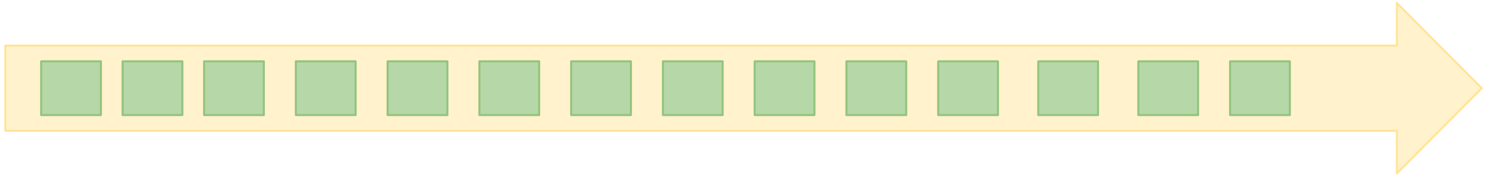
KDD2016
22nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining
August 13 - 17, 2016 | San Francisco, California

Streaming Analytics

Ashish Gupta (LinkedIn)
Neera Agarwal

What is a Data Stream

- Unbounded Data
- Data arriving continuously at high rate
- Too large to first store and then process
- Need to be processed in one pass
- May display Temporal Locality - patterns may evolve over time



