JOINS:

SQL

- Welcome to this section on JOINS.
- JOINS will allow us to combine information from multiple tables!
- Let's see what we will learn in this section

AS CONDITION:

- Example

Query Editor | Query History

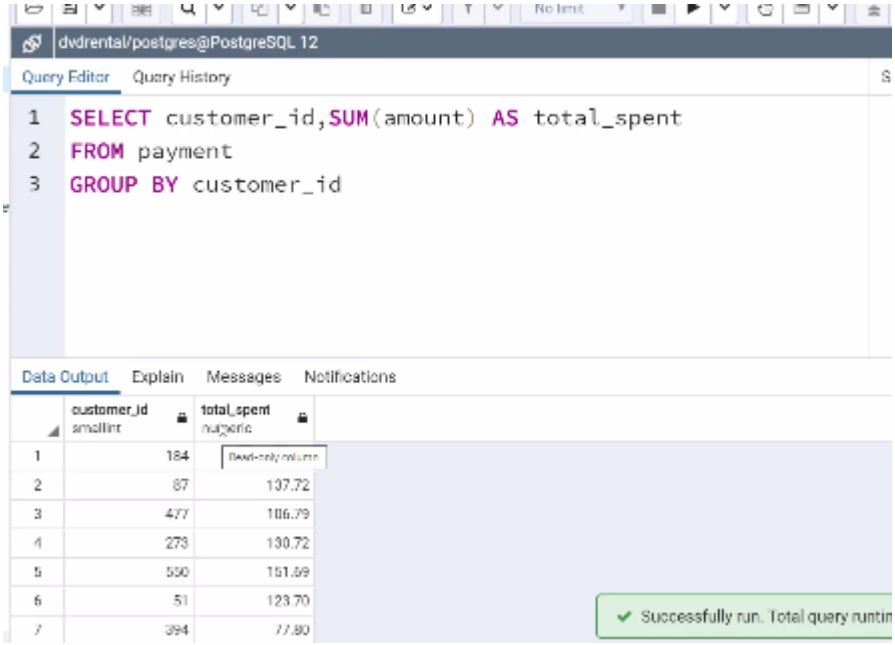
```
1  SELECT amount AS rental_price
2  FROM payment;
```

Data Output | Explain | Messages | Notifications

	rental_price numeric (5,2)
1	7.99
2	1.99

SQL

- The AS operator gets executed at the very end of a query, meaning that we can not use the ALIAS inside a WHERE operator.

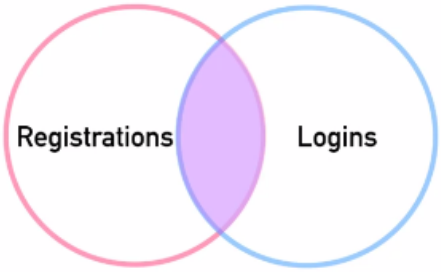


INNER JOIN

SQL

- SELECT * FROM Registrations
  INNER JOIN Logins
  ON Registrations.name = Logins.name

