

SQL QUERIES (LAB-11)

- 1) List the names of doctors from each department with highest consultation charges.

Relational Algebra:

$\pi_{D.DOCOR_NAME, D.CONSULTATION_CHARGE, D.DEPARTMENT_ID}(\rho(D, DOCTOR) \bowtie D.DEPARTMENT_ID=R.DEPARTMENT_ID \text{ AND } D.CONSULTATION_CHARGE=R.MAX_CHARGE(\rho(DEPARTMENT_ID) \mathcal{F}_{MAX(CONSULTATION_CHARGE) \rightarrow MAX_CHARGE}(DOCTOR), R)))$

SQL:

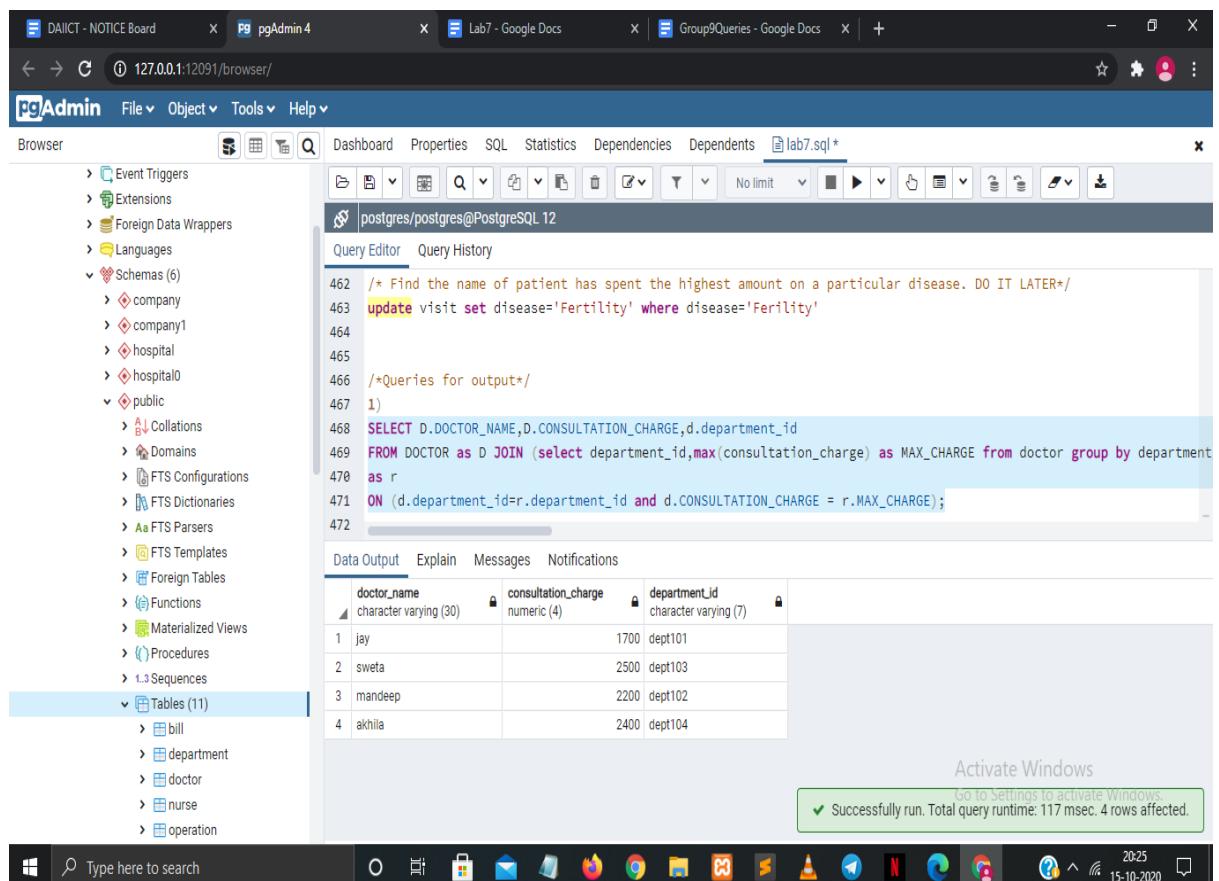
```
SELECT D.DOCTOR_NAME,D.CONSULTATION_CHARGE,d.department_id
```

FROM DOCTOR as D JOIN (select department_id,max(consultation_charge) as MAX_CHARGE from doctor group by department_id)

as r

ON (d.department_id=r.department_id and d.CONSULTATION_CHARGE = r.MAX_CHARGE);

OUTPUT:



```
/* Find the name of patient has spent the highest amount on a particular disease. DO IT LATER*/
update visit set disease='Fertility' where disease='Fertility'

/*Queries for output*/
1)
SELECT D.DOCTOR_NAME,D.CONSULTATION_CHARGE,d.department_id
FROM DOCTOR as D JOIN (select department_id,max(consultation_charge) as MAX_CHARGE from doctor group by department_id
as r
ON (d.department_id=r.department_id and d.CONSULTATION_CHARGE = r.MAX_CHARGE);
```

doctor_name	consultation_charge	department_id
jay	1700	dept101
sweta	2500	dept103
mandeep	2200	dept102
akhila	2400	dept104

2) Count the number of doctors in each department.

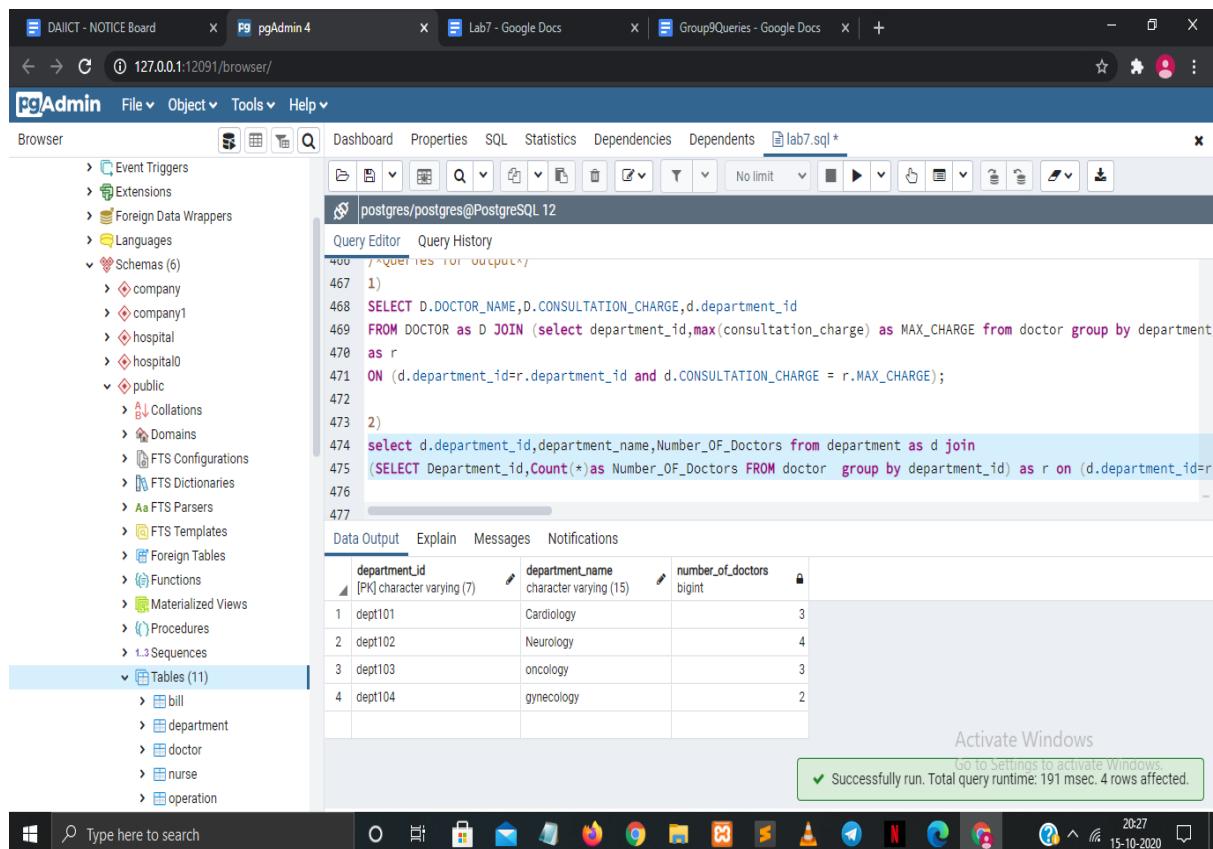
Relational Algebra:

$\pi_{D.DEPARTMENT_ID, DEPARTMENT_NAME}$,
 $\text{NUMBER_OF_DOCTORS}(\rho(D, \text{DEPARTMENT}) \bowtie_{D.DEPARTMENT_ID=R.DEPARTMENT_ID} (\rho(\text{DEPARTMENT_ID},$
 $\mathcal{F}_{\text{COUNT}(*)} \rightarrow \text{NUMBER_OF_DOCTORS}(DOCTOR), R))$

SQL:

```
select d.department_id,department_name,Number_OF_Doctors from department as d join  
(SELECT Department_id,Count(*)as Number_OF_Doctors FROM doctor group by  
department_id) as r on (d.department_id=r.department_id);
```

OUTPUT



```
467 1)  
468 SELECT D.DOCTOR_NAME,D.CONSULTATION_CHARGE,d.department_id  
469 FROM DOCTOR as D JOIN (select department_id,max(consultation_charge) as MAX_CHARGE from doctor group by department  
470 as r  
471 ON (d.department_id=r.department_id and d.CONSULTATION_CHARGE = r.MAX_CHARGE);  
472  
473 2)  
474 select d.department_id,department_name,Number_OF_Doctors from department as d join  
475 (SELECT Department_id,Count(*)as Number_OF_Doctors FROM doctor group by department_id) as r on (d.department_id=r.  
476  
477
```

department_id	department_name	number_of_doctors
dept101	Cardiology	3
dept102	Neurology	4
dept103	oncology	3
dept104	gynecology	2

Activate Windows
Go to Settings to activate Windows.
✓ Successfully run. Total query runtime: 191 msec. 4 rows affected.

3) List the details of nurses working in a particular shift.

Relational Algebra:

$\Pi^*(\sigma_{SHIFT_ID='S1'} (NURSE))$

SQL:

SELECT * from nurse where shift_id='s1';

OUTPUT:

The screenshot shows the pgAdmin 4 interface. In the top navigation bar, there are tabs for 'DAIIC - NOTICE Board', 'pgAdmin 4', 'Lab7 - Google Docs', and 'Group9Queries - Google Docs'. Below the tabs, the main window has a title bar 'pgAdmin' with dropdown menus for 'File', 'Object', 'Tools', and 'Help'. The 'File' menu is currently selected. The left sidebar is titled 'Browser' and contains a tree view of database objects, with 'Tables (11)' being the selected node. The central area is divided into two sections: 'Query Editor' and 'Data Output'. The 'Query Editor' contains the following SQL code:

```
458 SELECT d.DOCTOR_NAME,d.CONSULTATION_CHARGE,d.department_id
459 FROM DOCTOR AS D JOIN (select department_id,max(consultation_charge) as MAX_CHARGE from doctor group by department
460 as r
461 ON (d.department_id=r.department_id and d.CONSULTATION_CHARGE = r.MAX_CHARGE);
462
463 2)
464 select d.department_id,department_name,Number_OF_Doctors from department as d join
465 (SELECT Department_id,Count(*)as Number_OF_Doctors FROM doctor group by department_id) as r on (d.department_id=r
466 3)
467 SELECT * from nurse where shift_id='s1';
468
469
```

The 'Data Output' section displays the results of the query, which are the details of nurses working in shift 's1'. The table has columns: nurse_id, nurse_name, gender, date_of_birth, contact, address, and shift_id. The data is as follows:

nurse_id	nurse_name	gender	date_of_birth	contact	address	shift_id
1 N001	Natasha	F	1990-03-12	9856433387	flat-7, andheri west	s1
2 N004	Meera	F	1991-03-02	9856493387	flat-25, andheri west	s1
3 N007	Lily	F	1991-06-02	9956493387	flat-5, colaba	s1
4 N010	Riya	F	1996-08-02	9954493387	flat-5, juhu	s1
5 N013	Iksha	F	1998-03-11	9856033387	flat-7, Anushakti nagar	s1
6 N016	Diya	F	1997-08-02	9856003387	flat-25, audit bawani	s1
7 N019	Rose	F	1999-01-01	9856003387	flat-10, Andheri east	s1

A green success message at the bottom right of the data table says 'Successfully run. Total query runtime: 111 msec. 8 rows affected.'

4) List the name of doctors who have done more than 2 operations in a month.

Relational Algebra:

$\Pi_{\text{doctor}.*}(\text{doctor} \bowtie_{\text{doctor.doctor_id}=r.\text{doctor_id}} (\rho(r, \sigma_{\text{count}(*)>2 \text{ and operation_id is not null and visit_date}>='2020-10-01' \text{ and visit_date}<='2020-10-31')}(\text{doctor_id} \mathcal{F}_{\text{count}(*)}(\text{visit}))))$

SQL:

select doctor.* from doctor join

(select doctor_id from visit where operation_id is not null and visit_date>='2020-10-01' and visit_date<='2020-10-31' group by doctor_id having count(*)>2)

as r on (doctor.doctor_id=r.doctor_id);

OUTPUT:

```
472
473 2)
474 select d.department_id,department_name,Number_OF_Doctors from department as d join
475 (SELECT Department_id,Count(*)as Number_OF_Doctors FROM doctor group by department_id) as r on (d.department_id=r
476 3)
477 SELECT * from nurse where shift_id='s1';
478 4)
479 select doctor.* from doctor join
480 (select doctor_id from visit where operation_id is not null and visit_date>='2020-10-01' and visit_date<='2020-10
481 as r on (doctor.doctor_id=r.doctor_id)
482
483
```

doctor_id	doctor_name	gender	date_of_birth	contact	address	qualification
1 doc101001	alpesh	M	1975-08-26	9537507549	ahmedabad	MBBS

5) List the name of doctors working on a particular shift.

