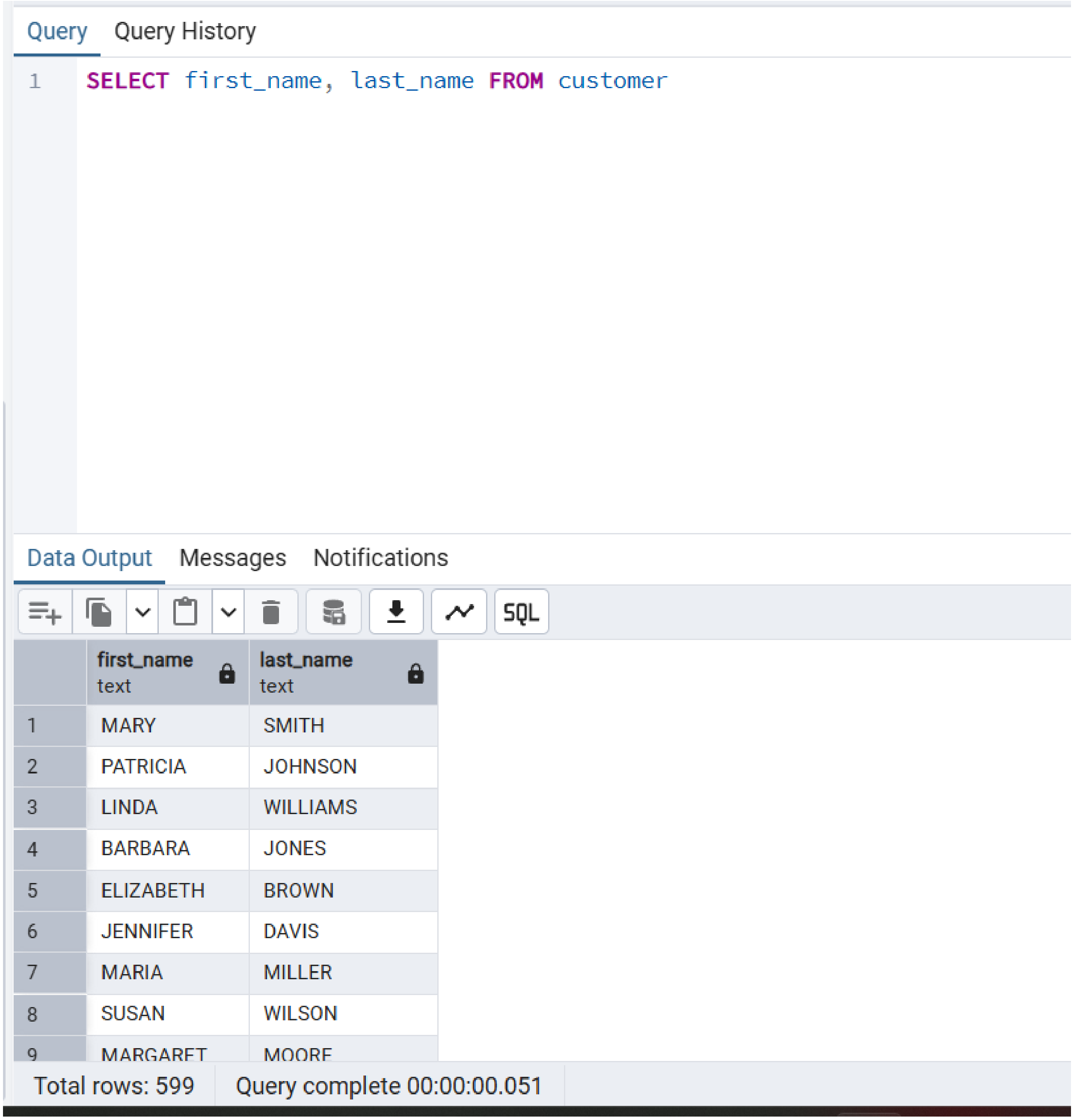
LAB 1 task

الاسم: شهاب شريف محمد الغريب

1- List all customers' first and last names.

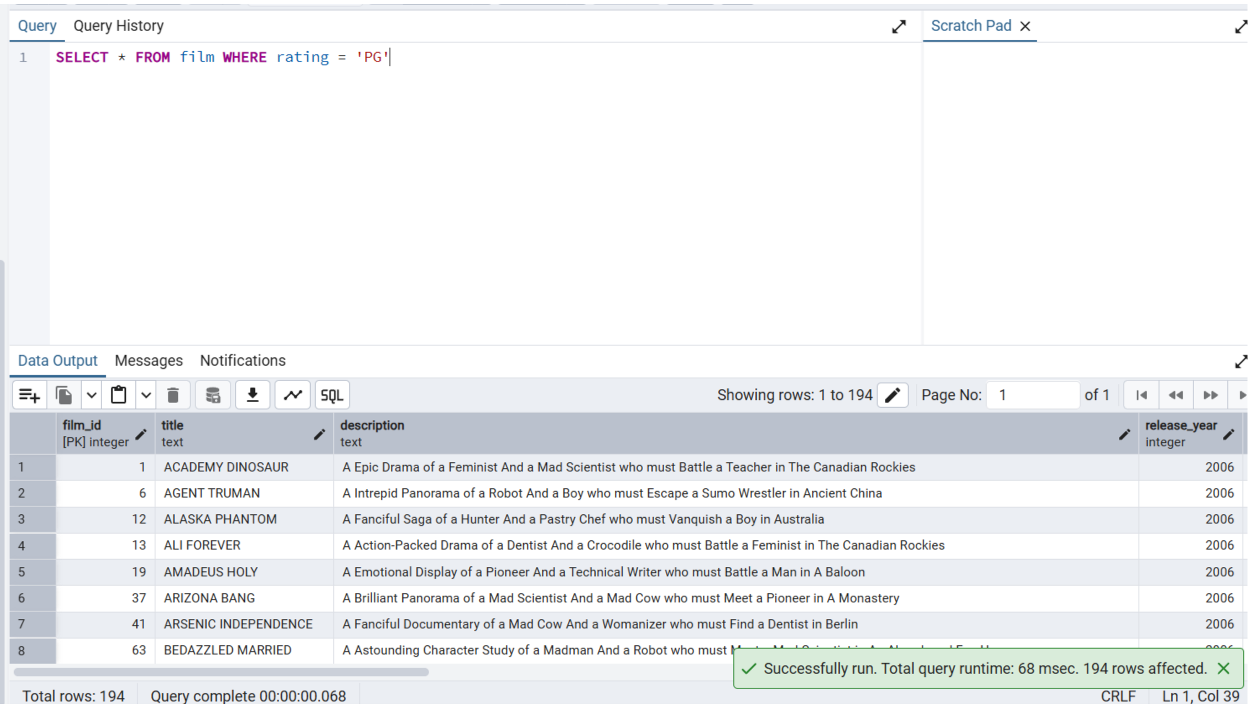


```sql
SELECT first_name, last_name FROM customer
```

	first_name text	last_name text
1	MARY	SMITH
2	PATRICIA	JOHNSON
3	LINDA	WILLIAMS
4	BARBARA	JONES
5	ELIZABETH	BROWN
6	JENNIFER	DAVIS
7	MARIA	MILLER
8	SUSAN	WILSON
9	MARGARET	MOORE

```
Total rows: 599    Query complete 00:00:00.051
```

2- Show all films with a rating of 'PG'.



