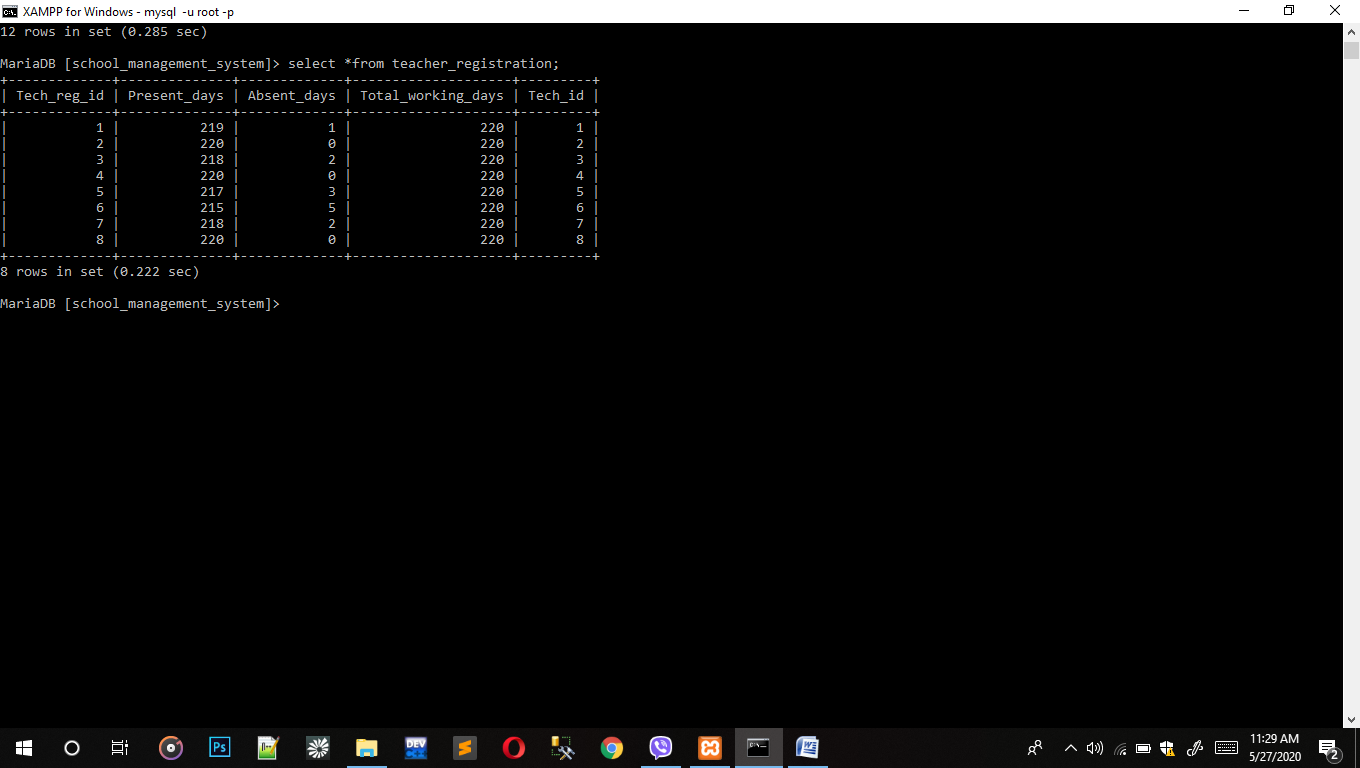


Syntex: insert into student\_registration values(1,215,5,220,1), (2,211,9,220,2), (3,213,7,220,3), (4,214,6,220,4), (5,217,3,220,5), (6,215,5,220,6), (7,215,3,220,7), (8,218,2,220,8), (9,209,11,220,9), (10,220,0,220,10), (11,219,1,220,11), (12,218,2,220,12), (13,215,5,220,13), (14,217,3,220,17), (15,215,5,220,15), (16,220,0,220,16), (17,217,3,220,17), (18,216,4,220,18);

### 5.2.4 Teacher Registration

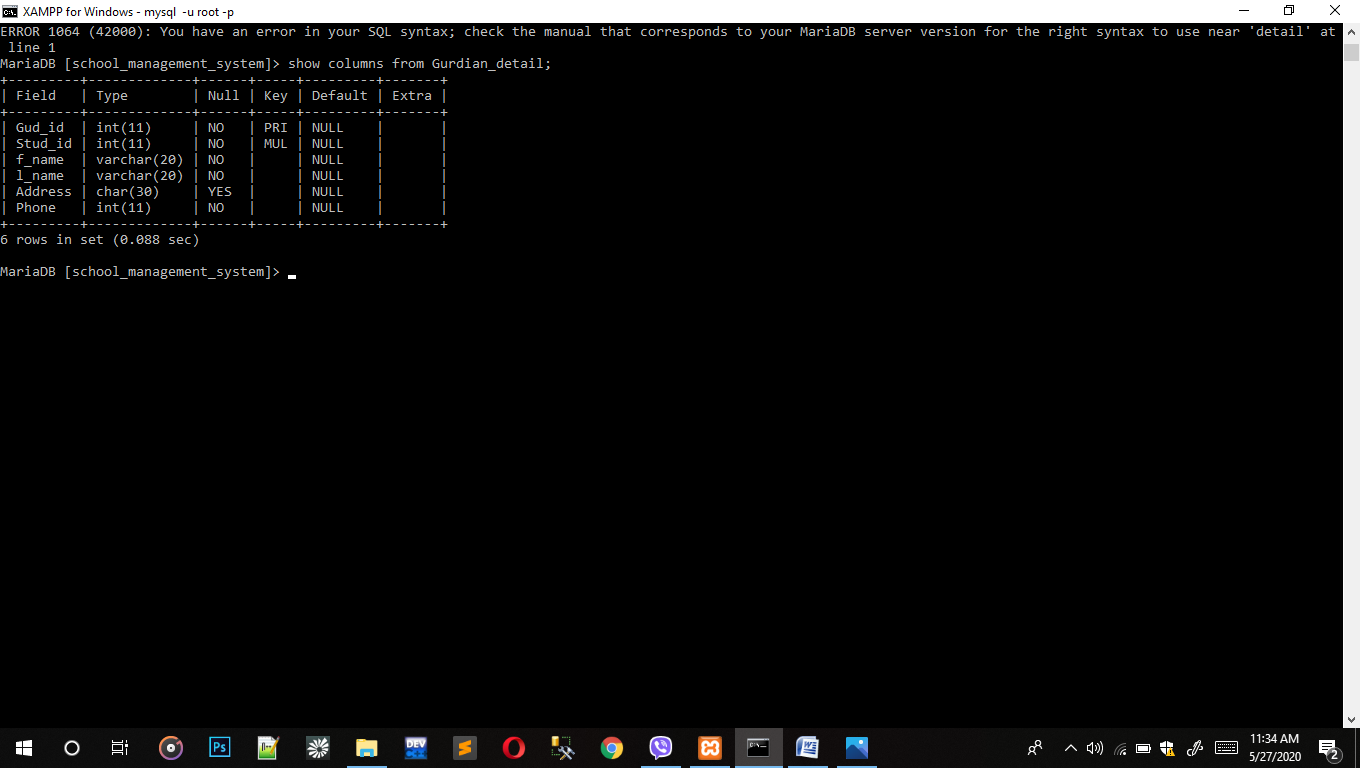


Syntax: create table Teacher\_registration(Tech\_reg\_id int not null primary key, Present\_days int not null, Absent\_days int not null,Total\_working\_days int not null, Tech\_id int not null, foreign key(Tech\_id) references Students(Tech\_id));

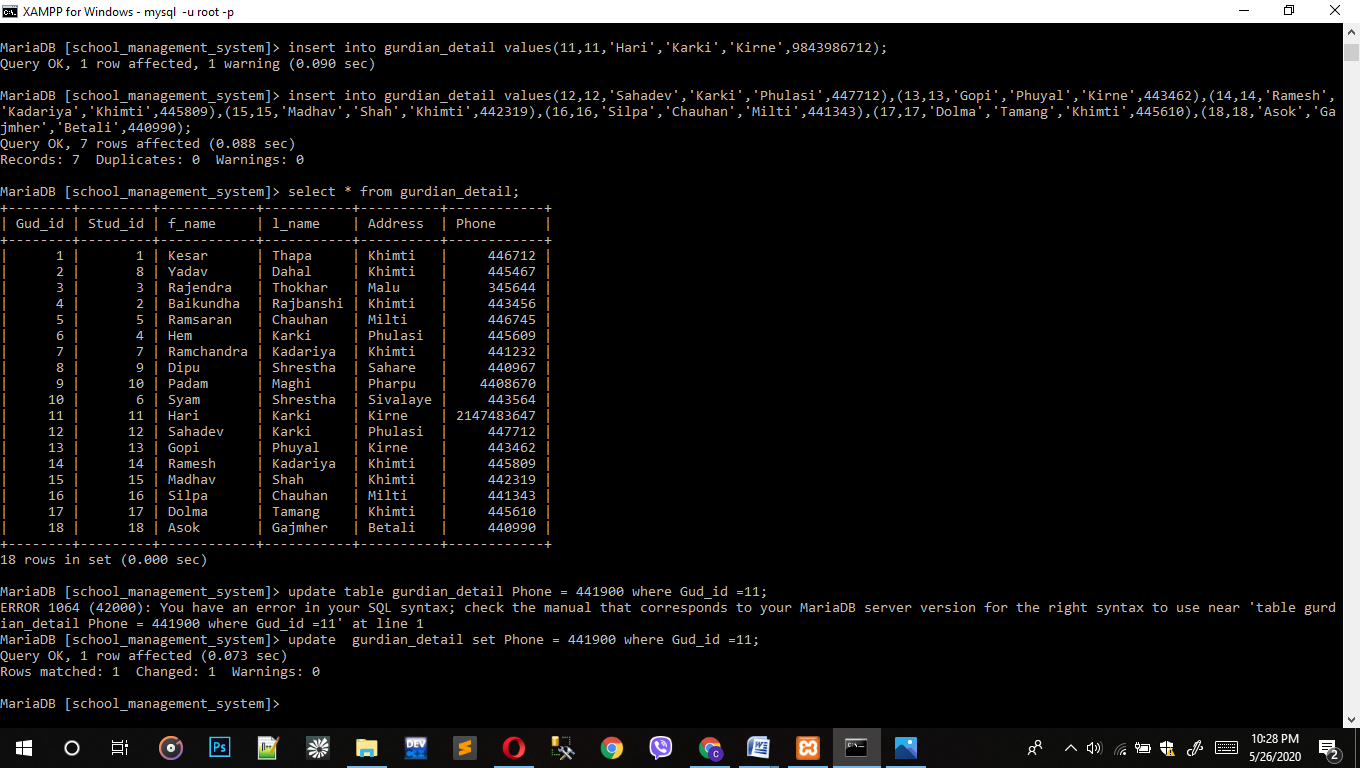


Syntax: insert into teacher\_registratiom values(1,219,1,220,1), (2,220,0,220,2), (3,218,2,220,3), (4,220,0,220,4), (5,215,5,220,5), (6,215,5,220,6), (7,2180,2,220,7), (8,220,0,220,8);

### 5.2.5 Guardian Detail

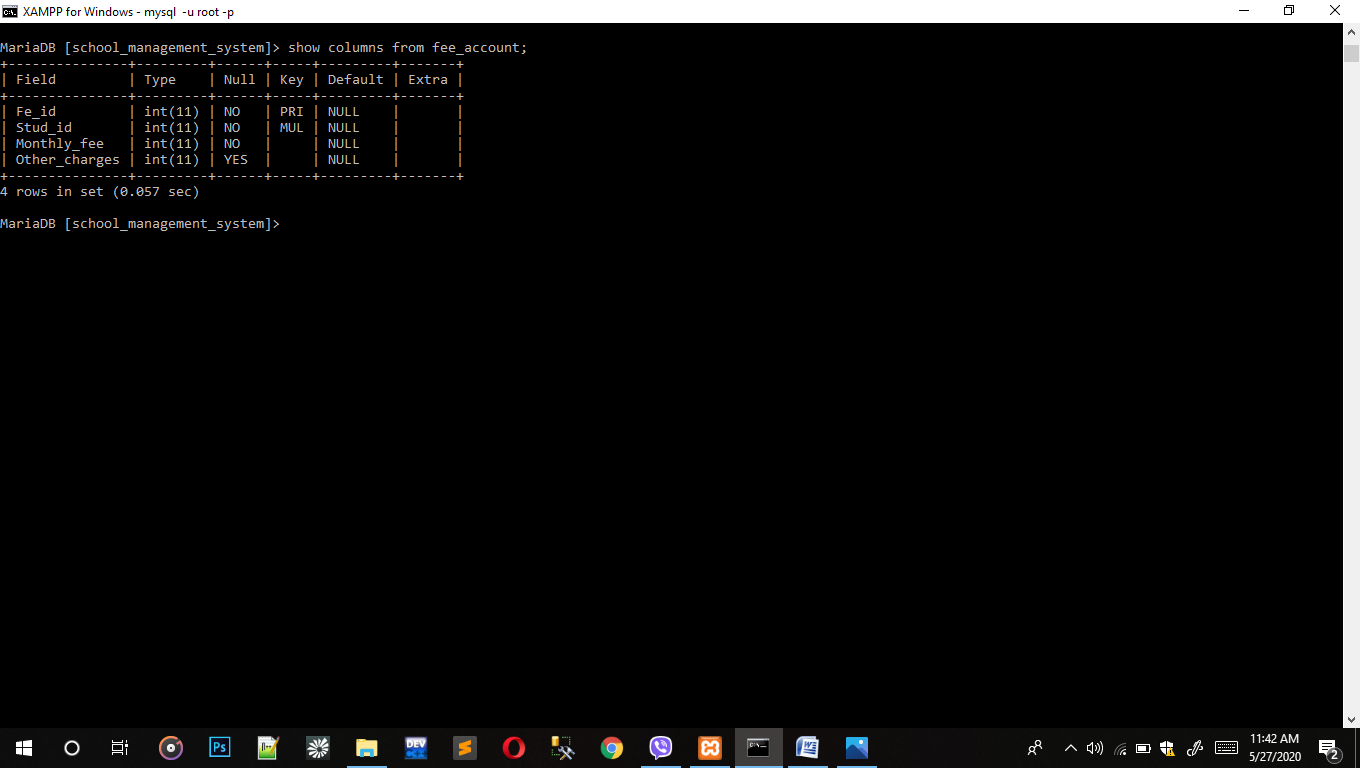


Syntax: create table Gurdian\_detail(Gud\_id int not null primary key, Stud\_id int not null, f\_name varchar(20), l\_name varchar(20) not null, Address char(30) null, phone int not null, foreign key(Stud\_id) references Students(Stud\_id));

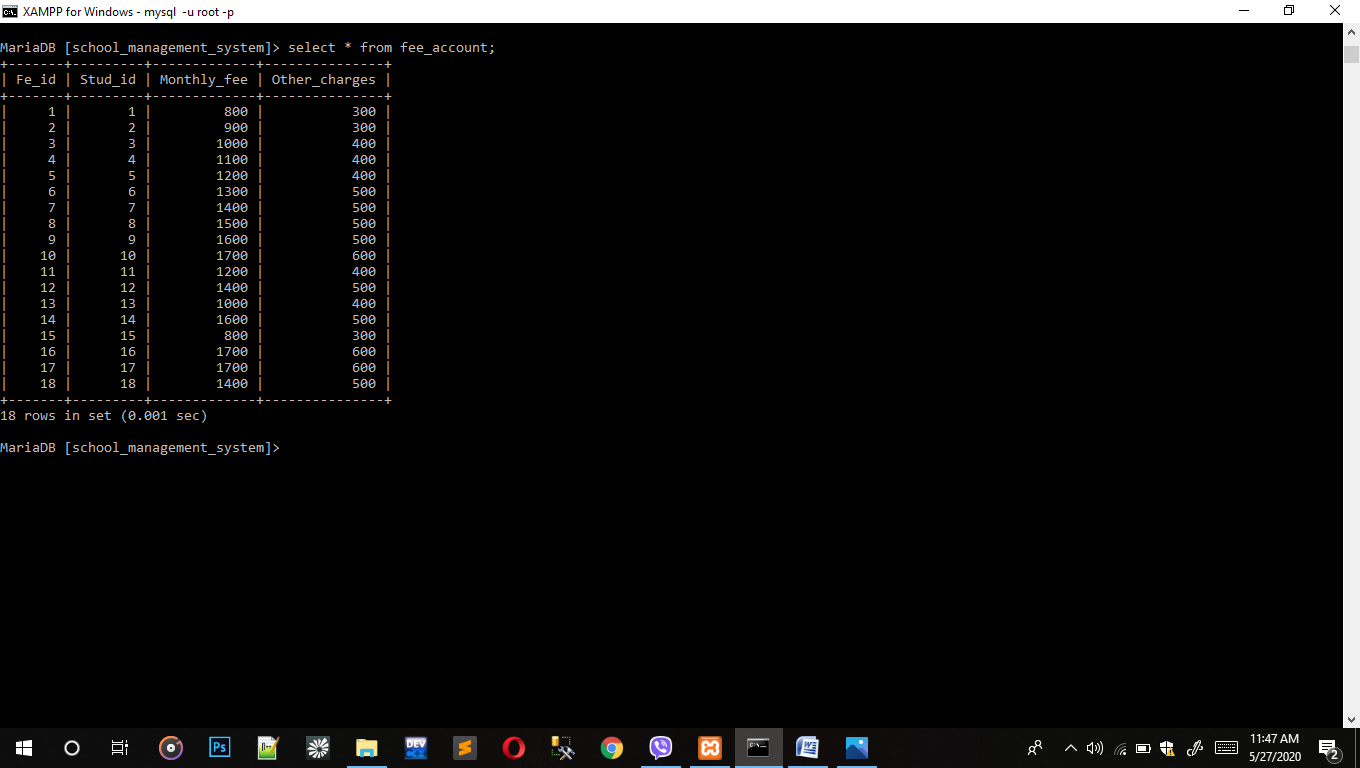


Syntax: insert into guardian\_detail values (1,1,'Kesar','Thapa','Khimti',446712), (2,8,'Yadav','Dahal','Khimti',445467),(3,3,'Rajendra','Thokhar','Malu',345644),(4,2,'Baikhunda','Rajbanshi','Khimti',443456),(5,5,'Ramsaran','Chauhan','Milti',446745),(6,4,'Hem','Karki','Phulasi',445609),(7,7,'Ramchandra','Kadariya','Khimti',441232),(8,9,'Dipu','Shrestha','Sahare',440967),(9,10,'Padam','Maghi','Pharpu',4408670),(10,6,'Syam','Shrestha','Sivalaye',443564),(11,11,'Hari','Karki','Kirne',441900),(12,12,'Sahadev','Karki','Phulasi',447712),(13,13,'Gopi','Phuyal','Kirne',443462),(14,14,'Ramesh','Kadariya','Khimti',445809),(15,15,'Madav','Shah','Khimti',442319),(16,16,'Silpa','Chauhan','Milti',441343),(17,17,'Dolma','Tamang','Khimti',445610),(18,18,'Asok','Gajmher','Betali',440990);

### 5.2.6 Fee Account



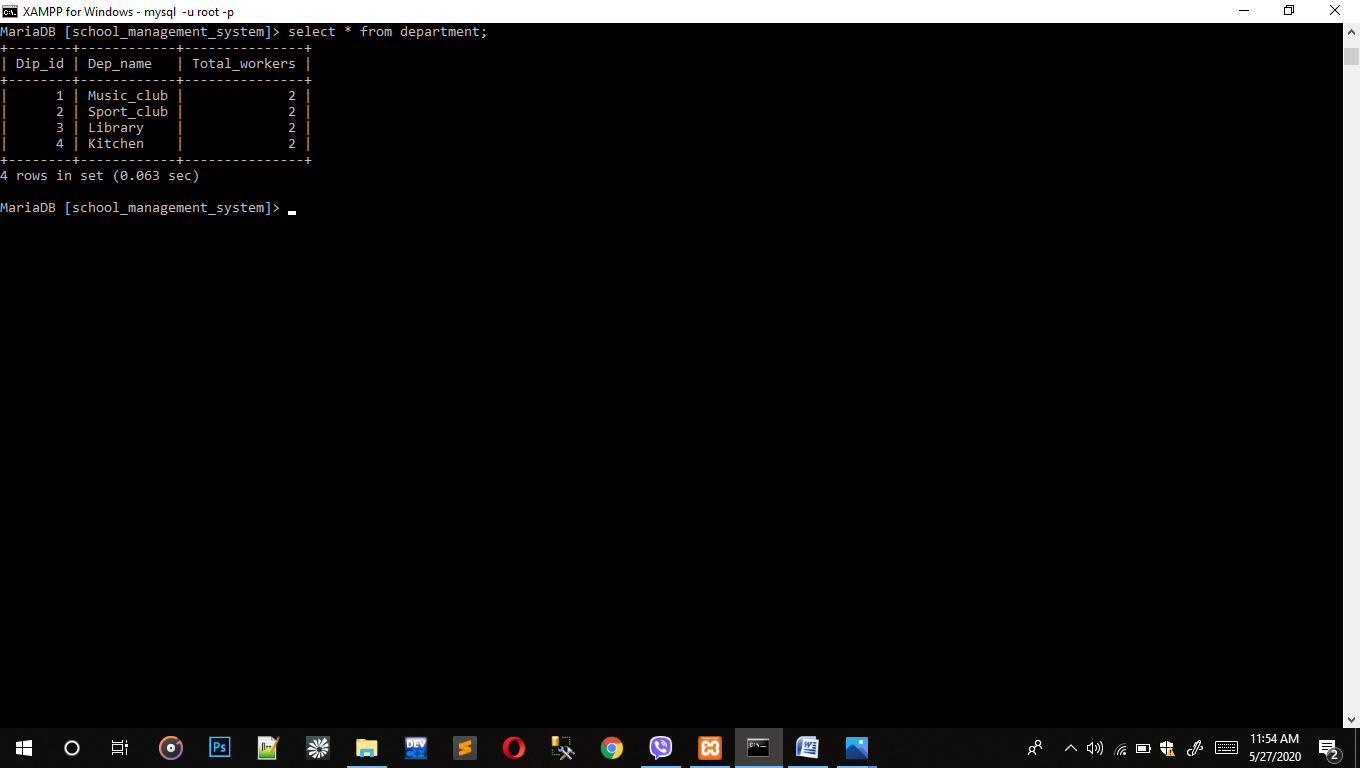
Syntax: create table fee\_account(fe\_id int not null primary key, stud\_id int not null, monthly\_fee int not null, other\_charges int not null, foreign key(Stud\_id) references Students(Stud\_id));



Syntax: insert into fee\_account(1,1,800,300),(2,2,900,300), (3,3,1000,300), (4,4,1000,400), (5,5,1200,400),(6,6,1300,500), (7,7,1400,500), (8,8,1500,500),(9,9,1600,500),(10,10,1700,300), (11,11,1200,400), (12,12,1400,500), (13,13,1000,400), (1,414,1600,500), (15,15,800,300), (16,16,1700,600), (17,17,1700,600), (18,18,1400,500);

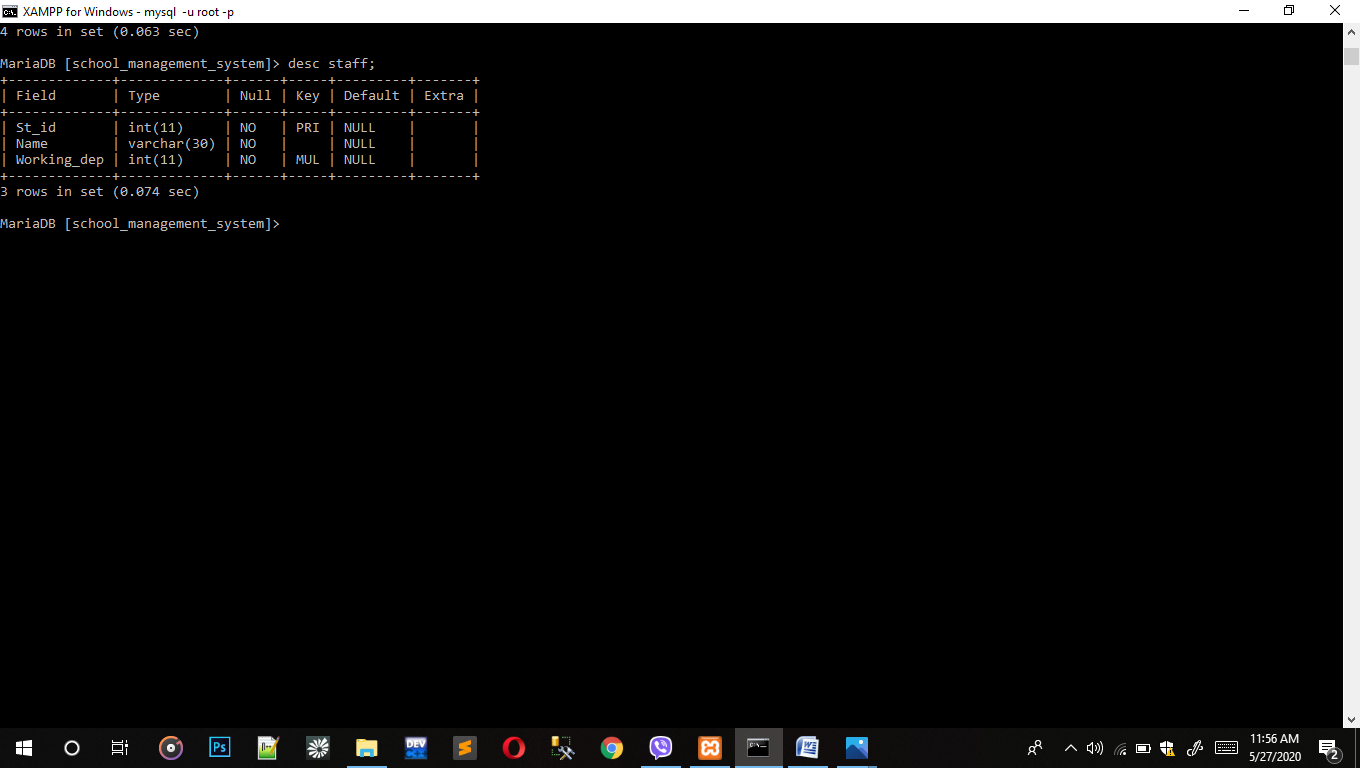
### 5.2.7 Department

Syntax: create table department(dip\_id int not null primary key, dep\_name varchar(20) null, total\_workers int not null);

