DATABASE MANAGEMENT SYSTEMS

TOPIC : Coaching Centre Database Management System

FACULTY : Prof. Bimal Kumar Ray

GROUP MEMBERS :

1. Piyush Sahu	(19BIT0038)
2. Krittika Chaturvedi	(19BIT0071)
3. Ishita Tiwari	(19BIT0077)

REVIEW - 1

INTRODUCTION:

The Miniworld selected by us is the Coaching Center. In the field of education, there has been a tremendous soar in the sheer amount of vying among the students for seats in the best colleges. The reason behind this is not only the rising population in India (1.0 % population growth per annum as per recorded in the year 2018) but also the need for earning money by choosing a rather safe field of work. This has further led to a surge in the number of coaching institutes as well as their branches which are placed in multiple cities across different states of India.

These coachings like any other educational institute are needed to have a well recorded detail of each and every member associated with it irrespective of the post or role played by the individual in any of their bases. Needless to say, the students as well as the faculty's data must under all conditions be recorded. An incomplete record could lead to various problems such as fee payment mismatches, salary issues, missing logs of personal details under states of emergency etc.

While it is important for them to keep a track of student, his personal details, his course, his fee payments, faculties, their personal details, their salaries and departments, it is also equally important to keep a well documented record of the admission process, the exams in which the student or to-be student is appearing as well as the administration details. These details must constantly be updated which tells us that the data must be easily accessible and modifying it must never be an ordeal.

DATA REQUIREMENTS:

Entities :

1. Administration Department: is an entity type which manages all departments of the coaching institute.It has following attributes-departments, total number of staff employed, offices.

2. Student: is an entity type which has attributes - composite attribute Name(first name, middle name, last name), attribute Personal Details(date of birth, gender, caste, nationality, address), multivalued attribute Phone Number of students,vehicle id, composite attribute Student admission(fees, year of joining,batch, scholarship amount), Programme opted, branch of coaching institute (determined by the location/area), Attendance (Subject wise, selected duration, selected subject, student) . Every student must have a unique id which cannot have null value .

3. Faculty: is an entity type which has attributes - Name(first name, middle name, last name), Subject taught, Area of specialization, No. of years of experience of, post , year of joining and Attendance. Each faculty must have a unique id which cannot have null value.

4. Transportation: students use transportation services. Each vehicle must have a unique vehicle id , multivalued attribute driver phone no. and driver id. The address of the student is used to determine the area to which the transportation service is provided .

5. Programmes : every student chooses a programme having the programme period in years. Each programme must have a unique programme type that can't be null. Each programme has a particular no. of students enrolled, and the no. of faculties.

6. Course Material: is a weak entity type which has Student and Programme as its strong entity type. The programme provides course material to every student. Each course has subjects, course type, quantity of material required and the quantity of material still remaining in the stack (status).

7. Examination : every student writes an exam on a regular basis. The exam details include the id of student ,exam type of string data type, marks obtained and date of numeric data type, venue, time of exam, if he/she was present or absent in the exam and the attribute Results having class rank, centre rank, success percentage and unique AIR Rank .

8. Admission : each student writes a particular type of entrance exam in order to get admission in the coaching institute. Each student gets a unique registration no. for the entrance exam. The fee structure, course opted and the admission year are also stored

7. Examination : every student writes an exam on a regular basis. The exam details include the id of student ,exam type of string data type, marks obtained and date of numeric data type, venue, time of exam, if he/she was present or absent in the exam and the attribute Results having class rank, centre rank, success percentage and unique AIR Rank .

8. Admission : each student writes a particular type of entrance exam in order to get admission in the coaching institute. Each student gets a unique registration no. for the entrance exam. The fee structure, course opted and the admission year are also stored.

RELATIONSHIPS:

1. Manages (1-M)

Admin manages every Faculty. This implies that the admin keeps a track of the faculty’s personal and professional details including factors such as attendance and experience.

2. Handles (1-M)

Admin handles every Student. It means that each student’s personal detail as well as the academic record is maintained by the admin.

3. Control_ad (1-M)

Admin controls every prospective student’s Admission process irrespective of the branch or the city.

4. Control_e (1-M)

Admin controls each Exam. That is, the admin is incharge of conducting each exam held by the institute on every scale.

5. Control_tr (1-M)
Admin controls Transport. The admin is hence responsible for providing and assigning transportation to the students of the institute.
6. Teaches (M-N)
Faculties teach Students. Every teacher must be incharge of a particular set of students and a coaching centre is bound to have multiple faculties. Therefore, resulting in an M-N relation.
7. Takes(1-1)
Every student must take admission in order to be a part of the institute.
8. Gets (1-1)
Each student necessarily gets c_mat (course material) from the institute. A student gets a carefully curated set of material from the coaching to help the students follow their desired course.
9. C_mat (1-M)
A program must provide c_mat (course material) to the students. This course material depends on the program selected by the student (as c_mat is a weak entity type).
10. Choose (M-1)
Each student must choose a program. A program can be chosen by more than one student, hence justifying why it is an M-1 relation.
11. Uses (M-1)
A student may or may not use the provided transportation. The specific mode of transportation will most definitely carry multiple students. Therefore, the relation is M-1.
12. Writes (M-N)
Students write exams. This is an M-N relation as multiple students write different exams which include class tests and mock tests.

FUNCTIONAL REQUIREMENT:

Removal of old data :

1. Remove the information of the student or faculty if he/she leaves the coaching institute .
2. Removal of the information of the driver if he/she leaves the job.
3. If the exam got cancelled for some reason the particular exam details will not be shown .
4. If the student was absent in the exam then exam results will not be shown .

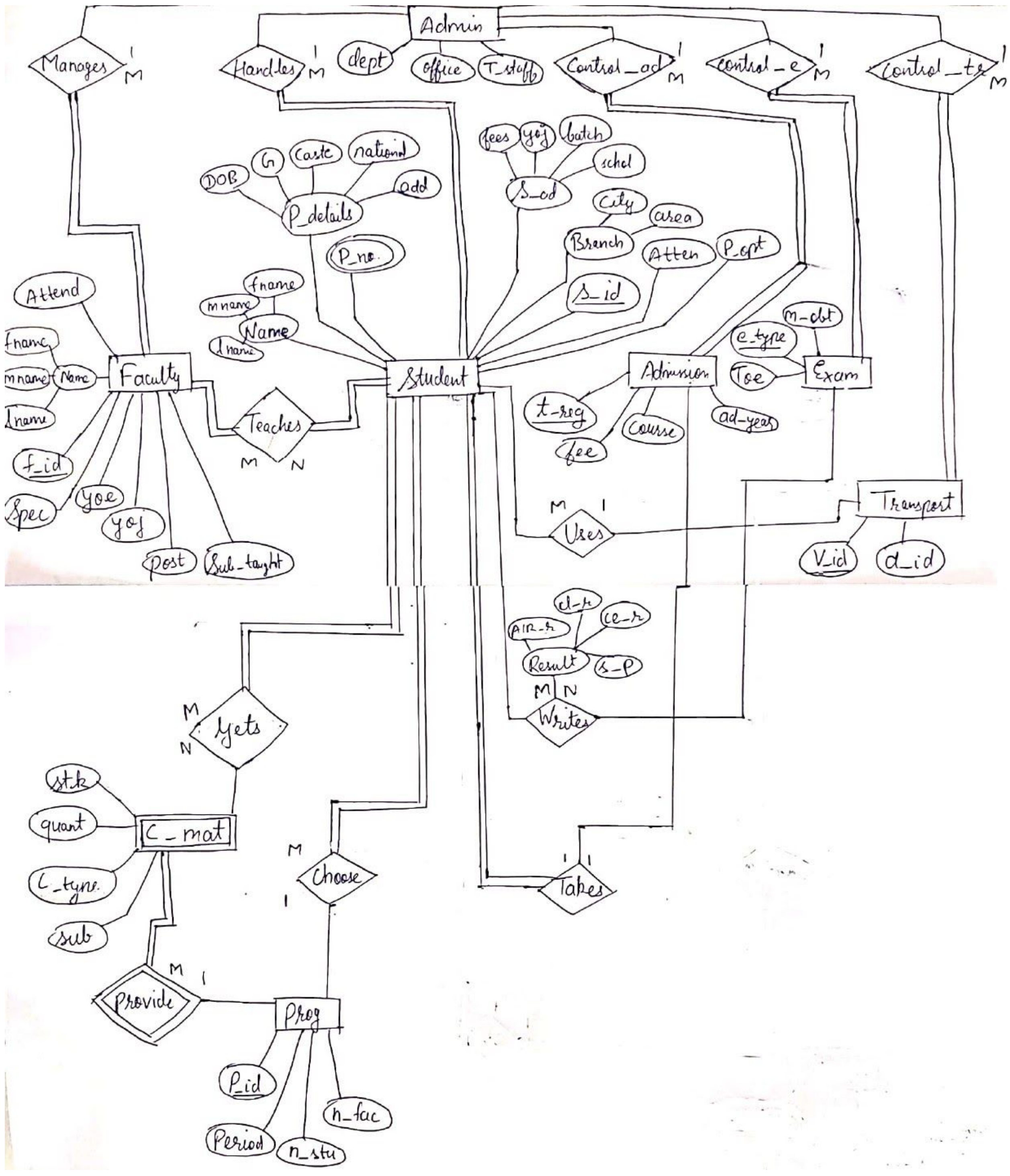
Modification of data :

1. Update results and student information yearly , as new students are selected every year .
2. Update results of weekly tests of admitted students on a weekly basis.
3. Update attendance of student and faculty.
4. Updation of course & batch can be done simply.

Data Retrieval :

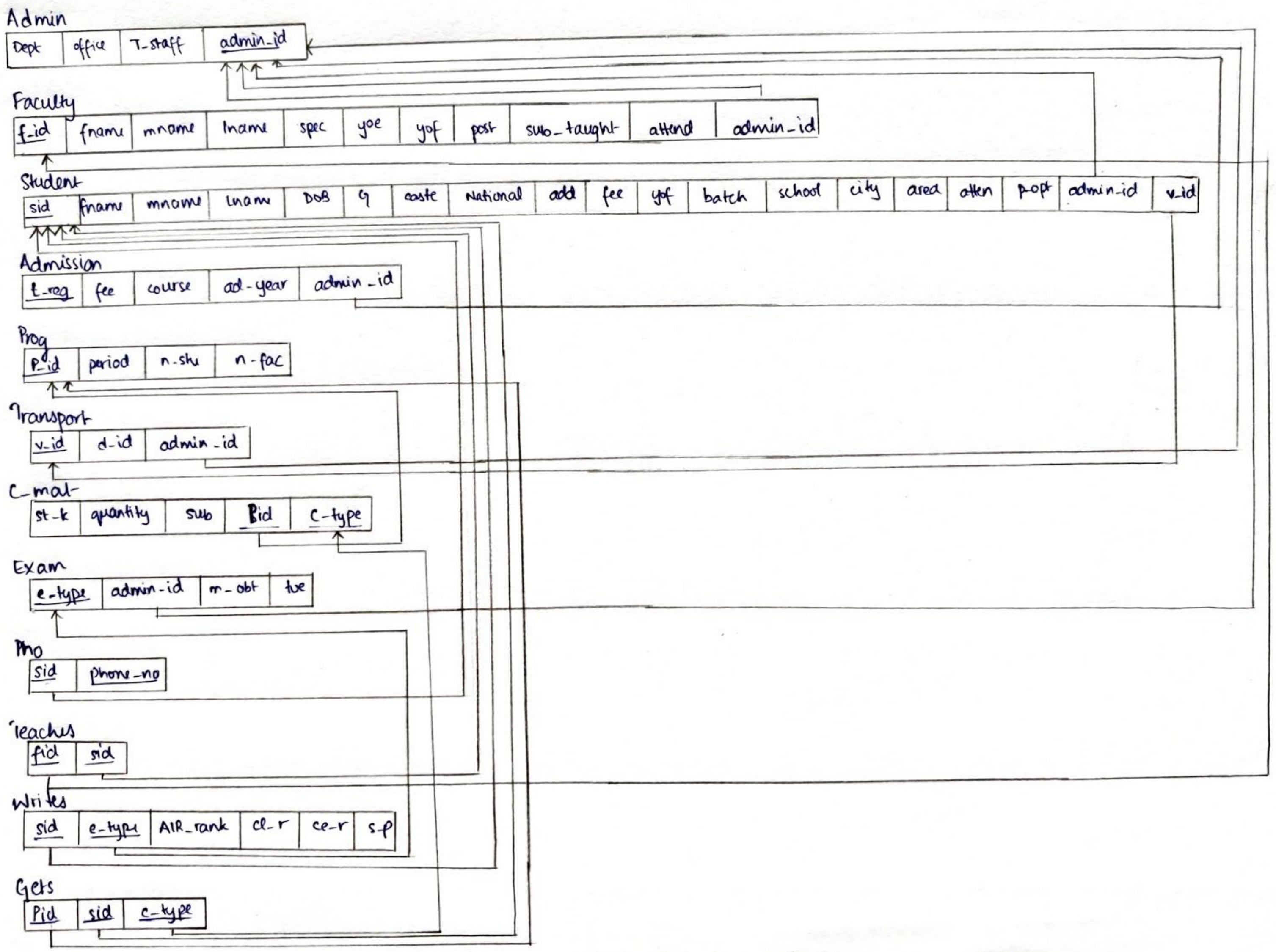
1. Retrieve information of a particular student or faculty .
2. Get details of every test or exam, student wise, course wise or batch wise.
3. View the yearly progress of the coaching centre according to the results in competitive examinations.
4. View the details related to transportation facilities provided to the student by the centre.
5. Also view the status of the course materials provided and keep a record of the distribution.
6. You can check the fee structure of different classes .
7. Amount paid and amount dues can be viewed as per the classes, courses or individual student.
8. Overall absentee reports with remarks can be viewed.

ER DIAGRAM FOR THE ABOVE MINIWORLD:



REVIEW - 2

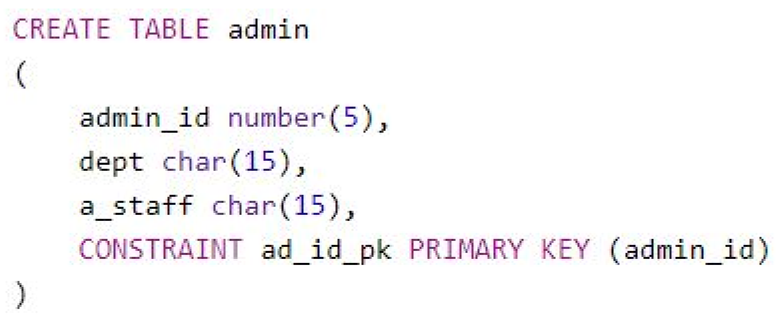
RELATIONAL DATABASE SCHEMA DIAGRAM



CREATING TABLES

Creating table admin

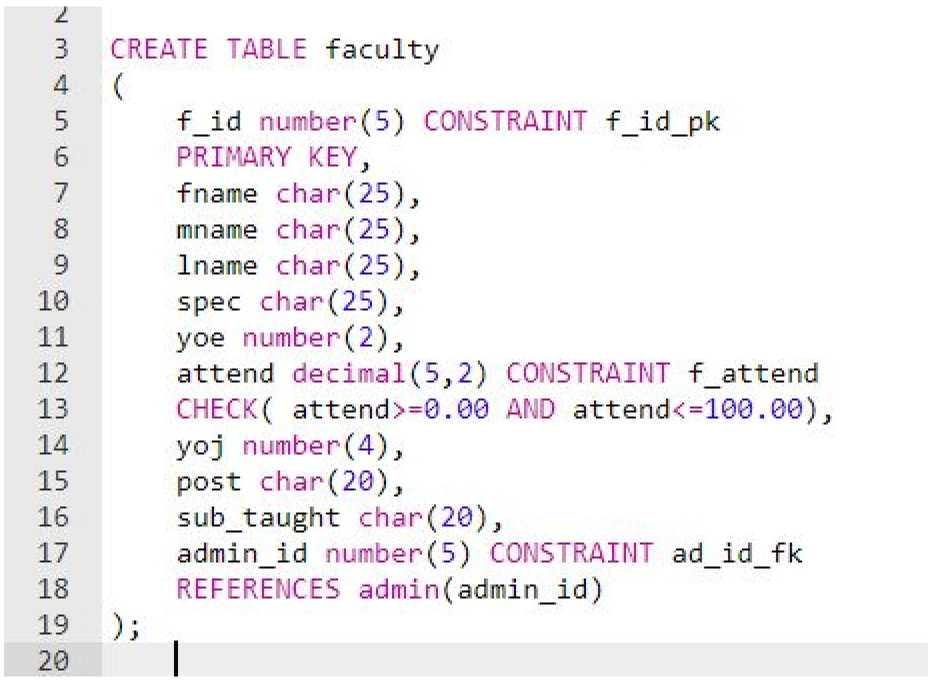
```
CREATE TABLE admin
(
        admin_id number(5),
        dept char(15),
        a_staff char(15),
        CONSTRAINT ad_id_pk PRIMARY KEY (admin_id)
);
```



```
CREATE TABLE admin
(
    admin_id number(5),
    dept char(15),
    a_staff char(15),
    CONSTRAINT ad_id_pk PRIMARY KEY (admin_id)
)
```

Creating table faculty

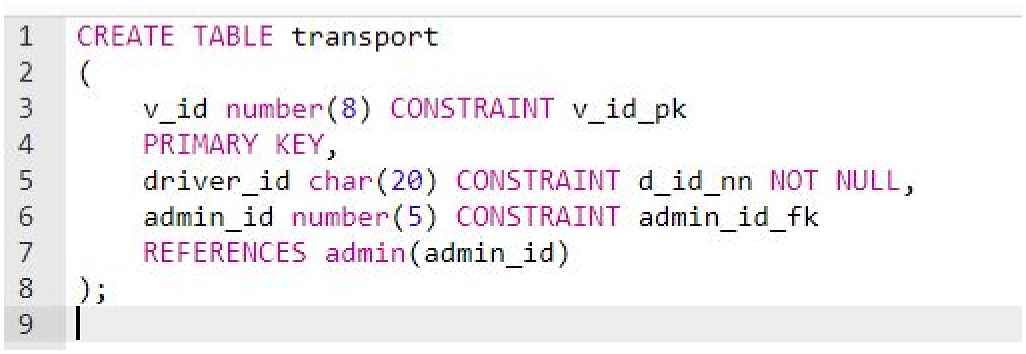
```
CREATE TABLE faculty
(
        f_id number(5) CONSTRAINT f_id_pk
        PRIMARY KEY,
        fname char(25),
        mname char(25),
        lname char(25),
        spec char(25),
        yoe number(2),
        attend decimal(5,2) CONSTRAINT f_attend
        CHECK( attend>=0.00 AND attend<=100.00),
        yoj number(4),
        post char(20),
        sub_taught char(20),
        admin_id number(5) CONSTRAINT ad_id_fk
        REFERENCES admin(admin_id)
);
```



```
CREATE TABLE faculty
(
    f_id number(5) CONSTRAINT f_id_pk
    PRIMARY KEY,
    fname char(25),
    mname char(25),
    lname char(25),
    spec char(25),
    yoe number(2),
    attend decimal(5,2) CONSTRAINT f_attend
    CHECK( attend>=0.00 AND attend<=100.00),
    yoj number(4),
    post char(20),
    sub_taught char(20),
    admin_id number(5) CONSTRAINT ad_id_fk
    REFERENCES admin(admin_id)
);
```

Creating table transport

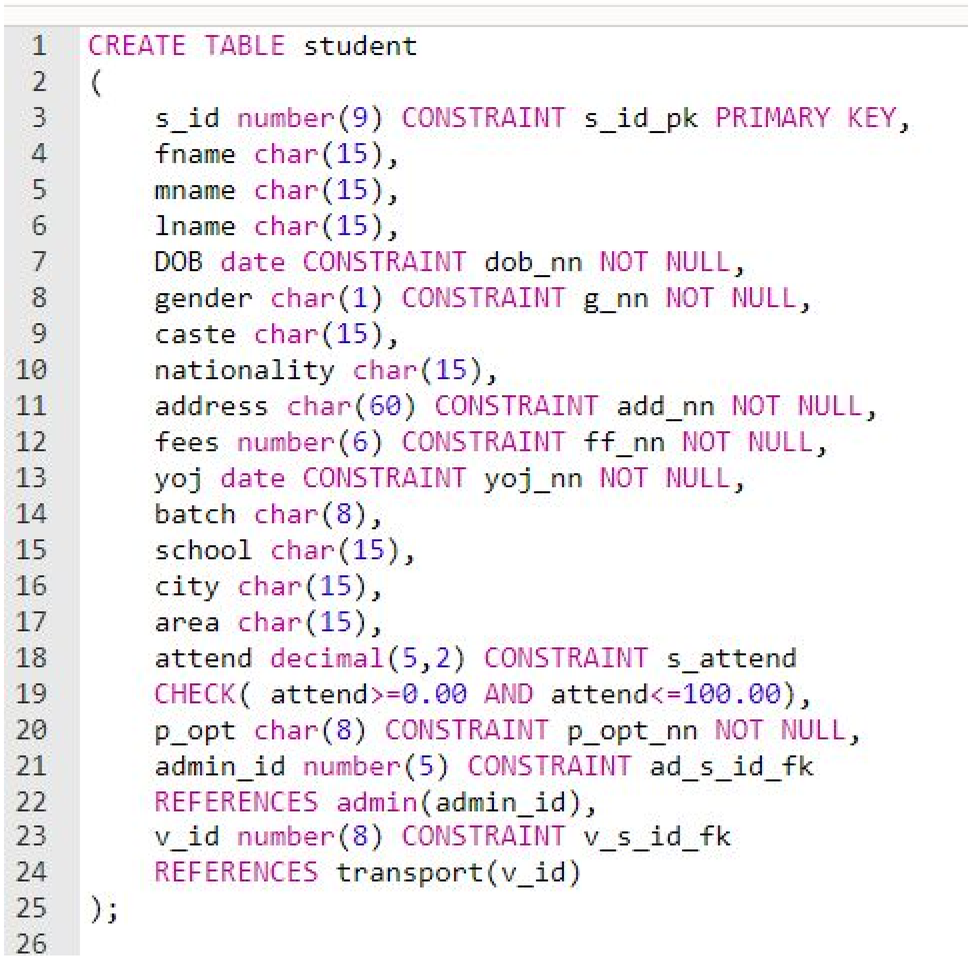
```
CREATE TABLE transport

(

        v_id number(8) CONSTRAINT v_id_pk

        PRIMARY KEY,

        driver_id char(20) CONSTRAINT d_id_nn NOT NULL,

        admin_id number(5) CONSTRAINT admin_id_fk

        REFERENCES admin(admin_id)

);
```



```
CREATE TABLE transport
(
    v_id number(8) CONSTRAINT v_id_pk
    PRIMARY KEY,
    driver_id char(20) CONSTRAINT d_id_nn NOT NULL,
    admin_id number(5) CONSTRAINT admin_id_fk
    REFERENCES admin(admin_id)
);
```