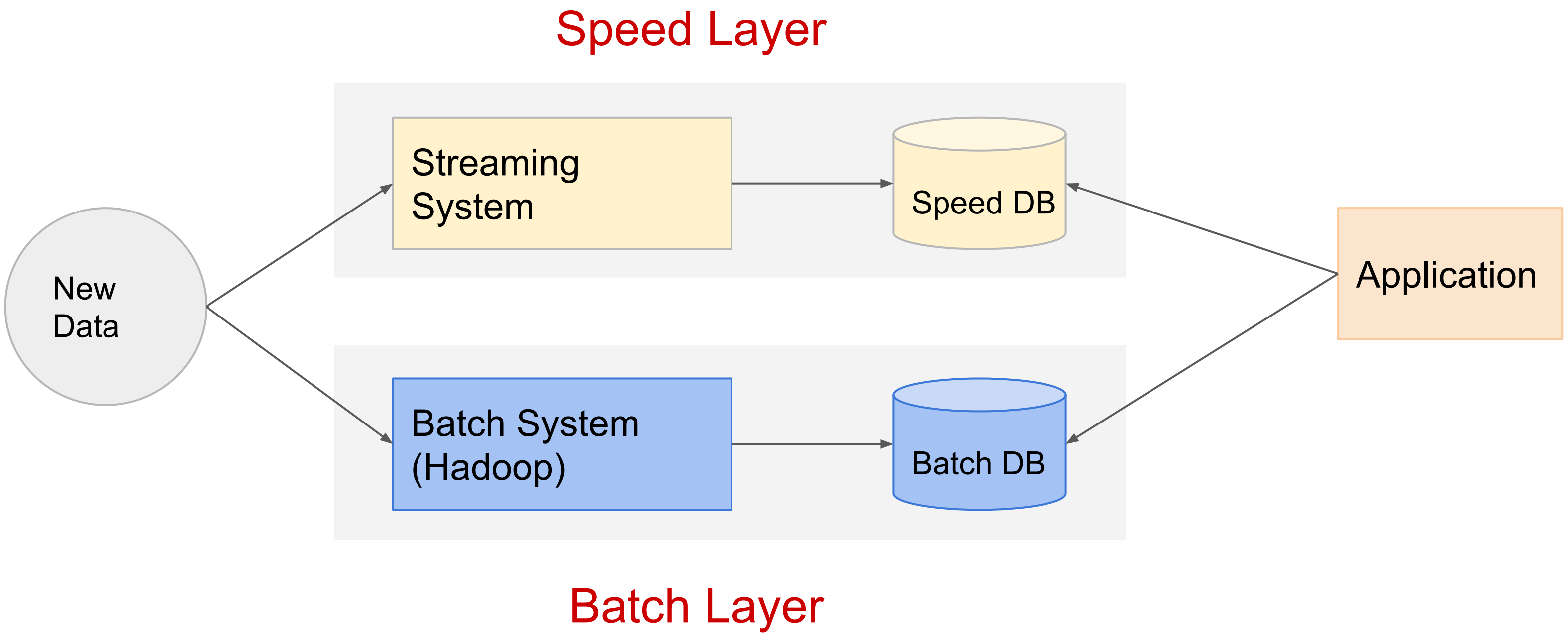
Contrast with Batch Processing

1. Process Bounded Files - such as files by ingestion time - files by last 15 minutes, last 1 hour, last 1 day
2. Program can go back and forth in data. Do multipass processing.
3. Sessions and joins can span files.
4. Many machine learning algorithms need full batch of data to train.
5. Very high Latency, but very high throughput as well.
    a. Wait for files to arrive. Ie wait for file window to close.
    b. Processing the whole file(s) take time.

