VIT®
BHOPAL
www.vitbhopal.ac.in

REPORT

PROJECT : RECORD MANGEMENT SYSTEM (LIST OF CANDIDATES)

MADE BY :-

SALAM KHAN

25BAI10036

B.TECH COMPUTER SCIENCE ENGINEERING (AIML)

Index

Sno.	Topic	Page no.
1	System requirements	3
2	Feasibility study	4
3	Errors and its types	5
4	Testing	6
5	Maintenance	7
6	Flow chart of program	8
7	Code	9
8	Output	16
9	Appendix	27
10	Bibliography	28

System Requirements

1. **HARDWARE:**
   - ✓ Processor
   - ✓ Keyboard
   - ✓ Minimum memory - 2GB
2. **SOFTWARE:**
   - ✓ Operating System –OS7, OS8
   - ✓ Python IDLE
   - ✓ MYSQL

Feasibility Study

Feasibility study is a system proposal according to its work, ability, impact on the operation ability to meet the needs of users and efficient use of resources. An important outcome of preliminary investigations the determination of that system requested feasible.

ECONOMICAL FEASIBILITY:

Economics analysis is the most frequent use method for evaluating the effectiveness of the candidates the benefits and savings that are expected from system and compare them with cost.

This software is not very costly. It just worth Rs.5500/-.So users records can be maintained at a cheaper cost and every school would like to use this software so that the student's records can be managed easily.

TECHNICAL FEASIBILTY:

Technical feasibility center on the existing computer system and to what extent it can support the proposed task. This involves financial consideration to accommodate technical enhancements.

It is technically feasible because whatever technology is needed to develop this software is easily available.

Error and its Types

An error, sometime called "A BUG" is anything in the code that prevents a program from compiling and running correctly. There are broadly three types of errors as follows:

1. **Compile- time errors**: Errors that occurs during compilation of a program is called compile time error. It has two types as follows:
    a. **Syntax error**: It refers to formal rules governing the construction of valid statements in a language.
    b. **Semantics error**: It refers to the set of rules which give the meaning of a statement.
2. **Run time Errors:** Errors that occur during the execution of program are run time errors. These are harder to detect errors. Some run-time error stop the execution of program which is then called program "Crashed".
3. **Logical Errors:** Sometimes, even if you don't encounter any error during compiling-time and runtime, your program does not provide the correct result. This is because of the programmer's mistaken analysis of the problem he or she is trying to solve. Such errors are called logical error.

TESTING

1. **Alpha Testing**: It is the most common type of testing used in the software industry. The objective of this testing is to identify all possible issues or defects before releasing it into the market or to the user. It is conducted at the developer's site.
2. **Beta Testing**: It is a formal type of software testing which is carried out by the customers. It is performed in a real environment before releasing the products into the market for the actual end-users. It is carried out to ensure that there are no major failures in the software or product and it satisfies the business requirement. Beta Testing is successful when the customer accepts the software.
3. **White Box Testing**: White box testing is based on the knowledge about the internal logic of an application's code. It is also known as Glass box Testing. Internal Software and code working should be known for performing this type of testing. These tests are based on the coverage of the code statements, branches, paths, conditions etc.
4. **Black Box Testing**: It is a software testing, method in which the internal structure or design of the item to be tested is not known to the tester. This method of testing

can be applied virtually to every level of the software testing.

Maintenance

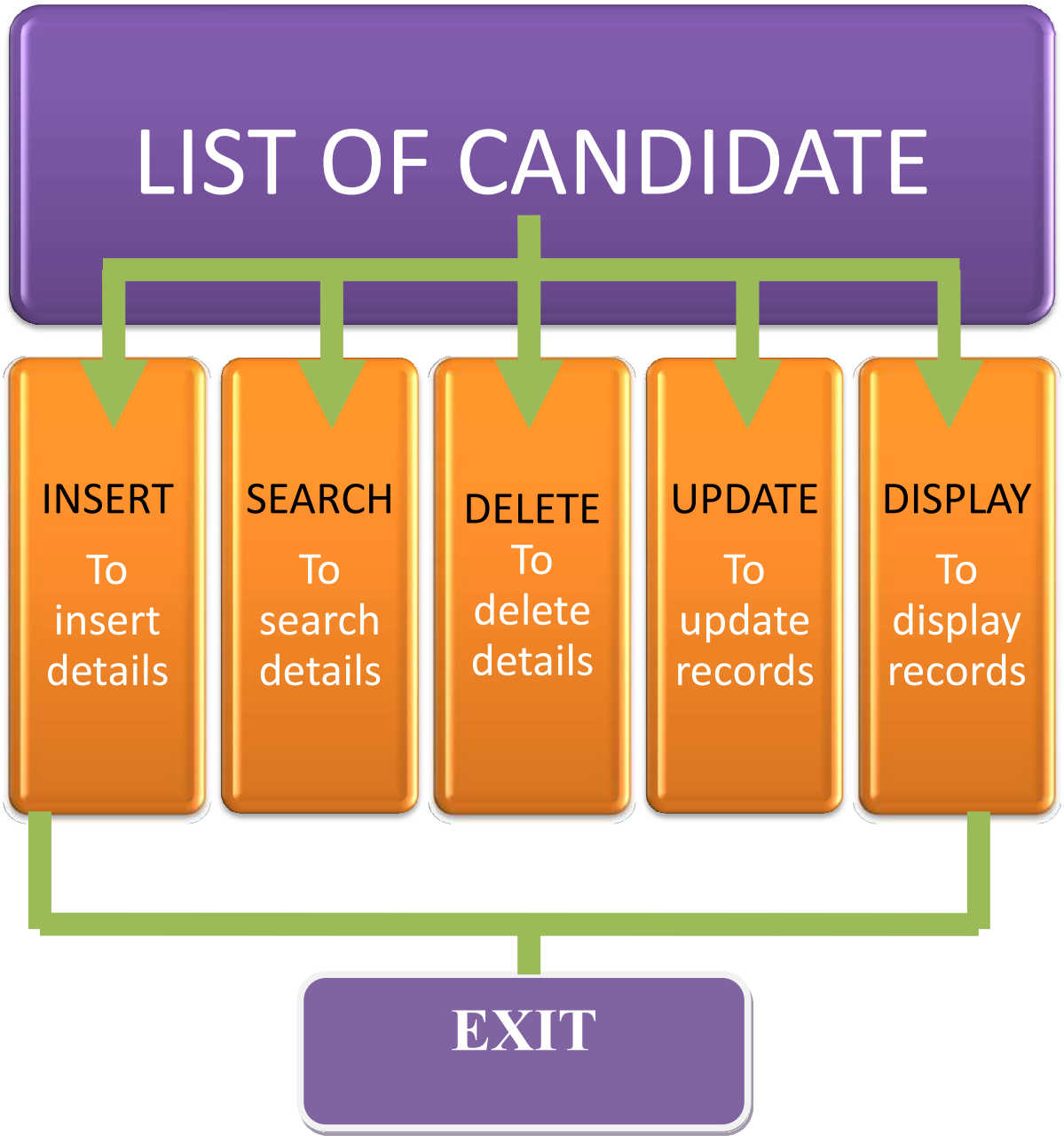
Programming maintenance refers to the modifications in the program. After it has been completed, in order to meet changing requirement or to take care of the errors that shown up. There are four types of maintenance:

