D191: Automating data integration
What are the sales registered by our stores for the first quarter of the year 2007?

Jackelinne Salazar Coutino
Student ID: 1130922

November 14, 2021

BUSINESS QUESTION

What are the total sales reported by ours stores for the first quarter of the year 2007?

BUSINESS REPORT

A1. Description of the data used for this report

The data used for this report is extracted from the DVD Rental Database on Labs on Demand. "The DVD rental Database represents the business processes of a DVD rental store." (PostgreSQL Tutorial, n.d)
The DVD rental database has the following 15 tables:

- actor – stores actors data including first name and last name.
- film – stores film data such as title, release year, length, rating, etc.
- film_actor – stores the relationships between films and actors.
- category – stores film's categories data.
- film_category- stores the relationships between films and categories.
- store – contains the store data including manager staff and address.
- inventory – stores inventory data.
- rental – stores rental data.
- payment – stores customer's payments.
- staff – stores staff data.
- customer – stores customer data.
- address – stores address data for staff and customers
- city – stores city names.
- country – stores country names.

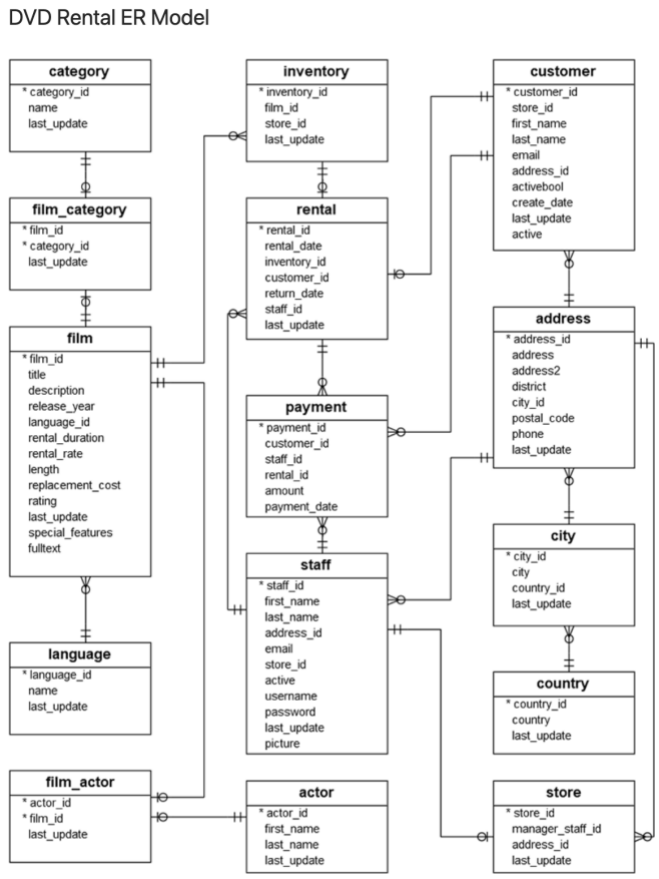

Figure 1. DVD Rental ER diagram (PostgreSQL Tutorial, n.d)

A2. Detailed table and summary table

This report consists of two tables that will provide us with an answer to the business question. The detailed table contains every record that is relevant to the business question. Furthermore, the rental table, payment table and staff table are used to create the detailed table. Within the SQL code the detailed table is identified as “quarter_sales”.

