Codd Movie Rentals

Database Management

April 19, 2016

Zachary Fong

INSERT TABLE OF CONTENTS HERE

Table of Contents

Executive Summary3

Entity Relationship Diagram4

Tables5

Conditions6

Genre7

Actors8

Movies9

Rent\_Status10

Cust\_rentals11

Payment12

Views13

PayStatus14

CustomerRoster15

MissingMovie16

Reports17

Total Income18

Average Income19

Triggers20

Security21

Known Problems22

Future Enhancements22

# Executive Summary

The purpose of this document is to outline a database to record resources within a movie rental store. Specifically, this database records the condition of a move, its genre, title of the movie, its price, and customer information. This will allow the manager of the store to keep track of different movies and whether or not a customer returned the movie in a worse condition or if they even returned it at all.

This document will provide an overview of the database. This includes the various tables in the database, the purpose of each table, and their functional dependencies, and more.

## https://documents.lucidchart.com/documents/c6237a7b-f50b-44ed-932f-45d4938ba6ca/pages/0_0?a=443&x=42&y=185&w=1276&h=773&store=1&accept=image%2F*&auth=LCA%20e97e9000425b0a6f1e7b10f48832bbdc8d438dd5-ts%3D1460926679ENTITY RELATIONSHIP DIAGRAM

### TABLES

CONDITION

**Purpose**

This table is used to store the state of the conditions of different movies

**Create Statement**

CREATE TABLE Condition (

cid char(4) not null,

description text,

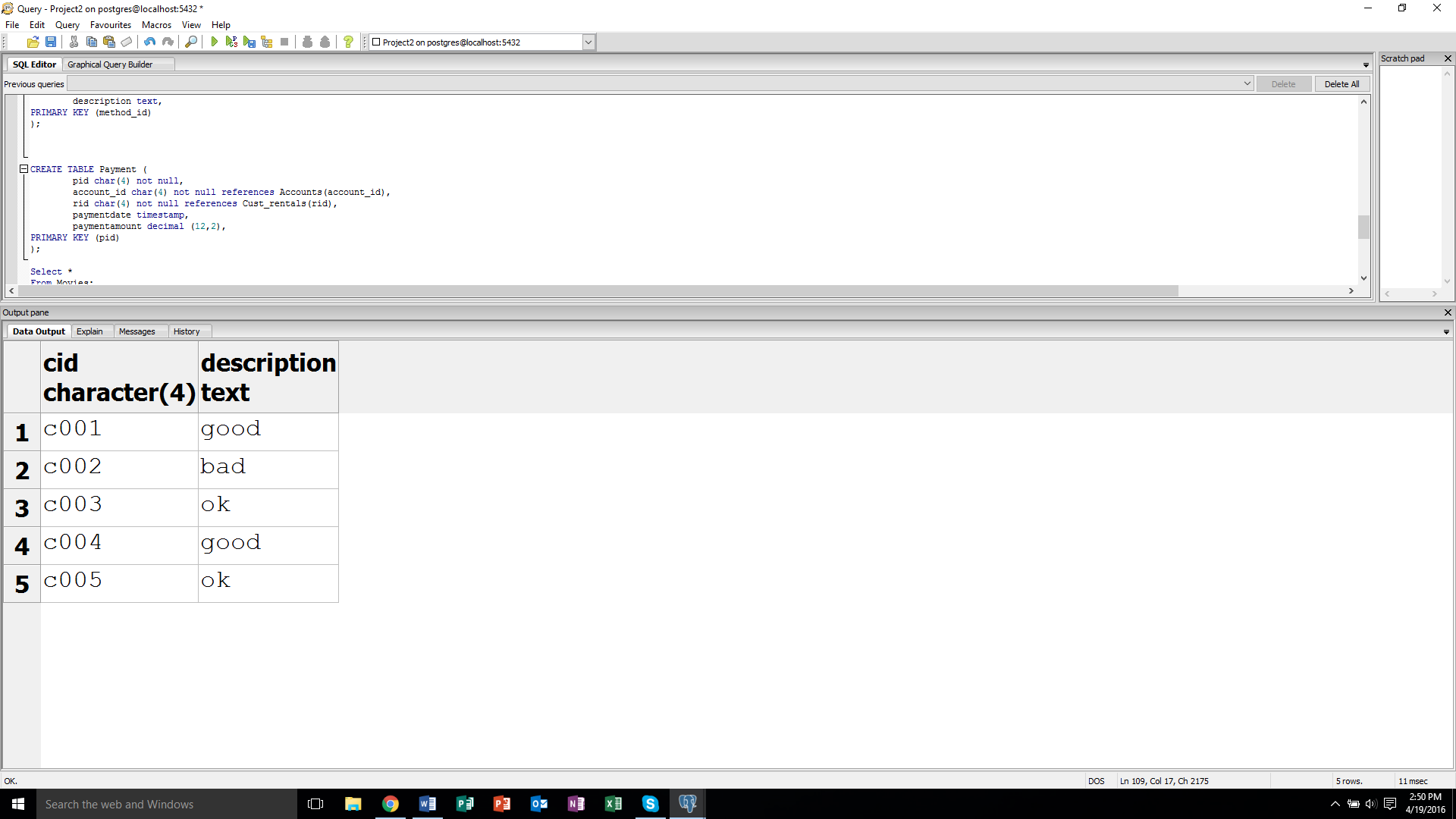
PRIMARY KEY (cid)

);

**Functional Dependencies**

cid🡪 desc

**Sample Data**



GENRE

**Purpose**

This table is used to identify the genres for each movie.

**Create Statement**

CREATE TABLE Genre (

gid char(4) not null,

description text,

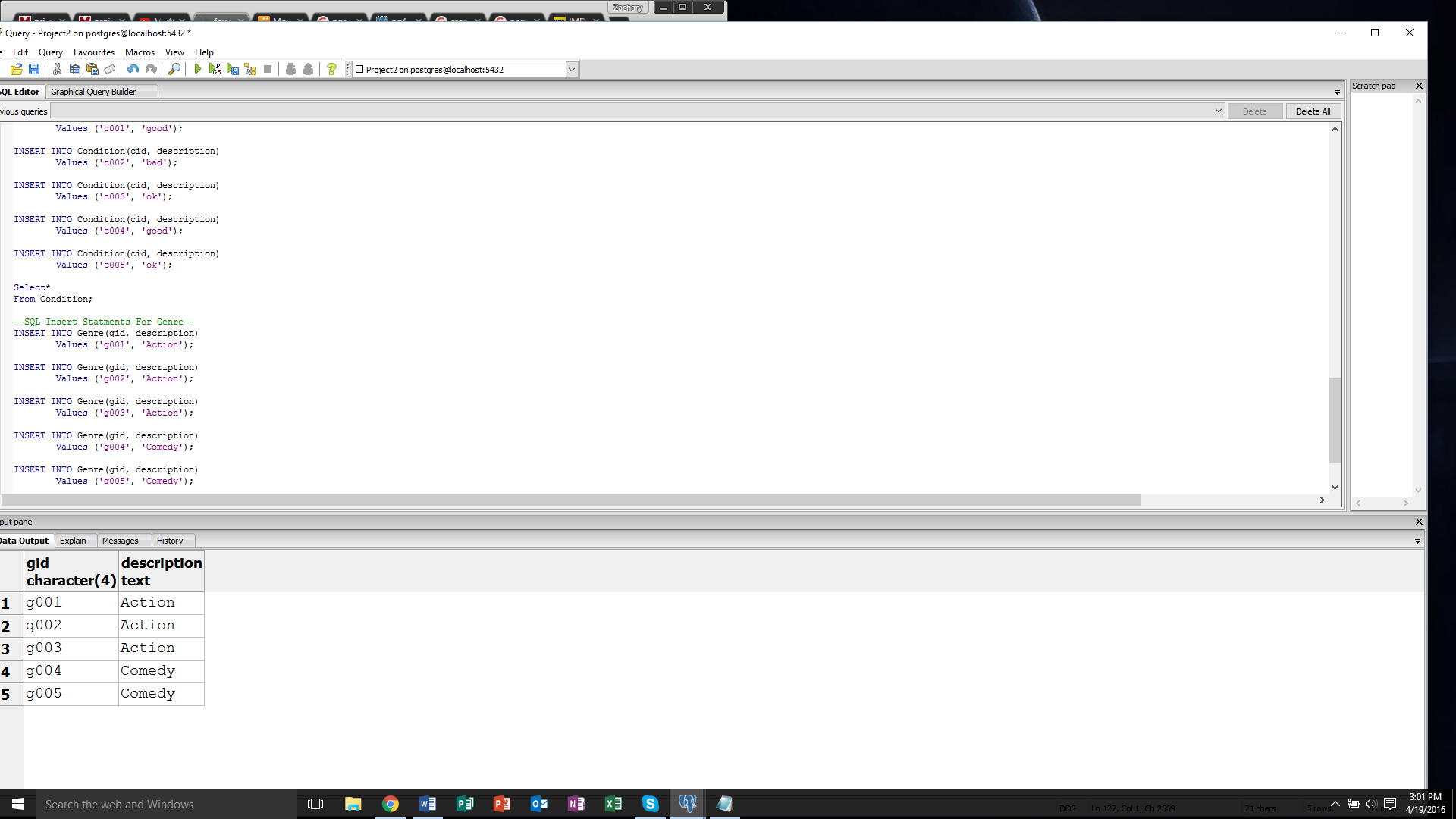
PRIMARY KEY (gid)

