

## MSDE 631: Lab 4

Etana Disasa

### Part 1: Install PostgreSQL

#### Pre Installation Summary

Installation Directory: /Library/PostgreSQL/10

Server Installation Directory: /Library/PostgreSQL/10

Data Directory: /Library/PostgreSQL/10/data

Database Port: 5433

Database Superuser: postgres

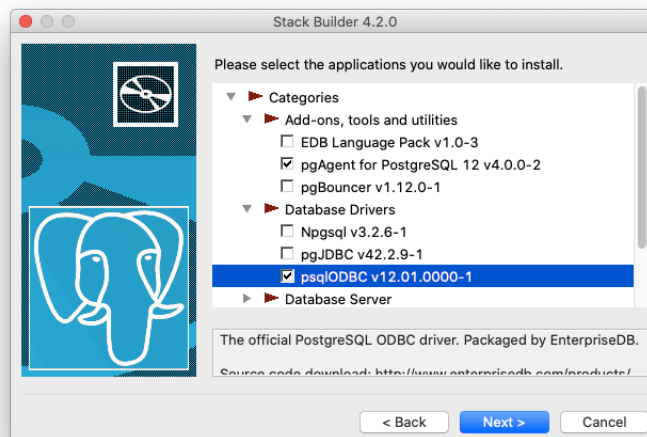
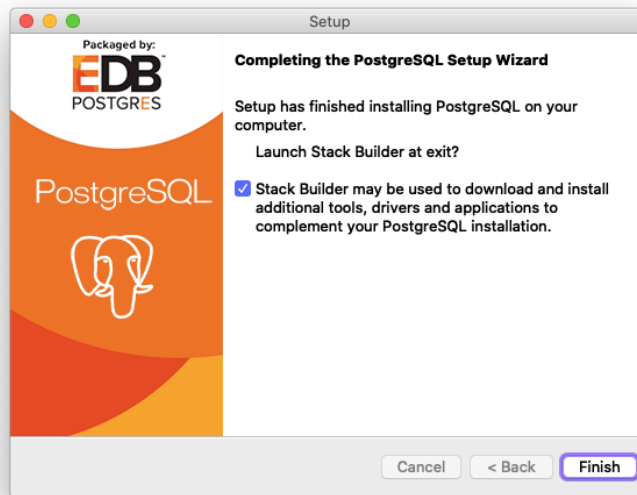
Operating System Account: postgres

Database Service: postgresql-10

Command Line Tools Installation Directory: /Library/PostgreSQL/10

pgAdmin4 Installation Directory: /Library/PostgreSQL/10/pgAdmin 4

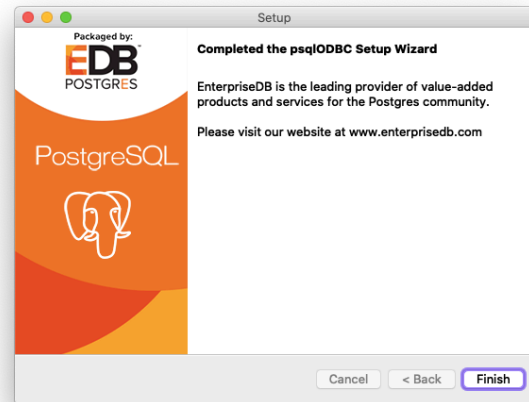
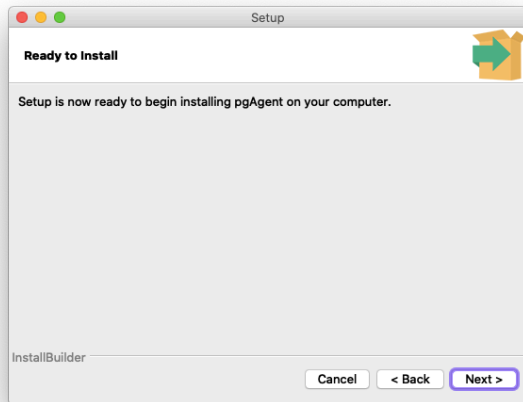
Stack Builder Installation Directory: /Library/PostgreSQL/10



