

# Modelo relacional. Vistas y disparadores

Lihao Zhu, Darío Domínguez González.

1. Realice la restauración de la base de datos [alquilerdvd.tar](#). Observe que la base de datos no tiene un formato SQL como el empleado en actividades anteriores.

Para restaurar la base de datos, usamos los siguientes comandos:

```
sudo -u postgres createdb Alquiler
pg_restore -d Alquiler -U postgres -h localhost -p 5432
/home/usuario/ADBD/AlquilerPractica.tar
```

Siendo /home/usuario/ADBD/AlquilerPractica.tar la ruta al tar.

2. Identifique las tablas, vistas y secuencias.

```
Alquiler=# \dt
          List of relations
Schema |      Name      | Type  | Owner
-----+-----+-----+-----
public | actor           | table | postgres
public | address         | table | postgres
public | category        | table | postgres
public | city            | table | postgres
public | country         | table | postgres
public | customer        | table | postgres
public | film            | table | postgres
public | film_actor      | table | postgres
public | film_category   | table | postgres
public | inventory       | table | postgres
public | language        | table | postgres
public | payment         | table | postgres
public | rental          | table | postgres
public | staff           | table | postgres
public | store           | table | postgres
(15 rows)
```

List of relations				
Schema	Name	Type	Owner	Table
public	actor_pkey	index	postgres	actor
public	address_pkey	index	postgres	address
public	category_pkey	index	postgres	category
public	city_pkey	index	postgres	city
public	country_pkey	index	postgres	country
public	customer_pkey	index	postgres	customer
public	film_actor_pkey	index	postgres	film_actor
public	film_category_pkey	index	postgres	film_category
public	film_fulltext_idx	index	postgres	film
public	film_pkey	index	postgres	film
public	idx_actor_last_name	index	postgres	actor
public	idx_fk_address_id	index	postgres	customer
public	idx_fk_city_id	index	postgres	address
public	idx_fk_country_id	index	postgres	city
public	idx_fk_customer_id	index	postgres	payment
public	idx_fk_film_id	index	postgres	film_actor
public	idx_fk_inventory_id	index	postgres	rental
public	idx_fk_language_id	index	postgres	film
public	idx_fk_rental_id	index	postgres	payment
public	idx_fk_staff_id	index	postgres	payment
public	idx_fk_store_id	index	postgres	customer
public	idx_last_name	index	postgres	customer
public	idx_store_id_film_id	index	postgres	inventory
public	idx_title	index	postgres	film
public	idx_unq_manager_staff_id	index	postgres	store
public	idx_unq_rental_rental_date_inventory_id_customer_id	index	postgres	rental
public	inventory_pkey	index	postgres	inventory
public	language_pkey	index	postgres	language
public	payment_pkey	index	postgres	payment
public	rental_pkey	index	postgres	rental
public	staff_pkey	index	postgres	staff
public	store_pkey	index	postgres	store
(32 rows)				

### 3. Identifique las tablas principales y sus principales elementos.

#### 1. Tabla customer

- Propósito: Almacena información sobre los clientes.
- Elementos principales:
  - customer\_id: Clave primaria para identificar a cada cliente.
  - first\_name y last\_name: Nombre y apellido del cliente.
  - email: Dirección de correo electrónico del cliente.
  - address\_id: Referencia a la tabla address para la dirección del cliente.
  - active: Indica si el cliente está activo (valores 0 o 1).
  - create\_date y last\_update: Fechas de creación y última actualización del registro.

#### 2. Tabla film

- Propósito: Almacena información sobre cada película disponible.
- Elementos principales:
  - film\_id: Clave primaria para identificar cada película.
  - title: Título de la película.
  - description: Breve descripción de la película.

- release\_year: Año de lanzamiento.
- language\_id: Referencia al idioma de la película.
- rental\_duration y rental\_rate: Duración y tarifa del alquiler.
- length: Duración de la película en minutos.
- replacement\_cost: Costo de reemplazo en caso de pérdida o daño.
- rating: Clasificación MPAA de la película.
- last\_update: Fecha de la última actualización.

### 3. Tabla inventory

- Propósito: Registra las copias físicas de cada película y su ubicación.
- Elementos principales:
  - inventory\_id: Clave primaria para cada inventario.
  - film\_id: Referencia a la película en la tabla film.
  - store\_id: Identifica la tienda que posee el inventario.
  - last\_update: Fecha de la última actualización del inventario.

