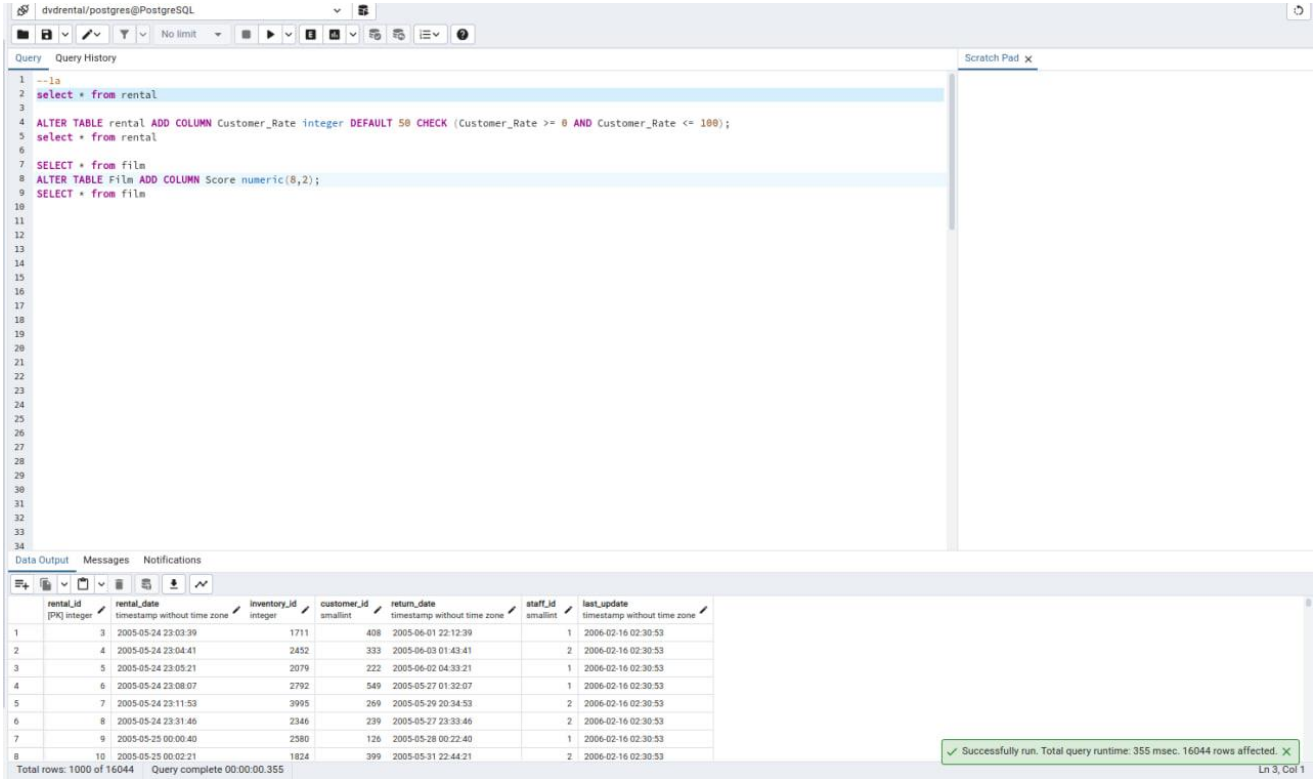
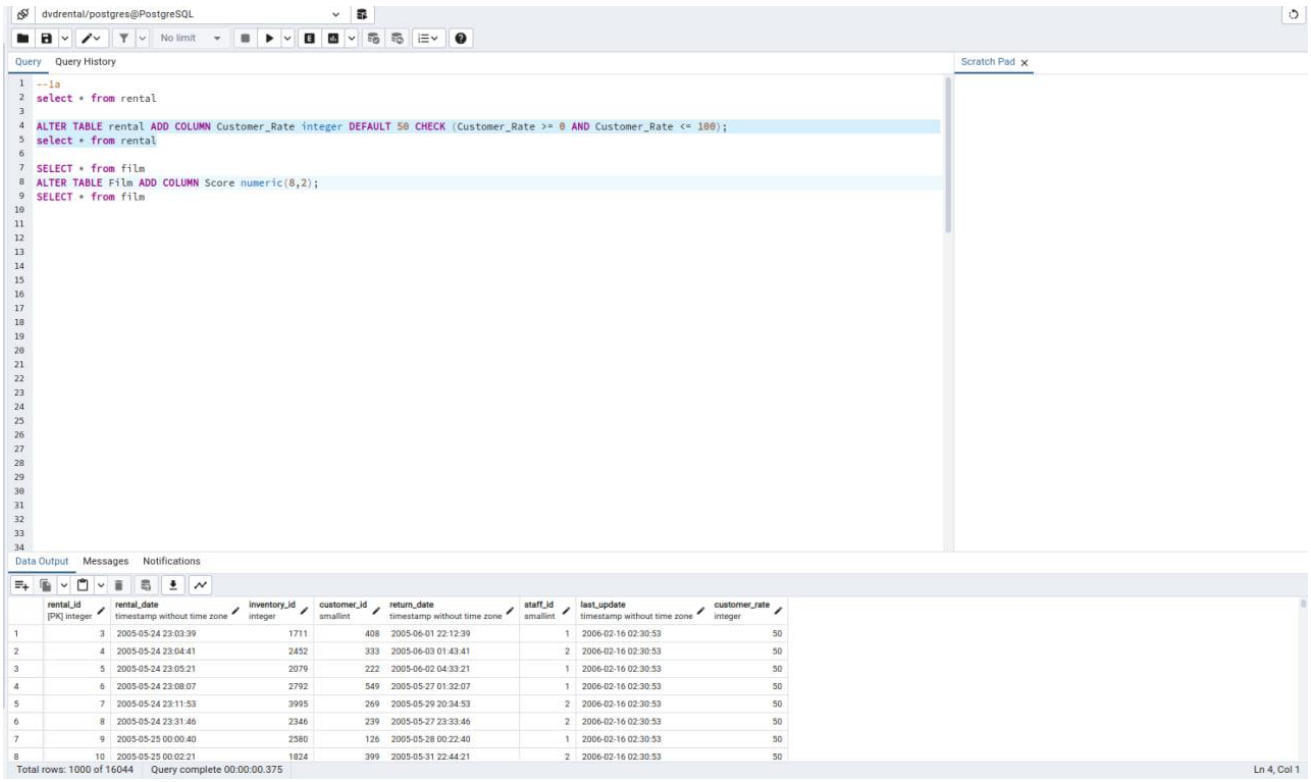