Relational Algebra:

$\sigma_{shift_id=s1}(doctor)$

SQL:

select * from doctor where shift_id='s1';

OUTPUT:

The screenshot shows the pgAdmin 4 interface with a query editor window open. The query is:

```
475 (SELECT department_id, count(*) AS number_of_doctors FROM doctor GROUP BY department_id) AS T1 UNION (SELECT department_id, count(*) AS number_of_nurses FROM nurse GROUP BY department_id) AS T2
476 3)
477 SELECT * FROM nurse WHERE shift_id='s1';
478 4)
479 SELECT doctor.* FROM doctor JOIN
480 (SELECT doctor_id FROM visit WHERE operation_id IS NOT NULL AND visit_date >= '2020-10-01' AND visit_date <= '2020-10-31') AS r ON (doctor.doctor_id=r.doctor_id)
481 AS r ON (doctor.doctor_id=r.doctor_id)
482
483 5)
484 SELECT * FROM doctor WHERE shift_id='s1'
485
486
```

The results table shows the following data:

doctor_id	doctor_name	gender	date_of_birth	contact	address	qualification
1 doc101001	alipesh	M	1975-08-26	9537507549	ahmedabad	MBBS
2 doc102004	mehul	M	1984-11-18	9990410587	South ahmedabad	BHMS
3 doc103007	sweta	F	1989-07-21	9845734985	North-West ahmedabad	BHMS
4 doc104010	mandeep	M	1967-05-19	9999127456	out of ahmedabad	BHMS

A message at the bottom right says: "Successfully run. Total query runtime: 135 msec. 4 rows affected."

6) Given the doctor's id, retrieve the details of patients he has treated so far.

Relational Algebra:

$$\pi_{\text{distinct } P.PATIENT_ID, P.PATIENT_NAME, GENDER, Blood_Group} (\sigma_{\text{DOCTOR_ID} = \text{'DOC101001'}} (\rho(P, PATIENT) \bowtie P.PATIENT_ID = VISIT.PATIENT_ID (VISIT)))$$

SQL:

```
SELECT DISTINCT p.PATIENT_ID, p.PATIENT_NAME, GENDER, Blood_Group
  from patient as p
 join visit on (p.patient_id=visit.patient_id)
 where doctor_id='doc101001';
```

OUTPUT:

The screenshot shows the pgAdmin 4 interface with a query editor window open. The query is as follows:

```
477 SELECT * FROM nurse WHERE shift_id='s1';
478 4)
479 select doctor.* from doctor join
480 (select doctor_id from visit where operation_id is not null and visit_date>='2020-10-01' and visit_date<='2020-10-01'
481 as r on (doctor.doctor_id=r.doctor_id)
482
483 5)
484 select * from doctor WHERE shift_id='s1'
485 6)
486 SELECT DISTINCT p.PATIENT_ID, p.PATIENT_NAME, GENDER, Blood_Group
  from patient as p
 join visit on (p.patient_id=visit.patient_id)
 where doctor_id='doc101001';
```

The results are displayed in a table:

patient_id	patient_name	gender	blood_group
1	sambhav	M	A-
2	sejal	O	AB-
3	khush	M	O+
4	priyanka	F	A+
5	kajal	F	AB+
6	payal	F	B+
7	amit	M	A-

A message at the bottom right indicates: "Successfully run. Total query runtime: 165 msec. 7 rows affected."

7) Retrieve details of nurses assigned to a particular room.

Relational Algebra:

$\sigma_{\text{room_id}=101}(\text{nurse})$

SQL:

select * from nurse where room_id='101';

OUTPUT:

The screenshot shows the pgAdmin 4 interface. The left sidebar displays various database objects like Event Triggers, Extensions, Foreign Data Wrappers, Languages, Schemas, and Tables. The 'Tables' section is currently selected, showing 11 tables: bill, department, doctor, nurse, and operation. The main window contains a query editor with the following SQL code:

```
479 select * from nurse where room_id='101'
```

The results pane shows the following data:

nurse_id	nurse_name	gender	date_of_birth	contact	address	shift_id
1 N001	Natasha	F	1990-03-12	9856433387	flat-7, andheri west	s1
2 N002	Asha	F	1992-09-02	9856483387	flat-9, andheri east	s2
3 N003	Shankar	M	1990-03-12	8856433387	flat-17, goregaon	s3

At the bottom right of the results pane, there is a message: "Successfully run. Total query runtime: 130 msec. 3 rows affected."

8) Count the number of patients preferring a particular category of room.(to understand which room category is preferred so the infrastructure of rooms in hospital is adjusted accordingly)

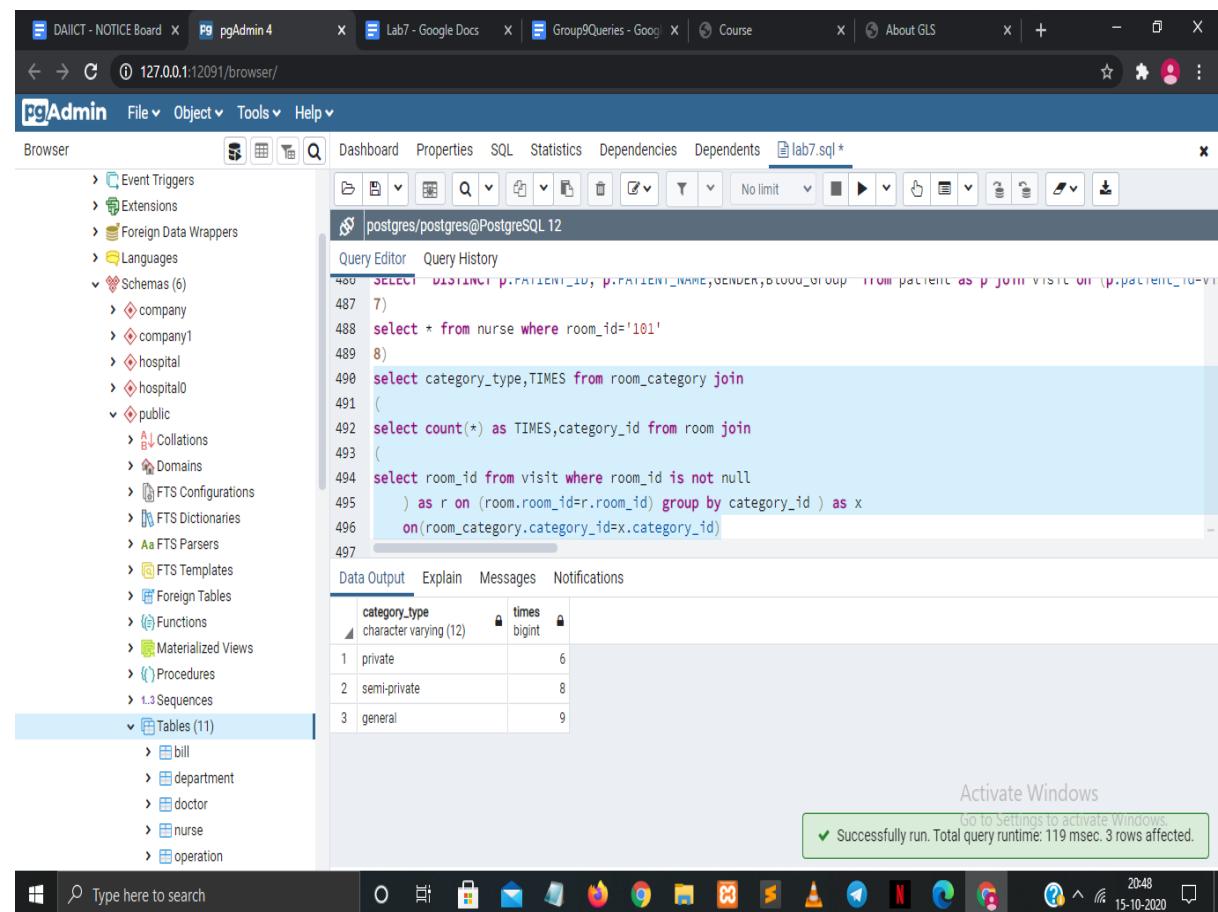
Relational Algebra:

$$\pi_{\text{category_type}, \text{times}} \\ (\rho(\text{room_category}) \bowtie_{\text{room_category.category_id} = x.\text{category_id}} (\rho(\mathcal{F}_{\text{COUNT}(*)} \rightarrow \text{times}, \text{category_id}(x, \text{room})) \bowtie_{\text{room.room_id} = r.\text{room_id}} (\rho(\sigma_{\text{room_id} \text{ not null}}(r, \text{visit}))))$$

SQL:

```
select category_type, TIMES from room_category join
(
  select count(*) as TIMES, category_id from room join
  (
    select room_id from visit where room_id is not null
    ) as r on (room.room_id=r.room_id) group by category_id ) as x
  on(room_category.category_id=x.category_id);
```

OUTPUT:



The screenshot shows the pgAdmin 4 interface with a query editor window open. The code in the editor is:

```
480 SELECT DISTINCT p.PATIENT_ID, p.PATIENT_NAME, GENDER, BLOOD_GROUP FROM patient AS p JOIN VISIT ON (p.PATIENT_ID = v.PATIENT_ID)
481 )
482 select * from nurse where room_id='101'
483 )
484 select category_type, TIMES from room_category join
485 (
486   select count(*) as TIMES, category_id from room join
487   (
488     select room_id from visit where room_id is not null
489     ) as r on (room.room_id=r.room_id) group by category_id ) as x
490   on(room_category.category_id=x.category_id)
491
492
493
494
495
496
497
```

The results table below the code shows the following data:

category_type	times
private	6
semi-private	8
general	9

A green message bar at the bottom right indicates: "Successfully run. Total query runtime: 119 msec. 3 rows affected."

9) From date A to B retrieve the data of visited patients.

Relational Algebra:

$\pi_{VISIT_ID, PATIENT_NAME}(\rho(P, PATIENT) \bowtie$

$P.PATIENT_ID=R.PATIENT_ID \rho(R, \pi_{VISIT_ID, PATIENT_NAME}(\sigma_{visit_date \geq '2020/10/01' \text{ and } visit_date \leq '2030/02/27'}(VISIT)))$

SQL:

SELECT visit_id, PATIENT_name FROM PATIENT as p JOIN

(select patient_id, visit_id from visit where visit_date >= '2020/10/01' and visit_date <= '2030/02/27')

as r ON (p.patient_id=r.patient_id);

OUTPUT:

The screenshot shows the pgAdmin 4 interface with the following details:

- Query Editor:** Contains the SQL query:

```
492 select count(*) as TIMES,category_id from room
493 (
494 select room_id from visit where room_id is not null
495 ) as r on (room.room_id=r.room_id) group by category_id ) as x
496 on(room_category.category_id=x.category_id)
497
498 9)
499 SELECT visit_id,PATIENT_name FROM PATIENT as p JOIN
500 (select patient_id,visit_id from visit where visit_date >= '2020/10/01' and visit_date <= '2030/02/27')
501 as r ON (p.patient_id=r.patient_id);
502
503
```
- Data Output:** Displays the results of the query in a table:

visit_id	patient_name
1	4 parshva
2	3 parshva
3	2 parshva
4	8 abid
5	6 abid
6	12 sambhav
7	11 sambhav
8	?? which

- Status Bar:** Shows "Activate Windows" and "Successfully run. Total query runtime: 164 msec. 20 rows affected."

10) Get the admit history of a particular patient.

Relational Algebra:

$\pi_{P.PATIENT_ID, P.PATIENT_NAME, GENDER, VISIT_DATE, DISCHARGE_DATE, ROOM_ID}(P \bowtie_{P.PATIENT_ID=R.PATIENT_ID} R \sigma_{PATIENT_ID=XXX \text{ AND } ROOM_ID \text{ IS NOT NULL}} (VISIT))$

SQL:

```
SELECT p.PATIENT_ID,p.PATIENT_NAME,GENDER,visit_date,discharge_date,room_id  
FROM PATIENT as p JOIN
```

```
(select patient_id,visit_date,discharge_date,room_id from visit where patient_id=1 and  
room_id is not null)
```

```
as r ON (p.patient_id=r.patient_id);
```

OUTPUT:

The screenshot shows the pgAdmin 4 interface with a query editor window titled 'lab7.sql'. The query is as follows:

```
497 9)  
498 SELECT visit_id,PATIENT_name FROM PATIENT as p JOIN  
499 (select patient_id,visit_id from visit where visit_date >= '2020/10/01' and visit_date <= '2030/02/27')  
500 as r ON (p.patient_id=r.patient_id);  
501  
502 10)  
503 SELECT p.PATIENT_ID,p.PATIENT_NAME,GENDER,visit_date,discharge_date,room_id FROM PATIENT as p JOIN  
504 (select patient_id,visit_date,discharge_date,room_id from visit where patient_id=1 and room_id is not null)  
505 as r ON (p.patient_id=r.patient_id);  
506  
507  
508
```

The results of the query are displayed in a Data Output tab, showing one row:

patient_id	patient_name	gender	visit_date	discharge_date	room_id
1	1 harsh	M	2020-01-01	2020-03-10	102

A status bar at the bottom right indicates: 'Successfully run. Total query runtime: 117 msec. 1 rows affected.'

11) List of medicines prescribed to a patient on a particular visit.

