**Functional Specification Document**

**Visual Investigator**

Network Analysis

Version 1.0



**Network Linking**

**Objective –**

This document provides detailed information on the Visual representation of graph in network analysis focusing on customer’s network of payments(transactions)

We can identify influential suspicious customer with the help of degree centrality values. A high degree of centrality value indicates the node has a high influence on the network, and the associated customer is highly risk-oriented.

We will create a graph based on following relationships and node weights

**Graph Relationships**

1. **Direct SAR relationship –**

This relationship will help us identify how many distinct SAR parties does a bank customer deals with and plot the edges between them

1. **Indirect relationship with SAR customers -**

This relationship will be a level n (currently 2) analysis to identify customer who are indirectly linked to SAR customers

**Node Weight:**

**Degree centrality measure -**

This features uses number of transactions as proxy for importance to find our influential customers

Degree centrality (node\_i) = number of edges (node\_i)/N-1

**Metrics:**

1. **Direct SAR relationship –**

This feature will help us identify how many distinct SAR parties does a bank customer deals with

**Calculation –**

We will be using name matching algorithm to identify which non SAR bank customers are dealing with SAR parties

Step 1 – Extract transactions for SAR customers

Step 2 – Using name matching algorithm identify which counterparties are also SAR vs SAR

Step 3 – Retain Non SAR counterparties which are bank customers to get SAR vs Non SAR

Variables –

* # Transactions with SAR parties
* # Distinct SAR parties they are linked with

1. **Indirect relationship with SAR customers -**

This feature will be a level 2 analysis to identify customers who are indirectly linked to SAR customers

**Calculation –**

From previous step, we had a list of non SAR counterparties who were dealing with SAR customers. Now we will extract network of these NON SAR counterparties to identify bank customers who were indirectly linked with SAR parties

Variables –

* # Transactions
* # Indirect linked counterparties who were dealing with SAR parties
* sum of (distinct SAR parties corresponding to all counterparties linked with the node)

**Data model:**

**Data Mart creation from Data source:**

|  |  |
| --- | --- |
| DATA\_MART | Description |
| date | transaction date time |
| transaction\_cdi\_code | transaction code, debit or credit |
| primary\_medium\_desc | Check, wire etc. |
| currency\_amount | amount of the transaction |
| flag | Suspicious transaction flag |
| segment | Individual or org |
| account\_number | account number of the customer |
| counterparty | counter party id |

|  |
| --- |
| Transaction Master |
| transaction key |
| party key |
| account key |
| date key |
| currency amount |
| country key |
| x\_transaction\_ope\_country\_key |
| cmov\_party\_name |
| cprt\_party\_name |
| flag |
| Transaction cdi code |
| Primary medium |
| secondary medium |
| tertioary medium |

**Data model of Graph Storage**:

|  |  |
| --- | --- |
| **columns** | **Description** |
| account\_number | account number of the customer |
| graph\_gexf | graph representation in xml format to reader in UI using Sigma library |
| direct\_sar | JSON hold the direct sar transaction, sum and count per month |
| in\_direct\_sar | JSON hold the indirect sar transaction, sum and count per month |
| created\_dttm | create time |
| updated\_dttm | update time |

**Metrics Calculations:**

**Direct SAR:**

**Pyspark Queries:**

1. create table sar\_lkp as select distinct account\_number from data\_mart where flag =1 ;
2. create table sartrxn as select \* from data\_mart where account\_number in (select distinct account\_number from lkp);
3. create table sarvsar as select \* from sartrxn where counterparty in (select distinct account\_number from lkp);
4. create table sarvnonsar as select \* from sartrxn where counterparty not in (select distinct account\_number from lkp);

#direct SAR count

1. create table direct\_sar\_count as select account\_number , count(distinct counterparty) as directsar, count(\*) as trx\_L1 from sarvsar group by account\_number;

