GTW 2014 – MIMOS Accelerated Library

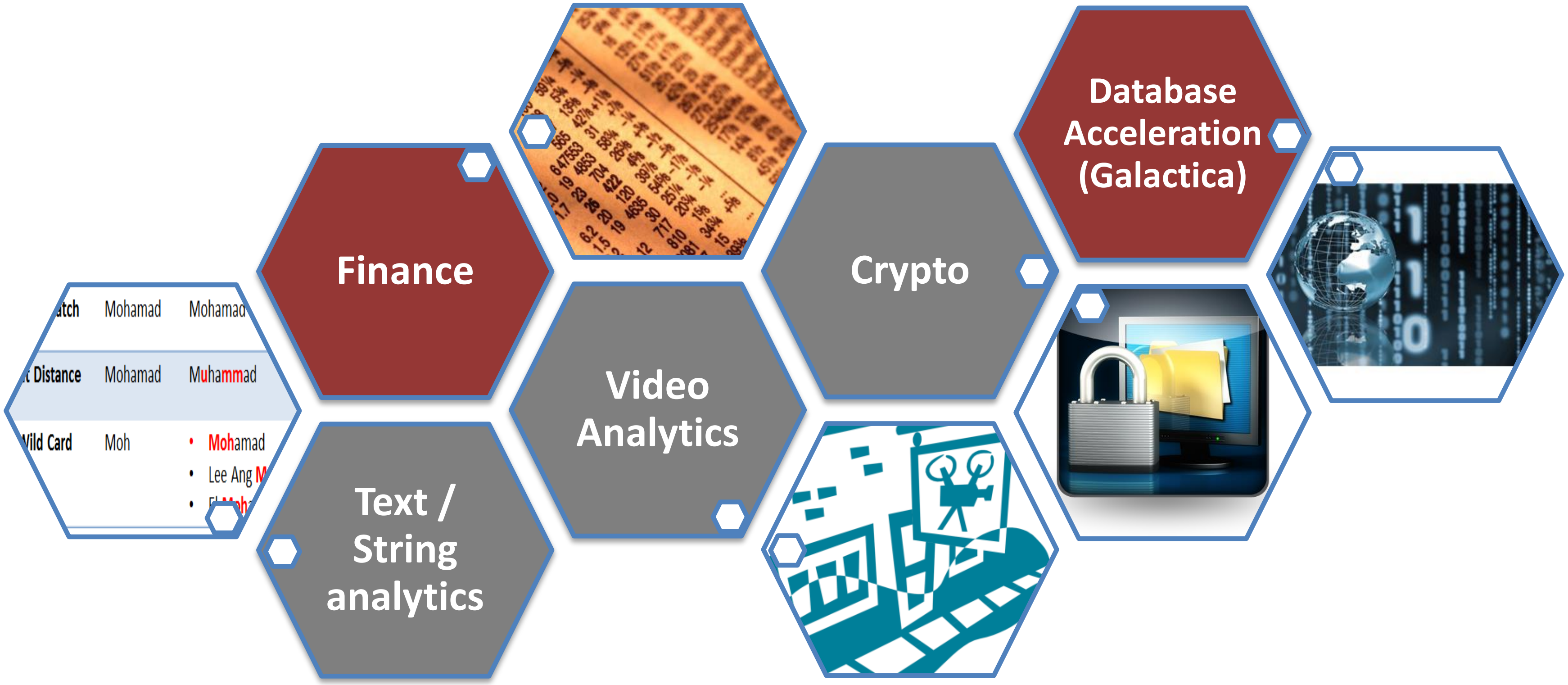
Yaszrina Mohamad Yassin

Accelerative Technology Lab
MIMOS Berhad, Malaysia

Innovation for Life™

MIMOS Accelerated Library

acclib



Finance

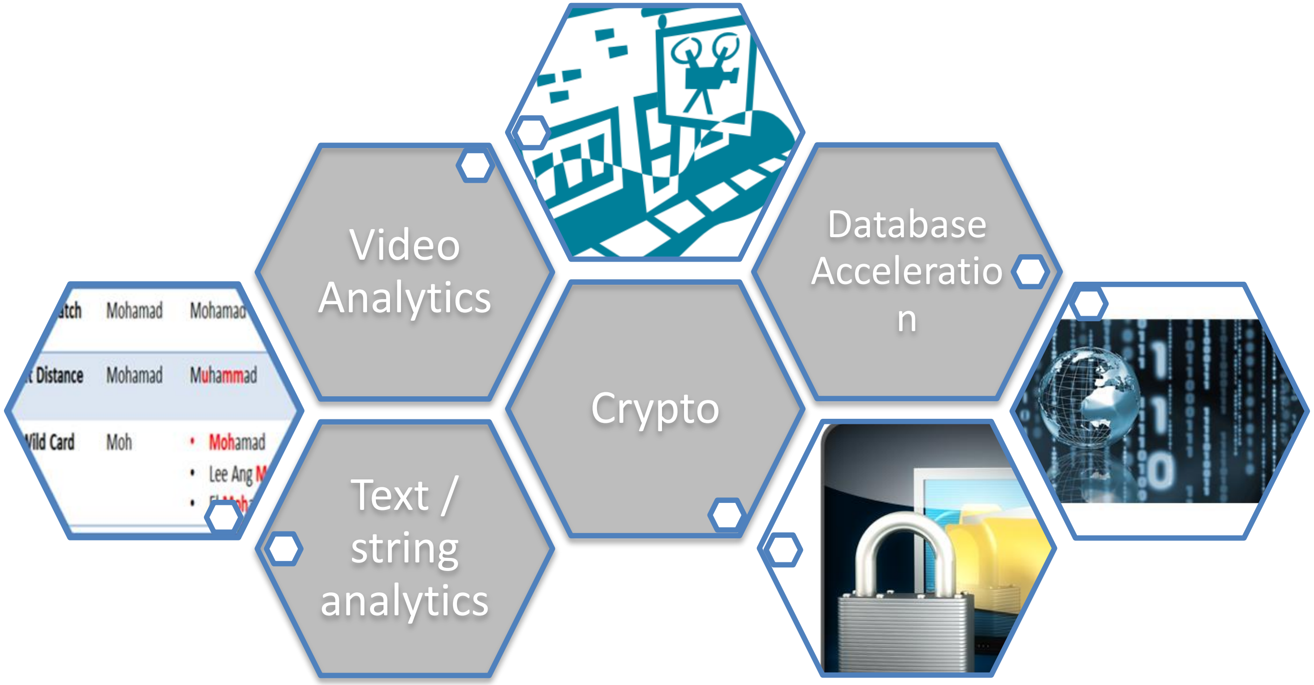
Crypto

Database Acceleration (Galactica)

Video Analytics

Text / String analytics

Finance



Video Analytics

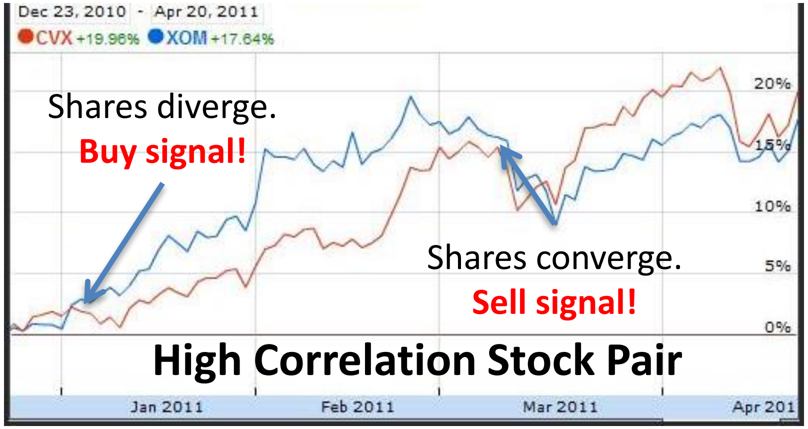
Database Acceleratio n

Crypto

Text / string analytics

Financial Analysis With MiAccLib

Stock Pair Trading



Shares diverge.
Buy signal!

Shares converge.
Sell signal!

High Correlation Stock Pair

Pair Correlation = $$\frac{\sum(A_i-\overline{A})(B_i-\overline{B})}{\sqrt{\sum(A_i-\overline{A})^2\sum(B_i-\overline{B})^2}}$$

- Provides the user a historical view on the correlation between different pairs of shares.
- Shares with high correlation historically move in the same direction.

Multiple Portfolios with Mutliple Stocks

Historical Data for Selected Stocks (Daily)

Trading Parameters (e.g):

- Ratio versus Spread
- Moving Average
- Standard DeviatioΩ
- Maximum day in trade
- Stop Loss
- Critical Entry and Exit
- Start and End Date

Accelerated Calculation:

Price Ratio (**PR**)/Spread (**PS**)
PR/PS Moving Average
PR/PS Moving standard Deviation
Normal Deviation (**ND**)
Average Spread **Correlation**
Spread **Co-integration**

Buy/Sell/Hold Parallel Stocks in Different/Multiple Portfolios

Compute Intense & Parallelizable Algorithms!

