Monica Nieckula
D191 Advanced Data Management
Performance Assessment

Business Report

Summarize one real-world business report that can be created from the attached Data Sets and Associated Dictionaries.

With employee turnover at an all-time high, the stakeholders at DVDbuster, have decided to start rewarding their staff members who sold the highest number of rentals with a $200 quarterly bonus. By rewarding productivity, the stakeholders expect to see profits increase and employee turnover decrease. The stakeholders have requested that their current data be used to discover which employee provides the highest revenue for DVDbuster.

Describe the data used for the report.

The DVD Rental database contains the data necessary for this report. Using the Payment and Staff tables, I can accurately share which staff member should be rewarded with the stakeholders. First, I will review the data for the payments made (Payment table) and then compare them with the data in the Staff table. The data pulled from the Payment table will include the payment_id (a unique integer assigned to each payment) and the payment_date (a timestamp stating when the payment was made) columns. The data pulled from the Staff table will include the staff_id (a unique integer assigned to each staff member), first_name (a varying character data type with a maximum length, or amount of characters, of 25), last_name (a varying character data type with a maximum length, or amount of characters, of 25), and the email (a varying character data type with a maximum length, or amount of characters, of 100) columns. The varying character data types allow for longer staff member first name, last name, and email inputs into their designated columns. The two tables are related via their staff_id attributes. For example, a row in our new Summary table might have the following inputs: a staff_id of 1, a full_name consisting of the combined data from the first_name and last_name columns (from the Detailed table) of Mike Hillyer, and a total_sales output of 7292. An example of the data that appears inside our Detailed table includes a payment_id of 17504, staff_id of 1, first_name of Mike, last_name of Hillyer, email of mike.hillyer@sakilastaff.com, and a payment_date of 2007-02-16 17:23:14.996577. In this instance, payment_date column's date is February 2nd, 2007 at 5:23pm and 14.9 seconds.

Identify *two* or more specific tables from the given dataset that will provide the data necessary for the detailed and summary sections of the report.

The Payment and Staff tables are the only tables needed to generate this report. The Payment table contains specific data pertaining to the payments processed by the staff members and the customers who made the payments. The Payment table does not provide information about the staff members who processed the payments-for this reason; I will need to use the Staff table to connect the staff member and the payment they had processed. The two tables are connected via their 'staff_id' attribute.

Identify the specific fields that will be included in the detailed and summary sections of the report.

Within the detailed report, stakeholders will find the staff member's name, email, payment IDs associated with their staff ID, the payment date of said payment ids, and the staff member's ID. The summary report will only include the staff member's name, staff ID, and the number of payments they have processed.

Identify *one* field in the detailed section that will require a custom transformation and explain why it should be transformed. For example, you might translate a field with a value of 'N' to 'No' and 'Y' to 'Yes.'

For readability, two fields will need to be transformed. I will concatenate the first and last names into a column labeled 'full_name'. The number of payments will also be counted and grouped by staff ID to see who processed the most payments.

Explain the different business uses of the detailed and summary sections of the report.

There are a few different business uses that come to mind when viewing the summary and detailed reports. The summary report can be used to see which employee is most productive or a better salesperson when renting DVDs. The detailed table can be analyzed deeper by seeing a list of exactly which payments were processed by each employee. If a customer has an issue with a payment, the detailed table will be able to tell the manager precisely who processed the payment without needing to look up the staff member's name and email.

Explain how frequently your report should be refreshed to remain relevant to stakeholders.

This report should be updated quarterly to reflect recent sales and determine which staff member has earned the monthly sales reward. The report should be updated before offering a bonus to the staff member with the most payments processed.