1. **Corrective Maintenance**: When the program after compilation shows error because of some unexpected situations, untested areas such errors are fixed up by Corrective maintenance.
2. **Adaptive Maintenance**: Changes in the environment in which an information system operates may lead to system management. To accommodate changing needs time to time maintenance is done and is called Adaptive maintenance.
3. **Preventive Maintenance**: If possible the errors could be anticipated before they actually occur; the maintenance is called Preventive maintenance.

Perfective Maintenance: In this rapidly changing world, information technology is the fastest growing area. If te existing system is maintained to keep tuned with the new

features, new facilities, new capabilities, it is said to be Perfective maintenance

Flow Chart of the Program



Code

```
import mysql.connector as mcon

import sys

con = mcon.connect (host="localhost" ,port="3306" ,user="root"  ,passwd="root")

mycursor = con.cursor()

if con.is_connected():

        print("MySql DataBase is connected Successfully.")

        mycursor.execute("create database if not exists LOC")

        mycursor.execute("use LOC")

        mycursor.execute("create table if not exists user \              (uname varchar(20)
 primary key,upwd varchar(20)\

,utype char(5),ustatus char(5))")

      Q = "insert into user(uname,upwd,utype) values (\'LOC\',\'LOC\',\'S\')"

      #print(Q)

      #mycursor.execute(Q)

      con.commit()

      at = 1

      while at <= 3:

            at += 1

            uid = input("Enter User Name : ")

            pwd = input("Enter User Password : ")
```

```
        status = 'A'
        mycursor.execute("select * from user where uname = '{}' and upwd = '{}' and
ustatus = '{}'".format(uid,pwd,status))
        data = mycursor.fetchone()
        count = mycursor.rowcount
        #print(count)
        if count == 1:
        print("Login Successfully.")
        print("Perform CRUD Operations.")
    #-------------------------*CHOICES*----------------------------
        while True:
            print("Input 'I' for Insertion a New Record.")
              print("Input 'U' for Update an Existing Record.")
              print("Input 'R' for Removal an Existing Record.")
            print("Input 'S' for Searching a Record.")
            print("Input 'D' for Display All Records.")
            print("Input 'E' for Exit the Program.")
            ch = input("Enter Your Option: ")
  #-------------------------*TABLE CREATION*----------------------------
            if ch == 'I' or ch == 'i':
                ins = "create table if not exists students(\
  reg_num int(20) primary key, loc_sr_num integer NOT NULL, yr_pass_xi int(5) NOT
NULL, exam_cat char(5) NOT NULL, cand_name char(50) NOT NULL, mother_name
```

```
char(50) NOT NULL, father_name char(50) NOT NULL, gender varchar(5), category1
varchar(5), minority varchar(5), PwD_status varchar(20), mob_num bigint NOT NULL,
email_id varchar(50), aadhar_num bigint, sub_1 char(15), sub_2 char(15) NOT NULL,
sub_3 char(15) NOT NULL, sub_4 char(15) NOT NULL, sub_5 char(15) NOT NULL,
add_sub_6 char(15) NOT NULL, int_grade_sub1 char(30), int_grade_sub2 char(30),
int_grade_sub3 char(30), annual_income varchar(25), roll_num_of_equi_exam_passed
integer, exam_of_equi_exam_passed char(20), board_of_equi_exam_passed char(20),
single_child char(5), migration_certificate char(5), adm_no integer, adm_date date)"

                #print(ins)

                mycursor.execute(ins)

#-------------------------*INSERTION OF RECORDS*-------------------

                print("Insertion Operation.")

                reg = int(input("Enter student's registration_num: "))

                locsr = int(input("Enter student's loc_sr_num: "))

                yrpassc11 = int(input("Enter student's year_passing_class11: "))

                ecat = input("Enter student's exam_cat: ")

               cname = input("Enter student's Name: ")

               mname = input("Enter student's mother's name: ")

               fname = input("Enter student's father's name: ")

               gender = input("Enter student's gender: ")

               cat = input("Enter student's category: ")

               minor = input("Enter if student belongs to minority section(y/n): ")

               pwdis = input("Enter if student have disability (type of disability): ")

              mnum = int(input("Enter student's mobile_num: "))
```

```
email = input("Enter student's email_id: ")
ad_num = int(input("Enter student's addhar number: "))
s1 = input("Enter subject1(compulsory language): ")
s2 = input("Enter subject2: ")
s3 = input("Enter subject3: ")
s4 = input("Enter subject4: ")
s5 = input("Enter subject5: ")
s6 = input("Enter subject6(additional): ")
intsub1 = input("Enter name of internal grade subject1: ")
intsub2 = input("Enter name of internal grade subject2: ")
intsub3 = input("Enter name of internal grade subject3: ")
aninc = int(input("Enter annual income of student's parents: "))
eexrnum = int(input("Enter student's rollnum of equivalent exam passed:"))
eexam = input("Enter student's exam of equivalent exam passed:")
eexboard = input("Enter student's board of equivalent exam passed:")
sch = input("Enter if student is single girl child or not:")
mgcr = input("Enter if migration certificate is required or not:")
adm_num = int(input("Enter student's admission num:"))
adm_date = input("Enter student's admission date as (yyyy-mm-dd):")

q = "insert into students (reg_num, loc_sr_num,\ yr_pass_xi,exam_cat,
cand_name, mother_name, father_name, gender,\ category1, minority, PwD_status,
mob_num, email_id, aadhar_num,\ sub_1, sub_2, sub_3, sub_4, sub_5, add_sub_6,
```

```
int_grade_sub1,\ int_grade_sub2, int_grade_sub3, annual_income,\
roll_num_of_equi_exam_passed, exam_of_equi_exam_passed,\
board_of_equi_exam_passed, single_child, migration_certificate,\ adm_no, adm_date)
values ({}, {}, {}, '{}', '{}', '{}', '{}', '{}', '{}', '{}', '{}', {}, '{}', {}, '{}', '{}', '{}', '{}', '{}', '{}',
'{}', '{}', '{}', {}, {}, '{}', '{}', '{}', '{}', {}, '{}') ".format (reg, locsr, yrpassc11, ecat, cname,
mname, fname, gender, cat, minor, pwdis, mnum, email, ad_num ,s1, s2, s3, s4, s5, s6,
intsub1, intsub2, intsub3, aninc, eexrnum, eexam, eexboard, sch, mgcr, adm_num,
adm_date)

                mycursor.execute(q)

                con.commit()

                print("Record is inserted Successfully.")

#-------------------------*UPDATION*----------------------------

            elif ch == 'U' or ch == 'u':

                print("Updation of Record.")

                reg = input("Enter Student's registration Number: ")

                sn = input("Enter New student's Name: ")

                mn = input("Enter New student Mother's Name: ")

                fn = input("Enter New student Father's Name: ")

                qry = "update students set cand_name = '{}', mother_name = '{}' ,
 father_name = '{}' where reg_num = {}".format(sn,mn,fn,reg)

                mycursor.execute(qry)

                con.commit()

                print("Record is updated Successfully.")

#-------------------------*DELETION*----------------------------

            elif ch == 'R' or ch == 'r':
```