N2N CONNECT

Complexity of Computation for Pairwise Correlation

Pair correlation = $\frac{\sum(A_i - \overline{A})(B_i - \overline{B})}{\sqrt{\sum(A_i - \overline{A})^2 \sum(B_i - \overline{B})^2}}$

$$= \frac{(A_1 - \overline{A})*(B_1 - \overline{B})}{\sqrt{\sum^{2500}(A_1 - \overline{A})^2 * \sum^{2500}(B_1 - \overline{B})^2}} + \frac{(A_2 - \overline{A})*(B_2 - \overline{B})}{\sqrt{\sum^{2500}(A_2 - \overline{A})^2 * \sum^{2500}(B_2 - \overline{B})^2}}$$

$$+ \dots\dots\dots\dots + \frac{(A_{2500} - \overline{A})*(B_{2500} - \overline{B})}{\sqrt{\sum^{2500}(A_{2500} - \overline{A})^2 \sum^{2500}(B_{2500} - \overline{B})^2}}$$

Note:
2500 = 250 trading days * 10 years
319,600 = 800 stocks pair combination = $\frac{800*(800-1)}{2}$ →

For 1 pair:
Complexity = 2500^2

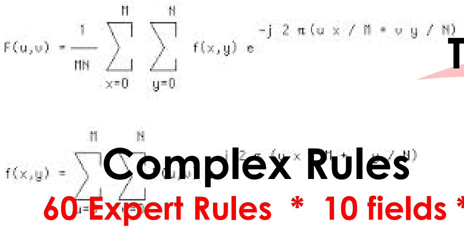
For 319,600 pairs:
Complexity = 319,600 * 2500^2

Data-points = 1.99 x 10^{12}

1 GPU card (2496 Cores) = **99** Minutes

1 PC (4 Cores)= 21,307 Minutes (~14.8 days)

Financial Compliance: Anti Money Laundering



Large Daily Transaction Pool
(Millions Daily)

Complex Rules
60 Expert Rules * 10 fields *
20 million customer

Large Financial Terrorist
(~ 6 Million Individuals)

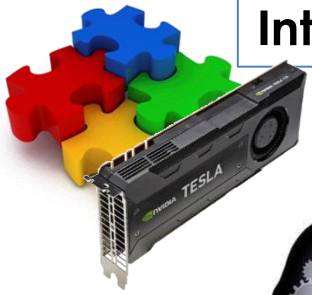
36 Trillion Combinations

High False Positive Rate
False positive = 99.96%

Improved Analytical Integrity

Accelerated &
Parallelized Algorithms

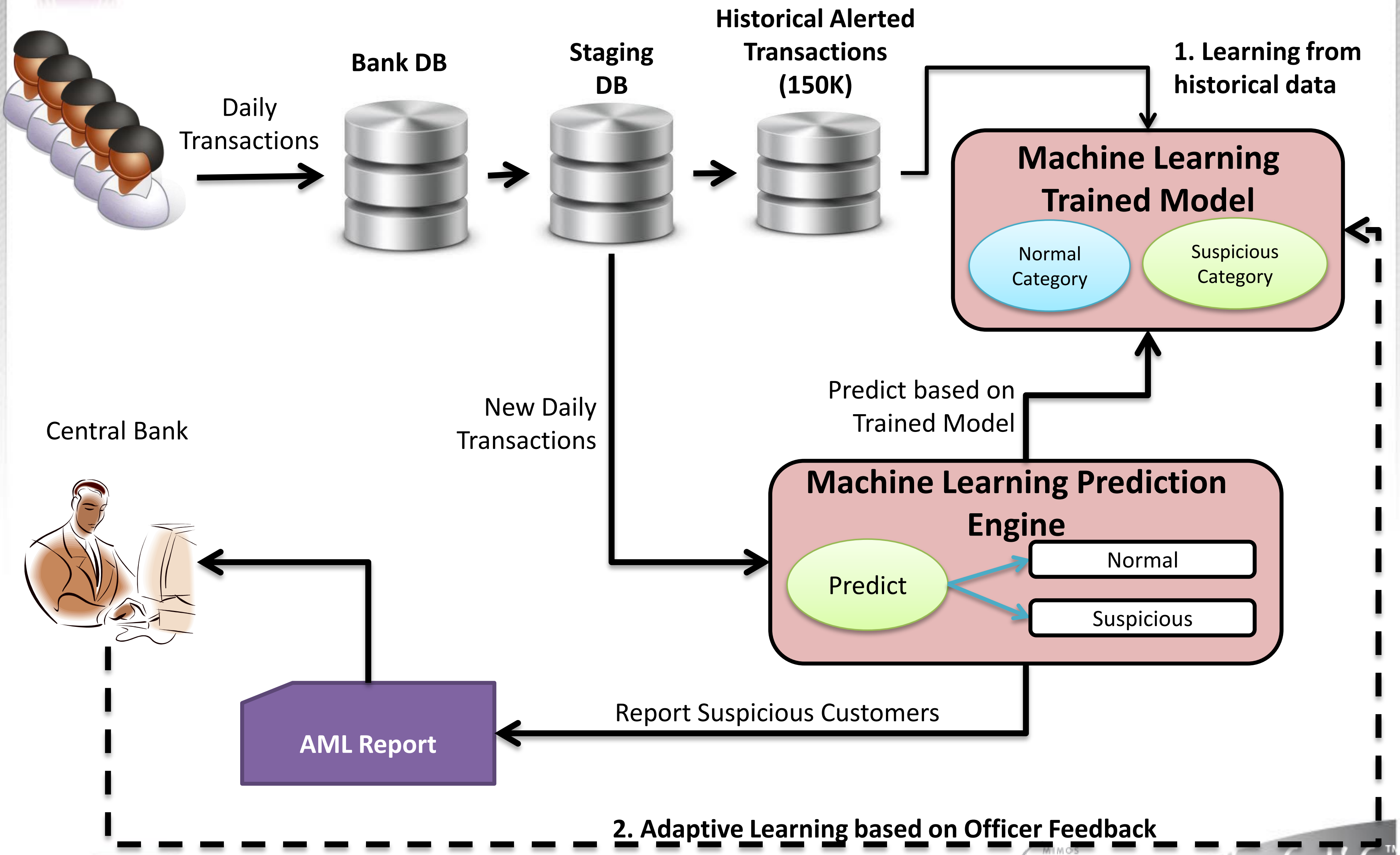
High Speed Search
(Seconds)



