Portfolio Project, Option #1: Johnson Video Store

Irfan Peterson

Colorado State University – Global Campus

June 7th, 2020

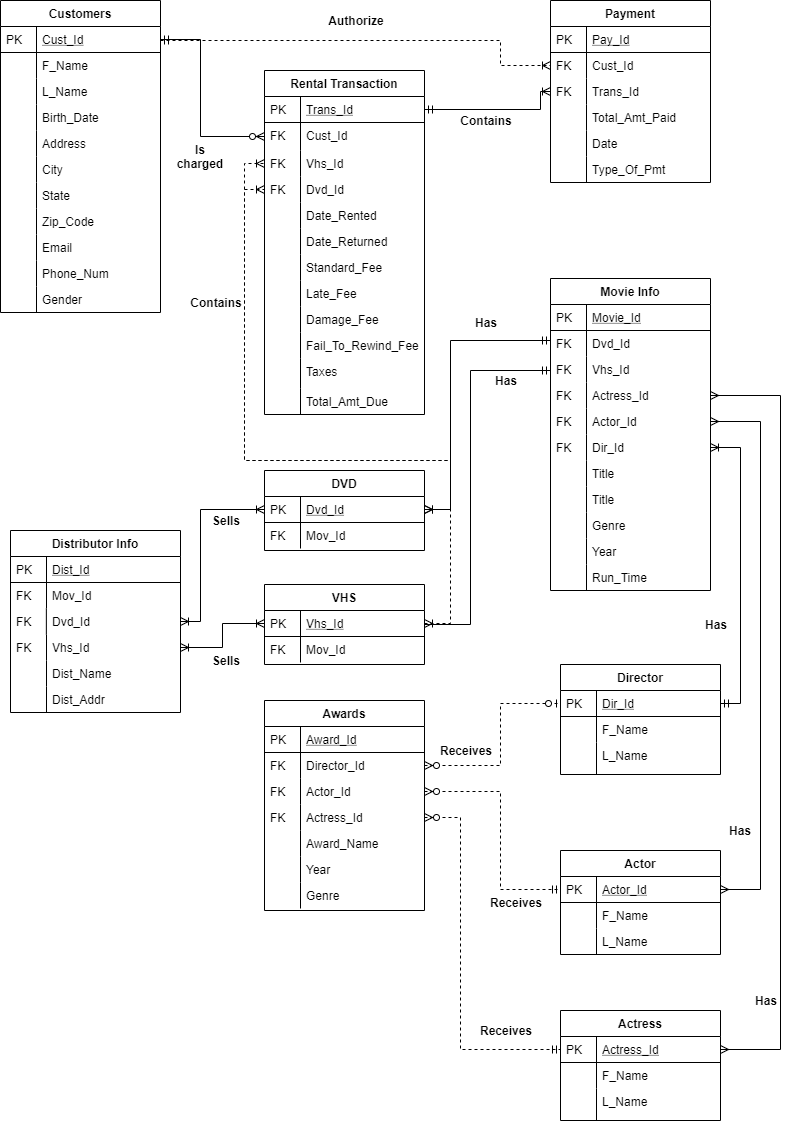
Portfolio Project, Option #1: Johnson Video Store

For the portfolio project, I have chosen option one, which is create a database for a small business called Johnson Video Store. A little background of the project, the owners have been keeping their records of videos and DVDs purchased from distributors and rented to customers in stacks of paper files and forms. To keep up with a modern business model and increasing demands, they have decided to automate their record keeping with a database. In the next few pages, I will go over how the ERD is created, how the metadata is derived from the ERD, how to execute a script file of DDL to create database, how to execute a script file of DML to insert data in the tables and how to execute various script to perform queries outlined in this project.

**ERD of Johnson Video Store**

Before I can begin to explain the elements of the Johnson Video Store ERD, it is beneficial to identify the parts of the diagram and the terminology that was used. An ERD has three basic elements: entity types, attributes, and relationships (Mannino, 2002, p.145). Entity types are a collection of data interests represented by a rectangle or a table in the diagram. Attributes are properties of the entity types and contain a primary key and other description, (Mannino, 2002, p.145).

When looking at the ERD, displayed on figure one below, we can see there are eleven tables displayed: Customers, Rental Transaction, Payment, Movie Info, DVD, VHS, Director, Actor, Actress, Awards, and Distributor Info. The eleven tables represent the entity types and when looking closer, you can see their attributes, or description, with the primary key being represented by PK next to it, followed by the foreign key underneath, represented by FK. The normal attributes will be represented below the foreign keys

****

*Figure 1*. Entity Relationship Diagram for Johnson Video Store Database

**Relationship Amongst Diagrams**

Next, we need to provide a high-level overview of the relationships amongst all eleven diagrams. According to Manninon (2002, p.145), relationships are named associations among entity types. The first relationship we are analyzing is between Customers and Rental Transaction. The relationship between the two is mandatory, as represented by the solid line and has a cardinality of one to many; a customer can have none or multiple rental (transaction) and a transaction can have on one customer. Next, we will look at the relationship between Customer and Payment; a customer can have one or more payments and a payment can only have one customer. The relationship between the two is optional and the cardinality is one to many. Subsequently, we will look at the relationship between Rental Transaction and DVD. The relationship is optional, and the cardinality is many to many; a rental transaction may contain one or many DVDs and a rental DVD may have one or many transactions. Then, we will look at the relationship between Rental Transaction and VHS. The relationship is optional, and the cardinality is many to many; a rental transaction may contain one or many VHS’ and a rental VHS’ may have one or many transactions. The relationship between Movie Info and DVD is mandatory to one. The cardinality between the two entities is one to many; a movie can be copied into one or more DVD and a DVD has only one movie. As for the relationship between Movie Info and VHS, it is mandatory to one. The cardinality between the two entities is one to many; a movie can be copied to one or more VHS and a VHS has only one movie.

