西北工業大學

NORTHWESTERN POLYTECHNICAL UNIVERSITY

Lab report

Name : ABID ALI

Student_no : 2019380141

Experiment 3

Experiment No:3

Data Integrity and Security Goal

Goal:

1. To practice how to define the data integrity.
2. To practice how to create users
3. To practice how to grant/revoke privileges of databases and tables.

Content

1．Add the following constraint and index with GUI(for university database)（10 points）.

(1) Not null：add the not null constraint to **S**(SBITH).

(2) Primary key：set the SNO as the primary key.

(3) Unique constraint：add unique constraint for the primary key of SNAME, the constraint name is uk_Sname.

(4) Default constraint：add the default value to **S**(SGENDER), the default value is “male”.

2. Add forgein key constraints to table **SC** in the database **Student** with GUI, set SNO(foreign ke name is: sc_fk_sid) as a foreign key referencing table S, and set CNO as another foreign key referencing table C **,** name it with sc_fk_ cno. Try and validate different strategies in violation of the foreign key constraints : NO Action/restrict/cascade/set to null（10 points）

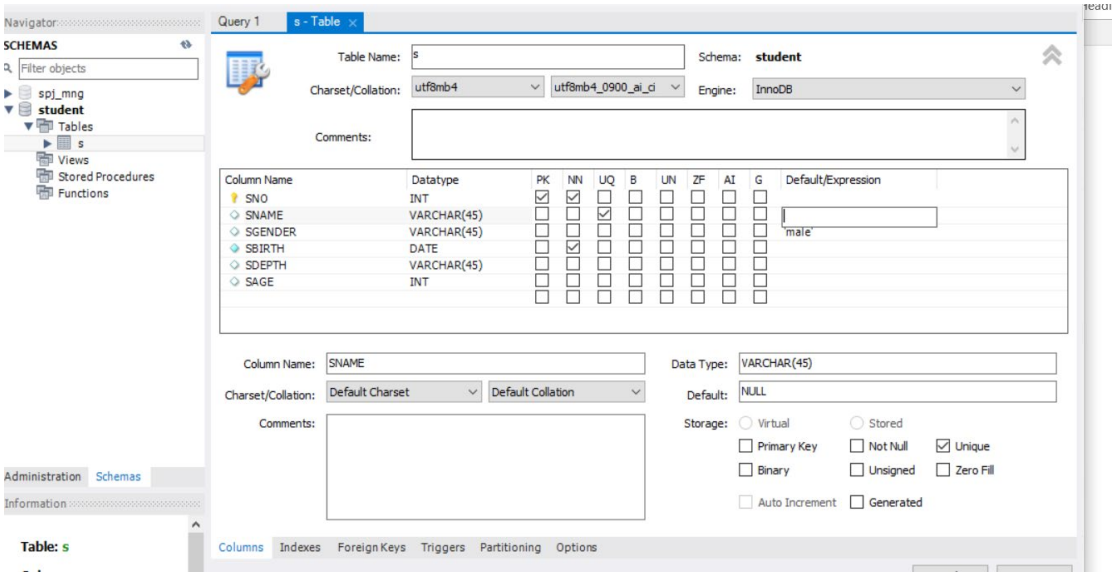
3. Drop the three tables in database **Student**, and create some tables through SQL statements with the following constraints.(10 points）
   - ✧ Table ***S***：same to the constraints set in question 1(1)。
   - ✧ Table ***C***：set CPNO as a foreign key，referencing table c itself with the attribute CNO.
   - ✧ Table **SC**：set the foreign key constraints same to question2, and set the valid range of attribute GRADE with[0,100]. In addition, add one attribute ID to table SC, and set it as a primary key, and it can increase automaticlly. When a new tuple is inserted to the table, its(ID) value will increase by +1.

4. Add or remove the following integrity constraints with SQL language. (10 points)

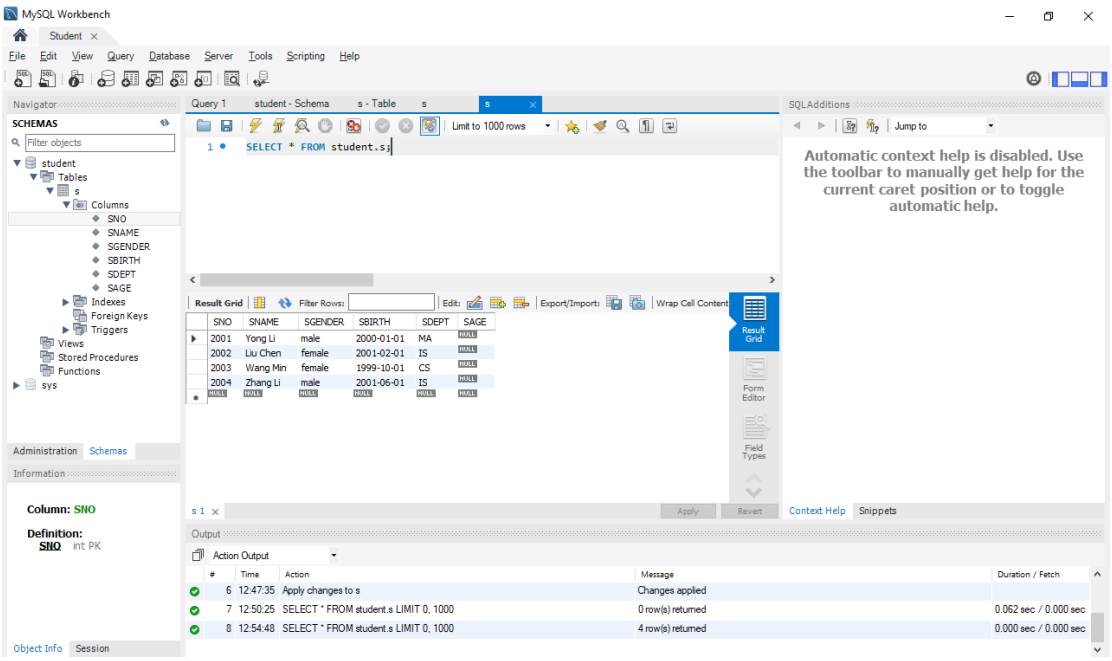
Solutions:

Answer No:1

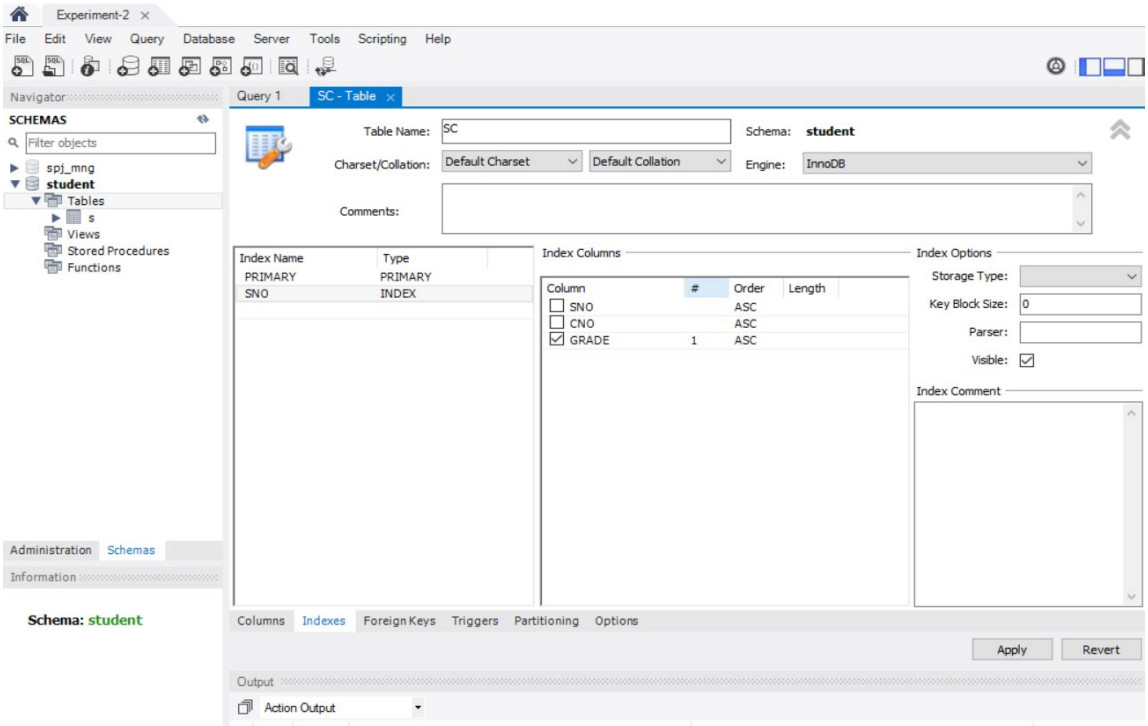
We are using the options provided by the workbench in mysql(GUI)

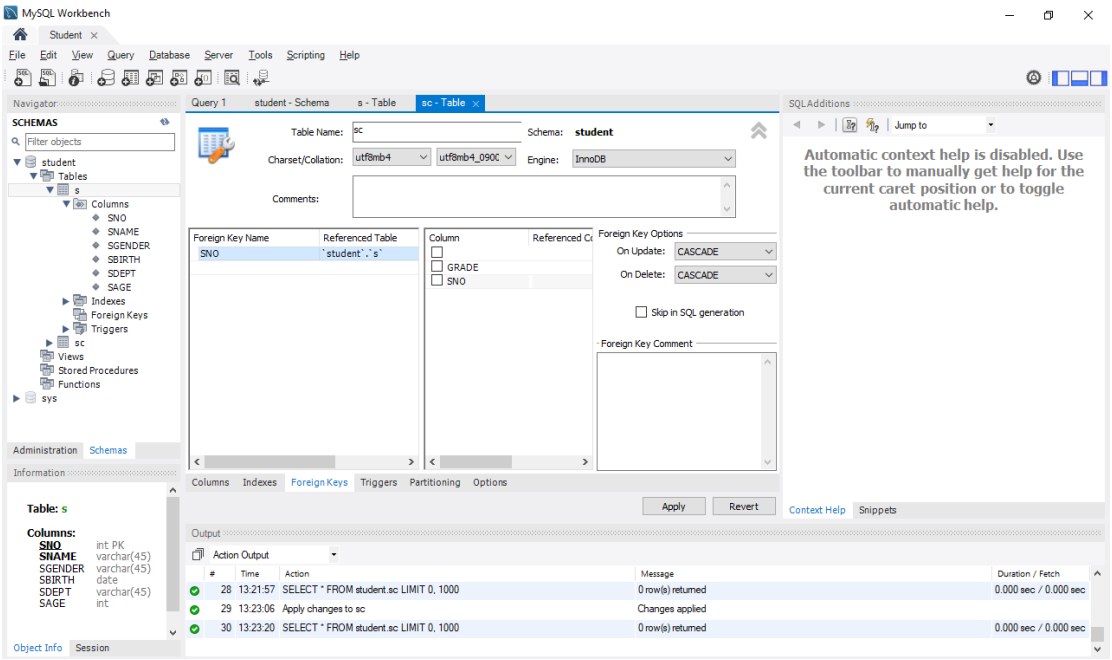


Creating the tables in GUI first with name of ‘S’. In this the SNO is primary key, ‘SBIRTH’ is not null constraint, ‘SNAME’ with unique key constraint and ‘SGENDER’ have default value of ‘male’



