**a.**       **Number of users per country, SQL code:**

SELECT country, COUNT(id) as total\_users

FROM users

GROUP BY country

ORDER BY total\_users DESC;

Number of users per country: The SQL query provided returns the total number of users for each country, ordered from highest to lowest. According to the results, the largest user base is in the USA with 14,772 users, followed by Brazil with 9,434 users, Mexico with 5,738, Germany with 3,854, and Turkey with 3,732. There are also 643 users for which the country is not known or not provided ("null").

**b.**      **Group test distribution per country:**

SELECT users.country, groups.group, COUNT(groups.uid) as total\_users

FROM groups

JOIN users ON groups.uid = users.id

GROUP BY users.country, groups.group

ORDER BY total\_users DESC;

**c.**       **Converted users per country and per gender:**

SELECT users.country, users.gender, COUNT(DISTINCT activity.uid) as converted\_users

FROM activity

JOIN users ON activity.uid = users.id

GROUP BY users.country, users.gender

ORDER BY converted\_users DESC;

**d. Conversion rate for each group (control and test):** To start our analysis of the A/B test results, we need to calculate the conversion rate for each group (control and test). The conversion rate is defined as the number of users who made at least one purchase (converted) divided by the total number of users in each group.

We use the following SQL query to get the conversion rate per group:

SELECT

  g.Group AS group\_type,

  COUNT(DISTINCT u.Id) AS total\_users,

  COUNT(DISTINCT a.Uid) AS converted\_users,

  (COUNT(DISTINCT a.Uid) \* 100.0 / COUNT(DISTINCT u.Id)) AS conversion\_rate

FROM

  Users u

  JOIN Groups g ON u.Id = g.Uid

  LEFT JOIN Activity a ON u.Id = a.Uid

WHERE

  g.join\_dt >= '2023-02-01' AND g.join\_dt <= '2023-02-28'

GROUP BY   g.Group;

**e**. **The average revenue generated by users**:

The purpose of the next SQL analysis is to examine the average revenue generated by users in both. This SQL query aims to analyze the performance of groups A and B by calculating the total revenue, total users, and average revenue per user across all device types. By comparing the average revenue per user for both groups, we can gain insights into the effectiveness of the tested banner in driving revenue for the GloBox online marketplace.

Here's the SQL code for this analysis:

WITH group\_revenue AS (

    SELECT

        g."group",

        SUM(a."spent") AS total\_revenue

    FROM

        "groups" g

        JOIN "activity" a ON g."uid" = a."uid"

GROUP BY

        g."group"

),

group\_users AS (

    SELECT

        "group",

        COUNT("uid") AS total\_users

    FROM

        "groups"

    GROUP BY

        "group"

)

SELECT

    gr."group",

    gr.total\_revenue,

    gu.total\_users,

    gr.total\_revenue::decimal / gu.total\_users::decimal AS avg\_revenue\_per\_user

FROM

    group\_revenue gr

    JOIN group\_users gu ON gr."group" = gu."group";

**f. Comparing the conversion rates and average revenue per user across different countries:**

The next SQL analysis will focus on comparing the conversion rates and average revenue per user across different countries. This will help us understand if the impact of the banner varies across different regions. To do this, we will:

1. Join the 'users', 'groups', and 'activity' tables.

2. Calculate the total number of users, total revenue, and total conversions for each country and group (A or B).

3. Calculate the conversion rate and average revenue per user for each country and group.

4. Display the results in a tabular format for easy comparison.

Here's the SQL query to perform the analysis as explained:

WITH country\_data AS (

    SELECT u.country,

           g.group,

           COUNT(DISTINCT u.id) AS total\_users,

           COUNT(a.uid) AS total\_converted\_users,

           SUM(a.spent) AS total\_revenue

    FROM users u

    JOIN groups g ON u.id = g.uid

    LEFT JOIN activity a ON u.id = a.uid

    WHERE g.device = 'A' AND a.device = 'A' OR a.device IS NULL

    GROUP BY u.country, g.group

)

SELECT country\_data.country,

       country\_data.group,

       total\_users,

       total\_converted\_users,

       total\_revenue,

       (total\_converted\_users \* 1.0 / total\_users) \* 100 AS conversion\_rate,

       total\_revenue / total\_users AS avg\_revenue\_per\_user

FROM country\_data

ORDER BY country\_data.country, country\_data.group;

This query will join the tables, group the data by country and test group, and then calculate the total users, total converted users, total revenue, conversion rate, and average revenue per user for each country and group. The results will be displayed in a table for easy comparison.

