The attached dataset contains information for players that signed up between July and September, from a small group of Countries.

You'll find a unique identifier for each player, the signup date, Country, currency and the dates of their deposit with the status of the deposit and the amount deposited. The date of the deposit is formatted as "days from signup" (for example when the value is 3,it means that the player deposited 3 days after completing registration).

Could you please prepare a small presentation for the Acquisition Manager on :
- Signups trend
- FTDs trend
- New registrations deposits behaviour

If you would have to recreate this dataset starting from the following tables, how would you do it? (in SQL)

Table 1 has the following columns:

- **account_id: varchar**
- **userID: varchar**
- **userName: varchar**
- **first_name: varchar**
- **last_name: varchar**
- **test_account: bool**
- **country_id: varchar**
- **brand_id: varchar**
- **date_created: varchar**
- **status_id: varchar**

Table 2:

- **deposit_id: int**
- **deposit_started_date: varchar**

- deposit_account_id: float
- deposit_updated_date: varchar
- deposit_payment_method_id: int
- deposit_amount: varchar
- deposit_amount_eur: float
- deposit_status: varchar
- deposit_currency_id: varchar
- deposit_currency: varchar
- deposit_reversed: bool

Table 3:

- currency_id: varchar
- currency_from: varchar
- currency_to: varchar
- currency_exchange_rate: float

Table 4:

- account_id: int
- status_id: int
- user_locked: bool

Extra questions:

- Please provide a distribution of the deposits, including the 5th and 95th percentiles.
- Lets suppose we need to run this distribution but per country in order to spot any trend difference.
- How would you provide these distributions in a table, by countries (row) in an automated way?
- Please include the cumulative successful and total deposits sums, per userID.

SECTION 1
The presentation can be found the attached .pptx and supporting visualization in .pbix

SECTION 2
Assumptions

- Since country_id: varchar is an id and there are no country table, it assumed the country id is the country name.
- Date attributes are in date format.
- Since deposit_amount is varchar, it is assumed the equivalent deposit amount is deposit_amount_eur: float and the equivalent currency deposited is recorded as deposit_currency: varchar
- Include only unlocked users.

Question 1: Recreate this dataset starting from the following tables.

```
SELECT
    a.userID AS userID,
    a.date_created::date AS signup_date,
    a.country_id AS Country,
    b.deposit_currency AS currency,
    DATEDIFF(b.deposit_started_date::date, a.date_created::date) AS days_till_deposit,
-- Calculate days until deposit
    b.deposit_started_date::date AS deposit_date,
    b.deposit_status AS deposit_status,
    CASE
        WHEN b.deposit_currency = 'EUR' THEN b.deposit_amount_eur  -- No conversion
needed for EUR
        ELSE b.deposit_amount_eur / c.currency_exchange_rate  -- Convert from EUR to
target currency
    END AS deposit_amount
FROM
    Table1 a
JOIN Table2 b
    ON a.account_id = b.deposit_account_id
JOIN Table3 c
    ON c.currency_from = 'EUR'  -- Ensure exchange rate is from EUR
    AND c.currency_to = b.deposit_currency  -- Convert to the deposit's currency
JOIN Table4 d
    ON a.account_id = d.account_id
WHERE
```

```
    a.date_created BETWEEN '2023-07-01' AND '2023-09-30'
    AND a.country_id IN ('UK', 'Italy', 'Canada', 'New Zealand')  -- Specified country IDs
    AND b.deposit_currency IN ('GBP', 'NZD', 'CAD', 'EUR')
    AND d.user_locked = FALSE  -- Include only unlocked users
ORDER BY
    a.userID, b.deposit_started_date;
```

