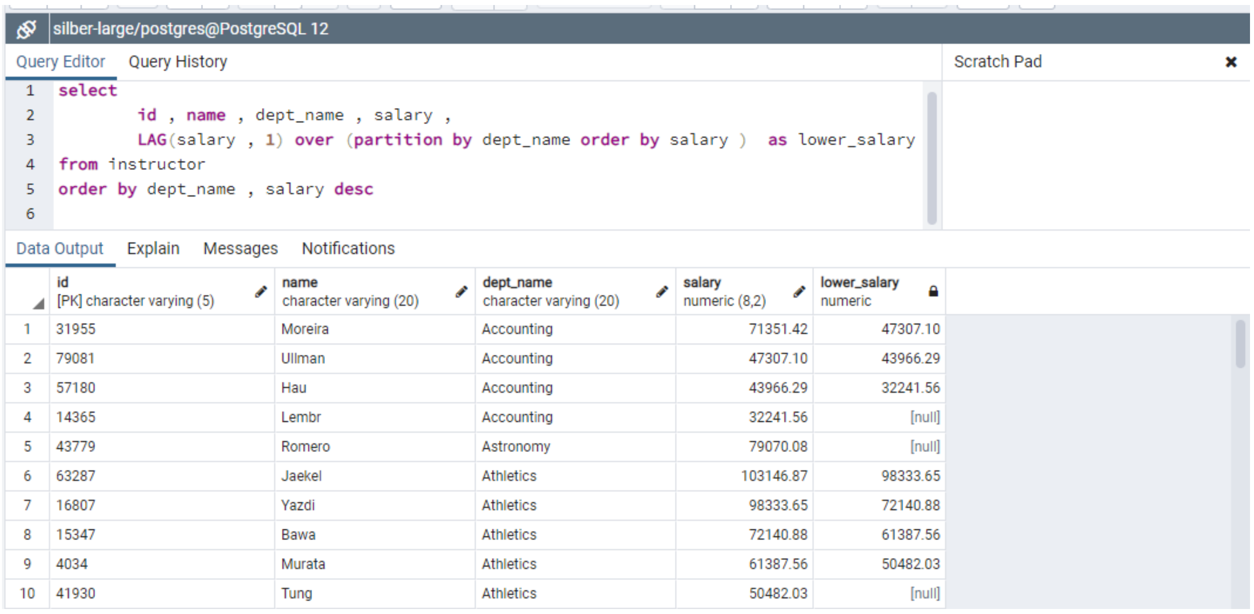
DataBase | HW05

Fatemeh Nadi | 9636753

.1

a.

R1a:



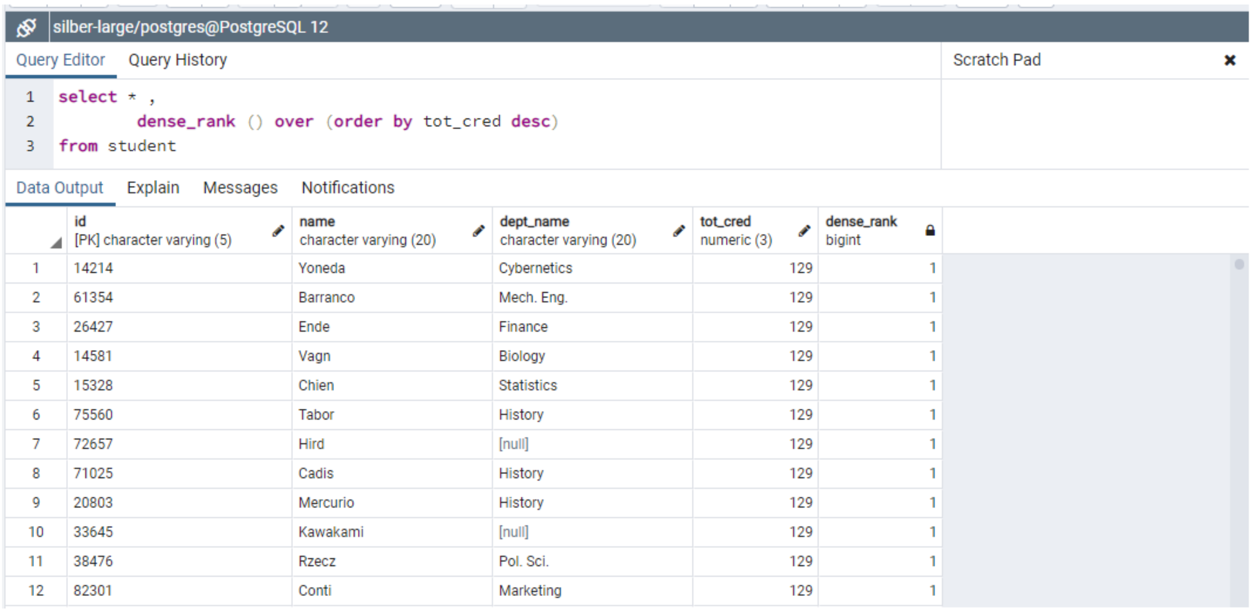
silber-large/postgres@PostgreSQL 12

```
select
        id , name , dept_name , salary ,
        LAG(salary , 1) over (partition by dept_name order by salary )  as lower_salary
from instructor
order by dept_name , salary desc
```

	id [PK] character varying (5)	name character varying (20)	dept_name character varying (20)	salary numeric (8,2)	lower_salary numeric
1	31955	Moreira	Accounting	71351.42	47307.10
2	79081	Ullman	Accounting	47307.10	43966.29
3	57180	Hau	Accounting	43966.29	32241.56
4	14365	Lembr	Accounting	32241.56	[null]
5	43779	Romero	Astronomy	79070.08	[null]
6	63287	Jaekel	Athletics	103146.87	98333.65
7	16807	Yazdi	Athletics	98333.65	72140.88
8	15347	Bawa	Athletics	72140.88	61387.56
9	4034	Murata	Athletics	61387.56	50482.03
10	41930	Tung	Athletics	50482.03	[null]

b.