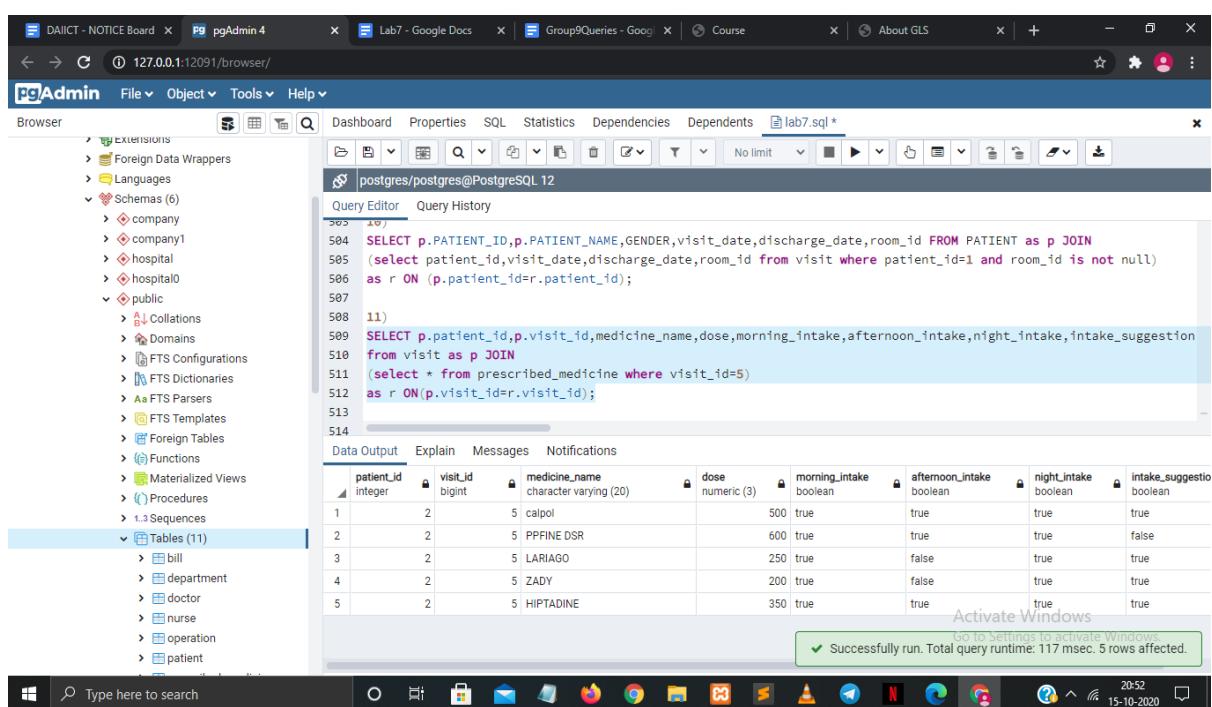
Relational Algebra:

$\pi_{P.PATIENT_ID, P.VISIT_ID, MEDICINE_NAME, DOSE, MORNING_INTAKE, AFTERNOON_INTAKE, NIGHT_INTAKE, INTAKE_SUGGESTION} (\rho(P, VISIT) \bowtie P.VISIT_ID=R.VISIT_ID \rho(R, \sigma_{VISIT_ID=X} (PRESCRIBED_MEDICINE)))$

SQL:

```
SELECT
p.patient_id,p.visit_id,medicine_name,dose,morning_intake,afternoon_intake,night_intake,intake_suggestion
from visit as p JOIN
(select * from prescribed_medicine where visit_id=1)
as r on (p.visit_id=r.visit_id);
```

OUTPUT:



```
504 SELECT p.PATIENT_ID,p.PATIENT_NAME,GENDER,visit_date,discharge_date,room_id FROM PATIENT as p JOIN
505 (select patient_id,visit_date,discharge_date,room_id from visit where patient_id=1 and room_id is not null)
506 as r ON (p.patient_id=r.patient_id);
507
508 11)
509 SELECT p.patient_id,p.visit_id,medicine_name,dose,morning_intake,afternoon_intake,night_intake,intake_suggestion
510 from visit as p JOIN
511 (select * from prescribed_medicine where visit_id=5)
512 as r ON(p.visit_id=r.visit_id);
513
514
```

patient_id	visit_id	medicine_name	dose	morning_intake	afternoon_intake	night_intake	intake_suggestion
1	2	5 calpol	500	true	true	true	true
2	2	5 PPPINE DSR	600	true	true	true	false
3	2	5 LARIAGO	250	true	false	true	true
4	2	5 ZADY	200	true	false	true	true
5	2	5 HIPTADINE	350	true	true	true	true

12) List of the disease of a particular patient in previous visits.

Relational Algebra:

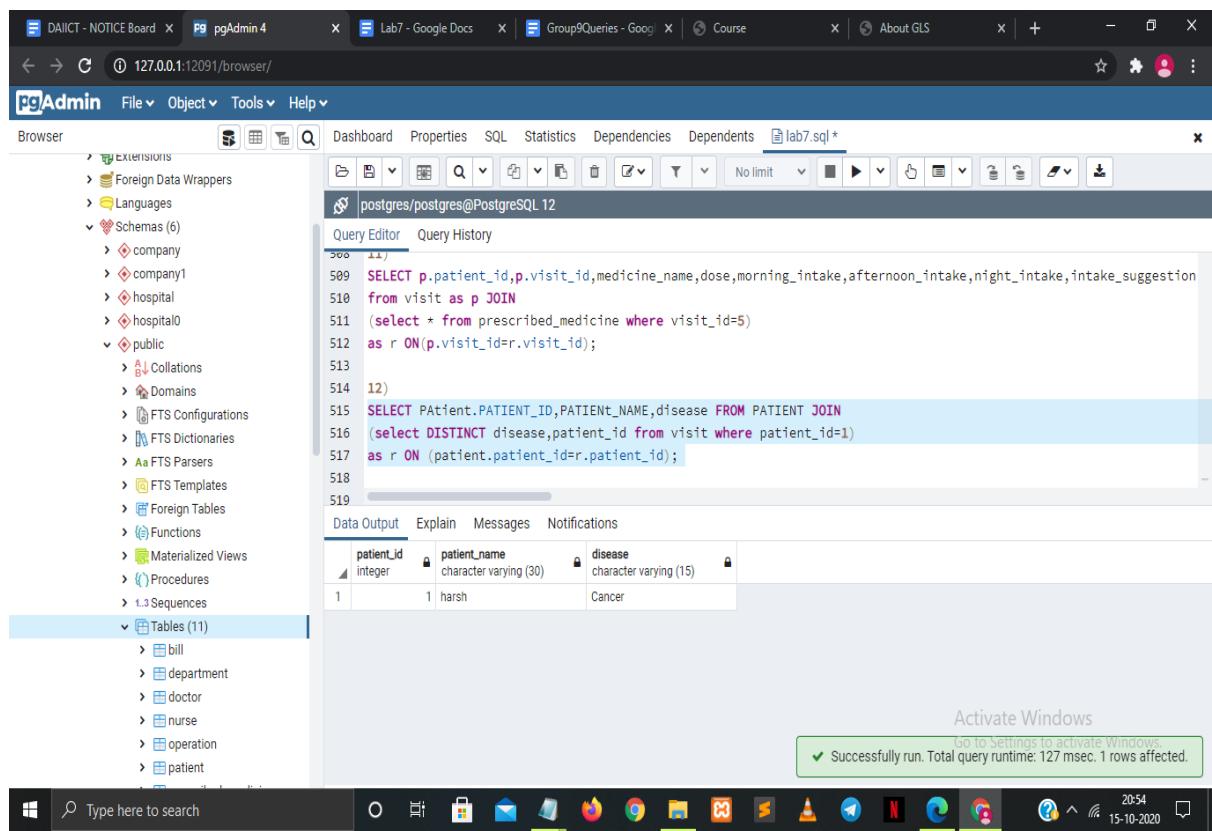
$\pi_{\text{patient.patient_id}, \text{patient_name}, \text{disease}}(\rho(r, \sigma_{\text{patient_id}=1}(\text{visit})))$

$\bowtie_{r.\text{patient_id}=\text{patient.patient_id}} \text{patient}$

SQL:

```
SELECT PATIENT.PATIENT_ID, PATIENT_NAME, disease FROM PATIENT JOIN
(select DISTINCT disease, patient_id from visit where patient_id=1)
as r ON (patient.patient_id=r.patient_id);
```

OUTPUT:



```
509 SELECT p.patient_id,p.visit_id,medicine_name,dose,morning_intake,afternoon_intake,night_intake,intake_suggestion
510 from visit as p JOIN
511 (select * from prescribed_medicine where visit_id=5)
512 as r ON(p.visit_id=r.visit_id);
513
514 12)
515 SELECT PATIENT.PATIENT_ID, PATIENT_NAME, disease FROM PATIENT JOIN
516 (select DISTINCT disease, patient_id from visit where patient_id=1)
517 as r ON (patient.patient_id=r.patient_id);
518
519
```

patient_id	patient_name	disease
1	harsh	Cancer

Activate Windows
Go to Settings to activate Windows.
✓ Successfully run. Total query runtime: 127 msec. 1 rows affected.

13) Show the details of patients treated by a particular doctor.

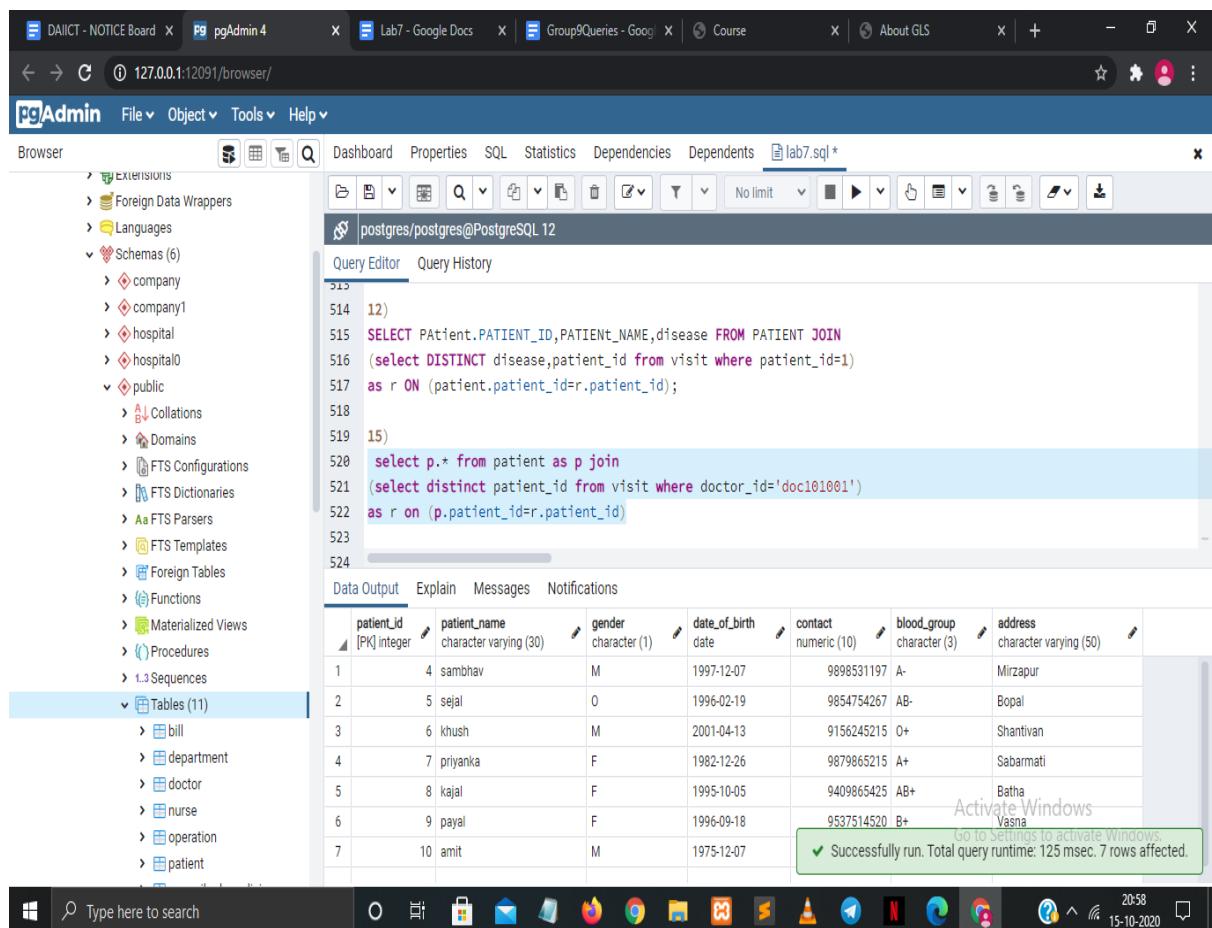
Relational Algebra:

$$\Pi * (\rho(p, \text{patient}) \bowtie_{p.\text{patient_id}=r.\text{patient_id}} \rho(r, \Pi \text{ distinct patient_id} (\sigma_{\text{doctor_id}='doc101001'} (\text{visit})))$$

SQL:

```
select p.* from patient as p join
(select distinct patient_id from visit where doctor_id='doc101001')
as r on (p.patient_id=r.patient_id);
```

OUTPUT:



The screenshot shows the pgAdmin 4 interface with the following details:

- Query Editor:** Contains the SQL query:

```
12)
SELECT PATIENT.PATIENT_ID,PATIENT_NAME,disease FROM PATIENT JOIN
(select DISTINCT disease,patient_id from visit where patient_id=1)
as r ON (patient.patient_id=r.patient_id);

15)
select p.* from patient as p join
(select distinct patient_id from visit where doctor_id='doc101001')
as r on (p.patient_id=r.patient_id)
```
- Data Output:** Displays the results of the query in a table format:

patient_id	patient_name	gender	date_of_birth	contact	blood_group	address
1	sambhav	M	1997-12-07	9898531197	A-	Mirzapur
2	sejal	O	1996-02-19	9854754267	AB-	Bopal
3	khush	M	2001-04-13	9156245215	O+	Shantivan
4	priyanka	F	1982-12-26	9879865215	A+	Sabarmati
5	kajal	F	1995-10-05	9409865425	AB+	Batha
6	payal	F	1996-09-18	9537514520	B+	Vasna
7	amit	M	1975-12-07			

- Status Bar:** Shows "Activate Windows" and "Successfully run. Total query runtime: 125 msec. 7 rows affected."