Extra Questions

1. *Please provide a distribution of the deposits, including the 5th and 95th percentiles.*

```
WITH DepositData AS (
SELECT
    a.userID AS userID,
    a.date_created::date AS signup_date,
    a.country_id AS Country,
    b.deposit_currency AS currency,
    DATEDIFF(b.deposit_started_date::date, a.date_created::date) AS days_till_deposit,
-- Calculate days until deposit
    b.deposit_started_date::date AS deposit_date,
    b.deposit_status AS deposit_status,
    CASE
        WHEN b.deposit_currency = 'EUR' THEN b.deposit_amount_eur  -- No conversion
needed for EUR
        ELSE b.deposit_amount_eur / c.currency_exchange_rate  -- Convert from EUR to
target currency
    END AS deposit_amount
FROM
    Table1 a
JOIN Table2 b
    ON a.account_id = b.deposit_account_id
JOIN Table3 c
    ON c.currency_from = 'EUR'  -- Ensure exchange rate is from EUR
    AND c.currency_to = b.deposit_currency  -- Convert to the deposit's currency
JOIN Table4 d
    ON a.account_id = d.account_id
WHERE
    a.date_created BETWEEN '2023-07-01' AND '2023-09-30'
    AND a.country_id IN ('UK', 'Italy', 'Canada', 'New Zealand')  -- Specified country IDs
    AND b.deposit_currency IN ('GBP', 'NZD', 'CAD', 'EUR')
```

```
    AND d.user_locked = FALSE  -- Include only unlocked users
ORDER BY
    a.userID, b.deposit_started_date;
)

SELECT
    COUNT(*) AS total_deposits,
    MIN(deposit_amount) AS min_deposit,
    MAX(deposit_amount) AS max_deposit,
    PERCENTILE_CONT(0.05) WITHIN GROUP (ORDER BY deposit_amount) AS
percentile_5,
    PERCENTILE_CONT(0.95) WITHIN GROUP (ORDER BY deposit_amount) AS
percentile_95,
    AVG(deposit_amount) AS average_deposit
FROM
    DepositData;
```

2. *Lets suppose we need to run this distribution but per country in order to spot any trend difference.*

```
WITH DepositData AS (
SELECT
    a.userID AS userID,
    a.date_created::date AS signup_date,
    a.country_id AS Country,
    b.deposit_currency AS currency,
    DATEDIFF(b.deposit_started_date::date, a.date_created::date) AS days_till_deposit,
-- Calculate days until deposit
    b.deposit_started_date::date AS deposit_date,
    b.deposit_status AS deposit_status,
    CASE
        WHEN b.deposit_currency = 'EUR' THEN b.deposit_amount_eur  -- No conversion
needed for EUR
        ELSE b.deposit_amount_eur / c.currency_exchange_rate  -- Convert from EUR to
target currency
    END AS deposit_amount
FROM
    Table1 a
JOIN Table2 b
    ON a.account_id = b.deposit_account_id
```

```
JOIN Table3 c
   ON c.currency_from = 'EUR'  -- Ensure exchange rate is from EUR
   AND c.currency_to = b.deposit_currency  -- Convert to the deposit's currency
JOIN Table4 d
   ON a.account_id = d.account_id
WHERE
   a.date_created BETWEEN '2023-07-01' AND '2023-09-30'
   AND a.country_id IN ('UK', 'Italy', 'Canada', 'New Zealand')  -- Specified country IDs
   AND b.deposit_currency IN ('GBP', 'NZD', 'CAD', 'EUR')
   AND d.user_locked = FALSE  -- Include only unlocked users
ORDER BY
   a.userID, b.deposit_started_date;
)

SELECT
   Country,
   COUNT(*) AS total_deposits,
   MIN(deposit_amount) AS min_deposit,
   MAX(deposit_amount) AS max_deposit,
   PERCENTILE_CONT(0.05) WITHIN GROUP (ORDER BY deposit_amount) AS
percentile_5,
   PERCENTILE_CONT(0.95) WITHIN GROUP (ORDER BY deposit_amount) AS
percentile_95,
   AVG(deposit_amount) AS average_deposit
FROM
   DepositData
GROUP BY
   Country
ORDER BY
   Country;
```

3. *How would you provide these distributions in a table, by countries (row) in an automated way?*