### 4. Tabla rental

- Propósito: Lleva un registro de cada alquiler.
- Elementos principales:
  - rental\_id: Clave primaria para cada alquiler.
  - rental\_date: Fecha y hora en que se realizó el alquiler.
  - inventory\_id: Referencia al inventario (copia específica de la película alquilada).
  - customer\_id: Referencia al cliente que realizó el alquiler.
  - return\_date: Fecha y hora de la devolución.
  - staff\_id: Identifica al miembro del personal que gestionó el alquiler.
  - last\_update: Fecha de la última actualización del registro.

### 5. Tabla payment

- Propósito: Registra los pagos de los clientes.
- Elementos principales:
  - payment\_id: Clave primaria para cada pago.
  - customer\_id: Referencia al cliente que realizó el pago.
  - staff\_id: Referencia al miembro del personal que gestionó el pago.
  - rental\_id: Referencia al alquiler relacionado con el pago.
  - amount: Cantidad pagada.
  - payment\_date: Fecha y hora en que se realizó el pago.

### 6. Tabla address

- Propósito: Contiene la información de direcciones, que puede ser referenciada por clientes y personal.

- Elementos principales:
  - address\_id: Clave primaria para cada dirección.
  - address, address2, district, city\_id: Campos para la dirección completa.
  - postal\_code y phone: Código postal y teléfono de contacto.
  - last\_update: Fecha de la última actualización de la dirección.

#### 4. Realice las siguientes consultas.

- Obtenga las ventas totales por categoría de películas ordenadas descendientemente.

```


1  SELECT
2      s.store_id AS tienda,
3      CONCAT(ci.city, ', ', co.country) AS ubicacion,
4      CONCAT(st.first_name, ' ', st.last_name) AS encargado,
5      SUM(p.amount) AS ventas_totales
6  FROM
7      payment p
8  JOIN
9      rental r ON p.rental_id = r.rental_id
10 JOIN
11     inventory i ON r.inventory_id = i.inventory_id
12 JOIN
13     store s ON i.store_id = s.store_id
14 JOIN
15     address a ON s.address_id = a.address_id
16 JOIN
17     city ci ON a.city_id = ci.city_id
18 JOIN
19     country co ON ci.country_id = co.country_id
20 JOIN
21     staff st ON s.manager_staff_id = st.staff_id
22 GROUP BY
23     s.store_id, ubicacion, encargado
24 ORDER BY
25     ventas_totales DESC;

```

	categoria character varying (25) 🔒	ventas_totales numeric 🔒
1	Sports	4892.19
2	Sci-Fi	4336.01
3	Animation	4245.31
4	Drama	4118.46
5	Comedy	4002.48
6	New	3966.38
7	Action	3951.84
8	Foreign	3934.47
9	Games	3922.18
10	Family	3830.15
11	Documentary	3749.65
12	Horror	3401.27
13	Classics	3353.38
14	Children	3309.39
15	Travel	3227.36
16	Music	3071.52

- b. Obtenga las ventas totales por tienda, donde se refleje la ciudad, el país (concatenar la ciudad y el país empleando como separador la "\","), y el encargado. Pudiera emplear GROUP BY, ORDER BY

```
1  SELECT
2      c.name AS categoria,
3      SUM(p.amount) AS ventas_totales
4  FROM
5      payment p
6  JOIN
7      rental r ON p.rental_id = r.rental_id
8  JOIN
9      inventory i ON r.inventory_id = i.inventory_id
10 JOIN
11     film f ON i.film_id = f.film_id
12 JOIN
13     film_category fc ON f.film_id = fc.film_id
14 JOIN
15     category c ON fc.category_id = c.category_id
16 GROUP BY
17     c.name
18 ORDER BY
19     ventas_totales DESC;
```




	tienda 	ubicacion 	encargado 	ventas_totales 
	integer	text	text	numeric
1	2	Woodridge, Australia	Jon Stephens	30683.13
2	1	Lethbridge, Canada	Mike Hillyer	30628.91