R1b:



silber-large/postgres@PostgreSQL 12

Query Editor | Query History | Scratch Pad

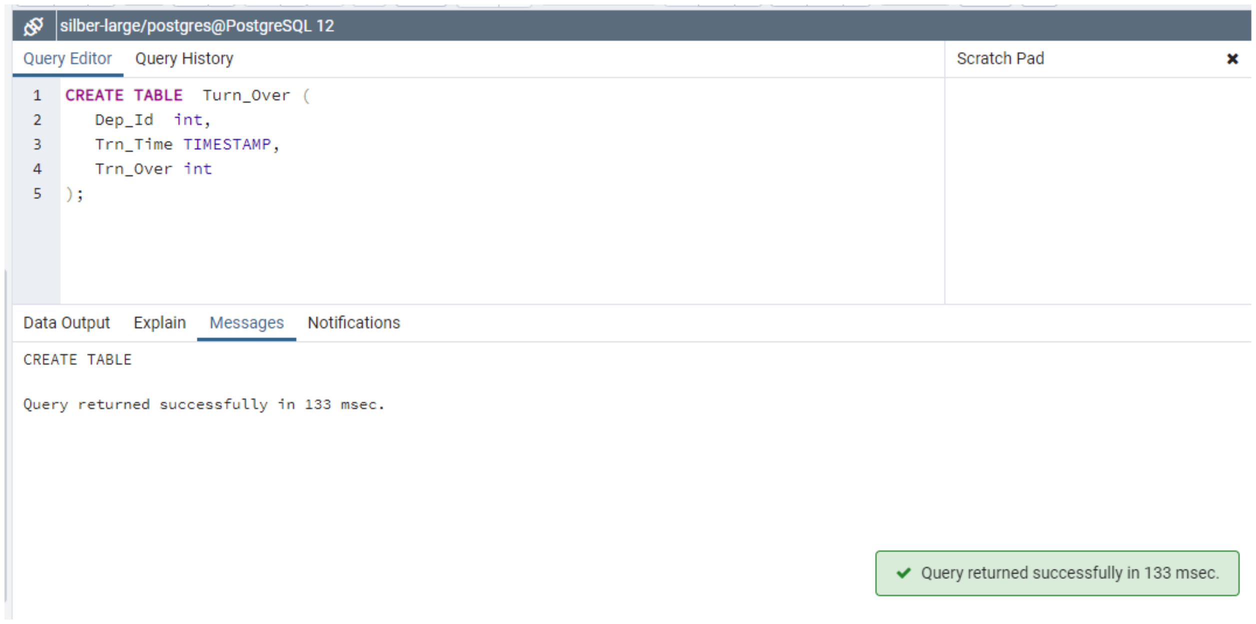
```
select * ,
        dense_rank () over (order by tot_cred desc)
from student
```

Data Output | Explain | Messages | Notifications

	id [PK] character varying (5)	name character varying (20)	dept_name character varying (20)	tot_cred numeric (3)	dense_rank bigint
1	14214	Yoneda	Cybernetics	129	1
2	61354	Barranco	Mech. Eng.	129	1
3	26427	Ende	Finance	129	1
4	14581	Vagn	Biology	129	1
5	15328	Chien	Statistics	129	1
6	75560	Tabor	History	129	1
7	72657	Hird	[null]	129	1
8	71025	Cadis	History	129	1
9	20803	Mercurio	History	129	1
10	33645	Kawakami	[null]	129	1
11	38476	Rzecz	Pol. Sci.	129	1
12	82301	Conti	Marketing	129	1

.2

در ابتدا جدول را ایجاد میکنم:



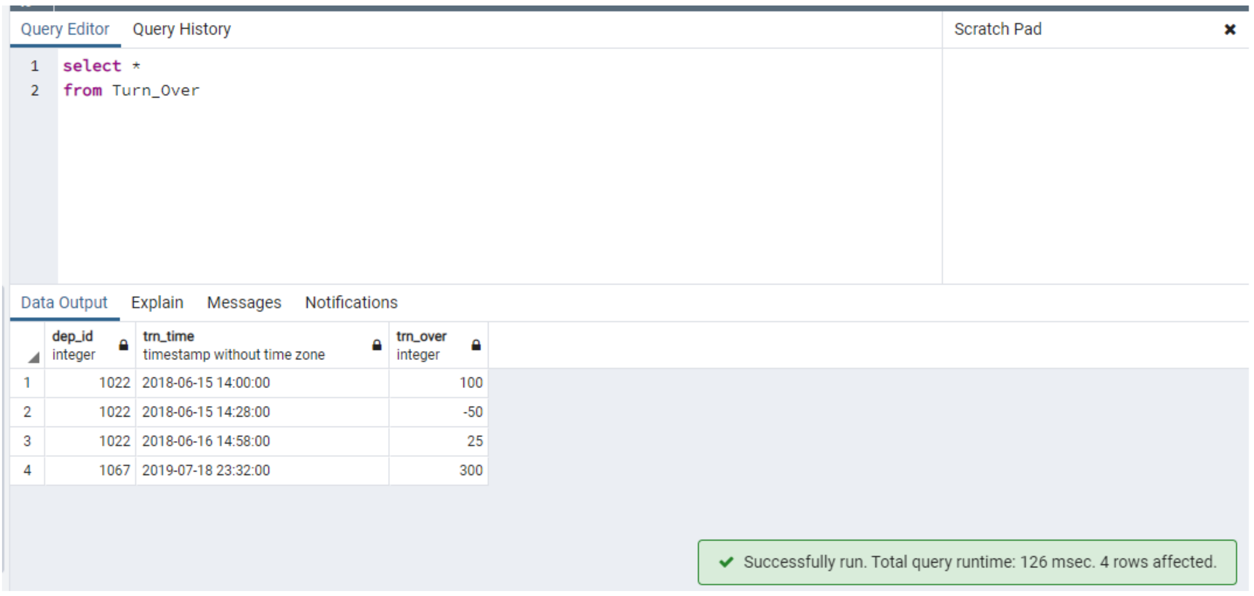
```sql
CREATE TABLE  Turn_Over (
    Dep_Id  int,
    Trn_Time TIMESTAMP,
    Trn_Over int
);
```

```
CREATE TABLE

Query returned successfully in 133 msec.
```

✓ Query returned successfully in 133 msec.

سپس مقادیر را وارد جدول میکنیم

و مقادیر را طبق مسئله مشاهده میکنید:



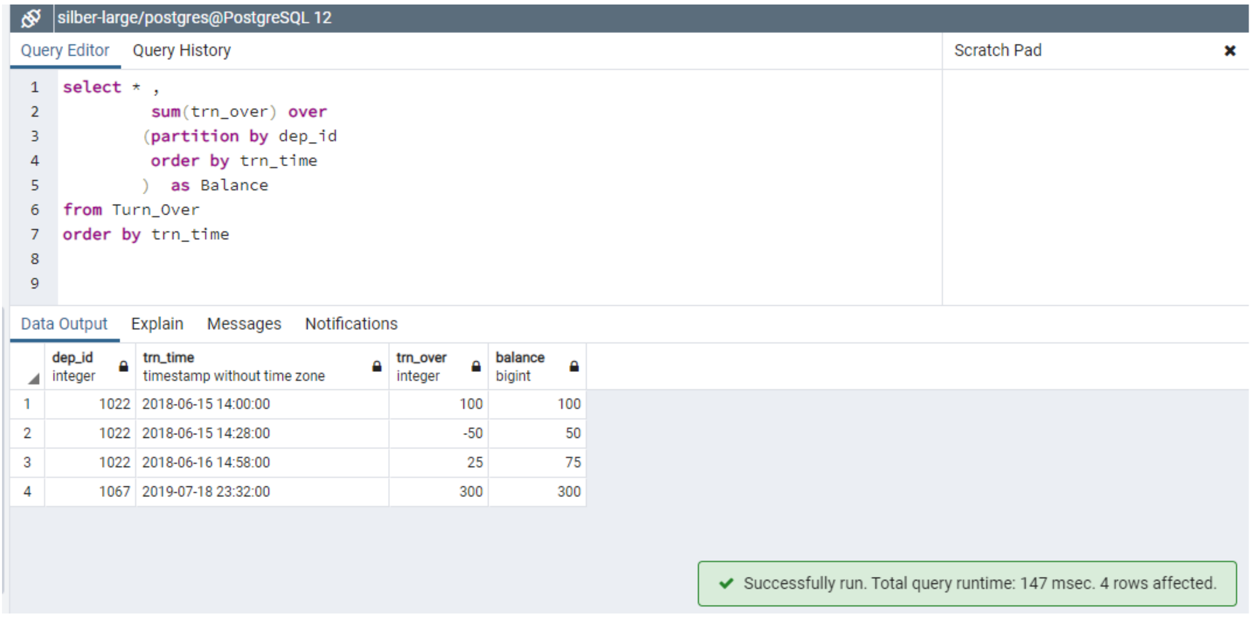
```sql
select *
from Turn_Over
```

	dep_id integer	trn_time timestamp without time zone	trn_over integer
1	1022	2018-06-15 14:00:00	100
2	1022	2018-06-15 14:28:00	-50
3	1022	2018-06-16 14:58:00	25
4	1067	2019-07-18 23:32:00	300

✓ Successfully run. Total query runtime: 126 msec. 4 rows affected.

سپس آنچه در سوال خواسته شده را پیاده سازی کرده ام :

R2:



```sql
select * ,
          sum(trn_over) over
         (partition by dep_id
          order by trn_time
         )  as Balance
from Turn_Over
order by trn_time
```

	dep_id integer	trn_time timestamp without time zone	trn_over integer	balance bigint
1	1022	2018-06-15 14:00:00	100	100
2	1022	2018-06-15 14:28:00	-50	50
3	1022	2018-06-16 14:58:00	25	75
4	1067	2019-07-18 23:32:00	300	300

✓ Successfully run. Total query runtime: 147 msec. 4 rows affected.

