Non-Relational Database – Practicum 3

Fraud Transaction Detection Database

Scenario

In the world of money transactions, it's super important to keep sensitive info safe from fraudulent transactions. Cybercriminals constantly seek innovative methods to exploit vulnerabilities in online transactions, and demand for developing and implementing advanced fraud detection measures increases daily. These measures go beyond protecting against data breaches, and they involve proactive strategies to identify suspicious patterns, irregularities, or anomalies that may indicate fraudulent behavior. As we do more and more of our transactions online, it's crucial to have good ways to check for fraud.

Why NoSQL

That's where NoSQL databases come in handy. They offer a flexible and scalable solution for efficiently managing large volumes of transactional and user-related data. Unlike traditional relational databases, NoSQL databases can handle diverse data types, allowing for the storage and retrieval of unstructured or semi-structured information often encountered in user behavior logs and transaction records. This flexibility is especially beneficial when implementing fraud detection algorithms that analyze patterns and anomalies within user behavior. Additionally, the distributed and horizontally scalable nature of NoSQL databases enhances their performance and responsiveness, ensuring that financial institutions can process and analyze data in real-time, fortifying their ability to detect and prevent fraudulent activities swiftly.

NoSQL databases are suitable for handling lots of unstructured and semi-structured data. These databases are great because they can handle all kinds of data. These databases are also good at working quickly and can handle a ton of data simultaneously. This is important because we can catch any weird or suspicious activity in real time. So, using NoSQL databases makes it easier to keep our money systems safe and ensure everyone's transactions are up and up.

Options of NoSQL Database

CouchDB, a NoSQL database for document management, proves valuable in fraud detection due to its flexible design, robust data structure, and storage efficiency. Its horizontal scalability ensures flexibility to match industry requirement numbers, making it viable for financial institutions. Known for its real-time database capabilities, Firebase offers seamless integration with web and mobile applications. Integrating data in real-time enables rapid detection and response to suspicious activity, making Firebase a useful solution for dynamic online transaction environments. Neo4j is a graph database focusing on complex relationships represented and analyzed in connected data. Its graph-based framework excels in uncovering patterns and anomalies of fraudulent interactions, especially in situations involving complex interactions of networks and user interactions.

Reasons for choosing MongoDB

Among the various options available, MongoDB stands out as the first choice. Its unique strengths make it ideally suited to the demanding fraud detection requirements. MongoDB's document-oriented structure is key to facilitating the storage and easy analysis of complex data structures. This structure effectively represents complex relationships and structures in a single document, and provides a view of user profiles, transaction records, and interconnected services. Its aggregation pipelines and geospatial queries provide efficient data filtering, aggregation, and manipulation, making spotting patterns and anomalies indicative of fraudulent activity easier. For example, MongoDB's ability to create complex aggregates and transformations on data can be due to the limitations of SQL databases, which can hinder the ability to perform microanalysis and discover subtle patterns in the data.

Easy integration is another important benefit offered by MongoDB. Its seamless integration with various third-party tools and services is commonly used in fraud detection. It streamlines overall infrastructure operations. MongoDB's compatibility with the most widely accepted logging, monitoring, and security tools provides its profitability development in a complex system. In contrast, integrating with other NoSQL databases may require additional development efforts or custom solutions, which may delay the project timeline and result in a complex architecture. MongoDB's flexibility in adapting existing technology stacks ensures a smooth integration process and reduces development costs and fraud detection identity authentication systems.

MongoDB's document-oriented structure aligns well with the many facets of user behavior data in fraud detection. MongoDB changes how data is stored and analyzed by allowing complex relationships to be represented in a single document. Users profiles, transaction logs, and associated entities can coexist in an integrated system, enabling comprehensive analysis that reveals patterns and complex relationships in data between points. This is in contrast to traditional databases, where integrating data from disparate sources can be difficult, affecting productivity and performance.

MongoDB reliability is critical in the case of fraud detection and identity authentication, where the consequences of a system failure or data breach can be significant. MongoDB's ability to handle large amounts of data and its robust architecture make it suitable for high performance under heavy caseloads and ensure continuous and safe operation of the systems.

While other NoSQL databases may offer distinct advantages in some areas, MongoDB's unique combination of strengths, including scalability, flexibility, rich query capabilities, ease of integration, active community, and mature platform, make it a possible choice for fraud detection and identity authentication. Query capabilities and easy integration position MongoDB as a leader in this area.

