



DATA BANK

#8WeekSQLChallenge

Course: MIS 395 - Business Data Management

Lecturers: Mr. Dang Thai Doan

Ms. Huynh Tuyet Ngan

Presented by: 3A group

OUR MEMBERS



Hoang Vinh



Tam Nhu



Thanh Dat

CASE OVERVIEW

- **Neo-bank** providing banking and data storage services via digital platforms
- A merger between banking, cryptocurrency, and data storage services
- Provides customers with secure, distributed data storage platforms
- Track and forecast customer storage needs
- Calculate metrics and growth for planning



3 DIFFERENT DATA SOURCES

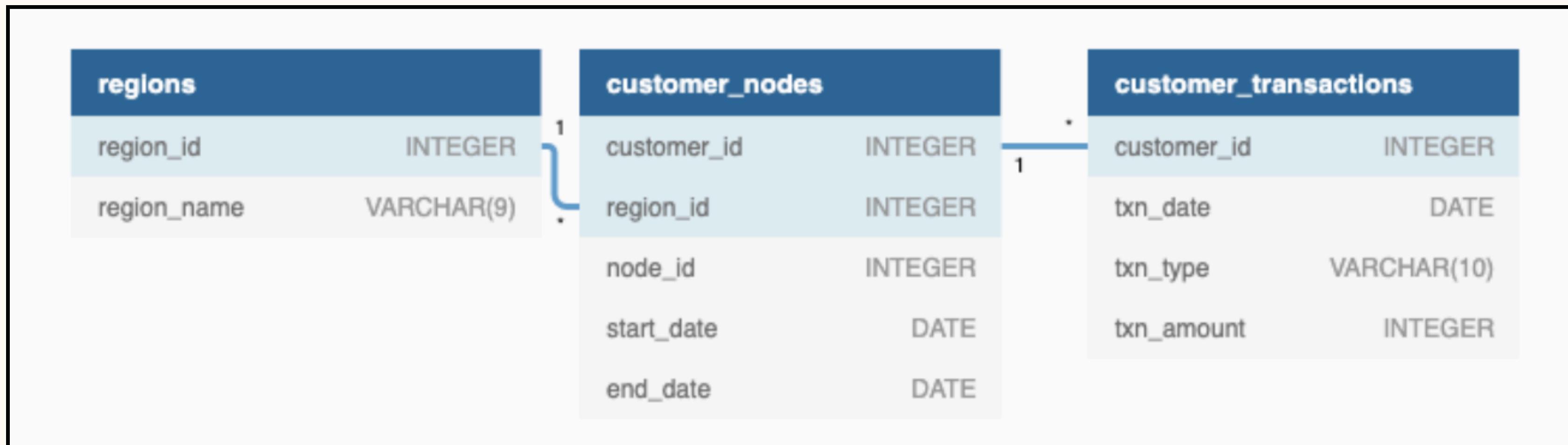
- ① **Regions:** Contains some regions across the globe, each with a unique ID.
- ② **Customer Nodes:** Includes information about customers in a single node from the beginning to the end.
- ③ **Customer Transactions:** Encompasses records of customers' actions in the Data Bank system.



CASE STUDY QUESTION GROUPS

- A Customer Nodes Exploration
- B Customer Transactions
- C Data Allocation Challenge

RELATIONSHIP DIAGRAM



SAMPLE INPUT

Regions

region_id	region_name
1	Africa
2	America
3	Asia
4	Europe
5	Oceania

Customer Nodes

customer_id	region_id	node_id	start_date	end_date
1	3	4	2020-01-02	2020-01-03
2	3	5	2020-01-03	2020-01-17
3	5	4	2020-01-27	2020-02-18

Customer Transactions

customer_id	txn_date	txn_type	txn_amount
429	2020-01-21	deposit	82
155	2020-01-10	deposit	712
398	2020-01-01	deposit	196

A. CUSTOMER NODES EXPLORATION

1. How many unique nodes are there on the Data Bank system?

SQL Query:

```
SELECT
    COUNT(DISTINCT node_id) as total_unique_nodes
FROM
    data_bank.customer_nodes;
```

SQL Output:

	total_unique_nodes	lock
	bigint	
1	5	

Insight:

The Data Bank system has **5 unique nodes**. This random distribution is frequently updated to **minimize the risk of hackers** breaching Data Bank's system and stealing customers' funds or data.

