

What is the count of movies across ratings?

The screenshot shows the pgAdmin 4 interface. On the left, the 'Catalog' tree is expanded to 'Schemas (1)' > 'public' > 'Tables'. The 'Query' tab is active, showing the following SQL query:

```
1 select rating, count(*)
2 from film
3 group by rating
```

The 'Data Output' tab shows the results of the query in a table with 5 rows and 2 columns: 'rating' (mpaa_rating) and 'count' (bigint). The results are:

rating	count
PG-13	223
NC-17	210
R	195
G	178
PG	194

The status bar at the bottom indicates 'Total rows: 5' and 'Query complete 00:00:00.191'.

rating, count(*)
from film
group by rating

2)What is the average length of a movie across ratings?

select rating, avg(length) as avg_length_movie
from film
group by rating

select

pgAdmin 4

File Object Tools Edit View Window Help

dvd rental/postgres... X dvd rental/postgres@PostgreSQL 14* X

dvd rental/postgres@PostgreSQL 14

Query Query History

```

1 select rating, avg(length) as avg_length_movie
2 from film
3 group by rating

```

Data Output Messages Graph Visualiser X Notifications

Showing rows: 1 to 5 Page No: 1 of 1

	rating	mpaa_rating	avg_length_movie
1	PG-13		120.4439461883408072
2	NC-17		113.2285714285714286
3	R		118.6615384615384615
4	G		111.0505617977528090
5	PG		112.0051546391752577

Total rows: 5 Query complete 00:00:00.170

CRLF Ln 1, Col 47

28°C Light rain 12:59 29-08-2025

3) What is the average rental rate across rating & rental duration?

select rating,rental_duration, avg(rental_rate) as avg_rental_rate

from film

group by rating,rental_duration

limit 5

pgAdmin 4

File Object Tools Edit View Window Help

dvd rental/postgres... X dvd rental/postgres@PostgreSQL 14* X dvd rental/postgres... X

dvd rental/postgres@PostgreSQL 14

Query Query History

```

1 select rating,rental_duration, avg(rental_rate) as avg_rental_rate
2 from film
3 group by rating,rental_duration
4 limit 5
5

```

Data Output Messages Graph Visualiser X Notifications

Showing rows: 1 to 5 Page No: 1 of 1

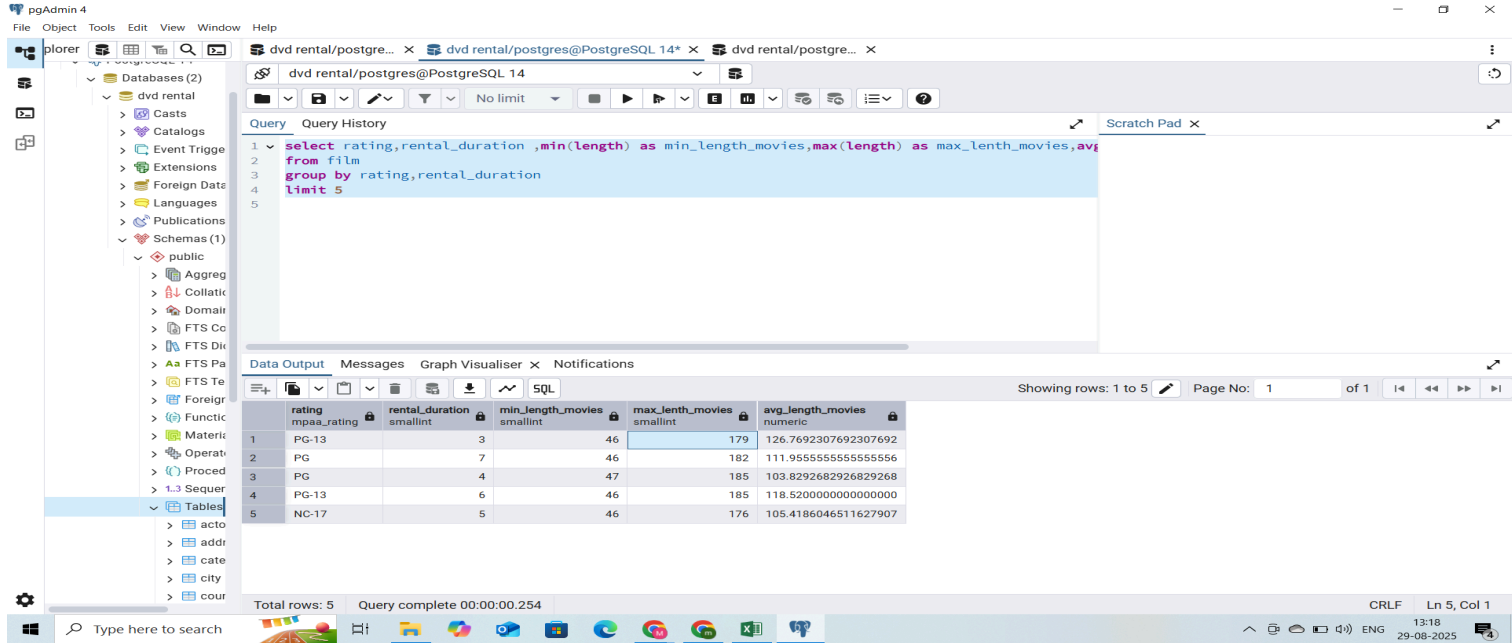
	rating	mpaa_rating	rental_duration	smallint	avg_rental_rate	numeric
1	PG-13		3		2.9900000000000000	
2	PG		7		3.4344444444444444	
3	PG		4		3.0875609756097561	
4	PG-13		6		3.1900000000000000	
5	NC-17		5		3.2690697674418605	

