Designing and Building Next Generation Data Pipelines at Scale with Structured Streaming

Burak Yavuz

October 4th 2018, London

#SAISDev15

databricks

Who am I

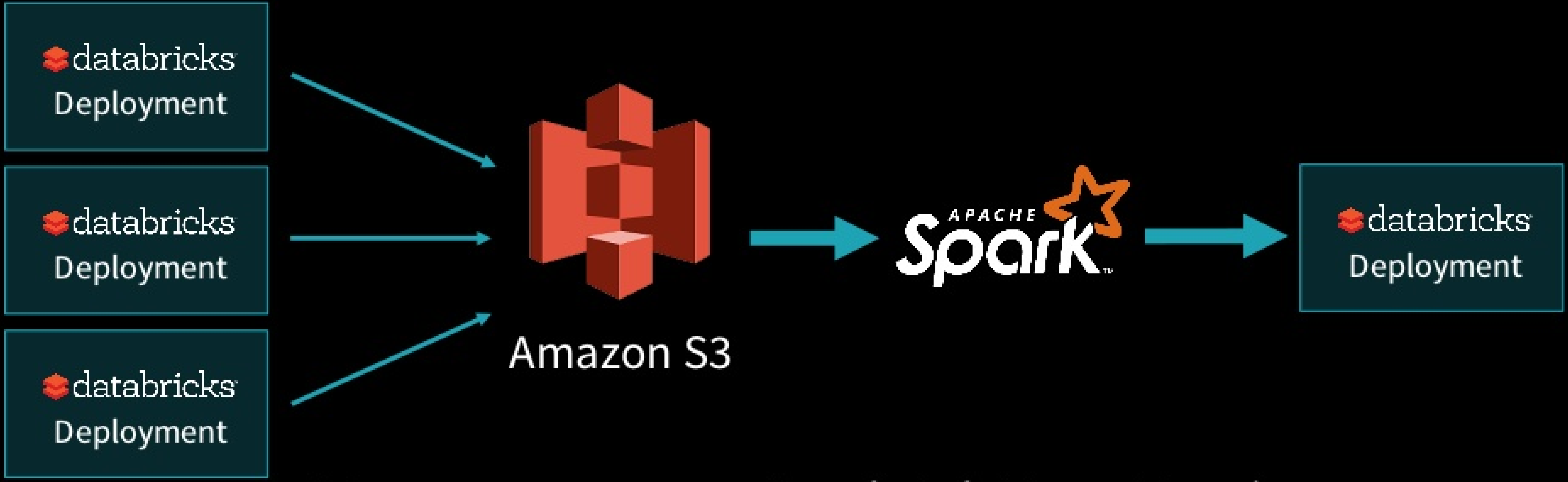
databricks

- Software Engineer – Databricks
  - “We make your streams come true”
- Apache Spark Committer
- MS in Management Science & Engineering - Stanford University
- BS in Mechanical Engineering - Bogazici University, Istanbul

