SCHOOL MANAGEMENT SYSTEM

Queries:

1) **Given a class retrieve number of students studying in it.**

Relational Algebra:

$$\sigma_{class\_id=x}(\mathcal{F}_{count(*)}(student))$$

SQL Query:

SELECT COUNT(*) FROM student

WHERE (class_id=x)

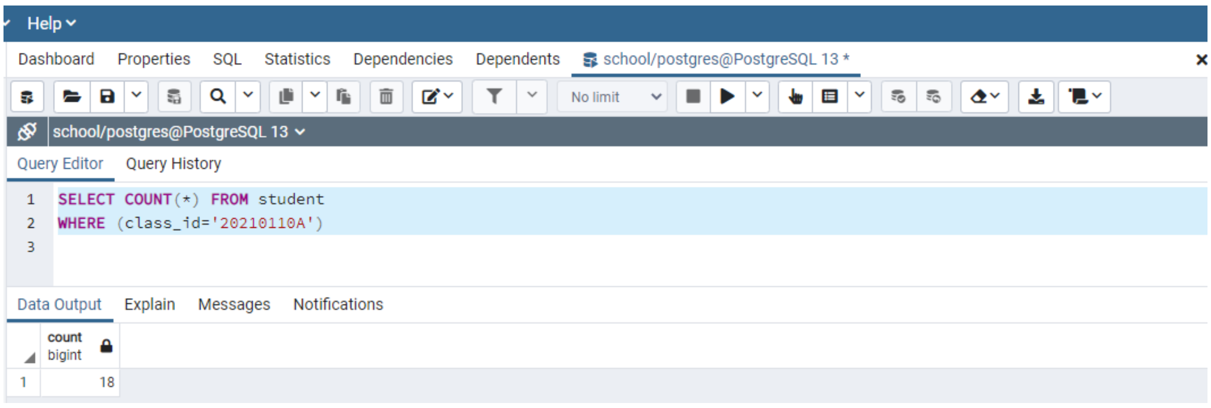
X is any class id which can be user input.

FOR EG:-x=20200110A

SELECT COUNT(*) FROM student

WHERE (class_id='20210110A')

OUTPUT:-



```
SELECT COUNT(*) FROM student
WHERE (class_id='20210110A')
```

```
   count
   bigint
1      18
```

2) **Find the names and student ids of students who scored minimum marks for each exam.**

Relational Algebra:

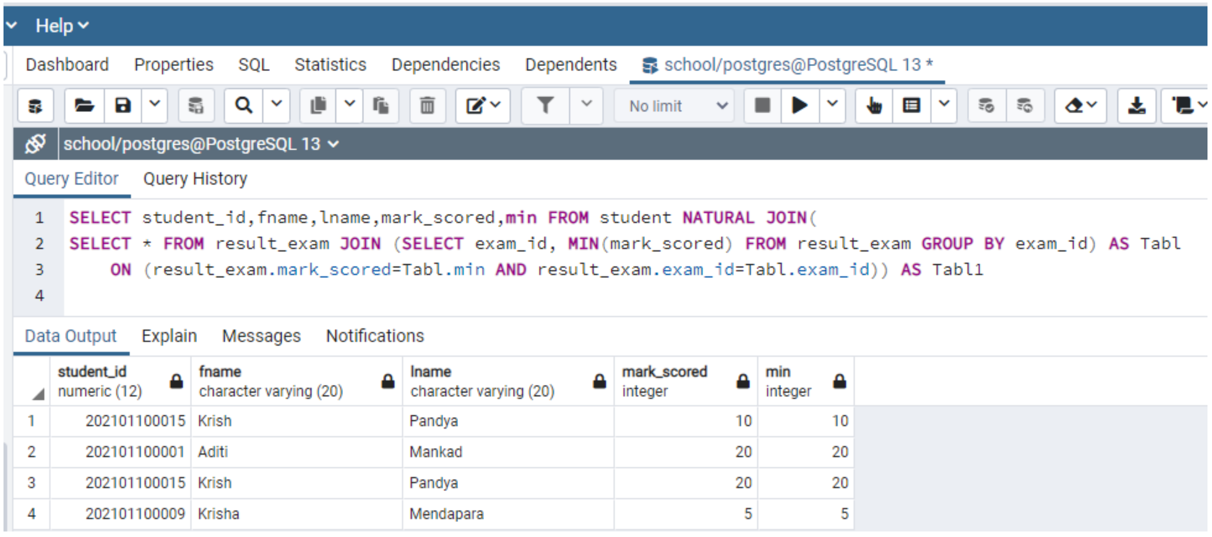
$$\Pi_{student\_id,fname,lname,mark\_scored,min}(student \bowtie \rho((result\_exam \bowtie_{result\_exam.mark\_scored=Tabl.min}$$

AND

$\text{result\_exam.exam\_id=Tabl.exam\_id}\ \rho((_{\text{exam\_id}}\mathcal{F}_{\text{MIN(mark\_scored)}}(\text{result\_exam}),\text{Tabl}),\text{Tabl1}))$

SQL Query:
SELECT student_id,fname,lname,mark_scored,min FROM student
NATURAL JOIN(
SELECT * FROM result_exam JOIN (SELECT exam_id, MIN(mark_scored)
FROM result_exam GROUP BY exam_id) AS Tabl
        ON (result_exam.mark_scored=Tabl.min AND
result_exam.exam_id=Tabl.exam_id)) AS Tabl1

Output:



```
SELECT student_id,fname,lname,mark_scored,min FROM student NATURAL JOIN(
SELECT * FROM result_exam JOIN (SELECT exam_id, MIN(mark_scored) FROM result_exam GROUP BY exam_id) AS Tabl
    ON (result_exam.mark_scored=Tabl.min AND result_exam.exam_id=Tabl.exam_id)) AS Tabl1
```

	student_id numeric (12)	fname character varying (20)	lname character varying (20)	mark_scored integer	min integer
1	202101100015	Krish	Pandya	10	10
2	202101100001	Aditi	Mankad	20	20
3	202101100015	Krish	Pandya	20	20
4	202101100009	Krisha	Mendapara	5	5

3) **Find the names and student ids of students who scored maximum marks for each exam.**
Relational Algebra:

$\Pi_{\text{student\_id,fname,lname,mark\_scored,max}}(\text{student}\ \bowtie$

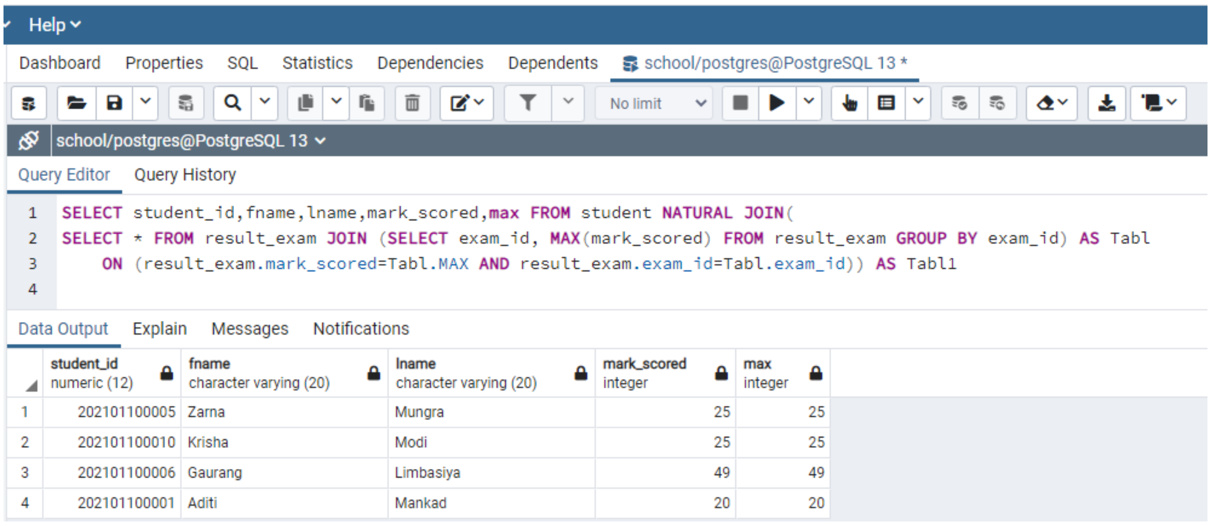
$\rho((\text{result\_exam}\ \bowtie_{\text{result\_exam.mark\_scored=Tabl.max}}$

AND

result_exam.exam_id=Tabl.exam_id $\rho((_{\text{exam\_id}}\mathcal{F}_{\text{MAX(mark\_scored)}}(\text{result\_exam}),\text{Tabl}),\text{Tabl1}))$

SQL Query:
SELECT student_id,fname,lname,mark_scored,max FROM student NATURAL JOIN(
SELECT * FROM result_exam JOIN (SELECT exam_id, MAX(mark_scored) FROM result_exam GROUP BY exam_id) AS Tabl
    ON (result_exam.mark_scored=Tabl.MAX AND result_exam.exam_id=Tabl.exam_id)) AS Tabl1

Output:



```
SELECT student_id,fname,lname,mark_scored,max FROM student NATURAL JOIN(
SELECT * FROM result_exam JOIN (SELECT exam_id, MAX(mark_scored) FROM result_exam GROUP BY exam_id) AS Tabl
    ON (result_exam.mark_scored=Tabl.MAX AND result_exam.exam_id=Tabl.exam_id)) AS Tabl1
```

	student_id numeric (12)	fname character varying (20)	lname character varying (20)	mark_scored integer	max integer
1	202101100005	Zarna	Mungra	25	25
2	202101100010	Krisha	Modi	25	25
3	202101100006	Gaurang	Limbasiya	49	49
4	202101100001	Aditi	Mankad	20	20

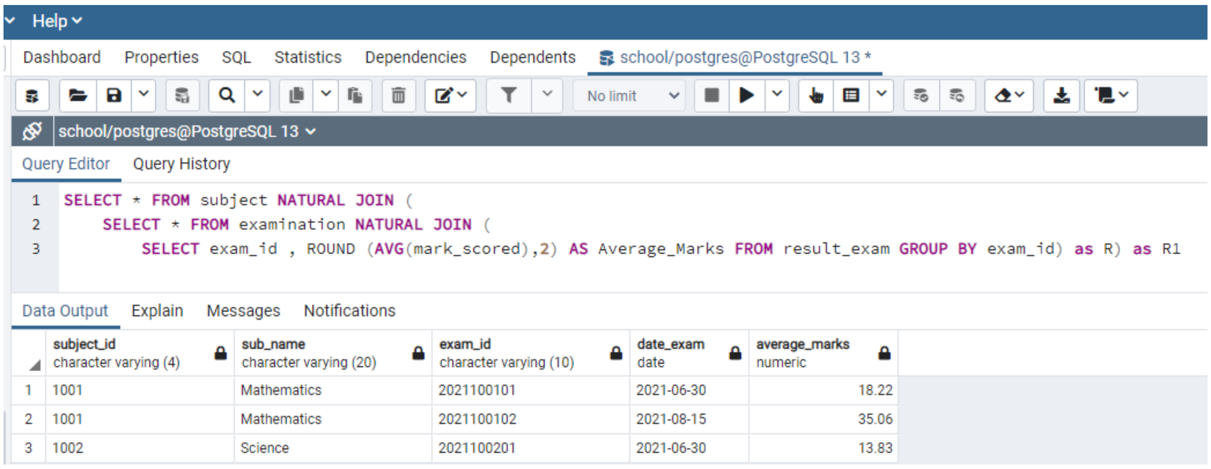
4) **Find average marks for each subject for each examination.**
Relational Algebra:

$$\text{Subject} \bowtie \rho(\text{examination} \bowtie \rho(_{\text{exam\_id}}\mathcal{F}_{\text{AVG(mark\_scored)->Average\_Marks,R}}),\text{R1})$$

SQL Query:
SELECT * FROM subject NATURAL JOIN (
    SELECT * FROM examination NATURAL JOIN (

SELECT exam_id , ROUND (AVG(mark_scored),2) AS Average_Marks FROM result_exam GROUP BY exam_id) as R) as R1

Output:



```
SELECT * FROM subject NATURAL JOIN (
    SELECT * FROM examination NATURAL JOIN (
        SELECT exam_id , ROUND (AVG(mark_scored),2) AS Average_Marks FROM result_exam GROUP BY exam_id) as R) as R1
```

	subject_id character varying (4)	sub_name character varying (20)	exam_id character varying (10)	date_exam date	average_marks numeric
1	1001	Mathematics	2021100101	2021-06-30	18.22
2	1001	Mathematics	2021100102	2021-08-15	35.06
3	1002	Science	2021100201	2021-06-30	13.83

5) Find number of Classrooms in given school branch.