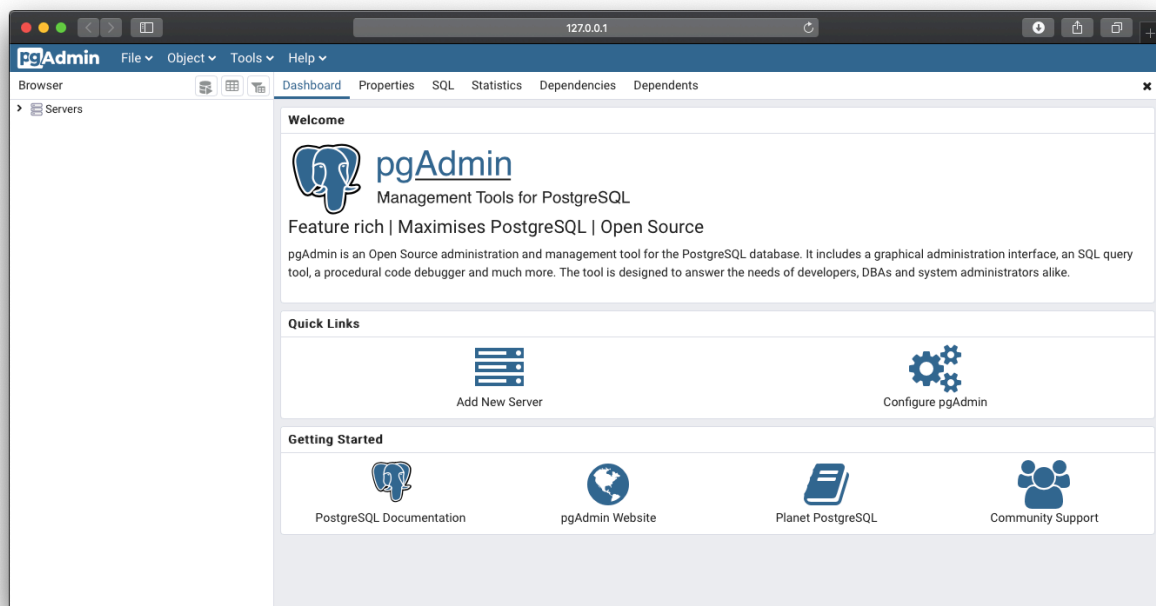
## MSDE 631: Lab 4

Etana Disasa

During this installation, pgAgent and psqLODBC failed to install. I had individually run the installation files from where Stack Builder had downloaded them for the installation.



And now pgAdmin 4 is up and running.



END OF PART ONE

## MSDE 631: Lab 4

Etana Disasa

### Part 2: How to Use PostgreSQL

SQL Shell (psql)

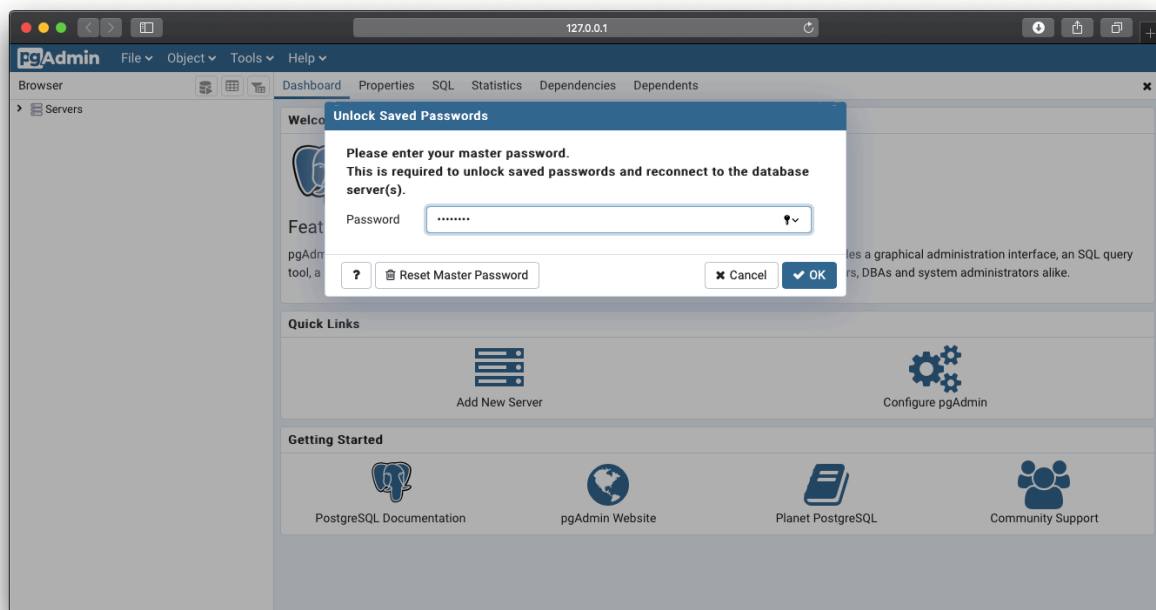
```
msds — more · runpsql.sh — 124x22
Last login: Tue Apr 7 12:58:56 on ttys000

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) ETANAmac:~ msds$ /Library/PostgreSQL/12/scripts/runpsql.sh; exit
Server [localhost]:
Database [postgres]:
Port [5432]:
Username [postgres]:
[Password for user postgres:
psql (12.2)
Type "help" for help.

[postgres=# SELECT VERSION();

              version
-----
 PostgreSQL 12.2 on x86_64-apple-darwin, compiled by Apple LLVM version 6.0 (clang-600.0.54) (based on LLVM 3.5svn), 64-bit
(1 row)

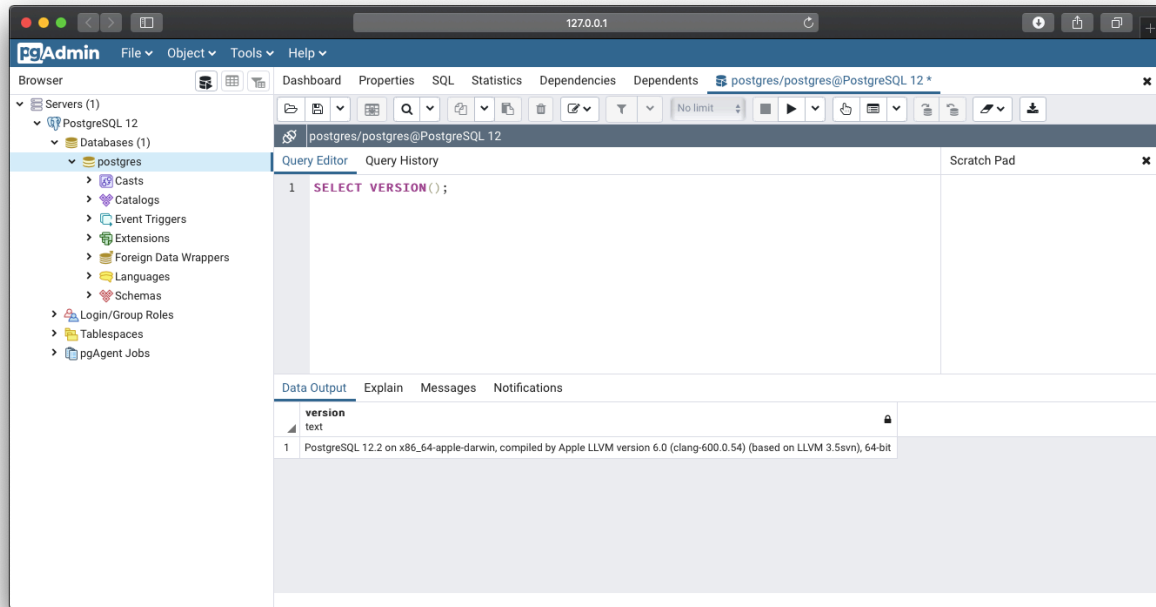
[END]
```