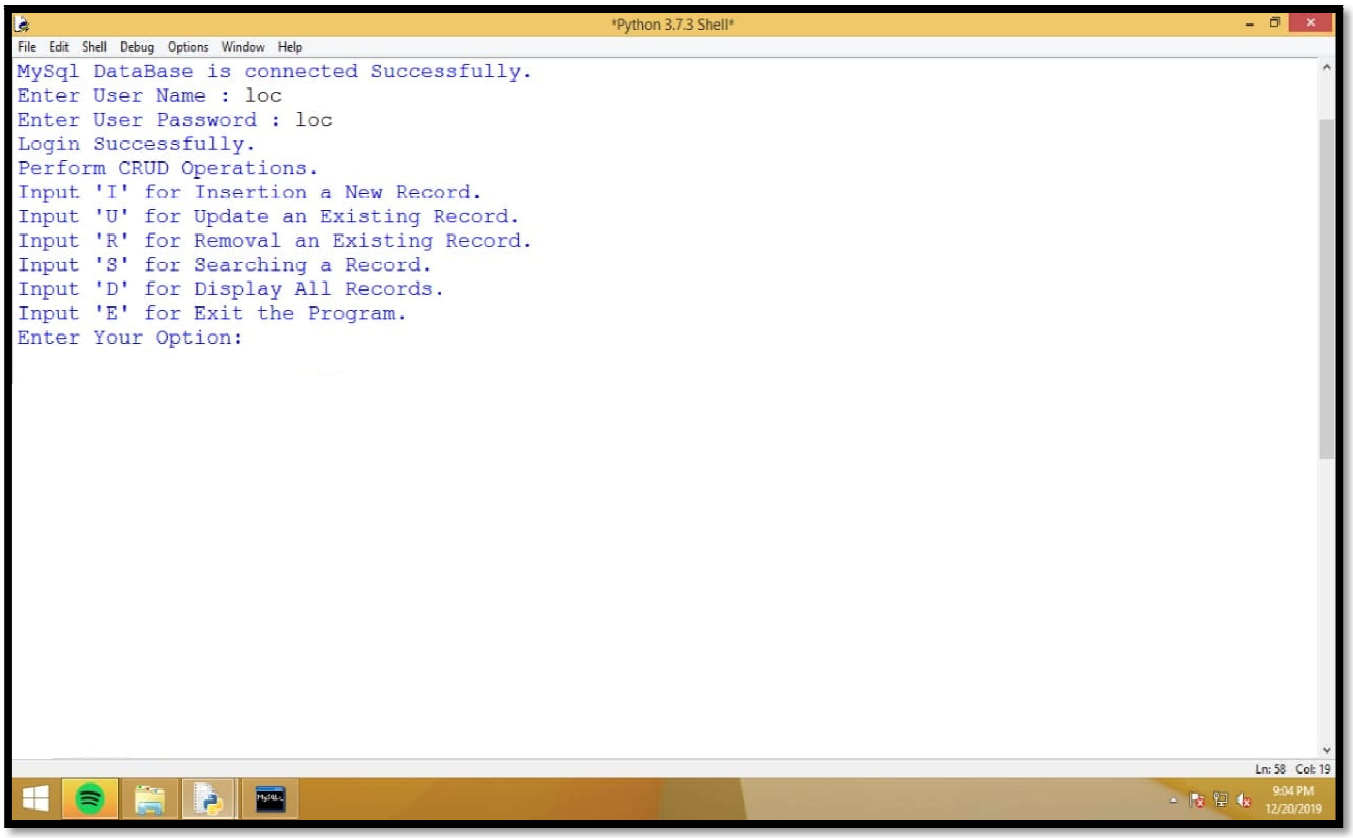
```
                print("Removal of Record.")
                reg = input("Enter Student's registration Number: ")
                qry = "delete from students where reg_num = {}".format(reg)
                mycursor.execute(qry)
                con.commit()
                print("Record is deleted Successfully.")
#-------------------------*SEARCHING*----------------------------
        elif ch == 'S' or ch == 's':
                print("Searching Operation.")
                reg = input("Enter Student's registration Number: ")
                qry = "select * from students where reg_num = {} ".format(reg)
                #print(qry)
                mycursor.execute(qry)
                print("Record is found Successfully.")
                data = mycursor.fetchone()
                count = mycursor.rowcount
                print("Total No. of Record:",count)
                for row in data:
                        print(row)
#-------------------------*DISPLAY*----------------------------
        elif ch == 'D' or ch == 'd':
                print("Display ALl Records.")
```