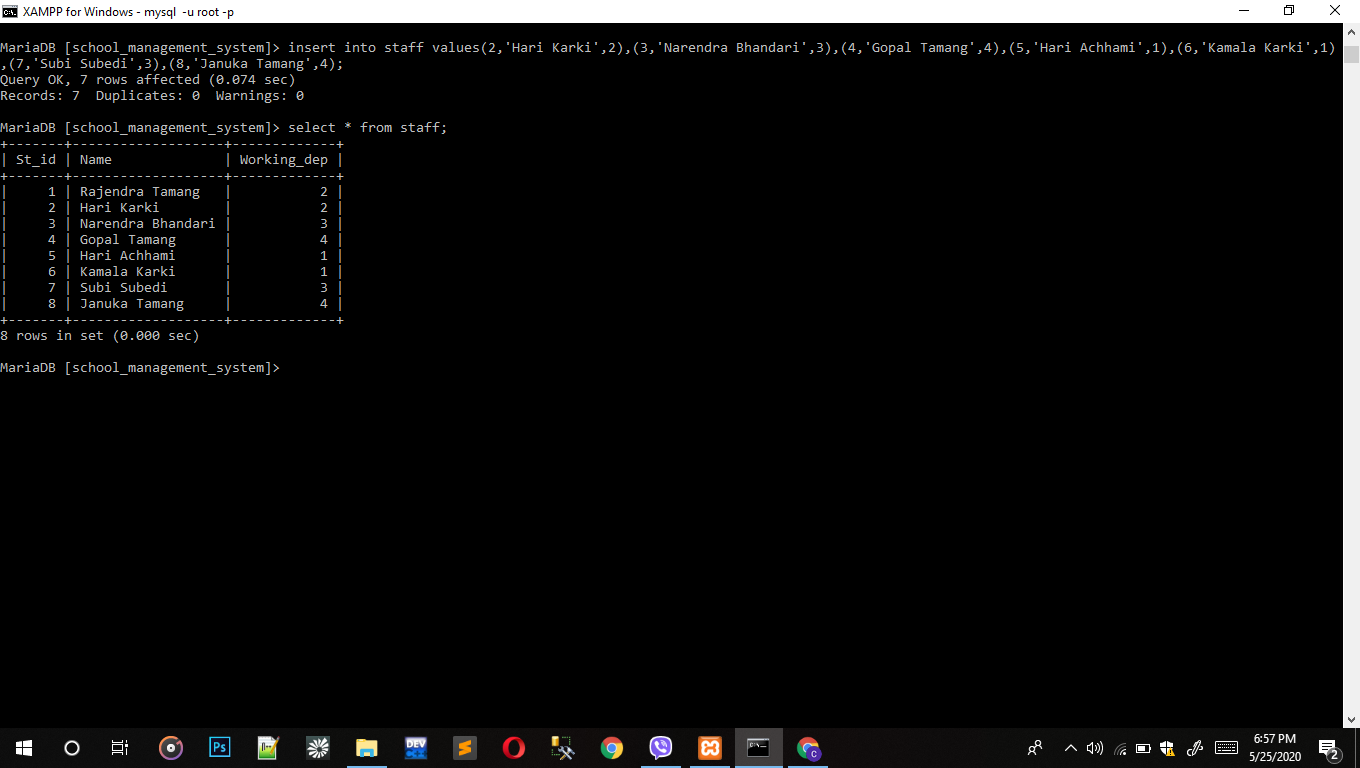


Syntax: insert into department values(1,'Music\_club',2), (2,'Sport\_club',2), (3,'Library',3), (4,'Kitchen',2);

### 5.2.8 Staff

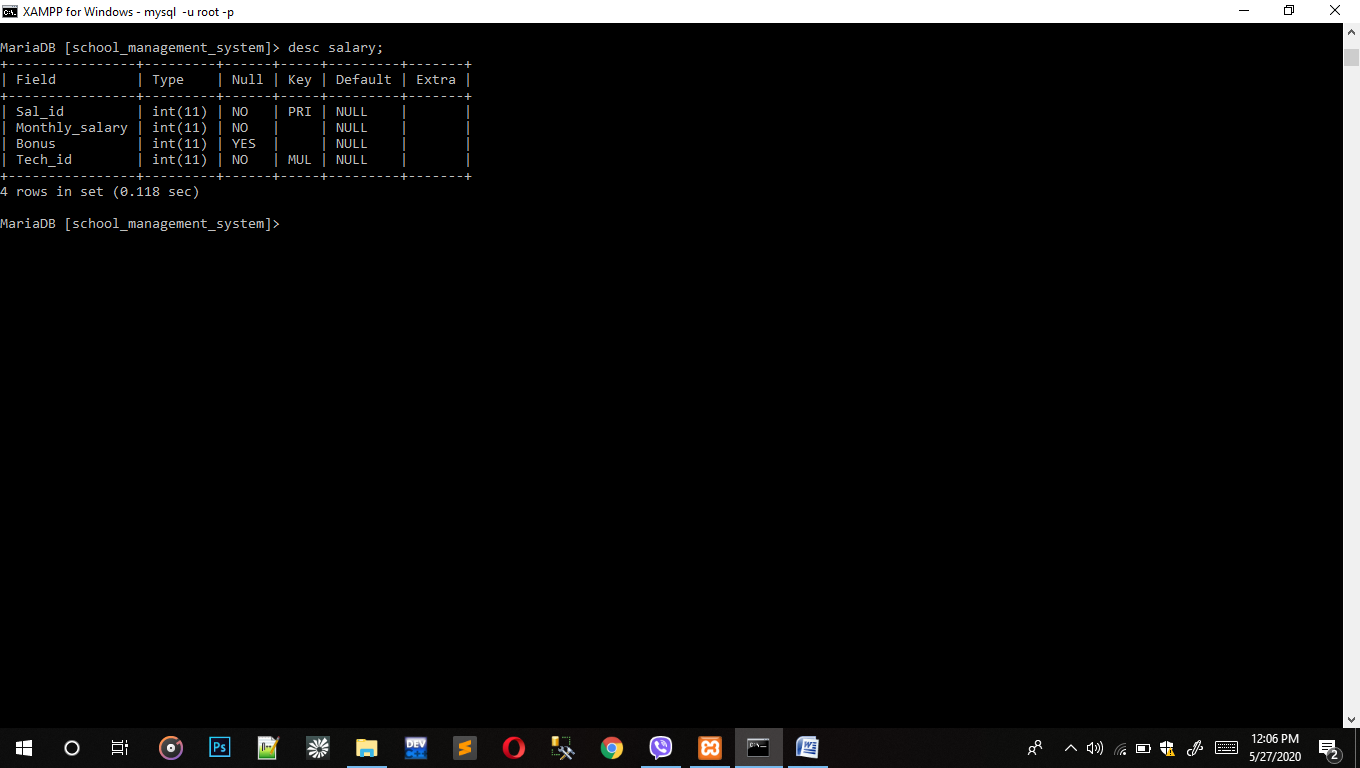


Syntax: create table staff(st\_id int not null primary key, name varchar(30) not null, working\_dep int not null,foreign key(working\_dep) references department(Dep\_id));

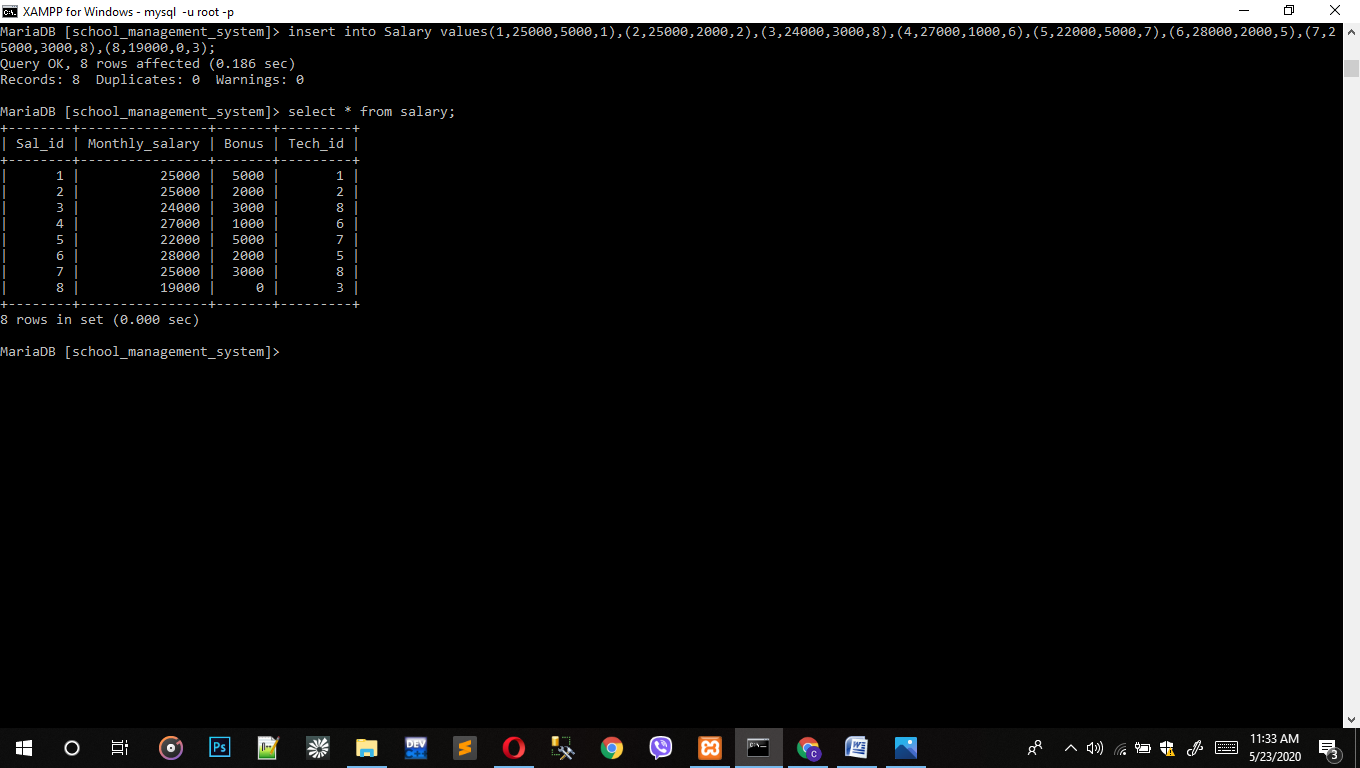


Syntax: insert into staff values(1,'Rajendra','Tamang',2), (2,'Hari','Karki',2), (3,'Narendra','Bhandari',3), (4,'Gopa','Tamang',4), (5,'Hari','Achhami',1), (6,'Kamala','Karki',1), (7,'Subi','Subedi',3), (8,'Januka','Tamang',4);

### 5.2.9 Salary

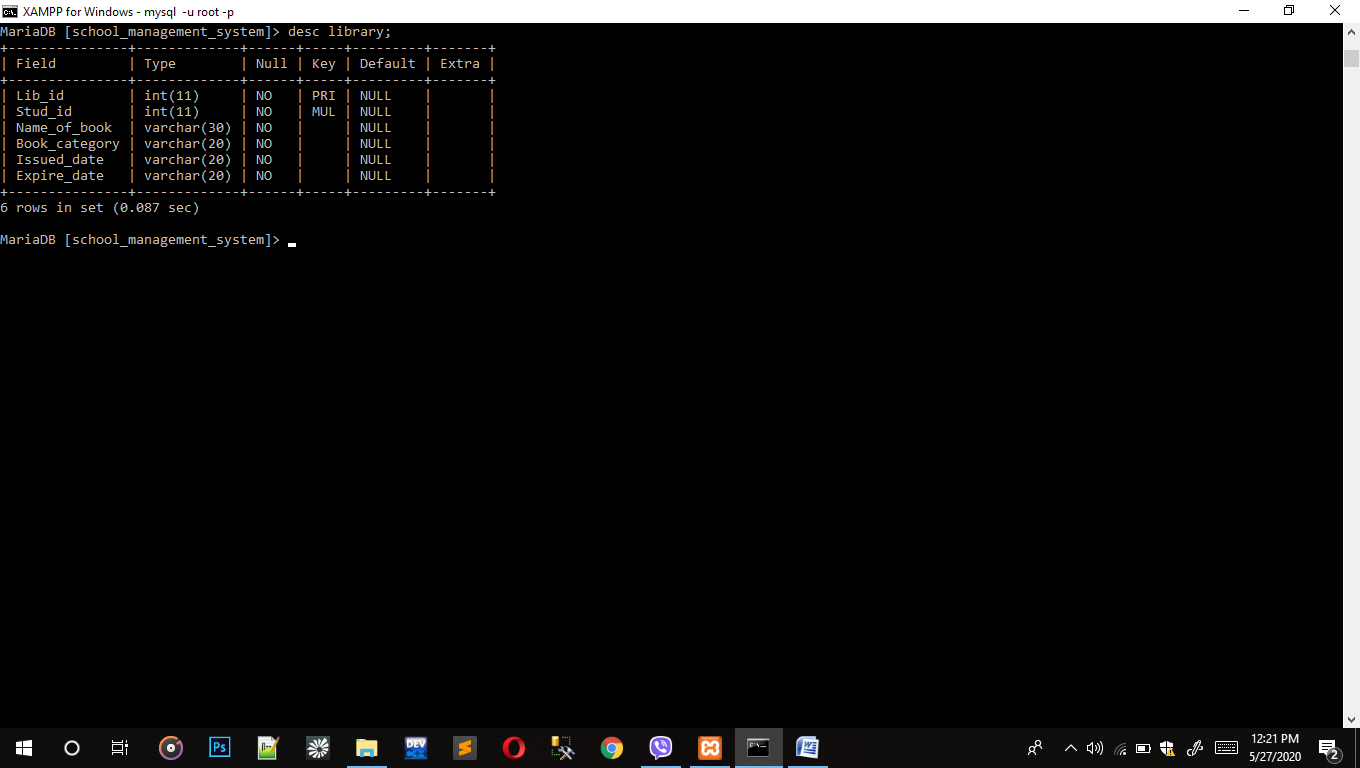


Syntax: create table salary(sal\_id int not null primary key, monthly\_salary int not null, bonus int not null, tech\_id int not null, foreign key(Tech\_id) references Teacher(Tech\_id));

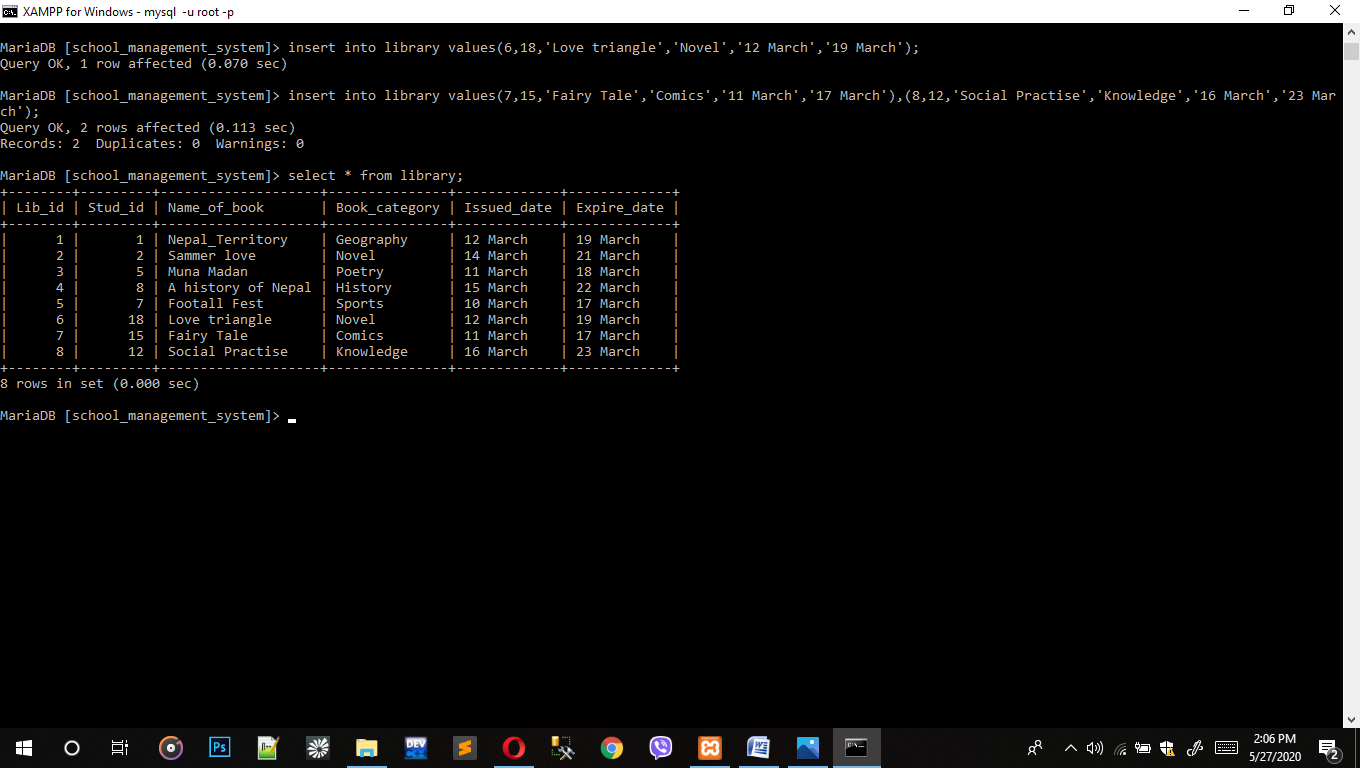


Syntax: insert into salary values(1,25000,5000,1), (2,25000,2000,2), (3,24000,3000,8), (4,27000,1000,6), (5,22000,5000,7), (6,28000,2000,5), (7,25000,3000,8), (8,19000,0,3);

### 5.2.10 Library

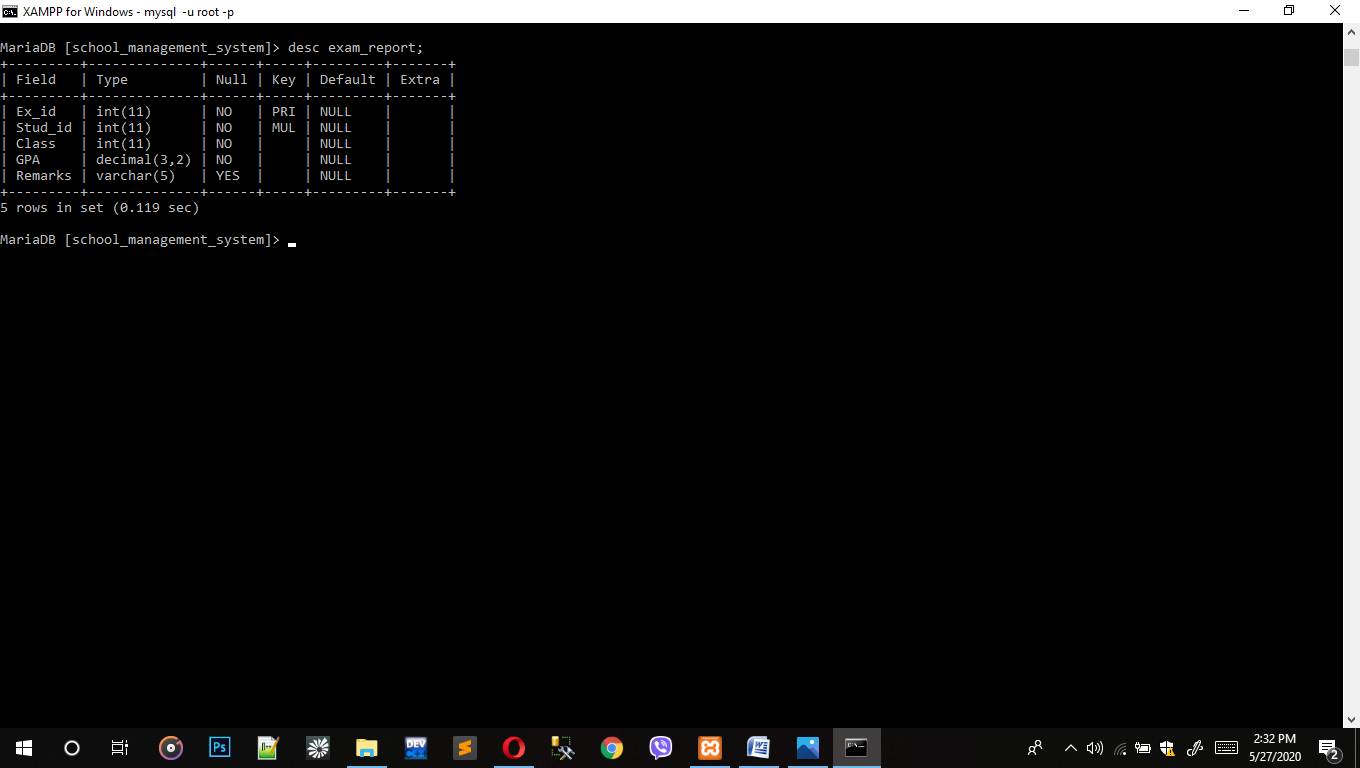


Syntax: create table library(lib\_id int not null primary key, stud\_id int not null, name\_of \_book varchar(30) not null, book\_category varchar(20) not null, issued date int not null, expire\_date varchar(20) not null, foreign key(Stud\_id) references Students(Stud\_id));

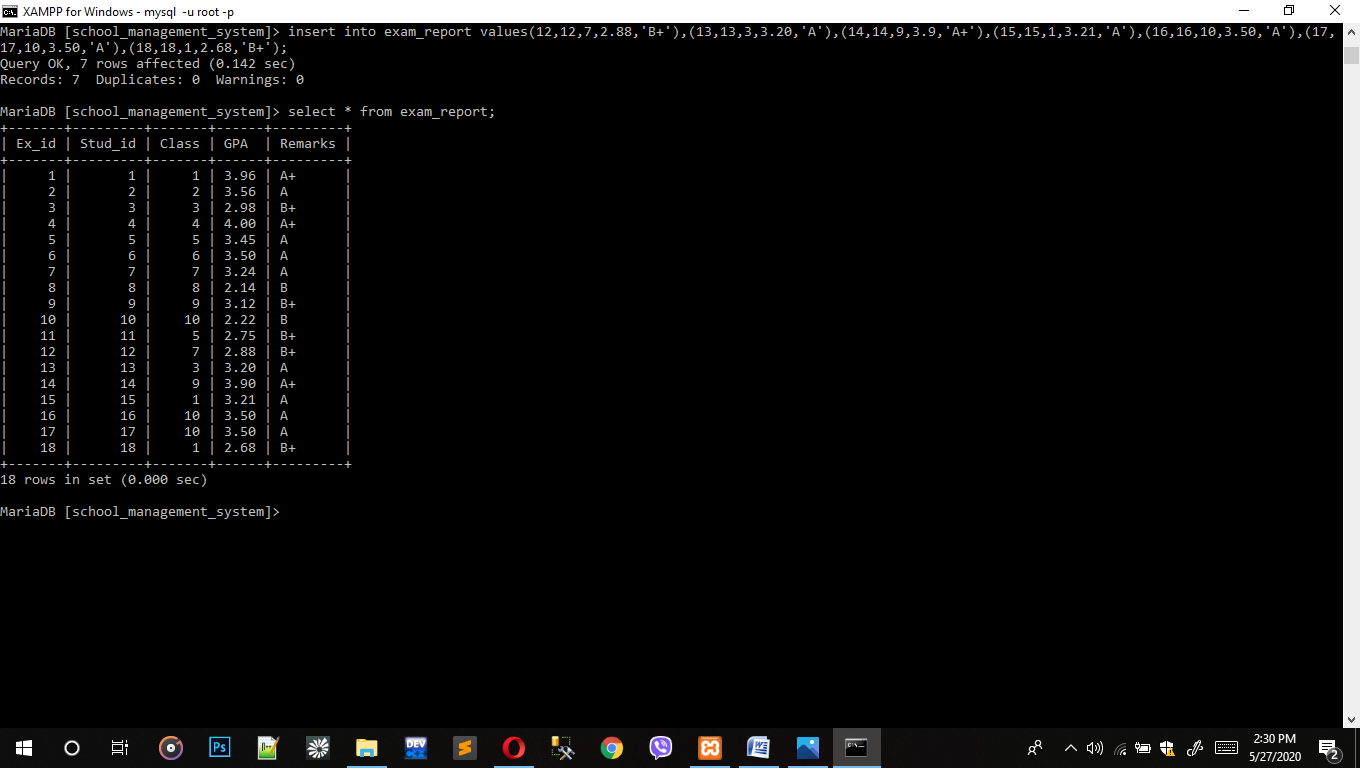


Syntax: insert into library values(1,1,'Nepal\_Territory,'Geography','12 March','19 March'), (2,2,'Summer love','Novel','14 March','21 March'),(3,5,'Muna Madan','Poetry','11 March','18 March'), (4,8,'A history of Nepal','History','15 March','22 March'), (5,7,'Football fest','Sports','10 March','17 March'), (6,18,'Love triangle','Novel','12 March','19 March'), (7,15,'Fairy Tale,'Comics','11 March','18 March'), (8,12,'Social practise','Knowledge','16 March','23 March'),

### 5.2.11 Exam Report



Syntax: create table exam\_report(ex\_id int not null primary key, stud\_id int not null, class int not null,gpa decimal(3,2) not null, remarks varchar(5) null, foreign key(Stud\_id) references Students(Stud\_id));