14) Show the details of patients operated by a particular doctor.

Relational Algebra:

$\Pi_{p.*}(\rho(p, \text{patient}) \bowtie_{p.\text{patient_id}=r.\text{patient_id}} \rho(r, \Pi_{\text{distinct patient_id}} \sigma_{\text{doctor_id}='doc101001' \text{ and } \text{operation_id} \text{ is not null}(\text{visit}))$

SQL:

SELECT p.* FROM patient as p JOIN

(SELECT DISTINCT patient_id FROM visit where doctor_id='doc101001' and operation_id is not null)

as r ON (p.patient_id=r.patient_id);

OUTPUT:

The screenshot shows the pgAdmin 4 interface with a query editor window open. The query is as follows:

```
517 as t1 on (patient.patient_id=r.patient_id),
518 15)
519 select p.* from patient as p join
520 (select distinct patient_id from visit where doctor_id='doc101001')
521 as r on (p.patient_id=r.patient_id)
522 16)
523 select p.* from patient as p join
524 (select distinct patient_id from visit where doctor_id='doc101001' and operation_id is not null)
525 as r on (p.patient_id=r.patient_id)
526
527
528
```

The results are displayed in a table:

patient_id	patient_name	gender	date_of_birth	contact	blood_group	address
1	4 sambhav	M	1997-12-07	9898531197	A-	Mirzapur
2	6 khush	M	2001-04-13	9156245215	0+	Shantivan
3	7 priyanka	F	1982-12-26	9879865215	A+	Sabarmati
4	8 kajal	F	1995-10-05	9409865425	AB+	Batha
5	10 amit	M	1975-12-07	9898846781	A-	unknown bunglow near vs_h...

A message at the bottom right of the results pane says "Successfully run. Total query runtime: 200 msec. 5 rows affected."

15) Count the number of visits of a particular patient as per his disease.

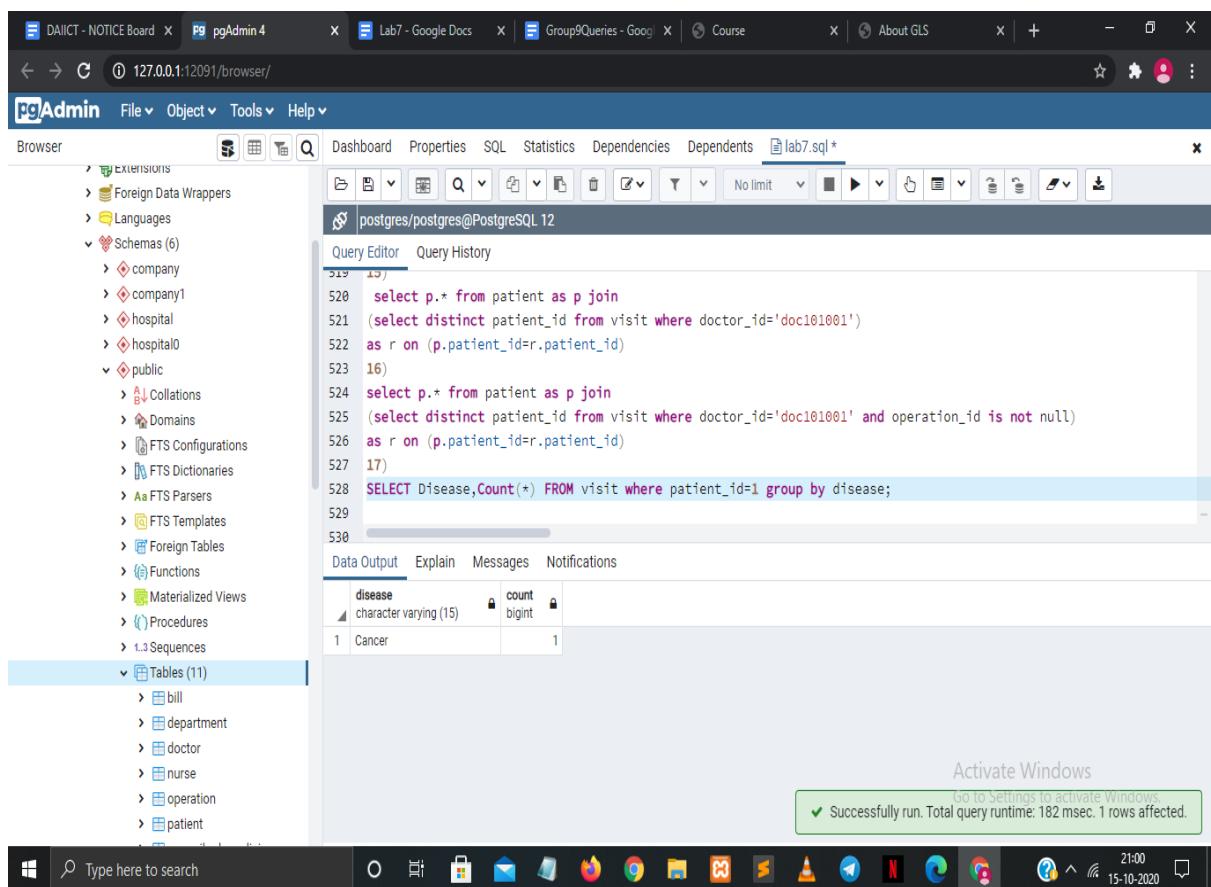
Relational Algebra:

disease $\mathcal{F}_{\text{count}(*)} (\sigma_{\text{patient_id}=1}(\text{visit}))$

SQL:

SELECT Disease,Count(*) FROM visit where patient_id=1 group by disease;

OUTPUT:



The screenshot shows the pgAdmin 4 interface with a query editor window open. The query is:

```
519
520 select p.* from patient as p join
521 (select distinct patient_id from visit where doctor_id='doc101001')
522 as r on (p.patient_id=r.patient_id)
523 (16)
524 select p.* from patient as p join
525 (select distinct patient_id from visit where doctor_id='doc101001' and operation_id is not null)
526 as r on (p.patient_id=r.patient_id)
527 (17)
528 SELECT Disease,Count(*) FROM visit where patient_id=1 group by disease;
529
530
```

The results table shows:

disease	count
Cancer	1

A message at the bottom right says: "Successfully run. Total query runtime: 182 msec. 1 rows affected."

16) List the details of the doctor working in a particular department and shift.

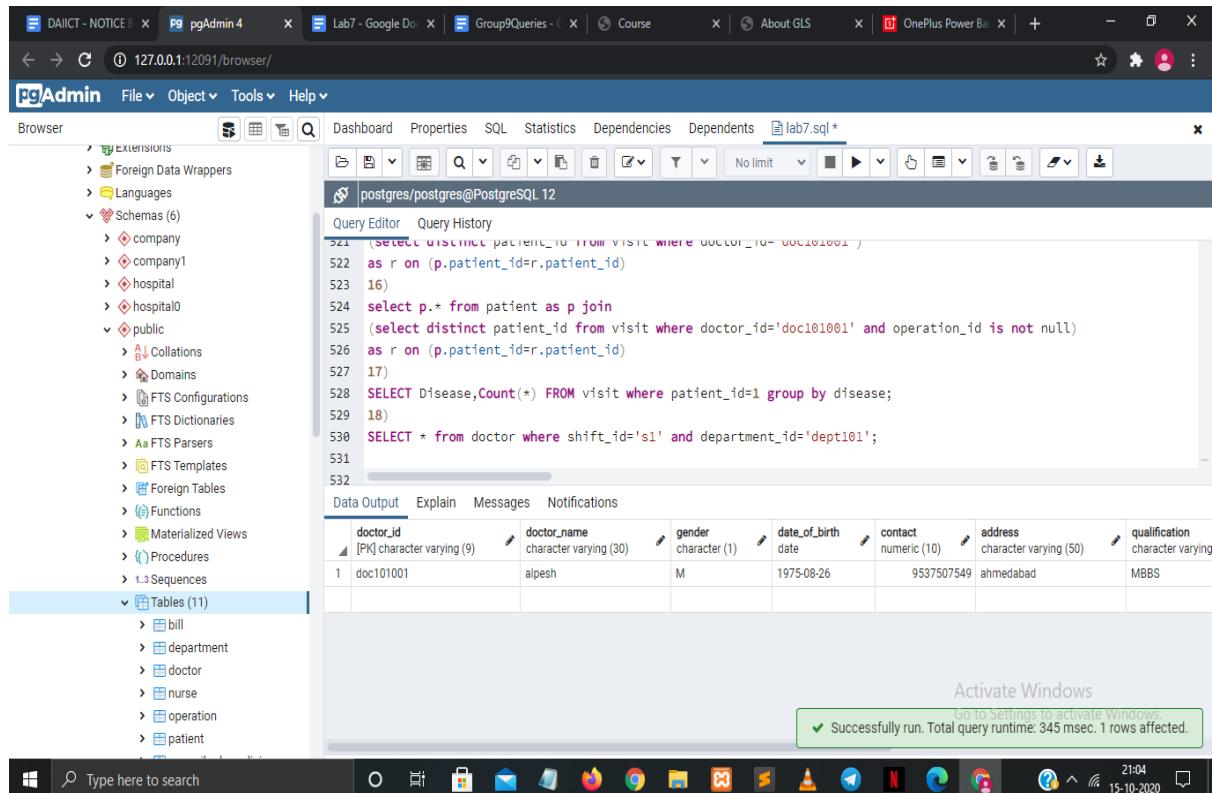
Relational Algebra:

$\Pi^*(\sigma_{SHIFT_ID='S1' \text{ AND } DEPARTMENT_ID='DEPT101'}(DOCTOR))$

SQL:

SELECT * from doctor where shift_id='s1' and department_id='dept101';

OUTPUT:



The screenshot shows the pgAdmin 4 interface. The left sidebar displays various database objects like extensions, schemas, and tables. The main area has a query editor window with the following SQL code:

```
521 (select distinct patient_id from visit where doctor_id='doc101001')
522 as r on (p.patient_id=r.patient_id)
523 16)
524 select p.* from patient as p join
525 (select distinct patient_id from visit where doctor_id='doc101001' and operation_id is not null)
526 as r on (p.patient_id=r.patient_id)
527 17)
528 SELECT Disease,Count(*) FROM visit where patient_id=1 group by disease;
529 18)
530 SELECT * from doctor where shift_id='s1' and department_id='dept101';
531
532
```

Below the query editor is a data output table with one row of results:

doctor_id	doctor_name	gender	date_of_birth	contact	address	qualification
1 doc101001	alipesh	M	1975-08-26	9537507549	ahmedabad	MBBS

A status bar at the bottom right indicates: "Activate Windows", "Go to Settings to activate Windows.", "Successfully run. Total query runtime: 345 msec. 1 rows affected.", "21:04", "15-10-2020".

17) Get the revenue generated in a particular month.

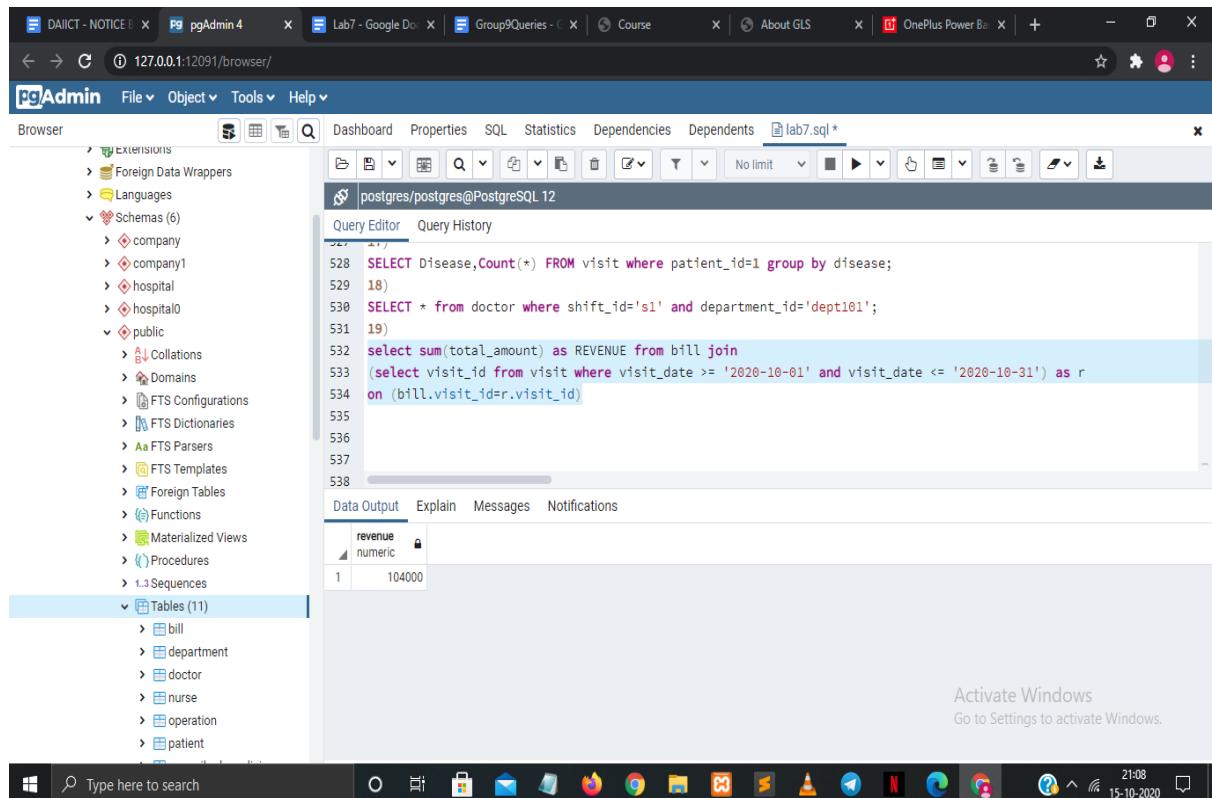
Relational Algebra:

$\mathcal{F}_{\text{sum}(\text{total_amount}) \rightarrow \text{revenue}}(\text{bill} \bowtie_{\text{bill.visit_id} = \text{r.visit_id}} \rho(\text{r}, \Pi_{\text{visit_id}}(\sigma_{\text{visit_date} \geq 2020-10-01 \text{ and } \text{visit_date} \leq 2020-10-31}(\text{visit})))$

SQL:

```
SELECT SUM(total_amount) as REVENUE FROM bill JOIN
(SELECT visit_id FROM visit where visit_date >= '2020-10-01' and visit_date <= '2020-10-31') as r
ON (bill.visit_id=r.visit_id);
```

OUTPUT:



The screenshot shows the pgAdmin 4 interface with a query editor window open. The query is as follows:

```
528 SELECT Disease,Count(*) FROM visit where patient_id=1 group by disease;
529 18)
530 SELECT * from doctor where shift_id='s1' and department_id='dept101';
531 19)
532 select sum(total_amount) as REVENUE from bill join
533 (select visit_id from visit where visit_date >= '2020-10-01' and visit_date <= '2020-10-31') as r
534 on (bill.visit_id=r.visit_id)
```

The results pane shows a single row of data:

revenue	numeric
1	104000

18) Get the last prescription of a particular patient.

