Project Title: Maven Movies SQL Analysis

Background:

Maven Movies is a traditional DVD rental business aiming to leverage MySQL to gain comprehensive insights into its operations. With access to the mavenmovies database, the objective is to develop a deep understanding of the 16 database tables and their interrelationships. This analysis encompasses all facets of the company's data, including transactions, customer profiles, staff details and inventory management.

Project Objectives:

1. Explore the mavenmovies database to analyze staff information, store inventory, customer demographics, and transactional data.
2. Develop SQL queries to extract relevant information and address specific inquiries regarding staff, inventory, customer base, and financial metrics.
3. Present findings in a clear and concise manner to demonstrate proficiency in database analysis and SQL querying techniques.

Project Queries:

1. Retrieve a comprehensive list of staff members, including their names, email addresses, and corresponding store IDs.

Query:

```
SELECT
    first_name,
    last_name,
    email,
    store_id
FROM staff;
```

Output:

	first_name	last_name	email	store_id
▶	Mike	Hillyer	Mike.Hillyer@sakilastaff.com	1
	Jon	Stephens	Jon.Stephens@sakilastaff.com	2

2. Generate separate counts of inventory items available at each of the two Maven Movies store locations.

Query:

```
SELECT
    store_id,
    COUNT(inventory_id) AS inventory_items
FROM inventory
GROUP BY
    store_id;
```

Output:

	store_id	inventory_items
▶	1	2270
	2	2311

3. Determine the count of active customers for each store location to assess customer engagement.

Query:

```
SELECT
    store_id,
    COUNT(customer_id) AS active_customers
FROM customer
WHERE active = 1
GROUP BY
    store_id;
```

Output:

	store_id	active_customers
▶	1	318
	2	266

4. Calculate the total count of customer email addresses stored in the database to evaluate data breach liability.

Query:

```
SELECT
    COUNT(email) AS emails
FROM customer;
```

Output:

	emails
▶	599

5. Assess the diversity of film offerings by providing counts of unique film titles and categories available at each store.

Query:

```
SELECT
    store_id,
    COUNT(DISTINCT film_id) AS unique_films
FROM inventory
GROUP BY
    store_id;

SELECT
    COUNT(DISTINCT name) AS unique_categories
FROM category;
```

Output:

	store_id	unique_films
▶	1	759
	2	762

	unique_categories
▶	16

6. Analyze film replacement costs by determining the least expensive, most expensive, and average replacement costs.

Query:

```
SELECT
    MIN(replacement_cost) AS least_expensive,
    MAX(replacement_cost) AS most_expensive,
    AVG(replacement_cost) AS average_replacement_cost
FROM film;
```

Output:

	least_expensive	most_expensive	average_replacement_cost
▶	9.99	29.99	19.984000

7. Implement payment monitoring systems by calculating the average and maximum payment processed by the company.

Query:

```
SELECT
    AVG(amount) AS average_payment,
    MAX(amount) AS max_payment
FROM payment;
```

Output:

	average_payment	max_payment
▶	4.200667	11.99

8. Gain insights into customer behavior by providing a list of customer IDs sorted by rental volume, highlighting high-volume customers.

Query:

```
SELECT
    customer_id,
    COUNT(rental_id) AS number_of_rentals
FROM rental
GROUP BY
    customer_id
ORDER BY
    COUNT(rental_id) DESC;
```

Output:

customer_id	number_of_rentals
236	42
75	41
197	40
469	40
137	39
178	39
468	39
5	38
295	38
410	38
459	38
176	37
198	37
257	37
366	37
29	36
267	36
348	36
354	36

9. Extract data on store managers and locations to facilitate in-person visits for potential investors.

Query:

```
SELECT
    staff.first_name AS manager_first_name,
    staff.last_name AS manager_last_name,
    address.address,
    address.district,
    city.city,
    country.country

FROM store
    LEFT JOIN staff ON store.manager_staff_id = staff.staff_id
    LEFT JOIN address ON store.address_id = address.address_id
    LEFT JOIN city ON address.city_id = city.city_id
    LEFT JOIN country ON city.country_id = country.country_id;
```

Output:

manager_first_name	manager_last_name	address	district	city	country
Mike	Hillyer	47 MySakila Drive	Alberta	Lethbridge	Canada
Jon	Stephens	28 MySQL Boulevard	QLD	Woodridge	Australia

10. Compile an inventory list with film details, including ratings, rental rates, and replacement costs.

Query:

```
SELECT
    inventory.store_id,
    inventory.inventory_id,
    film.title,
    film.rating,
    film.rental_rate,
    film.replacement_cost
FROM inventory
    LEFT JOIN film
        ON inventory.film_id = film.film_id;
```