Machine Learning Algorithms
(Minutes)

Adaptive Domain Specific Algorithms
(Reduction to 20% & Hours)

Accelerated Machine Learning Solution



Daily Transactions

Bank DB

Staging DB

Historical Alerted Transactions (150K)

1. Learning from historical data

Machine Learning Trained Model

Normal Category

Suspicious Category

Predict based on Trained Model

New Daily Transactions

Central Bank

Machine Learning Prediction Engine

Predict

Normal

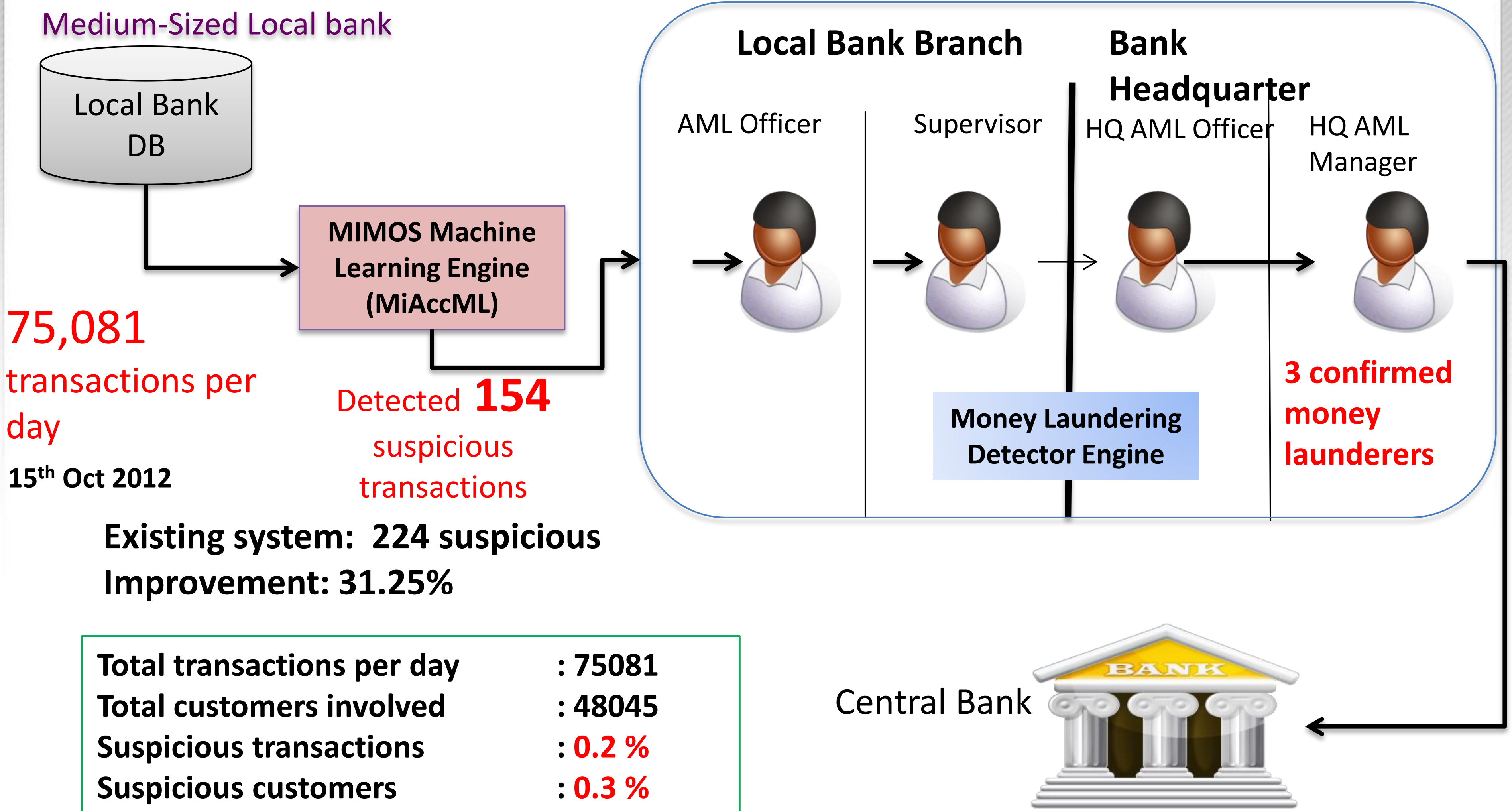
Suspicious

Report Suspicious Customers