## MSDE 631: Lab 4

Etana Disasa

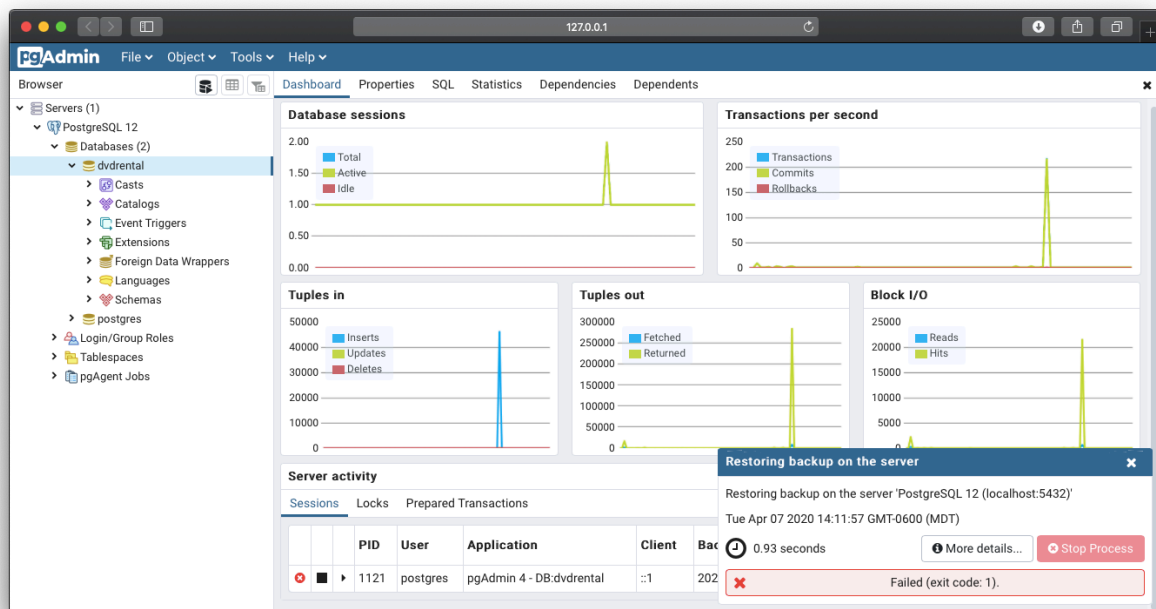
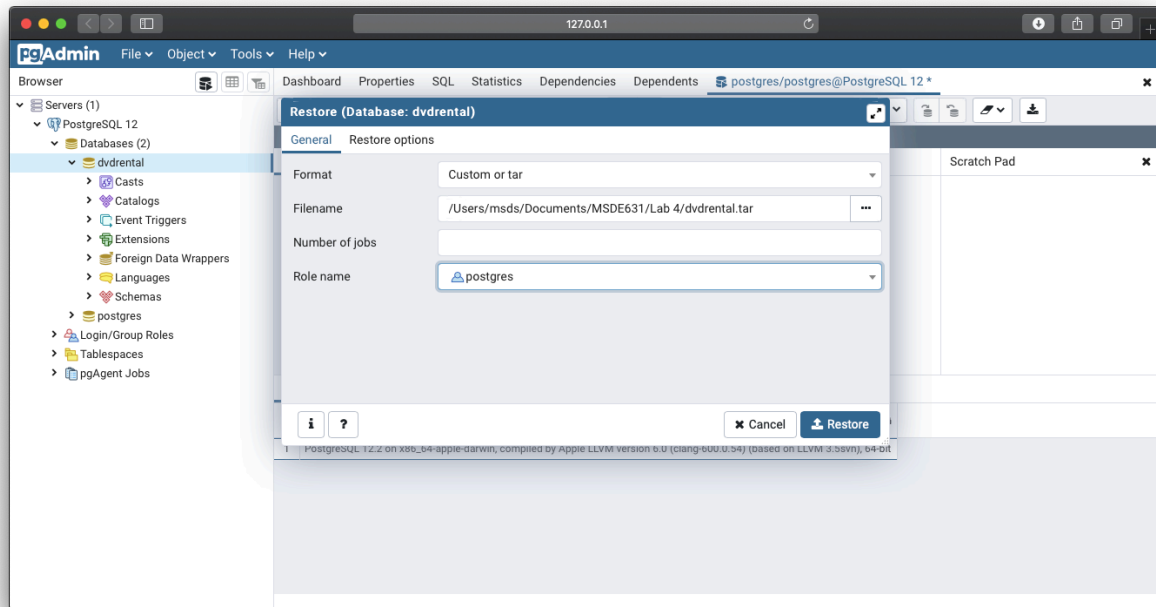
In this new version the ⚡ (lightening bolt) is not changed into a traditional ▶ (execute) button.