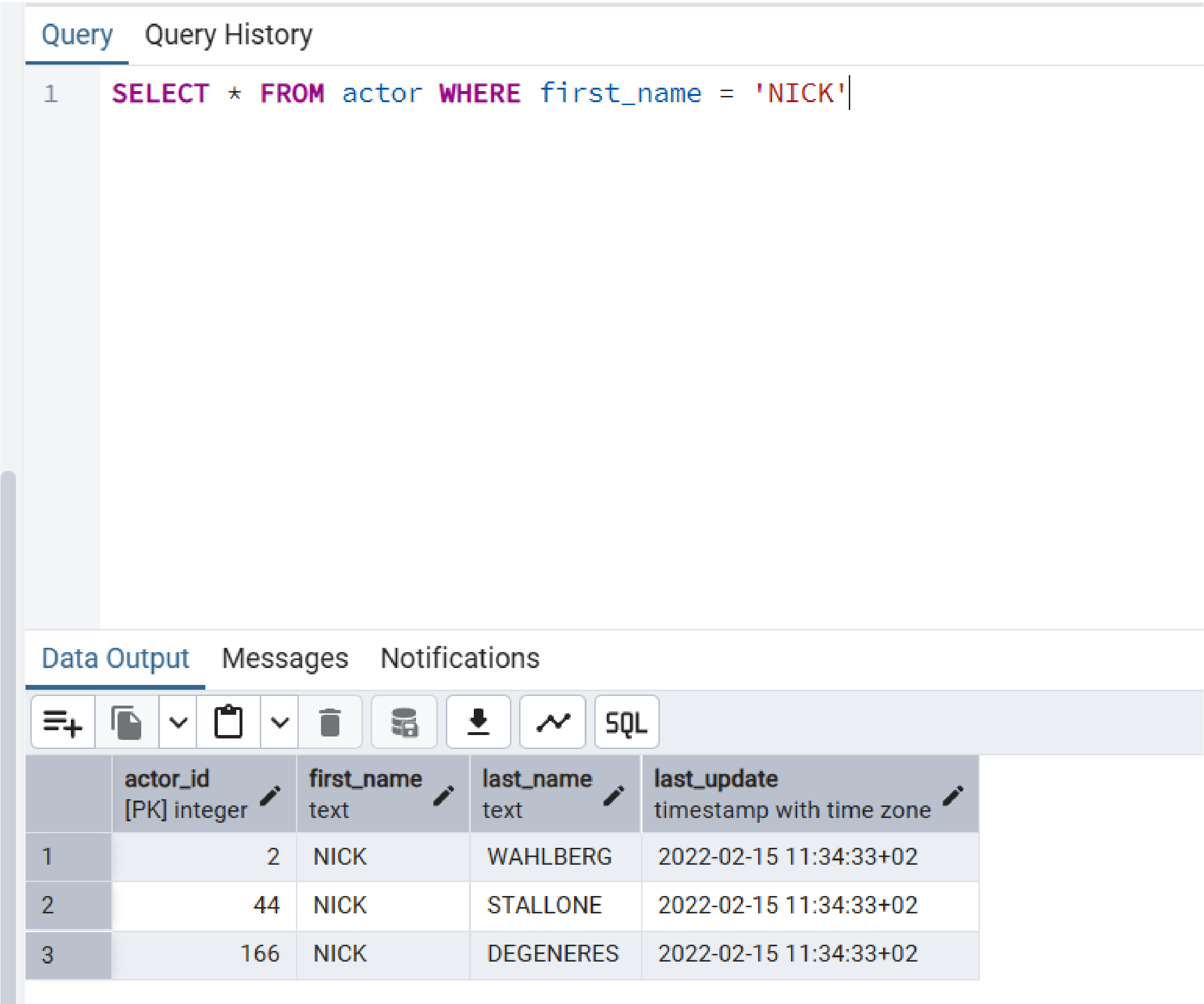
```sql
SELECT * FROM film WHERE rating = 'PG'
```

	film_id [PK] integer	title text	description text	release_year integer
1	1	ACADEMY DINOSAUR	A Epic Drama of a Feminist And a Mad Scientist who must Battle a Teacher in The Canadian Rockies	2006
2	6	AGENT TRUMAN	A Intrepid Panorama of a Robot And a Boy who must Escape a Sumo Wrestler in Ancient China	2006
3	12	ALASKA PHANTOM	A Fanciful Saga of a Hunter And a Pastry Chef who must Vanquish a Boy in Australia	2006
4	13	ALI FOREVER	A Action-Packed Drama of a Dentist And a Crocodile who must Battle a Feminist in The Canadian Rockies	2006
5	19	AMADEUS HOLY	A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon	2006
6	37	ARIZONA BANG	A Brilliant Panorama of a Mad Scientist And a Mad Cow who must Meet a Pioneer in A Monastery	2006
7	41	ARSENIC INDEPENDENCE	A Fanciful Documentary of a Mad Cow And a Womanizer who must Find a Dentist in Berlin	2006
8	63	BEDAZZLED MARRIED	A Astounding Character Study of a Madman And a Robot who must [illegible]	[illegible]