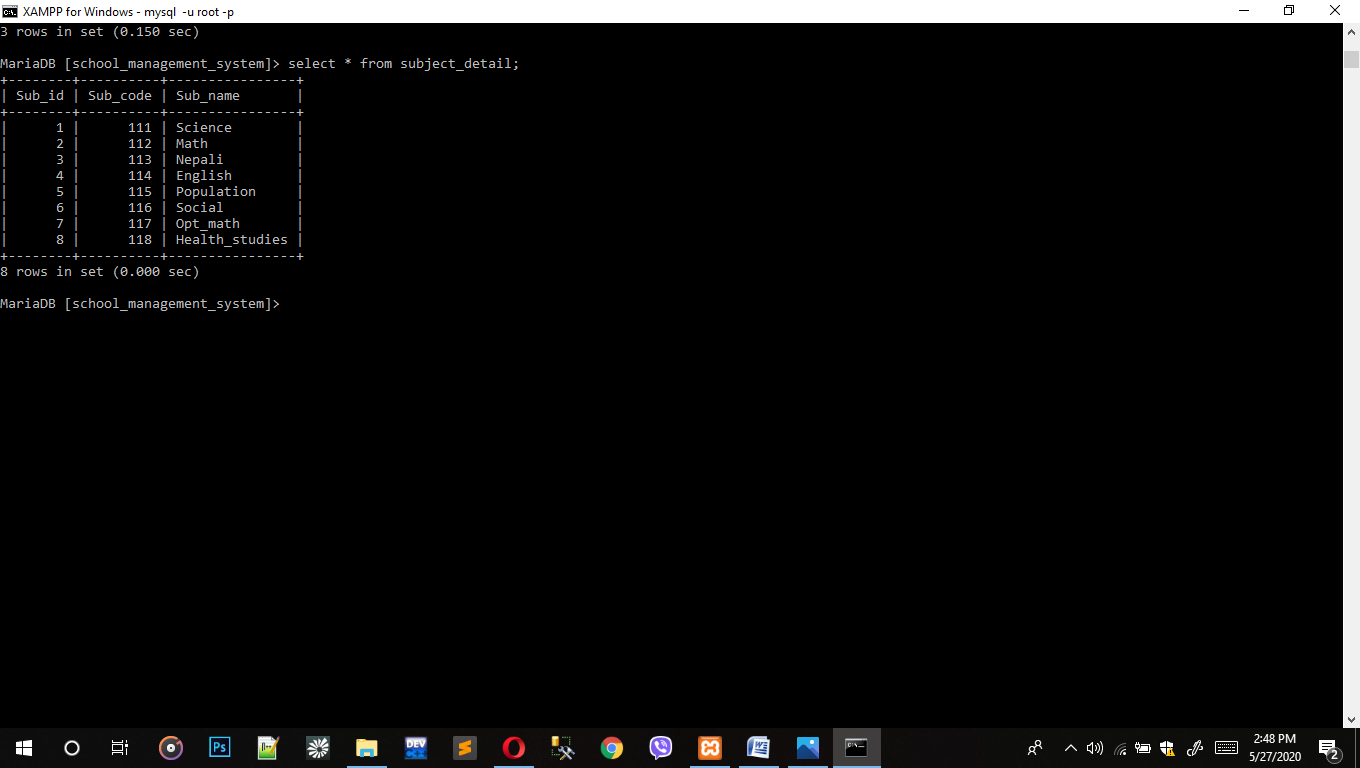


Syntax: insert into exam\_report values(1,1,1,3.96,'A+'), (2,2,2,3.56,'A'), (3,3,3,2.98,'B+'), (4,4,4,4.00,'A+'),(5,5,3.45,'A'),(6,6,6,3.50,'A'),(7,7,7,3.24,'A'),(8,8,8,2.14,'B'),(9,9,9,3.12,'A'),(10,10,10,2.22,'B'),(11,11,5,2.75,'A'), (12,12,7,2.88,'B+'), (13,13,3,3.20,'B+'), (14,14,9,3.90,'A+'), (15,15,1,3.21,'A'), (16,16,10,3.50,'A'), (17,17,10,3.50,'A'), (18,18,1,2.68,'B+');

### 5.2.12 Subject Detail



Syntax: create table subject\_detail(sub\_id int not null primary key, sub\_name varchar(200) not null, sub\_code int not null);

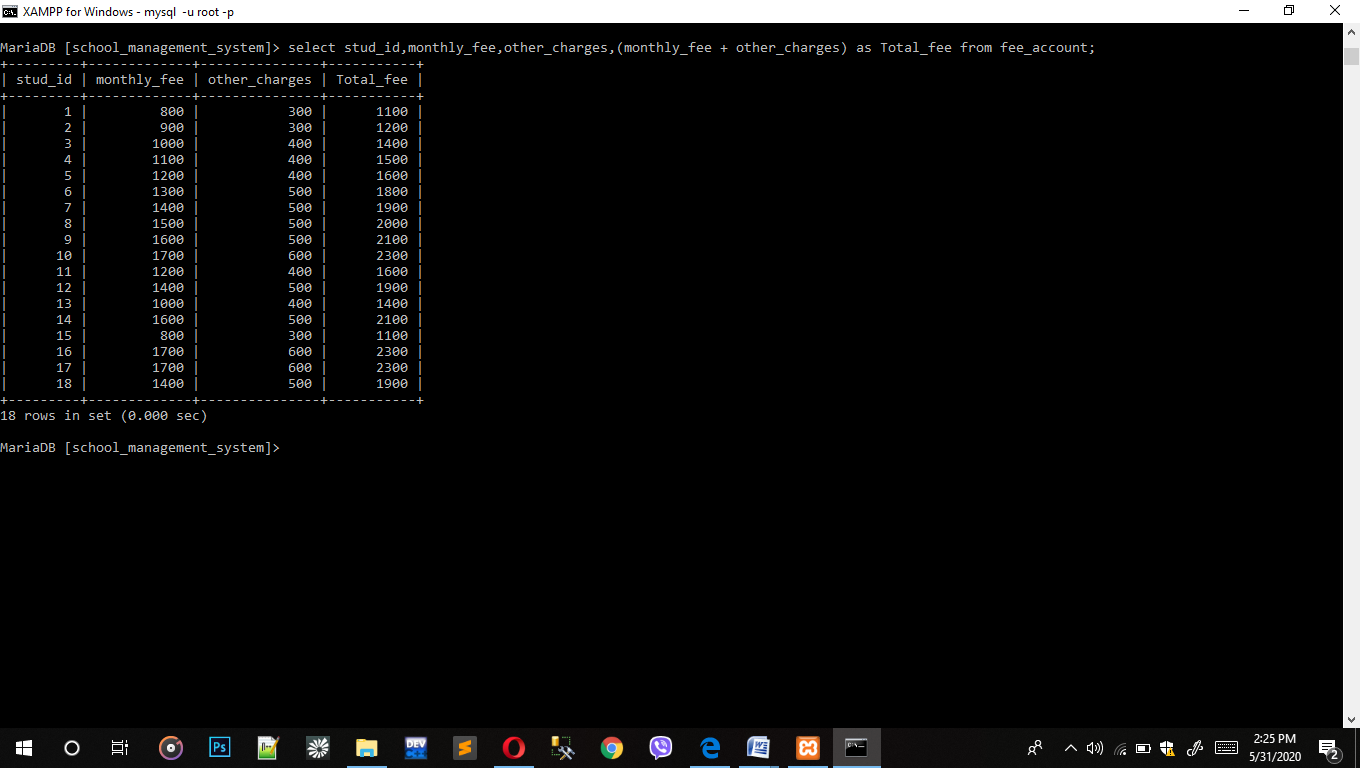


Syntax: insert into subject\_detail valus(1,111,'Science'), (2,112,'Math'), (3,113,'Nepali'), (4,114,'English'), (5,115,'Population'), (6,116,'Social'), (7,117,'Opt\_math'), (8,118,'Health\_studies');

# 6. Select Statement Using Different Function

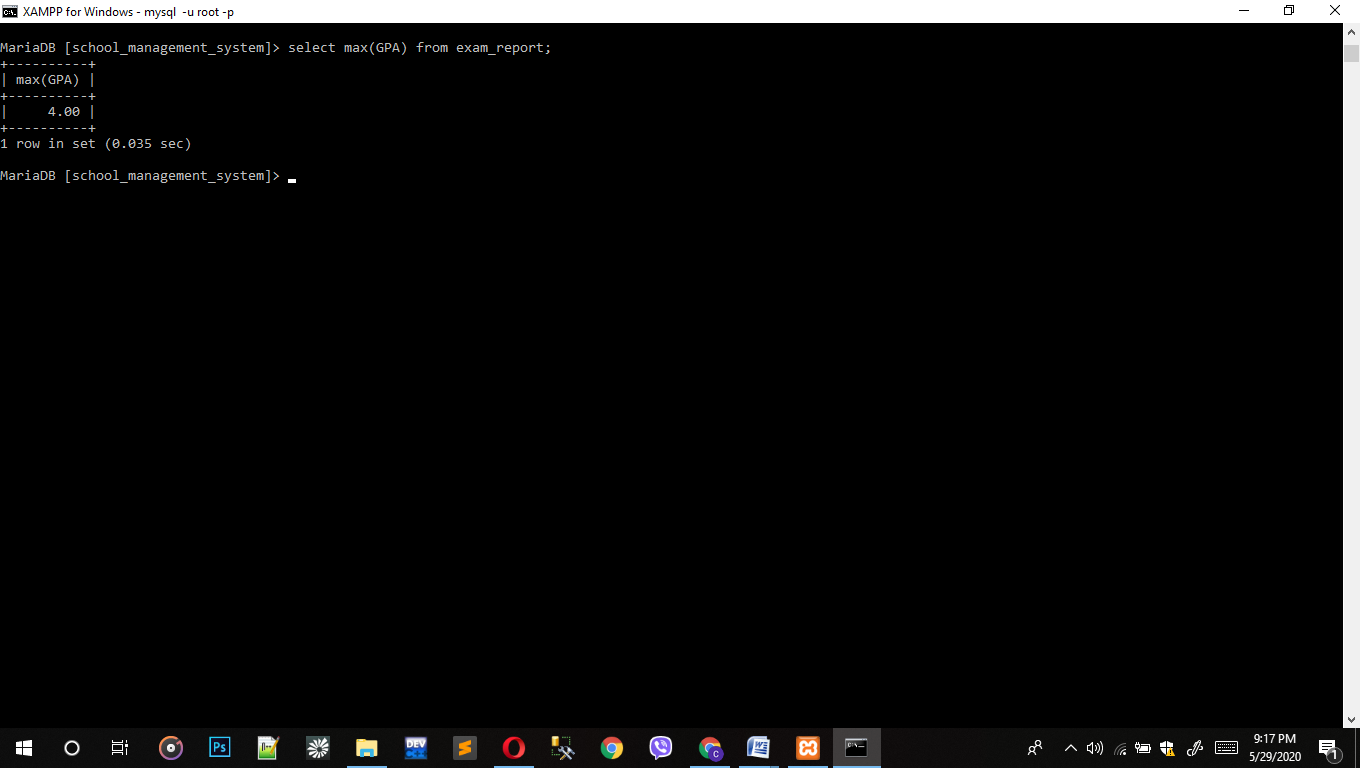
1. Query to display total fee of student by merging monthly fees and other charges.

* Select count(\*) from students where age=14;



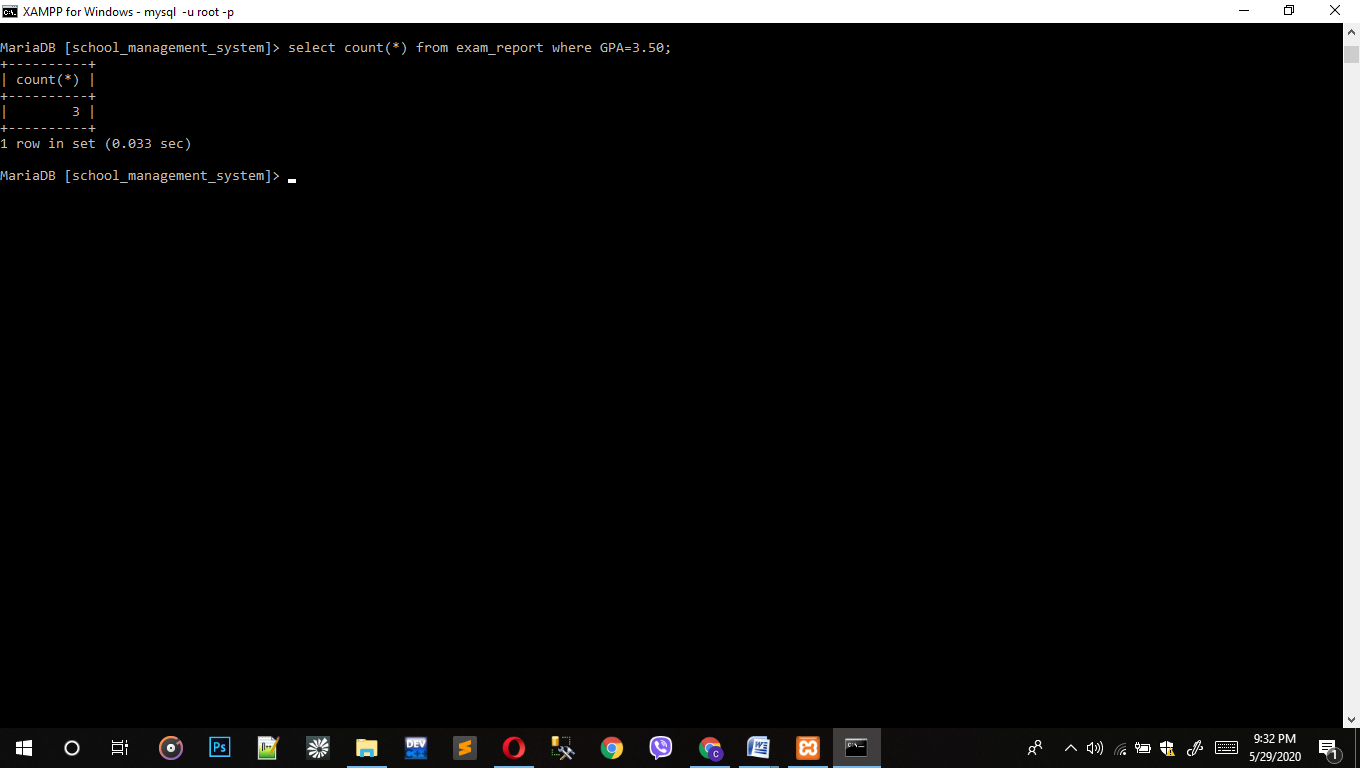
2. Query to display maximum GPA.

* Select max(GPA) from exam\_report;



3. Query to display total number of exam\_report who got 3.50 GPA.

* Select count(\*) from exam\_report where GPA=3.50;



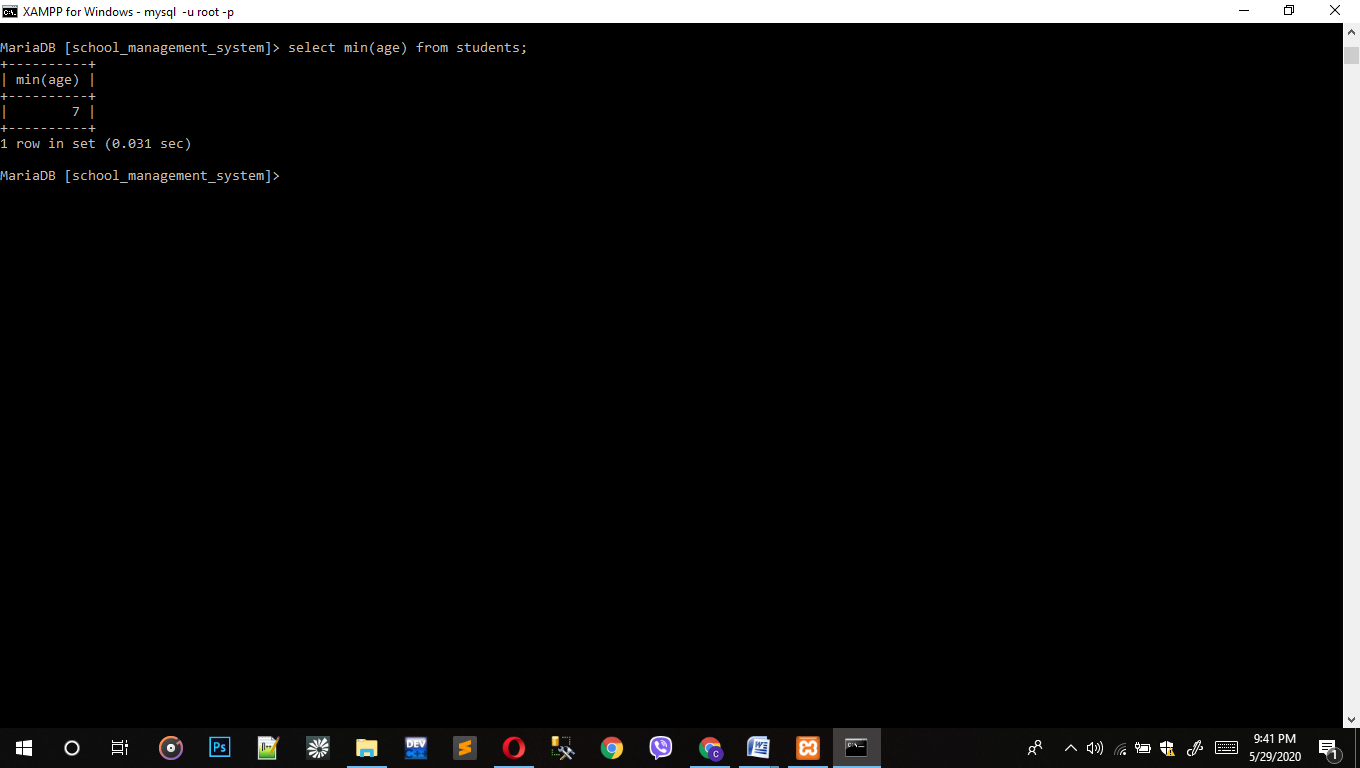
1. Query to select character from ASCII value.

* Select char(65) as NumberCodeToCharecter;



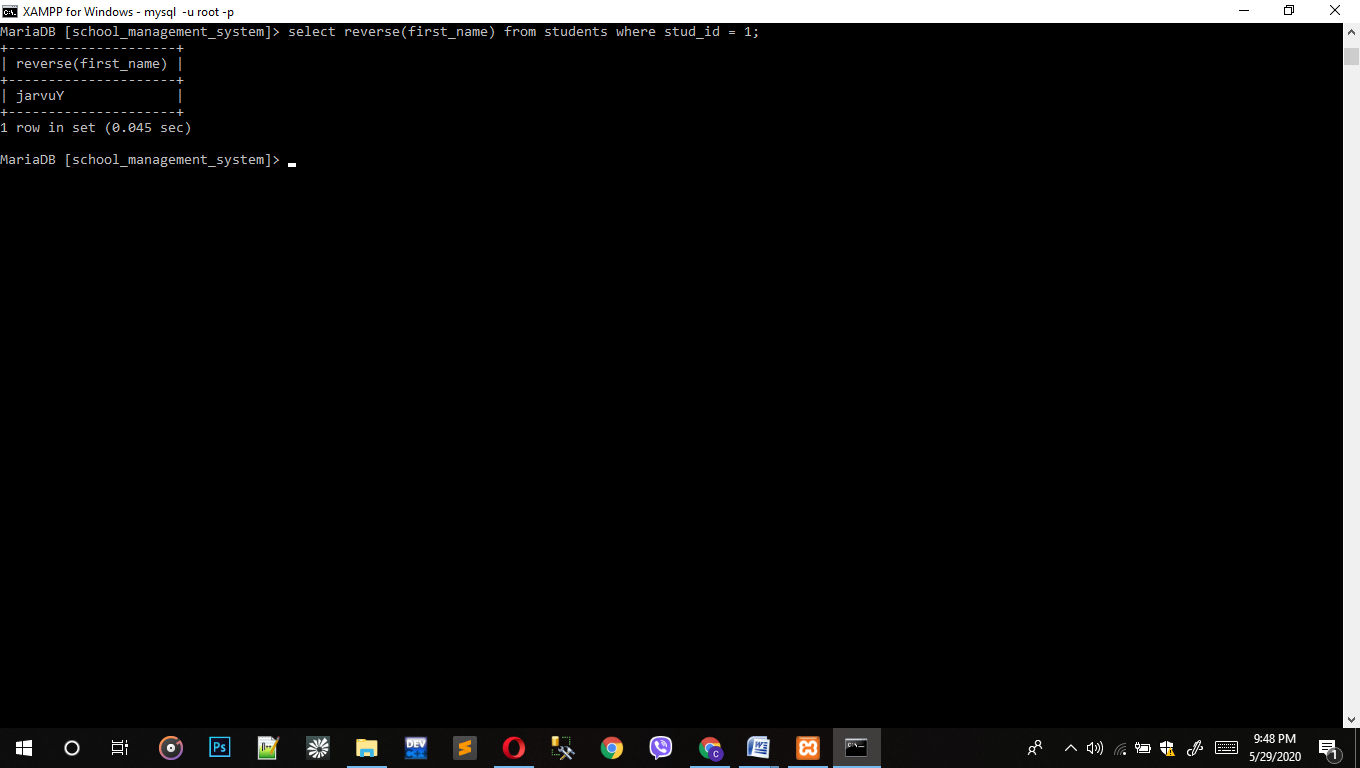
1. Query to display minimum age of students.

* Select min(age) from students;



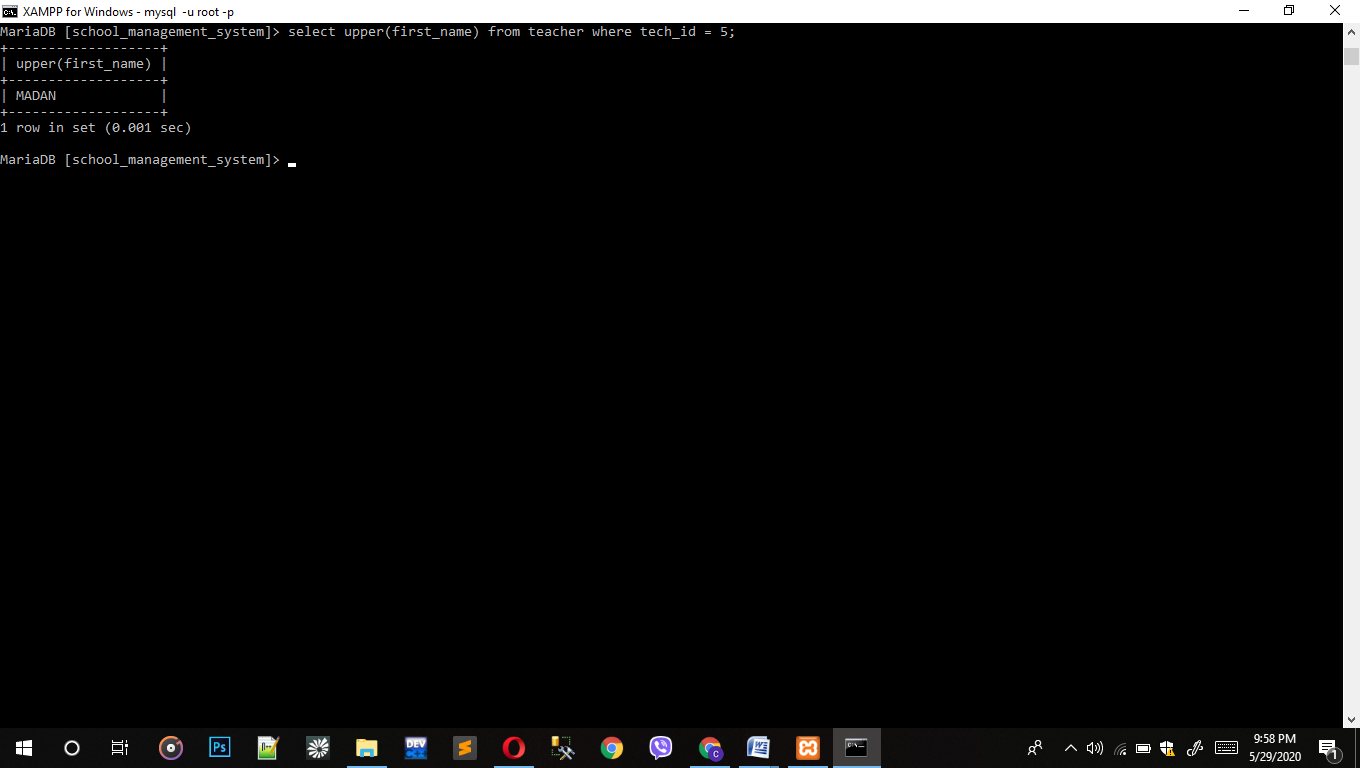
1. Query to display reverse of student name.

* Select reverse(first\_name) from students where stud\_id = 1;



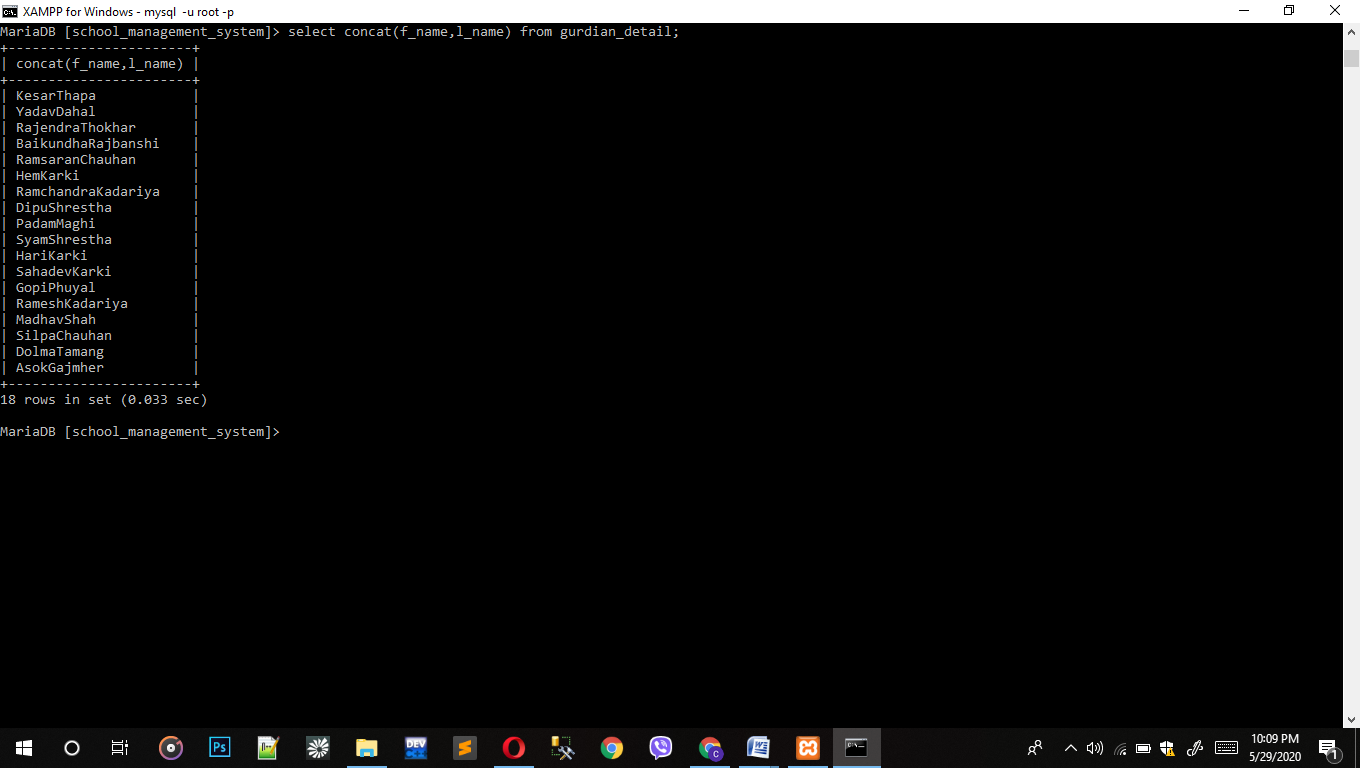
1. Query to display upper case character from teacher.