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David



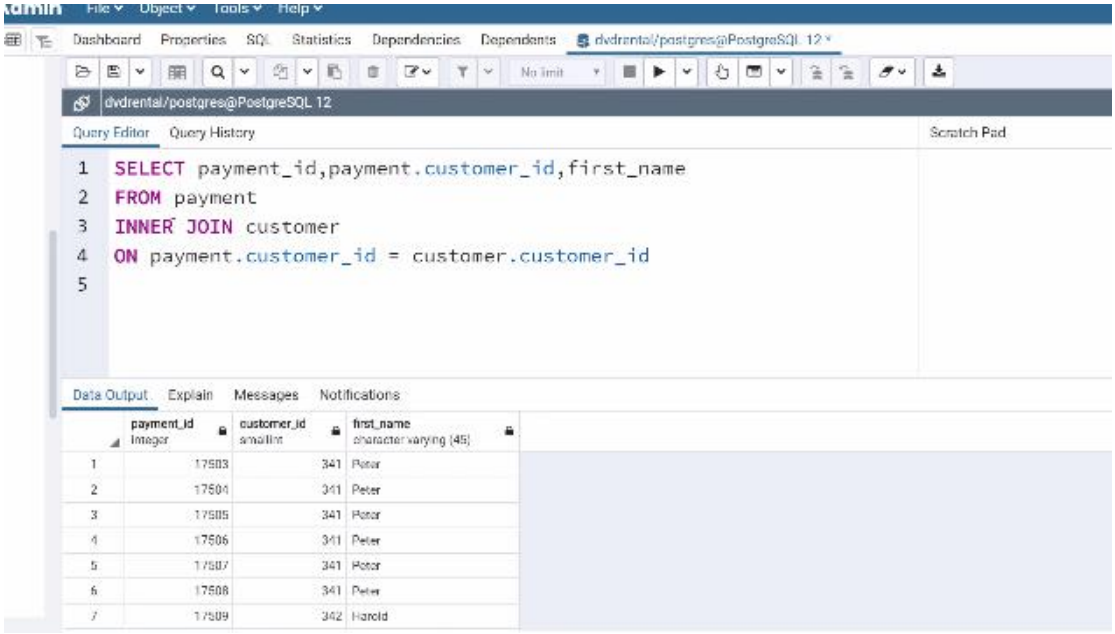
LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

PIERIAN DATA

SQL

```
SELECT reg_id,Logins.name,log_id
FROM Registrations
INNER JOIN Logins
ON Registrations.name = Logins.name
```

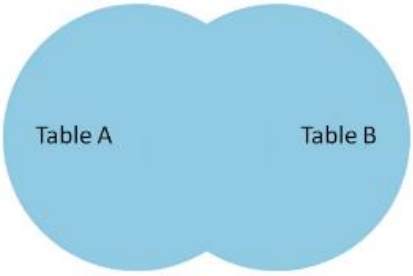
RESULTS		
reg_id	name	log_id
1	Andrew	2
2	Bob	4



FULL OUTER JOIN:

- There are few different types of OUTER JOINs
- They will allow us to specify how to deal with values only present in one of the tables being joined.
- These are the more complex JOINs, take your time when trying to understand them!

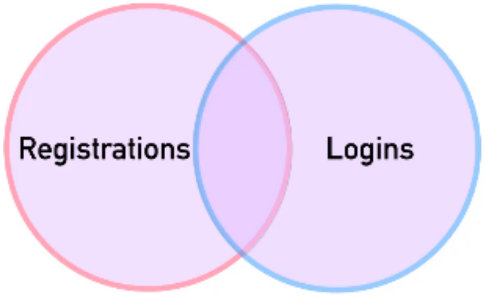
- SELECT * FROM **TableB**
  FULL OUTER JOIN **TableA**
  ON **TableA**.col_match = **TableB**.col_match



PIERIAN DATA

- SELECT * FROM Registrations
  FULL OUTER JOIN Logins
  ON Registrations.name = Logins.name

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David



LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

PIERIAN DATA

SQL

- SELECT * FROM Registrations FULL OUTER JOIN Logins ON Registrations.name = Logins.name

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

RESULTS			
reg_id	name	log_id	name
1	Andrew	2	Andrew
2	Bob	4	Bob
3	Charlie	null	null
4	David	null	null
null	null	1	Xavier
null	null	3	Yolanda

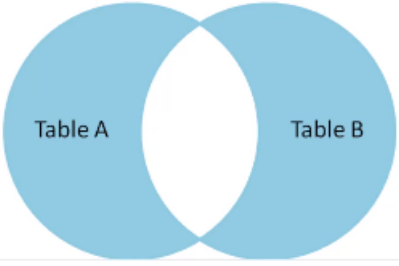
LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

FULL OUTER JOIN with WHERE

Get rows unique to either table
(rows not found in both tables)

SQL

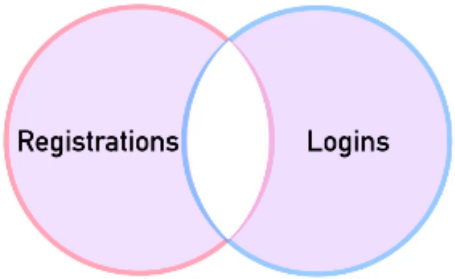
- SELECT * FROM **TableA**
  FULL OUTER JOIN **TableB**
  ON **TableA**.col_match = **TableB**.col_match
  WHERE **TableA**.id IS null OR
  TableB.id IS null



IERIAN DATA

SQL

```
SELECT * FROM Registrations FULL OUTER
JOIN Logins
ON Registrations.name = Logins.name
WHERE Registrations.reg_id IS null OR
Logins.log_id IS null
```



written a note here.