✓ Successfully run. Total query runtime: 68 msec. 194 rows affected. ✕

Total rows: 194 Query complete 00:00:00.068 CRLF Ln 1, Col 39

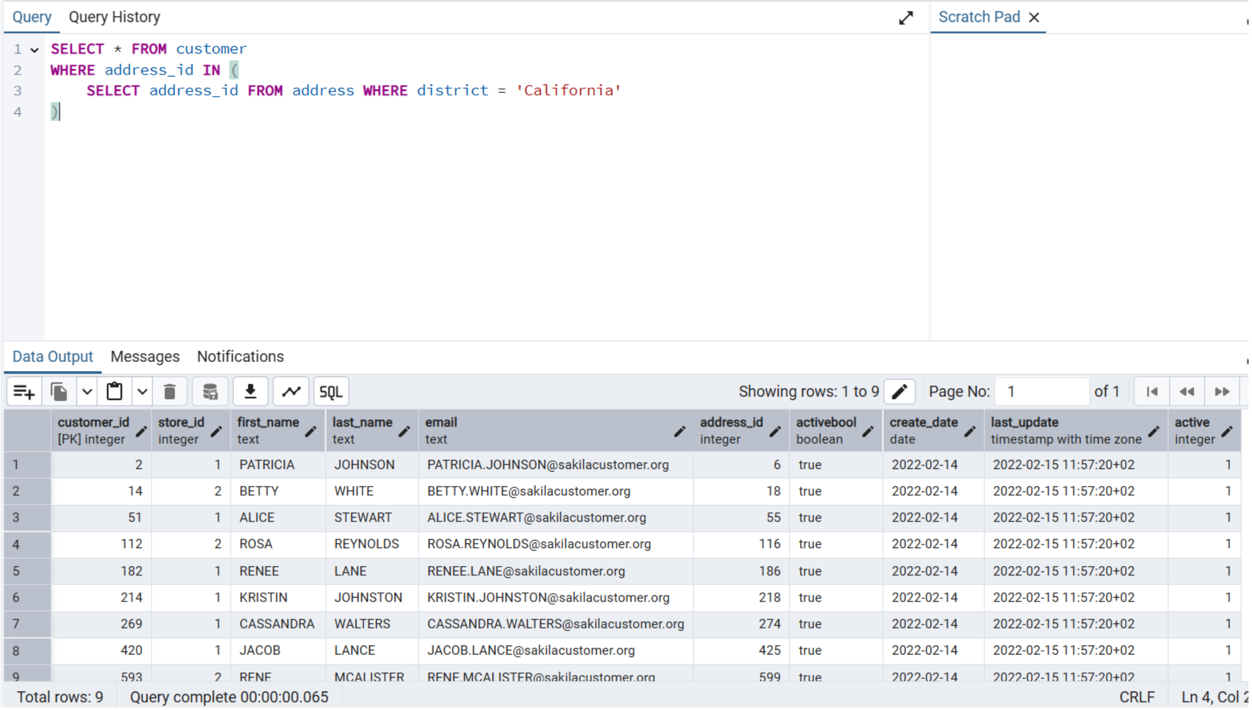
3- List all actors with the first name 'Nick'.



```
SELECT * FROM actor WHERE first_name = 'NICK'
```

	actor_id [PK] integer	first_name text	last_name text	last_update timestamp with time zone
1	2	NICK	WAHLBERG	2022-02-15 11:34:33+02
2	44	NICK	STALLONE	2022-02-15 11:34:33+02
3	166	NICK	DEGENERES	2022-02-15 11:34:33+02

4- Get all customers who live in the 'California'. (using subquery)



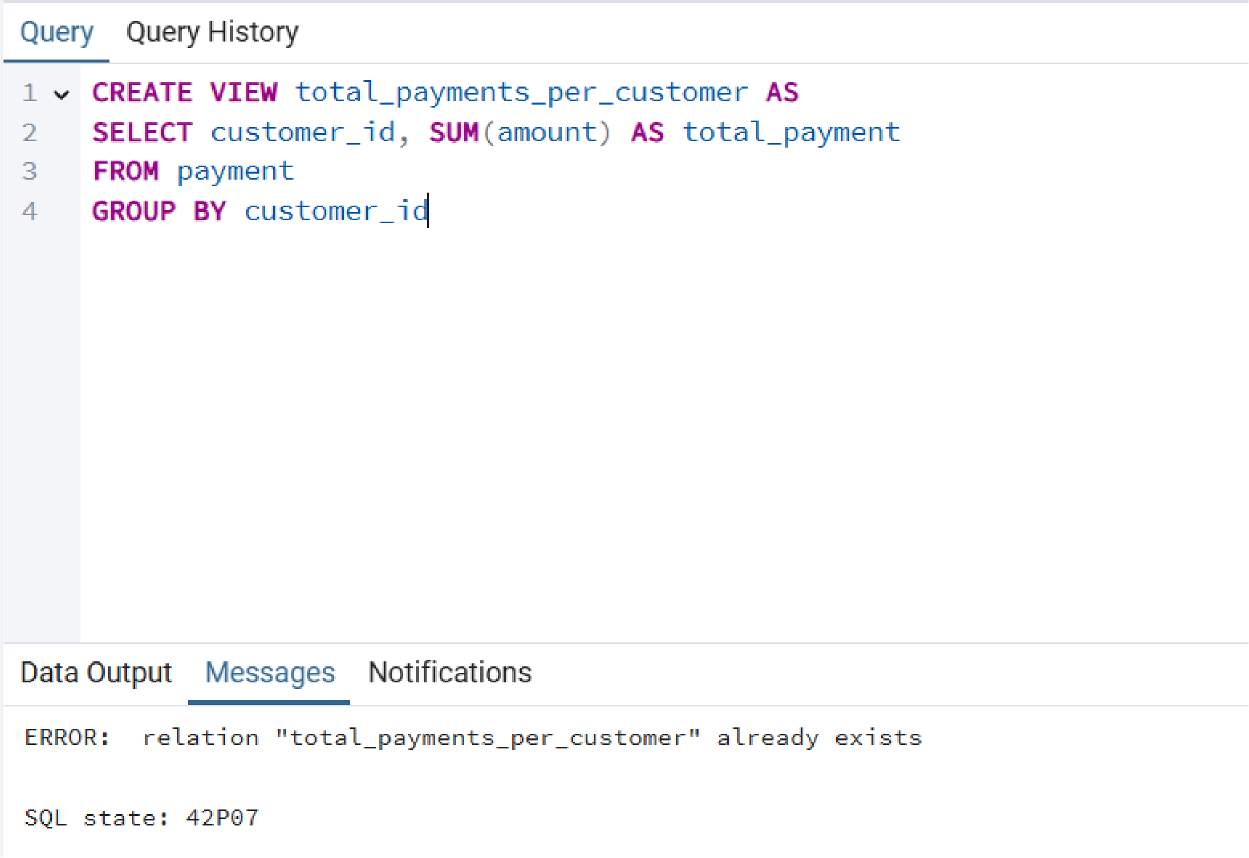
```sql
SELECT * FROM customer
WHERE address_id IN (
    SELECT address_id FROM address WHERE district = 'California'
)
```

	customer_id [PK] integer	store_id integer	first_name text	last_name text	email text	address_id integer	activebool boolean	create_date date	last_update timestamp with time zone	active integer
1	2	1	PATRICIA	JOHNSON	PATRICIA.JOHNSON@sakilacustomer.org	6	true	2022-02-14	2022-02-15 11:57:20+02	1
2	14	2	BETTY	WHITE	BETTY.WHITE@sakilacustomer.org	18	true	2022-02-14	2022-02-15 11:57:20+02	1
3	51	1	ALICE	STEWART	ALICE.STEWART@sakilacustomer.org	55	true	2022-02-14	2022-02-15 11:57:20+02	1
4	112	2	ROSA	REYNOLDS	ROSA.REYNOLDS@sakilacustomer.org	116	true	2022-02-14	2022-02-15 11:57:20+02	1
5	182	1	RENEE	LANE	RENEE.LANE@sakilacustomer.org	186	true	2022-02-14	2022-02-15 11:57:20+02	1
6	214	1	KRISTIN	JOHNSTON	KRISTIN.JOHNSTON@sakilacustomer.org	218	true	2022-02-14	2022-02-15 11:57:20+02	1
7	269	1	CASSANDRA	WALTERS	CASSANDRA.WALTERS@sakilacustomer.org	274	true	2022-02-14	2022-02-15 11:57:20+02	1
8	420	1	JACOB	LANCE	JACOB.LANCE@sakilacustomer.org	425	true	2022-02-14	2022-02-15 11:57:20+02	1
9	593	2	RENE	MCALISTER	RENE.MCALISTER@sakilacustomer.org	599	true	2022-02-14	2022-02-15 11:57:20+02	1