);

**Functional Dependencies**

gid 🡪 desc

**Sample Data**



ACTORS

**Purpose**

This table is used to store a list of actor so a user can search for movies by actors.

**Create Statement**

CREATE TABLE Actors (

aid char(4) not null,

gender text,

fname text,

lname text,

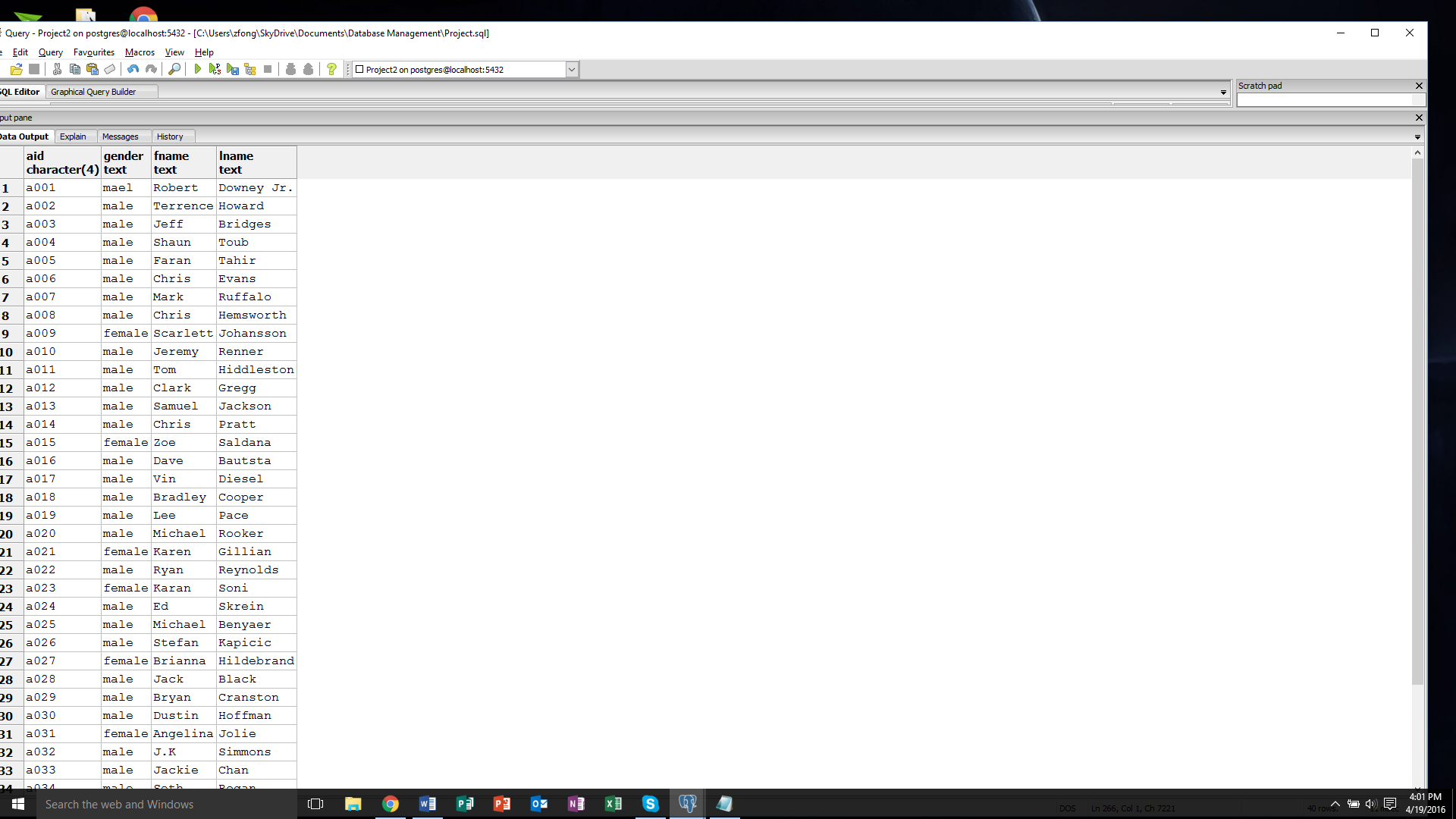
Primary Key (aid)

);

**Functional Dependencies**

aid 🡪 gender, fname, lname

**Sample Data**



MOVIES

**Purpose**

This table is used to store the different movies available for rent.

**Create Statement**

CREATE TABLE Movies (

mid char(4) not null,

cid char(4) not null references Condition(cid),

gid char(4) not null references Genre(gid),

yearrelased int,

title text,

rentprice int,

saleprice int,

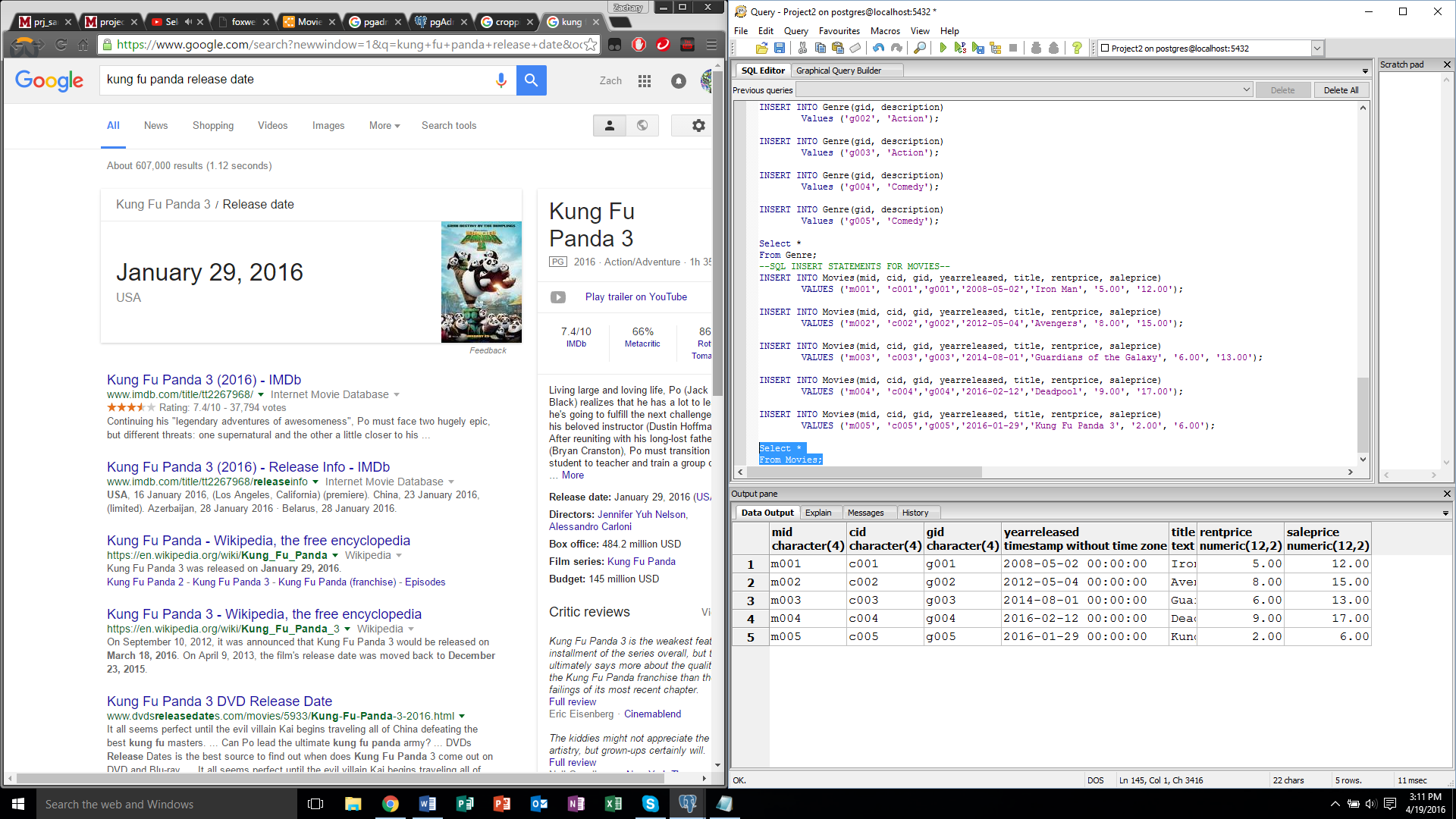
PRIMARY KEY (mid)