## MSDE 631: Lab 4

Etana Disasa

### Creating and Restoring Database (dvdrental)



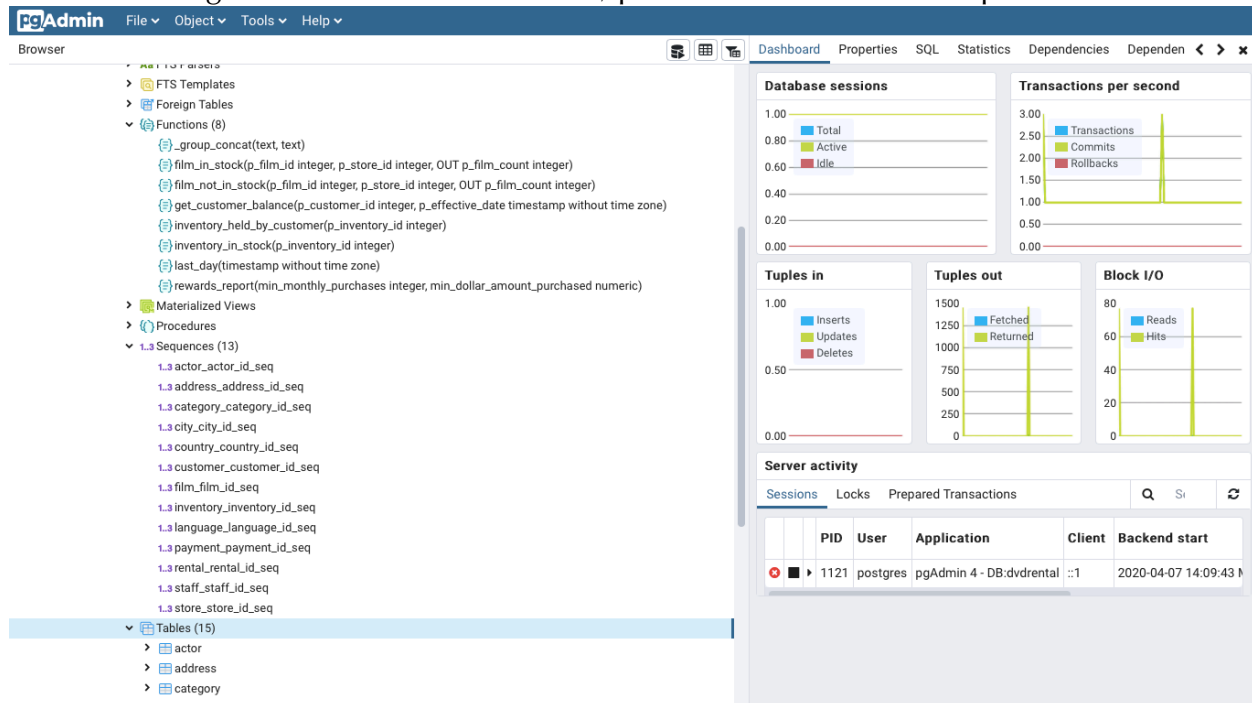
My installation was giving me problems. It appears that this exit code-1 error appears if the restore process is attempting to add an IPS\_name to the Schema. "If the IPS\_username is already configured in the Schema or is different from the IPS\_username in the old PostgreSQL database, it will return exit code 1. This error usually can be ignored."<sup>1</sup>