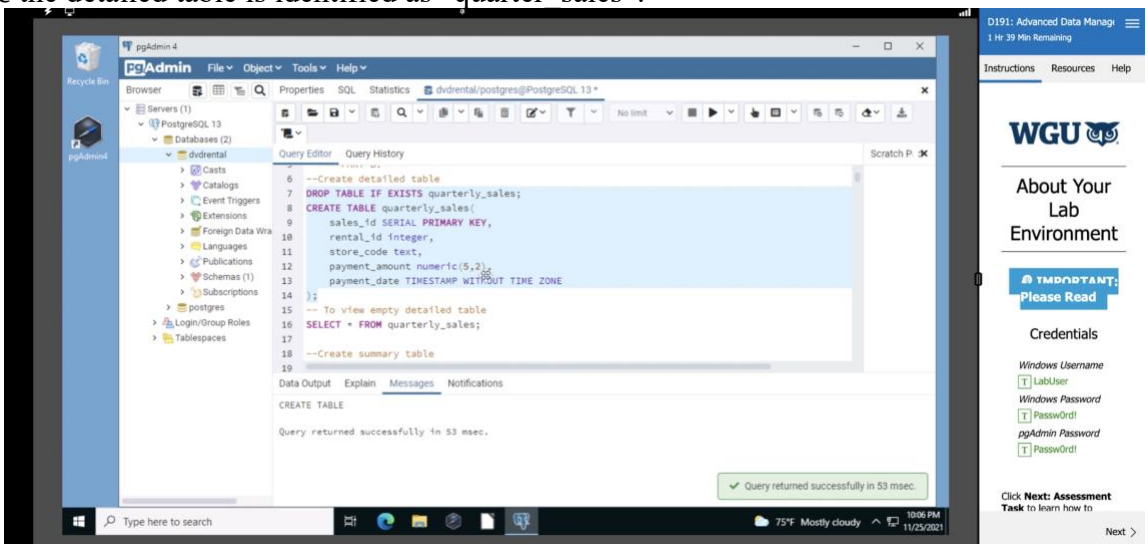

Figure 2. SQL code to create the detailed table.

The summary table contains the total retail sales of each store of the DVD rental database. This is the result of aggregating the data of the detailed table. This table is called “summary_sales”. The “summary_sales” table depicts the sales performance of each store during the first quarter of the year.

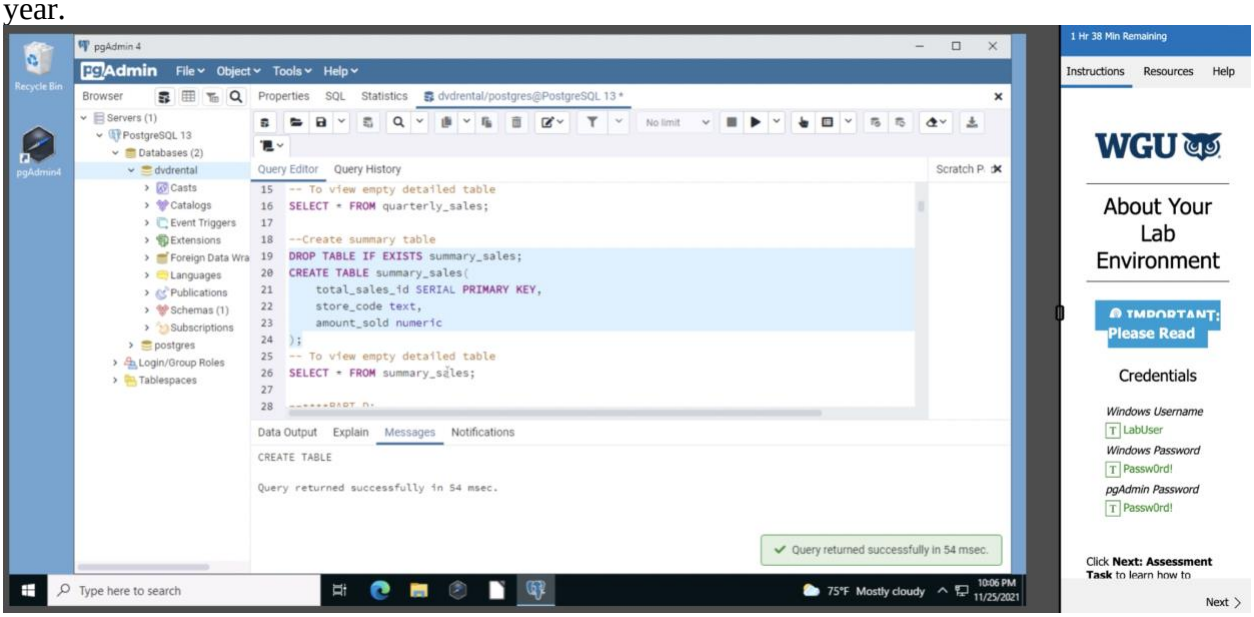

Figure 3. SQL code to create the summary table.

