VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

AAT PROJECT REPORT

on

Fraud Detection Dashboard using TigerGraph

Submitted by

Adarsh K N (1BM19CS004)
Anvitha Aravinda (1BM19CS021)
Bharath Mahesh Gera (1BM19CS035)
Bhavya Sharma (1BM19CS036)

Under the Guidance of

Dr. Selva kumar S
Assistant Professor, BMSCE

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

B. M. S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019
Oct-2022 to Feb-2023

B. M. S. College of Engineering

Bull Temple Road, Bangalore 560019

(Affiliated to Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the project work entitled "**Fraud Detection Dashboard using TigerGraph**" carried out by **Adarsh K N (1BM19CS004), Anvitha Aravinda (1BM19CS021), Bharath Mahesh Gera (1BM19CS035) and Bhavya Sharma (1BM19CS036)** who are bonafide students of **B. M. S. College of Engineering.** It is in partial fulfilment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visveswaraiah Technological University, Belgaum during the year 2022-2023. The project report has been approved as it satisfies the academic requirements in respect of **AAT NoSQL (21CS7PENSD)** work prescribed for the said degree.

Signature of the Guide
Dr. Selva kumar S
Assistant Professor
BMSCE, Bengaluru

Signature of the HOD
Dr. Jyothi S Nayak
Associate Prof. & Head, Dept. of CSE

Table of Contents

1. Introduction

Fraud detection is the process of identifying fraudulent activity by analyzing and reviewing information and data for signs of suspicious behavior. This can be done manually by humans, or with the help of specialized software and algorithms. Fraud detection is used in a variety of industries, including finance, insurance, and e-commerce, to protect against financial loss and damage to reputation.

There are various methods and techniques used in fraud detection, including:

Data mining and analysis: This involves analyzing large amounts of data to look for patterns and anomalies that may indicate fraudulent activity.

Rule-based systems: These are systems that are programmed to flag transactions that meet certain predetermined criteria, such as unusually large amounts or unusual patterns of activity.

Artificial intelligence and machine learning: These technologies can be used to analyze data and detect fraud in real-time by learning from past fraudulent activity and adapting to new patterns as they emerge.

Overall, the goal of fraud detection is to identify and prevent fraudulent activity before it causes harm.

Data visualization's significance cannot be overstated. Visualizations enable developers to show "meaningless" data in an efficient, effective, and compelling manner. It's critical for comprehending data and making decisions, and it's useful in any sector. This, of course, encompasses the worlds of business and finance.

As vital as data is, organisations cannot effectively analyse it unless it is presented in an easy-to-understand way. Visualizations are critical in sectors such as fraud detection, where accurately comprehending data is critical to halting harmful activities and saving billions of dollars.

2. Client-side Design

2.1 Software tools:

Dash is a Python framework for building web applications. It is built on top of the Plotly.js JavaScript library and is designed to make it easy to build interactive, data-driven applications. With Dash, you can create beautiful visualizations and use them to build powerful, interactive dashboards.

Some key features of Dash include:

Easy to use: Dash is designed to be easy to use for Python developers, with a simple API and built-in support for a wide range of data visualization libraries.

Interactive: Dash allows you to create interactive, data-driven apps that allow users to explore and interact with your data in real-time.

Customizable: Dash gives you complete control over the look and feel of your app, so you can create a unique and professional-looking dashboard that meets your specific needs.

Whether you're a data scientist, a business analyst, or a developer, Dash is a powerful tool that can help you build interactive, data-driven applications quickly and easily.

It also comes with widgets such as sliders, text boxes, and navigation bars, all of which are extremely easy to use. Dash abstracts away the complicated HTML, CSS and JS and allows you to add components with jsut a few lines of code. But, they also offer the option to add custom styling or custom components, giving users true freedom in creating their dashboard.

Plotly Dash is a powerful open-source library for creating interactive web-based data visualization applications. It is built on top of the popular Plotly JavaScript library and allows developers to create sophisticated visualizations by writing just a few lines of Python code.

One of the key features of Plotly Dash is its ability to handle large and complex datasets, making it well-suited for tasks like real-time streaming data visualization and analytics. It also supports a wide range of chart types, including line charts, bar charts, scatter plots, and more, and allows users to easily customize the appearance of their visualizations using a variety of formatting options.

In addition to its visualization capabilities, Plotly Dash also includes a built-in server and web application framework, allowing developers to build and deploy their applications without the need for any additional software. This makes it easy to build interactive dashboards and data applications that can be accessed from any device with a web browser.