AML Report

2. Adaptive Learning based on Officer Feedback

Accuracy Test using historical data



75,081 transactions per day

15th Oct 2012

Detected **154** suspicious transactions

3 confirmed money launderers

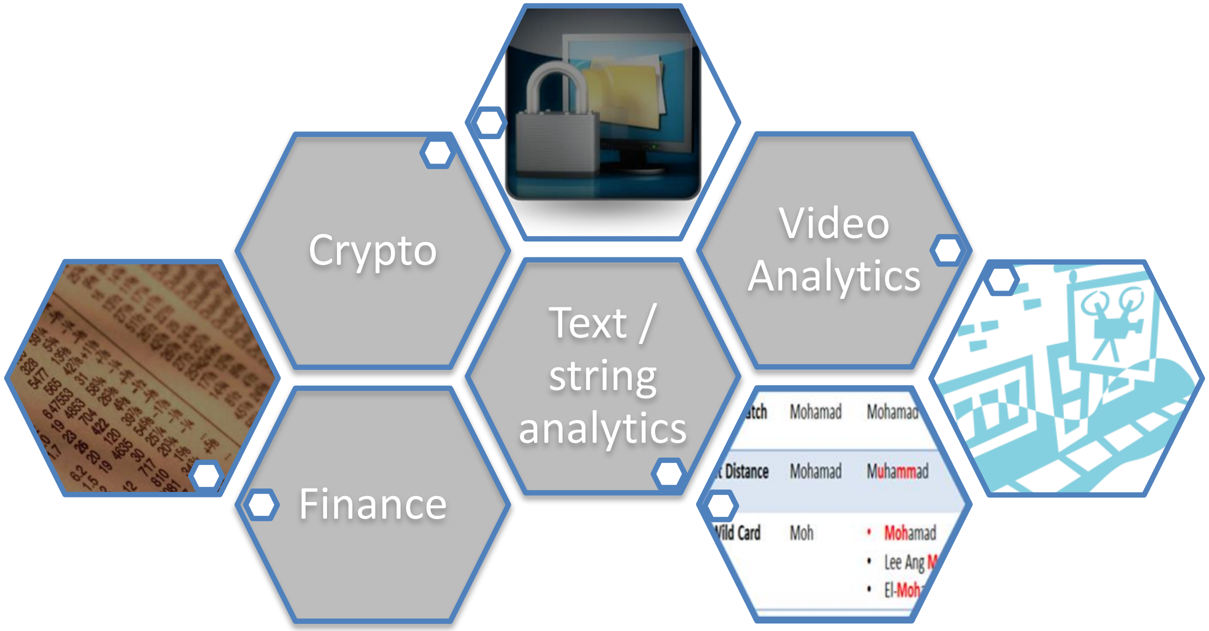
Existing system: 224 suspicious

Improvement: 31.25%

Total transactions per day	**: 75081**
Total customers involved	**: 48045**
Suspicious transactions	**: 0.2 %**
Suspicious customers	**: 0.3 %**
Actual money launderers	**: 0.004 %**

Database Acceleration

(Galactica)