Adding values using GUI

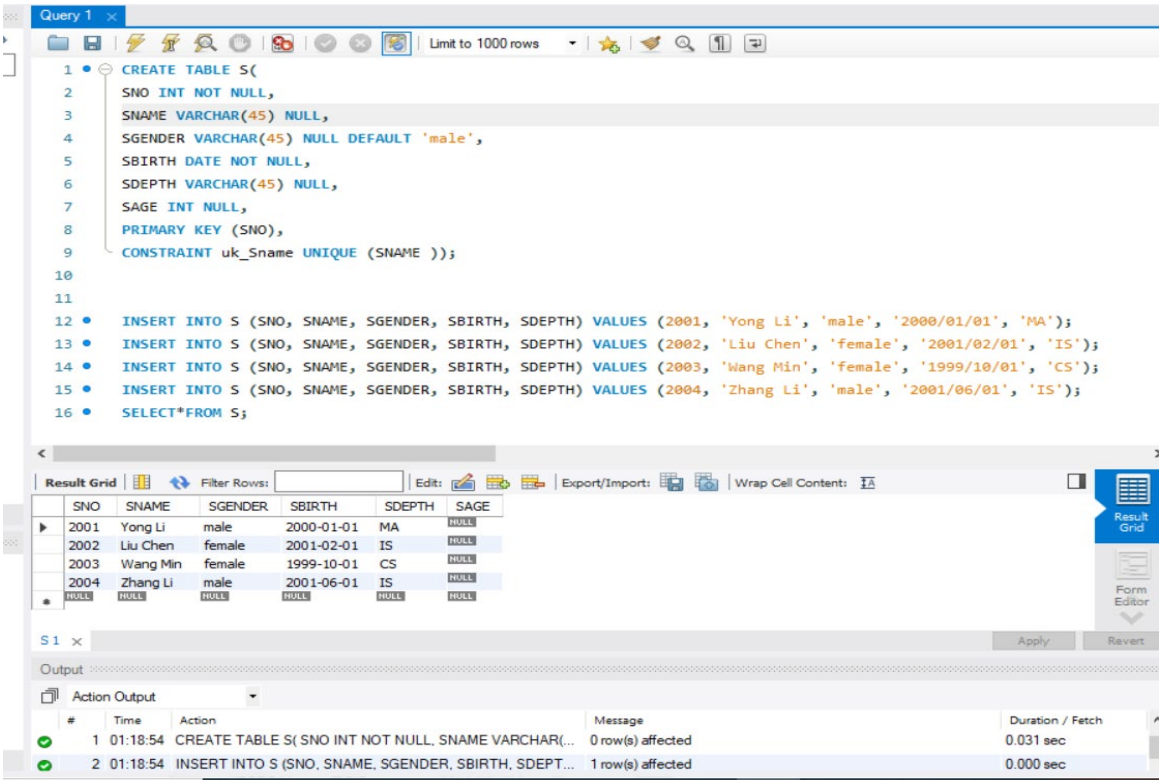




Then, adding the foreign key constraints to table ‘SC’ by using GUI and set ‘SNO’ as a foreign key referencing table ‘S’ and set ‘CNO’ as another foreign key referencing table ‘C’

Alternate way:

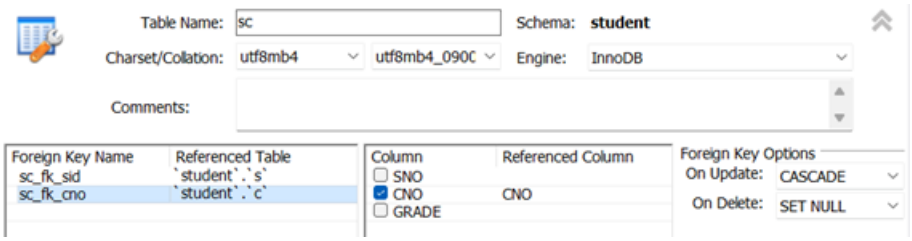
We are using the statements in mysql to form the table with constrains(Mysql statements):



```
CREATE TABLE S(
SNO INT NOT NULL,
SNAME VARCHAR(45) NULL,
SGENDER VARCHAR(45) NULL DEFAULT 'male',
SBIRTH DATE NOT NULL,
SDEPTH VARCHAR(45) NULL,
PRIMARY KEY (SNO),
CONSTRAINT uk_Sname UNIQUE (SNAME));
```

Answer No:2

Using GUI



Alternate way(Using mysql language)

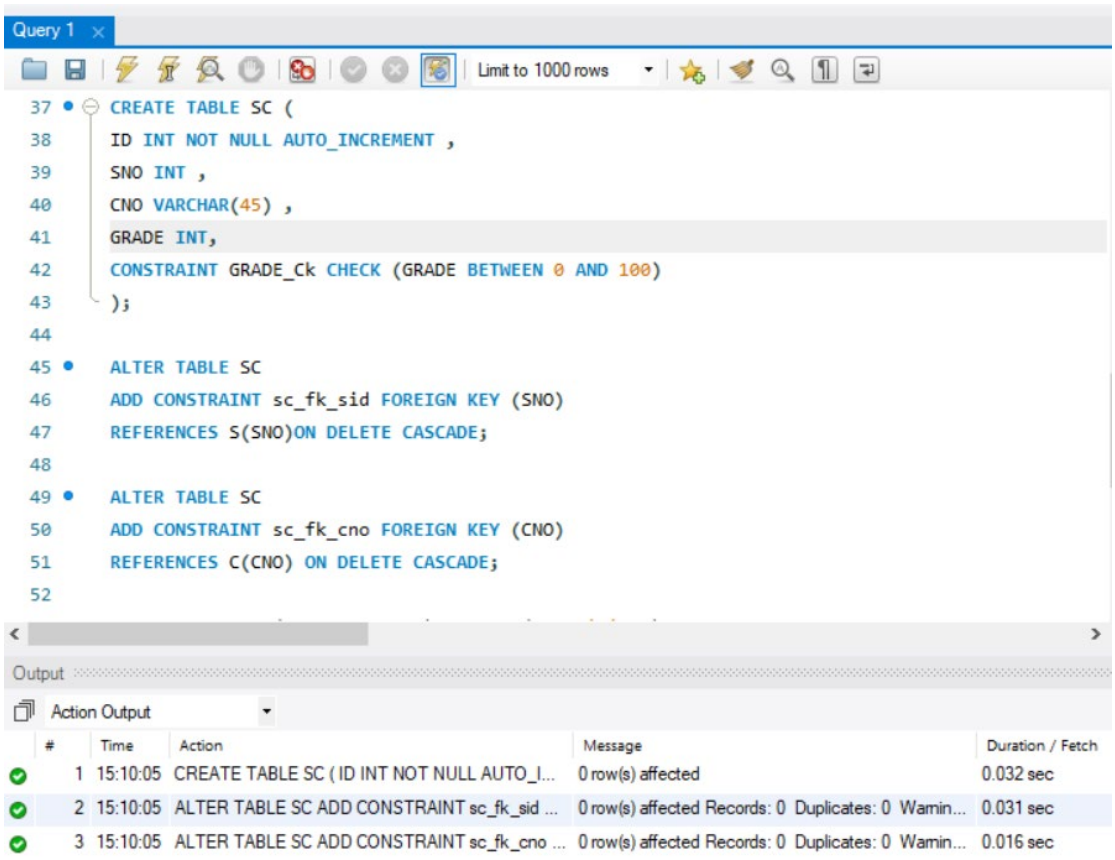
```
ALTER TABLE SC

ADD CONSTRAINT sc_fk_sid FOREIGN KEY (SNO)

REFERENCES S(SNO)ON DELETE CASCADE;

ALTER TABLE SC

ADD CONSTRAINT sc_fk_cno FOREIGN KEY (CNO)

REFERENCES C(CNO) ON DELETE CASCADE;
```

Table: sc

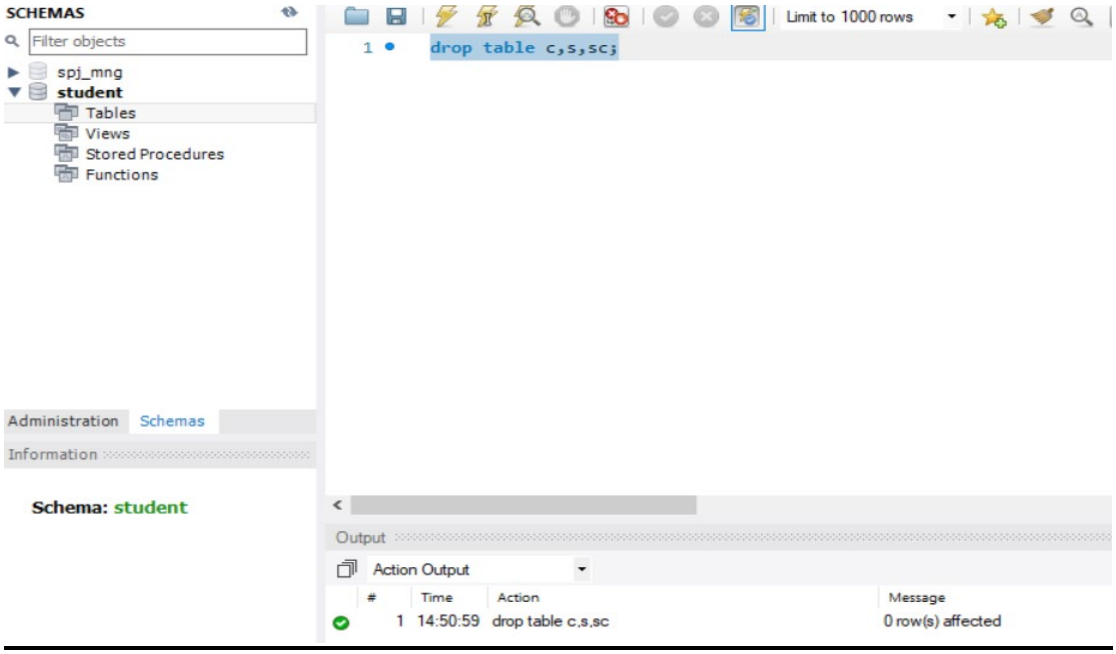
Columns:

ID	int AI PK
SNO	int
CNO	varchar(45)
GRADE	int

If you don't want to turn key checking on and off, you can permanently modify it to `ON DELETE SET NULL`