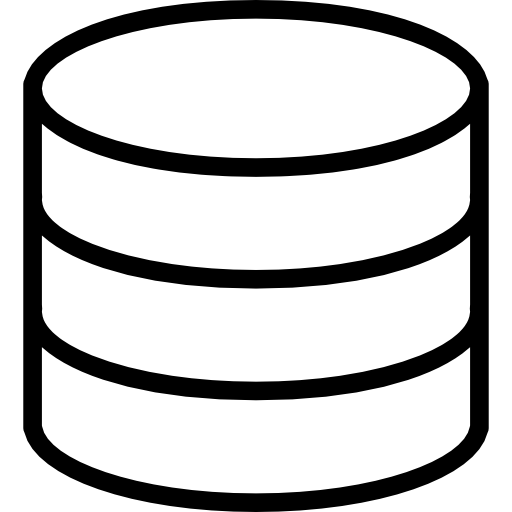
**Indirect SAR:**

**Pyspark Queries:**

1. create table nonsarlkp as select counterparty from data\_mart;
2. create table level2 as select \* from data\_mart where counterparty in (select distinct counterparty from nonsarlkp) ;
3. create table level2v2 as select \* from level2 where account\_number not in (select distinct account\_number from lkp);
4. create table feature2 as select account\_number, count(distinct counterparty) as cntrparty, count(\*) as trxn\_l2 from level2v2 group by account\_number
5. create table abc as select counterparty, count(distinct account\_number) as l2sarcount from sarvnonsar group by counterparty
6. create table level2v3 as select a.\*,b.l2sarcount from level2v2 as a left join abc as b on a.counterparty = b.counterparty
7. create table feature3 as select distinct account\_number ,l2sarcount from level2v3

create table feature4 as select account\_number , sum(l2sarcount) from feature3 group by account\_number

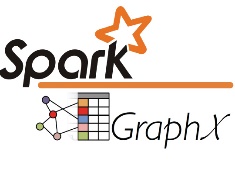
**Graph Creation:**



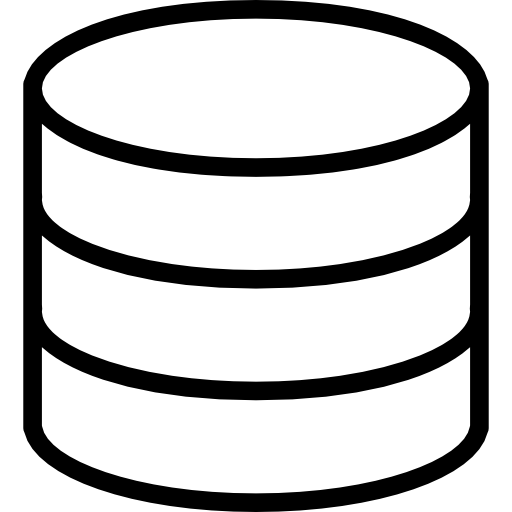
**Presentation DB**



**NetworkX**



**Spark GraphX**



**Data mart**

* Read graph from CSV file
* Construct the graph
* Export graph in GEFX format
* Save the graph data to DB
* Read data from data mart
* create graph dataframe from the relationships direct SAR, and indirect SAR
* calculate degree of centrality
* find out all the nodes with shortest distance n (currently 2) for particular customer
* Save graph in CSV format

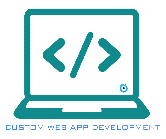
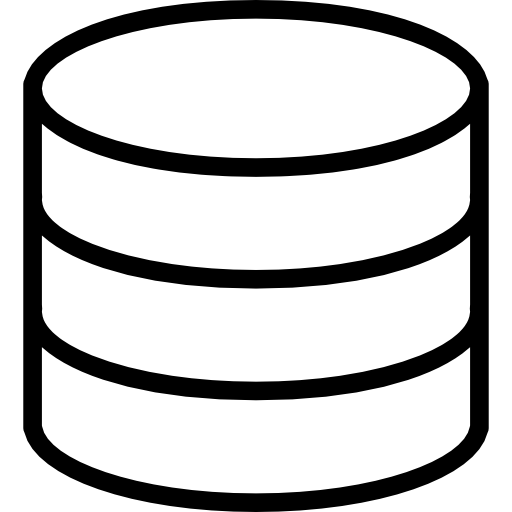
**Graph Render on UI:**

