Parul University

Name: Prerak Doshi
Email: prerak12102004@gmail.com
Roll no: 25UG033422
Phone: 8849921118
Branch: Parul University
Department: CSE10_Batch 1
Batch: 2028
Degree: B.Tech - CSE

Scan to verify results

PIET_Oracle DBMS_Course

PIET_Oracle DBMS_Session 7_PAH

Attempt : 1
Total Mark : 50
Marks Obtained : 50

Section 1 : COD

1. Problem Statement

Emma, a data analyst at a social media analytics company, has been tasked with analyzing user interactions with posts to provide valuable insights for the marketing team. To carry out this analysis, she needs to perform join queries between the USERS and POSTS tables.

Table Details:

Sample Input Records:

Table: USERS

</span>

Table: POSTS

</span>

Tasks to perform:

Write a query to retrieve the POST_ID, POST_CONTENT, POST_DATE, USER_ID, and USERNAME of posts that have more than 200 likes or more than 15 comments. Perform a LEFT OUTER JOIN between the POSTS and USERS tables, ensuring that even posts without a corresponding user are included in the results.Write a query to retrieve the USER_ID, USERNAME, POST_ID, and POST_DATE by performing a LEFT OUTER JOIN between the USERS and POSTS tables. The query should return all users, including those who haven't made any posts, and order the results by POST_ID with null values appearing first(use NULLS FIRST).Write a query to retrieve the POST_DATE, USER_ID, USERNAME, and SIGNUP_DATE for posts made by users who signed up before March 1, 2024. The query should perform a LEFT OUTER JOIN between the POSTS and USERS tables to include all posts, even if the corresponding user has no matching record.

Note: The user must write only the join queries to select the required records.

Answer

oracle.sql

```
SELECT
   P.POST_ID,
   P.POST_CONTENT,
   P.POST_DATE,
   U.USER_ID,
   U.USERNAME
FROM
   POSTS P
LEFT OUTER JOIN
   USERS U
ON
   P.USER_ID = U.USER_ID
WHERE
   P.LIKES > 200 OR P.COMMENTS > 15;
```

```
SELECT
   U.USER_ID,
   U.USERNAME,
   P.POST_ID,
   P.POST_DATE
FROM
   USERS U
LEFT OUTER JOIN
   POSTS P
ON
   U.USER_ID = P.USER_ID
ORDER BY
   P.POST_ID NULLS FIRST;

SELECT
   P.POST_DATE,
   U.USER_ID,
   U.USERNAME,
   U.SIGNUP_DATE
FROM
   POSTS P
LEFT OUTER JOIN
   USERS U
ON
   P.USER_ID = U.USER_ID
WHERE
   U.SIGNUP_DATE < DATE '2024-03-01';
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

Alex, a sales manager at a retail company, is tasked with analyzing the performance of sales representatives. He needs to join data from the SALES_REPRESENTATIVES and SALES tables to get a comprehensive view of which representatives are handling sales for specific products.

Table Details:

Sample Input Records:

Table: SALES_REPRESENTATIVES

</span>

Table: SALES

</span>

Tasks to Perform

Write a query to retrieve pairs of sales (SALE_ID1, SALE_ID2) for the same product S1.PRODUCT where the first sale (SALE_AMOUNT1) is less than the second sale (SALE_AMOUNT2). The query should compare each sale with every other sale for the same product and display the SALE_ID of both sales, the PRODUCT, and their respective sale amounts.Write a query to retrieve pairs of sales (SALE_ID1, SALE_ID2) made by the same sales representative (REP_ID), ensuring that the SALE_ID values are different. The query should return the SALE_ID of both sales and the REP_ID.Write a query to retrieve pairs of sales (SALE_ID1, SALE_ID2) made by the same sales representative (REP_ID) where the first sale (SALE_AMOUNT1) is greater than the second sale (SALE_AMOUNT2). The query should return the SALE_ID of both sales, the REP_ID, and their respective sale amounts.

Note: The user must write only the join queries to select the required records.

Answer

oracle.sql

