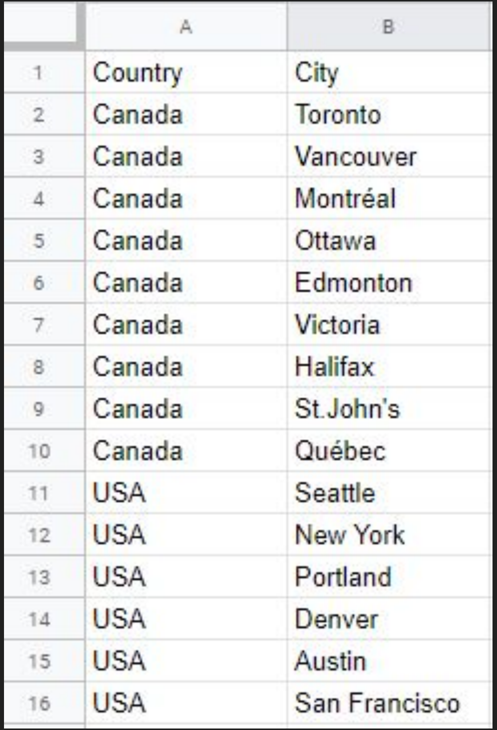
SASSA SQL Workshop

Fall 2019

Why DBMS



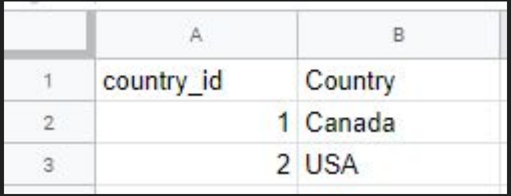
	A	B
1	Country	City
2	Canada	Toronto
3	Canada	Vancouver
4	Canada	Montréal
5	Canada	Ottawa
6	Canada	Edmonton
7	Canada	Victoria
8	Canada	Halifax
9	Canada	St.John's
10	Canada	Québec
11	USA	Seattle
12	USA	New York
13	USA	Portland
14	USA	Denver
15	USA	Austin
16	USA	San Francisco

Example

```
Search for
    County == “USA” && City ==
“Austin”
```

Average case: O(n)

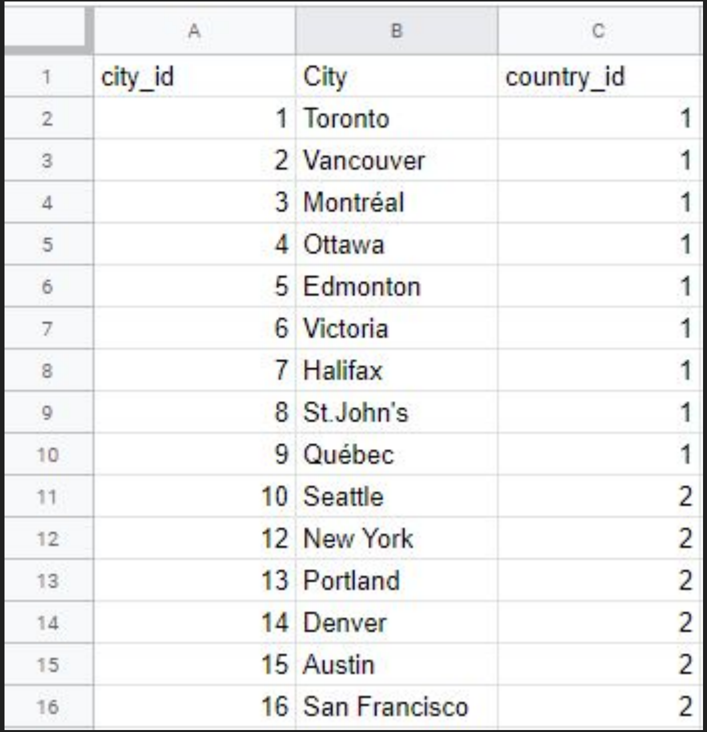
INDEXING



	A	B
1	country_id	Country
2	1	Canada
3	2	USA

Search for
 County == "USA" && City == "Austin"

Average case: O(log(n))



	A	B	C
1	city_id	City	country_id
2	1	Toronto	1
3	2	Vancouver	1
4	3	Montréal	1
5	4	Ottawa	1
6	5	Edmonton	1
7	6	Victoria	1
8	7	Halifax	1
9	8	St.John's	1
10	9	Québec	1
11	10	Seattle	2
12	12	New York	2
13	13	Portland	2
14	14	Denver	2
15	15	Austin	2
16	16	San Francisco	2

SCHEDULING

Example

Two people buying MOVIE TICKETS

What if the data was stored in a CSV…

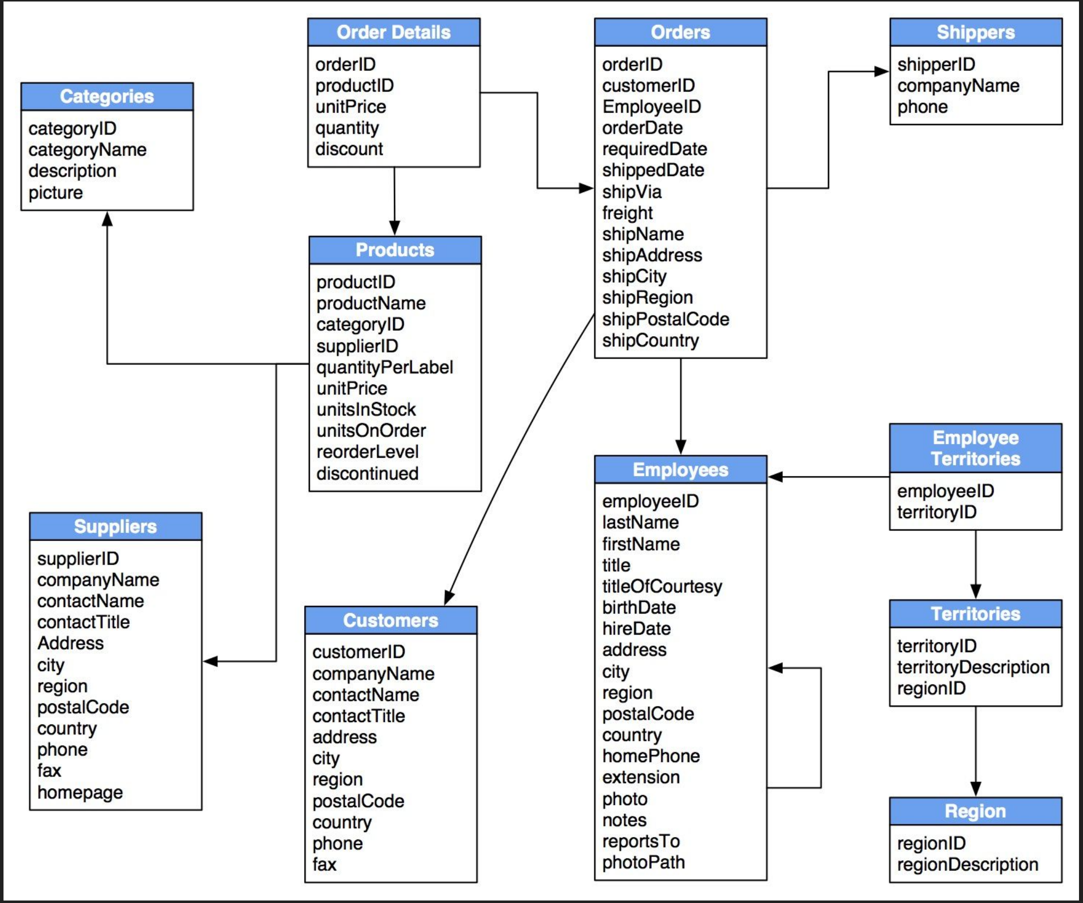
Can we WRITE to this document simultaneously?