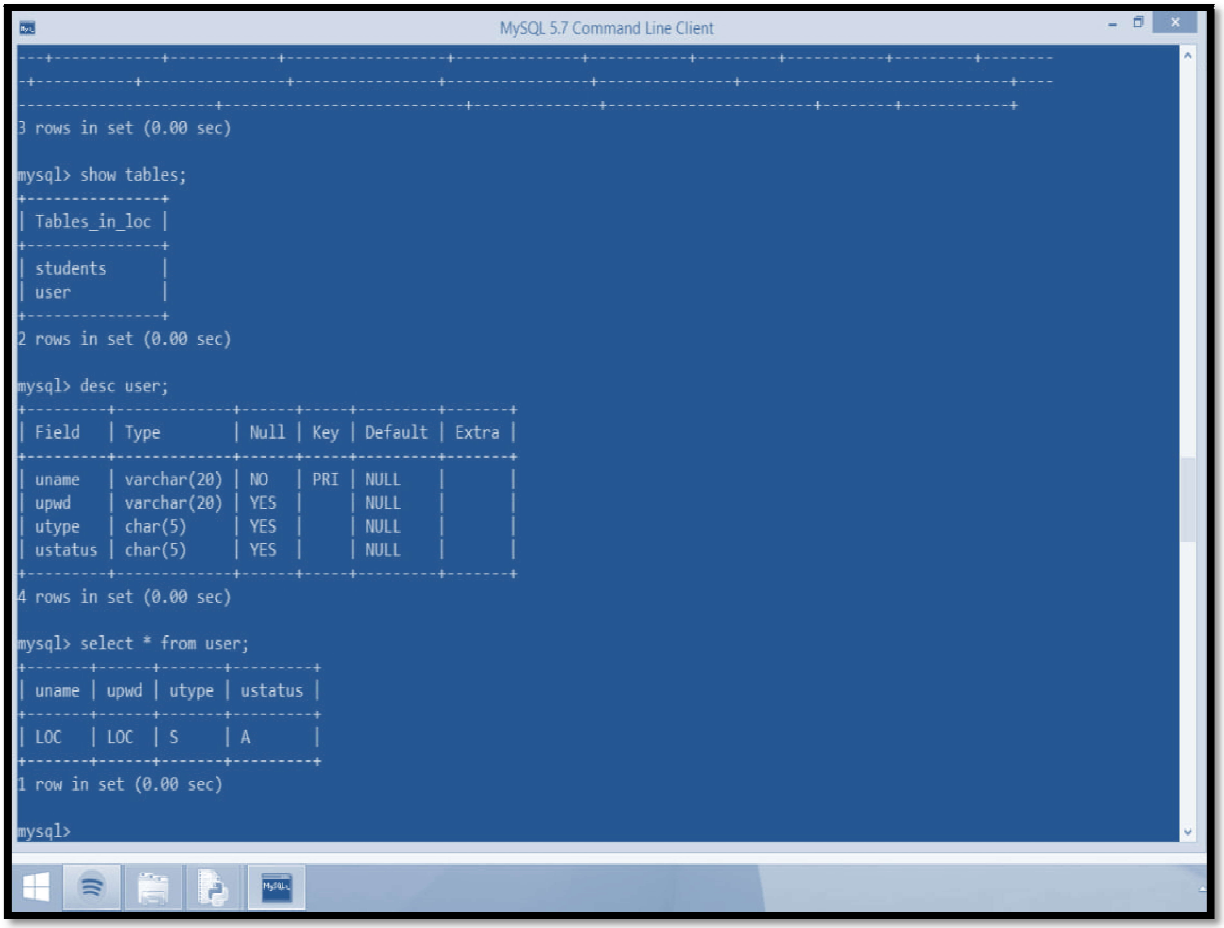
```
                qry = "select * from students"
                mycursor.execute(qry)
                data = mycursor.fetchall()
                count = mycursor.rowcount
                print("Total No. of Record: ",count)
                print("{0:<9s} {1:<9s} {2:<9s} {3:<9s} {4:<9s} {5:<9s} {6:<9s} {7:<9s} {8:<9s}
 {9:<9s}" .format ('Sl.No', 'Name', 'MName', 'FName', 'Subject1', 'Subject2',
 'Subject3','Subject4','Subject5','Subject6'))
                print("_____ ____
_____"
                for row in data:
                        print ("{0:<9s} {1:<9s} {2:<9s} {3:<9s} {4:<9s} {5:<9s} {6:<9s} {7:<9s}
{8:<9s} {9:<9s}" .format (str(row[1]), row[4], row[5], row[6], row[14], row[15], row[16],
row[17],row[18],row[19]))
            elif ch == 'E' or ch == 'e':
                print("Exiting Program.")
                sys.exit(0)
            else:
                print("Wrong Input. Try Again!!!!!")

        else:
            print("Login Failed")
            if at !=4:
                print("Try Again")
```

```
else:
    print("MySql DataBase Connection Failed.Terminating....")
```

