```sql
-- Data Extraction and Analysis

-- Q1: Can a user show up more than once in the activity table
-- Ans: Yes, a user can make multiply purchase

Select uid as userid, count(spent) as spent from activity
group by uid
order by count(spent) desc
limit 10;


-- Q2 : What type of join should we use to join the user table to the activity table

-- Ans : Left Join

Select u.id as user_id, u.country as country, u.gender as gender, max(a.dt) as date_purchase, a.device as device,round(coalesce(sum(a.spent),0),2)
from users u
left join activity a
on u.id = a.uid
group by u.id,a.device
order by u.id;

-- Q3 :  What SQL can we use to fill in NULL Values
-- Ans : coalesce(sum(a.spent),0)

Select uid as user_id,coalesce(sum(spent),0) from activity
group by uid
limit 10;

-- Q4: What are the start and end date of the experiment
-- Ans : start date: 2023-01-25 , end date: 2023-02-26

Select min(join_dt) as start_date,max(join_dt) as end_date from groups;

-- Q5 :How many total users were in the experiment
-- Answer : 48,943

Select count(uid) from groups;

-- Q6: How many users were in the control and treatment groups
-- Ans: A = 24,343 , B = 24,600

select "group" as "treatment group", count(uid) as "count user" from groups
group by "group";


-- Q7: What was the conversion rate of all user
-- Ans: 4.278
```

```
Select
        (count(distinct case when spent is not null then uid
END)::float/count(distinct uid)::float)*100 as conversion_rate
  from groups g
  left join
  activity a
  using (uid);


-- Q8: What is the user conversion rate for the control and treatment group
-- Ans: A = 3.923 , B = 4.630

Select g.group as group,
        (count(distinct case when spent is not null then uid
END)::float/count(distinct uid)::float)*100 as conversion_rate
  from groups g
  left join
  activity a
  using (uid)
  group by g.group;




-- Q9: What is the average amount spent per user for the control and treatment
groups, including users who did not convert?
-- Ans: A = 3.375 , B = 3.391

Select g.group, sum(a.spent)/count(distinct g.uid) as "avg spent per user"
from groups g
left join activity a
using (uid)
group by g.group;


--Q10: Dataset upload for Tableau visual

Select u.id as User_ID,u.country as Country,u.gender as Gender,g.device as
Device,g.group as Test_grp,g.join_dt as join_date,
max(a.dt) as purchase_date,Count(a.spent) as
Quantity,round(sum(coalesce(a.spent,0)),2) as Amount_spent,
Case
        WHEN sum(coalesce(a.spent,0)) = 0 THEN 0
 ELSE
                1
  END as Convert
from users u
left join groups g
on u.id = g.uid
```

```
left join activity a
using (uid)
group by u.id,u.country,u.gender,g.device,g.group,g.join_dt;


-- Other insight to the query

-- country segmentation
select country, count(ID) AS "Count_country" from users
group by country;


--gender segmentation
select gender, count(ID) AS "count of gender" from users
group by gender;

-- Total count of users
select count(distinct id) as total_user from users;

--segmentation of groups count
select "group",count(distinct uid) from groups
group by "group";

--segmentation by device
select device,count(distinct uid) from groups
group by "device";


* Novelty Effect Analysis:

-- Converted Users Average Amount Spent Over Join Date:
SELECT
      g.join_dt AS join_date,
      g.group,
      COUNT(DISTINCT g.uid) AS total_users,
      COUNT(DISTINCT a.uid) AS paid_users,
      round(SUM(a.spent),2) AS total_spent
FROM
    groups AS g
LEFT JOIN activity AS a ON g.uid = a.uid
GROUP BY
        g.group,
        g.join_dt
ORDER BY 1;

-- All Users’ Metrics Over Join Date:
SELECT
      n.join_date,
      n.group,
      ROUND(CAST(SUM(n.paid_users) / MAX(n.total_users) * 100 AS
      DECIMAL(10,2)), 2) AS conversion_rate,
      ROUND(CAST(SUM(n.total_spent)/MAX(n.total_users) AS DECIMAL(10,2)),2) AS
      average_spent
FROM(SELECT
           g.join_dt AS join_date,
           g.group,
           COUNT(DISTINCT g.uid) AS total_users,
           COUNT(DISTINCT a.uid) AS paid_users,
           SUM(a.spent) AS total_spent
      FROM
           groups AS g
      LEFT JOIN activity AS a ON g.uid = a.uid
      GROUP BY
              g.group,
              g.join_dt
      ORDER BY 1) AS n
GROUP BY 1, 2;

-- Date Difference and Converted Users:
SELECT n.group, COUNT(n.user_id), n.date_difference
FROM(SELECT
           a.uid AS user_id,
     g.group,
     g.join_dt AS date_registered,
     a.dt AS date_converted,
     SUM(COALESCE(a.spent, 0)) AS total_spent_usd,
     a.dt - g.join_dt AS date_difference
      FROM groups AS g
JOIN activity AS a
ON g.uid = a.uid
GROUP BY 1,2,3,4) AS n
GROUP BY 1,3
ORDER BY 3;
```