Today, we're going to ride a time machine

Let’s go back to 2014…

Evolution of Data Pipelines @ Databricks
Circa MMXIV



S3-copy

Batch Job Every Hour*

Data Pipeline V1

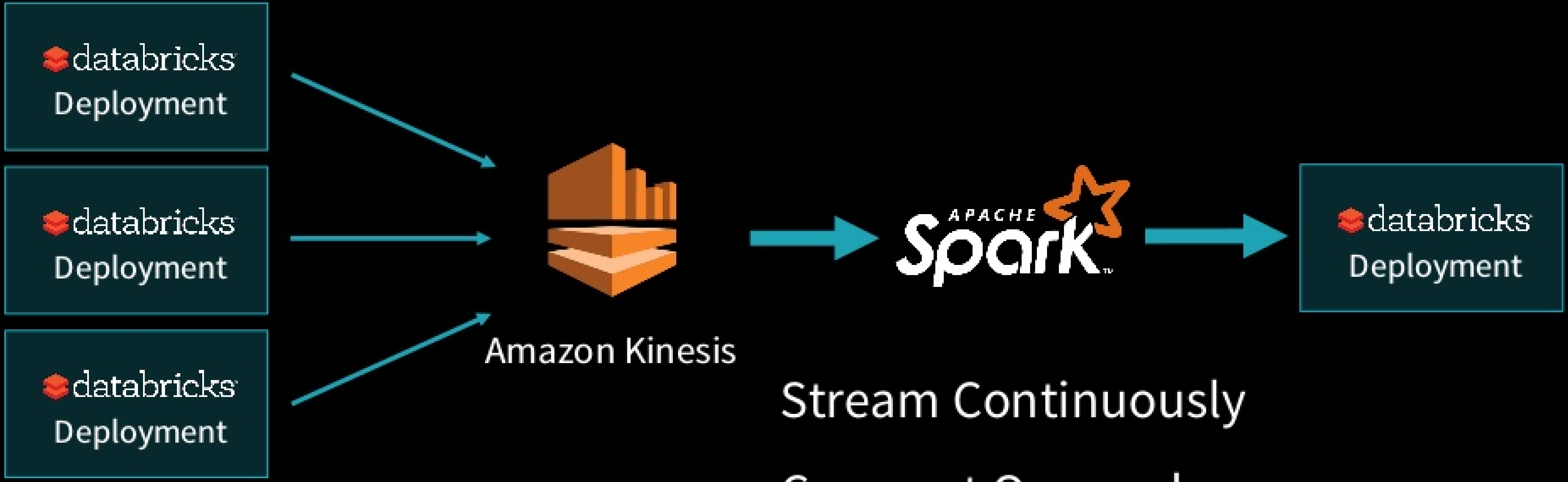
- Took 1 engineer ~1 week to implement
- Was pretty robust for the early days of Databricks

- … until we got to 30+ customers
- File listing on S3 quickly became the bottleneck
- *eventually, job duration rose to 8 hours

Fast-forward to 2016…

databricks

Evolution of Data Pipelines @ Databricks
Circa 2016



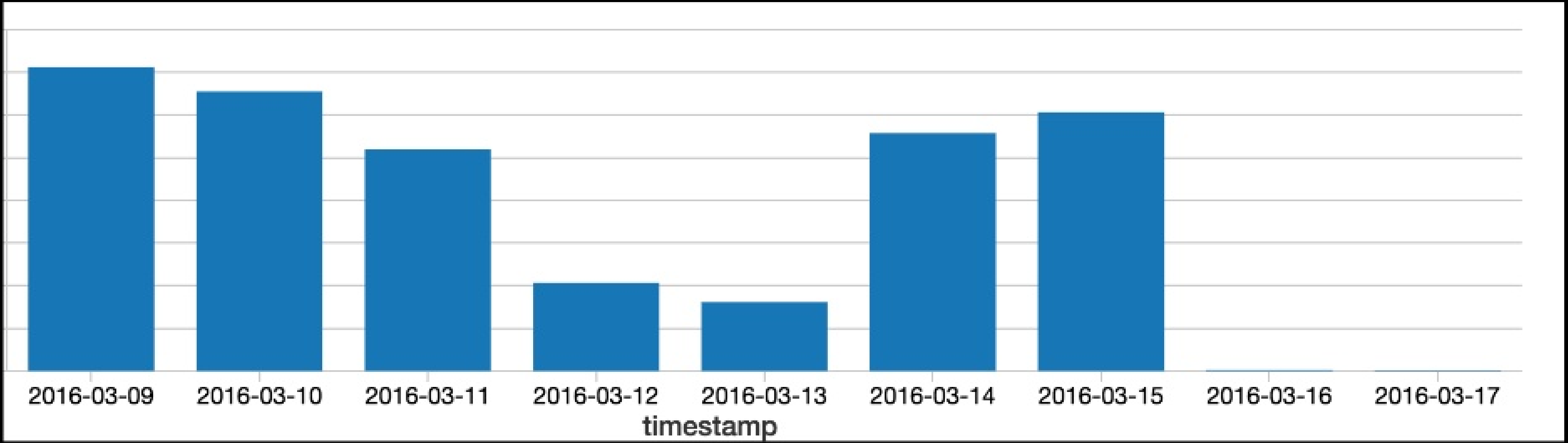
Stream Continuously

Compact Once a day

Data Pipeline V2

- Scaled very well
- ETL'ing the data became fast
- Took 2 engineers ~8 months
- Query performance / experience got worse
  - Lots of small files
  - Compaction jobs impacted queries (FileNotFoundExceptions)
  - HiveMetaStore quickly became bottleneck
  - REFRESH TABLE / MSCK REPAIR TABLE / ALTER TABLE ADD PARTITIONS
- Logic became more complicated. Pipeline was less robust
- Fixing mistakes in the data became harder

What Happened?