Creating table student

```
CREATE TABLE student

(

        s_id number(9) CONSTRAINT s_id_pk PRIMARY KEY,

        fname char(15),

        mname char(15),
        lname char(15),
        DOB date CONSTRAINT dob_nn NOT NULL,
        gender char(1) CONSTRAINT g_nn NOT NULL,
        caste char(15),
        nationality char(15),
        address char(60) CONSTRAINT add_nn NOT NULL,
        fees number(6) CONSTRAINT ff_nn NOT NULL,
        yoj date CONSTRAINT yoj_nn NOT NULL,
        batch char(8),
        school char(15),
        city char(15),
        area char(15),
        attend decimal(5,2) CONSTRAINT s_attend
        CHECK( attend>=0.00 AND attend<=100.00),
        p_opt char(8) CONSTRAINT p_opt_nn NOT NULL,
        admin_id number(5) CONSTRAINT ad_s_id_fk
        REFERENCES admin(admin_id),
        v_id number(8) CONSTRAINT v_s_id_fk
        REFERENCES transport(v_id)
);
```

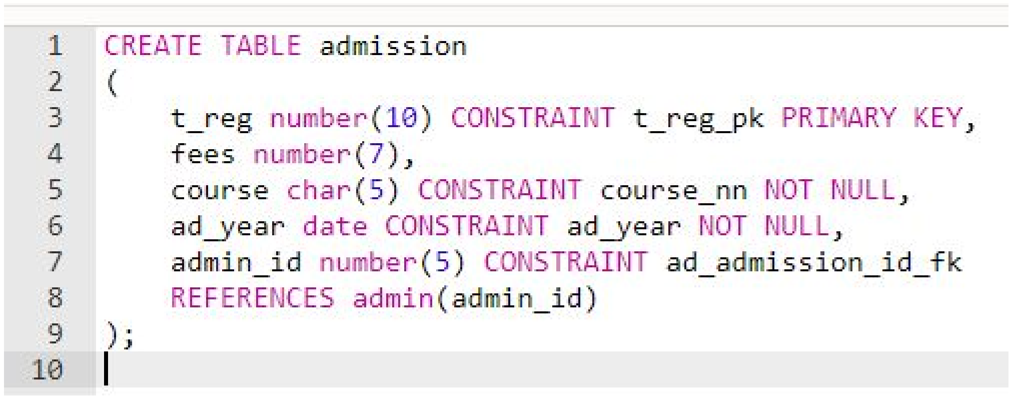


```sql
CREATE TABLE student
(
    s_id number(9) CONSTRAINT s_id_pk PRIMARY KEY,
    fname char(15),
    mname char(15),
    lname char(15),
    DOB date CONSTRAINT dob_nn NOT NULL,
    gender char(1) CONSTRAINT g_nn NOT NULL,
    caste char(15),
    nationality char(15),
    address char(60) CONSTRAINT add_nn NOT NULL,
    fees number(6) CONSTRAINT ff_nn NOT NULL,
    yoj date CONSTRAINT yoj_nn NOT NULL,
    batch char(8),
    school char(15),
    city char(15),
    area char(15),
    attend decimal(5,2) CONSTRAINT s_attend
    CHECK( attend>=0.00 AND attend<=100.00),
    p_opt char(8) CONSTRAINT p_opt_nn NOT NULL,
    admin_id number(5) CONSTRAINT ad_s_id_fk
    REFERENCES admin(admin_id),
    v_id number(8) CONSTRAINT v_s_id_fk
    REFERENCES transport(v_id)
);

```

Creating table admission

```sql
CREATE TABLE admission

(

        t_reg number(10) CONSTRAINT t_reg_pk PRIMARY KEY,

        fees number(7),

        course char(5) CONSTRAINT course_nn NOT NULL,

        ad_year date CONSTRAINT ad_year NOT NULL,

        admin_id number(5) CONSTRAINT ad_admission_id_fk

        REFERENCES admin(admin_id)
```

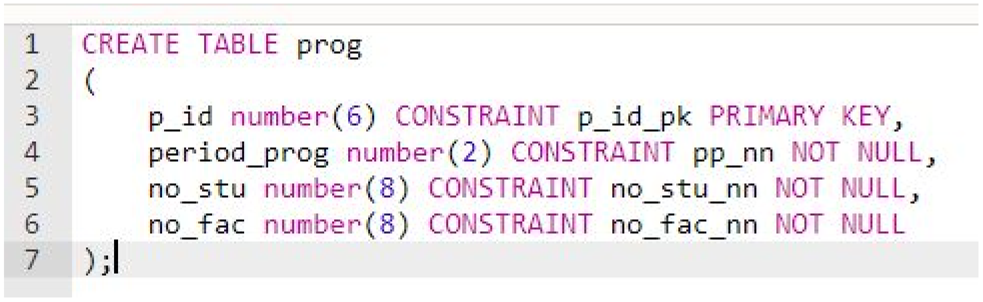
);



```
CREATE TABLE admission
(
    t_reg number(10) CONSTRAINT t_reg_pk PRIMARY KEY,
    fees number(7),
    course char(5) CONSTRAINT course_nn NOT NULL,
    ad_year date CONSTRAINT ad_year NOT NULL,
    admin_id number(5) CONSTRAINT ad_admission_id_fk
    REFERENCES admin(admin_id)
);
```

<u>Creating table prog</u>

```
CREATE TABLE prog
(
        p_id number(6) CONSTRAINT p_id_pk PRIMARY KEY,
        period_prog number(2) CONSTRAINT pp_nn NOT NULL,
        no_stu number(8) CONSTRAINT no_stu_nn NOT NULL,
        no_fac number(8) CONSTRAINT no_fac_nn NOT NULL
);
```



```
CREATE TABLE prog
(
    p_id number(6) CONSTRAINT p_id_pk PRIMARY KEY,
    period_prog number(2) CONSTRAINT pp_nn NOT NULL,
    no_stu number(8) CONSTRAINT no_stu_nn NOT NULL,
    no_fac number(8) CONSTRAINT no_fac_nn NOT NULL
);
```

Creating table course material (c_mat)

CREATE TABLE c_mat

(

st_k number(6) CONSTRAINT st_k_nn NOT NULL,

quantity number(6) CONSTRAINT quan_nn NOT NULL,

c_type char(8) CONSTRAINT c_type_unq UNIQUE,

subject char(8) CONSTRAINT subject_nn NOT NULL,

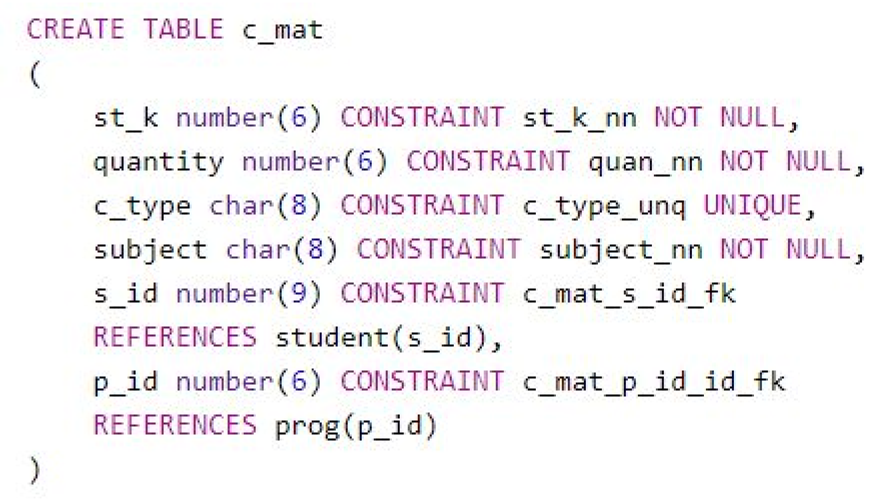
s_id number(9) CONSTRAINT c_mat_s_id_fk

REFERENCES student(s_id),

p_id number(6) CONSTRAINT c_mat_p_id_id_fk

REFERENCES prog(p_id)

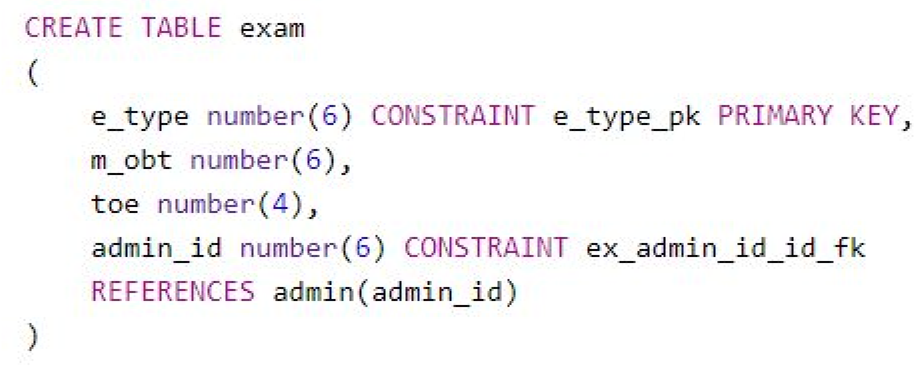
)



```
CREATE TABLE c_mat
(
    st_k number(6) CONSTRAINT st_k_nn NOT NULL,
    quantity number(6) CONSTRAINT quan_nn NOT NULL,
    c_type char(8) CONSTRAINT c_type_unq UNIQUE,
    subject char(8) CONSTRAINT subject_nn NOT NULL,
    s_id number(9) CONSTRAINT c_mat_s_id_fk
    REFERENCES student(s_id),
    p_id number(6) CONSTRAINT c_mat_p_id_id_fk
    REFERENCES prog(p_id)
)
```

Creating table exam

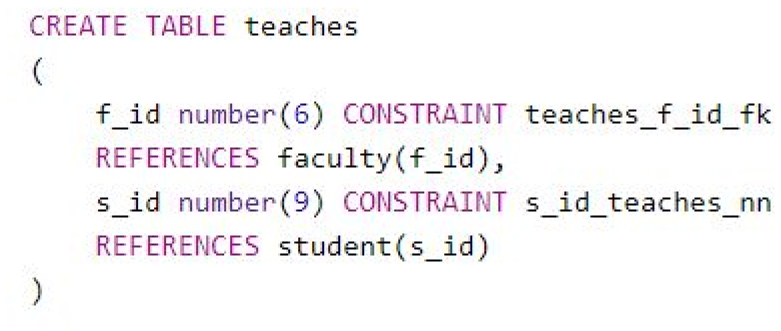
```
CREATE TABLE exam

(

        e_type number(6) CONSTRAINT e_type_pk PRIMARY KEY,

        m_obt number(6),

        toe number(4),

        admin_id number(6) CONSTRAINT ex_admin_id_id_fk

        REFERENCES admin(admin_id)

);
```



```
CREATE TABLE exam
(
    e_type number(6) CONSTRAINT e_type_pk PRIMARY KEY,
    m_obt number(6),
    toe number(4),
    admin_id number(6) CONSTRAINT ex_admin_id_id_fk
    REFERENCES admin(admin_id)
)
```

Creating table teaches

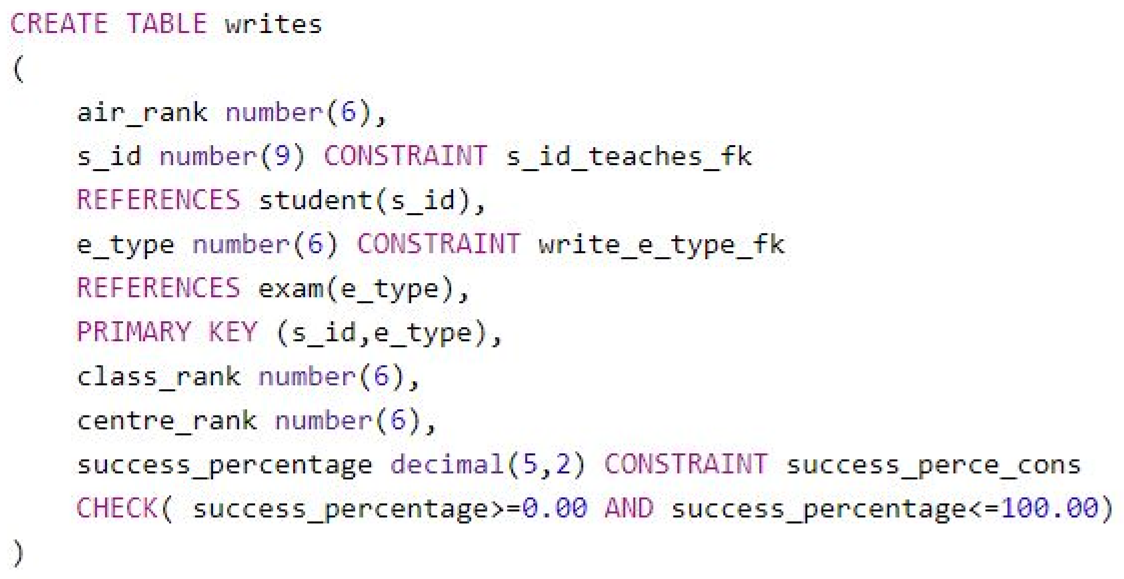
```
CREATE TABLE teaches

(

        f_id number(6) CONSTRAINT teaches_f_id_fk

        REFERENCES faculty(f_id),

        s_id number(9) CONSTRAINT s_id_teaches_nn

        REFERENCES student(s_id)

);
```



```
CREATE TABLE teaches
(
    f_id number(6) CONSTRAINT teaches_f_id_fk
    REFERENCES faculty(f_id),
    s_id number(9) CONSTRAINT s_id_teaches_nn
    REFERENCES student(s_id)
)
```

Creating table writes

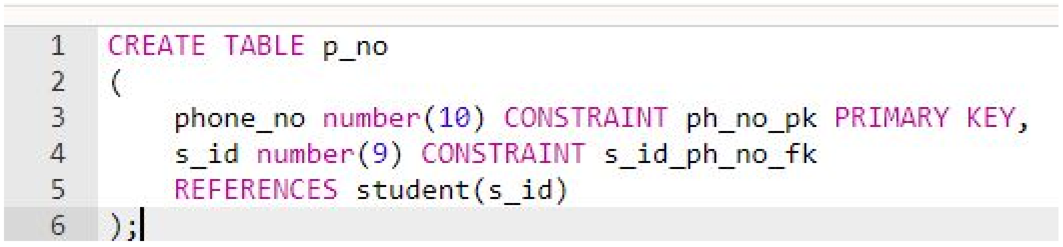
```
CREATE TABLE writes
(
        air_rank number(6) ,
        s_id number(9) CONSTRAINT s_id_teaches_fk
        REFERENCES student(s_id),
        e_type number(6) CONSTRAINT write_e_type_fk
        REFERENCES exam(e_type),
        PRIMARY KEY (s_id,e_type),
        class_rank number(6),
        centre_rank number(6),
        success_percentage decimal(5,2) CONSTRAINT success_perce_cons
        CHECK( success_percentage>=0.00 AND success_percentage<=100.00)
);
```



```
CREATE TABLE writes
(
    air_rank number(6),
    s_id number(9) CONSTRAINT s_id_teaches_fk
    REFERENCES student(s_id),
    e_type number(6) CONSTRAINT write_e_type_fk
    REFERENCES exam(e_type),
    PRIMARY KEY (s_id,e_type),
    class_rank number(6),
    centre_rank number(6),
    success_percentage decimal(5,2) CONSTRAINT success_perce_cons
    CHECK( success_percentage>=0.00 AND success_percentage<=100.00)
)
```

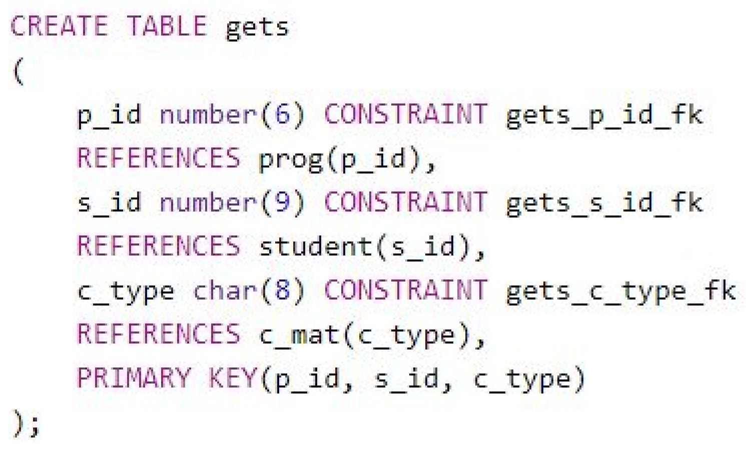
Creating table phone number (p_no)

```
CREATE TABLE p_no

(

        phone_no number(10) CONSTRAINT ph_no_pk PRIMARY KEY,

        s_id number(9) CONSTRAINT s_id_ph_no_fk

        REFERENCES student(s_id)

);
```



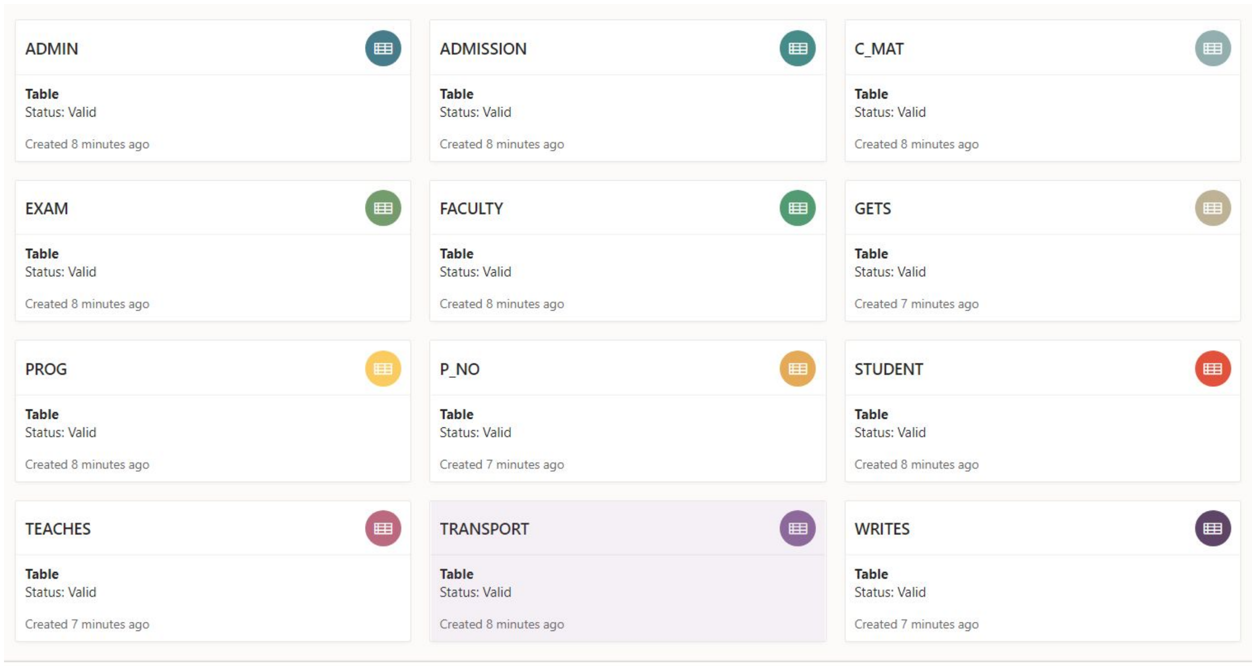
```
CREATE TABLE p_no
(
    phone_no number(10) CONSTRAINT ph_no_pk PRIMARY KEY,
    s_id number(9) CONSTRAINT s_id_ph_no_fk
    REFERENCES student(s_id)
);
```

```
CREATE TABLE gets

(

  p_id number(6) CONSTRAINT gets_p_id_fk

  REFERENCES prog(p_id),

  s_id number(9) CONSTRAINT gets_s_id_fk

  REFERENCES student(s_id),

  c_type char(8) CONSTRAINT gets_c_type_fk

  REFERENCES c_mat(c_type),

  PRIMARY KEY(p_id, s_id, c_type)

);
```



```
CREATE TABLE gets
(
    p_id number(6) CONSTRAINT gets_p_id_fk
    REFERENCES prog(p_id),
    s_id number(9) CONSTRAINT gets_s_id_fk
    REFERENCES student(s_id),
    c_type char(8) CONSTRAINT gets_c_type_fk
    REFERENCES c_mat(c_type),
    PRIMARY KEY(p_id, s_id, c_type)
);
```

