|  |
| --- |
| **Overview of the Lab** |
| Sometimes we are interested in the result of aggregating multiple data items rather than in individual data items. For example, a store may be interested in the monetary amount of a single sale, but may be equally or more interested in the sum the monetary amount of all sales that occurred on a specific day. SQL provides many useful ways to aggregate data. One objective of this lab is for you to learn to aggregate data using SQL.  Other objectives include learning to normalize a schema’s tables to BCNF, and learning to visualize SQL results with data visualizations.  From a technical perspective, together, we will learn:   * how to use aggregate functions generally. * how to count and add items in a table. * how to determine minimum and maximum values. * how to filter rows based upon aggregate values. * how to use aggregation with joins together to answer more complex use cases with related data. * how to visualize SQL results with basic data visualizations. |

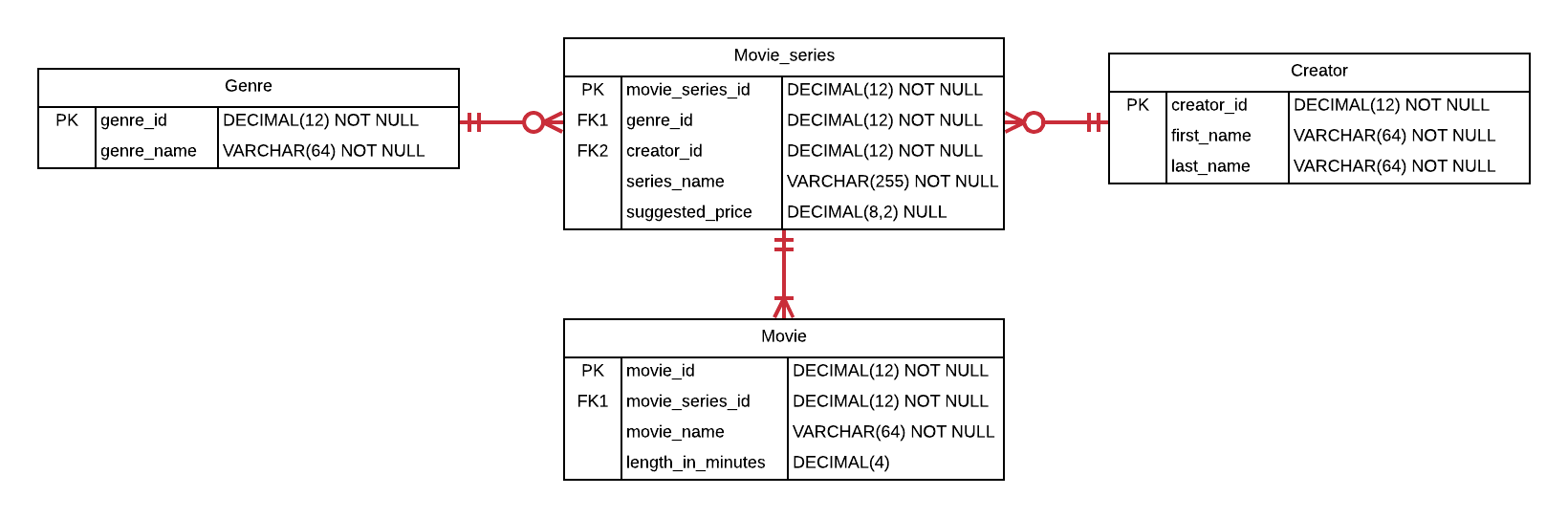
|  |
| --- |
| **Lab 3 Explanations Reminder** |
| As a reminder, it is important to read through the Lab 3 Explanation document to successfully complete this lab, available in the assignment inbox alongside this lab. The explanation document illustrates how to correctly execute each SQL construct step-by-step, and explains important theoretical and practical details. |

|  |
| --- |
| **Other Reminders** |
| * The examples in this lab will execute in modern versions of Oracle, Microsoft SQL Server, and PostgreSQL as is. * The screenshots in this lab display execution of SQL in the default SQL clients supported in the course – Oracle SQL Developer, SQL Server Management Studio, and pgAdmin – but your screenshots may vary somewhat as different version of these clients are released. * Don’t forget to commit your changes if you work on the lab in different sittings, using the “COMMIT” command, so that you do not lose your work. |

**Section One – Aggregating Data**

**Section Background**

To practice aggregating data, you will be working with the following simplified Movie Series schema.



This schema contains basic information about various movie series and the movies that comprise them, such as the Star Wars series with its movies.

