MET CS 669 Database Design and Implementation for Business Term Project: Online DVD Rental Business

**Nicholas Scala**

**June 2015**

Objective

This document serves as the initial design and implementation of a database for a DVD rental business similar to the DVD rental portion of the business pioneered by Netflix. This document includes a list of business rules, conceptual and logical ERDs, use case executions, and indexes.

Table of Contents

[1. Project Description & Business Requirements 3](#_Toc450133016)

[2. Business Rules 4](#_Toc450133017)

[3. Conceptual Entity Relationship Diagram 5](#_Toc450133018)

[4. Logical Entity Relationship Diagram 6](#_Toc450133019)

[5. Business Use Cases 7](#_Toc450133020)

[5.1 Query for All Movies by a Particular Director 7](#_Toc450133021)

[5.2 Query for All Active Customers and Plan Information 8](#_Toc450133022)

[5.3 Stored Procedure for Customer Adding a Movie to the Queue 9](#_Toc450133023)

[5.4 Query for All Movies a Customer Has Not Yet Rented 11](#_Toc450133024)

[5.5 Stored Procedure for Customer Cancelling Membership 12](#_Toc450133025)

[5.6 Query for Sold Out Movies 14](#_Toc450133026)

[5.7 Query for the Plan with the Most Customers Enrolled 15](#_Toc450133027)

[5.8 Query for Customers Who Rented Movie More Than Once 16](#_Toc450133028)

[5.9 Query for All Movies and Number of Unique Renters 17](#_Toc450133029)

[6. Indexes 18](#_Toc450133030)

[6.1 Customer ID Index 18](#_Toc450133031)

[6.2 Rental ID Index 19](#_Toc450133032)

[6.3 Customer State Index 20](#_Toc450133033)

[7. Appendix 21](#_Toc450133034)

[7.1 Plan\_Info Table Data 21](#_Toc450133035)

[7.2 Customer Table Data 21](#_Toc450133036)

[7.3 Fee Table Data 21](#_Toc450133037)

[7.4 Movie\_Director Table Data 22](#_Toc450133038)

[7.5 Movie\_Genre Table Data 22](#_Toc450133039)

[7.6 Movie Table Data 23](#_Toc450133040)

[7.7 Queue\_Item Table Data 24](#_Toc450133041)

[7.8 Copy Table Data 25](#_Toc450133042)

[7.9 Rental Table Data 26](#_Toc450133043)

# 1. Project Description & Business Requirements

This project is to design a database for a DVD rental business such as Netflix.com. The store rents DVDs online. The customer business model is illustrated below:



In order to rent a movie, a person must be enrolled at the online store. There are two different membership programs. As quoted from Netflix:

“With Netflix you can rent as many DVDs from the comfort of your home and have them delivered to your door in about 1 business day. There are no late fees and no due dates, and shipping is free both ways. Programs start at $11.99 plus any applicable tax. With our most popular program, 3-at-a-time, you can rent as many DVDs as you want for just $17.99 a month. You keep a revolving library of up to three DVDs at a time and can exchange them for new available DVDs as often as you like. Our 2-at-a-time program (limit 4 DVDs per month) is $11.99 plus any applicable tax per month.”  
(From http://www.netflix.com)

Below is an example use case to aid in understanding the Netflix operation:

1. A customer signs up for the 3-at-a-time program.
2. The same customer adds 10 movies to their queue.
3. Netflix mails to the customer the first three movies in their queue.
4. The customer watches and returns the first movie to Netflix.
5. Netflix mails out the next movie in the queue to the customer, which is the fourth movie added to the queue in Step #3.
6. The customer closes their account, but only returns two of the three movies the customer has at home.
7. Netflix charges the customer $25 for the missing movie.