OVERVIEW OF TABLE CREATED



ADMIN
Table
Status: Valid
Created 8 minutes ago

ADMISSION
Table
Status: Valid
Created 8 minutes ago

C_MAT
Table
Status: Valid
Created 8 minutes ago

EXAM
Table
Status: Valid
Created 8 minutes ago

FACULTY
Table
Status: Valid
Created 8 minutes ago

GETS
Table
Status: Valid
Created 7 minutes ago

PROG
Table
Status: Valid
Created 8 minutes ago

P_NO
Table
Status: Valid
Created 7 minutes ago

STUDENT
Table
Status: Valid
Created 8 minutes ago

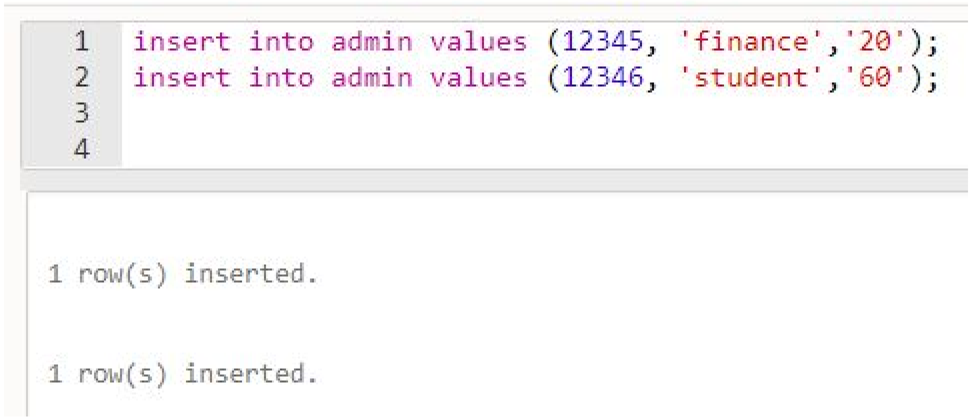
TEACHES
Table
Status: Valid
Created 7 minutes ago

TRANSPORT
Table
Status: Valid
Created 8 minutes ago

WRITES
Table
Status: Valid
Created 7 minutes ago

QUERY INSERTION

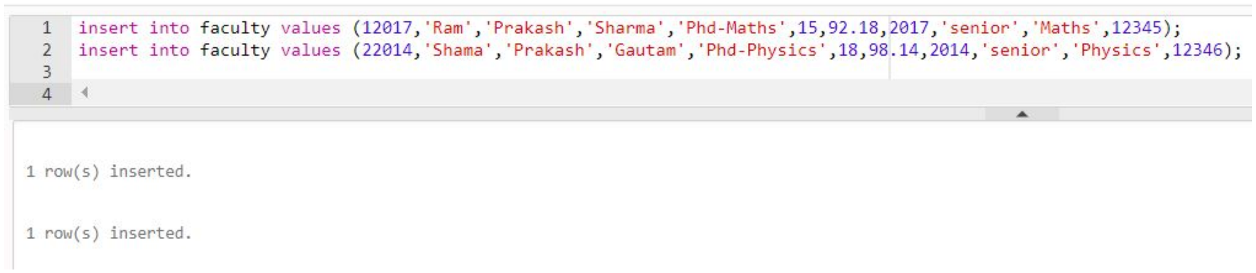
Admin Table



```sql
insert into admin values (12345, 'finance','20');
insert into admin values (12346, 'student','60');
```

```
1 row(s) inserted.

1 row(s) inserted.
```

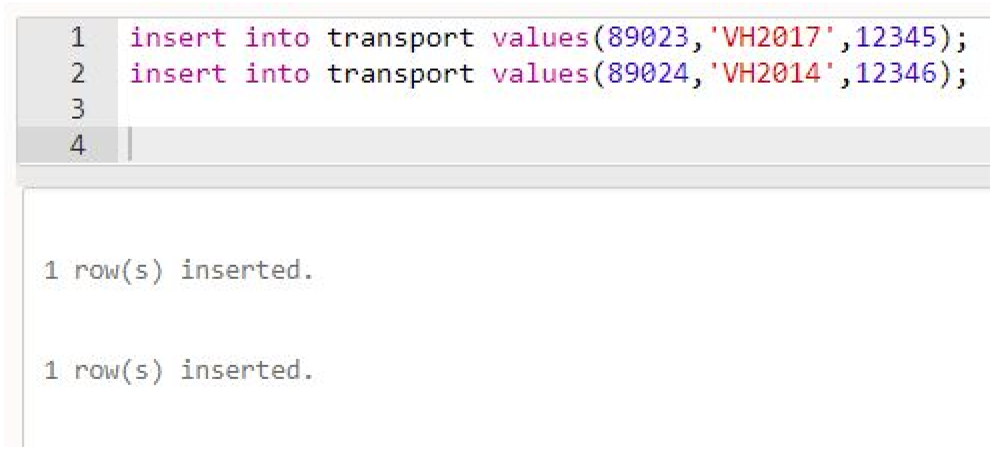
Faculty Table



```sql
insert into faculty values (12017,'Ram','Prakash','Sharma','Phd-Maths',15,92.18,2017,'senior','Maths',12345);
insert into faculty values (22014,'Shama','Prakash','Gautam','Phd-Physics',18,98.14,2014,'senior','Physics',12346);
```

```
1 row(s) inserted.

1 row(s) inserted.
```

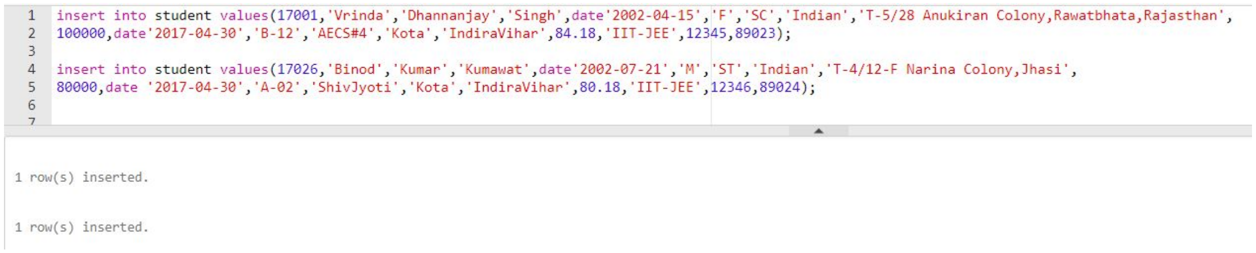
Transport Table



```sql
insert into transport values(89023,'VH2017',12345);
insert into transport values(89024,'VH2014',12346);
```

```
1 row(s) inserted.

1 row(s) inserted.
```

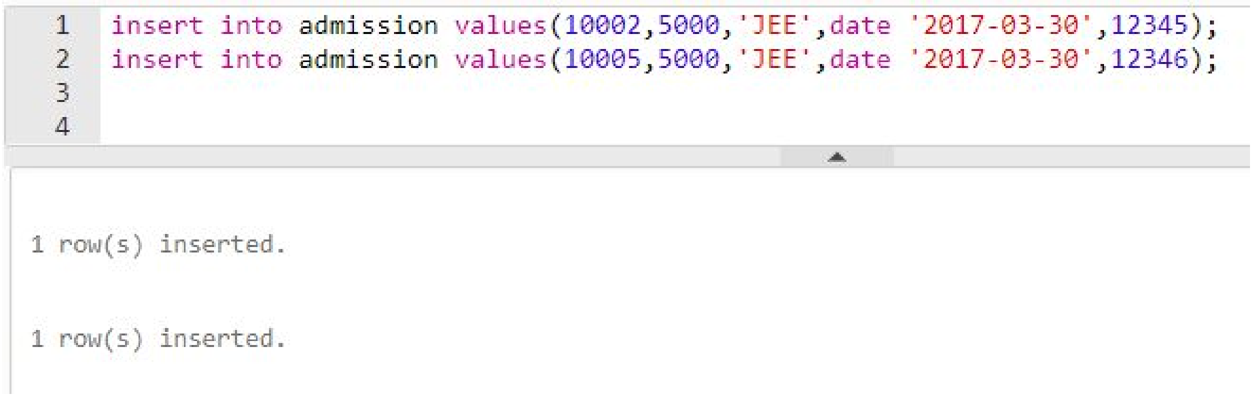
Student Table



```
insert into student values(17001,'Vrinda','Dhannanjay','Singh',date'2002-04-15','F','SC','Indian','T-5/28 Anukiran Colony,Rawatbhata,Rajasthan',
100000,date'2017-04-30','B-12','AECS#4','Kota','IndiraVihar',84.18,'IIT-JEE',12345,89023);

insert into student values(17026,'Binod','Kumar','Kumawat',date'2002-07-21','M','ST','Indian','T-4/12-F Narina Colony,Jhasi',
80000,date '2017-04-30','A-02','ShivJyoti','Kota','IndiraVihar',80.18,'IIT-JEE',12346,89024);
```

```
1 row(s) inserted.

1 row(s) inserted.
```

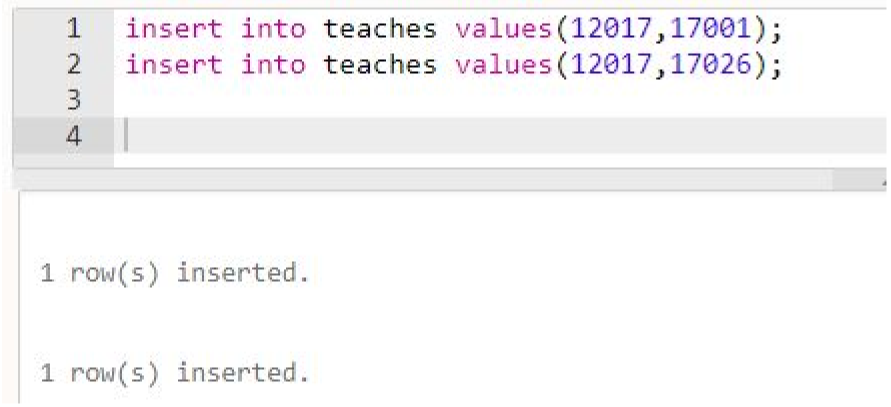
Admission Table



```
insert into admission values(10002,5000,'JEE',date '2017-03-30',12345);
insert into admission values(10005,5000,'JEE',date '2017-03-30',12346);
```

```
1 row(s) inserted.

1 row(s) inserted.
```

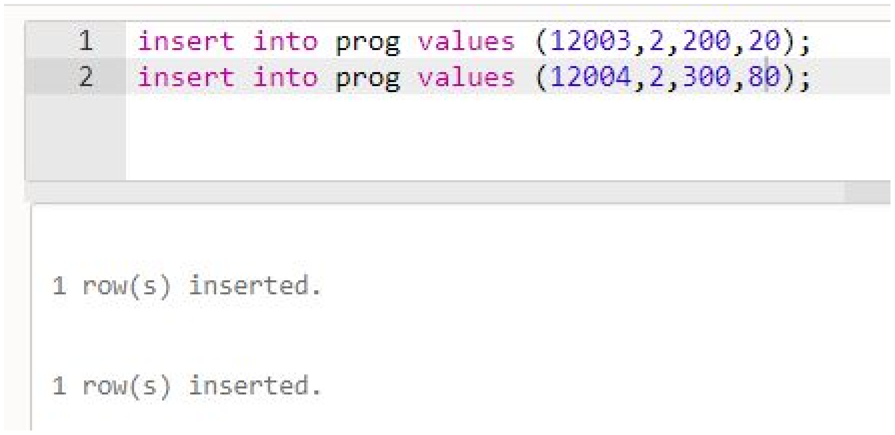
Teaches Table



```
insert into teaches values(12017,17001);
insert into teaches values(12017,17026);
```

```
1 row(s) inserted.

1 row(s) inserted.
```

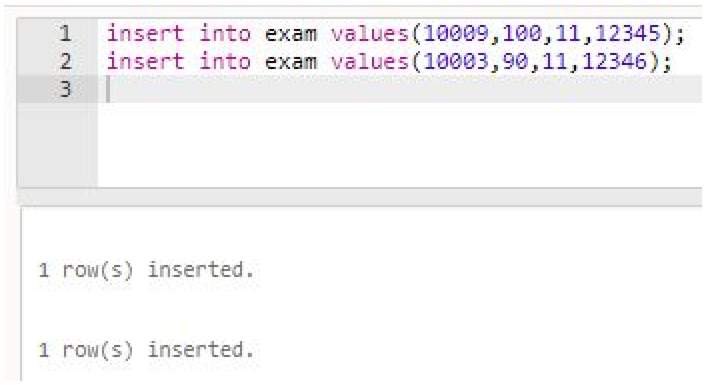
Prog Table



```
insert into prog values (12003,2,200,20);
insert into prog values (12004,2,300,80);
```

```
1 row(s) inserted.

1 row(s) inserted.
```

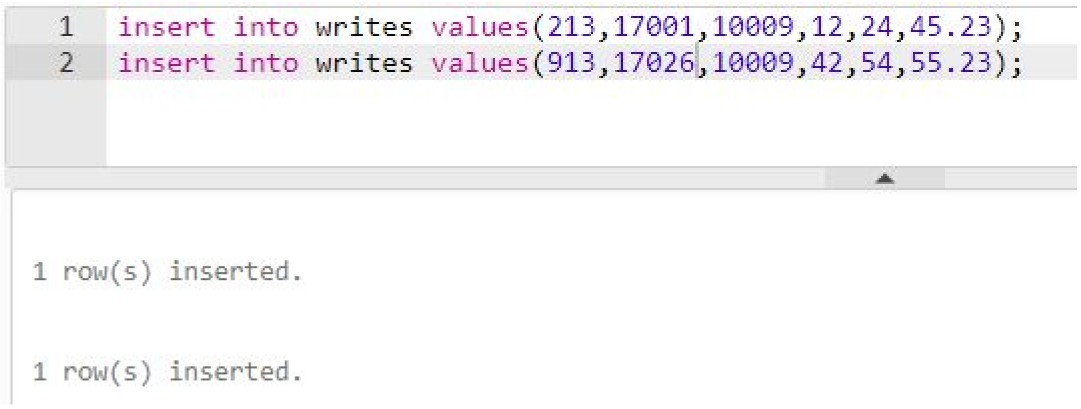
Exam Table



```
insert into exam values(10009,100,11,12345);
insert into exam values(10003,90,11,12346);
```

```
1 row(s) inserted.

1 row(s) inserted.
```

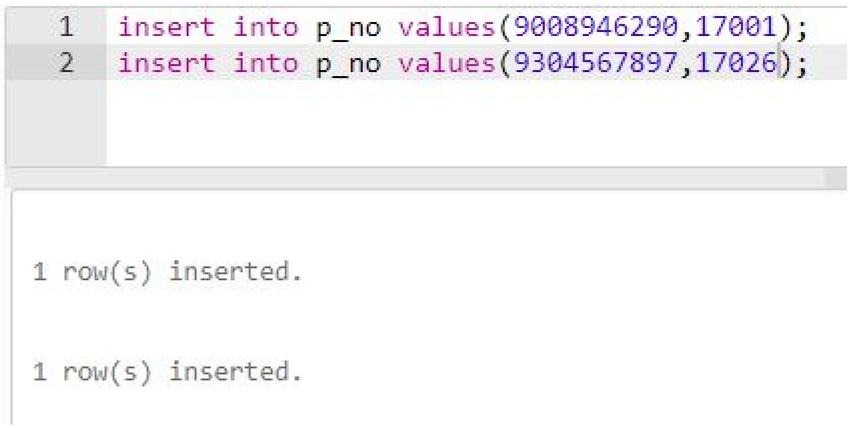
Writes Table



```
insert into writes values(213,17001,10009,12,24,45.23);
insert into writes values(913,17026,10009,42,54,55.23);
```

```
1 row(s) inserted.

1 row(s) inserted.
```

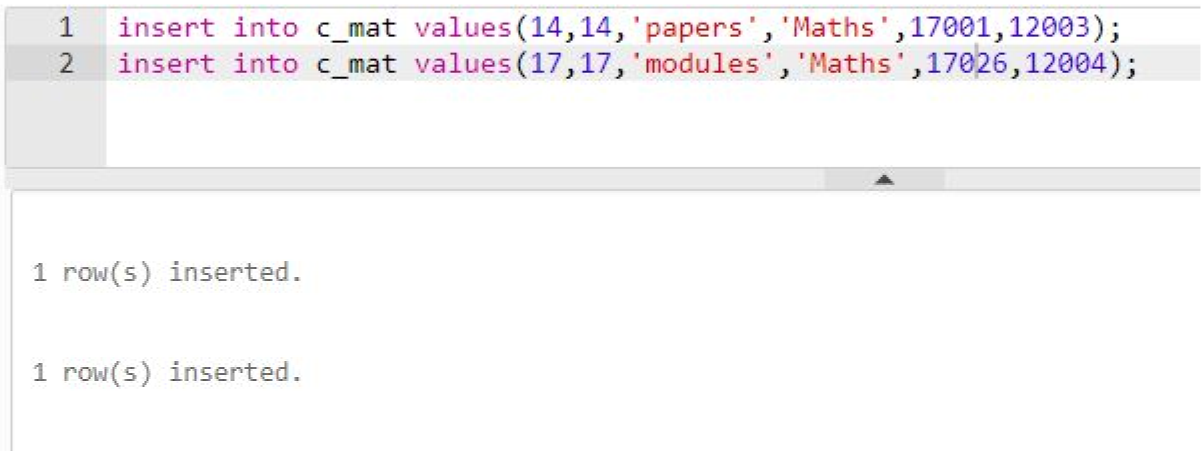
P_no Table



```
insert into p_no values(9008946290,17001);
insert into p_no values(9304567897,17026);
```

```
1 row(s) inserted.

1 row(s) inserted.
```

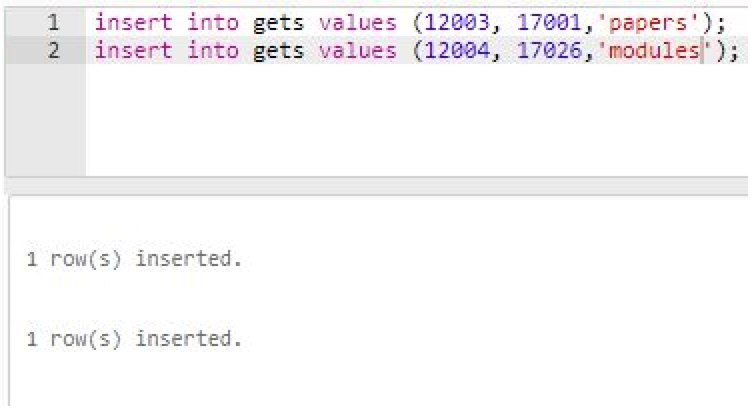
C_mat Table



```
insert into c_mat values(14,14,'papers','Maths',17001,12003);
insert into c_mat values(17,17,'modules','Maths',17026,12004);
```

```
1 row(s) inserted.

1 row(s) inserted.
```

Gets Table

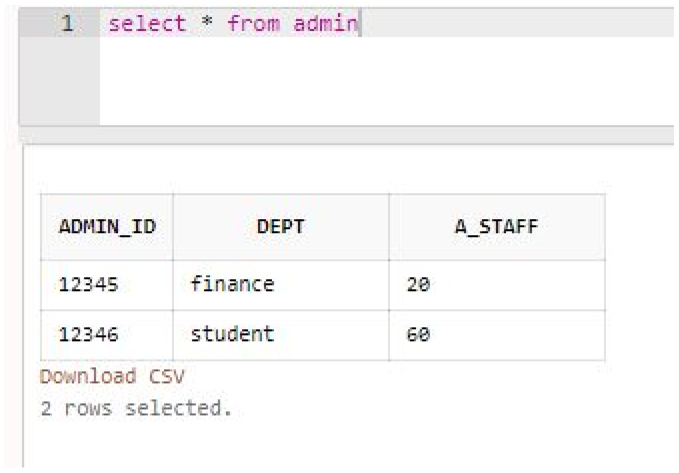


```
insert into gets values (12003, 17001,'papers');
insert into gets values (12004, 17026,'modules');
```

```
1 row(s) inserted.

1 row(s) inserted.
```

Display Content

Admin

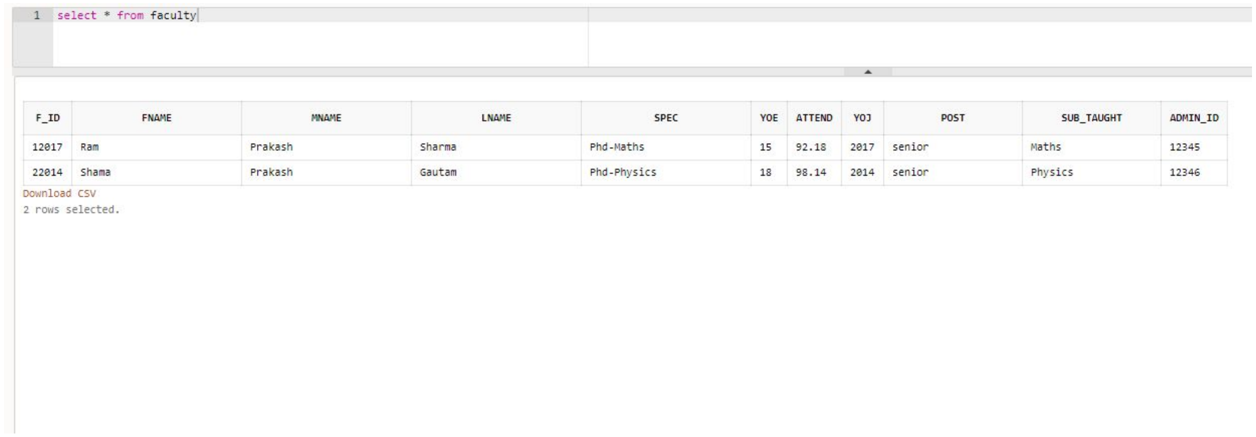


```
select * from admin
```

ADMIN_ID	DEPT	A_STAFF
12345	finance	20
12346	student	60

Download CSV
2 rows selected.