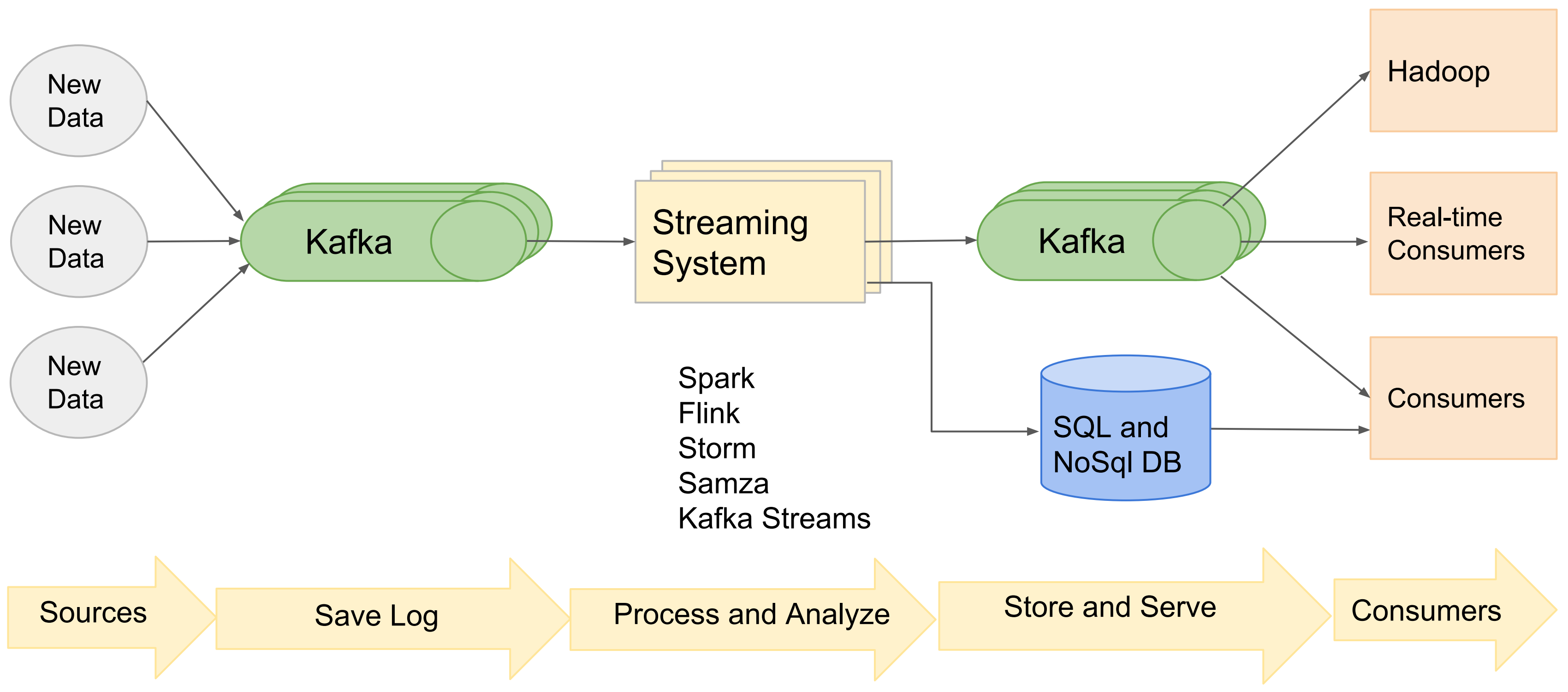
Streaming Applications

- Joining Clicks and Impressions
- Mobile applications - User activity
- Session based analysis
- Fraud detection
- Industrial IOT
- LinkedIn’s Streaming Standardization Platform

Lambda Architecture



Streaming Systems Architecture



What is Streaming Analytics

“ Continuous processing on unbounded data”

“Software that can filter, aggregate, enrich, and analyze a high throughput of data from multiple disparate live data sources and in any data format to identify simple and complex patterns to visualize business in real-time, detect urgent situations, and automate immediate actions.” - Forrester

