Bank Fraud Detection Design

by

Tahsin Nayeem Shrestha

Introduction

Tahsin Nayeem Shrestha

Bank Fraud Detection Design

MOTIVATION

THE ENTITIES

- **Customer** - This entity represents the bank's customers and includes information.
- **Account** - The Account entity is used to store details about bank accounts.
- **Merchant Transaction** - This entity tracks individual transactions associated with bank accounts and an external merchant.
- **Card** - The Card entity manages information related to credit/debit cards issued to customers.
- **Merchants** - Represents information about merchants involved in transactions. They are not the customers of the bank (account holders).
- **Alerts** - The Alert entity is used to store information about fraud alerts triggered by suspicious activities.
- **Case** - Every Case has some transactions that stem from a same investigation. An investigator is assigned to the case

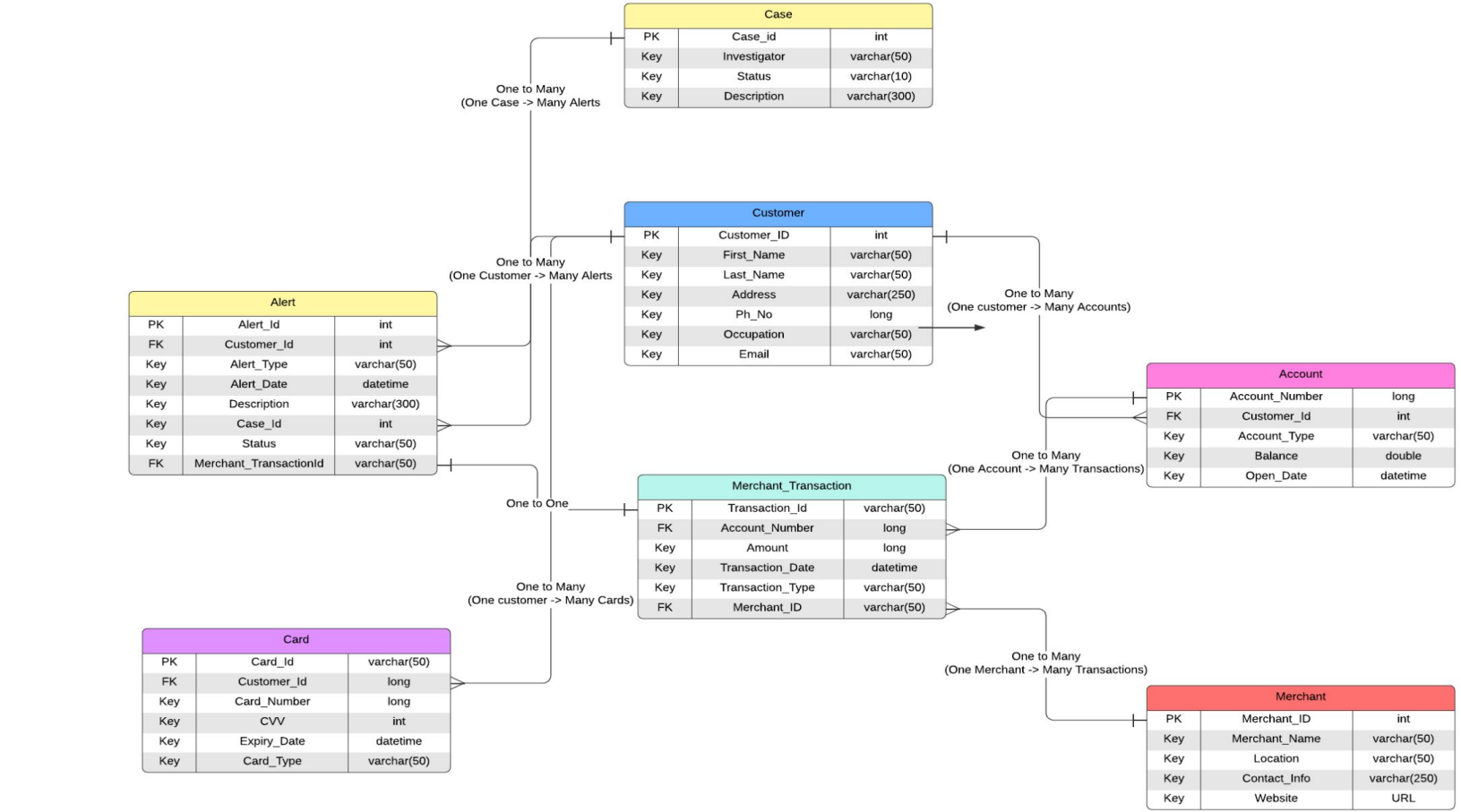
VS Code

MySQL

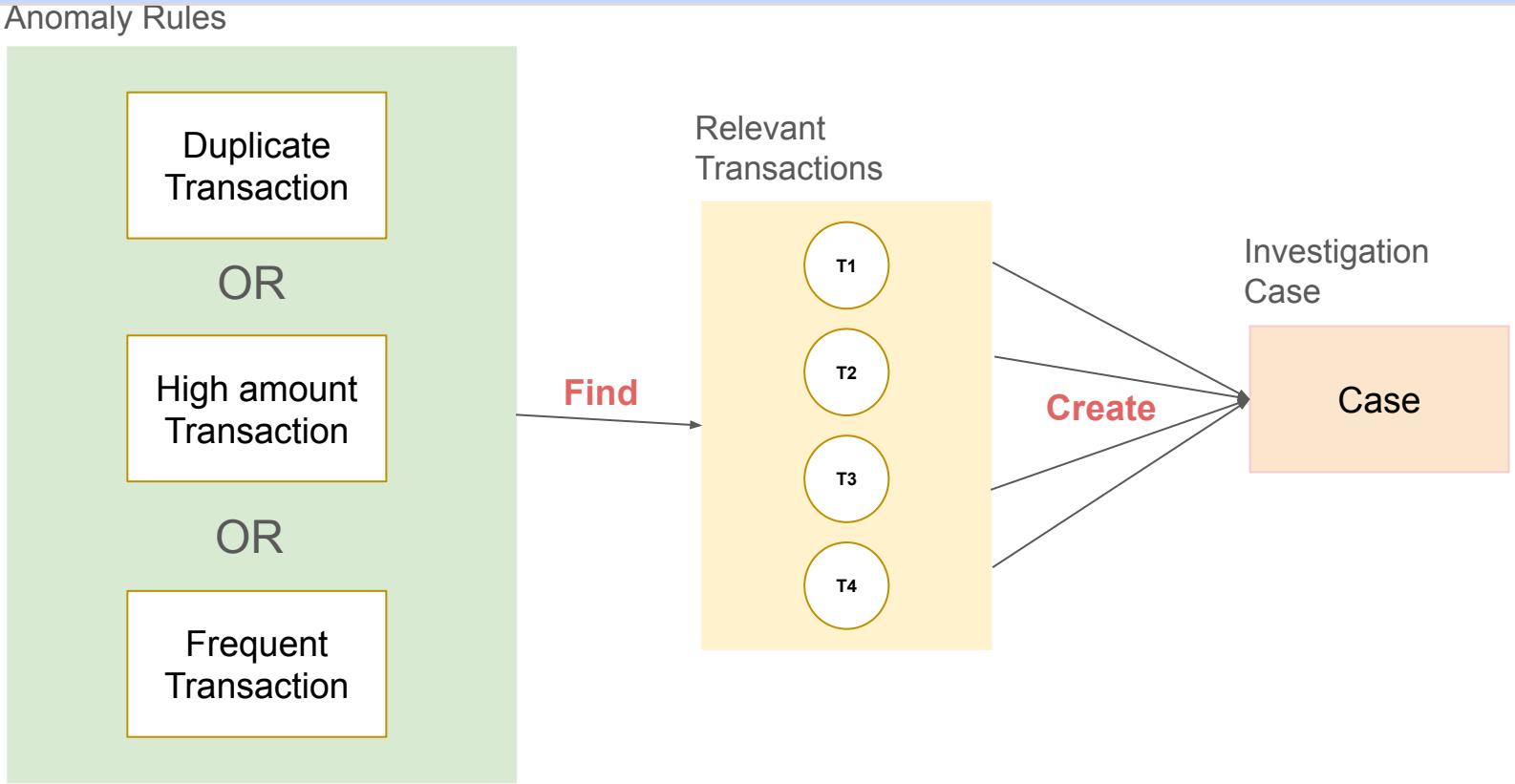
Colab

Python

ENTITY RELATIONSHIP DIAGRAM



FRAUD CASE FILING PIPELINE



NOTE - Fraud is any kind of anomaly detected in the amount transactions

Anomaly Case - 1

Description - Finding suspicious transactions by searching for transactions made more than $5000 in one day