);

**Functional Dependencies**

mid🡪 cid, gid, yearreleased, title, rentprice, saleprice

**Sample Data**



RENT\_STATUS

**Purpose**

This table is used to keep track of the status of rented movies.

**Create Statement**

CREATE TABLE RentStatus (

status\_id char(4) not null,

description text,

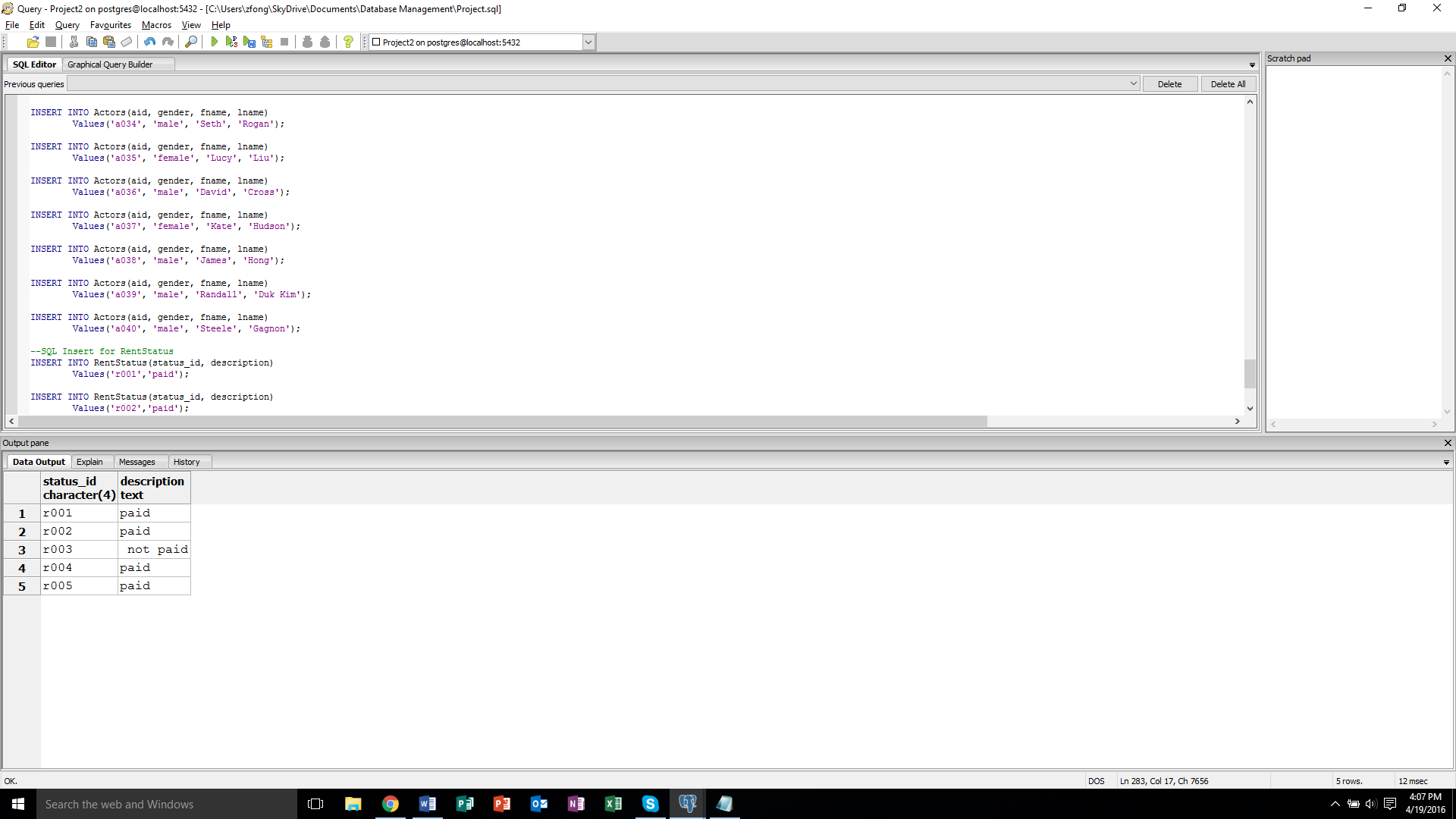
PRIMARY KEY(status\_id)

);

**Functional Dependencies**

status\_id🡪desc

**Sample Data**



CUST\_RENTALS

**Purpose**

This table is used to keep track of when a customer rented a movie and whether or not they returned it.

**Create Statement**

CREATE TABLE Cust\_rentals (

rid char(4) not null,

cust\_id char(4) not null references Customers(cust\_id),

mid char(4) not null references Movies(mid),

status\_id char(4) not null references RentStatus(status\_id),

dayrented timestamp not null,

dayreturned timestamp,

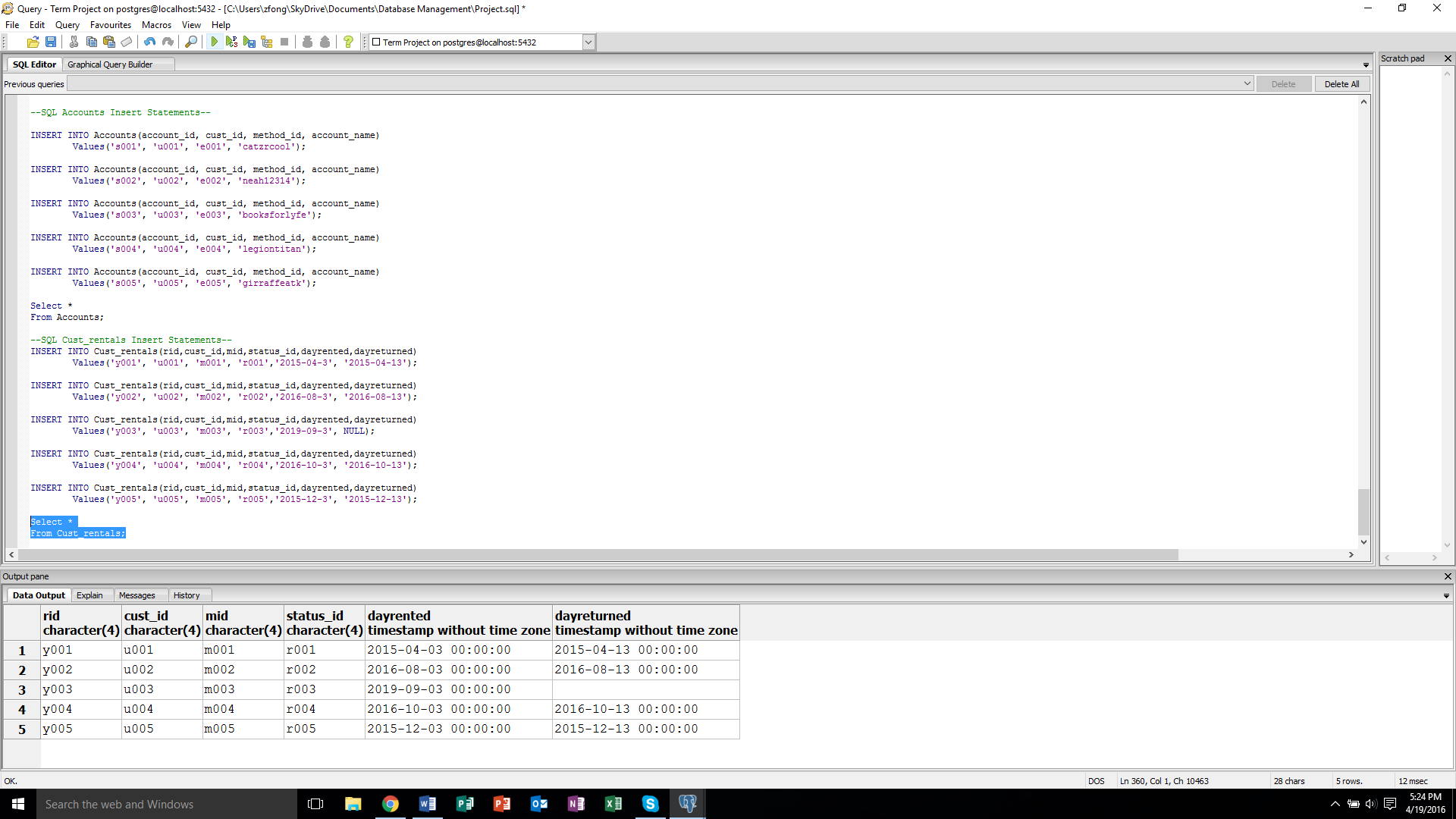
PRIMARY KEY(rid)