The database will store membership information for each person, the movies she/he rented, movies in the queue to be rented, when were these movies returned, and so on. The rental history is used for two purposes:

* To give employees a basis to work from when they are asked what movie the customer has rented out or if it was lost in mail
* To determine if the movie was never mailed back by the customer

# 2. Business Rules

Below is a list of business rules used to create the initial database model derived from the problem statement described in [Part 1](#_1._Project_Description):

1. Each customer can subscribe to zero or one plans; each plan is subscribed to by zero to many customers.
2. Each customer can make zero to many rentals; each rental must be made by one and only one customer.
3. Each customer can be charged zero to many fees; each fee must be charged to one and only one customer.
4. Each rental must be of one and only one copy of a movie; each copy of a movie can belong to zero to many rentals.
5. Each customer can queue zero to many movies; each movie can be queued by zero to many customers.
6. Each copy is of one and only one movie; each movie can have zero to many copies.
7. Each movie must have one and only one genre; each genre can belong to zero to many movies.
8. Each movie must be directed by one and only one director; each director can direct zero to many movies.

# 3. Conceptual Entity Relationship Diagram

Below is a Conceptual ERD for the DVD rental database:



# 4. Logical Entity Relationship Diagram

Below is a Logical ERD for the DVD rental database including attributes, primary key and foreign key designations, and normalization:



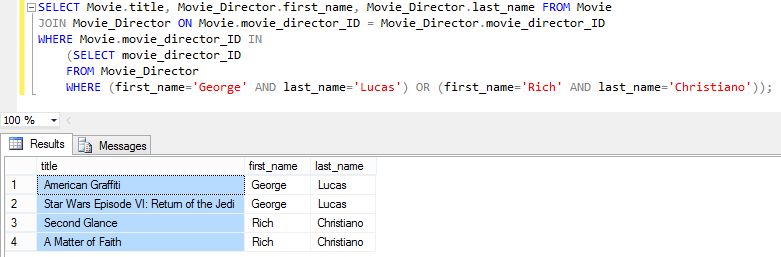
# 5. Business Use Cases

## 5.1 Query for All Movies by a Particular Director

**Use Case Description:**

A customer requests the titles of all movies that are directed by "George Lucas" or by “Rich Christiano”.

**Query Execution:**

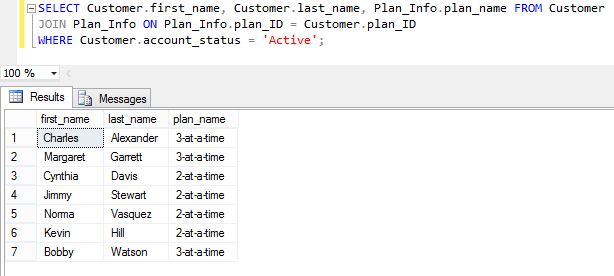


## 5.2 Query for All Active Customers and Plan Information

**Use Case Description:**

Management requests the names of all currently active customers, as well as the name of the current plan in which each of these customers is enrolled.