SQL Command:

```
SELECT
GROUP_CONCAT(MerchantTransactionID
) AS
TransactionIDs,FromAccountNumber,
MerchantID,
DATE(TransactionDate) AS
TransactionDate,
GROUP_CONCAT(Amount) AS AmountList
FROM MerchantTransaction
WHERE Amount > 5000
GROUP BY DATE(TransactionDate),
FromAccountNumber,MerchantID
HAVING COUNT(*) >= 2;
```

```
sql> #case-1
    •SELECT
      GROUP_CONCAT(MerchantTransactionID) AS TransactionIDs,
      FromAccountNumber,
      MerchantID,
      DATE(TransactionDate) AS TransactionDate,
      GROUP_CONCAT(Amount) AS AmountList
    FROM
      MerchantTransaction
    WHERE
      Amount > 5000
    GROUP BY
      DATE(TransactionDate), FromAccountNumber, MerchantID
    HAVING
      COUNT(*) >= 2;
```

TransactionIDs	FromAccountNumber	MerchantID	TransactionDate	AmountList
69,70,71	2222222222222228	8	03/09/2022	133330.25,70434.50,8135.00
82,83	2222222222222228	8	03/21/2022	8888.00,30000.00

OK, 2 records retrieved in 1.952ms

Anomaly Case - 2

Description - Sending alert message for suspicious transactions to the Merchant

SQL Command:

```
CREATE TABLE MerchantTransaction
(MerchantTransactionID INT PRIMARY
KEY,
    FromAccountNumber BIGINT,
    Amount DECIMAL(10, 2),
    TransactionDate DATETIME,
    TransactionType VARCHAR(50),
    MerchantID INT(100),
    FOREIGN KEY (MerchantID)
REFERENCES Merchant(MerchantID),
    FOREIGN KEY (FromAccountNumber)
REFERENCES Account(AccountNumber)
);
SELECT * FROM MerchantTransaction
WHERE Amount>=5000;
```

```
sql> SELECT * FROM MerchantTransactionAlert;
```

AlertID	MerchantTransactionID	AlertType	AlertDate	CaseID
1	69	HighTransaction	03/10/2022	1
2	70	HighTransaction	03/10/2022	1
3	71	HighTransaction	03/10/2022	1
4	82	HighTransaction	03/22/2022	2
5	83	HighTransaction	03/22/2022	2

OK, 5 records retrieved in 0.805ms

```
sql> #case-2
    SELECT * FROM MerchantTransaction WHERE Amount>=5000;
```

MerchantTransactionID	FromAccountNumber	Amount	TransactionDate	TransactionType	MerchantID
6	2222222222222226	11320.00	01/06/2022	Purchase	6
13	2222222222222233	70000.50	01/13/2022	Purchase	13
69	2222222222222228	133330.25	03/10/2022	Purchase	8
70	2222222222222228	70434.50	03/10/2022	Purchase	8
71	2222222222222228	8135.00	03/10/2022	Purchase	8
82	2222222222222228	8888.00	03/22/2022	Purchase	8
83	2222222222222228	30000.00	03/22/2022	Purchase	8

OK, 7 records retrieved in 0.893ms

Anomaly Case - 3

Description - Finding Duplicate Transaction made by Merchant with amount, type and date

SQL Command:

```
SELECT MerchantTransactionID,
FromAccountNumber,Amount,Transaction
Date,TransactionType,MerchantID
FROM MerchantTransaction mt1
WHERE
Amount IN (SELECT Amount FROM
 MerchantTransaction mt2
   WHERE mt2.MerchantTransactionID
<> mt1.MerchantTransactionID
     AND mt2.TransactionDate BETWEEN
DATE_SUB(mt1.TransactionDate,
INTERVAL 1 MINUTE)AND
DATE_ADD(mt1.TransactionDate,
INTERVAL 1 MINUTE) );
```

```
sql> #case-3
SELECT
  MerchantTransactionID,
  FromAccountNumber,
  Amount,
  TransactionDate,
  TransactionType,
  MerchantID
FROM
  MerchantTransaction mt1
WHERE
  Amount IN (
    SELECT
      Amount
    FROM
      MerchantTransaction mt2
    WHERE
      mt2.MerchantTransactionID <> mt1.MerchantTransactionID
      AND mt2.TransactionDate BETWEEN
        DATE_SUB(mt1.TransactionDate, INTERVAL 1 MINUTE)
        AND DATE_ADD(mt1.TransactionDate, INTERVAL 1 MINUTE)
  );
```

MerchantTransactionID	FromAccountNumber	Amount	TransactionDate	TransactionType	MerchantID
84	2222222222222221	999.99	03/22/2022	Purchase	5
85	2222222222222221	999.99	03/22/2022	Purchase	5

OK, 2 records retrieved in 30.037ms

Anomaly Case - 4

Description - Finding no.of Alerts for different customer account in order to detect fraud

SQL Command:

