Analytics Pipeline at Lyft

Shenghu Yang | March 2018

Agenda

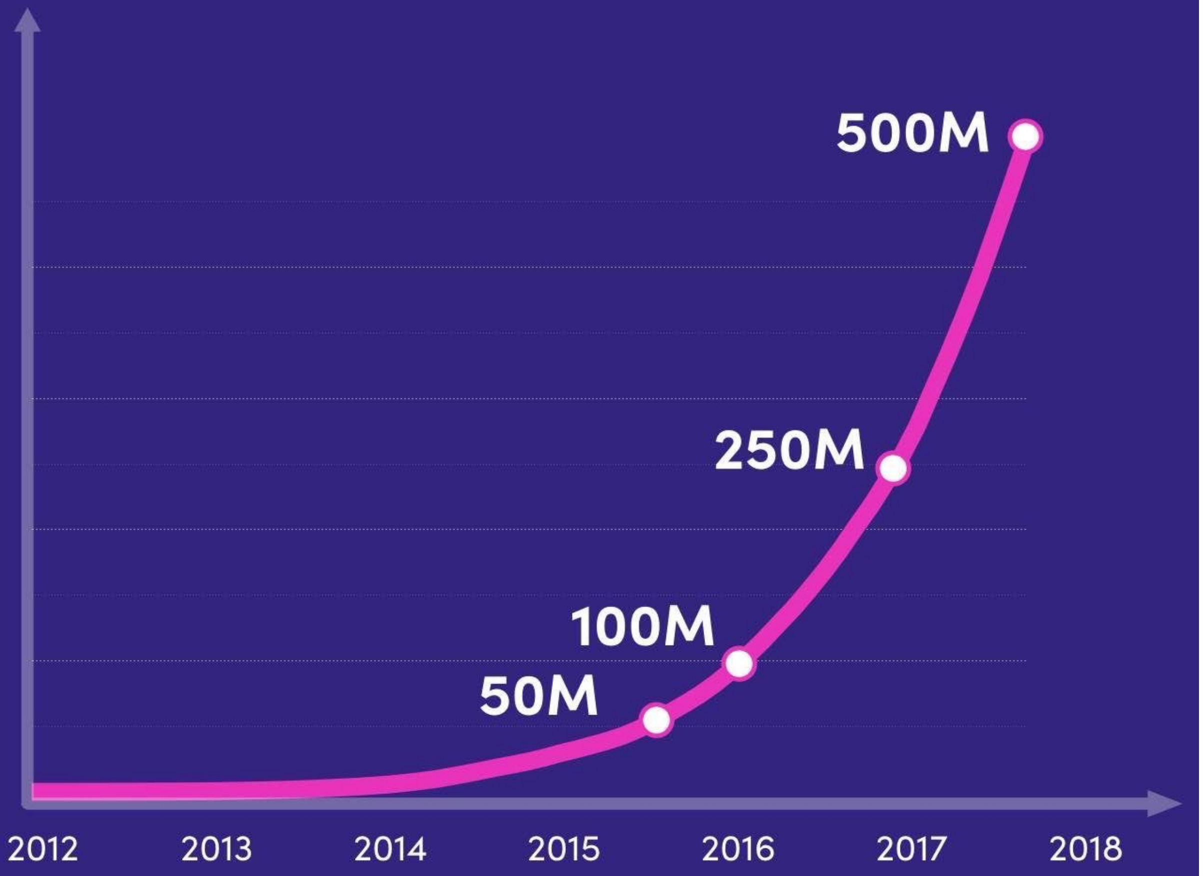
- Lyft at a glance
- Lyft analytics data audience
- How Lyft analytics pipeline evolved
- Principles & challenges
- What we solved
- What's next

Lyft at a glance

Lyft at a glance

- **Mission**
  - “Improve people's lives with the world's best transportation”
- **600+** cities
- **95%** US population
- Growing fast