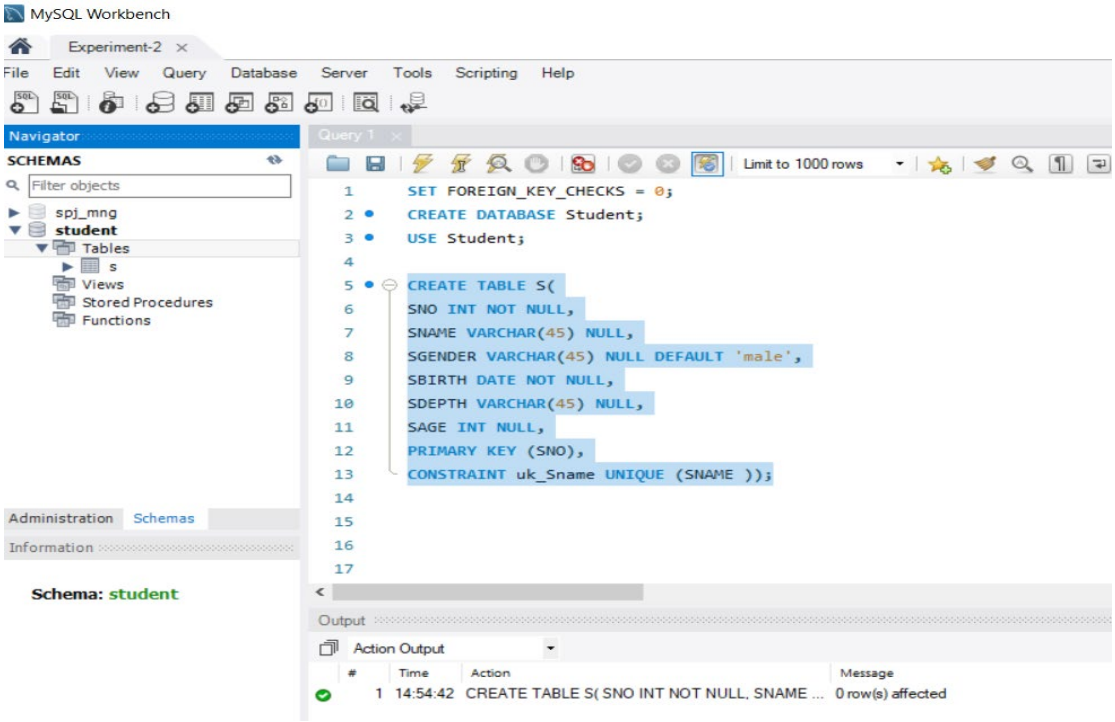


Answer No:3



We drop the 3 tables c, s and sc

TABLE S



CREATE TABLE S(

```
SNO INT NOT NULL,
SNAME VARCHAR(45) NULL,
SGENDER VARCHAR(45) NULL DEFAULT 'male',
SBIRTH DATE NOT NULL,
SDEPTH VARCHAR(45) NULL,
SAGE INT NULL,
PRIMARY KEY (SNO),
CONSTRAINT uk_Sname UNIQUE (SNAME ));
```

Table: s

Columns:

SNO	int PK
SNAME	varchar(45)
SGENDER	varchar(45)
SBIRTH	date
SDEPTH	varchar(45)
SAGE	int

TABLE C

Table: c

Columns:

CNO	varchar(45) PK
CNAME	varchar(45)
CPNO	varchar(45)
CREDIT	int

```
CREATE TABLE C (
CNO VARCHAR(45),
CNAME VARCHAR(45),
CPNO VARCHAR(45),
CREDIT INT,
PRIMARY KEY (CNO),
CONSTRAINT FOREIGN KEY(CPNO)
REFERENCES C (CNO) ON DELETE CASCADE);
```

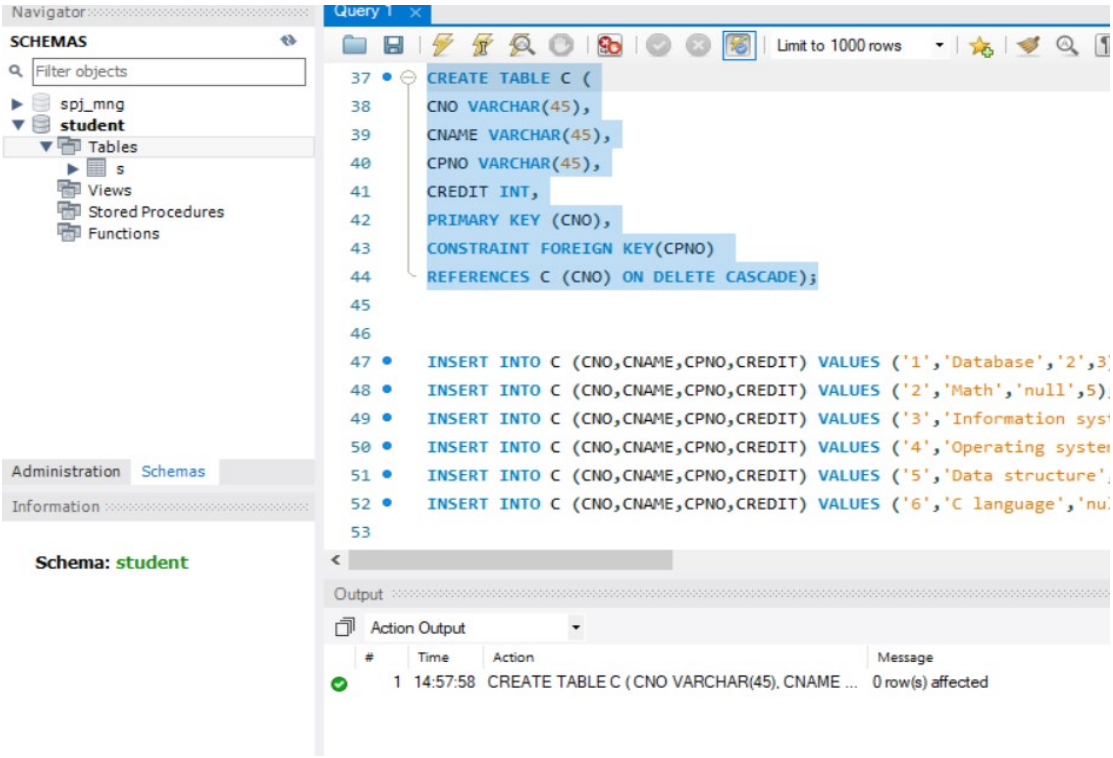


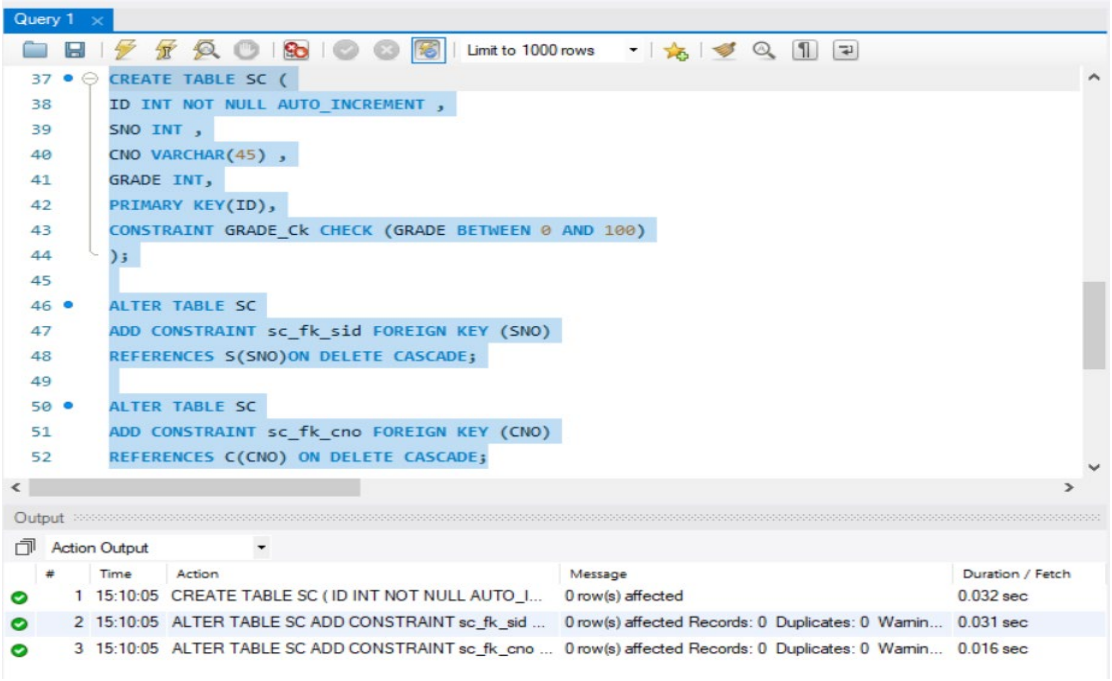
TABLE SC

```
CREATE TABLE SC (
ID INT NOT NULL AUTO_INCREMENT ,
SNO INT ,
CNO VARCHAR(45) ,
GRADE INT,
PRIMARY KEY(ID),
CONSTRAINT GRADE_Ck CHECK (GRADE BETWEEN 0 AND 100)
);

ALTER TABLE SC
ADD CONSTRAINT sc_fk_sid FOREIGN KEY (SNO)
REFERENCES S(SNO)ON DELETE CASCADE;

ALTER TABLE SC
ADD CONSTRAINT sc_fk_cno FOREIGN KEY (CNO)
REFERENCES C(CNO) ON DELETE CASCADE;
```

Table: sc

Columns:

ID	int AI PK
SNO	int
CNO	varchar(45)
GRADE	int

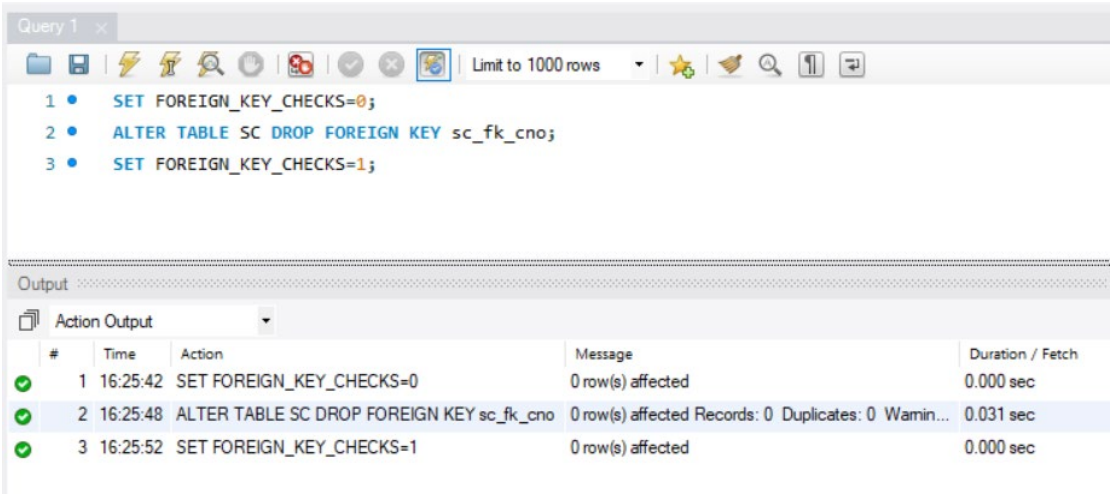


Answer No:4

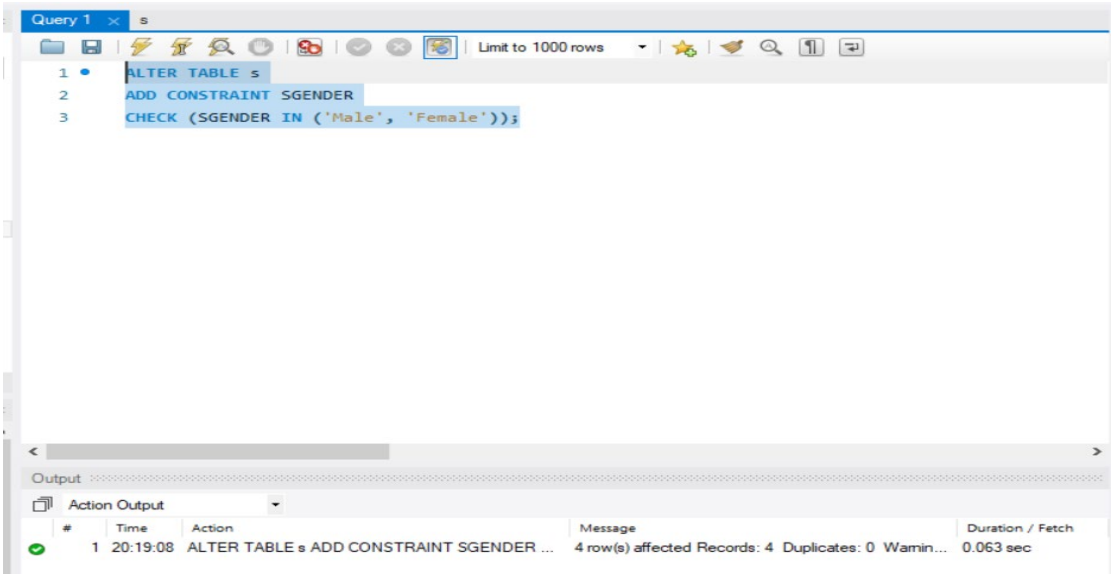
Note that `DISABLE KEYS` does not work on InnoDB tables as it works properly for MyISAM.

This is how we remove the foreign key constrain:

```
SET FOREIGN_KEY_CHECKS=0;
ALTER TABLE SC DROP FOREIGN KEY sc_fk_cno;
SET FOREIGN_KEY_CHECKS=1;
```

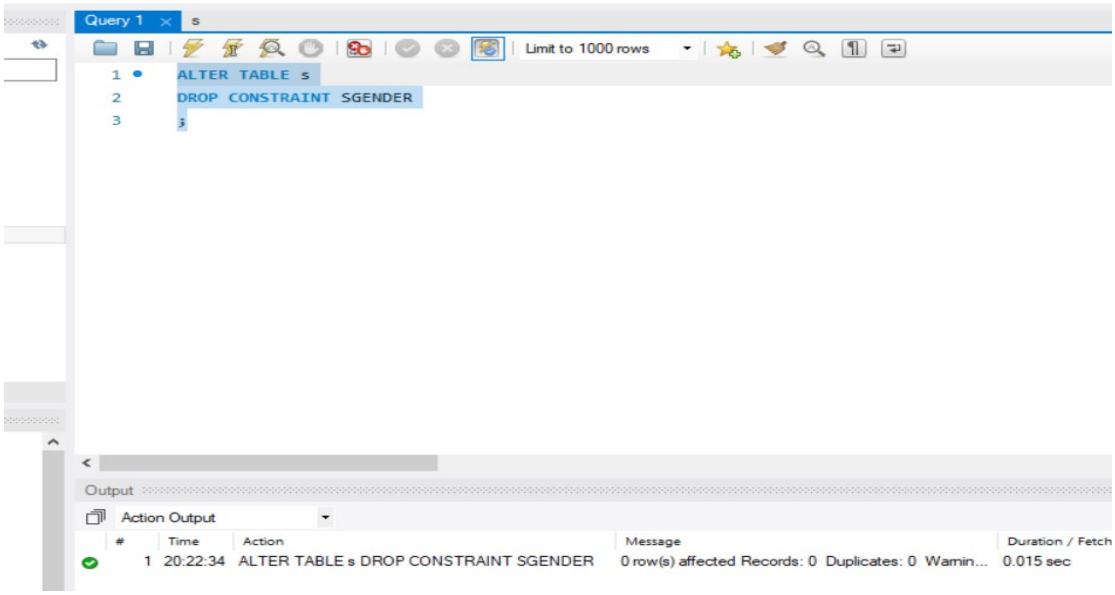


Answer No:4(1)



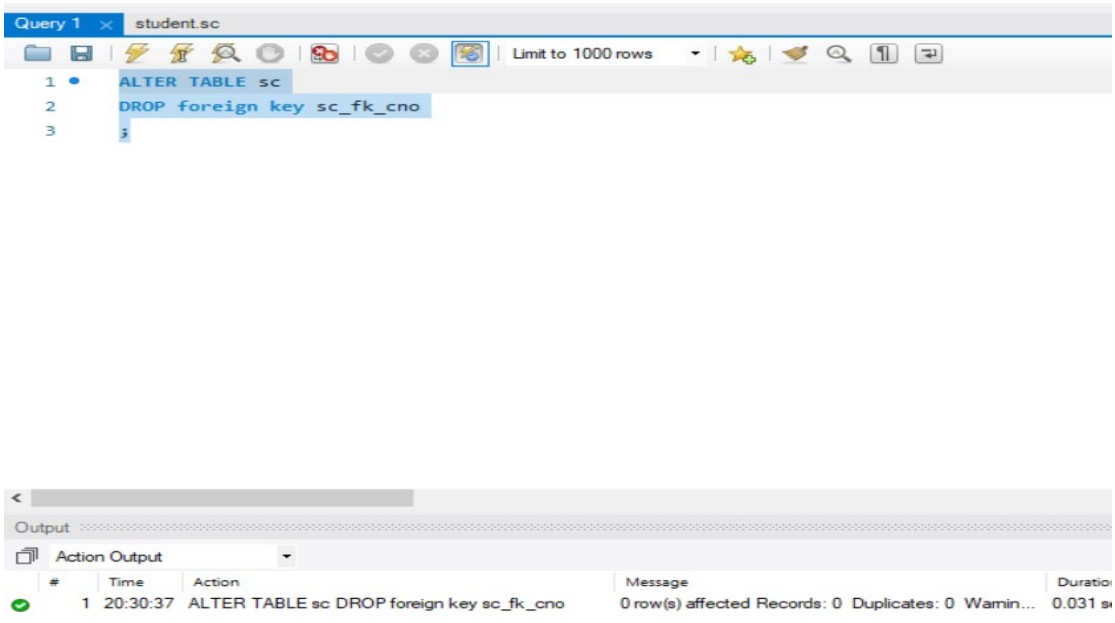
ALTER TABLE s
ADD CONSTRAINT SGENDER
CHECK (SGENDER IN ('Male', 'Female'));

Answer No:4(2)



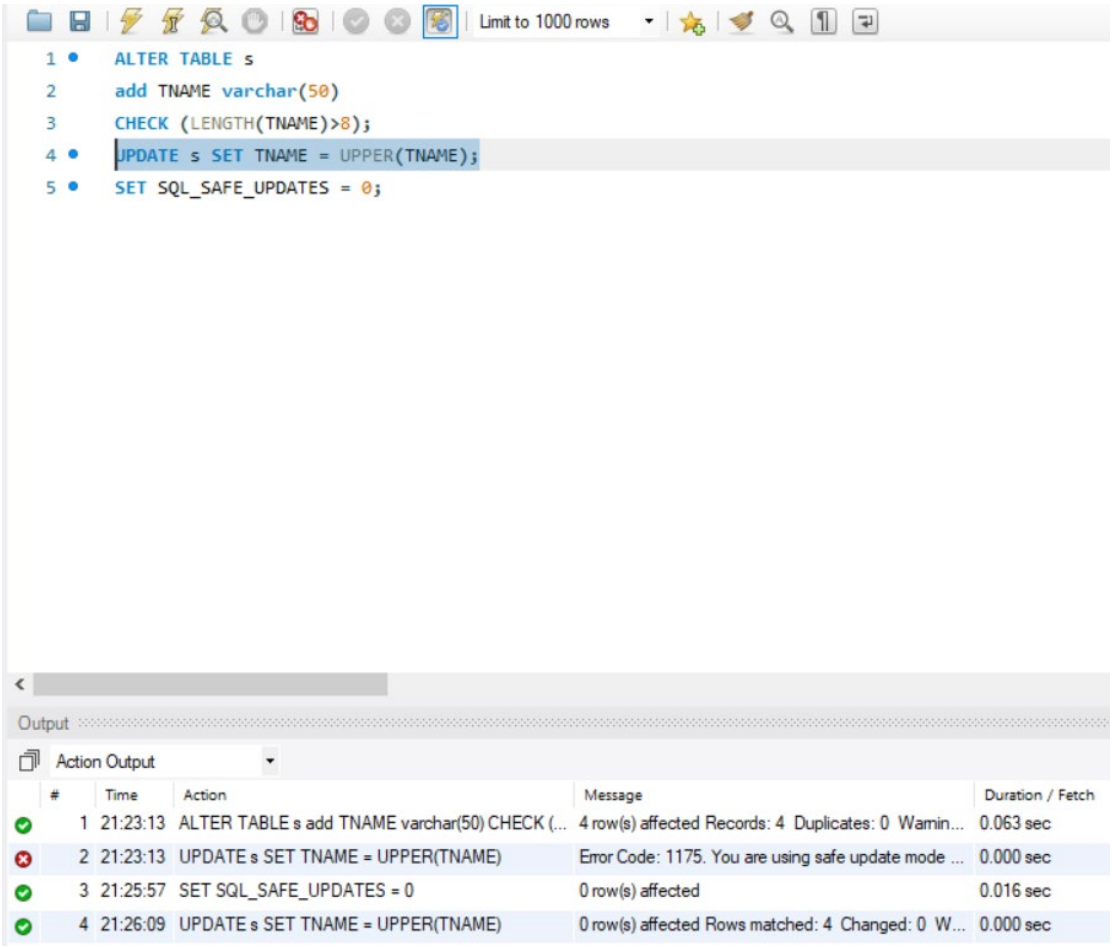
ALTER TABLE s
DROP CONSTRAINT SGENDER
;

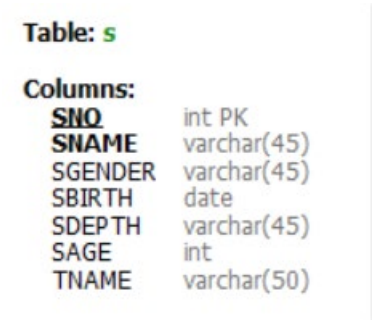
Answer No:4(3)



ALTER TABLE sc
DROP foreign key sc_fk_cno
;

Answer No:4(4)