A. CUSTOMER NODES EXPLORATION

2. What is the number of nodes per region?

SQL Query:

```

SELECT
    rg.region_name, COUNT(DISTINCT nd.node_id) AS total_nodes
FROM
    data_bank.customer_nodes AS nd
INNER JOIN
    data_bank.regions AS rg ON rg.region_id = nd.region_id
GROUP BY
    rg.region_name
ORDER BY
    rg.region_name;
    
```

SQL Output:

	region_name character varying (9)	total_nodes bigint
1	Africa	5
2	America	5
3	Asia	5
4	Australia	5
5	Europe	5

Insight:

Each region in the Data Bank system contains **exactly 5 unique nodes**. It reflects a strategy to ensure consistent **data storage and security, reducing regional disparities** and **providing balanced access** to their services across different regions.

A. CUSTOMER NODES EXPLORATION

3. How many customers are allocated to each region?

SQL Query:

```

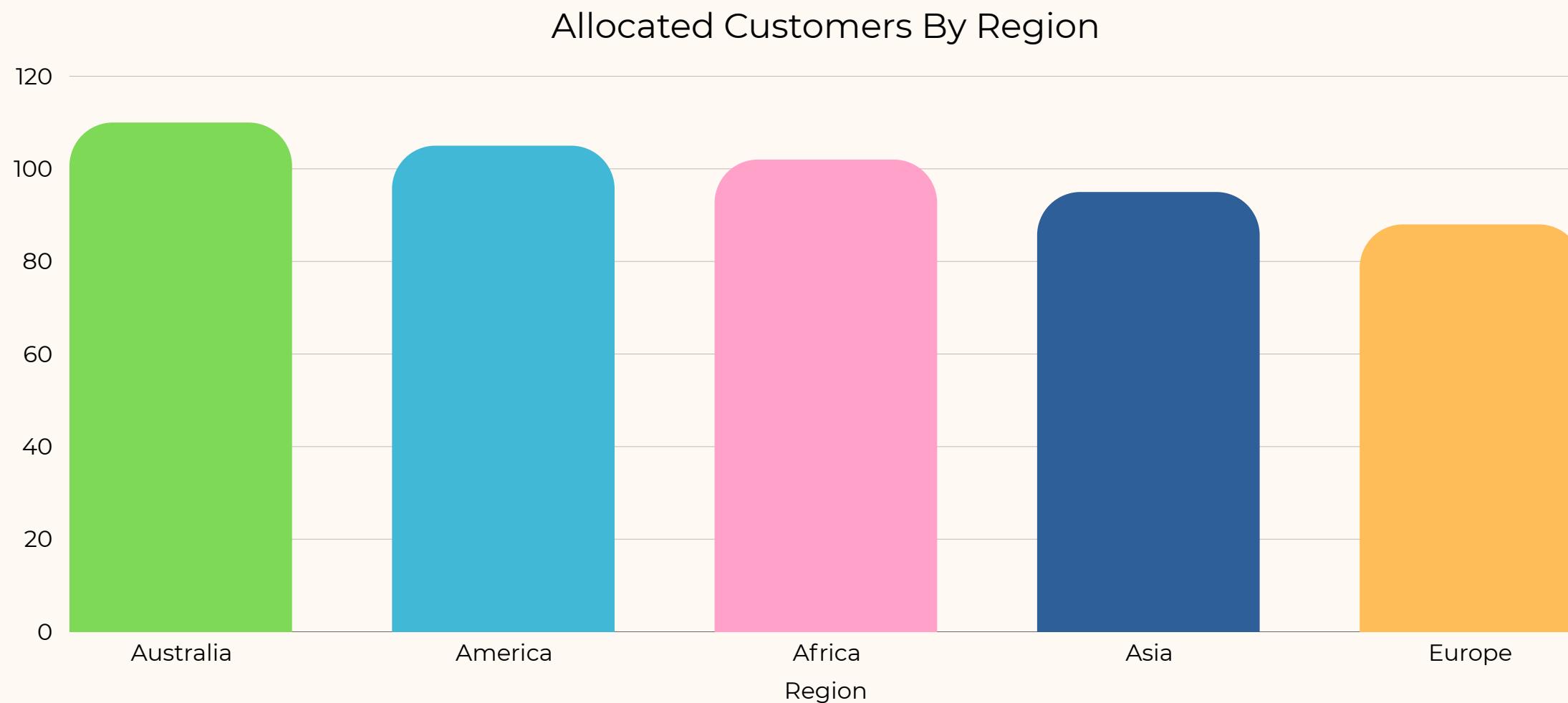
SELECT
    rg.region_name, COUNT(DISTINCT nd.customer_id) AS total_customers
FROM
    data_bank.customer_nodes AS nd
INNER JOIN
    data_bank.regions AS rg ON rg.region_id = nd.region_id
GROUP BY
    rg.region_name
ORDER BY
    total_customers DESC;
    
```