APACHE Spark™

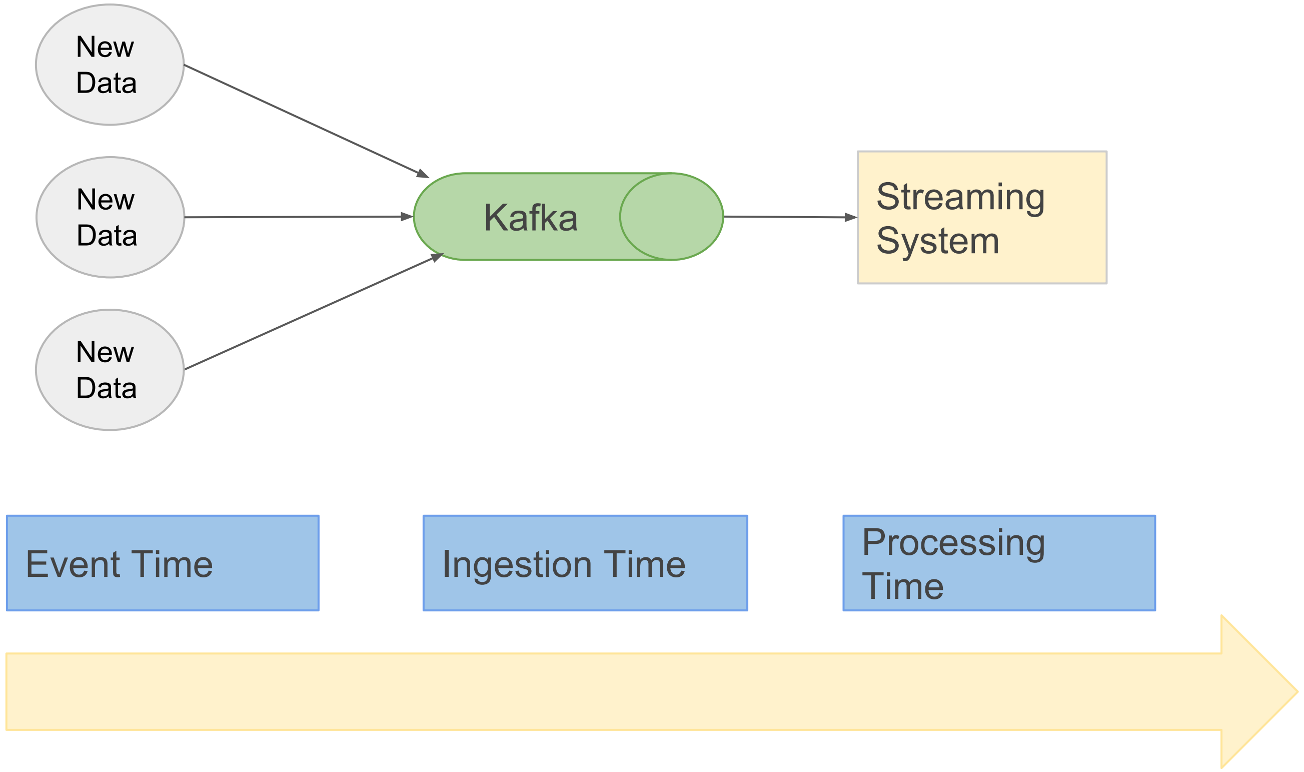
Flink

APACHE STORM™

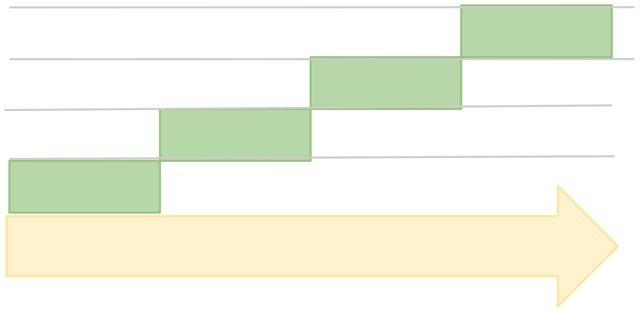
Streaming Concepts

Time	Window	Order	Correctness
Event Time	Fixed Window	Delayed data	Consistency
Processing Time	Sliding Window	Out of order data	At least Once
	Sessions		Exactly Once
			Checkpointing