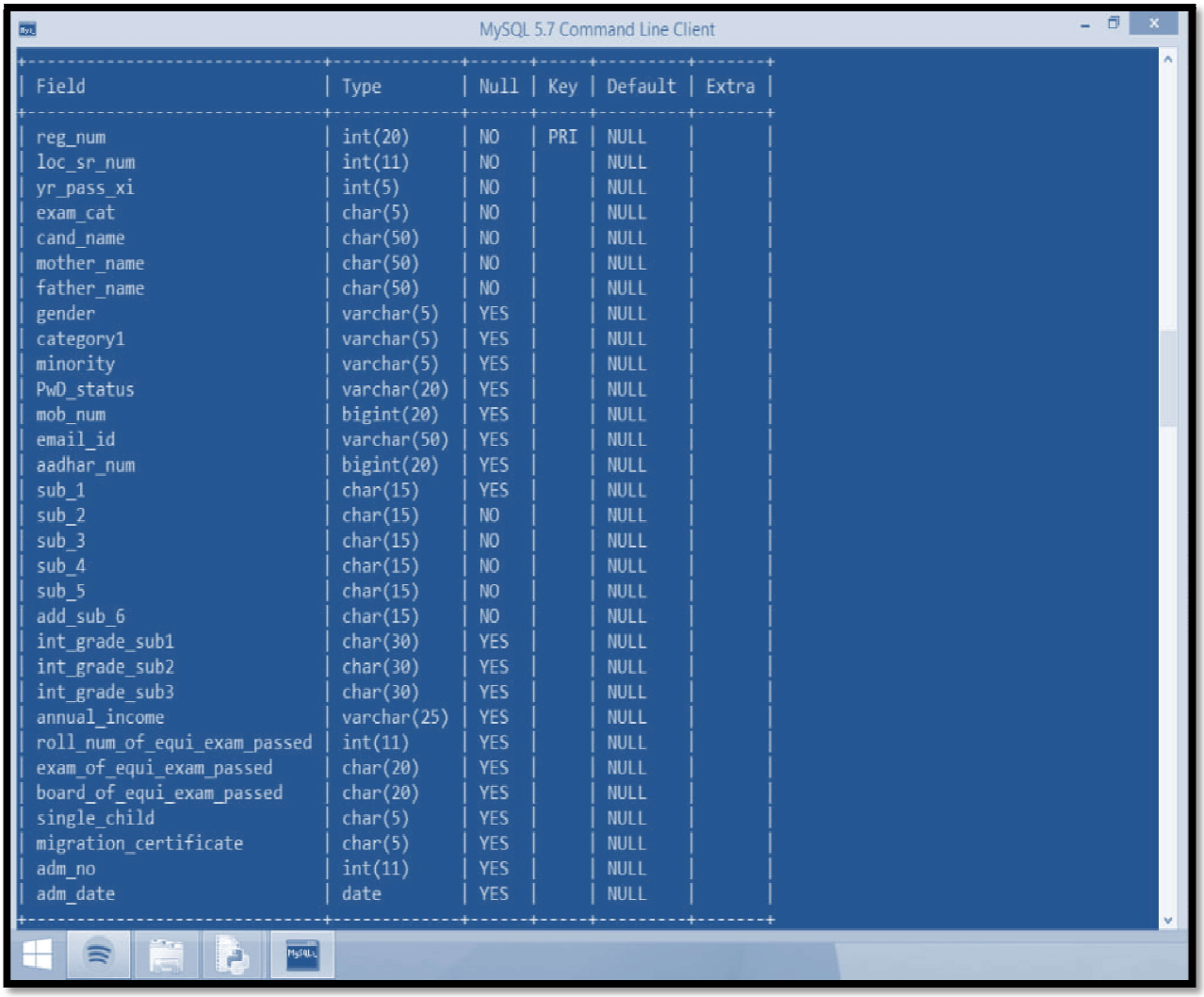
Output

1.Table created successfully.




MySQL 5.7 Command Line Client
3 rows in set (0.00 sec)
mysql> show tables;
| Tables_in_loc |
| students |
| user |
2 rows in set (0.00 sec)
mysql> desc user;
Field	Type	Null	Key	Default	Extra
uname	varchar(20)	NO	PRI	NULL	
upwd	varchar(20)	YES		NULL	
utype	char(5)	YES		NULL	
ustatus	char(5)	YES		NULL	
4 rows in set (0.00 sec)
mysql> select * from user;
uname	upwd	utype	ustatus
LOC	LOC	S	A
1 row in set (0.00 sec)
mysql>

STRUCTURE OF THE TABLE



Field	Type	Null	Key	Default	Extra
reg_num	int(20)	NO	PRI	NULL	
loc_sr_num	int(11)	NO		NULL	
yr_pass_xi	int(5)	NO		NULL	
exam_cat	char(5)	NO		NULL	
cand_name	char(50)	NO		NULL	
mother_name	char(50)	NO		NULL	
father_name	char(50)	NO		NULL	
gender	varchar(5)	YES		NULL	
category1	varchar(5)	YES		NULL	
minority	varchar(5)	YES		NULL	
PwD_status	varchar(20)	YES		NULL	
mob_num	bigint(20)	YES		NULL	
email_id	varchar(50)	YES		NULL	
aadhar_num	bigint(20)	YES		NULL	
sub_1	char(15)	YES		NULL	
sub_2	char(15)	NO		NULL	
sub_3	char(15)	NO		NULL	
sub_4	char(15)	NO		NULL	
sub_5	char(15)	NO		NULL	
add_sub_6	char(15)	NO		NULL	
int_grade_sub1	char(30)	YES		NULL	
int_grade_sub2	char(30)	YES		NULL	
int_grade_sub3	char(30)	YES		NULL	
annual_income	varchar(25)	YES		NULL	
roll_num_of_equi_exam_passed	int(11)	YES		NULL	
exam_of_equi_exam_passed	char(20)	YES		NULL	
board_of_equi_exam_passed	char(20)	YES		NULL	
single_child	char(5)	YES		NULL	
migration_certificate	char(5)	YES		NULL	
adm_no	int(11)	YES		NULL	
adm_date	date	YES		NULL	