```
SELECT S1.SALE_ID AS SALE_ID1, S2.SALE_ID AS SALE_ID2, S1.PRODUCT,
S1.SALE_AMOUNT AS SALE_AMOUNT1, S2.SALE_AMOUNT AS SALE_AMOUNT2
FROM SALES S1
JOIN SALES S2
ON S1.PRODUCT = S2.PRODUCT
AND S1.SALE_AMOUNT < S2.SALE_AMOUNT;

SELECT S1.SALE_ID AS SALE_ID1, S2.SALE_ID AS SALE_ID2, S1.REP_ID
FROM SALES S1
JOIN SALES S2
ON S1.REP_ID = S2.REP_ID
```

```
AND S1.SALE_ID <> S2.SALE_ID;

SELECT S1.SALE_ID AS SALE_ID1, S2.SALE_ID AS SALE_ID2, S1.REP_ID,
S1.SALE_AMOUNT AS SALE_AMOUNT1, S2.SALE_AMOUNT AS SALE_AMOUNT2
FROM SALES S1
JOIN SALES S2
ON S1.REP_ID = S2.REP_ID
AND S1.SALE_AMOUNT > S2.SALE_AMOUNT;
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

James, a data analyst at a travel agency, has been tasked with analyzing customer feedback alongside booking details to provide insights into customer satisfaction and booking trends. To achieve this, James needs to join data from the CUSTOMER_FEEDBACK and BOOKINGS tables.

</span>

Sample Table Records:

Table Name: BOOKINGS

</span>

Table Name: CUSTOMER_FEEDBACK

</span>

Tasks to be performed:

Write a query to retrieve feedback information along with the associated booking details where the feedback date matches the booking date for the same customer. The query should return the FEEDBACK_ID, FEEDBACK_DATE from CUSTOMER_FEEDBACK, and the corresponding BOOKING_ID, BOOKING_DATE from BOOKINGS.Write a query to find pairs of feedbacks that have the same rating but different feedback IDs. The query should return Feedback1_ID, Feedback1_Rating from the first feedback, and Feedback2_ID, Feedback2_Rating from the second feedback.Write a query to retrieve feedback information along with the booking details where the feedback date matches the booking date for the same customer. The query should return FEEDBACK_ID, BOOKING_ID, CUSTOMER_ID, and FEEDBACK_DATE.

Note: The user must write only the join queries to select the required records.

Answer

oracle.sql

```
SELECT CF.FEEDBACK_ID, CF.FEEDBACK_DATE, B.BOOKING_ID,
B.BOOKING_DATE
FROM CUSTOMER_FEEDBACK CF
JOIN BOOKINGS B
ON CF.CUSTOMER_ID = B.CUSTOMER_ID
AND CF.FEEDBACK_DATE = B.BOOKING_DATE;

SELECT CF1.FEEDBACK_ID AS Feedback1_ID, CF1.RATING AS
Feedback1_Rating, CF2.FEEDBACK_ID AS Feedback2_ID, CF2.RATING AS
Feedback2_Rating
FROM CUSTOMER_FEEDBACK CF1
JOIN CUSTOMER_FEEDBACK CF2
ON CF1.RATING = CF2.RATING
AND CF1.FEEDBACK_ID <> CF2.FEEDBACK_ID;

SELECT F.FEEDBACK_ID, B.BOOKING_ID, F.CUSTOMER_ID, F.FEEDBACK_DATE
FROM CUSTOMER_FEEDBACK F
JOIN BOOKINGS B ON F.CUSTOMER_ID = B.CUSTOMER_ID AND
F.FEEDBACK_DATE = B.BOOKING_DATE;
```

Status : Correct ***Marks : 10/10***

4. Problem Statement:

Sophia, a business analyst at a subscription-based streaming service, is tasked with analyzing subscriber data to gain insights into viewing patterns and subscription status across different subscription tiers. To accomplish this, she needs to join data from the SUBSCRIPTIONS and VIEWING_LOGS tables.