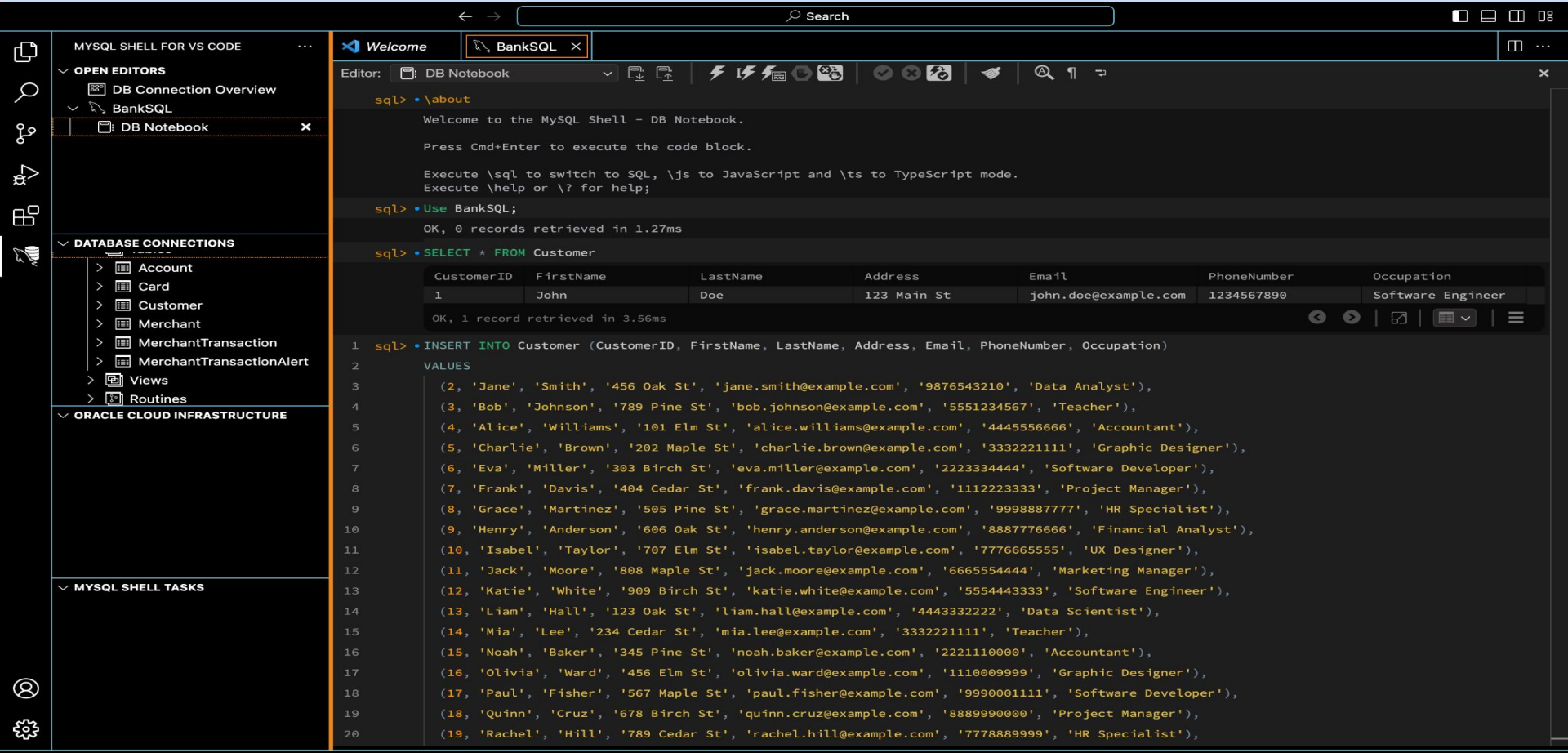
```
SELECT c.CustomerID,
COUNT(mta.AlertID) AS AlertCount FROM
Customer c
LEFT JOIN
Account a ON c.CustomerID =
a.CustomerID
LEFT JOIN MerchantTransaction mt ON
a.AccountNumber = mt.FromAccountNumber
LEFT JOIN
 MerchantTransactionAlert mta ON
mt.MerchantTransactionID =
mta.MerchantTransactionID
GROUP BY c.CustomerID
ORDER BY AlertCount DESC;
```

```
sql> #Case-4
SELECT
    c.CustomerID,
    COUNT(mta.AlertID) AS AlertCount
FROM
    Customer c
LEFT JOIN
    Account a ON c.CustomerID = a.CustomerID
LEFT JOIN
    MerchantTransaction mt ON a.AccountNumber = mt.FromAccountNumber
LEFT JOIN
    MerchantTransactionAlert mta ON mt.MerchantTransactionID = mta.MerchantTransactionID
GROUP BY
    c.CustomerID
ORDER BY
    AlertCount DESC;
```