Faculty

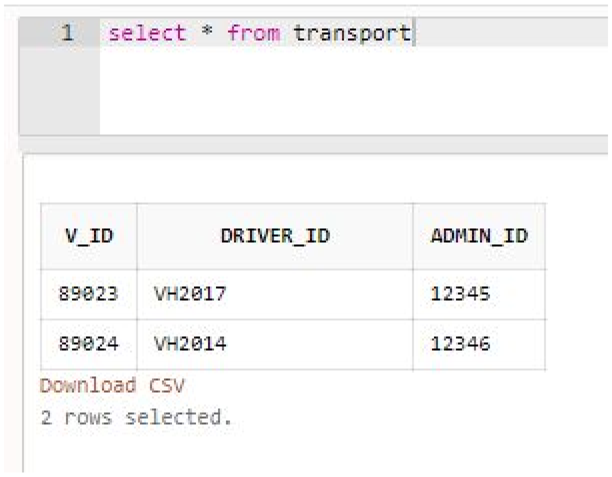


```
select * from faculty
```

F_ID	FNAME	MNAME	LNAME	SPEC	YOE	ATTEND	YOJ	POST	SUB_TAUGHT	ADMIN_ID
12017	Ram	Prakash	Sharma	Phd-Maths	15	92.18	2017	senior	Maths	12345
22014	Shama	Prakash	Gautam	Phd-Physics	18	98.14	2014	senior	Physics	12346

Download CSV
2 rows selected.

Transport

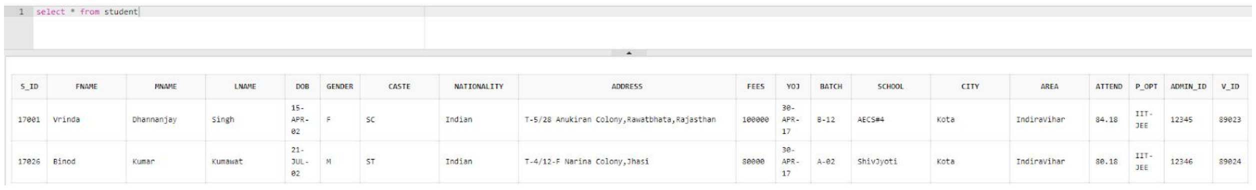


```
select * from transport
```

V_ID	DRIVER_ID	ADMIN_ID
89023	VH2017	12345
89024	VH2014	12346

Download CSV
2 rows selected.

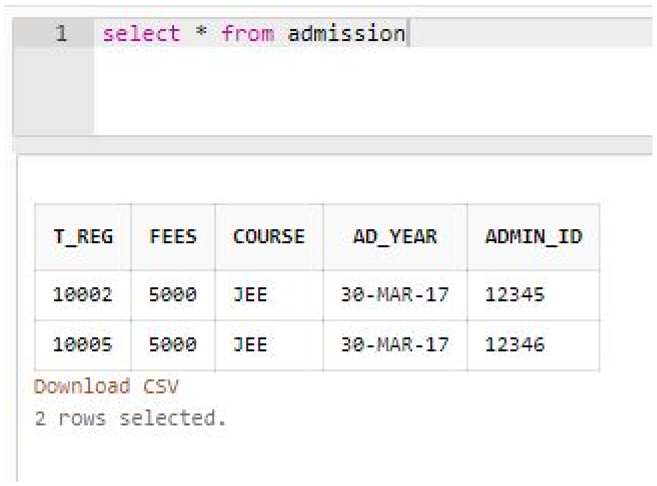
Student



```
select * from student
```

S_ID	FNAME	MNAME	LNAME	DOB	GENDER	CASTE	NATIONALITY	ADDRESS	FEES	YOJ	BATCH	SCHOOL	CITY	AREA	ATTEND	P_OPT	ADMIN_ID	V_ID
17001	Vrinda	Dhannanjay	Singh	15-APR-02	F	SC	Indian	T-5/28 Anukiran Colony,Rawatbhata,Rajasthan	100000	30-APR-17	B-12	AECS#4	Kota	IndiraVihar	84.18	IIT-JEE	12345	89023
17026	Binod	Kumar	Kumawat	21-JUL-02	M	ST	Indian	T-4/12-F Narina Colony,Jhasi	80000	30-APR-17	A-02	ShivJyoti	Kota	IndiraVihar	80.18	IIT-JEE	12346	89024

Admission



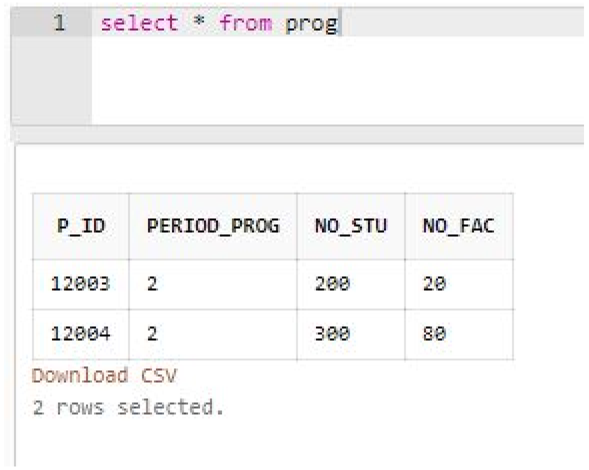
```
select * from admission
```

T_REG	FEES	COURSE	AD_YEAR	ADMIN_ID
10002	5000	JEE	30-MAR-17	12345
10005	5000	JEE	30-MAR-17	12346

Download CSV

2 rows selected.

Prog



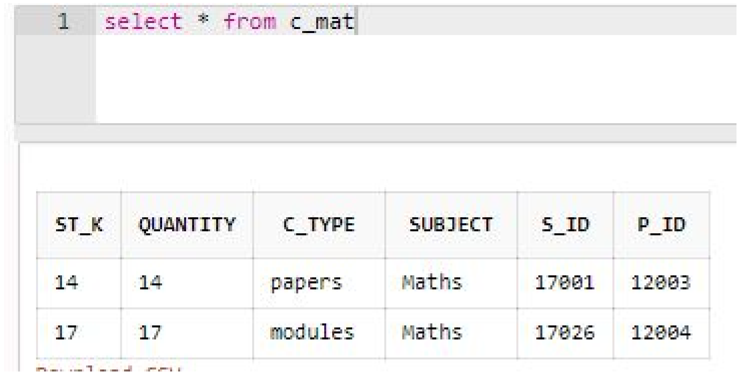
```
select * from prog
```

P_ID	PERIOD_PROG	NO_STU	NO_FAC
12003	2	200	20
12004	2	300	80

Download CSV

2 rows selected.

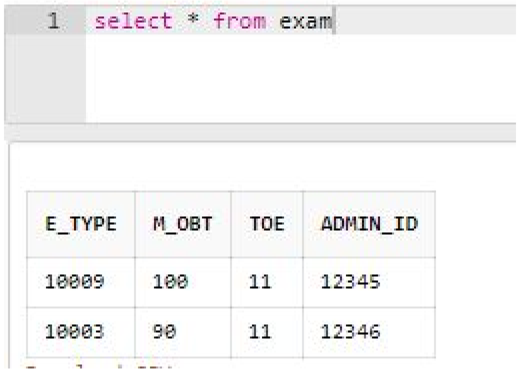
C_mat



```
select * from c_mat
```

ST_K	QUANTITY	C_TYPE	SUBJECT	S_ID	P_ID
14	14	papers	Maths	17001	12003
17	17	modules	Maths	17026	12004

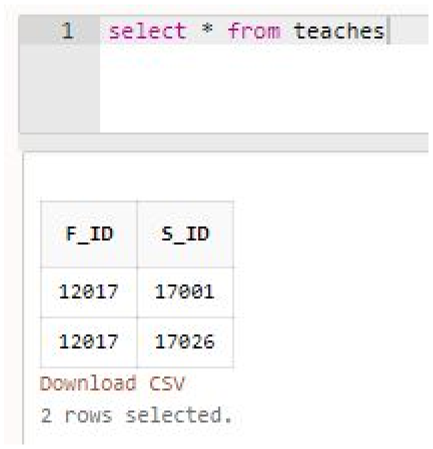
Exam



```
select * from exam
```

E_TYPE	M_OBT	TOE	ADMIN_ID
10009	100	11	12345
10003	90	11	12346

Teaches

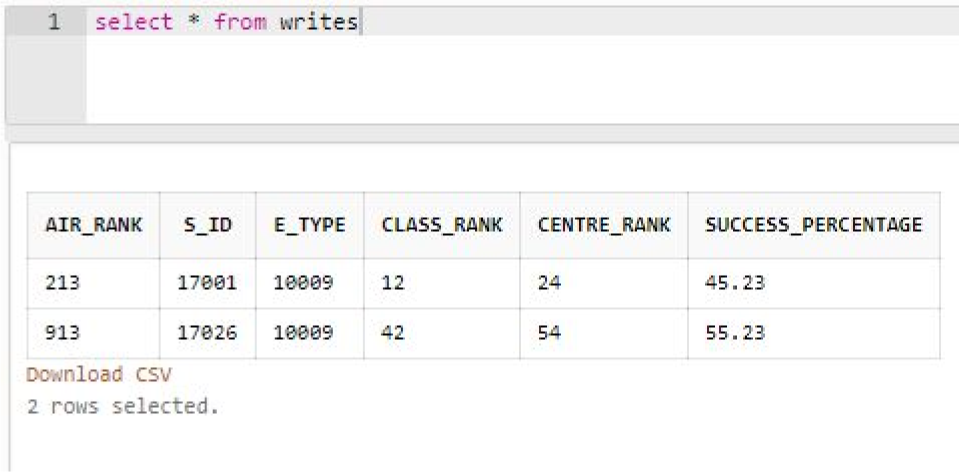


```
select * from teaches
```

F_ID	S_ID
12017	17001
12017	17026

Download CSV
2 rows selected.

Writes

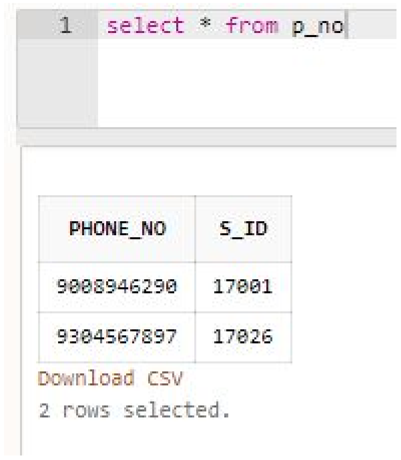


```
select * from writes
```

AIR_RANK	S_ID	E_TYPE	CLASS_RANK	CENTRE_RANK	SUCCESS_PERCENTAGE
213	17001	10009	12	24	45.23
913	17026	10009	42	54	55.23

Download CSV
2 rows selected.

p_no



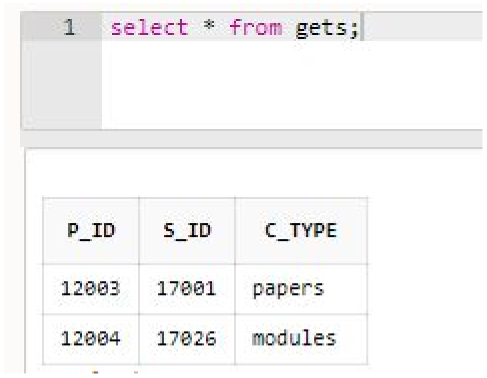
```
select * from p_no
```

PHONE_NO	S_ID
9008946290	17001
9304567897	17026

Download CSV

2 rows selected.

Gets



```
select * from gets;
```

P_ID	S_ID	C_TYPE
12003	17001	papers
12004	17026	modules

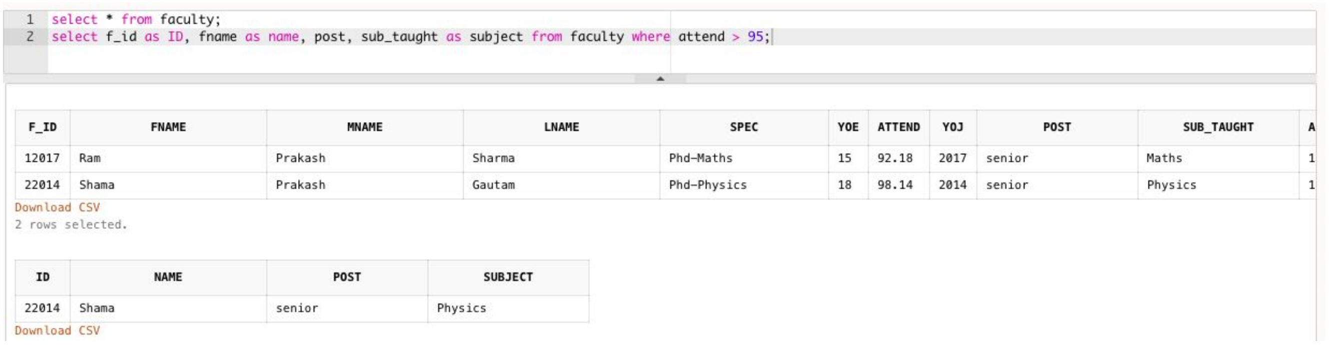
REVIEW - 3

SQL STATEMENTS FOR IMPLEMENTATION OF FUNCTIONAL REQUIREMENTS

1. SELECT

select * from faculty;

select f_id as ID, fname as name, post, sub_taught as subject from faculty where attend > 95;



```
select * from faculty;
select f_id as ID, fname as name, post, sub_taught as subject from faculty where attend > 95;
```

F_ID	FNAME	MNAME	LNAME	SPEC	YOE	ATTEND	YOJ	POST	SUB_TAUGHT	A
12017	Ram	Prakash	Sharma	Phd-Maths	15	92.18	2017	senior	Maths	1
22014	Shama	Prakash	Gautam	Phd-Physics	18	98.14	2014	senior	Physics	1

Download CSV
2 rows selected.

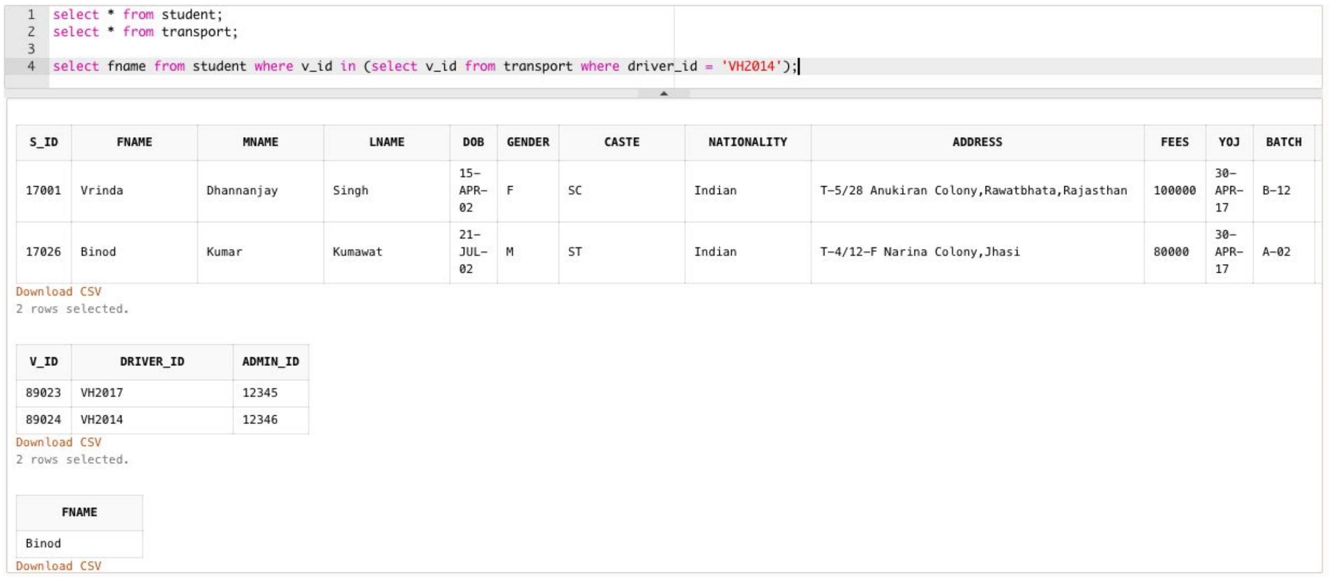
ID	NAME	POST	SUBJECT
22014	Shama	senior	Physics

Download CSV

select * from student;

select * from transport;

select fname from student where v_id in (select v_id from transport where driver_id = 'VH2014');



```
select * from student;
select * from transport;

select fname from student where v_id in (select v_id from transport where driver_id = 'VH2014');
```

S_ID	FNAME	MNAME	LNAME	DOB	GENDER	CASTE	NATIONALITY	ADDRESS	FEES	YOJ	BATCH
17001	Vrinda	Dhannanjay	Singh	15-APR-02	F	SC	Indian	T-5/28 Anukiran Colony,Rawatbhata,Rajasthan	100000	30-APR-17	B-12
17026	Binod	Kumar	Kumawat	21-JUL-02	M	ST	Indian	T-4/12-F Narina Colony,Jhasi	80000	30-APR-17	A-02

Download CSV
2 rows selected.

V_ID	DRIVER_ID	ADMIN_ID
89023	VH2017	12345
89024	VH2014	12346

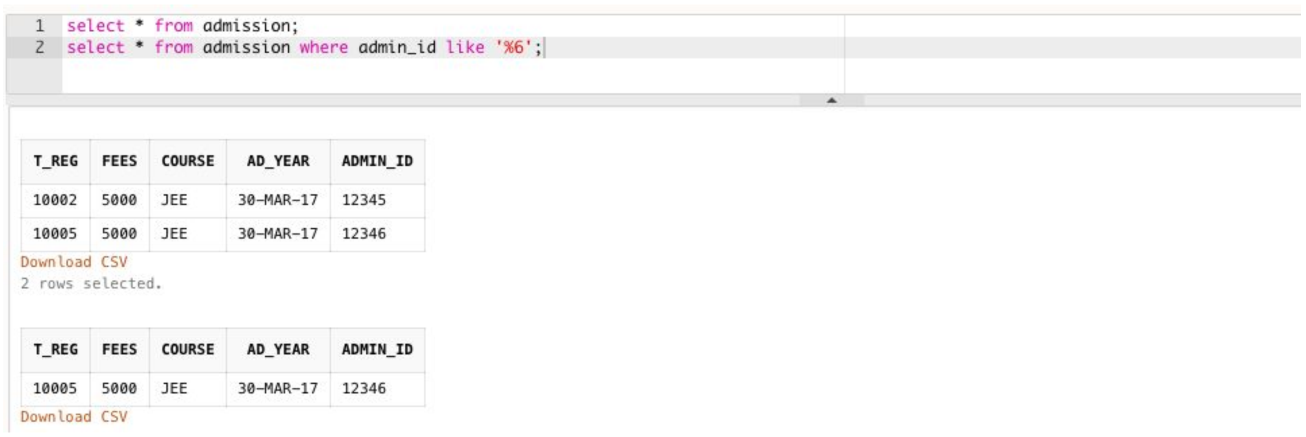
Download CSV
2 rows selected.

FNAME
Binod

Download CSV

select * from admission;

select * from admission where admin_id like '%6';



```
select * from admission;
select * from admission where admin_id like '%6';
```

T_REG	FEES	COURSE	AD_YEAR	ADMIN_ID
10002	5000	JEE	30-MAR-17	12345
10005	5000	JEE	30-MAR-17	12346

Download CSV
2 rows selected.

T_REG	FEES	COURSE	AD_YEAR	ADMIN_ID
10005	5000	JEE	30-MAR-17	12346

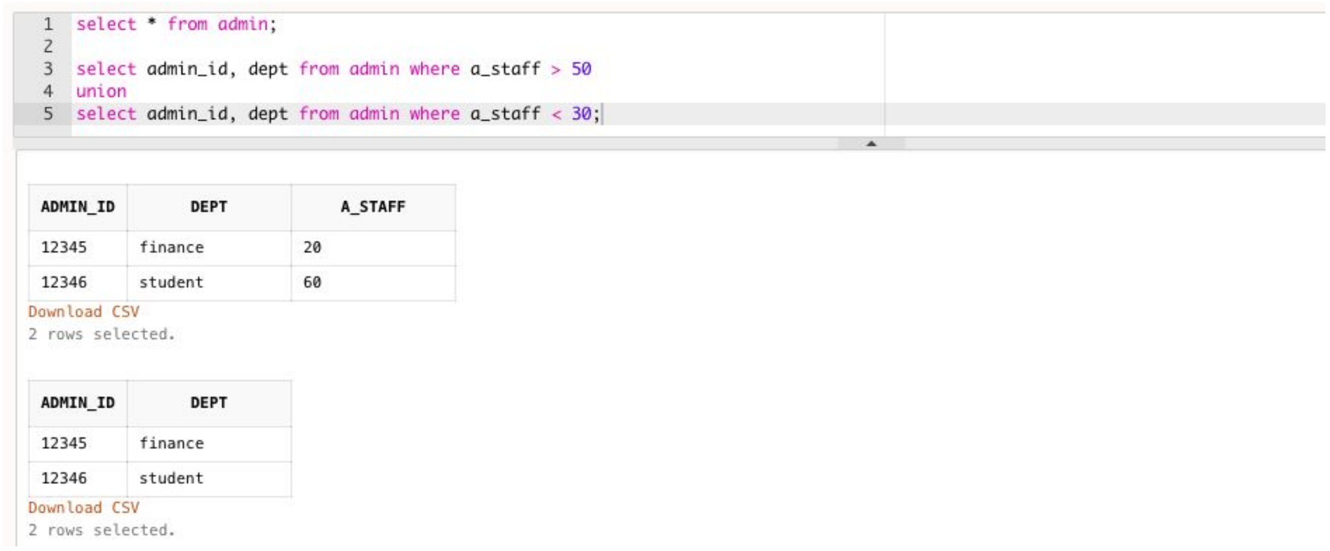
Download CSV

select * from admin;

select admin_id, dept from admin where a_staff > 50

union

select admin_id, dept from admin where a_staff < 30;



```
select * from admin;

select admin_id, dept from admin where a_staff > 50
union
select admin_id, dept from admin where a_staff < 30;
```

ADMIN_ID	DEPT	A_STAFF
12345	finance	20
12346	student	60

Download CSV
2 rows selected.