SQL Output:

	region_name	total_customers
	character varying (9)	bigint
1	Australia	110
2	America	105
3	Africa	102
4	Asia	95
5	Europe	88

A. CUSTOMER NODES EXPLORATION

3. How many customers are allocated to each region?



Insight:

The number of customers allocated to each region varies, with **Australia having the highest** at 110 customers and **Europe having the lowest** at 88 customers. This distribution suggests that customer density is higher in certain regions, likely due to factors such as regional popularity or market penetration. Regions with fewer customers may require targeted strategies to increase customer acquisition and engagement.

A. CUSTOMER NODES EXPLORATION

4. How many days on average are customers reallocated to a different node?

SQL Query:

```
WITH active_days_in_nodes AS
  (SELECT
    nd.customer_id,
    nd.node_id,
    SUM(nd.end_date - nd.start_date) AS active_days_in_nodes
   FROM
    data_bank.customer_nodes AS nd
   WHERE
    end_date != '9999-12-31'
   GROUP BY
    nd.customer_id, nd.node_id
   ORDER BY
    customer_id, node_id)
SELECT round(AVG(adin.active_days_in_nodes)) AS active_days_in_nodes
FROM active_days_in_nodes AS adin
```

SQL Output:

	active_days_in_nodes	numeric
1		24

Insight:

On average, customers are reallocated to a different node **every 24 days**. This reflects how often Data Bank reassigned customers to new storage nodes, possibly due to their security protocols or data management strategies.

A. CUSTOMER NODES EXPLORATION

Question 5: What is the median, 80th and 95th percentile for this same reallocation days metric for each region?

SQL Query:

```

56  SELECT
57    rg.region_name,
58    PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY nd.end_date - nd.start_date) AS median_days,
59    PERCENTILE_CONT(0.8) WITHIN GROUP (ORDER BY nd.end_date - nd.start_date) AS p80_days,
60    PERCENTILE_CONT(0.95) WITHIN GROUP (ORDER BY nd.end_date - nd.start_date) AS p95_days
61  FROM
62    data_bank.customer_nodes as nd
63  INNER JOIN
64    data_bank.regions as rg on rg.region_id = nd.region_id
65  WHERE
66    end_date != '9999-12-31'
67  GROUP BY
68    rg.region_name
69  ORDER BY
70    rg.region_name;

```

SQL Output:

	region_name character varying (9) 	median_days double precision 	p80_days double precision 	p95_days double precision 
1	Africa	15	24	28
2	America	15	23	28
3	Asia	15	23	28
4	Australia	15	23	28
5	Europe	15	24	28

Insight:

The median, 80th, and 95th percentiles for reallocation days vary slightly across regions.