**Query Execution:**

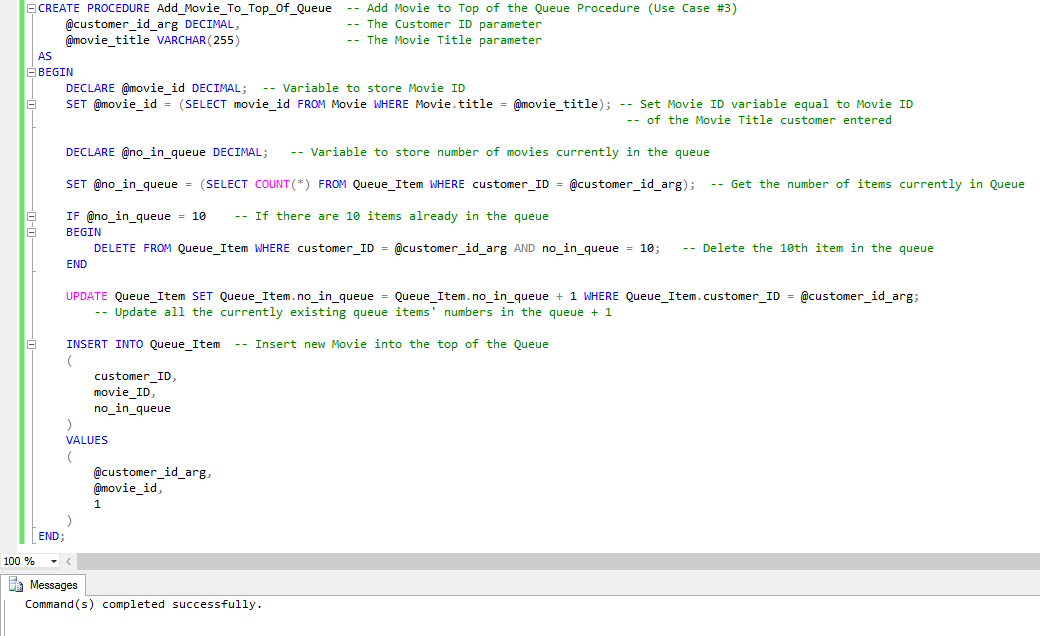


## 5.3 Stored Procedure for Customer Adding a Movie to the Queue

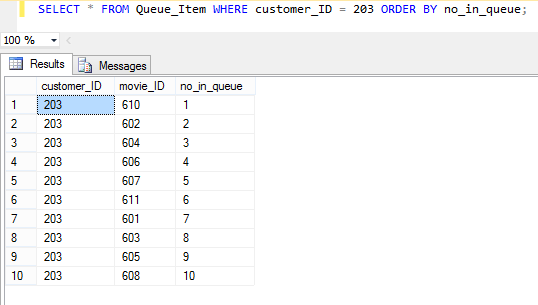
**Use Case Description:**

A customer wants to add a movie to their queue so that the newly added movie will be the next movie they receive.

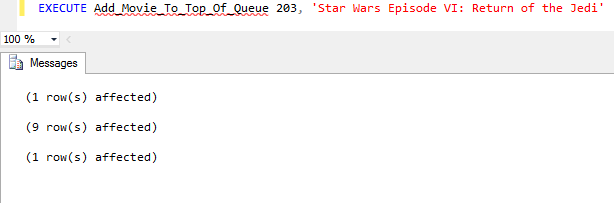
**Stored Procedure Creation:**



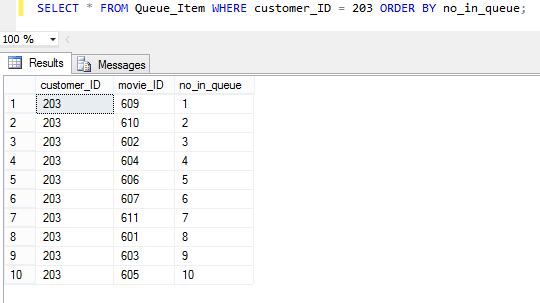
**Queue\_Item Table for Customer “203” – Before Execution:**



**Stored Procedure Execution:**



**Queue\_Item Table for Customer “203” – After Execution:**

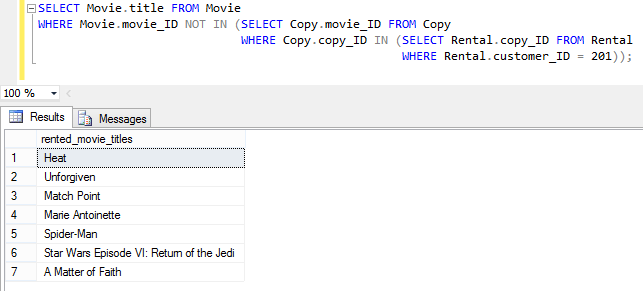


## 5.4 Query for All Movies a Customer Has Not Yet Rented

**Use Case Description:**

A customer requests the titles of all the DVDs that he or she has not rented.