Total rows: 9 Query complete 00:00:00.065 CRLF Ln 4, Col 2

Views:

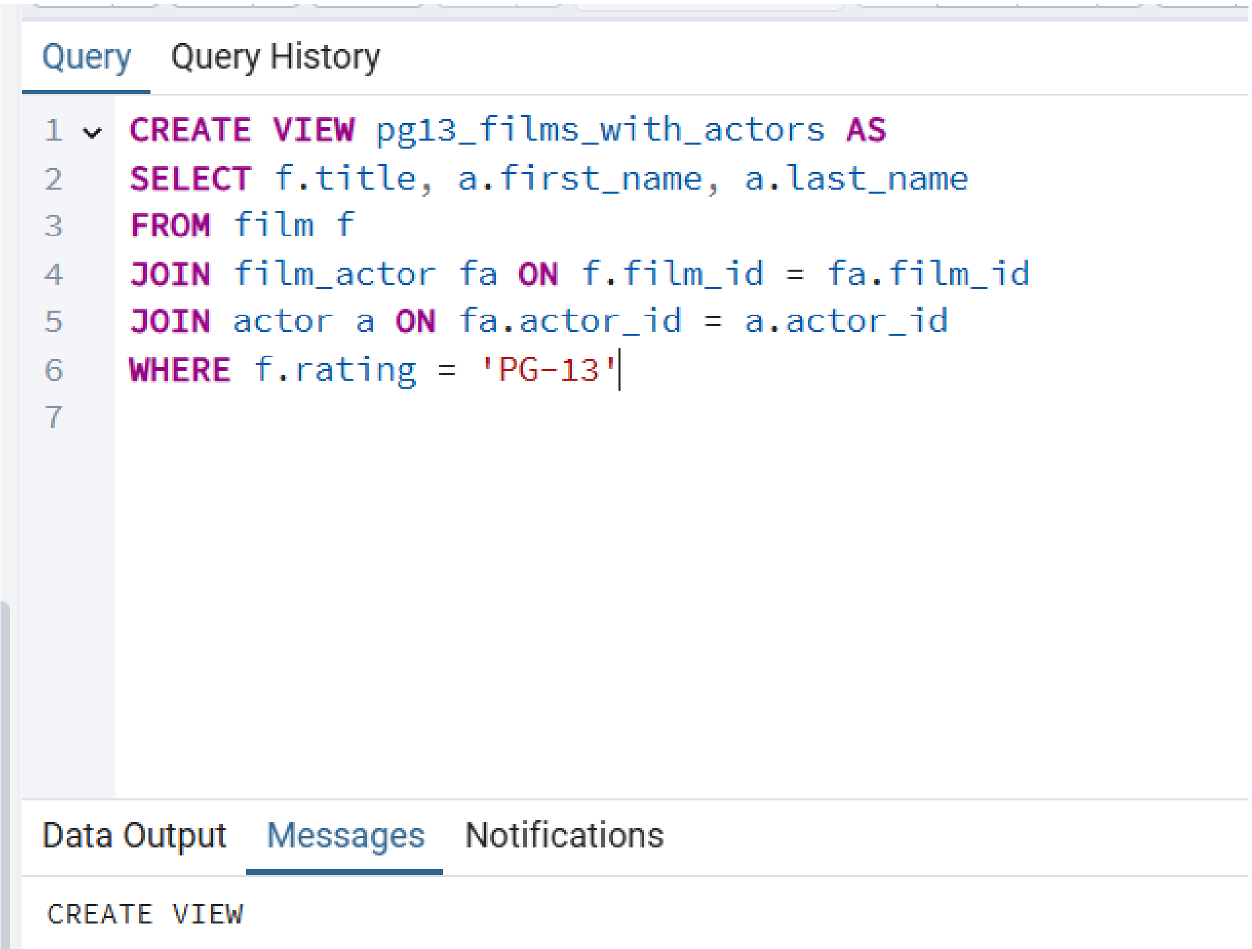
1. Create a view showing total payments per customer. I run the code twice



```
CREATE VIEW total_payments_per_customer AS
SELECT customer_id, SUM(amount) AS total_payment
FROM payment
GROUP BY customer_id
```

```
ERROR:  relation "total_payments_per_customer" already exists

SQL state: 42P07
```

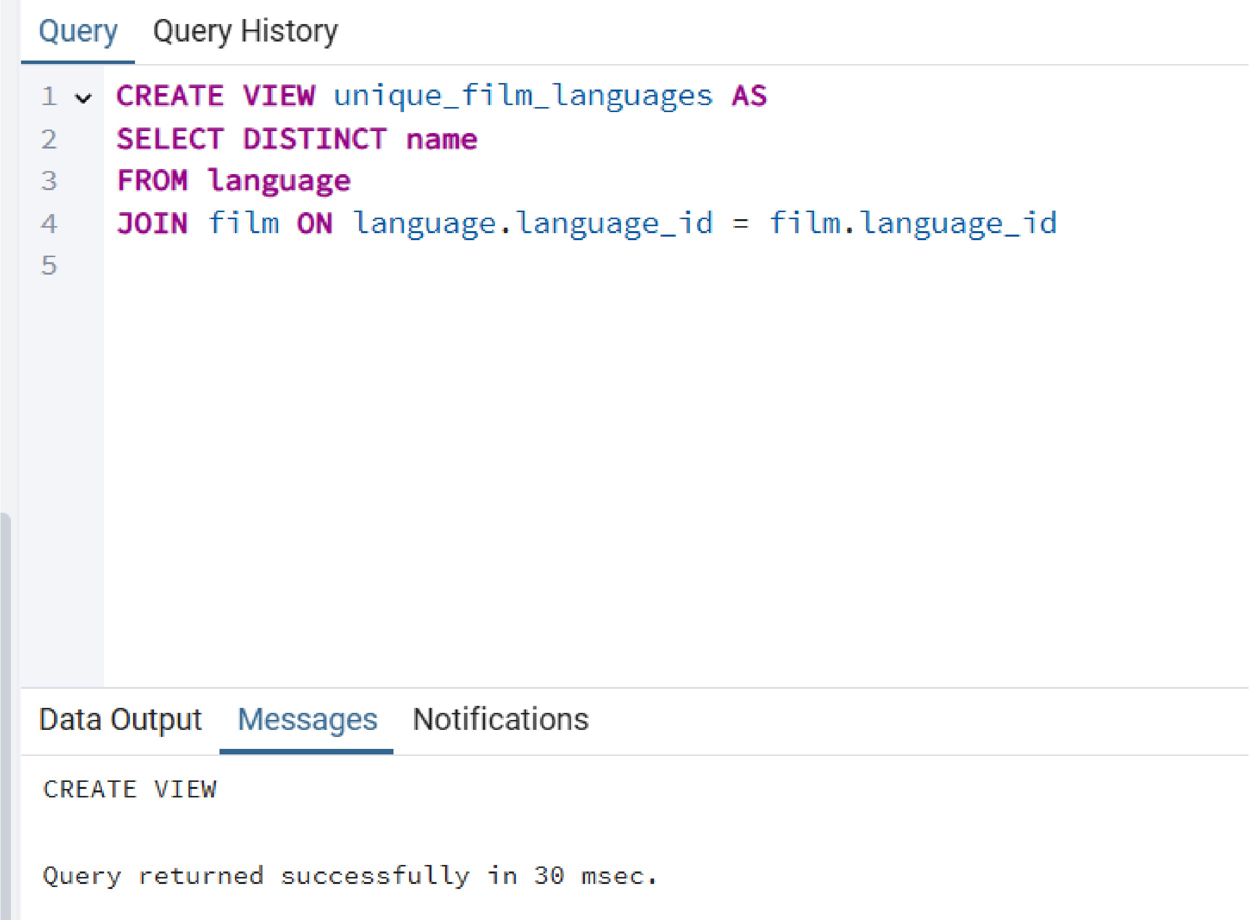
2. Create a view of all PG-13 rating films with their actors.



```sql
CREATE VIEW pg13_films_with_actors AS
SELECT f.title, a.first_name, a.last_name
FROM film f
JOIN film_actor fa ON f.film_id = fa.film_id
JOIN actor a ON fa.actor_id = a.actor_id
WHERE f.rating = 'PG-13'