When looking at the relationship of the Movie Info, Actor and Actress, we can see that it is a mandatory relationship. The cardinality is many to many; a Movie Info can have one or many actors and actresses. The relationship of a Movie Info and Director is mandatory with a cardinality of one to many; a move can have a director and a director can have multiple movies. Another relationship that will be analyzed are amongst the Award entity with Director, Actor, and Actresses entity. The relationship is optional with a many to many cardinalities; an actor, actresses and directors can have zero or multiple awards, and an award can be given to one actor, actress or director. Last, a Distributor’s relationship with DVD and VHS are mandatory with cardinality of many to many; a distributor can supply one or many DVD and VHS, and a VHS or DVD can have one or many distributors.

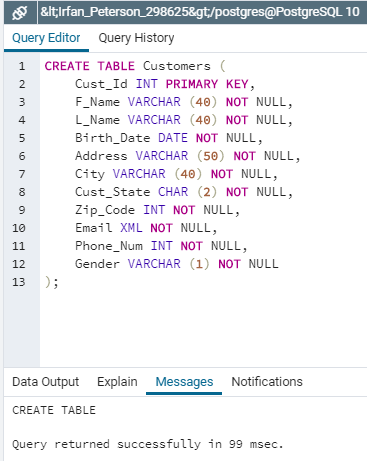
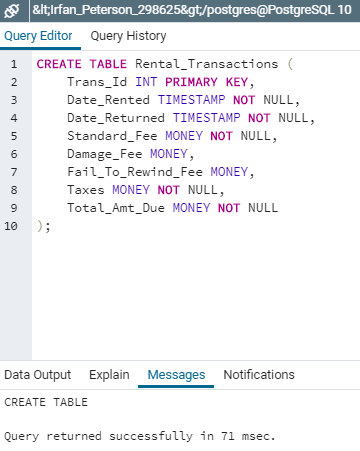
**Metadata Extracted from the ERD**

Using the ERD, as displayed on figure one, we can extract various information to create the metadata. As outlined in the attached metadata excel sheet, we can see the various attributes, foreign keys, and primary keys of the entities in the Johnson Video Store database. Additionally, we can see the key type, data type, size, and null classification, all which will assists us in creating the database by using various DDL SQL scripts.

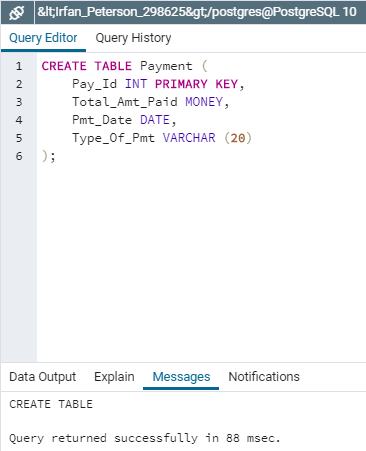
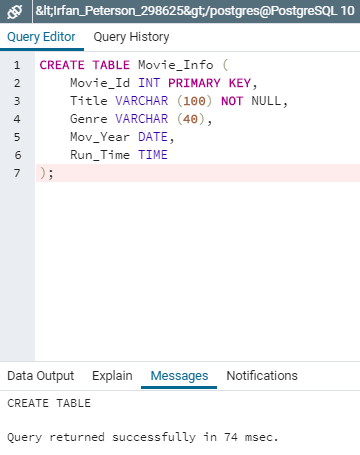
**Creating Database Using DDL SQL Scripts**

In Postgre SQL, there are various ways we can create table. However, since not all database has the same functionality as Postgre SQL, I will be creating the tables using DDL SQL Scripts. A CREATE TABLE statement can be used to define the heading of the table (Mannino, 2002, p.48). The column name and data types are specified next. There are different data types that can be specified for each column, which includes CHAR for fixed-length text entries, VARCHAR for variable length text, INTEGER for whole numbers, etc (Mannino, 2002, p. 49). Additionally, a (NOT) NULL statement can be used for columns that require data for each entry. A null statement is a requirement when assigning a Primary Key, a specially designated candidate key, unique from one another, and its used as a part of the integrity rule. To implement a primary key, a CONSTRAINT clause is used at the end of the statement, (Mannino, 2002, p.52).