- c. Obtenga una lista de películas, donde se reflejen el identificador, el título, descripción, categoría, el precio, la duración de la película, clasificación, nombre y apellidos de los actores (puede realizar una concatenación de ambos). Pudiera emplear GROUP BY

```

1  SELECT
2      f.film_id AS identificador,
3      f.title AS titulo,
4      f.description AS descripcion,
5      c.name AS categoria,
6      f.rental_rate AS precio,
7      f.length AS duracion,
8      f.rating AS clasificacion,
9      STRING_AGG(CONCAT(a.first_name, ' ', a.last_name), ', ') AS actores
10 FROM
11     film f
12 JOIN
13     film_category fc ON f.film_id = fc.film_id
14 JOIN
15     category c ON fc.category_id = c.category_id
16 JOIN
17     film_actor fa ON f.film_id = fa.film_id
18 JOIN
19     actor a ON fa.actor_id = a.actor_id
20 GROUP BY
21     f.film_id, f.title, f.description, c.name, f.rental_rate, f.length, f.rating;

```

	 titulo character varying (255)	 descripcion text	
1	Academy Dinosaur	A Epic Drama of a Feminist And a Mad Scientist who must Battle a Teacher in The Canadian Rockies	
2	Ace Goldfinger	A Astounding Epistle of a Database Administrator And a Explorer who must Find a Car in Ancient China	
3	Adaptation Holes	A Astounding Reflection of a Lumberjack And a Car who must Sink a Lumberjack in A Baloon Factory	
4	Affair Prejudice	A Fanciful Documentary of a Frisbee And a Lumberjack who must Chase a Monkey in A Shark Tank	
5	African Egg	A Fast-Paced Documentary of a Pastry Chef And a Dentist who must Pursue a Forensic Psychologist in The Gulf of Mexico	
6	Agent Truman	A Intrepid Panorama of a Robot And a Boy who must Escape a Sumo Wrestler in Ancient China	
7	Airplane Sierra	A Touching Saga of a Hunter And a Butler who must Discover a Butler in A Jet Boat	
8	Airport Pollock	A Epic Tale of a Moose And a Girl who must Confront a Monkey in Ancient India	
9	Alabama Devil	A Thoughtful Panorama of a Database Administrator And a Mad Scientist who must Outgun a Mad Scientist in A Jet Boat	
10	Aladdin Calendar	A Action-Packed Tale of a Man And a Lumberjack who must Reach a Feminist in Ancient China	
11	Alamo Videotape	A Boring Epistle of a Butler And a Cat who must Fight a Pastry Chef in A MySQL Convention	
12	Alaska Phantom	A Fanciful Saga of a Hunter And a Pastry Chef who must Vanquish a Boy in Australia	
13	Ali Forever	A Action-Packed Drama of a Dentist And a Crocodile who must Battle a Feminist in The Canadian Rockies	

d. Obtenga la información de los actores, donde se incluya sus nombres y apellidos, las categorías y sus películas. Los actores deben de estar agrupados y, las categorías y las películas deben estar concatenados por ":"

```

1  SELECT
2      a.actor_id AS id_actor,
3      CONCAT(a.first_name, ' ', a.last_name) AS actor,
4      STRING_AGG(CONCAT(c.name, ': ', f.title), ', ') AS categorias_peliculas
5  FROM
6      actor a
7  JOIN
8      film_actor fa ON a.actor_id = fa.actor_id
9  JOIN
10     film f ON fa.film_id = f.film_id
11  JOIN
12     film_category fc ON f.film_id = fc.film_id
13  JOIN
14     category c ON fc.category_id = c.category_id
15  GROUP BY
16     a.actor_id, actor;