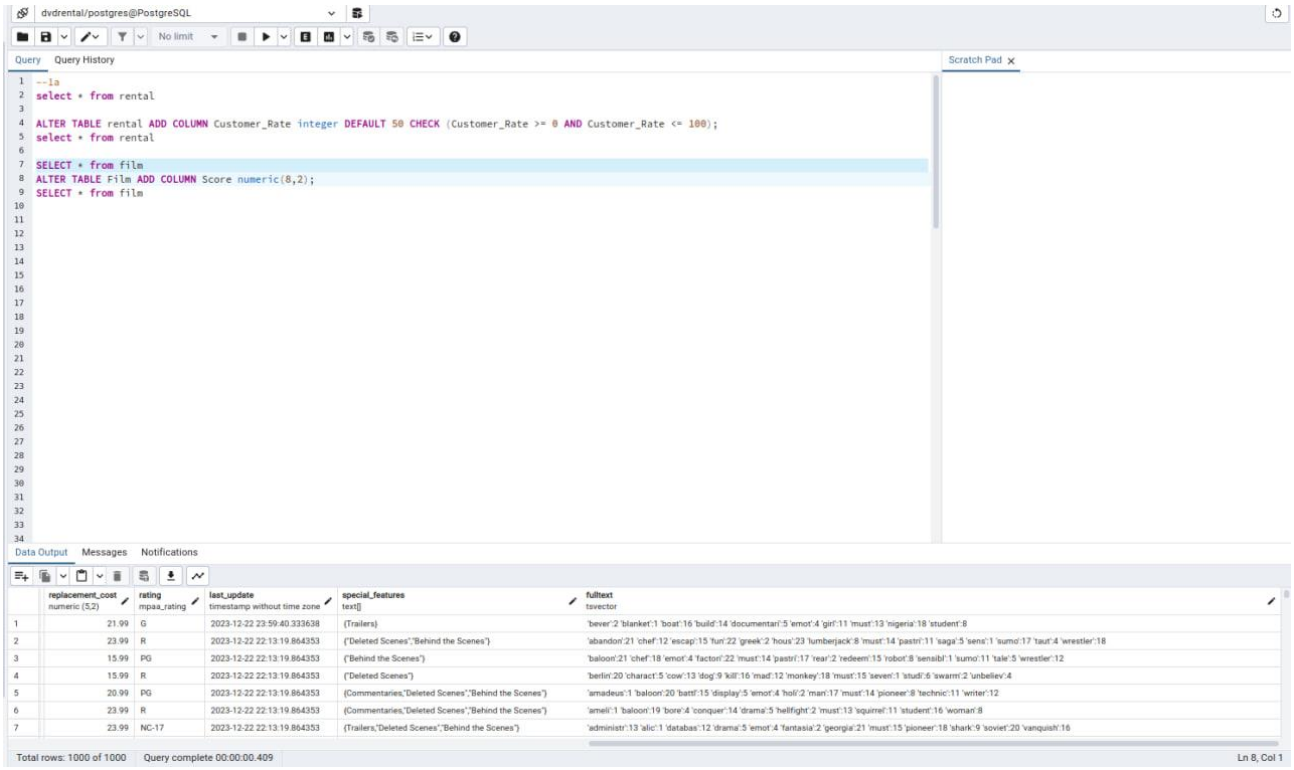
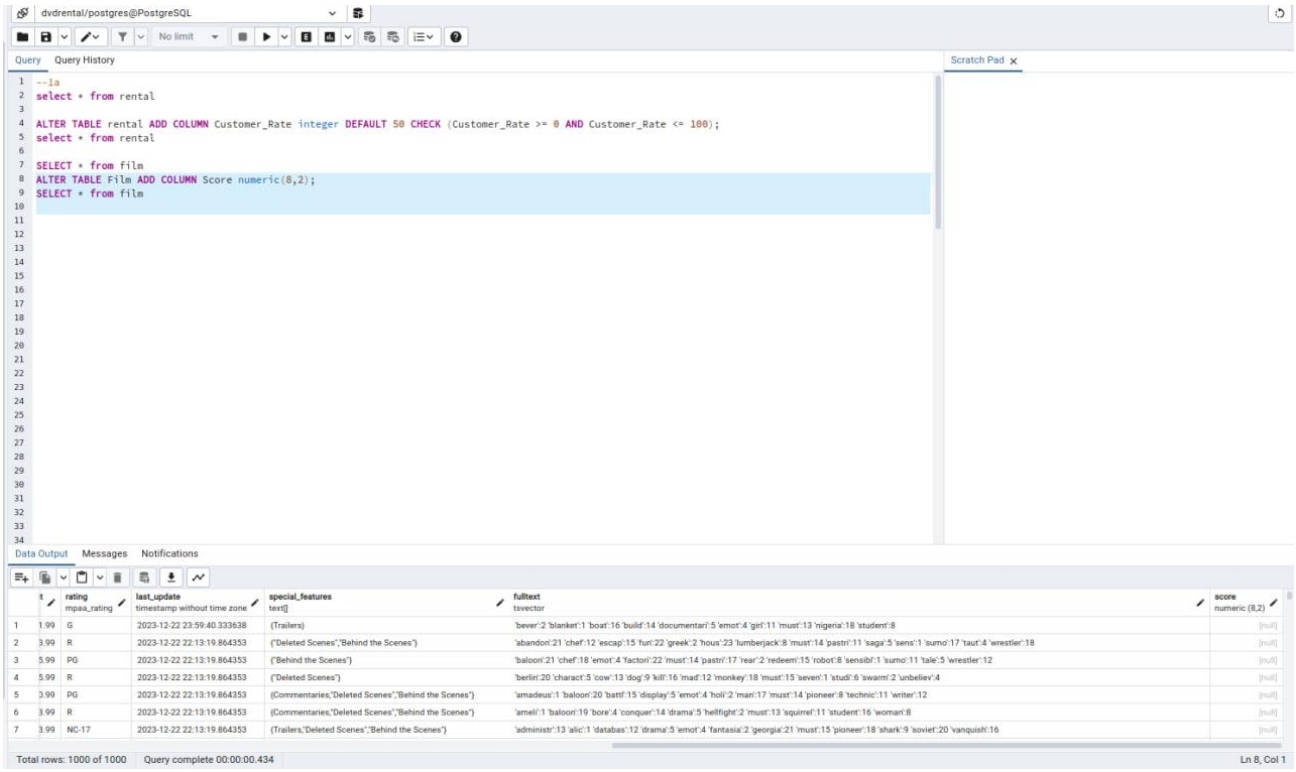
1a