Whose data is added first?

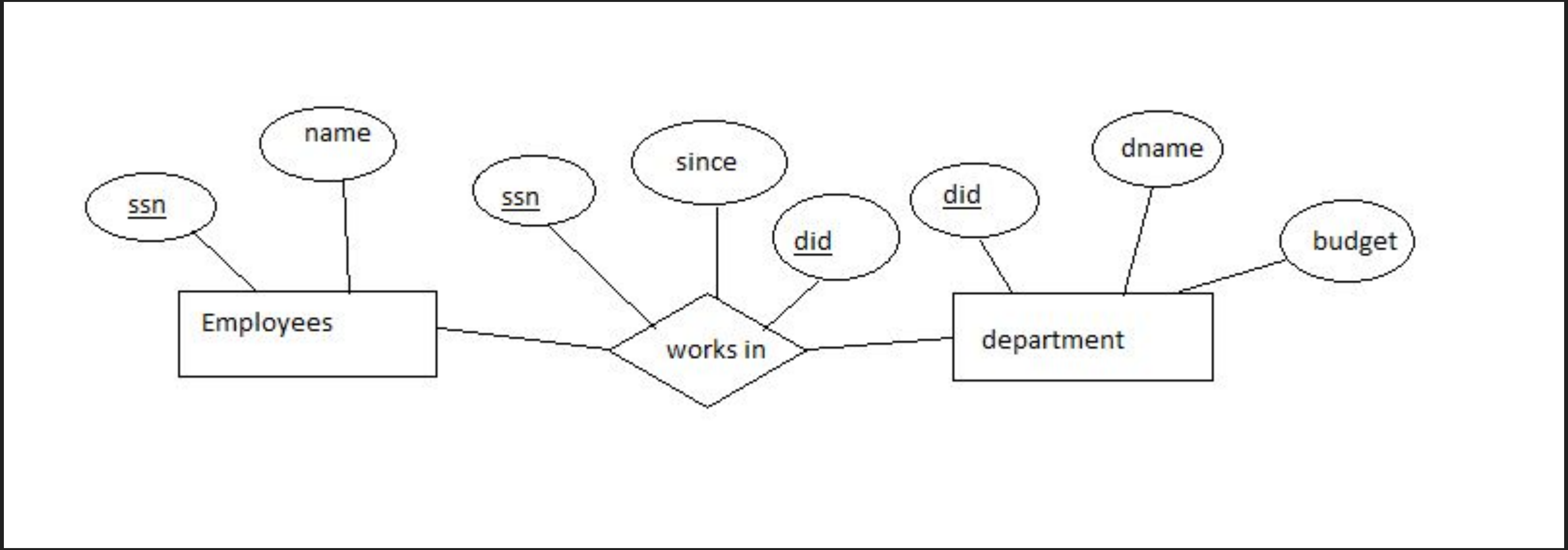
RELATIONS

ER diagrams

dbdiagram.io



ENTITY RELATIONSHIP DIAGRAM



OUR ER DIAGRAM

We will have 7 ENTITIES :

FILM - stores films data such as title, release year, length, rating, etc.

CATEGORY - stores film's categories data.

FILM_CATEGORY- stores the relationships between films and categories.

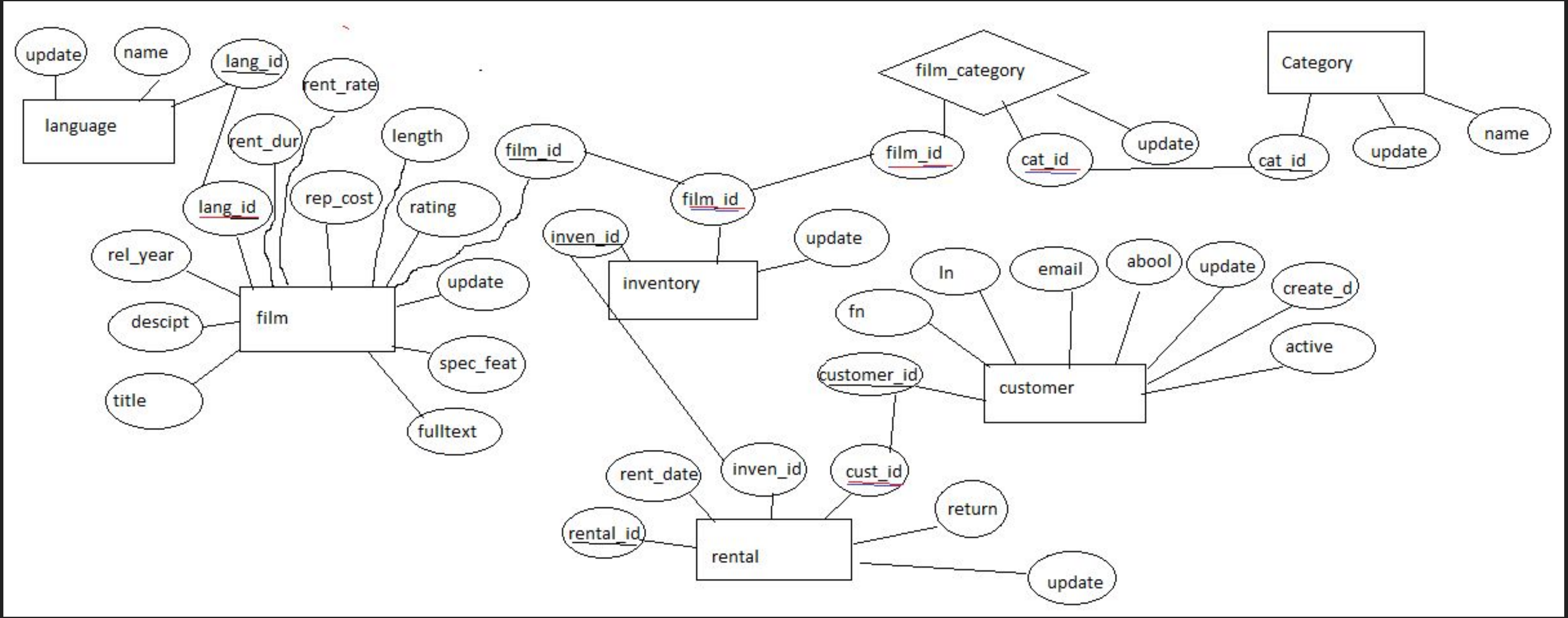
INVENTORY - stores inventory data.

RENTAL - stores rental data.