**Query Execution:**

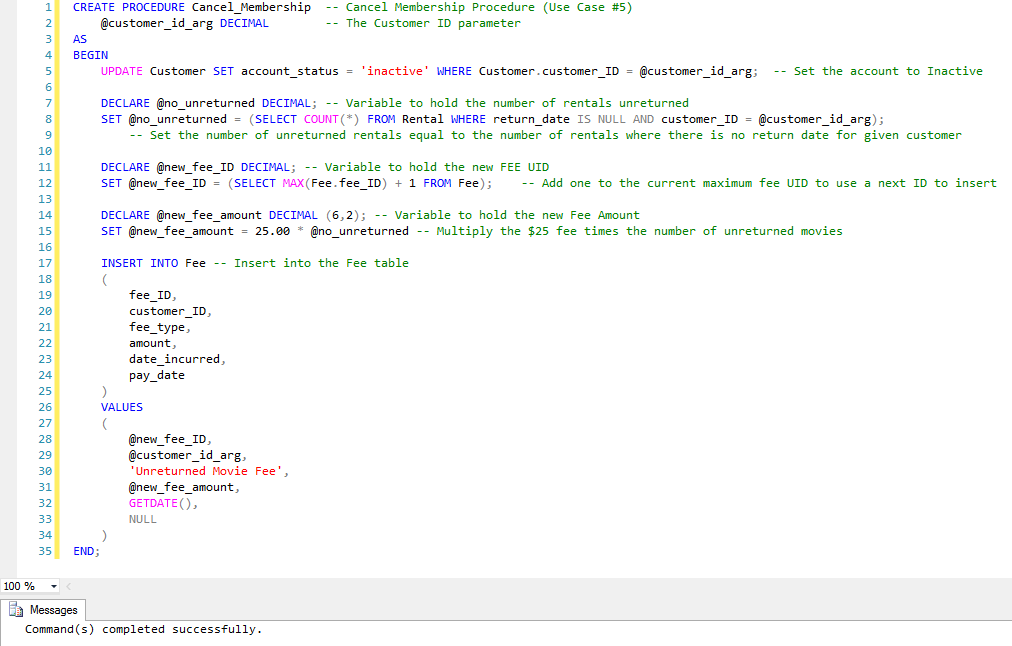


## 5.5 Stored Procedure for Customer Cancelling Membership

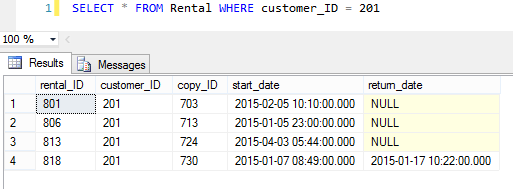
**Use Case Description:**

A customer cancels their membership and does not return a rented DVD, necessitating that a $25 DVD replacement fee be charged to their account. When a customer cancels their membership, they become inactive, but their DVD queue and rental history remains in the database, in the event they return as a customer.

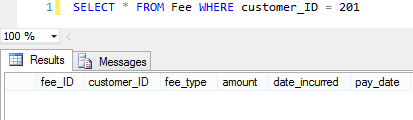
**Stored Procedure Creation:**



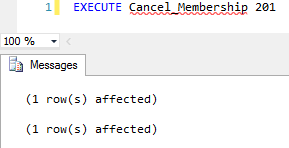
**Rental Table Showing Three Unreturned Movies for Customer “201”:**



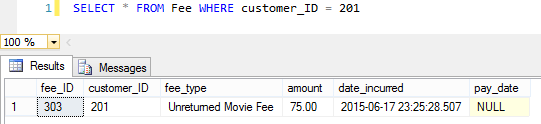
**Fee Table for Customer “201” – Before Execution:**