Examples of Data Mistakes

- A field’s unit changed from MB to GB
- Schema inference / mismatch
  - An integer column in JSON started getting inferred as longs after Spark upgrade. Some Parquet files had ints, some had longs
  - A different type of log got introduced to the system. All of a sudden a table with 8 columns had 32 new columns introduced
- Garbage data caused by partial failures

databricks

Problems in Data Pipelines

- Correctness
  - Lack of atomicity leads to pockets of duplicate data
  - Bookkeeping of what to process is tedious – late / out-of-order data
  - Schema Management
  - Maintaining Data Hygiene – checks/corrections
- Performance
  - Listing files on blob storage systems is slow
  - Lots of small files hurt query performance
  - HiveMetaStore experience is horrendous
    - Doesn't scale well
    - Having to call MSCK REPAIR TABLE and REFRESH TABLE all the time

Enter Structured Streaming

You care about your business logic

Structured Streaming cares about incrementally running your logic on new data over time

databricks

Correctness Problems in Data Pipelines

- ~~Lack of atomicity leads to pockets of duplicate data~~
  - Structured Streaming writes a manifest file to "commit" data to a file sink, which atomically appear to Spark
- ~~Bookkeeping of what to process is tedious – late / out-of-order data~~
  - The engine keeps track of what data is processed, and what data is new
  - Watermark support allows "correct" processing of late – out of order data.
- Schema Management
- Maintaining Data Hygiene – checks/corrections

databricks

Performance Problems in Data Pipelines

- ~~Listing files on blob storage systems is slow~~
  - The manifest lists which files were written by Structured Streaming, therefore no longer need to list
- Lots of small files hurt query performance
- HiveMetaStore experience is horrendous
  - Doesn't scale well
  - Having to call MSCK REPAIR TABLE and REFRESH TABLE all the time

Enter Databricks Delta

- Separates Compute from Storage
- No dependency on HiveMetaStore – Manages metadata internally
- Scales to Billions of partitions and/or files
- Supports Batch/Stream Reads/Writes
- ACID Transactions
- Compaction and Indexing
- Schema Management / Invariant Support
- Tables Auto-Update and provide Snapshot Isolation
- DELETE / UPDATE / MERGE
- Leverages data locality through DBIO Caching
- Coming Q4 2018 => Querying old versions of the table + Rollbacks