.3

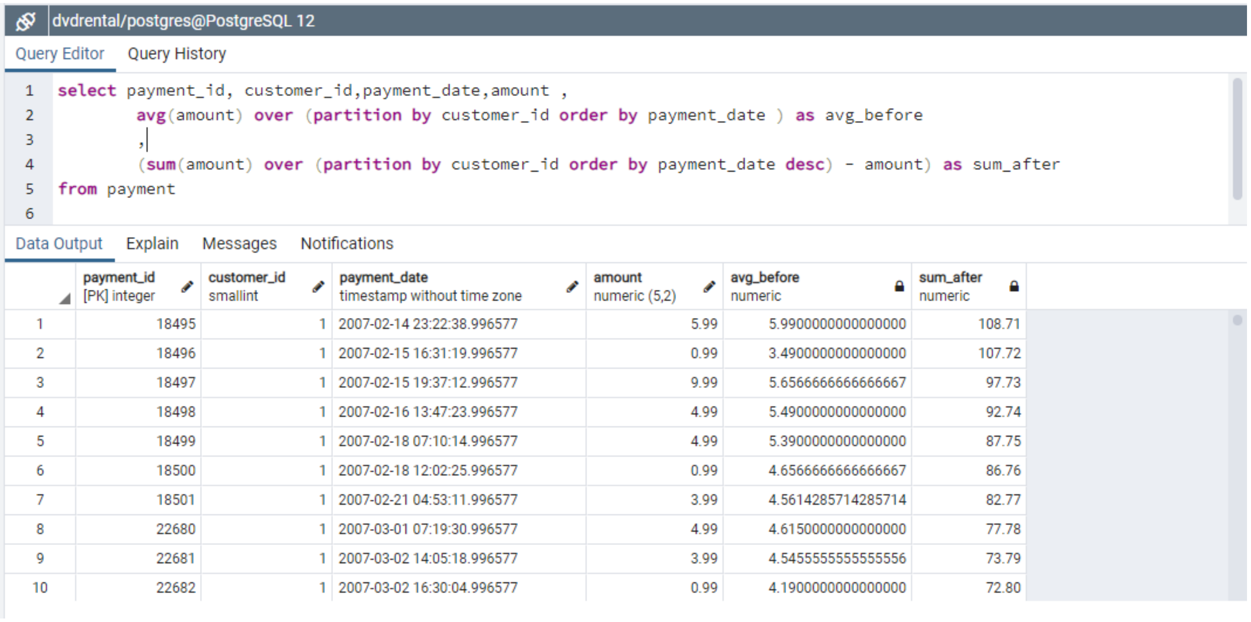
a.

چون صورت سوال مبهم بود و برداشت های متفاوت وجود داشت

من با این فرض سوال را حل کردم که avg_before میانگین کل پرداختی ها تا الان (از اول تا همین پرداخت)

و sum_after جمع کل پرداختی ها بعد از این پرداخت (کل پرداختی ها منهای از اول تا الان) :

R3a:



dvdrental/postgres@PostgreSQL 12

Query Editor | Query History

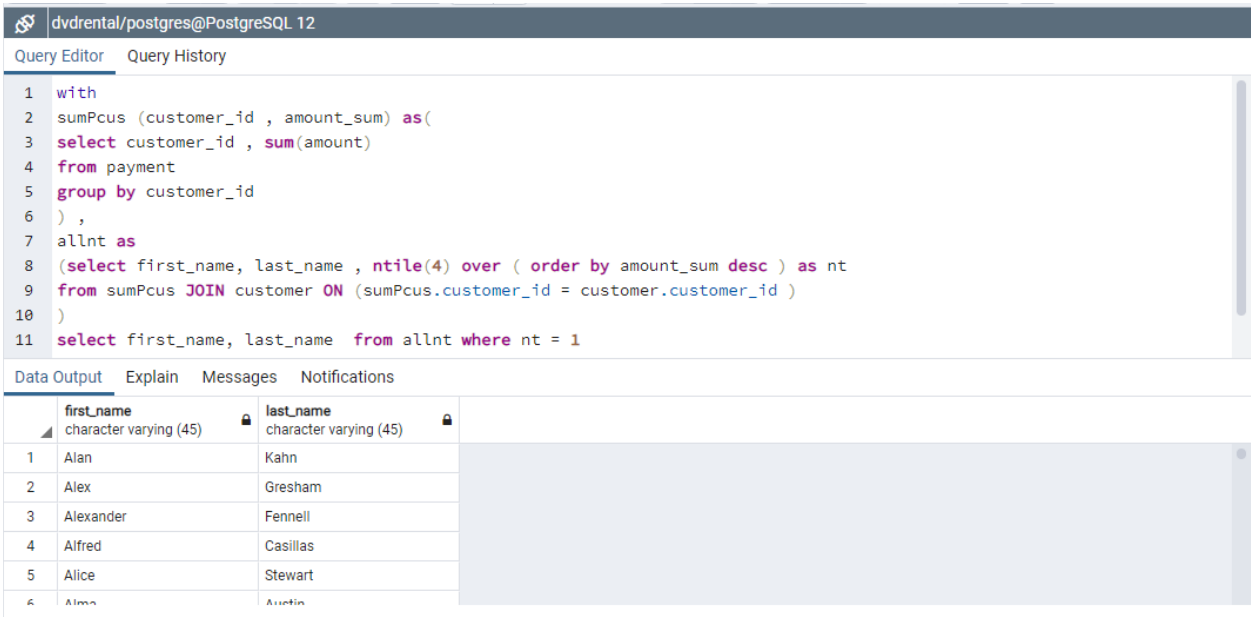
```
select payment_id, customer_id,payment_date,amount ,
         avg(amount) over (partition by customer_id order by payment_date ) as avg_before
         ,
         (sum(amount) over (partition by customer_id order by payment_date desc) - amount) as sum_after
from payment