Overall, Plotly Dash is a powerful and flexible tool for creating interactive data visualizations and applications, and is widely used in a variety of industries and applications.

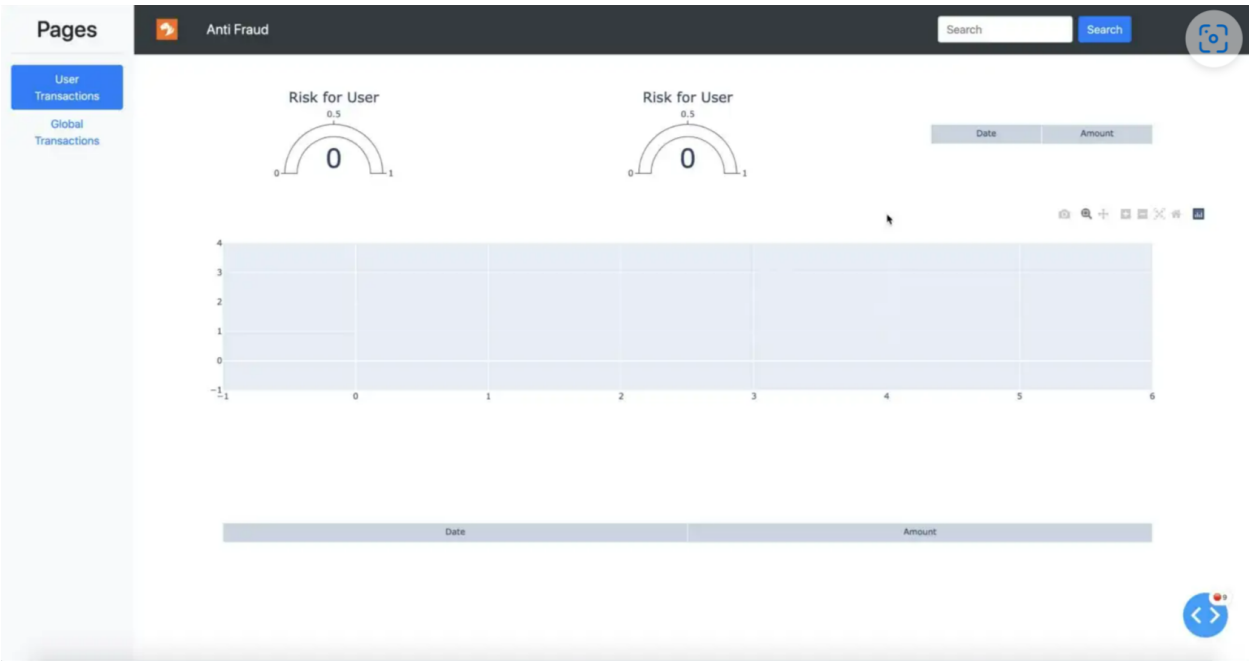
Google Colab (short for Colaboratory) is a cloud-based programming environment that allows users to write and execute code in a variety of languages, including Python, R, and Julia. It is designed to be a powerful and flexible tool for data scientists, machine learning engineers, and researchers, and can be used for tasks such as data analysis, machine learning model development and training, and scientific computing.

One of the key features of Colab is that it acts as a client tool, meaning that it runs in a web browser and connects to cloud-based resources for processing and storage. This makes it easy for users to access their Colab notebooks and code from any device with an internet connection, and eliminates the need to install and maintain local software.

Colab also includes a variety of built-in features that make it easy to use, such as integration with Google Drive for file storage and collaboration, support for Jupyter notebooks, and access to powerful computing resources, including GPUs and TPUs, for running machine learning and other computationally-intensive tasks.

Overall, Colab is a powerful and convenient tool for working with data and code, and is widely used by data scientists and other technical professionals around the world.