```

	id_actor integer	actor text	categorias_peliculas text
1	70	Michelle McConaughey	Family: Baked Cleopatra, Family: Bang Kwai, Horror: Bowfinger Gables, Comedy: Daddy Pittsburgh, Games: Details Packer, Classics: Dracula Crystal, Foreign: Ev
2	65	Angela Hudson	Sci-Fi: Armageddon Lost, Games: Autumn Crow, Action: Bride Intrigue, Games: Bulworth Commandments, Games: Candles Grapes, Travel: Cassidy Wyoming, M
3	198	Mary Keitel	Documentary: Academy Dinosaur, New: Butterfly Chocolat, Travel: Cassidy Wyoming, Drama: Craft Outfield, Family: Dumbo Lust, Games: Dwarfs Alter, Action: Fi
4	80	Ralph Cruz	Sci-Fi: Beverly Outlaw, Animation: Canyon Stock, Children: Casper Dragonfly, Family: Confused Candles, Foreign: Dangerous Uptown, Action: Darn Forrester, Spo
5	100	Spencer Depp	Music: Alone Trip, Animation: Canyon Stock, Action: Dragon Squad, Music: Heavenly Gun, Foreign: Hellfighters Sierra, Travel: Leathernecks Dwarfs, Games: Mas
6	76	Angelina Astaire	Classics: Beast Hunchback, Children: Beneath Rush, Children: Betrayed Rear, New: Breakfast Goldfinger, Horror: Carrie Bunch, Sports: Cranes Reservoir, Animati
7	131	Jane Jackman	Children: Backlash Undefeated, Children: Beneath Rush, Family: Braveheart Human, Sports: Caribbean Liberty, Family: Chocolat Harry, Documentary: Dancing Fe
8	62	Jayne Neeson	Foreign: Agent Truman, Sports: Artist Coldblooded, Music: Banger Pinocchio, Foreign: Brooklyn Desert, Documentary: Brotherhood Blanket, Documentary: Caus
9	54	Penelope Pinkett	Travel: Boiled Dares, Documentary: Cause Date, Documentary: Cider Desire, Classics: Core Suit, Sci-Fi: English Bulworth, Action: Excitement Eve, Horror: Family
10	153	Minnie Kilmer	Foreign: Baby Hall, Drama: Beethoven Exorcist, New: Chaplin License, Classics: Conspiracy Spirit, Sci-Fi: Daisy Menagerie, Family: Dinosaur Secretary, Comedy:
11	104	Penelope Cronyn	Action: Amadeus Holy, Sci-Fi: Armageddon Lost, Documentary: Army Flintstones, Children: Bear Graceland, Animation: Bikini Borrowers, New: Chaplin License, I
12	173	Alan Dreyfuss	Sci-Fi: Badman Dawn, Sci-Fi: Barbarella Streetcar, Music: Birch Antitrust, Family: Blanket Beverly, Games: Bulworth Commandments, Animation: Clash Freddy, A
13	180	Jeff Silverstone	Music: Alaska Phantom, Drama: Apollo Teen, New: Chinatown Gladiator, Sci-Fi: Crowds Telemark, Horror: Drums Dynamite, Documentary: Hunter Alter, Horror: L

Las vistas creadas con esencialmente lo mismo que las consultas con la diferencia de que se añade la línea **CREATE VIEW [nombre de vista] AS** previo a la consulta, siendo estos:

10_actor   actor	categorias_peliculas
	<p>70   Michelle McKeownagh   Family: Robot Cavewar, Family: Bang Bang, Horror: Bowfinger Gables, Comedy: Duddy Pittsburgh, Games: Details Packer, Classics: Dracula Crystal, Foreign: Everyone Craft, Children: Fargo Gwahl, Children: Full Flatliners, Foreign: Hellfighters Sierra, Sports: Image Perish, Documentary: Intolerable Intentions, Drama: Male Homeward, Foreign: Mixed Doors, Sports: Northwest Polish, Games: Penny Submarine, Classics: Requin Tycoon, Action: South Mat, Documentary: Spoilers Hellfighters, Horror: Streator Intensions, Action: Suspects Gulls, Animation: Walt Cider, Action: Waterfront Deliverance</p> <p>61   Angela Hudson   1 Sci-Fi: Armageddon Lost, Action: Crown Arow, Action: Bridge Intrigue, Games: Bulworth Commandments, Games: Candies Grapes, Travel: Cassidy Wyoming, Music: Clones Pinocchio, Comedy: Elment Freddy, New: Fatal Haunted, Sci-Fi: Frisco Forrest, Travel: Games: Dogfingers, Action: Gasford Bonds, Music: Harver Gault, Classics: Island Sorcrist, Horror: Japanese Run, Family: Jason Iron, Children: Jaming trail, Travel: Kiss Kiss Swam, Music: Legend Bell, Sports: Lesson Gnostora, Animation: Lute Kamy, Games: Walste Hape, Documentary: Male Arasquod, Sports: Hilo Maly, Animation: Nash Ockard, Horror: Paris Weeled, Drama: Pity Bond, Classics: Prevalite Ockard, Sci-Fi: Ranzon 60, Children: Robbers Jope, Children: Stranglehoe Desire, Comedy: Velvet Tumble, Travel: Voyage Legally, Horror: Watership Frontier</p> <p>58   Mary Keitel   1 Documentary: Academy Dinosaur, New: Butterfly Choclat, Travel: Cassidy Wyoming, Drama: Craft Outfield, Music: Dumbo Lust, Games: Dwarfs Alter, Action: Fantasy Troopers, Family: Feud Frogsen, Foreign: Fiction Christmas, Action: Feud Frost, Travel: Games: Bowfinger, Drama: Grendy Rocks, Action: Handicap Boundry, Games: Hunting Planets, Children: Idols Snatchers, Animation: Intention Emory, Classics: Jeopardy Encho, Family: King Evolution, Classics: Lovely Zingo, Animation: Lute Pamy, Foreign: Madnes Attacks, Comedy: Mollrats United, Comedy: Memento Zoalander, Sports: Mermad Intents, Children: Mord Fish, Games: Noomalord Zoo, Sports: Northwest Polish, Sports: Roses Treasure, Drama: Saints Bride, Sports: Sierra Divide, New: Sleepy Japanese, Sci-Fi: Soldiers Evolution, Classics: Stela Santa, Comedy: Submarine Bed, Comedy: Sweden Shining, Animation: Theory Mermad, Animation: Titanic Boondock, Sci-Fi: Unforgiven Zoalander, Documentary: Wagon Jaws, Music: Youth Kick</p> <p>40   Ralph Cruz   Sci-Fi: Beverly Outlaw, Animation: Canyon Stone, Children: Casper Dragonfly, Family: Confused Gables, Foreign: Dangerous outow, Action: Dorn Forrester, Sports: Duke Blindness, Family: Dumbo Lust, Children: Empire Velovich, Classics: Frost Head, Travel: Fugitive Handre, Children: Full Flatliners, Games: Gley Tracy, New: Hars Rape, Horror: Japanese Run, Children: Hare Gahaly, Sports: Neighbors Darade, Documentary: Newlies Story, Comedy: Pinocchio Sam, Documentary: Focus Pulp, Classics: Polish Brooklyn, Drama: Racer Egg, Documentary: Tact Wonderland, Games: Sleuth Orkney, Comedy: Submarine Bed, Documentary: Thin Sagebrush, Games: Videopage Arsenic, Drama: Mitches Pan</p>