Crypto

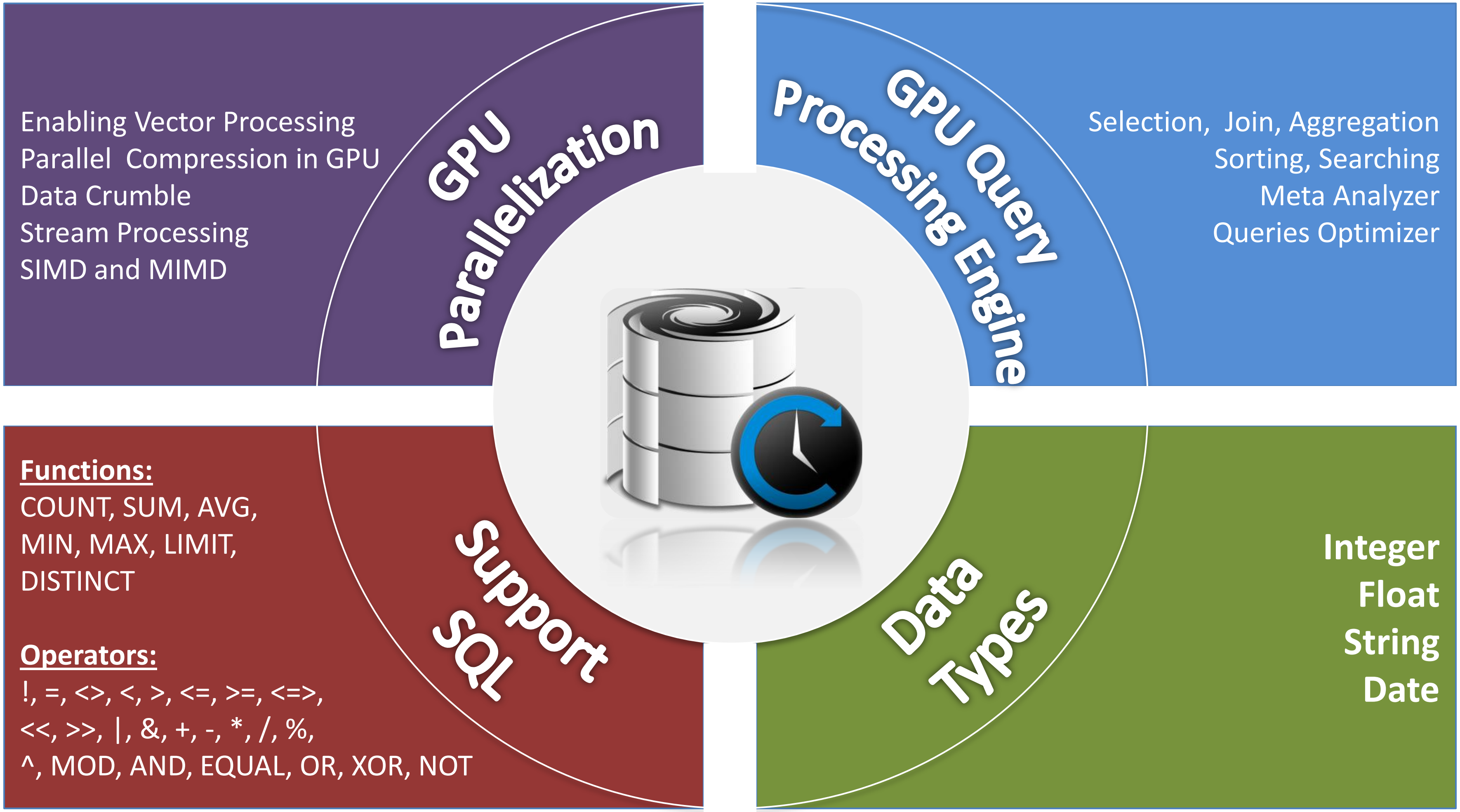
Video Analytics

Text / string analytics

Finance

GPU-based Database Acceleration (Galactica)

acclib



GPU Parallelization

Enabling Vector Processing
Parallel Compression in GPU
Data Crumble
Stream Processing
SIMD and MIMD