<sup>1</sup> <https://community.ipswitch.com/s/article/Exit-Code-1-When-Restoring-PostgreSQL-Database?r=6&ui-force-components-controllers-recordGlobalValueProvider.RecordGvp.getRecord=1>

## MSDE 631: Lab 4

Etana Disasa

Below the image shows that all the schema, procedures and tables are present.



## Writing and running queries

dvdrental/postgres@PostgreSQL 12

Query Editor

Query History

Scratch Pad

1

SELECT \* FROM actor;

Data Output

Explain

Messages


Notifications

	actor_id [PK] integer	first_name character varying (45)	last_name character varying (45)	last_update timestamp without time zone
1	1	Penelope	Guinness	2013-05-26 14:47:57.62
2	2	Nick	Wahlberg	2013-05-26 14:47:57.62
3	3	Ed	Chase	2013-05-26 14:47:57.62
4	4	Jennifer	Davis	2013-05-26 14:47:57.62
5	5	Johnny	Lollobrigida	2013-05-26 14:47:57.62
6	6	Bette	Nicholson	2013-05-26 14:47:57.62
7	7	Grace	Mostel	2013-05-26 14:47:57.62

# MSDE 631: Lab 4

## Etana Disasa

### Views

 dvrental/postgres@PostgreSQL 12

Query Editor


Query History

Scratch Pad

```
1 SELECT * FROM customer_list;
```

	<b>id</b> integer	<b>name</b> text	<b>address</b> character varying (50)	<b>zip code</b> character varying (10)	<b>phone</b> character varying (20)	<b>city</b> character varying (50)	<b>country</b> character varying (50)	<b>notes</b> text
1	524	Jared Ely	1003 Qinhuangdao Street	25972	35533115997	Purwakarta	Indonesia	active
2	1	Mary S...	1913 Hanoi Way	35200	28303384290	Sasebo	Japan	active
3	2	Patricia...	1121 Loja Avenue	17886	838635286649	San Bernardino	United States	active
4	3	Linda ...	692 Joliet Street	83579	448477190408	Athenai	Greece	active
5	4	Barbar...	1566 Inegl Manor	53561	705814003527	Myingyan	Myanmar	active
6	5	Elizabe...	53 Idfu Parkway	42399	10655648674	Nantou	Taiwan	active
7	6	Jennife...	1795 Santiago de Composte...	18743	860452626434	Laredo	United States	active
8	7	Maria ...	900 Santiago de Compostel...	93896	716571220373	Kragujevac	Yugoslavia	active
9	8	Susan ...	478 Joliet Way	77948	657282285970	Hamilton	New Zealand	active

View properties in PostgreSQL 12 appears a bit different. The *Code* tab displays the sql code for the view which would have been found in the *Definition* tab in older versions.

 **customer\_list**

General



Definition

**Code**

Security

SQL

```
1 SELECT cu.customer_id AS id,  
2 (cu.first_name::text || ' '::text) || cu.last_name::text AS name,  
3 a.address,  
4 a.postal_code AS "zip code",  
5 a.phone,  
6 city.city,  
7 country.country,  
8 CASE  
9 WHEN cu.activebool THEN 'active'::text  
10 ELSE ''::text  
11 END AS notes,  
12 cu.store_id AS sid  
13 FROM customer cu  
14 JOIN address a ON cu.address_id = a.address_id  
15 JOIN city ON a.city_id = city.city_id  
16 JOIN country ON city.country_id = country.country_id;
```

