```sql
USE film_rental;

/* 1. What is the total revenue generated from all rentals in the database? */

SELECT

   SUM(amount) AS 'Total Revenue'

FROM

   payment;



/* 2. How many rentals were made in each month_name? */

SELECT

   MONTHNAME(payment_date) AS Month,

   COUNT(payment_id) AS 'No of Rentals'

FROM

   payment

GROUP BY

   Month

ORDER BY

   2 DESC;



/* 3. What is the rental rate of the film with the longest title in the database? */

SELECT

   title,

   LENGTH(title) AS 'length Title',

   rental_rate

FROM

   Film

WHERE

   LENGTH(title) = (SELECT MAX(LENGTH(title)) FROM film);
```

```
/* 4. What is the average rental rate for films that were taken from the last 30 days from the date
("2005-05-05 22:04:30")? */

WITH RecentRentals AS (
  SELECT
    a.title,
    DATEDIFF(c.rental_date, '2005-05-05 22:04:30') AS Difference,
    AVG(rental_rate) AS avg_rent
  FROM
    film a
    LEFT JOIN inventory b ON a.film_id = b.film_id
    LEFT JOIN rental c ON b.inventory_id = c.inventory_id
  WHERE
    DATEDIFF(c.rental_date, '2005-05-05 22:04:30')  <=  30
  GROUP BY 1 , 2
  ORDER BY 1 , 2
)

SELECT * FROM RecentRentals;

/* 5. What is the most popular category of films in terms of the number of rentals? */
SELECT
  e.name AS Category,
  COUNT(c.rental_id) AS Rentals
FROM
  film a
  INNER JOIN inventory b ON a.film_id = b.film_id
  INNER JOIN rental c ON b.inventory_id = c.inventory_id
  INNER JOIN film_category d ON a.film_id = d.film_id
  INNER JOIN category e ON e.category_id = d.category_id
GROUP BY 1
ORDER BY Rentals DESC
```

```
LIMIT 1;

/* 6. Find the longest movie duration from the list of films that have not been rented by any
customer.  */
WITH FilmRentalsCount AS (
  SELECT
    title,
    COUNT(c.rental_id) AS Rentals
  FROM
    film a
    LEFT JOIN inventory b ON a.film_id = b.film_id
    LEFT JOIN rental c ON b.inventory_id = c.inventory_id
  GROUP BY
    1
  ORDER BY
    Rentals ASC
)

SELECT
  a.*,
  b.length
FROM
  FilmRentalsCount a
  INNER JOIN film b ON a.title = b.title
WHERE
  a.Rentals = 0
ORDER BY
  3 DESC
LIMIT 1;
```

```
/* 7. What is the average rental rate for films, broken down by category? */
SELECT
   e.name,
   a.title,
   AVG(rental_rate) AS avg_rent
FROM
   film a
   INNER JOIN film_category d ON a.film_id = d.film_id
   INNER JOIN category e ON e.category_id = d.category_id
GROUP BY
   1, 2;
```

```
/*  8. What is the total revenue generated from rentals for each actor in the database?  */
SELECT
   a.actor_id,
   a.first_name,
   a.last_name,
   SUM(c.rental_rate * c.rental_duration) AS Revenue
FROM
   actor a
   INNER JOIN film_actor b ON a.actor_id = b.actor_id
   INNER JOIN film c ON b.film_id = c.film_id
GROUP BY
   1, 2, 3
ORDER BY
   1;
```

```
/* 9. Show all the actresses who worked in a film having a "Wrestler" in the description.   */

SELECT DISTINCT

   a.first_name,

   a.last_name

FROM

   actor a

   INNER JOIN film_actor b ON a.actor_id = b.actor_id

   INNER JOIN film c ON b.film_id = c.film_id

WHERE

   c.description LIKE '%Wrestler%'

ORDER BY

   1;


/* 10. No column specifying the gender was given in any of the tables, so the whole actors were
taken. */

SELECT

   a.first_name,

   a.last_name,

   d.title,

   COUNT(d.title) AS Times_rented

FROM

   customer a

   INNER JOIN rental b ON a.customer_id = b.customer_id

   INNER JOIN inventory c ON b.inventory_id = c.inventory_id

   INNER JOIN film d ON c.film_id = d.film_id

GROUP BY

   1, 2, 3

HAVING

   Times_rented > 1

ORDER BY

   Times_rented DESC;
```

```
/* 11.   How many films in the comedy category have a rental rate higher than the average rental
rate?   */

SELECT
   c.name,
   COUNT(DISTINCT a.film_id) AS 'Total films'