RIDE GROWTH



Lyft Analytics Data Audience

Lyft analytics data audience

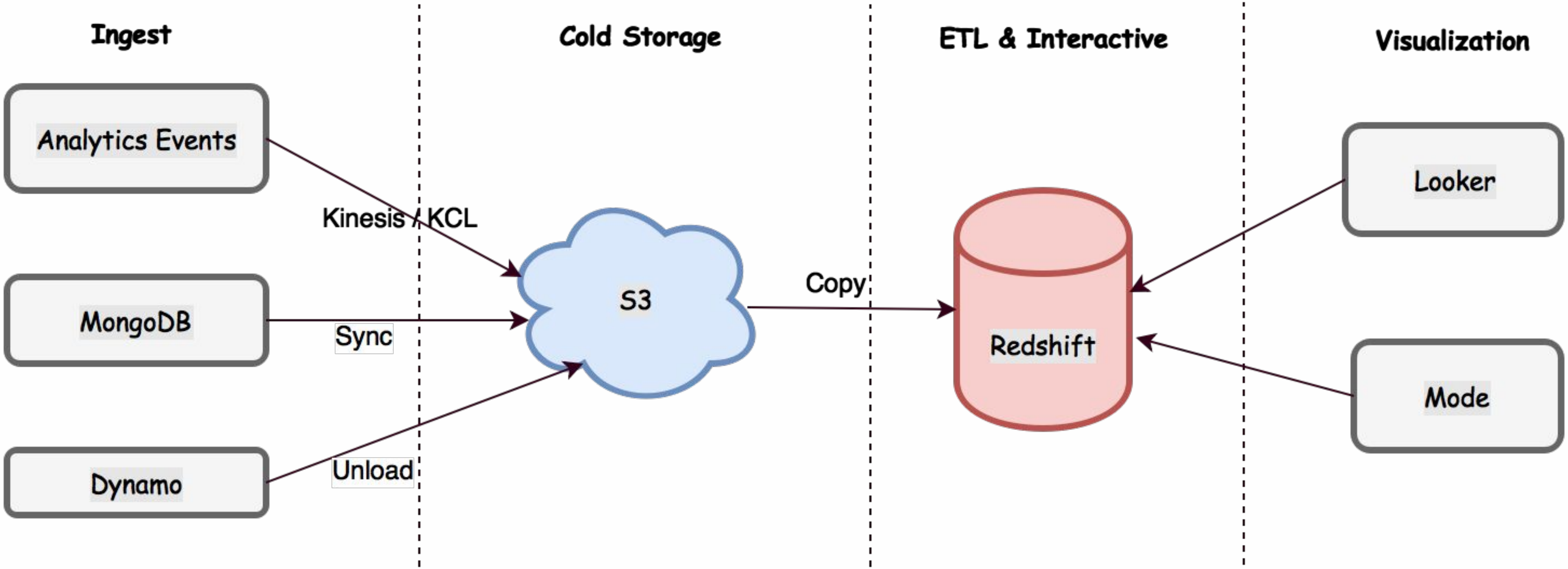
- Growth
  - driver acquisition & engagement, passenger activation & retention
- City team - Ops
  - market health, local marketing
- Data Science / Analytics
  - rides, conversion, driver hours, finance, marketing
- Engineering / Product / Design
  - fraud, ETA, pricing, routing, feature design
- Experimentation Platform
  - A/B testing