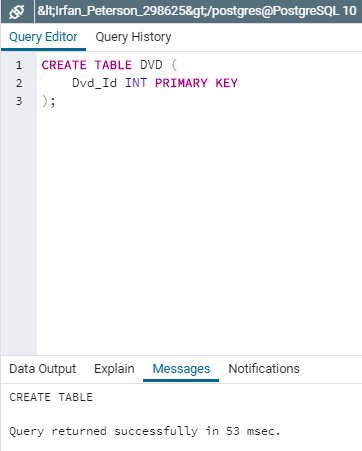
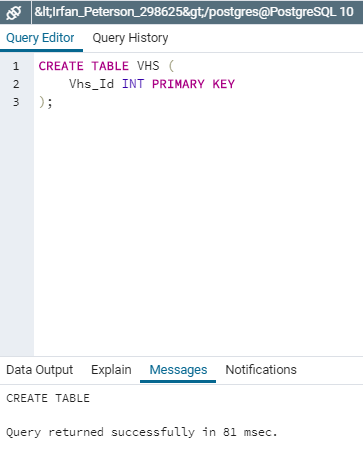
The following screenshots, shows the scripts that was used to create the eleven tables, note that the primary key and various attributes are included in the script:

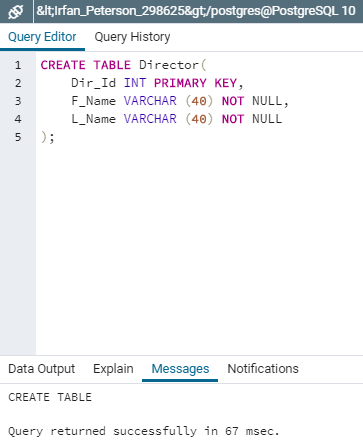
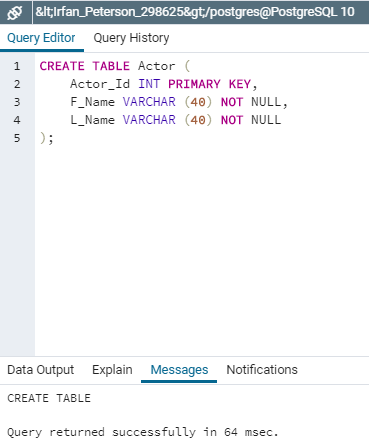
*Figure 2.* CREATE SQL statement for Customer and Rental table.

* *

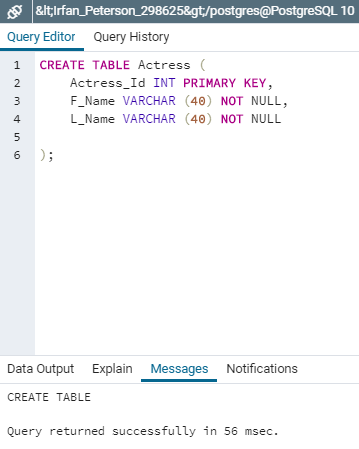
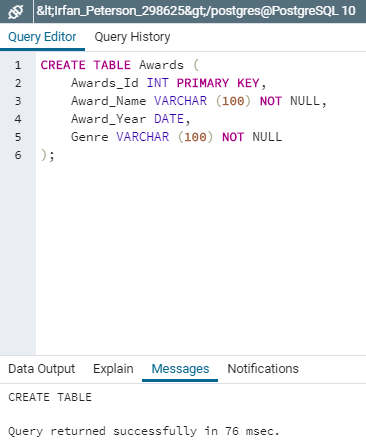
*Figure 3.* CREATE SQL statement for Payment and Movie\_Info table.

* *

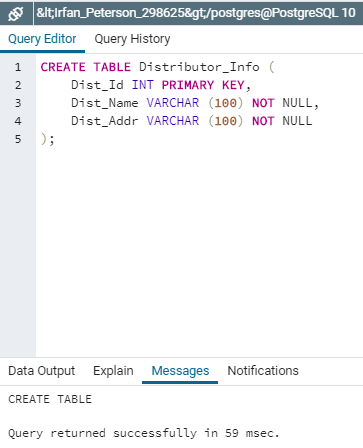
*Figure 4.* CREATE SQL statement for DVD and VHS table.

* *

*Figure 5.* CREATE SQL statement for Director and Actor table.

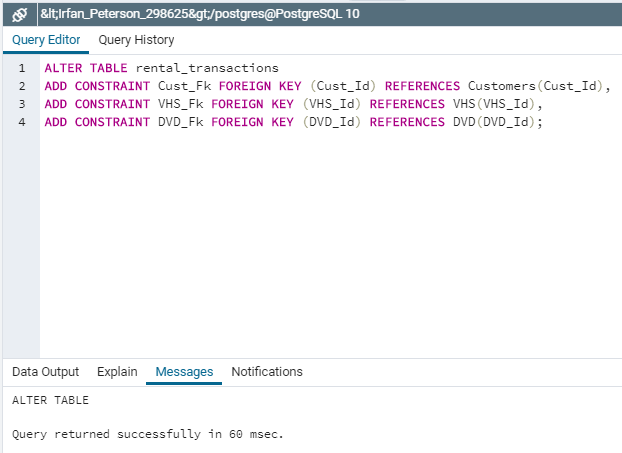
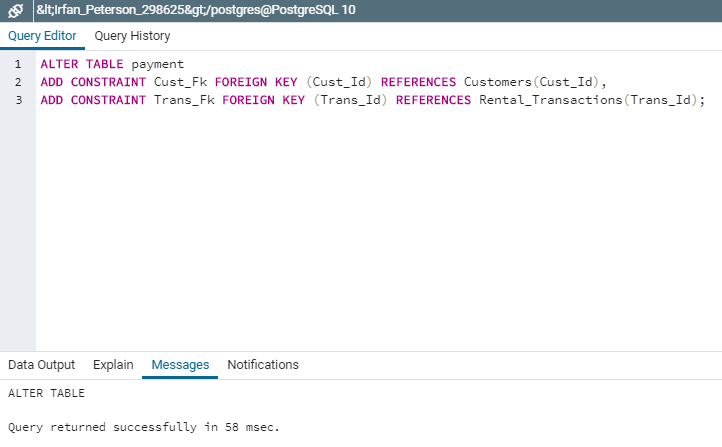
* *