);

**Functional Dependencies**

rid🡪cust\_id, mid, status\_id, dayrented, dayreturned

**Sample Data**



PAYMENT

**Purpose**

This table is used to track when the customer paid and how much they paid.

**Create Statement**

CREATE TABLE Payment (

pid char(4) not null,

account\_id char(4) not null references Accounts(account\_id),

rid char(4) not null references Cust\_rentals(rid),

paymentdate timestamp,

paymentamount decimal (12,2),

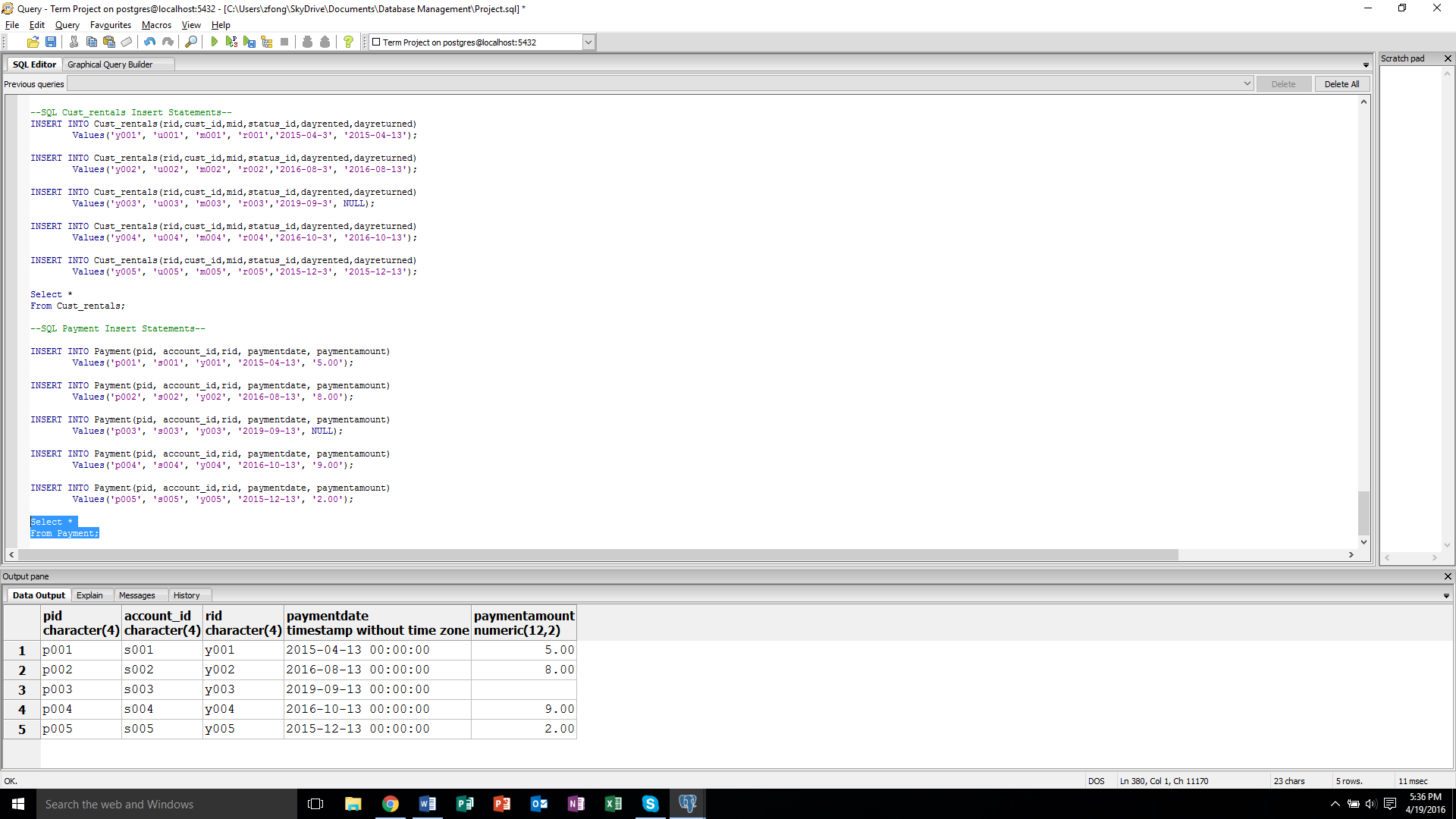
PRIMARY KEY (pid)

);

**Functional Dependencies**

pid🡪account\_id, rid, paymentdate, paymentamount

**Sample Data**



CUSTOMERS

**Purpose**

This table is used to store the customer’s information.