- The median is consistently **15 days** for all regions.
- The 80th percentile ranges from **23 to 24 days**.
- The 95th percentile remains **28 days**.

B. CUSTOMER TRANSACTIONS

1. What is the unique count and total amount for each transaction type?

SQL Query:

```

SELECT
    txn_type,
    COUNT(customer_id) AS total_deposits,
    sum(txn_amount) AS total_amount
FROM
    data_bank.customer_transactions
GROUP BY
    txn_type
ORDER BY
    txn_type;
  
```

SQL Output:

	txn_type character varying (10)	total_deposits bigint	total_amount bigint
1	deposit	2671	1359168
2	purchase	1617	806537
3	withdrawal	1580	793003

Insights:

- **Deposits dominate** the system with 2,671 transactions totaling **1,359,168**, indicating a strong focus on savings and account growth.
- **Purchases** (1,617 transactions, **806,537**) and **withdrawals** (1,580 transactions, **793,003**) occur at similar levels, reflecting balanced spending and cash-out behavior.

B. CUSTOMER TRANSACTIONS

2. What is the average total historical deposit counts and amounts for all customers?

SQL Query:

```

SELECT
    ROUND(AVG(deposit_count)) AS avg_deposit_times,
    ROUND(AVG(deposit_amount)) AS avg_deposit_amount
FROM (
    SELECT
        customer_id,
        COUNT(*) AS deposit_count,
        AVG(txn_amount) AS deposit_amount
    FROM data_bank.customer_transactions
    WHERE txn_type = 'deposit'
    GROUP BY customer_id
) AS deposit;
    
```

SQL Output:

	avg_deposit_times	avg_deposit_amount
	numeric	numeric
1	5	509

Insights:

On average, each customer has made **5 deposits** historically, with an average deposit amount of **509**. This suggests that customers tend to make a moderate number of deposits over time, with relatively consistent transaction sizes.

B. CUSTOMER TRANSACTIONS

3. For each month - how many Data Bank customers make more than 1 deposit and either 1 purchase or 1 withdrawal in a single month?