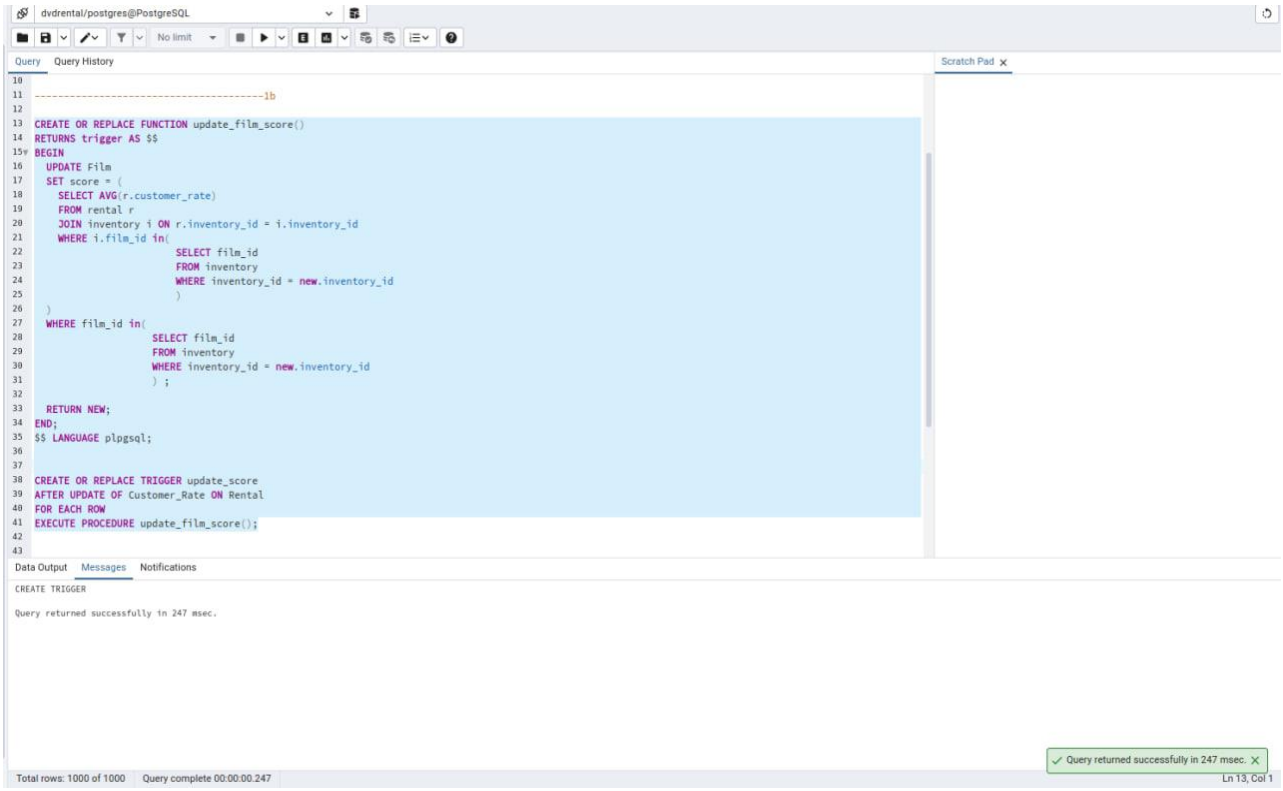

dvdrental/postgres@PostgreSQL
No limit
Query
Query History
Scratch Pad
--1a
select * from rental
ALTER TABLE rental ADD COLUMN Customer_Rate integer DEFAULT 50 CHECK (Customer_Rate >= 0 AND Customer_Rate <= 100);
select * from rental
SELECT * from film
ALTER TABLE Film ADD COLUMN Score numeric(8,2);
SELECT * from film
Data Output
Messages
Notifications
rental_id [PK] integer
rental_date timestamp without time zone
inventory_id integer
customer_id smallint
return_date timestamp without time zone
staff_id smallint
last_update timestamp without time zone
1 3 2005-05-24 23:03:39 1711 408 2005-06-01 22:12:39 1 2006-02-16 02:30:53
2 4 2005-05-24 23:04:41 2452 333 2005-06-03 01:43:41 2 2006-02-16 02:30:53
3 5 2005-05-24 23:05:21 2079 222 2005-06-02 04:33:21 1 2006-02-16 02:30:53
4 6 2005-05-24 23:08:07 2792 549 2005-05-27 01:32:07 1 2006-02-16 02:30:53
5 7 2005-05-24 23:11:53 3995 269 2005-05-29 20:34:53 2 2006-02-16 02:30:53
6 8 2005-05-24 23:31:46 2346 239 2005-05-27 23:33:46 2 2006-02-16 02:30:53
7 9 2005-05-25 00:00:40 2580 126 2005-05-28 00:22:40 1 2006-02-16 02:30:53
8 10 2005-05-25 00:02:21 1824 399 2005-05-31 22:44:21 2 2006-02-16 02:30:53
Successfully run. Total query runtime: 355 msec. 16044 rows affected.
Total rows: 1000 of 16044
Query complete 00:00:00.355
Ln 3, Col 1


dvdrental/postgres@PostgreSQL
No limit
Query
Query History
Scratch Pad
--1a
select * from rental
ALTER TABLE rental ADD COLUMN Customer_Rate integer DEFAULT 50 CHECK (Customer_Rate >= 0 AND Customer_Rate <= 100);
select * from rental
SELECT * from film
ALTER TABLE Film ADD COLUMN Score numeric(8,2);
SELECT * from film
Data Output
Messages
Notifications
rental_id [PK] integer
rental_date timestamp without time zone
inventory_id integer
customer_id smallint
return_date timestamp without time zone
staff_id smallint
last_update timestamp without time zone
customer_rate integer
1 3 2005-05-24 23:03:39 1711 408 2005-06-01 22:12:39 1 2006-02-16 02:30:53 50
2 4 2005-05-24 23:04:41 2452 333 2005-06-03 01:43:41 2 2006-02-16 02:30:53 50
3 5 2005-05-24 23:05:21 2079 222 2005-06-02 04:33:21 1 2006-02-16 02:30:53 50
4 6 2005-05-24 23:08:07 2792 549 2005-05-27 01:32:07 1 2006-02-16 02:30:53 50
5 7 2005-05-24 23:11:53 3995 269 2005-05-29 20:34:53 2 2006-02-16 02:30:53 50
6 8 2005-05-24 23:31:46 2346 239 2005-05-27 23:33:46 2 2006-02-16 02:30:53 50
7 9 2005-05-25 00:00:40 2580 126 2005-05-28 00:22:40 1 2006-02-16 02:30:53 50
8 10 2005-05-25 00:02:21 1824 399 2005-05-31 22:44:21 2 2006-02-16 02:30:53 50
Total rows: 1000 of 16044
Query complete 00:00:00.375
Ln 4, Col 1