```

```
CREATE VIEW
```

3. Create a view listing all unique film languages.

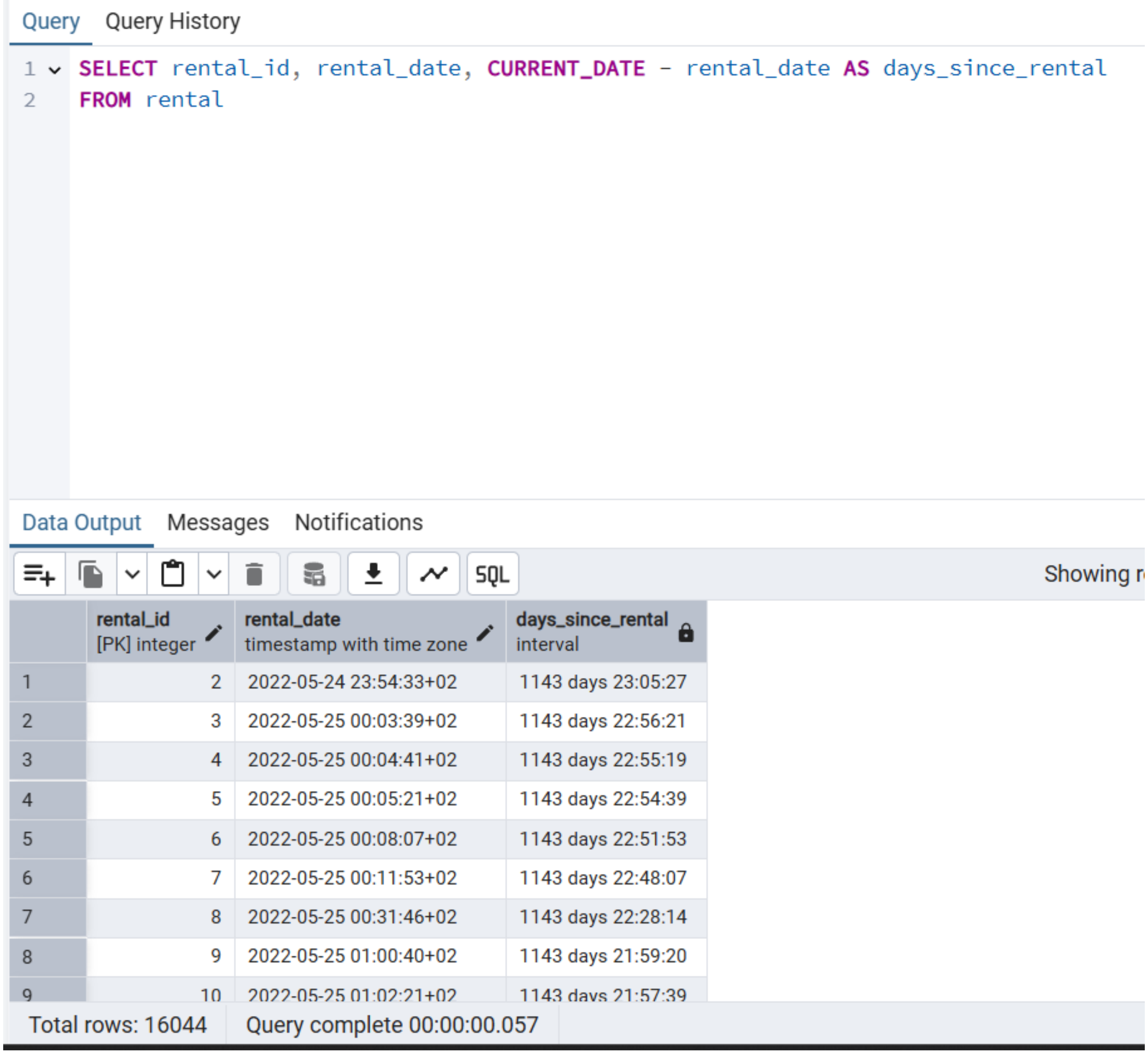


```sql
CREATE VIEW unique_film_languages AS
SELECT DISTINCT name
FROM language
JOIN film ON language.language_id = film.language_id

```

```
CREATE VIEW

Query returned successfully in 30 msec.
```

Built-in functions:

4. Show the rental date and how many days have passed since each rental.

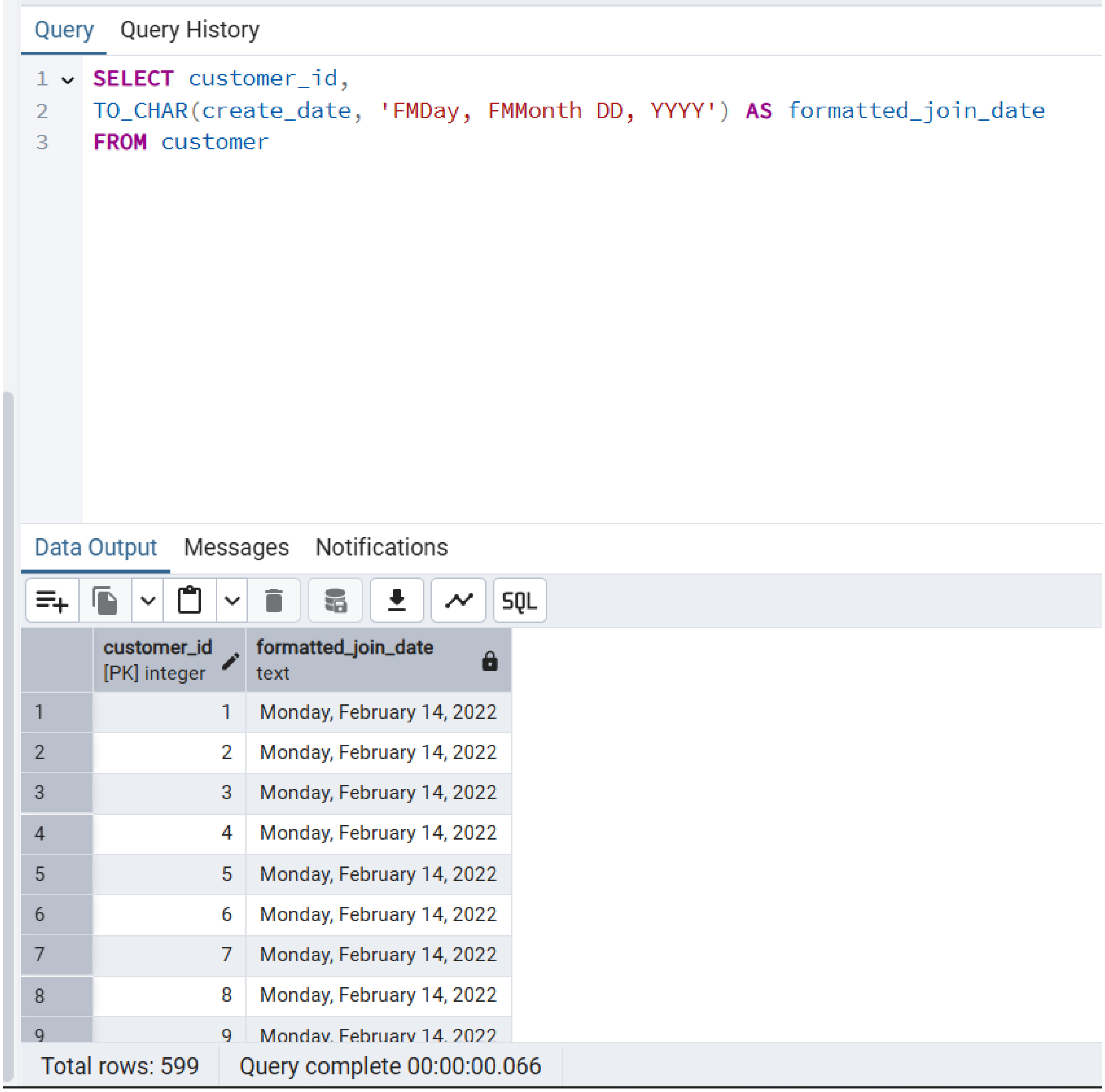


```sql
SELECT rental_id, rental_date, CURRENT_DATE - rental_date AS days_since_rental
FROM rental
```

	rental_id [PK] integer	rental_date timestamp with time zone	days_since_rental interval
1	2	2022-05-24 23:54:33+02	1143 days 23:05:27
2	3	2022-05-25 00:03:39+02	1143 days 22:56:21
3	4	2022-05-25 00:04:41+02	1143 days 22:55:19
4	5	2022-05-25 00:05:21+02	1143 days 22:54:39
5	6	2022-05-25 00:08:07+02	1143 days 22:51:53
6	7	2022-05-25 00:11:53+02	1143 days 22:48:07
7	8	2022-05-25 00:31:46+02	1143 days 22:28:14
8	9	2022-05-25 01:00:40+02	1143 days 21:59:20
9	10	2022-05-25 01:02:21+02	1143 days 21:57:39

Total rows: 16044 Query complete 00:00:00.057

5. Show customer join dates as "Weekday, Month Day, Year" (e.g., "Monday, January 01").

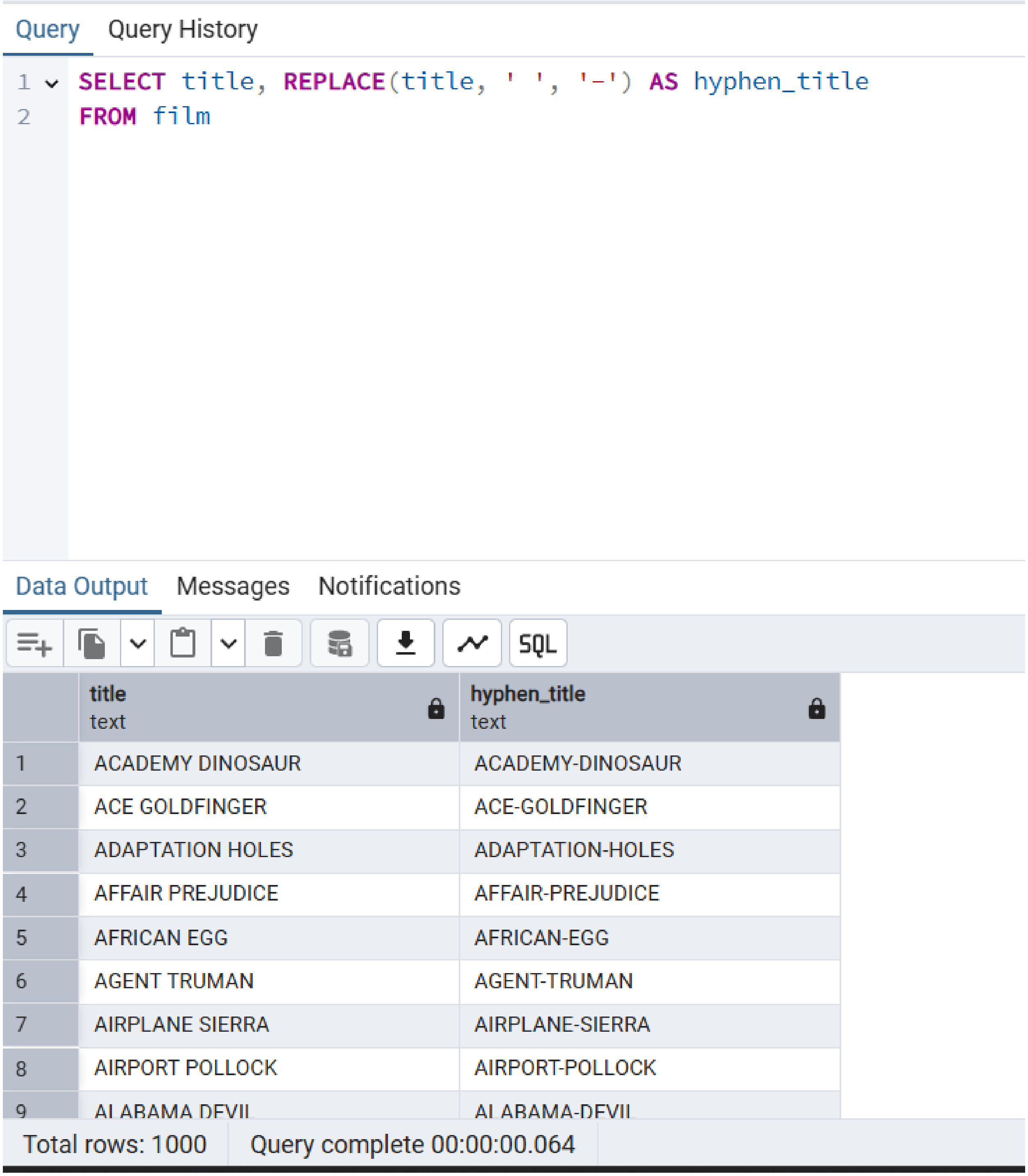


```sql
SELECT customer_id,
TO_CHAR(create_date, 'FMDay, FMMonth DD, YYYY') AS formatted_join_date
FROM customer
```