* Select upper(first\_name) from teacher where tech\_id =5;



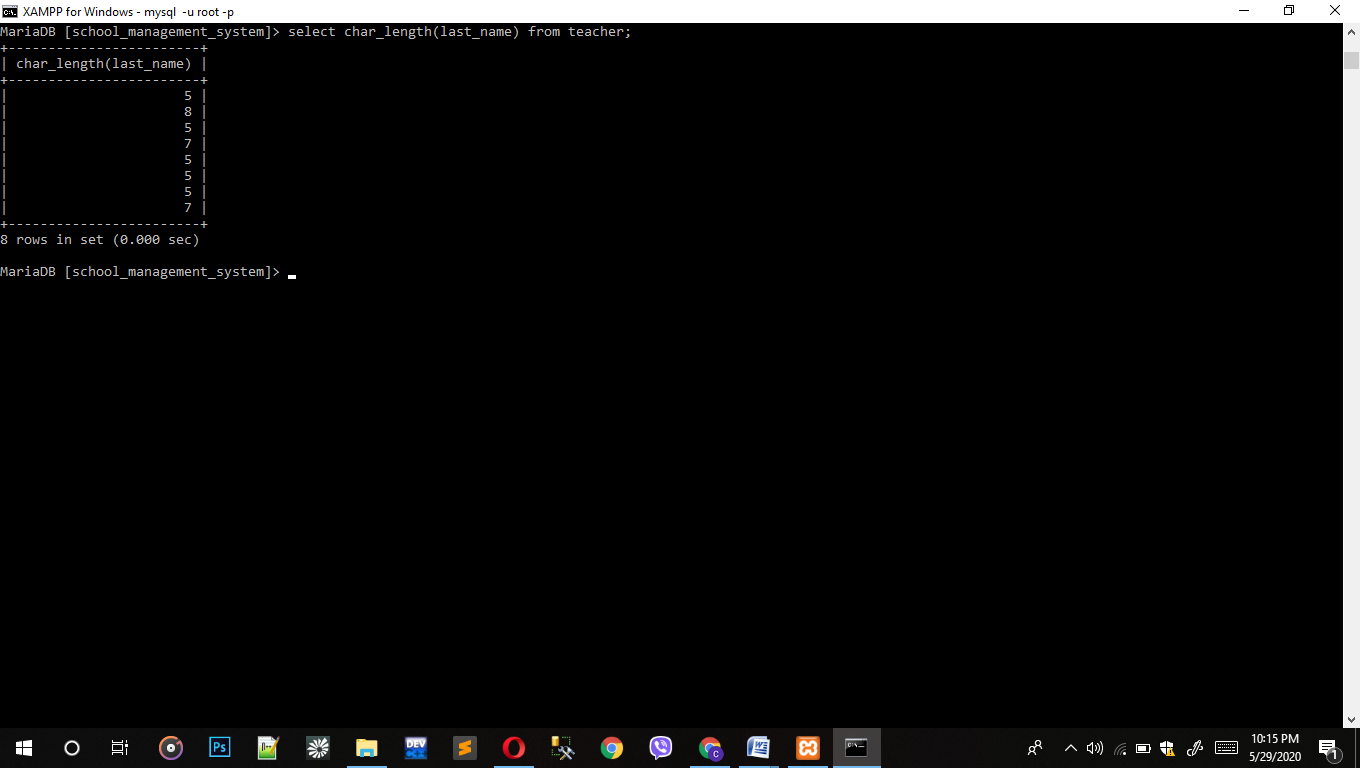
1. Query to show full name of guardian\_detail.

* Select concat (f\_name,l\_name) from guardian\_detail;



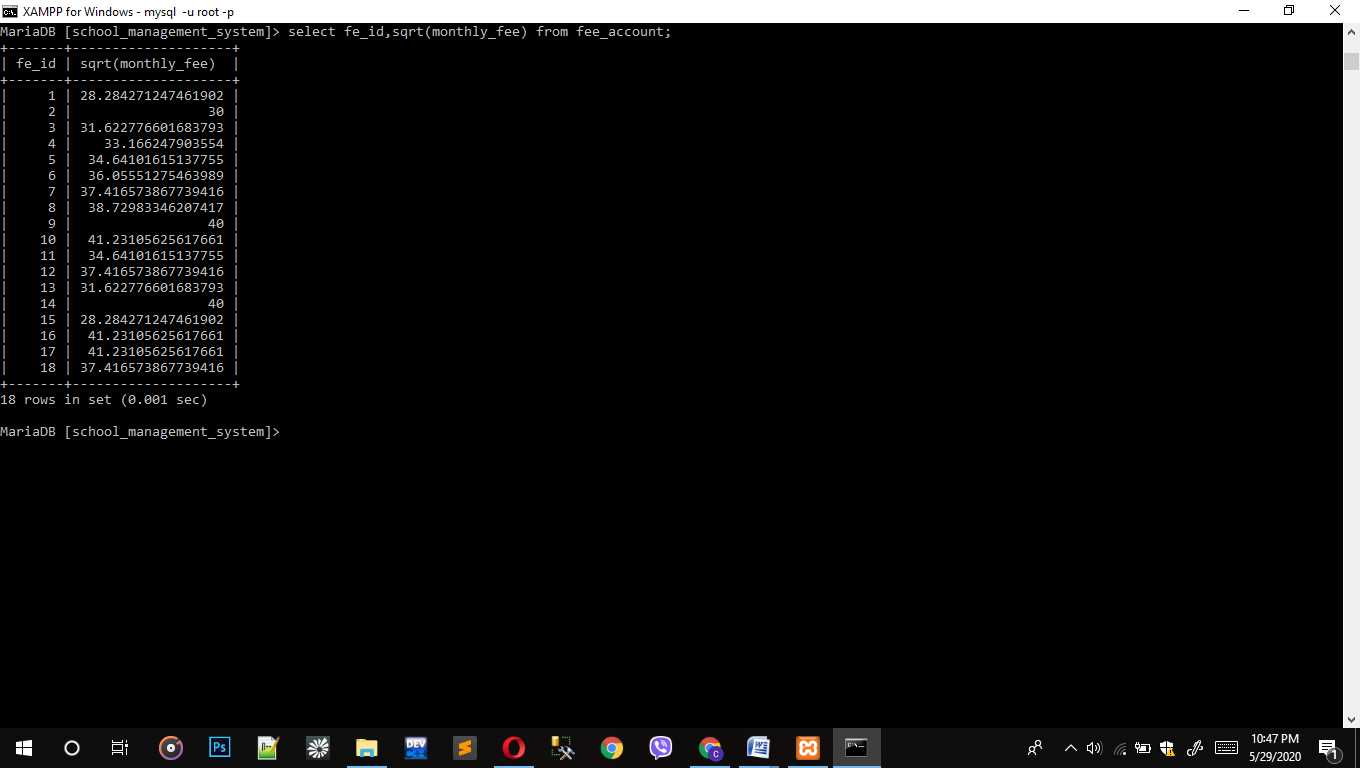
1. Query to display to count character of last\_name of teacher.

* Select char\_length(last\_name) from teacher;



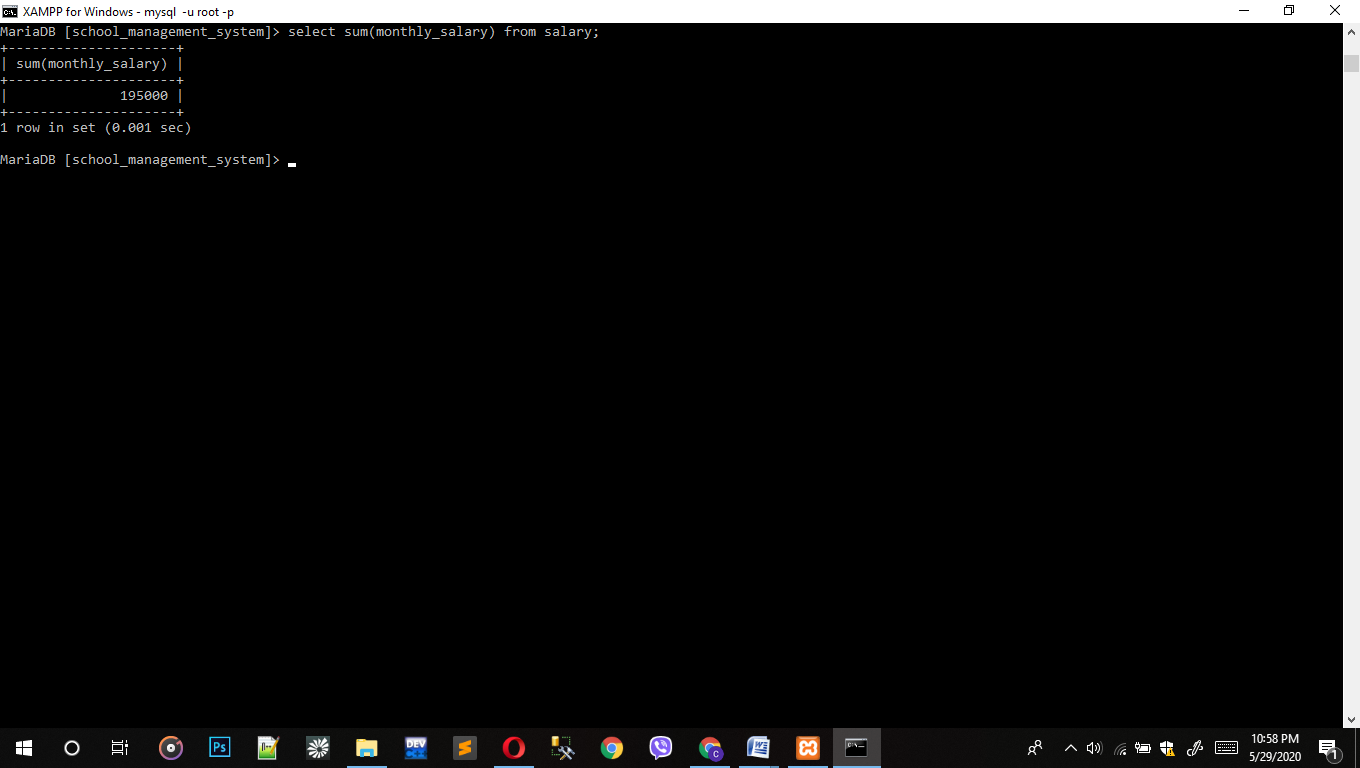
1. Query to display square root of monthly\_fee from fee\_account.

* Select fe\_id,sqrt(monthly\_fee) from fee\_account.



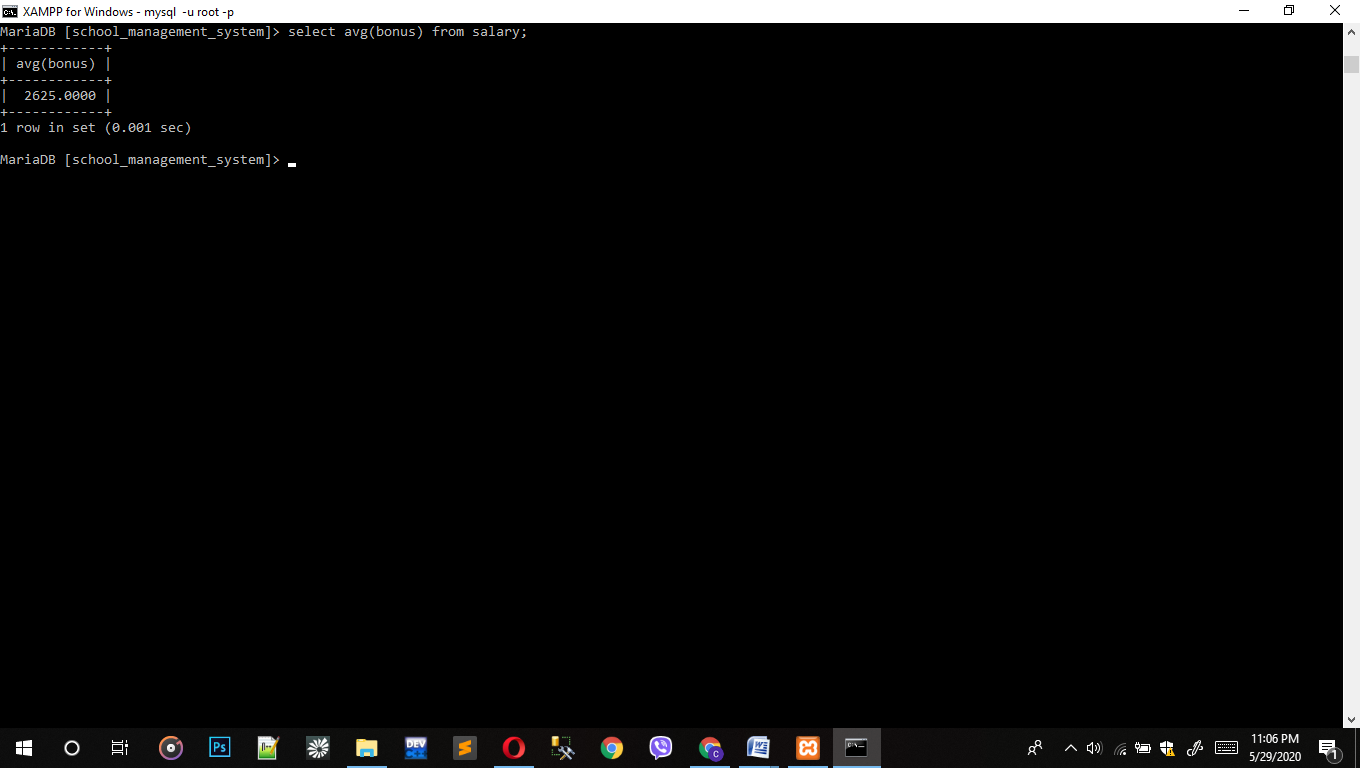
1. Query to display of salary.

* Select sum(monthly\_salary) from salary;



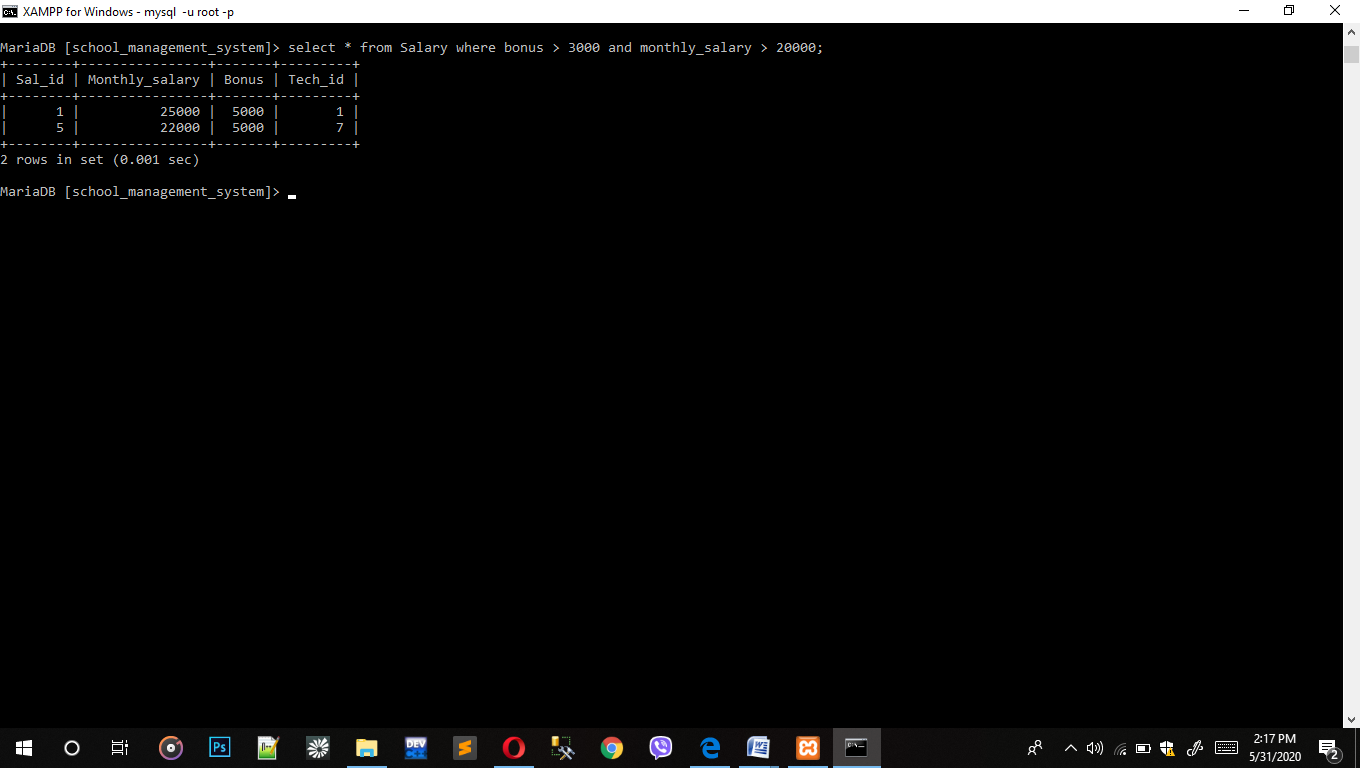
1. Query to display average bonuses from salary.

* Select upper(first\_name) from teacher where tech\_id =5;



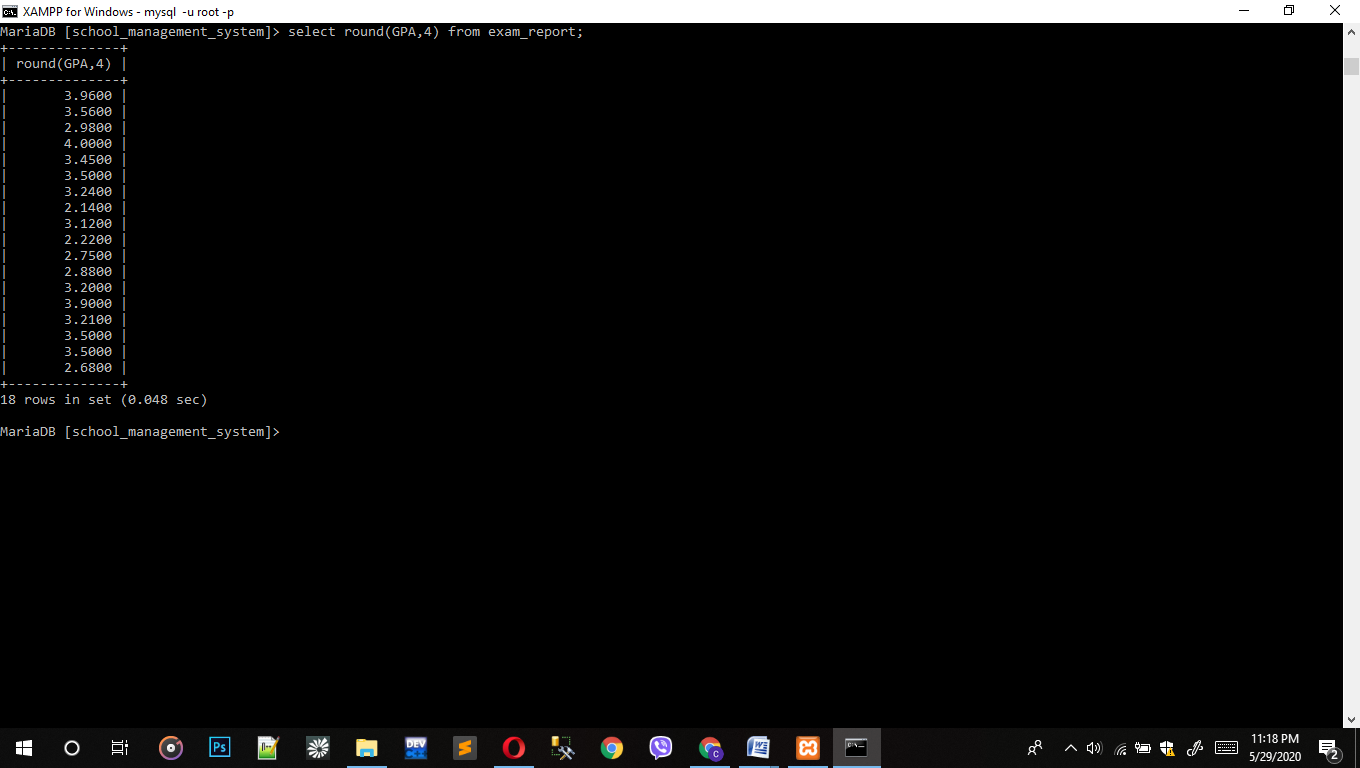
1. Query to display number of rows in library.

* Select \* from salary where bonus > 3000 and monthly\_salary >20000;



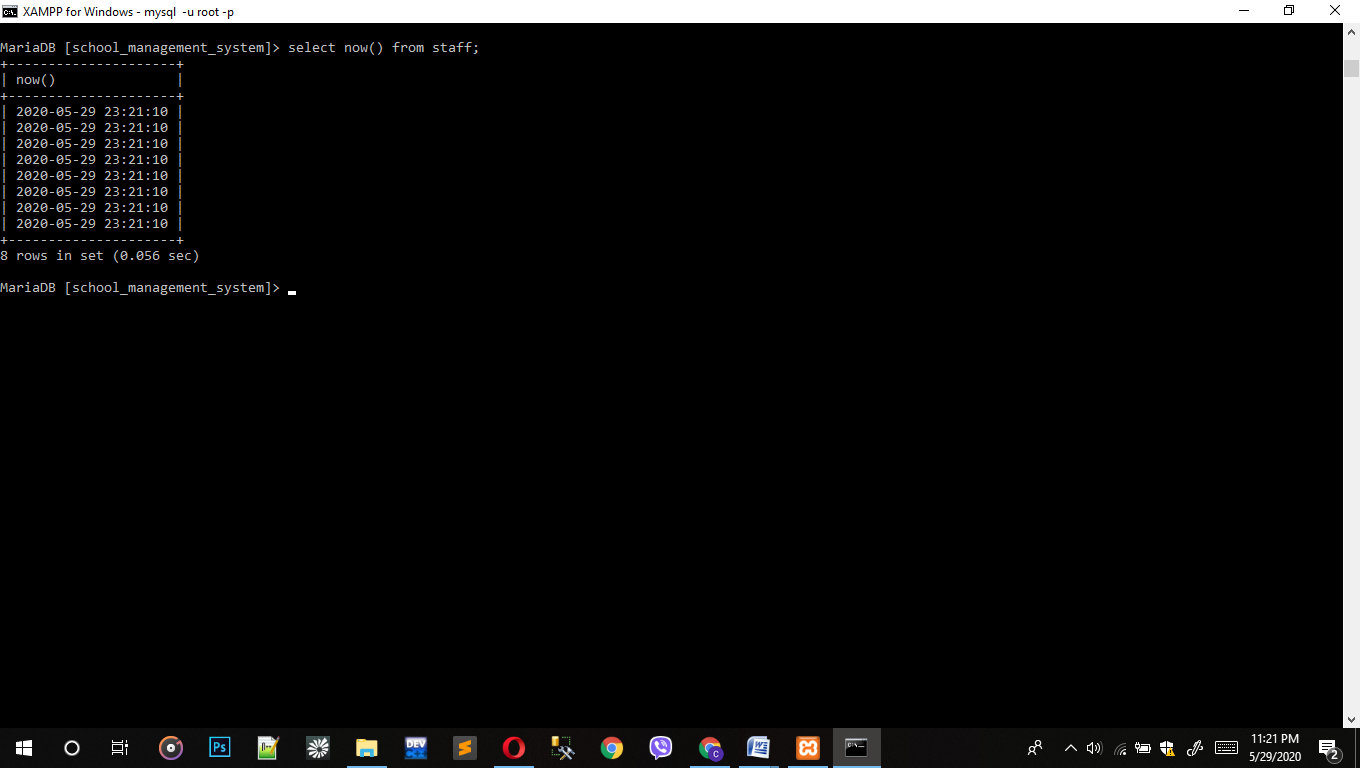
1. Query to display round value of decimals from Exam\_report

* Select round(GPA,4) from exam report;



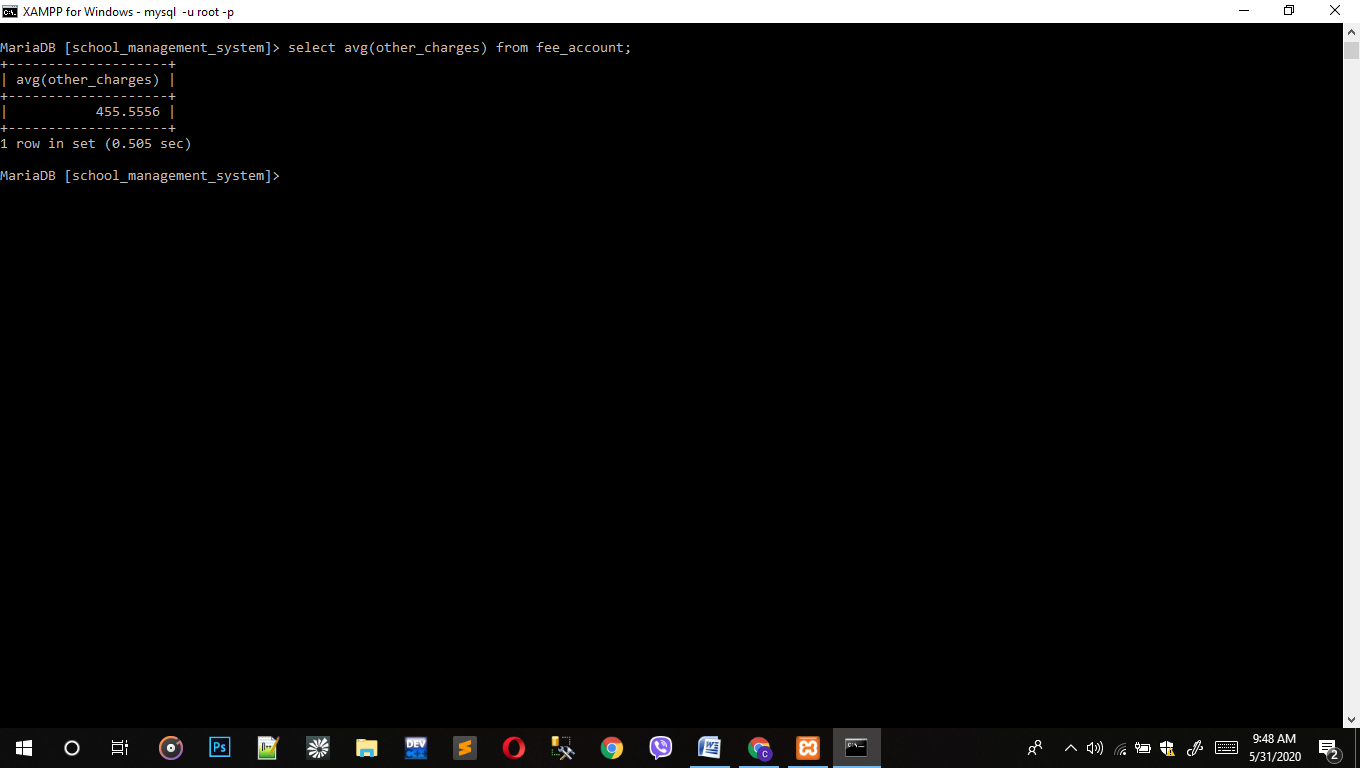
1. Query to display the system date and time.

