* **users\_customuser:** customer demographic and contact information
* **savings\_savingsaccount:** records of deposit transactions
* **plans\_plan:** records of plans created by customers
* **withdrawals\_withdrawal:**  records of withdrawal transactions

1. The **plans\_plan** contains the following columns:

id, name, description, amount, start\_date, last\_charge\_date, next\_charge\_date, created\_on, frequency\_id, owner\_id, status\_id, interest\_rate, withdrawal\_date, default\_plan, plan\_type\_id, goal, locked, next\_returns\_date, last\_returns\_date, cowry\_amount, debit\_card, is\_archived, is\_deleted, is\_goal\_achieved, is\_a\_goal, is\_interest\_free, plan\_group\_id, is\_deleted\_from\_group, is\_a\_fund, purchased\_fund\_id, is\_a\_wallet, currency\_is\_dollars, is\_auto\_rollover, is\_vendor\_plan, plan\_source, is\_donation\_plan, donation\_description, donation\_expiry\_date, donation\_link, link\_code, charge\_payment\_fee, donation\_is\_approved, is\_emergency\_plan, is\_personal\_challenge, currency\_id, is\_a\_usd\_index, usd\_index\_id, open\_savings\_plan, new\_cycle, recurrence, is\_bloom\_note, is\_managed\_portfolio, portfolio\_holdings\_id, is\_fixed\_investment, is\_regular\_savings, preset\_id

1. The **savings\_savingsaccount** contains the following columns:

`id`, `savings\_id`, `maturity\_start\_date`, `maturity\_end\_date`, `amount`, `confirmed\_amount`, `deduction\_amount`, `new\_balance`, `transaction\_date`, `transaction\_reference`, `transaction\_status`, `verification\_call\_amount`, `verification\_call\_message`, `verification\_call\_code`, `verification\_transaction\_date`, `book\_returns`, `available\_returns`, `returns\_on\_hold`, `last\_returns\_date`, `next\_returns\_date`, `created\_on`, `card\_billed\_id`, `channel\_id`, `charging\_method\_id`, `owner\_id`, `plan\_id`, `transaction\_type\_id`, `verification\_status\_id`, `gateway\_response\_message`, `fee\_in\_kobo`, `donor\_id`, `is\_anonymous`, `description`, `payment\_gateway`, `source\_bank\_account`, `currency`, `fee\_in\_cents`

1. The **users\_customuser** contains the following columns:

`password`, `last\_login`, `is\_superuser`, `id`, `email`, `name`, `first\_name`, `last\_name`, `phone\_number`, `date\_of\_birth`, `is\_staff`, `is\_active`, `date\_joined`, `is\_admin`, `username`, `created\_on`, `gender\_id`, `invite\_code`, `avatar\_firebase\_reference`, `avatar\_local\_uri`, `avatar\_url`, `risk\_apetite`, `current\_latitude`, `current\_longitude`, `postal\_address`, `pin`, `ambassador\_profile\_id`, `is\_ambassador`, `account\_source`, `is\_vendor\_account`, `is\_business\_account`, `is\_account\_deleted`, `vendor\_code`, `is\_halal\_account`, `address\_city`, `address\_country`, `address\_state`, `address\_street`, `monthly\_expense`, `monthly\_salary`, `is\_account\_disabled`, `authy\_id`, `fraud\_score`, `account\_campaign`, `account\_medium`, `last\_password\_change`, `last\_pin\_change`, `is\_private`, `disabled\_at`, `is\_disabled\_by\_owner`, `is\_account\_deleted\_by\_owner`, `proposed\_deletion\_date`, `reason\_for\_deletion`, `enabled\_at`, `signup\_device`, `proposed\_enablement\_date`, `tier\_id`

1. The **withdrawals\_withdrawal:**  contains the following columns:

`id`, `amount`, `amount\_withdrawn`, `transaction\_reference`, `transaction\_date`, `new\_balance`, `bank\_id`, `owner\_id`, `plan\_id`, `transaction\_channel\_id`, `transaction\_status\_id`, `transaction\_type\_id`, `fee\_in\_kobo`, `description`, `gateway`, `gateway\_response`, `session\_id`, `currency`, `fee\_in\_cents`, `payment\_id`, `created\_on`, `updated\_on`, `withdrawal\_intent\_id`

**SOLUTIONS**

**Question 1: High-Value Customers with Multiple Products**

To find customers with at least one funded savings plan AND one funded investment plan:

Note: I'm dividing confirmed\_amount by 100 since you mentioned amounts are stored in kobo.

