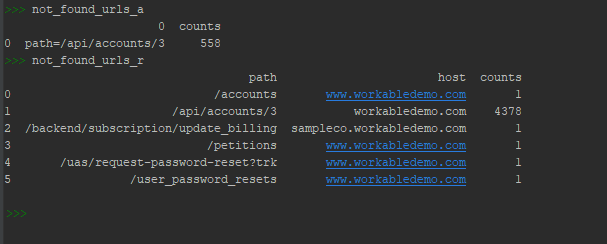
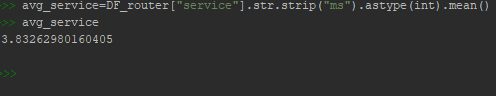
**TASK 1**

1. In order to answer to this question, only the heroku/[router] and app/web.X log entries were used. According to heroku’s documentation page, the inbound requests are received by a set of routers that forward the request to one of the application’s web dynos. The web dynos are the only type of dynos that can receive HTTP requests. The router logs can be correlated with the web dynos logs using the request\_id field. Two different dataframes were created, one for the router logs and one for the web dynos logs. To count how many times the status=404 had appeared in each dataframe, it was ensured that the same request\_id is only counted once.

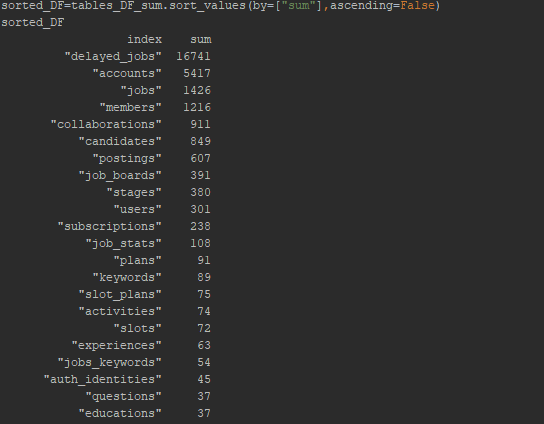
The not found url with the highest count is the : workabledemo.com/api/accounts/3



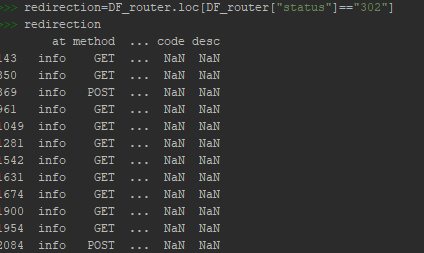
2.



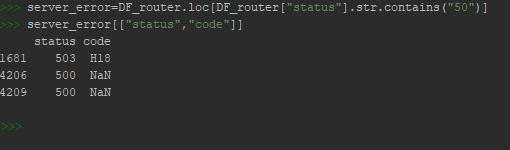
3. The table “delayed jobs” is more frequently loaded.



4. Yes URL redirection is taking place.



5. Yes, for example the error with code=H18, which according to heroku’s documentation, it stands for “Server Request Interrupted” and signifies that the socket connected, some data was sent as part of a response by the app, but then the socket was destroyed without completing the response.

****

**TASK 2**

1.

**SQL\_query=** *SELECT c.last\_name,c.first\_name, c.store\_id, count(r.rental\_id) FROM customer AS c INNER JOIN rental AS r ON c.customer\_id=r.customer\_id GROUP BY c.customer\_id HAVING c.store\_id=2 ORDER BY count(rental\_id) DESC LIMIT 1;*