In this schema, the Movie\_series table represents the overall movie series, and contains a primary key, the name of the series, foreign keys to its genre and creator, and a suggested price for the entire series. The Genre table represents the genre of a movie such as “Fantasy”, “Family Film”, and the like. It contains a primary key and the name of the genre. The Creator table represents who created the series, and contains a primary key and the name of each creator. The Movie table represents movies that comprise each movie series, and contains a primary key, a foreign key to the movie’s series, the name of the movie, and the length of the movie, in minutes.

The schema is intentionally simplified compared to what you might see in a real-world production schema. Many attributes and entities that would exist in a production database are not present. Nevertheless, there is sufficient complexity in the existing relationships and attributes to challenge you to learn various aggregation scenarios you encounter in real-world schemas.

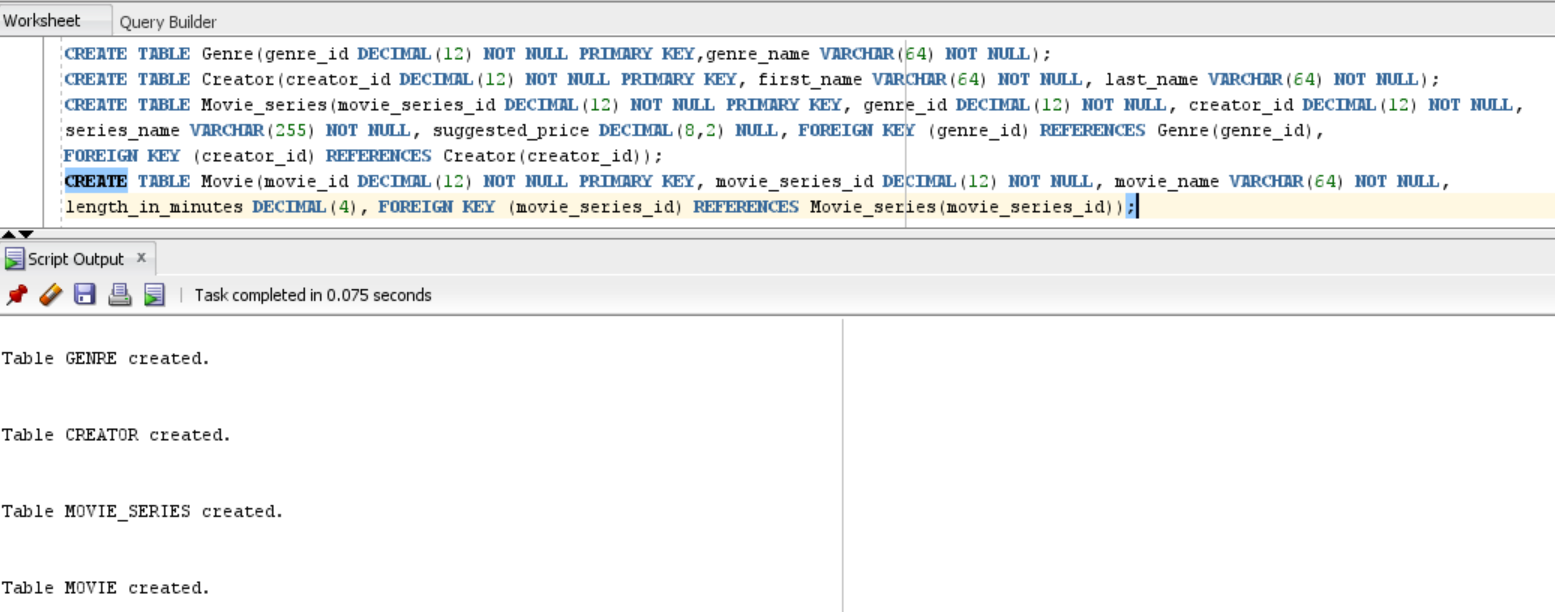
As a reminder, for each step that requires SQL, make sure to capture a screenshot of the command and the results of its execution. *Further, make sure to eliminate unneeded columns from the result set, to name your columns something user-friendly and human readable, and to format any prices as currencies.*