A view allows querying distribution data by country as if it were a regular table. This approach is especially useful for regularly accessing the data without recalculating it each time. The view can be updated using a scheduler or orchestrator in the database, making it ideal when real-time data is required for each query of the distribution. This setup is well-suited for dashboards or reporting systems where data is frequently accessed and needs to remain current.

```
CREATE OR REPLACE VIEW DepositDistributionView AS
WITH DepositData AS (
   SELECT
      a.userID AS userID,
      a.date_created::date AS signup_date,
      a.country_id AS Country,
      b.deposit_currency AS currency,
      DATEDIFF(b.deposit_started_date::date, a.date_created::date) AS
days_till_deposit,
      b.deposit_started_date::date AS deposit_date,
      b.deposit_status AS deposit_status,
      CASE
         WHEN b.deposit_currency = 'EUR' THEN b.deposit_amount_eur
         ELSE b.deposit_amount_eur / c.currency_exchange_rate
      END AS deposit_amount
   FROM
      Table1 a
   JOIN Table2 b
      ON a.account_id = b.deposit_account_id
   JOIN Table3 c
      ON c.currency_from = 'EUR'
      AND c.currency_to = b.deposit_currency
   JOIN Table4 d
      ON a.account_id = d.account_id
   WHERE
      a.date_created BETWEEN '2023-07-01' AND '2023-09-30'
      AND a.country_id IN ('UK', 'Italy', 'Canada', 'New Zealand')
      AND b.deposit_currency IN ('GBP', 'NZD', 'CAD', 'EUR')
      AND d.user_locked = FALSE
)
SELECT
   Country,
   COUNT(*) AS total_deposits,
   MIN(deposit_amount) AS min_deposit,
   MAX(deposit_amount) AS max_deposit,
   PERCENTILE_CONT(0.05) WITHIN GROUP (ORDER BY deposit_amount) AS
percentile_5,
```

```
    PERCENTILE_CONT(0.95) WITHIN GROUP (ORDER BY deposit_amount) AS
percentile_95,
    AVG(deposit_amount) AS average_deposit
FROM
    DepositData
GROUP BY
    Country;

– to call view
SELECT * FROM DepositDistributionView;
```

4. Please include the cumulative successful and total deposits sums, per userID.

```
CREATE OR REPLACE VIEW DepositDistributionView AS
WITH DepositData AS (
    SELECT
        a.userID AS userID,
        a.date_created::date AS signup_date,
        a.country_id AS Country,
        b.deposit_currency AS currency,
        DATEDIFF(b.deposit_started_date::date, a.date_created::date) AS
days_till_deposit,
        b.deposit_status AS deposit_status,
        CASE
            WHEN b.deposit_currency = 'EUR' THEN b.deposit_amount_eur
            ELSE b.deposit_amount_eur / c.currency_exchange_rate
        END AS deposit_amount
    FROM
        Table1 a
    JOIN Table2 b
        ON a.account_id = b.deposit_account_id
    JOIN Table3 c
        ON c.currency_from = 'EUR'
        AND c.currency_to = b.deposit_currency
    JOIN Table4 d
        ON a.account_id = d.account_id
    WHERE
        a.date_created BETWEEN '2023-07-01' AND '2023-09-30'
        AND a.country_id IN ('UK', 'Italy', 'Canada', 'New Zealand')
```

```
        AND b.deposit_currency IN ('GBP', 'NZD', 'CAD', 'EUR')
        AND d.user_locked = FALSE
),
AggregateData AS (
    SELECT
        userID,
        Country,
        COUNT(*) AS total_deposits,
        MIN(deposit_amount) AS min_deposit,
        MAX(deposit_amount) AS max_deposit,
        PERCENTILE_CONT(0.05) WITHIN GROUP (ORDER BY deposit_amount) AS
percentile_5,
        PERCENTILE_CONT(0.95) WITHIN GROUP (ORDER BY deposit_amount) AS
percentile_95,
        AVG(deposit_amount) AS average_deposit,
        SUM(deposit_amount) AS total_deposit_sum,
        SUM(CASE WHEN deposit_status = 'successful' THEN deposit_amount ELSE 0
END) AS cumulative_successful_deposit_sum
    FROM
        DepositData
    GROUP BY
        userID, Country
)

SELECT
    *,
    SUM(total_deposit_sum) OVER (PARTITION BY Country) AS
country_total_deposit_sum,
    SUM(cumulative_successful_deposit_sum) OVER (PARTITION BY Country) AS
country_successful_deposit_sum
FROM
    AggregateData;
```