**g. Conversion rates and average revenue per user by gender**

In the next analysis, we will examine the conversion rates and average revenue per user by gender. This will help us to understand if there is any significant difference in user behaviour between different genders. The SQL query will perform the following steps:

1. Join the Users, groups, and activity tables on the uid column.
2. Group the data by the group and gender columns.
3. Calculate the total number of users, the total number of converted users (those who made a purchase), the total revenue, the conversion rate (total\_converted\_users / total\_users \* 100), and the average revenue per user (total\_revenue / total\_users).

Here's the SQL query:

SELECT

    g.group,

    u.gender,

    COUNT(DISTINCT u.id) as total\_users,

    COUNT(DISTINCT a.uid) as total\_converted\_users,

    SUM(a.spent) as total\_revenue,

    (COUNT(DISTINCT a.uid) \* 100.0 / COUNT(DISTINCT u.id)) as conversion\_rate,

    (SUM(a.spent) / COUNT(DISTINCT u.id)) as avg\_revenue\_per\_user

FROM

    Users u

JOIN

    groups g ON u.id = g.uid

LEFT JOIN

    activity a ON u.id = a.uid

GROUP BY

    g.group, u.gender

ORDER BY

    g.group, u.gender;

**h.**   **Device Analysis:**

The following SQL code will calculate the conversion rates and average revenue per user for each device type and group.

SELECT

    g.device,

    g.group,

    COUNT(DISTINCT g.uid) AS total\_users,

    COUNT(DISTINCT a.uid) AS total\_converted\_users,

    COALESCE(SUM(a.spent), 0) AS total\_revenue,

    (COUNT(DISTINCT a.uid) \* 100.0 / COUNT(DISTINCT g.uid)) AS conversion\_rate,

    (COALESCE(SUM(a.spent), 0) / COUNT(DISTINCT g.uid)) AS avg\_revenue\_per\_user

FROM

    groups g

LEFT JOIN

    activity a ON g.uid = a.uid

GROUP BY

    g.device,

    g.group;

 This SQL query first joins the groups and activity tables on the uid field. It then groups the data by device and group, and calculates the total number of users, the number of converted users (those who made a purchase), the total revenue, the conversion rate (the number of converted users divided by the total number of users), and the average revenue per user.

**i.**        **Test Duration Analysis:**

For this, we first need to find the duration for which each user was in the test. We can find this by subtracting the join date from the last purchase date for each user. Then we can group by duration to see the conversion rates and average revenue.

SELECT

    g.group,

    COUNT(DISTINCT g.uid) AS total\_users,

    COUNT(DISTINCT a.uid) AS total\_converted\_users,

    COALESCE(SUM(a.spent), 0) AS total\_revenue,

    (COUNT(DISTINCT a.uid) \* 100.0 / COUNT(DISTINCT g.uid)) AS conversion\_rate,

    (COALESCE(SUM(a.spent), 0) / COUNT(DISTINCT g.uid)) AS avg\_revenue\_per\_user,

    CAST(MAX(a.dt) AS DATE) - CAST(MIN(g.join\_dt) AS DATE) AS test\_duration\_days

FROM

    groups g

LEFT JOIN

    activity a ON g.uid = a.uid

GROUP BY

    g.group;

**j.**   **User Retention Analysis:**

We can define retained users as those who made more than one purchase during the test period. Let's calculate the number of retained users and their contribution to the total revenue.

SELECT

    g.group,

    COUNT(DISTINCT g.uid) AS total\_users,

    COUNT(DISTINCT CASE WHEN purchase\_count > 1 THEN a.uid END) AS retained\_users,

    COALESCE(SUM(a.spent), 0) AS total\_revenue,

    COALESCE(SUM(CASE WHEN purchase\_count > 1 THEN a.spent END), 0) AS retained\_user\_revenue

FROM

    groups g

LEFT JOIN (

    SELECT

        uid,

        COUNT(\*) as purchase\_count,

        SUM(spent) as spent

    FROM

        activity

    GROUP BY

        uid

) a ON g.uid = a.uid

GROUP BY

    g.group;