Why Database Matters In Card Fraud Detection

Big Data LDN : 13-14 Nov, 2019

Dheeraj Mudgil
Enterprise Architect [Fraud]
Introduction

Barclays

- Is 329 year old Bank (1690); it is older than the UK Itself
- Moves, lends, invests and protects money for customers and clients worldwide
- Offers Banking, Cards, Corporate Banking, Insurance, Markets, Mortgages, Payments Acceptance, Ecommerce, Wealth & Investment Management …
- Financial Innovation

Dheeraj Mudgil

- Enterprise / Solutions Architect with software engineering background
- 20+ years in the Industry mainly in Financial sector
- Working with Barclays in Chief Technology Office
- Architecture owner for the Fraud Platform
Question

How much data do we generate today?
Trends - General

- Devices
- New Channels
- Automation
- Desire to release products fast

- Social Media & Info Sharing
- Innovative Products
- Software Complexity
- Vulnerabilities
Trends – Customer

- Customer Patience
- Instant / on-demand

- Loyalty towards Product
  ➢ Enabler – Options/Choices

- CX Expectations
- Expects protection from Loss

- Complex customer needs
- Complex customer behaviour
Trends – Fraud & Fraudster

- Barrier to HPC
  - Enabler - Cloud

- Barriers to collect Info
  - Enabler – Social Media Platforms

- Sophistication of Fraud Techniques
- Fraud supply chain – more organised

- Subtlety of Fraud
- Complex fraudster behaviour
Typical Card Payment Journey (Card Not Present)

* Card Present journey is mostly similar with slight variation
Use-Case Analysis

- Consumer - Swift Checkout & Fraud Protection
- Merchant - Fast Approval & Zero declines
- Issuer – Encourage Card usage & Protect Fraud losses

Key Characteristics:
- Latency Sensitive
- Complex Processing Rules
- Large Datasets
- Consistency

Challenge

Complexity
Latency
Operations

False Positive

False Negative

TCO

Build Cost

Operational Costs

Time to Market
How to solve?

Very Simple!

Set up a Database – To store customer history & patterns

Build an Application – Containing logic & expose via API
Which Database?

How to Select:

- **Known** - Let's use what I am familiar with
- **Generic** - Let's pick a general purpose database
- **Safety** - Look at Industry reports (Gartner etc.)
- **Enterprise Constraints** – No more than one/two DB allowed
- **Retrofit** - We already have something let's use it
- **Luck** – Toss a coin / dice!

Credit for Infographic – [https://mattturck.com/bidata2018/](https://mattturck.com/bidata2018/)
Simple Solution

Traditional DBMS

Client

Application

Database

Physical Disk

Shared Nothing DBMS

Client

Application

Database System

Application Logic
Common Patterns [Simplistic View]

Basic Caching

DB Clustering

Cache Aside

Cache Through*

* Other variations also exist
Common Patterns [Simplistic View]

1. In-Memory DB aside
2. In-Memory DB through
3. In-Memory DB with Persistence
Common Patterns [Simplistic View]

* Deploying application in database is roughly similar
# Complexity of Financial Fraud

## Financial Crime

| 1st Party Fraud | 3rd Party Fraud | 2nd Party Fraud | Mules | Scams | ...

## Key Concepts

- Card Fraud
- Account Takeovers
- Cheque Fraud
- Payment Fraud
- Invoice Fraud
- Bank Account Fraud
- Cyber Fraud
- Cash Point Fraud
- Identity Fraud
- Plastic Fraud
- Mandate Fraud
- ...

## Fraud Types

### Extending use-case: Implications?

- Complexity & Criticality
- Additional data needs
- Scale, throughput and other NFRs…
Regulatory requirements

- Strong Customer Authentication (SCA)
- Transaction Risk Assessment (TRA)
- Confirmation of Payee
- Payment Validations
- Contingency Reimbursement Model (CRM)
- ...

- Impact on the use-case?
- Impact on Architecture?
- Impact on the Data handling?
- Impact on DB technology choice?
Summary

- **Database Matters in Fraud use-case**
  - It’s extremely important component in the Fraud detection process

- **Each DB Technology**
  - Has its pros and cons
  - Scale differently to others
  - Each technology may offer unique feature as its core strength but may have weakness in other areas

- **No Silver Bullet in selecting a DB**
  - 80% use-cases may fall into general purpose where selection may be relatively easy
  - Remaining 10 - 20% use-case are more complex and difficult

- **Functionally similar use-cases**
  - May only be functionally alike but
  - Additional use-cases may increase complexity and scale exponentially
  - NFRs may completely warrant changing the solution
  - Operational Environment can change / dictate solution

- **What may work**
  - Deep understanding of use-case
  - Understanding Data itself
  - Meticulously matching detailed use-case requirements with product features
Barclays Fraud Use-Case

- **Performance** – Quick access to large data sets
  - One hop to the data from client
  - Fast disk access [Secret sauce - Patented Technology]
  - No cache misses
  - Supports IMDB
  - Parallel Fetch

- **Predictability** - Helps utilizing set ‘Time Budget’ effectively
  - Known path of data retrieval
  - No cache miss
  - Reduced Jitter [written in C]

- **Simple Architecture** – Help TCO & Extensibility
  - No caching layer to setup and manage
  - Reduced RAM footprint & Cluster size

- **Supported** – The use-case’s
  - Scaling needs
  - Strong consistency & durability
  - Standard security features

- **Non Relational** - Key Value pair only with Maps & List feature
- Does not support SQL, has its own API like most NO-SQL DBMS
- Has bit of a learning curve
- Not a general purpose DB, may not be the best investment for all use-cases
• Choosing right DB technology is more difficult (for extreme 10-20% use-cases)

• Aspects frequently misunderstood or often taken lightly
  ➢ Scale & Throughput requirements
  ➢ Sensitivity towards Latency
  ➢ Jitter in DB operations and its Implication on overall application behaviour
  ➢ Matching CAP theorem aspects with use-case
  ➢ Data characteristics and usage [Hot /Cold, All Hot, volume, variety etc.]
  ➢ TCO calculation considering all factors and environments

• Key aspects to consider to narrow down choices
  ➢ Pedigree – One specific problem the product was originally designed to solve i.e. product’s core competence and matching with the use-case
  ➢ Jepsen Report – If consistency is important; wonderful resource for DB community
  ➢ References - Drawing comparable from industry use-cases
• Operational aspects
  - Knowing your operational environment and constraints imposed by overall landscape, wider architecture, practices, pipelines etc.
  - Security & Monitoring
  - Patching strategy of DB and its alignment with the UNIX node patching
  - Vendor Support & SLAs
  - Other standard aspect such as Resilience, Cloud/On-Premise, DR, Active – Active, Rolling Upgrades, Backup, Accessibility to data/API etc.

• Investment in following pays off
  - PoC/s for right DB selection
  - Training the Build & Operational teams on product
  - Right Data-Modelling for the application/s

• Things change
  - Better products to support your use-case may be available tomorrow, so keep eyes open and be aware of Vendor-Lock-In