PIERIAN DATA

SQL

- SELECT * FROM Registrations FULL OUTER JOIN Logins
  ON Registrations.name = Logins.name

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

RESULTS			
reg_id	name	log_id	name
1	Andrew	2	Andrew
2	Bob	4	Bob
3	Charlie	null	null
4	David	null	null
null	null	1	Xavier
null	null	3	Yolanda

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

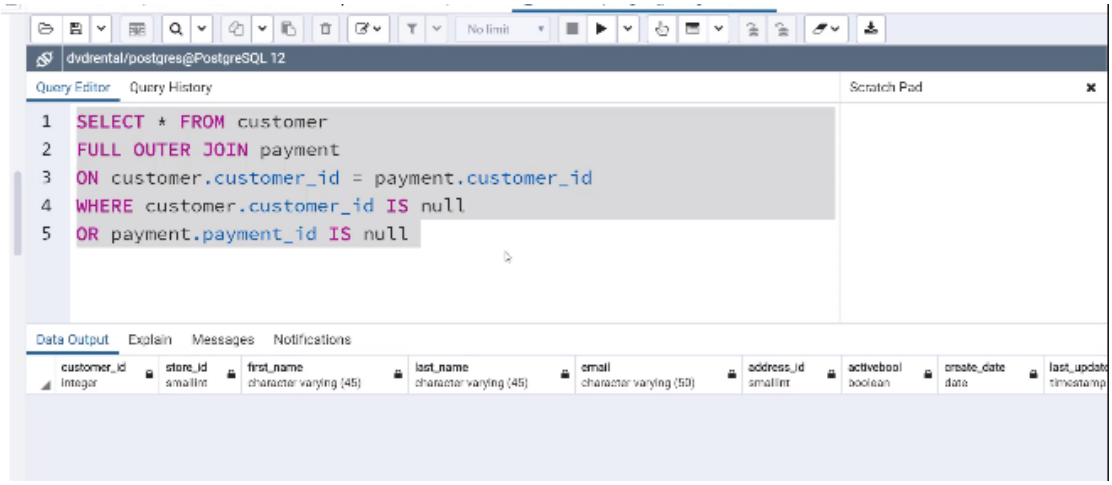
SQL

- SELECT * FROM Registrations FULL OUTER JOIN Logins
  ON Registrations.name = Logins.name
  WHERE Registrations.reg_id IS null OR
  Logins.log_id IS null

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

RESULTS			
reg_id	name	log_id	name
3	Charlie	null	null
4	David	null	null
null	null	1	Xavier
null	null	3	Yolanda

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob



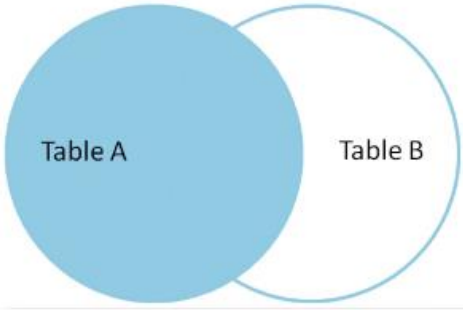
LEFT JOIN:

SQL

- A LEFT OUTER JOIN results in the set of records that are in the left table, if there is no match with the right table, the results are null.
- Later on we will learn how to add WHERE statements to further modify a LEFT OUTER JOIN

SQL

- SELECT * FROM **TableA**
  LEFT OUTER JOIN **TableB**
  ON **TableA**.col_match = **TableB**.col_match