**Section Steps**

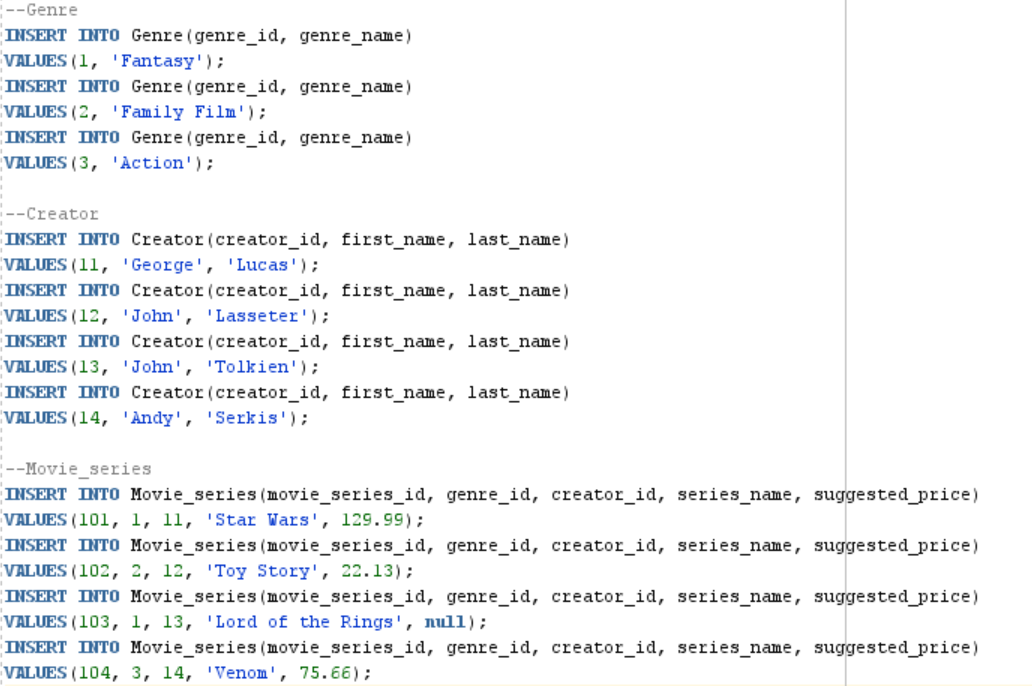
1. *Creating Table Structure and Data* – Create the tables in the schema, including all of their columns, datatypes, and constraints, and populate the tables with data. Most but not all of the data is given to you in the table below; *you should also insert information for one additional movie series of your choosing.* Although the data is in flattened representation below, you will of course insert the data relationally into the schema with foreign keys referencing the appropriate primary keys.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Genre** | **Creator** | **Series** | **Suggested Price** | **Movie** | **Length** |
| Fantasy | George Lucas | Star Wars | $129.99 | Episode I: The Phantom Menace | 136 |
| Fantasy | George Lucas | Star Wars | $129.99 | Episode II: Attack of the Clones | 142 |
| Fantasy | George Lucas | Star Wars | $129.99 | Episode III: Revenge of the Sith | 140 |
| Fantasy | George Lucas | Star Wars | $129.99 | Episode IV: A New Hope | 121 |
| Family Film | John Lasseter | Toy Story | $22.13 | Toy Story | 121 |
| Family Film | John Lasseter | Toy Story | $22.13 | Toy Story 2 | 135 |
| Family Film | John Lasseter | Toy Story | $22.13 | Toy Story 3 | 148 |
| Fantasy | John Tolkien | Lord of the Rings |  | The Lord of the Rings: The Fellowship of the Ring | 228 |
| Fantasy | John Tolkien | Lord of the Rings |  | The Lord of the Rings: The Two Towers | 235 |
| Fantasy | John Tolkien | Lord of the Rings |  | The Lord of the Rings: The Return of the King | 200 |

