Social Media Schema

01 June 2025 14:39

1)Get the username and join_date for users who joined on a Monday.

```
select
        username,
   join_date
from users
where dayofweek(join_date)=02;
```

NOTE:

Dayofweek() is used for week days like Monday, Tuesday, etc for those we can provide number of day we want out of 7 week days. Here 2nd weekofday means Monday.
Day() is used to get the date of Month, like 10th, 15th, 22nd, etc. day of month

2)Find post_id, post_content, and post_date for posts created between 10 AM and 2 PM (inclusive) on any day.

```
select
        post_id,
   post_content,
   post_date
from posts
where hour(post_date) between 10 and 14;
```

NOTE:

To get minute from date time we can use minute() and for seconds we can use second()

3)List the username and the day of the week they joined.

```
select
        username,
   JOIN_DATE,
   weekday(JOIN_DATE)
from users;
```

NOTE:

The weekday function in MySQL returns 0 for Monday, 1 for Tuesday, and so on up to 6 for Sunday. This is just the way the function is designed.

4)Find the username of users who joined in the same month as 'Alice Smith'

```
with CTE as(select
        month(join_date) as JoinMonth
from users
where full_name='Alice Smith')
select
```

```
      username,
  join_date
from users
where month(join_date)=(select JoinMonth from CTE);
```

5)Retrieve post_id and comment_id for comments made on the same day as their corresponding post was created.

```
select
      p.post_id,
  c.comment_id
from
      posts p join comments c
on p.post_id=c.post_id
where
      c.comment_date=p.post_date;
```

6)Find the average number of days between a user's join date and their first post date.

```
WITH CTE AS (
  SELECT
    user_id,
    MIN(post_date) AS firstPost
  FROM
    posts
  GROUP BY
    user_id
)
SELECT
  AVG(DATEDIFF(firstPost, join_date)) AS AvgFirstPostDate
FROM
  CTE
JOIN
  users ON CTE.user_id = users.user_id;
```

7)List post_id, post_content, and the post_date of posts created on an odd-numbered day of the month.

```
select
      post_id,
  post_content,
  post_date
from
      posts
where
      day(post_date)%2!=0
```

8)Find usernames that have a number in them.

```
SELECT username
FROM Users
WHERE username REGEXP '[0-9]';
```

9)Find the username of users who have commented on 'Alice Smith's posts.

```
SELECT DISTINCT
   c_user.username
FROM
   users p_user
JOIN
   posts p ON p_user.user_id = p.user_id
JOIN
   comments c ON p.post_id = c.post_id
JOIN
   users c_user ON c.user_id = c_user.user_id
WHERE
   p_user.username = 'Alice Smith';
```

10)For each post, list the post_content and the username of the user who made the post, along with the username of any user who commented on it. Include posts with no comments.

```
SELECT
   P.post_content,
   U_poster.username AS poster_username,
   U_commenter.username AS commenter_username
FROM
   Posts AS P
JOIN
   Users AS U_poster ON P.user_id = U_poster.user_id
LEFT JOIN
   Comments AS C ON P.post_id = C.post_id
LEFT JOIN
   Users AS U_commenter ON C.user_id = U_commenter.user_id
ORDER BY
   P.post_id, U_commenter.username;
```

11)For each username, list their total number of posts and total number of comments.

```
SELECT
   u.username,
   (SELECT COUNT(c.comment_id) FROM comments c WHERE c.user_id = u.user_id) AS
comment_count,
   (SELECT COUNT(p.post_id) FROM posts p WHERE p.user_id = u.user_id) AS post_count
FROM
   users u;
```

12)Rank users by the total number of likes_count received across all their posts.

```
select
        user_id,
   sum(likes_count) as AllLikes,
        dense_rank()over(order by sum(likes_count)) as LikeRank
from
        posts
group by user_id;
```

13)For each user, find the post_id and post_date of their second post (if any).

```
with CTE as(select
        user_id,
        post_id,
   post_date,
   dense_rank()over(partition by user_id order by post_date) Postrank
from
        posts)
select
        user_id
        post_id,
   post_date
from CTE where Postrank=2;
```

14)Determine the average likes_count for posts created in the same month as their respective user joined.

```
select
        post_id,
   avg(likes_count)over(partition by month(post_date) ) AvgLikes
from
        posts p join users u
on p.user_id=u.user_id
where month(p.post_date)=month(u.join_date)
```

15)Find the username of users who made a post and then commented on that same post.

```
SELECT DISTINCT
   u.username
FROM
   posts p
JOIN
   comments c ON p.post_id = c.post_id AND p.user_id = c.user_id
JOIN
   users u ON p.user_id = u.user_id;
```

16)For each user, find the post_id and comment_id of the post they commented on immediately after making a post themselves.

```
WITH UserPosts AS (
   SELECT
      user_id,
      post_id,
      post_date,
      LEAD(post_date) OVER (PARTITION BY user_id ORDER BY post_date) AS NextPostDate
   FROM
      posts
),
UserComments AS (
   SELECT
      user_id,
        comment_id,
        post_id,
        comment_date
    FROM
        comments
)
SELECT
    up.user_id,
    up.post_id,
    uc.comment_id
FROM
    UserPosts up
JOIN
    UserComments uc ON up.user_id = uc.user_id AND up.post_id = uc.post_id
WHERE
    uc.comment_date > up.post_date AND uc.comment_date < up.NextPostDate;
```