*Figure 6.* CREATE SQL statement for Actress and Awards table.

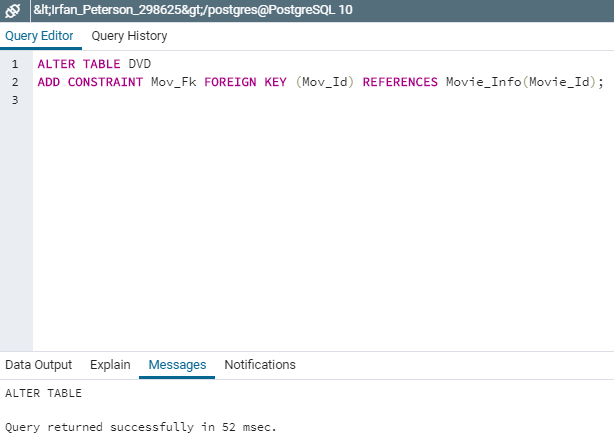
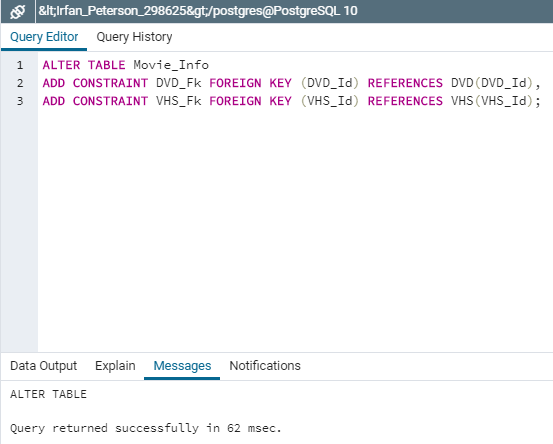
**

*Figure 7.* CREATE SQL statement for Distributor\_Info table.

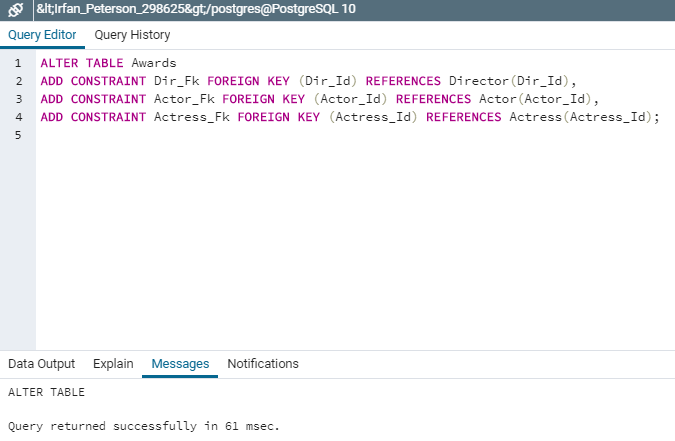
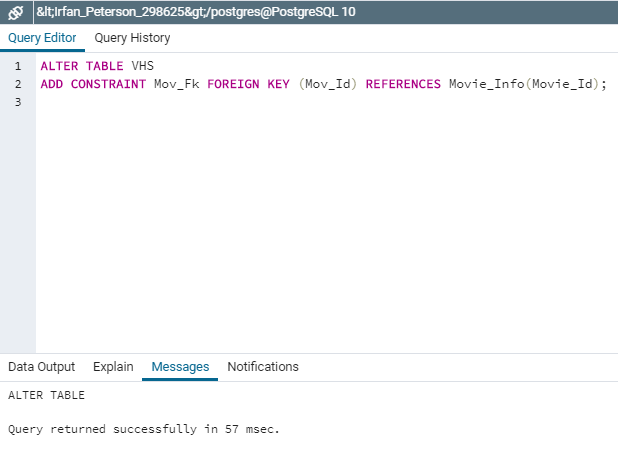
After all eleven tables are created, we can then assign foreign keys using the ALTER SCRIPT.