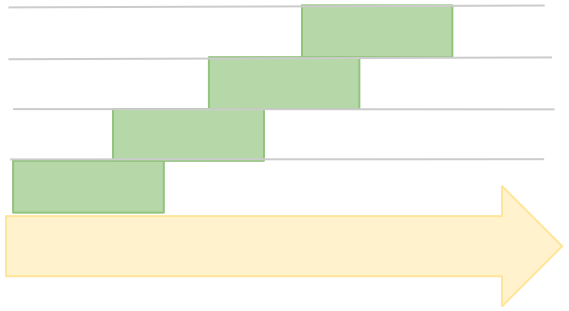
Streaming Concepts - Time



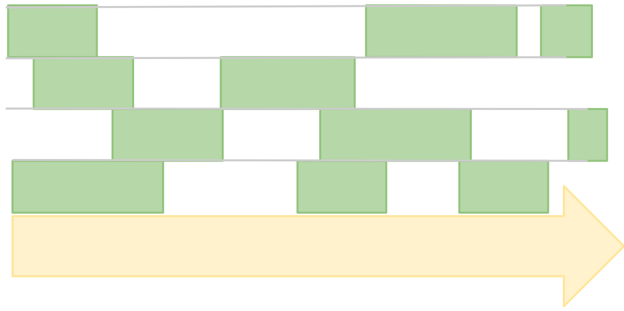
Streaming Concepts - Windows



Fixed Window/
Tumbling Window



Sliding Window

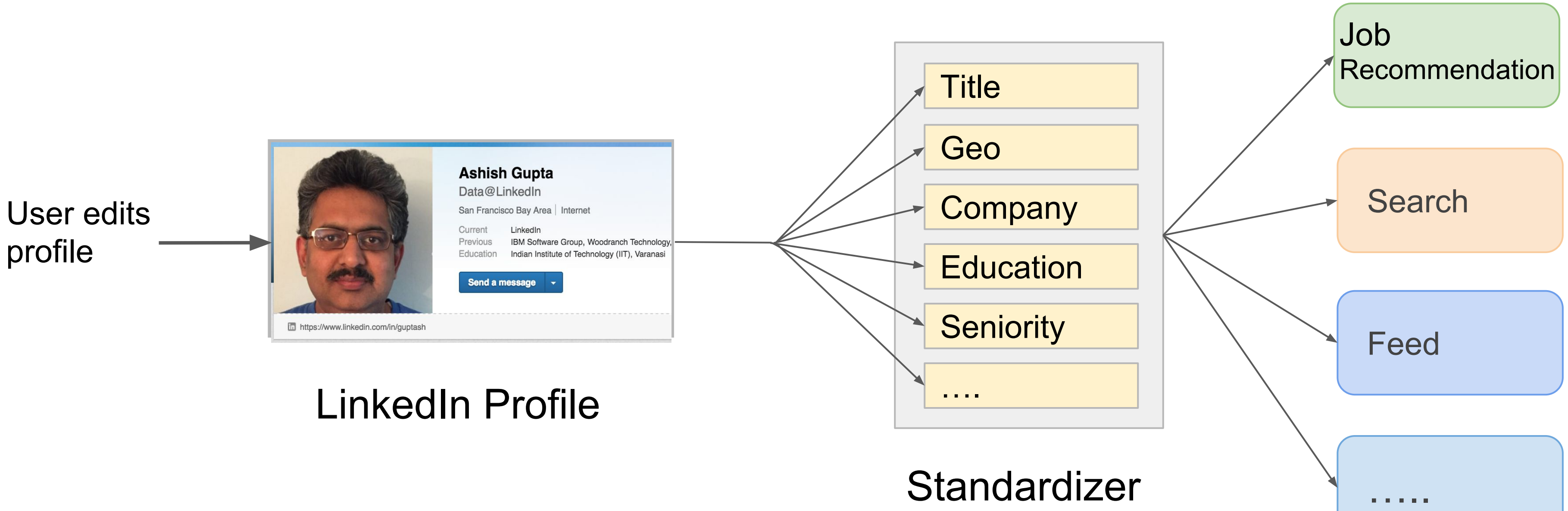


Session Window

Open Source Streaming Systems

	APACHE Spark	Flink	APACHE STORM	samza	Kafka Streams
Processing Model	Mini Batch	Event level	Event level	Event level	Event level
Guarantee	Exactly Once	Exactly Once	At least once	At least once	At least once
State Management	Yes	Yes	No	Yes	Yes
Latency	Medium	Low	Low	Low	Low
Built in primitives	Batch and streaming	Batch and streaming	Low Level API	Low level API	Streaming only
Back Pressure	Yes	Yes	No	via Kafka	via Kafka

Case Study: LinkedIn Standardization Platform



Ashish Gupta
Data@LinkedIn
San Francisco Bay Area | Internet

Current	LinkedIn
Previous	IBM Software Group, Woodranch Technology,
Education	Indian Institute of Technology (IIT), Varanasi