Correctness Problems in Data Pipelines

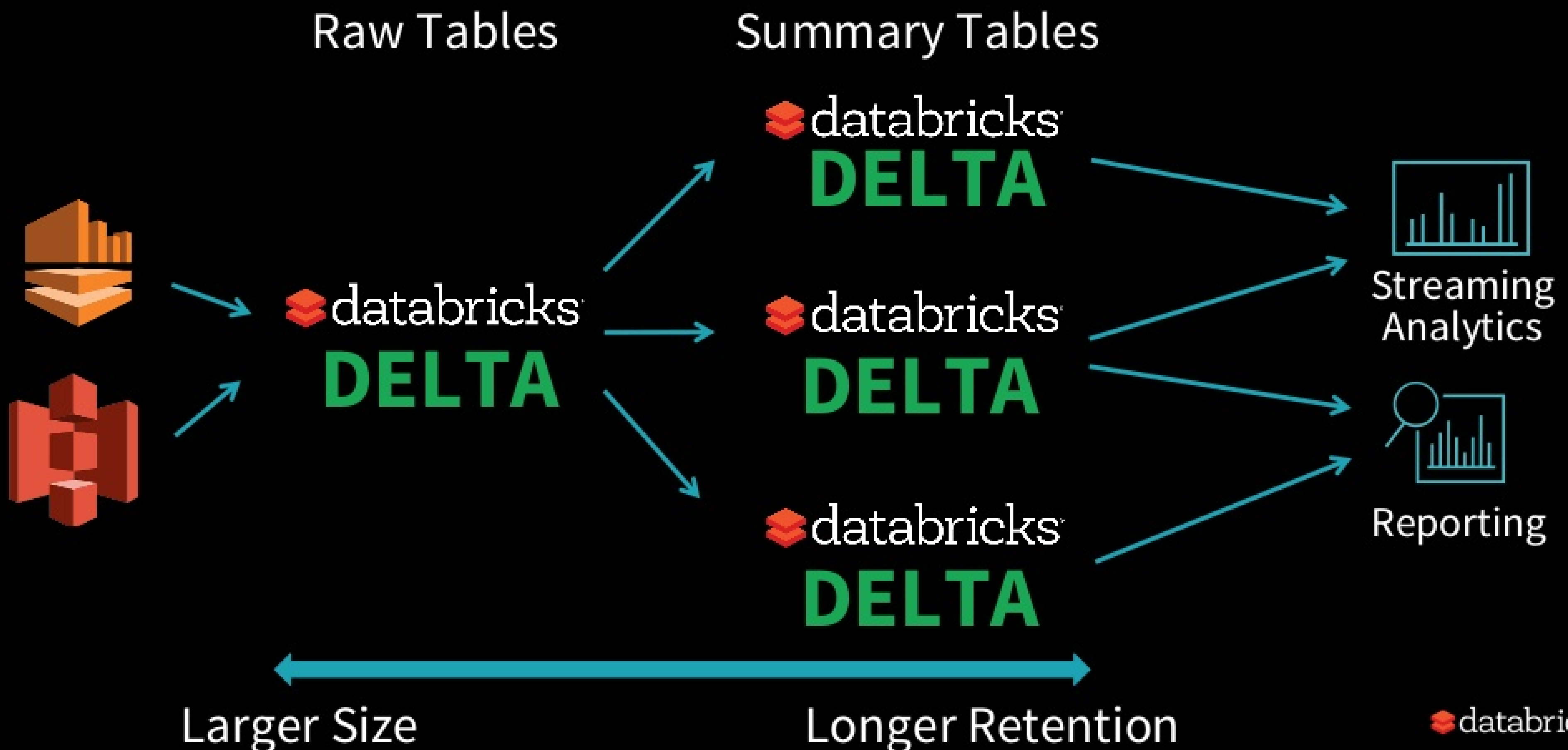
- ~~Lack of atomicity leads to pockets of duplicate data~~
  - Delta provides ACID transactions and Snapshot Isolation
- ~~Bookkeeping of what to process is tedious – late / out-of-order data~~
  - Structured Streaming handles this. Streaming into Delta provides exactly-once semantics
- ~~Schema Management~~
  - Delta manages the schema of the table internally and allows “safe” (opt-in) evolutions
- ~~Maintaining Data Hygiene – checks/corrections~~
  - Delta supports DELETE / UPDATE to delete/fix records
  - Delta supports Invariants (NOT NULL, enum in (‘A’, ‘B’, ‘C’))

databricks

Performance Problems in Data Pipelines

- ~~Listing files on blob storage systems is slow~~
  - Delta doesn’t need to list files. It keeps file information in its state
- ~~Lots of small files hurt query performance~~
  - Delta’s OPTIMIZE method compacts data without affecting in-flight queries
- ~~HiveMetaStore experience is horrendous~~
  - Delta uses Spark jobs to compute its state, therefore metadata is scalable!
  - Delta auto-updates tables, therefore you don’t need REFRESH TABLE / MSCK REPAIR TABLE / ALTER TABLE ADD PARTITIONS, etc