How Lyft Analytics Pipeline Evolved

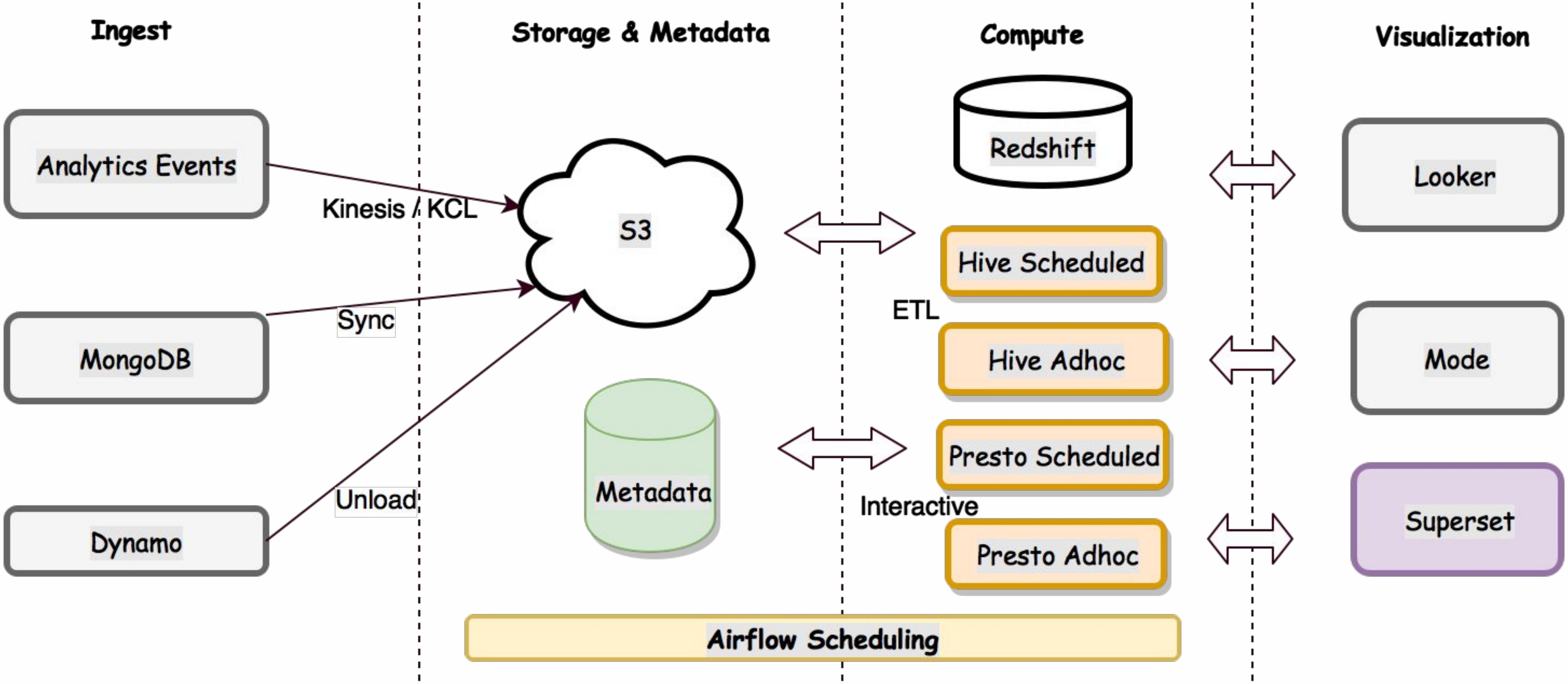
How analytics pipeline evolved

- 2015 - Redshift, Kinesis, MongoDB, Dynamo
- 2016 - Hive, Spark, Airflow
- 2017 - Presto, Kafka, Flink
- <u>2018 - Druid, Superset</u>

Once upon a time ...



Current in Production



Quick Stats

Data volume:

- 20PB Warehouse
- 3B+ events / day

Query stats:

- Hive - 60k / day
- Presto - 20k / day
- Redshift - 40k / day

Principles & Challenges

Principles:

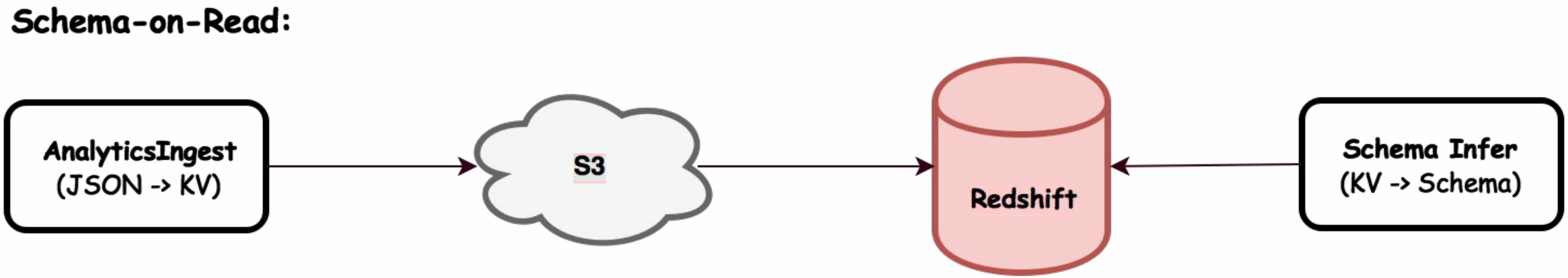
- Keep business up & running fast
- Forward looking

Challenges:

- Schema Management
- Operation vs. Performance
- Backfill Orchestration
- Data Replication
- User Expectation / Onboarding