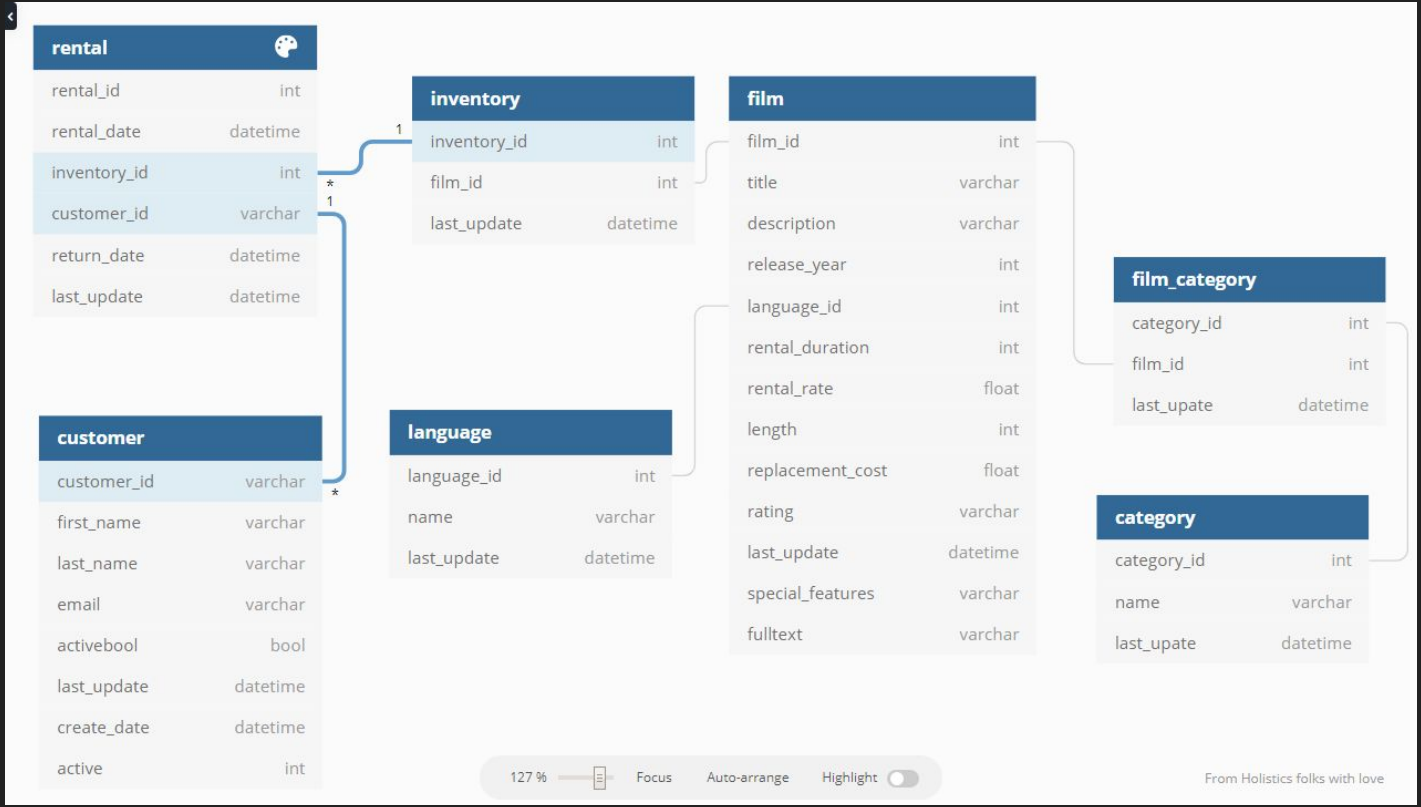
CUSTOMER - stores customers data.

LANGUAGE - stores the film language

OUR ER DIAGRAM



OUR RELATIONSHIP MODEL



rental	
rental_id	int
rental_date	datetime
inventory_id	int
customer_id	varchar
return_date	datetime
last_update	datetime

inventory	
inventory_id	int
film_id	int
last_update	datetime

film	
film_id	int
title	varchar
description	varchar
release_year	int
language_id	int
rental_duration	int
rental_rate	float
length	int
replacement_cost	float
rating	varchar
last_update	datetime
special_features	varchar
fulltext	varchar

film_category	
category_id	int
film_id	int
last_upate	datetime

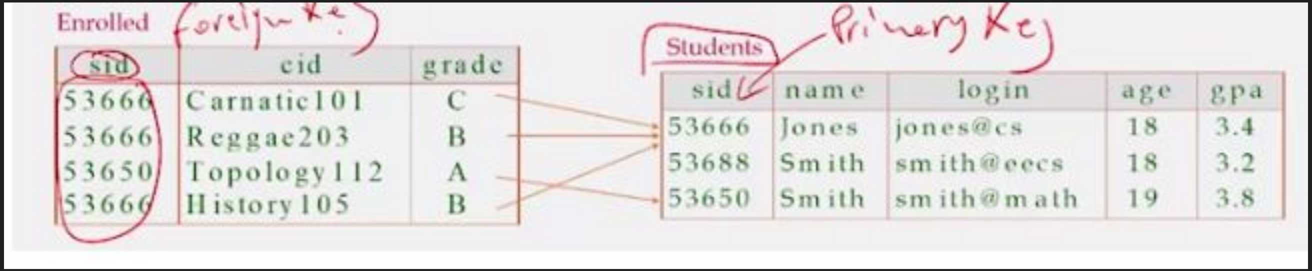
customer	
customer_id	varchar
first_name	varchar
last_name	varchar
email	varchar
activebool	bool
last_update	datetime
create_date	datetime
active	int

language	
language_id	int
name	varchar
last_update	datetime

category	
category_id	int
name	varchar
last_upate	datetime

KEYS

EXAMPLE



Enrolled

sid	cid	grade
53666	Carnatic101	C
53666	Reggae203	B
53650	Topology112	A
53666	History105	B

Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@eecs	18	3.2
53650	Smith	smith@math	19	3.8

Self study KEY CONSTRAINTS

CREATE TABLE

```sql
CREATE TABLE rental (
    rental_id SERIAL,
    rental_date timestamp with time zone NOT NULL,
    inventory_id integer NOT NULL,
    customer_id integer NOT NULL,
    return_date timestamp with time zone,
    last_update timestamp with time zone DEFAULT now() NOT NULL,
    PRIMARY KEY(rental_id),
    FOREIGN KEY(customer_id) REFERENCES customer
);
```

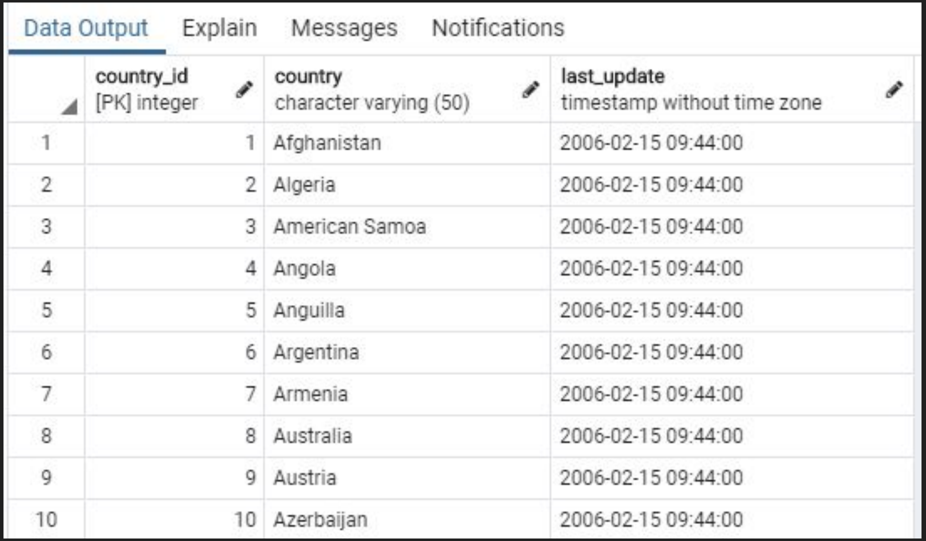
INSERT

```sql
INSERT INTO inventory VALUES (612, 133, '2017-02-15 10:09:17');
```

SELECT Statements

SELECT statements are used to select data from a database

```
SELECT
  -- All columns
  *
FROM
  -- The Schema 'public'
  -- The Table 'country'
  public.country
```