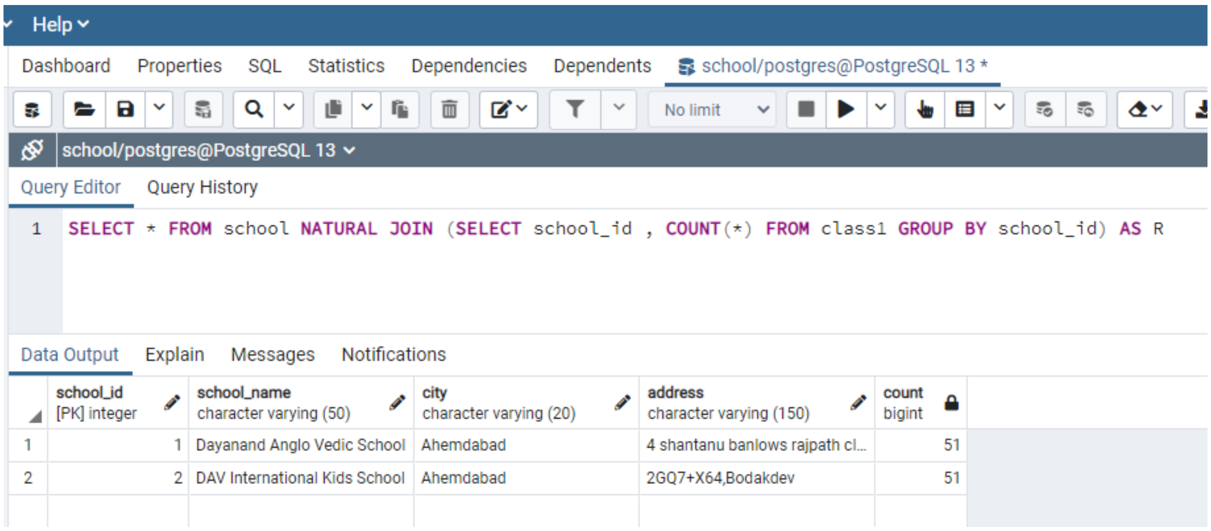
Relational Algebra:

$$\text{School} \bowtie \rho\left({}_{\text{school\_id}}\mathcal{F}_{\text{COUNT}(*)}(\text{class1}), R\right)$$

SQL Query:

SELECT * FROM school NATURAL JOIN (SELECT school_id , COUNT(*) FROM class1 GROUP BY school_id) AS R

Output:



```
SELECT * FROM school NATURAL JOIN (SELECT school_id , COUNT(*) FROM class1 GROUP BY school_id) AS R
```

	school_id [PK] integer	school_name character varying (50)	city character varying (20)	address character varying (150)	count bigint
1	1	Dayanand Anglo Vedic School	Ahemdabad	4 shantanu banlows rajpath cl...	51
2	2	DAV International Kids School	Ahemdabad	2GQ7+X64,Bodakdev	51

6) **Find the number of instruments which are in good condition.**

Relational Algebra:

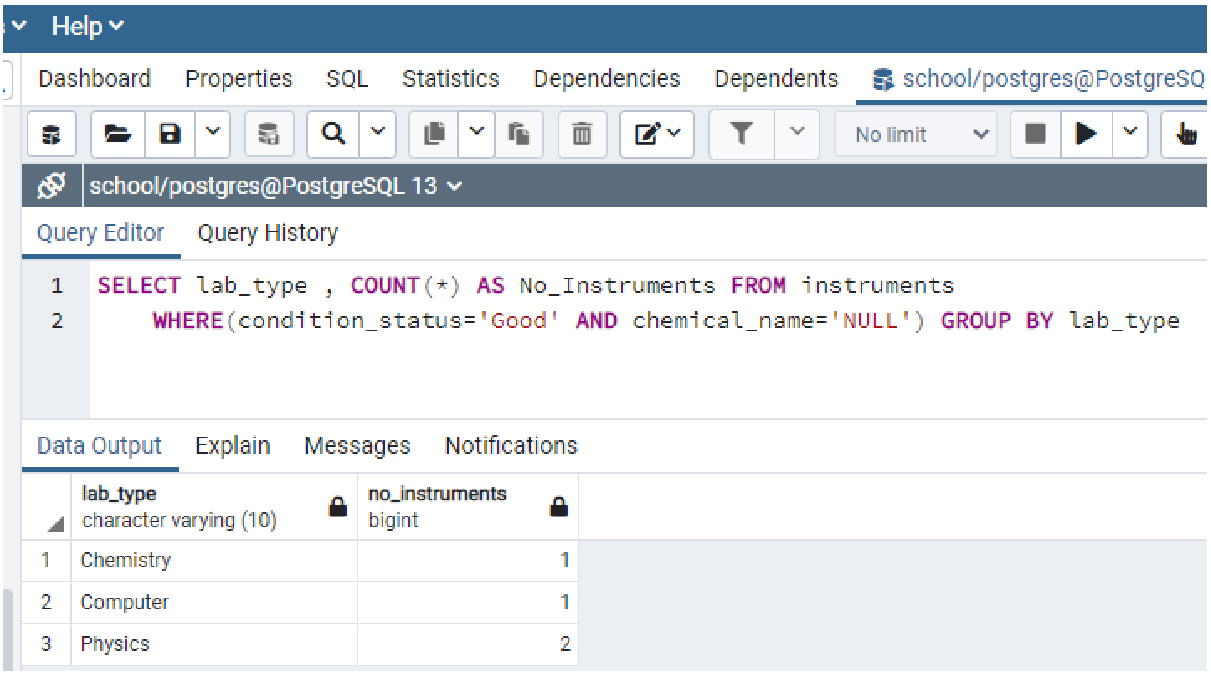
$$_{lab\_type}\mathcal{F}_{count(*)\text{->}Instruments}(instruments)$$

WHERE(condition_status=’Good’ AND chemical_name=’NULL’)

SQL Query:

SELECT lab_type , COUNT(*) AS No_Instruments FROM instruments
WHERE(condition_status='Good' AND chemical_name='NULL')
GROUP BY lab_type

Output:



```
SELECT lab_type , COUNT(*) AS No_Instruments FROM instruments
    WHERE(condition_status='Good' AND chemical_name='NULL') GROUP BY lab_type
```

	lab_type character varying (10)	no_instruments bigint
1	Chemistry	1
2	Computer	1
3	Physics	2

7) **Find the number of instruments which are to be repaired.**

Relational Algebra:

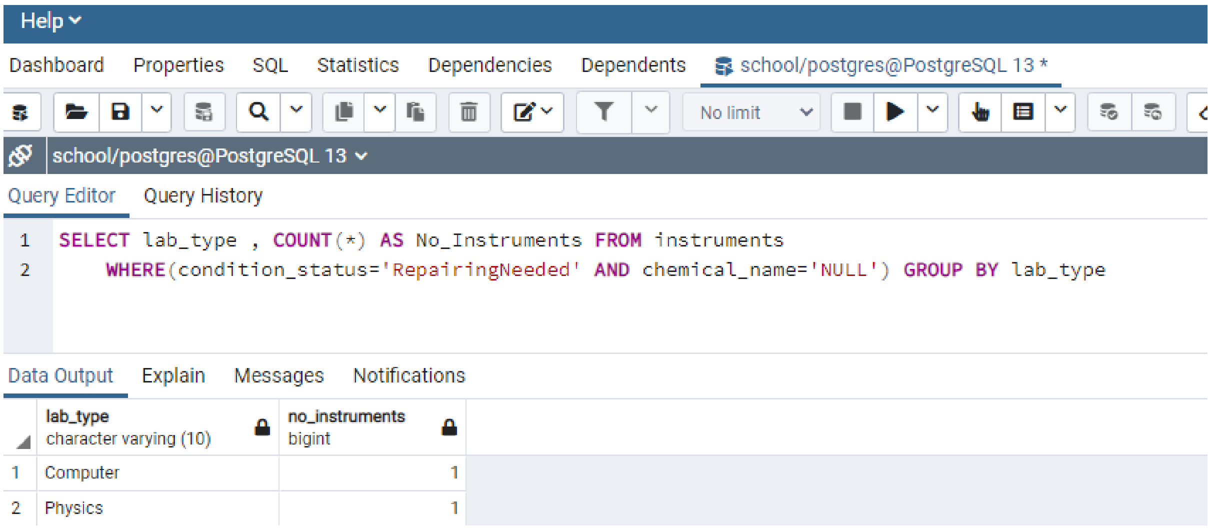
$$_{lab\_type}\mathcal{F}_{count(*)\text{->}Instruments}(instruments)$$

WHERE(condition_status=’RepairingNeeded’ AND chemical_name=’NULL’)

SQL Query:

SELECT lab_type , COUNT(*) AS No_Instruments FROM instruments WHERE(condition_status='RepairingNeeded' AND chemical_name='NULL') GROUP BY lab_type

Output:



```
SELECT lab_type , COUNT(*) AS No_Instruments FROM instruments
    WHERE(condition_status='RepairingNeeded' AND chemical_name='NULL') GROUP BY lab_type
```

	lab_type character varying (10)	no_instruments bigint
1	Computer	1
2	Physics	1

8) Find the number of instruments which are to be replaced.