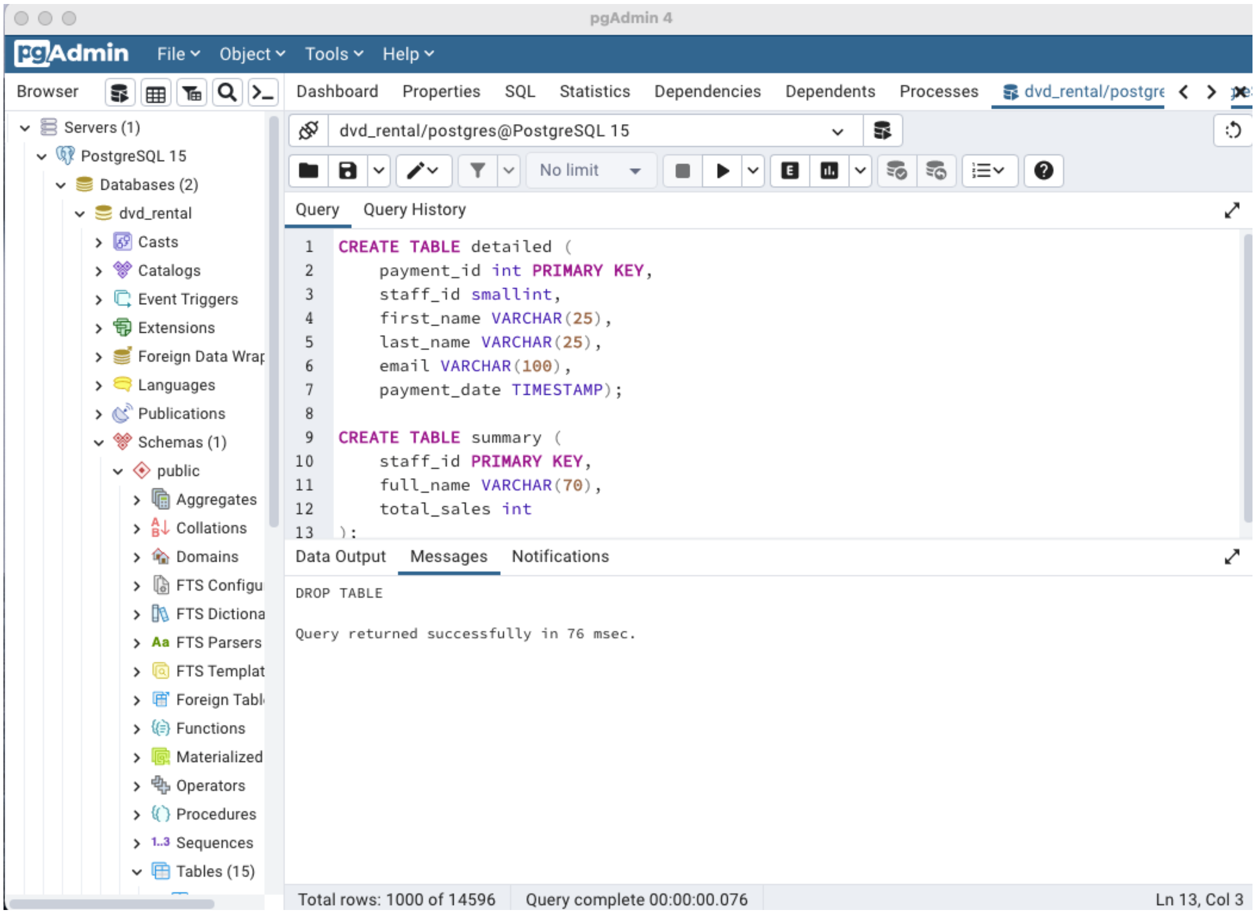
Explain how the stored procedure can be run on a schedule to ensure data freshness.

Anytime the detailed and summary table data is required, which should be quarterly, the stored procedure can be executed. The old data will remain in the report if it isn't re-run, which will cause the company to have an inaccurate picture of which staff member is processing the most payments. The stored procedure to update the data for the detail and the summary tables must be carried out before the quarterly business meeting to guarantee the freshness of data. The stored procedure, renew(), can be scheduled with an external scheduling tool, such as psAgent.