Delta @ databricks



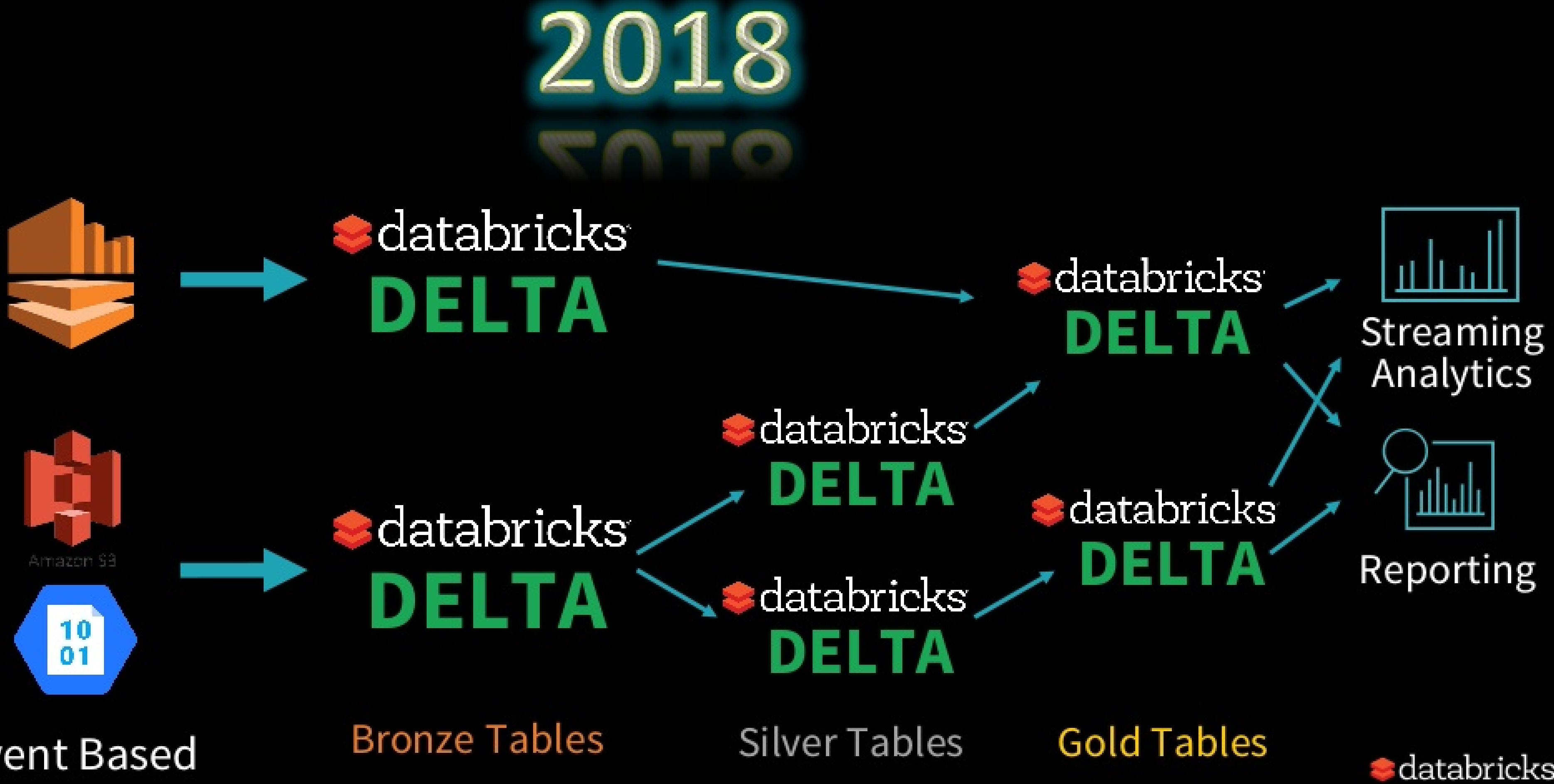
New Requirements

- Launched Azure Databricks
- Can’t leverage Kinesis anymore
- Have to replicate pipeline in many Azure Regions

New Requirements

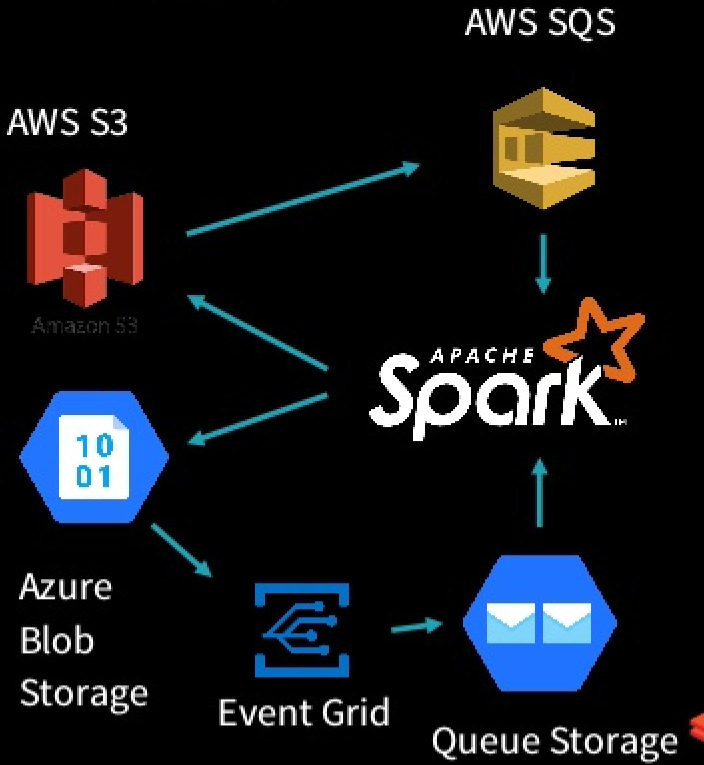
databricks

Evolution of Data Pipelines @ Databricks



Event Based File Sources

- Launched Structured Streaming connectors:
  - s3-sqs on AWS (DBR 3.5)
  - abs-aqs on Azure (DBR 5.0)
- As blobs are generated:
  - Events are published to SQS/AQS
  - Spark reads these events
  - Then reads original files from blob storage system



databricks

Properties of Bronze/Silver/Gold

- Bronze tables
  - No data processing
  - Deduplication + JSON => Parquet conversion
  - Data kept around for a couple weeks in order to fix mistakes just in case
- Silver tables
  - Directly queryable tables
  - PII masking/redaction
- Gold tables
  - Materialized views of silver tables
  - Curated tables by the Data Science team