* Select now() from staff;



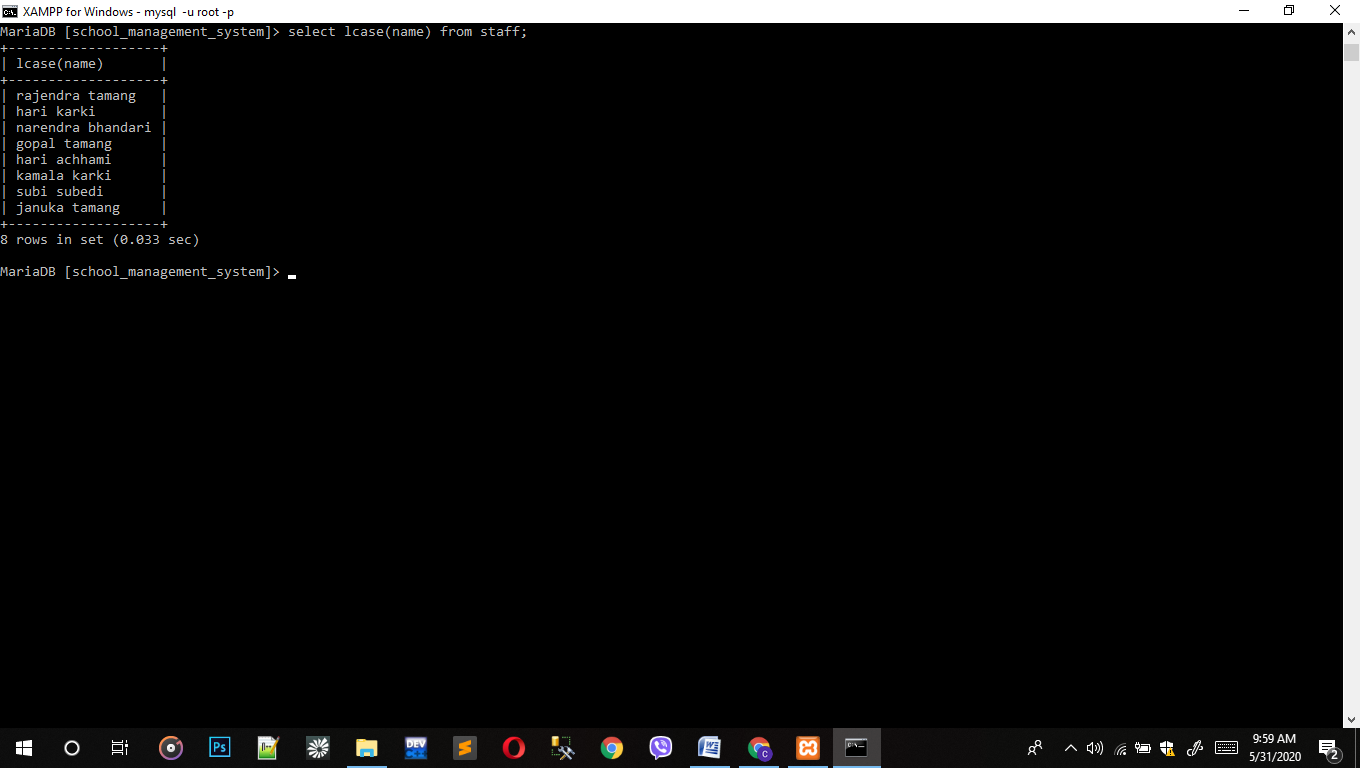
1. Query to display the average of other\_charges from Fee Account.

* Select avg(other\_charges) from fee\_account;



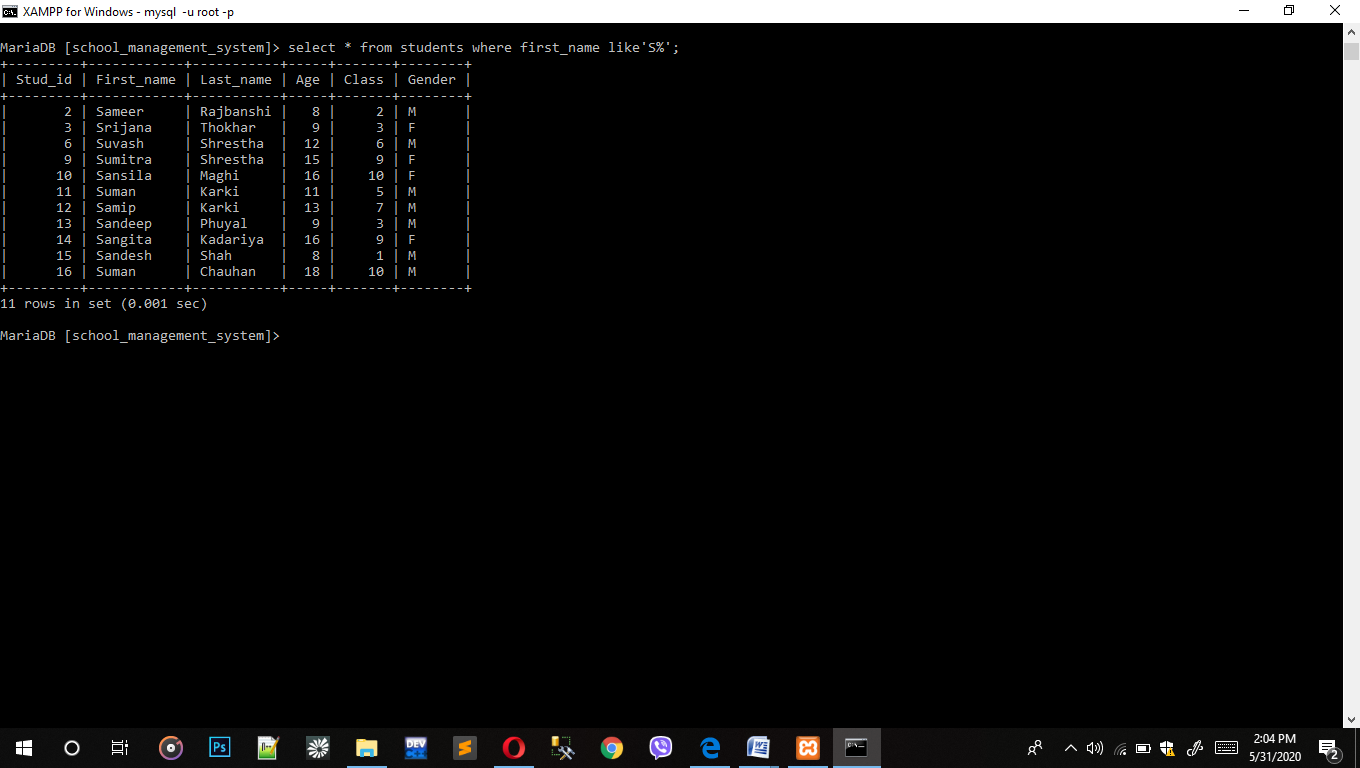
1. Query to display character in lowercase.

* Select lcase(name) from staff;



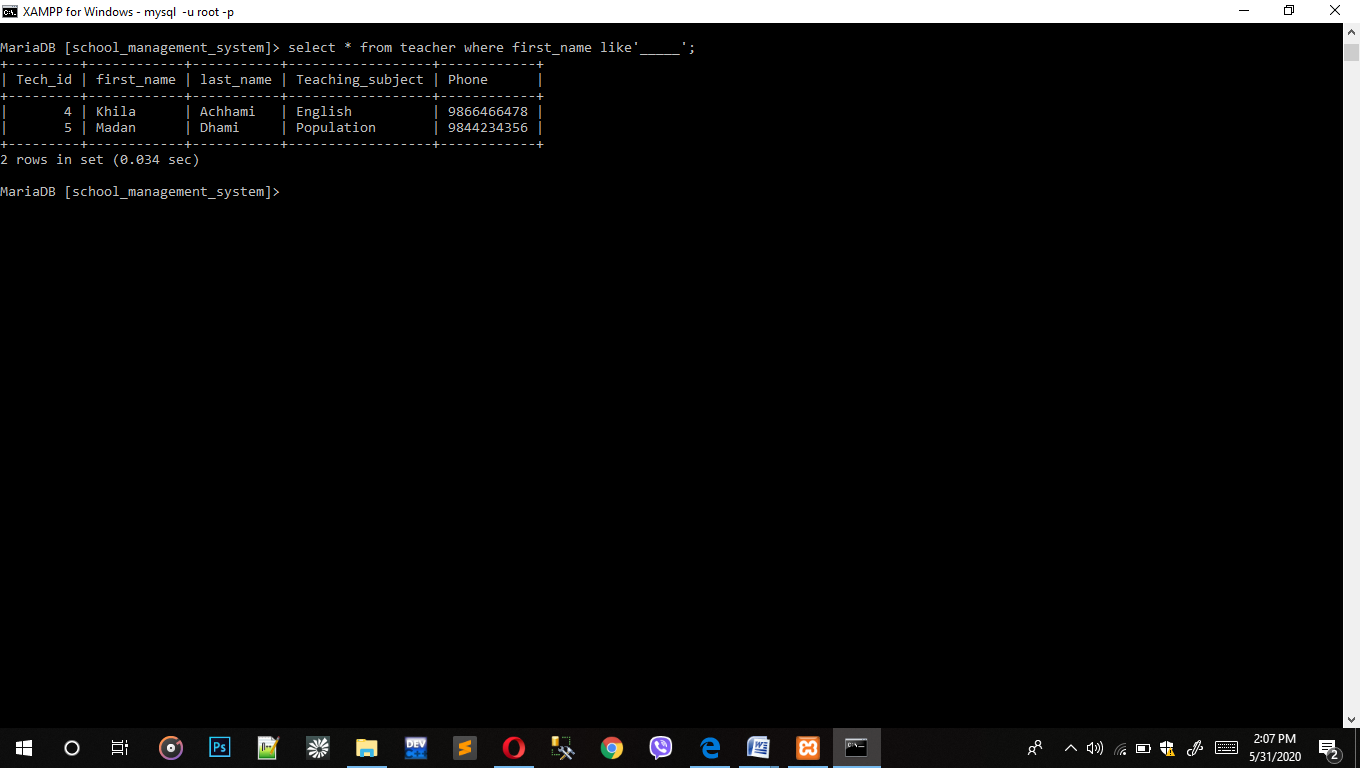
1. Query to display the students whose first name starts with 'S'.

* Select \* from students where first\_name like 'S%';



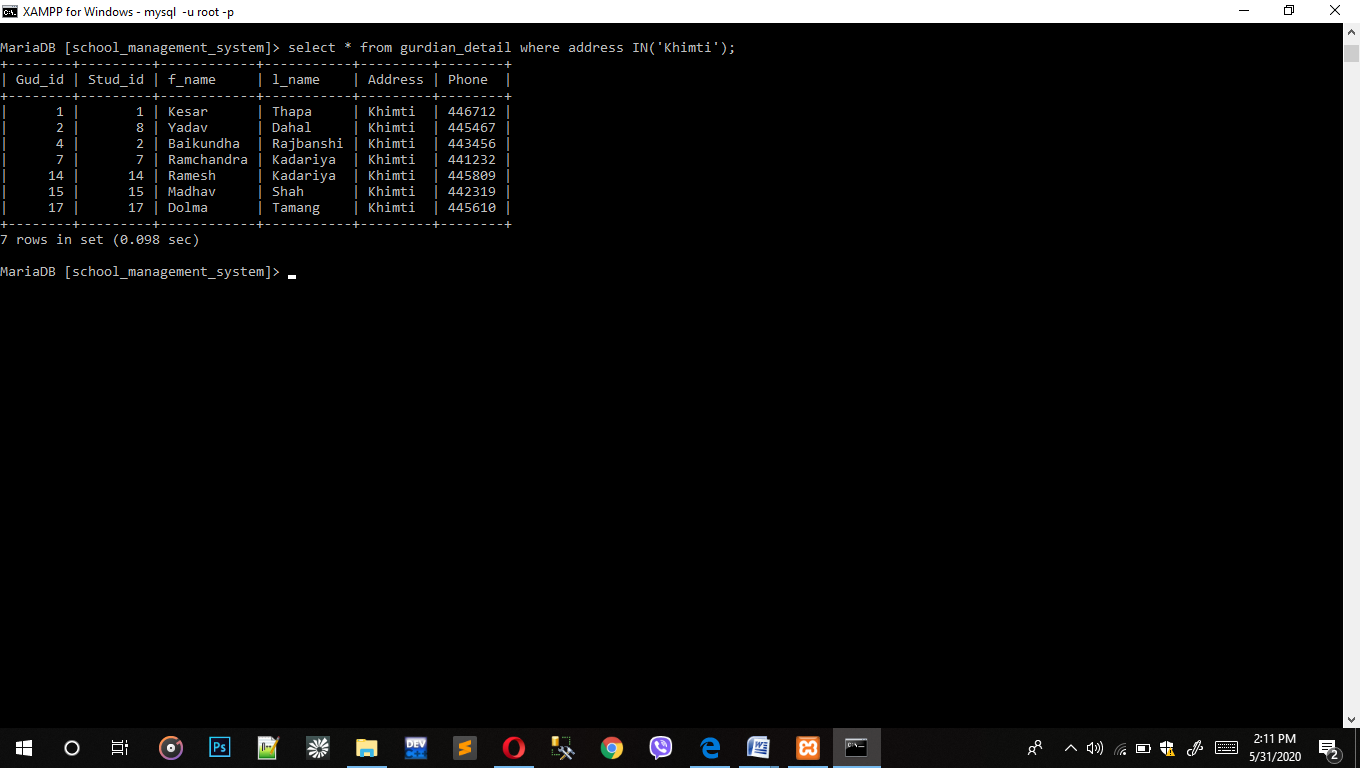
1. Query to display the teacher whose first name has 5 character.

* Select \*from teacher where first\_name like'\_\_\_\_\_';



1. Query to display the guardian whose address is Khimti.

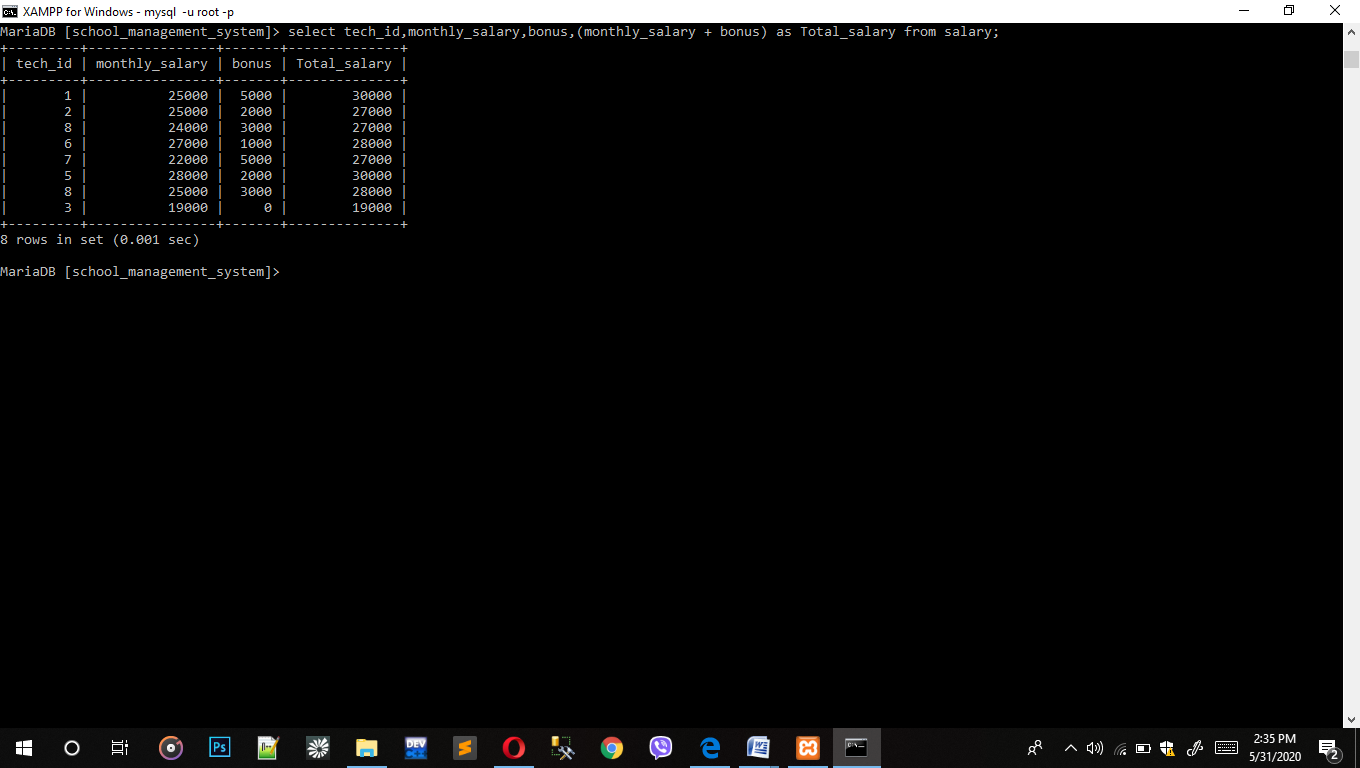
* Select \* from guardian\_detail where aderss IN('Khimti');



# Select statements Using Sub Query

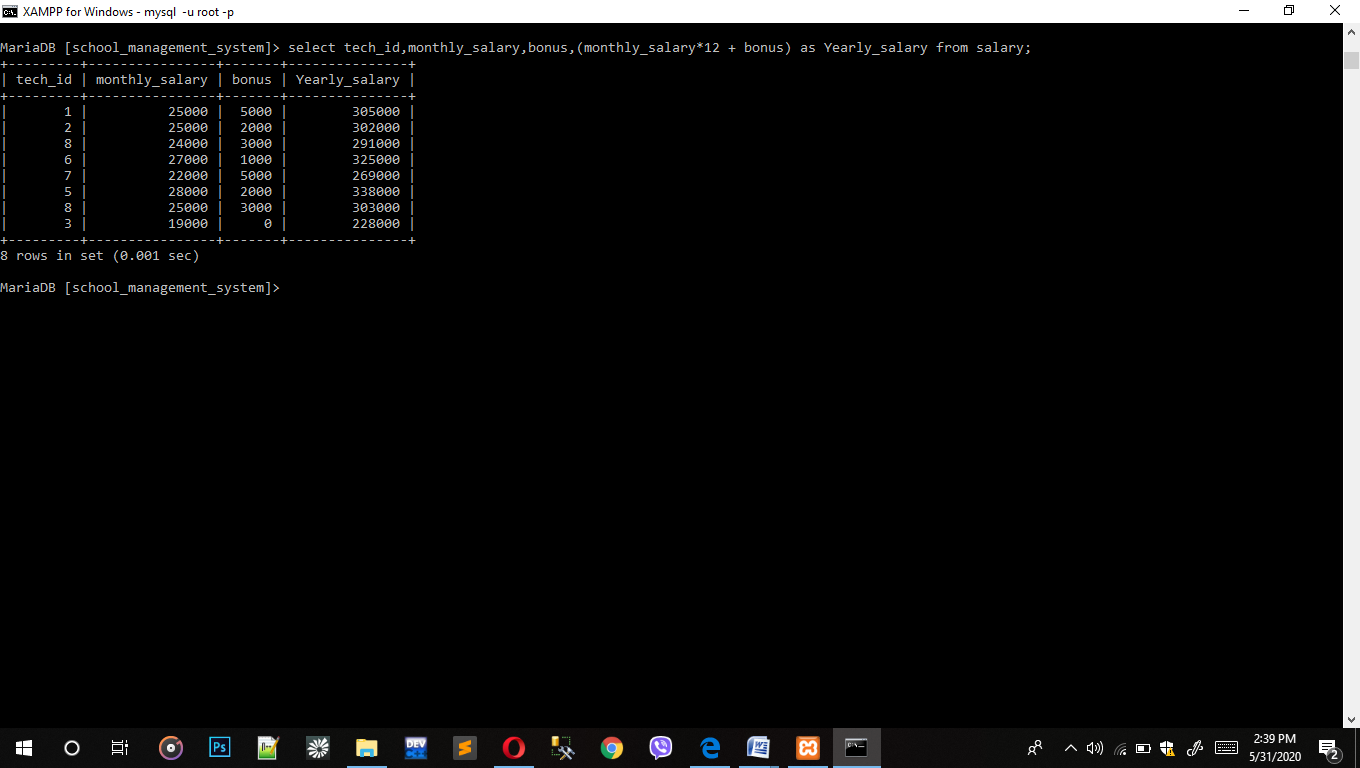
1. Query to display the total salary of teacher by adding bonus and monthly salary.

* Select tech\_id,monthly\_salary,bonus(monthly\_salary + bonus) as Total\_salary from salary;



1. Query to display the yearly salary of teacher.

* Select tech\_id,monthly\_salary,bonus(monthly\_salary\*12) as Yearly\_salary from salary;



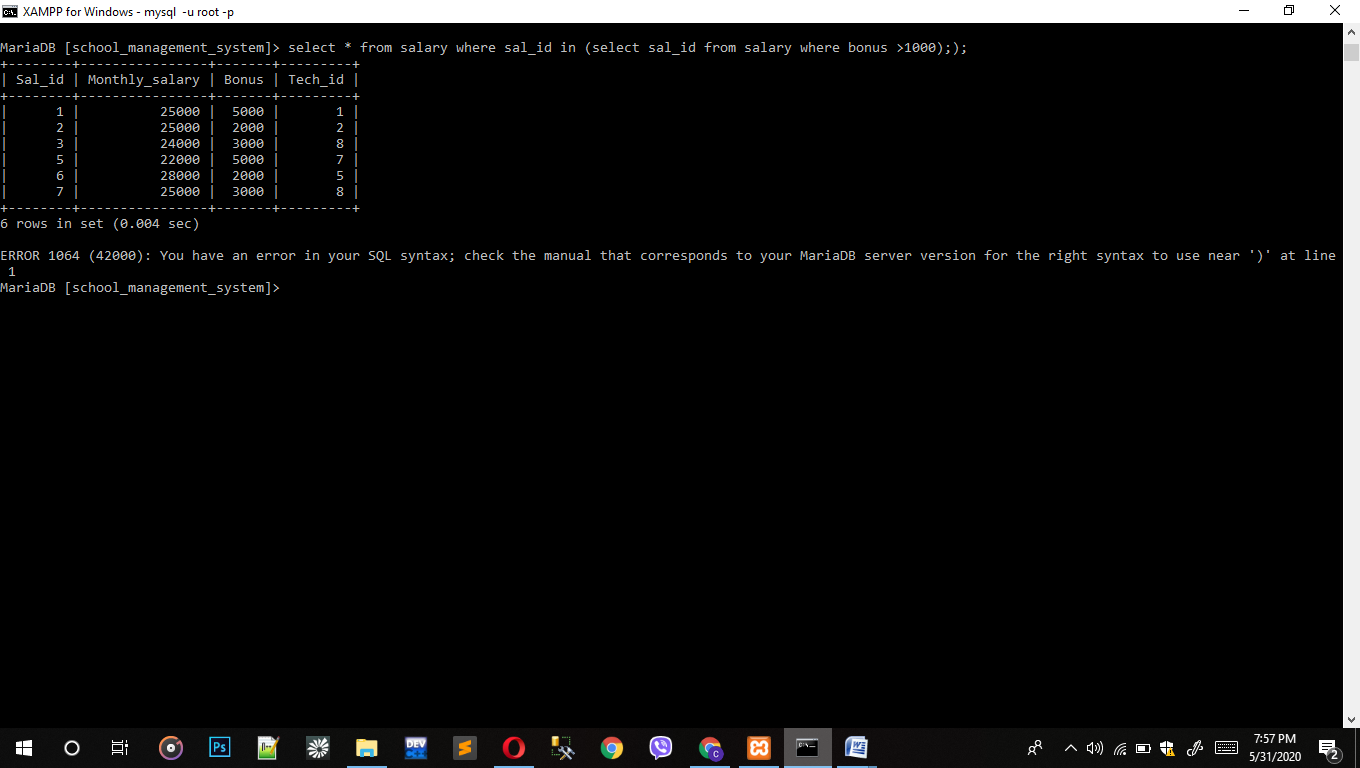
1. Query to display the total fee of student.

