Objective:
Create a stored procedure that inserts rental data on the primary server, and verify that changes replicate to the standby server. Add a logging mechanism to track each operation.

Tasks to Complete:
Set up streaming replication (if not already done):
Primary on port 5432
Standby on port 5433

Create a table on the primary:

```
CREATE TABLE rental_log (
   log_id SERIAL PRIMARY KEY,
   rental_time TIMESTAMP,
   customer_id INT,
   film_id INT,
   amount NUMERIC,
   logged_on TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

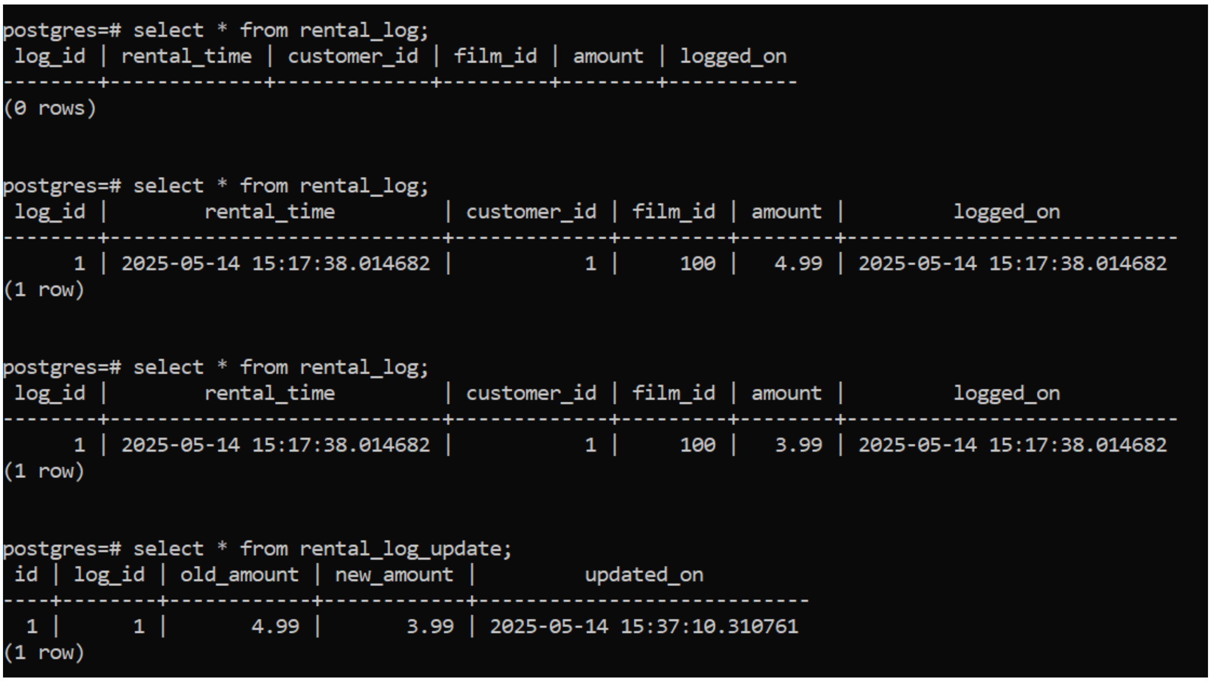
Ensure this table is replicated.
Write a stored procedure to:
Insert a new rental log entry
Accept customer_id, film_id, amount as inputs
Wrap logic in a transaction with error handling (BEGIN...EXCEPTION...END)

```
CREATE OR REPLACE PROCEDURE sp_add_rental_log(
   p_customer_id INT,
   p_film_id INT,
   p_amount NUMERIC
)
LANGUAGE plpgsql
AS $$
BEGIN
   INSERT INTO rental_log (rental_time, customer_id, film_id, amount)
   VALUES (CURRENT_TIMESTAMP, p_customer_id, p_film_id, p_amount);
EXCEPTION WHEN OTHERS THEN
   RAISE NOTICE 'Error occurred: %', SQLERRM;
END;
$$;
```

Call the procedure on the primary:

```
CALL sp_add_rental_log(1, 100, 4.99);
```

On the standby (port 5433):
Confirm that the new record appears in rental_log
Run:SELECT * FROM rental_log ORDER BY log_id DESC LIMIT 1;
Add a trigger to log any UPDATE to rental_log

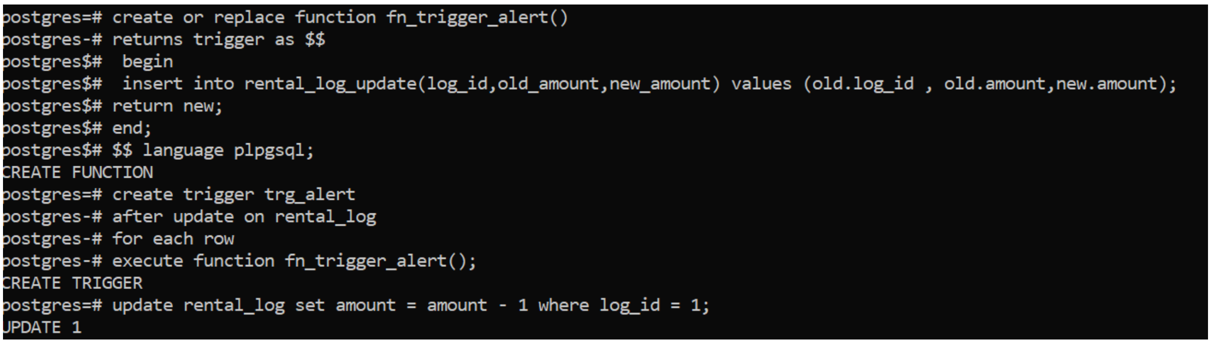


```
postgres=# select * from rental_log;
 log_id | rental_time | customer_id | film_id | amount | logged_on
--------+-------------+-------------+---------+--------+-----------
(0 rows)


postgres=# select * from rental_log;
 log_id |        rental_time         | customer_id | film_id | amount |         logged_on
--------+----------------------------+-------------+---------+--------+----------------------------
      1 | 2025-05-14 15:17:38.014682 |           1 |     100 |   4.99 | 2025-05-14 15:17:38.014682
(1 row)


postgres=# select * from rental_log;
 log_id |        rental_time         | customer_id | film_id | amount |         logged_on
--------+----------------------------+-------------+---------+--------+----------------------------
      1 | 2025-05-14 15:17:38.014682 |           1 |     100 |   3.99 | 2025-05-14 15:17:38.014682
(1 row)


postgres=# select * from rental_log_update;
 id | log_id | old_amount | new_amount |         updated_on
----+--------+------------+------------+----------------------------
  1 |      1 |       4.99 |       3.99 | 2025-05-14 15:37:10.310761
(1 row)
```



```
postgres=# create or replace function fn_trigger_alert()
postgres-# returns trigger as $$
postgres$#  begin
postgres$#  insert into rental_log_update(log_id,old_amount,new_amount) values (old.log_id , old.amount,new.amount);
postgres$# return new;
postgres$# end;
postgres$# $$ language plpgsql;
CREATE FUNCTION
postgres=# create trigger trg_alert
postgres-# after update on rental_log
postgres-# for each row
postgres-# execute function fn_trigger_alert();
CREATE TRIGGER
postgres=# update rental_log set amount = amount - 1 where log_id = 1;
UPDATE 1
```