SQL Query:

```

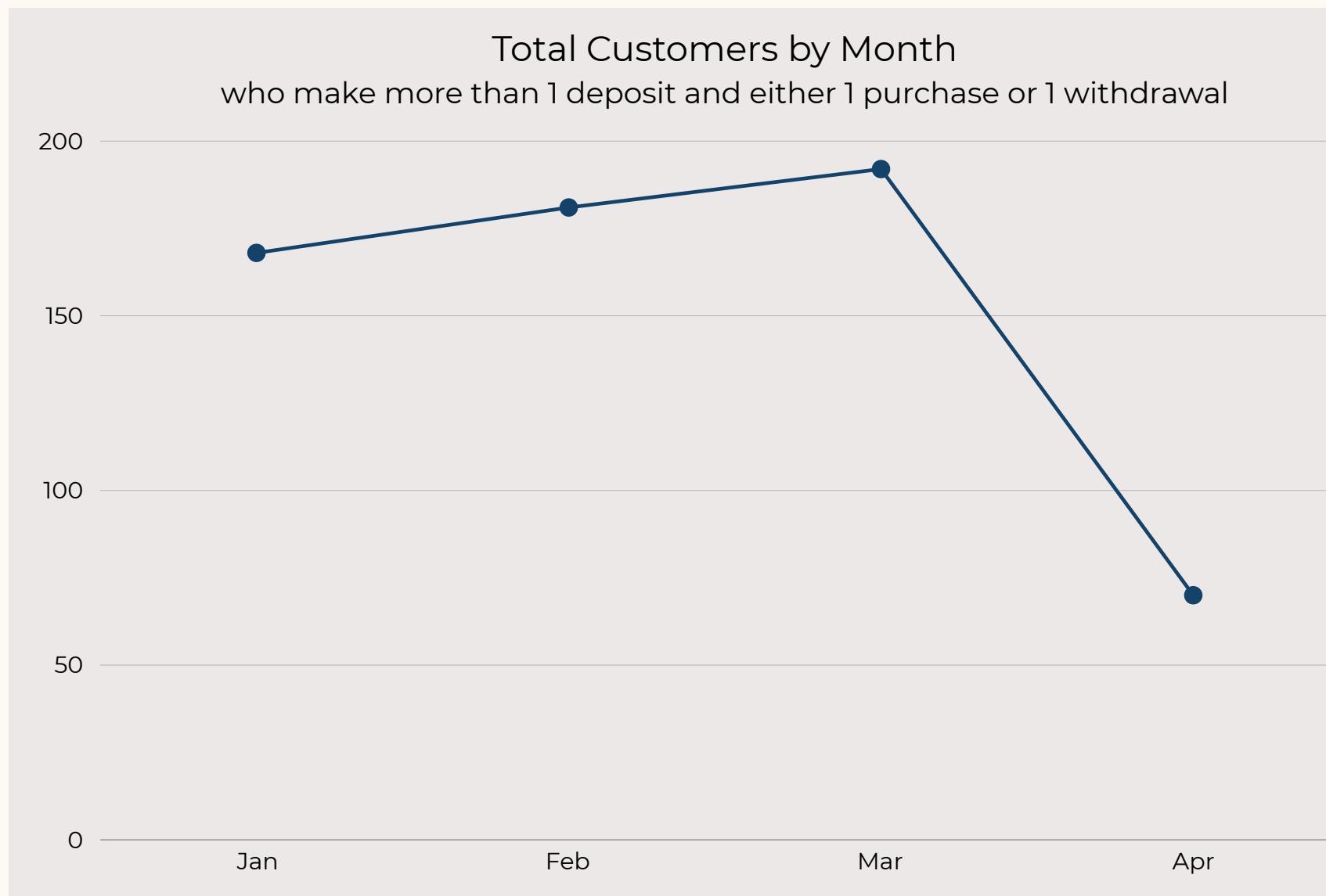
WITH customer_dep_pur_with_count AS (SELECT
    customer_id,
    EXTRACT(MONTH FROM txn_date) AS month,
    SUM(CASE WHEN txn_type = 'deposit' then 1 else 0 end) as total_deposit_count,
    SUM(CASE WHEN txn_type = 'purchase' then 1 else 0 end) as total_purchase_count,
    SUM(CASE WHEN txn_type = 'withdrawal' then 1 else 0 end) as total_withdrawal_count
FROM
    data_bank.customer_transactions
GROUP BY
    customer_id, month
ORDER BY
    customer_id, month)
SELECT
    MONTH, COUNT(DISTINCT customer_id) as total_customers
FROM
    customer_dep_pur_with_count
WHERE
    total_deposit_count > 1 AND (total_purchase_count >= 1 OR total_withdrawal_count >=1)
GROUP BY month
ORDER BY month;
  
```

SQL Output:

	month numeric	total_customers bigint
1	1	168
2	2	181
3	3	192
4	4	70

B. CUSTOMER TRANSACTIONS

3. For each month - how many Data Bank customers make more than 1 deposit and either 1 purchase or 1 withdrawal in a single month?



Insights:

- The number of customers making **more than 1 deposit** and **at least 1 purchase or withdrawal** increased from **168 in January** to **192 in March**, showing a trend of rising account engagement.
- April saw a drop to **70 customers**, indicating a possible seasonal or situational decline in activity.