* Select stud\_id, monthly\_fee, other\_charges,(monthly\_fee + other\_charges)\*12) as Total\_fee from fee\_account;



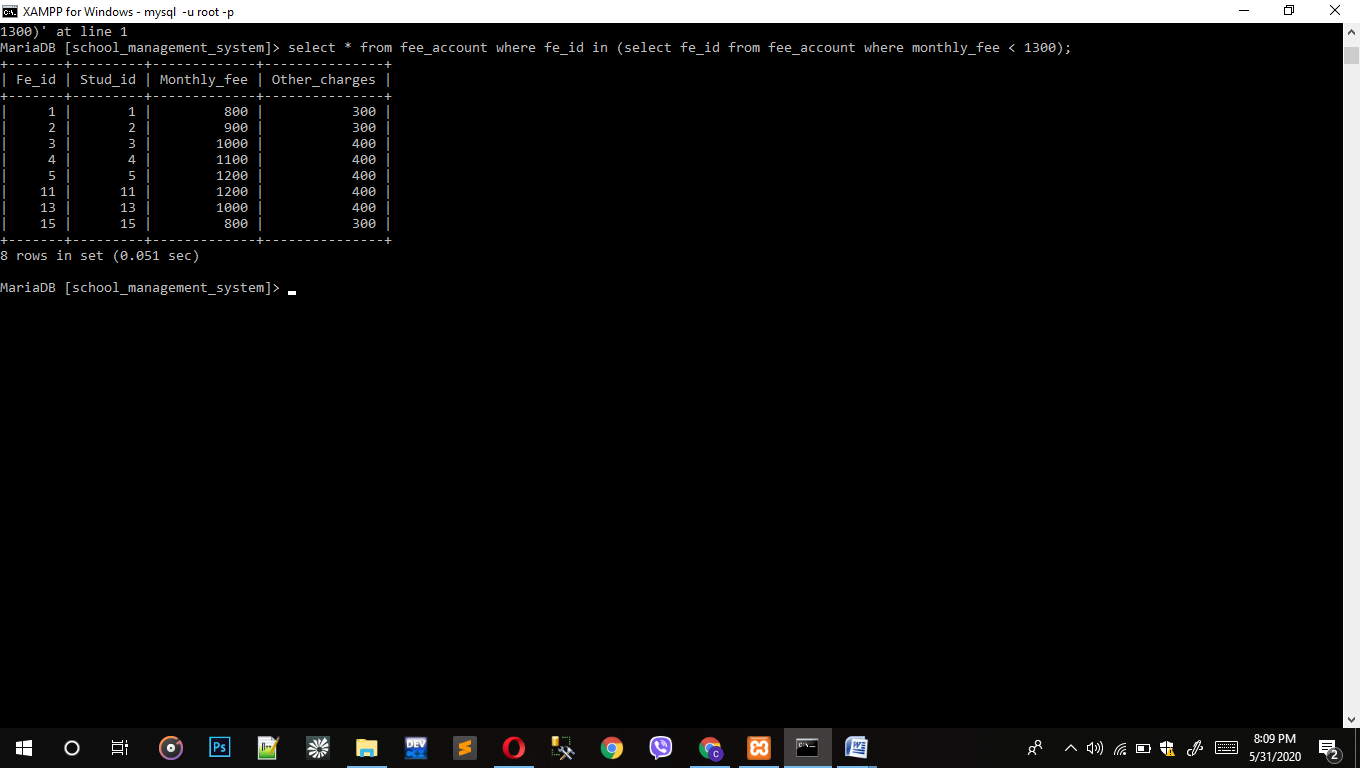
1. Query to display the salary whose bonus is over 1000.

* Select \* from salary where sal\_id in (select sal\_id from salary where bonus > 1000);



1. Query to display the students whose fee is less than 1300.

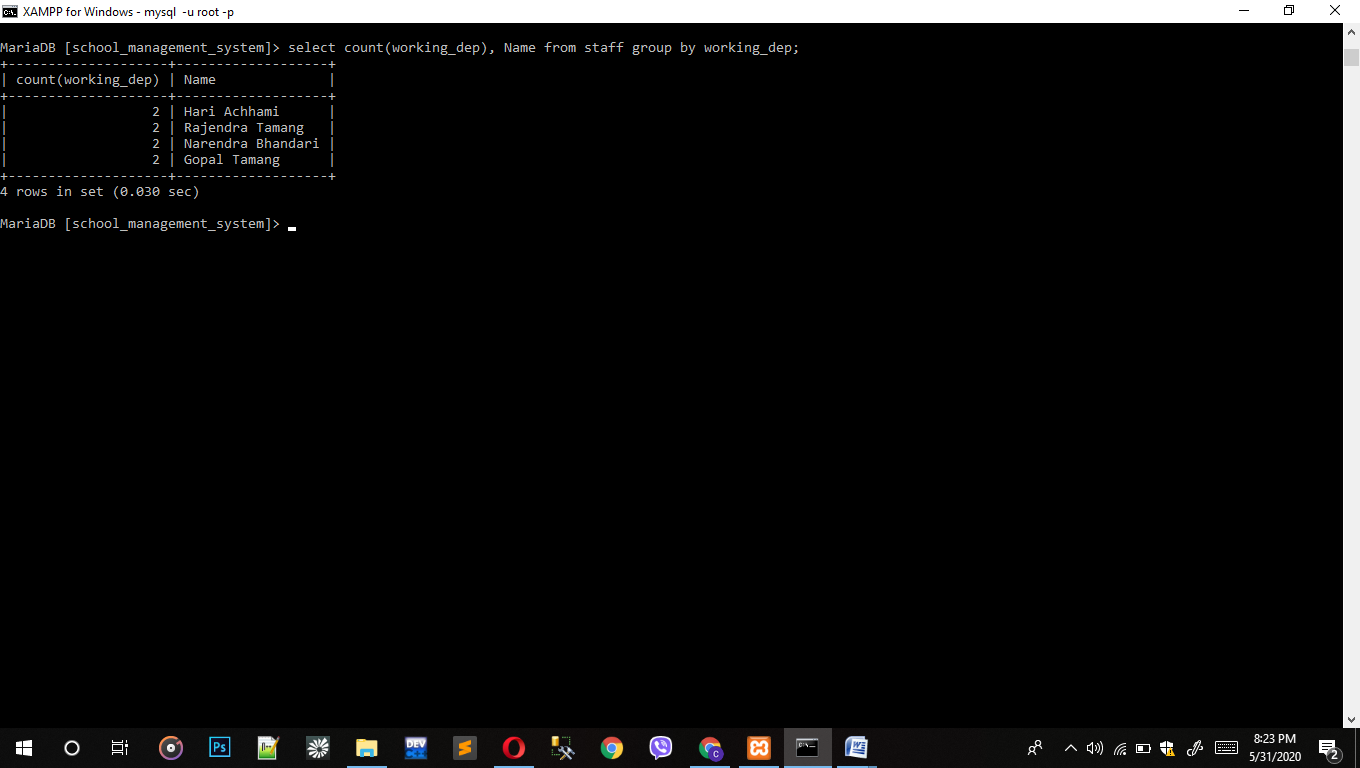
* Select \* from fee\_account where fe\_id in (select fe\_id from fee\_account where monthly\_fee <1300);



# Select Statements using Count and Group Functions

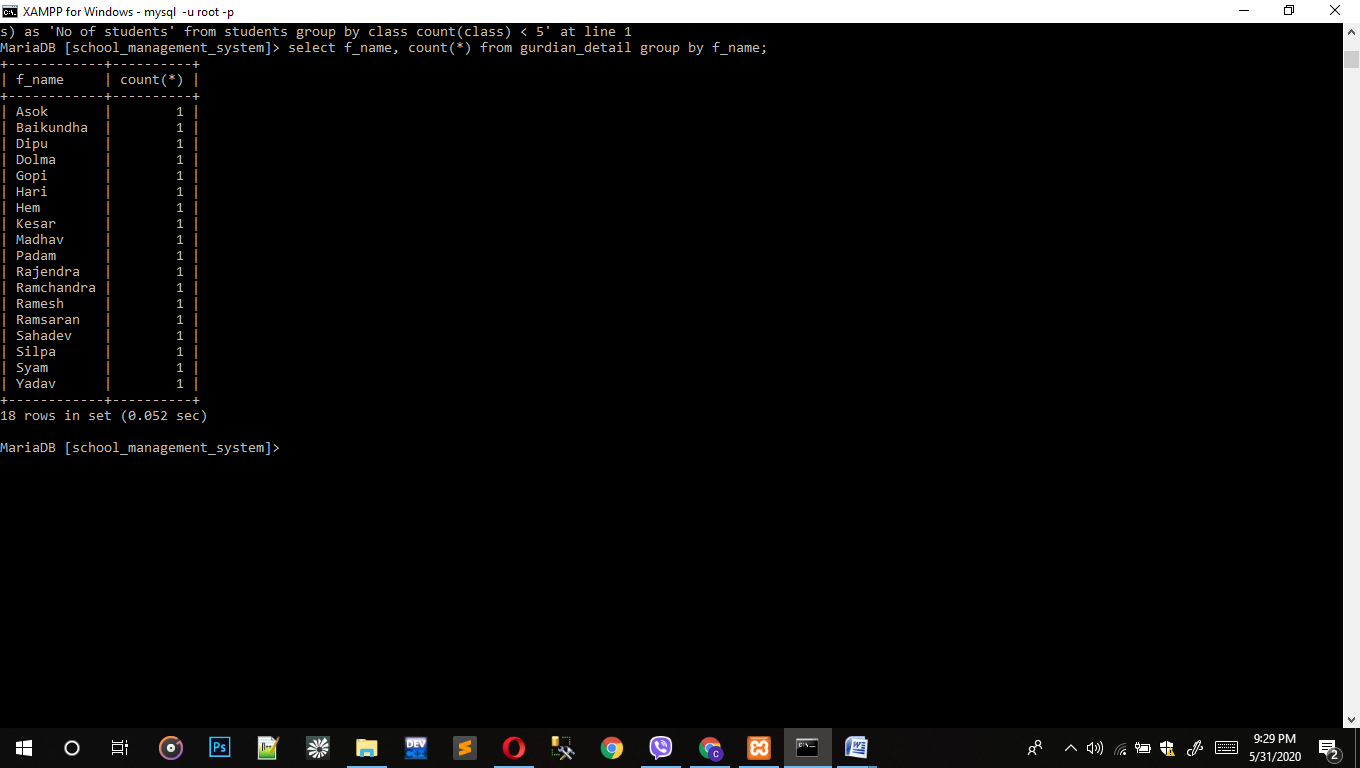
1. Query to count the name of staff working in particular department.

* Select count(working\_dep), name from staff group by working\_dep;



1. Query to count the first name of guardian group by first name;

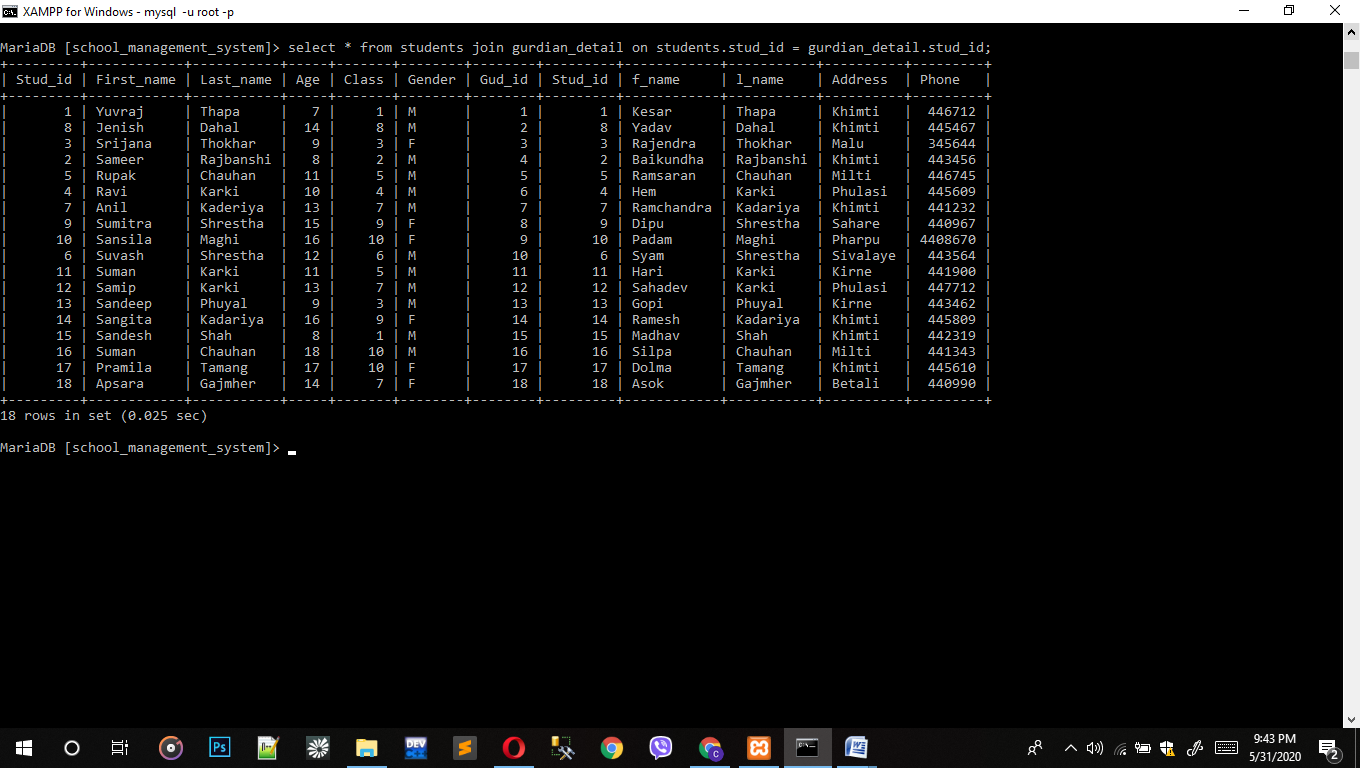
* Select f\_name, count(\*) from guardian\_detail group by f\_name;



# Select Statements Using Different Joins

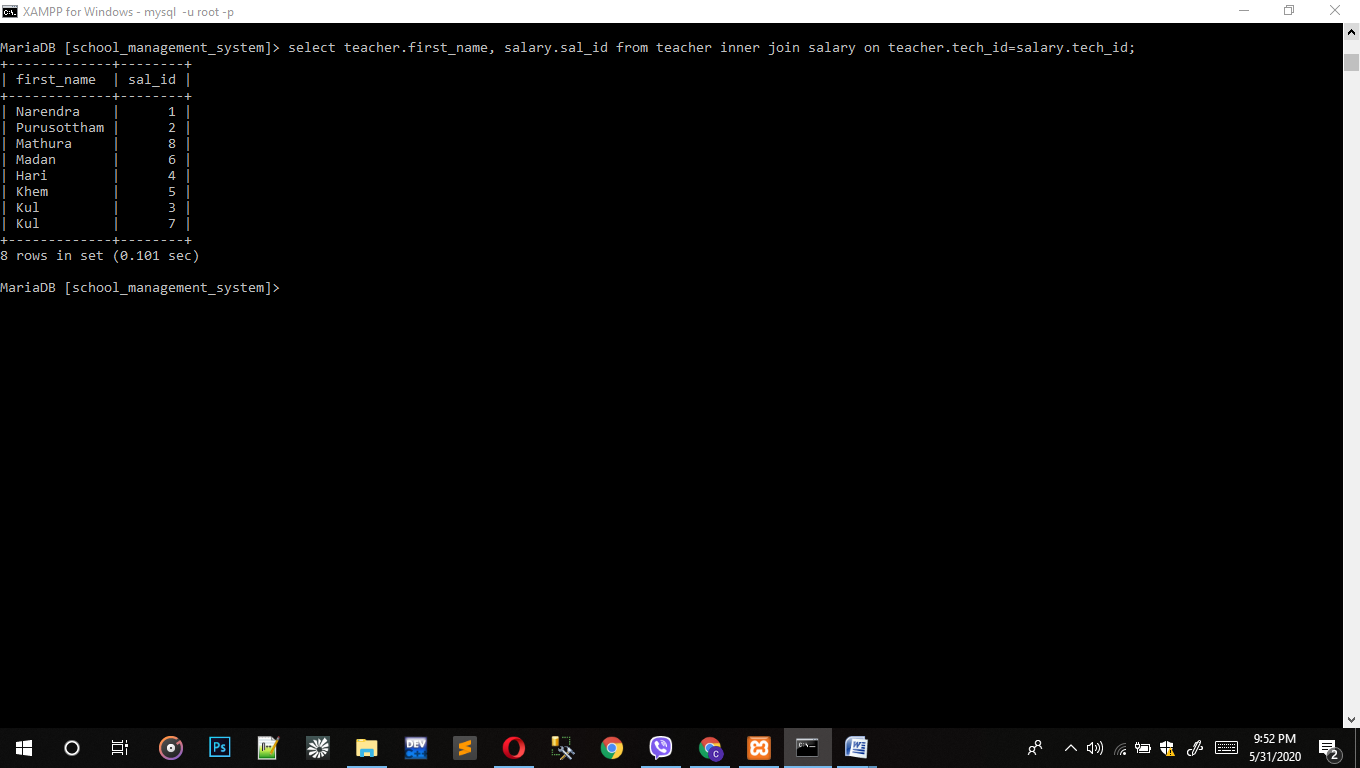
1. Query to join students and guardian.

* Select \* from students join gurdian\_detail on students.stud\_id = gurdian\_detail.stud\_id;



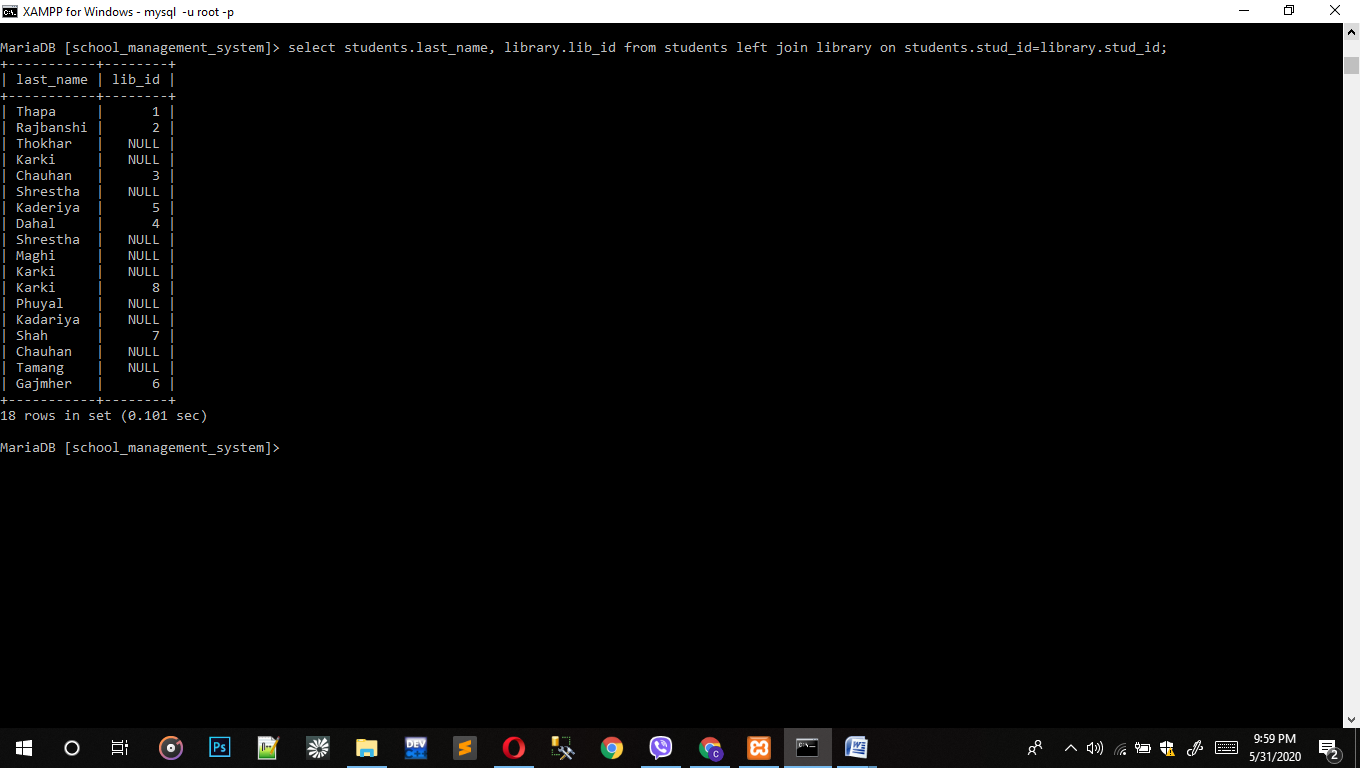
1. Query to inner joins table teacher and salary.

* Select teacher.first\_name, salary.sal\_id from teacher inner joins salary on teacher.tech\_id = salary.tech\_id;



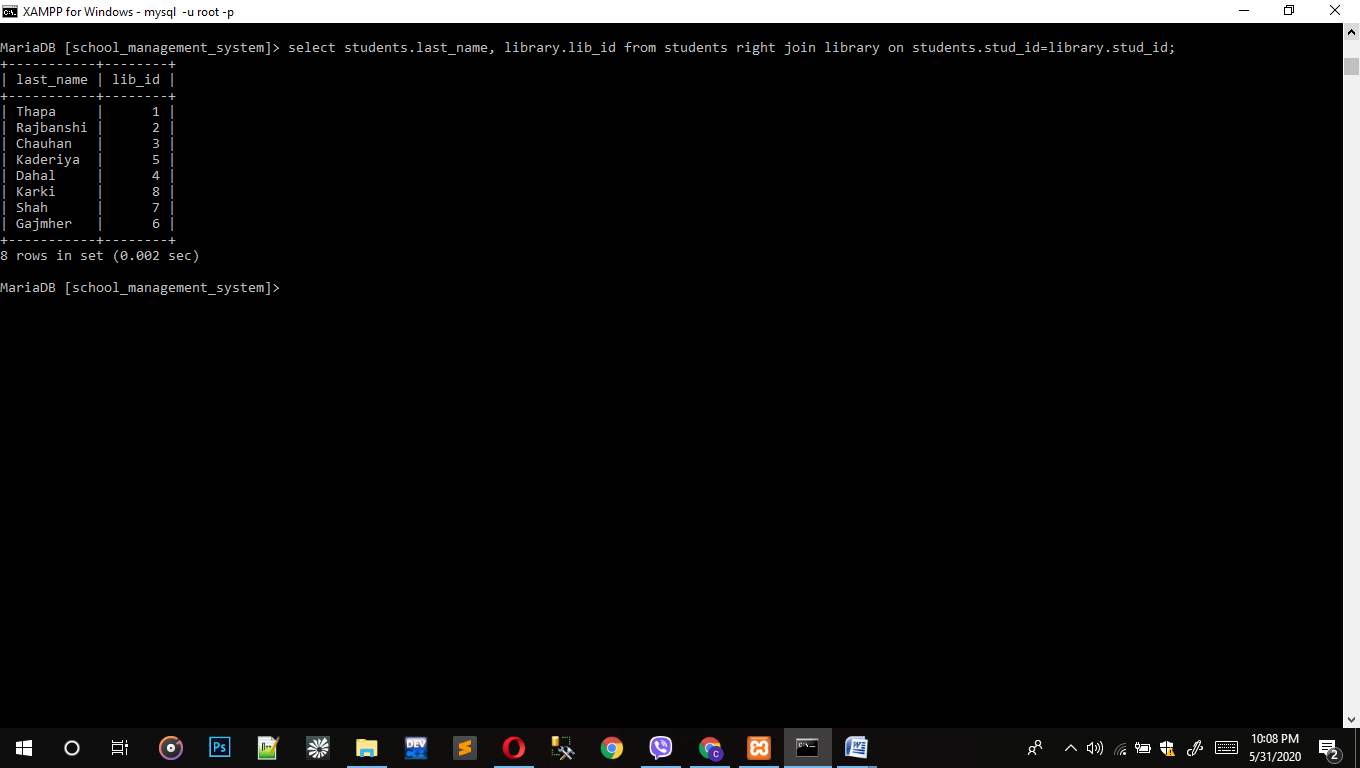
1. Query to left join table students and library.

* Select students.last\_name,library.lib\_id from students left join library on students.stud\_id=library.stud\_id;



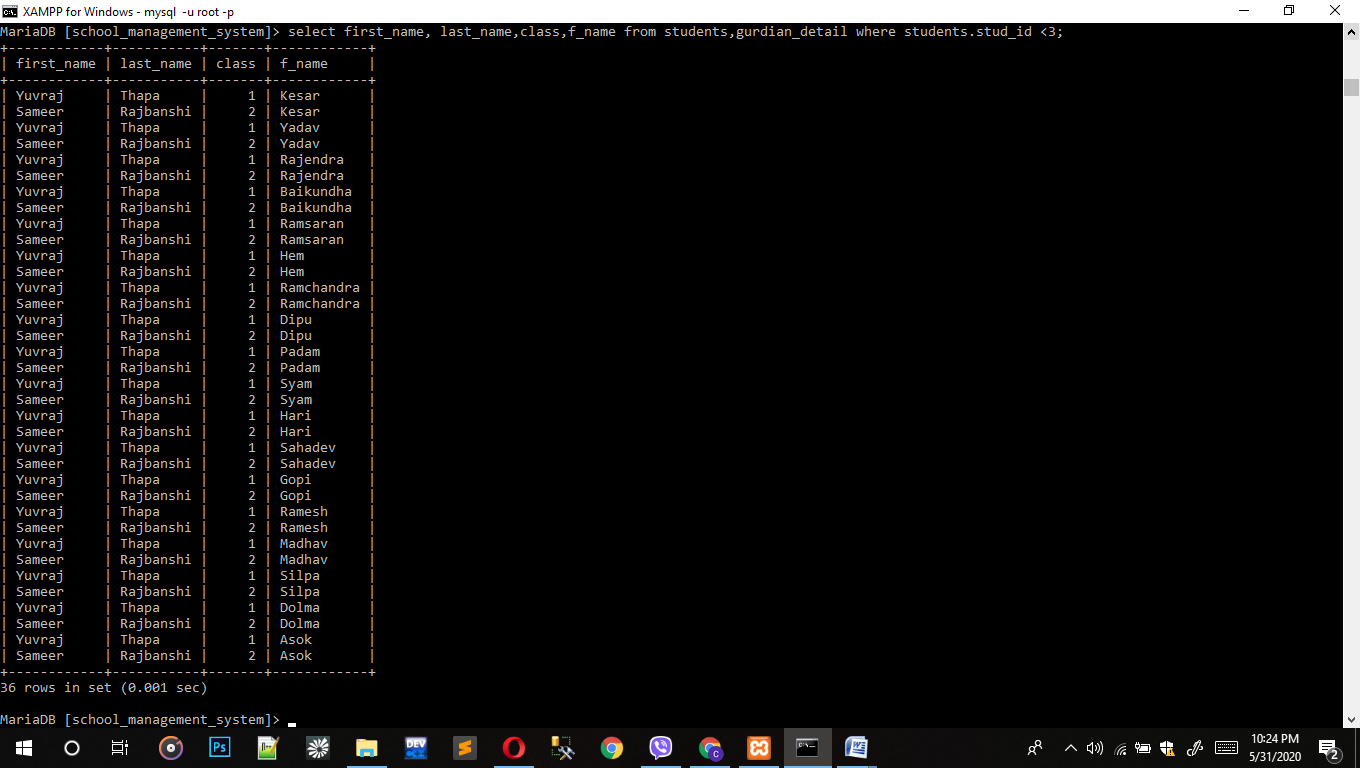
1. Query to left join table students and library.

