Case Study
Motion Pictures
Data Analysis

Sakila Movie Rental Store Data

Sakila is a movie rental store with a wide collection of movies in DVDs and blue ray disc formats. The management decided to analyze the data to understand that what sort of movies and actors are popularly rented.

The analysis would help them stock up the inventory of movies to improve their business. This analysis involves a set of tasks to commence with as an analyst.

Sakila Database Schema

Click the below link to understand the schema structure of the 'Sakila database'

https://dev.mysql.com/doc/sakila/en/sakila-structure.html

Objectives

- **The analysis would help the management to stock up the inventory as per the most popular movies and actors.**
- **The motive of analysis would also help the management to maximize the overall rental sales.**
- **The final report would give the overview of the current inventory in stock.**

Motion Pictures Data Analysis

/* TASK 1: Display the first_name, last_name, actor, and the details of the last_updated column */

```
USE sakila;
SELECT * FROM actor;
```

- INTERPRETATION: The above syntax selects the particular database to work with and displays the data of 200 actors_id

Motion Pictures Data Analysis

/*TASK 2: A. Display the full names of all actors

B.Display the firstnames of actors along with the count of repeated first name.

C. Display the lastnames of actors along with the count of repeated lastnames.*/

```
SELECT CONCAT(first_name,last_name)as full_name FROM actor;
SELECT COUNT(first_name),first_name FROM actor GROUP BY first_name;
SELECT COUNT(last_name),last_name FROM actor GROUP BY last_name;
```

- *INTERPRETATION- The 'Concat' is used to join the separated first & last names*

And 'Count' is an arithmetic operator that displays the count of data.

Motion Pictures Data Analysis

-- TASK 3 : Display the count of movies grouped by the ratings

```
SELECT * FROM film;
SELECT rating,COUNT(title) FROM film GROUP BY rating;
```

- INTERPRETATION- The above syntax displays the ratings and the count of title

Motion Pictures Data Analysis

-- TASK 4: Display the average rental rates based on movie ratings

```
SELECT rating,AVG(rental_rate) FROM film GROUP BY rating;
```

- *INTERPRETATION- The above syntax displays the ratings and the average of rental rate*

Motion Pictures Data Analysis

-- TASK 5 : A.Display the movie title where the replacement cost is up to $9

-- B. Display the movie titles where the replacement cost is between $15 and $20.

--C. Display the movie titles with the highest replacement cost and lowest replacement cost.

```
SELECT title,replacement_cost FROM film WHERE replacement_cost=9.99;
SELECT title,replacement_cost FROM film WHERE replacement_cost BETWEEN '15' AND '20';
SELECT title, MAX(replacement_cost),MIN(rental_rate) FROM film GROUP BY title;
```

- INTERPRETATION- The above syntaxs displays the movie title according to the changes in replacement cost.

Motion Pictures Data Analysis

-- TASK 6: Display the list of movies with number of actors listed for each

```
SELECT*FROM film_actor;
SELECT film.film_id,film.title,COUNT(film_actor.actor_id)FROM film
INNER JOIN film_actor ON film.film_id=film_actor.film_id GROUP BY film.film_id;
```

- *INTERPRETATION- The syntax displays the lists the number of actors liste*

Motion Pictures Data Analysis

-- TASK 7: Display the movie titles with "k" & "Q“

```
SELECT title FROM film WHERE title LIKE 'K%';
```

```
SELECT title FROM film WHERE title LIKE 'Q%';
```

- INTERPRETATION- The above syntax displays the titles of thee movie that starts with “k”,”Q”.

Motion Pictures Data Analysis

-- TASK 8: Display first and last names of all actors who are a part of 'Agent Truman'

```
SELECT first_name,last_name from actor
    WHERE actor_id IN(SELECT actor_id FROM film_actor WHERE film_id=
    (SELECT film_id FROM film WHERE title ='AGENT TRUMAN'));
```

- INTERPRETATION- The above syntax displays the first and last names of all actors who worked in the movie names ‘Agent Truman’

Motion Pictures Data Analysis

-- TASK 9: Display the movies fall under family category

```
SELECT*FROM category;
SELECT title FROM film WHERE film_id IN (SELECT film_id FROM film_category WHERE
category_id=(SELECT category_id FROM category WHERE name ='family'));
```

- INTERPRETATION - The above syntax shows the movies fall under the family category

Motion Pictures Data Analysis

-- TASK 10: Display the title of movies as per the most frequently rented movies

```
SELECT *FROM rental;
SELECT title FROM film WHERE last_update IN (SELECT last_update FROM rental
 WHERE rental_date=current_date());
```

- INTERPRETATION- The above syntax displays the title of movies according to the most frequently rented.

Motion Pictures Data Analysis

--/*TASK 11: Display the number of movie categories where the average difference between the movie replacement cost and the rental rate is greater than $15*/

```
SELECT COUNT(name) FROM category WHERE category_id IN(
SELECT category_id FROM film_category WHERE film_id=(
SELECT film_id FROM film GROUP BY film_id HAVING AVG(replacement_cost)-AVG(rental_rate) >15));
```

- INTERPRETATION- The above syntax shows the number of movie categories where the average difference between movie replacement and rental is greater than $15.

Motion Pictures Data Analysis

-- TASK 12: Display the names of categories and the number of movies per category

```
SELECT category.name,film.film_id FROM category
INNER JOIN film_category ON category.category_id=film_category.category_id
INNER JOIN film ON film_category.film_id=film.film_id;
```

- INTERPRETATION- The above syntax show the categories and the number of movies per categories.

Thank you!