Report Queries

Creating Tables

The SQL code used to create the tables to hold the report sections.



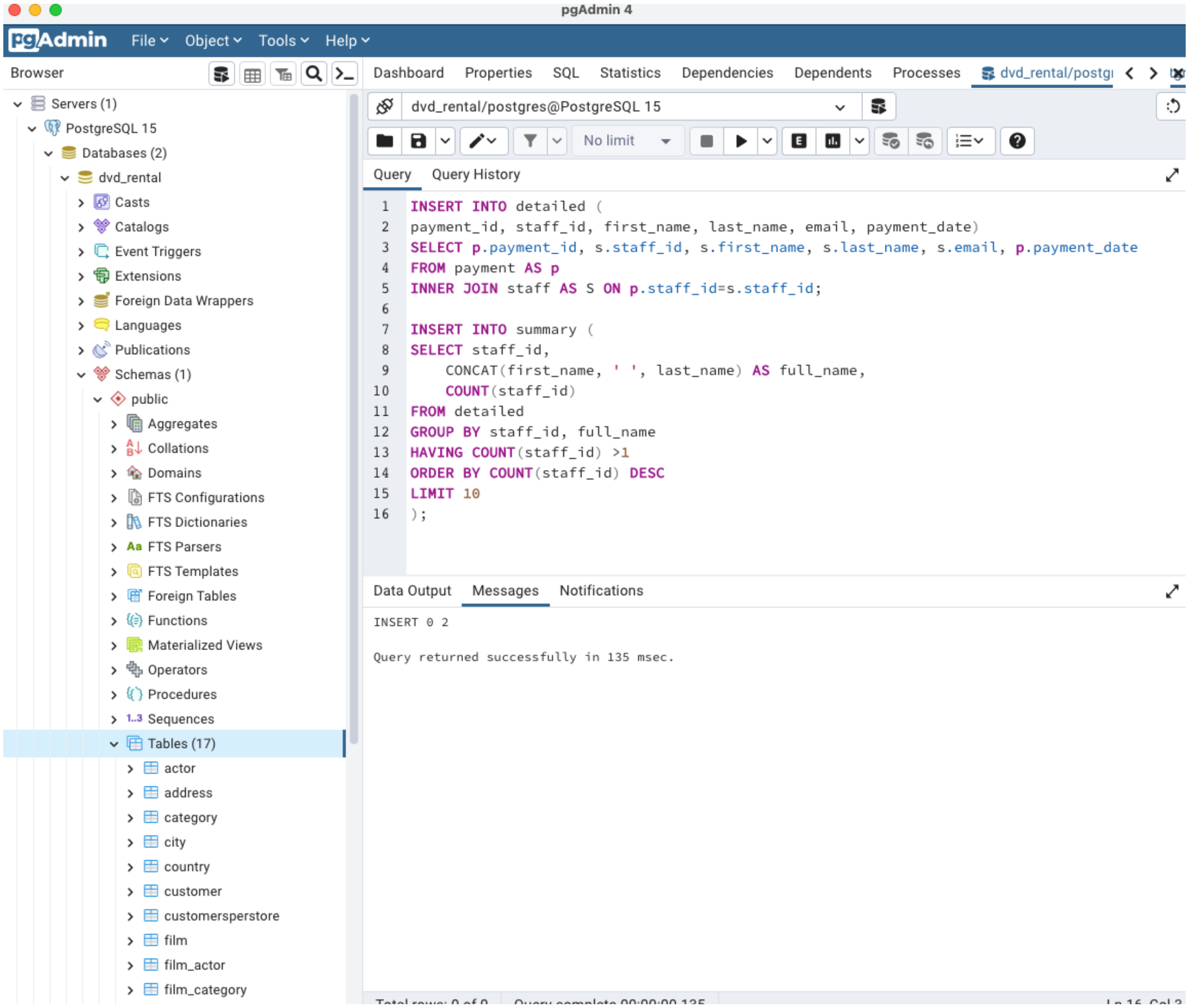
```sql
CREATE TABLE detailed (
    payment_id int PRIMARY KEY,
    staff_id smallint,
    first_name VARCHAR(25),
    last_name VARCHAR(25),
    email VARCHAR(100),
    payment_date TIMESTAMP);

CREATE TABLE summary (
    staff_id PRIMARY KEY,
    full_name VARCHAR(70),
    total_sales int
);
```

```
DROP TABLE

Query returned successfully in 76 msec.
```