Cancel

Reset

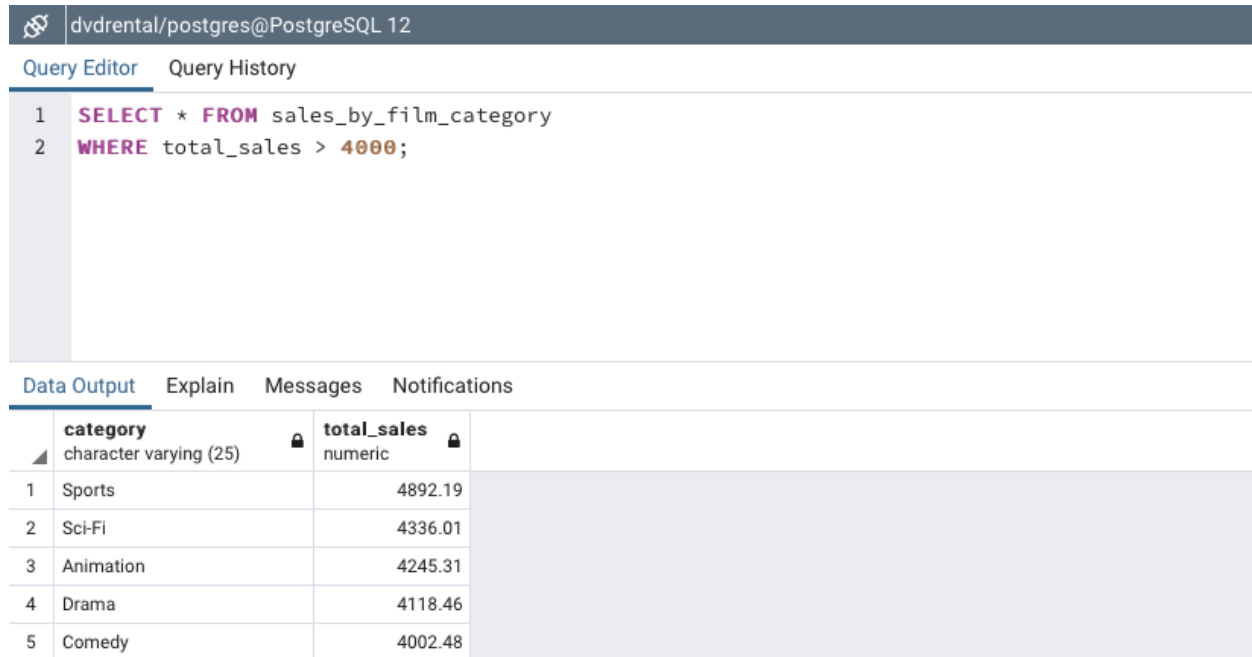
Save

## MSDE 631: Lab 4

Etana Disasa

Writing queries with views

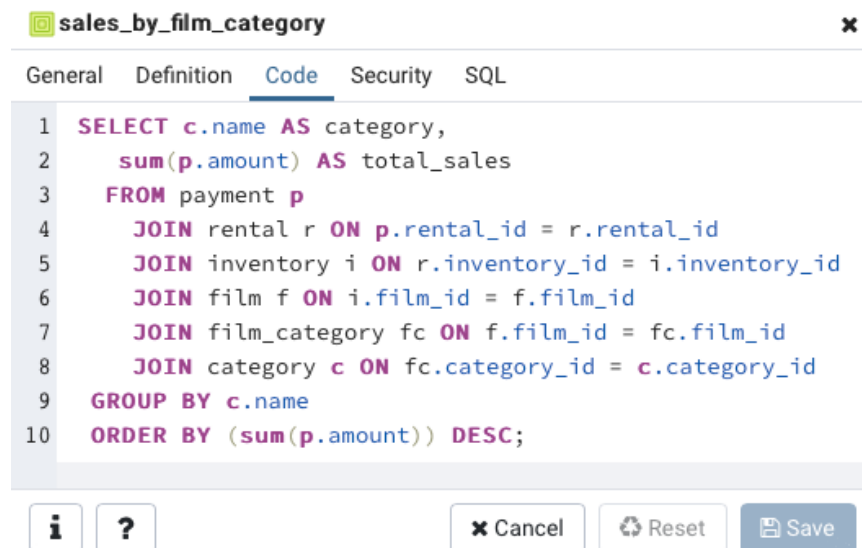
```
SELECT * FROM sales_by_film_category
WHERE total_sales > 4000;
```



The screenshot shows a PostgreSQL Query Editor interface. At the top, the connection is 'dvdrental/postgres@PostgreSQL 12'. Below the connection bar are tabs for 'Query Editor' and 'Query History'. The 'Query Editor' tab is active, displaying a SQL query with line numbers 1 and 2. Below the query editor are tabs for 'Data Output', 'Explain', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing a table with two columns: 'category' (character varying (25)) and 'total\_sales' (numeric). The table contains five rows of data, numbered 1 to 5.

	category	total_sales
1	Sports	4892.19
2	Sci-Fi	4336.01
3	Animation	4245.31
4	Drama	4118.46
5	Comedy	4002.48

The sales\_by\_film\_category view looks like this



The screenshot shows the 'Code' tab of a PostgreSQL Query Editor for a view named 'sales\_by\_film\_category'. The view definition is a SQL query that selects the category name and the sum of payment amounts, grouped by category and ordered by total sales in descending order. The query uses joins to connect the payment, rental, inventory, film, film\_category, and category tables.

```
SELECT c.name AS category,
       sum(p.amount) AS total_sales
FROM payment p
      JOIN rental r ON p.rental_id = r.rental_id
      JOIN inventory i ON r.inventory_id = i.inventory_id
      JOIN film f ON i.film_id = f.film_id
      JOIN film_category fc ON f.film_id = fc.film_id
      JOIN category c ON fc.category_id = c.category_id
GROUP BY c.name
ORDER BY (sum(p.amount)) DESC;
```