A3. Fields included in the tables for this report

The fields that meet the criteria asked in the business question: “What are the total sales reported by ours stores for the first quarter of the year 2007?” to create the detailed table are the following:

- sales_id: Primary key of the detailed table.
- Retail_id: Rental ID extracted from the rental table.
- Store_code: Store number.
- Amount: Price of the movie rented.
- Payment_date: Date the payment was effectively made.

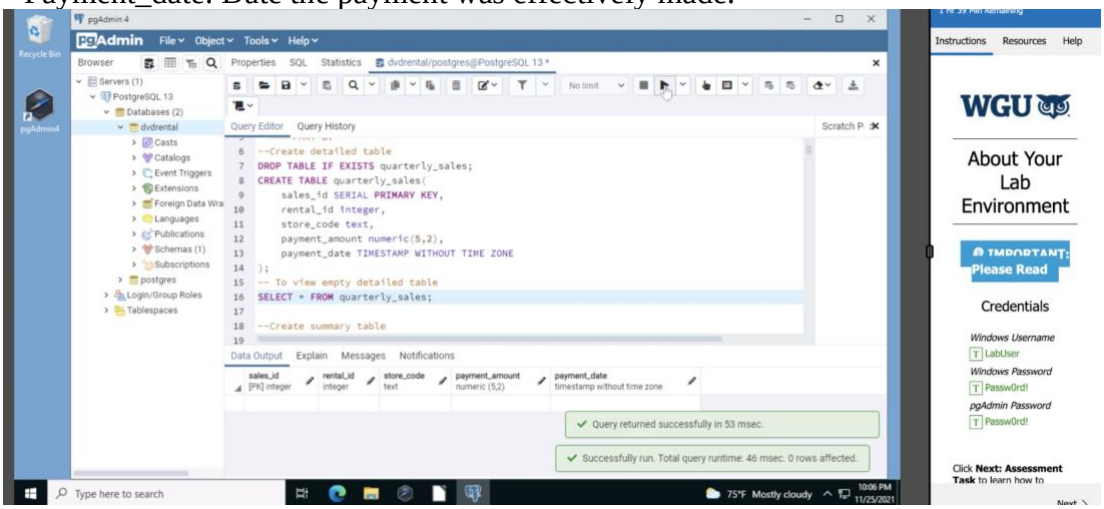

Figure 4. SQL code to depict all the columns in the detailed table.

The fields that are included in the summary table are the following:

- Total_sales_Id: Primary key of the summary table.
- Store_code: Store number.
- Amount_sold: Sum of all the movies rented for each individual store during the first quarter of the year.