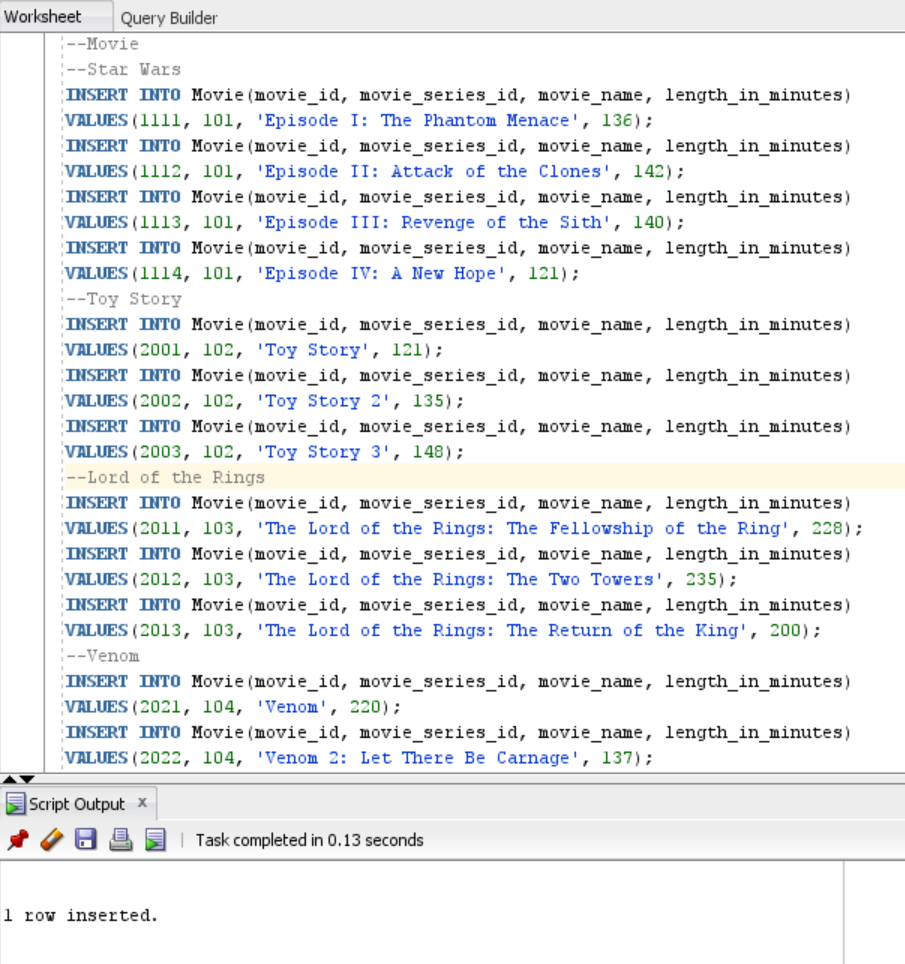
Note that the suggested price for the Lord of the Rings series is null (has no value).



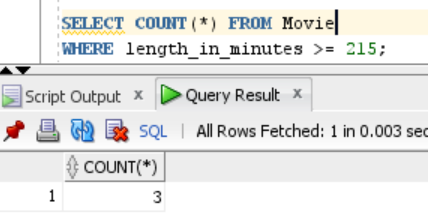
Inserting values into Genre, Creator, and Movie\_series tables:

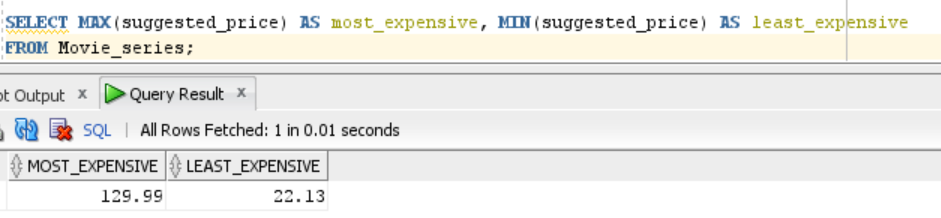


Inserting values into the Movie table:



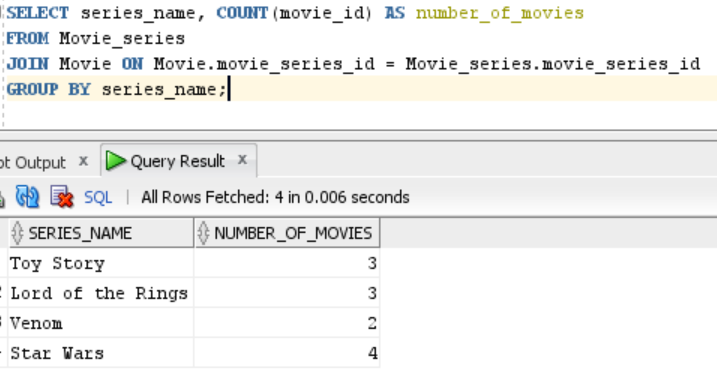
1. *Counting Matches* – A video reseller needs to know how many movies are available that are at least two hours and fifteen minutes long. Write a single query to fulfill this request.