**

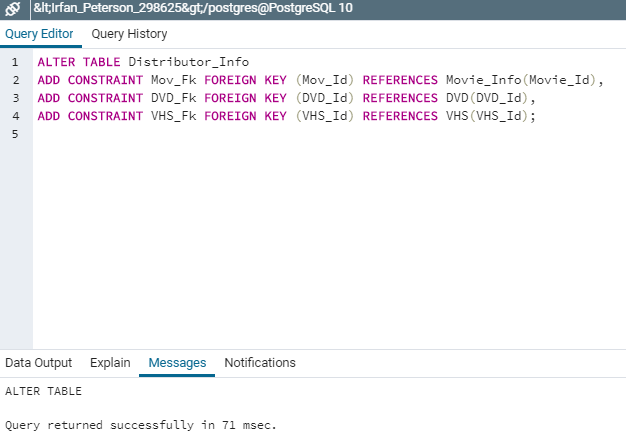
*Figure 8.* ALTER and ADD CONSTRAINT SQL statement to declare foreign keys on Rental\_Transaction and Payment table.

**

*Figure 9.* ALTER and ADD CONSTRAINT SQL statement to declare foreign keys on Movie\_Info and DVD table.

**

*Figure 10.* ALTER and ADD CONSTRAINT SQL statement to declare foreign keys on Movie\_Info and DVD table.

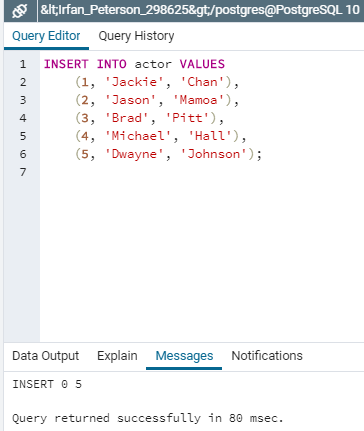
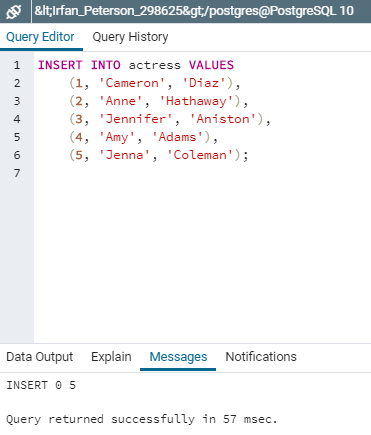
**

*Figure 11.* ALTER and ADD CONSTRAINT SQL statement to declare foreign keys on Distributor\_Info table.

**Inserting Data Into Tables**

****

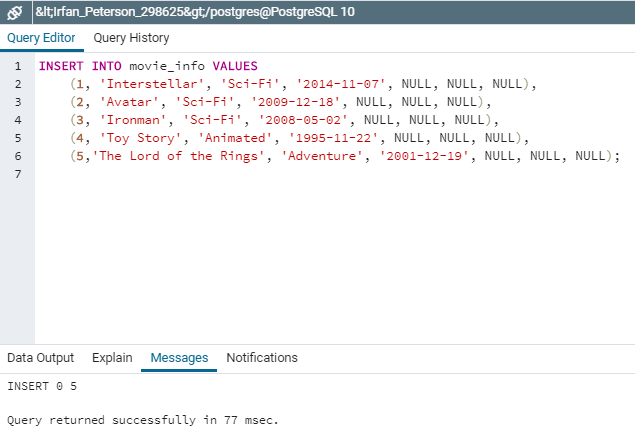
*Figure 12.* INSERT INTO SQL statement to populate data into Customer table.

** **

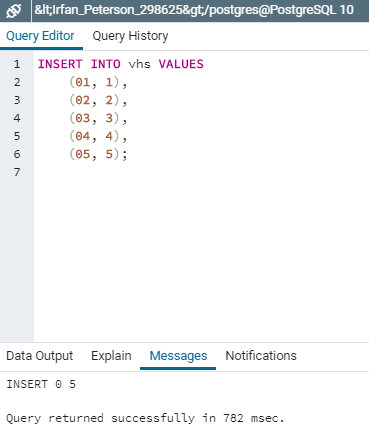
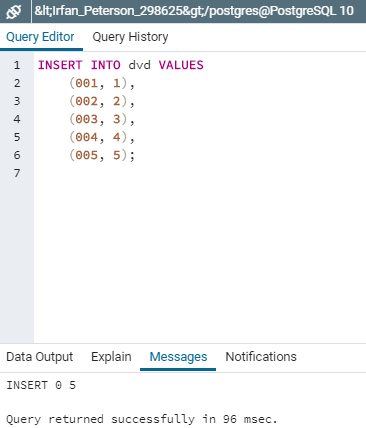
*Figure 13.* INSERT INTO SQL statement to populate data into Actor and Actress table.