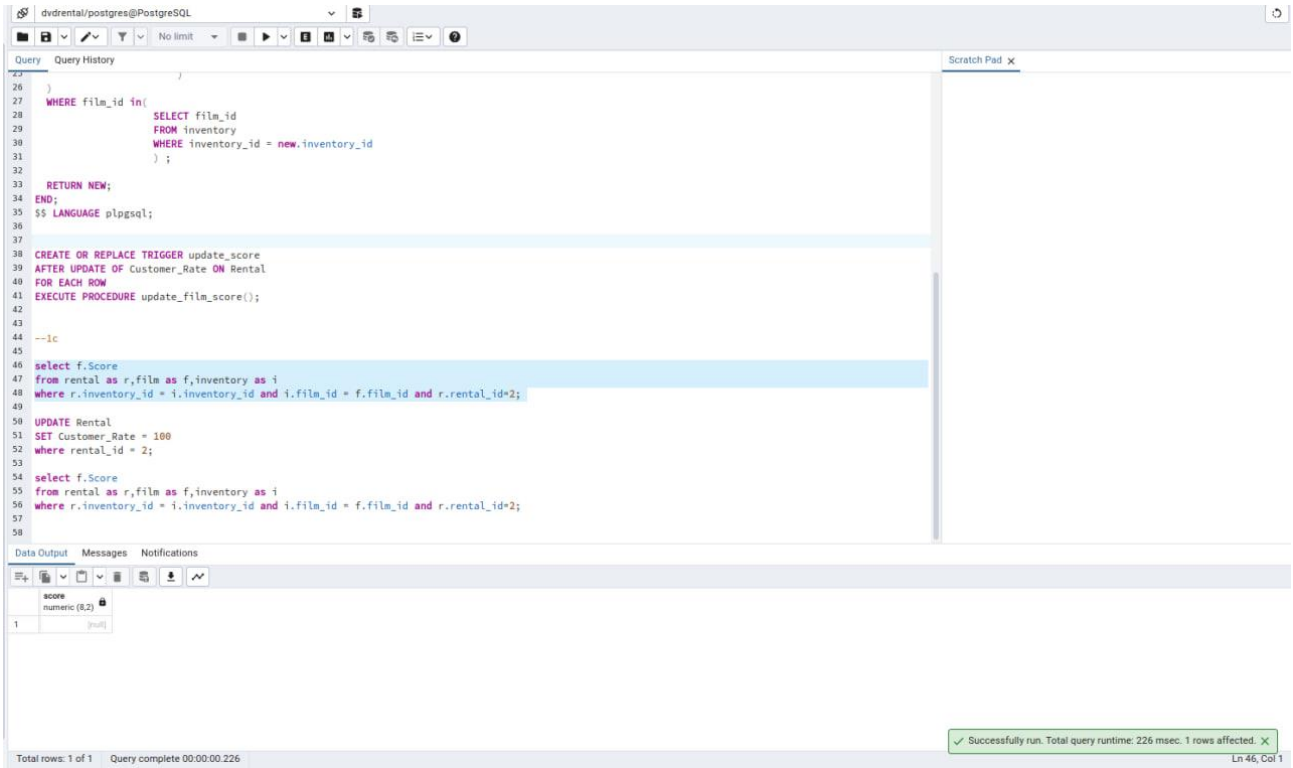

dvdrental/postgres@PostgreSQL
No limit
Query
Query History
Scratch Pad
--1a
select * from rental
ALTER TABLE rental ADD COLUMN Customer_Rate integer DEFAULT 50 CHECK (Customer_Rate >= 0 AND Customer_Rate <= 100);
select * from rental
SELECT * from film
ALTER TABLE Film ADD COLUMN Score numeric(8,2);
SELECT * from film
Data Output
Messages
Notifications
replacement_cost numeric (5,2)
rating mpaa_rating
last_update timestamp without time zone
special_features text[]
fulltext tsvector
1 21.99 G 2023-12-22 23:59:40.333638 {Trailers} 'bever':2 'blanket':1 'boat':16 'build':14 'documentari':5 'emot':4 'girl':11 'must':13 'nigeria':18 'student':8
2 23.99 R 2023-12-22 22:13:19.864353 {"Deleted Scenes","Behind the Scenes"} 'abandon':21 'chef':12 'escap':15 'fun':22 'greek':2 'hous':23 'lumberjack':8 'must':14 'pastri':11 'saga':5 'sens':1 'sumo':17 'taut':4 'wrestler':18
3 15.99 PG 2023-12-22 22:13:19.864353 {"Behind the Scenes"} 'baloon':21 'chef':18 'emot':4 'factori':22 'must':14 'pastri':17 'rear':2 'redeem':15 'robot':8 'sensibl':1 'sumo':11 'tale':5 'wrestler':12
4 15.99 R 2023-12-22 22:13:19.864353 {"Deleted Scenes"} 'berlin':20 'charact':5 'cow':13 'dog':9 'kill':16 'mad':12 'monkey':18 'must':15 'seven':1 'studi':6 'swarm':2 'unbeliev':4
5 20.99 PG 2023-12-22 22:13:19.864353 {Commentaries,"Deleted Scenes","Behind the Scenes"} 'amadeus':1 'baloon':20 'battl':15 'display':5 'emot':4 'holi':2 'man':17 'must':14 'pioneer':8 'technic':11 'writer':12
6 23.99 R 2023-12-22 22:13:19.864353 {Commentaries,"Deleted Scenes","Behind the Scenes"} 'ameli':1 'baloon':19 'bore':4 'conquer':14 'drama':5 'hellfight':2 'must':13 'squirrel':11 'student':16 'woman':8
7 23.99 NC-17 2023-12-22 22:13:19.864353 {Trailers,"Deleted Scenes","Behind the Scenes"} 'administr':13 'alic':1 'databas':12 'drama':5 'emot':4 'fantasia':2 'georgia':21 'must':15 'pioneer':18 'shark':9 'soviet':20 'vanquish':16
Total rows: 1000 of 1000
Query complete 00:00:00.409
Ln 8, Col 1