B. CUSTOMER TRANSACTIONS

4. What is the closing balance for each customer at the end of the month?

SQL Query:

```

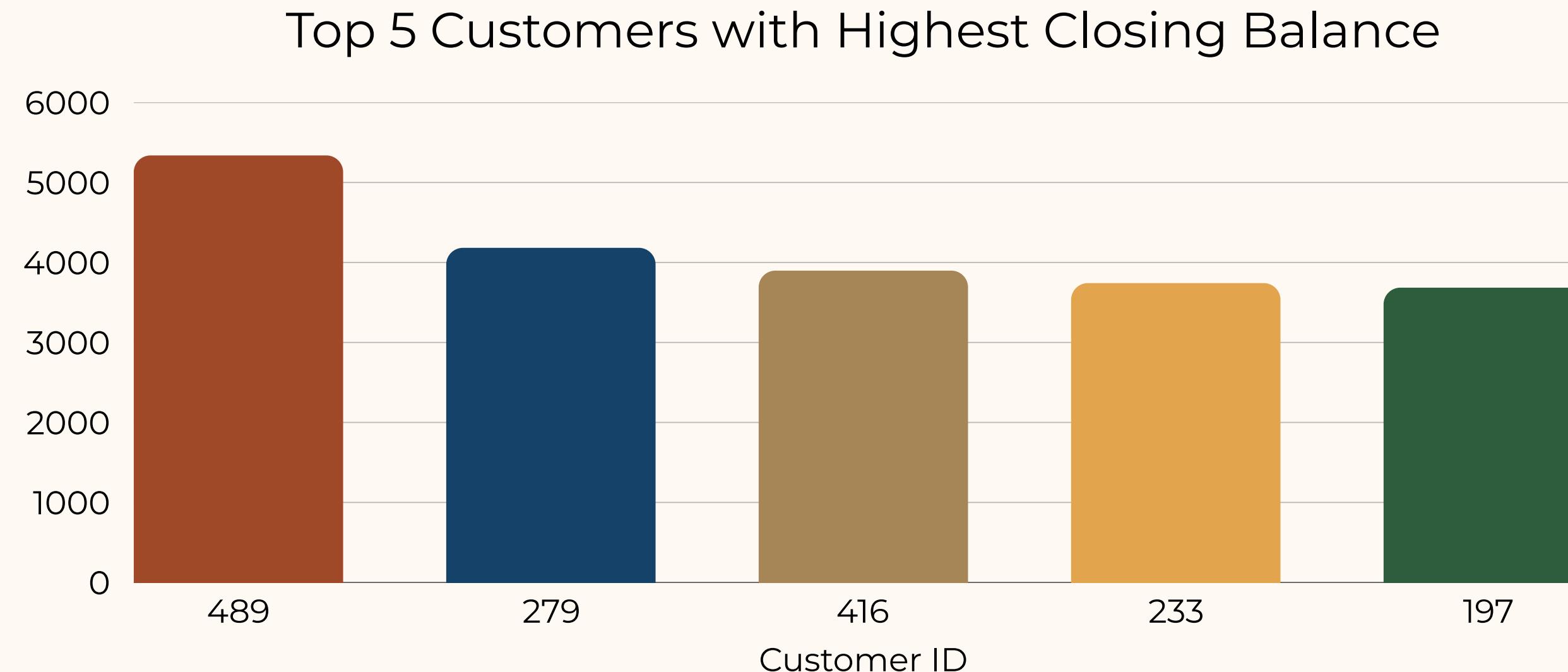
SELECT
  customer_id,
  txn_month,
  SUM(net_change) OVER (
    PARTITION BY customer_id
    ORDER BY txn_month
  ) AS closing_balance
FROM (
  SELECT
    customer_id,
    DATE_TRUNC('month', txn_date) AS txn_month,
    SUM(
      CASE
        WHEN txn_type = 'deposit' THEN txn_amount
        WHEN txn_type IN ('withdrawal', 'purchase') THEN -txn_amount
        ELSE 0
      END
    ) AS net_change
  FROM data_bank.customer_transactions
  GROUP BY customer_id, DATE_TRUNC('month', txn_date)
) AS monthly_change
ORDER BY customer_id, txn_month;
  