To render the graph on UI we will use Sigma js library. Sigma library convert XML format graph to visual graph and render it on UI



**Backend Service**

**Presentation DB**



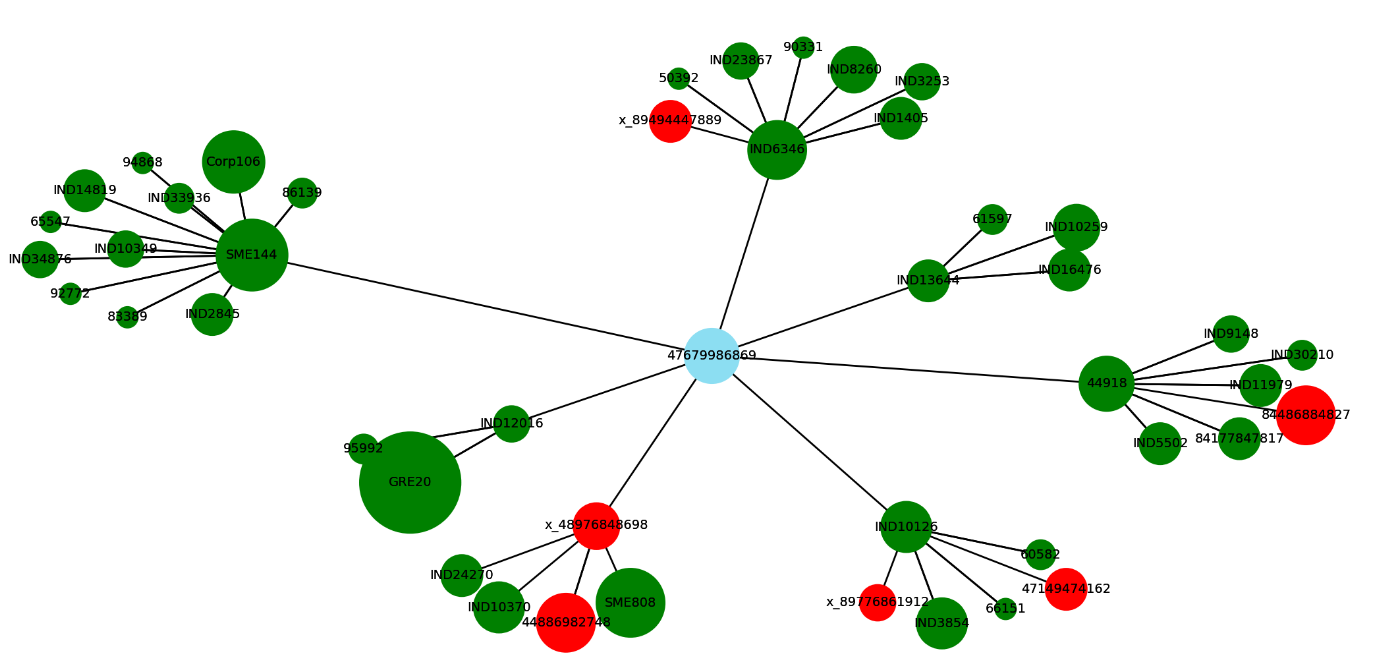
**Web APP**

Read graph data for particular customer

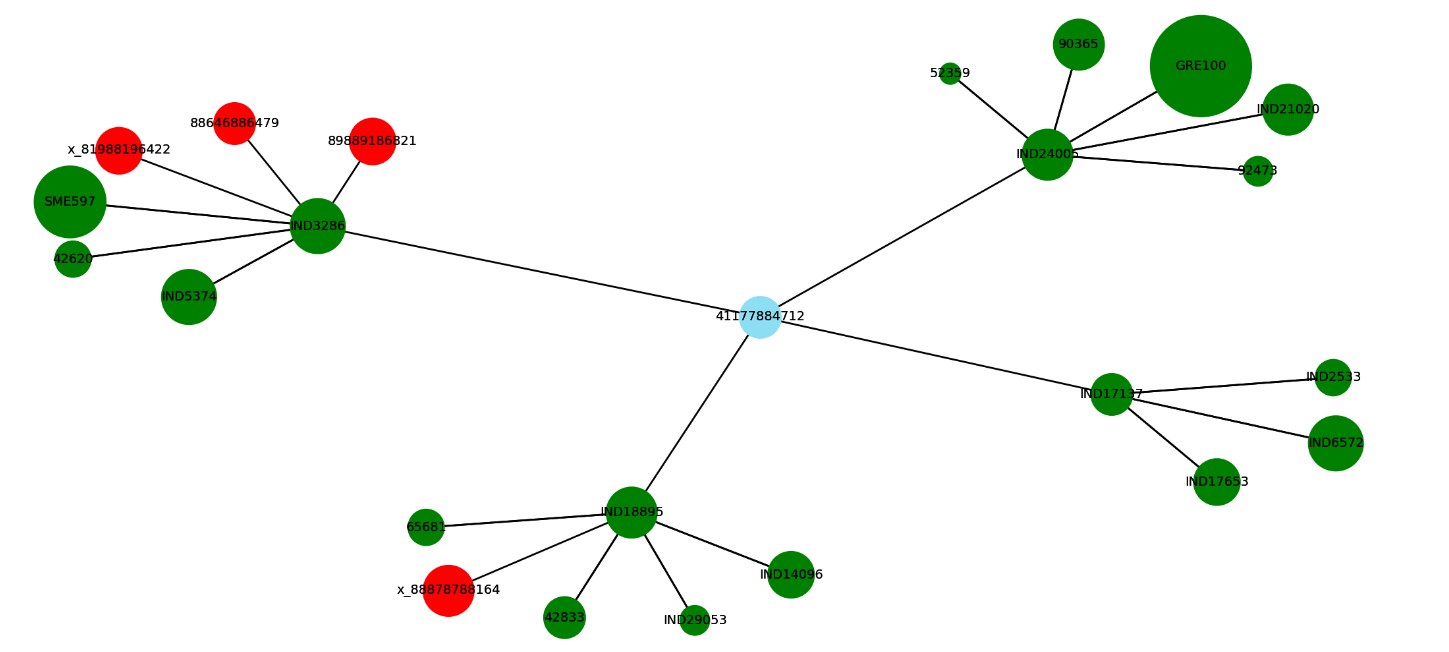
Fetch Graph from backend REST API

**Rendered Graph with centrality measure between customer and its neighbors**

**Example 1:**



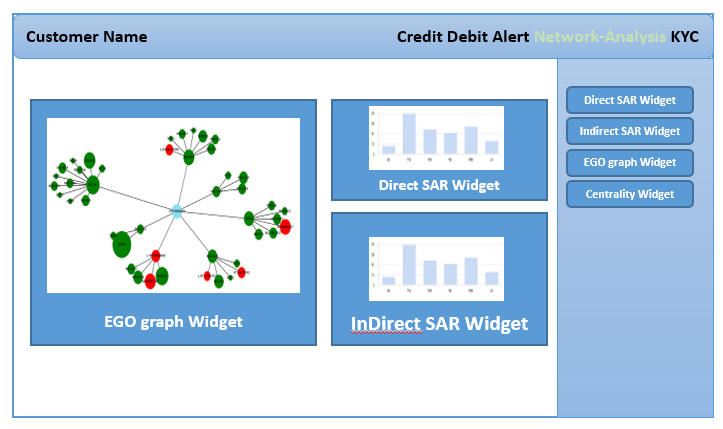
**Example 2:**

****

**Graph Specifics:**

1. **Red color:**  SAR counterparty
2. **Green color:** Non SAR counterparty
3. **Blue:** customer
4. **Radius of circle:** centrality value - A high degree of centrality value indicates the node has a high influence on the network, and the associated customer is highly risk-oriented.

**Network Linking Page:**

****