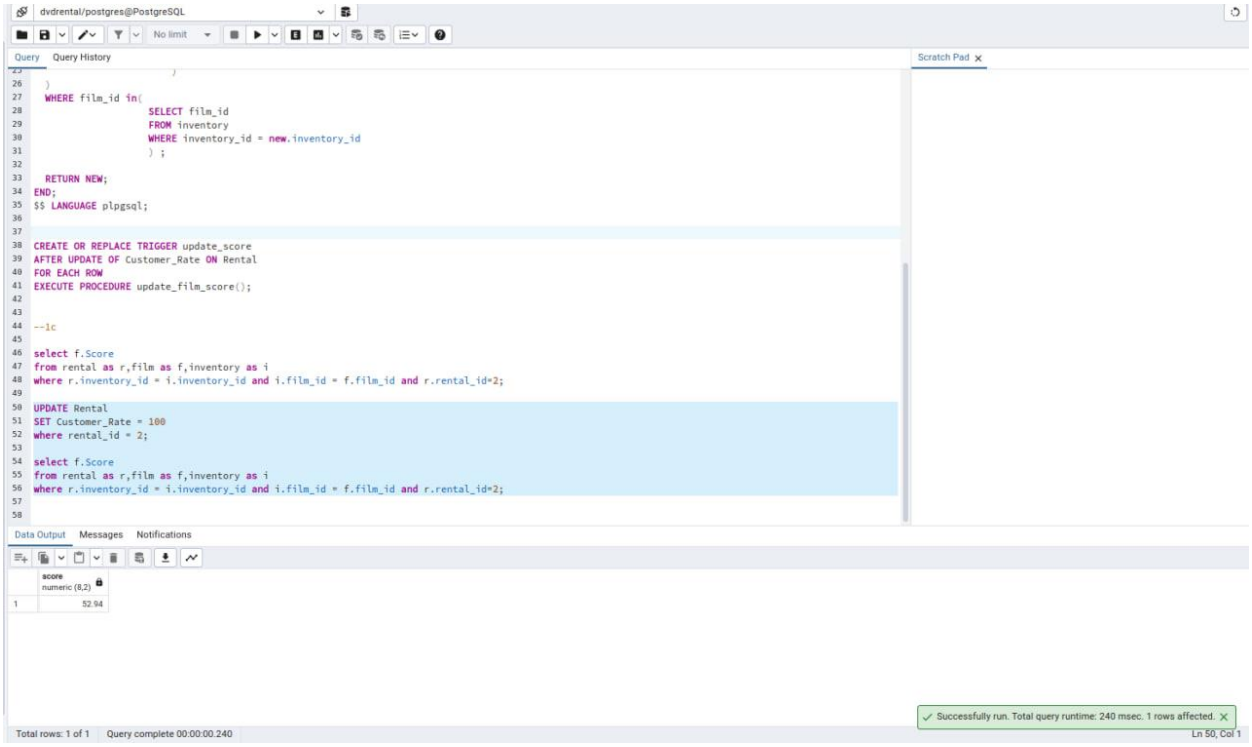

dvdrental/postgres@PostgreSQL
No limit
Query
Query History
Scratch Pad
--1a
select * from rental
ALTER TABLE rental ADD COLUMN Customer_Rate integer DEFAULT 50 CHECK (Customer_Rate >= 0 AND Customer_Rate <= 100);
select * from rental
SELECT * from film
ALTER TABLE Film ADD COLUMN Score numeric(8,2);
SELECT * from film
Data Output
Messages
Notifications
t
rating mpaa_rating
last_update timestamp without time zone
special_features text[]
fulltext tsvector
score numeric (8,2)
1 1.99 G 2023-12-22 23:59:40.333638 {Trailers} 'bever':2 'blanket':1 'boat':16 'build':14 'documentari':5 'emot':4 'girl':11 'must':13 'nigeria':18 'student':8 [null]
2 3.99 R 2023-12-22 22:13:19.864353 {"Deleted Scenes","Behind the Scenes"} 'abandon':21 'chef':12 'escap':15 'fun':22 'greek':2 'hous':23 'lumberjack':8 'must':14 'pastri':11 'saga':5 'sens':1 'sumo':17 'taut':4 'wrestler':18 [null]
3 5.99 PG 2023-12-22 22:13:19.864353 {"Behind the Scenes"} 'baloon':21 'chef':18 'emot':4 'factori':22 'must':14 'pastri':17 'rear':2 'redeem':15 'robot':8 'sensibl':1 'sumo':11 'tale':5 'wrestler':12 [null]
4 5.99 R 2023-12-22 22:13:19.864353 {"Deleted Scenes"} 'berlin':20 'charact':5 'cow':13 'dog':9 'kill':16 'mad':12 'monkey':18 'must':15 'seven':1 'studi':6 'swarm':2 'unbeliev':4 [null]
5 3.99 PG 2023-12-22 22:13:19.864353 {Commentaries,"Deleted Scenes","Behind the Scenes"} 'amadeus':1 'baloon':20 'battl':15 'display':5 'emot':4 'holi':2 'man':17 'must':14 'pioneer':8 'technic':11 'writer':12 [null]
6 3.99 R 2023-12-22 22:13:19.864353 {Commentaries,"Deleted Scenes","Behind the Scenes"} 'ameli':1 'baloon':19 'bore':4 'conquer':14 'drama':5 'hellfight':2 'must':13 'squirrel':11 'student':16 'woman':8 [null]
7 3.99 NC-17 2023-12-22 22:13:19.864353 {Trailers,"Deleted Scenes","Behind the Scenes"} 'administr':13 'alic':1 'databas':12 'drama':5 'emot':4 'fantasia':2 'georgia':21 'must':15 'pioneer':18 'shark':9 'soviet':20 'vanquish':16 [null]
Total rows: 1000 of 1000
Query complete 00:00:00.434
Ln 8, Col 1

1b


dvdrental/postgres@PostgreSQL
No limit
Query
Query History
Scratch Pad
----------1b
CREATE OR REPLACE FUNCTION update_film_score()
RETURNS trigger AS $$
BEGIN
UPDATE Film
SET score = (
SELECT AVG(r.customer_rate)
FROM rental r
JOIN inventory i ON r.inventory_id = i.inventory_id
WHERE i.film_id in(
SELECT film_id
FROM inventory
WHERE inventory_id = new.inventory_id
)
)
WHERE film_id in(
SELECT film_id
FROM inventory
WHERE inventory_id = new.inventory_id
) ;
RETURN NEW;
END;
$$ LANGUAGE plpgsql;
CREATE OR REPLACE TRIGGER update_score
AFTER UPDATE OF Customer_Rate ON Rental
FOR EACH ROW
EXECUTE PROCEDURE update_film_score();
Data Output
Messages
Notifications
CREATE TRIGGER
Query returned successfully in 247 msec.
Query returned successfully in 247 msec.
Total rows: 1000 of 1000
Query complete 00:00:00.247
Ln 13, Col 1

1c


dvdrental/postgres@PostgreSQL
No limit
Query
Query History
Scratch Pad
WHERE film_id in(
SELECT film_id
FROM inventory
WHERE inventory_id = new.inventory_id
) ;
RETURN NEW;
END;
$$ LANGUAGE plpgsql;
CREATE OR REPLACE TRIGGER update_score
AFTER UPDATE OF Customer_Rate ON Rental
FOR EACH ROW
EXECUTE PROCEDURE update_film_score();
--1c
select f.Score
from rental as r,film as f,inventory as i
where r.inventory_id = i.inventory_id and i.film_id = f.film_id and r.rental_id=2;
UPDATE Rental
SET Customer_Rate = 100
where rental_id = 2;
select f.Score
from rental as r,film as f,inventory as i
where r.inventory_id = i.inventory_id and i.film_id = f.film_id and r.rental_id=2;
Data Output
Messages
Notifications
score
numeric (8,2)
1
[null]
Successfully run. Total query runtime: 226 msec. 1 rows affected.
Total rows: 1 of 1
Query complete 00:00:00.226
Ln 46, Col 1


dvdrental/postgres@PostgreSQL
No limit
Query
Query History
Scratch Pad
)
WHERE film_id in(
SELECT film_id
FROM inventory
WHERE inventory_id = new.inventory_id
) ;
RETURN NEW;
END;
$$ LANGUAGE plpgsql;
CREATE OR REPLACE TRIGGER update_score
AFTER UPDATE OF Customer_Rate ON Rental
FOR EACH ROW
EXECUTE PROCEDURE update_film_score();
--1c
select f.Score
from rental as r,film as f,inventory as i
where r.inventory_id = i.inventory_id and i.film_id = f.film_id and r.rental_id=2;
UPDATE Rental
SET Customer_Rate = 100
where rental_id = 2;
select f.Score
from rental as r,film as f,inventory as i
where r.inventory_id = i.inventory_id and i.film_id = f.film_id and r.rental_id=2;
Data Output
Messages
Notifications
score
numeric (8,2)
1
52.94
Successfully run. Total query runtime: 240 msec. 1 rows affected.
Total rows: 1 of 1
Query complete 00:00:00.240
Ln 50, Col 1

2-

Query | Query History

```
58
59  --2
60  CREATE OR REPLACE FUNCTION get_actor_films(p_actor_id integer)
61  RETURNS TABLE(film_id integer, rental_count bigint, customer_satisfaction numeric)
62  AS $$
63  BEGIN
64      RETURN QUERY
65      SELECT f.film_id, COUNT(r.rental_id)  , CAse when AVG(r.customer_rate) is NULL then 0 else AVG(r.customer_rate) end
66      FROM film f
67      JOIN film_actor fa ON f.film_id = fa.film_id
68      left JOIN inventory i ON f.film_id = i.film_id
69      left JOIN rental r ON i.inventory_id = r.inventory_id
70      WHERE fa.actor_id = p_actor_id
71      GROUP BY f.film_id
72      order by f.film_id;
73  END;
74  $$ LANGUAGE plpgsql;
75
76  select * from get_actor_films(42)
77
78  --3
79  WITH film_summary AS (
80    SELECT
```

quartile Aa

Data Output | Messages | Notifications

	film_id integer	rental_count bigint	customer_satisfaction numeric
15	673	14	50.0000000000000000
16	687	26	50.0000000000000000
17	713	0	0
18	738	33	50.0000000000000000
19	798	12	50.0000000000000000
20	861	21	50.0000000000000000
21	865	24	50.0000000000000000