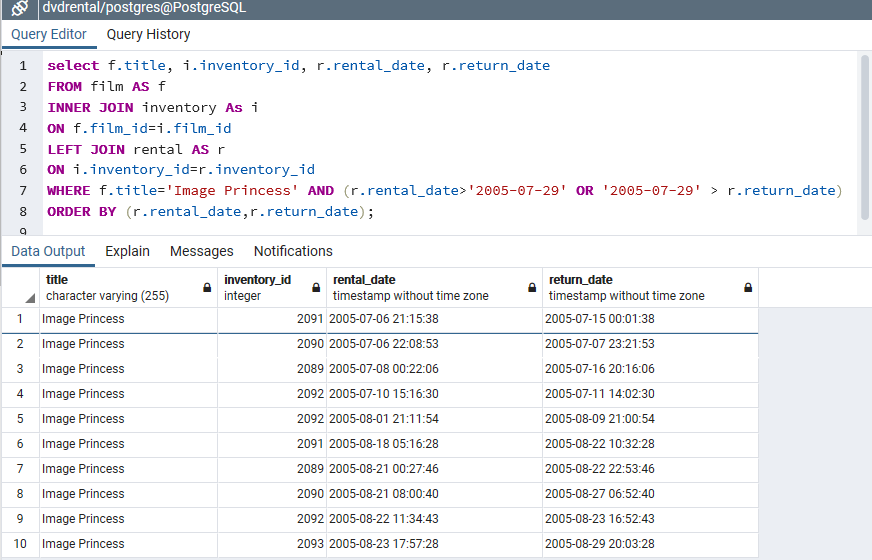
**Answer=**Seal Karl has the most rentals at store 2.



2.

**SQL\_query=** *select f.title, i.inventory\_id, r.rental\_date, r.return\_date FROM film AS f INNER JOIN inventory As i ON f.film\_id=i.film\_id LEFT JOIN rental AS r ON i.inventory\_id=r.inventory\_id WHERE f.title='Image Princess' AND (r.rental\_date>’2005-07-29’ OR ’2005-07-29’ >r.return\_date) ORDER BY (r.rental\_date,r.return\_date);*

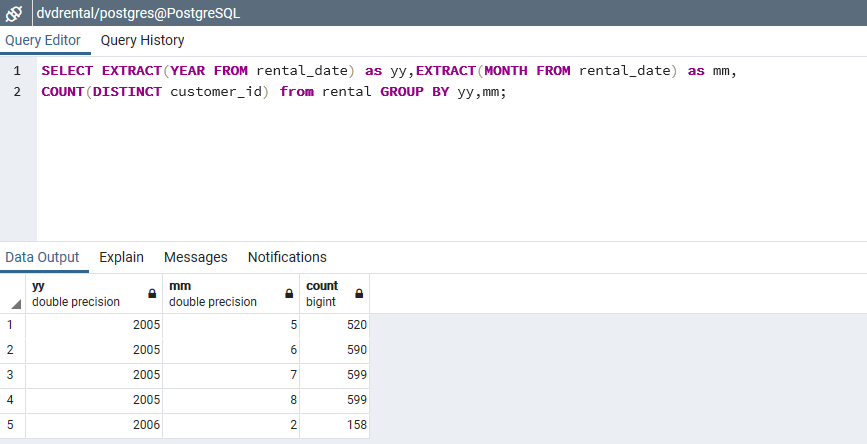
**Answer=**Yes he would be able to rent the copy of “Image Princess” with inventory id=2092, under the assumption that both stores have access to all inventory\_ids(copys).



3.

**SQL\_query=** *SELECT EXTRACT(YEAR FROM rental\_date) as yy,EXTRACT(MONTH FROM rental\_date) as mm, COUNT(DISTINCT customer\_id) from rental GROUP BY yy,mm;*