Relational Algebra:

$\Pi_{p.*}(\rho(p, \text{prescribed_medicine}) \bowtie_{p.visit_id=r.last_visit} \rho(r, \sigma_{\text{patient_id}=2}(\mathcal{F}_{\max(\text{visit_id}) \rightarrow \text{last_visit}}(\text{visit}))))$

SQL:

```
SELECT p.* FROM prescribed_medicine as p JOIN
(SELECT MAX(visit_id) as last_visit FROM visit where patient_id=2 ) as r
ON (p.visit_id=r.last_visit);
```

OUTPUT:

```
19)
530 SELECT * from doctor where shift_id='s1' and department_id='dept101';
531 19)
532 select sum(total_amount) as REVENUE from bill join
533 (select visit_id from visit where visit_date >= '2020-10-01' and visit_date <= '2020-10-31') as r
534 on (bill.visit_id=r.visit_id)
535 20)
536 select p.* from prescribed_medicine as p join
537 (select max(visit_id) as last_visit from visit where patient_id=2 ) as r
538 on (p.visit_id=r.last_visit)
539
540
```

visit_id	medicine_name	dose	morning_intake	afternoon_intake	night_intake	intake_suggestion
1	5 calpol	500	true	true	true	true
2	5 PFFINE DSR	600	true	true	true	false
3	5 LARIAGO	250	true	false	true	true
4	5 ZADY	200	true	false	true	true
5	5 HIPTADINE	350	true	true	true	true

Activate Windows
Go to Settings to activate Windows
✓ Successfully run. Total query runtime: 158 msec. 5 rows affected.

19) Total revenue contributed by a doctor in a financial year.

Relational Algebra:

$\mathcal{F} \text{ sum(consultation_charges)} \rightarrow \text{doctor_contribution(bill)} \bowtie_{\text{bill.visit_id} = \text{r.visit_id}} \rho(\text{r}, \Pi_{\text{visit_id}}(\sigma_{\text{doctor_id} = \text{'doc101001'}} \text{ and } \text{visit_date} \geqslant \text{'2020-04-01'} \text{ and } \text{visit_date} \leqslant \text{'2021-03-31'}(\text{visit})))$

SQL:

```
SELECT SUM(consultation_charges) as Doctor_Contribution FROM bill JOIN  
(SELECT visit_id FROM visit where doctor_id = 'doc101001' and visit_date >= '2020-04-01'  
and visit_date <= '2021-03-31')as r  
ON (bill.visit_id=r.visit_id);
```

OUTPUT:

The screenshot shows the pgAdmin 4 interface with multiple browser tabs open. The main window displays a query editor with the following SQL code:

```
550 ) as r
551 on(room.room_id=r.room_id) as x on (room_category.category_id=x.category_id)
552
553 22)
554 select sum(consultation_charges) as Doctor_Contribution from bill join
555 (select visit_id from visit where doctor_id = 'doc1@1001' and visit_date >= '2020-04-01' and visit_date <= '2021-
556 as r on (bill.visit_id=r.visit_id)
557
558
559
560
```

The Data Output tab shows the result of the query:

doctor_contribution
10500

The left sidebar shows the database schema with tables like bill, department, doctor, nurse, operation, and patient.

20) Count the number of particular operations in a year.

Relational Algebra:

$\pi_{\text{operation_type}, \text{times}} (\text{operation} \bowtie \text{operation.operation_id} = \text{r.operation_id} \rho(\text{r}, \sigma_{\text{operation_id} \text{ not null} \text{ and } \text{visit_date} \geq '2020-04-01' \text{ and } \text{visit_date} \leq '2021-03-31'} (\text{operation_id} \mathcal{F} \text{ count(*)} \rightarrow \text{times} (\text{visit}))))$

SQL:

```
SELECT Operation_type, TIMES FROM operation JOIN
(SELECT count(*) as TIMES, operation_id FROM visit WHERE operation_id is not null and
visit_date >= '2020-04-01' and visit_date <= '2021-03-31' group by operation_id) as r
ON (operation.operation_id=r.operation_id);
```

OUTPUT:

The screenshot shows the pgAdmin 4 interface with a query editor window open. The query is as follows:

```
553 22)
554 select sum(consultation_charges) as Doctor_Contribution from bill join
555 (select visit_id from visit where doctor_id = 'doc101001' and visit_date >= '2020-04-01' and visit_date <= '2021-
556 as r on (bill.visit_id=r.visit_id)
557
558 23)
559 select Operation_type, TIMES from operation join
560 (
561 select count(*) as TIMES, operation_id from visit WHERE operation_id is not null and visit_date >= '2020-04-01' and
562 group by operation_id) as r on (operation.operation_id=r.operation_id)
563
564
```

The results are displayed in a table:

operation_type	times
Anangioplasty	1
Pacemaker Implant	1
craniotomy	3
Aneurysm Repair	5
Bowel resection	2
chemotherapy	7

A message at the bottom right indicates: "Successfully run. Total query runtime: 164 msec. 6 rows affected."

21) Count number of operations done by doctors in a year.

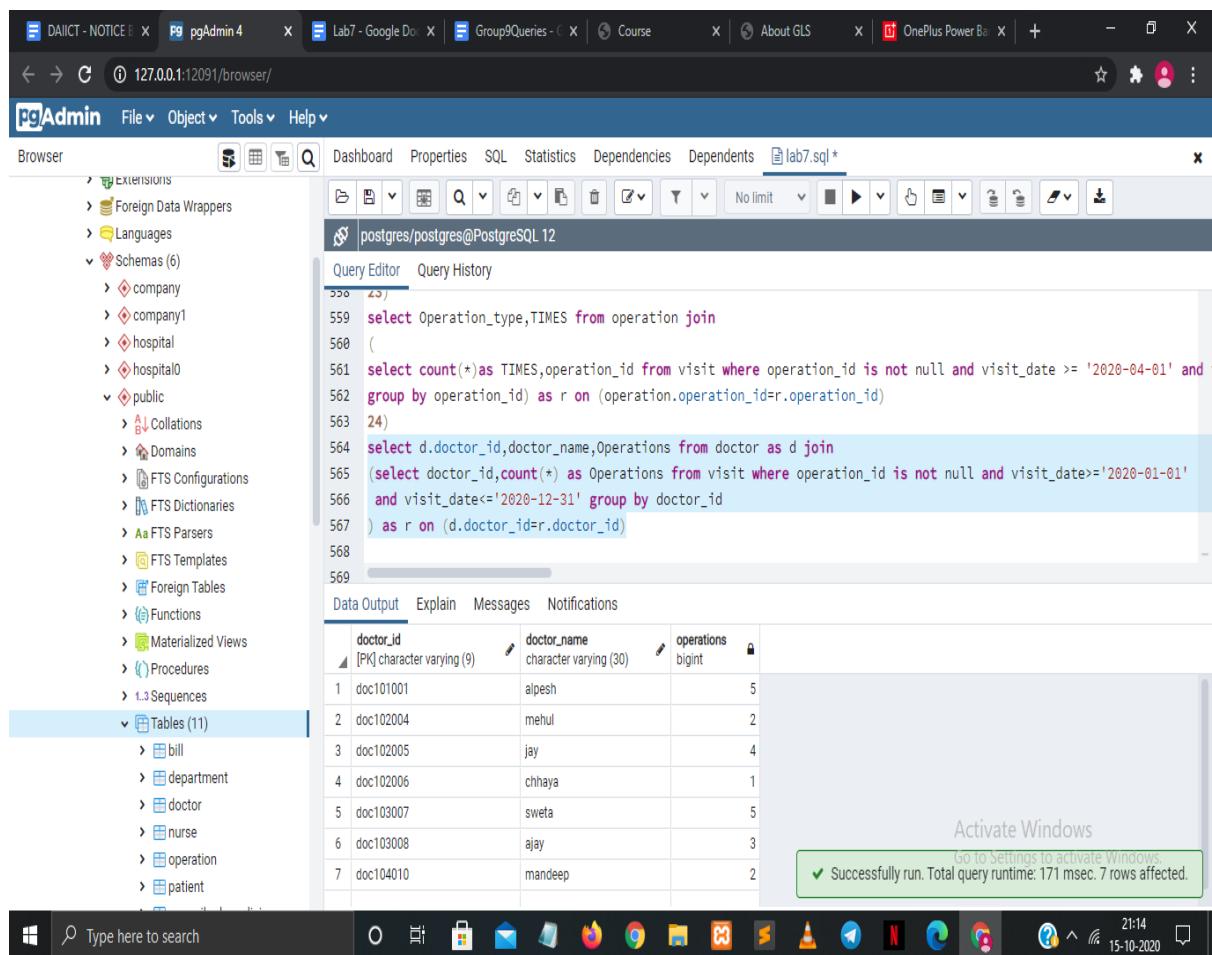
Relational Algebra:

$$\Pi_{d.\text{doctor_id}, d.\text{doctor_name}, \text{Operations}}(\rho(d, \text{doctor}) \bowtie_{d.\text{doctor_id}=r.\text{doctor_id}} \rho(r, \sigma_{\text{operation_id} \text{ not null} \text{ and } \text{visit_date} \geq= 2020-01-01 \text{ and } \text{visit_date} \leq= 2020-12-31}(\text{doctor_id} \mathcal{F}_{\text{count}(*)} \rightarrow \text{operations}(\text{visit}))))$$

SQL:

```
select d.doctor_id, doctor_name, Operations from doctor as d join
(select doctor_id, count(*) as Operations from visit where operation_id is not null and
visit_date >='2020-01-01' and visit_date <='2020-12-31' group by doctor_id ) as r
on (d.doctor_id=r.doctor_id);
```

OUTPUT:



```
558 559 select Operation_type, TIMES from operation join
560 (
561 select count(*) as TIMES, operation_id from visit where operation_id is not null and visit_date >= '2020-01-01' and
562 group by operation_id) as r on (operation.operation_id=r.operation_id)
563 24)
564 select d.doctor_id, doctor_name, Operations from doctor as d join
565 (select doctor_id, count(*) as Operations from visit where operation_id is not null and visit_date >= '2020-01-01'
566 and visit_date <= '2020-12-31' group by doctor_id
567 ) as r on (d.doctor_id=r.doctor_id)
568
569
```

doctor_id	doctor_name	operations
1 doc101001	ajpesh	5
2 doc102004	mehul	2
3 doc102005	jay	4
4 doc102006	chhaya	1
5 doc103007	sweta	5
6 doc103008	ajay	3
7 doc104010	mandeep	2

22) Count the number of patients who took appointments with at least two different physicians.

Relational Algebra:

$$\Pi_{\text{patient}.*}(\text{patient}) \bowtie_{\text{patient.patient_id}=r.\text{patient_id}} \rho(r, \sigma_{\text{count}(*)>1}(\text{patient_id} \not\in \text{F distinct count}(*) \text{ (visit)}))$$

SQL:

```
SELECT patient.* FROM patient JOIN
(SELECT DISTINCT count(*),patient_id FROM visit GROUP BY patient_id having
count(*)>1) as r
ON (patient.patient_id=r.patient_id) ;
```

OUTPUT:

The screenshot shows the pgAdmin 4 interface with the following details:

- Query Editor:** Contains the SQL query for the question.
- Data Output:** Shows the results of the query, which are 9 rows of patient information.
- Table:** Shows the structure of the patient table with columns: patient_id, patient_name, gender, date_of_birth, contact, blood_group, address.
- Messages:** A green message at the bottom right indicates the query was successfully run with a runtime of 224 msec and 9 rows affected.

patient_id	patient_name	gender	date_of_birth	contact	blood_group	address
1	parshva	M	1998-10-05	9409805584	AB+	Paldi
2	abid	M	1999-09-18	9537502845	B+	mirzapur
3	sambhav	M	1997-12-07	9898531197	A-	Mirzapur
4	sejal	O	1996-02-19	9854754267	AB-	Bopal
5	khush	M	2001-04-13	9156245215	O+	Shantivan
6	priyanka	F	1982-12-26	9879865215	A+	Sabarmati
7	kajal	F	1995-10-05			
8	naval	F	1000-00-00	09375114507	R+	Vaenna

23) Obtain the names of all patients whose operation is done by a doctor who is head of a department.