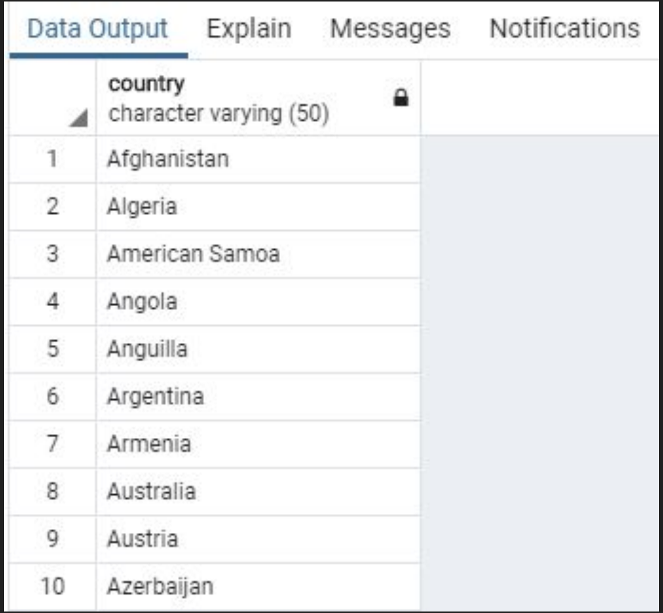
Data Output | Explain | Messages | Notifications

	country_id [PK] integer	country character varying (50)	last_update timestamp without time zone
1	1	Afghanistan	2006-02-15 09:44:00
2	2	Algeria	2006-02-15 09:44:00
3	3	American Samoa	2006-02-15 09:44:00
4	4	Angola	2006-02-15 09:44:00
5	5	Anguilla	2006-02-15 09:44:00
6	6	Argentina	2006-02-15 09:44:00
7	7	Armenia	2006-02-15 09:44:00
8	8	Australia	2006-02-15 09:44:00
9	9	Austria	2006-02-15 09:44:00
10	10	Azerbaijan	2006-02-15 09:44:00

SELECT Statements

SELECT statements are used to select data from a database

```
SELECT
  -- The column country
  country
FROM
  -- The Schema 'public'
  -- The Table 'country'
  public.country
```



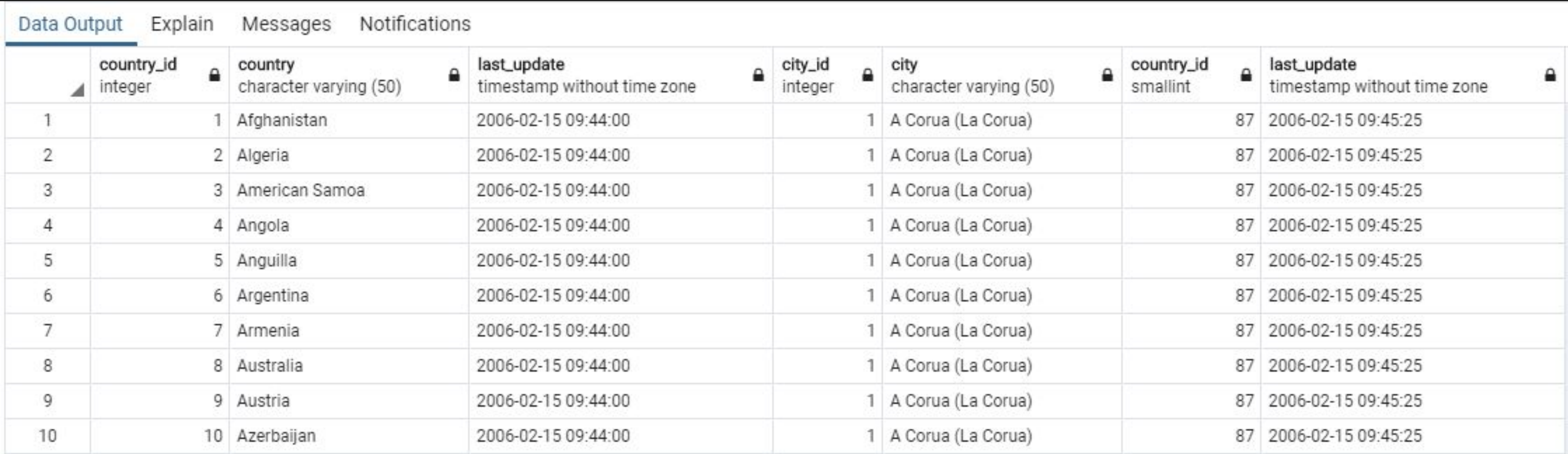
	country character varying (50)
1	Afghanistan
2	Algeria
3	American Samoa
4	Angola
5	Anguilla
6	Argentina
7	Armenia
8	Australia
9	Austria
10	Azerbaijan

SELECT Statements

```
SELECT
  *
FROM
  public.country,
  public.city
```

Two tables are included in this SELECT statement

Does anyone see what’s wrong?


Data Output Explain Messages Notifications

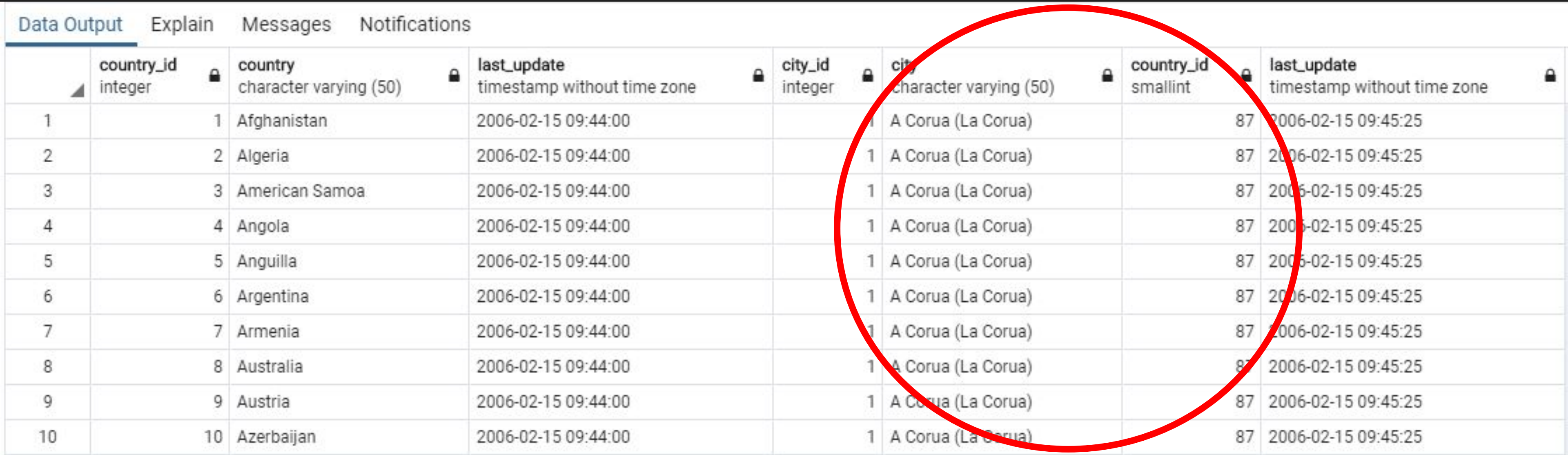
	country_id integer	country character varying (50)	last_update timestamp without time zone	city_id integer	city character varying (50)	country_id smallint	last_update timestamp without time zone
1	1	Afghanistan	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
2	2	Algeria	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
3	3	American Samoa	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
4	4	Angola	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
5	5	Anguilla	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
6	6	Argentina	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
7	7	Armenia	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
8	8	Australia	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
9	9	Austria	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
10	10	Azerbaijan	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25