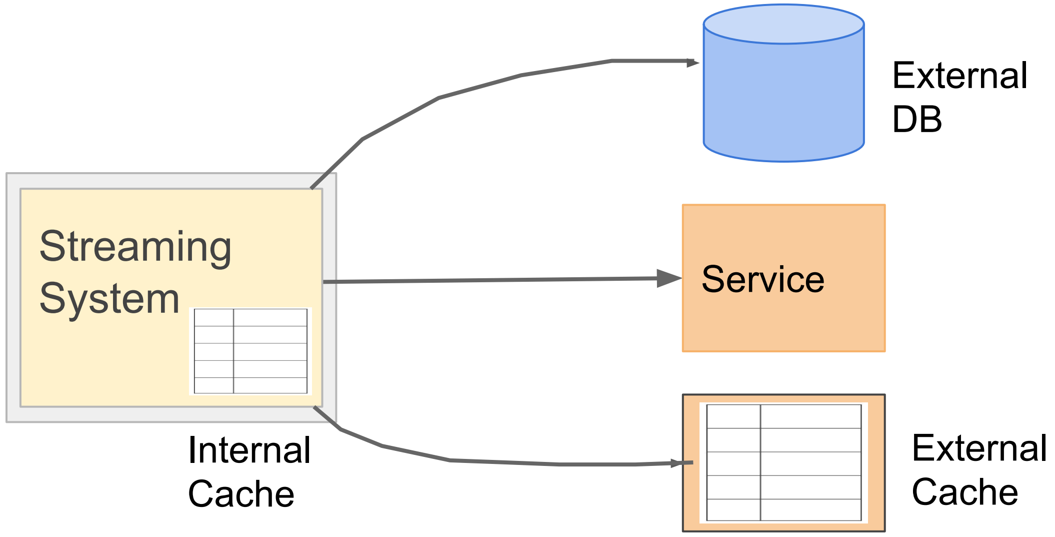
Send a message

https://www.linkedin.com/in/guptash

LinkedIn Profile

Standardizer

Pattern : External Lookup/Stream to Table Join

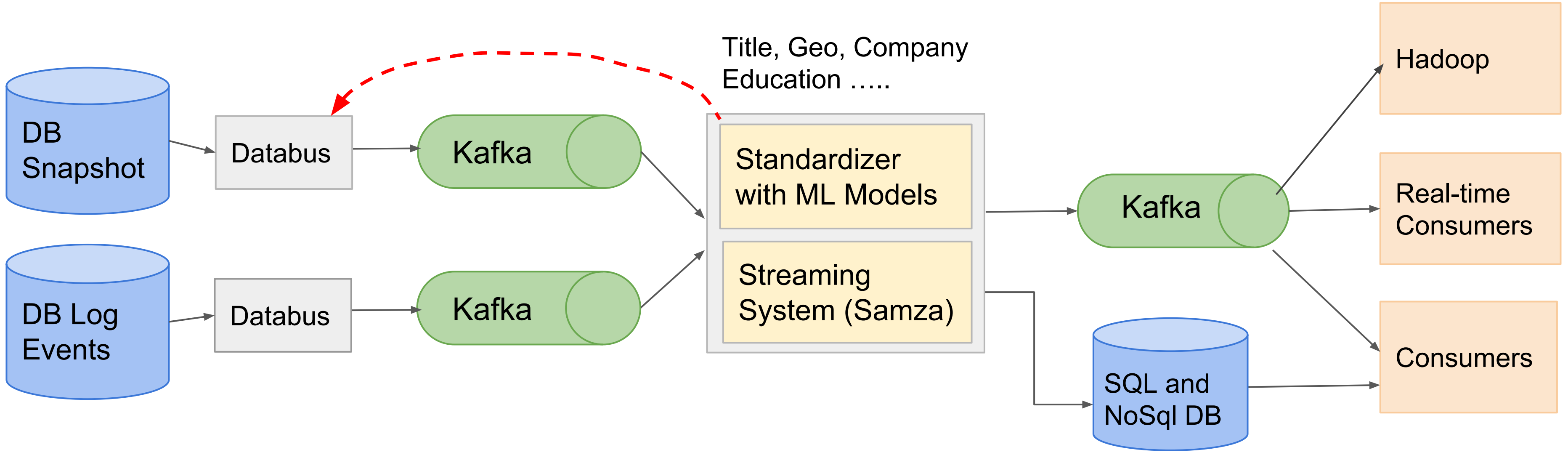


Decide based on size of the data, latency needs and QPS of external systems

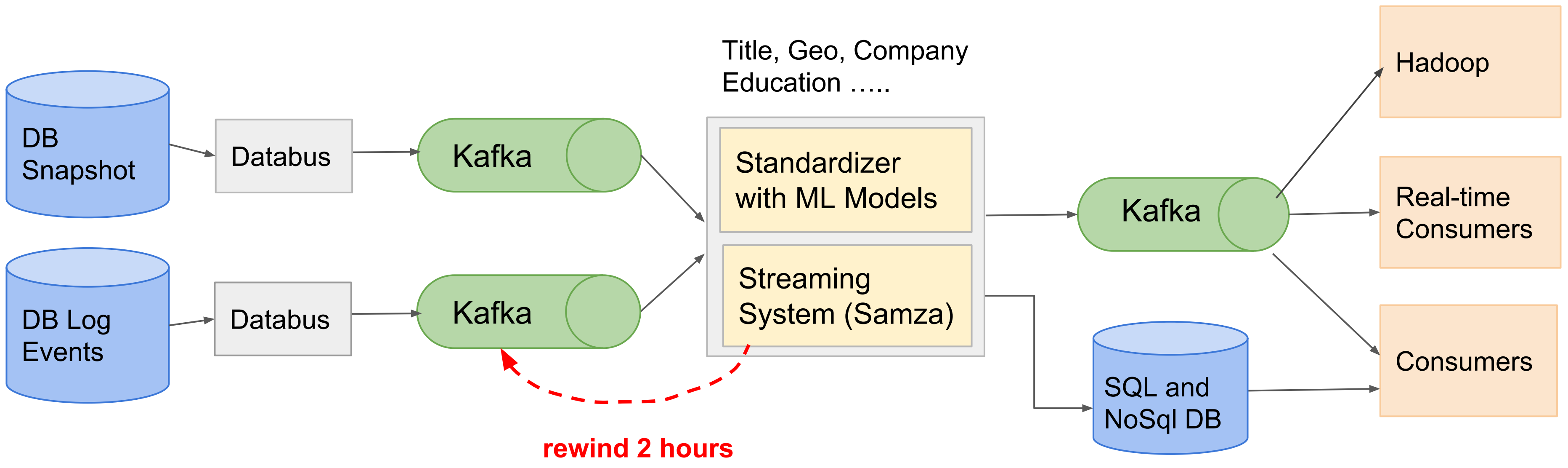
Pattern: Stream to Stream Joining

- Joins are expensive
- If partitions of two streams not collocated, then expensive shuffle
- Broadcast join if one file is small

Pattern: Reprocessing



Pattern: Reprocessing

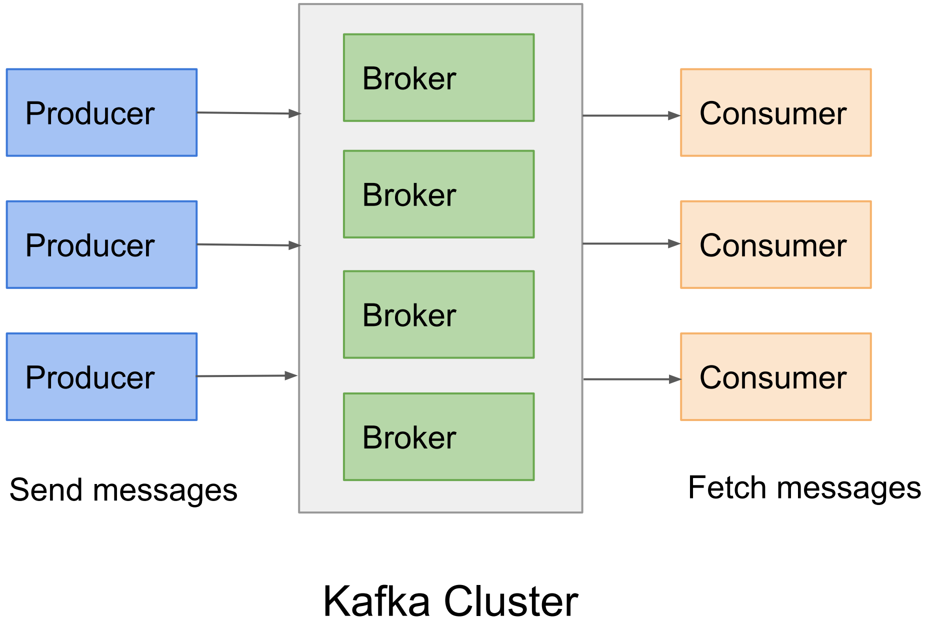


Apache Kafka

- Highly scalable messaging system
- Distributed commit log
- Developed in LinkedIn back in 2010
- At LinkedIn - more than 1.4 trillion messages per day across over 1400 brokers
- Distributed, partitioned, replicated
- Message retention - based on time and size

Some Kafka use cases

- Queuing/Messaging
- Metrics
- Auditing
- Logging



References

- MillWheel: http://research.google.com/pubs/pub41378.html
- DataFlow:http://research.google.com/pubs/pub43864.html
- Samza: http://samza.apache.org/
- Spark Streaming Paper: Discretized streams
- Models and Issues in Data Stream Systems

Contact Us

Ashish Gupta - ahgupta@linkedin.com

https://www.linkedin.com/in/guptash

Neera Agarwal - neera8work@gmail.com

https://www.linkedin.com/in/neera-agarwal-21b9473