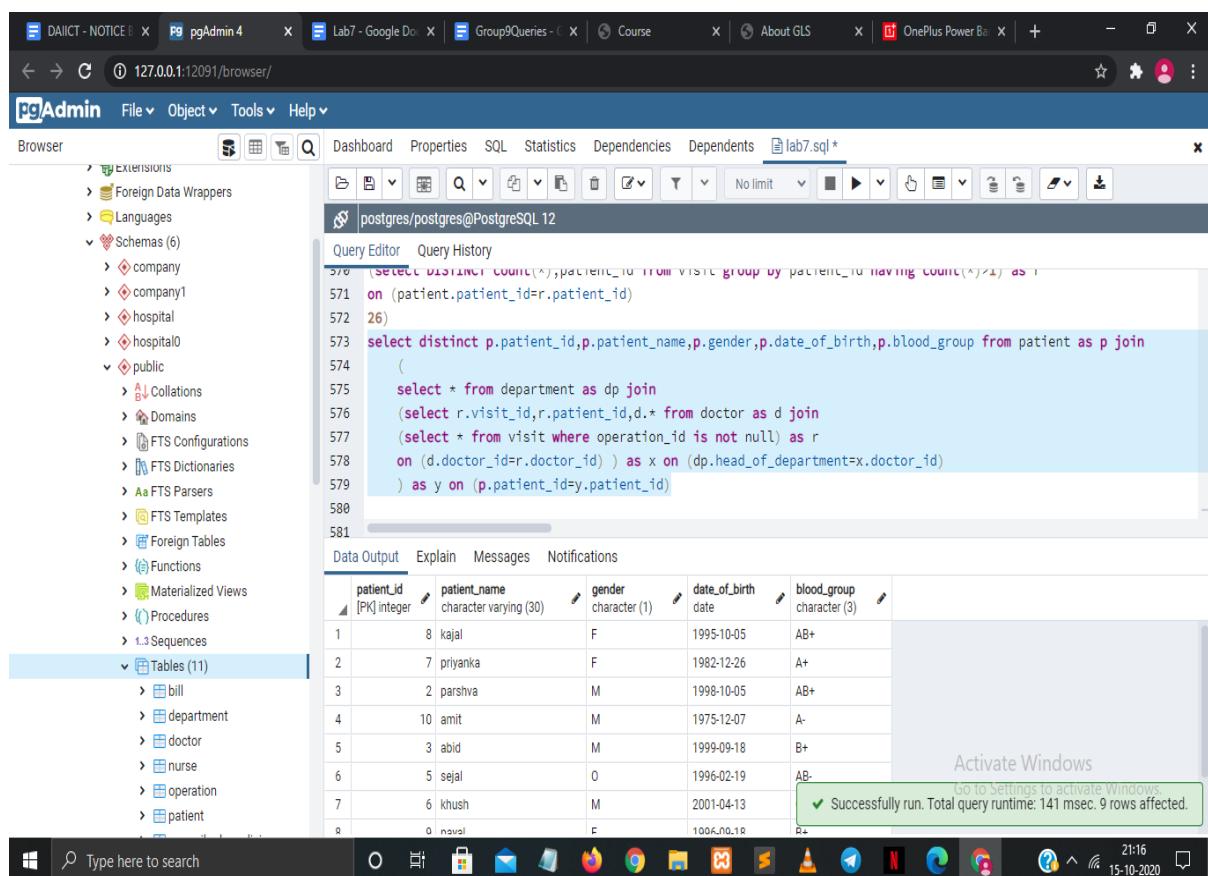
Relational Algebra:

$$\begin{aligned} & \Pi_{\text{distinct } p} (p.\text{patient_id}, p.\text{patient_name}, p.\text{gender}, p.\text{date_of_birth}, p.\text{blood_group}) (\rho(p, \text{patient}) \bowtie_{p.\text{patient_id}=y.\text{patient_id}} \\ & \rho(dp, \Pi^*(\text{department})) \bowtie_{dp.\text{head_of_department}=x.\text{doctor_id}} \rho(d, \Pi^r_{r.\text{visit_id}}, r.\text{patient_id}, d.^* (\text{doctor})) \\ & \bowtie_{d.\text{doctor_id}=r.\text{doctor_id}} \Pi^* \rho(r, \sigma_{\text{operation_id} \text{ not null}} (\text{visit}))) \end{aligned}$$

SQL:

```
SELECT distinct p.patient_id,p.patient_name,p.gender,p.date_of_birth,p.blood_group FROM patient as p JOIN
(SELECT * FROM department as dp JOIN
(SELECT r.visit_id,r.patient_id,d.* FROM doctor as d JOIN
(SELECT * FROM visit where operation_id is not null) as r
ON (d.doctor_id=r.doctor_id) ) as x
ON (dp.head_of_department=x.doctor_id)) as y
ON (p.patient_id=y.patient_id);
```

OUTPUT:



```
570 (select distinct count(*),patient_id from visit group by patient_id having count(*)>2) as t
571 on (patient.patient_id=r.patient_id)
572 26)
573 select distinct p.patient_id,p.patient_name,p.gender,p.date_of_birth,p.blood_group from patient as p join
574 (
575 select * from department as dp join
576 (select r.visit_id,r.patient_id,d.* from doctor as d join
577 (select * from visit where operation_id is not null) as r
578 on (d.doctor_id=r.doctor_id) ) as x on (dp.head_of_department=x.doctor_id)
579 ) as y on (p.patient_id=y.patient_id)
580
581
```

patient_id	patient_name	gender	date_of_birth	blood_group
1	kajal	F	1995-10-05	AB+
2	priyanka	F	1982-12-26	A+
3	parshva	M	1998-10-05	AB+
4	amit	M	1975-12-07	A-
5	abid	M	1999-09-18	B+
6	sejal	O	1996-02-19	AB-
7	khush	M	2001-04-13	
8	naval	F	1005-00-10	

24) List the Nurses who have taken leave on a particular date.

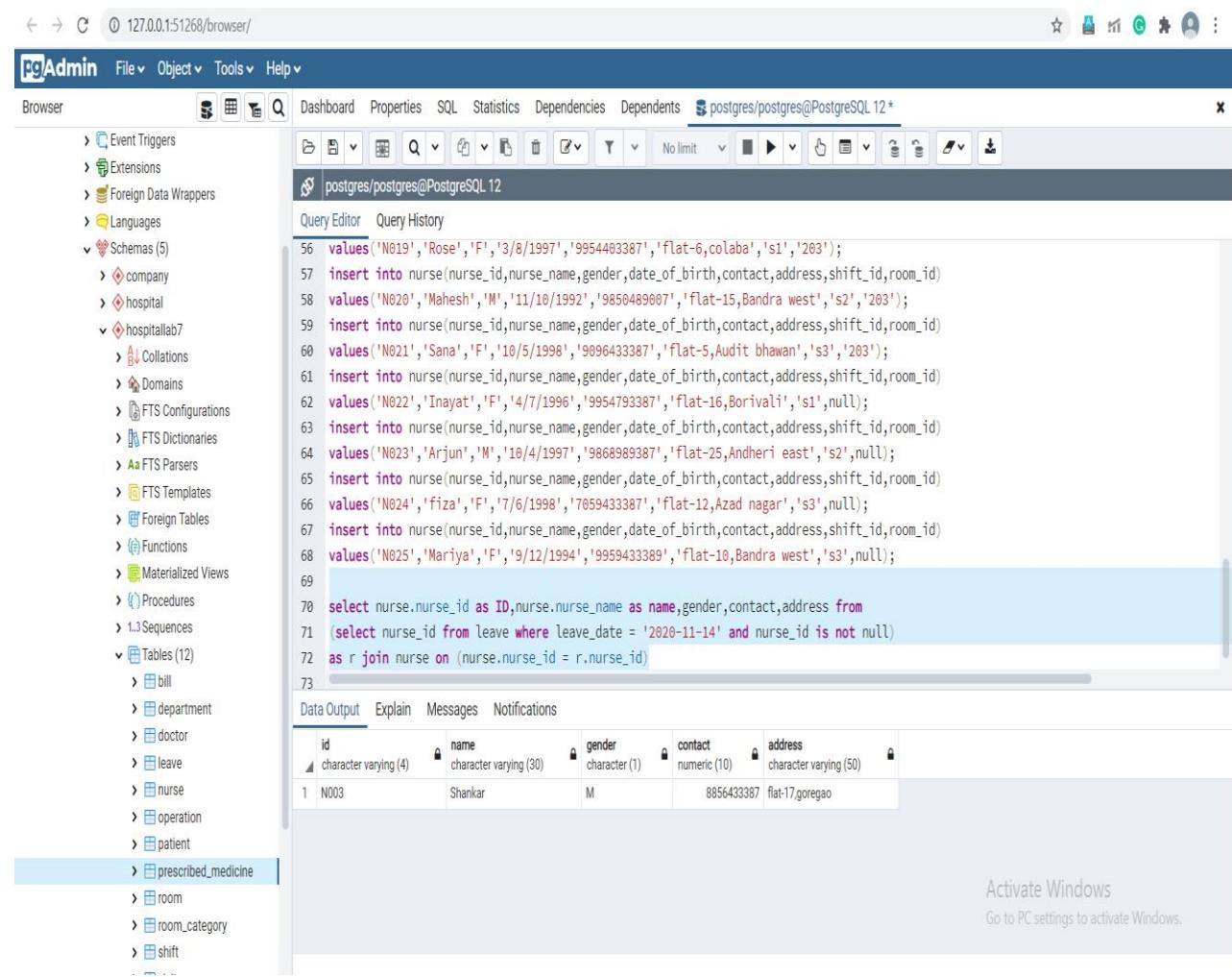
Relational Algebra:

$\Pi_{\text{nurse_id} \rightarrow \text{ID}, \text{nurse_name} \rightarrow \text{name}, \text{gender}, \text{contact}, \text{address}} \rho(\Pi_{\text{nurse_id}} (\sigma \text{leave_date} = '2020-11-14' \text{ and } \text{nurse_id} \text{ is not null}) \text{Leave}), r \bowtie_{\text{nurse.nurse_id} = \text{r.nurse_id}} \text{Nurse}$

SQL:

select nurse.nurse_id as ID,nurse.nurse_name as name,gender,contact,address from (select nurse_id from leave where leave_date = '2020-11-14' and nurse_id is not null) as r join nurse on (nurse.nurse_id = r.nurse_id)

OUTPUT:



The screenshot shows the pgAdmin 4 interface. On the left is the Object Browser tree, which includes Event Triggers, Extensions, Foreign Data Wrappers, Languages, Schemas (5), company, hospital, hospitalab7, Collations, Domains, FTS Configurations, FTS Dictionaries, FTS Parsers, FTS Templates, Foreign Tables, Functions, Materialized Views, Procedures, Sequences, and Tables (12). The 'prescribed_medicine' table is currently selected. The main window contains a Query Editor tab with the following SQL query:

```
values('N019','Rose','F','3/8/1997','9954403387','flat-6,colaba','s1','203');
insert into nurse(nurse_id,nurse_name,gender,date_of_birth,contact,address,shift_id,room_id)
values('N020','Mahesh','M','11/10/1992','9850489087','flat-15,Bandra west','s2','203');
insert into nurse(nurse_id,nurse_name,gender,date_of_birth,contact,address,shift_id,room_id)
values('N021','Sana','F','10/5/1998','9096433387','flat-5,Audit bawan','s3','203');
insert into nurse(nurse_id,nurse_name,gender,date_of_birth,contact,address,shift_id,room_id)
values('N022','Inayat','F','4/7/1996','9954793387','flat-16,Borivali','s1',null);
insert into nurse(nurse_id,nurse_name,gender,date_of_birth,contact,address,shift_id,room_id)
values('N023','Arjun','M','10/4/1997','9868989387','flat-25,Andheri east','s2',null);
insert into nurse(nurse_id,nurse_name,gender,date_of_birth,contact,address,shift_id,room_id)
values('N024','Fiza','F','7/6/1998','7059433387','flat-12,Azad nagar','s3',null);
insert into nurse(nurse_id,nurse_name,gender,date_of_birth,contact,address,shift_id,room_id)
values('N025','Mariya','F','9/12/1994','9959433389','flat-10,Bandra west','s3',null);
select nurse.nurse_id as ID,nurse.nurse_name as name,gender,contact,address from
(select nurse_id from leave where leave_date = '2020-11-14' and nurse_id is not null)
as r join nurse on (nurse.nurse_id = r.nurse_id)
```

Below the query editor is a Data Output tab showing the results of the query:

	id	name	gender	contact	address
1	N003	Shankar	M	8856433387	flat-17,goregao

25) Get the revenue generated in a particular month by rooms (operation).