Relational Algebra:

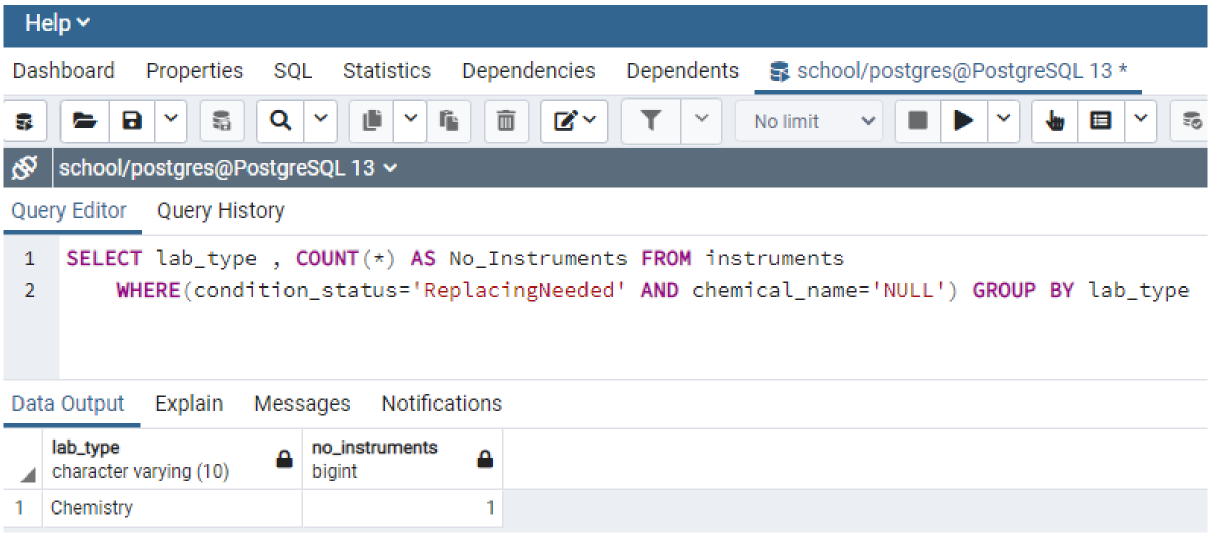
$$_{\text{lab\_type}}\mathcal{F}_{\text{count(*)->Instruments}}(\text{instruments})$$

WHERE(condition_status='ReplacingNeeded' AND chemical_name='NULL')

SQL Query:

SELECT lab_type , COUNT(*) AS No_Instruments FROM instruments WHERE(condition_status='ReplacingNeeded' AND chemical_name='NULL') GROUP BY lab_type

Output:



```sql
SELECT lab_type , COUNT(*) AS No_Instruments FROM instruments
    WHERE(condition_status='ReplacingNeeded' AND chemical_name='NULL') GROUP BY lab_type
```

```
   lab_type                  no_instruments
   character varying (10)    bigint
1  Chemistry                              1
```

9) Find the number of chemicals which are to be replaced.

Relational Algebra:

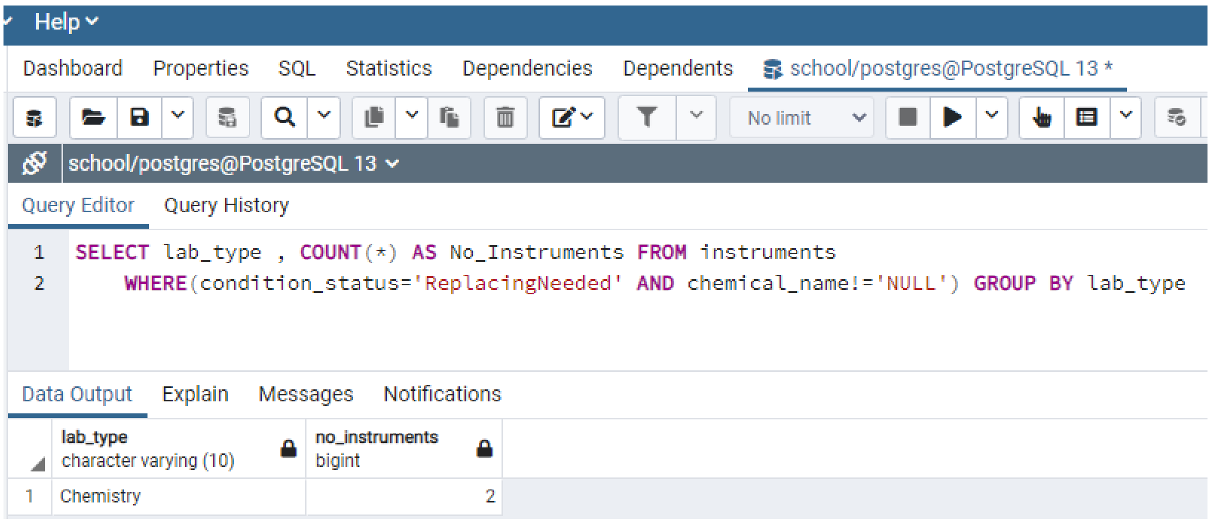
$$ {}_{lab\_type}F_{count(*)\text{->}Instruments}(instruments) $$

WHERE(condition_status=’ReplacingNeeded’ AND chemical_name!=’NULL’)

SQL Query:

SELECT lab_type , COUNT(*) AS No_Instruments FROM instruments WHERE(condition_status='ReplacingNeeded' AND chemical_name!='NULL') GROUP BY lab_type

Output:



```sql
SELECT lab_type , COUNT(*) AS No_Instruments FROM instruments
    WHERE(condition_status='ReplacingNeeded' AND chemical_name!='NULL') GROUP BY lab_type
```

```
   lab_type                  no_instruments
   character varying (10)    bigint
1  Chemistry                              2
```

10) List the names of chemicals which are to be replaced.

Relational Algebra:

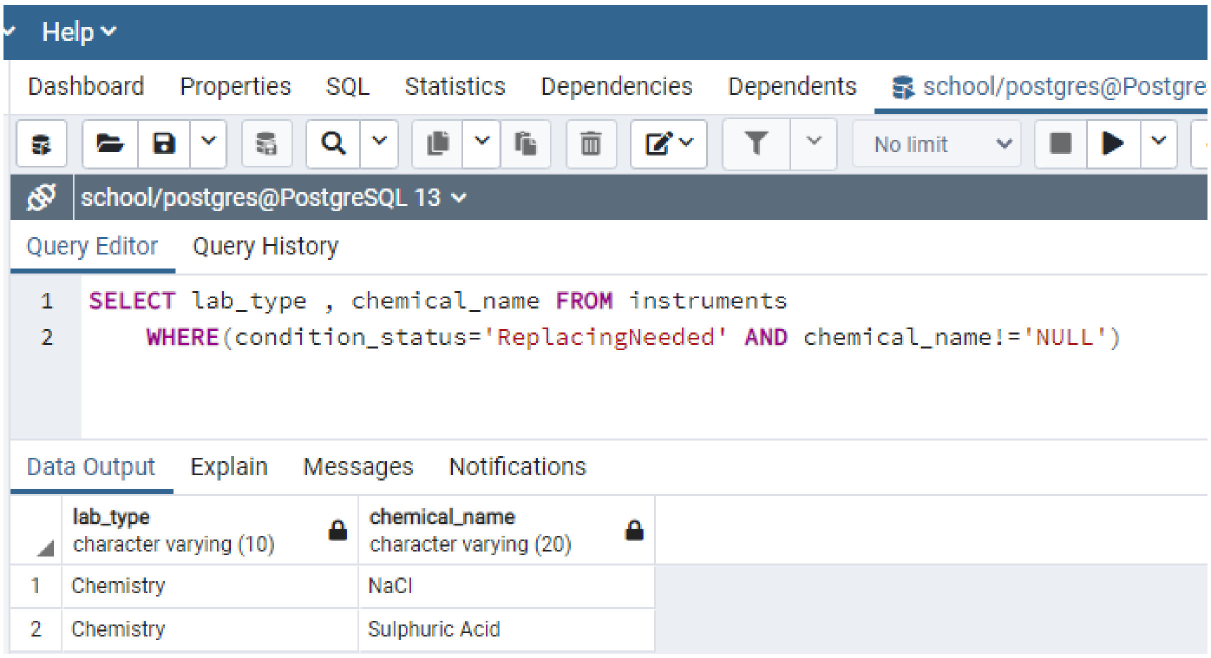
$$\Pi_{lab\_type,chemical\_name}(instruments)$$

WHERE(condition_status=’ReplacingNeeded’ AND chemical_name!=’NULL’)

SQL Query:

SELECT lab_type , chemical_name FROM instruments WHERE(condition_status='ReplacingNeeded' AND chemical_name!='NULL')

Output:



```
SELECT lab_type , chemical_name FROM instruments
    WHERE(condition_status='ReplacingNeeded' AND chemical_name!='NULL')
```

	lab_type character varying (10)	chemical_name character varying (20)
1	Chemistry	NaCl
2	Chemistry	Sulphuric Acid

11) List the names of instruments which are to be replaced.

Relational Algebra:

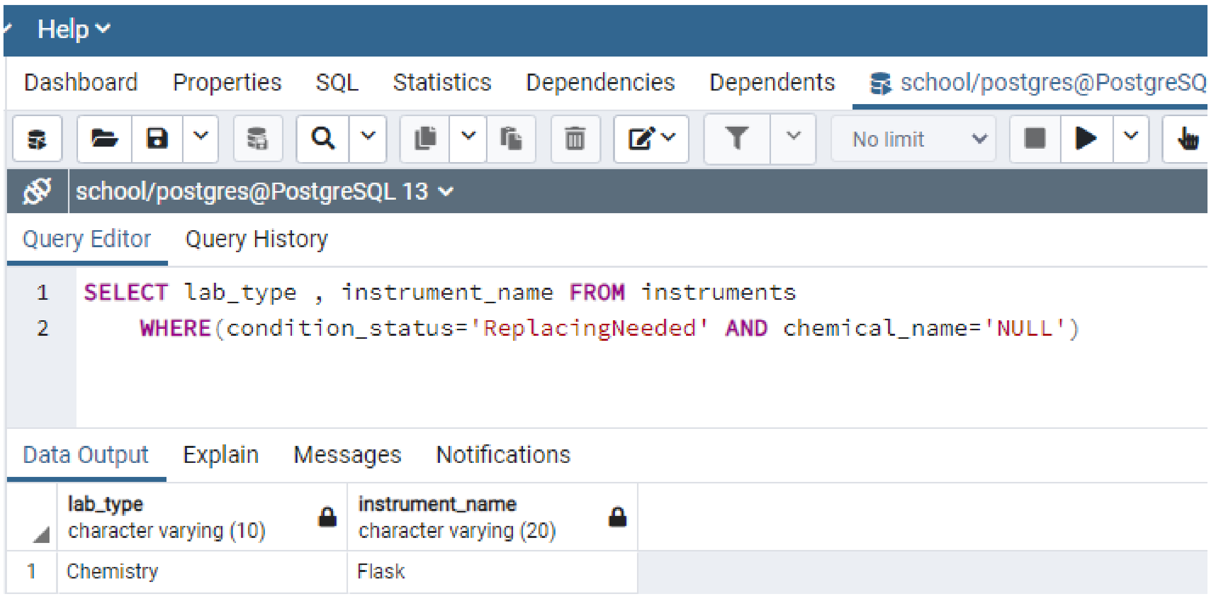
$$\Pi_{lab\_type,chemical\_name}(instruments)$$

WHERE(condition_status=’ReplacingNeeded’ AND chemical_name=’NULL’)

SQL Query:

SELECT lab_type , instrument_name FROM instruments WHERE(condition_status='ReplacingNeeded' AND chemical_name='NULL')

Output:



```
SELECT lab_type , instrument_name FROM instruments
    WHERE(condition_status='ReplacingNeeded' AND chemical_name='NULL')
```

	lab_type character varying (10)	instrument_name character varying (20)
1	Chemistry	Flask

12) List the names of instruments which are to be repaired.