ADMIN_ID	DEPT
12345	finance
12346	student

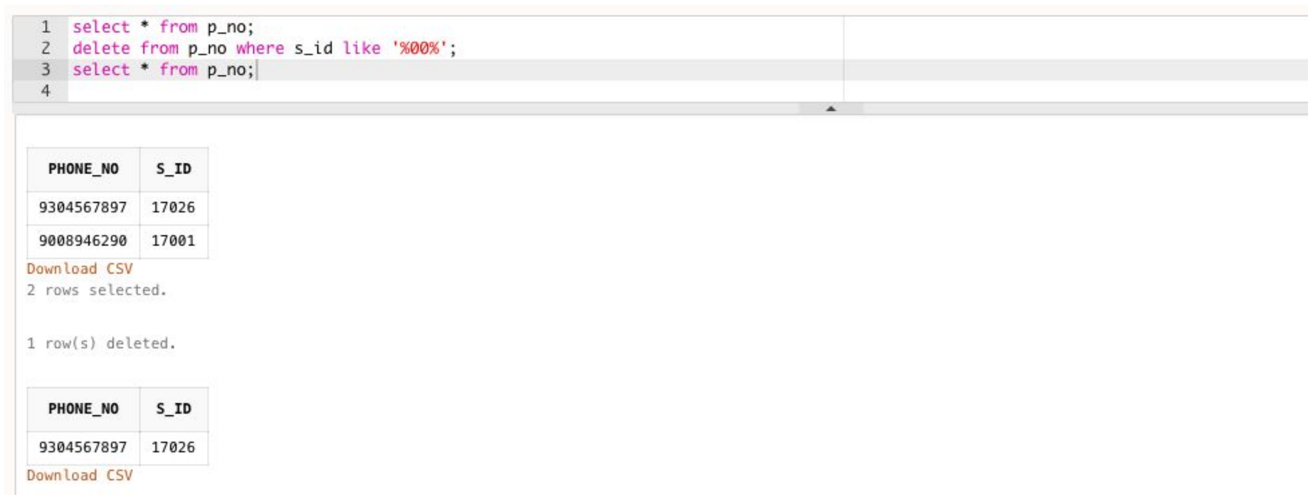
Download CSV
2 rows selected.

2. DELETE

select * from p_no;

delete from p_no where s_id like '%00%';

select * from p_no;



```
select * from p_no;
delete from p_no where s_id like '%00%';
select * from p_no;
```

PHONE_NO	S_ID
9304567897	17026
9008946290	17001

```
Download CSV
2 rows selected.

1 row(s) deleted.
```

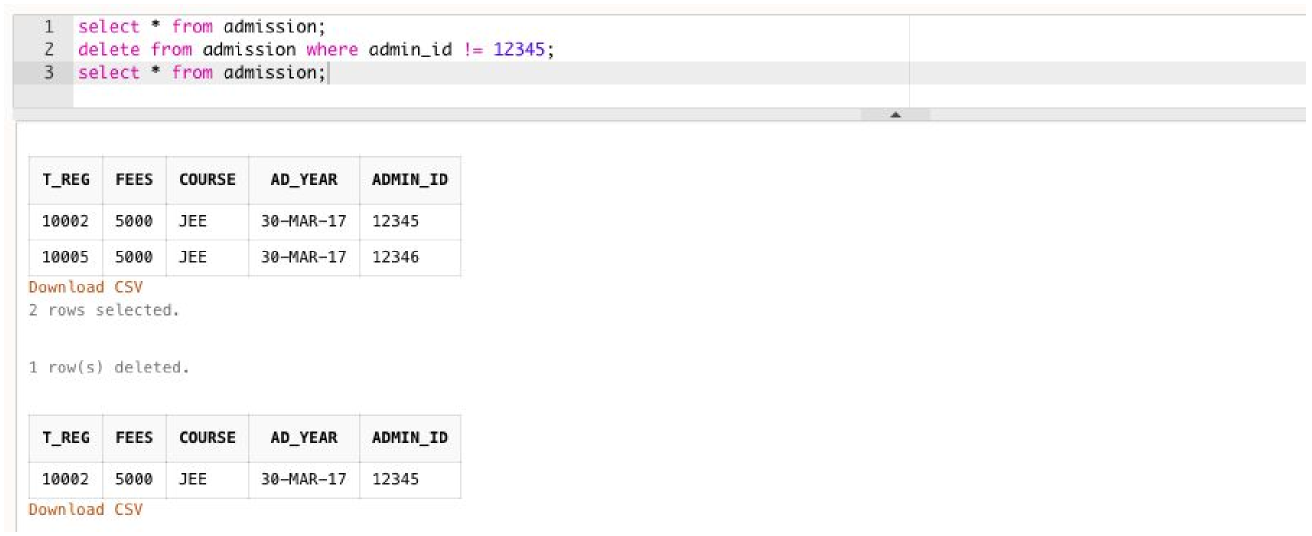
PHONE_NO	S_ID
9304567897	17026

```
Download CSV
```

select * from admission;

delete from admission where admin_id != 12345;

select * from admission;



```
select * from admission;
delete from admission where admin_id != 12345;
select * from admission;
```

T_REG	FEES	COURSE	AD_YEAR	ADMIN_ID
10002	5000	JEE	30-MAR-17	12345
10005	5000	JEE	30-MAR-17	12346

```
Download CSV
2 rows selected.

1 row(s) deleted.
```

T_REG	FEES	COURSE	AD_YEAR	ADMIN_ID
10002	5000	JEE	30-MAR-17	12345

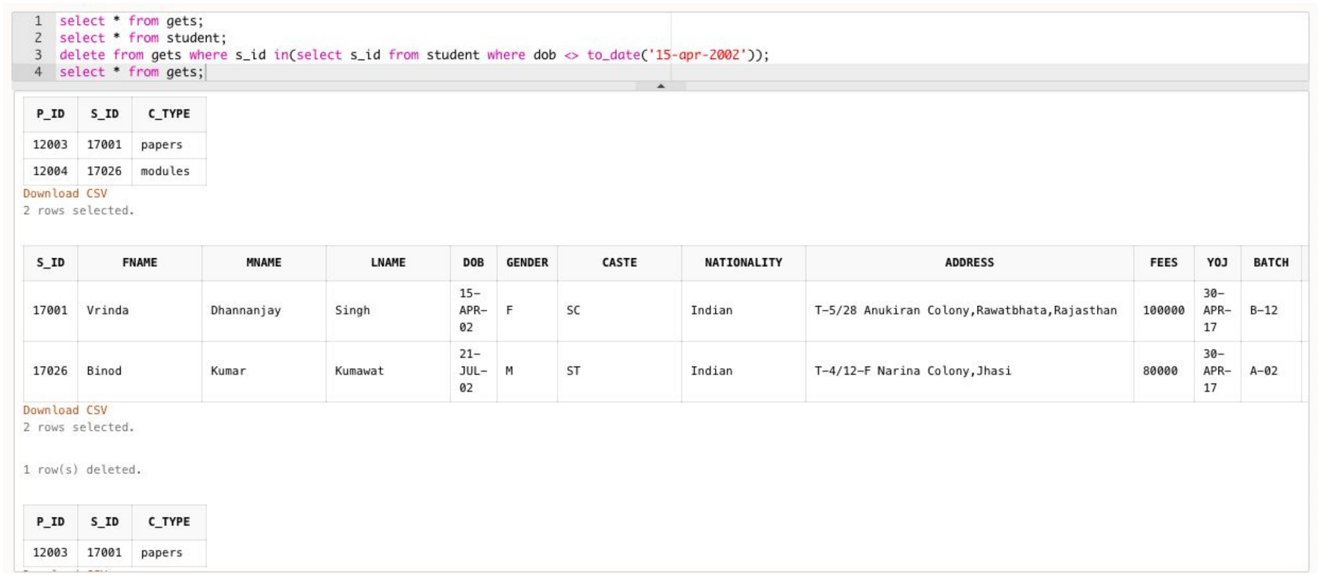
```
Download CSV
```

select * from gets;

select * from student;

delete from gets where s_id in(select s_id from student where dob <> to_date('15-apr-2002'));

select * from gets;



```
select * from gets;
select * from student;
delete from gets where s_id in(select s_id from student where dob <> to_date('15-apr-2002'));
select * from gets;
```

P_ID	S_ID	C_TYPE
12003	17001	papers
12004	17026	modules

Download CSV
2 rows selected.

S_ID	FNAME	MNAME	LNAME	DOB	GENDER	CASTE	NATIONALITY	ADDRESS	FEES	YOJ	BATCH
17001	Vrinda	Dhannanjay	Singh	15-APR-02	F	SC	Indian	T-5/28 Anukiran Colony,Rawatbhata,Rajasthan	100000	30-APR-17	B-12
17026	Binod	Kumar	Kumawat	21-JUL-02	M	ST	Indian	T-4/12-F Narina Colony,Jhasi	80000	30-APR-17	A-02

Download CSV
2 rows selected.

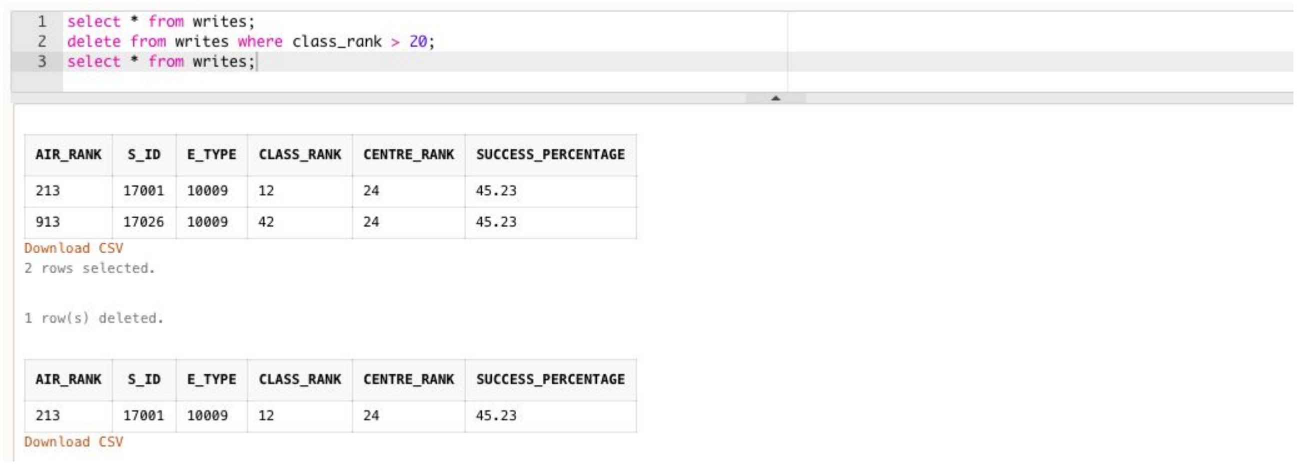
1 row(s) deleted.

P_ID	S_ID	C_TYPE
12003	17001	papers

select * from writes;

delete from writes where class_rank > 20;

select * from writes;



```
select * from writes;
delete from writes where class_rank > 20;
select * from writes;
```

AIR_RANK	S_ID	E_TYPE	CLASS_RANK	CENTRE_RANK	SUCCESS_PERCENTAGE
213	17001	10009	12	24	45.23
913	17026	10009	42	24	45.23

Download CSV
2 rows selected.

1 row(s) deleted.

AIR_RANK	S_ID	E_TYPE	CLASS_RANK	CENTRE_RANK	SUCCESS_PERCENTAGE
213	17001	10009	12	24	45.23

Download CSV

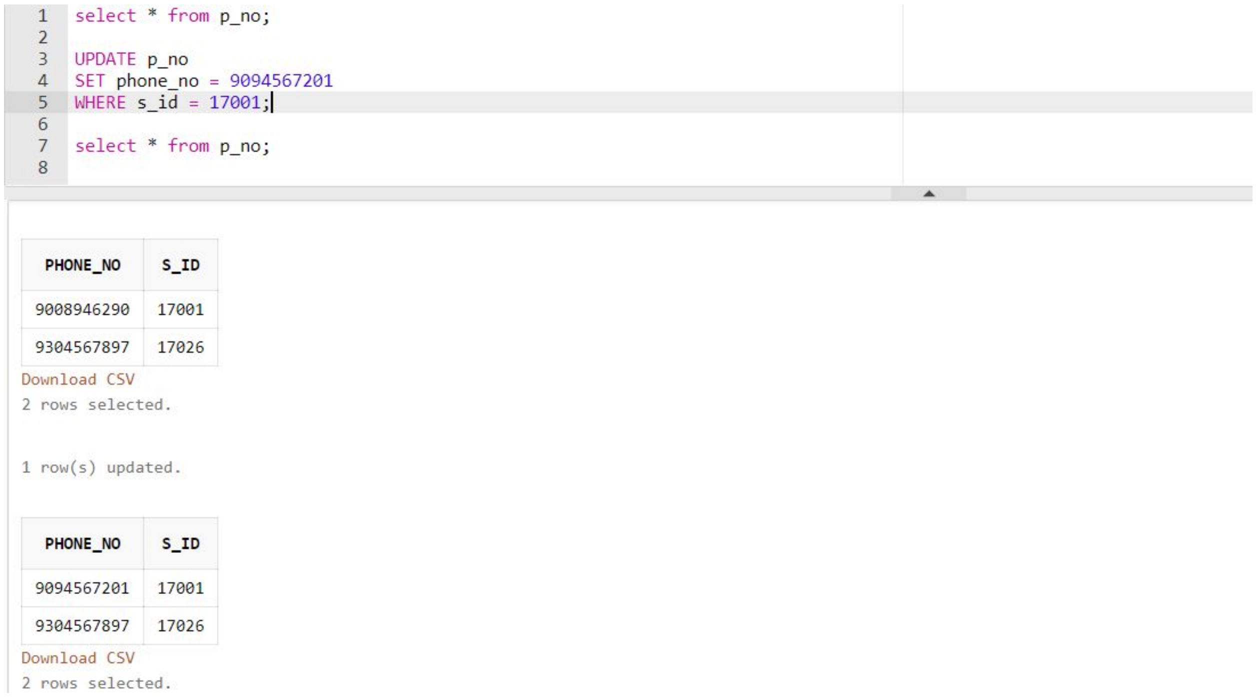
3. UPDATE

Query to update phone number of the student.

UPDATE p_no

SET phone_no = 9094567201

WHERE s_id = 17001;



```
select * from p_no;

UPDATE p_no
SET phone_no = 9094567201
WHERE s_id = 17001;

select * from p_no;
```

PHONE_NO	S_ID
9008946290	17001
9304567897	17026

```
Download CSV
2 rows selected.


1 row(s) updated.
```

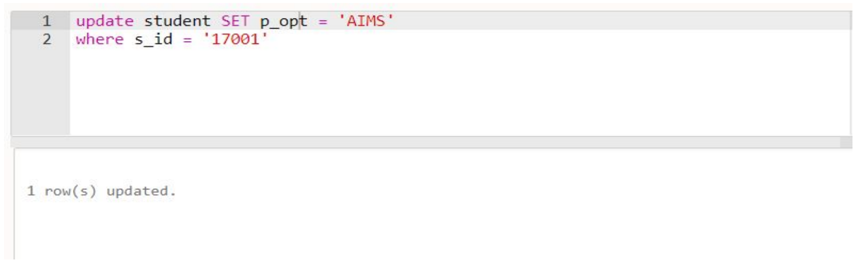
PHONE_NO	S_ID
9094567201	17001
9304567897	17026

```
Download CSV
2 rows selected.
```

Update course student opted if he/she wishes to change it.

update student SET p_opt = 'AIMS'

where s_id = '17001'

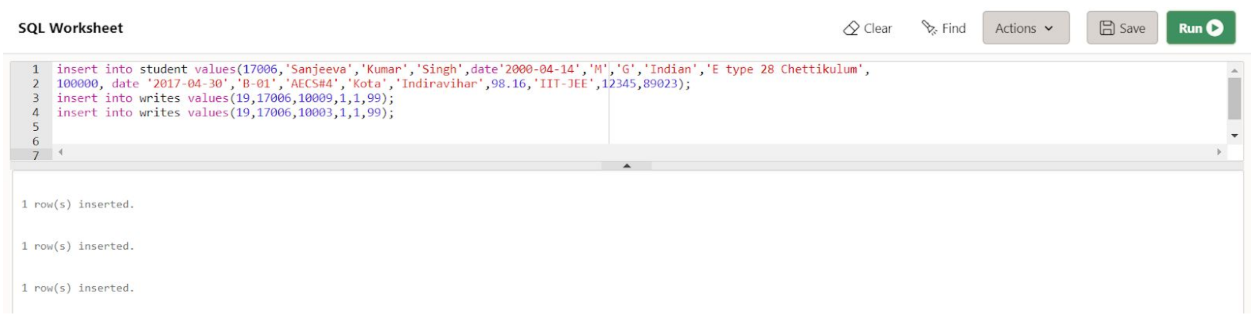


```
update student SET p_opt = 'AIMS'
where s_id = '17001'
```

```
1 row(s) updated.
```

Update student batch to A_01 if the success percentage of the student is greater than 90% in equal to or more than two exams.

Insertion :-



```
insert into student values(17006,'Sanjeeva','Kumar','Singh',date'2000-04-14','M','G','Indian','E type 28 Chettikulum',
100000, date '2017-04-30','B-01','AECS#4','Kota','Indiravihar',98.16,'IIT-JEE',12345,89023);
insert into writes values(19,17006,10009,1,1,99);
insert into writes values(19,17006,10003,1,1,99);
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

Updation:-

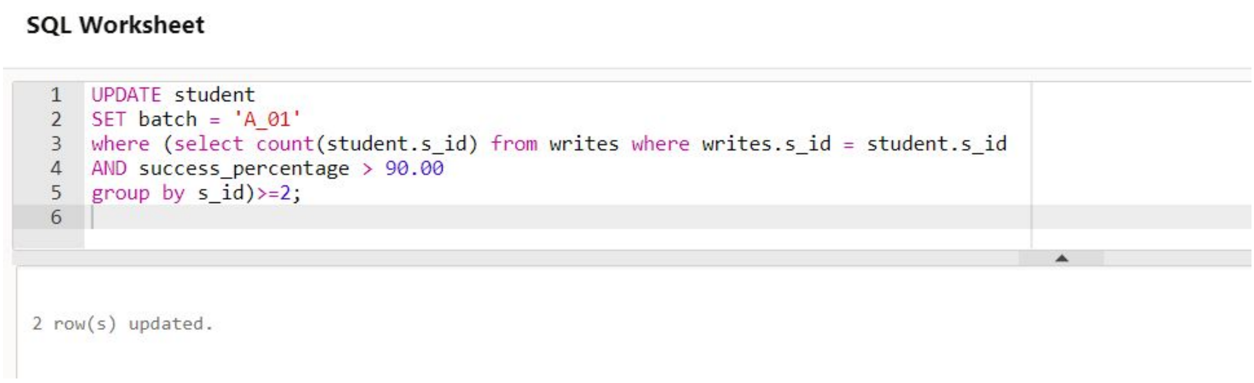
UPDATE student

SET batch = 'A_01'

where (select count(student.s_id) from writes where writes.s_id = student.s_id

AND success_percentage > 90.00

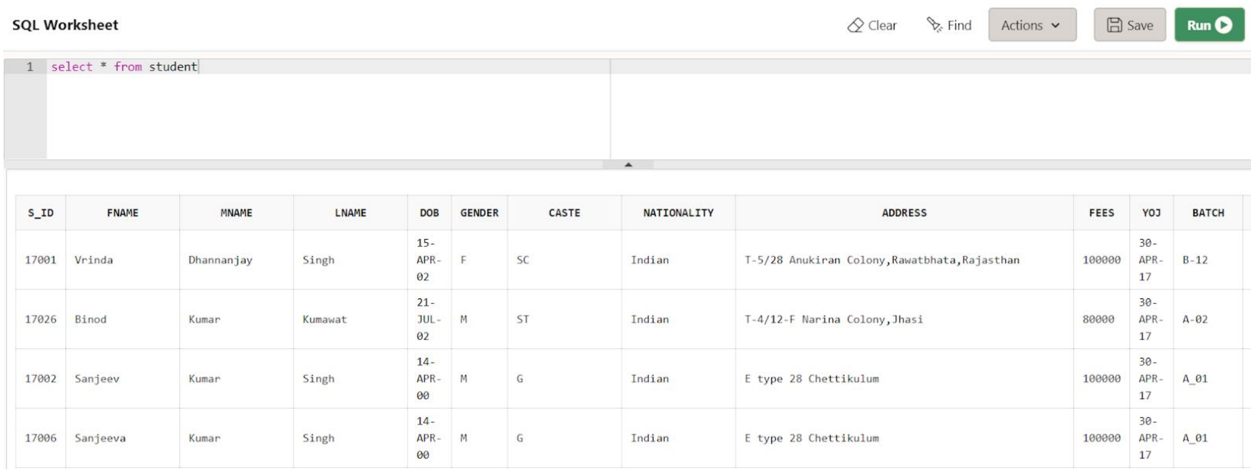
group by s_id)>=2;