Relational Algebra:

$\Pi_{*} (\text{room_category}) \bowtie \text{room_category.category_id} = x.\text{category_id}$

$\rho(\Pi_{*} (\text{room}) \bowtie \text{room.room_id} = r.\text{room_id}$

$\rho(\Pi_{\text{visit_id}, \text{room_id}, (\text{discharge_date}-\text{visit_date} \rightarrow \text{days})} (\sigma_{\text{visit_date} \geq '2020-10-01' \text{ and } \text{visit_date} \leq '2020-10-31' \text{ and } \text{discharge_date} \leq '2020-10-31'} \text{visit})$

\cup

$\Pi_{\text{visit_id}, \text{room_id}, (\text{discharge_date}-\text{visit_date} \rightarrow \text{days})} (\sigma_{\text{visit_date} < '2020-10-01' \text{ and } \text{discharge_date} \leq '2020-10-31' \text{ and } \text{discharge_date} \geq '2020-10-01' \text{ and } \text{discharge_date} \text{ is not null}}$

$\text{visit})$

\cup

$\Pi_{\text{visit_id}, \text{room_id}, (\text{discharge_date}-\text{visit_date} \rightarrow \text{days})} (\sigma_{\text{visit_date} \geq '2020-10-01' \text{ and } \text{visit_date} \leq '2020-10-31' \text{ and } \text{discharge_date} > '2020-10-31'} \text{visit})$

$, r, x)$

SQL:

select sum(category_charges*days) from room_category join

(select * from room join

(select visit_id, room_id, discharge_date-visit_date as days from visit where visit_date >= '2020-10-01' and visit_date <= '2020-10-31' and discharge_date <= '2020-10-31'

union

select visit_id, room_id, discharge_date-'2020-10-01' as days from visit where visit_date < '2020-10-01' and discharge_date <= '2020-10-31' and discharge_date >= '2020-10-01' and discharge_date is not null union

select visit_id, room_id, '2020-10-31'-visit_date as days from visit where visit_date >= '2020-10-01' and visit_date <= '2020-10-31' and discharge_date > '2020-10-31') as r

on(room.room_id=r.room_id)) as x on (room_category.category_id=x.category_id)

pgAdmin 4 Group9Queries - Google Docs

127.0.0.1:11931/browser/

File Object Tools Help

Browser postgres/postgres@PostgreSQL 12*

Tables (17)

- bill
- billnew
- billnew1
- billnew2
- billnew3
- billnew4
- department
- doctor
- leave

Columns (5)

- doctor_leave
- nurse_leave
- leave_date
- reason
- nurse_substitution

Constraints (4)

- leave_doctor_leave_f
- leave_doctor_leave_r
- leave_nurse_leave_fk
- leave_nurse_substitu

Indexes

RLS Policies

Rules

Triggers

nurse

Query Editor Query History

```
101
102 /*25*/
103 select sum(category_charges*days) from room_category join
104 (
105 select * from room join
106 (
107 select visit_id,room_id,discharge_date-visit_date as days from visit where visit_date >= '2020-10-01' and visit_da
108
109 union
110 select visit_id,room_id,discharge_date-'2020-10-01' as days from visit where visit_date<'2020-10-01' and discharge
111 union
112 select visit_id,room_id,'2020-10-31'-visit_date as days from visit where visit_date >= '2020-10-01' and visit_date
113
114 ) as r
115 on(room_room_id=r.room_id) ) as x on (room_category_category_id=x.category_id)
```

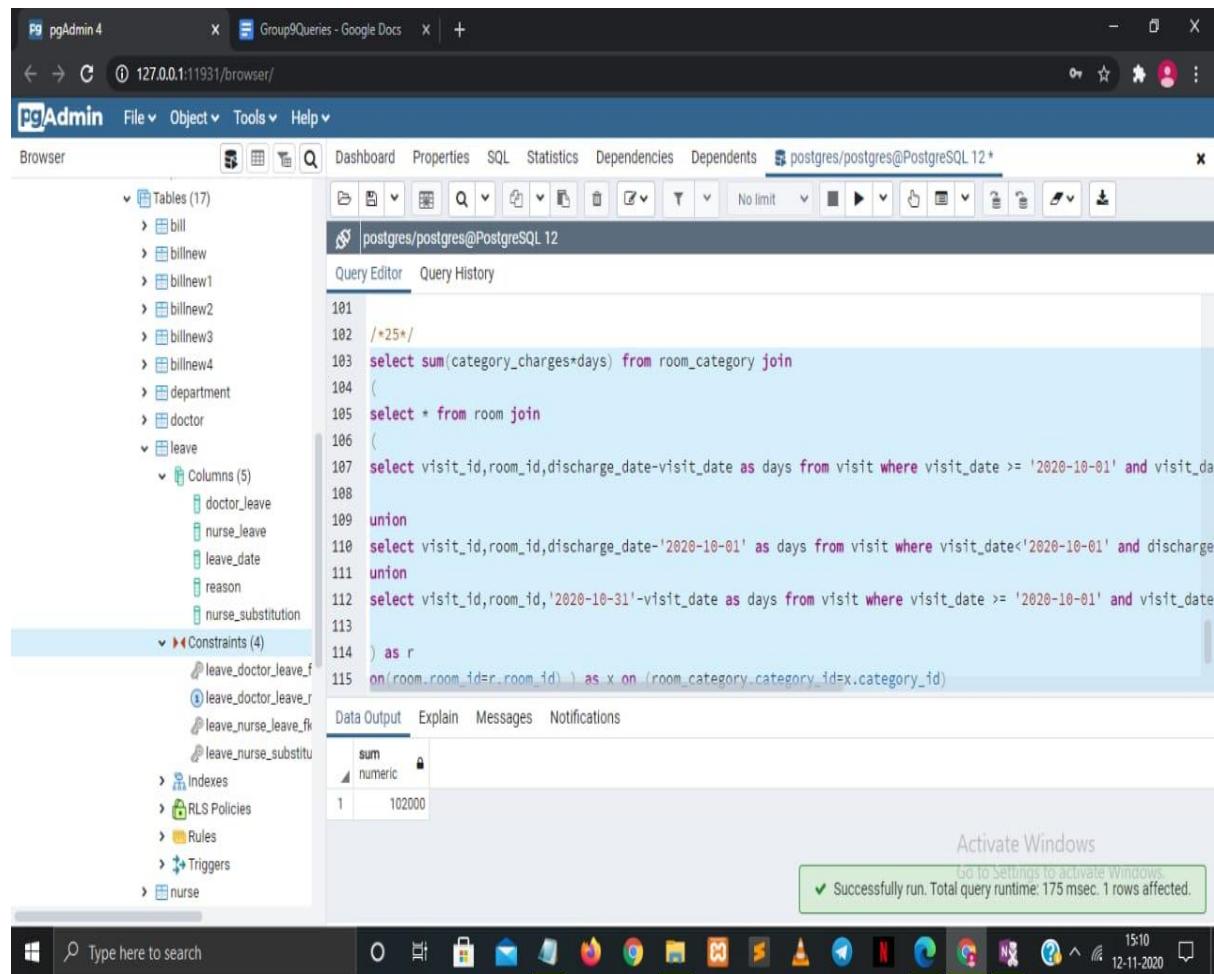
Data Output Explain Messages Notifications

sum
1 102000

Activate Windows
Go to Settings to activate Windows
✓ Successfully run. Total query runtime: 175 msec. 1 rows affected.

Type here to search

15:10 12-11-2020



26) List the substitution of nurses on a particular date.

Relational Algebra:

$$\Pi_{\text{nurse}.\text{nurse_id} \rightarrow \text{Leave_Taker}, \text{nurse}.\text{nurse_name}, \text{x}. \text{sid} \rightarrow \text{Substitute}, \text{nurse}.\text{nurse_name}, \text{x}. \text{nurse_name}} (\bigwedge (\pi_{*}(\bigwedge (\pi_{\text{nurse}.\text{id} \rightarrow \text{nid}}, \text{substitute}.\text{id} \rightarrow \text{sid}) (\sigma_{\text{leave_date} = '2020-11-14' \text{ and } \text{substitute}.\text{id} \text{ is not null}} \text{Leave}), \text{R}) \bowtie_{\text{nurse}.\text{nurse_id} = \text{r}. \text{sid}} \text{Nurse}), \text{X}) \bowtie_{\text{nurse}.\text{nurse_id} = \text{x}. \text{nid}} \text{Nurse})$$

SQL:

```
select nurse.nurse_id as Leave_Taker,nurse.nurse_name, x.sid as Substitute,x.nurse_name  
from(
```

```
select * from
```

```
(
```

```
select nurse_id as nid,substitute_id as sid from leave where leave_date='2020-11-12' and  
substitute_id is not null) as
```

```
r join nurse on (nurse.nurse_id=r.sid) ) as x join nurse on (nurse.nurse_id=x.nid)
```

OUTPUT:

The screenshot shows the PgAdmin 4 interface. The left sidebar displays the database structure under 'Servers (1)'. The 'Tables (12)' section contains 'leave', 'nurse', 'bill', 'department', 'doctor', 'leave', and 'nurse'. The main window has a 'Query Editor' tab active, showing the following SQL query:

```
1 set search_path to hospitallab7;  
2  
3 select * from leave;  
4  
5 select x.nd as leave_taker,x.nurse_name,nurse.nurse_id as substitution_name,nurse.nurse_name from  
(select * from  
6 (select nurse_id as nd,substitute_id from leave where leave_date='2020-11-12' and substitute_id is not nu  
7 r join nurse on (nurse.nurse_id=r.substitute_id))  
8 as x join nurse on (nurse.nurse_id=x.nd);
```

The 'Data Output' tab shows the results of the query:

	leave_taker	nurse_name	substitution_name	nurse_name
1	N012	Rose	N012	kshama

27) Count the number of leaves taken by particular Nurse in a given time.

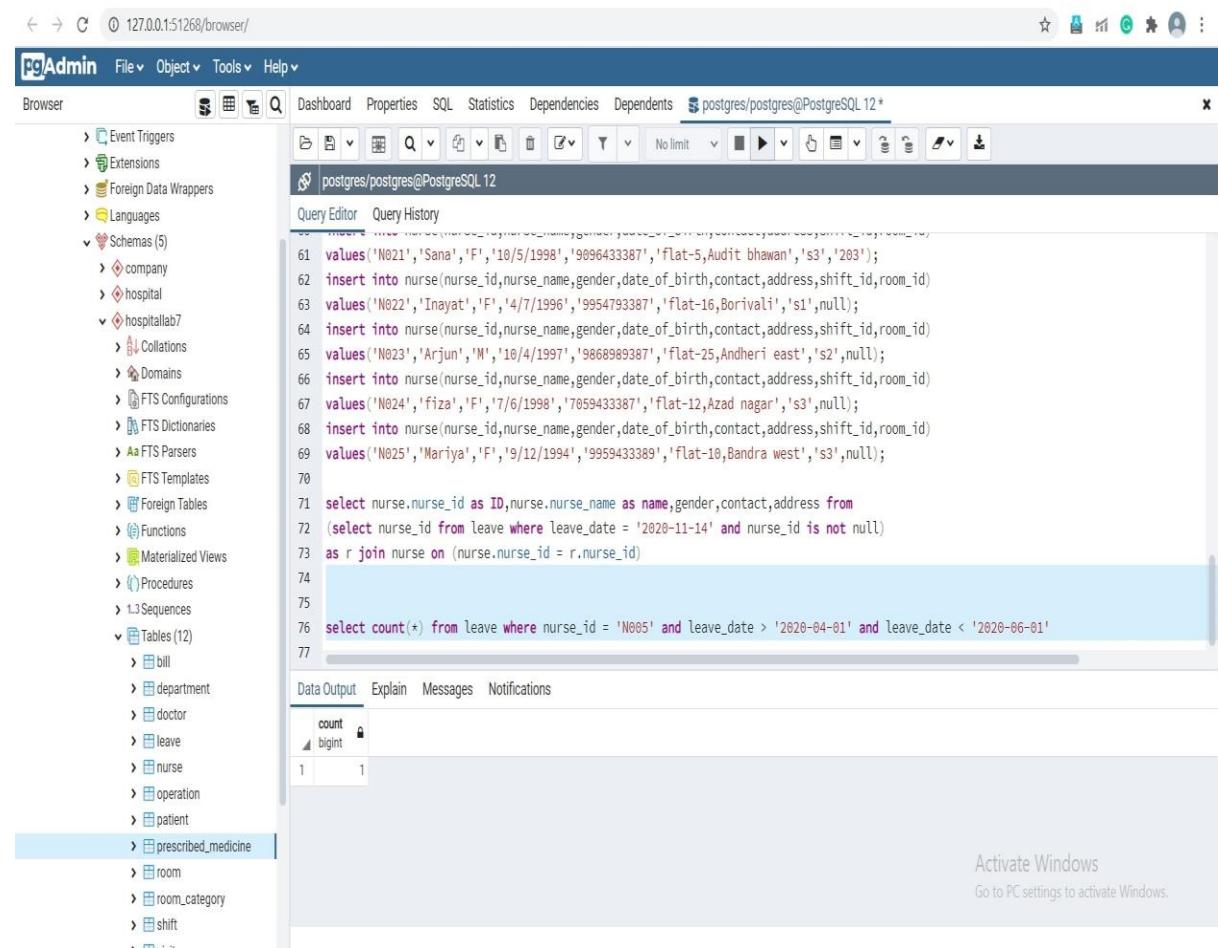
Relational Algebra:

$\mathcal{F}_{\text{count}^*} (\sigma_{\text{nurse_id} = 'N005' \text{ and } \text{leave_date} > '2020-04-01' \text{ and } \text{leave_date} < '2020-06-01'}$
Leave)

SQL:

select count(*) from leave where nurse_id = 'N005' and leave_date > '2020-04-01' and leave_date < '2020-06-01'

OUTPUT:



The screenshot shows the pgAdmin 4 interface. The left sidebar is titled 'Browser' and lists various database objects like Event Triggers, Extensions, Foreign Data Wrappers, Languages, Schemas, Collations, Domains, FTS Configurations, FTS Dictionaries, FTS Parsers, FTS Templates, Foreign Tables, Functions, Materialized Views, Procedures, Sequences, and Tables. The 'Tables' section is currently selected. The main area is a 'Query Editor' window with the following SQL query:

```
values('N021','Sana','F','10/5/1998','9896433387','flat-5,Audit bhawan','s3','203');
insert into nurse(nurse_id,nurse_name,gender,date_of_birth,contact,address,shift_id,room_id)
values('N022','Inayat','F','4/7/1996','9954793387','flat-16,Borivali','s1',null);
insert into nurse(nurse_id,nurse_name,gender,date_of_birth,contact,address,shift_id,room_id)
values('N023','Arjun','M','10/4/1997','9868989387','flat-25,Andheri east','s2',null);
insert into nurse(nurse_id,nurse_name,gender,date_of_birth,contact,address,shift_id,room_id)
values('N024','fiza','F','7/6/1998','7859433387','flat-12,Azad nagar','s3',null);
insert into nurse(nurse_id,nurse_name,gender,date_of_birth,contact,address,shift_id,room_id)
values('N025','Mariya','F','9/12/1994','9959433389','flat-10,Bandra west','s3',null);
select nurse.nurse_id as ID,nurse.nurse_name as name,gender,contact,address from
(select nurse_id from leave where leave_date = '2020-11-14' and nurse_id is not null)
as r join nurse on (nurse.nurse_id = r.nurse_id)
select count(*) from leave where nurse_id = 'N005' and leave_date > '2020-04-01' and leave_date < '2020-06-01'
```