* Select students.last\_name,library.lib\_id from students right join library on students.stud\_id=library.stud\_id;



1. Query to self join students and guardian detail.

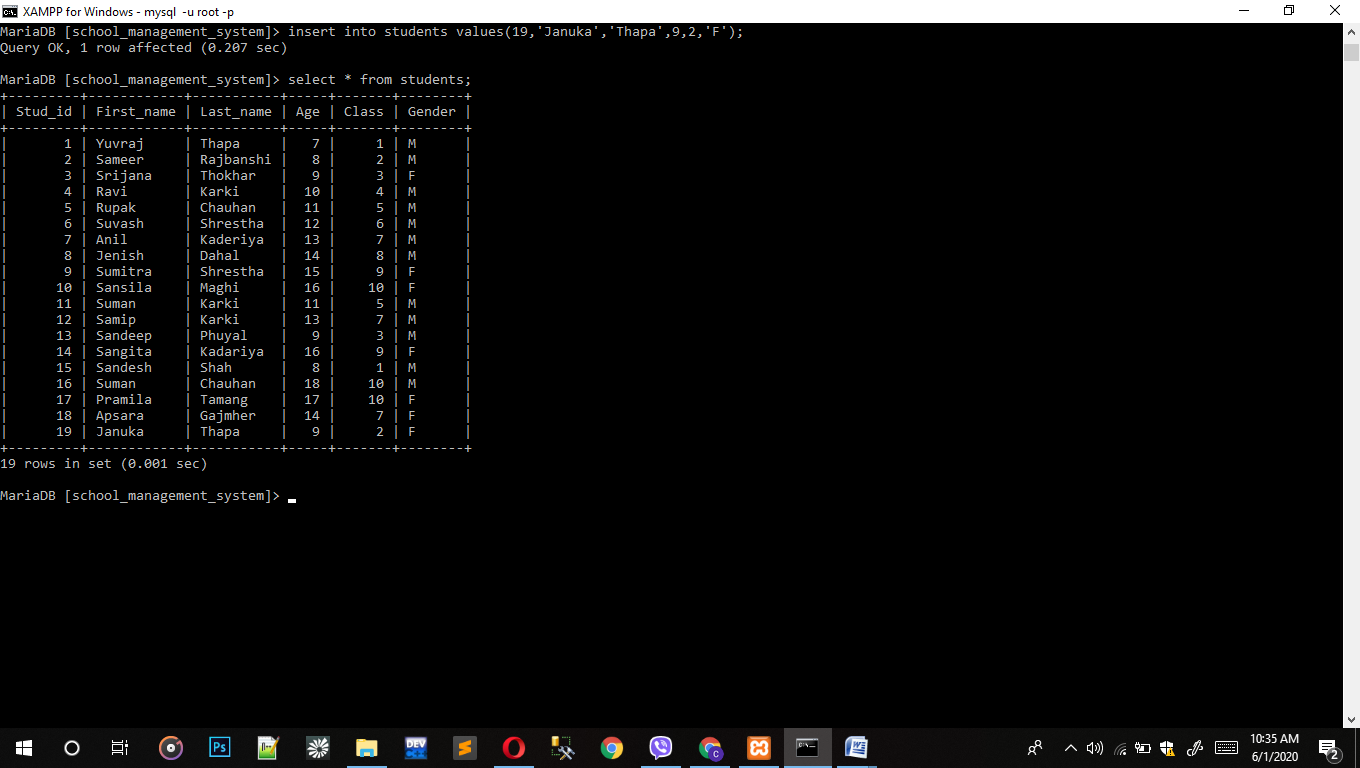
* Select first\_name, last\_name,class,f\_name from students,gurdian\_detail where students.stud\_id <3;



# Insert Statements

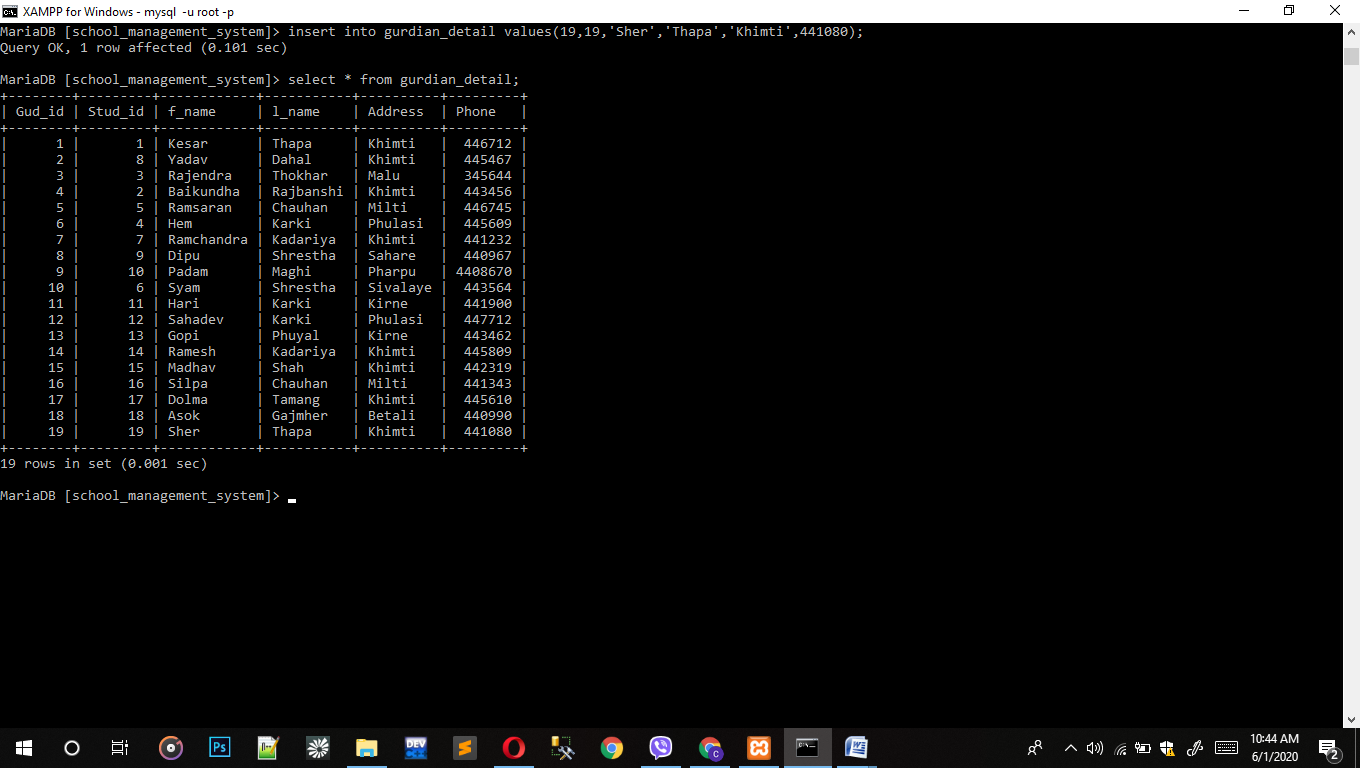
1. Insert into students.

* Insert into students values(19,'Januka','Thapa',9,2,'F');



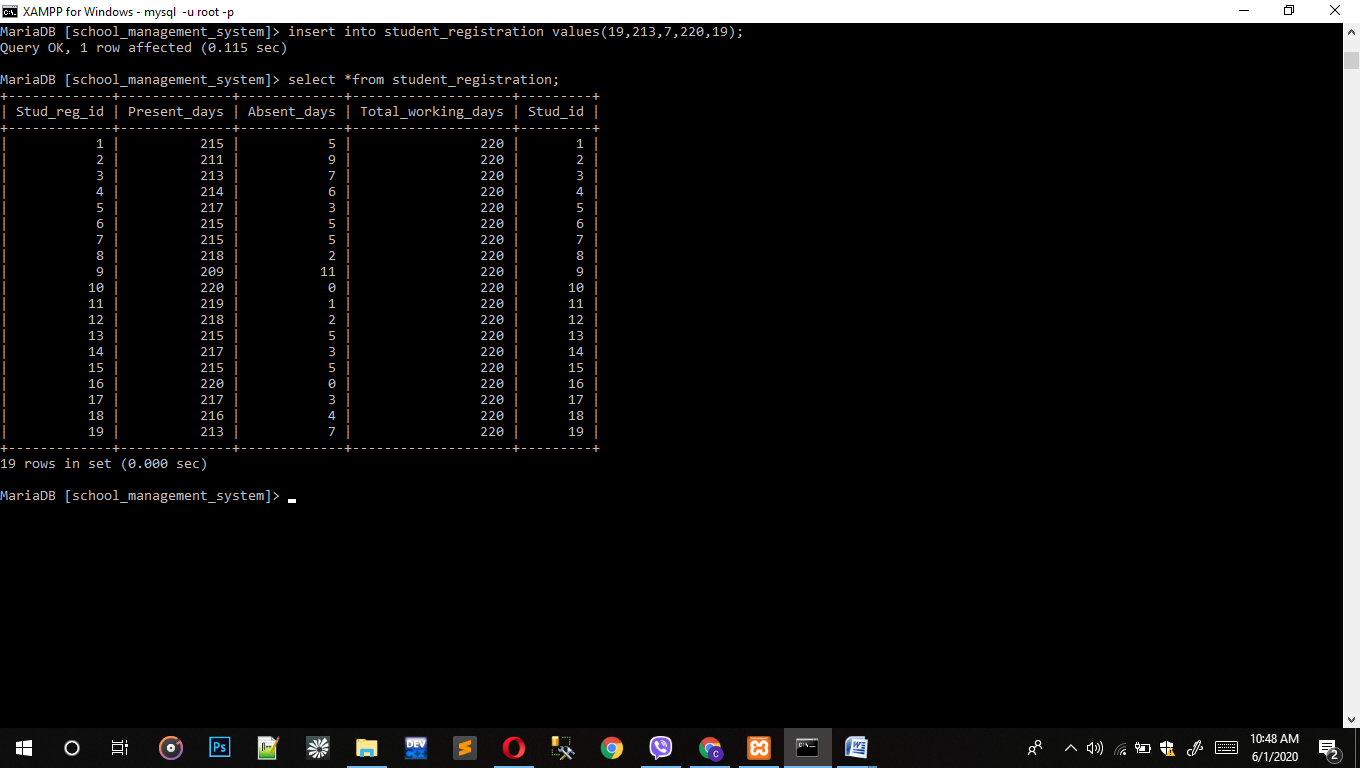
1. Insert into Gurdian\_detail.

* Insert into gurdian\_detail values(19,19,'Sher','Thapa','Khimti',441080);



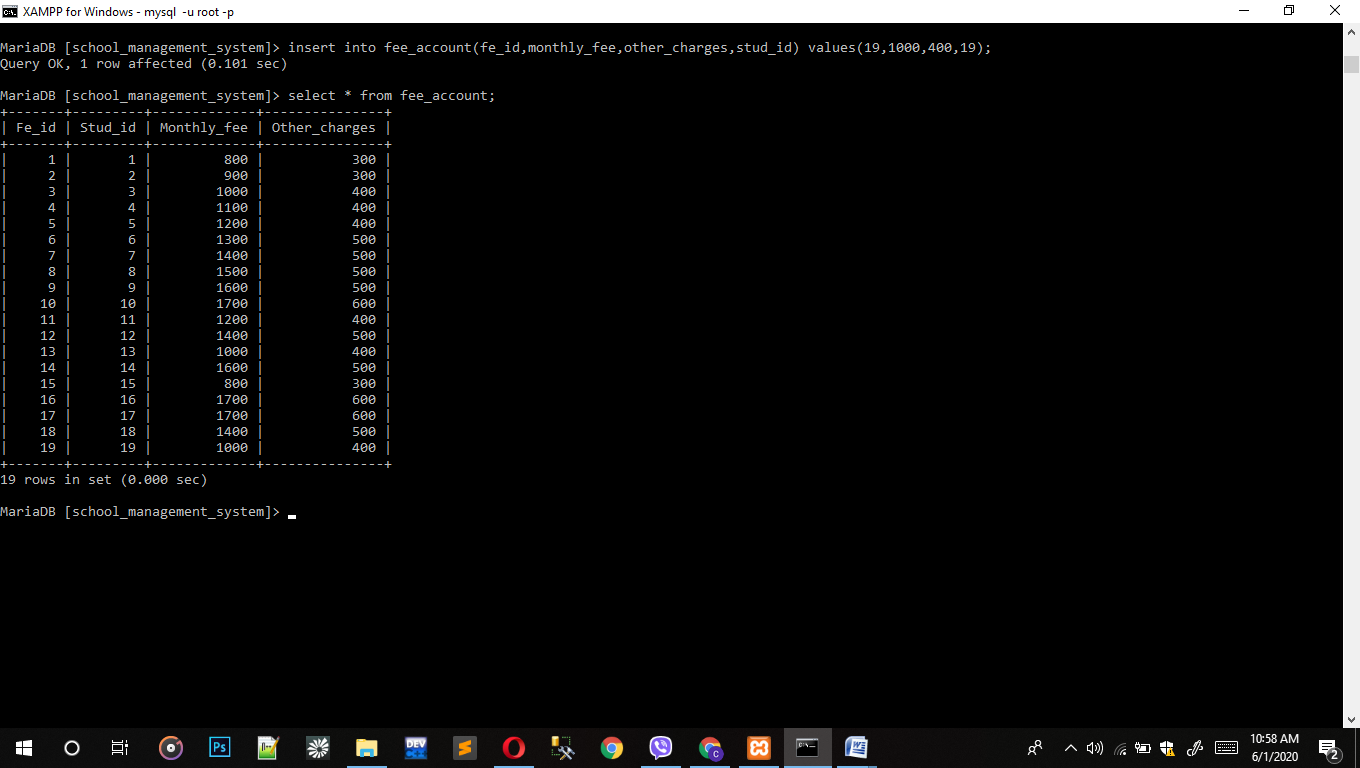
1. Insert into student registration.

* Insert into student\_registration values(19,213,7,220,19);



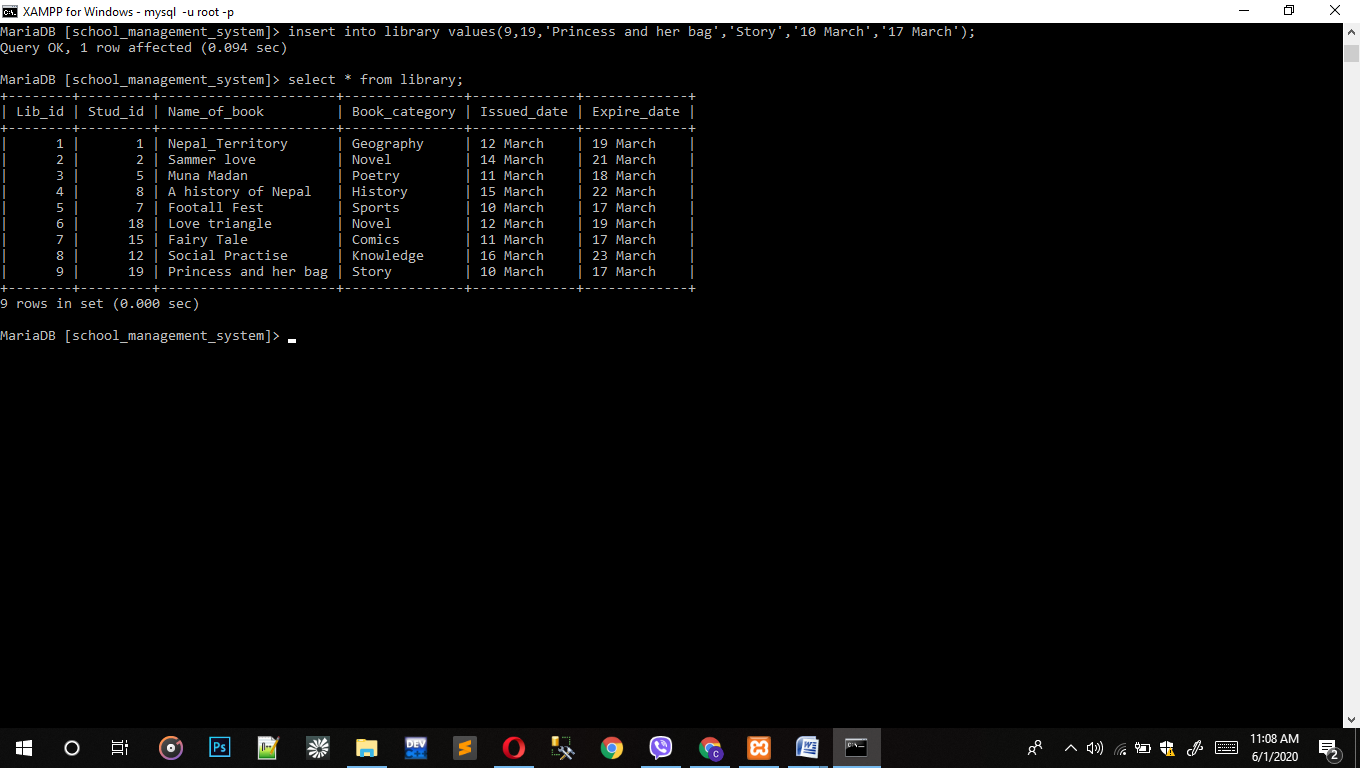
1. Insert into fee account.

* Insert into fee\_account(fe\_id,monthly\_fee,other\_charges,stud\_id) values(19,1000,400,19);



1. Insert into library.

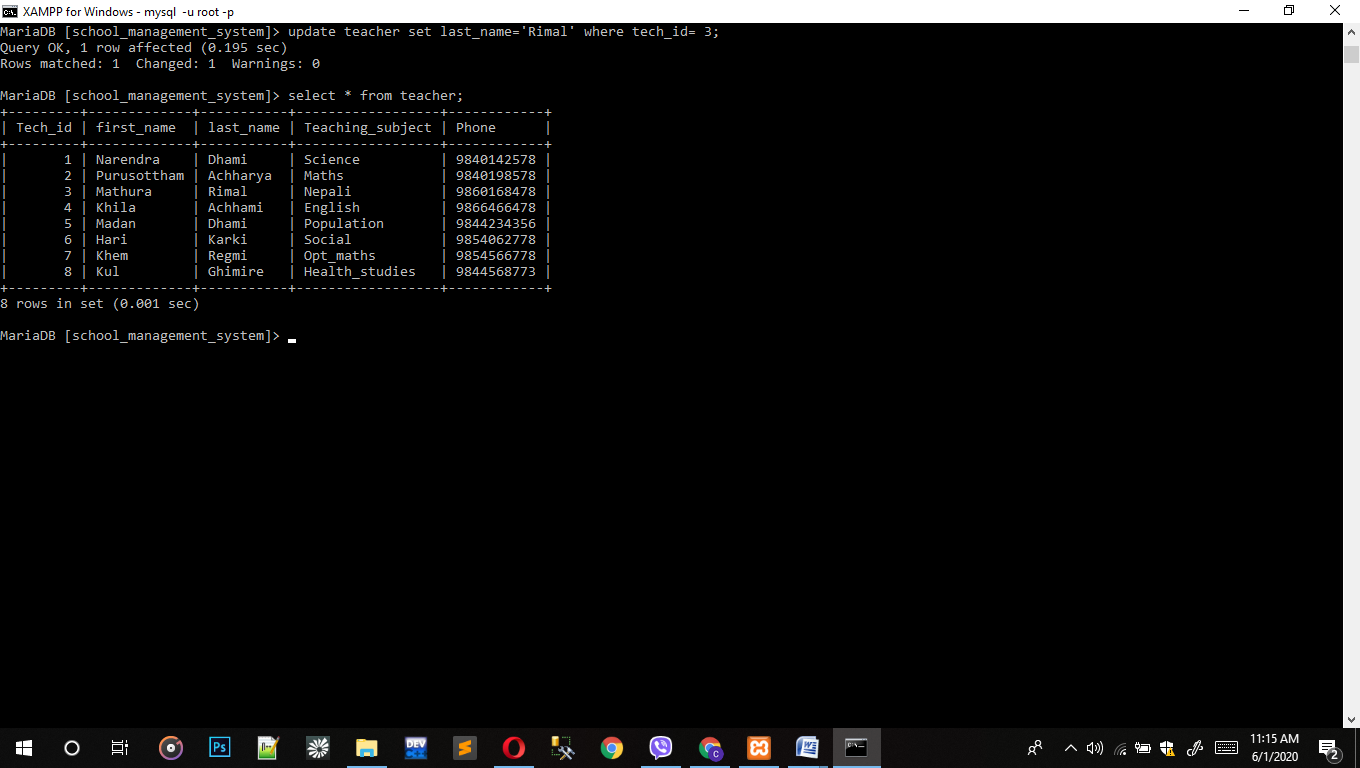
* Insert into library values(9,19,'Princess and her bag','Story','10 March','17 March');



# Update Statements

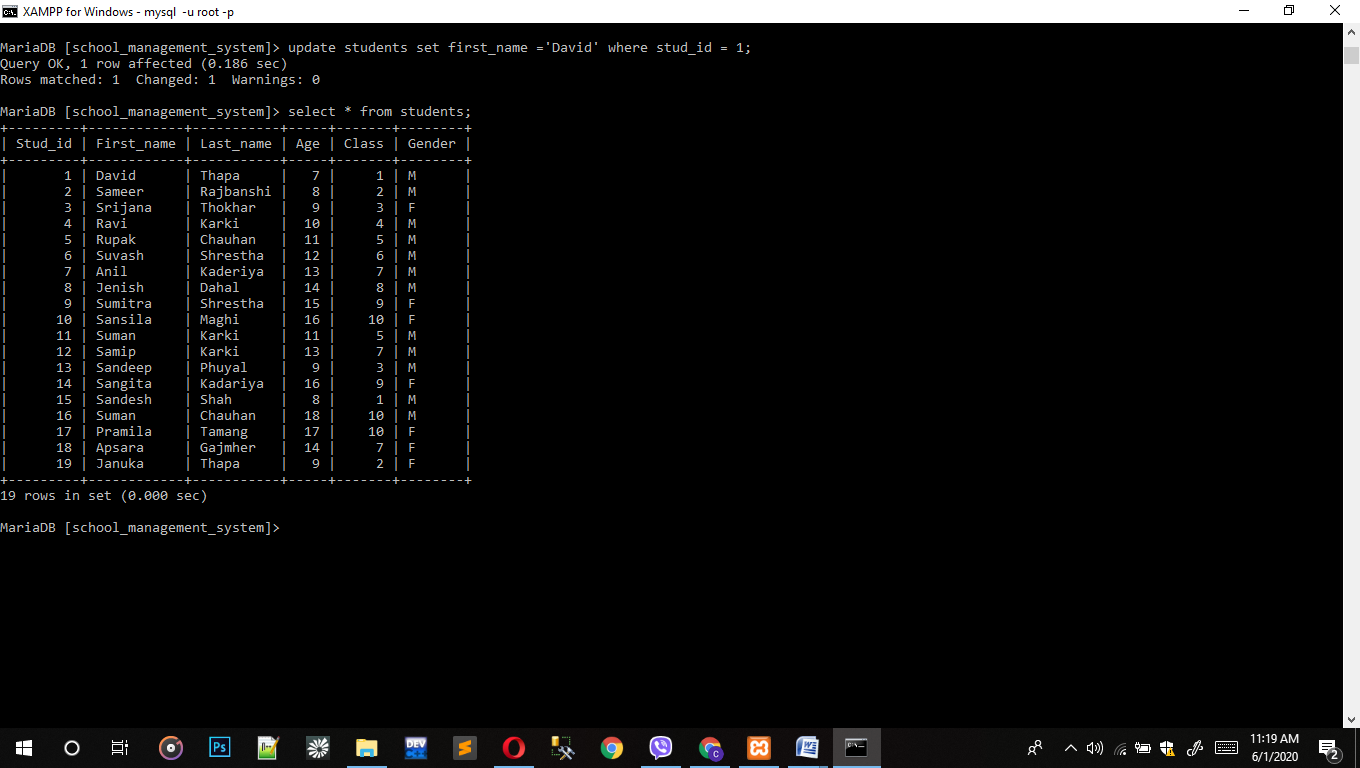
1. Update teacher.