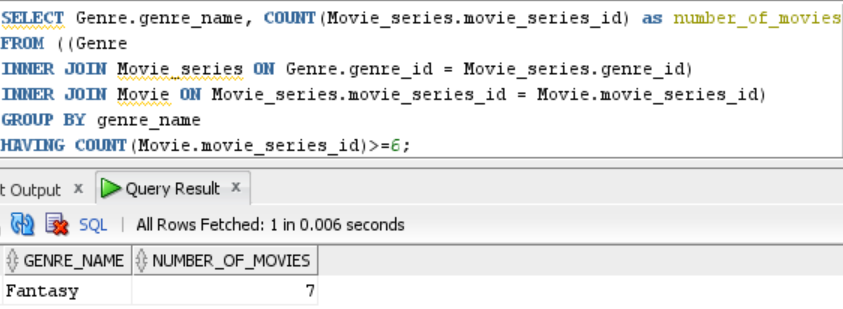


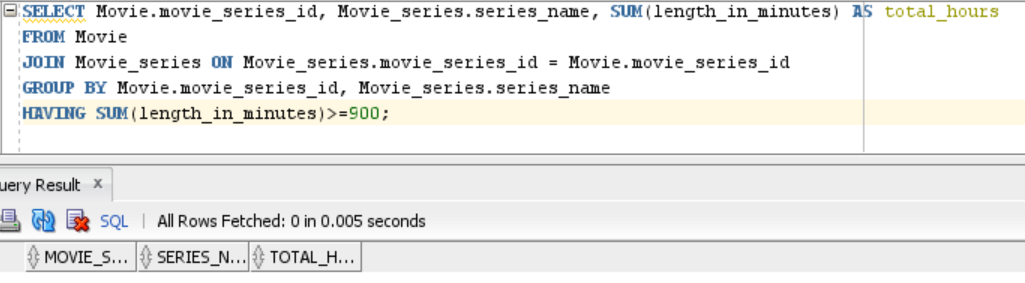
1. *Determining Highest and Lowest –* The same video reseller needs to know the price of the most expensive and least expensive series. Write a single query that fulfill this request. Explain how and why the SQL processor treated the suggested price for the Lord of the Rings series differently than the other suggested price values.  
   

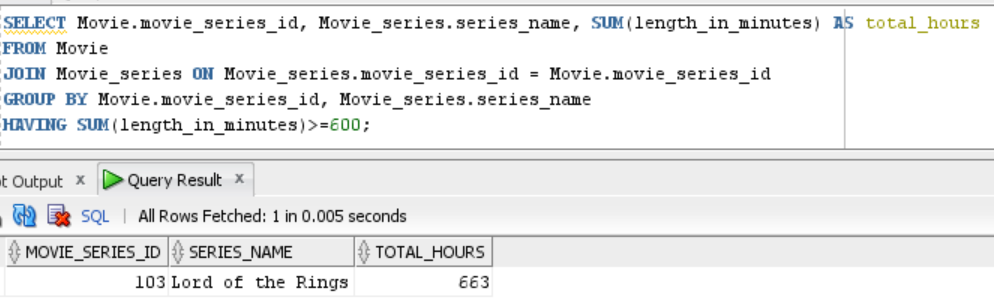
The SQL query uses MIN and MAX functions to return the prices of the least expensive series and the most expensive series, respectively. The lowest value in the suggested\_price column $22.13 maps to the Toy Story movie series and the highest value in the suggested\_price column $129.99 maps to Star Wars series. The suggested\_price column has no value for the Lord of the Rings series. MIN and MAX functions do not evaluate null values, so the null value present for the Lord of the Rings series is overlooked. Some aggregate functions may order null values at the start or end of the result or allow the querying user to decide how null values are evaluated.

1. *Grouping Aggregate Results –* A film production company is considering purchasing the rights to extend a series, and needs to know the name of each movie series, along with the number of movies in each series. Write a single query to fulfill this request.