```

Data Output | Explain | Messages | Notifications

	payment_id [PK] integer	customer_id smallint	payment_date timestamp without time zone	amount numeric (5,2)	avg_before numeric	sum_after numeric
1	18495	1	2007-02-14 23:22:38.996577	5.99	5.9900000000000000	108.71
2	18496	1	2007-02-15 16:31:19.996577	0.99	3.4900000000000000	107.72
3	18497	1	2007-02-15 19:37:12.996577	9.99	5.6566666666666667	97.73
4	18498	1	2007-02-16 13:47:23.996577	4.99	5.4900000000000000	92.74
5	18499	1	2007-02-18 07:10:14.996577	4.99	5.3900000000000000	87.75
6	18500	1	2007-02-18 12:02:25.996577	0.99	4.6566666666666667	86.76
7	18501	1	2007-02-21 04:53:11.996577	3.99	4.5614285714285714	82.77
8	22680	1	2007-03-01 07:19:30.996577	4.99	4.6150000000000000	77.78
9	22681	1	2007-03-02 14:05:18.996577	3.99	4.5455555555555556	73.79
10	22682	1	2007-03-02 16:30:04.996577	0.99	4.1900000000000000	72.80

b.

R3b:


dvdrental/postgres@PostgreSQL 12

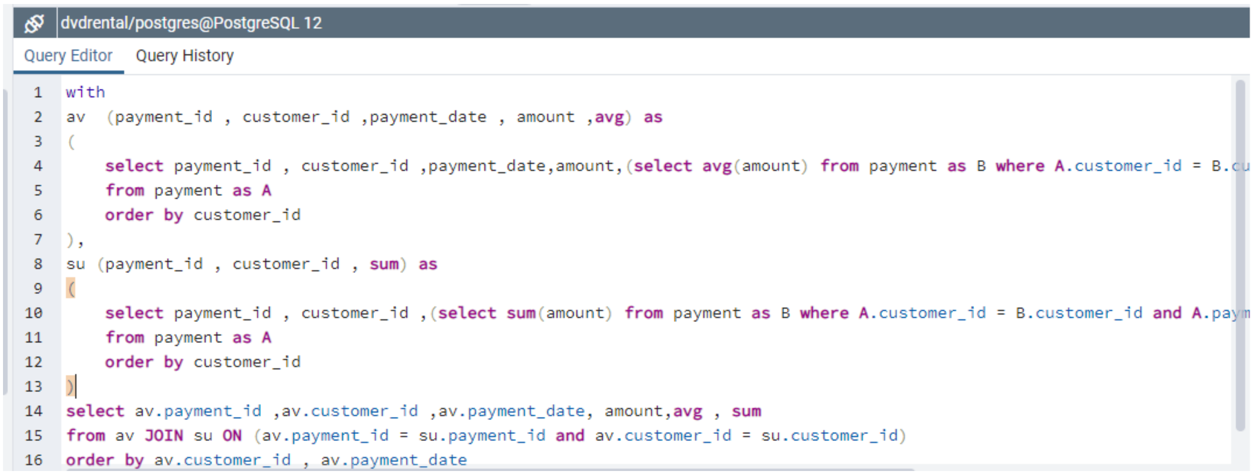
Query Editor   Query History

```
with
sumPcus (customer_id , amount_sum) as(
select customer_id , sum(amount)
from payment
group by customer_id
) ,
allnt as
(select first_name, last_name , ntile(4) over ( order by amount_sum desc ) as nt
from sumPcus JOIN customer ON (sumPcus.customer_id = customer.customer_id )
)
select first_name, last_name  from allnt where nt = 1
```

Data Output   Explain   Messages   Notifications

```
   first_name                  last_name
   character varying (45)      character varying (45)
1  Alan                        Kahn
2  Alex                        Gresham
3  Alexander                   Fennell
4  Alfred                      Casillas
5  Alice                       Stewart
```