2.2 User Transactions



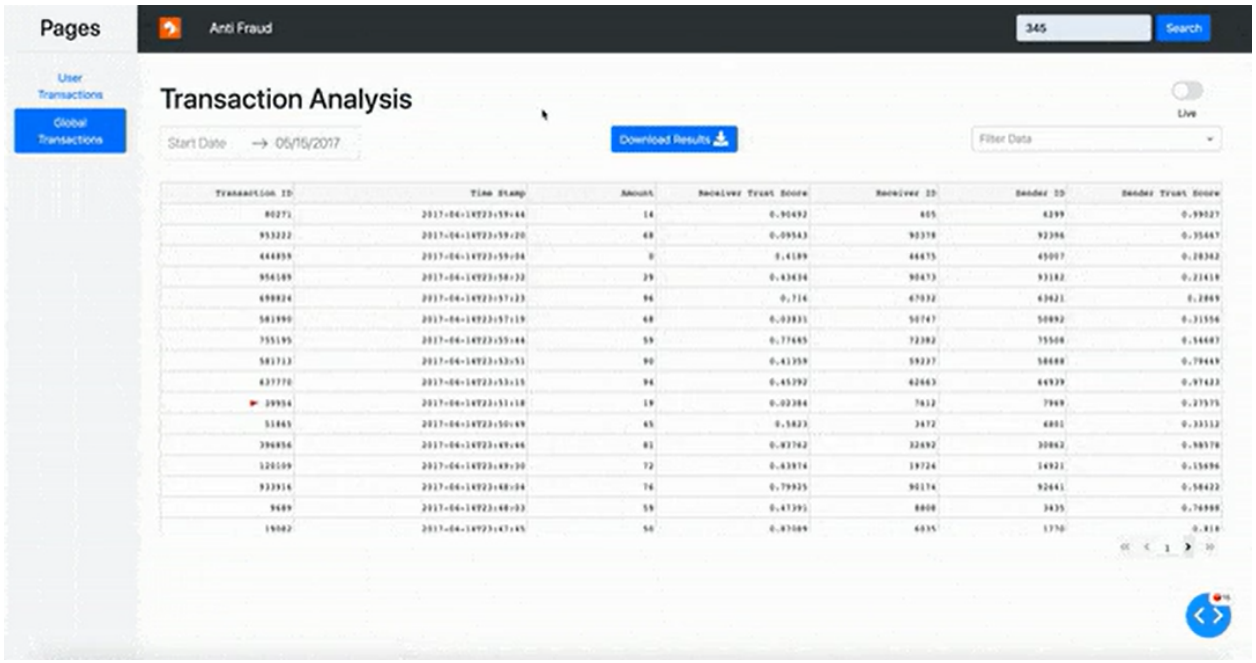
Query for User 333.

Average Trust Score for User 333

The user page consists of :

1. 2 dials : 1 dial shows the avg trust score and other one the current trust score of a particular user
2. Bullet Chart : This plots the aggregate of the user's transactions based on months
3. Table : This displays the user's transaction in terms of table. Consists of Date of transaction, Amount of transaction and Type of transaction (Transferred or Received)

2.3 Global Transactions



Transaction Analysis consists of transaction of whole user database. The same can be downloaded from the website.

The transaction table consists of :

1. ID
2. Time Stamps
3. Amount of transaction
4. Receiver Trust Score
5. Receiver ID
6. Sender ID
7. Sender Trust Score

The date of transaction can be set and even filters can be applied to get transactions above or below the certain limit.

3. Server-side Design

3.1 Software tools/Programming language:

TigerGraph Server is a software component that is part of the TigerGraph platform. It is the main server component of TigerGraph and is responsible for managing and executing graph queries, as well as providing access to the graph data and analytics functionality of TigerGraph.

Some key features of TigerGraph Server include:

Graph query execution: TigerGraph Server is responsible for executing graph queries written in the GSQL query language. It is optimized for fast query processing, making it well-suited for applications that require low-latency queries and real-time analytics.

Data management: TigerGraph Server manages the storage and retrieval of graph data, as well as handling data import and export.

User management: TigerGraph Server provides support for managing users and their permissions, allowing you to control access to your graph data and analytics functionality.

Graph analytics: TigerGraph Server provides a range of built-in graph analytics functions, such as shortest path, centrality, and community detection, that can be used to gain insights from your graph data.

pyTigerGraph is Python package that manages connecting to your TigerGraph server and working with the built-in REST endpoints from TigerGraph.

