

Rockbuster Stealth LLC.

A movie rental company with stores around the world, planning to use its existing movie licenses to launch an online video rental service.



Objective Prepare a launch strategy for Rockbuster's new online video rental service and planned marketing budget reallocation for next year.



Data ●Relational database made up of 17 tables which contains information on film rentals, movies, customers, payment [here](#)



Limitations ●The data contains only the internal records of Rockbuster.



Skills Relational Databases, Entity Relationship Diagram (ERD), Data Dictionary, Database Querying, Filtering Data, Data Cleaning and Summarizing, Joining Tables, Subqueries, Common Table Expressions.



Excel, SQL, Word, Tableau, and PowerPoint.

Database queried with SQL, filtered, joined tables, subqueries and CTE visualization with Tableau.

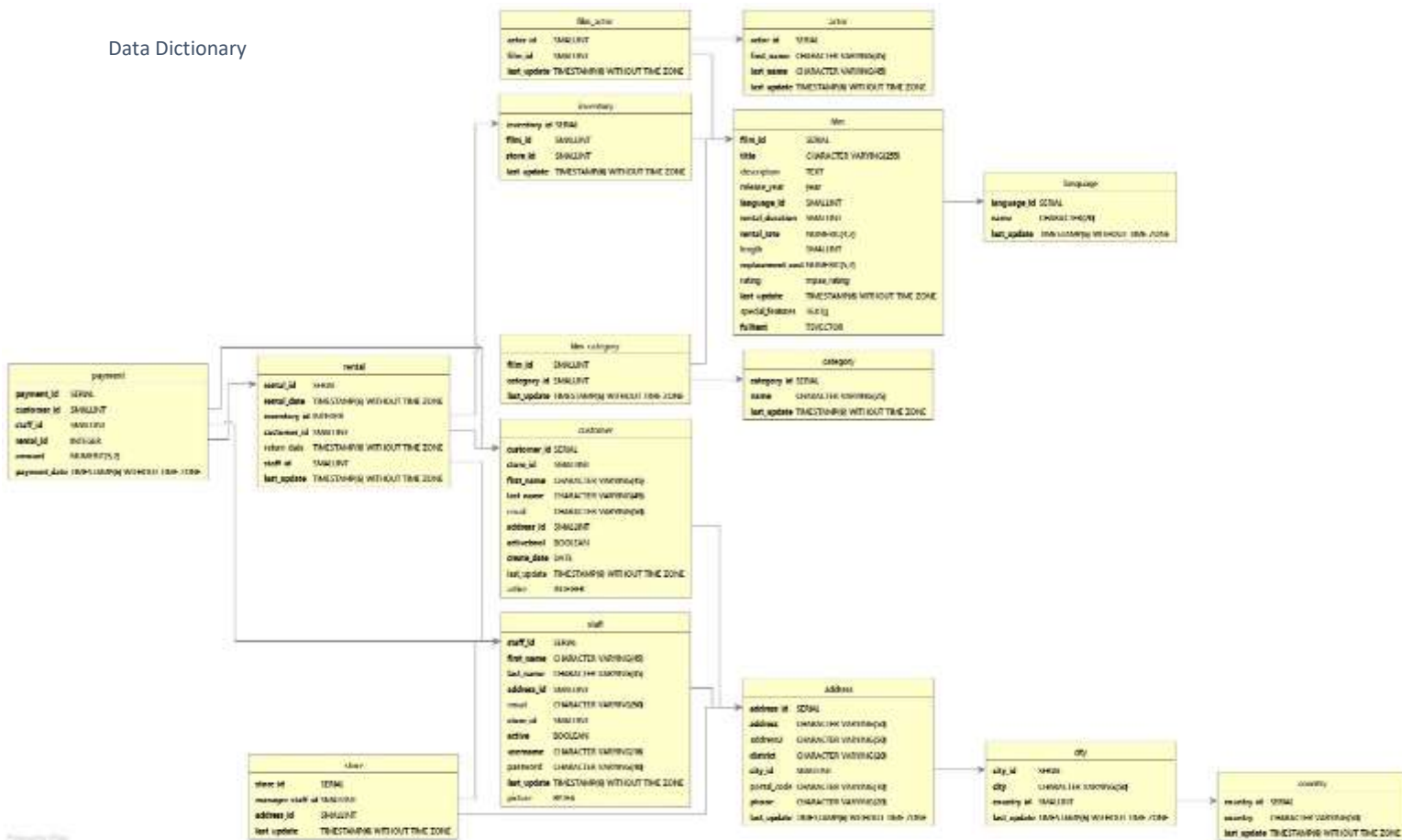


- Check the SQL scripts [here](#)
- Check the Business Managers Presentation [here](#)
- Check Data Dictionary [here](#) and Analysis [here](#)
- Check the Technical Presentation [here](#)
- Visualization in Tableau [here](#)


Initial Analysis

- Created an ERD Diagram using DBVisualizer to understand the Database structure and make the Schema Accessible.