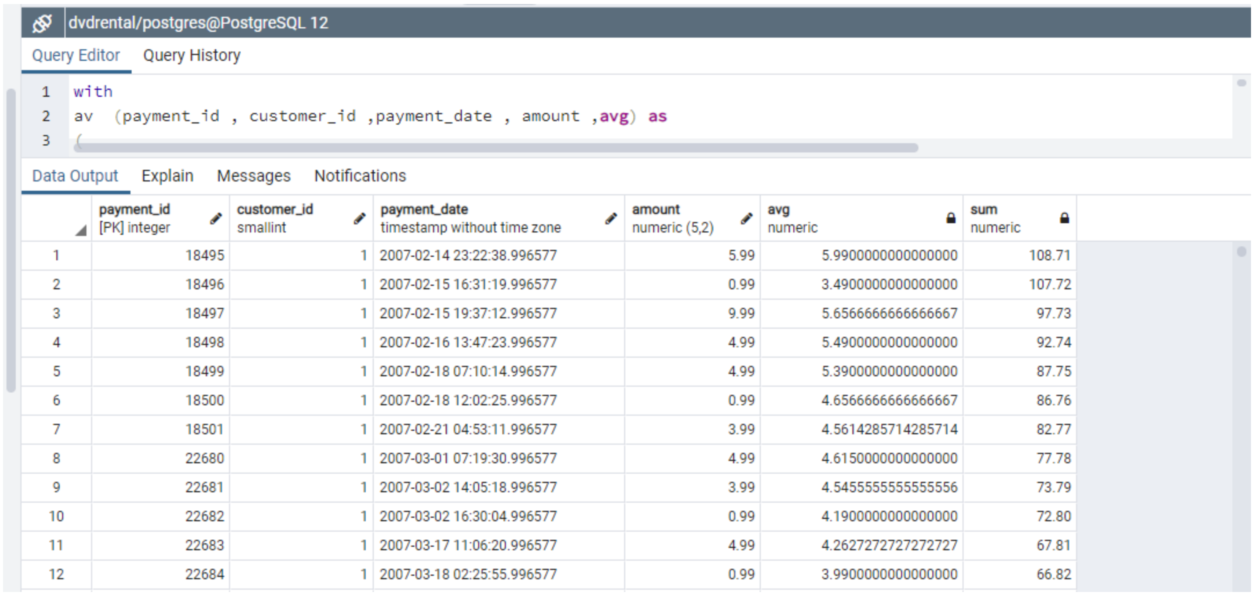
c.


dvdrental/postgres@PostgreSQL 12

Query Editor   Query History

```
with
av  (payment_id , customer_id ,payment_date , amount ,avg) as
(
    select payment_id , customer_id ,payment_date,amount,(select avg(amount) from payment as B where A.customer_id = B.cu
    from payment as A
    order by customer_id
),
su (payment_id , customer_id , sum) as
(
    select payment_id , customer_id ,(select sum(amount) from payment as B where A.customer_id = B.customer_id and A.paym
    from payment as A
    order by customer_id
)
select av.payment_id ,av.customer_id ,av.payment_date, amount,avg , sum
from av JOIN su ON (av.payment_id = su.payment_id and av.customer_id = su.customer_id)
order by av.customer_id , av.payment_date
```

R3c:



dvdrental/postgres@PostgreSQL 12

Query Editor Query History

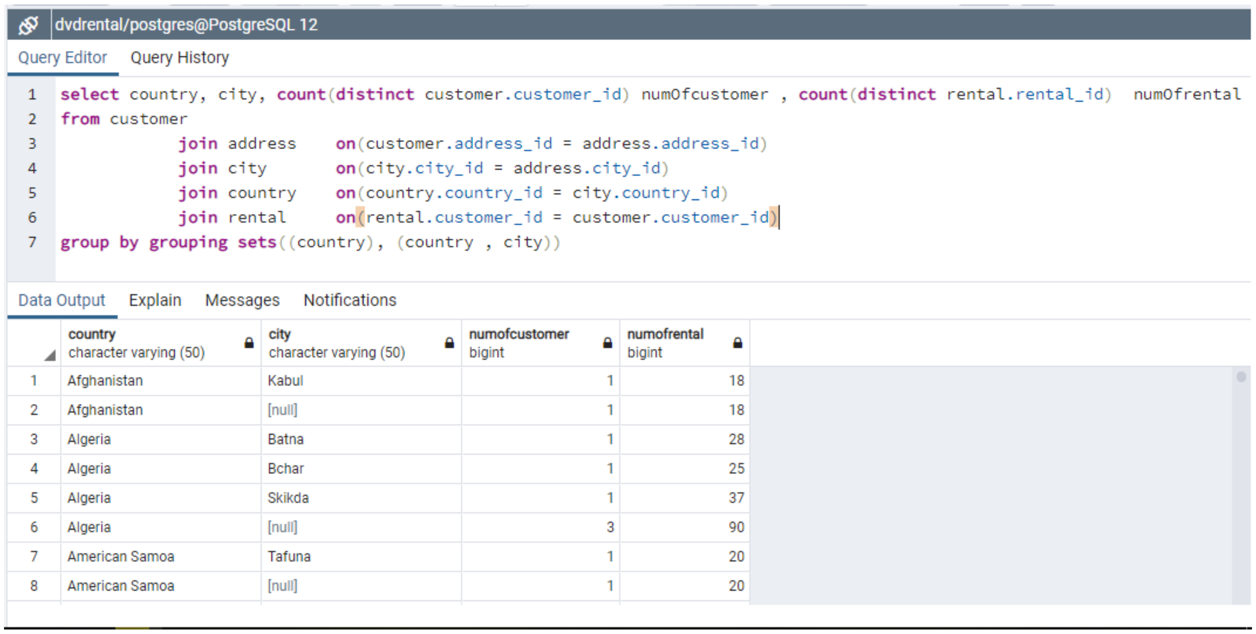
```sql
with
av  (payment_id , customer_id ,payment_date , amount ,avg) as
(
```