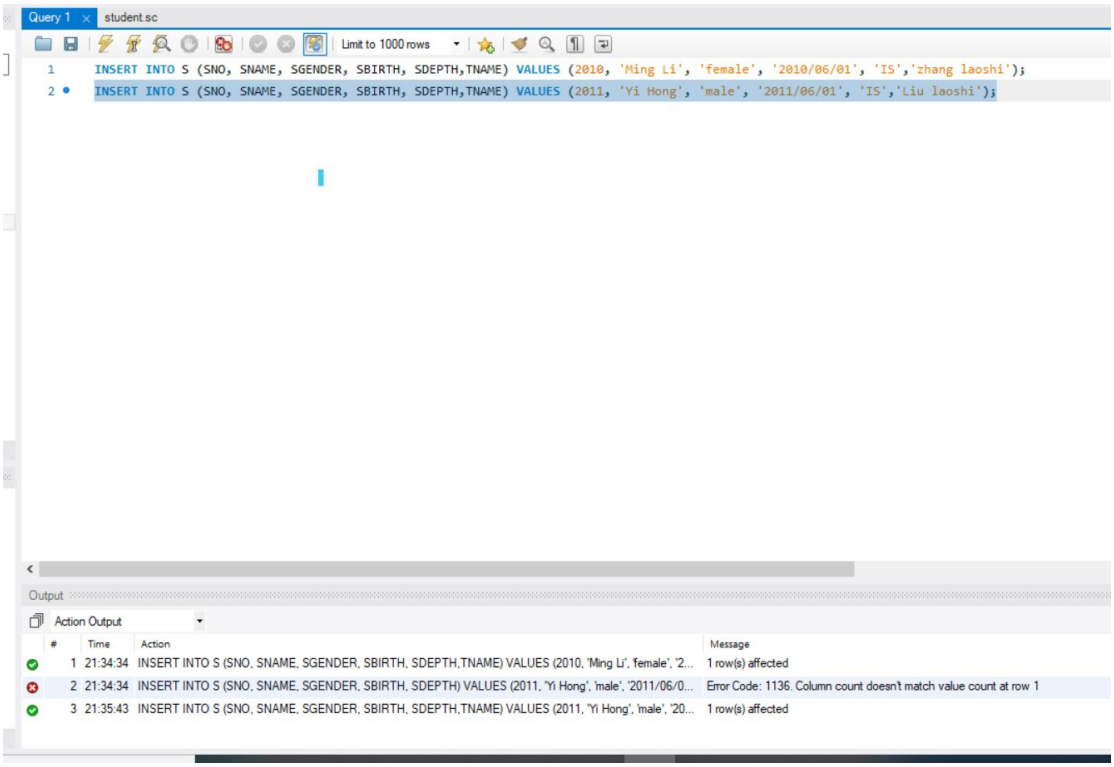


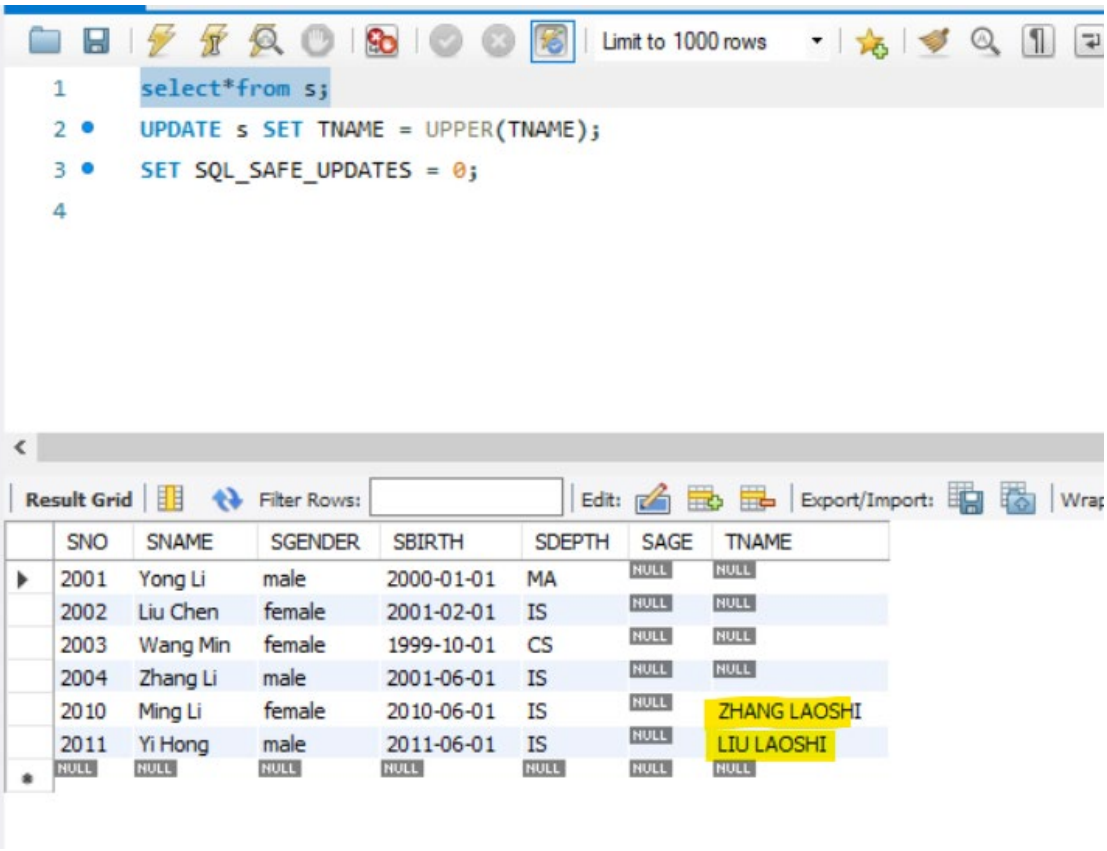



```
ALTER TABLE s
add TNAME varchar(50)
CHECK (LENGTH(TNAME)>8);
UPDATE s SET TNAME = UPPER(TNAME);
SET SQL_SAFE_UPDATES = 0;
```

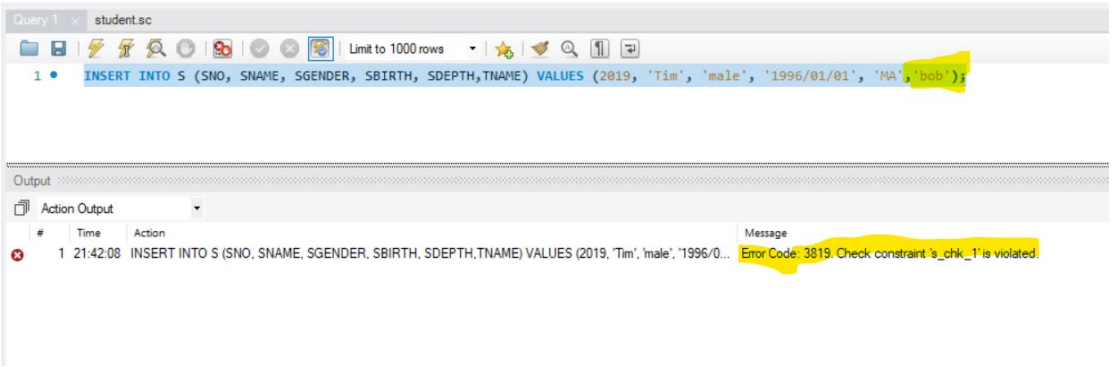
We have created the new column “TNAME”.

We have named the teacher’s name more than 8 char,so we didn’t get error.Because as per the condition of question ,teacher’s name needs to be more than 8 letters

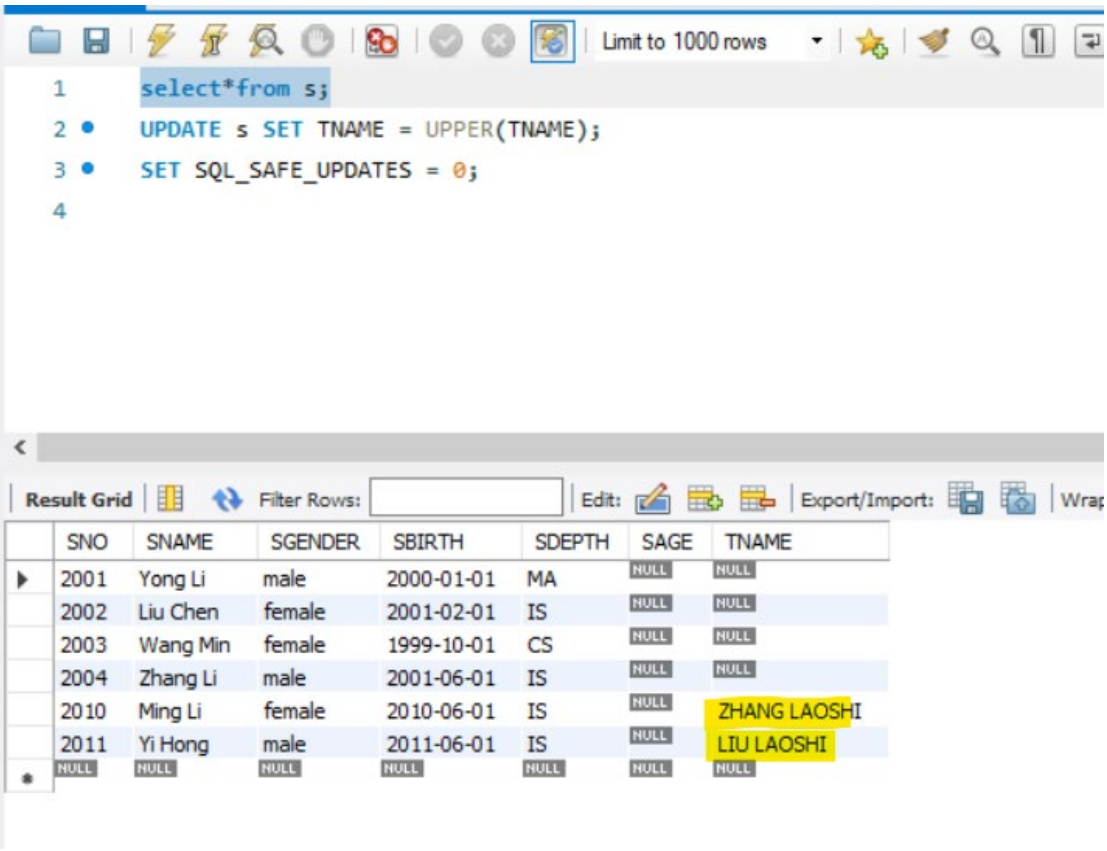




We can see that ,when we give a name smaller than 8 characters then it shows error.Then,it will make the make the teacher to give a name bigger than 8 characters.