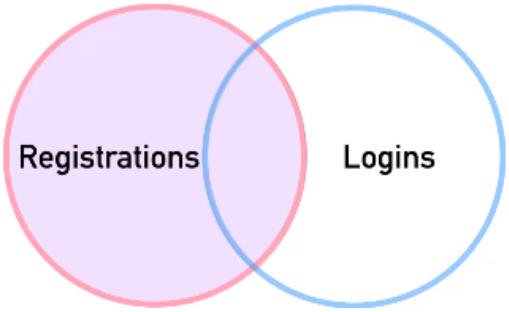


PIERIAN DATA

SQL

- SELECT * FROM Registrations
  LEFT OUTER JOIN Logins
  ON Registrations.name = Logins.name

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David



LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

SQL

- SELECT * FROM Registrations
  LEFT OUTER JOIN Logins
  ON Registrations.name = Logins.name

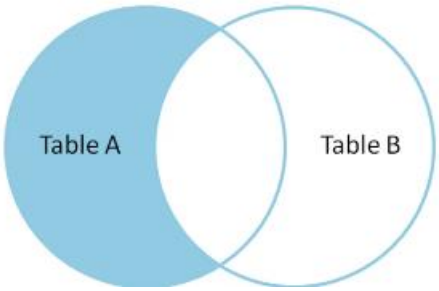
REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

RESULTS			
reg_id	name	log_id	name
1	Andrew	2	Andrew
2	Bob	4	Bob
3	Charlie	null	null
4	David	null	null

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

SQL

- What if we only wanted entries unique to Table A? Those rows found in Table A and **not** found in Table B.



PIERIAN DATA

SQL

```
SELECT * FROM Registrations
LEFT OUTER JOIN Logins
ON Registrations.name = Logins.name
WHERE Logins.log_id IS null
```

REGISTRATIONS	
reg_id	name
1	Andrew
2	Bob
3	Charlie
4	David

RESULTS			
reg_id	name	log_id	name
3	Charlie	null	null
4	David	null	null

LOGINS	
log_id	name
1	Xavier
2	Andrew
3	Yolanda
4	Bob

PIERIAN DATA

Dashboard Properties SQL Statistics Dependencies Dependents dvdrental/postgres@PostgreSQL 12 *

No limit

dvdrental/postgres@PostgreSQL 12

Query Editor Query History

```
1  SELECT film.film_id,title,inventory_id,store_id
2  FROM film
3  LEFT JOIN inventory ON
4  inventory.film_id = film.film_id
```

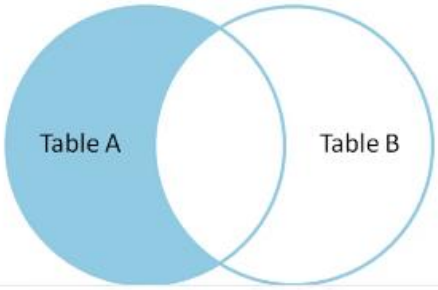
Data Output Explain Messages Notifications

	film_id integer	title character varying (255)	inventory_id integer	store_id smallint
1	1	Academy Dinosaur	1	1
2	1	Academy Dinosaur	2	1
3	1	Academy Dinosaur	3	1
4	1	Academy Dinosaur	4	
5	1	Academy Dinosaur	5	