Relational Algebra:

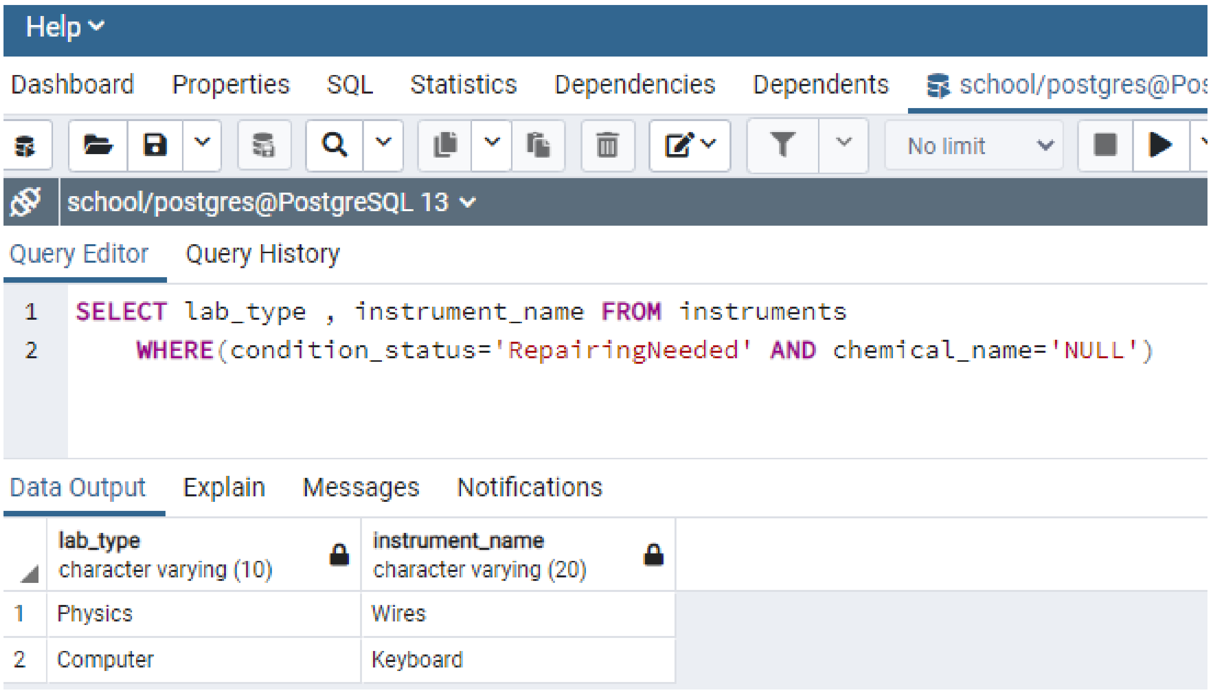
$$\Pi_{lab\_type,chemical\_name}(\text{instruments})$$

WHERE(condition_status=’RepairingNeeded’ AND chemical_name=’NULL’)

SQL Query:

SELECT lab_type , instrument_name FROM instruments WHERE(condition_status='RepairingNeeded' AND chemical_name='NULL')

Output:



```
SELECT lab_type , instrument_name FROM instruments
    WHERE(condition_status='RepairingNeeded' AND chemical_name='NULL')
```

	lab_type character varying (10)	instrument_name character varying (20)
1	Physics	Wires
2	Computer	Keyboard

13) List the names of instruments which are in good condition.

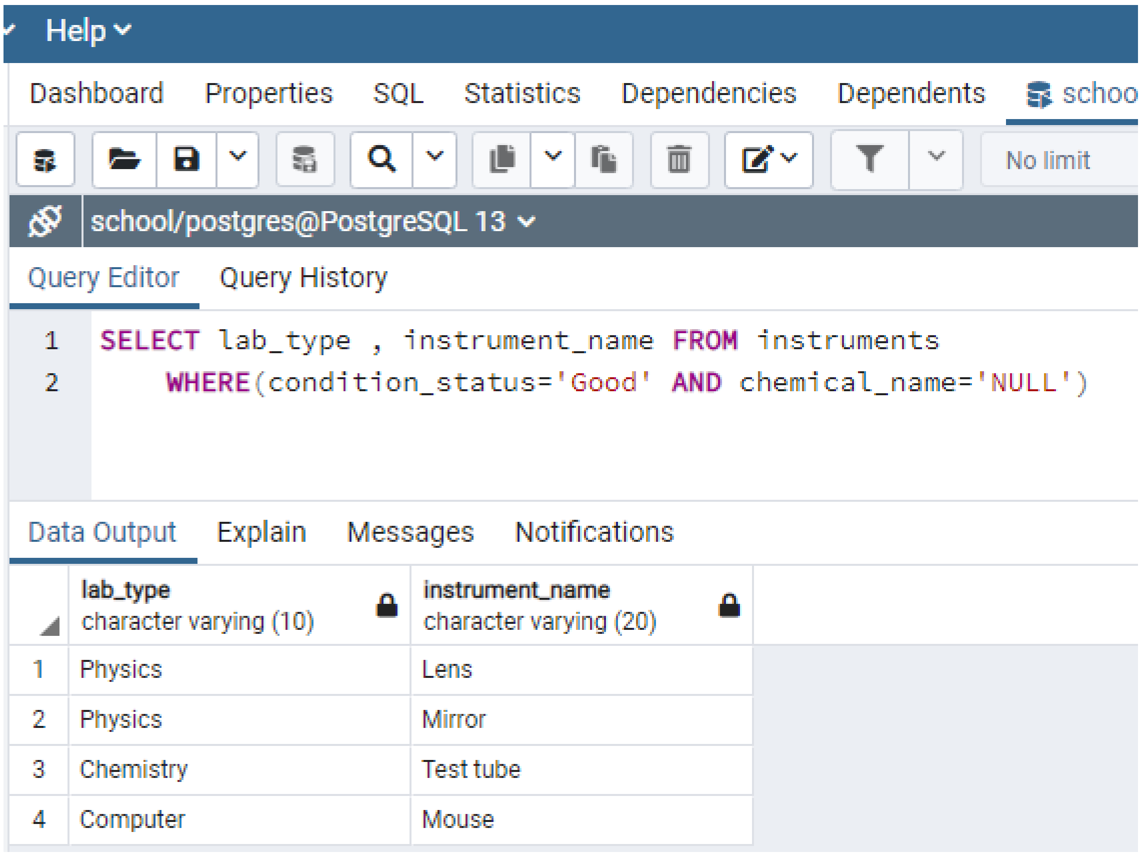
Relational Algebra:

$$\Pi_{\text{lab\_type,chemical\_name}}(\text{instruments})$$

WHERE(condition_status=’Good’ AND chemical_name=’NULL’)

SQL Query:
SELECT lab_type , instrument_name FROM instruments
WHERE(condition_status='Good' AND chemical_name='NULL')

Output:



```
SELECT lab_type , instrument_name FROM instruments
    WHERE(condition_status='Good' AND chemical_name='NULL')
```

	lab_type character varying (10)	instrument_name character varying (20)
1	Physics	Lens
2	Physics	Mirror
3	Chemistry	Test tube
4	Computer	Mouse

14) **List the names of employees whose salary is more than the average salary.**

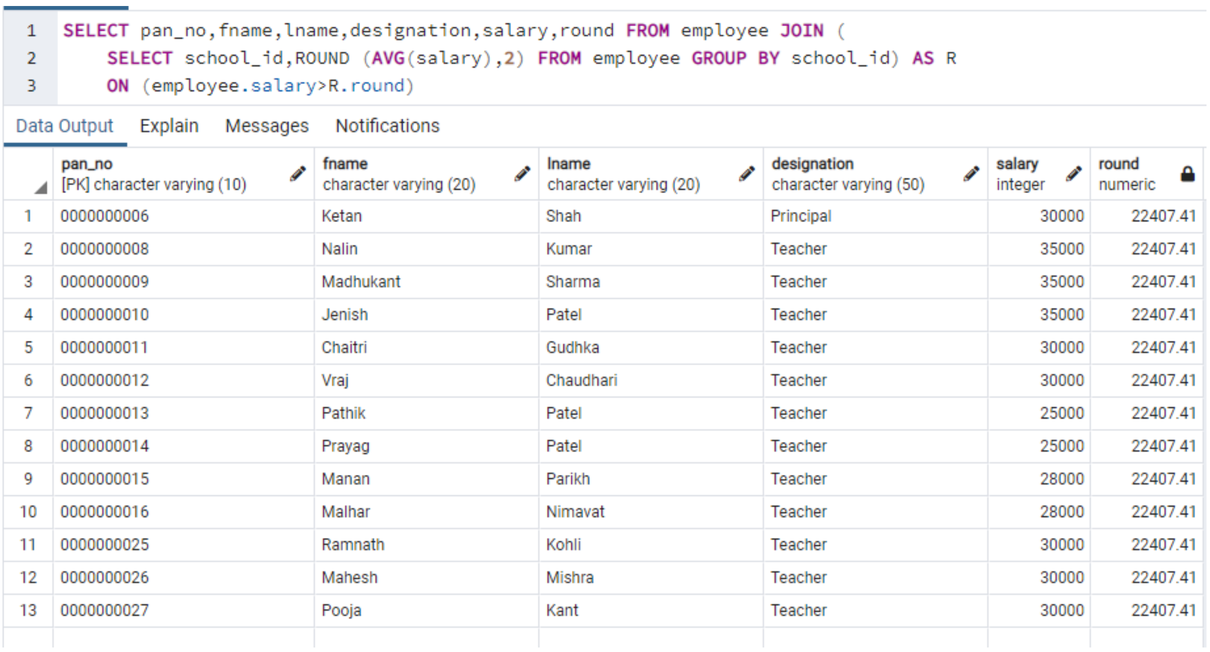
Relational Algebra:

$$\Pi_{pan\_no,fname,lname,designation,salary,round}(employee \bowtie_{employee.salary>R.round} \rho({}_{school\_id}\mathcal{F}_{ROUND(AVG(salary),2)}(employee),R)$$

SQL Query:

SELECT pan_no,fname,lname,designation,salary,round FROM employee JOIN (
	SELECT school_id,ROUND (AVG(salary),2) FROM employee GROUP BY school_id) AS R
	ON (employee.salary>R.round)

Output:



```sql
SELECT pan_no,fname,lname,designation,salary,round FROM employee JOIN (
    SELECT school_id,ROUND (AVG(salary),2) FROM employee GROUP BY school_id) AS R
    ON (employee.salary>R.round)
```

	pan_no [PK] character varying (10)	fname character varying (20)	lname character varying (20)	designation character varying (50)	salary integer	round numeric
1	0000000006	Ketan	Shah	Principal	30000	22407.41
2	0000000008	Nalin	Kumar	Teacher	35000	22407.41
3	0000000009	Madhukant	Sharma	Teacher	35000	22407.41
4	0000000010	Jenish	Patel	Teacher	35000	22407.41
5	0000000011	Chaitri	Gudhka	Teacher	30000	22407.41
6	0000000012	Vraj	Chaudhari	Teacher	30000	22407.41
7	0000000013	Pathik	Patel	Teacher	25000	22407.41
8	0000000014	Prayag	Patel	Teacher	25000	22407.41
9	0000000015	Manan	Parikh	Teacher	28000	22407.41
10	0000000016	Malhar	Nimavat	Teacher	28000	22407.41
11	0000000025	Ramnath	Kohli	Teacher	30000	22407.41
12	0000000026	Mahesh	Mishra	Teacher	30000	22407.41
13	0000000027	Pooja	Kant	Teacher	30000	22407.41

15) List the names of employees whose salary is less than the average salary.