****

**Stored Procedure Execution:**

****

**Fee Table for Customer 201 – After Execution**

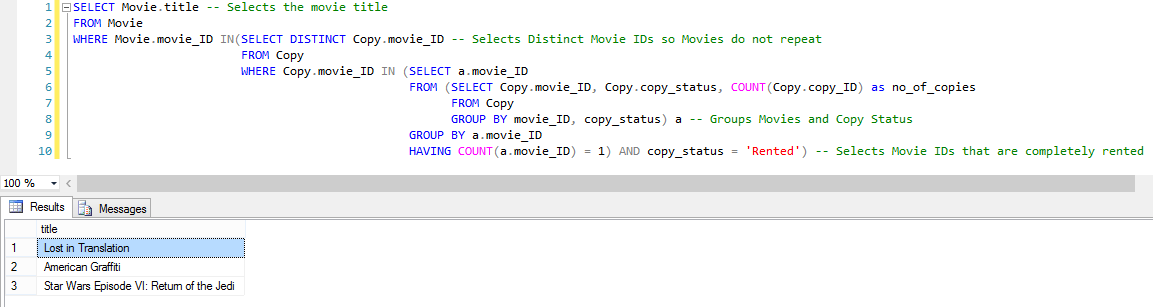
****

## 5.6 Query for Sold Out Movies

**Use Case Description:**

Management requests the names of all movies that are currently sold out. A movie is sold out if all copies of the movie are currently rented and not yet returned.

**Query Execution:**

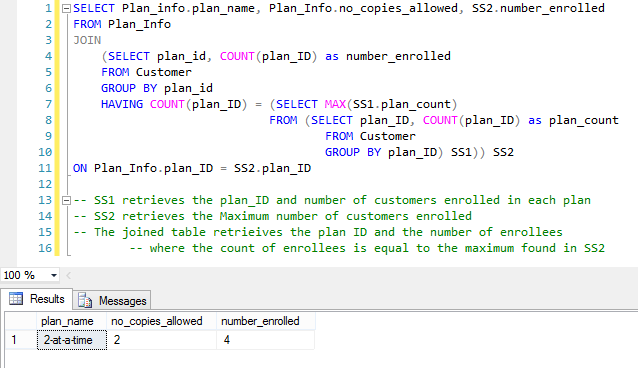


## 5.7 Query for the Plan with the Most Customers Enrolled

**Use Case Description:**

Management requests identification of the plan with the most customer enrollees, and for that plan, the name, number of DVDs allowed at one time, and the number of customer enrollees.

**Query Execution:**

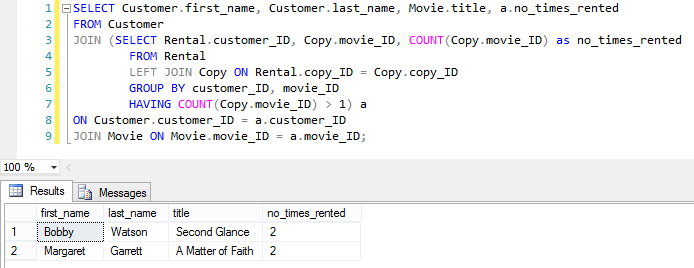


## 5.8 Query for Customers Who Rented Movie More Than Once

**Use Case Description:**

Management requests the names of all customers, and for each customer, the titles of the movies that they rented multiple times. For each title, management would like to see the number of times it was rented by the customer, only including titles that the customer rented more than once. If a customer has no rentals, or did not rent any movies multiple times, management does not want to see them in the list.