SELECT Statements

```
SELECT
  *
FROM
  public.country,
  public.city
```

Two tables are included in this SELECT statement

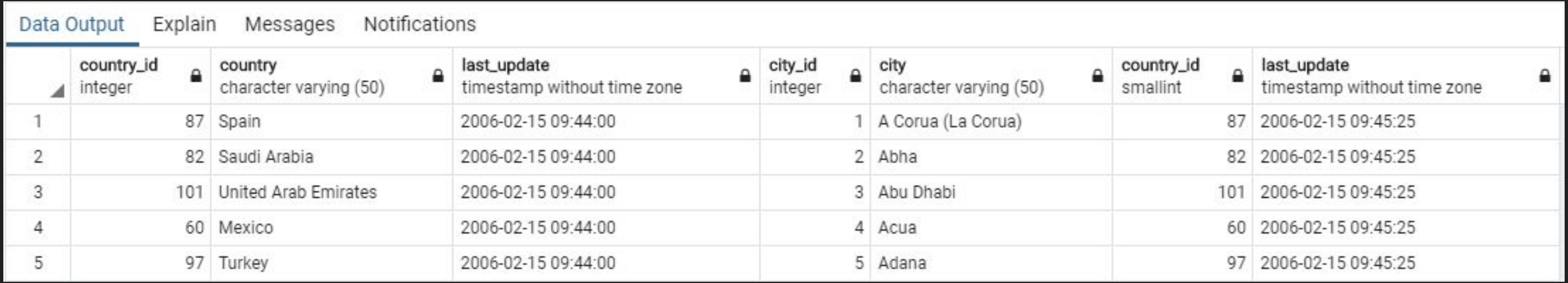
Does anyone see what's wrong?



	country_id integer	country character varying (50)	last_update timestamp without time zone	city_id integer	city character varying (50)	country_id smallint	last_update timestamp without time zone
1	1	Afghanistan	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
2	2	Algeria	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
3	3	American Samoa	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
4	4	Angola	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
5	5	Anguilla	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
6	6	Argentina	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
7	7	Armenia	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
8	8	Australia	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
9	9	Austria	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
10	10	Azerbaijan	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25

SELECT Statements

```
SELECT
  *
FROM
  public.country,
  public.city
WHERE
  country.country_id = city.country_id
```

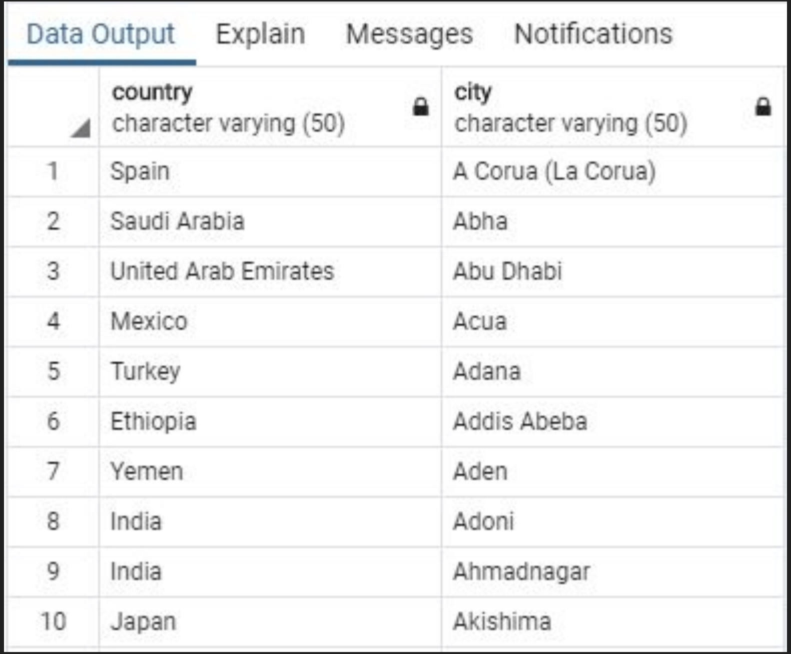


Data Output | Explain | Messages | Notifications

	country_id integer	country character varying (50)	last_update timestamp without time zone	city_id integer	city character varying (50)	country_id smallint	last_update timestamp without time zone
1	87	Spain	2006-02-15 09:44:00	1	A Corua (La Corua)	87	2006-02-15 09:45:25
2	82	Saudi Arabia	2006-02-15 09:44:00	2	Abha	82	2006-02-15 09:45:25
3	101	United Arab Emirates	2006-02-15 09:44:00	3	Abu Dhabi	101	2006-02-15 09:45:25
4	60	Mexico	2006-02-15 09:44:00	4	Acua	60	2006-02-15 09:45:25
5	97	Turkey	2006-02-15 09:44:00	5	Adana	97	2006-02-15 09:45:25

SELECT Statements

```
SELECT
  country.country,
  city.city
FROM
  public.country,
  public.city
WHERE
  country.country_id = city.country_id
```



	country character varying (50)	city character varying (50)
1	Spain	A Corua (La Corua)
2	Saudi Arabia	Abha
3	United Arab Emirates	Abu Dhabi
4	Mexico	Acua
5	Turkey	Adana
6	Ethiopia	Addis Abeba
7	Yemen	Aden
8	India	Adoni
9	India	Ahmadnagar
10	Japan	Akishima

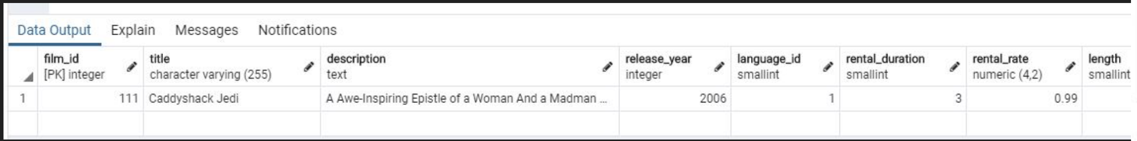
SELECT Statements - Equi Join

These aren’t very useful results though.

Let’s say we wanted to find out all the people who have rented the movie ‘Caddyshack Jedi’

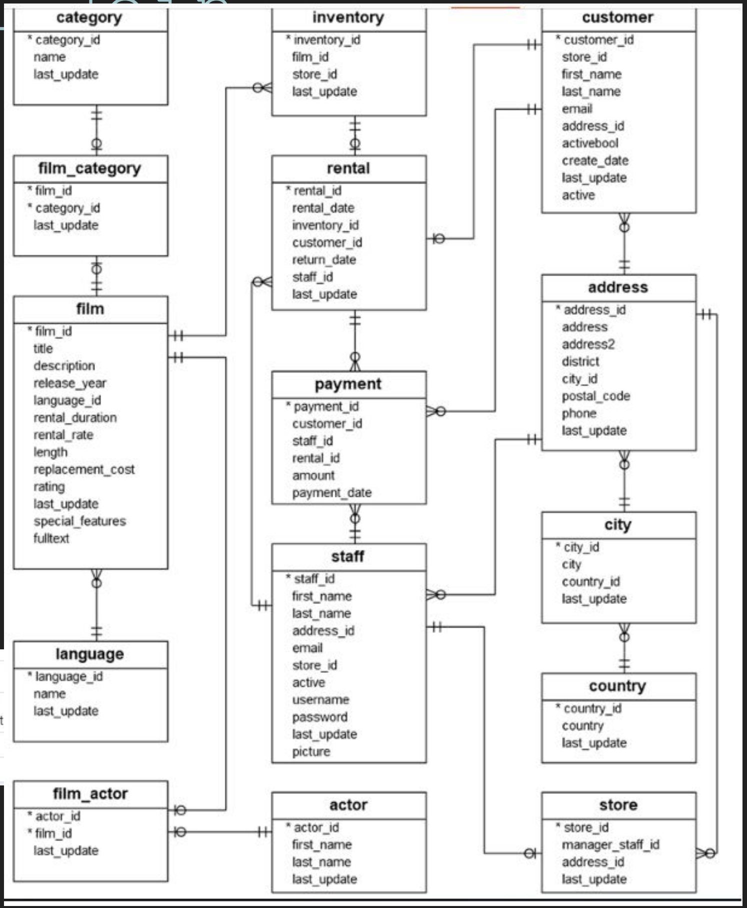
SELECT Statements - Equi

```
SELECT
  *
FROM
  public.film
WHERE
  public.film.title = 'Caddyshack Jedi'
```