	customer_id [PK] integer	formatted_join_date text
1	1	Monday, February 14, 2022
2	2	Monday, February 14, 2022
3	3	Monday, February 14, 2022
4	4	Monday, February 14, 2022
5	5	Monday, February 14, 2022
6	6	Monday, February 14, 2022
7	7	Monday, February 14, 2022
8	8	Monday, February 14, 2022
9	9	Monday, February 14, 2022

Total rows: 599   Query complete 00:00:00.066

6. Display the film title and replace all spaces with hyphens (-).

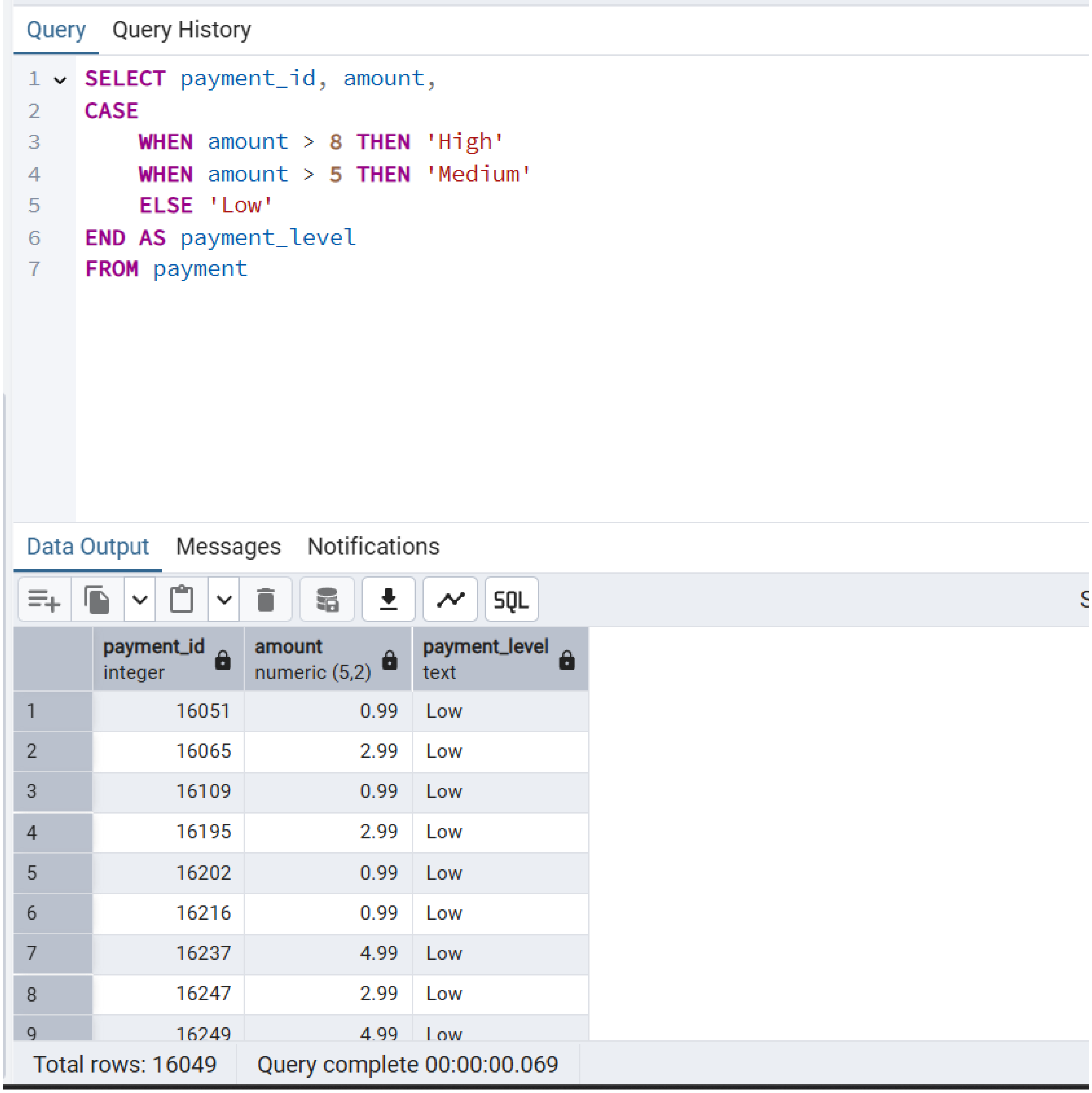


```sql
SELECT title, REPLACE(title, ' ', '-') AS hyphen_title
FROM film
```

	title text	hyphen_title text
1	ACADEMY DINOSAUR	ACADEMY-DINOSAUR
2	ACE GOLDFINGER	ACE-GOLDFINGER
3	ADAPTATION HOLES	ADAPTATION-HOLES
4	AFFAIR PREJUDICE	AFFAIR-PREJUDICE
5	AFRICAN EGG	AFRICAN-EGG
6	AGENT TRUMAN	AGENT-TRUMAN
7	AIRPLANE SIERRA	AIRPLANE-SIERRA
8	AIRPORT POLLOCK	AIRPORT-POLLOCK
9	ALABAMA DEVIL	ALABAMA-DEVIL

Total rows: 1000   Query complete 00:00:00.064

7. For each payment, show a label: 'High' if amount > 8, 'Medium' if > 5, else 'Low'.



```
SELECT payment_id, amount,
CASE
    WHEN amount > 8 THEN 'High'
    WHEN amount > 5 THEN 'Medium'
    ELSE 'Low'
END AS payment_level
FROM payment
```

	payment_id integer	amount numeric (5,2)	payment_level text
1	16051	0.99	Low
2	16065	2.99	Low
3	16109	0.99	Low
4	16195	2.99	Low
5	16202	0.99	Low
6	16216	0.99	Low
7	16237	4.99	Low
8	16247	2.99	Low
9	16249	4.99	Low