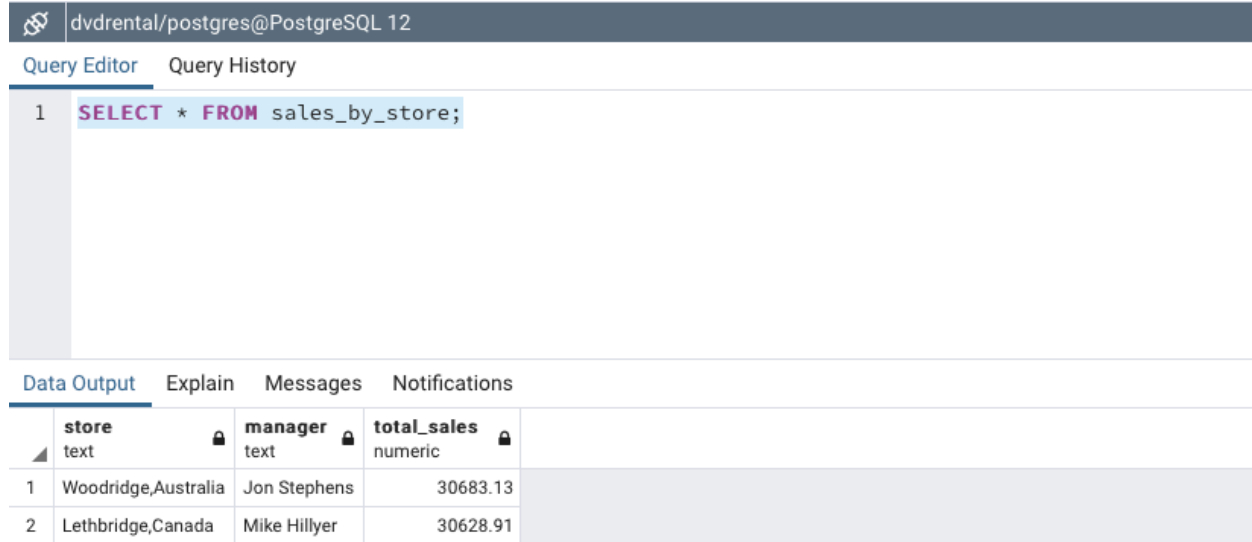
At the bottom of the editor, there are buttons for 'i' (info), '?' (help), 'Cancel', 'Reset', and 'Save'.



## MSDE 631: Lab 4

Etana Disasa

```
SELECT * FROM sales_by_store
```



Query Editor Query History

```
1 SELECT * FROM sales_by_store;
```

Data Output Explain Messages Notifications

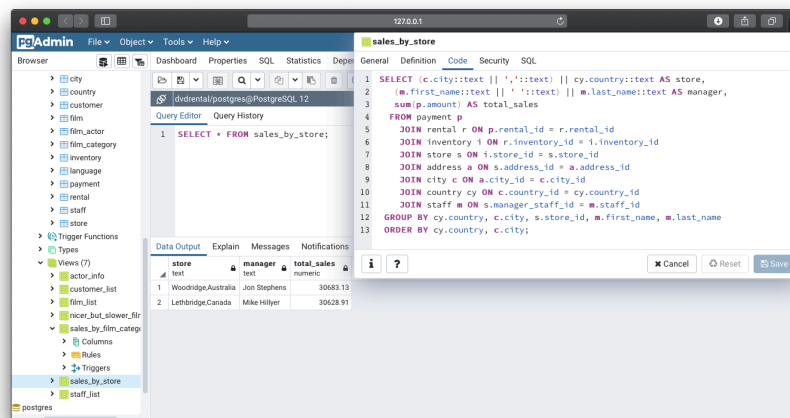
	store text	manager text	total_sales numeric
1	Woodridge,Australia	Jon Stephens	30683.13
2	Lethbridge,Canada	Mike Hillyer	30628.91

### Question

1. How many tables are joined by this SQL statement within the view?
2. Why does the “group by” clause contain so many column names?

### Answer

1. There are 8 tables that are joined together. And these tables are **payment**, **rental**, **inventory**, **store**, **address**, **city**, **country**, and **staff**.
2. The “group by” clause contains these many columns in order to determine the total sum of sales completed by individuals (**m.first\_name**, **m.last\_name**), by their store id (**s.store\_id**), by city (**c.city**) and by country (**cy.country**). Otherwise, we will have multiple individual’s that share the same first/last names from different cities will have their sales added together. The same thing to stores with similar id but in different cities or even countries can have their sales added together. Therefore, the “group by” columns rule out any error of adding the sales of individuals by error.



pgAdmin

Browser

- city
- country
- customer
- film
- film\_actor
- film\_category
- inventory
- language
- payment
- rental
- staff
- store
- Trigger Functions
- Types
- Views (7)
  - actor\_info
  - customer\_list
  - film\_list
  - nicer\_but\_slower\_film\_list
  - sales\_by\_film\_category
  - sales\_by\_store
  - staff\_list

