MINISTRY OF EDUCATION AND TRAINING

EASTERN INTERNATIONAL UNIVERSITY

MIS 443

BUSINESS DATA MANAGEMENT

Final Project

Case Study 4 - Data Bank

Lecturers: ***Mr. Dang Thai Doan***

Ms. Huynh Tuyet Ngan

GROUP 3A	
Name	**IRN**
Nguyễn Thành Đạt	2132300562
Lê Nguyễn Tâm Như	2132300065
Nguyễn Hoàng Vinh	2132300522

Quarter 4/2024-2025

Date Submission: 13/8/2025

Table Of Content

1. Objective

The objective of this report is to analyze the Data Bank case study by applying SQL queries to address key business questions. The analysis involves understanding customer behavior, transaction patterns, and regional performance using real-world data. The aim is to extract actionable insights that align with business objectives, which can inform decisions about customer segmentation, regional marketing strategies, and operational optimizations. This report highlights the SQL skills employed to develop queries and derive meaningful conclusions from the data, focusing on enhancing analytical thinking and problem-solving abilities in a business context.

2. Project Overview: Data Bank

2.1. Introduction

The "Data Bank" project simulates a digital-only bank integrated with a secure distributed data storage platform. The bank allocates data storage limits based on the amount of money customers hold in their accounts. The aim of the project is to analyze various datasets related to customer transactions and data allocations, and to provide insights into Data Bank's operations. This analysis will help in better understanding the customer base, forecasting future storage needs, and addressing core business questions for improved strategic planning.

In this project, we utilize SQL queries to explore, aggregate, and analyze data from three main tables: regions, customer_nodes, and customer_transactions. The goal is to uncover patterns, summarize important metrics, and answer business-critical questions that will assist in Data Bank's growth and operational efficiency.

2.2. Key Data Elements

- Regions - Information about different geographical regions where Data Bank operates.
- Customer Nodes - Details on customer allocation to specific nodes (data storage locations) along with their active dates.
- Customer Transactions - Records of customer financial activities, including deposits, withdrawals, and purchases made using their Data Bank debit card.

3. Deliverables

A. Customer Nodes Exploration

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

SQL Query:

```sql
select
    count(distinct node_id) as total_unique_nodes
from
    data_bank.customer_nodes;
```

SQL output:

	total_unique_nodes bigint
1	5

Insights:

The Data Bank system has 5 unique nodes, indicating a distributed infrastructure that enhances data security and redundancy. This setup ensures efficient storage management, reduces the risk of data loss, and supports scalability as the customer base grows.

Question 2 What is the number of nodes per region?

SQL Query:

```sql
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

Insights:

Each region in the Data Bank system contains exactly 5 unique nodes. This uniform distribution across all regions suggests that Data Bank aims for equal infrastructural capacity globally. It reflects a strategy to ensure consistent data storage and security, reducing regional disparities and providing balanced access to their services across different regions.

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

Insights:

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.

Question 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

Insights:

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. Regular node reallocation ensures that data is distributed evenly and reduces the risk of data breaches or system vulnerabilities, providing customers with enhanced security and reliable service.

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

SQL Query:

```
SELECT
    rg.region_name,
    PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY nd.end_date - nd.start_date) AS median_days,
    PERCENTILE_CONT(0.8) WITHIN GROUP (ORDER BY nd.end_date - nd.start_date) AS p80_days,
    PERCENTILE_CONT(0.95) WITHIN GROUP (ORDER BY nd.end_date - nd.start_date) AS p95_days
FROM
    data_bank.customer_nodes as nd
INNER JOIN
    data_bank.regions as rg on rg.region_id = nd.region_id
WHERE
    end_date != '9999-12-31'
GROUP BY
    rg.region_name
ORDER BY
    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

Insights:

The median, 80th, and 95th percentiles for reallocation days vary slightly across regions. The median is consistently 15 days for all regions, while the 80th percentile ranges from 23 to 24 days, and the 95th percentile remains 28 days. This indicates that while most customers are reallocated within 15 days, some regions experience slightly longer reallocation periods, especially for a small portion of customers at the 80th percentile. These variations suggest that Data Bank's reallocation strategy may differ slightly by region, likely influenced by factors such as data load or operational differences.

B. Customer Transactions

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

SQL Query:

```
SELECT
    txn_type,
    COUNT(customer_id) as total_transactions,
    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_transactions bigint	total_amount bigint
1	deposit	2671	1359168
2	purchase	1617	806537
3	withdrawal	1580	793003

Insights:

The analysis shows that deposit transactions are the most frequent, with 2,671 occurrences and a total of 1,359,168. Purchase transactions follow with 1,617 occurrences and 806,537, while withdrawals have 1,580 occurrences totaling 793,003. This indicates that deposits are the primary activity in the system, with purchases and withdrawals representing balanced

financial behavior. The high deposit total underscores the focus on customer savings and account growth.

Question 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 numeric	avg_deposit_amount 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. The deposit amount indicates typical individual deposit values, reflecting steady but not excessively large contributions into customer accounts.

Question 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:

```sql
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

Insights:

The results show how many customers made more than 1 deposit and at least 1 purchase or withdrawal in each month. In the first month (January), 168 customers met this criterion, while 181 customers did so in February and 192 customers in March. The data indicates a growing trend of active engagement among customers, with a noticeable increase in transaction activity, likely reflecting more frequent use of their accounts for deposits and other transactions as the months progress.