GPU Query Processing Engine

Selection, Join, Aggregation
Sorting, Searching
Meta Analyzer
Queries Optimizer

Support SQL

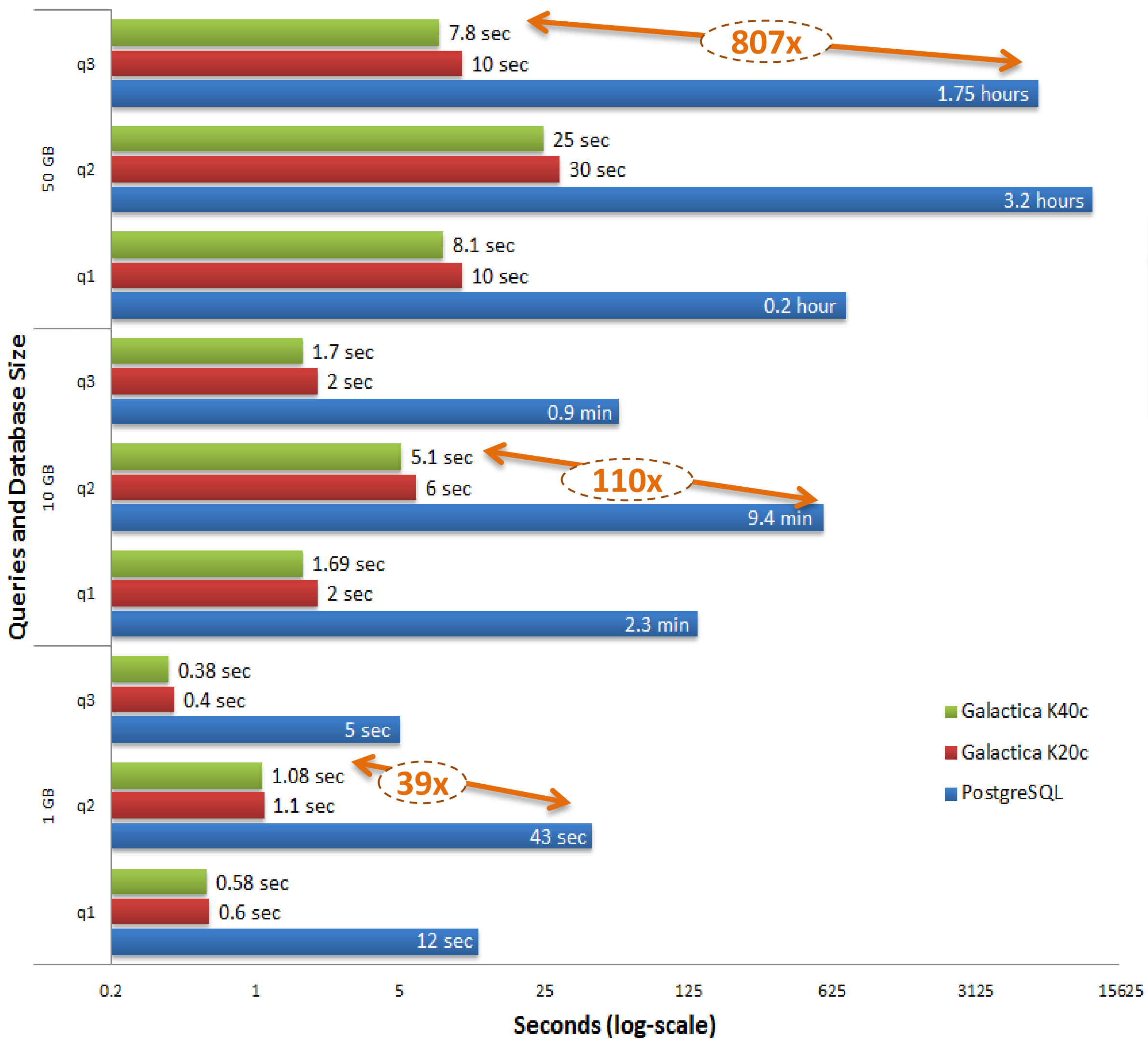
Functions:
COUNT, SUM, AVG,
MIN, MAX, LIMIT,
DISTINCT

Operators:
!, =, <>, <, >, <=, >=, <=>,
<<, >>, |, &, +, -, *, /, %,
^, MOD, AND, EQUAL, OR, XOR, NOT

Data Types

Integer
Float
String
Date

Result of *Galactica* vs *PostgreSQL*



***Data is taken from TPC-H benchmarking**

SQL Query 1

Compute amount of business that was billed, shipped and returned

SQL Query 2

Compute the total revenue, quantity and orders from the “Building” customer

SQL Query 3

Compute the revenue, total and average amount of quantity along with the average price from transactions

** Setup Config:

CPU	Intel(R) Xeon(R) CPU X5680 @ 3.33GHz (2 processors)
RAM	22 GB
GPU	NVIDIA Tesla K40c / K20c
CUDA	5.5
Storage	WD HDD 1TB
Database	PostgreSQL 9.3
OS	Windows 7 (64 bits)