3.2 Database connectivity code

```
configs = {
   "host": "https://YOUR_URL",
   "password": "YOUR_PASSWORD",
   "graphname": "YOUR_GRAPHNAME"
}


conn = tg.TigerGraphConnection(host=configs['host'], password=configs['password'], gsqlVersion="3.0.5", useCert=True, graphname=configs['graphname'])

conn.apiToken = conn.getToken(conn.createSecret())
```

3.3 CRUD operations code

Creating a new schema(changing schema) -

```
USE GRAPH AntiFraud


CREATE SCHEMA_CHANGE JOB addTimeTree FOR GRAPH AntiFraud {
   ADD VERTEX Year (PRIMARY_ID id INT, text STRING) WITH
primary_id_as_attribute="true";
   ADD VERTEX Month (PRIMARY_ID id INT, text STRING) WITH
primary_id_as_attribute="true";
   ADD VERTEX Day (PRIMARY_ID id INT, text STRING, dateValue DATETIME) WITH
primary_id_as_attribute="true";
   ADD UNDIRECTED EDGE DAY_TO_TRANSACTION (FROM Day, TO Transaction);
   ADD UNDIRECTED EDGE MONTH_TO_DAY (FROM Month, TO Day);
   ADD UNDIRECTED EDGE YEAR_TO_MONTH (FROM Year, TO Month);
}


RUN SCHEMA_CHANGE JOB addTimeTree
```

Inserting and updating -

```
CREATE QUERY TransactionTimes() FOR GRAPH AntiFraud {
  /* Inserts transaction time data into day, month, and year vertices in the graph */


  Seed = {Transaction.*};


  results = SELECT s FROM Seed:s
     ACCUM
        INSERT INTO Day (PRIMARY_ID, text, dateValue) VALUES
(str_to_int(to_string(year(epoch_to_datetime(s.ts))) + to_string(month(epoch_to_datetime(s.ts)))
```

```
+ to_string(day(epoch_to_datetime(s.ts)))), to_string(day(epoch_to_datetime(s.ts))),
epoch_to_datetime(s.ts)),

        INSERT INTO Month (PRIMARY_ID, text) VALUES
(str_to_int(to_string(year(epoch_to_datetime(s.ts))) +
to_string(month(epoch_to_datetime(s.ts)))), to_string(month(epoch_to_datetime(s.ts)))),

        INSERT INTO Year (PRIMARY_ID, text) VALUES (year(epoch_to_datetime(s.ts)),
to_string(year(epoch_to_datetime(s.ts)))),


        INSERT INTO YEAR_TO_MONTH (FROM, TO) VALUES
(year(epoch_to_datetime(s.ts)), str_to_int(to_string(year(epoch_to_datetime(s.ts))) +
to_string(month(epoch_to_datetime(s.ts))))),

        INSERT INTO MONTH_TO_DAY (FROM, TO) VALUES
(str_to_int(to_string(year(epoch_to_datetime(s.ts))) +
to_string(month(epoch_to_datetime(s.ts)))), str_to_int(to_string(year(epoch_to_datetime(s.ts)))
+ to_string(month(epoch_to_datetime(s.ts))) + to_string(day(epoch_to_datetime(s.ts))))),

        INSERT INTO DAY_TO_TRANSACTION (FROM, TO) VALUES
(str_to_int(to_string(year(epoch_to_datetime(s.ts))) + to_string(month(epoch_to_datetime(s.ts)))
+ to_string(day(epoch_to_datetime(s.ts)))), s);

}
```

Display -

```
CREATE QUERY GetRecentTransactionID(STRING startDate = "2017-01-15", STRING
endDate = "2017-04-15") FOR GRAPH AntiFraud{


  /* Grabs all transactions between given start and end date */


  ListAccum<VERTEX> @receiverSet, @senderSet;

  SumAccum<FLoat> @receiverTrust, @senderTrust;


  Seed = {Day.*};
```

```
  s1 = SELECT t FROM Seed:d -(DAY_TO_TRANSACTION:e) -:t
       WHERE d.dateValue < to_datetime(endDate) AND d.dateValue > to_datetime(startDate);


  s2 = SELECT t FROM s1:t
-((User_Recieve_Transaction_Rev|User_Transfer_Transaction_Rev):e) - User:u
      ACCUM
         CASE WHEN e.type == "User_Recieve_Transaction_Rev" THEN
            t.@receiverSet += u,
            t.@receiverTrust += u.trust_score
         ELSE
            t.@senderSet += u,
            t.@senderTrust += u.trust_score
         END
     ORDER BY t.ts DESC;


  PRINT s2;
}
```

3. Database Design

3.1 Software tools:

TigerGraph is a fast, scalable graph database that is designed for handling large-scale graph data and providing real-time analytics. It is a native parallel graph database, meaning that it was designed from the ground up to handle graph data and is optimized for storing and processing large-scale graphs efficiently.