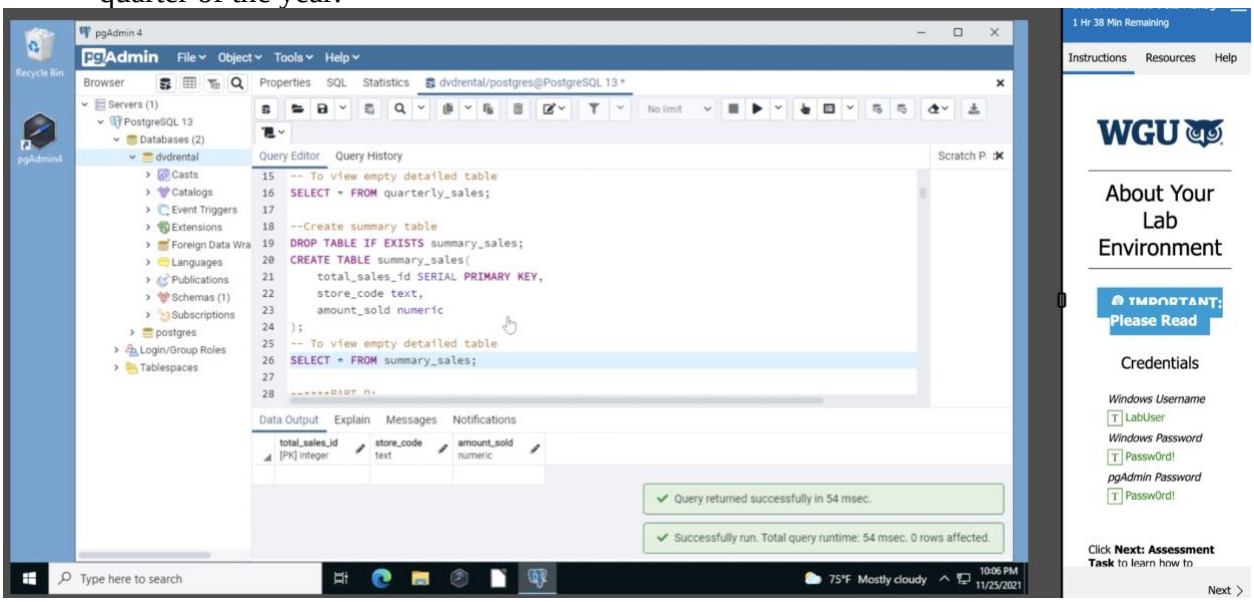

Figure 5. SQL code to depict all the columns in the summary table.

A4. Custom transformation

A function called "get_store_code(smallint)" was created. Its functionality is to perform a transformation on the field "store_code". This function receives a parameter type smallint named current_staff_id.

Considering that this report's goal is to depict the total retail sales of each store in the DVD rental database, this translation permits an understandable way to read the store_id of both the detailed and summary table. For example, the store_id with value 1 will be displayed as "Store # 1".

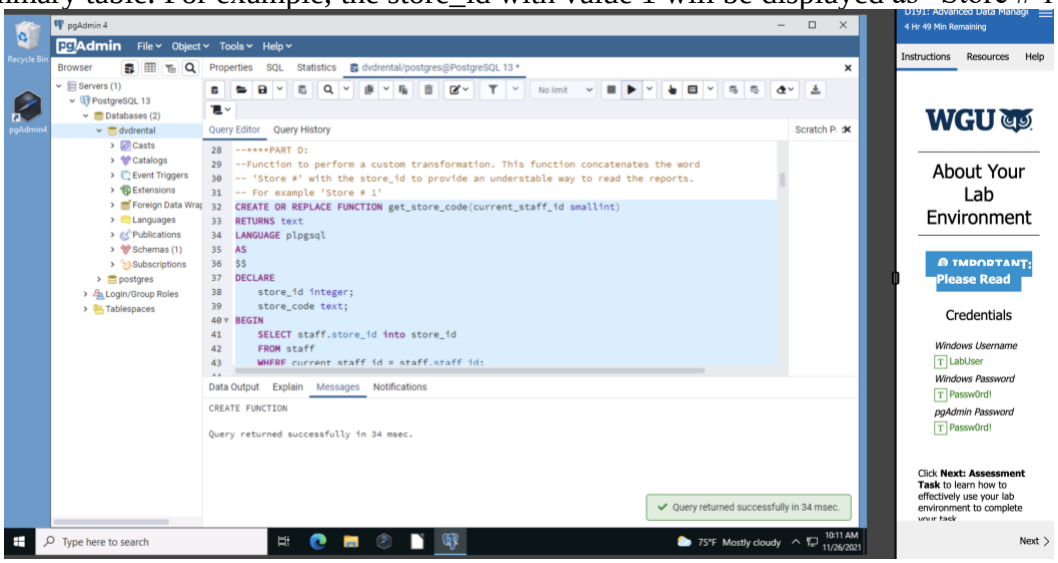

Figure 6. This figure depicts the get_store_code() function run successfully.

A5. Business uses of the tables used in this report

The detailed table of this report could be also used to analyze the number of movies rented by month within the quarter. This result will give us which month has the largest and smallest rental demand. With this information we could identified how the market is behaving in that period. For instance, if a holiday has an influence in people to rent a movie, we could increase our profit margin by improving our inventory.

Furthermore, the summary table could also be used to obtain the total retail sales registered in the first quarter of the year 2007. With this information, we could set accountable goals for the following quarter to increase our average order value.

A6. How frequently should the report be refreshed?

The objective of this report is to depict the quarterly retail sales of each store in the DVD rentals database. Since this report will be run after each quarter of the year 2007, I will recommend executing this action at the end of each month to have updated data. Furthermore, it will provide us with an overview of our sales in case we need to meet an accountable goal by the following quarter.

A stored procedure was created with the intention to clear the contents of the detailed and summary tables to meet this requirement. The stored procedure is called "refresh_reports(quarter int)". This procedure receives a parameter that can be any number between one and four. Each number depicts the any of the four quarters of the year.