FROM
   film a
   INNER JOIN film_category b ON a.film_id = b.film_id
   INNER JOIN category c ON b.category_id = c.category_id
WHERE
   c.name LIKE '%comedy%'
   AND a.rental_rate > (SELECT AVG(rental_rate) FROM film)
GROUP BY
   1;

/* 12. Which films have been rented the most by customers living in each city? */
WITH m_rented AS (
   SELECT
      f.city,
      d.title,
      COUNT(d.title) AS Times_rented,
      ROW_NUMBER() OVER (PARTITION BY f.city ORDER BY COUNT(d.title) DESC) AS Most_rented
   FROM
      customer a
      INNER JOIN rental b ON a.customer_id = b.customer_id
      LEFT JOIN inventory c ON b.inventory_id = c.inventory_id
      LEFT JOIN film d ON c.film_id = d.film_id
      LEFT JOIN address e ON e.address_id = a.address_id
      LEFT JOIN city f ON f.city_id = e.city_id
   GROUP BY
```

```
        1, 2
)
SELECT
    DISTINCT city,
    title,
    Times_rented
FROM
    m_rented
WHERE
    Most_rented = 1
ORDER BY
    Times_rented DESC;


/* 13.    What is the total amount spent by customers whose rental payments exceed $200?   */
SELECT
    b.customer_id,
    a.first_name,
    a.last_name,
    SUM(b.amount) AS Total_amount
FROM
    customer a
    INNER JOIN payment b ON a.customer_id = b.customer_id
GROUP BY
    a.customer_id
HAVING
    Total_amount > 200;
```

```
/* 14. Display the fields which are having foreign key constraints related to the "rental" table.   */

SELECT

   *

FROM

   information_schema.key_column_usage

WHERE

   referenced_table_name = 'rental';



/* 15.   Create a View for the total revenue generated by each staff member, broken down by store
city with the country name.   */

CREATE VIEW Revenue_Generated AS

SELECT

   c.city,

   d.country,

   e.first_name,

   e.last_name,

   SUM(f.amount) AS total_amount

FROM

   store a

   INNER JOIN address b ON a.address_id = b.address_id

   INNER JOIN city c ON b.city_id = c.city_id

   INNER JOIN country d ON c.country_id = d.country_id

   INNER JOIN staff e ON a.store_id = e.store_id

   INNER JOIN payment f ON e.staff_id = f.staff_id

GROUP BY

   1, 2, 3, 4;

SELECT * FROM Revenue_Generated;
```

```
/* 16.   Create a view based on rental information consisting of visiting_day, customer_name, the
title of the film,  no_of_rental_days, the amount paid by the customer along with the percentage of
customer spending.   */

CREATE VIEW Rental_Info AS

SELECT

   b.rental_date AS visiting_day,

   a.first_name,

   a.last_name,

   e.title,

   DATEDIFF(b.return_date, b.rental_date) AS no_of_rental_days,

   c.amount,

   ROUND(c.amount / (SUM(c.amount) OVER (PARTITION BY a.first_name)) * 100, 2) AS
Percentage_spent

FROM

   customer a

   INNER JOIN rental b ON a.customer_id = b.customer_id

   INNER JOIN payment c ON b.rental_id = c.rental_id

   INNER JOIN inventory d ON b.inventory_id = d.inventory_id

   INNER JOIN film e ON d.film_id = e.film_id

HAVING

   no_of_rental_days IS NOT NULL;

SELECT * FROM Rental_Info;



/* 17.   Display the customers who paid 50% of their total rental costs within one day.   */

WITH base AS (

   SELECT

      payment_date,

      customer_id,

      SUM(amount) AS amount

   FROM

      payment
```

```
    GROUP BY
        payment_date, customer_id
), base2 AS (
    SELECT
        payment_date,
        customer_id,
        amount,
        SUM(amount) OVER (PARTITION BY customer_id) AS total_amount
    FROM
        base
)
SELECT
    a.payment_date,
    a.customer_id,
    b.first_name,
    b.last_name,
    a.amount,
    a.total_amount
FROM
    base2 a
    INNER JOIN customer b ON a.customer_id = b.customer_id
WHERE
    total_amount > 0  -- Ensure total_amount is not zero to avoid division by zero
    AND amount / total_amount >= 0.5;
```