Some key features of TigerGraph include:

Scalability: TigerGraph is designed to scale horizontally across multiple servers, allowing it to handle extremely large-scale graph data and real-time analytics.

High performance: TigerGraph is optimized for fast graph processing, making it well-suited for applications that require low-latency queries and real-time analytics.

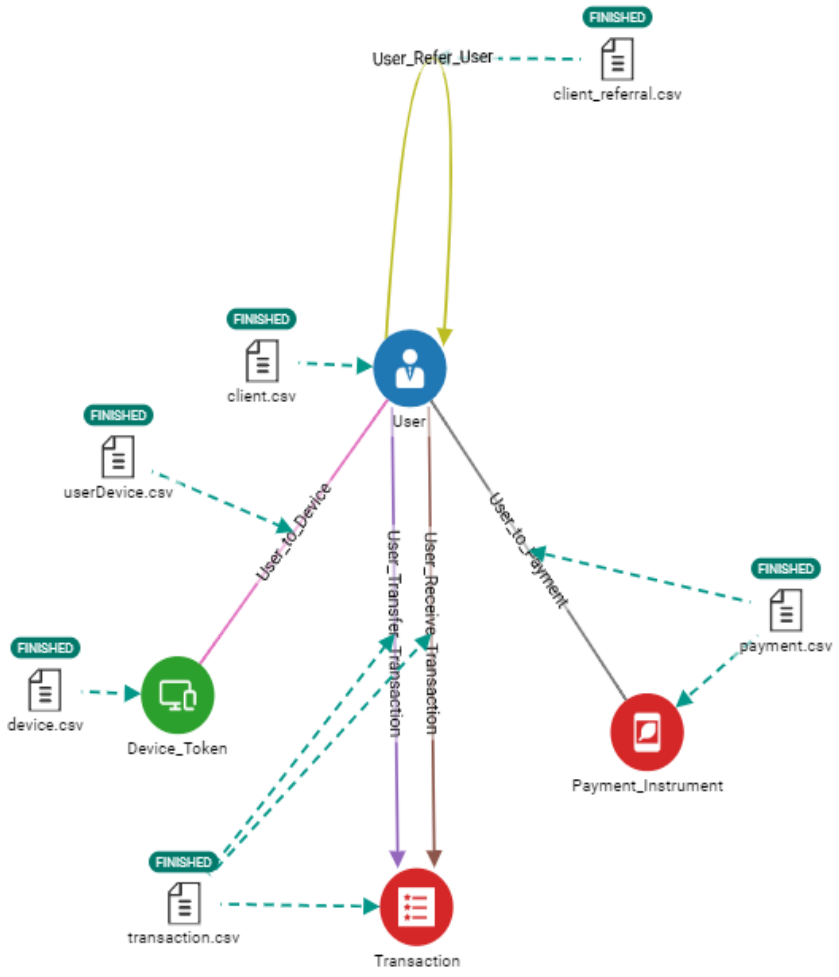
SQL-like query language: TigerGraph uses a SQL-like query language called GSQL, which is designed to be easy to learn and use for developers familiar with SQL.

Flexible data model: TigerGraph's data model is flexible and can support both graph and non-graph data, making it well-suited for a wide range of applications.