identificador	titulo	actores	descripcion	categoria	precio	duracion	clasificacion	
1	Academy Dinosaur		A Epic Drama of a Feminist and a Mad Scientist who must Battle a Teacher in The Canadian Rockies	Documentary	0.99	86	PG	Rock Dukakis, Mary Keitel, Johnny Ca
ge, Penelope Guinness, Sanders Peak, Christian		Gable, Oprah Kilmer, Warren Nolte, Lucille Tracy, Mena Temple						
2	Ace Golderinger		A Astounding Epistle of a Database Administrator And a Explorer who must Find a Car in Ancient China	Horror	4.99	48	6	Minnie Zellenger, Chris Depp, Bob Fe
uett, Sean Guinness								
3	Adaptation Holes		A Astounding Reflection of a Lumberjack And a Car who must Sink a Lumberjack in A Baloon Factory	Documentary	2.99	58	NC-17	Cameron Streep, Bob Fawcett, Nick He
ilberg, Ray Johansson, Julianne Donch								
4	Affair: Prejudice		A Fasciful Documentary of a Frisbee And a Lumberjack who must Chase a Monkey In A Shark Tank	Horror	2.99	117	6	Jodie Degeneres, Kenneth Pesci, Fay F
Winslet, Oprah Kilmer, Scarlett Damon								
5	African Ice		A Fast-Paced Documentary of a Pastry Chef And a Dentist who must Pursue a Forensic Psychologist In The Gulf of Mexico	Familly	2.99	138	6	Dustin Tazout, Matthew Leigh, Gary F
heonits, Matthew Carrey, Thora Thople								
6	Agent Truman		A Intrepid Panorama of a Robot And a Boy who must Escape a Sumo Wrestler in Ancient China	Foreign	2.99	169	PG	Warren Nolte, Sandra Kilmer, Jayne H
eson, Morgan Hume, Kirsten Paltrow, Kenneth		Hoffman, Reese West						
7	Airplane Sierra		A Truacing Saga of a Hunter And a Butler who must Discover a Butler in A Jet Boat	Comedy	4.99	62	PG-13	Mena Hopper, Jim Mostel, Michael Bol
ger, Oprah Kilmer, Richard Penn								
8	Arlo Povolok		A Epic Tale of a Moose And a Girl who must Confront a Monkey in Ancient India	Horror	4.99	54	R	Lucille Dee, Susan Davis, Fay Kilmer
Gene Mills								
9	Atlambas Dell		A Thoughtful Panorama of a Database Administrator And a Mad Scientist who must Outgun a Mad Scientist in A Jet Boat	Horror	2.99	114	PG-13	William Hucksan, Rip Crawford, Rip K
inslet, Greta Keitel, Christian Gable, Mena		Temple, Mary Allen, Warren Nolte, Elvis Har						
10	Aviation Explorer		A Action-Rocket Tale of a Man And a Lumberjack who must Reach a Feminist in Ancient China	Sports	4.99	63	NC-17	Greta Walden, Rock Dukakis, Ray Joh
esson, Renee West, Val Roberts, Judy Dean		Jedi Potter, Alice Munn						

```
Alquiler=# SELECT * FROM view_total_sales;
```