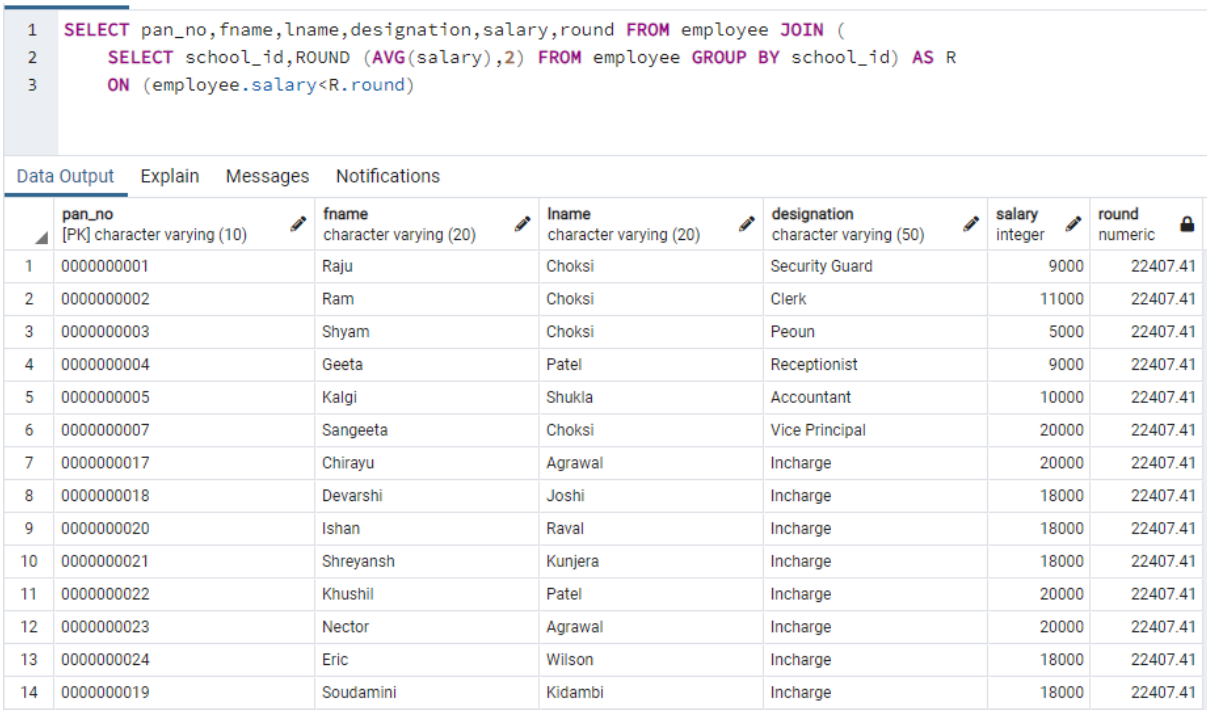
Relational Algebrs:

$$\Pi_{pan\_no,fname,lname,designation,salary,round}(employee) \bowtie_{employee.salary<R.round} \rho({}_{school\_id}\mathcal{F}_{ROUND(AVG(salary),2)}(employee),R)$$

SQL Query:

SELECT pan_no,fname,lname,designation,salary,round FROM employee JOIN (
    SELECT school_id,ROUND (AVG(salary),2) FROM employee GROUP BY school_id) AS R
    ON (employee.salary<R.round)

Output:



```
SELECT pan_no,fname,lname,designation,salary,round FROM employee JOIN (
    SELECT school_id,ROUND (AVG(salary),2) FROM employee GROUP BY school_id) AS R
    ON (employee.salary<R.round)
```

Data Output | Explain | Messages | Notifications

	pan_no [PK] character varying (10)	fname character varying (20)	lname character varying (20)	designation character varying (50)	salary integer	round numeric
1	0000000001	Raju	Choksi	Security Guard	9000	22407.41
2	0000000002	Ram	Choksi	Clerk	11000	22407.41
3	0000000003	Shyam	Choksi	Peoun	5000	22407.41
4	0000000004	Geeta	Patel	Receptionist	9000	22407.41
5	0000000005	Kalgi	Shukla	Accountant	10000	22407.41
6	0000000007	Sangeeta	Choksi	Vice Principal	20000	22407.41
7	0000000017	Chirayu	Agrawal	Incharge	20000	22407.41
8	0000000018	Devarshi	Joshi	Incharge	18000	22407.41
9	0000000020	Ishan	Raval	Incharge	18000	22407.41
10	0000000021	Shreyansh	Kunjera	Incharge	18000	22407.41
11	0000000022	Khushil	Patel	Incharge	20000	22407.41
12	0000000023	Nector	Agrawal	Incharge	20000	22407.41
13	0000000024	Eric	Wilson	Incharge	18000	22407.41
14	0000000019	Soudamini	Kidambi	Incharge	18000	22407.41

16) Find the names of students who scored more than average marks.

Relational Algebra:

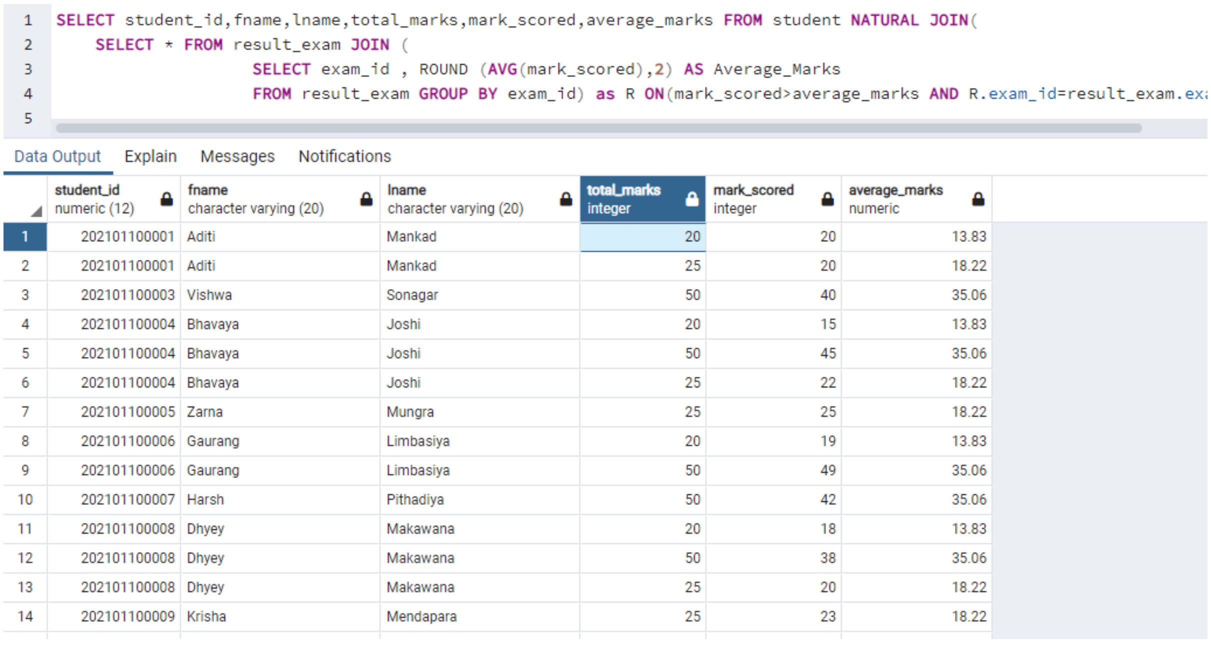
Πstudent_id,fname,lname,total_marks,mark_scored,average_marks student $\bowtie$ ρ(result_exam$\bowtie$(mark_scored>average_marks AND R.exam_id=result_exam.exam_id)ρ(exam_id$\mathcal{F}$ROUND(AVG(mark_scored),2)->Average_Marksresult_exam,R),R1)

SQL Query:

SELECT student_id,fname,lname,total_marks,mark_scored,average_marks FROM student NATURAL JOIN(
SELECT * FROM result_exam JOIN (
SELECT exam_id , ROUND (AVG(mark_scored),2) AS Average_Marks

FROM result_exam GROUP BY exam_id) as R ON(mark_scored>average_marks AND R.exam_id=result_exam.exam_id)) as R1

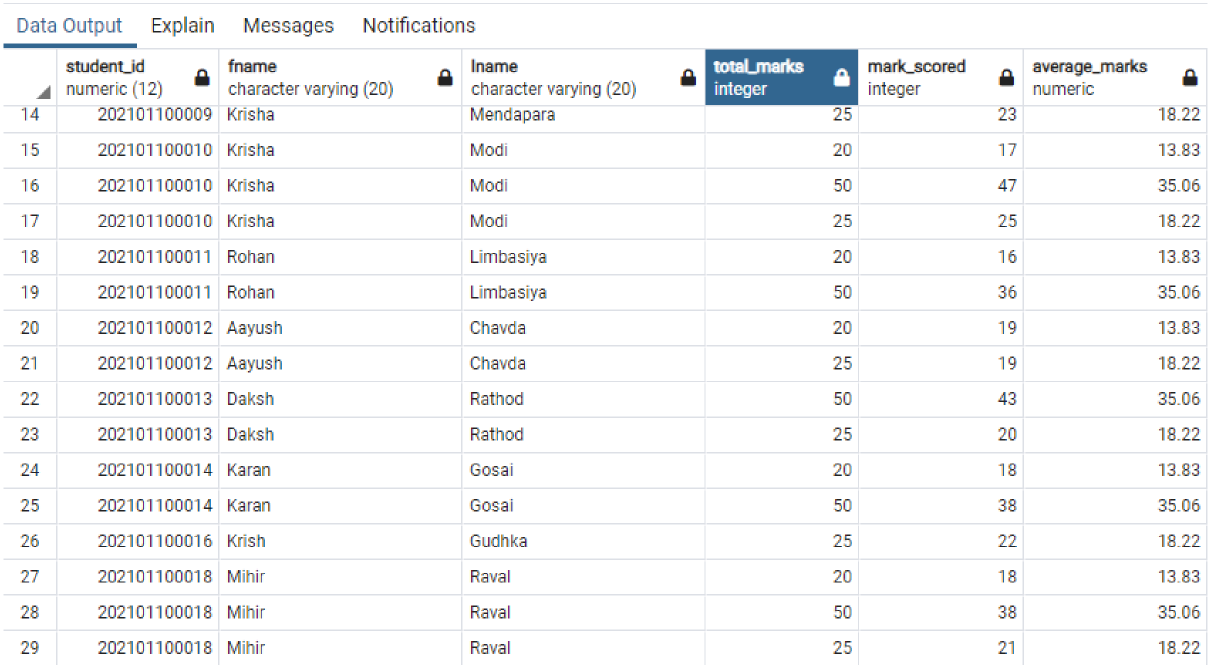
Output:



```
SELECT student_id,fname,lname,total_marks,mark_scored,average_marks FROM student NATURAL JOIN(
    SELECT * FROM result_exam JOIN (
                SELECT exam_id , ROUND (AVG(mark_scored),2) AS Average_Marks
                FROM result_exam GROUP BY exam_id) as R ON(mark_scored>average_marks AND R.exam_id=result_exam.ex
```