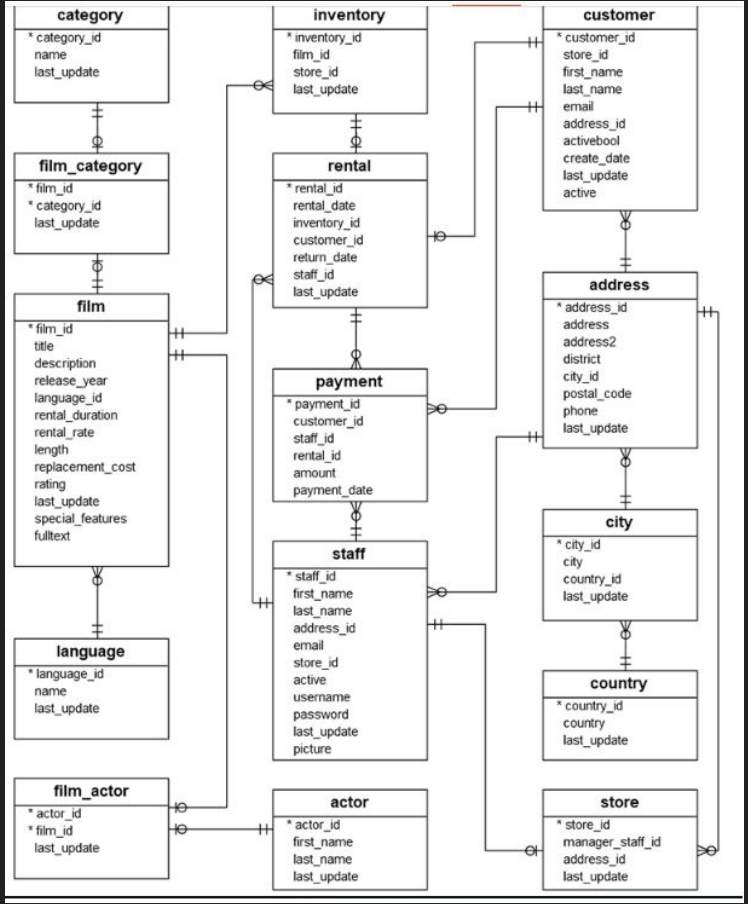
Data Output | Explain | Messages | Notifications

	film_id [PK] integer	title character varying (255)	description text	release_year integer	language_id smallint	rental_duration smallint	rental_rate numeric (4,2)	length smallint
1	111	Caddyshack Jedi	A Awe-Inspiring Epistle of a Woman And a Madman ...	2006	1	3	0.99	



SELECT Statements - Equi Join

```sql
SELECT
  public.film.title
FROM
  public.film,
  public.inventory
WHERE
  public.film.film_id = public.inventory.film_id
  AND public.film.title = 'Caddyshack Jedi'
```



SELECT Statements - Equi Join

```
SELECT
  public.film.title
FROM
  public.film,
  public.inventory,
  public.rental
WHERE
  public.film.film_id = public.inventory.film_id
  AND public.inventory.inventory_id = public.rental.inventory_id
  AND public.film.title = 'Caddyshack Jedi'
```

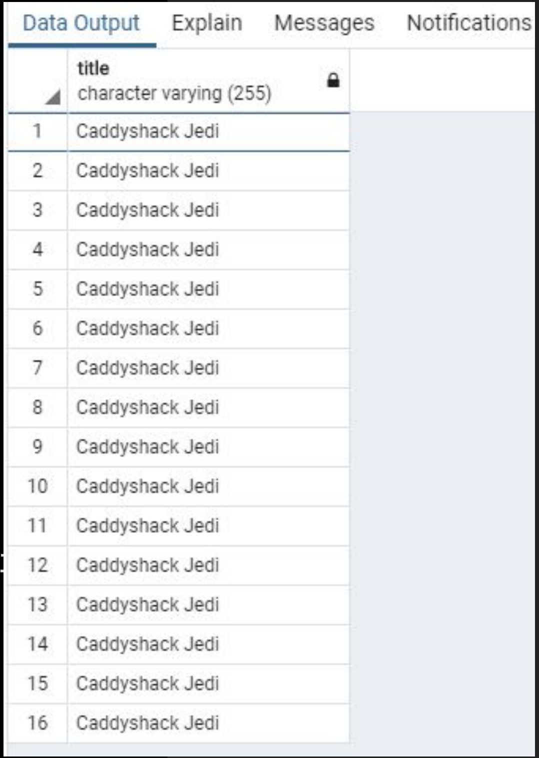
SELECT Statements - Equi Join

```
SELECT
  public.film.title
FROM
  public.film,
  public.inventory,
  public.rental,
  public.customer
WHERE
  public.film.film_id = public.inventory.film_id
  AND public.inventory.inventory_id = public.rental.inventory_id
  AND public.rental.customer_id = public.customer.customer_id
  AND public.film.title = 'Caddyshack Jedi'
```

SELECT Statements - Equi Join

```
SELECT
  public.film.title
FROM
  publ
  publ
  publ
  publ
WHERE
  public.film.film_id = public.inventory.film_id
  AND public.inventory.inventory_id = public.rental.inventory_
  AND public.rental.customer_id = public.customer.customer_id
  AND public.film.title = 'Caddyshack Jedi'
```

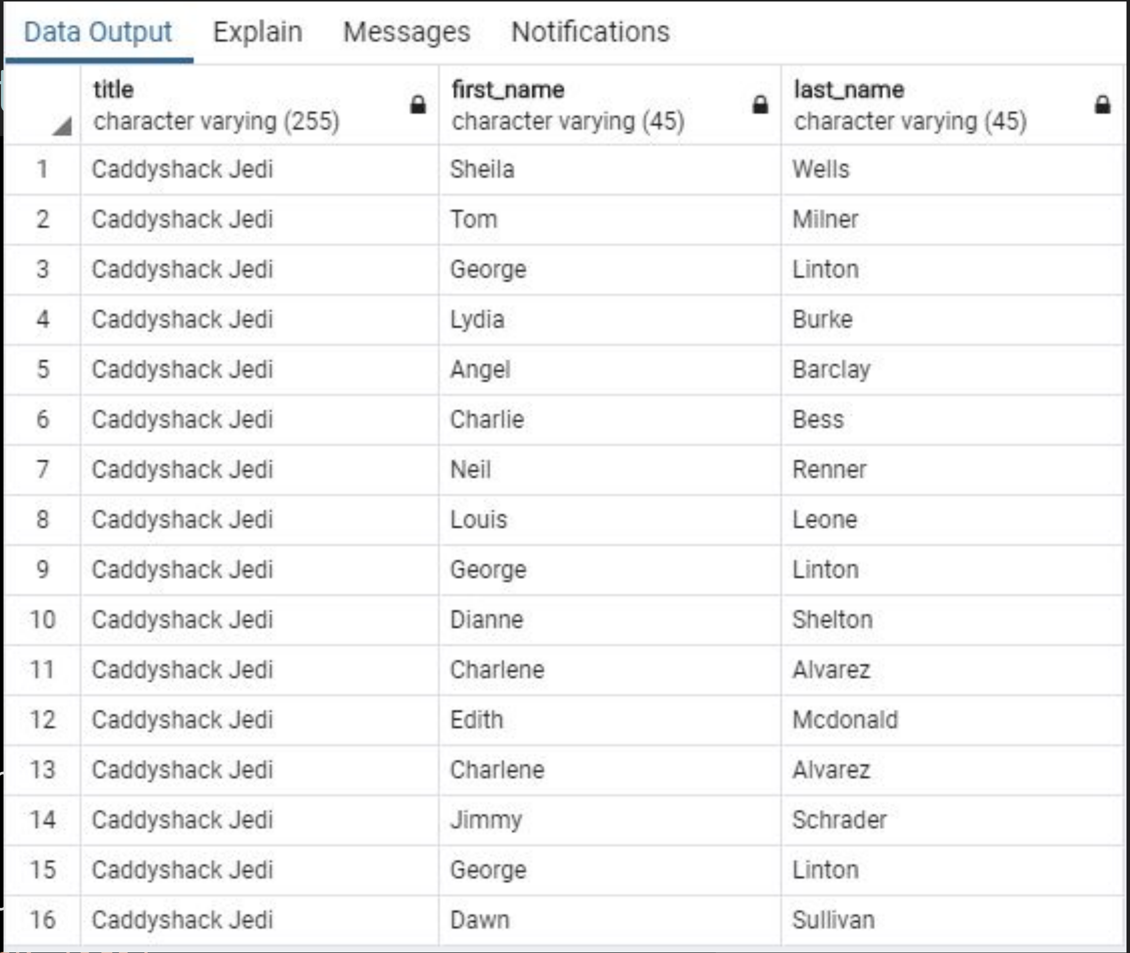
How do we fix this?