Query Editor Query History

```
1 SELECT * FROM sales_by_store;
```

Data Output Explain Messages Notifications

	store text	manager text	total_sales numeric
1	Woodridge,Australia	Jon Stephens	30683.13
2	Lethbridge,Canada	Mike Hillyer	30628.91

sales\_by\_store

General Definition Code Security SQL

```
1 SELECT (c.city::text || ', '::text) || cy.country::text AS store,
2 (m.first_name::text || ' ' || m.last_name::text) AS manager,
3 sum(p.amount) AS total_sales
4 FROM payment p
5 JOIN rental r ON r.rental_id = r.rental_id
6 JOIN inventory i ON r.inventory_id = i.inventory_id
7 JOIN store s ON i.store_id = s.store_id
8 JOIN address a ON s.address_id = a.address_id
9 JOIN city c ON a.city_id = c.city_id
10 JOIN country cy ON c.country_id = cy.country_id
11 JOIN staff m ON s.manager_staff_id = m.staff_id
12 GROUP BY cy.country, c.city, s.store_id, m.first_name, m.last_name
13 ORDER BY cy.country, c.city;
```

## MSDE 631: Lab 4

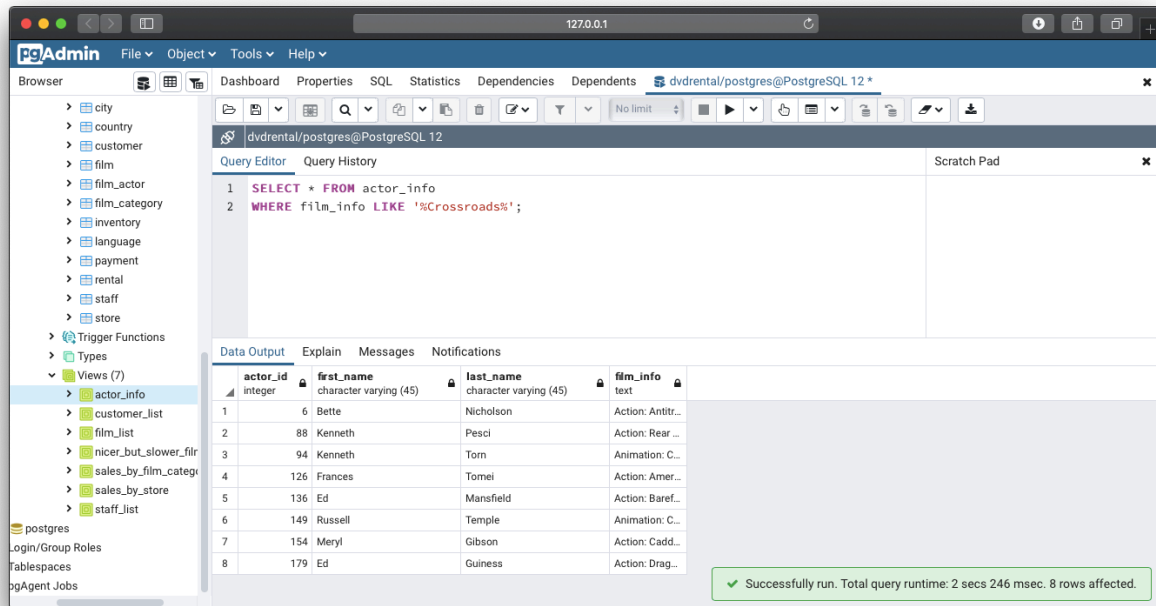
Etana Disasa

### Problem

Modify the query to select from the actor\_info view. Your result set should match the results shown below exactly using some type of filtering command. You want to find rows that contain the words 'Crossroads' as shown below.

### Solution

```
SELECT * FROM public.actor_info
WHERE film_info LIKE '%Crossroads%';
```



### Question

3. What does the 'Group By' do in the SQL?
4. What does the nested query do in the SQL? If you are not sure, grab the nested SQL and run the SQL by itself.
5. Why did they use Left Joins in this query instead of regular inner joins which we saw in prior queries?



## MSDE 631: Lab 4

Etana Disasa

### Answer

3. The "**GROUP BY** fa\_1.actor\_id" clause groups the joined tables by the actor\_id of the film\_actor table.
4. The nested query is about the fourth column to be called which is "film\_info". The first portion of the query tries to find the distinct "c.name" (category name) separated by ":". Then the nested query fetches all the films with the **same category** (with the same film\_category.category\_id and category.category\_id) and with the **same actor** (film\_actor.actor\_id and actor.actor\_id) and populates them all in the "film\_info" column.
5. Left join still joins and captures entries in the table on the left, even though there are missing observations in the table to the right. In other words, there will be entries in the actor table that do not have corresponding entry in the film\_actor table with a null value. When the happens, the joined table will still manage to join the tables. Therefore, there will not be rows from the table on the left, that will be left out in the merger just because there are not any corresponding values in columns in table to the right.

**END OF PART TWO**