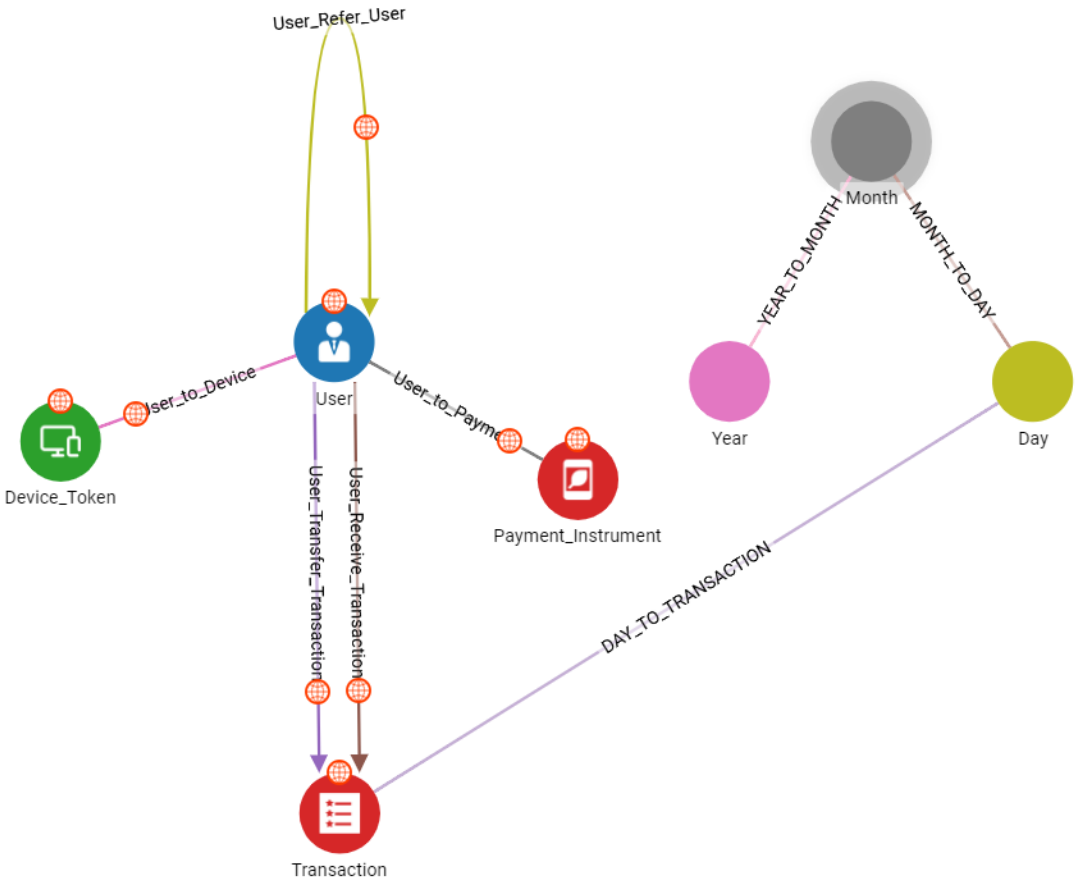
Overall, TigerGraph is a powerful tool for handling large-scale graph data and providing real-time analytics. It is used in a variety of industries, including finance, telecom, and social networking.

3.2 Database design architecture

Schema before adding time tree



After adding time tree



Vertex information:

Vertex: Transaction
(PRIMARY ID) id: STRING
amount: FLOAT
ts: UINT

Vertex: User
(PRIMARY ID) id: STRING
mobile: STRING
signupEpoch: UINT
trust_score: FLOAT

Vertex: Device_Token
(PRIMARY ID) id: STRING
carrier: STRING
device_name: STRING
is_banned: BOOL
is_emulator: BOOL
is_rooted: BOOL
model: STRING
os_name: STRING

Vertex: Payment_Instrument
(PRIMARY ID) id: STRING
card_bin: STRING
card_issuing_bank: STRING
card_issuing_country_iso2: STRING
token_handle: STRING
token_type: STRING
trust_score: FLOAT

Vertex: Year
(PRIMARY ID) id: INT
text: STRING

Vertex: Month
(PRIMARY ID) id: INT
text: STRING

Vertex: Day
(PRIMARY ID) id: INT
dateValue: DATETIME
text: STRING

Edge: User_Transfer_Transaction
Source: User
Target: Transaction
reverse edge: User_Transfer_Transaction_Rev

Edge: User_to_Device
Source: User
Target: Device_Token

Edge: User_to_Payment
Source: User
Target: Payment_Instrument

Edge: User_Refer_User
Source: User
Target: User
reverse edge: User_Referred_By_User

Edge: User_Receive_Transaction
Source: User
Target: Transaction
reverse edge: User_Receive_Transaction_Rev

Edge: DAY_TO_TRANSACTION
Source: Day
Target: Transaction

Edge: MONTH_TO_DAY
Source: Month
Target: Day

Edge: YEAR_TO_MONTH
Source: Year
Target: Month

4. Conclusion

In conclusion, a fraud detection dashboard using TigerGraph can be a powerful and effective tool for identifying and preventing fraudulent activity. By leveraging the power of graph analytics, it is possible to analyze complex interconnected data and identify patterns and anomalies that may indicate fraud.

The use of a dashboard allows for real-time monitoring and analysis of data, enabling timely detection and response to potential instances of fraud. Additionally, the integration of machine learning techniques can improve the accuracy and efficiency of the fraud detection process.

Overall, a fraud detection dashboard using TigerGraph can be a valuable asset for organizations looking to protect against fraudulent activity and improve the security and integrity of their systems.