**Query Execution:**

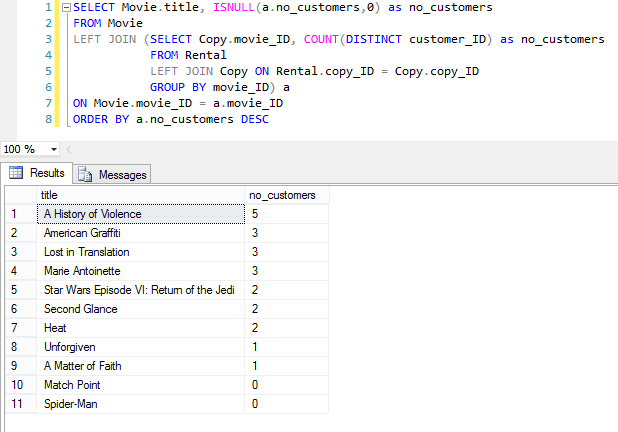


## 5.9 Query for All Movies and Number of Unique Renters

**Use Case Description:**

Management requests the titles of all movies, and for each movie, the number of different customers that rented the movie. They would like the list to be ordered from the highest number of different rentals to the lowest number. Multiple rentals of the same movie by the same customer only count as one unique rental. Management is interested in the number of different customers that rented the movie, but not whether the same customer rented the same movie more than once.

**Query Execution:**



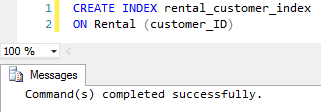
# 6. Indexes

## 6.1 Customer ID Index

**Index Description:**

The first index is on the customer\_ID foreign key in the Rental table. It is obvious that Netflix would be very interested in which customers belong to which rentals. It is likely that Netflix leadership and customers will utilize several queries (as evident by the use cases above) that involve tracking the customer to rentals. For example, Netflix leadership would want to track the activeness of each customer to incentivize those that use the system less or reward more frequent customers. Customers, on the other hand, would be very interested in seeing their rental history. Indexing this column would help speed up these queries because instead of scanning the entire table for a particular set of customer\_IDs, the query can just look to the index. Besides being a foreign key, customer\_ID makes a strong candidate for an index because the Rental table will contain a vast amount of data and the customer\_ID data will exhibit high sparsity. It is a non-unique index because customer\_ID will repeat many times in the Rental table when the customer rents more than one movie.

**Index Execution:**

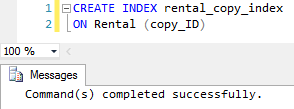


## 6.2 Rental ID Index

**Index Description:**

The second index is on the copy\_ID foreign key in the Rental table. Similarly, to related customer data, Netflix would be very interested in which copies belong to which rentals. It is likely that Netflix leadership will utilize several queries (as evident by the use cases above) that involve tracking the copy to rentals. For example, Netflix leadership may want to replace a particular copy of a movie if it has been rented extensively and physically worn down. Indexing this column would help speed up these queries because instead of scanning the entire table for a particular set of copy\_IDs, the query can just look to the index. Besides being a foreign key, copy\_ID makes a strong candidate for an index because the Rental table will contain a vast amount of data and the copy\_ID data will exhibit high sparsity. It is a non-unique index because copy\_ID will repeat many times in the Rental table when a particular copy is rented by many customers throughout its lifecycle.

**Index Execution:**

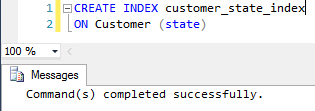


## 6.3 Customer State Index

**Index Description:**

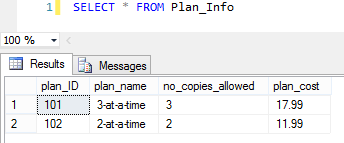
The third index is on the state attribute of the Customer table. Netflix leadership and marketing teams would be interested in which states their customers live in. It is likely that Netflix leadership will utilize several queries that involve customer demographics. For example, Netflix leadership may want to advertise more in regions where there are less customers in order to gain customers for that region. Indexing this column would help speed up these queries because instead of scanning the entire table for a particular set of states, the query can just look to the index. State makes a strong candidate for an index because the Customer table will contain a vast amount of data and state data will exhibit high sparsity. It is a non-unique index because states will repeat many times in the Customer table when there are multiple customers from the same state.

