The Challenges of Managing Real-Time Financial Market Data Storage

SREcon23 Asia/Pacific
June 16, 2023

Kiran Kasichayanula, System Reliability Engineer
Nishith Nedungadi, SRE Team Leader
Feeds Engineering

TechAtBloomberg.com
Agenda

- Introduction to market data
- Existing workflows and challenges in managing data storage
- Deep dive into our solution
- The road ahead
The **Bloomberg Terminal** is software that delivers a diverse array of information, news and analytics to facilitate financial decision-making.
Primer: What is Market Data?

Exchanges - Source of market data

- **Instrument**
- **Orders**
- **Quotes**
- **Trades**
Primer: What is Market Data?

Normalization: The representation of information as basic as trade price and trade quantity can vary substantially across major exchanges.

Price and quantity have different data types

Ultra fast, tightly packed price and volume with venue identification

<table>
<thead>
<tr>
<th>Tag</th>
<th>Tag Name</th>
<th>Req</th>
<th>Data Type</th>
<th>Comment</th>
<th>EQT</th>
<th>DER</th>
</tr>
</thead>
<tbody>
<tr>
<td>→</td>
<td>270</td>
<td>C</td>
<td>Price</td>
<td>Price of the Market Data Entry. Required when this market data entry involves a price. Represents the notional value for trade volume (B). Other entry types that do not involve price do not require this tag.</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>→</td>
<td>271</td>
<td>C</td>
<td>Qty</td>
<td>Quantity or volume represented by the Market Data Entry. Required when MDUpdateAction = New (0) and MDEntryType = Bid (0), Offer (1), Trade (2), Trade Volume (8) or Opening Price (4).</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Trade Price</td>
<td>2</td>
<td>Short</td>
</tr>
<tr>
<td>Trade Volume</td>
<td>2</td>
<td>Short</td>
</tr>
<tr>
<td>Primary Listing Market Participant ID</td>
<td>1</td>
<td>Char</td>
</tr>
</tbody>
</table>
Feed Handler Application

**Normalization**: The representation of information as basic as trade price and trade quantity can vary substantially across major exchanges.

<table>
<thead>
<tr>
<th>Tag</th>
<th>Tag Name</th>
<th>Req</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>270</td>
<td>MDEntryPx</td>
<td>C</td>
<td>Prices</td>
</tr>
<tr>
<td>271</td>
<td>MDEntrySize</td>
<td>C</td>
<td>Qty</td>
</tr>
</tbody>
</table>

**Example Normalized Market Data**:

```
TradeWithTradeID 102|PriceClass 1|FeSubSequence1 0|FeSubSequence2 0|Sequence 952101|Timestamp 06/08/2023 06:59:28|Key TEST TradePrice 10|TradeSize 100|ConditionCodes 1: 19|TradeID 0001|
```

<table>
<thead>
<tr>
<th>Trade Price</th>
<th>2</th>
<th>Short</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trade Volume</td>
<td>2</td>
<td>Short</td>
</tr>
<tr>
<td>Primary Listing Market Participant ID</td>
<td>1</td>
<td>Char</td>
</tr>
</tbody>
</table>
Bloomberg Market Data Pipeline

Exchanges
Source of market data

Feed Handlers
Connectivity Normalization

Storage
Consolidates and stores all data from all Exchanges

Data Distribution Infrastructure
Global distribution Subscription management

The Terminal, Microsoft Excel, and Enterprise applications present data to clients using a unified interface
Feed Handler Application

Exchanges → Session → Business → Raw → Normalized → Downstream
Challenge: Managing Raw / Normalized Data Storage

Business analysts and Engineering need a system to access the incoming raw market data and the normalized market data generated by the feed handlers.