Data Output Explain Messages Notifications

	payment_id [PK] integer	customer_id smallint	payment_date timestamp without time zone	amount numeric (5,2)	avg numeric	sum numeric
1	18495	1	2007-02-14 23:22:38.996577	5.99	5.9900000000000000	108.71
2	18496	1	2007-02-15 16:31:19.996577	0.99	3.4900000000000000	107.72
3	18497	1	2007-02-15 19:37:12.996577	9.99	5.6566666666666667	97.73
4	18498	1	2007-02-16 13:47:23.996577	4.99	5.4900000000000000	92.74
5	18499	1	2007-02-18 07:10:14.996577	4.99	5.3900000000000000	87.75
6	18500	1	2007-02-18 12:02:25.996577	0.99	4.6566666666666667	86.76
7	18501	1	2007-02-21 04:53:11.996577	3.99	4.5614285714285714	82.77
8	22680	1	2007-03-01 07:19:30.996577	4.99	4.6150000000000000	77.78
9	22681	1	2007-03-02 14:05:18.996577	3.99	4.5455555555555556	73.79
10	22682	1	2007-03-02 16:30:04.996577	0.99	4.1900000000000000	72.80
11	22683	1	2007-03-17 11:06:20.996577	4.99	4.2627272727272727	67.81
12	22684	1	2007-03-18 02:25:55.996577	0.99	3.9900000000000000	66.82

d.

R3d:



dvdrental/postgres@PostgreSQL 12

Query Editor Query History

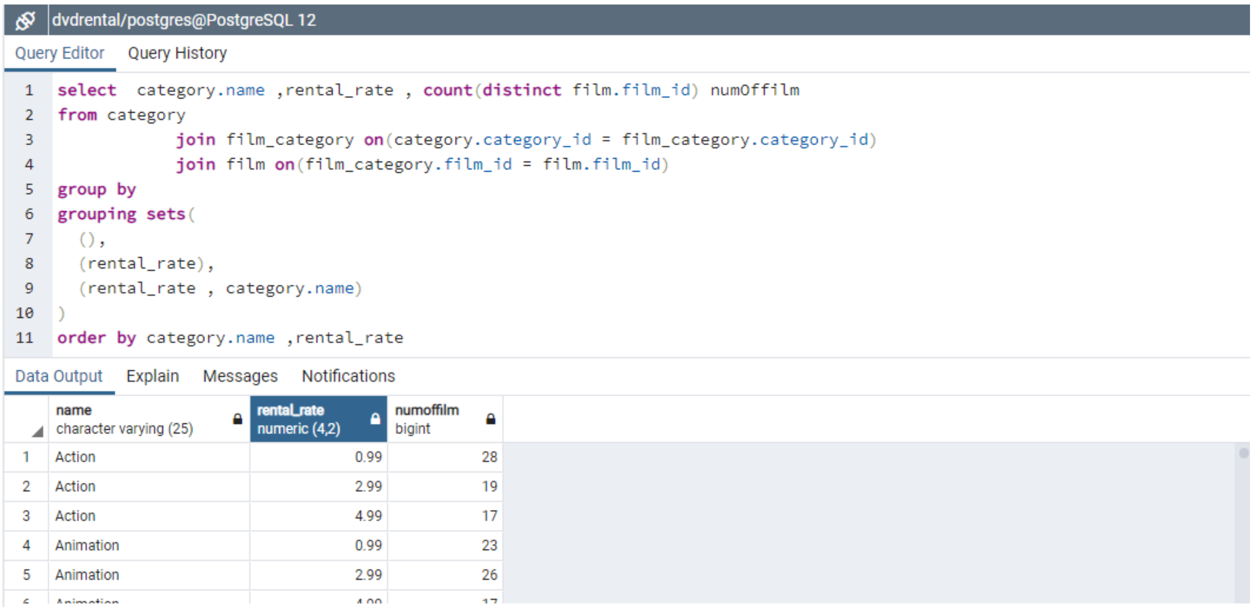
```sql
select country, city, count(distinct customer.customer_id) numOfcustomer , count(distinct rental.rental_id)  numOfrental
from customer
            join address    on(customer.address_id = address.address_id)
            join city       on(city.city_id = address.city_id)
            join country    on(country.country_id = city.country_id)
            join rental     on(rental.customer_id = customer.customer_id)
group by grouping sets((country), (country , city))
```

Data Output Explain Messages Notifications

	country character varying (50)	city character varying (50)	numofcustomer bigint	numofrental bigint
1	Afghanistan	Kabul	1	18
2	Afghanistan	[null]	1	18
3	Algeria	Batna	1	28
4	Algeria	Bchar	1	25
5	Algeria	Skikda	1	37
6	Algeria	[null]	3	90
7	American Samoa	Tafuna	1	20
8	American Samoa	[null]	1	20

e.