**Create Statement**

CREATE TABLE Customers (

cust\_id char(4) not null,

cname text,

address text,

phone bigint not null,

email text,

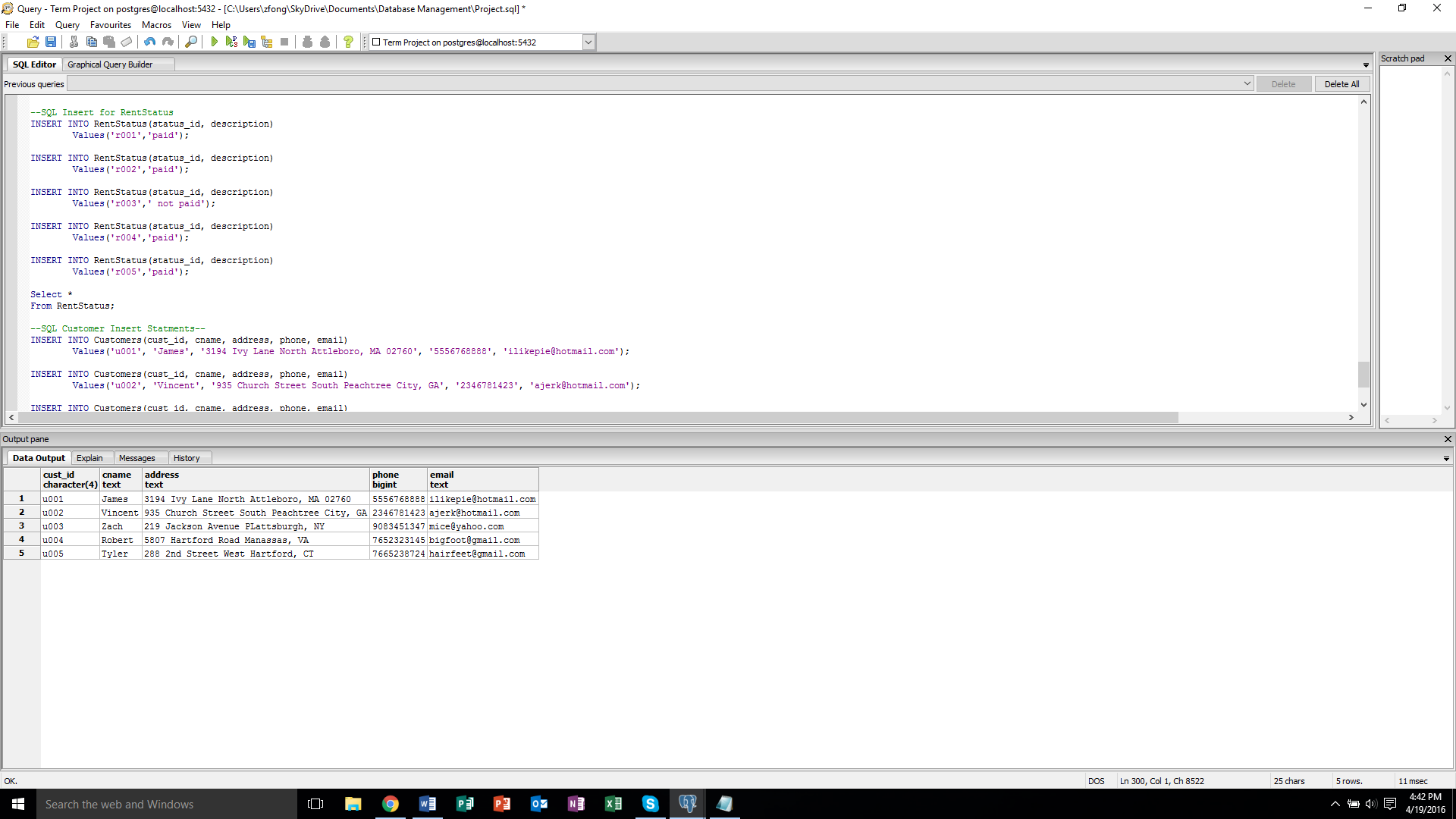
PRIMARY KEY (cust\_id)

);

**Functional Dependencies**

cust\_id🡪name, address, phone, email

**Sample Data**



ACCOUNTS

**Purpose**

This table is used to keep a record of the accounts of customers.

**Create Statement**

CREATE TABLE Accounts (

account\_id char(4) not null,

cust\_id char(4) not null references Customers(cust\_id),

method\_id char(4) not null references PaymentMethod(method\_id),

account\_name text,

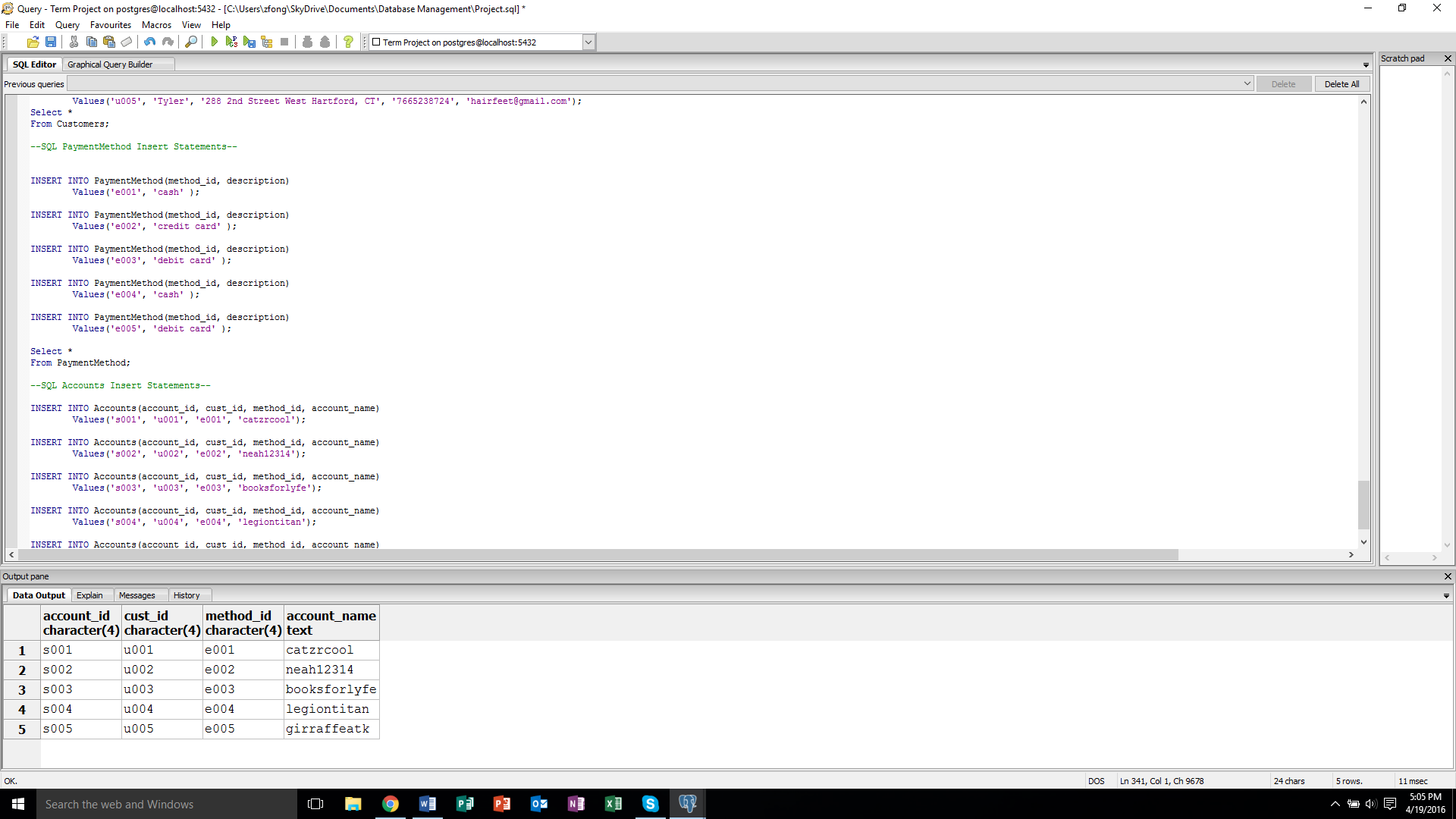
PRIMARY KEY (account\_id)

);

**Functional Dependencies**

account\_id🡪cust\_id, method\_id, account\_name

**Sample Data**



PAYMENT METHOD

**Purpose**

This table is used to store the methods customers used to pay.