Result of *Galactica* vs *Hadoop*

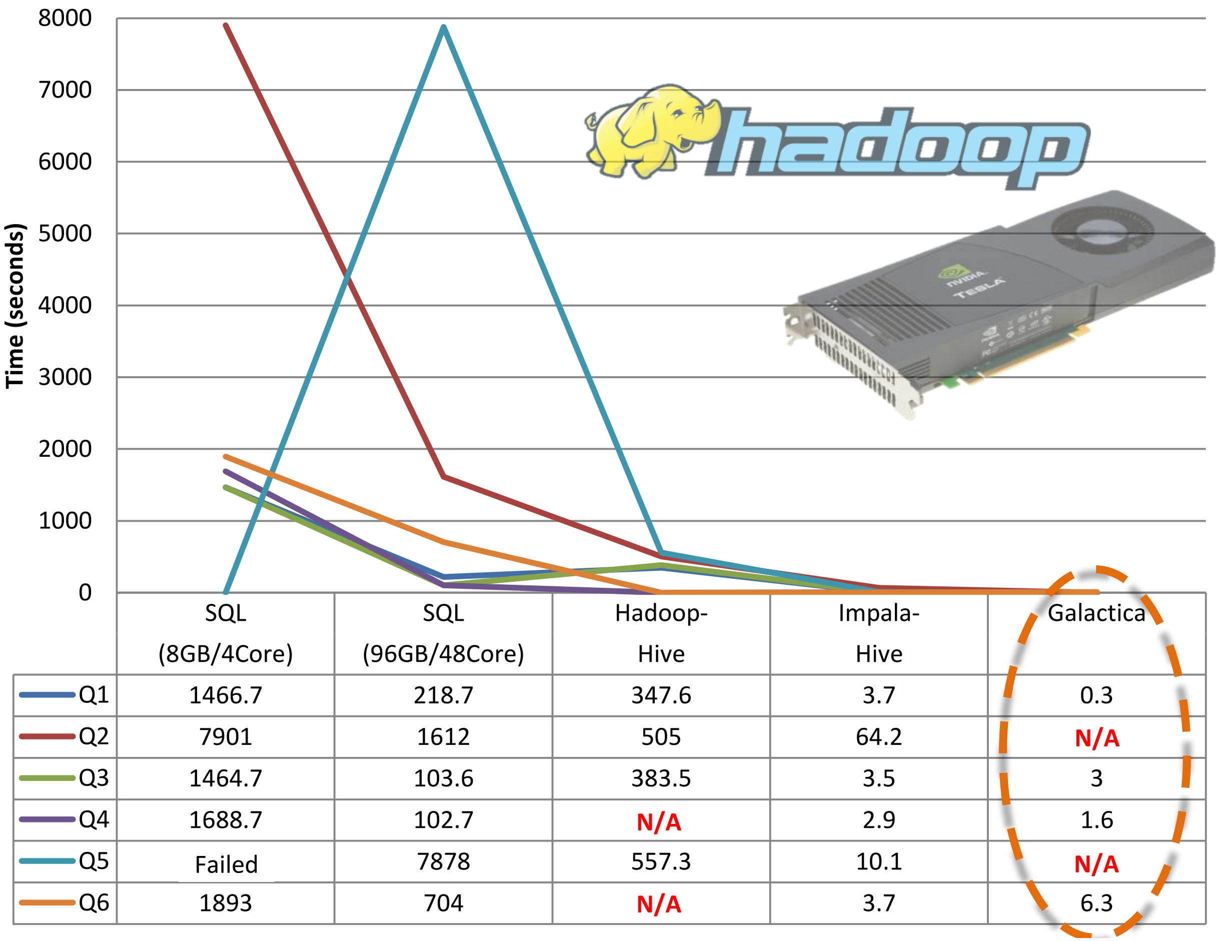
32GB Data

- Best performing on sum
- Cost saving HPC
- Failed queries operation because Galactica does not support the feature yet

Hadoop
7 Virtual Machines with one master node (8 cores)
6 worker nodes (4 cores each) running on a few of
HP DL380p G8 servers installed with Apache Hadoop,
Cloudera's Hadoop and Impala.
Postgres on another same model of HP server with 8GB RAM with 4 cores and another high end HP machine with 96 GB RAM and 48 cores.

GPU
DELL Precision T5500 workstation
NVIDIA Tesla K20c
on Intel Xeon E5630@2.53GHz processor
12GB RAM
1 TB Sata Hard drive (7200rpm).
Windows Server 2008 R2 Enterprise SP1 64-bit

Comparison of Queries Processing Time



	SQL (8GB/4Core)	SQL (96GB/48Core)	Hadoop-Hive	Impala-Hive	Galactica
Q1	1466.7	218.7	347.6	3.7	0.3
Q2	7901	1612	505	64.2	N/A
Q3	1464.7	103.6	383.5	3.5	3
Q4	1688.7	102.7	N/A	2.9	1.6
Q5	Failed	7878	557.3	10.1	N/A
Q6	1893	704	N/A	3.7	6.3