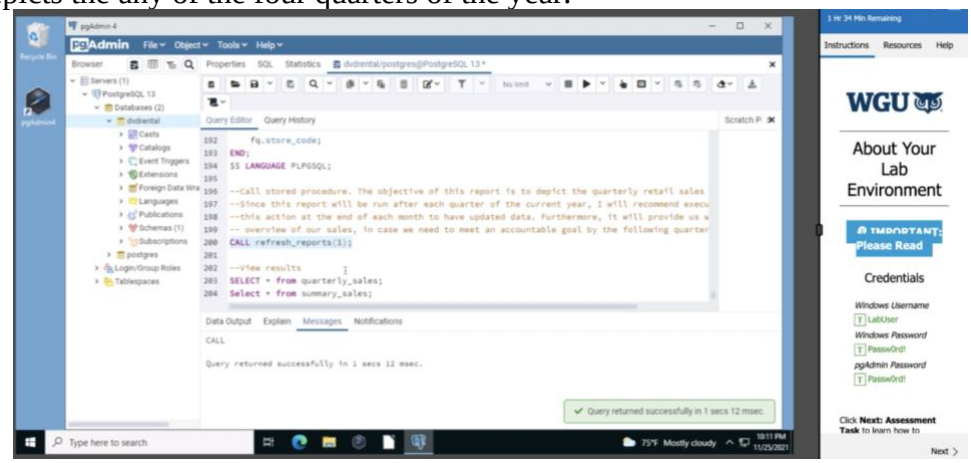

Figure 7. This figure depicts the query to run the function refresh_reports().