Total rows: 5 Query complete 00:00:00.176

CRLF Ln 4, Col 8

28°C Light rain 13:04 29-08-2025

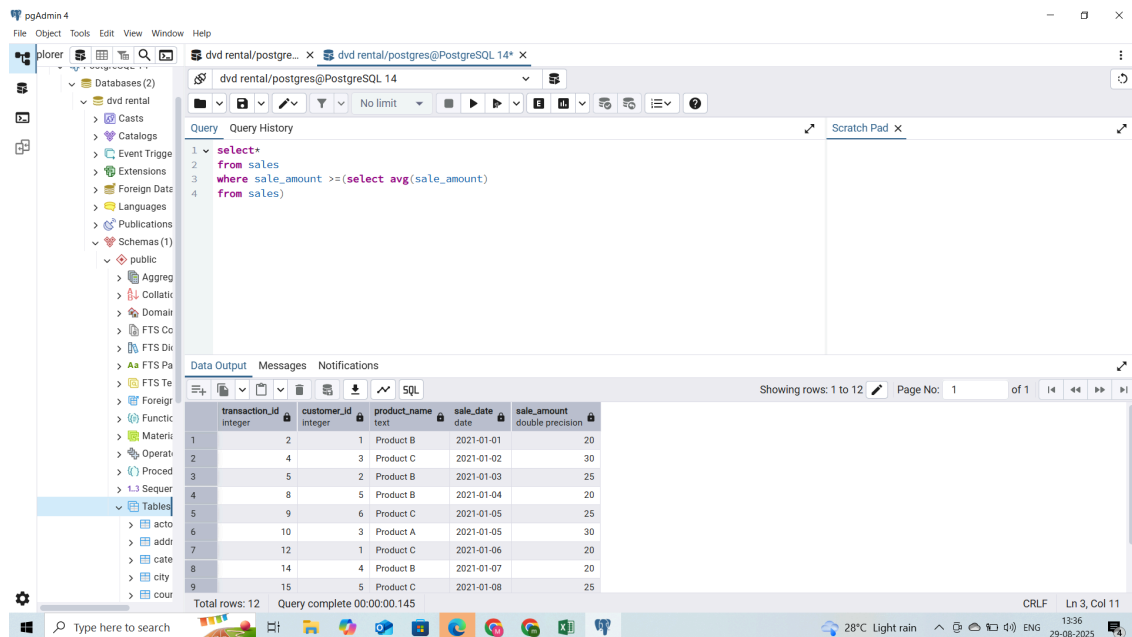
```
select rating,rental_duration ,min(length) as min_length_movies,max(length) as
max_lenth_movies,avg(length)as avg_length_movies
from film
group by rating,rental_duration
limit 5
```



//Subquery to filter results//

1)Get all the transactions greater than equal to avg of sale amount .

```
select*
from sales
where sale_amount >=(select avg(sale_amount)
from sales)
```



The screenshot shows the pgAdmin 4 interface. The SQL query editor contains the following query:

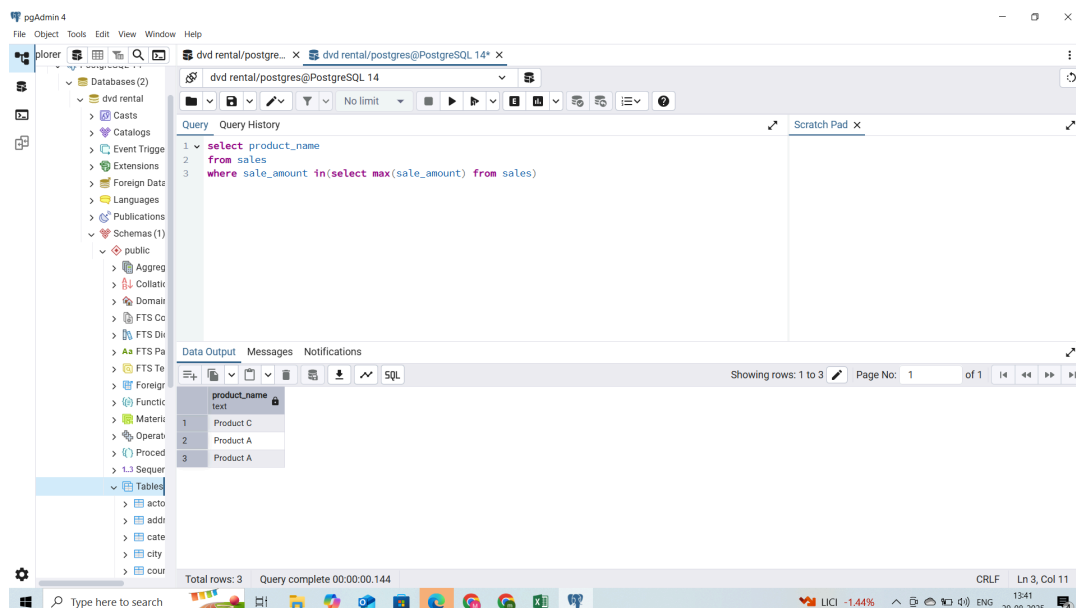
```
1 select*
2 from sales
3 where sale_amount >=(select avg(sale_amount)
4 from sales)
```

The Data Output pane shows the results of the query, displaying 12 rows. The columns are transaction_id, customer_id, product_name, sale_date, and sale_amount.

transaction_id	customer_id	product_name	sale_date	sale_amount
1	2	Product B	2021-01-01	20
2	4	Product C	2021-01-02	30
3	5	Product B	2021-01-03	25
4	8	Product B	2021-01-04	20
5	9	Product C	2021-01-05	25
6	10	Product A	2021-01-05	30
7	12	Product C	2021-01-06	20
8	14	Product B	2021-01-07	20
9	15	Product C	2021-01-08	25
Total rows: 12				

2)product_name which sale was the maximum sales

```
select product_name
from sales
where sale_amount in(select max(sale_amount) from sales)
```



The screenshot shows the pgAdmin 4 interface. The SQL query editor contains the following query:

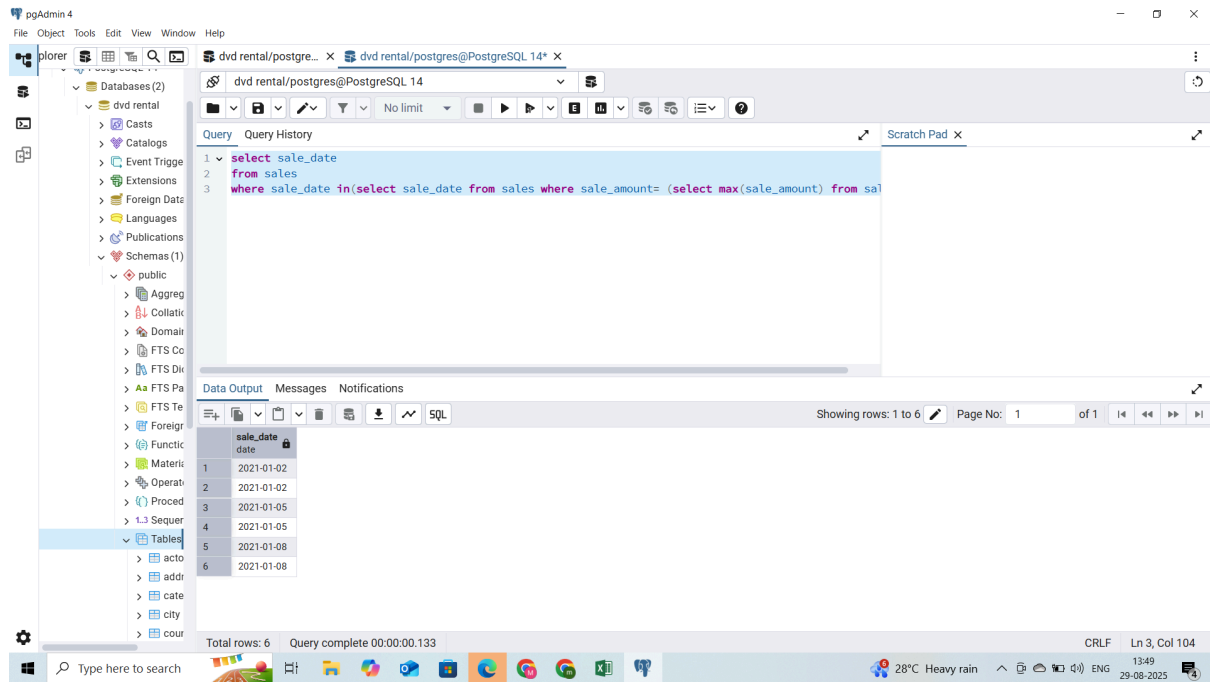
```
1 select product_name
2 from sales
3 where sale_amount in(select max(sale_amount) from sales)
```

The Data Output pane shows the results of the query, displaying 3 rows. The column is product_name.

product_name
Product C
Product A
Product A

3) count of transactions on dates on which sale was maximum sales amount

```
select sale_date
from sales
where sale_date in(select sale_date from sales where sale_amount= (select
max(sale_amount) from sales))
```



//Using subquery to create column//

1) Create column which will be difference of sale amount and minimum of sale amount

```
select *,sale_amount -(select min(sale_amount)from sales)as diff_sales_amount_min
from sales
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure for 'dvd rental', including schemas like 'public'. The main query editor contains the following SQL query:

```

1 select *,sale_amount -(select min(sale_amount)from sales)as diff_sales_amount_min
2 from sales
3 limit 5

```

The 'Data Output' tab shows the results of the query in a table with 6 columns: transaction_id, customer_id, product_name, sale_date, sale_amount, and diff_sales_amount_min. The results are as follows:

transaction_id	customer_id	product_name	sale_date	sale_amount	diff_sales_amount_min
1	1	Product A	2021-01-01	10	0
2	2	Product B	2021-01-01	20	10
3	3	Product A	2021-01-02	15	5
4	4	Product C	2021-01-02	30	20
5	5	Product B	2021-01-03	25	15

The status bar at the bottom indicates 'Total rows: 5' and 'Query complete 00:00:00.189'.

//Subquery to subset a table(filter a table)(for which values u want table in select)//
 1)Sum of sale_amount for date before 6th Jan 2021

```

select sum(sale_amount) as sale_sum
from
(select * from sales where sale_date<
'2021-01-06'
)

```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure for 'dvd rental'. The main query editor contains the following SQL query:

```

1 select sum(sale_amount) as sale_sum
2 from
3 (select * from sales where sale_date<
4 '2021-01-06'
5 )

```

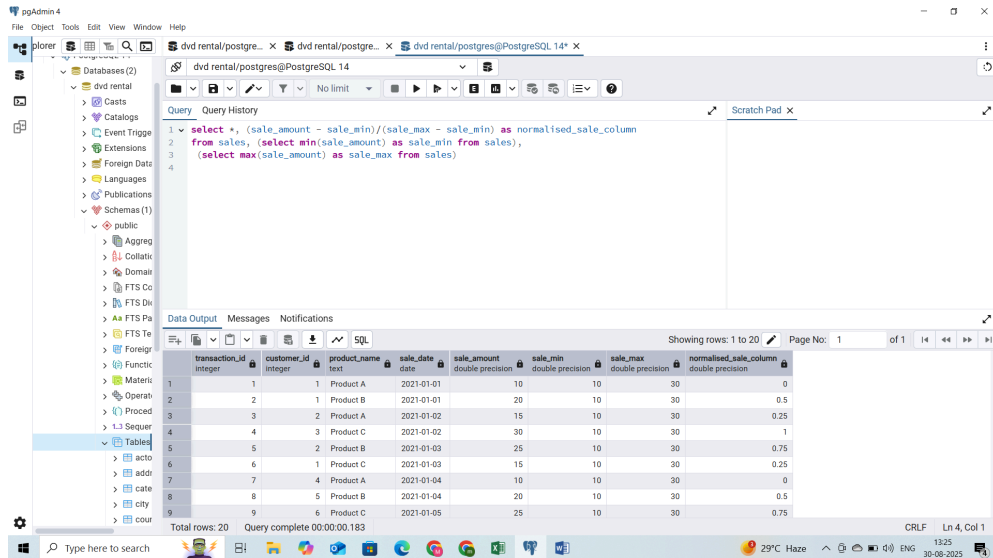
The 'Data Output' tab shows the results of the query in a table with 1 column: sale_sum. The results are as follows:

sale_sum
200

The status bar at the bottom indicates 'Total rows: 1' and 'Query complete 00:00:00.202'.

//Subquery to concat values//

**select *, (sale_amount - sale_min)/(sale_max - sale_min) as normalised_sale_column
from sales, (select min(sale_amount) as sale_min from sales), (select max(sale_amount)
as sale_max from sales)**



The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, including 'dvd rental' and 'public' schemas. The main pane shows a SQL query in the 'Query' tab. The query is as follows:

```
1 select *, (sale_amount - sale_min)/(sale_max - sale_min) as normalised_sale_column
2 from sales, (select min(sale_amount) as sale_min from sales),
3 (select max(sale_amount) as sale_max from sales)
4
```

The 'Data Output' pane shows the results of the query. The table has 9 rows and 7 columns. The columns are: transaction_id, customer_id, product_name, sale_date, sale_amount, sale_min, sale_max, and normalised_sale_column. The data is as follows:

transaction_id	customer_id	product_name	sale_date	sale_amount	sale_min	sale_max	normalised_sale_column
1	1	Product A	2021-01-01	10	10	30	0
2	2	Product B	2021-01-01	20	10	30	0.5
3	3	Product A	2021-01-02	15	10	30	0.25
4	4	Product C	2021-01-02	30	10	30	1
5	5	Product B	2021-01-03	25	10	30	0.75
6	6	Product C	2021-01-03	15	10	30	0.25
7	7	Product A	2021-01-04	10	10	30	0
8	8	Product B	2021-01-04	20	10	30	0.5
9	9	Product C	2021-01-05	25	10	30	0.75

The status bar at the bottom indicates 'Total rows: 20', 'Query complete 00:00:00.183', and 'CRLF Ln 4, Col 1'.

Q1. Find the number of male and female patients.

```
select gender, count(*) as pt_cnt  
from patients
```

Group by gender

The screenshot shows the pgAdmin 4 interface. The Object Explorer on the left lists the database structure, with 'Tables (16)' expanded. The main query editor displays the following SQL query:

```
1  
2 select gender, count(*) as pt_cnt  
3 from patients  
4 group by gender  
5  
6  
7
```

The 'Data Output' pane at the bottom shows the results of the query:

	gender character varying (10)	pt_cnt bigint
1	Male	3
2	Female	2

The status bar at the bottom indicates 'Total rows: 2' and 'Query complete 00:00:00.201'.

Q2. Find the details of patient as pratient_name, age, gender, doctor_id, appointment_date, reason_for_visit

```
select ap.appointment_date,pt.patient_name,pt.age,  
pt.gender,ap.doctor_id,ap.reason_for_visit from patients as pt join appointments as ap on  
pt.patient_id = ap.patient_id
```

The screenshot shows the pgAdmin 4 interface. The Object Explorer on the left lists the database structure, with 'Tables (16)' expanded. The main query editor displays the following SQL query:

```
1 select ap.appointment_date,pt.patient_name,pt.age, pt.gender,ap.doctor_id,ap.reason_for_visit  
2 from patients as pt  
3 join appointments as ap on pt.patient_id = ap.patient_id  
4  
5  
6
```

The 'Data Output' pane at the bottom shows the results of the query:

	appointment_date date	patient_name character varying (100)	age integer	gender character varying (10)	doctor_id integer	reason_for_visit character varying (255)
1	2024-04-01	John Smith	45	Male	1	Chest pain
2	2024-04-02	Alice Johnson	35	Female	2	Child checkup
3	2024-04-03	Michael Brown	60	Male	3	Knee pain
4	2024-04-04	Emily Davis	28	Female	4	Annual checkup

The status bar at the bottom indicates 'Total rows: 4' and 'Query complete 00:00:00.119'.

Q3. Display all doctors along with the number of appointments they have scheduled.