Output:

store_id	inventory_id	title	rating	rental_rate	replacement_cost
1	1	ACADEMY DINOSAUR	PG	0.99	20.99
1	2	ACADEMY DINOSAUR	PG	0.99	20.99
1	3	ACADEMY DINOSAUR	PG	0.99	20.99
1	4	ACADEMY DINOSAUR	PG	0.99	20.99
1	16	AFFAIR PREJUDICE	G	2.99	26.99
1	17	AFFAIR PREJUDICE	G	2.99	26.99
1	18	AFFAIR PREJUDICE	G	2.99	26.99
1	19	AFFAIR PREJUDICE	G	2.99	26.99
1	26	AGENT TRUMAN	PG	2.99	17.99
1	27	AGENT TRUMAN	PG	2.99	17.99
1	28	AGENT TRUMAN	PG	2.99	17.99
1	32	AIRPLANE SIERRA	PG-13	4.99	28.99
1	33	AIRPLANE SIERRA	PG-13	4.99	28.99
1	41	ALABAMA DEVIL	PG-13	2.99	21.99
1	42	ALABAMA DEVIL	PG-13	2.99	21.99
1	43	ALABAMA DEVIL	PG-13	2.99	21.99
1	46	ALADDIN CALENDAR	NC-17	4.99	24.99
1	47	ALADDIN CALENDAR	NC-17	4.99	24.99
1	48	ALADDIN CALENDAR	NC-17	4.99	24.99
1	49	ALADDIN CALENDAR	NC-17	4.99	24.99
1	53	ALAMO VIDEOTAPE	G	0.99	16.99
1	54	ALAMO VIDEOTAPE	G	0.99	16.99
1	55	ALAMO VIDEOTAPE	G	0.99	16.99

11. Summarize inventory data to understand the distribution of film ratings across stores

Query:

```
SELECT
    inventory.store_id,
    film.rating,
    COUNT(inventory_id) AS inventory_items
FROM inventory
    LEFT JOIN film
        ON inventory.film_id = film.film_id
GROUP BY
    inventory.store_id,
    film.rating
;
```

Output:

store_id	rating	inventory_items
1	PG	444
1	G	394
1	PG-13	525
1	NC-17	465
1	R	442
2	PG	480
2	G	397
2	NC-17	479
2	PG-13	493
2	R	462

12. Analyze inventory diversification in terms of replacement costs and film categories.

Query:

```
SELECT
    store_id,
    category.name AS category,
    COUNT(inventory.inventory_id) AS films,
    AVG(film.replacement_cost) AS avg_replacement_cost,
    SUM(film.replacement_cost) AS total_replacement_cost

FROM inventory
    LEFT JOIN film
        ON inventory.film_id = film.film_id
    LEFT JOIN film_category
        ON film.film_id = film_category.film_id
    LEFT JOIN category
        ON category.category_id = film_category.category_id

GROUP BY
    store_id,
    category.name

ORDER BY
    SUM(film.replacement_cost) DESC;
```

Output:

store_id	category	films	avg_replacement_cost	total_replacement_cost
2	Sports	181	20.697182	3746.19
1	Action	169	21.191183	3581.31
1	Drama	162	21.934444	3553.38
2	Animation	174	19.995747	3479.26
2	Documentary	164	20.544878	3369.36
1	Sports	163	20.578957	3354.37
2	Sci-Fi	163	20.493067	3340.37
1	Animation	161	20.387516	3282.39
1	Sci-Fi	149	21.795369	3247.51
1	Family	157	20.537771	3224.43
2	Action	143	21.500490	3074.57
2	Games	148	20.773784	3074.52
2	Family	153	19.512876	2985.47
2	Drama	138	21.461014	2961.62
2	Classics	139	21.292158	2959.61
1	New	148	19.267027	2851.52
1	Foreign	153	18.558627	2839.47
1	Comedy	142	19.440704	2760.58
2	Foreign	147	18.636259	2739.53
2	Children	140	19.504286	2730.60
1	Games	128	21.130625	2704.72
1	Classics	131	20.615954	2700.69
1	Documentary	130	20.728462	2694.70
2	Horror	136	19.563529	2660.64
1	Children	129	19.990000	2578.71

13. Compile customer profiles, including active status and address details, to assess customer demographics.

Query:

```
SELECT
    customer.first_name,
    customer.last_name,
    customer.store_id,
    customer.active,
    address.address,
    city.city,
    country.country

FROM customer
    LEFT JOIN address ON customer.address_id = address.address_id
    LEFT JOIN city ON address.city_id = city.city_id
    LEFT JOIN country ON city.country_id = country.country_id
;
```