Table Details:

Sample input records:

Table Name: SUBSCRIPTIONS

</span>

Table Name: VIEWING_LOGS

</span>

Tasks to Perform:

Write a query to calculate the average view duration (AVG_DURATION) for each subscription. Retrieve the SUBSCRIPTION_ID and TIER from the SUBSCRIPTIONS table, and the average of VIEW_DURATION from the VIEWING_LOGS table. The query should join these tables on the SUBSCRIPTION_ID field and group the results by SUBSCRIPTION_ID and TIER.Write a query to retrieve the SUBSCRIPTION_ID, VIEW_DATE, and VIEW_DURATION from the VIEWING_LOGS table for the record with the maximum view duration. Use a subquery to determine the maximum VIEW_DURATION in the VIEWING_LOGS table.Write a query to retrieve distinct subscription start and end dates (START_DATE, END_DATE) for subscriptions where the view duration in the VIEWING_LOGS table exceeds 100. The query should join the SUBSCRIPTIONS and VIEWING_LOGS tables on the SUBSCRIPTION_ID field and filter the results based on the view duration.

Note: The user must write only the join queries to select the required records.

Answer

oracle.sql

```
SELECT s.SUBSCRIPTION_ID, S.TIER, AVG(v.VIEW_DURATION) AS
AVG_DURATION
FROM SUBSCRIPTIONS s
JOIN VIEWING_LOGS v ON s.SUBSCRIPTION_ID = v.SUBSCRIPTION_ID
GROUP BY s.SUBSCRIPTION_ID, s.TIER;

SELECT v.SUBSCRIPTION_ID, v.VIEW_DATE, v.VIEW_DURATION
FROM VIEWING_LOGS v
WHERE v.VIEW_DURATION = (SELECT MAX(VIEW_DURATION) FROM
VIEWING_LOGS);

SELECT DISTINCT s.START_DATE, s.END_DATE
FROM SUBSCRIPTIONS s
JOIN VIEWING_LOGS v ON s.SUBSCRIPTION_ID = v.SUBSCRIPTION_ID
WHERE v.VIEW_DURATION > 100;
```

Status : Correct ***Marks : 10/10***

5. Problem Statement

Alex, a database administrator at a social media platform, has been tasked with extracting and transforming data for analysis. To do this, he needs to perform joins on the SOCIAL_MEDIA_POSTS and USER_PROFILES tables.

Table Details:

Sample Input Records:

Table Name: SOCIAL_MEDIA_POSTS

Table Name: USER_PROFILES

Tasks to perform:

Write a query to retrieve the USER_ID, USERNAME, and TAGS from the USER_PROFILES and SOCIAL_MEDIA_POSTS tables for posts that include the tag 'travel'. The query should join the SOCIAL_MEDIA_POSTS and USER_PROFILES tables on the USERNAME field and filter the results where TAGS contains the word 'travel'.Write a query to retrieve the POST_ID and

BIO from the SOCIAL_MEDIA_POSTS and USER_PROFILES tables for users located in 'San Francisco, USA'. The query should join the SOCIAL_MEDIA_POSTS and USER_PROFILES tables on the USERNAME field and filter the results where the LOCATION in the USER_PROFILES table is 'San Francisco, USA'.Write a query to retrieve the POST_ID, CONTENT, and USERNAME from the SOCIAL_MEDIA_POSTS and USER_PROFILES tables for posts where the user's bio contains the word 'tech'. The query should join the SOCIAL_MEDIA_POSTS and USER_PROFILES tables on the USERNAME field and filter the results where the BIO in the USER_PROFILES table contains the word 'tech'.

Note: The user must write only the join queries to select the required records.

Answer

oracle.sql

```
SELECT up.USER_ID, up.USERNAME, sp.TAGS
FROM SOCIAL_MEDIA_POSTS sp
INNER JOIN USER_PROFILES up
ON sp.USERNAME = up.USERNAME
WHERE sp.TAGS LIKE '%travel%';

SELECT sp.POST_ID, up.BIO
FROM SOCIAL_MEDIA_POSTS sp
INNER JOIN USER_PROFILES up
ON sp.USERNAME = up.USERNAME
WHERE up.LOCATION = 'San Francisco, USA';

SELECT sp.POST_ID, sp.CONTENT, up.USERNAME
FROM SOCIAL_MEDIA_POSTS sp
INNER JOIN USER_PROFILES up
ON sp.USERNAME = up.USERNAME
WHERE up.BIO LIKE '%tech%';
```

Status : Correct ***Marks : 10/10***