Data Output Explain Messages Notifications

	student_id numeric (12)	fname character varying (20)	lname character varying (20)	total_marks integer	mark_scored integer	average_marks numeric
1	202101100001	Aditi	Mankad	20	20	13.83
2	202101100001	Aditi	Mankad	25	20	18.22
3	202101100003	Vishwa	Sonagar	50	40	35.06
4	202101100004	Bhavaya	Joshi	20	15	13.83
5	202101100004	Bhavaya	Joshi	50	45	35.06
6	202101100004	Bhavaya	Joshi	25	22	18.22
7	202101100005	Zarna	Mungra	25	25	18.22
8	202101100006	Gaurang	Limbasiya	20	19	13.83
9	202101100006	Gaurang	Limbasiya	50	49	35.06
10	202101100007	Harsh	Pithadiya	50	42	35.06
11	202101100008	Dhyey	Makawana	20	18	13.83
12	202101100008	Dhyey	Makawana	50	38	35.06
13	202101100008	Dhyey	Makawana	25	20	18.22
14	202101100009	Krisha	Mendapara	25	23	18.22



Data Output Explain Messages Notifications

	student_id numeric (12)	fname character varying (20)	lname character varying (20)	total_marks integer	mark_scored integer	average_marks numeric
14	202101100009	Krisha	Mendapara	25	23	18.22
15	202101100010	Krisha	Modi	20	17	13.83
16	202101100010	Krisha	Modi	50	47	35.06
17	202101100010	Krisha	Modi	25	25	18.22
18	202101100011	Rohan	Limbasiya	20	16	13.83
19	202101100011	Rohan	Limbasiya	50	36	35.06
20	202101100012	Aayush	Chavda	20	19	13.83
21	202101100012	Aayush	Chavda	25	19	18.22
22	202101100013	Daksh	Rathod	50	43	35.06
23	202101100013	Daksh	Rathod	25	20	18.22
24	202101100014	Karan	Gosai	20	18	13.83
25	202101100014	Karan	Gosai	50	38	35.06
26	202101100016	Krish	Gudhka	25	22	18.22
27	202101100018	Mihir	Raval	20	18	13.83
28	202101100018	Mihir	Raval	50	38	35.06
29	202101100018	Mihir	Raval	25	21	18.22

17) Find the information about student's guardian for a given student id.

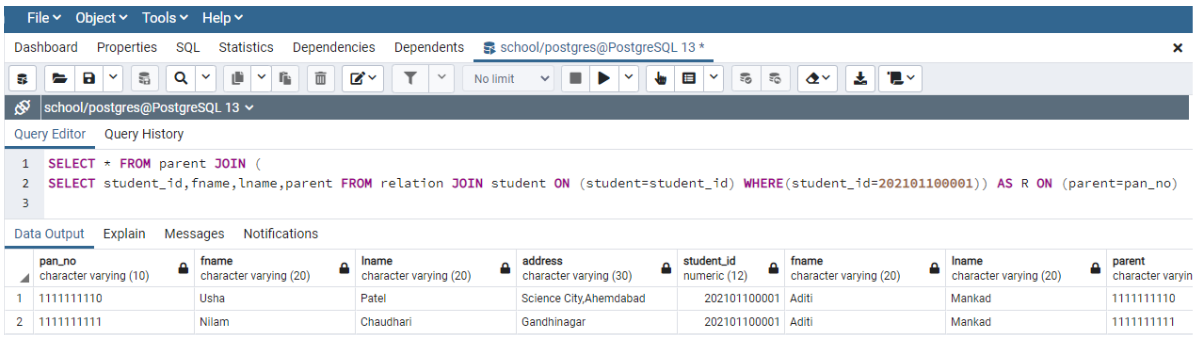
Relational Algebra:

$$\text{Parent} \bowtie_{\text{parent=pan\_no}} \rho(\Pi_{\text{student\_id,fname,lname,lname,parent}}\ \text{realtion} \bowtie_{\text{student=student\_id}} \text{student,R})$$

SQL Query:
SELECT * FROM parent JOIN (
SELECT student_id,fname,lname,parent FROM relation JOIN student ON (student=student_id) WHERE(student_id=x)) AS R ON (parent=pan_no)
WHERE x is user input student id.

For Eg:-x=202101100001
SELECT * FROM parent JOIN (
SELECT student_id,fname,lname,parent FROM relation JOIN student ON (student=student_id) WHERE(student_id=202101100001)) AS R ON (parent=pan_no)

OUTPUT:-



```
SELECT * FROM parent JOIN (
SELECT student_id,fname,lname,parent FROM relation JOIN student ON (student=student_id) WHERE(student_id=202101100001)) AS R ON (parent=pan_no)
```

	pan_no character varying (10)	fname character varying (20)	lname character varying (20)	address character varying (30)	student_id numeric (12)	fname character varying (20)	lname character varying (20)	parent character varyin
1	1111111110	Usha	Patel	Science City,Ahemdabad	202101100001	Aditi	Mankad	1111111110
2	1111111111	Nilam	Chaudhari	Gandhinagar	202101100001	Aditi	Mankad	1111111111

18).Find the list of students who are studying in class_id='20210110B'

Relational Algebra:

$$\Pi_{\text{student.class\_id,student\_id,fname,lname}}(\text{student} \bowtie \text{class1 EXCEPT } ((\Pi_{\text{student.class\_id,student\_id,fname,lname}}(\text{student CROSS JOIN class1 WHERE(class1.class\_id='20210110B')) EXCEPT}$$

$(\Pi_{\text{student.class\_id,student\_id,fname,lname}}(\text{student} \bowtie_{\text{student.class\_id=class1.class\_id AND class1.class\_id='20210110B'}} \text{class1}))))$

SQL Query:

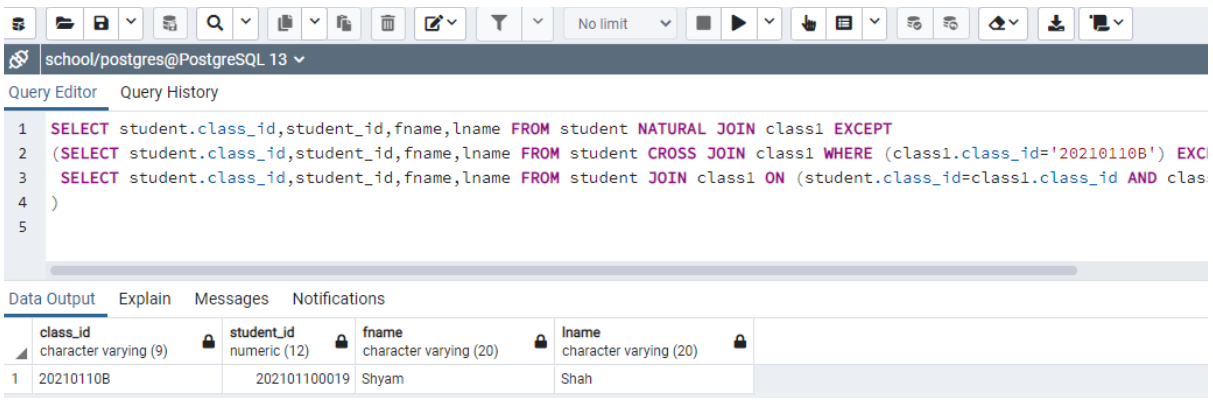
SELECT student.class_id,student_id,fname,lname FROM student NATURAL JOIN class1 EXCEPT

(SELECT student.class_id,student_id,fname,lname FROM student CROSS JOIN class1 WHERE (class1.class_id='20210110B') EXCEPT

 SELECT student.class_id,student_id,fname,lname FROM student JOIN class1 ON (student.class_id=class1.class_id AND class1.class_id='20210110B')

)

Output:



```
SELECT student.class_id,student_id,fname,lname FROM student NATURAL JOIN class1 EXCEPT
(SELECT student.class_id,student_id,fname,lname FROM student CROSS JOIN class1 WHERE (class1.class_id='20210110B') EXC
 SELECT student.class_id,student_id,fname,lname FROM student JOIN class1 ON (student.class_id=class1.class_id AND clas
)
```

	class_id character varying (9)	student_id numeric (12)	fname character varying (20)	lname character varying (20)
1	20210110B	202101100019	Shyam	Shah

19).Find the teachers who teaches in class_id='20210110A'.

Relational Algebra:

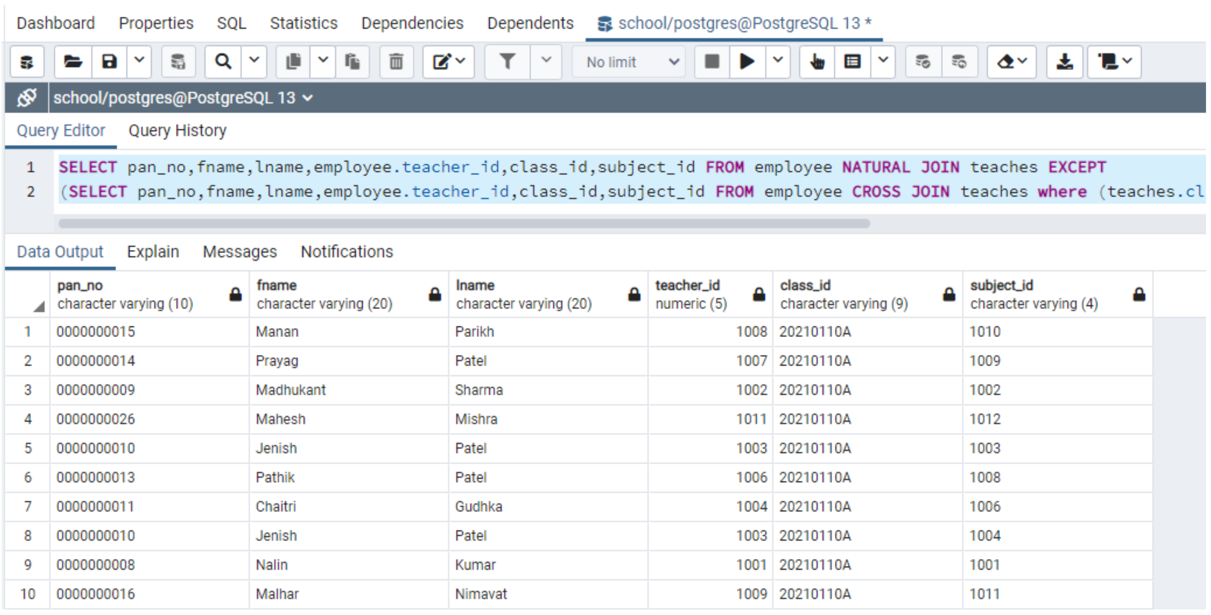
$\Pi_{\text{pan\_no,fname,lname,employee.teacher\_id,class\_id,subject\_id}}$ (employee $\bowtie$ teaches EXCEPT ($\Pi_{\text{pan\_no,fname,lname,employee.teacher\_id,class\_id,subject\_id}}$(employee CROSS JOIN teaches where (teaches.class_id!='20210110A' AND employee.teacher_id=teaches.teacher_id))))

SQL Query:

SELECT pan_no,fname,lname,employee.teacher_id,class_id,subject_id FROM employee NATURAL JOIN teaches EXCEPT

(SELECT pan_no,fname,lname,employee.teacher_id,class_id,subject_id FROM employee CROSS JOIN teaches where (teaches.class_id!='20210110A' AND employee.teacher_id=teaches.teacher_id))

Output:



```
SELECT pan_no,fname,lname,employee.teacher_id,class_id,subject_id FROM employee NATURAL JOIN teaches EXCEPT
(SELECT pan_no,fname,lname,employee.teacher_id,class_id,subject_id FROM employee CROSS JOIN teaches where (teaches.cl
```

	pan_no character varying (10)	fname character varying (20)	lname character varying (20)	teacher_id numeric (5)	class_id character varying (9)	subject_id character varying (4)
1	0000000015	Manan	Parikh	1008	20210110A	1010
2	0000000014	Prayag	Patel	1007	20210110A	1009
3	0000000009	Madhukant	Sharma	1002	20210110A	1002
4	0000000026	Mahesh	Mishra	1011	20210110A	1012
5	0000000010	Jenish	Patel	1003	20210110A	1003
6	0000000013	Pathik	Patel	1006	20210110A	1008
7	0000000011	Chaitri	Gudhka	1004	20210110A	1006
8	0000000010	Jenish	Patel	1003	20210110A	1004
9	0000000008	Nalin	Kumar	1001	20210110A	1001
10	0000000016	Malhar	Nimavat	1009	20210110A	1011

20). List down all the books which arrived before 20 years.