SQL CODE

This section contains a copy of the SQL Code created to build this business report.

```
--****PART A:
--Business question: What are the sales registered by our stores for
--the first quarter of the year 2007?

--****PART B:
--Create detailed table
```

```
DROP TABLE IF EXISTS quarterly_sales;
CREATE TABLE quarterly_sales(
        sales_id SERIAL PRIMARY KEY,
        rental_id integer,
        store_code text,
        payment_amount numeric(5,2),
        payment_date TIMESTAMP WITHOUT TIME ZONE
);
-- To view empty detailed table
SELECT * FROM quarterly_sales;

--Create summary table
DROP TABLE IF EXISTS summary_sales;
CREATE TABLE summary_sales(
        total_sales_id SERIAL PRIMARY KEY,
        store_code text,
        amount_sold numeric
);
-- To view empty detailed table
SELECT * FROM summary_sales;

--****PART D:
--Function to perform a custom transformation. This function concatenates the word
-- 'Store #' with the store_id to provide an understable way to read the reports.
-- For example 'Store # 1'
CREATE OR REPLACE FUNCTION get_store_code(current_staff_id smallint)
RETURNS text
LANGUAGE plpgsql
AS
$$
        DECLARE
        store_id integer;
        store_code text;
        BEGIN
        SELECT staff.store_id into store_id FROM staff WHERE current_staff_id =
staff.staff_id;
        SELECT CONCAT('Store # ', store_id) into store_code;
RETURN store_code;
END;
$$;

--****PART C:
--Extract raw data from dvdrental db into detailed table
INSERT INTO quarterly_sales(
        rental_id,
        store_code,
```

```
        payment_amount,
        payment_date
)
SELECT
        r.rental_id, store_code,pmt.amount, pmt.payment_date
FROM
        rental AS r INNER JOIN payment AS pmt USING(rental_id),
LATERAL get_store_code(pmt.staff_id) store_code
WHERE (SELECT EXTRACT (QUARTER FROM payment_date) = 1);

-- To view contents of detailed table
SELECT * FROM quarterly_sales;

--Evaluate data accuracy.
--      Extracts the number of movies rented for the first quarter of 2007.
--      The number of rows of the detailed table should be the same as the result of
--      this statement. (7660 rows).
SELECT
        COUNT (rental_id)
FROM
        rental INNER JOIN payment USING(rental_id) WHERE (SELECT EXTRACT
(QUARTER FROM payment_date) = 1);

--Compare the result with the detailed table
SELECT COUNT(*) FROM quarterly_sales;

--Insert values to the summary table (with aggregation)
INSERT INTO summary_sales(
        store_code,
        amount_sold
)
SELECT
        q.store_code, SUM(q.payment_amount) AS total
FROM
        quarterly_sales q
GROUP BY
        q.store_code ;

-- To view contents of summary table
SELECT * FROM summary_sales;

--Evaluate data accuracy.
--      Extracts the total retail sales for the first quarter of 2007 grouped by store.
--      The result of this statement should be the same as the summary table.
SELECT
        store_id, SUM(amount)
```

```
FROM
	payment INNER JOIN staff USING(staff_id) WHERE (SELECT EXTRACT
(QUARTER FROM payment_date) = 1)
GROUP BY
	store_id;

--****PART E:
--Create function to update the summary table when data is added to the detailed table.
CREATE OR REPLACE FUNCTION sales_summary_update()
RETURNS TRIGGER AS $sales_summary_update$
DECLARE
	new_amount_cost numeric(15,2);
	new_store_code text;
BEGIN
	new_amount_cost = NEW.payment_amount;
	new_store_code = NEW.store_code;
	--Update the summary row with the new values
	UPDATE summary_sales
	SET amount_sold = amount_sold + new_amount_cost
	WHERE store_code = new_store_code;
	RETURN NULL;
END;
$sales_summary_update$ LANGUAGE PLPGSQL;

-- Create trigger on detailed table
CREATE OR REPLACE TRIGGER sales_summary_update
AFTER INSERT ON quarterly_sales
FOR EACH ROW EXECUTE FUNCTION sales_summary_update();

-- To view contents of summary table before the update trigger
SELECT * FROM summary_sales;
--Test trigger functionality.
--Insert 2 new rows to modify the total sales on the summary table.
INSERT INTO quarterly_sales(
	rental_id,
	store_code,
	payment_amount,
	payment_date
)
VALUES(
	5052, 'Store # 1', 3.59, LOCALTIMESTAMP
);

INSERT INTO quarterly_sales(
	rental_id,
	store_code,
```

```
        payment_amount,
        payment_date
)
VALUES(
        5051, 'Store # 2', 7.99, LOCALTIMESTAMP
);

-- To view contents of summary table after creating the update trigger
SELECT * FROM summary_sales;

--****PART F:
--Create stored procedure to refresh data in both detailed and summary tables.
CREATE OR REPLACE PROCEDURE refresh_reports(quarter int)
AS $$
BEGIN
        --Refresh detailed table
        DELETE FROM quarterly_sales;
        INSERT INTO quarterly_sales(
                rental_id,
                store_code,
                payment_amount,
                payment_date
        )
        SELECT
                r.rental_id, store_code, pmt.amount, pmt.payment_date
        FROM
                rental AS r INNER JOIN payment AS pmt USING(rental_id),
                LATERAL get_store_code(pmt.staff_id) store_code
        WHERE
                (SELECT EXTRACT (QUARTER FROM payment_date) = quarter);
        --Refresh summary table
        DELETE FROM summary_sales;
        INSERT INTO summary_sales(
                store_code,
                amount_sold
        )
        SELECT
                fq.store_code, SUM(fq.payment_amount) AS Total
        FROM
                quarterly_sales fq
        GROUP BY
                fq.store_code;
END;
$$ LANGUAGE PLPGSQL;
```

```
--Call stored procedure. The objective of this report is to depict the quarterly retail sales of DVD rentals.
--Since this report will be run after each quarter of the current year, I will recommend to execute
--this action at the end of each month to have updated data. Furthermore, it will provide us with a
-- overview of our sales, in case we need to meet an accountable goal by the following quarter.
CALL refresh_reports(1);

--View results
SELECT * from quarterly_sales;
Select * from summary_sales;
```

PRE-ASSESSMENT RECORDING

This section contains a link to the Panopto video I created to demonstrate the functionality of the code used for the analysis and summary of the programming environment.
https://d2y36twrtb17ty.cloudfront.net/sessions/1b860e08-db78-48a7-ab57-adec0032f9c1/a3685ee6-eed3-4166-b595-adec0032f9ca-f647e59d-3ec3-451e-bc6c-adec0035b4a1.mp4?invocationId=f1f2fbd7-d14e-ec11-8289-12b1cb861383

REFERENCE LIST

PostgreSQL Tutorial. (n.d). PostgreSQL sample Database.
https://www.postgresqltutorial.com/postgresql-sample-database/