Barebone

This barebones prototype demonstrates MongoDB's fraud detection and identity authentication capabilities. It focuses on the data model and basic applications.

Data Model

Users:

_id: Unique identifier for the user.

username: Username for login.

password: Password for User.

email: User's email address.

name: User's full name.

phone_number: User's phone number.

address: User's address.

created_at: Timestamp of user creation.

updated_at: Timestamp of last user update.

Transactions:

_id: Unique identifier for the transaction.

user_id: Foreign key referencing the user who performed the transaction.

amount: Amount of the transaction.

currency: Currency used for the transaction.

type: Type of transaction.

status: transaction status.

timestamp: Timestamp of the transaction.

ip_address: IP address used for the transaction.

device_id: Unique identifier for the device used for the transaction.

merchant_details: Includes details of merchant.

Implementation

Users:

```
{
  "_id": "11928",
  "username": "ruthvick",
  "password": "Fall@2023",
  "email": "ruthvick@kent.com",
  "name": "Ruthvick B",
  "phone_number": "123-456-7890",
  "address": "118 Summit Street, kent, USA",
  "created_at": "2023-12-01T12:00:00Z",
  "updated_at": "2023-12-02T15:30:00Z",
  "behavior_analysis": {
    "login_history": [
      {
        "timestamp": "2023-12-01T12:00:00Z",
        "location": "Kent"
      },
      {
        "timestamp": "2023-12-02T09:45:00Z",
        "location": "Kent"
      }
    ],
    "purchase_history": [
      {
        "timestamp": "2023-12-01T15:00:00Z",
        "amount": 97.8,
        "merchant": "Puma"
      }
    ]
  }
```

}

Transactions:
Genuine Transaction:

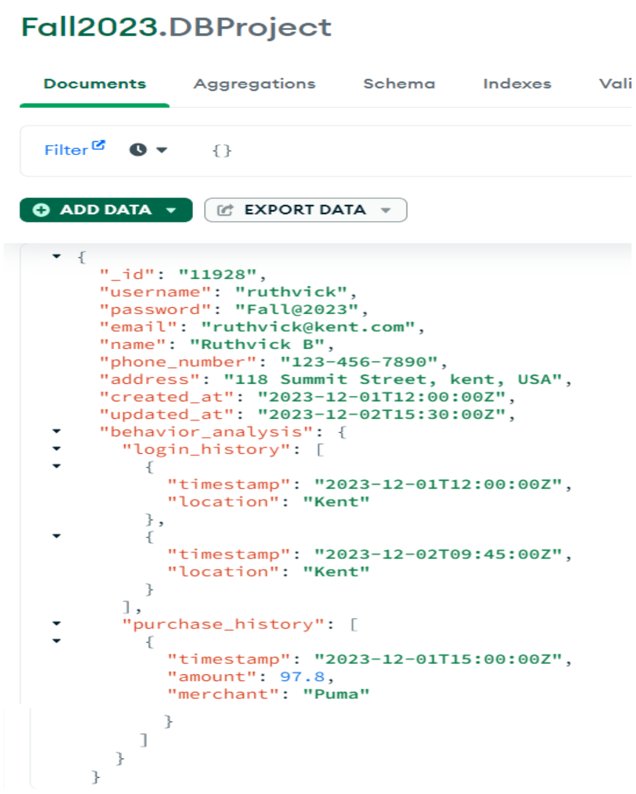
```
{
 "_id": "23",
 "user_id": "11928",
 "amount": 97.8,
 "currency": "USD",
 "type": "purchase",
 "status": "completed",
 "timestamp": "2023-12-02T10:30:00Z",
 "ip_address": "192.168.1.100",
 "device_id": "ABC123XYZ789",
 "merchant_details": {
   "merchant_name": "Puma",
   "location": "Kent",
   "category": "Retail"
 }
}
```

Fraudulent Transaction:

```
{
 "_id": "65",
 "user_id": "11928",
 "amount": 292.16,
 "currency": "USD",
 "type": "purchase",
 "status": "pending",
 "timestamp": "2023-12-13T14:30:00Z",
 "ip_address": "192.168.1.200",
 "device_id": "ZZZ999YYY111",
 "merchant_details": {
   "merchant_name": "LuxuryGoods",
   "location": "Unknown",
   "category": "Luxury"
 }
}
```