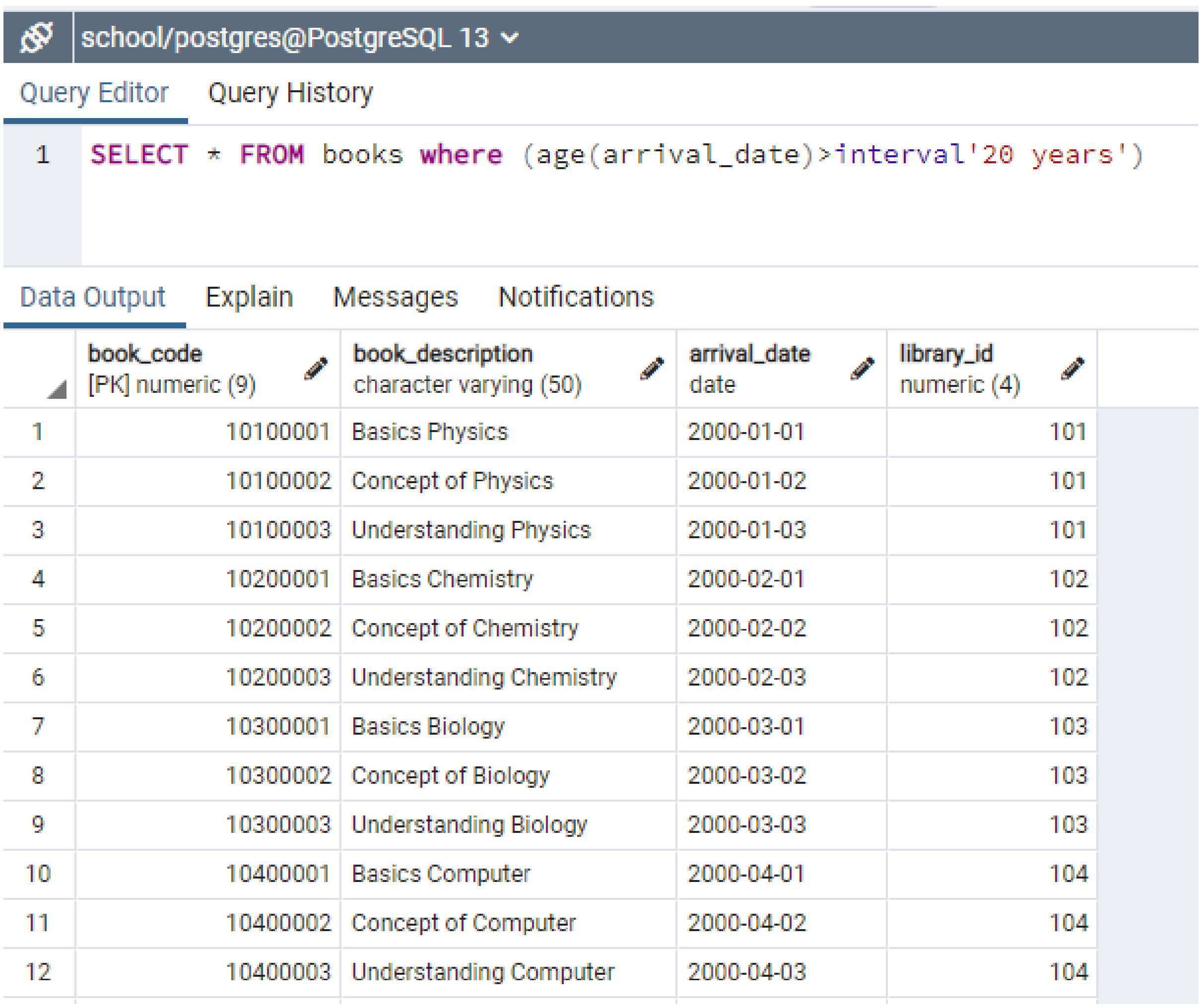
Relational Algebra:

$$\sigma_{\text{age(arrival\_date)>inetrval'20years'}}(\text{books})$$

SQL Query:

SELECT * FROM books where (age(arrival_date)>interval'20 years')

Output:



```sql
SELECT * FROM books where (age(arrival_date)>interval'20 years')
```

	book_code [PK] numeric (9)	book_description character varying (50)	arrival_date date	library_id numeric (4)
1	10100001	Basics Physics	2000-01-01	101
2	10100002	Concept of Physics	2000-01-02	101
3	10100003	Understanding Physics	2000-01-03	101
4	10200001	Basics Chemistry	2000-02-01	102
5	10200002	Concept of Chemistry	2000-02-02	102
6	10200003	Understanding Chemistry	2000-02-03	102
7	10300001	Basics Biology	2000-03-01	103
8	10300002	Concept of Biology	2000-03-02	103
9	10300003	Understanding Biology	2000-03-03	103
10	10400001	Basics Computer	2000-04-01	104
11	10400002	Concept of Computer	2000-04-02	104
12	10400003	Understanding Computer	2000-04-03	104

21).Find the teachers whose salary is greater than average salary of teachers.

Relational Algebra:

$$\Pi_{pan\_no,fname,lname,salary,round,Teacher\_ID}(employee \bowtie_{employee.salary>R.round} \rho(_{school\_id}\digamma_{ROUND(AVG(salary),2)}(employee)\ WHERE(teacher\_id\ is\ not\ null),R)$$

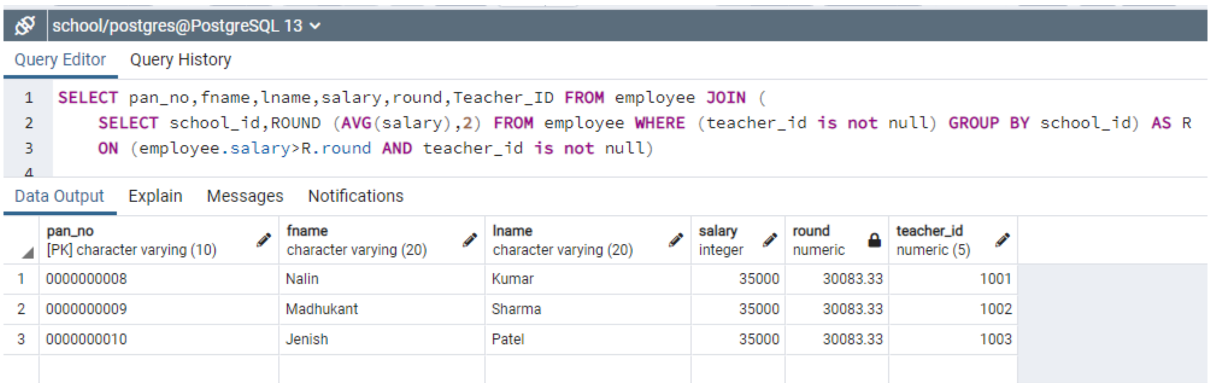
SQL Query:

SELECT pan_no,fname,lname,salary,round,Teacher_ID FROM employee JOIN (

SELECT school_id,ROUND(AVG(salary),2) FROM employee WHERE (teacher_id is not null) GROUP BY school_id) AS R

ON (employee.salary>R.round AND teacher_id is not null)

Output:


school/postgres@PostgreSQL 13

Query Editor | Query History

```
SELECT pan_no,fname,lname,salary,round,Teacher_ID FROM employee JOIN (
    SELECT school_id,ROUND (AVG(salary),2) FROM employee WHERE (teacher_id is not null) GROUP BY school_id) AS R
    ON (employee.salary>R.round AND teacher_id is not null)
```

Data Output | Explain | Messages | Notifications

	pan_no [PK] character varying (10)	fname character varying (20)	lname character varying (20)	salary integer	round numeric	teacher_id numeric (5)
1	0000000008	Nalin	Kumar	35000	30083.33	1001
2	0000000009	Madhukant	Sharma	35000	30083.33	1002
3	0000000010	Jenish	Patel	35000	30083.33	1003

22).Find all the students who didn't paid any fees(excluding those whose total fees is zero).

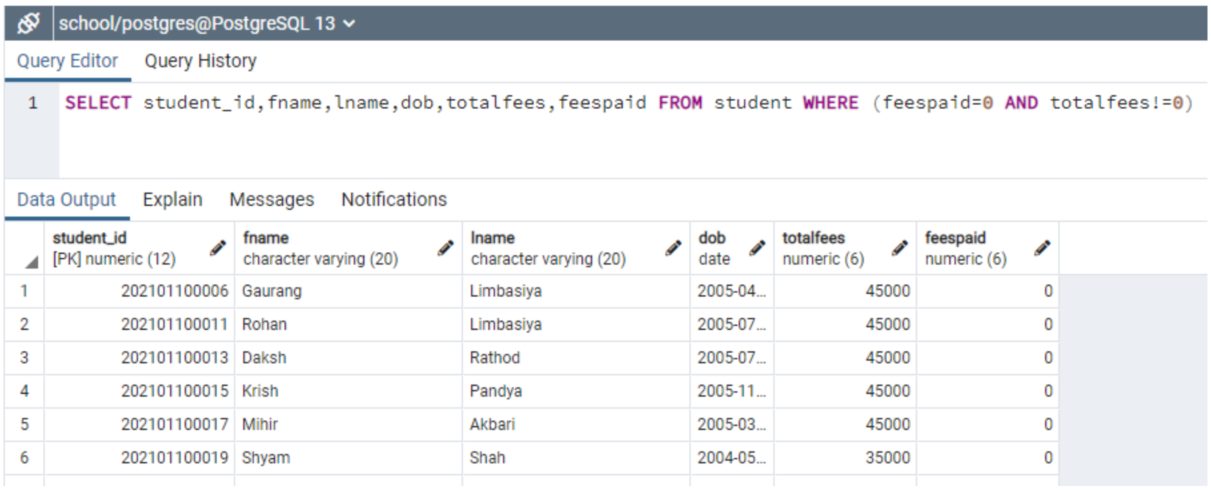
Relational Algebra:

$$\Pi_{student\_id,fname,lname,dob,totalfees,feespaid}(student)\ \text{WHERE}\ (feespaid=0\ \text{AND}\ totalfees!=0)$$

SQL Query:

SELECT student_id,fname,lname,dob,totalfees,feespaid FROM student WHERE (feespaid=0 AND totalfees!=0)

Output:


school/postgres@PostgreSQL 13

Query Editor | Query History

```
SELECT student_id,fname,lname,dob,totalfees,feespaid FROM student WHERE (feespaid=0 AND totalfees!=0)
```

Data Output | Explain | Messages | Notifications

	student_id [PK] numeric (12)	fname character varying (20)	lname character varying (20)	dob date	totalfees numeric (6)	feespaid numeric (6)
1	202101100006	Gaurang	Limbasiya	2005-04...	45000	0
2	202101100011	Rohan	Limbasiya	2005-07...	45000	0
3	202101100013	Daksh	Rathod	2005-07...	45000	0
4	202101100015	Krish	Pandya	2005-11...	45000	0
5	202101100017	Mihir	Akbari	2005-03...	45000	0
6	202101100019	Shyam	Shah	2004-05...	35000	0

23).Find all the book names and book_codes for books written by HC Verma.