=========================================================

SELECT

u.id AS owner\_id,

CONCAT(u.first\_name, ' ', u.last\_name) AS name,

COUNT(DISTINCT CASE WHEN p.is\_regular\_savings = 1 THEN p.id END) AS savings\_count,

COUNT(DISTINCT CASE WHEN p.is\_a\_fund = 1 THEN p.id END) AS investment\_count,

SUM(s.confirmed\_amount) / 100 AS total\_deposits

FROM

users\_customuser u

JOIN

plans\_plan p ON u.id = p.owner\_id

JOIN

savings\_savingsaccount s ON p.id = s.plan\_id

GROUP BY

u.id, name

HAVING

savings\_count > 0 AND investment\_count > 0

ORDER BY

total\_deposits DESC;

**Question 2: Transaction Frequency Analysis**

This query categorizes customers by transaction frequency:

WITH monthly\_transactions AS (

SELECT

s.owner\_id,

DATE\_FORMAT(s.transaction\_date, '%Y-%m') AS month,

COUNT(\*) AS transaction\_count

FROM

savings\_savingsaccount s

GROUP BY

s.owner\_id, month

),

customer\_averages AS (

SELECT

owner\_id,

AVG(transaction\_count) AS avg\_transactions\_per\_month,

CASE

WHEN AVG(transaction\_count) >= 10 THEN 'High Frequency'

WHEN AVG(transaction\_count) >= 3 THEN 'Medium Frequency'

ELSE 'Low Frequency'

END AS frequency\_category

FROM

monthly\_transactions

GROUP BY

owner\_id

)

SELECT

frequency\_category,

COUNT(\*) AS customer\_count,

ROUND(AVG(avg\_transactions\_per\_month), 1) AS avg\_transactions\_per\_month

FROM

customer\_averages

GROUP BY

frequency\_category

ORDER BY

CASE

WHEN frequency\_category = 'High Frequency' THEN 1

WHEN frequency\_category = 'Medium Frequency' THEN 2

ELSE 3

END;

**Question 3: Account Inactivity Alert**

To find active accounts with no transactions in the last year:

SELECT

p.id AS plan\_id,

p.owner\_id,

CASE

WHEN p.is\_regular\_savings = 1 THEN 'Savings'

WHEN p.is\_a\_fund = 1 THEN 'Investment'

ELSE 'Other'

END AS type,

MAX(s.transaction\_date) AS last\_transaction\_date,

DATEDIFF(CURDATE(), MAX(s.transaction\_date)) AS inactivity\_days

FROM

plans\_plan p

LEFT JOIN

savings\_savingsaccount s ON p.id = s.plan\_id

WHERE

p.status\_id != 3 -- Assuming status\_id 3 means inactive/closed

GROUP BY

p.id, p.owner\_id, type

HAVING

inactivity\_days > 365 OR last\_transaction\_date IS NULL

ORDER BY

inactivity\_days DESC;

Question 4: Customer Lifetime Value Estimation

WITH customer\_data AS (

SELECT

u.id AS customer\_id,

CONCAT(u.first\_name, ' ', u.last\_name) AS name,

TIMESTAMPDIFF(MONTH, u.date\_joined, CURDATE()) AS tenure\_months,

COUNT(s.id) AS total\_transactions,

SUM(s.confirmed\_amount) / 100 \* 0.001 AS total\_profit -- 0.1% of transaction value

FROM

users\_customuser u

LEFT JOIN

savings\_savingsaccount s ON u.id = s.owner\_id

GROUP BY

u.id, name, tenure\_months

HAVING

tenure\_months > 0 -- Avoid division by zero

)

SELECT

customer\_id,

name,

tenure\_months,

total\_transactions,

ROUND((total\_profit / tenure\_months) \* 12, 2) AS estimated\_clv

FROM

customer\_data

ORDER BY

estimated\_clv DESC;

For the CLV calculation, I'm using the formula:

* Calculate profit per transaction (0.1% of transaction value)
* Calculate monthly rate (total\_transactions / tenure\_months)
* Annualize it (multiply by 12)
* Multiply by average profit per transaction