```
UPDATE student
SET batch = 'A_01'
where (select count(student.s_id) from writes where writes.s_id = student.s_id
AND success_percentage > 90.00
group by s_id)>=2;
```

```
2 row(s) updated.
```

Showing result:-

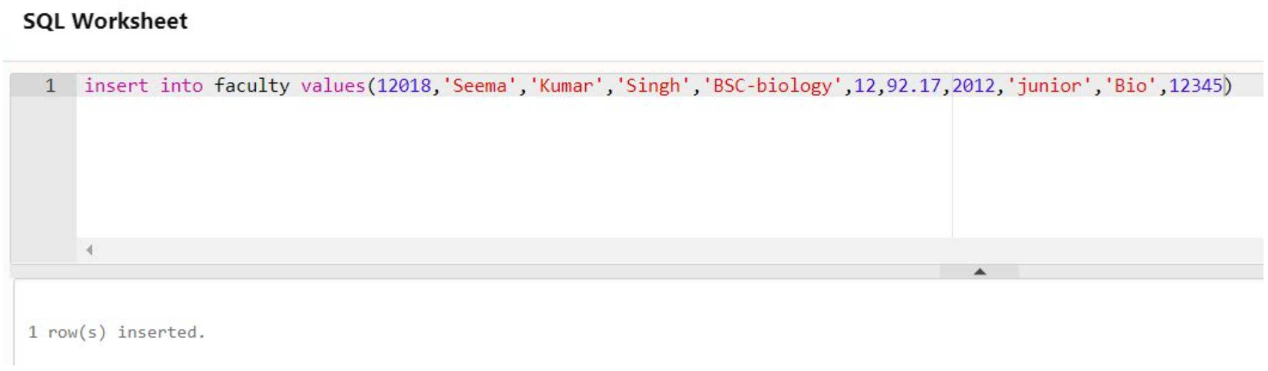


```sql
select * from student
```

S_ID	FNAME	MNAME	LNAME	DOB	GENDER	CASTE	NATIONALITY	ADDRESS	FEES	YOJ	BATCH
17001	Vrinda	Dhannanjay	Singh	15-APR-02	F	SC	Indian	T-5/28 Anukiran Colony,Rawatbhata,Rajasthan	100000	30-APR-17	B-12
17026	Binod	Kumar	Kumawat	21-JUL-02	M	ST	Indian	T-4/12-F Narina Colony,Jhasi	80000	30-APR-17	A-02
17002	Sanjeev	Kumar	Singh	14-APR-00	M	G	Indian	E type 28 Chettikulum	100000	30-APR-17	A_01
17006	Sanjeeva	Kumar	Singh	14-APR-00	M	G	Indian	E type 28 Chettikulum	100000	30-APR-17	A_01

Update post of the faculty to senior if the year of experience of teaching is greater than 10 .

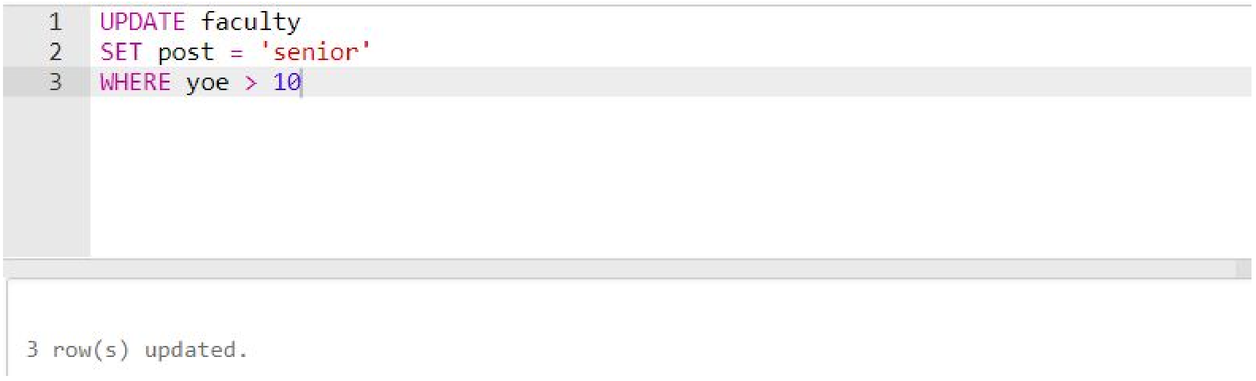
Query Insertion :-



```sql
insert into faculty values(12018,'Seema','Kumar','Singh','BSC-biology',12,92.17,2012,'junior','Bio',12345)
```

```
1 row(s) inserted.
```

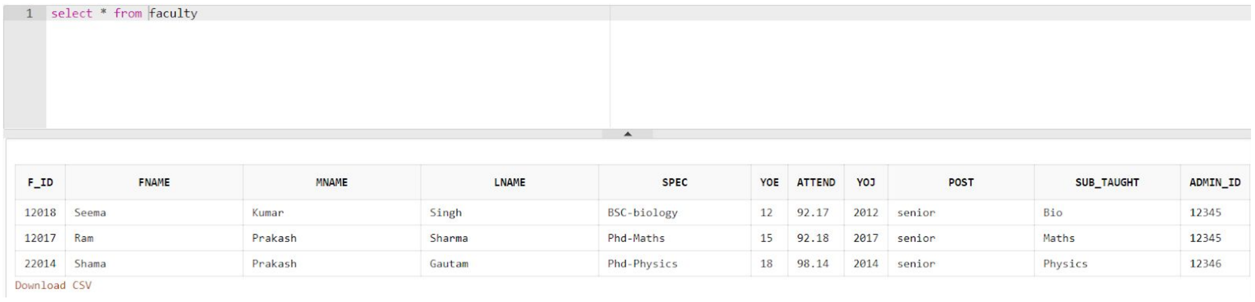
Updation:-



```sql
UPDATE faculty
SET post = 'senior'
WHERE yoe > 10
```

```
3 row(s) updated.
```

Showing Result:-



```
select * from faculty
```

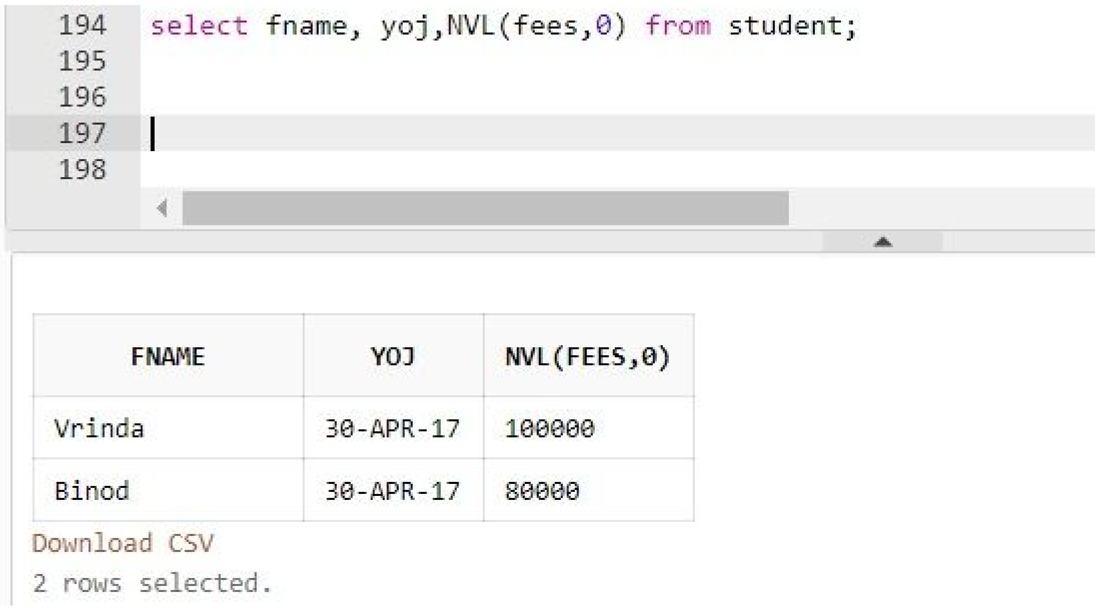
F_ID	FNAME	MNAME	LNAME	SPEC	YOE	ATTEND	YOJ	POST	SUB_TAUGHT	ADMIN_ID
12018	Seema	Kumar	Singh	BSC-biology	12	92.17	2012	senior	Bio	12345
12017	Ram	Prakash	Sharma	Phd-Maths	15	92.18	2017	senior	Maths	12345
22014	Shama	Prakash	Gautam	Phd-Physics	18	98.14	2014	senior	Physics	12346

Download CSV

4. NVL

Show name and student admission details(fees, yoj, batch) for example if students 100% scholarship, his/her fees should be zero as compared to null.

select fname, yoj,NVL(fees,0) from student;



```
select fname, yoj,NVL(fees,0) from student;
```

FNAME	YOJ	NVL(FEES,0)
Vrinda	30-APR-17	100000
Binod	30-APR-17	80000

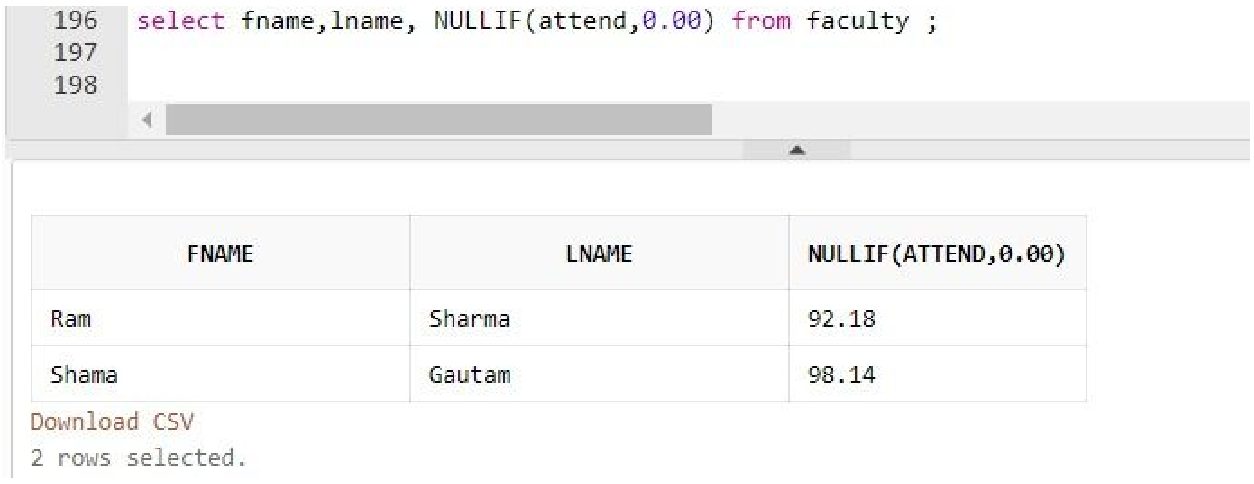
Download CSV

```
2 rows selected.
```

5. NULLIF

Show name and attendance of faculty and show attendance as null if attendance is zero.

select fname,lname, NULLIF(attend,0.00) from faculty ;



```
select fname,lname, NULLIF(attend,0.00) from faculty ;
```

FNAME	LNAME	NULLIF(ATTEND,0.00)
Ram	Sharma	92.18
Shama	Gautam	98.14

Download CSV
2 rows selected.

6. JOIN

Show name of faculty and student id.

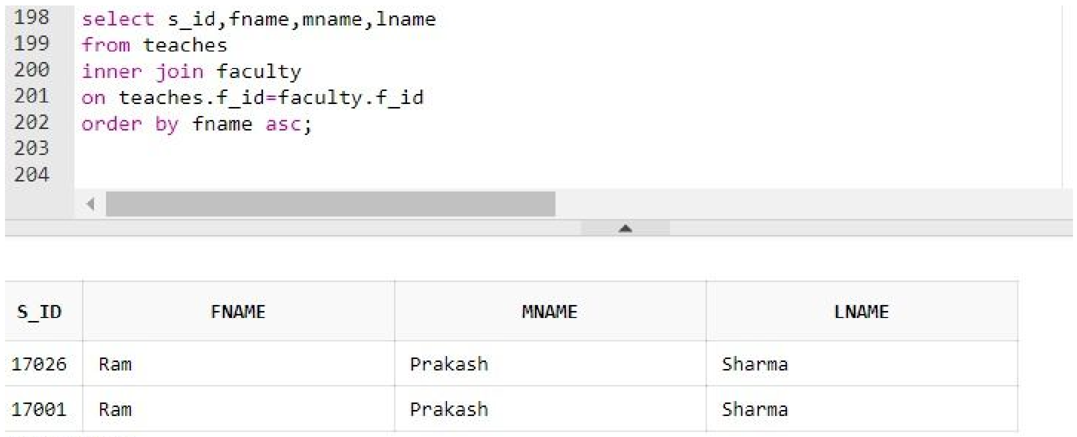
select s_id,fname,mname,lname

from teaches

inner join faculty

on teaches.f_id=faculty.f_id

order by fname asc;



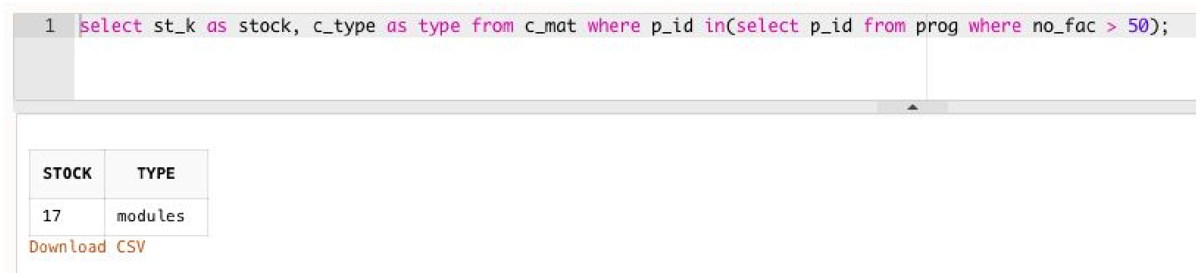
```
select s_id,fname,mname,lname
from teaches
inner join faculty
on teaches.f_id=faculty.f_id
order by fname asc;
```

S_ID	FNAME	MNAME	LNAME
17026	Ram	Prakash	Sharma
17001	Ram	Prakash	Sharma

7. UNCORRELATED NESTED

Display the stock and its type if the number of faculties assigned to the corresponding program is greater than 50.

select st_k as stock, c_type as type from c_mat where p_id in(select p_id from prog where no_fac > 50);



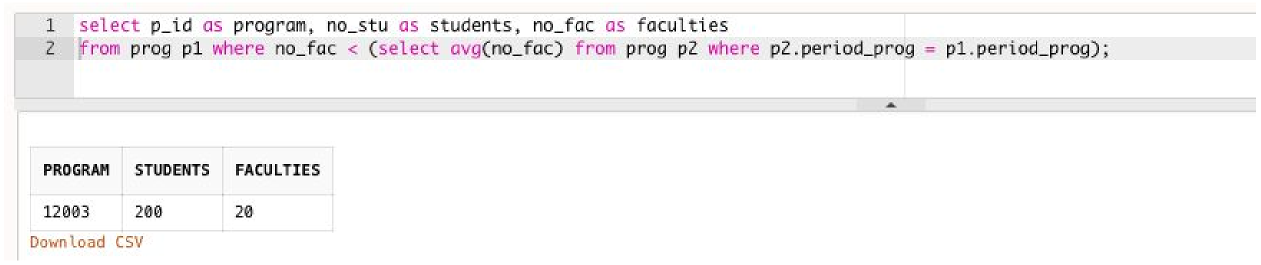
```
select st_k as stock, c_type as type from c_mat where p_id in(select p_id from prog where no_fac > 50);
```

STOCK	TYPE
17	modules

Download CSV

8. CORRELATED NESTED

select p_id as program, no_stu as students, no_fac as faculties from prog p1 where no_fac < (select avg(no_fac) from prog p2 where p2.period_prog = p1.period_prog);



```
select p_id as program, no_stu as students, no_fac as faculties
from prog p1 where no_fac < (select avg(no_fac) from prog p2 where p2.period_prog = p1.period_prog);
```

PROGRAM	STUDENTS	FACULTIES
12003	200	20

Download CSV

9. SET

Show name and specialization of the teacher whose specialization is not physics.

select fname,mname,lname,spec

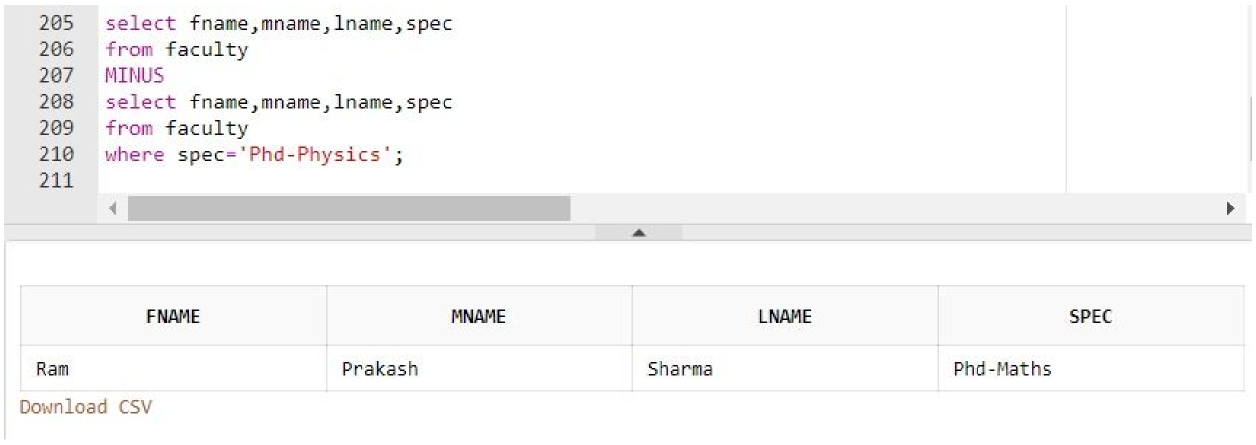
from faculty

MINUS

select fname,mname,lname,spec

from faculty

where spec='Phd-Physics';



```
select fname,mname,lname,spec
from faculty
MINUS
select fname,mname,lname,spec
from faculty
where spec='Phd-Physics';
```

FNAME	MNAME	LNAME	SPEC
Ram	Prakash	Sharma	Phd-Maths

Download CSV

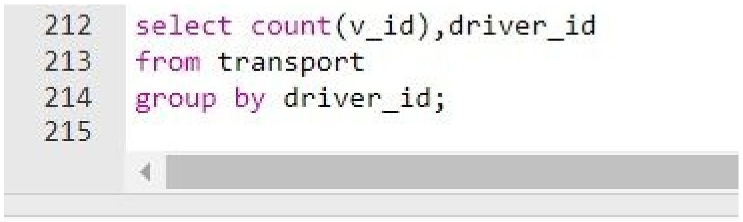
10. GROUP BY

show total no of vehicle and group them by driver_id

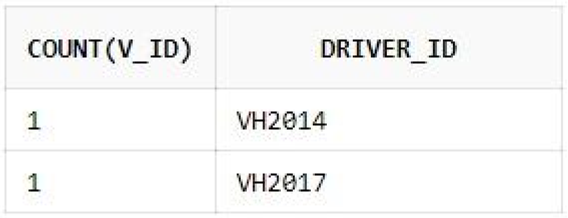
select count(v_id),driver_id

from transport

group by driver_id;



```
select count(v_id),driver_id
from transport
group by driver_id;
```



COUNT(V_ID)	DRIVER_ID
1	VH2014
1	VH2017

11. GROUP BY AND WHERE

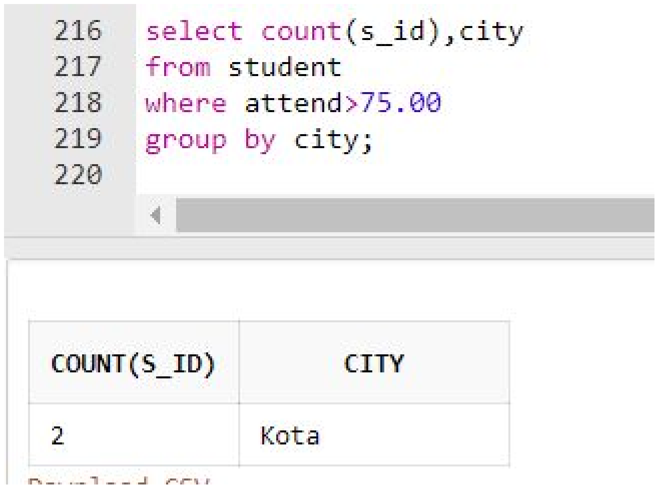
show number of students per city which have attendance greater than 75%

select count(s_id),city

from student

where attend>75.00

group by city;