CustomerID	AlertCount
8	5
1	2
2	0
3	0
4	0
5	0
6	0
7	0
9	0
10	0
11	0
12	0
13	0
14	0

OK, 30 records retrieved in 62.824ms

VS Studio


Search
MYSQL SHELL FOR VS CODE
OPEN EDITORS
DB Connection Overview
BankSQL
DB Notebook
DATABASE CONNECTIONS
Account
Card
Customer
Merchant
MerchantTransaction
MerchantTransactionAlert
Views
Routines
ORACLE CLOUD INFRASTRUCTURE
MYSQL SHELL TASKS
Welcome
BankSQL
Editor: DB Notebook
sql> \about
Welcome to the MySQL Shell - DB Notebook.
Press Cmd+Enter to execute the code block.
Execute \sql to switch to SQL, \js to JavaScript and \ts to TypeScript mode.
Execute \help or \? for help;
sql> Use BankSQL;
OK, 0 records retrieved in 1.27ms
sql> SELECT * FROM Customer
CustomerID FirstName LastName Address Email PhoneNumber Occupation
1 John Doe 123 Main St john.doe@example.com 1234567890 Software Engineer
OK, 1 record retrieved in 3.56ms
sql> INSERT INTO Customer (CustomerID, FirstName, LastName, Address, Email, PhoneNumber, Occupation)
VALUES
(2, 'Jane', 'Smith', '456 Oak St', 'jane.smith@example.com', '9876543210', 'Data Analyst'),
(3, 'Bob', 'Johnson', '789 Pine St', 'bob.johnson@example.com', '5551234567', 'Teacher'),
(4, 'Alice', 'Williams', '101 Elm St', 'alice.williams@example.com', '4445556666', 'Accountant'),
(5, 'Charlie', 'Brown', '202 Maple St', 'charlie.brown@example.com', '3332221111', 'Graphic Designer'),
(6, 'Eva', 'Miller', '303 Birch St', 'eva.miller@example.com', '2223334444', 'Software Developer'),
(7, 'Frank', 'Davis', '404 Cedar St', 'frank.davis@example.com', '1112223333', 'Project Manager'),
(8, 'Grace', 'Martinez', '505 Pine St', 'grace.martinez@example.com', '9998887777', 'HR Specialist'),
(9, 'Henry', 'Anderson', '606 Oak St', 'henry.anderson@example.com', '8887776666', 'Financial Analyst'),
(10, 'Isabel', 'Taylor', '707 Elm St', 'isabel.taylor@example.com', '7776665555', 'UX Designer'),
(11, 'Jack', 'Moore', '808 Maple St', 'jack.moore@example.com', '6665554444', 'Marketing Manager'),
(12, 'Katie', 'White', '909 Birch St', 'katie.white@example.com', '5554443333', 'Software Engineer'),
(13, 'Liam', 'Hall', '123 Oak St', 'liam.hall@example.com', '4443332222', 'Data Scientist'),
(14, 'Mia', 'Lee', '234 Cedar St', 'mia.lee@example.com', '3332221111', 'Teacher'),
(15, 'Noah', 'Baker', '345 Pine St', 'noah.baker@example.com', '2221110000', 'Accountant'),
(16, 'Olivia', 'Ward', '456 Elm St', 'olivia.ward@example.com', '1110009999', 'Graphic Designer'),
(17, 'Paul', 'Fisher', '567 Maple St', 'paul.fisher@example.com', '9990001111', 'Software Developer'),
(18, 'Quinn', 'Cruz', '678 Birch St', 'quinn.cruz@example.com', '8889990000', 'Project Manager'),
(19, 'Rachel', 'Hill', '789 Cedar St', 'rachel.hill@example.com', '7778889999', 'HR Specialist'),

Top Investigations / Future enhancements

- Which Merchant received the same transactions more than once at the same transaction datetime? case-3
- Flagging the Merchant - Which Merchant had most of the alerts associated with them? case-4
- Customer/Merchants that have the most anomalous amount of Transactions? case-2

- Which Customer has record of anomalous record of history and if they will be qualified for loans in the future. Features - associated alerts, qualifications, transaction history, occupation, credit score change, etc. Model - Binary classification ML model to predict qualification of customer for loan approval.
- Future case - If the fraud is positive, we can freeze the customer Cards and take actionable case on the Merchant.

THANK YOU

- Tahsin Nayeem Shrestha