

Movie Rental Database Project

A database-driven project analyzing customer behavior, rental patterns, and film details using SQL queries.

Movie Rental Database?

Database Design Overview

- **Customer:** Stores customer details
- **Film:** Contains movie information (title, length, rating, etc.)
- **Rental:** Tracks movie checkouts and returns
- **Payment:** Records customer payments
- **Staff & Store:** Manages employee and branch data

Purpose

To perform data analysis, generate reports, and understand customer and business performance using SQL.

Objectives of the Project



- To **apply SQL concepts** in a real-world dataset.
- To practice data retrieval using **SELECT**, JOIN, GROUP BY, and HAVING.
- To generate **insights** such as top customers, popular movies, and total revenue.
- To understand relationships between different entities in a **business model**.

Result

Improved understanding of data analytics and query optimization in SQL.



SQL Tasks Overview

The project included **10 SQL problems**, each focusing on a specific skill or analysis:

- 1.Retrieve customer details.
- 2.Find the most rented movies.
- 3.Analyze total payments by customers.
- 4.Identify top-performing staff.
- 5.Find movies by category or rating.
- 6.Calculate average rental duration.
- 7.Analyze revenue by city/store.
- 8.Get inactive or low-rental customers.
- 9.Combine multiple tables using JOINS.
- 10.Create a summary report with aggregated data.

Purpose To strengthen SQL query writing, logic-building, and data analysis understanding.

