HIVE + AMAZON EMR + S3 = ELASTIC BIG DATA SQL ANALYTICS PROCESSING IN THE CLOUD
A REAL WORLD CASE STUDY

Jaipaul Agonus
FINRA

Strata Hadoop World
New York, Sep 2015
FINRA - WHAT DO WE DO?

**Collect and Create**
- Up to 75 billion events per day
- 13 National Exchanges, 5 Reporting Facilities
- Reconstruct the market from trillions of events spanning years

**Detect & Investigate**
- Identify market manipulations, insider trading, fraud and compliance violations

**Enforce & Discipline**
- Ensure rule compliance
- Fine and bar broker dealers
- Refer matters to the SEC and other authorities
FINRA’S SURVEILLANCE ALGORITHMS

Hundreds of surveillance algorithms against massive amounts of data in multiple products (Equities, Options, etc.) and across multiple exchanges (NASDAQ, NYSE, CBOE, etc.).
FINRA’S SURVEILLANCE ALGORITHMS

Detecting abusive activity and compliance breaches
FINRA’S SURVEILLANCE ALGORITHMS

Dealing with big data before there was “Big Data”.

Over 430 batch analytics in the Surveillance suite.

“Massively Parallel Processing” methodology used to solve big-data problems in the legacy world.
Tiered storage design that struggles to balance Cost, Performance and Flexibility
PRE-HADOOP PAIN POINTS

Data Silos

Data distributed physically across MPP appliances, NAS and Tapes, affecting accessibility and efficiency.
PRE-HADOOP PAIN POINTS

Cost

Expensive, specialized hardware tuned for CPU, storage and network performance.

Proprietary software that comes with relative high cost & vendor lock-in.
Non-Elasticity

Can't grow or shrink easily with data volume, bound by the cost of hardware in the appliance and the relative high cost of the software.
ADDRESSING PAIN POINTS WITH...

HIVE
De facto standard for SQL-on-Hadoop

Amazon EMR
Amazon Elastic Map Reduce – Managed Hadoop Framework

Amazon S3
Amazon Simple Storage Service With Practically Infinite Storage
WHY SQL?

Heavily SQL based legacy application that works on source data that’s already available in structured format.

Hundreds of thousands of lines of legacy SQL code, iterated and tested rigorously through the years and readily available for easy Hive porting.

Developers, testers, data scientists and analysts with deep SQL skills and necessary business acumen already part of FINRA culture.
HIVE

Developed at Facebook and now open-source de facto SQL standard for Hadoop.

Been around for seven years, battle tested at scale, and widely used across industries.

Powerful abstraction over MapReduce jobs, translates HiveQL into map-reduce jobs that work against distributed dataset.
HIVE – EXECUTION ENGINES

**MapReduce**

- **Mature** batch-processing platform for the petabyte scale!
- Does not perform well enough for small data or iterative calculations with long data pipelines.

**Tez**

- Aims to translate complex SQL statements into an optimized, purpose-built data processing graphs.
- Strikes a balance between performance, throughput, and scalability.

**Spark**

- Fast in-memory computing, allows you to leverage available memory by fitting in all intermediate data.
AMAZON(AWS) S3

Cost-effective and durable object storage for a variety of content.

Allows separation of storage from compute resources, providing ability to scale each independently in pay-as-you-go pricing model.

Meets Hadoop’s file system requirements and integrates well with EMR.
EMR – ELASTIC MAP REDUCE

Managed **hadoop framework**, easy to deploy and manage Hadoop clusters.

Easy, fast, and cost-effective way to distribute and process vast amounts of data across dynamically **scalable Amazon EC2** Instances.

**Amazon EMR - Architecture**

- **Master instance**
  - Controls the cluster
- **Core instance**
  - Life of cluster
  - DataNode and TaskTracker daemons
- **Task instances**
  - Added or subtracted to perform work (SPOT)
- **S3 as underlying file system**
A wide selection of virtual Linux servers with varying combinations of CPU, memory, storage, and networking capacity.