Total rows: 27 of 27 | Query complete 00:00:00.051 | Ln 60, Col 1

3-

Query | Query History

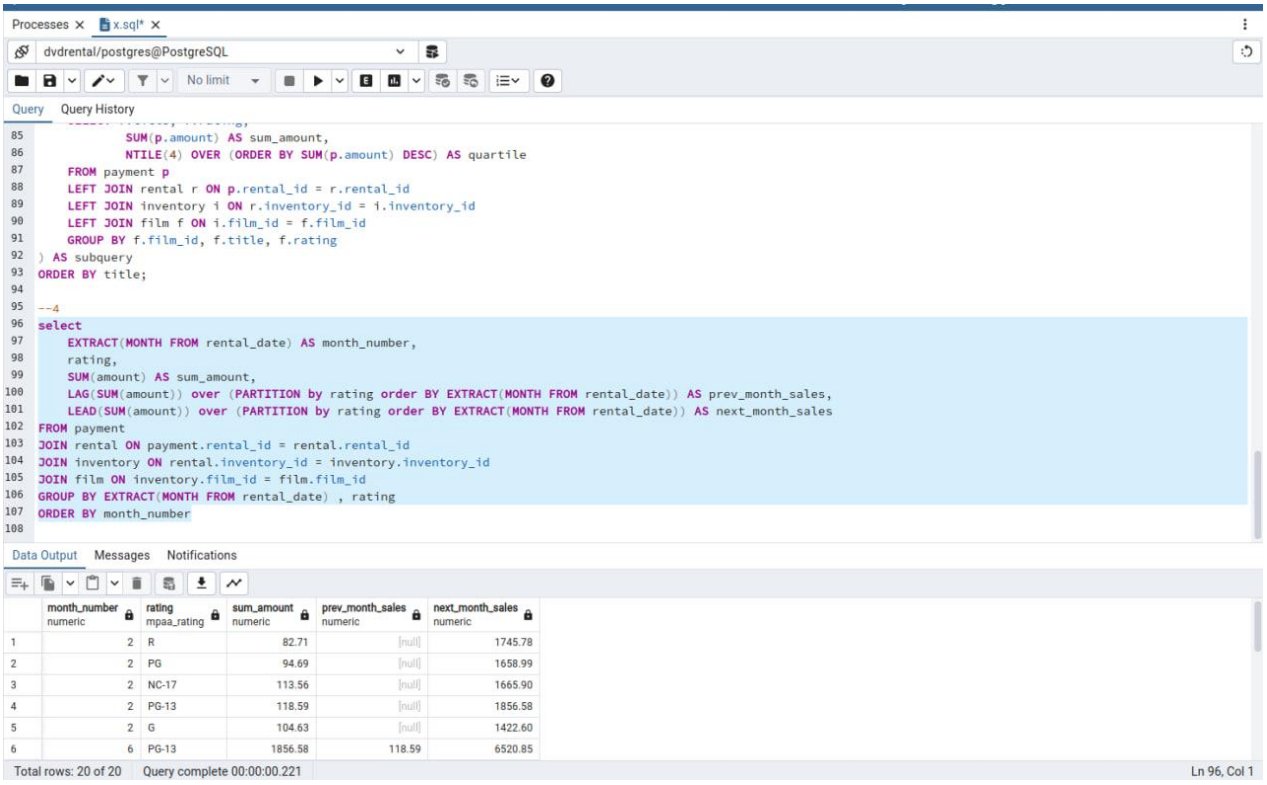
```
76  select * from get_actor_films(42)
77
78  --3
79  SELECT title, rating,
80    row_number() OVER (ORDER BY sum_amount DESC) AS rank_in_all,
81    dense_rank() OVER (PARTITION BY rating ORDER BY sum_amount DESC) AS rank_in_rating,
82    sum_amount,
83    CASE WHEN quartile = 4 THEN 1 ELSE 0 END AS is_in_fourth_quartile
84  FROM (
85    SELECT f.title, f.rating,
86           SUM(p.amount) AS sum_amount,
87           NTILE(4) OVER (ORDER BY SUM(p.amount) DESC) AS quartile
88    FROM payment p
89    left JOIN rental r ON p.rental_id = r.rental_id
90    left JOIN inventory i ON r.inventory_id = i.inventory_id
91    left JOIN film f ON i.film_id = f.film_id
92    GROUP BY f.film_id, f.title, f.rating
93  ) AS subquery
94  ORDER BY title;
95
96  --4
97
98
```

Data Output | Messages | Notifications

	title character varying (255)	rating mpaa_rating	rank_in_all bigint	rank_in_rating bigint	sum_amount numeric	is_in_fourth_quartile integer
1	Academy Dinosaur	PG	718	142	33.79	0
2	Ace Goldfinger	G	508	88	52.93	0
3	Adaptation Holes	NC-17	700	130	34.89	0
4	Affair Prejudice	G	261	46	83.79	0
5	African Egg	G	557	97	47.89	0
6	Agent Truman	PG	118	23	111.81	0
7	Airplane Sierra	PG-13	264	62	82.85	0

Total rows: 958 of 958 | Query complete 00:00:00.107 | Ln 79, Col 1

4-

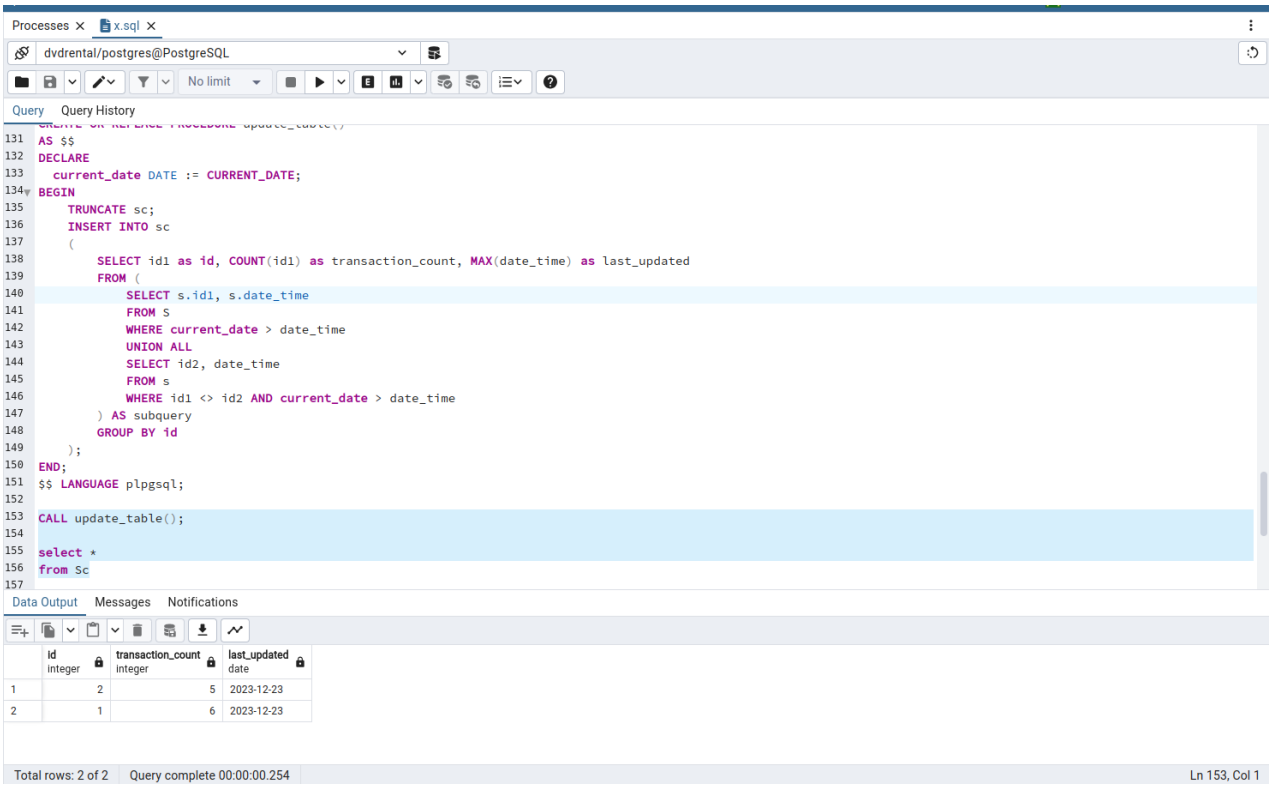


```
            SUM(p.amount) AS sum_amount,
            NTILE(4) OVER (ORDER BY SUM(p.amount) DESC) AS quartile
    FROM payment p
    LEFT JOIN rental r ON p.rental_id = r.rental_id
    LEFT JOIN inventory i ON r.inventory_id = i.inventory_id
    LEFT JOIN film f ON i.film_id = f.film_id
    GROUP BY f.film_id, f.title, f.rating
) AS subquery
ORDER BY title;

--4
select
    EXTRACT(MONTH FROM rental_date) AS month_number,
    rating,
    SUM(amount) AS sum_amount,
    LAG(SUM(amount)) over (PARTITION by rating order BY EXTRACT(MONTH FROM rental_date)) AS prev_month_sales,
    LEAD(SUM(amount)) over (PARTITION by rating order BY EXTRACT(MONTH FROM rental_date)) AS next_month_sales
FROM payment
JOIN rental ON payment.rental_id = rental.rental_id
JOIN inventory ON rental.inventory_id = inventory.inventory_id
JOIN film ON inventory.film_id = film.film_id
GROUP BY EXTRACT(MONTH FROM rental_date) , rating
ORDER BY month_number
```