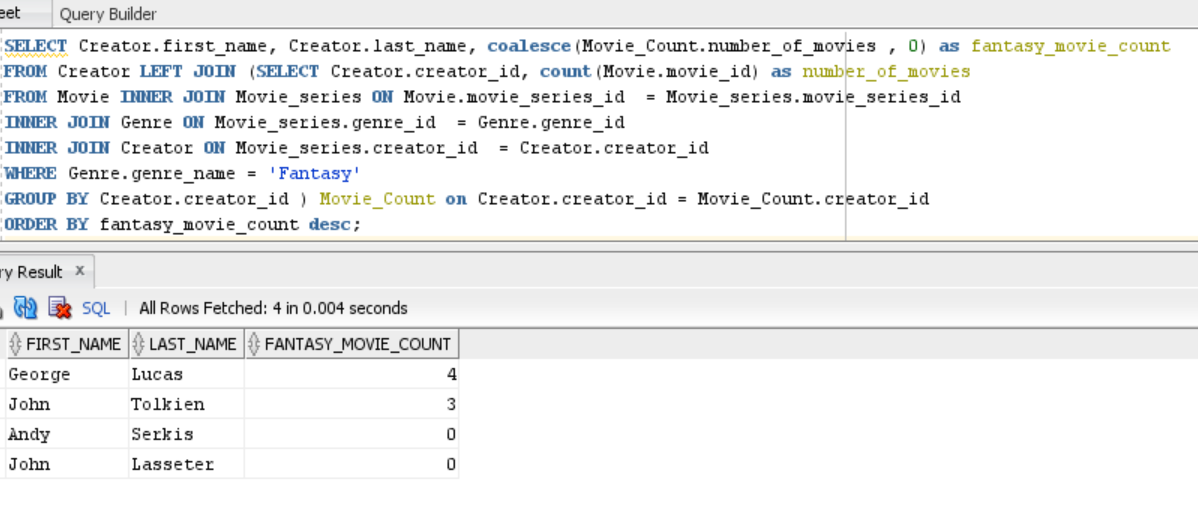


1. *Limiting Results by Aggregation* – A same film production company wants to search for genres that have at least 6 associated movies. Write a single query to fulfill this request, making sure to list only genres that have at least 6 movies, along with the number of movies for the genre.   
   
2. *Adding Up Values –* Boston University wants to offer its students a movie-binge weekend by playing every movie in a series. To make sure the series is as bingeable as possible, BU wants to be sure the series will run for at least 9 hours. Write a single query that gives this information, with useful columns.

  
In my table, none of the series run for at least 9 hours. Instead, I have written a query to provide series that will run for at least 6 hours.



1. *Integrating Aggregation with Other Constructs –* A research institution requests the names of all movie series’ creators, as well as the number of “Fantasy” movies they have created (even if they created none). The institution wants the list to be ordered from most to least; the creator who created the most fantasy films will be at the top of the list, and the one with the least will be at the bottom. Write a single query that gives this information, with useful columns.



**Section Two –****Data Visualization**

**Section Background**

Data visualization is presenting information in visual form, commonly with charts and graphs. People are adept at recognizing patterns, trends, and differences visually. Visual data stories are understood accurately and quickly; recognition comes much more slowly with pages and pages of text and tables.

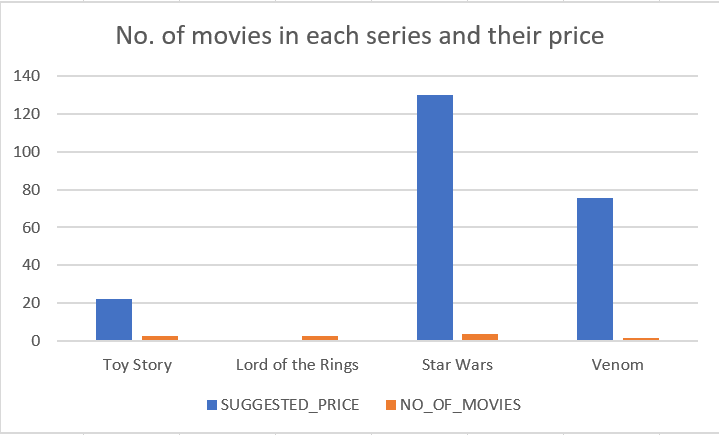
In the modern age of data driven decision-making, data stories are important for any field – sales, finance, human resources, engineering, information technology, just to name a few. Conveying those data stories effectively is just as important. If you can design and implement effective databases, and also build visualizations from your database to tell data stories, you will have a skillset desired by organizations worldwide. In this section, you have a chance to visualize data by writing queries to obtain results, and using those results to create commonly used charts.