Each instance type includes one or more instance sizes, allowing you to scale your resources to target workload.

Available in Spot (your bid price) or On-demand model.

[Source – Amazon]
CLOUD BASED HADOOP ARCHITECTURE

Postgres Surveillance Metadata

Management Services (Handling Process, Cluster and Data Management Needs)

Surveillance Patterns

Amazon S3

Virtual Private Cloud

Alert Management System

Reports

Splunk / Cloud Watch Analytics
**UNDERLYING DESIGN PATTERN**

**Transient Clusters with S3 as HDFS**

Cluster lives just for the duration of the job, shutdown when the job is done.

Persist **input and output data in S3**.

Run multiple jobs in **multiple Amazon EMR clusters** over the same data set (in S3) without overloading your HDFS nodes.

[Source – Amazon]
Control your cost, pay for what you use.

Minimum maintenance, cluster goes away when the job is done.

Persisting data in S3 enables easy reprocessing if the spot Instances are taken away due to an outbid.
Design your Hive batch analytics with focus on enabling direct data access and maximizing resource utilization.
Enable Direct Data Access

Partition data along natural query boundaries (e.g. trade date) and process only the data you need.

Improve join performance by **bucketing and sorting** data ahead of time and reduce I/O scans during join process.

Use **broadcast joins** when joining small tables.
HIVE – DESIGNING FOR PERFORMANCE

Tune Hive Configurations

Increase parallelism by tuning MapReduce split size to use all the available map slots in the cluster.

Increase the replication factor on dimension tables (not for additional resiliency but for better performance).

Compress intermediate data and reduce data transfer between mappers/reducers.
Measure and profile your clusters from the outset and adjust continuously keeping pace with changing data size, processing flow and execution frameworks.
Batch Process:
• Order Audit Trail Report Card Validation

Source Dataset:
• 1 month of market orders data
• Over 100 billion rows
• Around 5 terabytes in size
• Input stored in S3

Instance Choices:
• **m1.xlarge** *(General Purpose)* - 4 CPUs, 15 GB RAM, 4 x 420 Disk Storage, High network performance
• **c3.8xlarge** *(Compute Optimized)* - 32 CPUs, 60 GB RAM, 2 x 320 SSD Storage, High network performance

Run Options:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Option-1</th>
<th>Option-2</th>
<th>Option-3</th>
<th>Option-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance Type</td>
<td>m1.xl</td>
<td>C3.8xl</td>
<td>C3.8xl</td>
<td>C3.8xl</td>
</tr>
<tr>
<td>Instance Classfication</td>
<td>General Purpose</td>
<td>Compute Optimized</td>
<td>Compute Optimized</td>
<td>Compute Optimized</td>
</tr>
<tr>
<td>Cluster Size</td>
<td>100</td>
<td>16</td>
<td>32</td>
<td>100</td>
</tr>
</tbody>
</table>
CLUSTER PROFILE – **COMPARISON** RESULTS

100 M1s Vs 16 C3s Vs 32 C3s Vs 100 C3s

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Option-1</th>
<th>Option-2</th>
<th>Option-3</th>
<th>Option-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance Type</td>
<td>m1.xl</td>
<td>C3.8xl</td>
<td>C3.8xl</td>
<td>C3.8xl</td>
</tr>
<tr>
<td>Instance Classification</td>
<td>General Purpose</td>
<td>Compute Optimized</td>
<td>Compute Optimized</td>
<td>Compute Optimized</td>
</tr>
<tr>
<td>Cluster Size</td>
<td>100</td>
<td>16</td>
<td>32</td>
<td>100</td>
</tr>
<tr>
<td>Cost</td>
<td>$57.33</td>
<td>$111.86</td>
<td>$127.05</td>
<td>$312.00</td>
</tr>
<tr>
<td>Time</td>
<td>3 Hrs 12 mins</td>
<td>8 Hrs</td>
<td>4 Hrs 56 mins</td>
<td>3 Hrs</td>
</tr>
<tr>
<td>Approximate Spot Price per Instance</td>
<td>$0.14</td>
<td>$0.78</td>
<td>$0.66</td>
<td>$0.78</td>
</tr>
<tr>
<td>Peak Disk Usage</td>
<td>0.04%</td>
<td>2.10%</td>
<td>1.20%</td>
<td>0.45%</td>
</tr>
<tr>
<td>Peak Memory Usage</td>
<td>56%</td>
<td>43%</td>
<td>45%</td>
<td>41%</td>
</tr>
<tr>
<td>Peak CPU Usage</td>
<td>100%</td>
<td>95%</td>
<td>98%</td>
<td>92%</td>
</tr>
</tbody>
</table>
Abstract execution framework (MR, Tez, Spark) from business logic in your app architecture. This allows switching frameworks, keeping up with Hive community.
Hive UDFs (User Defined Functions) can solve your complex problems when SQL falls short.
HIVE UDFS