Therefore,we can see that we fulfilled the condition

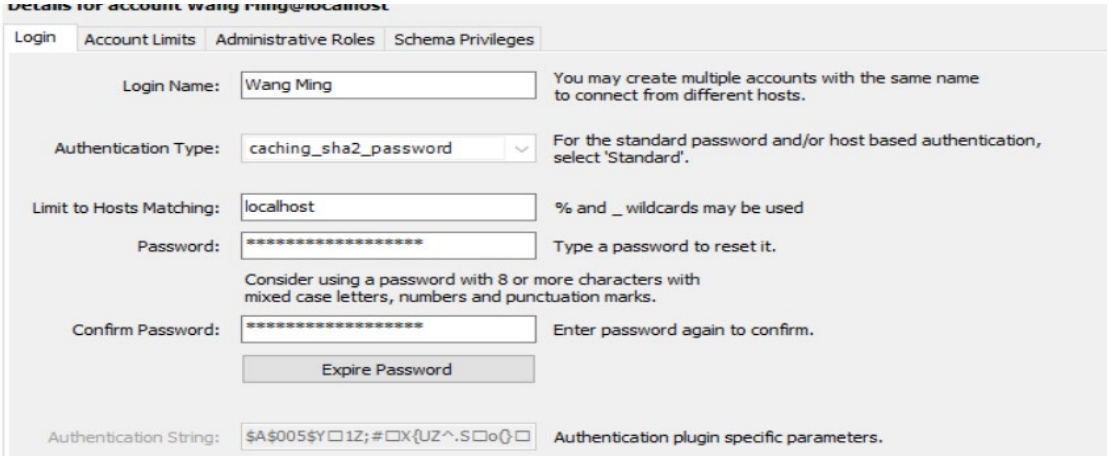


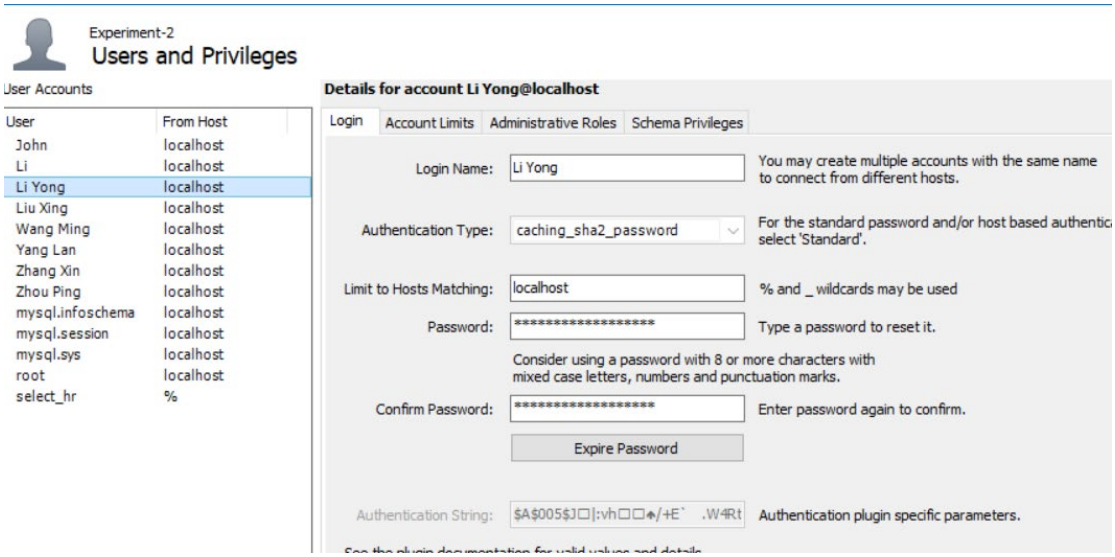
We can see that,the name of teacher will be in uppercase all the time.
We have fulfilled the two condition the name of teacher should more than 8 characters and capital letters.

5.Create and authorize new users in GUI

Answer No:5(1)

Create two users who can access the current student database: Wang Ming and Li Yong.
Using GUI





Alternate way using Sql Language

Code:

create user 'Wang Ming' @'localhost 'identified by' 123456 ';
create user 'Li Yong' @'localhost 'identified by' 123456 ';

This is how we create user

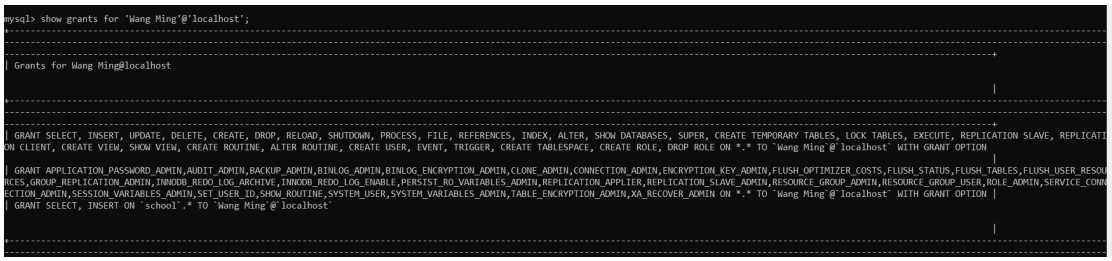
```
mysql> create user 'Li Yong' @'localhost 'identified by' 123456 ';
Query OK, 0 rows affected (0.02 sec)
```

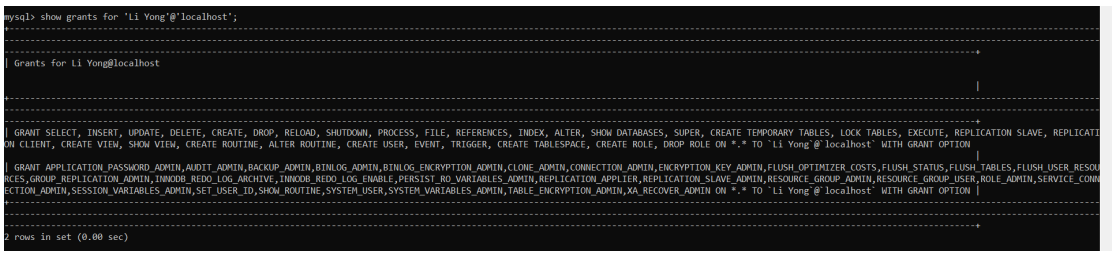
```
mysql> select user,host from mysql.user;
+------------------+-----------+
| user             | host      |
+------------------+-----------+
| John             | localhost |
| Li Yong          | localhost |
| Wang Ming        | localhost |
| mysql.infoschema | localhost |
| mysql.session    | localhost |
| mysql.sys        | localhost |
| root             | localhost |
+------------------+-----------+
7 rows in set (0.00 sec)
```

We can see that two users are created

```
mysql> GRANT ALL ON *.* TO 'Wang Ming'@'localhost' WITH GRANT OPTION;
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT ALL ON *.* TO 'Li Yong'@'localhost' WITH GRANT OPTION;
Query OK, 0 rows affected (0.00 sec)
```

Then,we grant some privileges so that they can access the student database.

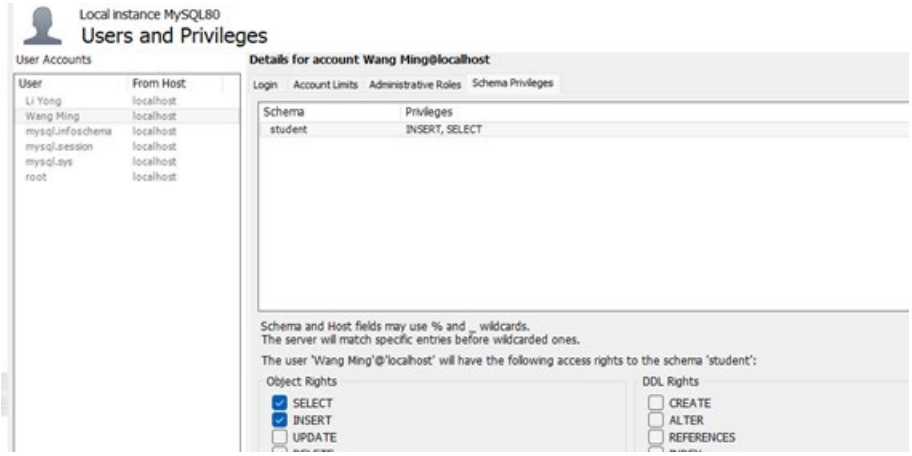




We can see in this two pictures that they have these privileges to do those operations and accessing database is one of them.

Answer No:5(2) Part-1

(1) User Wang Ming has the priviliges to select and insert all tables.



We can see that ,Wang Ming has the priviledge on the tables and she can select and insert all tables.

We can find the show grant query below to show that privileges are given to the user,to validate our answer is correct

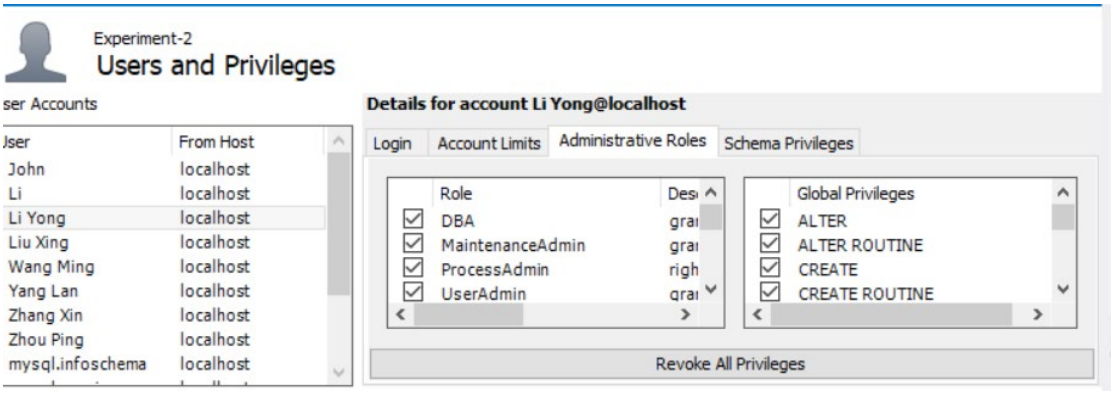
Code:

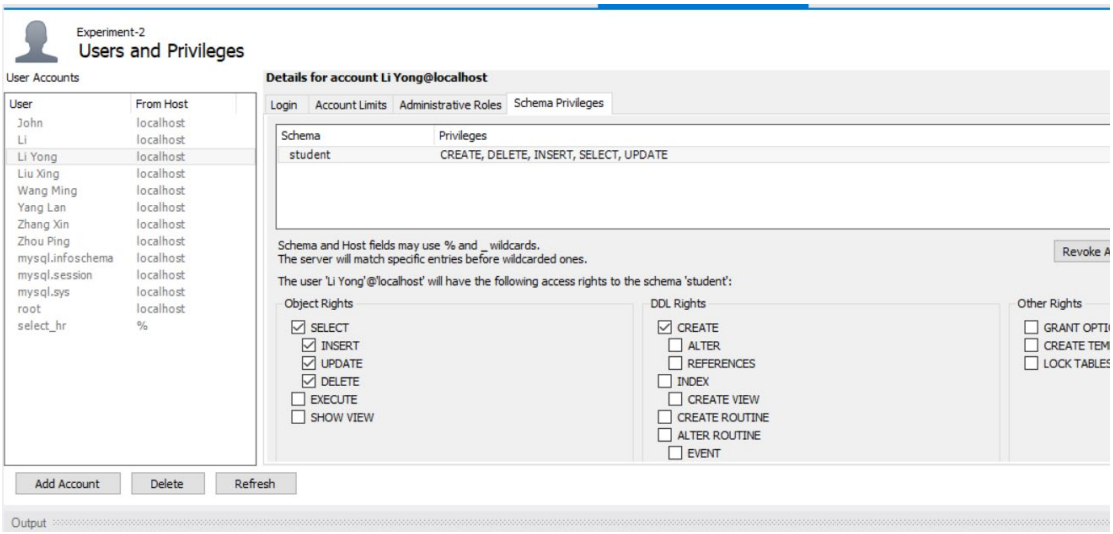
```
GRANT SELECT,INSERT ON school.*TO 'Wang Ming'@'localhost';
```

```
mysql> GRANT SELECT,INSERT ON school.*TO 'Wang Ming'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql> show grants for 'Wang Ming'@'localhost';
+-----------------------------------------------------------------+
| Grants for Wang Ming@localhost                                  |
+-----------------------------------------------------------------+
| GRANT USAGE ON *.* TO `Wang Ming`@`localhost`                   |
| GRANT SELECT, INSERT ON `school`.* TO `Wang Ming`@`localhost`   |
+-----------------------------------------------------------------+
2 rows in set (0.00 sec)
```

Answer No:5(2) Part-2

(2) User Li Yong has the priviliges of select, insert, delete, update and create on the database.