categoria	ventas_totales
Sports	4892.19
Sci-Fi	4336.01
Animation	4245.31
Drama	4118.46
Comedy	4002.48
New	3966.38
Action	3951.84
Foreign	3934.47
Games	3922.18
Family	3830.15
Documentary	3749.65
Horror	3401.27
Classics	3353.38
Children	3309.39
Travel	3227.36
Music	3071.52

(16 rows)



#### d) view\_total\_sales\_by\_store

```
Alquiler=# SELECT * FROM view_total_sales_by_store;
```

tienda	ubicacion	encargado	ventas_totales
2	Woodridge, Australia	Jon Stephens	30683.13
1	Lethbridge, Canada	Mike Hillyer	30628.91

(2 rows)

### 6. Haga un análisis del modelo e incluye las restricciones **CHECK** que considere necesarias.

El archivo contiene un volcado de una base de datos PostgreSQL llamada alquilerdvd. Cuenta con 15 tablas con sus respectivos atributos. Las restricciones que incluimos son:

- Tabla **customer**:

**CHECK (email ~\***

**'^[A-Za-z0-9.\_%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}\$')**: para verificar que el formato de los correos electrónicos sea válido.

- Tabla **film**:

**CHECK (rental\_duration > 0)**: para asegurar que la duración del alquiler sea positiva.

**CHECK (rental\_rate > 0)**: para validar que la tarifa de alquiler sea mayor que cero.

**CHECK (replacement\_cost > 0)**: para que el costo de reemplazo siempre sea positivo.

**CHECK (length IS NULL OR length > 0)**: para verificar que la duración de la película sea positiva o nula si no se proporciona.

- Tabla **address**:

**CHECK (postal\_code ~ '^\\d{5}(-\\d{4})?\$')**: para permitir solo formatos de código postal válidos (por ejemplo, 12345 o 12345-6789).

**CHECK (phone ~ '^\\+?\\d{1,15}\$')**: para asegurar que el número de teléfono tenga un formato numérico apropiado (con un máximo de 15 dígitos y opcionalmente un signo + al inicio).

## 7. Explique la sentencia que aparece en la tabla `customer`

### Triggers:

```
last_updated BEFORE UPDATE ON customer
```

```
FOR EACH ROW EXECUTE PROCEDURE last_updated()
```

### Identifique alguna tabla donde se utilice una solución similar.

La sentencia establece que cada vez que se actualice un registro en esta tabla, se ejecutará la función **last\_updated()** antes de completar la operación. Esta función actualiza automáticamente la columna `last_update` del registro con la fecha y hora actuales, permitiendo llevar un registro preciso de la última modificación de cada registro sin intervención manual.

Esta sentencia se utiliza para todas las tablas menos la tabla `payment`.