Used when Hive functionality falls short
- e.g. window function with ignore nulls – not supported in Hive
- e.g. date formatting functions – Hive 1.2 has better support

Used when non-procedural SQL can’t accomplish the task
- e.g. de-dupe many-to-many time series pairings
  A->B 10AM, A->C 11AM, D->B 11AM, D->C 11AM, E->C 12PM
Choose an optimal storage format (row, columnar) and compression type (splittable, space efficient, fast).
FILE STORAGE – FORMAT AND COMPRESSION

STORAGE FORMATS

Columnar or Row based storage

RCFILE, ORC, PARQUET – Columnar formats, skip loading unwanted columns!

COMPRESSION TYPES

Reduce the number of bytes written to/read from HDFS

Some are fast but offer less space reduction, some are space efficient but slower, some are splittable and some are not.

<table>
<thead>
<tr>
<th>Compression</th>
<th>Extension</th>
<th>Splittable</th>
<th>Encoding/Decoding Speed (Scale 1-4)</th>
<th>Space Savings % (scale 1-4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GZIP</td>
<td>GZ</td>
<td>NO</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>LZO</td>
<td>LZO</td>
<td>YES IF INDEXED</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>BZIP2</td>
<td>BZ2</td>
<td>YES</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>SNAPPY</td>
<td>SNAPPY</td>
<td>NO</td>
<td>4</td>
<td>1</td>
</tr>
</tbody>
</table>

[Source – Amazon]
Being at the bleeding edge of sql-on-hadoop has its risks, you need the right strategy to mitigate risks/challenges along the way.
RISKS AND MITIGATION STRATEGIES

Hive, released in late 2008, Tez/Spark backend added recently.

Traditional platforms like Oracle tested and iterated through four decades.

Discovered issues with Hadoop/Hive during migration that are related to compression, storage types and memory management.

Performance issues with interactive analytics – Impacts Data Scientists & Business Analysts.
RISKS AND MIGRATION STRATEGIES

Extensive parallel run comparison (apples-to-apples) against legacy production data to identify issues before production roll out.

Partnered with Hadoop and Cloud vendors to address Hadoop/Hive issues and feature requests quickly.

Push of a button automated regression test suites to kick the tires and test functionality for any minor software increments.

Analyzing Presto/Tez to solve interactive analytics problems.
Took advantage of cloud elasticity to complete parallel runs against production volume at scale to produce results swiftly.
COST

Cost savings in infrastructure relative to our legacy environment.

Minimal upfront spending, pay-as-you-go pricing model.

Variety of machine types and size to choose from based on cost and performance needs.
END STATE

**FLEXIBILITY**

Dynamic infrastructure provides great flexibility with faster reprocessing and testing needs.

Simplified software procurement and license management.

Ease of exploratory runs without affecting production workload.
SCALABILITY

Scale out at will easily on high volume days.

Cloud elasticity enables running multiple days at scale in parallel.

Reprocessing for historic days is completed in hours compared to weeks in our legacy environment.
QUESTIONS?

Jaipaul Agonus
FINRA
Jaipaul.agonus@finra.org
linkedin.com/in/jaipaulagonus