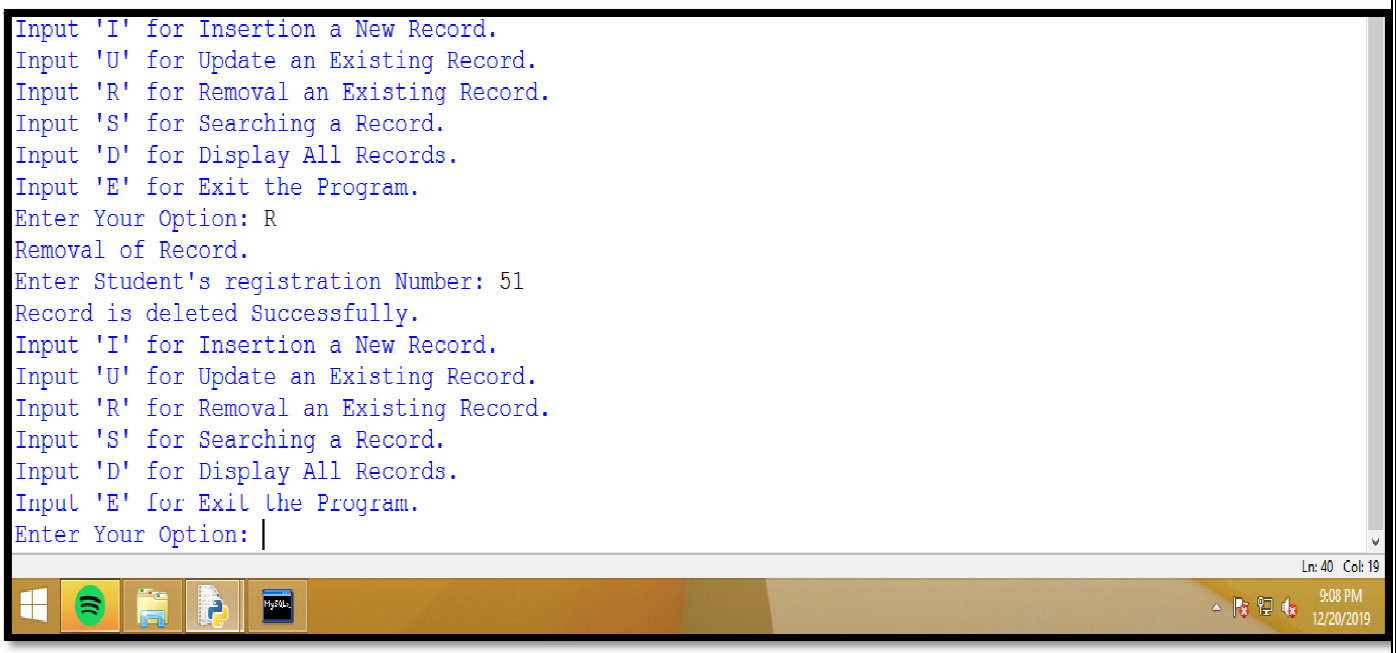
2. Insertion of record:


Enter name of internal grade subject2: GS
Enter name of internal grade subject3: PhE
Enter annual income of student's parents: 200000
Enter student's rollnum of equivalent exam passed:3131456
Enter student's exam of equivalent exam passed:AISSE
Enter student's board of equivalent exam passed:CBSE
Enter if student is single girl child or not:NO
Enter if migration certificate is required or not:YEs
Enter student's admission num:1478
Enter student's admission date as (yyyy-mm-dd):2008-12-12
Record is inserted Successfully.
Input 'I' for Insertion a New Record.
Input 'U' for Update an Existing Record.
Input 'R' for Removal an Existing Record.
Input 'S' for Searching a Record.
Input 'D' for Display All Records.
Input 'E' for Exit the Program.
Enter Your Option:

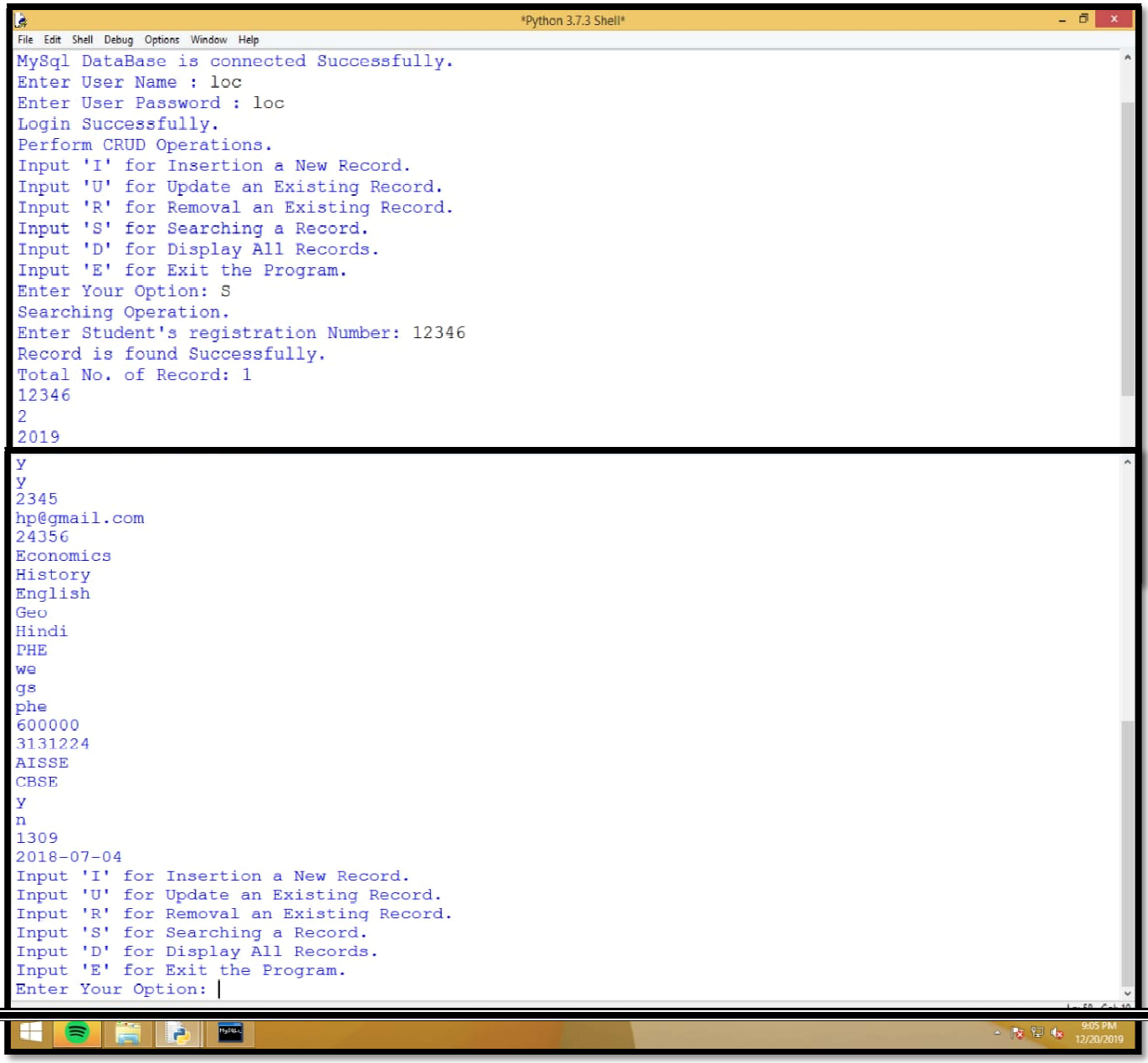
3. Updating record:

4. Deleting a record

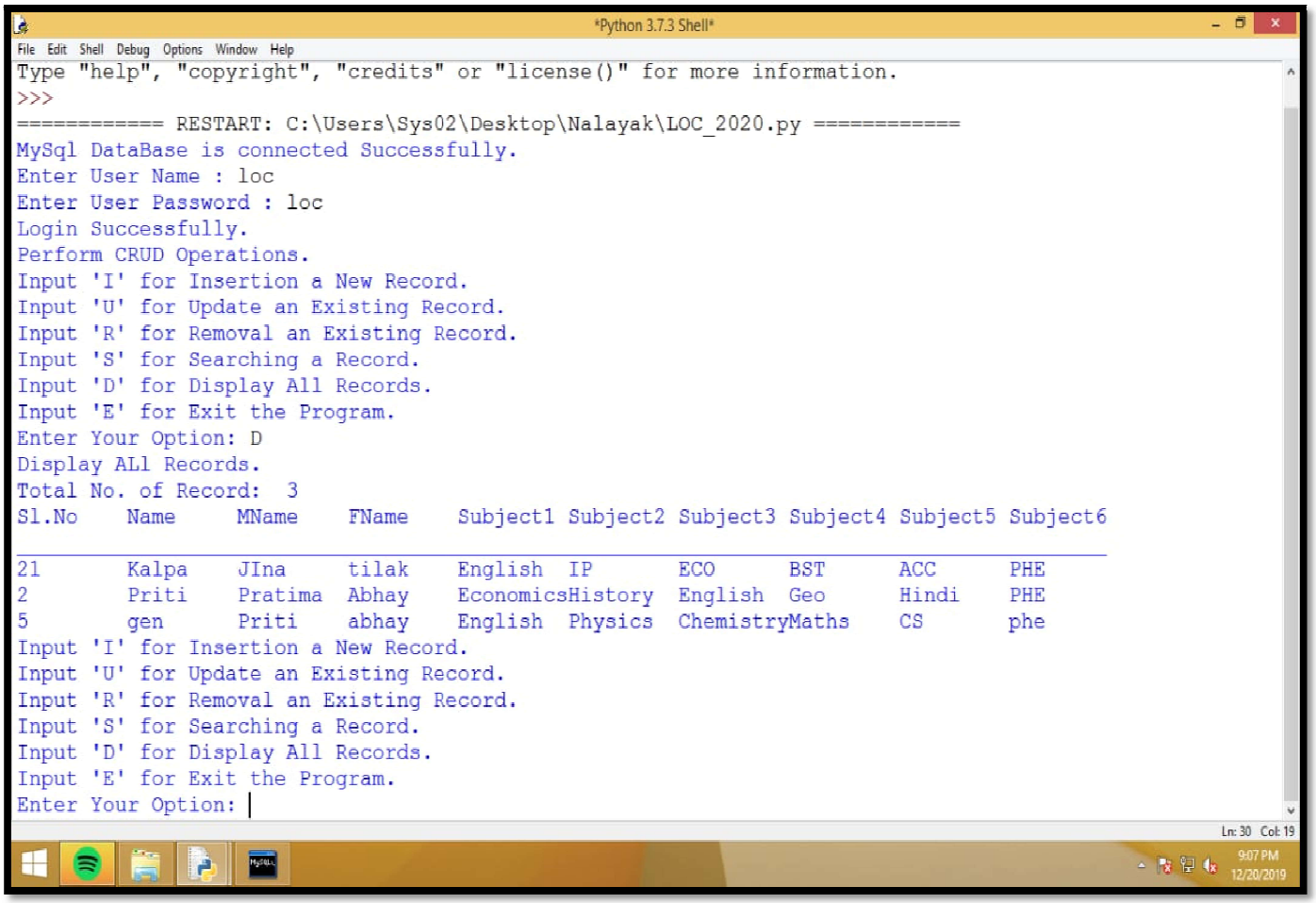
```
Input 'I' for Insertion a New Record.
Input 'U' for Update an Existing Record.
Input 'R' for Removal an Existing Record.
Input 'S' for Searching a Record.
Input 'D' for Display All Records.
Input 'E' for Exit the Program.
Enter Your Option: R
Removal of Record.
Enter Student's registration Number: 51
Record is deleted Successfully.
Input 'I' for Insertion a New Record.
Input 'U' for Update an Existing Record.
Input 'R' for Removal an Existing Record.
Input 'S' for Searching a Record.
Input 'D' for Display All Records.
Input 'E' for Exit the Program.
Enter Your Option:
```



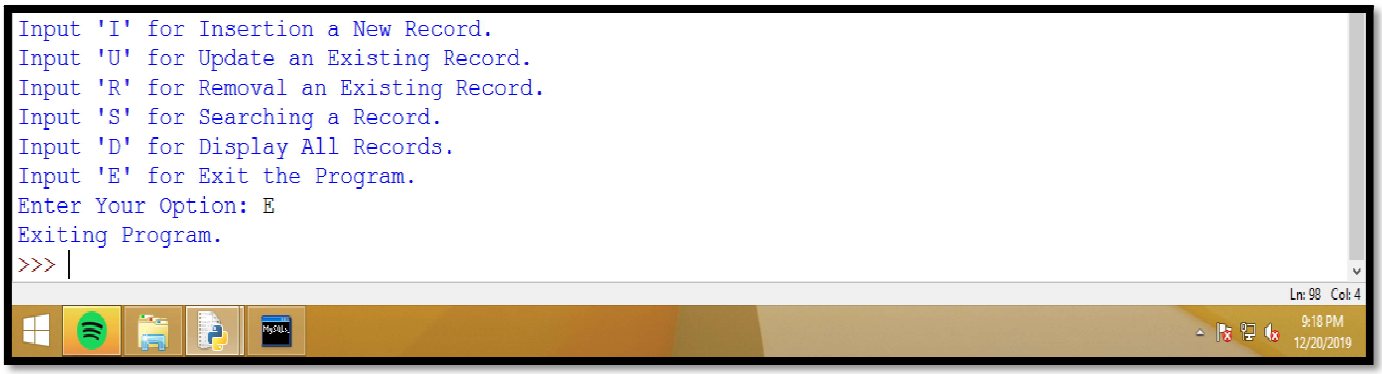
5. Searching a record:



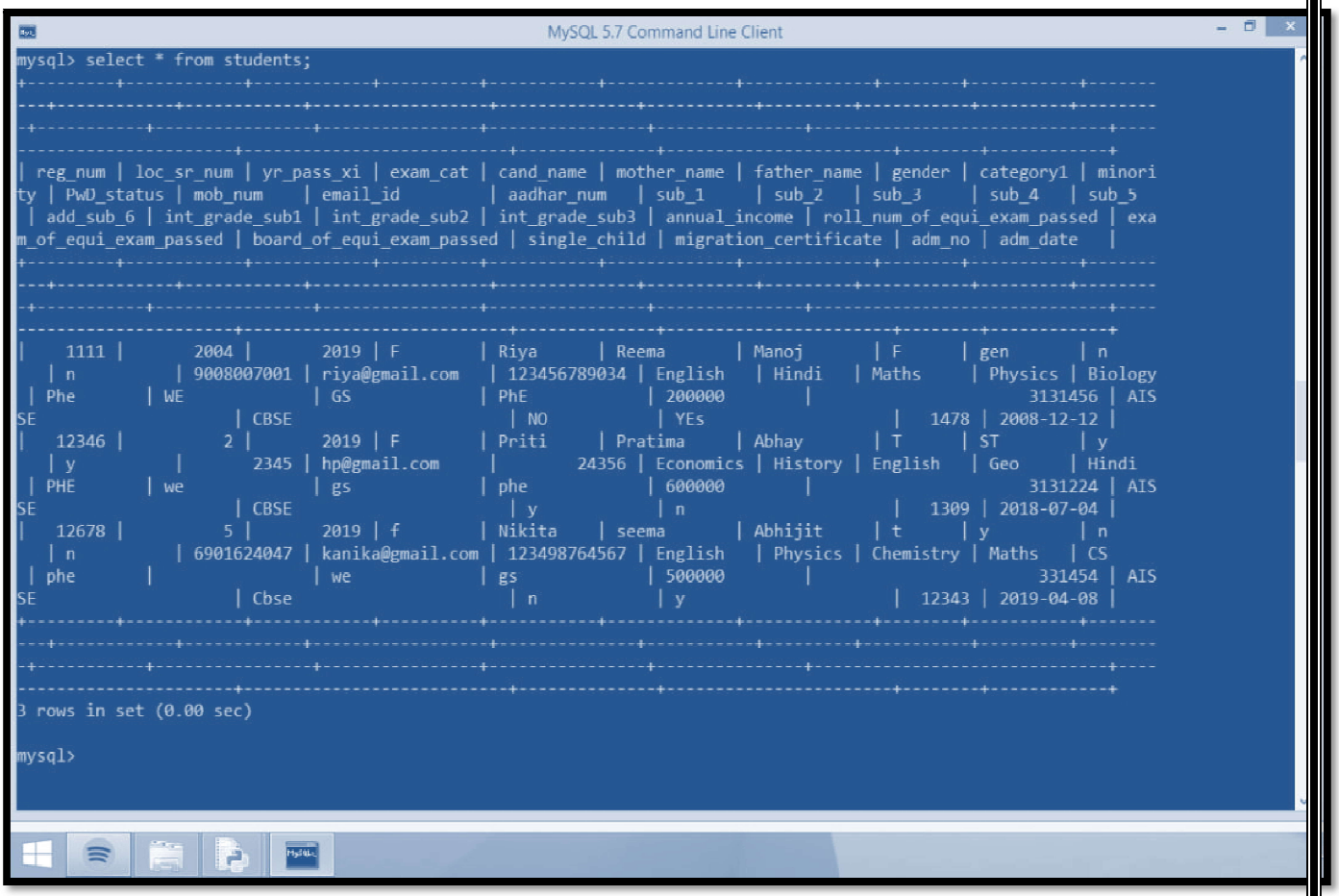
6. Display all Records:



7. Exit



RECORDS:


MySQL 5.7 Command Line Client
mysql> select * from students;
reg_num	loc_sr_num	yr_pass_xi	exam_cat	cand_name	mother_name	father_name	gender	category1	minority	PWD_status	mob_num	email_id	aadhar_num	sub_1	sub_2	sub_3	sub_4	sub_5	add_sub_6	int_grade_sub1	int_grade_sub2	int_grade_sub3	annual_income	roll_num_of_equi_exam_passed	exam_of_equi_exam_passed	board_of_equi_exam_passed	single_child	migration_certificate	adm_no	adm_date
1111	2004	2019	F	Riya	Reema	Manoj	F	gen	n	n	9008007001	riya@gmail.com	123456789034	English	Hindi	Maths	Physics	Biology	Phe	WE	GS	PhE	200000	3131456	AISSE	CBSE	NO	YEs	1478	2008-12-12
12346	2	2019	F	Priti	Pratima	Abhay	T	ST	y	y	2345	hp@gmail.com	24356	Economics	History	English	Geo	Hindi	PHE	we	gs	phe	600000	3131224	AISSE	CBSE	y	n	1309	2018-07-04
12678	5	2019	f	Nikita	seema	Abhijit	t	y	n	n	6901624047	kanika@gmail.com	123498764567	English	Physics	Chemistry	Maths	CS	phe	we	gs	500000	331454	AISSE	Cbse	n	y	12343	2019-04-08
3 rows in set (0.00 sec)
mysql>

Appendix

Module: mysql.connector: Package for database programming

Functions	Working
connect()	establishes connection between MySQL and Python
cursor()	facilitates the row by row processing of records in the resultset
is_connected()	check whether connection is established or not
execute()	for the execution of sql query
commit()	to save the changes that you have you made
fetchone()	only fetch one record
fetchall()	fetches all records
fetchmany()	fetches as many records as you want
rowcount()	returns the number of rows retrieved from the cursor

format()	to insert the records

Bibliography

- Computer Science with python
  - by Sumita Arora
- www.python.org/download
- www.py2exe.org

www.mysql.org