**

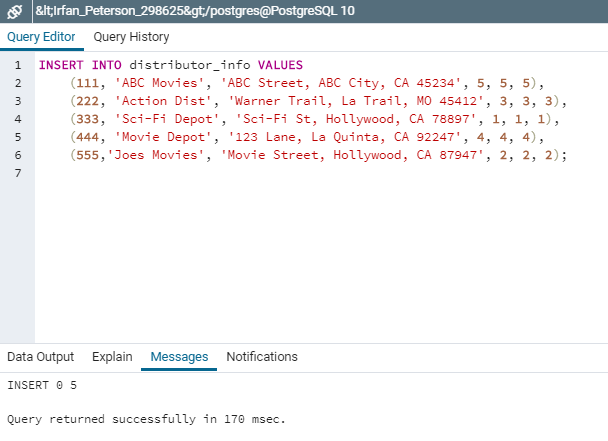
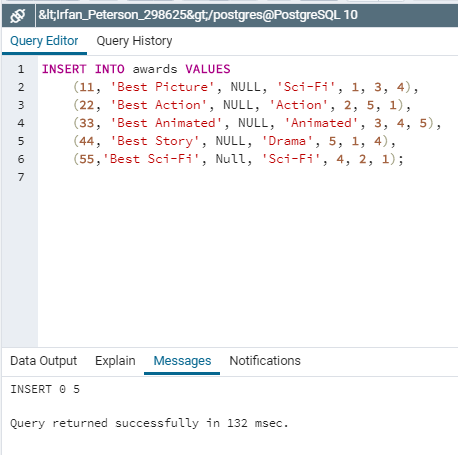
*Figure 14.* INSERT INTO SQL statement to populate data into Rental\_Transaction table.

* *

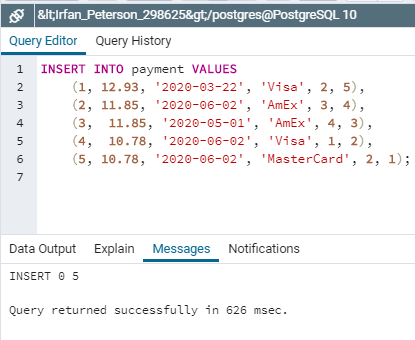
*Figure 14.* INSERT INTO SQL statement to populate data into Director and Movie\_Info table.

****

*Figure 15.* INSERT INTO SQL statement to populate data into DVD and VHS table.

**

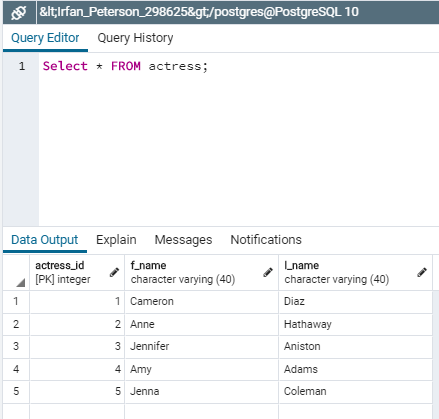
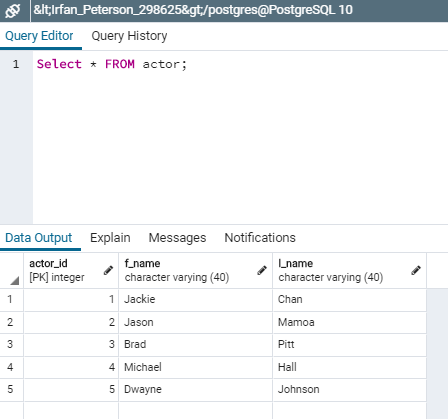
*Figure 16.* INSERT INTO SQL statement to populate data into Awards and Distributor\_Info table.

**

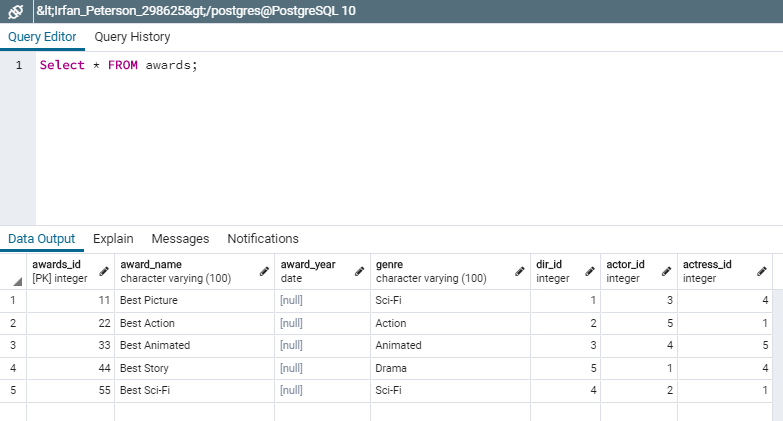
*Figure 17.* INSERT INTO SQL statement to populate data into Payment table.