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

SQL Query:

```sql
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 integer	txn_month timestamp with time zone	closing_balance 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
11	5	2020-01-01 00:00:00+07	954
12	5	2020-03-01 00:00:00+07	-1923
13	5	2020-04-01 00:00:00+07	-2413
14	6	2020-01-01 00:00:00+07	733
15	6	2020-02-01 00:00:00+07	-52

Insights:

The SQL query returns each customer's closing balance at the end of every month, with positive values indicating net deposits and negative values indicating net withdrawals or spending. From the output, we can observe varied financial patterns: some customers maintain positive balances across months, while others experience significant declines, sometimes turning negative. These fluctuations highlight differences in spending habits, saving behavior, and potential financial risk, which could inform targeted financial advice or interventions.

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

SQL Query:

```sql
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
),
```

```sql

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.

C. Data Allocation Challenge

Question: 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***

Using all of the data available - how much data would have been required for each option on a monthly basis?

a. *Running customer balance column that includes the impact each transaction*

SQL Query:

```
WITH running_balance_by_customer AS (
    SELECT
        customer_id,
        txn_date,
        txn_type,
        txn_amount,
        SUM(CASE
              WHEN txn_type = 'deposit' THEN txn_amount
              WHEN txn_type IN ('withdrawal', 'purchase') THEN - txn_amount
              ELSE 0
           END) OVER (
                    PARTITION BY customer_id
                      ORDER BY txn_date) AS running_balance
FROM data_bank.customer_transactions),
running_balance_by_month AS (
    SELECT
        EXTRACT(MONTH FROM txn_date) as month,
        SUM(running_balance)
    FROM running_balance_by_customer
    GROUP BY month
    ORDER BY month)
SELECT AVG(sum) as average_balance_needed FROM running_balance_by_month;
```

SQL output:

	average_balance_needed numeric
1	-245675.750000000000

Insights:

The running balance reflects the cumulative impact of each transaction, showing how each deposit, withdrawal, or purchase affects a customer's balance. In this case, the average running balance is -245675.75, indicating a significant fluctuation in customer accounts.

b. *Customer balance at the end of each month*

SQL Query:

```sql
WITH closing_balance_by_customer AS (SELECT
  customer_id,
  end_of_month,
  SUM(net_change)
  OVER (PARTITION BY customer_id
    ORDER BY end_of_month) AS closing_balance
FROM (SELECT
    customer_id,
    EXTRACT(MONTH FROM txn_date) AS end_of_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, end_of_month
  ORDER BY customer_id, end_of_month
) AS monthly_change
ORDER BY customer_id, end_of_month),

closing_balance_by_month AS (
    SELECT
        end_of_month, SUM(closing_balance)
    FROM
        closing_balance_by_customer
    GROUP BY end_of_month
    ORDER BY end_of_month)

SELECT  AVG(sum) as average_balance_needed
FROM    closing_balance_by_month;
```

SQL output:

	average_balance_needed numeric
1	-71007.500000000000

Insights:

The closing balance at the end of each month shows the final financial state of each customer, accounting for all transactions. Here, the closing balance is -71007.5, indicating that some customers have a negative balance, suggesting more withdrawals or purchases than deposits.

c. *Minimum, average and maximum values of the running balance for each customer*

SQL Query:

```sql
WITH running_balance AS (
    SELECT customer_id,
           txn_date,
           txn_type,
           txn_amount,
           SUM(CASE
                   WHEN txn_type = 'deposit' THEN txn_amount
                   WHEN txn_type IN ('withdrawal', 'purchase') THEN -txn_amount
                   ELSE 0
               END) OVER (PARTITION BY customer_id ORDER BY txn_date) AS running_balance
    FROM data_bank.customer_transactions
),
balance_by_customer AS (SELECT customer_id, EXTRACT(MONTH FROM txn_date) as month,
       MIN(running_balance) AS min_balance,
       AVG(running_balance) AS avg_balance,
       MAX(running_balance) AS max_balance
FROM running_balance
GROUP BY customer_id,month
ORDER BY customer_id,month),
balance_by_month as (
    SELECT
        month,
        sum(min_balance) as min,
        round(sum(avg_balance)) as avg,
        sum(max_balance) as max
    FROM
        balance_by_customer
    GROUP BY month
    ORDER BY month)
SELECT
    AVG(min) as total_min_balance,
    AVG(avg) as total_avg_balance,
    AVG(max) total_max_balance
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, average, and maximum balances provide a range of each customer's financial activity. The minimum balance of -194750.5 shows the lowest point of the balance, indicating a significant negative fluctuation. The average balance of -21176.25 reflects the typical financial status of the customer, showing an overall negative trend. Meanwhile, the maximum balance of 153039.5 represents the peak financial state, highlighting periods of higher account activity. These values are crucial for understanding the variability in customer balances and help determine the range of data allocation needs based on these fluctuations.

4. Final Summary

This report provides a comprehensive analysis of the Data Bank case study using SQL queries to answer key business questions, focusing on customer behavior, transaction patterns, and data allocation strategies.

4.1. Customer Nodes

Data Bank has an effective node distribution system across its regions, ensuring secure and efficient data storage. Customers are reallocated to different nodes approximately every 23-24 days, indicating a systematic and balanced approach to ensuring both data security and reliability.

4.2. Customer Transactions

Deposits are the most common type of transaction in the system, with customers regularly depositing funds. The average deposit per customer remains steady, and 13.20% of customers have seen their closing balance increase by more than 5% in a given month, indicating positive growth in their accounts.

4.3. Data Allocation Challenge

Three data allocation methods were explored:

- Option 1: Data allocation based on the end-of-month balance.
- Option 2: Data allocation based on the average balance over the past 30 days.
- Option 3: Real-time data updates for each transaction.

Each method offers unique advantages, with Option 2 providing a more dynamic allocation strategy that accounts for fluctuations in customer balances, making it a more flexible option for data management.