```
select count(s_id),city
from student
where attend>75.00
group by city;
```

COUNT(S_ID)	CITY
2	Kota

12. GROUP BY AND HAVING

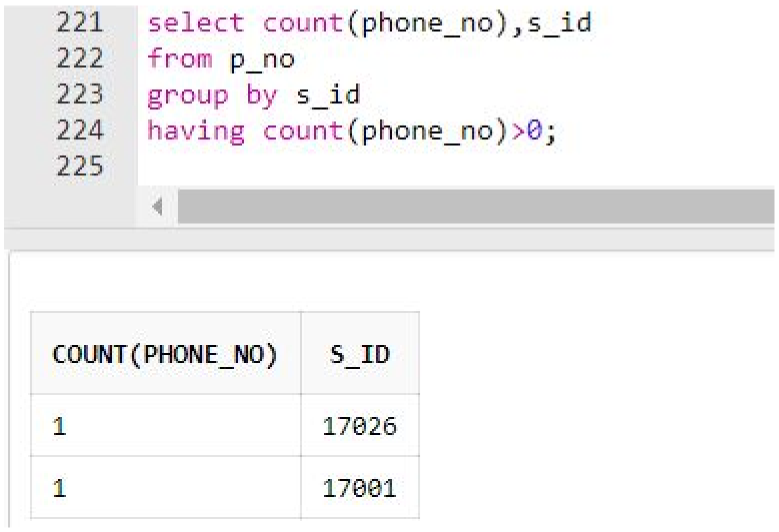
show student's id who have more than two phone no

select count(phone_no),s_id

from p_no

group by s_id

having count(phone_no)>0;



```
select count(phone_no),s_id
from p_no
group by s_id
having count(phone_no)>0;
```

COUNT(PHONE_NO)	S_ID
1	17026
1	17001

13. OUTER JOIN

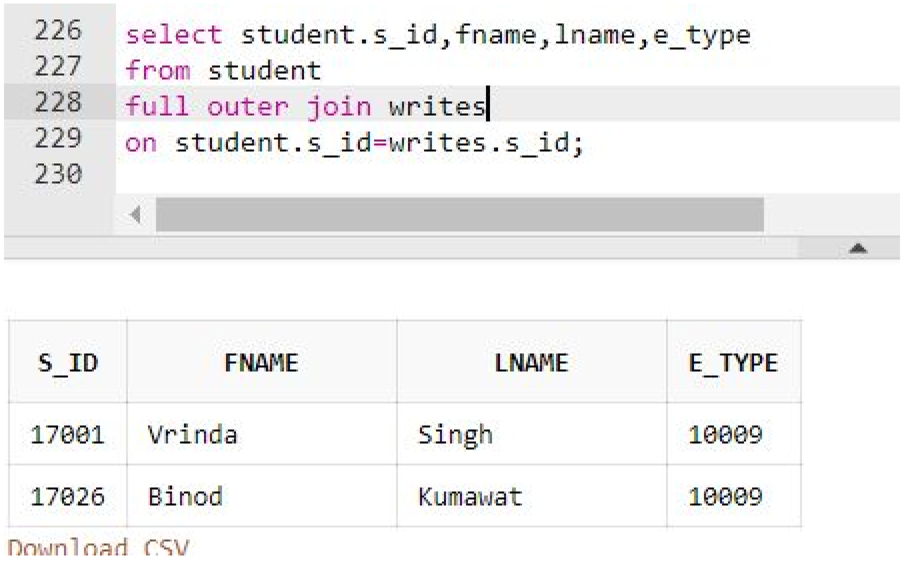
show all student names,id and the exam they write

select student.s_id,fname,lname,e_type

from student

full outer join writes

on student.s_id=writes.s_id;



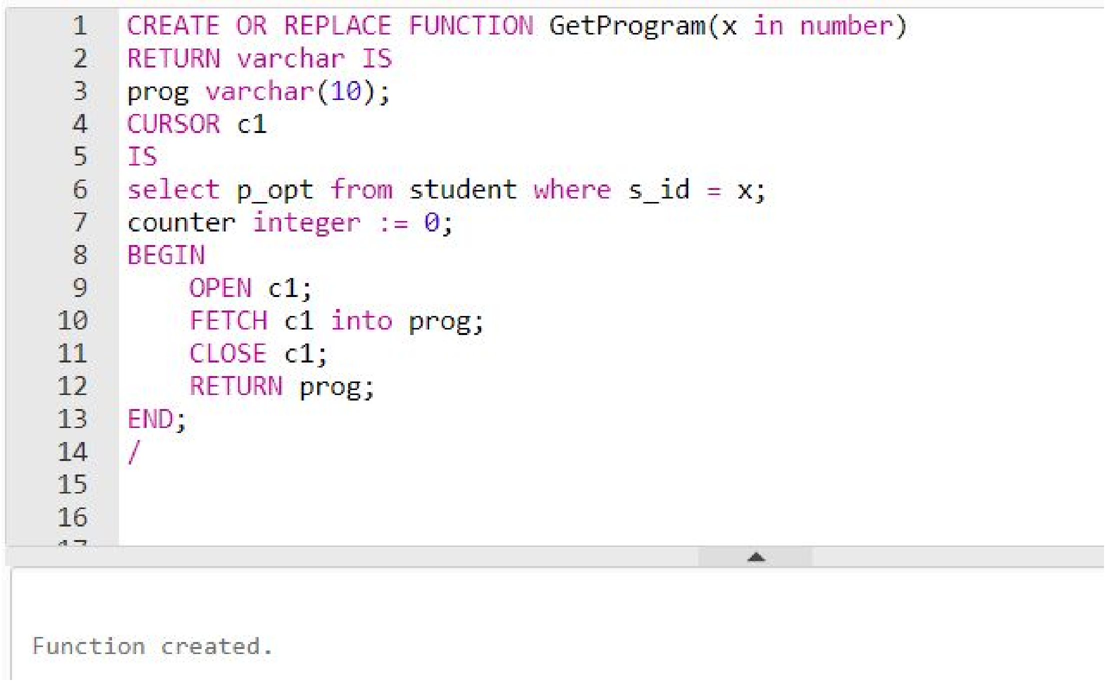
```
select student.s_id,fname,lname,e_type
from student
full outer join writes
on student.s_id=writes.s_id;
```

S_ID	FNAME	LNAME	E_TYPE
17001	Vrinda	Singh	10009
17026	Binod	Kumawat	10009

Download CSV

PL/SQL FUNCTION INVOLVING CURSOR

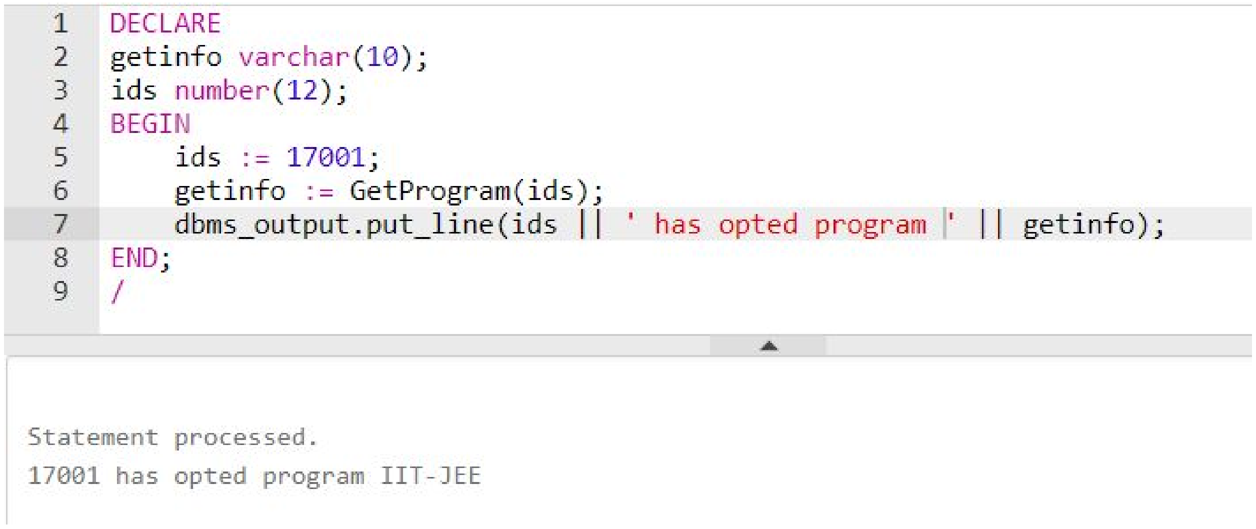
1. **The function to display the program opted by the student when the student ID is passed through it.**



```
CREATE OR REPLACE FUNCTION GetProgram(x in number)
RETURN varchar IS
prog varchar(10);
CURSOR c1
IS
select p_opt from student where s_id = x;
counter integer := 0;
BEGIN
    OPEN c1;
    FETCH c1 into prog;
    CLOSE c1;
    RETURN prog;
END;
/
```

```
Function created.
```

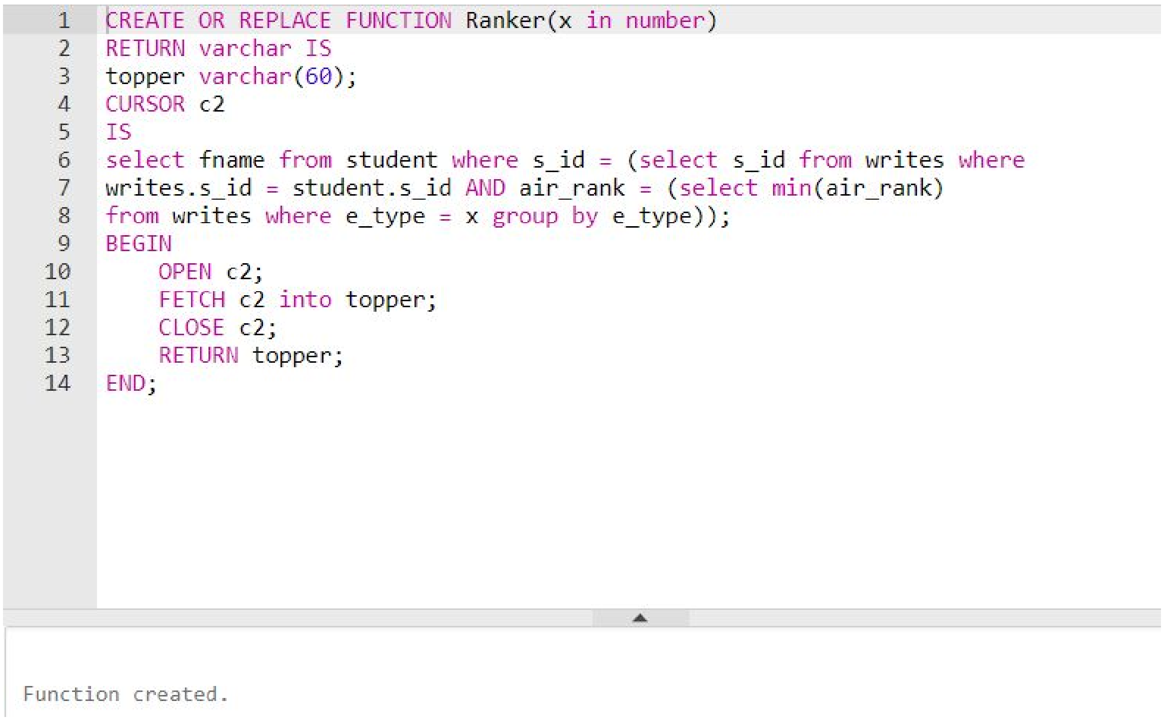
EXECUTING FUNCTION:-



```
DECLARE
getinfo varchar(10);
ids number(12);
BEGIN
    ids := 17001;
    getinfo := GetProgram(ids);
    dbms_output.put_line(ids || ' has opted program ' || getinfo);
END;
/
```

```
Statement processed.
17001 has opted program IIT-JEE
```

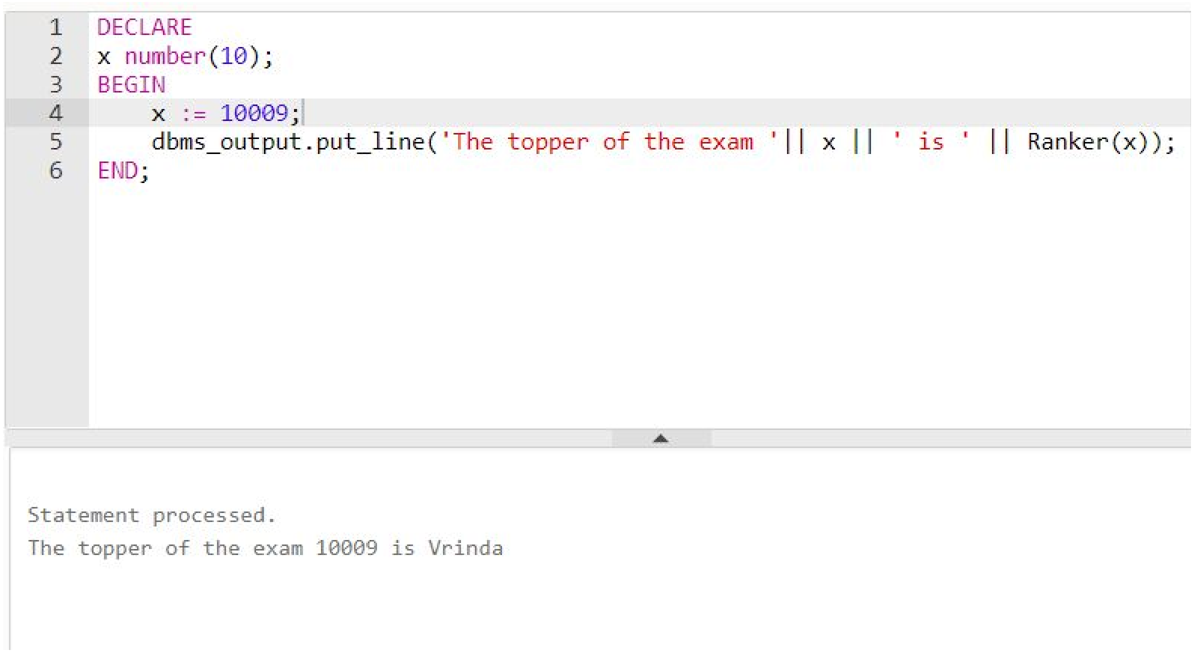
2. Function to display the name of the student who secured highest marks in the particular exam.



```
CREATE OR REPLACE FUNCTION Ranker(x in number)
RETURN varchar IS
topper varchar(60);
CURSOR c2
IS
select fname from student where s_id = (select s_id from writes where
writes.s_id = student.s_id AND air_rank = (select min(air_rank)
from writes where e_type = x group by e_type));
BEGIN
    OPEN c2;
    FETCH c2 into topper;
    CLOSE c2;
    RETURN topper;
END;
```

```
Function created.
```

EXECUTING:-



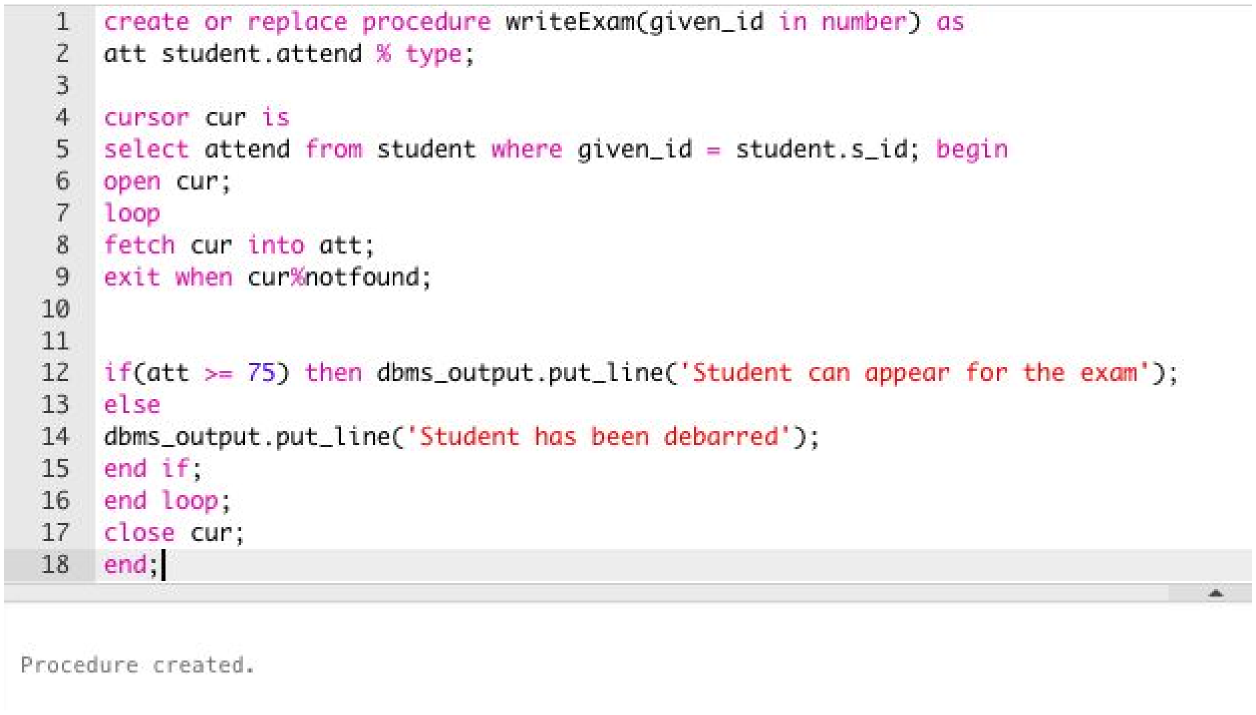
```
DECLARE
x number(10);
BEGIN
    x := 10009;
    dbms_output.put_line('The topper of the exam '|| x || ' is ' || Ranker(x));
END;
```

```
Statement processed.
The topper of the exam 10009 is Vrinda
```

PL/SQL PROCEDURE INVOLVING CURSOR

1. **Procedure to display whether a student is eligible to write his exam (attendance > 75%).**

```
create or replace procedure writeExam(given_id in number) as
att student.attend % type;
cursor cur is
select attend from student where given_id = student.s_id; begin
open cur;
loop
fetch cur into att;
exit when cur%notfound;
if(att >= 75) then dbms_output.put_line('Student can appear for the exam');
else
dbms_output.put_line('Student has been debarred');
end if;
end loop;
close cur;
end;
```



```
create or replace procedure writeExam(given_id in number) as
att student.attend % type;

cursor cur is
select attend from student where given_id = student.s_id; begin
open cur;
loop
fetch cur into att;
exit when cur%notfound;


if(att >= 75) then dbms_output.put_line('Student can appear for the exam');
else
dbms_output.put_line('Student has been debarred');
end if;
end loop;
close cur;
end;
```

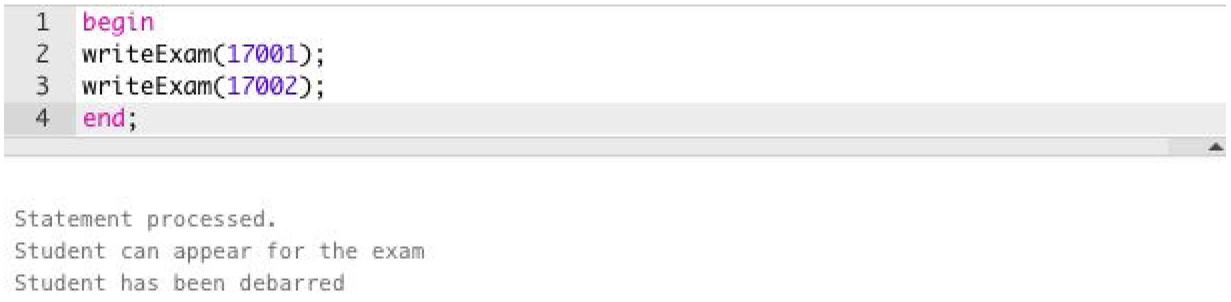
```
Procedure created.
```

begin

writeExam(17001);

writeExam(17002);

end;

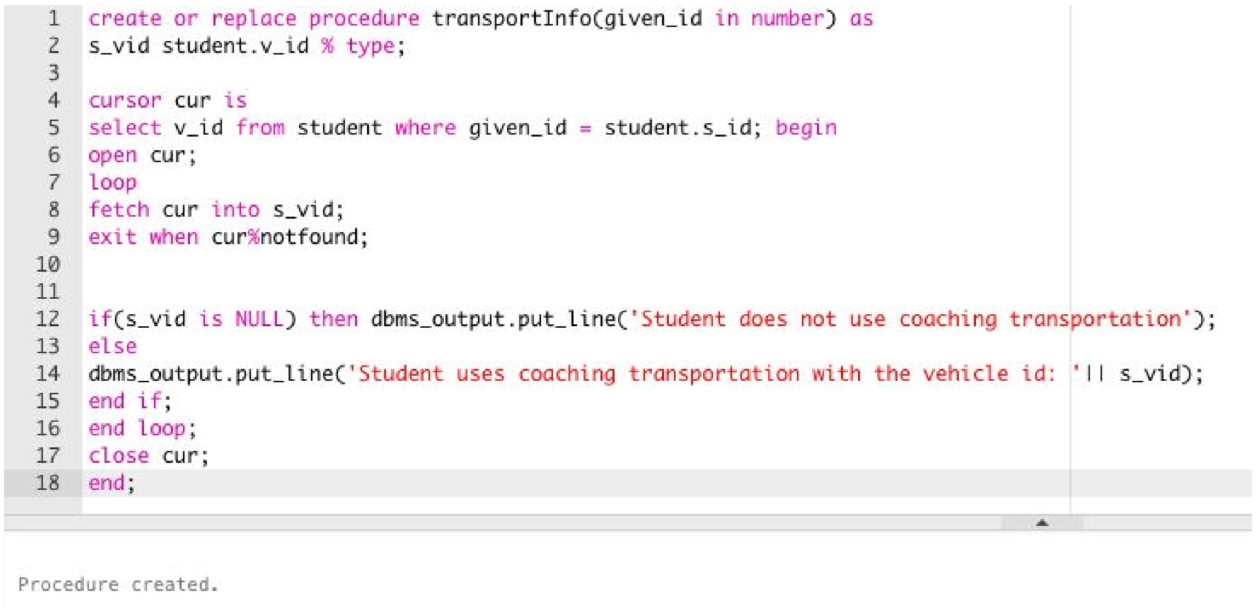