Entity Relationship Diagram (ERD) – Sakila Database

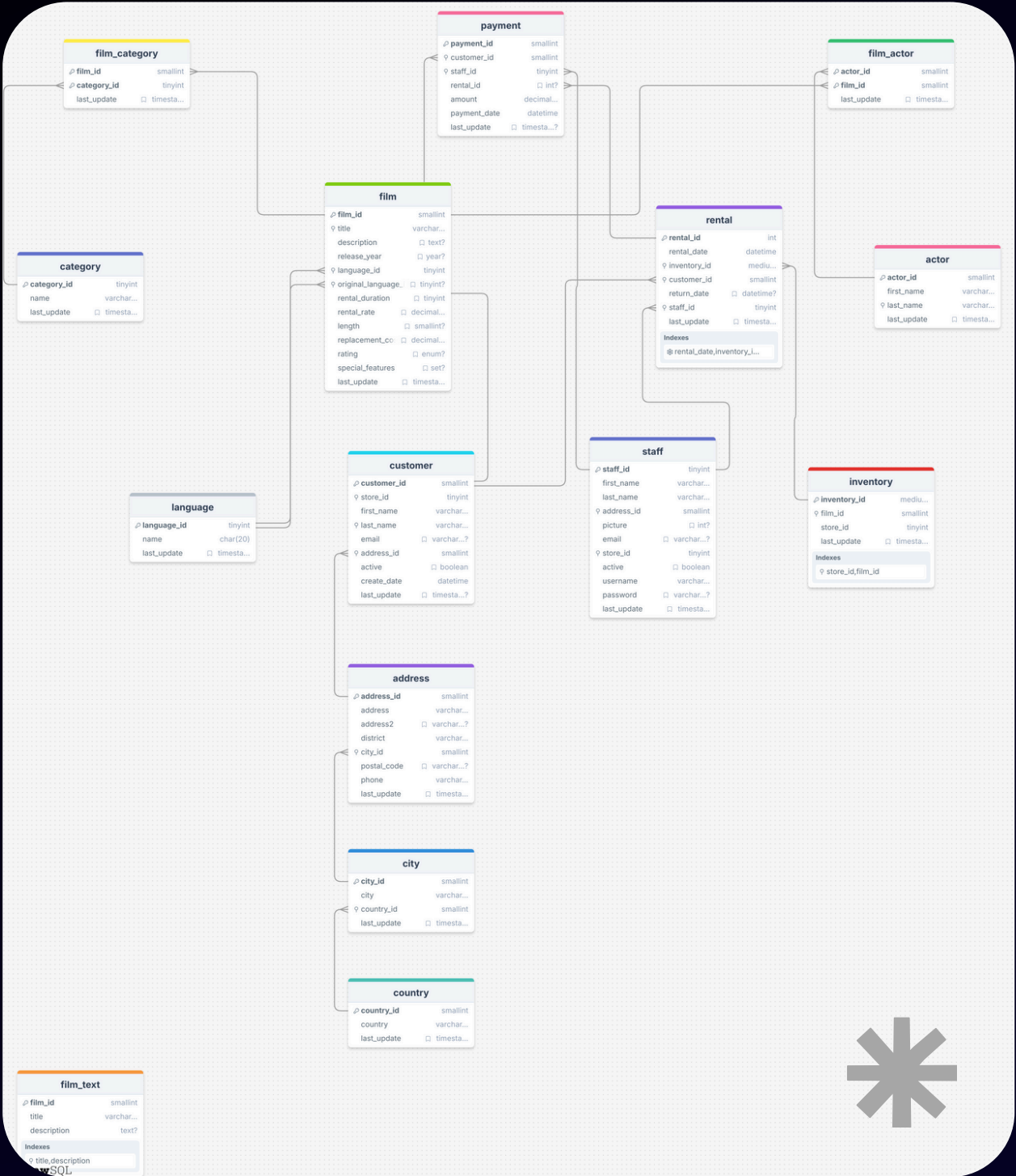


Table Name	Description
film	Stores detailed information about each movie available for rent
actor	Contains information about actors featured in films.
customer	Keeps records of customers who rent movies.
rental	Logs all movie rentals and returns made by customers.
payment	Tracks payments made by customers for their rentals.
inventory	Shows copies of each film available in specific stores.
store	Represents the physical movie rental stores.
staff	Contains details about employees working in each store
category	Lists all movie genres, like Action, Comedy, or Drama.
language	Stores the language in which each film is available.

SQL Problems (1–2)

Task 1: Total Revenue per Store *

joins and data
aggregation.

```
SELECT
  SUM(p.amount) AS revenue,
  s.store_id
FROM store s
JOIN staff st
  ON s.store_id = st.store_id
JOIN payment p
  ON st.staff_id = p.staff_id
GROUP BY s.store_id
ORDER BY revenue DESC;
```

	revenue	store_id
▶	33924.06	2
	33482.50	1

Task 2: Top 10 Most Rented Movies *

```
SELECT
  COUNT(r.rental_id) AS total_rentals,
  f.title
FROM film f
JOIN inventory i
  ON f.film_id = i.film_id
JOIN rental r
  ON i.inventory_id = r.inventory_id
GROUP BY f.film_id, f.title
ORDER BY total_rentals DESC
LIMIT 10;
```

	rentals_rate	title
▶	34	BUCKET BROTHERHOOD
	33	ROCKETEER MOTHER
	32	RIDGEMONT SUBMARINE
	32	GRIT CLOCKWORK
	32	SCALAWAG DUCK
	32	JUGGLER HARDLY
	32	FORWARD TEMPLE

SQL Problems (3–4)

Task 3: Monthly Revenue Trend *

Financial &
Customer
Analysis

```
SELECT
  DATE_FORMAT(payment_date, '%Y-%m') AS month,
  SUM(amount) AS total_revenue
FROM payment
GROUP BY month
ORDER BY month;
```

	month	total_revenue
▶	2005,05	4823.44
	2005,06	9629.89
	2005,07	28368.91
	2005,08	24070.14
	2006,02	514.18

Task 4: Active vs Inactive Customers *

```
SELECT
  active AS status,
  COUNT(customer_id) AS
total_customers
FROM customer
GROUP BY active;
```

	status	toal_customers
▶	Active	584
	Inactive	15

SQL Problems (5–6)

Task 5: Revenue by Movie Category *

Arithmetic &
Group By

Task 6: Staff Performance Report *

```
SELECT
  c.category_id,
  c.name AS category,
  SUM(p.amount) AS revenue
FROM category c
JOIN film_category fc
  ON c.category_id = fc.category_id
JOIN film f
  ON fc.film_id = f.film_id
JOIN inventory i
  ON f.film_id = i.film_id
JOIN rental r
  ON i.inventory_id = r.inventory_id
JOIN payment p
  ON r.rental_id = p.rental_id
GROUP BY c.category_id, c.name
ORDER BY revenue DESC;
```

	category_id	name	revenue
▶	1	Action	4375.85
	2	Animation	4656.30
	3	Children	3655.55
	4	Classics	3639.59
	5	Comedy	4383.58

```
SELECT
  s.staff_id,
  s.first_name,
  s.last_name,
  COUNT(r.rental_id) AS rentals_handled
FROM staff s
JOIN rental r
  ON s.staff_id = r.staff_id
GROUP BY s.staff_id, s.first_name,
s.last_name
ORDER BY rentals_handled DESC;
```

	performance	first_name	staff_id
▶	8040	Mike	1
	8004	Jon	2

SQL Problems (7–8)