**Index Execution:**

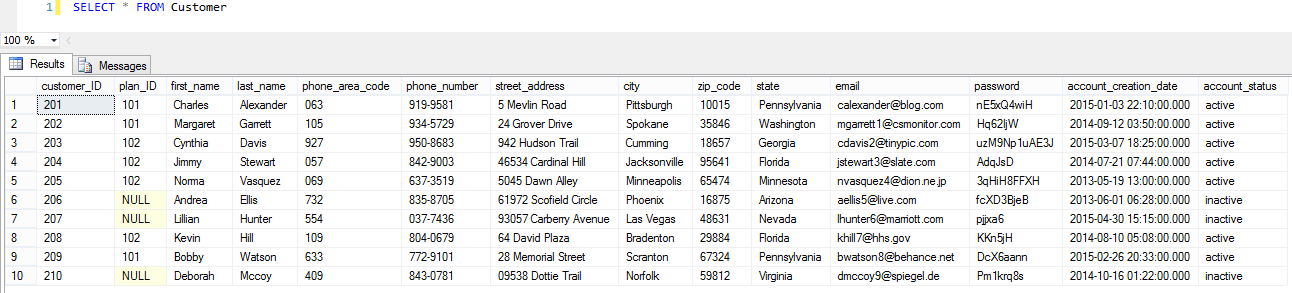


# 7. Appendix

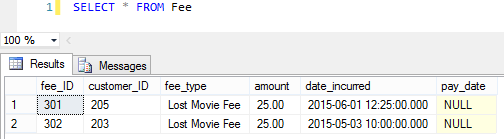
## 7.1 Plan\_Info Table Data



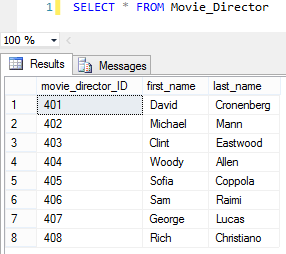
## 7.2 Customer Table Data



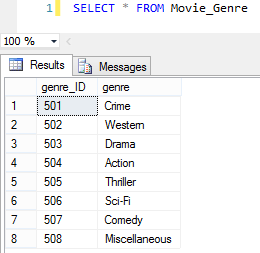
## 7.3 Fee Table Data



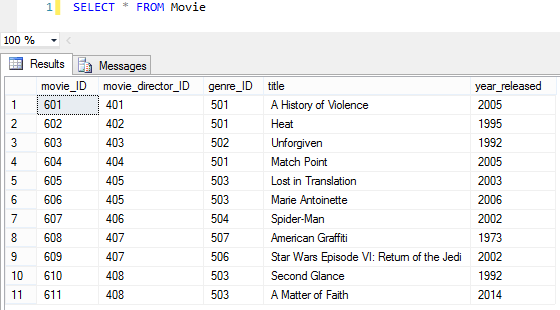
## 7.4 Movie\_Director Table Data



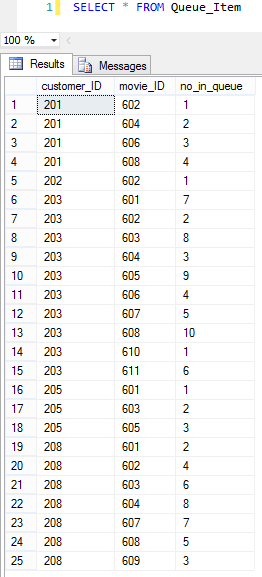
## 7.5 Movie\_Genre Table Data



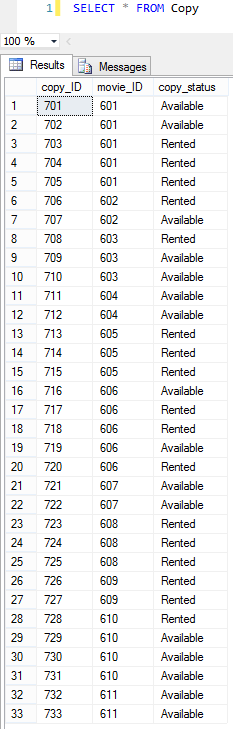
## 7.6 Movie Table Data



## 7.7 Queue\_Item Table Data



## 7.8 Copy Table Data



## 7.9 Rental Table Data