Total rows: 1000 of 14596   Query complete 00:00:00.076   Ln 13, Col 3

SQL Query

The SQL queries below used raw data from the source database to provide the Detailed section of the report.

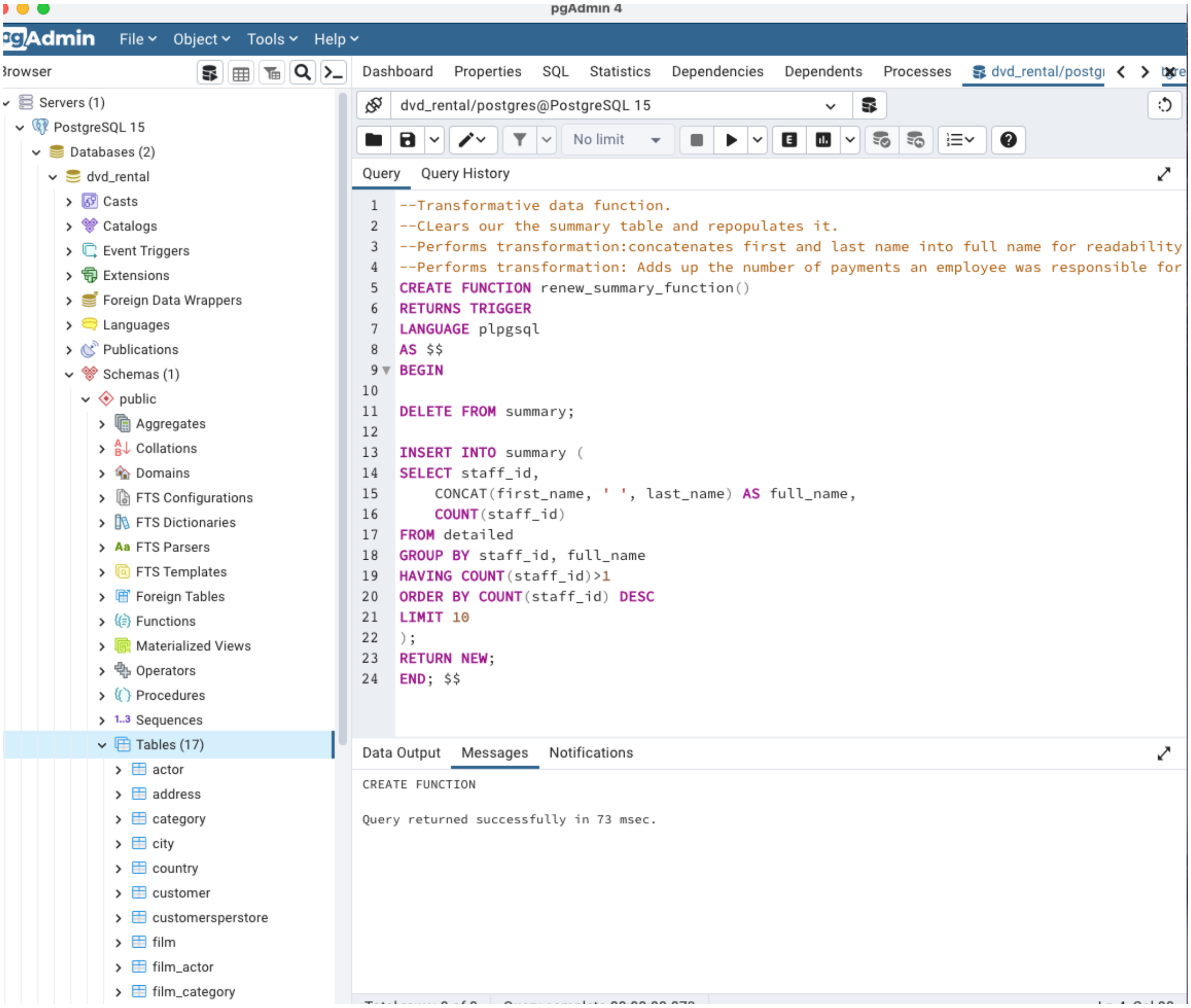


```sql
INSERT INTO detailed (
payment_id, staff_id, first_name, last_name, email, payment_date)
SELECT p.payment_id, s.staff_id, s.first_name, s.last_name, s.email, p.payment_date
FROM payment AS p
INNER JOIN staff AS S ON p.staff_id=s.staff_id;

INSERT INTO summary (
SELECT staff_id,
    CONCAT(first_name, ' ', last_name) AS full_name,
    COUNT(staff_id)
FROM detailed
GROUP BY staff_id, full_name
HAVING COUNT(staff_id) >1
ORDER BY COUNT(staff_id) DESC
LIMIT 10
);
```

```
INSERT 0 2

Query returned successfully in 135 msec.
```