Task 7: Average Rental Duration per Category *

Date functions
and average
calculations

```
SELECT
    AVG(DATEDIFF(r.return_date, r.rental_date)) AS
rental_duration,
    c.name
FROM rental r
JOIN inventory i ON r.inventory_id = i.inventory_id
JOIN film f ON i.film_id = f.film_id
JOIN film_category fc ON f.film_id = fc.film_id
JOIN category c ON fc.category_id = c.category_id
GROUP BY c.name
ORDER BY rental_duration DESC;
```

	rental_duration	category
►	5.1993	Sports
	5.1990	Games
	5.1481	Comedy
	5.0672	Music
	5.0594	Documentary
	5.0586	Sci-Fi
	5.0508	Family
	5.0468	Horror
	5.0362	Foreign
	5.0292	Action

Task 8: Top Customers by Spending *

```
SELECT
    SUM(p.amount) AS spending,
    c.customer_id,
    c.first_name,
    c.last_name
FROM payment p
JOIN customer c
    ON c.customer_id = p.customer_id
GROUP BY p.customer_id
ORDER BY spending DESC
LIMIT 10;
```

	total_spending	customer_id	first_name	last_name
►	221.55	526	KARL	SEAL
	216.54	148	ELEANOR	HUNT
	195.58	144	CLARA	SHAW
	194.61	137	RHONDA	KENNEDY
	194.61	178	MARION	SNYDER
	186.62	459	TOMMY	COLLAZO
	177.60	469	WESLEY	BULL
	175.61	468	TIM	CARY
	175.58	236	MARCIA	DEAN
	174.66	181	ANA	BRADLEY

SQL Problems (9–10)

Task 9: Late Returns by Customer *

```
SELECT
  c.customer_id,
  c.first_name,
  c.last_name,
  SUM(
    CASE
      WHEN DATEDIFF(r.return_date, r.rental_date) >
        f.rental_duration
      THEN 1
      ELSE 0
    END
  ) AS late_returns
FROM rental r
JOIN inventory i
  ON r.inventory_id = i.inventory_id
JOIN film f
  ON i.film_id = f.film_id
JOIN customer c
  ON r.customer_id = c.customer_id
GROUP BY c.customer_id, c.first_name, c.last_name
ORDER BY late_returns DESC;
```

	customer_id	first_name	last_name	late_returns
▶	526	KARL	SEAL	25
	469	WESLEY	BULL	24
	295	DAISY	BATES	24
	148	ELEANOR	HUNT	23
	144	CLARA	SHAW	22
	181	ANA	BRADLEY	22
	137	RHONDA	KENNEDY	22
	176	JUNE	CARROLL	22

Advance
Functions like
CASE WHEN

Task 10: City with Highest Rental Activity *

```
SELECT
  ct.city,
  COUNT(r.rental_id) AS rentals_booked
FROM rental r
JOIN customer c
  ON r.customer_id = c.customer_id
JOIN address a
  ON c.address_id = a.address_id
JOIN city ct
  ON a.city_id = ct.city_id
GROUP BY ct.city
ORDER BY rentals_booked DESC;
```

	product_id	product_name	month	monthly_sales
▶	1	Laptop Pro 15	2023-01	1200.00
	2	Smartphone X	2023-01	800.00
	3	Bluetooth Headphones	2023-02	300.00
	4	Smartwatch Z	2023-02	200.00
	5	Gaming Console	2023-03	1500.00
	6	LED TV 50"	2023-03	700.00
	7	Wireless Mouse	2023-06	120.00
	8	Office Chair	2023-09	360.00
	9	Coffee Maker	2023-04	180.00
	10	Air Purifier	2023-09	500.00

Conclusion & Key Insights

- Analyzed the Sakila Movie Rental Database to gain real-world business insights.
- Explored customer behavior, film performance, and revenue trends using SQL.

Key Insights

- **Top Performing Films:** Action and Comedy movies
- **Customer Activity:** loyal customers contributed to a large portion of rentals,
- **Store Performance:** Store 1 outperformed Store 2,
- **Revenue Trends:** The busiest rental months showed clear seasonal demand.
- **Staff Performance:** Certain employees processed more transactions.

Skills & Tools Learned

- joins, Subqueries, Aggregations, CTEs, and Data Analysis Logic.
- Strengthened problem-solving and data interpretation abilities.

Conclusion

This project helped bridge theoretical SQL knowledge with practical business analysis.

Thank You

This project helped me strengthen my data analysis, SQL problem-solving,
and business insight skills.

I appreciate your time and feedback!