**Executing Various Scripts to Perform Queries**

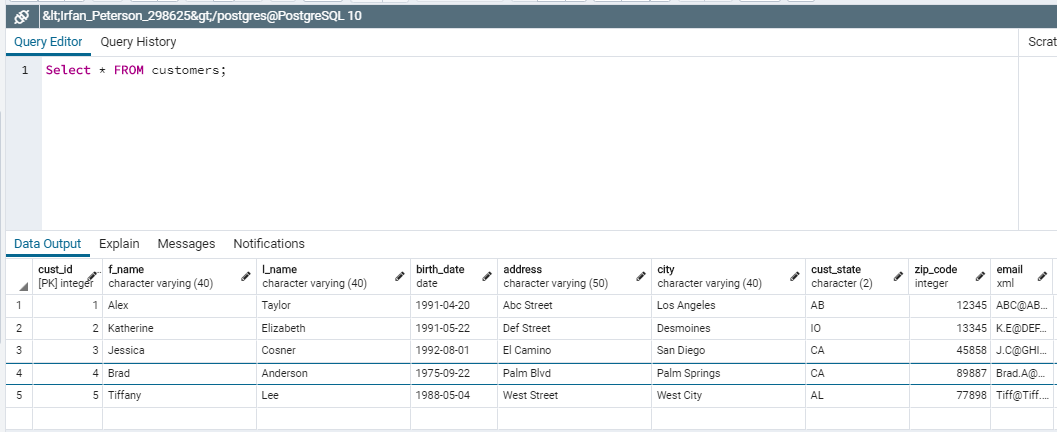
Our first query is to display all the tables that was populated. By using the Select \* From statement we can display the content of the tables:



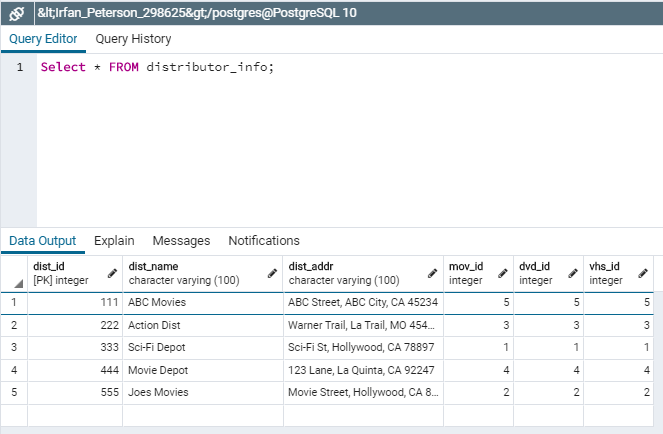
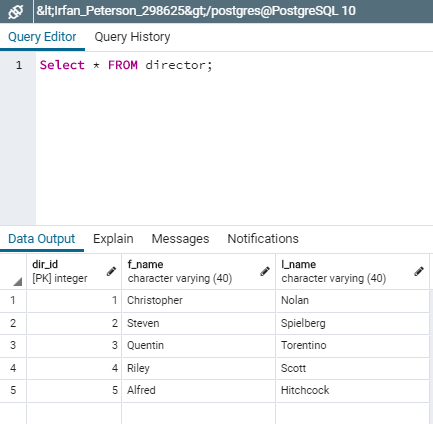
*Figure 18.* SELECT \* FROM SQL statement to display Actor and Actress tables.



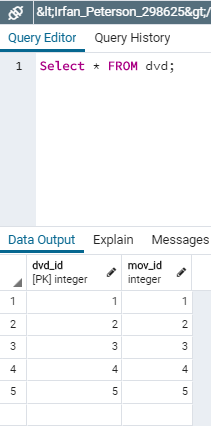
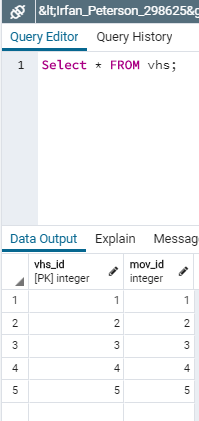
*Figure 19.* SELECT \* FROM SQL statement to display Awards table.

**

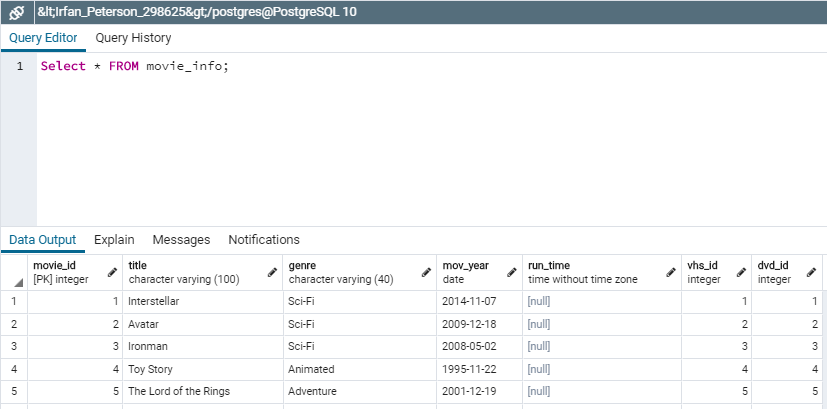
*Figure 20.* SELECT \* FROM SQL statement to display Customers tables.

**

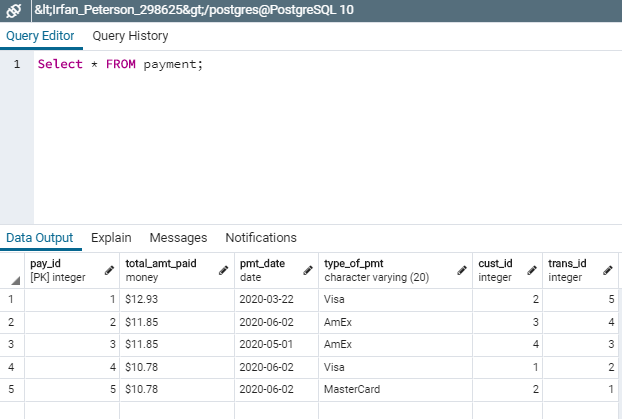
*Figure 21.* SELECT \* FROM SQL statement to display Director and Distributor\_Info tables.