* Update teacher set last\_name = 'Rimal' where tech\_id = 3;



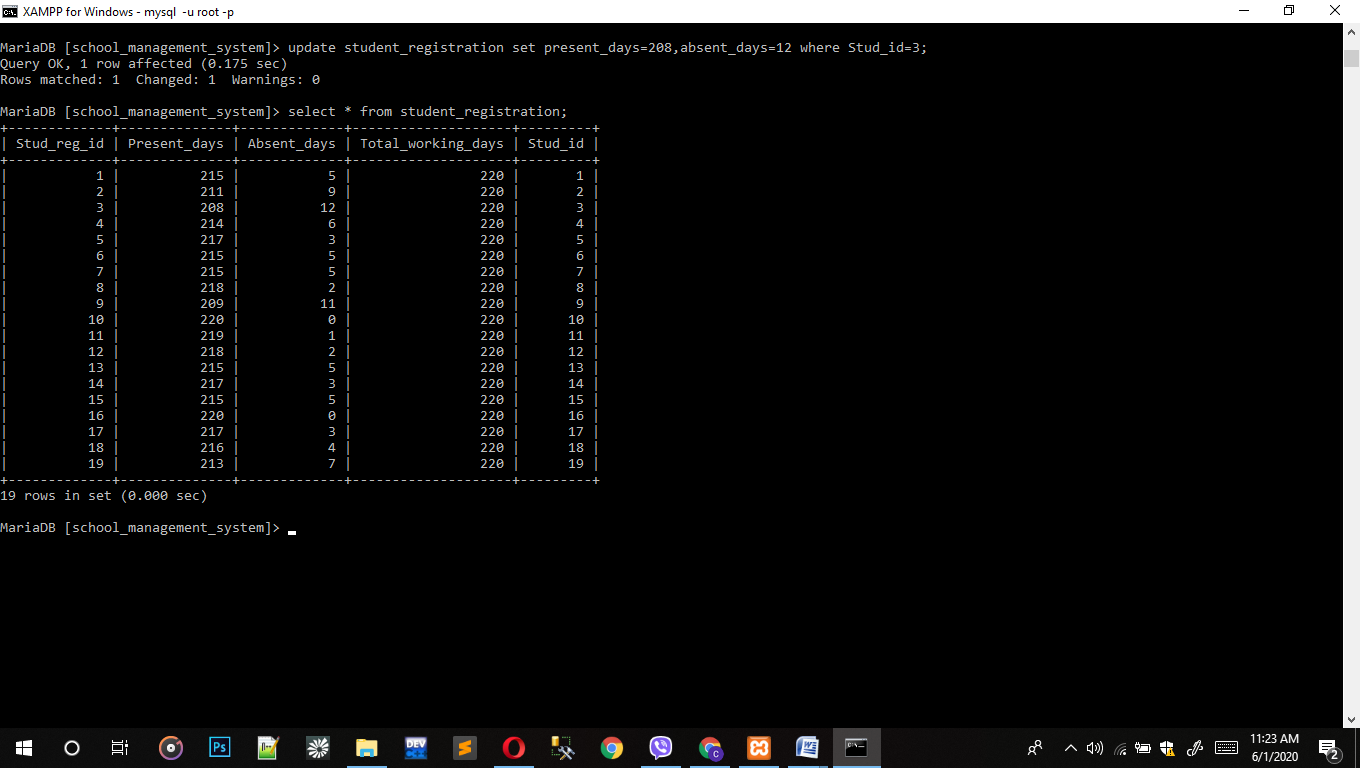
1. Update student.

* Update students set first\_name = 'David' where stud\_id = 1;



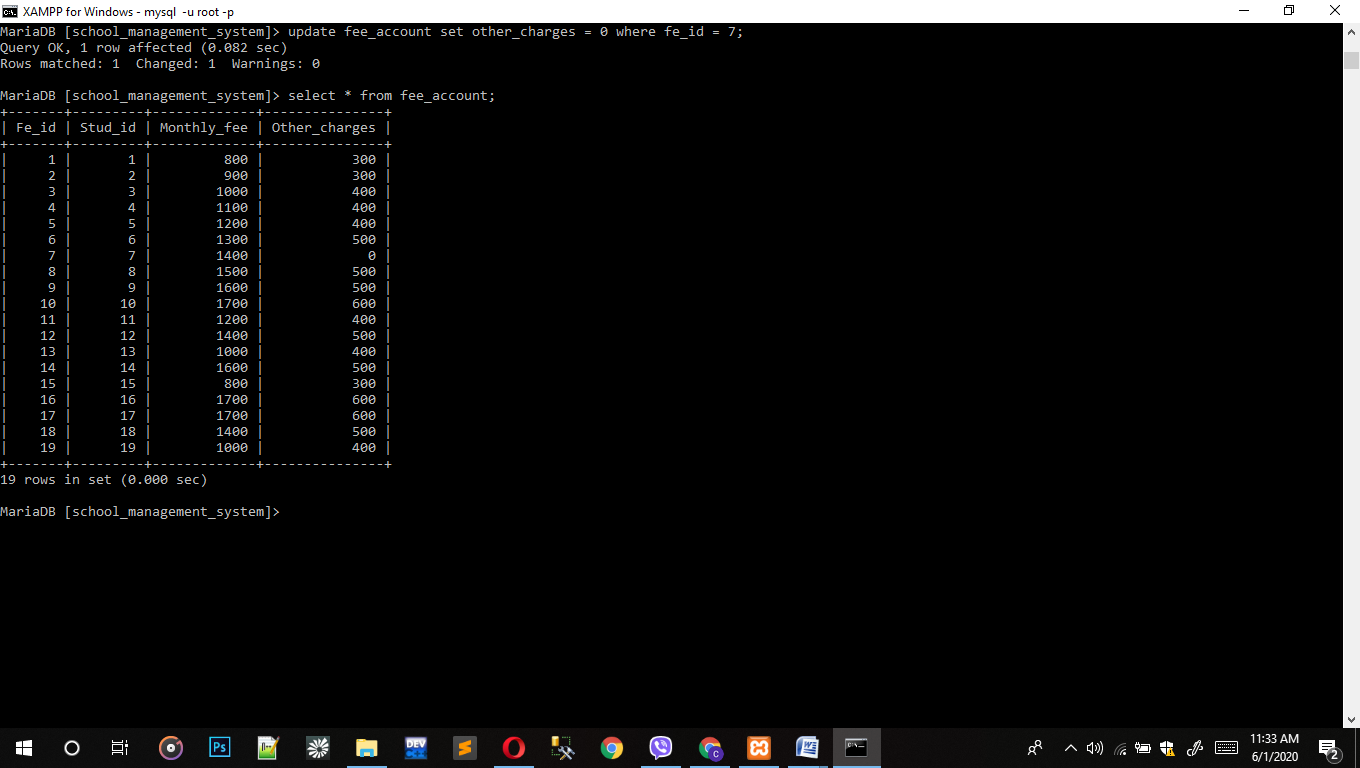
1. Update students registration.

* Update student\_registration set present\_days = 208, absent\_days = 12 where stud\_id=3;



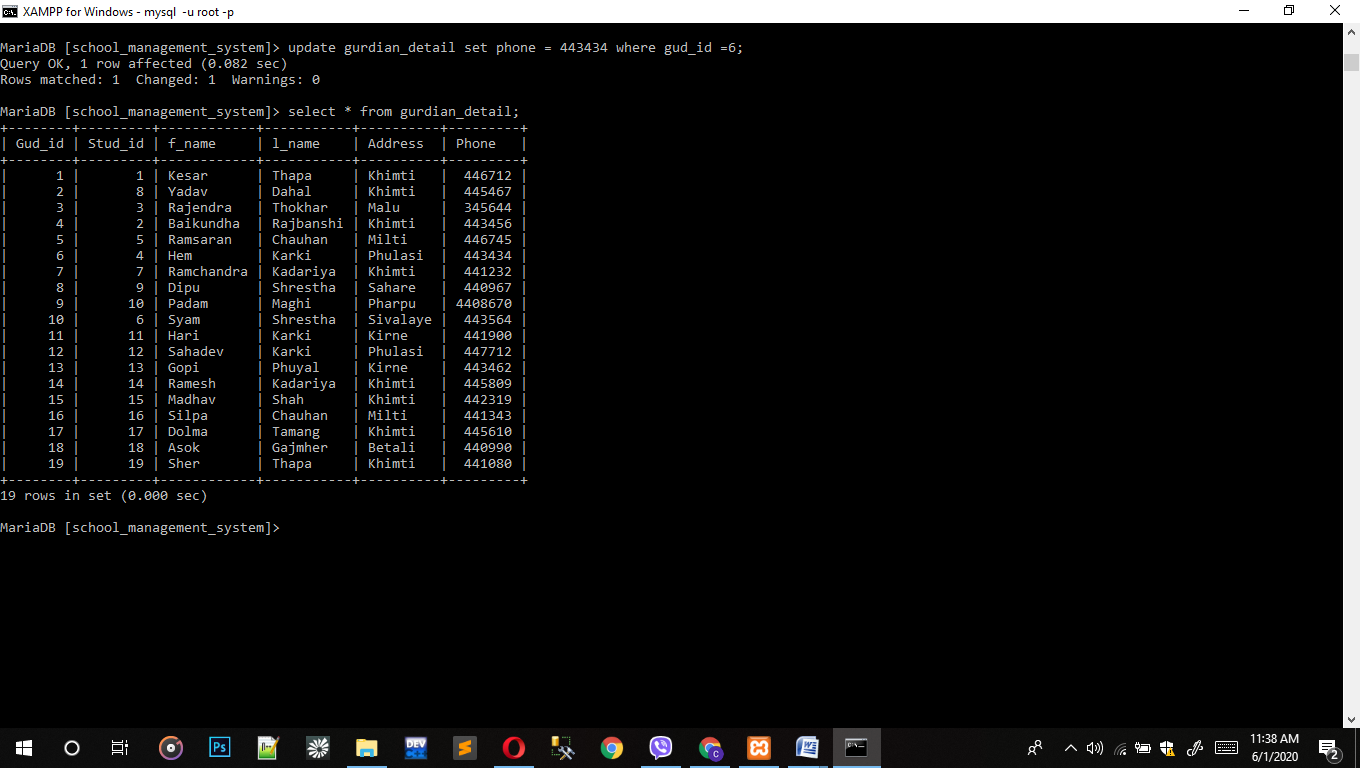
1. Update fee account

Update tfee\_account set other\_charges = 0 where fe\_id = 7;



1. Update Gurdian detail.

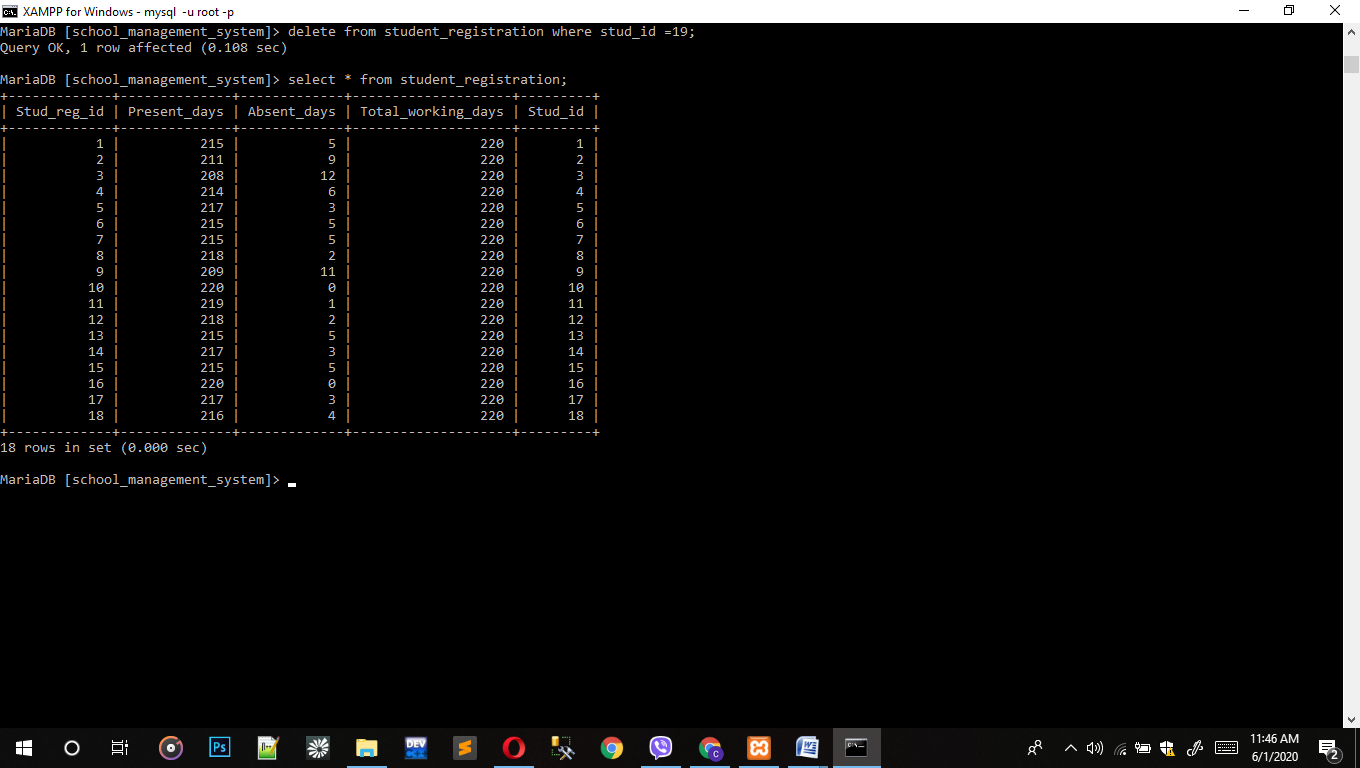
Update gurdian\_detail set phone = 443434 where gud\_id =6;



# Delete Statements

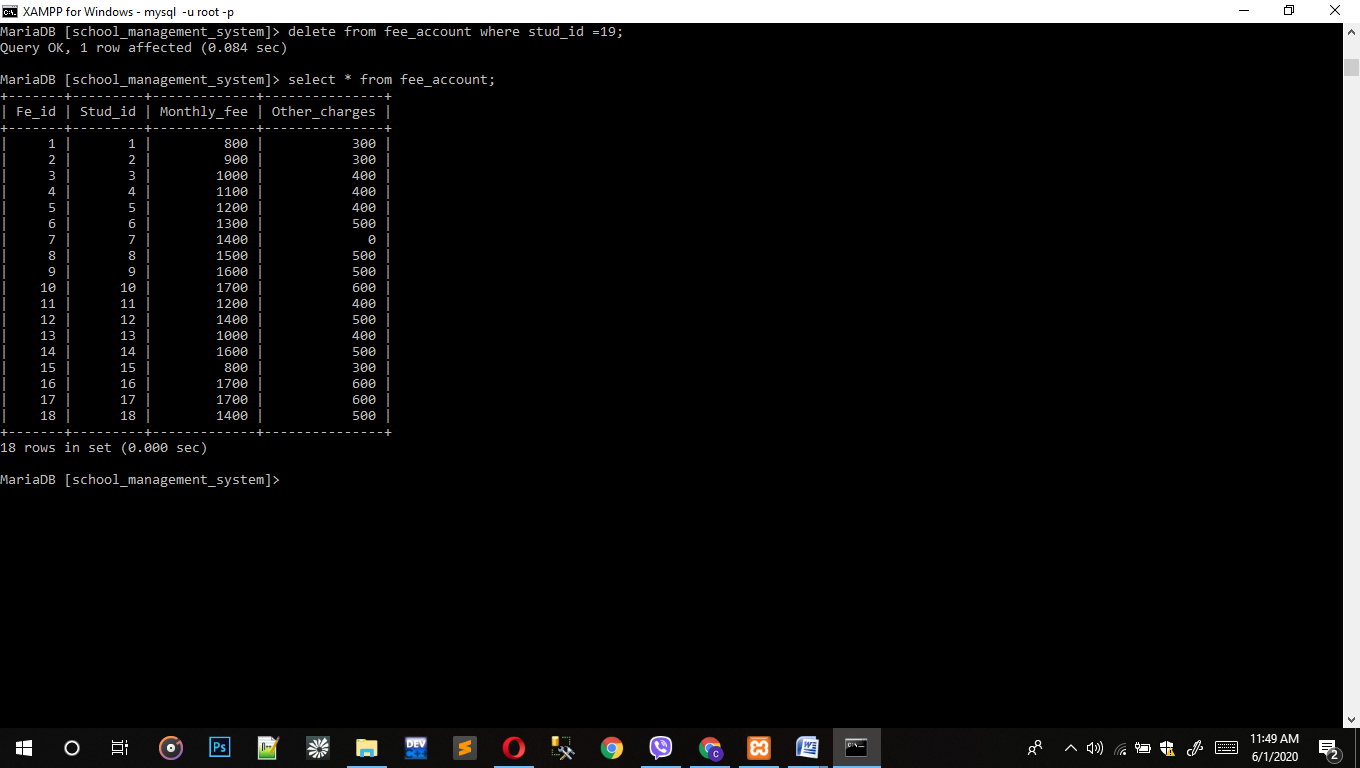
1. Delete from student registration.

* Delete from student\_registration where stud\_id = 19;



1. Delete from fee account

* Delete from fee\_accout where stud\_id = 19;



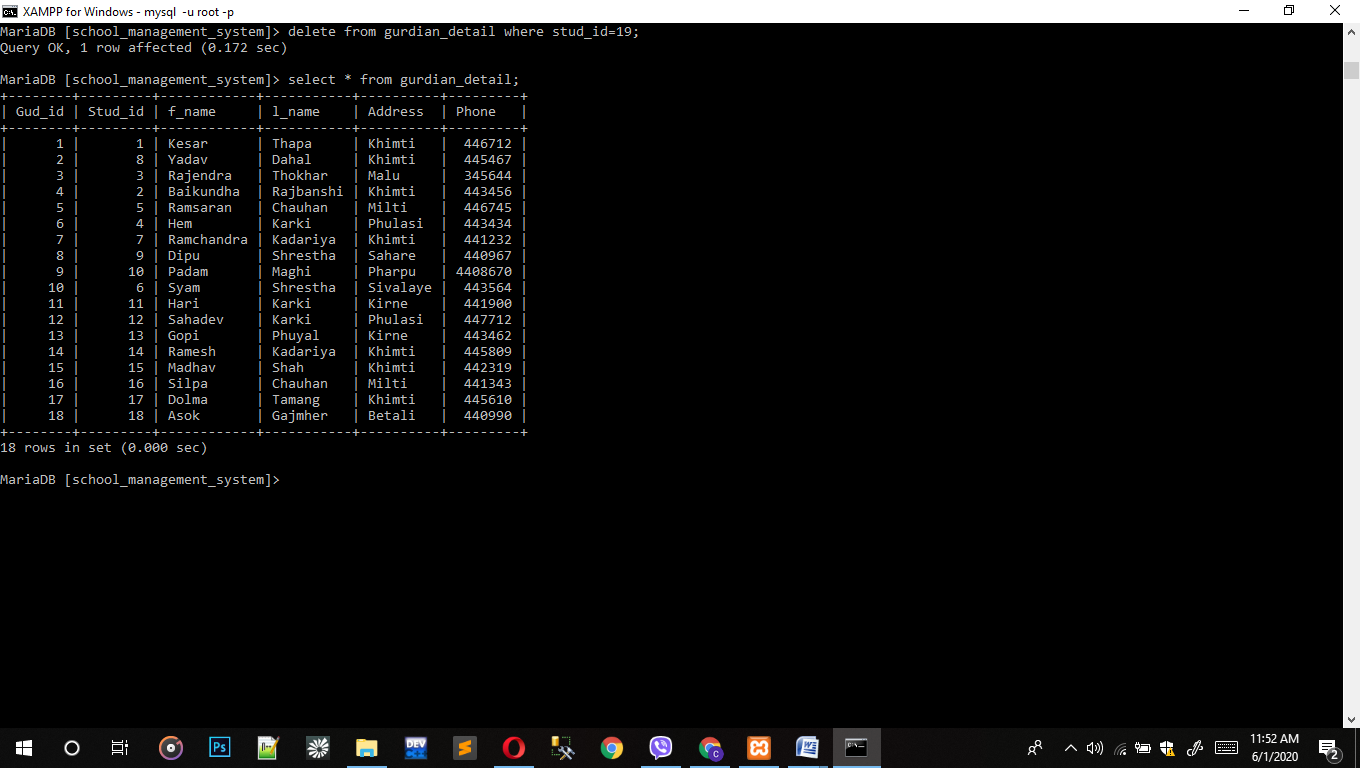
1. Delete from library.

* Delete from library where stud\_id = 19;



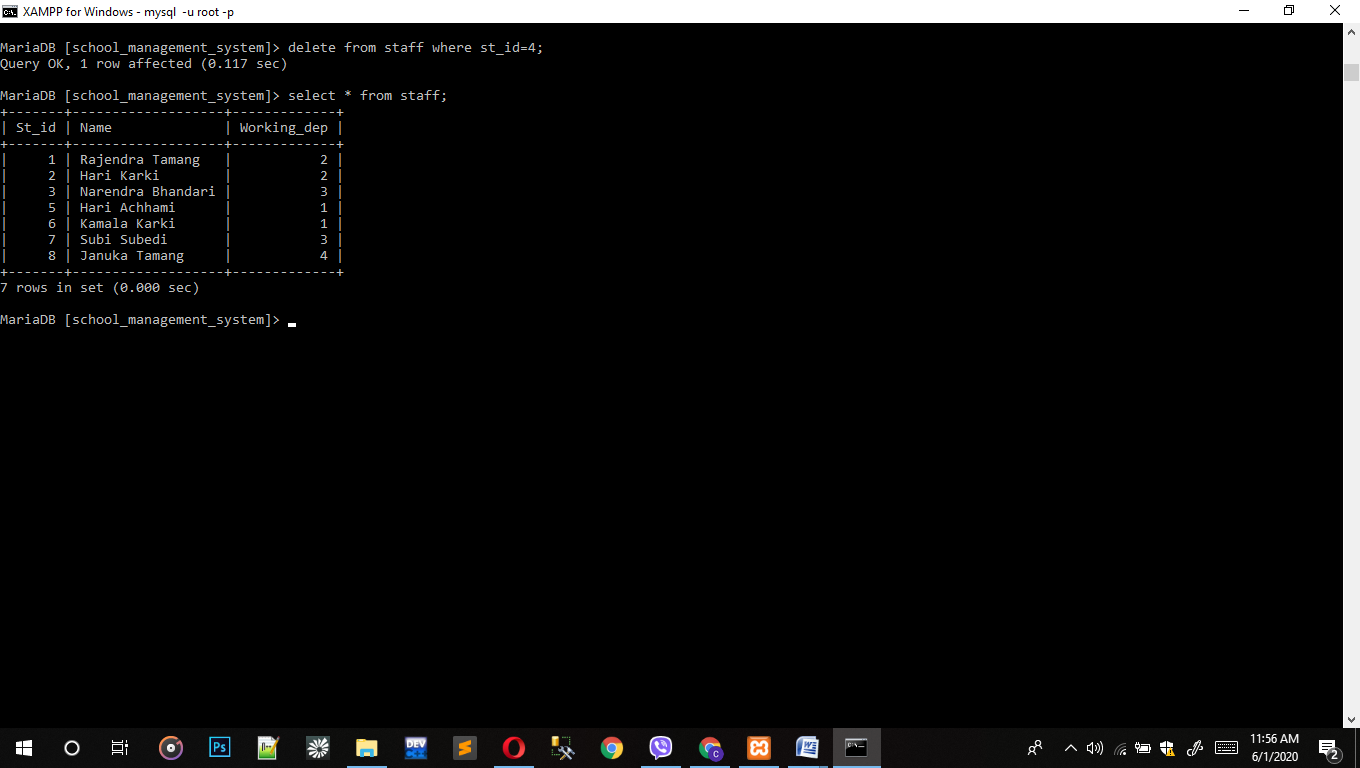
1. Delete from gurdian detail

* Delete from gurdian\_detail where stud\_id = 19;



1. Delete from staff

* Delete from staff where st\_id = 4;



# Normalization

Normalization in database is simply the technique to organize data of database. To define, Normalization is a disintegration of tables with the systematic way to eliminate data redundant and unwanted character like insertion, update and deletion. It follows different step to process in putting data into tabular form, then remove the duplicate from related tables. Normalization is divided according to its rules which are disused below:

## First Normal form (1NF)

For the 1NF, a table should have the following rules:

1. Table should have single valued attribute.
2. Domain should be same for the stored data in column.
3. Name should be unique for each column of tables.
4. The order of data store is not matter.

For examples, let us take a table having columns id, name, subject.

|  |  |  |
| --- | --- | --- |
| Id | Name | subject |
| 1 | Aron | Math |
| 2 | Sahil | Science |
| 3 | Subash | Population |
| 4 | Suman | Social |

This above tables follows the 1NF rule as it has satisfied the rules of 1NF.

## Second Normal form (2NF)

For the 2NF, a table should satisfy these conditions:

1. A table should be in 1NF.
2. Partial dependency is not allowed.

For example, let us take a table having columns st\_id, name, sub\_id,subject and teacher.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| St\_id | Name | Sub\_id | Score | subject | Teacher |
| 1 | Aaron | 1 | 434 | Math | Amar |
| 2 | Ankit | 3 | 398 | Science | Aahan |
| 3 | Anil | 2 | 443 | Social | Nelson |

In above table St\_id + Sub\_id are candidate key which can be the primary key. This table satisfies the 1Nf rule but it is partially dependent because teacher is directly related to subject.

|  |  |  |
| --- | --- | --- |
| St\_id | Name | score |
| 1 | Aaron | 434 |
| 2 | Ankit | 398 |
| 3 | Anil | 443 |

|  |  |  |
| --- | --- | --- |
| Sub\_id | subject | Teacher |
| 1 | Math | Amar |
| 3 | Science | Aahan |
| 2 | social | Nelson |

(I) (II)

Now the table (I) follows the rule of 2NF by separating teacher into another table.

## 3. Third Normal form (3NF)

For the 3NF, a table should satisfy these conditions:

1. It should be in 2NF.
2. It does not have transitive dependency.

For example, let take a table of columns book\_Id, genre\_Id, genre\_Type, price.

|  |  |  |  |
| --- | --- | --- | --- |
| Book\_Id | Genre\_Id | Genre\_type | Price |
| 1 | 1 | History | 200 |
| 2 | 2 | Story | 300 |
| 3 | 1 | History | 500 |

This table is transitive dependent because book id define genere id, genre id defines genre type and book id defines genre type via genre id. So if we decompose the table as book and genre in table as given below:

Book Genre

|  |  |  |
| --- | --- | --- |
| Book\_Id | Genre\_Id | price |
| 1 | 1 | 200 |
| 2 | 2 | 300 |
| 3 | 1 | 500 |

|  |  |
| --- | --- |
| Genre\_Id | Genre\_type |
| 1 | History |
| 2 | Story |
| 1 | History |

After break down it do not follow the transitive dependency and also it is in 2NF. So this table follows the rules of 3NF.