	month_number numeric	rating mpaa_rating	sum_amount numeric	prev_month_sales numeric	next_month_sales numeric
1	2	R	82.71	[null]	1745.78
2	2	PG	94.69	[null]	1658.99
3	2	NC-17	113.56	[null]	1665.90
4	2	PG-13	118.59	[null]	1856.58
5	2	G	104.63	[null]	1422.60
6	6	PG-13	1856.58	118.59	6520.85

Total rows: 20 of 20 Query complete 00:00:00.221 Ln 96, Col 1

5-

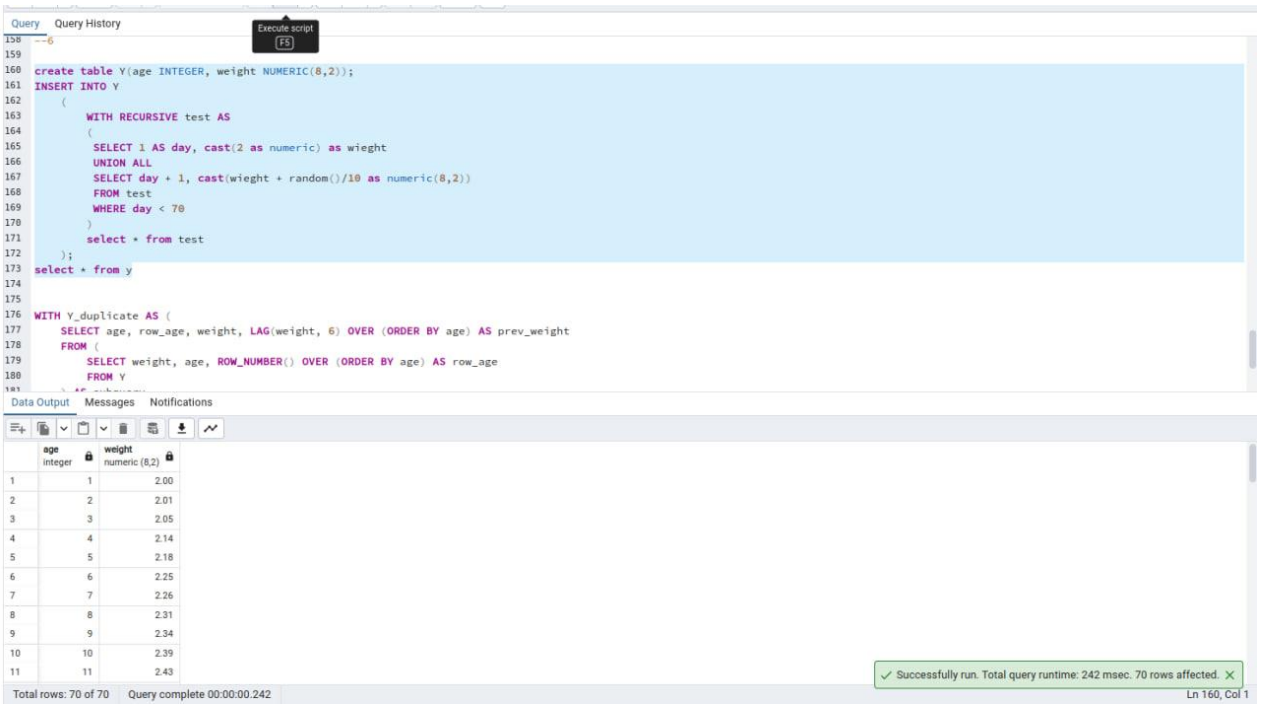


```
AS $$
DECLARE
  current_date DATE := CURRENT_DATE;
BEGIN
    TRUNCATE sc;
    INSERT INTO sc
    (
        SELECT id1 as id, COUNT(id1) as transaction_count, MAX(date_time) as last_updated
        FROM (
            SELECT s.id1, s.date_time
            FROM S
            WHERE current_date > date_time
            UNION ALL
            SELECT id2, date_time
            FROM s
            WHERE id1 <> id2 AND current_date > date_time
        ) AS subquery
        GROUP BY id
    );
END;
$$ LANGUAGE plpgsql;

CALL update_table();

select *
from Sc
```

	id integer	transaction_count integer	last_updated date
1	2	5	2023-12-23
2	1	6	2023-12-23