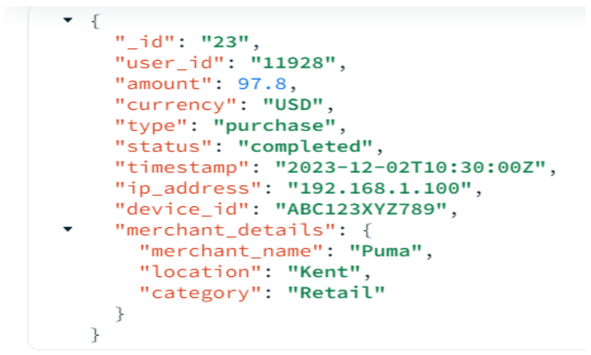
Screenshots of Implementation

Users:



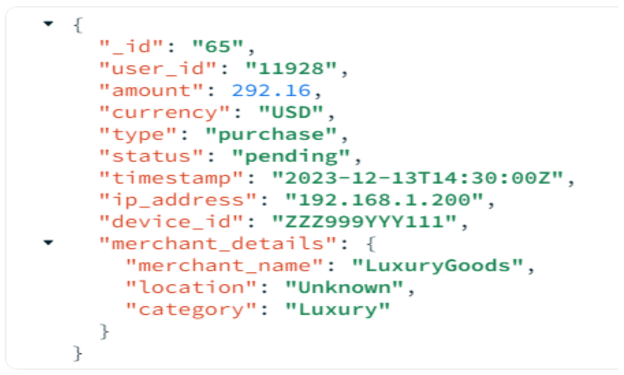
```
{
    "_id": "11928",
    "username": "ruthvick",
    "password": "Fall@2023",
    "email": "ruthvick@kent.com",
    "name": "Ruthvick B",
    "phone_number": "123-456-7890",
    "address": "118 Summit Street, kent, USA",
    "created_at": "2023-12-01T12:00:00Z",
    "updated_at": "2023-12-02T15:30:00Z",
    "behavior_analysis": {
      "login_history": [
        {
          "timestamp": "2023-12-01T12:00:00Z",
          "location": "Kent"
        },
        {
          "timestamp": "2023-12-02T09:45:00Z",
          "location": "Kent"
        }
      ],
      "purchase_history": [
        {
          "timestamp": "2023-12-01T15:00:00Z",
          "amount": 97.8,
          "merchant": "Puma"

        }
      ]
    }
  }
```

Genuine Transaction:



```
{
    "_id": "23",
    "user_id": "11928",
    "amount": 97.8,
    "currency": "USD",
    "type": "purchase",
    "status": "completed",
    "timestamp": "2023-12-02T10:30:00Z",
    "ip_address": "192.168.1.100",
    "device_id": "ABC123XYZ789",
    "merchant_details": {
      "merchant_name": "Puma",
      "location": "Kent",
      "category": "Retail"
    }
  }
```

Fraudulent Transaction:



```
{
    "_id": "65",
    "user_id": "11928",
    "amount": 292.16,
    "currency": "USD",
    "type": "purchase",
    "status": "pending",
    "timestamp": "2023-12-13T14:30:00Z",
    "ip_address": "192.168.1.200",
    "device_id": "ZZZ999YYY111",
    "merchant_details": {
      "merchant_name": "LuxuryGoods",
      "location": "Unknown",
      "category": "Luxury"
    }
  }
```

Conclusion

In evaluating transactions for potential fraud, a user "ruthvick" transaction adheres to established patterns, including routine logins from Kent, a typical purchase at the "Puma" store, and consistent device ID and IP address usage. Conversely, a fraudulent transaction on December 12, 2023, at 01:00 PM, reveals anomalies such as an unknown login location, an unfamiliar "LuxuryStore" merchant, a significantly higher amount of $292.16, and a different device ID and IP address. Effective fraud detection involves scrutinizing anomalies like irregular amounts, unfamiliar locations, and divergent user behavior, bolstered by real-time monitoring and thorough analysis of login and purchase patterns. Transactions with a "pending" status are scrutinized for potential fraudulent activity, ensuring a robust system for identifying and addressing security threats.

Reference

https://www.mongodb.com/industries/financial-services/fraud-prevention

https://journalofbigdata.springeropen.com/articles/10.1186/s40537-015-0025-0

https://www.datastax.com/guides/nosql-use-cases

https://aerospike.com/blog/how-nosql-helps-in-financial-crime-detection-and-fraud-management-medici/#:~:text=Through%20graph%2Dpowered%20investigation%2C%20NoSQL,fraudulent%20link%20between%20these%20entities.