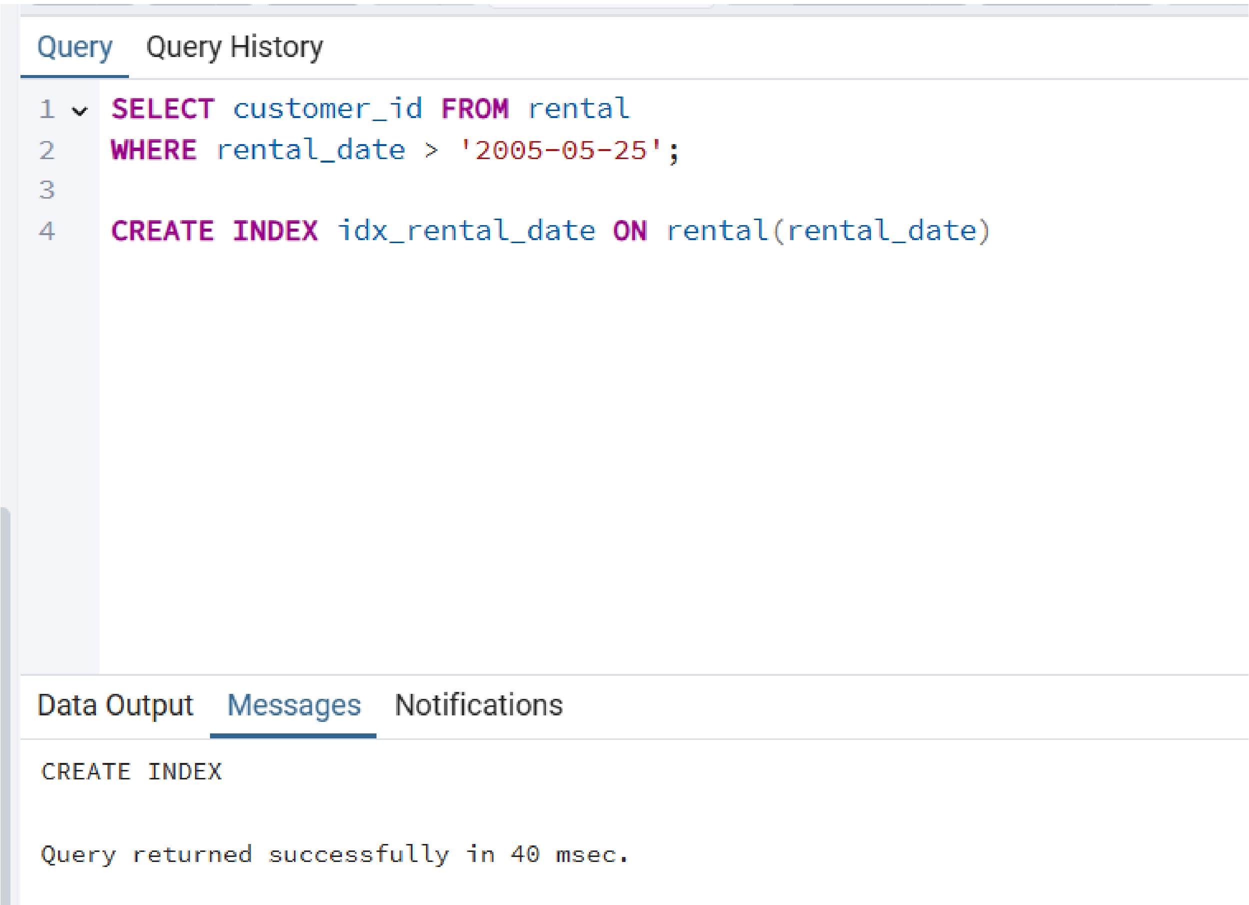
Total rows: 16049 Query complete 00:00:00.069

Index:

1. Identify which columns need indexes to optimize this query and create them.

SELECT customer_id FROM rental

WHERE rental_date > '2005-05-25';



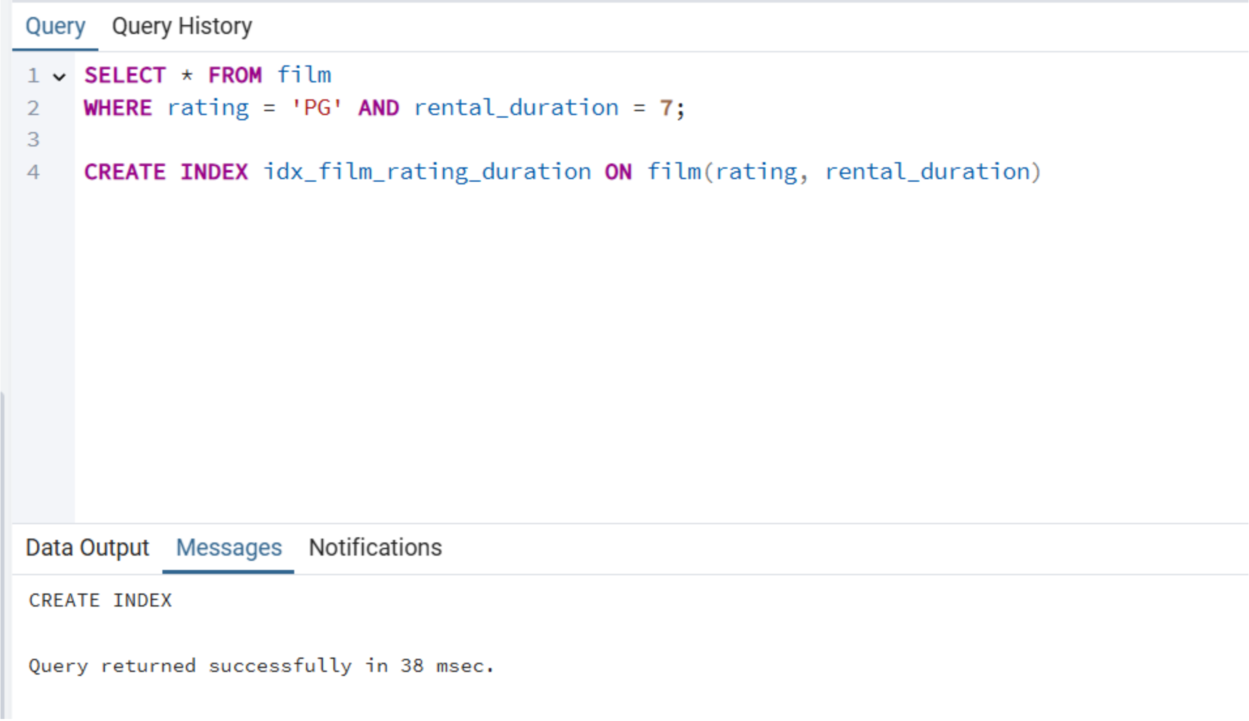
```
SELECT customer_id FROM rental
WHERE rental_date > '2005-05-25';

CREATE INDEX idx_rental_date ON rental(rental_date)
```

```
CREATE INDEX

Query returned successfully in 40 msec.
```

2. Create a composite index to speed up this query.

SELECT * FROM film

WHERE rating = 'PG' AND rental_duration= 7;



```sql
SELECT * FROM film
WHERE rating = 'PG' AND rental_duration = 7;

CREATE INDEX idx_film_rating_duration ON film(rating, rental_duration)
```

```
CREATE INDEX

Query returned successfully in 38 msec.
```