```

SQL Output:

	customer_id	txn_month	closing_balance
	integer	timestamp with time zone	numeric
1	1	2020-01-01 00:00:00+07	312
2	1	2020-03-01 00:00:00+07	-640
3	2	2020-01-01 00:00:00+07	549
4	2	2020-03-01 00:00:00+07	610
5	3	2020-01-01 00:00:00+07	144
6	3	2020-02-01 00:00:00+07	-821
7	3	2020-03-01 00:00:00+07	-1222
8	3	2020-04-01 00:00:00+07	-729
9	4	2020-01-01 00:00:00+07	848
10	4	2020-03-01 00:00:00+07	655

B. CUSTOMER TRANSACTIONS

4. What is the closing balance for each customer at the end of the month?



Insights: The chart highlights the top 5 customers with the highest closing balances, showing a clear lead by Customer 489, followed by four others with relatively similar balances.

B. CUSTOMER TRANSACTIONS

5. What is the percentage of customers who increase their closing balance by more than 5%?

SQL Query:

```

WITH monthly_closing_balance AS (
  SELECT
    customer_id,
    EXTRACT(MONTH FROM txn_date) AS month,
    SUM(CASE
      WHEN txn_type = 'deposit' THEN txn_amount
      WHEN txn_type IN ('withdrawal', 'purchase') THEN -txn_amount
      ELSE 0
    END) AS balance
  FROM data_bank.customer_transactions
  GROUP BY customer_id, month
  ORDER BY customer_id, month
),
balance_with_previous AS (
  SELECT
    customer_id,
    month,
    balance,
    LAG(balance) OVER (PARTITION BY customer_id ORDER BY month) AS prev_balance
  FROM monthly_closing_balance
),
increase_over_5_percent AS (SELECT *
  FROM balance_with_previous
  WHERE prev_balance IS NOT NULL AND balance > prev_balance AND ((balance/prev_balance) > 1.05))
,
final_percentage AS (SELECT
  COUNT(DISTINCT customer_id) * 100.0 /
  (SELECT COUNT(DISTINCT customer_id) FROM data_bank.customer_transactions) AS percent_increased
  FROM increase_over_5_percent)
SELECT round(percent_increased,2) || '%' AS increase_over_5_percent FROM final_percentage;
  
```

SQL Output:

increase_over_5_percent	
	text
1	13.20%

Insights:

Approximately 13.20% of customers have increased their closing balance by more than 5% in a given month. This indicates that a portion of customers are seeing steady growth in their account balances, which could reflect successful savings or positive financial activity. Monitoring this metric is valuable for Data Bank to assess customer engagement and identify strategies to help more customers achieve similar growth.

20

C. DATA ALLOCATION CHALLENGE

To test out a few different hypotheses - the Data Bank team wants to run an experiment where different groups of customers would be allocated data using 3 different options:

- Option 1: data is allocated based off the amount of money at the end of the previous month
- Option 2: data is allocated on the average amount of money kept in the account in the previous 30 days
- Option 3: data is updated real-time

For this multi-part challenge question - you have been requested to generate the following data elements to help the Data Bank team estimate how much data will need to be provisioned for each option:

- running customer balance column that includes the impact each transaction
- customer balance at the end of each month
- minimum, average and maximum values of the running balance for each customer