**Create Statement**

CREATE TABLE PaymentMethod (

method\_id char(4) not null,

description text,

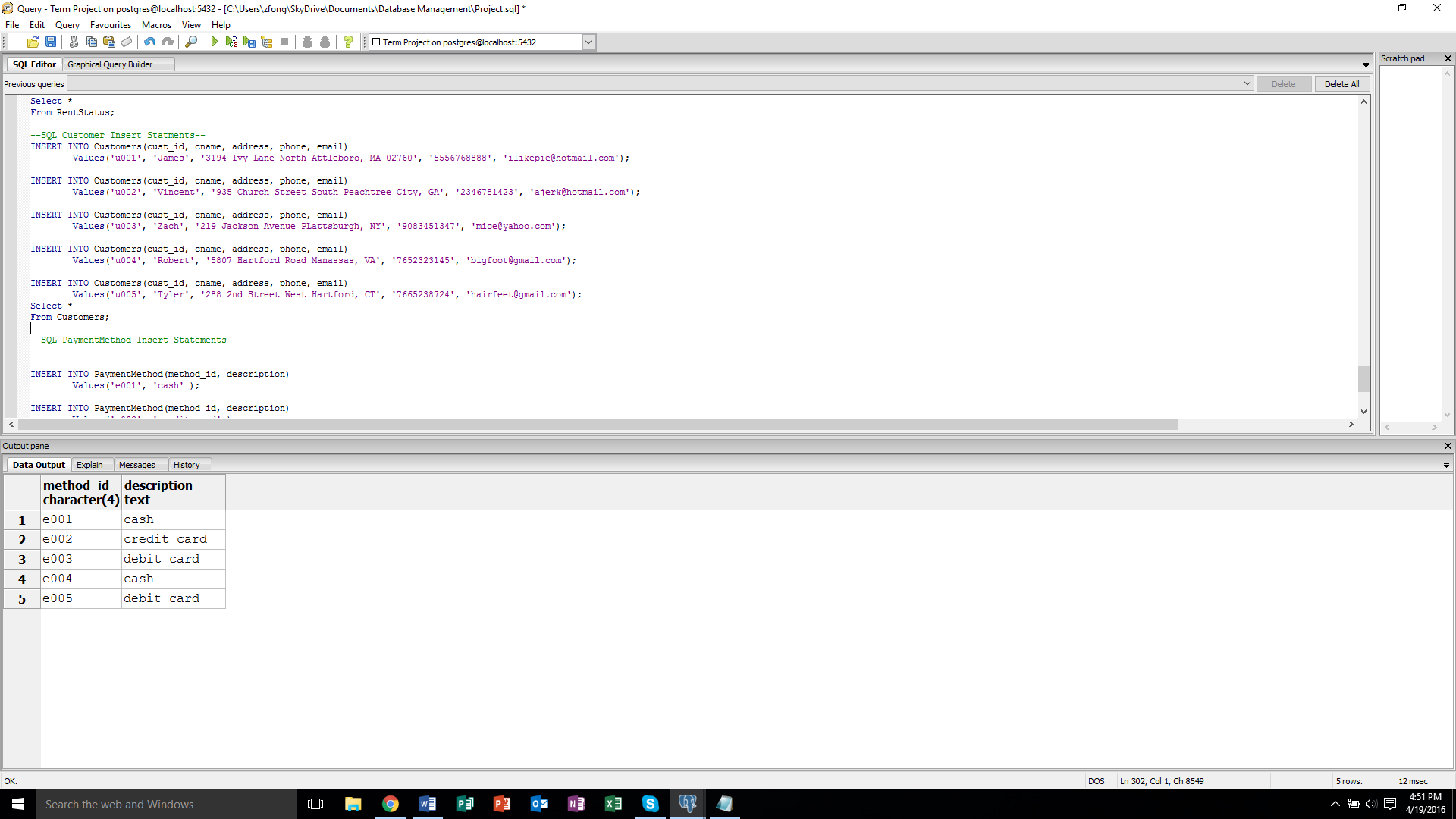
PRIMARY KEY (method\_id)

);

**Functional Dependencies**

method\_id🡪desc

**Sample Data**



#### VIEWS

PayStatus

**Purpose**

This view is used to determine the payment status of a movie once it is returned and what they paid with.

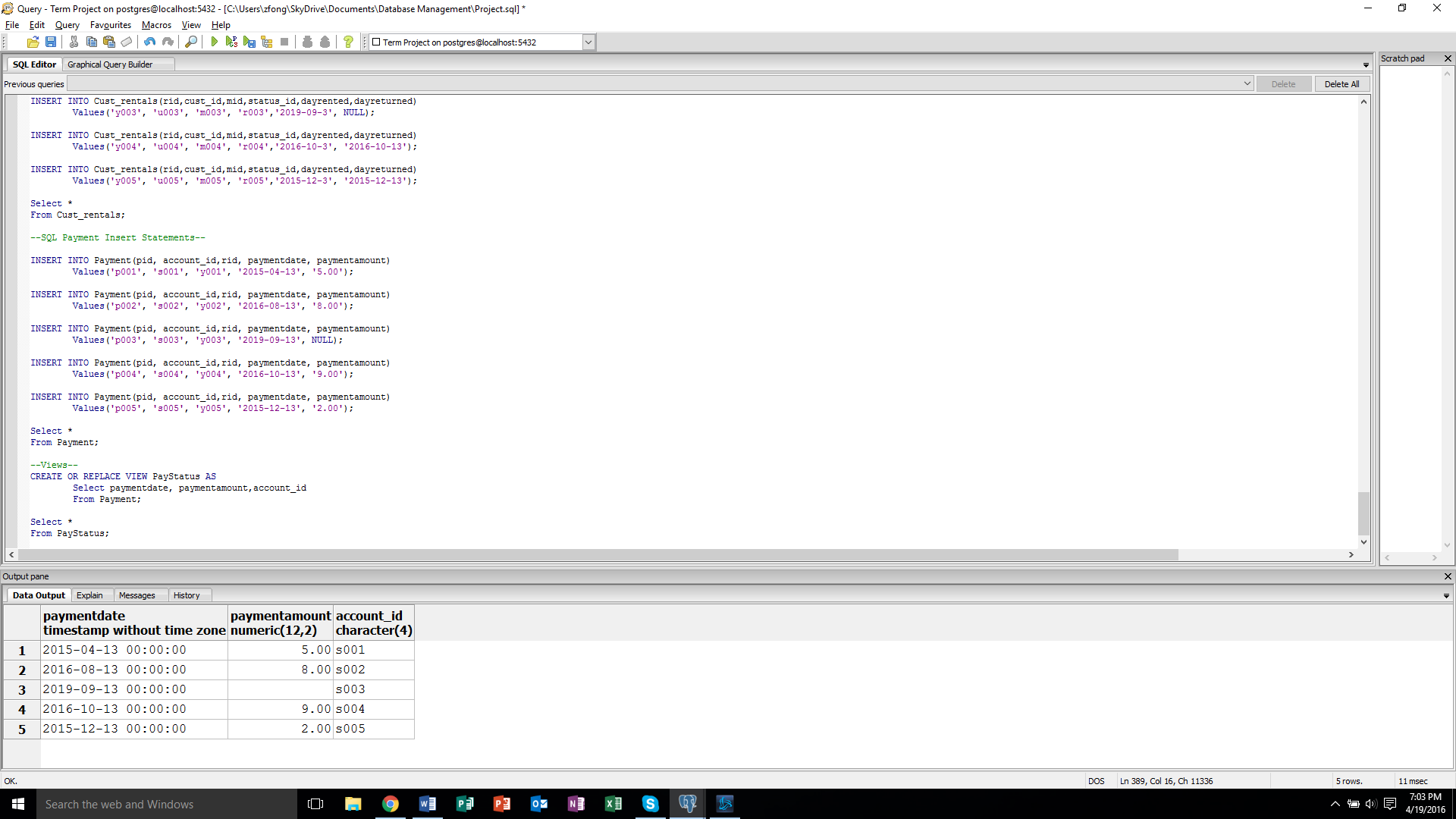
**Create Statement**

CREATE VIEW PayStatus AS

Select paymentdate, paymentamount

From Payment;

**Sample Data**

****

Customer Roster

**Purpose**

This view shows the entire list of customers and all their relevant information.