- Performed CRUD functions to make sure the Data is clean and free of duplicates or missing values.

Data Dictionary

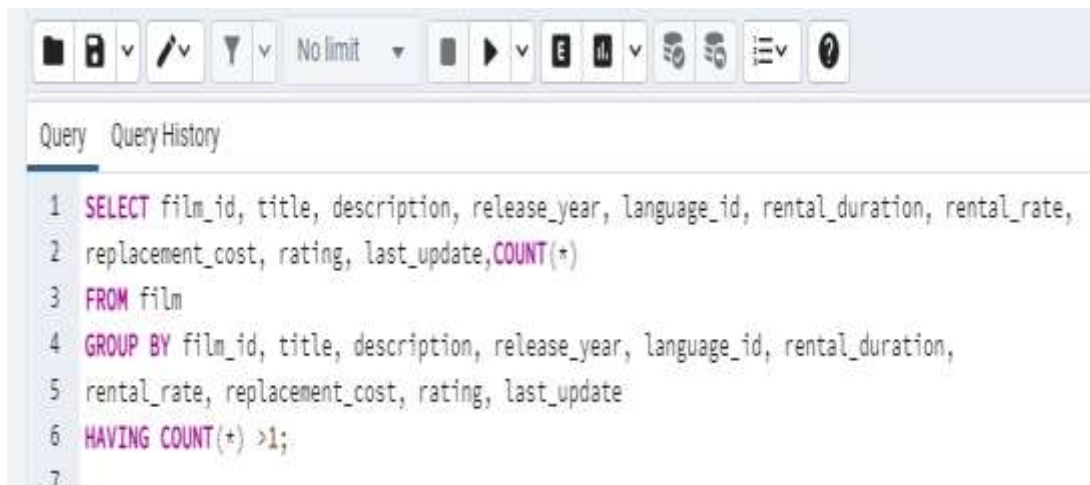


Check Data Dictionary in- [here](#)



```
1 SELECT film_id, title, description, release_year, language_id,
2 rental_duration, rental_rate,
3 replacement_cost, rating, last_update
4 FROM film
5
6
```

No duplicate



The screenshot shows a SQL query editor with a toolbar at the top containing icons for file operations, editing, and execution. Below the toolbar, there are tabs for 'Query' and 'Query History'. The 'Query' tab is active, displaying a SQL query designed to identify films with duplicate titles. The query uses a SELECT statement with various film attributes and a COUNT(*) function, grouped by these attributes and filtered by a HAVING clause to show only those with a count greater than 1.

```
1 SELECT film_id, title, description, release_year, language_id, rental_duration, rental_rate,  
2 replacement_cost, rating, last_update, COUNT(*)  
3 FROM film  
4 GROUP BY film_id, title, description, release_year, language_id, rental_duration,  
5 rental_rate, replacement_cost, rating, last_update  
6 HAVING COUNT(*) >1;  
7
```