```
begin
writeExam(17001);
writeExam(17002);
end;
```

```
Statement processed.
Student can appear for the exam
Student has been debarred
```

2. **Procedure to display information related to the student’s commute.**

```
create or replace procedure transportInfo(given_id in number) as

s_vid student.v_id % type;

cursor cur is

select v_id from student where given_id = student.s_id; begin

open cur;

loop

fetch cur into s_vid;

exit when cur%notfound;

if(s_vid is NULL) then dbms_output.put_line('Student does not use coaching transportation');

else

dbms_output.put_line('Student uses coaching transportation with the vehicle id: '|| s_vid);

end if;

end loop;

close cur;

end;
```



```
create or replace procedure transportInfo(given_id in number) as
s_vid student.v_id % type;

cursor cur is
select v_id from student where given_id = student.s_id; begin
open cur;
loop
fetch cur into s_vid;
exit when cur%notfound;


if(s_vid is NULL) then dbms_output.put_line('Student does not use coaching transportation');
else
dbms_output.put_line('Student uses coaching transportation with the vehicle id: '|| s_vid);
end if;
end loop;
close cur;
end;
```

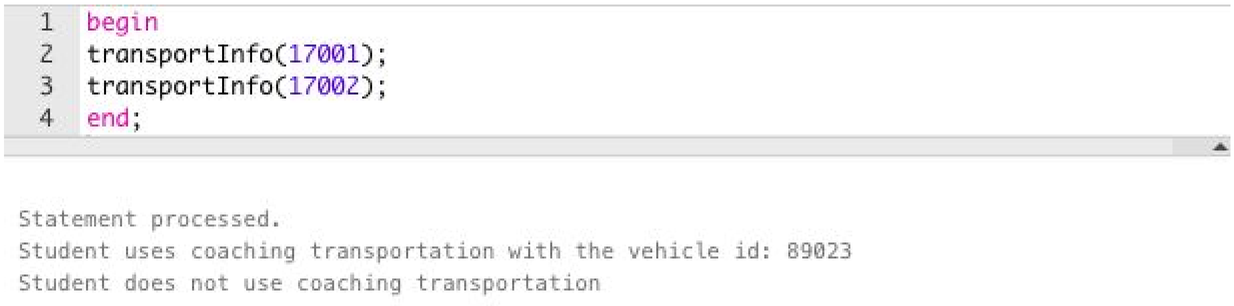
```
Procedure created.
```

begin

transportInfo(17001);

transportInfo(17002);

end;



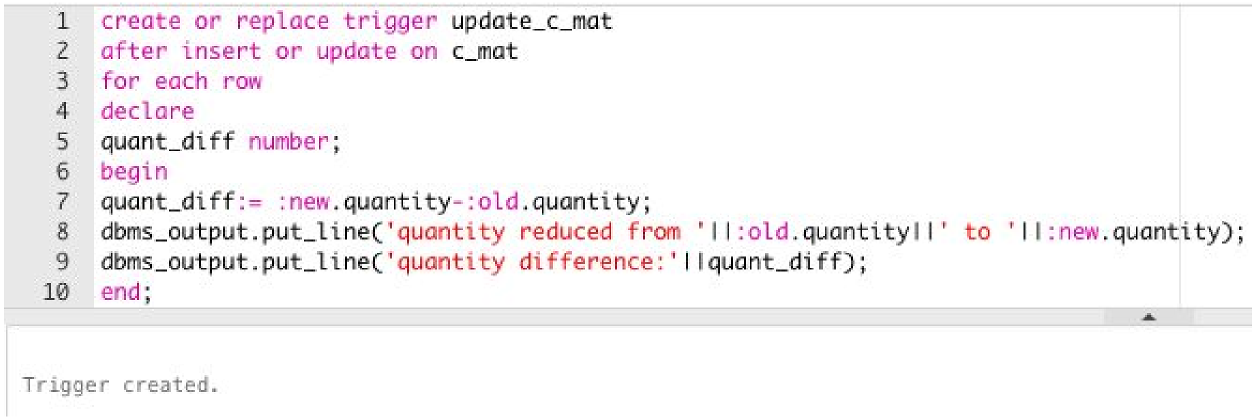
```
begin
transportInfo(17001);
transportInfo(17002);
end;
```

```
Statement processed.
Student uses coaching transportation with the vehicle id: 89023
Student does not use coaching transportation
```

IMPLEMENTATION OF BUSINESS RULES USING TRIGGER

1. **Show the number of materials in the stack as soon as the tuple on c_mat is updated.**

```
create or replace trigger update_c_mat

after insert or update on c_mat

for each row

declare

quant_diff number;

begin

quant_diff:= :new.quantity-:old.quantity;

dbms_output.put_line('quantity reduced from '||:old.quantity||' to '||:new.quantity);

dbms_output.put_line('quantity difference:'||quant_diff);

end;
```



```
create or replace trigger update_c_mat
after insert or update on c_mat
for each row
declare
quant_diff number;
begin
quant_diff:= :new.quantity-:old.quantity;
dbms_output.put_line('quantity reduced from '||:old.quantity||' to '||:new.quantity);
dbms_output.put_line('quantity difference:'||quant_diff);
end;
```

```
Trigger created.
```

insert into c_mat values(1234,250,'dpp','Physics',17002,12005);

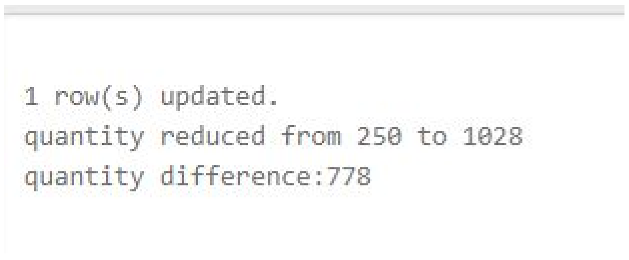
insert into student values(17002,'arjun','singh','rajput',date'2001-05-19','M','SC','indian','sharad nagar,saharanpur',90000,date'2018-04-21','A1','Shivjyoti','kota','indira vihar',96.66,'iit-jee',12345,89023);

insert into prog values(12005,1,2536,95);

update c_mat

set quantity=1028

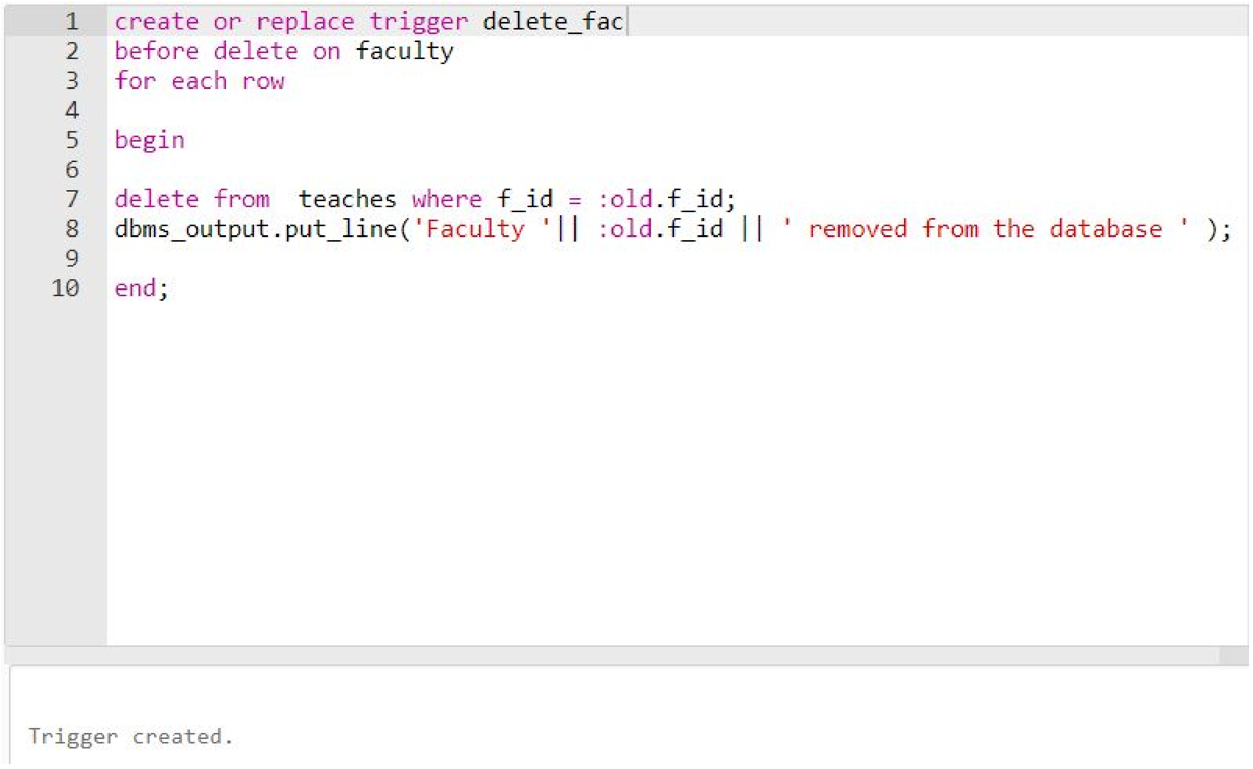
where s_id=17002;



```
1 row(s) updated.
quantity reduced from 250 to 1028
quantity difference:778
```

2. If the faculty leaves the institute then its record must be deleted fromall the other tables.

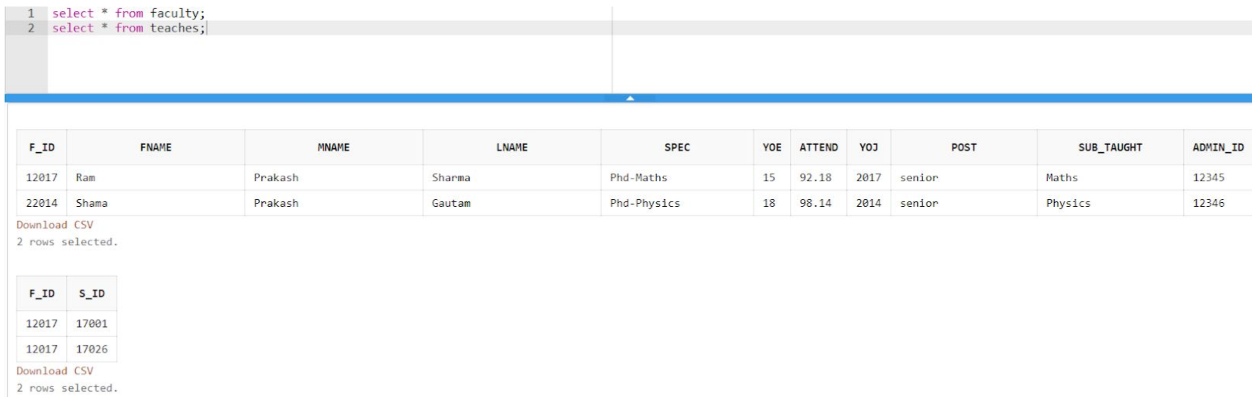
```
create or replace trigger delete_fac

before delete on faculty

for each row

begin

delete from  teaches where f_id = :old.f_id;

dbms_output.put_line('Faculty '|| :old.f_id || ' removed from the database ' );

end;
```



```
create or replace trigger delete_fac
before delete on faculty
for each row

begin

delete from  teaches where f_id = :old.f_id;
dbms_output.put_line('Faculty '|| :old.f_id || ' removed from the database ' );

end;
```

```
Trigger created.
```

Before Deleting :-



```
select * from faculty;
select * from teaches;
```

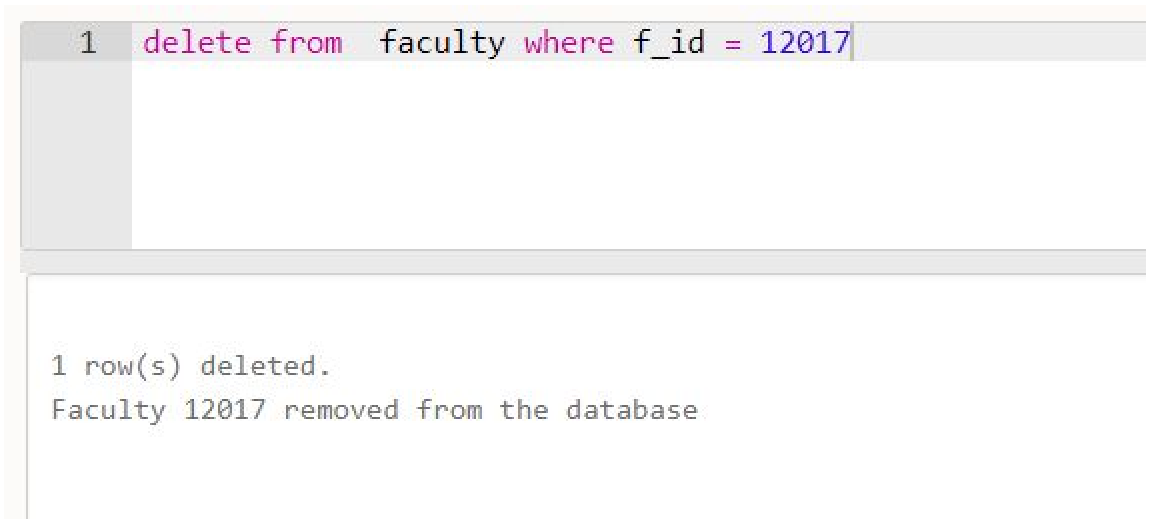
F_ID	FNAME	MNAME	LNAME	SPEC	YOE	ATTEND	YOJ	POST	SUB_TAUGHT	ADMIN_ID
12017	Ram	Prakash	Sharma	Phd-Maths	15	92.18	2017	senior	Maths	12345
22014	Shama	Prakash	Gautam	Phd-Physics	18	98.14	2014	senior	Physics	12346

Download CSV
2 rows selected.

F_ID	S_ID
12017	17001
12017	17026

Download CSV
2 rows selected.

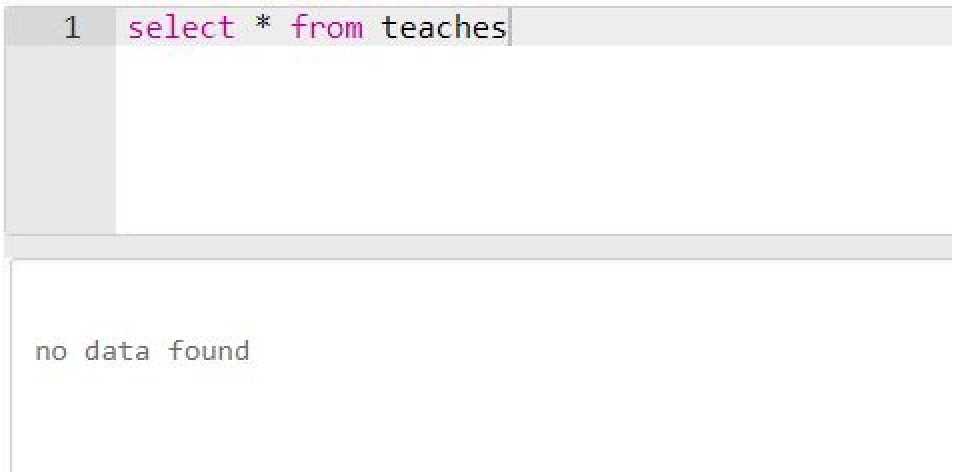
Deleting :-



```
delete from  faculty where f_id = 12017
```

```
1 row(s) deleted.
Faculty 12017 removed from the database
```

Teaches table after deleting :-

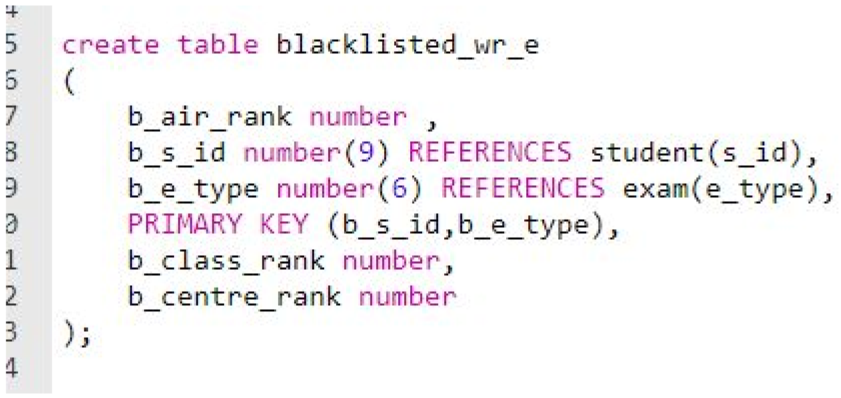


```
select * from teaches
```

```
no data found
```

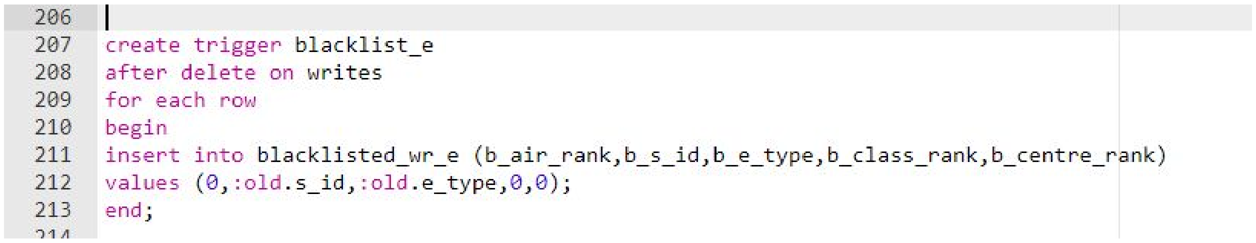
3. **If a student cheats on an exam then his/her student id is blacklisted along with exam type and the rank given is zero in all the cases.**

Creating a table for blacklisted student



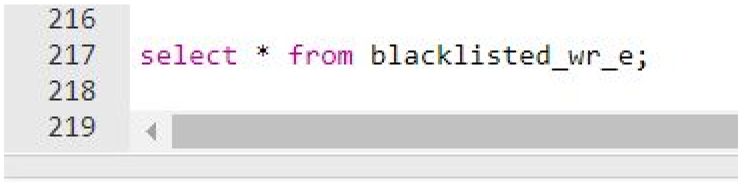
```
create table blacklisted_wr_e
(
    b_air_rank number ,
    b_s_id number(9) REFERENCES student(s_id),
    b_e_type number(6) REFERENCES exam(e_type),
    PRIMARY KEY (b_s_id,b_e_type),
    b_class_rank number,
    b_centre_rank number
);
```

Creating trigger for blacklisted student as soon he/she is deleted from student table

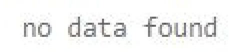


```
create trigger blacklist_e
after delete on writes
for each row
begin
insert into blacklisted_wr_e (b_air_rank,b_s_id,b_e_type,b_class_rank,b_centre_rank)
values (0,:old.s_id,:old.e_type,0,0);
end;
```

Before deletion the blacklisted_wr_e is empty

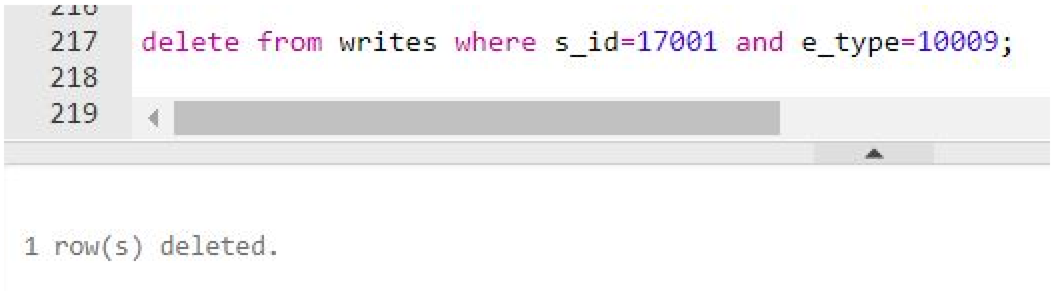


```
select * from blacklisted_wr_e;
```



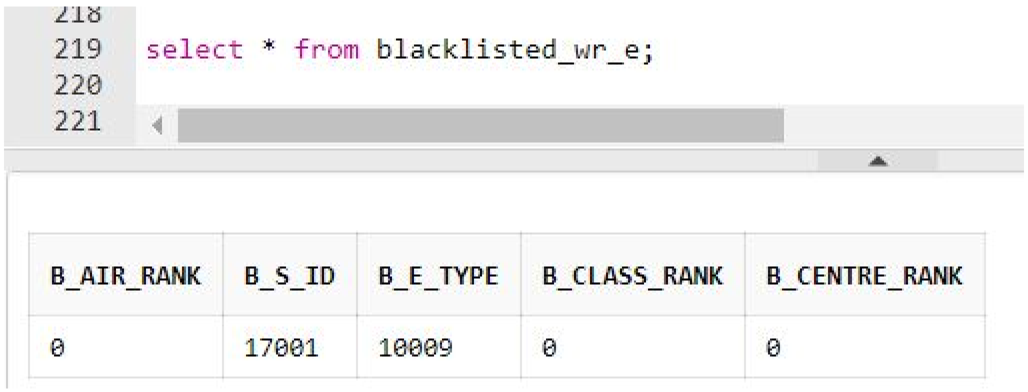
```
no data found
```

Deleting entry from student table



```
delete from writes where s_id=17001 and e_type=10009;
```

```
1 row(s) deleted.
```



```
select * from blacklisted_wr_e;
```

B_AIR_RANK	B_S_ID	B_E_TYPE	B_CLASS_RANK	B_CENTRE_RANK
0	17001	10009	0	0