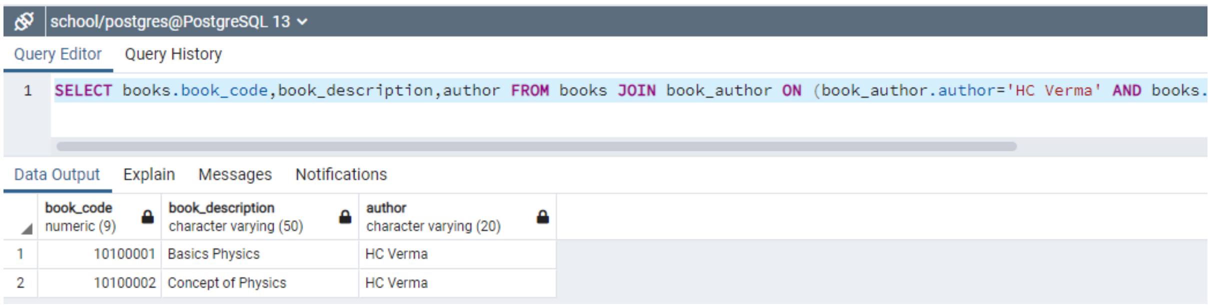
Relational Algebra:

$\Pi_{\text{books.book\_code,book\_description,author}}$(books $\bowtie_{\text{(book\_author='HC Verma' AND books.book\_code=book\_author.book\_code)}}$)

SQL Query:

SELECT books.book_code,book_description,author FROM books JOIN book_author ON (book_author.author='HC Verma' AND books.book_code=book_author.book_code)

Output:



```
SELECT books.book_code,book_description,author FROM books JOIN book_author ON (book_author.author='HC Verma' AND books.
```

	book_code numeric (9)	book_description character varying (50)	author character varying (20)
1	10100001	Basics Physics	HC Verma
2	10100002	Concept of Physics	HC Verma

24). Find all the books written by HC Verma and all the other authors who wrote the same book.

Relational Algebra:

$\Pi_{\text{books.book\_code,book\_description,author}}$ (books $\bowtie$($\Pi_{\text{b1.book\_code,b1.author}}$($\rho$(book_author,b1) $\bowtie_{\text{(b2.author='HC Verma' AND b1.book\_code=b2.book\_code)}}$ ρ(book_author,b2))))

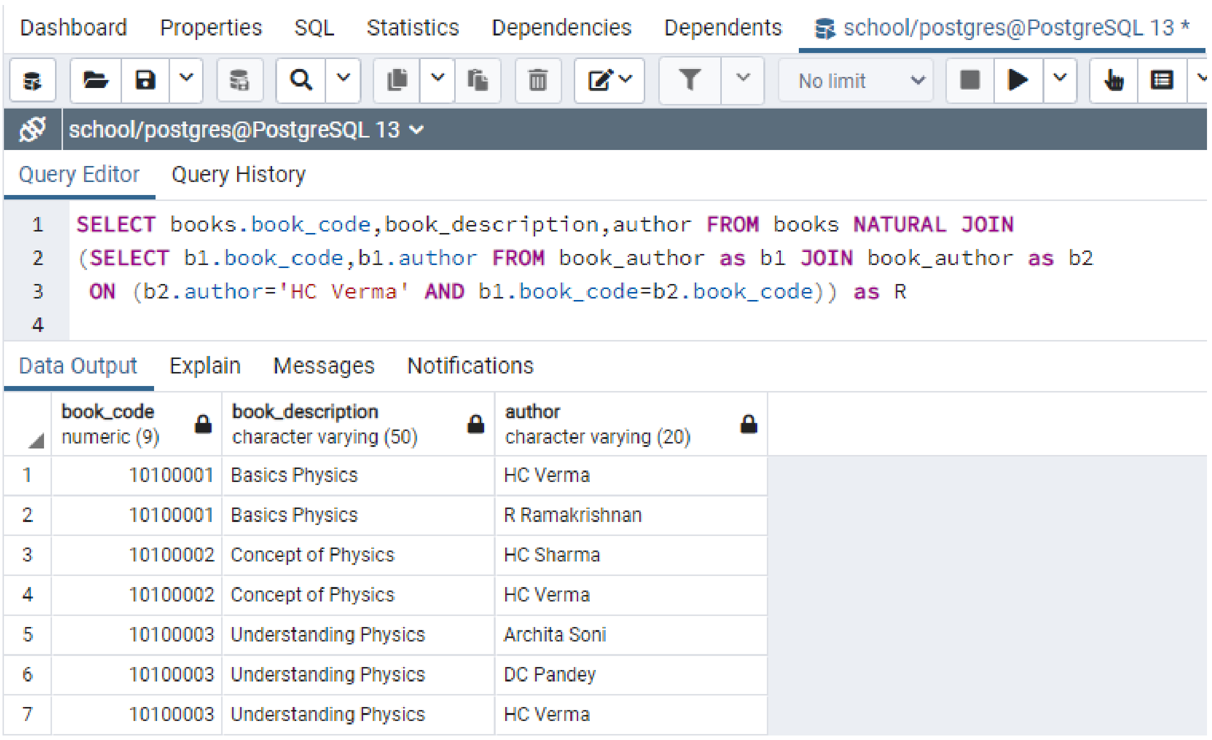
SQL Query:

SELECT books.book_code,book_description,author FROM books NATURAL JOIN

(SELECT b1.book_code,b1.author FROM book_author as b1 JOIN book_author as b2

ON (b2.author='HC Verma' AND b1.book_code=b2.book_code)) as R

Output:



```
SELECT books.book_code,book_description,author FROM books NATURAL JOIN
(SELECT b1.book_code,b1.author FROM book_author as b1 JOIN book_author as b2
 ON (b2.author='HC Verma' AND b1.book_code=b2.book_code)) as R
```

	book_code numeric (9)	book_description character varying (50)	author character varying (20)
1	10100001	Basics Physics	HC Verma
2	10100001	Basics Physics	R Ramakrishnan
3	10100002	Concept of Physics	HC Sharma
4	10100002	Concept of Physics	HC Verma
5	10100003	Understanding Physics	Archita Soni
6	10100003	Understanding Physics	DC Pandey
7	10100003	Understanding Physics	HC Verma

25). Find the student name and id who scored minimum marks in the maximum number of subjects.

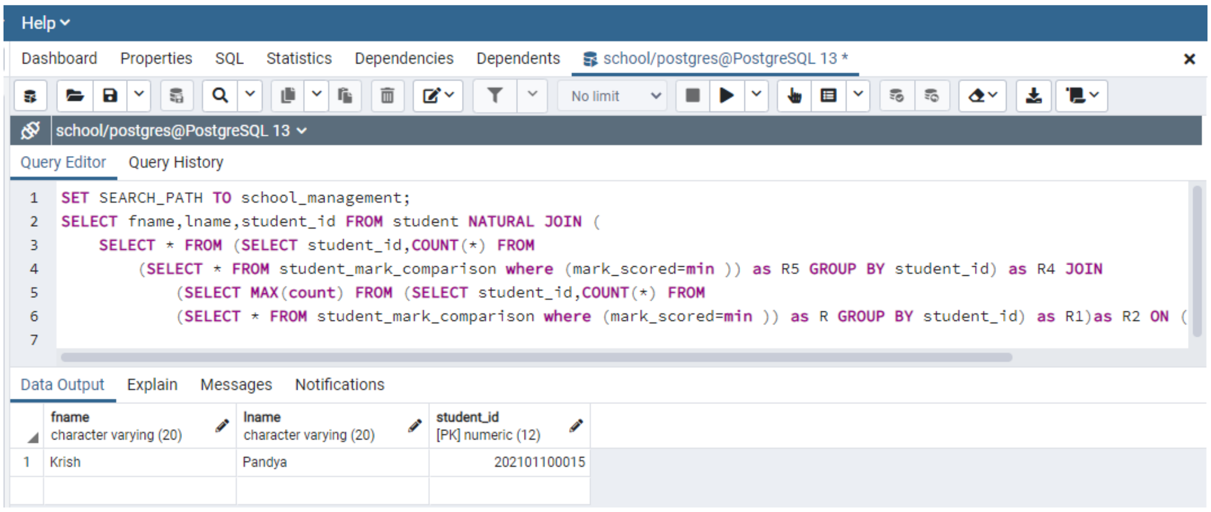
Relational Algebra:

$$\Pi_{fname,lname,student\_id}(student \bowtie ((_{student\_id}\mathcal{F}_{COUNT(*)}(\sigma_{mark\_scored=min}(student\_mark\_comparison))->R5)->R4 \bowtie_{R2.max=R4.count}(\mathcal{F}_{MAX(count)}(_{student\_id}\mathcal{F}_{COUNT(*)}(\sigma_{mark\_scored=min}(student\_mark\_comparison))->R)->R1)->R2)->R6)$$

SQL Query:
SET SEARCH_PATH TO school_management;
SELECT fname,lname,student_id FROM student NATURAL JOIN (
        SELECT * FROM (SELECT student_id,COUNT(*) FROM
                (SELECT * FROM student_mark_comparison where
(mark_scored=min)) as R5 GROUP BY student_id) as R4 JOIN
                        (SELECT MAX(count) FROM (SELECT
student_id,COUNT(*) FROM

(SELECT * FROM student_mark_comparison where (mark_scored=min)) as R GROUP BY student_id) as R1)as R2 ON (R2.max=R4.count)) as R6

Output:



```
SET SEARCH_PATH TO school_management;
SELECT fname,lname,student_id FROM student NATURAL JOIN (
    SELECT * FROM (SELECT student_id,COUNT(*) FROM
        (SELECT * FROM student_mark_comparison where (mark_scored=min )) as R5 GROUP BY student_id) as R4 JOIN
            (SELECT MAX(count) FROM (SELECT student_id,COUNT(*) FROM
            (SELECT * FROM student_mark_comparison where (mark_scored=min )) as R GROUP BY student_id) as R1)as R2 ON (
```

	fname character varying (20)	lname character varying (20)	student_id [PK] numeric (12)
1	Krish	Pandya	202101100015