We can see that at the top he has the privilege in database and can select, insert, delete, update and create on the database

We can find the show grant query below to show that privileges are given to the user,to validate our answer is correct

Alternate way by using Mysql Language

Code:

GRANT SELECT,INSERT,DELETE,UPDATE,CREATE ON *.*TO 'Li Hong'@'localhost';
show grants for 'Li Hong@'localhost';

```
mysql> grant select, insert, delete, update,create on *.* to 'Li Yong'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql>
```

```
mysql> create user 'Li Yong' @'localhost 'identified by' 123456 ';
Query OK, 0 rows affected (0.02 sec)

mysql> grant select, insert, delete, update,create on *.* to 'Li Yong'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.01 sec)

mysql> show grants for 'Li Yong'@'localhost';
+-----------------------------------------------------------------------------+
| Grants for Li Yong@localhost                                                |
+-----------------------------------------------------------------------------+
| GRANT SELECT, INSERT, UPDATE, DELETE, CREATE ON *.* TO `Li Yong`@`localhost` |
+-----------------------------------------------------------------------------+
1 row in set (0.00 sec)
```

(2) Use SQL statement to authorize and withdraw permissions and verify permissions.

Requirements: create relevant users and specified database tables, complete authorization and authority verification for each question, and then withdraw the authority and verify the authority.

<u>Answer No:5</u>

(1) User Wang Ming has query privilege on two tables.

```
mysql> GRANT ALL ON *.* TO 'Wang Ming'@'localhost';
Query OK, 0 rows affected (0.03 sec)

mysql> GRANT SELECT ON student.c TO 'Wang Ming'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT SELECT ON student.s TO 'Wang Ming'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql>
```

Code:

GRANT SELECT ON student.c TO 'Wang Ming'@'localhost';

GRANT SELECT ON student.s TO 'Wang Ming'@'localhost';

```
mysql> show grants for 'Wang Ming'@'localhost';
+--------------------------------------------------------------------+
| Grants for Wang Ming@localhost                                      |
+--------------------------------------------------------------------+
| GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, RELOAD, SHUTDOWN, PROCESS, FILE, REFERENCES, INDEX, ALTER, SHOW DATABASES, SUPER, CREATE TEMPORARY TABLES, LOCK TABLES, EXECUTE, REPLICATION SLA
ON CLIENT, CREATE VIEW, SHOW VIEW, CREATE ROUTINE, ALTER ROUTINE, CREATE USER, EVENT, TRIGGER, CREATE TABLESPACE, CREATE ROLE, DROP ROLE ON *.* TO `Wang Ming`@`localhost` |
| GRANT APPLICATION_PASSWORD_ADMIN,AUDIT_ADMIN,BACKUP_ADMIN,BINLOG_ADMIN,BINLOG_ENCRYPTION_ADMIN,CLONE_ADMIN,CONNECTION_ADMIN,ENCRYPTION_KEY_ADMIN,FLUSH_OPTIMIZER_COSTS,FLUSH_STATUS,FLUSH_TABLES,FLU
RCES,GROUP_REPLICATION_ADMIN,INNODB_REDO_LOG_ARCHIVE,INNODB_REDO_LOG_ENABLE,PERSIST_RO_VARIABLES_ADMIN,REPLICATION_APPLIER,REPLICATION_SLAVE_ADMIN,RESOURCE_GROUP_ADMIN,RESOURCE_GROUP_USER,ROLE_ADMIN
ECTION_ADMIN,SESSION_VARIABLES_ADMIN,SET_USER_ID,SHOW_ROUTINE,SYSTEM_USER,SYSTEM_VARIABLES_ADMIN,TABLE_ENCRYPTION_ADMIN,XA_RECOVER_ADMIN ON *.* TO `Wang Ming`@`localhost` |
| GRANT SELECT ON `student`.`c` TO `Wang Ming`@`localhost`            |
| GRANT SELECT ON `student`.`s` TO `Wang Ming`@`localhost`            |
+--------------------------------------------------------------------+
4 rows in set (0.01 sec)
```

(2) User Li Yong has insert and delete privileges on the two tables.

```
mysql> GRANT INSERT,DELETE ON student.s TO 'Li Yong'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT INSERT,DELETE ON student.c TO 'Li Yong'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql> show grants for 'Li Yong'@'localhost';
+-----------------------------------------------------------------------------+
| Grants for Li Yong@localhost                                                |
+-----------------------------------------------------------------------------+
| GRANT SELECT, INSERT, UPDATE, DELETE, CREATE ON *.* TO `Li Yong`@`localhost` |
| GRANT INSERT, DELETE ON `student`.`c` TO `Li Yong`@`localhost`              |
| GRANT INSERT, DELETE ON `student`.`s` TO `Li Yong`@`localhost`              |
+-----------------------------------------------------------------------------+
3 rows in set (0.00 sec)
```

(3) Each instructor only has the right to query his own record.

```
mysql> use student
Database changed
mysql> create user 'Li'@'localhost 'identified by' 123456 ';
Query OK, 0 rows affected (0.03 sec)

mysql> GRANT SELECT ON student.instructor TO 'Li'@'localhost ';
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT SHOW VIEW ON student.instructor TO 'Li'@'localhost ';
Query OK, 0 rows affected (0.01 sec)
```

CODE:

GRANT SELECT ON student. Instructor TO 'Li'@'localhost ';

GRANT SHOW VIEW ON student.instructor TO 'Li'@'localhost ';

```
mysql> show grants for 'Li'@'localhost';
+-----------------------------------------------------------------------------+
| Grants for Li@localhost                                                     |
+-----------------------------------------------------------------------------+
| GRANT USAGE ON *.* TO `Li`@`localhost`                                      |
| GRANT SELECT, SHOW VIEW ON `student`.`instructor` TO `Li`@`localhost`       |
+-----------------------------------------------------------------------------+
2 rows in set (0.00 sec)

mysql>
```

CREATE VIEW instructor AS
SELECT

teacher.teacher_id,teacher.teacher_name,teacher.birth,teacher.department
FROM teacher
where teacher_id ='1';

This view was used to create the view on instructor the I gave the privilege of this view to each instructor individually.

```
mysql> SHOW FULL TABLES
    -> WHERE table_type = 'VIEW';
+-------------------+------------+
| Tables_in_student | Table_type |
+-------------------+------------+
| instructor        | VIEW       |
+-------------------+------------+
1 row in set (0.01 sec)

mysql> SELECT teacher instructor
    -> ^C
mysql> SHOW CREATE VIEW instructor;
+------------+--------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------+----------------------+----------------------+
| View       | Create View
                                                                                           | character_set_client | collation_connection |
+------------+--------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------+----------------------+----------------------+
| instructor | CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY DEFINER VIEW `instructor` AS select `teacher`.`teacher_id` AS `teacher_id`,`teacher`.`teacher_name` AS `teacher_name`,`teacher`.`
birth` AS `birth`,`teacher`.`department` AS `department` from `teacher` where (`teacher`.`teacher_id` = '1') | utf8mb4              | utf8mb4_0900_ai_ci   |
+------------+--------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------+----------------------+----------------------+
1 row in set (0.00 sec)

mysql> SELECT * FROM student.instructor;
+------------+--------------+---------------------+------------+
| teacher_id | teacher_name | birth               | department |
+------------+--------------+---------------------+------------+
|          1 | Li           | 1980-01-01 00:00:00 | CS         |
+------------+--------------+---------------------+------------+
1 row in set (0.00 sec)
```

We can see that instructor can only see his information we have granted him that privileges.

(4)User Liu Xing has query privilege on employee table and update privilege on salary field.

```sql
CREATE VIEW instructor AS
SELECT teacher.teacher_id,teacher.teacher_name,teacher.birth,teacher.department
FROM teacher
where teacher_id ='1';
```

Output

Action Output

#	Time	Action	Message
1	15:09:04	CREATE VIEW instructor AS SELECT teacher.teacher_id,teacher.teacher_name,teacher.birth,teacher.departm...	0 row(s) affected
2	15:09:09	SELECT * FROM student.instructor LIMIT 0, 1000	1 row(s) returned
3	15:24:12	SHOW CREATE VIEW instructor	1 row(s) returned
4	15:24:57	SELECT * FROM student.instructor LIMIT 0, 1000	1 row(s) returned
5	15:34:53	Apply changes to employee	Changes applied
6	15:35:01	SELECT * FROM student.employee LIMIT 0, 1000	0 row(s) returned
7	15:39:29	create view EmployeeV as select * from employee where salary <= 50000 with check option	0 row(s) affected
8	15:39:39	SELECT * FROM student.employeev LIMIT 0, 1000	5 row(s) returned

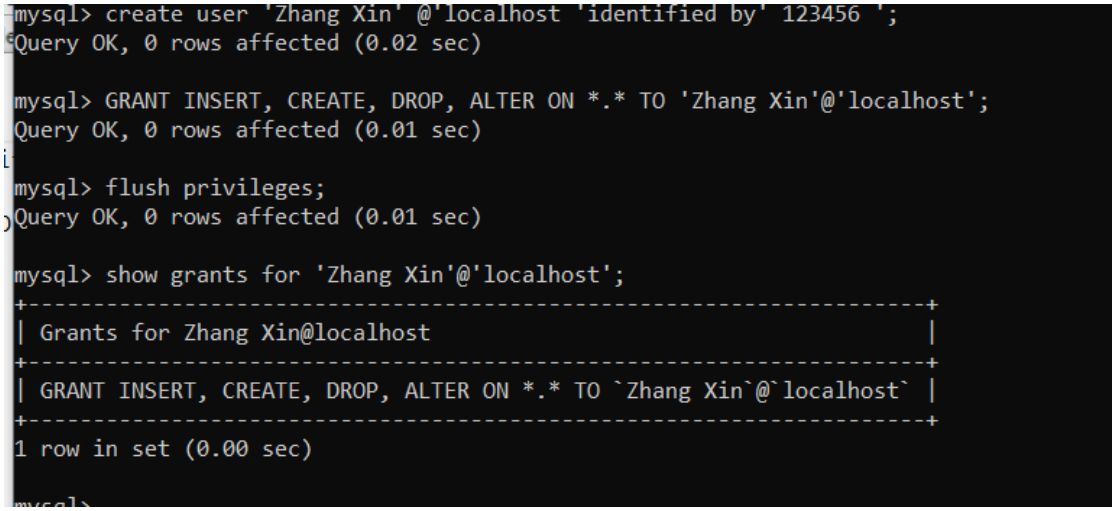
Code:

view EmployeeV as select * from employee
where salary <= 50000 with check option;