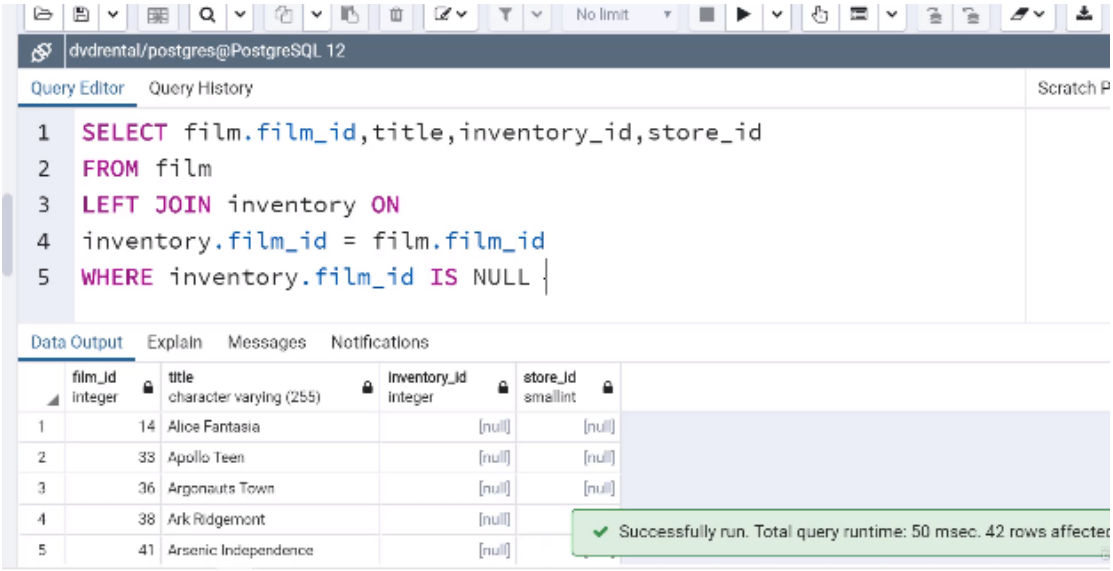
Successfully run. Total query runtime: 68 msec. 4623

SQL

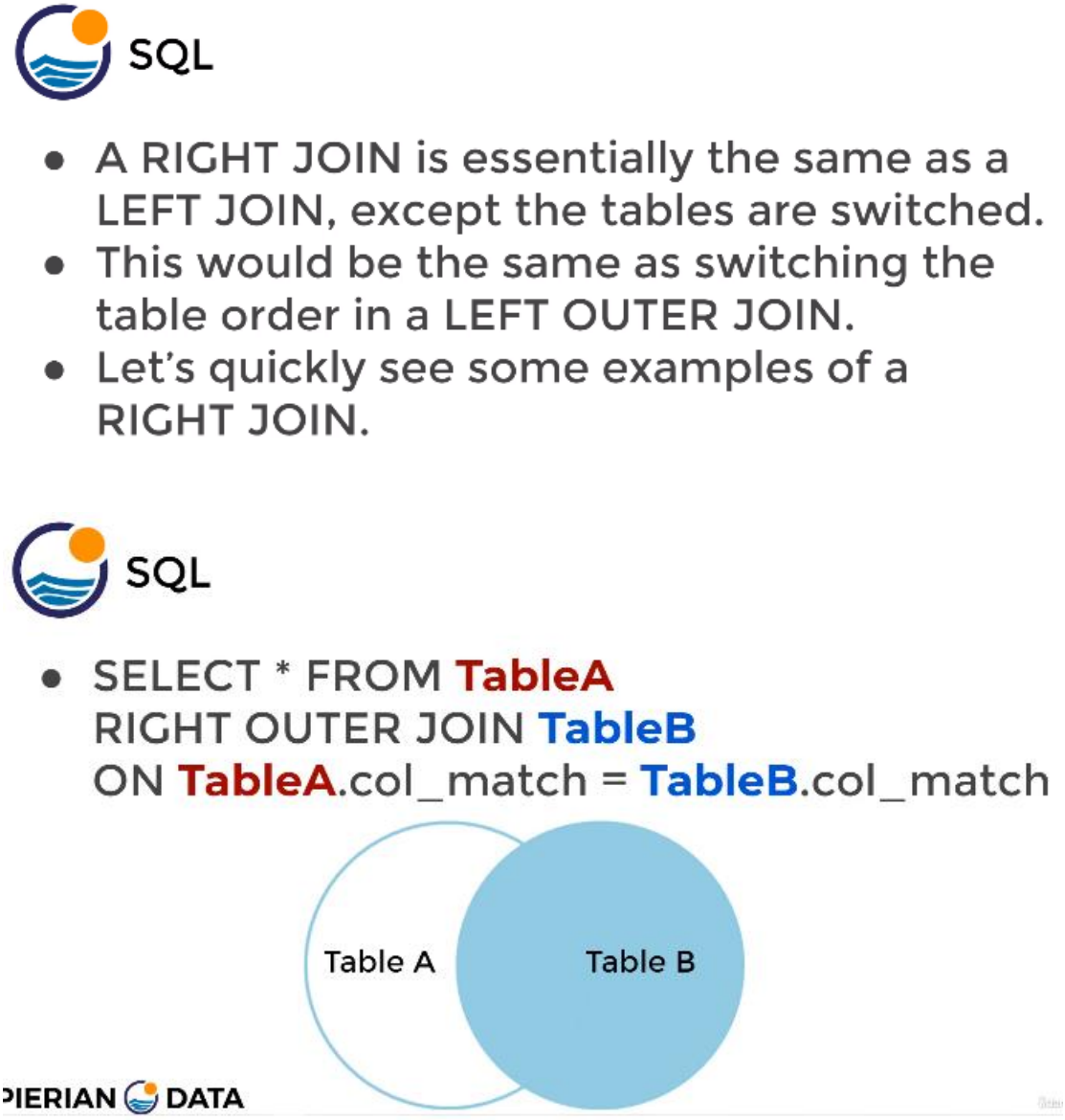
- What if we only wanted entries unique to Table A? Those rows found in Table A and **not** found in Table B.