**Section Steps**

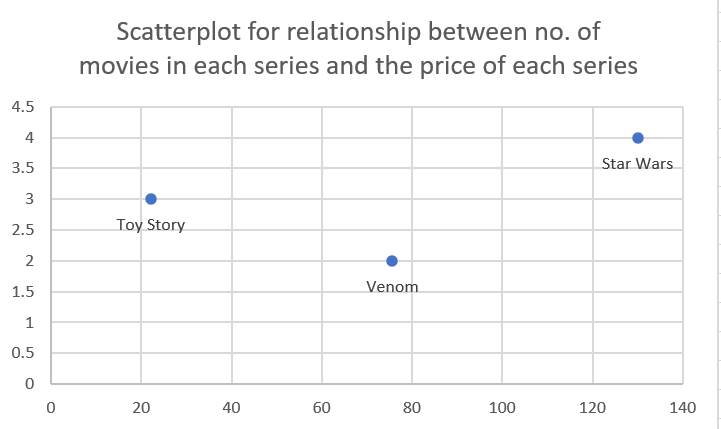
1. *Visualizing Data with One or Two Measures –* SQL results were obtained in #4, in particular the name each movies series along with total number of movies in each series. To address this step, you will need to expand this SQL to also include the suggested price of each series. The SQL will retrieve the name, suggested price, and the total number of movies in each move series. Use these results to address the following.
   1. Create a bar chart with the series’ name as one axis, and the series’ price as another axis. Explain the story this visualization describes.

The following bar chart illustrates the price and number of movies in each movie series. The blue bars represent the price of each series, and the orange bars represent the number of movies in each series.

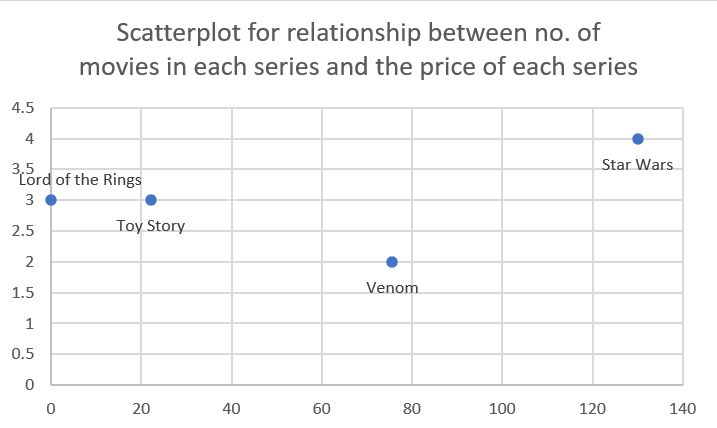
* The Toy Story series costs around $20 as depicted by the blue bar on the chart. The Lord of the Rings series does not show the blue bar as the price is not determined (has a null value). The Star Wars series costs the most, so its bar is taller compared to the other series. The Venom series is the second costliest series in the list.
* The orange bar for Star Wars is slightly higher than the rest of the series, meaning the series has the greatest number of movies compared to the other. On the flip side, the orange bar is at the lowest for Venom, meaning the Venom series has the least number of movies compared to the rest of the series in the list.



* 1. Create a scatterplot with the series’ price as one axis, and the number of movies in the series as another axis. Ensure that each series is labeled with its name, either directly or with a legend. Explain the story this visualization describes.



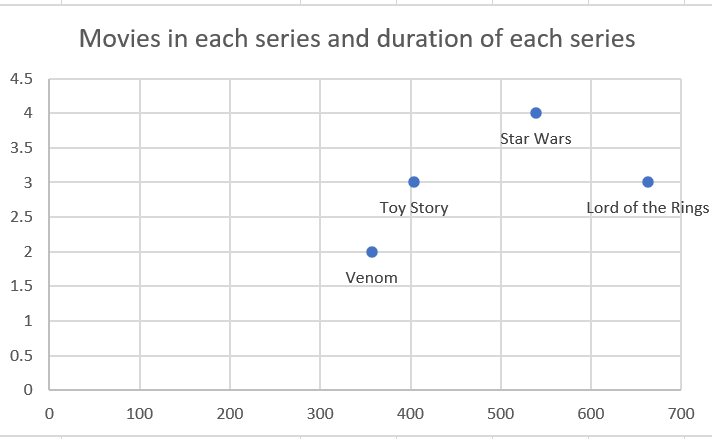
The chart above uses scatter plots to illustrate relationship between number of movies in each series and the price of each series. It can be observed that the Star Wars series is doing better compared to the other series in terms of the price and the number of movies. The Toy Story seems to be at its lowest compared to Star Wars and Venom in terms of price and the Venom series seems to be at its lowest compared to Star Wars and Toy Story in terms of number of movies. However, the Lord of the Rings has not been plotted as its price is NULL or N/A. Scatterplot displays variables with NULL values as a gap. The value can be changed to be zero to plot the graph as shown in the following figure. However, this is incorrect as it takes the price of the Lord of the Ring series as 0. In reality, the Lord of the Ring series may cost more than the Star Wars series.