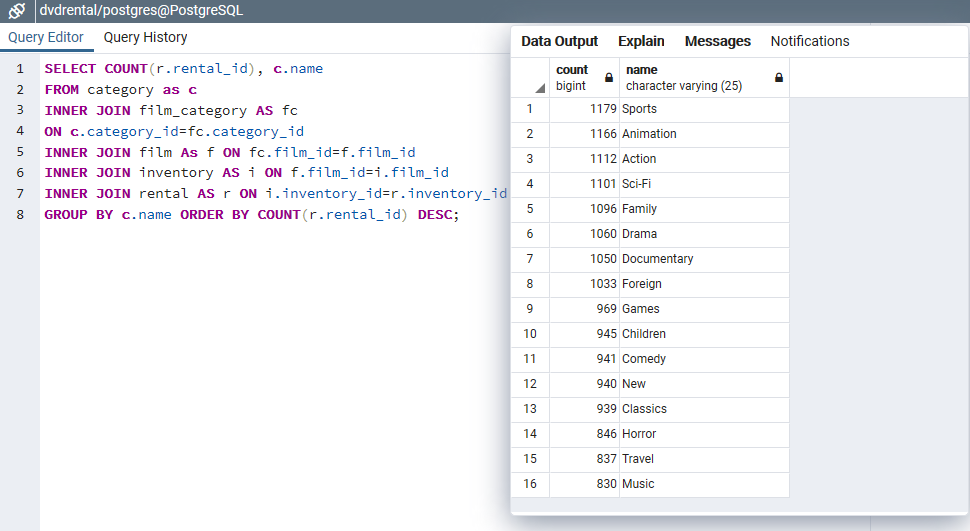
**Answer=**The counts are presented in the picture below.



4.

**SQL\_query=** *SELECT COUNT(r.rental\_id), c.name FROM category as c INNER JOIN film\_category AS fc ON c.category\_id=fc.category\_id INNER JOIN film As f ON fc.film\_id=f.film\_id INNER JOIN inventory AS i ON f.film\_id=i.film\_id INNER JOIN rental AS r ON i.inventory\_id=r.inventory\_id GROUP BY c.name ORDER BY COUNT(r.rental\_id) DESC;*

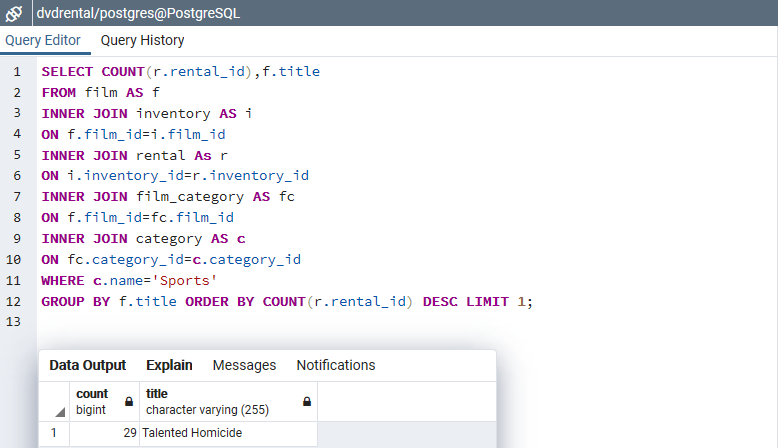
**Answer=**Sports is the most popular category among the store’s customers.

**

5.

**SQL\_query=** *SELECT COUNT(r.rental\_id),f.title FROM film AS f INNER JOIN inventory AS I ON f.film\_id=i.film\_id INNer JOIN rental As r ON i.inventory\_id=r.inventory\_id INNER JOIN film\_category AS fc ON f.film\_id=fc.film\_id INNER JOIN category AS c ON fc.category\_id=c.category\_id WHERE c.name='Sports' GROUP BY f.title ORDER BY COUNT(r.rental\_id) DESC LIMIT 1;*

**Answer=**Talented Homicide.



6. Examples of other insights that we can obtain from the data are the

following:

* Find out which customers are the oldest and more active in order to reward them with a discount.
* Find out which customers exceed the movies’ rental duration , in order to impose a penalty.
* Find out for each individual customer what their favorite category is and recommend them the highest rated movies from that category that they haven’t already seen.

These actions could increase the business’ income and ensure the

existence of a loyal customer base.

**TASK 3**

Web Application using the Flask framework for accessing TMDB API and storing information into MySQL Server.

The project contains 3 different files:

*app.py* : script for creating and launching the application.

*index.html*: html page to display the list of movies currently in theatres in Greece

*db.yaml*: configuration file for connection with db.

*DDL Commands*

CREATE DATABASE movies;

CREATE TABLE now\_playing(movie\_id varchar(30),original\_title varchar(30),title varchar(30),overview text(300));

ALTER TABLE now\_playing MODIFY overview text(1000);