```
select d.doctor_id,d.doctor_name, d.specialization,d.hospital_name, count(*) as  
appointment_count from doctors as d left join appointments as ap on d.doctor_id =  
ap.doctor_id group by d.doctor_id,d.specialization,d.doctor_name, d.hospital_name order by  
doctor_id
```

The screenshot shows the pgAdmin 4 interface. On the left, the Object Explorer displays the database structure, with 'Tables (16)' expanded. The main query editor shows the following SQL query:

```
1 select d.doctor_id,d.doctor_name, d.specialization,d.hospital_name, count(*) as appc  
2 from doctors as d  
3 left join appointments as ap on d.doctor_id = ap.doctor_id group by d.doctor_id,d.sp  
4  
5
```

The 'Data Output' pane at the bottom displays the results of the query in a table format. The table has five columns: doctor_id, doctor_name, specialization, hospital_name, and appointment_count. The results show four rows of data:

doctor_id	doctor_name	specialization	hospital_name	appointment_count
1	Dr. Smith	Cardiologist	City Hospital	1
2	Dr. Johnson	Pediatrician	Children Hospital	1
3	Dr. Brown	Orthopedic Surgeon	General Hospital	1
4	Dr. Davis	Gynecologist	City Hospital	1

A status bar at the bottom indicates: 'Total rows: 4 Query complete 00:00:00.154' and 'Successfully run. Total query runtime: 154 msec. 4 rows affected.'

Q4. Display the appointment dates for all appointments scheduled with doctors in Chicago.

```
select ap.appointment_date, d.doctor_name  
from appointments as ap  
left join doctors as d on ap.doctor_id = d.doctor_id where d.city = 'Chicago'
```

pgAdmin 4

File Object Tools Edit View Window Help

Object Explorer

- Foreign Tables
- Functions
- Materialized Views
- Operators
- Procedures
- Sequences
- Tables (16)
 - actor
 - address
 - category
 - city
 - country
 - customer
 - film
 - film_actor
 - film_category
 - inventory
 - language
 - payment
 - rental
 - sales
 - staff
 - store
- Trigger Functions
- Types
- Views
- Subscriptions
- postgres
- Login/Group Roles
- Tablespaces
- PostgreSQL 17

dvd rental/postgres@PostgreSQL 14*

Query

```

1 select ap.appointment_date, d.doctor_name
2 from appointments as ap
3 left join doctors as d on ap.doctor_id = d.doctor_id
4 where d.city = 'Chicago'

```

Data Output

appointment_date	doctor_name
2024-04-03	Dr. Brown

Showing rows: 1 to 1 Page No: 1 of 1

Total rows: 1 Query complete 00:00:00.133

CRLF Ln 4, Col 25

15:18 30-08-2025

Q5. Generate all possible combinations of patients and doctors for referral purpose

select p.patient_name,d.doctor_name

from patients as p

cross join doctors as d

pgAdmin 4

File Object Tools Edit View Window Help

Object Explorer

- Foreign Tables
- Functions
- Materialized Views
- Operators
- Procedures
- Sequences
- Tables (16)
 - actor
 - address
 - category
 - city
 - country
 - customer
 - film
 - film_actor
 - film_category
 - inventory
 - language
 - payment
 - rental
 - sales
 - staff
 - store
- Trigger Functions
- Types
- Views
- Subscriptions
- postgres
- Login/Group Roles
- Tablespaces
- PostgreSQL 17

dvd rental/postgres@PostgreSQL 14*

Query

```

1 select p.patient_name,d.doctor_name
2 from patients as p
3 cross join doctors as d
4

```

Data Output

patient_name	doctor_name
John Smith	Dr. Smith
John Smith	Dr. Johnson
John Smith	Dr. Brown
John Smith	Dr. Davis
Alice Johnson	Dr. Smith
Alice Johnson	Dr. Johnson
Alice Johnson	Dr. Brown
Alice Johnson	Dr. Davis
Michael Brown	Dr. Smith

Showing rows: 1 to 20 Page No: 1 of 1

Total rows: 20 Query complete 00:00:00.162

CRLF Ln 4, Col 1

15:19 30-08-2025