1. *Another Data Visualization –* Create a visualization of your choosing for data in the Movie schema. The visualization should tell a useful story. If you find that you need more movies in the schema to tell the story well, feel free to add them. Make sure to explain the data story, and to explain why you chose that particular chart or visualization.  
   The following table lists the series name, the duration of each series, and the number of movies in each series.

|  |  |  |
| --- | --- | --- |
| **SERIES NAME** | **TOTAL HOURS** | **NUMBER OF MOVIES** |
| Toy Story | 404 | 3 |
| Star Wars | 539 | 4 |
| Lord of the Rings | 663 | 3 |
| Venom | 357 | 2 |

The data visualization chosen to depict this data is scatterplot as it provides the relationship between two variables: duration and number of movies. It can be observed that Star Wars has the most movies (4 movies) compared to other series and Lord of the Rings is the lengthiest series with its duration close to 7 hours. Toy Story and Lord of the Rings seem to have the same number of movies in their series, but Toy Story series are shorter in duration compared to Lord of the Rings. Venom has the least number of movies (2 movies), hence is shortest in duration (< 4 hours) compared to all the other series.



# Evaluation

Your lab will be reviewed by your facilitator or instructor with the criteria outlined in the table below. Note that the grading process:

* involves the grader assigning an appropriate letter grade to each criterion.
* uses the following letter-to-number grade mapping – A+=100,A=96,A-=92,B+=88,B=85,B-=82,C+=88,C=85,C-=82,D=67,F=0.
* provides an overall grade for the submission based upon the grade and weight assigned to each criterion.
* allows the grader to apply additional deductions or adjustments as appropriate for the submission.
* applies equally to every student in the course.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Criterion** | **A** | **B** | **C** | **D** | **F** |
| **Section 1: Quality (70%)** | The results for all steps in Section 1 are complete and correct. Appropriate SQL constructs have been used for all steps, and supporting explanations are present and accurate. All screenshots in Section 1 are legible. The section is well organized. All supporting explanations are clear and understandable. | The results for most steps in Section 1 are complete and correct. The appropriate SQL constructs have been used for most steps, and supporting explanations are mostly present and accurate. Most screenshots in Section 1 are legible. The section is organized. Most supporting explanations are clear and understandable. | The results for some steps in Section 1 are complete and correct. Appropriate SQL constructs have been used for some steps, and some supporting explanations are present and accurate Some screenshots in Section 1 are legible. Some supporting explanations are clear and understandable. | The results for most steps in Section 1 are incomplete or incorrect. Appropriate SQL constructs have not been used for most steps. The screenshots in Section 1 are mostly illegible or missing. Most supporting explanations are unclear or missing. The section is disorganized. | The results for virtually all steps in Section 1 are incomplete or incorrect. Appropriate SQL constructs have not been used. Virtually all screenshots in Section 1 are illegible or missing. Virtually all supporting explanations are unclear or missing. The section is disorganized. |
| **Section 2 Presentation (20%)** | The visualizations present the SQL results entirely accurately. The visualizations are labeled well, use appropriate ranges, and are clearly understood. | The visualizations present the SQL results mostly accurately. The visualizations are labeled, use reasonable ranges, and are mostly clear. | The visualizations present the SQL results somewhat accurately. The visualizations are partly labeled, use somewhat reasonable ranges, and are somewhat clear. | The visualizations present the SQL results mostly inaccurately. The visualizations may not be labeled well, may not use reasonable ranges, and may be unclear. | The visualizations are missing, or represent the SQL results entirely inaccurately. The visualizations may not be labeled, may not use ranges, and may be entirely unclear. |
| **Section 2 Data Stories (10%)** | The data stories given are entirely clear and useful. The data stories accurately characterize the visualizations. | The data stories given are mostly clear and useful. The data stories mostly characterize the visualizations. | The data stories given are somewhat clear and useful. The data stories somewhat characterize the visualizations. | The data stories given are mostly unclear and not useful. The data stories do not characterize the visualizations well. | The data stories are missing, or are entirely unclear and not useful. The data stories do not characterize the visualizations. |

Use the **Ask the Teaching Team Forum** if you have any questions regarding how to approach this lab. Make sure to include your name in the filename and submit it in the *Assignments* section of the course.