Data Output Explain Messages Notifications

	title character varying (255)
1	Caddyshack Jedi
2	Caddyshack Jedi
3	Caddyshack Jedi
4	Caddyshack Jedi
5	Caddyshack Jedi
6	Caddyshack Jedi
7	Caddyshack Jedi
8	Caddyshack Jedi
9	Caddyshack Jedi
10	Caddyshack Jedi
11	Caddyshack Jedi
12	Caddyshack Jedi
13	Caddyshack Jedi
14	Caddyshack Jedi
15	Caddyshack Jedi
16	Caddyshack Jedi

SELECT Statement

```
SELECT
  public.film.title,
  public.customer.first_name,
  public.customer.last_name
FROM
  public.film,
  public.inventory,
  public.rental,
  public.customer
WHERE
  public.film.film_id = public.inver
  AND public.inventory.inventory_id
  AND public.rental.customer_id = pu
  AND public.film.title = 'Caddyshack Jedi'
```

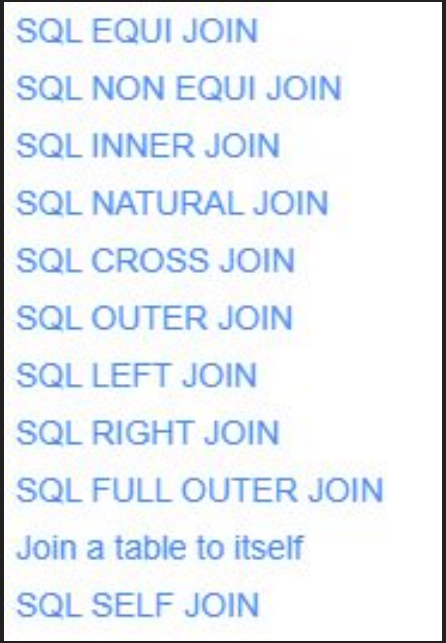


Data Output | Explain | Messages | Notifications

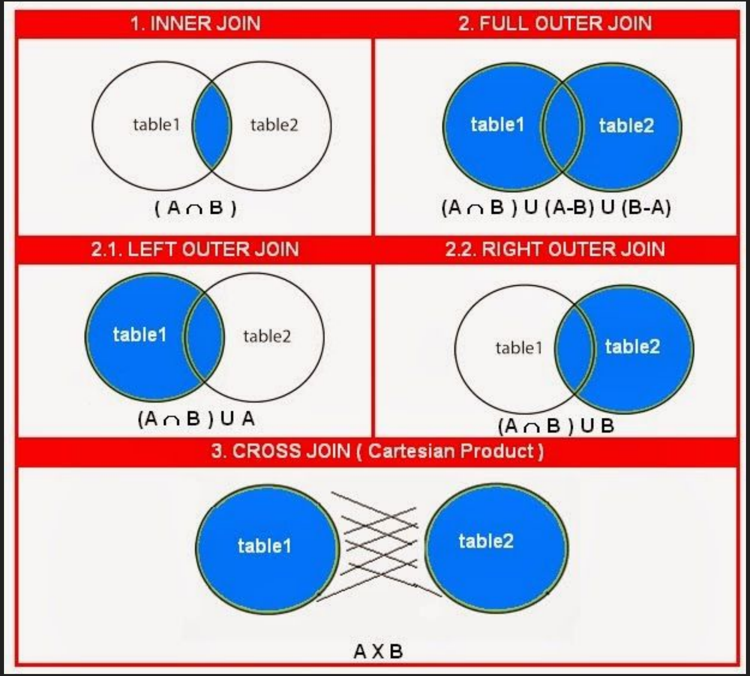
	title character varying (255)	first_name character varying (45)	last_name character varying (45)
1	Caddyshack Jedi	Sheila	Wells
2	Caddyshack Jedi	Tom	Milner
3	Caddyshack Jedi	George	Linton
4	Caddyshack Jedi	Lydia	Burke
5	Caddyshack Jedi	Angel	Barclay
6	Caddyshack Jedi	Charlie	Bess
7	Caddyshack Jedi	Neil	Renner
8	Caddyshack Jedi	Louis	Leone
9	Caddyshack Jedi	George	Linton
10	Caddyshack Jedi	Dianne	Shelton
11	Caddyshack Jedi	Charlene	Alvarez
12	Caddyshack Jedi	Edith	Mcdonald
13	Caddyshack Jedi	Charlene	Alvarez
14	Caddyshack Jedi	Jimmy	Schrader
15	Caddyshack Jedi	George	Linton
16	Caddyshack Jedi	Dawn	Sullivan

SELECT Statements

There are several types of joins for SELECT statements



SQL EQUI JOIN
SQL NON EQUI JOIN
SQL INNER JOIN
SQL NATURAL JOIN
SQL CROSS JOIN
SQL OUTER JOIN
SQL LEFT JOIN
SQL RIGHT JOIN
SQL FULL OUTER JOIN
Join a table to itself
SQL SELF JOIN



https://www.w3resource.com/sql/joins/sql-joins.php

SELECT Statements

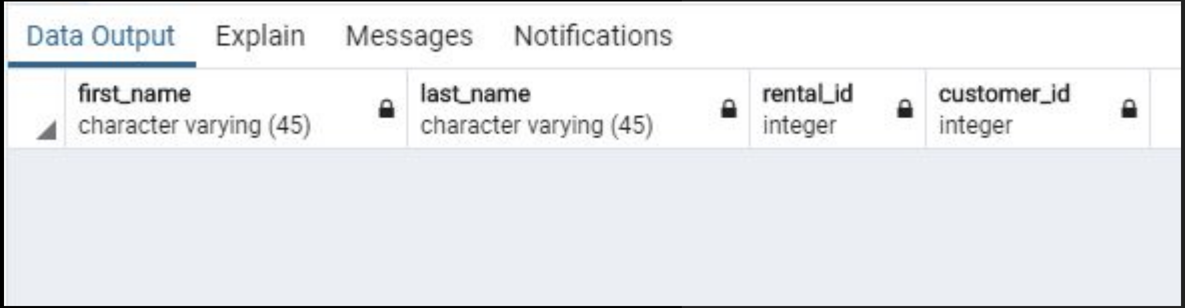
In the interest of time, I’ll only demonstrate a LEFT JOIN