Non-uniform data

Rockbuster/postgres@PostgreSQL 15

Query Query History

```

1 SELECT MIN(film_id) AS min_film_id,
2        MAX(film_id) AS max_film_id,
3        AVG(film_id) AS avg_film_id,
4        COUNT(film_id) AS count_rent_values,
5        MIN(release_year) AS min_release_year,
6        MAX(release_year) AS max_release_year,
7        AVG(release_year) AS avg_release_year,
8        MIN(language_id) AS min_language_id,
9        MAX(language_id) AS max_language_id,
10       AVG(language_id) AS avg_language_id,
11       MIN(rental_duration) AS min_rental_duration,
12       MAX(rental_duration) AS max_rental_duration,
13       AVG(rental_duration) AS avg_rental_duration,
14       MIN(rental_rate) AS min_rental_rate,
15       MAX(rental_rate) AS max_rental_rate,
16       AVG(rental_rate) AS avg_rental_rate,
17       MIN(replacement_cost) AS min_replacement_cost,
18       MAX(replacement_cost) AS max_replacement_cost,
19       AVG(replacement_cost) AS avg_replacement_cost,
20       COUNT(*) AS count_rows,
21       MODE () WITHIN GROUP (ORDER BY title) AS mode_title,
22       MODE () WITHIN GROUP (ORDER BY description) AS mode_description,
23       MODE () WITHIN GROUP (ORDER BY rating) AS mode_rating,
24       MODE () WITHIN GROUP (ORDER BY last_update) AS mode_last_update,
25       MODE () WITHIN GROUP (ORDER BY special_features) AS mode_special_features,
26       MODE () WITHIN GROUP (ORDER BY fulltext) AS mode_fulltext
27 FROM film;
28