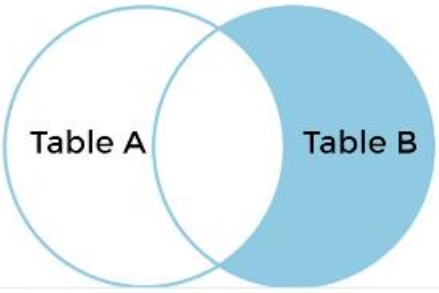
PIERIAN DATA


No limit
dvdrental/postgres@PostgreSQL 12
Query Editor
Query History
Scratch P
1 SELECT film.film_id,title,inventory_id,store_id
2 FROM film
3 LEFT JOIN inventory ON
4 inventory.film_id = film.film_id
5 WHERE inventory.film_id IS NULL
Data Output
Explain
Messages
Notifications
film_id integer
title character varying (255)
inventory_id integer
store_id smallint
1 14 Alice Fantasia [null] [null]
2 33 Apollo Teen [null] [null]
3 36 Argonauts Town [null] [null]
4 38 Ark Ridgemont [null]
5 41 Arsenic Independence [null]
Successfully run. Total query runtime: 50 msec. 42 rows affecte

RIGHT JOIN:


SQL
A RIGHT JOIN is essentially the same as a LEFT JOIN, except the tables are switched.
This would be the same as switching the table order in a LEFT OUTER JOIN.
Let's quickly see some examples of a RIGHT JOIN.
SQL
SELECT * FROM TableA
RIGHT OUTER JOIN TableB
ON TableA.col_match = TableB.col_match
Table A
Table B
PIERIAN DATA

- SELECT * FROM **TableA**
  RIGHT OUTER JOIN **TableB**
  ON **TableA**.col_match = **TableB**.col_match
  WHERE **TableA**.id IS null



- It is up to you and how you have the tables organized “in your mind” when it comes to choosing a LEFT vs RIGHT join, since depending on the table order you specify in the JOIN, you can perform duplicate JOINs with either method.

UNION:

- The UNION operator is used to combine the result-set of two or more SELECT statements.
- It basically serves to directly concatenate two results together, essentially “pasting” them together.

- SELECT * FROM Sales2021_Q1
  UNION
  SELECT * FROM Sales2021_Q2;

name	amount
David	100
Claire	50
David	200
Claire	100

Q & A:

- California sales tax laws have changed and we need to alert our customers to this through email.
- What are the emails of the customers who live in California?

SQL

- Expected Results

	district character varying (20)	email character varying (50)
1	California	patricia.johnson@sakilacust...
2	California	betty.white@sakilacustomer...
3	California	alice.stewart@sakilacustom...
4	California	rosa.reynolds@sakilacusto...
5	California	renee.lane@sakilacustomer....
6	California	kristin.johnston@sakilacust...
7	California	cassandra.walters@sakilacu...
8	California	jacob.lance@sakilacustome...
9	California	rene.mcalister@sakilacusto...