**Create Statement**

CREATE VIEW CustomerRoster AS

Select c.cid AS Customer ID,

lname, fname,

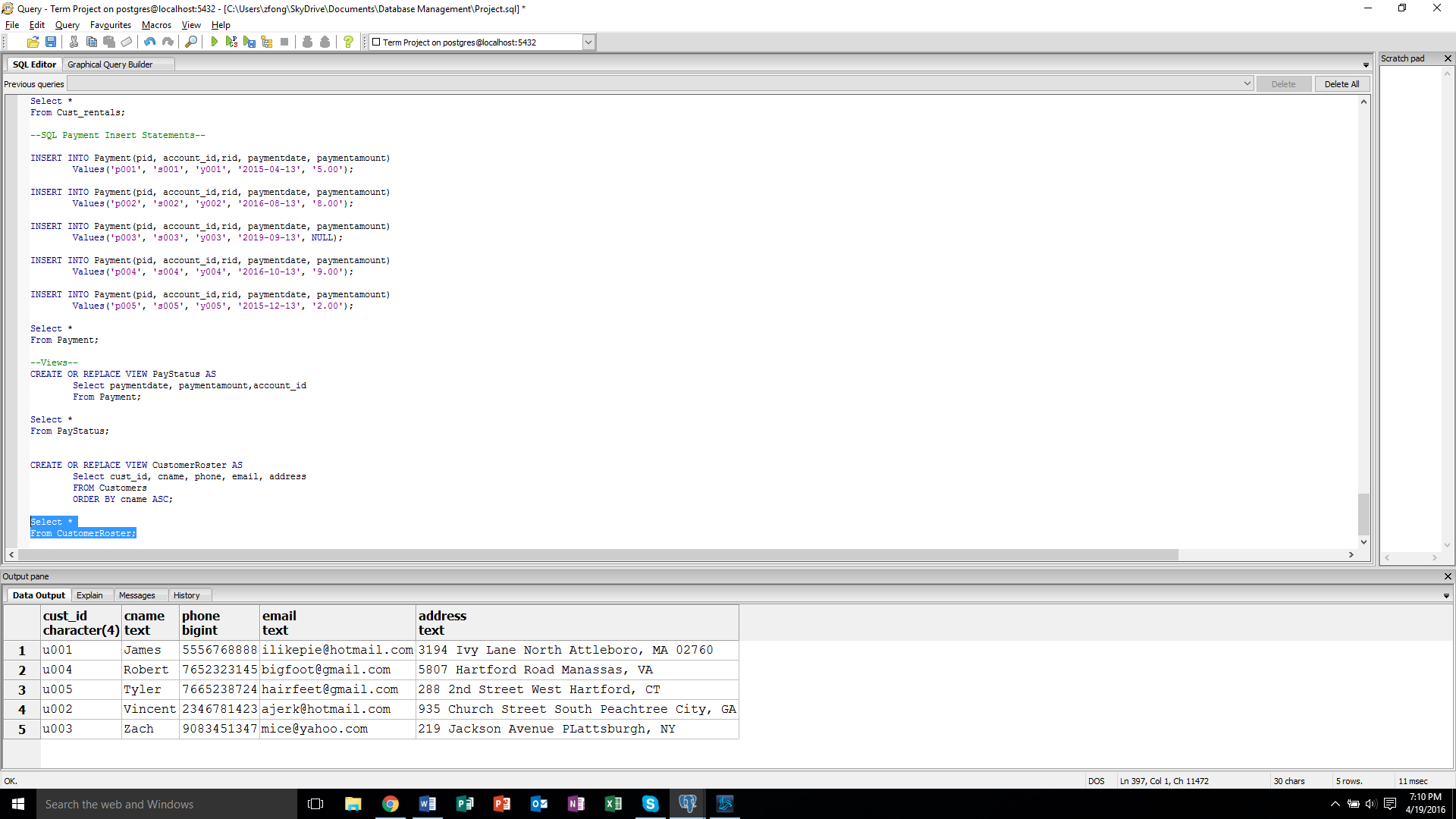
Phone, email,

Address,

From Customers c

Order By lname ASC;

**Sample Data**

****

Missing Movies

**Purpose**

The purpose of this view is to check whether any customers have not returned their movies.

**Create Statement**

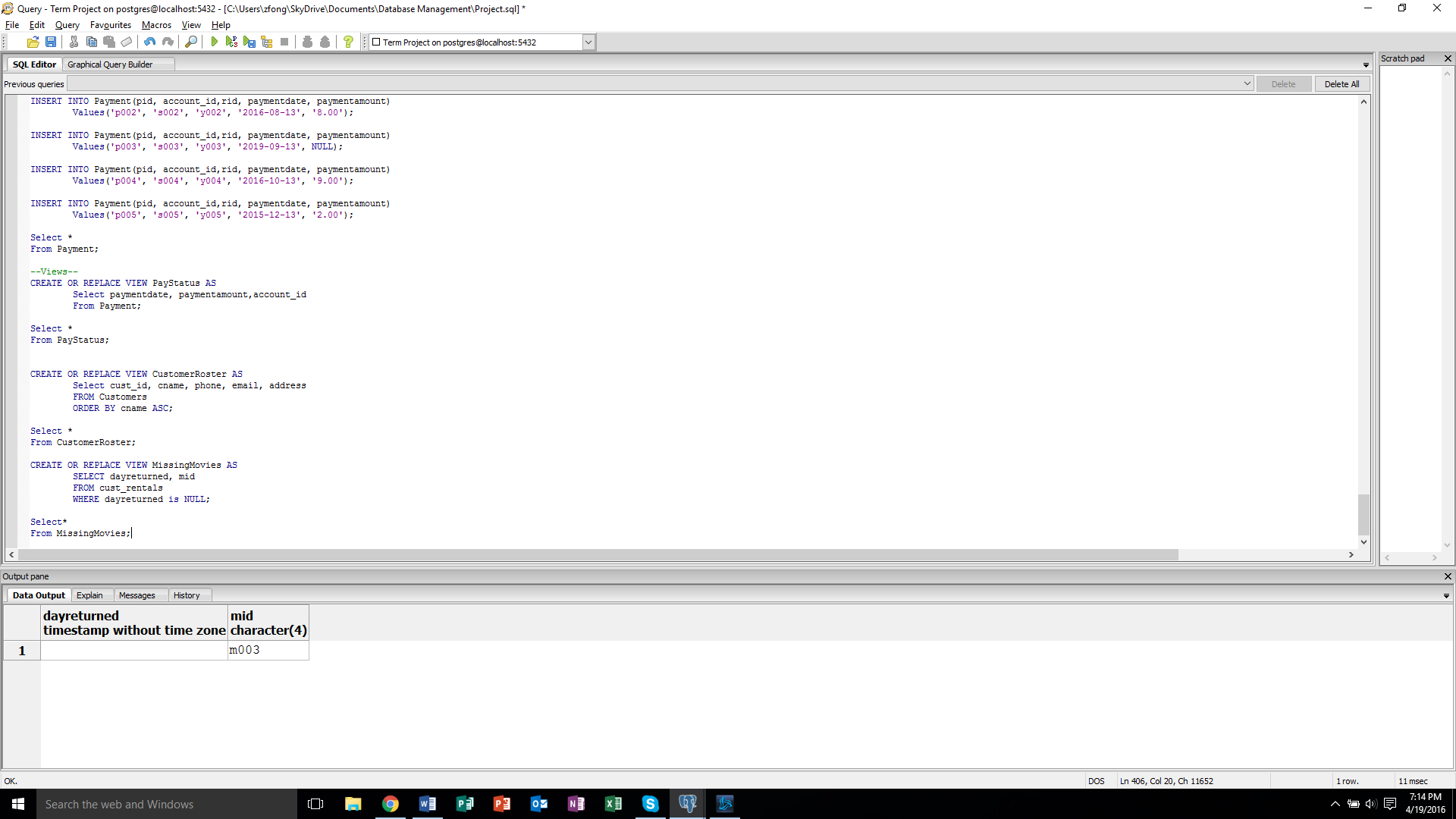
CREATE VIEW MissingMovies AS

Select dayreturned, mid

From Cust\_rentals

Where dayreturned IS NULL;