WHY?
- Business uses this data to understand and analyze raw data sent by exchanges
- To answer client queries
- Communication tool between Engineering and Business
- Debugging their applications
- To create new tests of interesting scenarios
- To validate the correctness of their processing
- Recover from outages
Managing Storage: Existing System
Managing Storage: Non-uniformity

Feed Machine
- Feed Handler 1
- Feed Handler 2
- Feed Handler 3
- Feed Handler N

Different Names
Different Locations
Number of Files

NFS
Dev
Managing Storage: Capacity impacts Uniform Policy

~100 TB in the last 24 hours
1000s of File per machine

300 Billion Messages per Day
200 Billion Messages per Day
100 Billion Messages per Day

2008 Financial Crisis
US Debt Ceiling Crisis (S&P Downgrade)
Japan Earthquake & Tsunami
Flash Crash
Stimulus Tapering Discussion
Managing Storage: Network zones security / bandwidth
Managing Storage: Why Files?

WHY FILES?
- Existing interface
- Characteristic of market data
Market Data Rates: Spikiness
Requirement

Business analysts and Engineering need a system to access the incoming raw data and the normalized market data generated by the feed handlers.
What we want

- Scalable
- Agile
- Secure
- Storage agnostic
- Datacenter agnostic
- Observable
- Correct
- Accessible
Architecture Deep Dive
Architecture: Object Data Store in the Cloud
Architecture: Object Data Store in the Cloud

Machine

File properties
- file_type: input
- machine_name: machine1
- feed_name: feed1
- creation_date: 2023-05-29
- version: 1
- file_name: input1.dat

Input File
1 2 3
4 5 n

Object storage organization in cloud
- input/machine1/input1.dat/2023-05-29/1/feed1.parts/part000001
- input/machine1/input1.dat/2023-05-29/1/feed1.parts/part000002
- input/machine1/input1.dat/2023-05-29/1/feed1.parts/partn
Architecture: Network Zones in the Cloud
Architecture: Local Agent Running on Machine

- Feeds Machine 1
  - Agent
- Feeds Machine 2
  - Agent
- Feeds Machine 3
  - Agent
  
- Feeds Machine N
  - Agent

Cloud Infrastructure

- Storage
- Storage
Architecture: Local Agent Running on Machine

Feed Machine

Feed Handler 1

Feed Handler 2

Feed Handler 3

Feed Handler N

Data Files

Data Files

Data Files

Data Files

Local Agent

Sidecar

Sidecar Properties:

Resource constrained
Appropriate NICE values
Architecture: Asyncio Library

- Find Files
- Read File
- Chunk File
- Upload Chunk

Event Loop

- Filesystem
- Network
- Process
Architecture: Local Agent Running on Machine
Architecture: Disk Rate Limiter

Disk Reader Task 1 (File X) → Chunk (32 MB) → Chunk (32 MB) → 32 MB * [0,N] → Disk Read Rate Limiter (Local) Capacity: 30 GB

Disk Reader Task K (File Y) → Chunk (32 MB) → Chunk (32 MB) → 32 MB * [0,N] → Disk Read Rate Limiter (Local) Capacity: 30 GB

Disk Reader Task N (File Z) → Chunk (32 MB) → Chunk (32 MB) → 32 MB * [0,N] → Disk Read Rate Limiter (Local) Capacity: 30 GB
Architecture: Local Agent Running on Machine
Network Rate Limiter
Application Aware

- Exchange
- Application space
  - ptr
- Agent
- Memory mapped file
- OS space
  - 1
  - 2
  - x bytes
Application aware
Architecture: Client APIs / Utilities
Client Access
(Internal Business / Engineering)

- Business users (Web API powered by Kubernetes clusters)
  - On-demand import requests
  - Intra-session data
  - Centralized decoding

- Developers (Development machines)
  - Match input and output
  - Debug intra-day issues
  - Testing new parser deployment
Monitoring and dashboards

- Max Chunk Upload Time:
  40 seconds, 137 milliseconds

- Median Chunk Upload Time:
  36 seconds, 917 milliseconds

- Min Chunk Upload Time:
  36 seconds, 917 milliseconds

- Last 30 minutes:
  1 minute, 52 seconds, 25 seconds
The road ahead

• Scalability
• Real-time
• Improving observability
Takeaways

- A glimpse into real-world market data management challenges
- Applying cloud storage
- Considerations in building out a system
Questions?
Visit our booth

https://TechAtBloomberg.com

https://www.bloomberg.com/careers

Contact us:
kkasichayanu@bloomberg.net | nnedungadi@bloomberg.net