R3e:

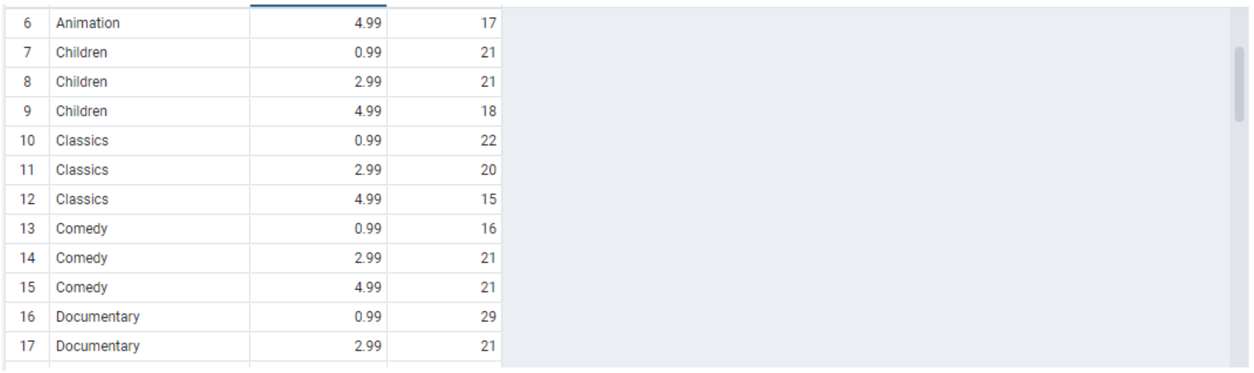


dvdrental/postgres@PostgreSQL 12

Query Editor Query History

```sql
select  category.name ,rental_rate , count(distinct film.film_id) numOffilm
from category
            join film_category on(category.category_id = film_category.category_id)
            join film on(film_category.film_id = film.film_id)
group by
grouping sets(
  (),
  (rental_rate),
  (rental_rate , category.name)
)
order by category.name ,rental_rate
```

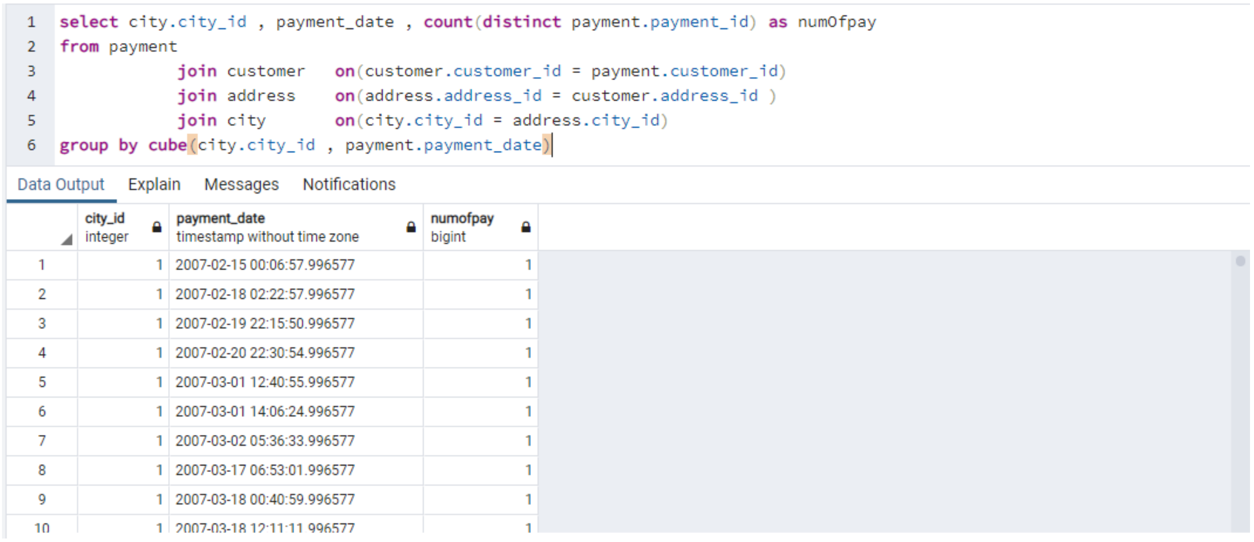
Data Output Explain Messages Notifications



	name character varying (25)	rental_rate numeric (4,2)	numoffilm bigint
1	Action	0.99	28
2	Action	2.99	19
3	Action	4.99	17
4	Animation	0.99	23
5	Animation	2.99	26
6	Animation	4.99	17
7	Children	0.99	21
8	Children	2.99	21
9	Children	4.99	18
10	Classics	0.99	22
11	Classics	2.99	20
12	Classics	4.99	15
13	Comedy	0.99	16
14	Comedy	2.99	21
15	Comedy	4.99	21
16	Documentary	0.99	29
17	Documentary	2.99	21

f.

R3f:



```sql
select city.city_id , payment_date , count(distinct payment.payment_id) as numOfpay
from payment
            join customer   on(customer.customer_id = payment.customer_id)
            join address    on(address.address_id = customer.address_id )
            join city       on(city.city_id = address.city_id)
group by cube(city.city_id , payment.payment_date)
```

Data Output | Explain | Messages | Notifications

	city_id integer	payment_date timestamp without time zone	numofpay bigint
1	1	2007-02-15 00:06:57.996577	1
2	1	2007-02-18 02:22:57.996577	1
3	1	2007-02-19 22:15:50.996577	1
4	1	2007-02-20 22:30:54.996577	1
5	1	2007-03-01 12:40:55.996577	1
6	1	2007-03-01 14:06:24.996577	1
7	1	2007-03-02 05:36:33.996577	1
8	1	2007-03-17 06:53:01.996577	1
9	1	2007-03-18 00:40:59.996577	1
10	1	2007-03-18 12:11:11.996577	1

4.

این سوال به صورت گروهی تحویل داده شده است

اعضای گروه : الناز رحمتی ، ریحانه حلوائی ، فاطمه نادی

که توسط خانم حلوائی آپلود شده است