Code:

GRANT UPDATE(salary) ON EmployeeV TO 'Liu Xing'@'localhost';
GRANT INSERT ON EmployeeV TO 'Liu Xing'@'localhost';

(5) User Zhang Xin has the right to modify the structure of the two tables.



```
mysql> GRANT INSERT,CREATE,DROP,ALTER ON student.s TO 'Zhang Xin'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql>  GRANT INSERT,CREATE,DROP,ALTER ON student.sc TO 'Zhang Xin'@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

Code:

GRANT INSERT,CREATE,DROP,ALTER ON student.s TO 'Zhang Xin'@'localhost';

GRANT INSERT,CREATE,DROP,ALTER ON student.sc TO 'Zhang Xin'@'localhost';

```
mysql> show grants for 'Zhang Xin'@'localhost';
+-----------------------------------------------------------------------------------------------+
| Grants for Zhang Xin@localhost                                                                |
+-----------------------------------------------------------------------------------------------+
| GRANT INSERT, CREATE, DROP, ALTER ON *.* TO `Zhang Xin`@`localhost`                           |
| GRANT INSERT, CREATE, DROP, ALTER ON `student`.`instructor` TO `Zhang Xin`@`localhost`       |
| GRANT INSERT, CREATE, DROP, ALTER ON `student`.`s` TO `Zhang Xin`@`localhost`                 |
| GRANT INSERT, CREATE, DROP, ALTER ON `student`.`sc` TO `Zhang Xin`@`localhost`                |
+-----------------------------------------------------------------------------------------------+
4 rows in set (0.00 sec)
```

```
mysql> ALTER TABLE `student`.`s` CHANGE COLUMN `SAGE` `STUDENT_AGE` INT NULL DEFAULT NULL ;
Query OK, 0 rows affected (0.03 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> show tables;
+-------------------+
| Tables_in_student |
+-------------------+
| c                 |
| employee          |
| employeev         |
| instructor        |
| s                 |
| sc                |
| teacher           |
+-------------------+
7 rows in set (0.01 sec)

mysql> select*from s;
+------+-----------+---------+------------+--------+-------------+--------------+
| SNO  | SNAME     | SGENDER | SBIRTH     | SDEPTH | STUDENT_AGE | TNAME        |
+------+-----------+---------+------------+--------+-------------+--------------+
| 2001 | Yong Li   | male    | 2000-01-01 | MA     |        NULL | NULL         |
| 2002 | Liu Chen  | female  | 2001-02-01 | IS     |        NULL | NULL         |
| 2003 | Wang Min  | female  | 1999-10-01 | CS     |        NULL | NULL         |
| 2004 | Zhang Li  | male    | 2001-06-01 | IS     |        NULL | NULL         |
| 2005 | Zhang Xin | female  | 2002-09-10 | Che    |        NULL | LIU LAOSHI   |
| 2010 | Ming Li   | female  | 2010-06-01 | IS     |        NULL | ZHANG LAOSHI |
| 2011 | Yi Hong   | male    | 2011-06-01 | IS     |        NULL | LIU LAOSHI   |
| 2012 | Liu Xing  | male    | 2000-09-09 | Phy    |        NULL | TIM LAOSHI   |
+------+-----------+---------+------------+--------+-------------+--------------+
8 rows in set (0.00 sec)
```

We can see that Zhang Xin has the privileges to alter the name of the column and she can also edit other stuctures of table if she want.It was

SAGE after gaving her the permission then I selected her as a user and used her privileges to change the name of the column to STUDENT_AGE

She needs all these operations INSERT,CREATE,DROP,ALTER to change the structures of tables.We were successfully manage to do it.

(6) User Zhou Ping has all permissions on the two tables, and has the authority to authorize other users.

Code:

GRANT ALL ON student.s TO 'Zhou Ping'@'localhost' WITH GRANT OPTION;

GRANT ALL ON student.sc TO 'Zhou Ping'@'localhost' WITH GRANT OPTION;

show grants for 'Zhou Ping'@'localhost';

```
mysql> GRANT ALL ON student.s TO 'Zhou Ping'@'localhost' WITH GRANT OPTION;
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT ALL ON student.sc TO 'Zhou Ping'@'localhost' WITH GRANT OPTION;
Query OK, 0 rows affected (0.01 sec)

mysql> show grants for 'Zhou Ping'@'localhost';
+-------------------------------------------------------------------------------------+
| Grants for Zhou Ping@localhost                                                      |
+-------------------------------------------------------------------------------------+
| GRANT USAGE ON *.* TO `Zhou Ping`@`localhost`                                       |
| GRANT ALL PRIVILEGES ON `student`.`s` TO `Zhou Ping`@`localhost` WITH GRANT OPTION  |
| GRANT ALL PRIVILEGES ON `student`.`sc` TO `Zhou Ping`@`localhost` WITH GRANT OPTION |
+-------------------------------------------------------------------------------------+
3 rows in set (0.00 sec)
```

create role select_hr;
grant select on student.sc to select_hr;
grant select on student.s to select_hr;;

```
mysql> CREATE ROLE select_hr;
Query OK, 0 rows affected (0.01 sec)

mysql> grant select on student.s to select_hr;
Query OK, 0 rows affected (0.01 sec)
```

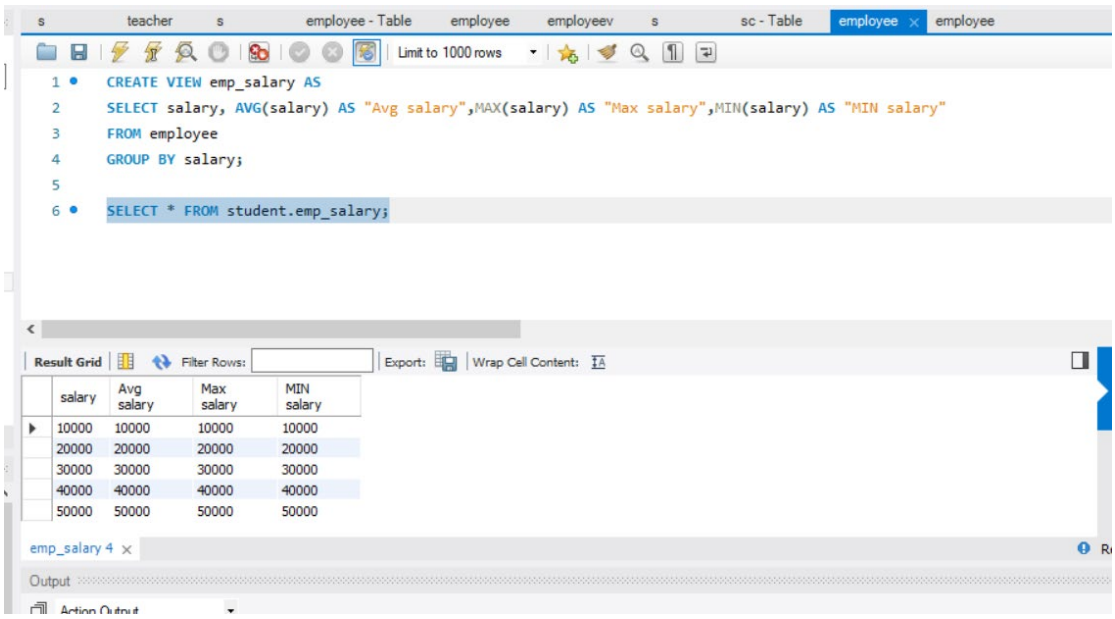
grant select_hr to 'Zhou Ping'@'localhost' with admin option;

grant select_hr to 'Zhang Xin'@'localhost';

(7) User Yang Lan has the privilige to query the maximum wage, minimum wage and average wage from each department employee, but he cannot view the salary of each person.

```
mysql> create user 'Yang Lan'@'localhost 'identified by' 123456 ';
Query OK, 0 rows affected (0.02 sec)

mysql> GRANT SHOW VIEW ON student.emp_salary TO 'Yang Lan'@'localhost ';
Query OK, 0 rows affected (0.01 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.01 sec)

mysql> show grants for 'Yang Lan'@'localhost';
+-----------------------------------------------------------------------+
| Grants for Yang Lan@localhost                                         |
+-----------------------------------------------------------------------+
| GRANT USAGE ON *.* TO `Yang Lan`@`localhost`                          |
| GRANT SHOW VIEW ON `student`.`emp_salary` TO `Yang Lan`@`localhost` |
+-----------------------------------------------------------------------+
2 rows in set (0.00 sec)

mysql>
```

Gave the permission to only see the avg,max and min but cann't see own salary



In Mysql workbench did the view function to separate min ,max and avg

```
mysql> SELECT salary, AVG(salary) AS "Avg salary",MAX(salary) AS "Max salary",MIN(salary) AS "MIN salary"
    -> FROM employee
    -> GROUP BY salary;
+--------+------------+------------+------------+
| salary | Avg salary | Max salary | MIN salary |
+--------+------------+------------+------------+
| 10000  |      10000 | 10000      | 10000      |
| 20000  |      20000 | 20000      | 20000      |
| 30000  |      30000 | 30000      | 30000      |
| 40000  |      40000 | 40000      | 40000      |
| 50000  |      50000 | 50000      | 50000      |
+--------+------------+------------+------------+
5 rows in set (0.00 sec)
```

Also,we can do by using command line did the view function to separate min ,max and avg.

CREATE VIEW emp_salary AS
SELECT salary, AVG(salary) AS "Avg salary",MAX(salary) AS "Max salary",MIN(salary) AS "MIN salary"
FROM employee
GROUP BY salary;

Successfully fulfilled the condition of the questions.

Problems:

At the beginning I had some GUI errors and many syntax errors because It felt very complicated at the beginning ,shifting from GUI to Sql statements.

Solutions:

To solve these problems which I faced during doing this practical, I took help from internet especially YouTube,StackOverflow and W3school to get information about these errors for the solution. I also asked the teacher to help me understand them. And provided instructions helped to solve some of my errors during the experiment.

Summary:

From this experiment I have learned SQL statement to create database and tables then I had learned how to manage the tables. Gradually becoming familiar with SQL statements of data insertion, modification and deletion of basic columns. Furthermore, I also learn that how to put a default value in a column. Learned all kinds of data operation about basic table in GUI. This experiment taught me to understand SQL statement of data query. Along with that this experiment also tells us about the foreign key.Also, I learned multi-user experience.

Attachments:

1) DB3_2019380141_ABID ALI.docx
2) DB3_2019380141_ABID ALI.pdf

References:

1)https://www.youtube.com/watch?v=wHvSTcg4Lyw&list=PLHcEm2S4__GGtrh3DaAT787NdBsak-j3Q&index=6

2)https://www.youtube.com/watch?v=HzZ3h-COam4&list=PLHcEm2S4__GGtrh3DaAT787NdBsak-j3Q&index=7

3) https://www.w3schools.com/

4) https://stackoverflow.com/

5) https://youtube.com/

6) https://dev.mysql.com/doc/refman/8.0/en/insert.html

7)https://codereview.stackexchange.com/questions/84171/class-method-to-insert-a-record-into-mysql