1. Running Customer Balance by Month

SQL Query:

```

204 v WITH running_balance_by_customer AS (
205   SELECT
206     customer_id,
207     txn_date,
208     txn_type,
209     txn_amount,
210     SUM(CASE
211       WHEN txn_type = 'deposit' THEN txn_amount
212       WHEN txn_type IN ('withdrawal', 'purchase') THEN - txn_amount
213       ELSE 0
214     END) OVER (
215       PARTITION BY customer_id
216         ORDER BY txn_date) AS running_balance
217   FROM data_bank.customer_transactions),
218   running_balance_by_month AS (
219     SELECT
220       EXTRACT(MONTH FROM txn_date) as month,
221       SUM(running_balance)
222     FROM running_balance_by_customer
223     GROUP BY month
224     ORDER BY month)
225   SELECT AVG(sum) as average_balance_needed FROM running_balance_by_month;

```

SQL Output:

average_balance_needed	numeric
-245675.750000000000	locked

Insights:

The average running balance is **-245,675.75**, which is the amount the bank would have on a monthly basis.

2. Closing Customer Balance by Month

SQL Query:

```

228 ✓ WITH closing_balance_by_customer AS (SELECT
229     customer_id,
230     end_of_month,
231     SUM(net_change)
232     OVER (PARTITION BY customer_id
233         ORDER BY end_of_month) AS closing_balance
234     FROM (SELECT
235         customer_id,
236         EXTRACT(MONTH FROM txn_date) AS end_of_month,
237         SUM(CASE
238             WHEN txn_type = 'deposit' THEN txn_amount
239             WHEN txn_type IN ('withdrawal', 'purchase') THEN -txn_amount
240             ELSE 0
241         END
242     ) AS net_change|
243     FROM data_bank.customer_transactions
244     GROUP BY customer_id, end_of_month
245     ORDER BY customer_id, end_of_month
246 ) AS monthly_change
247     ORDER BY customer_id, end_of_month),
248
249 closing_balance_by_month AS (
250     SELECT
251         end_of_month, SUM(closing_balance)
252     FROM
253         closing_balance_by_customer
254     GROUP BY end_of_month
255     ORDER BY end_of_month)
256
257     SELECT AVG(sum) as average_balance_needed
258     FROM closing_balance_by_month;

```

SQL Output:

	average_balance_needed	
1	-71007.500000000000	

Insights:

The average closing balance is **-71,007.5**, which is the amount the bank would have on a monthly basis.

3. Min, Max, and Average Running Balance by Month

SQL Query:

```

262 ✓ WITH running_balance AS (
263     SELECT customer_id,
264         txn_date,
265         txn_type,
266         txn_amount,
267         SUM(CASE
268             WHEN txn_type = 'deposit' THEN txn_amount
269             WHEN txn_type IN ('withdrawal', 'purchase') THEN -txn_amount
270             ELSE 0
271         END) OVER (PARTITION BY customer_id ORDER BY txn_date) AS running_balance
272     FROM data_bank.customer_transactions
273 ),
274 balance_by_customer AS (SELECT customer_id, EXTRACT(MONTH FROM txn_date) as month,
275     MIN(running_balance) AS min_balance,
276     AVG(running_balance) AS avg_balance,
277     MAX(running_balance) AS max_balance
278     FROM running_balance
279     GROUP BY customer_id,month
280     ORDER BY customer_id,month),
281 balance_by_month AS (
282     SELECT
283         month,
284         sum(min_balance) as min,
285         round(sum(avg_balance)) as avg,
286         sum(max_balance) as max
287     FROM
288         balance_by_customer
289     GROUP BY month
290     ORDER BY month)
291     SELECT
292         AVG(min) as total_min_balance,
293         AVG(avg) as total_avg_balance,
294         AVG(max) as total_max_balance
295     FROM balance_by_month;

```

SQL Output:

	total_min_balance numeric	total_avg_balance numeric	total_max_balance numeric
1	-194750.500000000000	-21176.250000000000	153039.500000000000

Insights:

- The minimum balance of **-194,750.5**
- The average balance of **-21,176.25**
- The maximum balance of **153,039.5**



THANK YOU FOR LISTENING

DO YOU HAVE ANY QUESTIONS FOR US?