Functions

The SQL code used for the transformations necessary to make the data easier to read for the stakeholders.

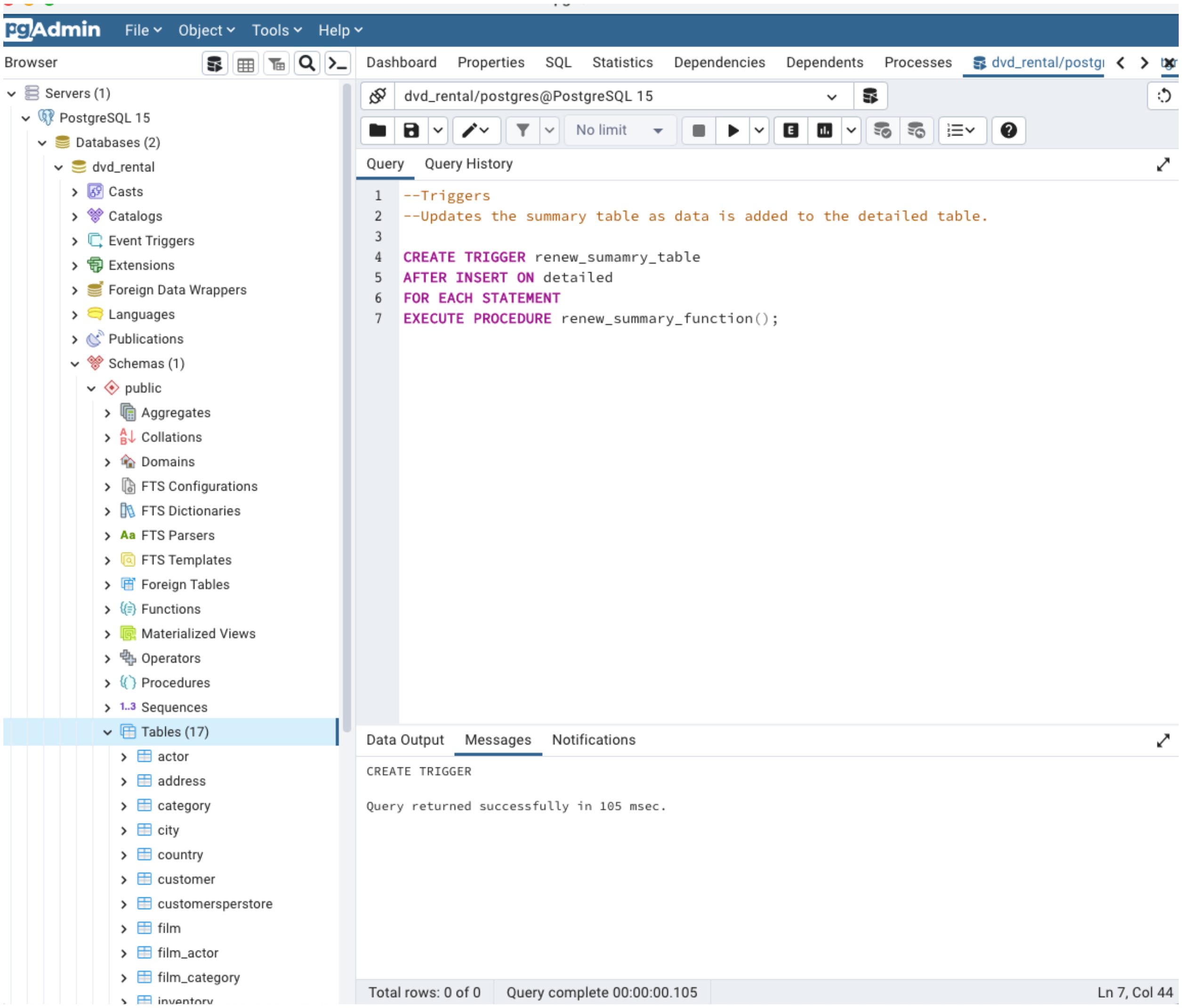


```
--Transformative data function.
--CLears our the summary table and repopulates it.
--Performs transformation:concatenates first and last name into full name for readability
--Performs transformation: Adds up the number of payments an employee was responsible for
CREATE FUNCTION renew_summary_function()
RETURNS TRIGGER
LANGUAGE plpgsql
AS $$
BEGIN

DELETE FROM summary;

INSERT INTO summary (
SELECT staff_id,
    CONCAT(first_name, ' ', last_name) AS full_name,
    COUNT(staff_id)
FROM detailed
GROUP BY staff_id, full_name
HAVING COUNT(staff_id)>1
ORDER BY COUNT(staff_id) DESC
LIMIT 10
);
RETURN NEW;
END; $$
```

```
CREATE FUNCTION

Query returned successfully in 73 msec.
```