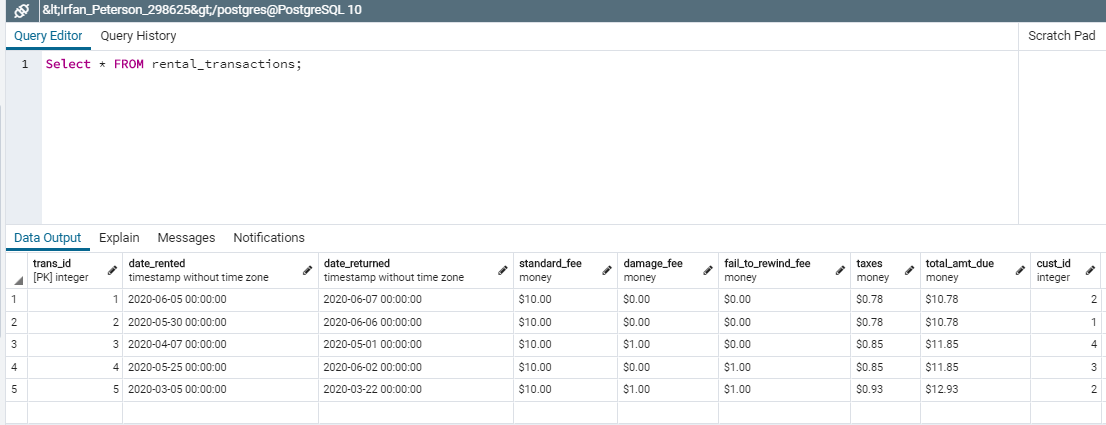
*Figure 22.* SELECT \* FROM SQL statement to display DVD and VHS tables.



*Figure 22.* SELECT \* FROM SQL statement to display Movie\_Info tables.

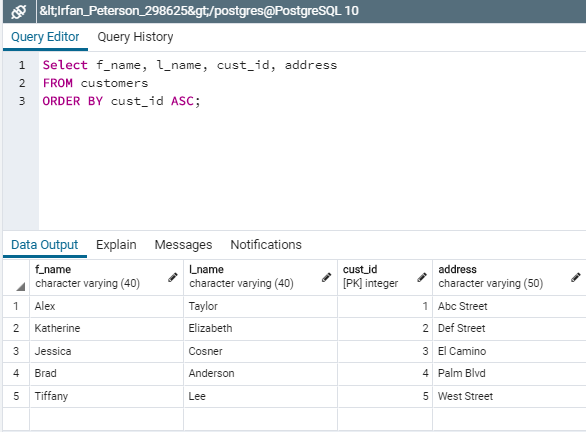


*Figure 22.* SELECT \* FROM SQL statement to display Payment table.



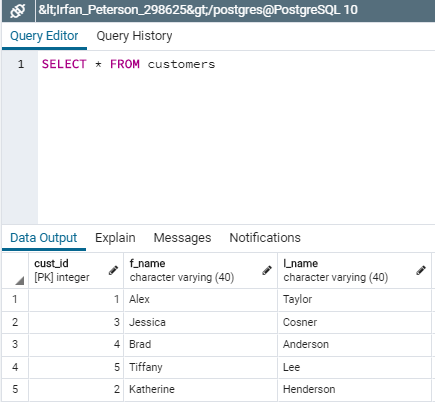
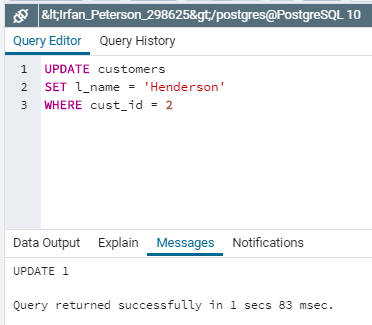
*Figure 23.* SELECT \* FROM SQL statement to display Rental\_Transaction table.

Figure 24 below shows the results of using a specific condition combined with the SELECT FROM ORDER BY statements to meet the query:



*Figure 24.* SELECT (Condition) FROM SQL statement to retrieve all of the customers' names, account numbers, and addresses, sorted by account number.

To update a customer maiden name we must use the the UPDATE SET WHERE script as seen on figure 25 below.



*Figure 25.* UPDATE SET WHERE SQL statement to change the last name of Katherine (id 2) from Elzabeth to Henderson.

Lastly, we can delete a row by using the DELETE FROM WHERE statement. Due to the customer row being aligned with foreign keys, the system did not allow me to delete an entry.

**Conclusion**

After configuring the entity-relationship diagram for the proposed Johnson Video Store database, extracting the metadata, creating the tables, populating them with data and ultimately using scripts to retrieve information, this project has taught me a lot about the structure and operations of a database. With only more practice, I will become more efficient in programing and designing SQL related databases.

References

Mannino, M. V. (2002). *Database application development and design*. Vancouver: B.C. College and Institute Library Services.