**Sample Data**

****

##### Reports

**Total Income**

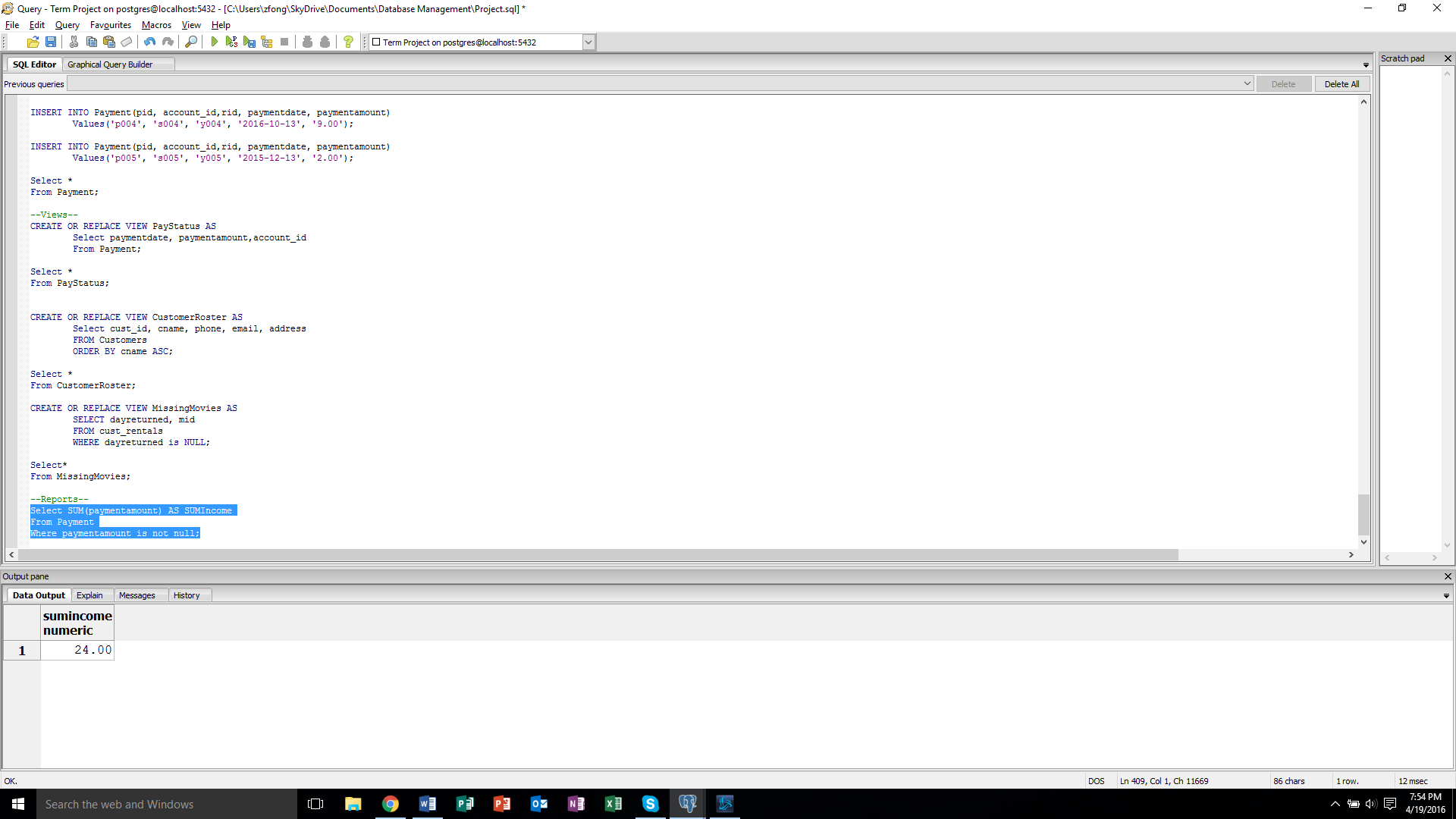
This is used to determine the total income of all movies that have either been rented or have been completely bought.

**Query**

Select SUM(paymentamount) AS SUMIncome

From Payment

Where paymentamount is not null;

Average Income

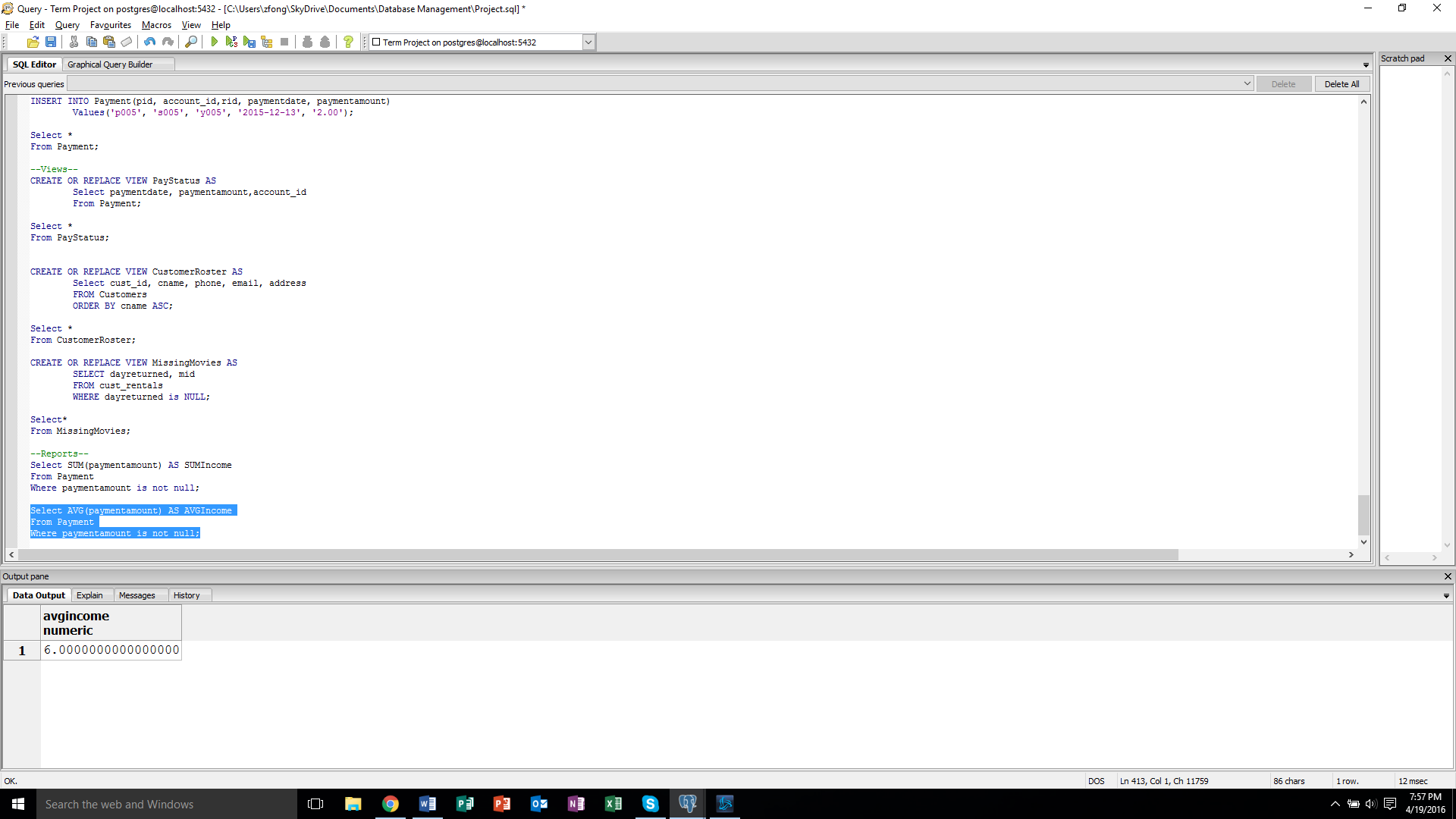
This is used for the manager to be able to see what the average income is for all returned rented movies and movie sales.

**Query**

Select AVG(paymentamount) AS AVGIncome

From Payment

Where paymentamount is not null;

****

###### Triggers

UnpaidRental

**Purpose**

When a customer has either forgotten to pay for their rented movie or has loaned the movie for an extended time period the table will immediately increase the amount due whenever the customer decides to return the movie

**Query**

CREATE TRIGGER UnpaidRental

AFTER UPDATE ON Payment

FOR EACH ROW EXECUETE PROCEDURE addpayment();

Security

**Database Administrator**

The Database administrator has access to everything.

GRANT ALL PRIVELEGES ON ALL TABLES IN SCHEMA public TO dbAdministator;

**Manager**

Manager is able to see all information in the database with the exception of not being able to see the customer’s financial information.

GRANT SELECT ON Movies TO manager;

GRANT SELECT, UPDATE ON Rent Status TO manager;

GRANT SELECT, INSERT, UPDATE ON Customers TO manager;

GRANT SELECT, INSERT, UPDATE ON Accounts TO manager;

GRANT SELECT, UPDATE Actors TO manager;

GRANT SELECT, INSERT, UPDATE ON Genre TO manager;

GRANT SELECT, INSERT, UPDATE ON Cust\_rentals TO manager;

GRANT SELECT, INSERT, UPDATE ON Condition TO manager;

Known Problems

* Did not account for actors being in multiple movies examples would be Robert Downey Jr being in multiple sequels of Iron Man and in the Avengers.
* Did not implement a way for customers to search for movies based on who directed the movie
* Some movies have multiple genres did not include all genres the movie may apply to only the main genre.

Future Enhancements

* Allow an employment table to keep track of an employee roster and see when they clock in and out
* The employment table should be separated from managerial positions and regular employees.