SELECT Statements - Left Join

```sql
SELECT
  public.customer.first_name,
  public.customer.last_name,
  public.rental.rental_id,
  public.customer.customer_id
FROM
  public.customer
  LEFT JOIN public.rental
    ON public.customer.customer_id = public.rental.customer_id
```

SELECT Statements - Left Join

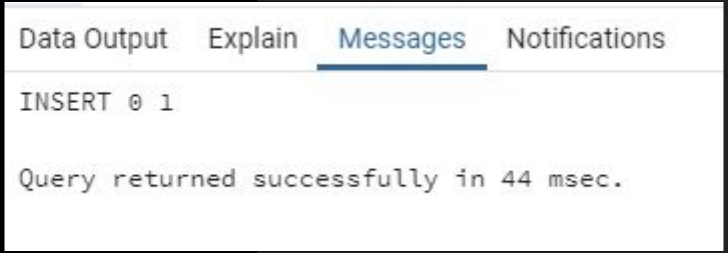
```
SELECT
  public.customer.first_name,
  public.customer.last_name,
  public.rental.rental_id,
  public.customer.customer_id
FROM
  public.customer
  LEFT JOIN public.rental
    ON public.customer.customer_id = public.rental.customer_id
WHERE
  rental_id IS NULL
```



first_name character varying (45)	last_name character varying (45)	rental_id integer	customer_id integer

SELECT Statements - Left Join

```
INSERT INTO
  Customer
VALUES (
  600,
  1,
  'Lily',
  'Potter',
  'lily_potter@hogwarts.com',
  1,
  true,
  now(),
  now()
)
```

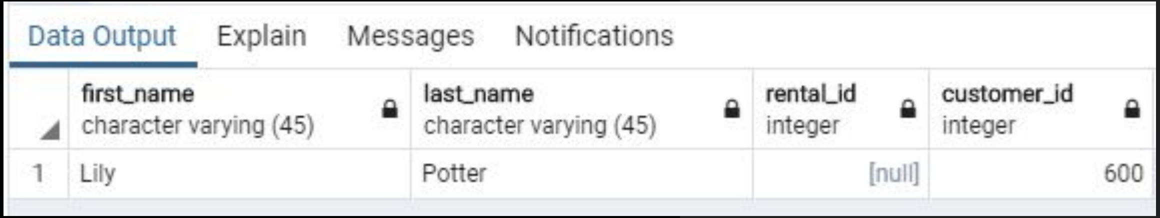


```
INSERT 0 1

Query returned successfully in 44 msec.
```

SELECT Statements - Left Join

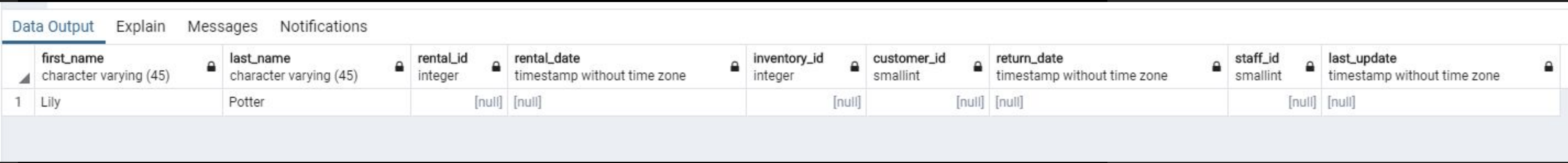
```
SELECT
  public.customer.first_name,
  public.customer.last_name,
  public.rental.rental_id,
  public.customer.customer_id
FROM
  public.customer
  LEFT JOIN public.rental
    ON public.customer.customer_id = public.rental.customer_id
WHERE
  rental_id IS NULL
```



	first_name character varying (45)	last_name character varying (45)	rental_id integer	customer_id integer
1	Lily	Potter	[null]	600

SELECT Statements - Left Join

```
SELECT
  public.customer.first_name,
  public.customer.last_name,
  public.rental.*
FROM
  public.customer
  LEFT JOIN public.rental
    ON public.customer.customer_id = public.rental.customer_id
WHERE
```

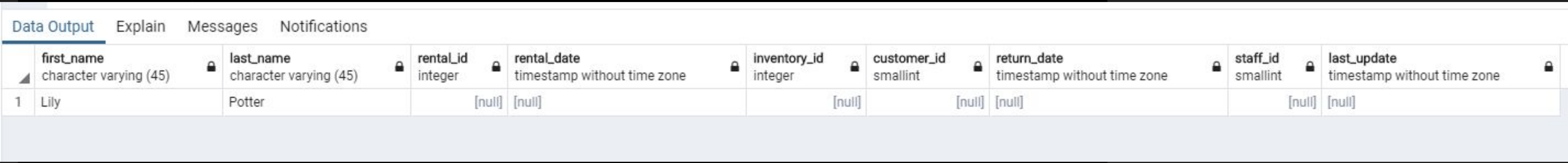


	first_name character varying (45)	last_name character varying (45)	rental_id integer	rental_date timestamp without time zone	inventory_id integer	customer_id smallint	return_date timestamp without time zone	staff_id smallint	last_update timestamp without time zone
1	Lily	Potter	[null]	[null]	[null]	[null]	[null]	[null]	[null]

SELECT Statements - Left Join

```
SELECT
  public
  public
  public
FROM
  public
  LEFT J
    ON p
WHERE
```

Lily has never rented a movie, hence all of her results are expectedly NULL



Data Output Explain Messages Notifications

	first_name character varying (45)	last_name character varying (45)	rental_id 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	Lily	Potter	[null]	[null]	[null]	[null]	[null]	[null]	[null]

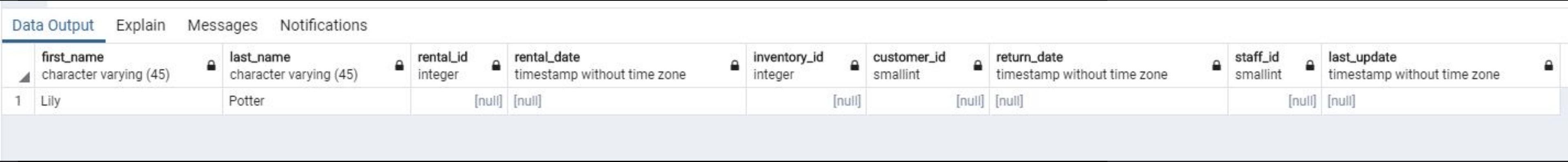
SELECT Statements - Left Join

```
SELECT
  public
  public
  public
FROM
  public
  LEFT J
    ON p
WHERE
```

Lily has never rented a movie, hence all of her results are expectedly NULL

You may want to do this to answer the question:

“What customers have not rented a movie yet?”



Data Output | Explain | Messages | Notifications

	first_name character varying (45)	last_name character varying (45)	rental_id 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	Lily	Potter	[null]	[null]	[null]	[null]	[null]	[null]	[null]

SELECT Statements - Some For You To Try

See if you can come up with SELECT statements for the following questions:

1. Find the First and Last Names of the customers who have rented ‘Caddyshack Jedi’ OR ‘Born Spinal’

SELECT Statements - Some For You To Try

See if you can come up with SELECT statements for the following questions:

2. Find the name of all the customers from ‘Vancouver’

SELECT Statements - Some For You To Try

See if you can come up with SELECT statements for the following questions:

3. Find all the customers who have paid more than $10