Dealing with GDPR

- Delta's in-built support for DELETE and UPDATE make data subject requests (DSR) tractable
  - Delete or update the records
  - Run VACUUM after 7 days (configurable) and the old data is gone!
- Check out blog post for more details!

databricks

Data Pipeline V3

- Event base file sources avoid file listing altogether
- Scales even better than V2
- Easy to replicate across Clouds and regions
- Run Once trigger gives all benefits of Structured Streaming with cost benefits of running batch jobs
- Delta makes GDPR easy
- Latency went from 15 seconds to 5 minutes for general case

Other Techniques

- Leverage S3 Inventory and Delta’s transaction log to unearth value from 500 million JSON files
- Using S3 Select to reduce data size
- Using Continuous Processing to process data from Kafka and write to Kafka at sub-millisecond latencies
- All available with Databricks Runtime!

databricks

Summary

- File listing and many small files hurt performance
  - Using Delta and event based notification sources help us avoid listing
  - Delta's in-built compaction and indexing alleviates small file problem
- HiveMetaStore can become the bottleneck for large tables
- Partial / distributed failures can taint tables
- Schema Management and Data Hygiene are hard problems
- GDPR adds extra complexity to pipelines through DSRs

Summary

- File listing and many small files hurt performance
- HiveMetaStore can become the bottleneck for large tables
  - Delta uses Spark jobs to manage its metadata to scale to billions of files
  - Delta auto-updates => No need to call REFRESH TABLE with Spark
  - No need to add/remove partitions, no need for MSCK REPAIR TABLE
- Partial / distributed failures can taint tables
- Schema Management and Data Hygiene are hard problems
- GDPR adds extra complexity to pipelines through DSRs

Summary

- File listing and many small files hurt performance
- HiveMetaStore can become the bottleneck for large tables
- Partial / distributed failures can taint tables
  - Delta's ACID transactions guard us against garbage data
  - Always get a consistent (possibly stale) view of your table with Delta
- Schema Management and Data Hygiene are hard problems
- GDPR adds extra complexity to pipelines through DSRs

Summary

- File listing and many small files hurt performance
- HiveMetaStore can become the bottleneck for large tables
- Partial / distributed failures can taint tables
- Schema Management and Data Hygiene are hard problems
  - Delta has in-built schema management to only allow safe changes
  - Invariants in Delta prevent unexpected data from polluting tables
  - Delta architecture (Bronze-Silver-Gold tables) combined with Delta makes backfills and corrections easier
  - Delta's upcoming support for rollbacks will make corrections effortless
- GDPR adds extra complexity to pipelines through DSRs

databricks

Summary

- File listing and many small files hurt performance
- HiveMetaStore can become the bottleneck for large tables
- Partial / distributed failures can taint tables
- Schema Management and Data Hygiene are hard problems
- GDPR adds extra complexity to pipelines through DSRs
  - UPDATE / DELETE support in Delta makes this easier

Further Reading

- On Structured Streaming
  - https://databricks.com/blog/2017/08/24/anthology-of-technical-assets-on-apache-sparks-structured-streaming.html

- On Delta:
  - https://databricks.com/blog/2017/10/25/databricks-delta-a-unified-management-system-for-real-time-big-data.html
  - https://databricks.com/blog/2018/07/31/processing-petabytes-of-data-in-seconds-with-databricks-delta.html
  - https://databricks.com/blog/2018/09/10/building-the-fastest-dnaseq-pipeline-at-scale.html
  - https://databricks.com/blog/2018/07/19/simplify-streaming-stock-data-analysis-using-databricks-delta.html
  - https://databricks.com/blog/2018/07/02/build-a-mobile-gaming-events-data-pipeline-with-databricks-delta.html

databricks

Thank You

“Do you have any questions for my prepared answers?”

– Henry Kissinger

databricks