Triggers

The SQL code that creates a trigger that will continually update the summary table as data is added to the detailed table.



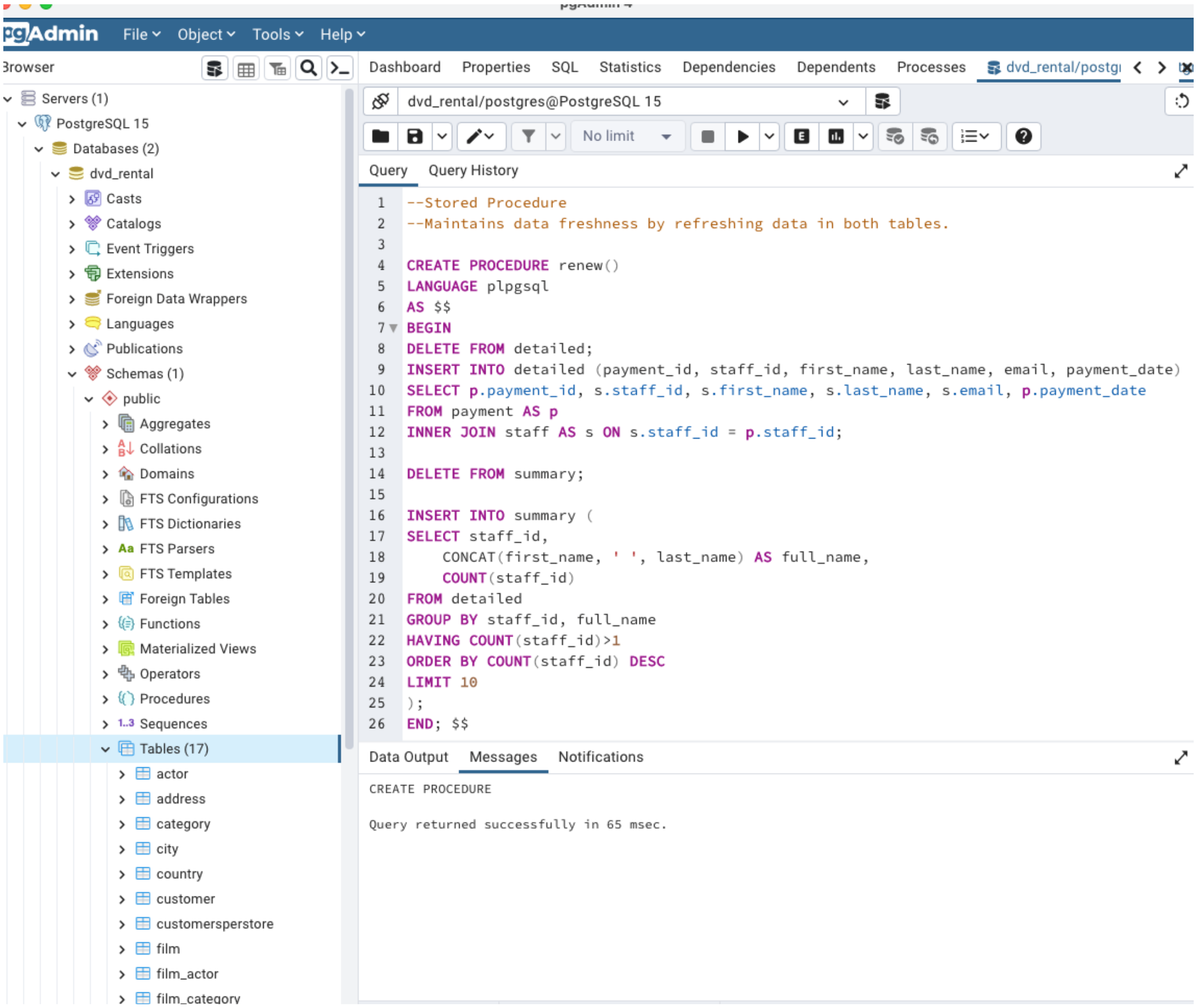
```
--Triggers
--Updates the summary table as data is added to the detailed table.

CREATE TRIGGER renew_sumamry_table
AFTER INSERT ON detailed
FOR EACH STATEMENT
EXECUTE PROCEDURE renew_summary_function();
```

```
CREATE TRIGGER

Query returned successfully in 105 msec.
```

Total rows: 0 of 0 Query complete 00:00:00.105 Ln 7, Col 44

Stored Procedure

The stored procedure refreshes the data in the detailed and summary tables. It can clear the contents of the two tables and performs an ETL load process.



```
--Stored Procedure
--Maintains data freshness by refreshing data in both tables.

CREATE PROCEDURE renew()
LANGUAGE plpgsql
AS $$
BEGIN
DELETE FROM detailed;
INSERT INTO detailed (payment_id, staff_id, first_name, last_name, email, payment_date)
SELECT p.payment_id, s.staff_id, s.first_name, s.last_name, s.email, p.payment_date
FROM payment AS p
INNER JOIN staff AS s ON s.staff_id = p.staff_id;

DELETE FROM summary;

INSERT INTO summary (
SELECT staff_id,
    CONCAT(first_name, ' ', last_name) AS full_name,
    COUNT(staff_id)
FROM detailed
GROUP BY staff_id, full_name
HAVING COUNT(staff_id)>1
ORDER BY COUNT(staff_id) DESC
LIMIT 10
);
END; $$
```

```
CREATE PROCEDURE

Query returned successfully in 65 msec.
```

Web Sources

I did not obtain data or pieces of third-party code from any external websites or sources.