The 'Data Output' tab shows the result of the final query:

count	bigint
1	1

28) Count the number of leaves taken by each nurse in a year

Relational Algebra:

$\Pi_{r.*} \text{nurse.nurse_name} \rho(\text{nurse_id} \not\in \text{count}^* \rightarrow \text{number_of_leaves}, \text{nurse_id}) (\sigma_{\text{leave_date} > '2020-11-01' \text{ and } \text{leave_date} < '2020-12-01'} \text{ and } \text{nurse_id} \text{ is not null} \text{ Leave}), r) \bowtie_{\text{nurse.nurse_id} = r.\text{nurse_id}} \text{Nurse}$

SQL:

```
select r.* ,nurse.nurse_name from (select nurse_id,count(*)as Number_OF_LEAVEs from leave where leave_date > '2020-04-01' and leave_date < '2021-04-01' and nurse_id is not null group by nurse_id ) as r join nurse on (nurse.nurse_id=r.nurse_id)
```

OUTPUT:

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema with various objects like Event Triggers, Extensions, Foreign Data Wrappers, Languages, Schemas, and Tables. The main area is the Query Editor with the following SQL code:

```
1 select nurse.nurse_id as ID,nurse.nurse_name as name,gender,contact,address from
2 (select nurse_id from leave where leave_date = '2020-11-14' and nurse_id is not null)
3 as r join nurse on (nurse.nurse_id = r.nurse_id)
4
5
6 select count(*) from leave where nurse_id = 'N005' and leave_date > '2020-04-01' and leave_date < '2020-06-01'
7
8 select r.* ,nurse.nurse_name from (select nurse_id,count(*)as Number_OF_LEAVEs from leave where leave_date > '2020-04-01' and
9 leave_date < '2021-04-01' and nurse_id is not null group by nurse_id )
10 as r join nurse on (nurse.nurse_id=r.nurse_id)
11
```

The Data Output tab shows the results of the last query:

nurse_id	number_of_leaves	nurse_name
1 N001	1	Natasha
2 N002	1	Asha
3 N003	1	Shankar
4 N005	3	Ashok
5 N006	1	Leela
6 N007	1	Lily
7 N008	1	Ramesh
8 N009	1	Rama
9 N011	1	Varun
10 N012	1	kshama

29) Select the details of doctors (department wise) whose consultation charge is greater than the average consultation charge.

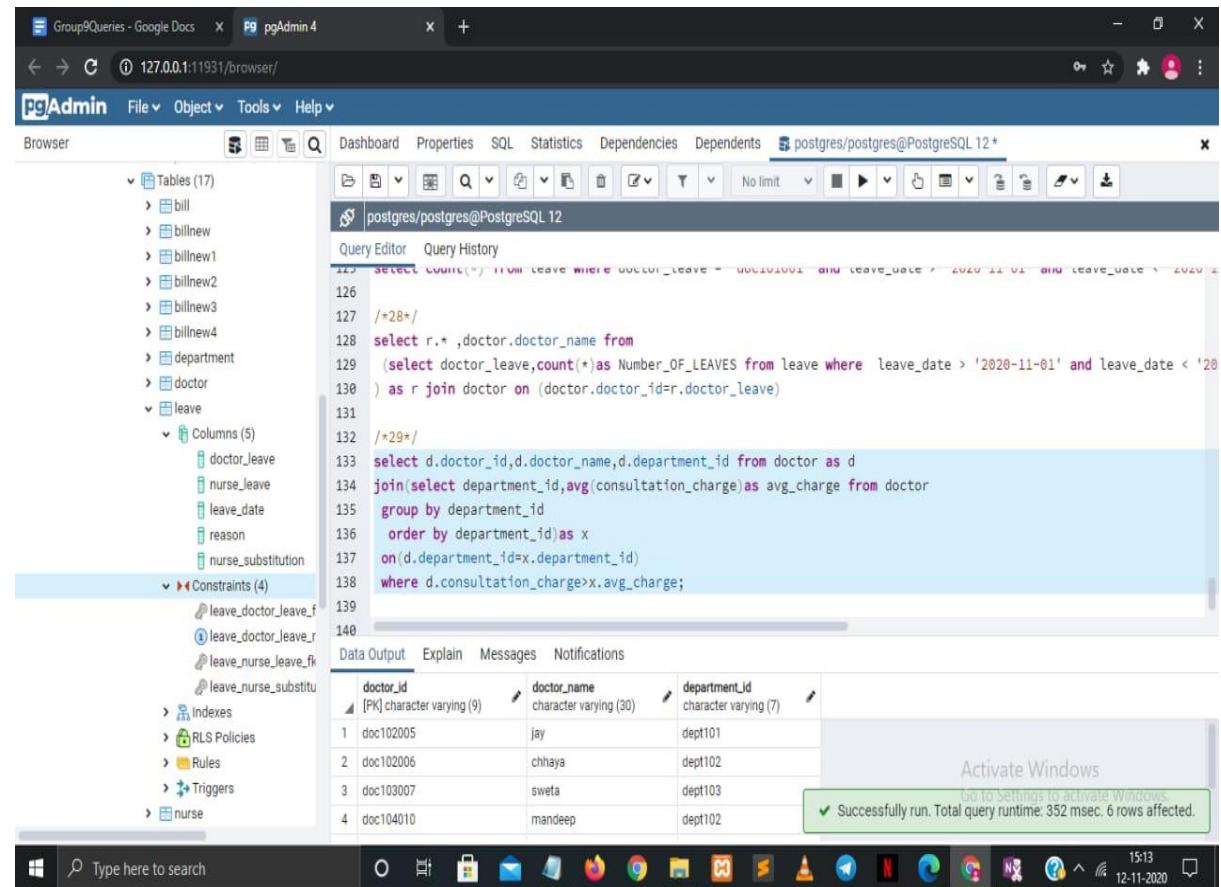
Relational Algebra:

$\sigma_{d.consultation_charge > x.avg_charge}(\text{doctor} \bowtie_{d.department_id=x.department_id} (\rho_{(department_id)} f_{avg(consultation_charge) \rightarrow avg_charge}(\text{doctor}), x))$

SQL:

```
select d.doctor_id,d.doctor_name,d.department_id from doctor as d
join(select department_id,avg(consultation_charge)as avg_charge from doctor
group by department_id
order by department_id)as x
on(d.department_id=x.department_id)
where d.consultation_charge>x.avg_charge;
```

OUTPUT:



```
Group9Queries - Google Docs  X pg pgAdmin 4
127.0.0.1:11931/browser/
File Object Tools Help
Browser
Tables (17)
  bill
  billnew
  billnew1
  billnew2
  billnew3
  billnew4
  department
  doctor
  leave
    Columns (5)
      doctor_leave
      nurse_leave
      leave_date
      reason
      nurse_substitution
  Constraints (4)
    leave_doctor_leave_f
    leave_doctor_leave_r
    leave_nurse_leave_fk
    leave_nurse_substitu
  Indexes
  RLS Policies
  Rules
  Triggers
  nurse
Dashboard Properties SQL Statistics Dependencies Dependents postgres/postgres@PostgreSQL 12*
Query Editor Query History
125 select count(*) from leave where doctor_leave = 'doc102001' and leave_date > '2020-11-01' and leave_date < '2020-12-01'
126 /*28*/
127
128 select r.* ,doctor.doctor_name from
129 (select doctor_leave,count(*)as Number_OF_LEAVEs from leave where leave_date > '2020-11-01' and leave_date < '2020-12-01') as r join doctor on (doctor.doctor_id=r.doctor_leave)
130
131 /*29*/
132
133 select d.doctor_id,d.doctor_name,d.department_id from doctor as d
134 join(select department_id,avg(consultation_charge)as avg_charge from doctor
135 group by department_id
136 order by department_id)as x
137 on(d.department_id=x.department_id)
138 where d.consultation_charge>x.avg_charge;
139
140
```

Data Output Explain Messages Notifications

doctor_id	doctor_name	department_id
1 doc102005	jay	dept101
2 doc102006	chhaya	dept102
3 doc103007	sweta	dept103
4 doc104010	mandeep	dept102

Activate Windows
Go to Settings to activate Windows.
✓ Successfully run. Total query runtime: 352 msec. 6 rows affected.

30) Number of new patient registered with hospital in a year

Relational Algebra:

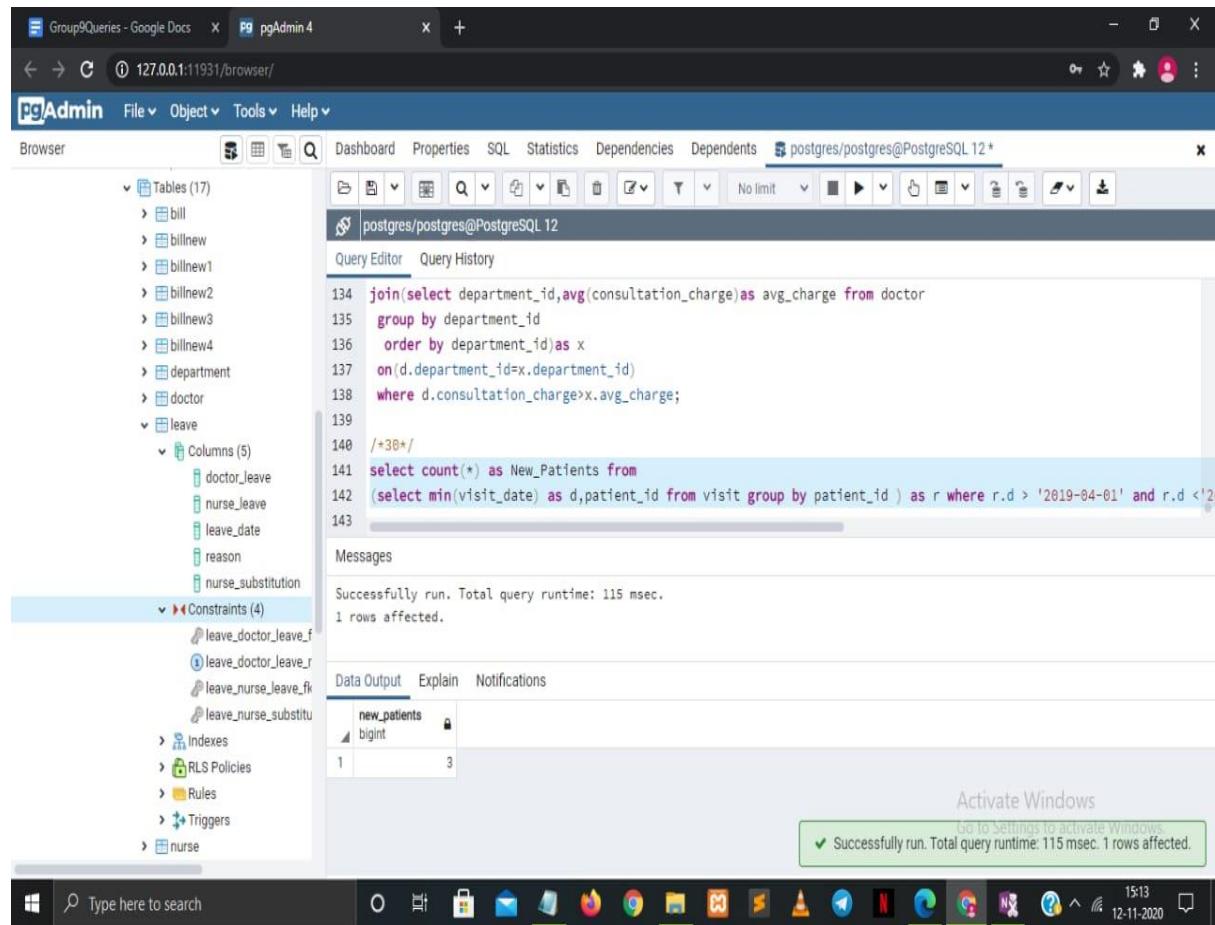
$f_{\text{count}(*)} \rightarrow \text{new_patients}(\rho(r, \text{patient_id}, f_{\text{min}(\text{visit_date})} \rightarrow d(\sigma_{r.d > '2019-04-01' \text{ and } r.d < '2020-03-31'} \text{ visit}))$

SQL:

select count(*) as New_Patients from

(select min(visit_date) as d, patient_id from visit group by patient_id) as r where r.d > '2019-04-01' and r.d <'2020-03-31'

OUTPUT:



The screenshot shows the pgAdmin 4 interface with a query editor window. The browser pane on the left lists tables like bill, billnew, department, doctor, leave, and nurse, along with constraints and indexes. The query editor contains the following SQL code:

```
134 join(select department_id,avg(consultation_charge)as avg_charge from doctor
135 group by department_id
136 order by department_id)as x
137 on(d.department_id=x.department_id)
138 where d.consultation_charge>x.avg_charge;
139
140 /*38*/
141 select count(*) as New_Patients from
142 (select min(visit_date) as d,patient_id from visit group by patient_id ) as r where r.d > '2019-04-01' and r.d <'2020-03-31'
```

The messages pane indicates the query was successfully run with a total runtime of 115 msec and 1 row affected.

new_patients	bigint
1	3

A green message bar at the bottom right says "Successfully run. Total query runtime: 115 msec. 1 rows affected."