Database Queries

Overview:

This section contains information on how I created a database and also the SQL queries used.

To create my initial tables I used DB Browser for SQLite.
Below, you will see an example of how I added data to each table.

How to creates a table. Example of Customers table.

```
CREATE TABLE Customers (
    ID INT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Address VARCHAR(100)
);
```

- CREATE TABLE Customers (); - Opening statement for creating table, instructions for fields are placed within the parenthesis.
    - ID INT PRIMARY KEY - The ID field is a primary key and is of Int datatype.
    - FirstName, LastName, Address, are all text fields.
    - ; Is used to end a Query

Adding data to Customers table

- I used ChatGPT for this so I wouldn't have to type out a bunch of data.

```
INSERT INTO Customers (ID, FirstName, LastName, Address)
VALUES
    (101, 'Skyla', 'Scrumplesworth', '2113 Dimble Street'),
    (102, 'Finnigan', 'Whistlebop', '2045 Dimble Street'),
    (103, 'Lila', 'Scrumplesby', '9034 Scrumple Lane'),
    (ect…  );
```

- INSERT INTO Customers (ID, FirstName, LastName, Address) VALUES - opening statement for inserting values into the Customers table. The (ID, FirstName, LastName, Address) are the fields.

- The (101, 'Skyla', 'Scrumplesworth', '2113 Dimble Street") and data that comes after is the data to be inserted into the table. 101 would be the ID, Skyla would be the FirstName and so on. Multiple entries can be made at the same time by separating each tuple with a ",".

SQL Queries:

Now that we have added all the necessary data to our database, we can start making some queries.

SELECT Statements:

1. **Selecting all customers first and last names who have a rental out:**

```
SELECT FirstName, LastName FROM Customers
JOIN Rentals ON Rentals.CustomerID = Customers.ID
WHERE Returned == "False";
```

- SELECT FirstName, LastName FROM Customers - Select the FirstName and LastName columns from Customers

- JOIN Rentals ON Rentals.CustomerID = Customers.ID - Join the Rentals Table to Customers table. The ID column in customers matches with the CustomerID in Rentals.

- WHERE Return == "False"; - Only entries that have "False" in the Returned field will be displayed. ";" is the end of the Query. This successfully displays a list of first and last names of customers who haven't returned a rented item.

2. **Selecting rental counts of media items that are DVDs**

```
SELECT COUNT(MediaItemTitle), MediaItemTitle FROM Media
JOIN Rentals ON Rentals.MediaID = Media.MediaID
WHERE MediaItemCategory == 1
GROUP BY MediaItemTitle;
```

- SELECT COUNT(MediaItemTitle), MediaItemTitle From Media - This will create two columns, one will be the number of times the MediaItemTitle occurs and the other is the MediaItemTitle itself.
- JOIN Rentals ON Rentals.MediaID = Media.MediaID - Connecting Rentals table to Media Table.
- WHERE MediaItemCategory == 1 - Only show media items that are DVDs
- GROUP BY MediaItemTitle. - This groups the counts by MediaItemTItle. Essentially this counts how many times a DVD has been rented.

3. **Selecting the top three renters:**

```
SELECT COUNT(ID), FirstName, LastName FROM Customers
JOIN Rentals ON Rentals.CustomerID = Customers.ID
GROUP BY FirstName
ORDER BY COUNT(ID) DESC
LIMIT(3);
```

- SELECT COUNT(ID), FirstName, LastName FROM Customers - Creates three columns. COUNT(ID) is the number of times a customers ID occurs, FirstName and LastName from the customers table.
- JOIN Rentals ON Rentals.CustomerID = Customers.ID - Connecting Rentals tables to Customers table
- GROUP BY FirstName - Group the COUNTs by FirstName
- ORDER BY COUNT(ID) DESC - Order by the ID count in descending order
- LIMIT(3); - Only show the top 3 results.