SQL

```
SELECT district,email FROM address
INNER JOIN customer ON
address.address_id = customer.address_id
WHERE district = 'California'
```

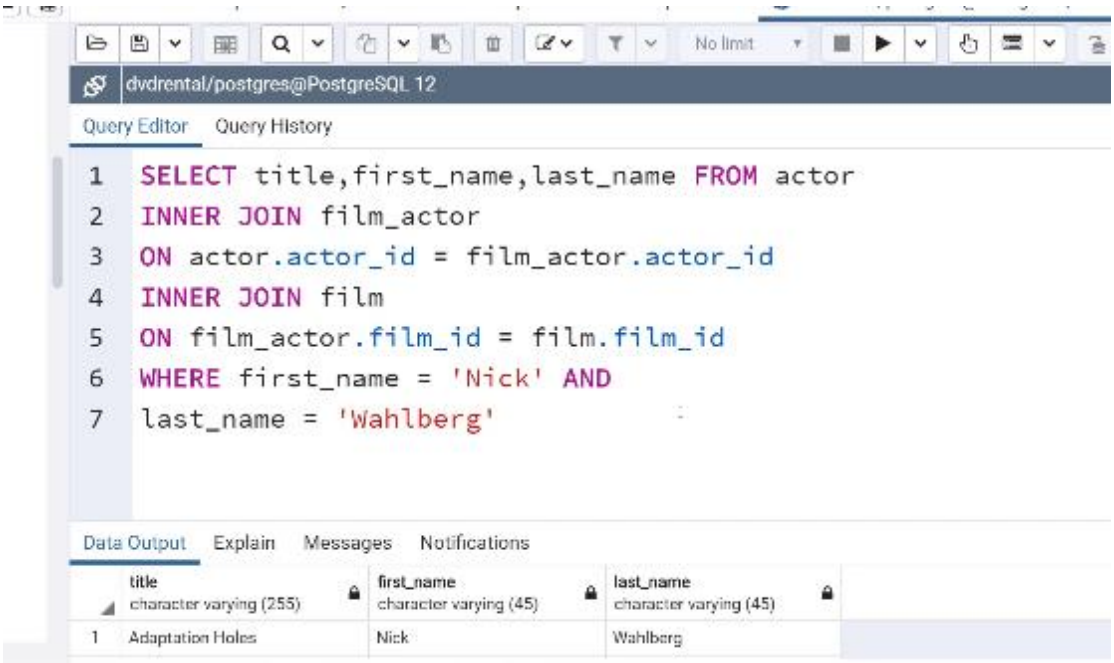
SQL

- A customer walks in and is a huge fan of the actor “Nick Wahlberg” and wants to know which movies he is in.
- Get a list of all the movies “Nick Wahlberg” has been in.

- Expected Results

	title character varying (255)	first_name character varying (45)	last_name character varying (45)
1	Adaptation Holes	Nick	Wahlberg
2	Apache Divine	Nick	Wahlberg
3	Baby Hall	Nick	Wahlberg
4	Bull Shawshank	Nick	Wahlberg
5	Chainsaw Uptown	Nick	Wahlberg
.....			
21	Mask Peach	Nick	Wahlberg
22	Roof Champion	Nick	Wahlberg
23	Rushmore Mermaid	Nick	Wahlberg
24	Smile Earring	Nick	Wahlberg
25	Wardrobe Phantom	Nick	Wahlberg


```
SELECT title,first_name,last_name
FROM film_actor INNER JOIN actor
ON film_actor.actor_id = actor.actor_id
INNER JOIN film
ON film_actor.film_id = film.film_id
WHERE first_name = 'Nick'
AND last_name = 'Wahlberg'
```


No limit
dvdrental/postgres@PostgreSQL 12
Query Editor
Query History
1 SELECT title,first_name,last_name FROM actor
2 INNER JOIN film_actor
3 ON actor.actor_id = film_actor.actor_id
4 INNER JOIN film
5 ON film_actor.film_id = film.film_id
6 WHERE first_name = 'Nick' AND
7 last_name = 'Wahlberg'
Data Output
Explain
Messages
Notifications
title
character varying (255)
first_name
character varying (45)
last_name
character varying (45)
1
Adaptation Holes
Nick
Wahlberg