Output:

first_name	last_name	store_id	active	address	city	country
MARY	SMITH	1	1	1913 Hanoi Way	Sasebo	Japan
PATRICIA	JOHNSON	1	1	1121 Loja Avenue	San Bernardino	United States
LINDA	WILLIAMS	1	1	692 Joliet Street	Athenai	Greece
BARBARA	JONES	2	1	1566 Inegl Manor	Myingyan	Myanmar
ELIZABETH	BROWN	1	1	53 Idfu Parkway	Nantou	Taiwan
JENNIFER	DAVIS	2	1	1795 Santiago de Compostela Way	Laredo	United States
MARIA	MILLER	1	1	900 Santiago de Compostela Parkway	Kragujevac	Yugoslavia
SUSAN	WILSON	2	1	478 Joliet Way	Hamilton	New Zealand
MARGARET	MOORE	2	1	613 Korolev Drive	Masqat	Oman
DOROTHY	TAYLOR	1	1	1531 Sal Drive	Esfahan	Iran
LISA	ANDERSON	2	1	1542 Tarlac Parkway	Sagamihara	Japan
NANCY	THOMAS	1	1	808 Bhopal Manor	Yamuna Nagar	India
KAREN	JACKSON	2	1	270 Amroha Parkway	Osmaniye	Turkey
BETTY	WHITE	2	1	770 Bydgoszcz Avenue	Citrus Heights	United States
HELEN	HARRIS	1	1	419 Iligan Lane	Bhopal	India
SANDRA	MARTIN	2	0	360 Toulouse Parkway	Southend-on-...	United Kingdom
DONNA	THOMPSON	1	1	270 Toulon Boulevard	Elista	Russian Fed...
CAROL	GARCIA	2	1	320 Brest Avenue	Kaduna	Nigeria
RUTH	MARTINEZ	1	1	1417 Lancaster Avenue	Kimberley	South Africa
SHARON	ROBINSON	2	1	1688 Okara Way	Mardan	Pakistan
MICHELLE	CLARK	1	1	262 A Corua (La Corua) Parkway	Tangail	Bangladesh
LAURA	RODRIG...	1	1	28 Charlotte Amalie Street	Sal	Morocco
SARAH	LEWIS	2	1	1780 Hino Boulevard	Liepaja	Latvia

14. Calculate customer spending and identify high-value customers based on total lifetime rentals and payments.

Query:

```
SELECT
    customer.first_name,
    customer.last_name,
    COUNT(rental.rental_id) AS total_rentals,
    SUM(payment.amount) AS total_payment_amount

FROM customer
    LEFT JOIN rental ON customer.customer_id = rental.customer_id
    LEFT JOIN payment ON rental.rental_id = payment.rental_id

GROUP BY
    customer.first_name,
    customer.last_name

ORDER BY
    SUM(payment.amount) DESC
    ;
```

Output:

first_name	last_name	total_rentals	total_payment_amount
KARL	SEAL	45	221.55
ELEANOR	HUNT	46	216.54
CLARA	SHAW	42	195.58
RHONDA	KENNEDY	39	194.61
MARION	SNYDER	39	194.61
TOMMY	COLLAZO	38	186.62
WESLEY	BULL	40	177.60
TIM	CARY	39	175.61
MARCIA	DEAN	42	175.58
ANA	BRADLEY	34	174.66
JUNE	CARROLL	37	173.63
DIANE	COLLINS	35	169.65
LENA	JENSEN	32	168.68
ARNOLD	HAVENS	33	167.67
CURTIS	IRBY	38	167.62
MIKE	WAY	35	166.65
DAISY	BATES	38	162.62

15. Provide a list of advisors and investors, along with their affiliations and roles

Query:

```
SELECT
    'investor' AS type,
    first_name,
    last_name,
    company_name
FROM investor

UNION

SELECT
    'advisor' AS type,
    first_name,
    last_name,
    NULL
FROM advisor;
```

Output:

type	first_name	last_name	company_name
investor	Montgomery	Burns	Springfield Syndicators
investor	Anthony	Stark	Iron Investors
investor	William	Wonka	Chocolate Ventures
advisor	Barry	Beenthere	NULL
advisor	Cindy	Smartypants	NULL
advisor	Mary	Moneybags	NULL
advisor	Walter	White	NULL

.

16. Evaluate the coverage of films featuring award-winning actors to gauge content diversity.

Query:

```
SELECT
    CASE
        WHEN actor_award.awards = 'Emmy, Oscar, Tony ' THEN '3 awards'
        WHEN actor_award.awards IN ('Emmy, Oscar','Emmy, Tony', 'Oscar, Tony') THEN '2 awards'
        ELSE '1 award'
    END AS number_of_awards,
    AVG(CASE WHEN actor_award.actor_id IS NULL THEN 0 ELSE 1 END) AS pct_w_one_film

FROM actor_award


GROUP BY
    CASE
        WHEN actor_award.awards = 'Emmy, Oscar, Tony ' THEN '3 awards'
        WHEN actor_award.awards IN ('Emmy, Oscar','Emmy, Tony', 'Oscar, Tony') THEN '2 awards'
        ELSE '1 award'
    END
```

Output:

	number_of_awards	pct_w_one_film
▶	3 awards	0.5714
	2 awards	0.9242
	1 award	0.8333