```

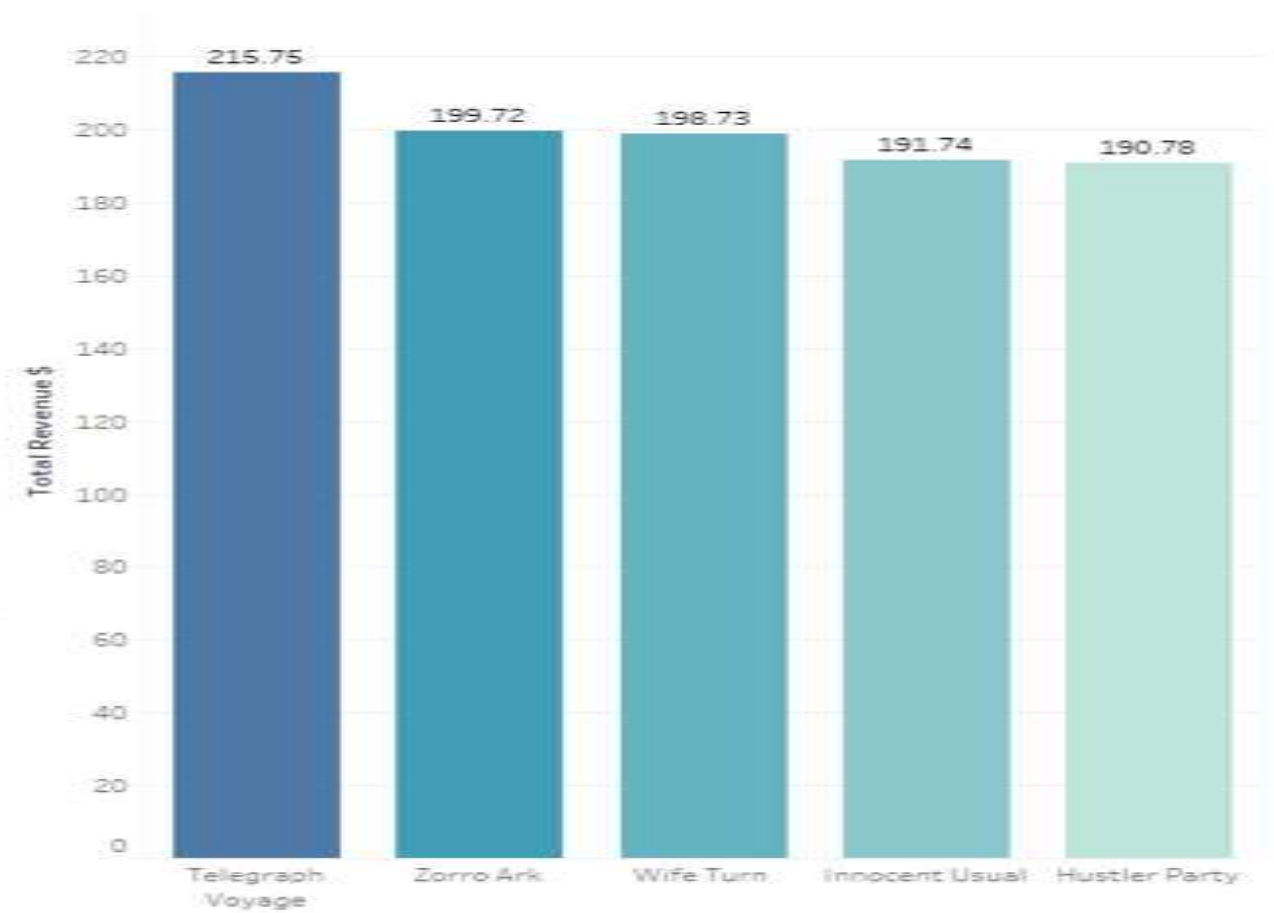
Statistical analysis

BUSINESS ANALYSIS

	country character varying (50)	customer_numbers bigint
1	India	60
2	China	53
3	United States	36
4	Japan	31
5	Mexico	30
6	Brazil	28
7	Russian Federation	28
8	Philippines	20
9	Turkey	15
10	Indonesia	14

Top 10 city per customer

Top 5 movies and revenue



Based on the above analysis, Rockbuster Stealth is operating in 108 countries with China and India (Asia) as the top 2 countries. The highest paying customers are also in these 2 countries. Also, the number of film language is 1 (English). The most watched rating is PG-12, with the most watched film as Academy Dinosaur.

These are the top customers and the revenue generated and resident countries.

Top 5 countries and revenue

Country	Customer Name	Total Amount Paid \$
India	Arlene Harvey	111.76
China	Kyle Spurlock	109.71
Japan	Marlene Welch	106.77
Mexico	Glen Talbert	100.77
United States	Clinton Buford	98.76





RECOMMENDATIONS

TARGET MARKET & PRODUCT ANALYSIS: Start the online services in the top 2 countries (China and India), with the most watched film, Academy Dinosaur, and other most rated PG-13 films. These films could be produced in the local languages of these countries.

REWARD SYSTEM: Incentivizing top-paying customers can establish a strong client relationship and an opportunity to offer new services that may appeal to their interests.

PROMOTION STRATEGY: Conducting research studies on high-performing countries can disclose useful information for effective promotional approaches. This is to understand consumer demands and replicate successful operations in other potential markets.

Key Takeaways

- * A bubble chart is a solution to visualize three metrics – number of transactions, revenue and average revenue per number of transactions. It allowed to include the addition of a third dimension as a bubble size/color to emphasize the most popular genres.
- * Common Table Expressions (CTE) is more readable than subqueries and can be reusable. However, subqueries and CTE have pros & cons and the choice between them should be made on a case-by-case basis.
- * SQL ranking functions allowed me to define top 20 movies with the highest revenue in a simple way and made my query more readable. RANK()/DENSE_RANK() functions are great for sequencing and comparing data across various factors.