## 8. Construya un disparador que guarde en una nueva tabla creada por usted la fecha de cuando se insertó un nuevo registro en la tabla `film`.

```
1  -- Crear tabla de registro de eliminaciones
2  CREATE TABLE film_deletes_log (
3      film_id INTEGER PRIMARY KEY,
4      fecha_eliminacion TIMESTAMP NOT NULL
5  );
6
7  -- Crear función para el disparador
8  CREATE OR REPLACE FUNCTION log_film_delete()
9  RETURNS TRIGGER AS $$
10 BEGIN
11     INSERT INTO film_deletes_log (film_id, fecha_eliminacion)
12     VALUES (OLD.film_id, NOW());
13     RETURN OLD;
14 END;
15 $$ LANGUAGE plpgsql;
16
17
18 -- Crear disparador que llama a la función después de cada eliminacion en 'film'
19 CREATE TRIGGER after_film_delete
20 AFTER DELETE ON film
21 FOR EACH ROW
22 EXECUTE FUNCTION log_film_delete();
```

```

Alquiler=# INSERT INTO film (title, description, release_year, language_id, rental_duration, rental_rate, length, replacement_cost, rating, last_update)
VALUES
('Inception', 'A mind-bending thriller where a thief steals corporate secrets through the use of dream-sharing technology.', 2010, 1, 7, 4.99, 148, 19.99, 'PG-13', NOW()),
('The Matrix', 'A hacker discovers the truth about the simulated reality that controls humanity.', 1999, 1, 5, 3.99, 136, 14.99, 'R', NOW()),
('The Dark Knight', 'Batman faces off against the Joker, a criminal mastermind who wants to bring Gotham to its knees.', 2008, 1, 7, 5.99, 152, 24.99, 'PG-13', NOW()),
('Avatar', 'A paraplegic Marine dispatched to the moon Pandora on a unique mission becomes torn between following his orders and protecting a native species.', 2009, 2, 5, 6.99, 162, 29.99, 'PG-13', NOW()),
('The Shawshank Redemption', 'Two imprisoned men bond over a number of years, finding solace and eventual redemption through acts of common decency.', 1994, 1, 7, 4.49, 142, 19.99, 'R', NOW());
INSERT 0 5
Alquiler=# SELECT * FROM film_inserts_log;
 film_id |      fecha_insercion
-----+-----
 1001    | 2024-11-10 18:20:11.161567
 1002    | 2024-11-10 18:20:11.161567
 1003    | 2024-11-10 18:20:11.161567
 1004    | 2024-11-10 18:20:11.161567
 1005    | 2024-11-10 18:20:11.161567
(5 rows)

```

- Construya un disparador que guarde en una nueva tabla creada por usted la fecha de cuando se eliminó un registro en la tabla `film` y el identificador del `film`.

```

1  -- Crear tabla de registro de inserciones
2  CREATE TABLE IF NOT EXISTS film_inserts_log (
3      film_id INTEGER PRIMARY KEY,
4      fecha_insercion TIMESTAMP NOT NULL
5  );
6
7  -- Crear función para el disparador
8  CREATE OR REPLACE FUNCTION log_film_insert()
9  RETURNS TRIGGER AS $$
10 BEGIN
11     INSERT INTO film_inserts_log (film_id, fecha_insercion)
12     VALUES (NEW.film_id, NOW());
13     RETURN NEW;
14 END;
15 $$ LANGUAGE plpgsql;
16
17 -- Crear disparador que llama a la función después de cada inserción en 'film'
18 CREATE TRIGGER after_film_insert
19 AFTER INSERT ON film
20 FOR EACH ROW
21 EXECUTE FUNCTION log_film_insert();

```

```

Alquiler=# DELETE FROM film
WHERE title IN ('Inception', 'The Matrix', 'The Dark Knight', 'Avatar', 'The Shawshank Redemption');
DELETE 5
Alquiler=# SELECT * FROM film_deletes_log;
 film_id |      fecha_eliminacion
-----+-----
 1004    | 2024-11-10 18:21:19.153706
 1001    | 2024-11-10 18:21:19.153706
 1003    | 2024-11-10 18:21:19.153706
 1002    | 2024-11-10 18:21:19.153706
 1005    | 2024-11-10 18:21:19.153706
(5 rows)

```

## 10. Comente el significado y la relevancia de las secuencias.

Las secuencias se utilizan para generar valores únicos y consecutivos que pueden ser empleados como claves primarias en las tablas correspondientes. Siendo estos:

- customer\_customer\_id\_seq
- actor\_actor\_id\_seq
- address\_address\_id\_seq
- category\_category\_id\_seq
- city\_city\_id\_seq
- country\_country\_id\_seq
- inventory\_inventory\_id\_seq
- language\_language\_id\_seq
- payment\_payment\_id\_seq
- rental\_rental\_id\_seq
- staff\_staff\_id\_seq
- store\_store\_id\_seq

De esta manera, aseguran que las claves primarias sean únicas, lo cual es fundamental para mantener la integridad referencial en la base de datos, permitiendo además un rendimiento eficiente cuando se insertan nuevos registros. La configuración START WITH 1 asegura que cada secuencia comienza desde el valor 1, y INCREMENT BY 1 garantiza que cada nuevo valor sea un número consecutivo.