Total rows: 2 of 2 Query complete 00:00:00.254 Ln 153, Col 1

6-



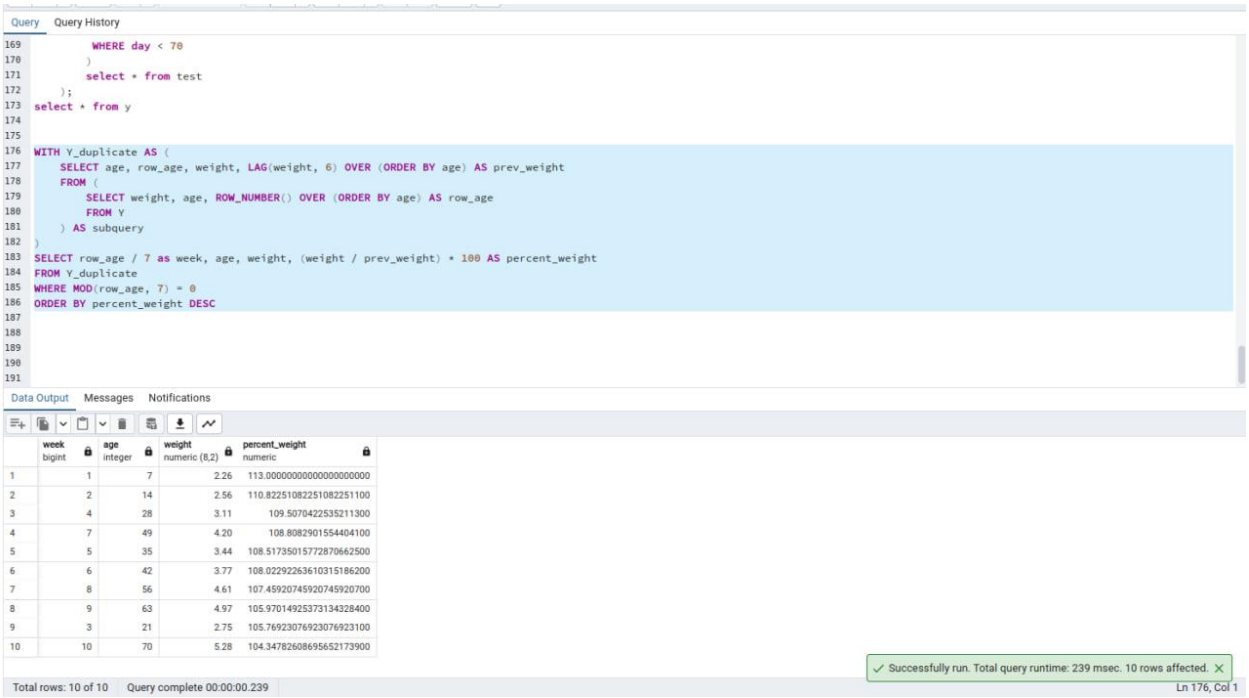
```
--6

create table Y(age INTEGER, weight NUMERIC(8,2));
INSERT INTO Y
    (
        WITH RECURSIVE test AS
        (
         SELECT 1 AS day, cast(2 as numeric) as wieght
         UNION ALL
         SELECT day + 1, cast(wieght + random()/10 as numeric(8,2))
         FROM test
         WHERE day < 70
        )
        select * from test
    );
select * from y


WITH Y_duplicate AS (
    SELECT age, row_age, weight, LAG(weight, 6) OVER (ORDER BY age) AS prev_weight
    FROM (
        SELECT weight, age, ROW_NUMBER() OVER (ORDER BY age) AS row_age
        FROM Y
```

	age integer	weight numeric (8,2)
1	1	2.00
2	2	2.01
3	3	2.05
4	4	2.14
5	5	2.18
6	6	2.25
7	7	2.26
8	8	2.31
9	9	2.34
10	10	2.39
11	11	2.43

Successfully run. Total query runtime: 242 msec. 70 rows affected.

Total rows: 70 of 70 Query complete 00:00:00.242 Ln 160, Col 1

ساختن جدول



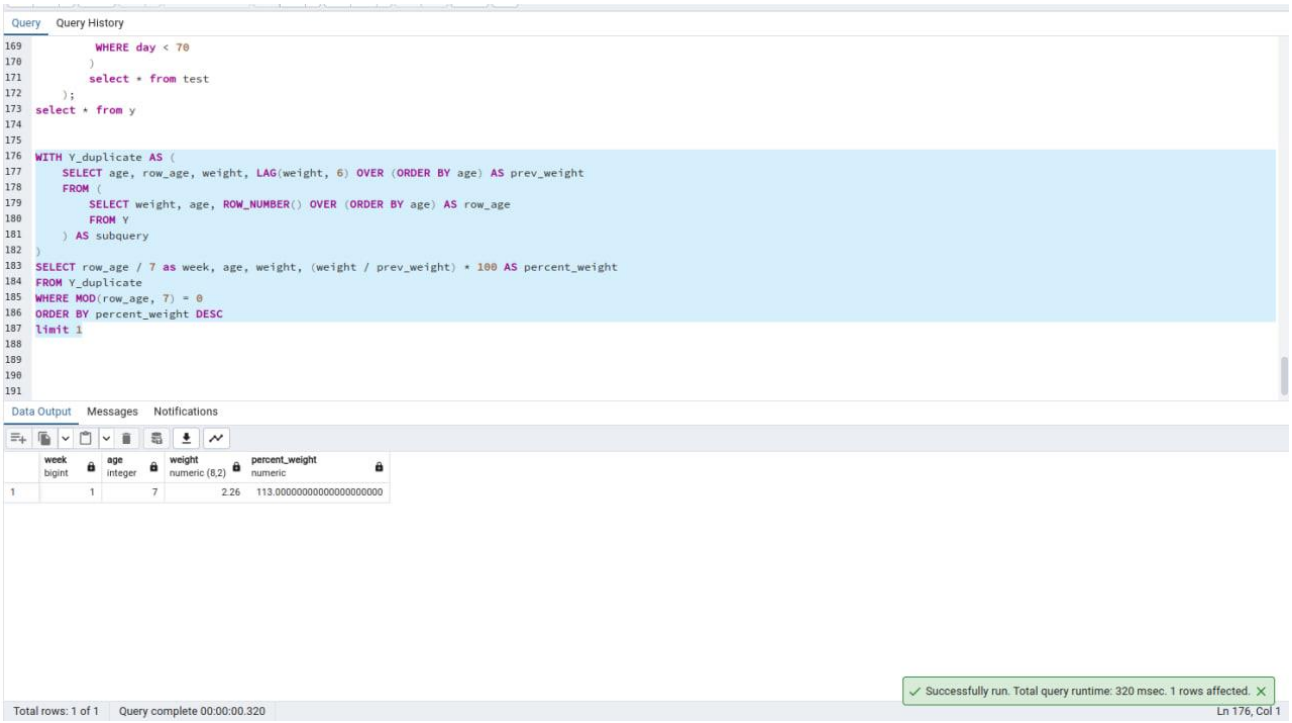
```
        WHERE day < 70
        )
        select * from test
    );
select * from y


WITH Y_duplicate AS (
    SELECT age, row_age, weight, LAG(weight, 6) OVER (ORDER BY age) AS prev_weight
    FROM (
        SELECT weight, age, ROW_NUMBER() OVER (ORDER BY age) AS row_age
        FROM Y
    ) AS subquery
)
SELECT row_age / 7 as week, age, weight, (weight / prev_weight) * 100 AS percent_weight
FROM Y_duplicate
WHERE MOD(row_age, 7) = 0
ORDER BY percent_weight DESC
```

	week bigint	age integer	weight numeric (8,2)	percent_weight numeric
1	1	7	2.26	113.00000000000000000000
2	2	14	2.56	110.82251082251082251100
3	4	28	3.11	109.5070422535211300
4	7	49	4.20	108.8082901554404100
5	5	35	3.44	108.51735015772870662500
6	6	42	3.77	108.02292263610315186200
7	8	56	4.61	107.45920745920745920700
8	9	63	4.97	105.97014925373134328400
9	3	21	2.75	105.76923076923076923100
10	10	70	5.28	104.34782608695652173900

Successfully run. Total query runtime: 239 msec. 10 rows affected.

Total rows: 10 of 10 Query complete 00:00:00.239 Ln 176, Col 1

نسبت رشد در هر هفته برای خوانایی بیشتر ۱۰۰ برابر شده است



لیمیت گذاشتن برای نمایش ۱ رکورد