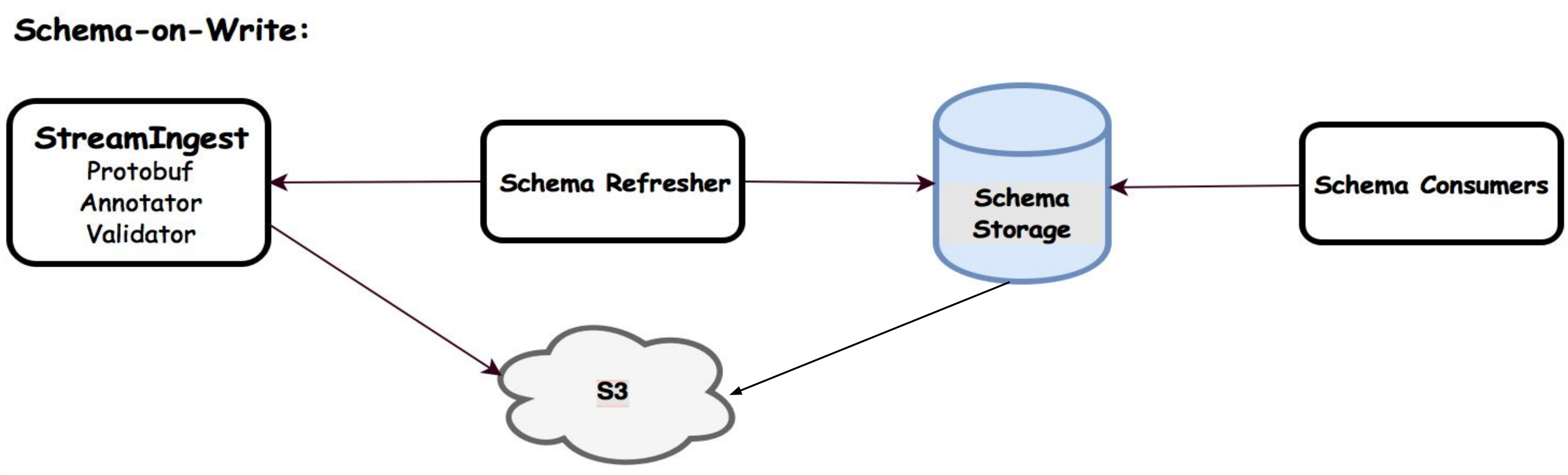
What we Solved

1. Schema Management - early days



- Very flexible (KV)
- Hard to scale
- No clear contract between producers & consumers
- No backward compatibility - data breakage

Schema Management - 2017



Schema Management - 2017

- Centralized schema
  - decouple producers and consumers
  - direct single source of truth
- Schema evolution - backward compatible
  - no removing field
  - no renaming field
  - no changing existing type
  - append only
- Support Parquet/Snappy for storage
  - 2-3X faster than json / gzip
  - 60% storage saved than KV

2. Operation vs. Performance

- We choose S3 over HDFS as our storage layer for operation and cost
- Pros:
  - Decouple compute with storage - instant new cluster launching
  - Capacity planning like a breeze
  - SLA (99.99% availability & 99.999999999% durability)
  - Backup and disaster recovery
  - Cost (compute node auto scaling & spot market)
- Cons:
  - Performance (less data locality)
  - Eventual consistency
  - No object renaming

Mitigation Plan

- s3a vs. s3n
- S3 bucket/prefix structure & file size (e.g 256M)
- Deep copy for regular ETL
  - s3 read-after-write consistency for new PUTs
  - on new partition/s
- Shallow copy for backfill jobs
  - s3 overwrite PUTs is eventual consistent
  - shallow copy eliminates the eventual consistency
  - no need to rename object

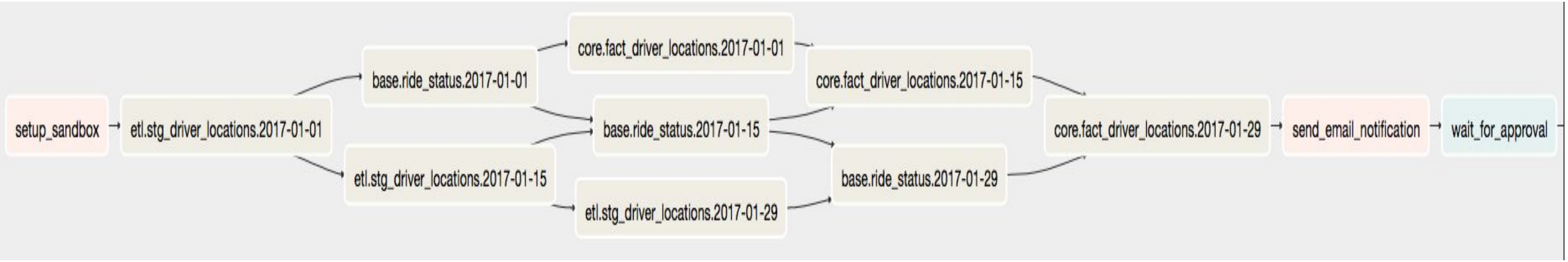
3. Backfill Orchestration

Pain points:

- Engineering hours / context switch
- Chance to break production
- Costly (time & money)

Backfill Orchestration Tool & Airflow DAG

```
[TARS] $ service_venv /srv/service/current/bin/hive_etl rebuild create \
                --dryrun \
                --start 2017-01-01 --end 2017-01-31 \
                --table etl.stg_driver_locations \
                --table base.ride_status \
                --table core.fact_driver_locations \
                --ds_step 14 | less
```



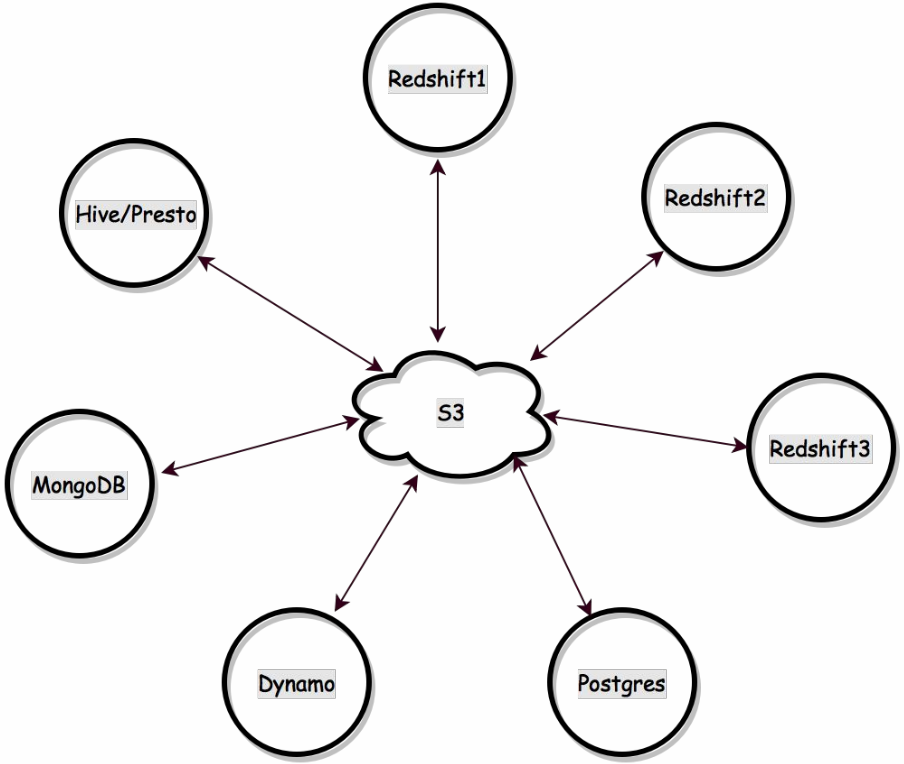
Backfill orchestration tool - results

- 12-18X gain on engineering productivity
- Chance to perform QA **before** promoting to prod
- 2-3X infra cost saving & speedup in wall clock time

4. Data Replication

Challenges:

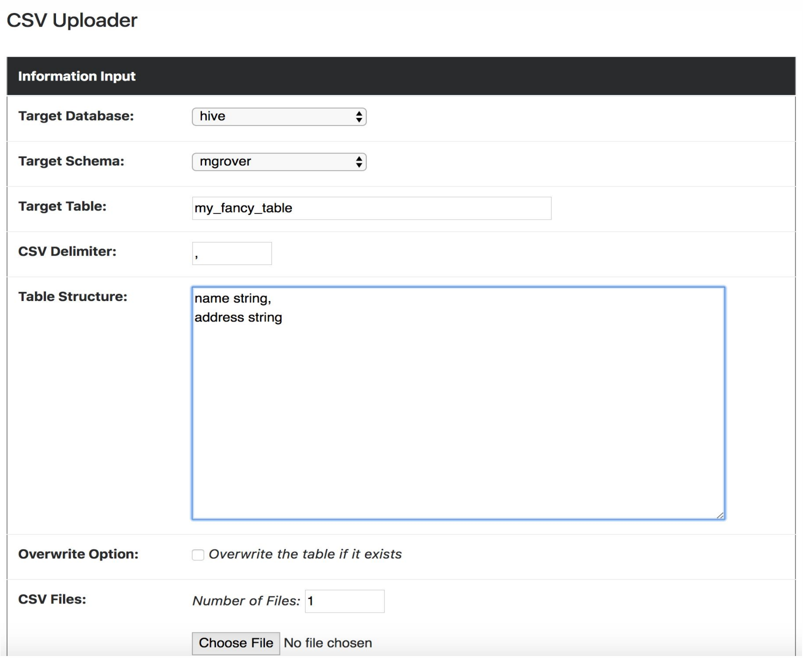
- Many databases
- Frequent schema changes
- Data truncation & append
- No SLAs

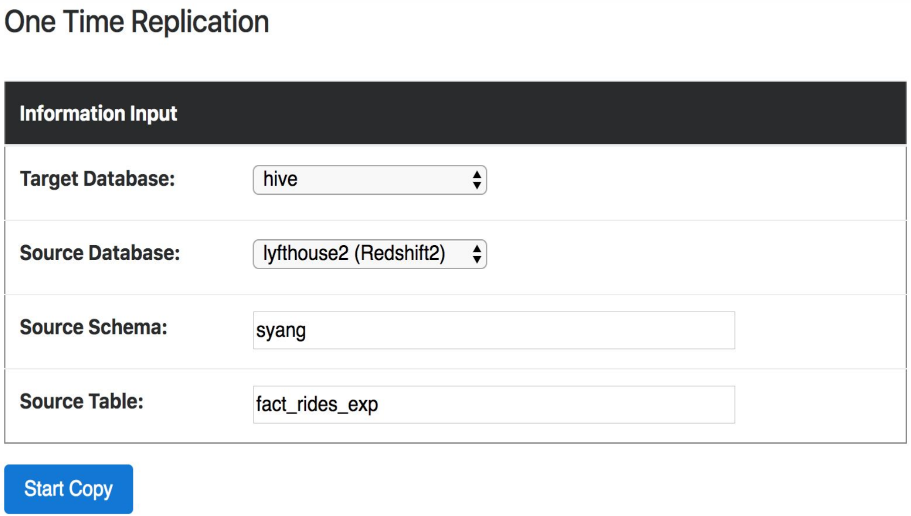


Replication - Scheduled Runs

		DAG	Schedule	Owner	Recent Tasks	Last Run	DAG Runs	Links
	On	financial_replication_d2r	0 2 * * *	data-platform	40	2018-03-04 02:00	276	
	On	financial_replication_d2r_fpna	0 10 * * 2	data-platform	21	2018-02-20 10:00	39	
	On	financial_replication_m2r	0 13 * * *	data-platform	20	2018-03-04 13:00	275	
	On	redshift3_replication_d2r	0 4 * * *	data-platform	59	2018-03-04 04:00	151	
	On	redshift3_replication_m2r	0 14 * * *	data-platform	29	2018-03-04 14:00	148	
	On	redshift_replication_driver_engagement	0 15 * * *	data-platform		2018-02-06 15:00	2	

Replication - self-service tools





5. User Expectation & Onboarding

User concerns / questions:

- Hive query is much slower than Redshift
- Hive ETL dev productivity is lower - lack of UDFs, tools, doc etc.
- Part of required data is not live in Hive/Presto
- Hive/Presto clusters is less available than Redshift
- When to use Hive vs. Presto?
- I really need something urgently, can I use Redshift?
- I am new to HiveQL/Presto query

User Expectation & Onboarding

Our answers:

- **Performance**: Use Presto for interactive query
- **Productivity**: We provide similar UDFs, dev tools and docs for best practices and gaps
- **Data availability**: We provide backfill tool, one time copy tool and csv uploader
- **Uptime**: We are striving to provide the same SLA, and the gap is shrinking
- **Hive vs. Presto**: Hive for big batch ETL, Presto for smaller adhoc query (<1TB)
- **One time exception**: We will examine it case by case
- **Newbie**: Data bootcamp (101: SQL, BI tools; 201: Hive, Presto, Event logging)

What’s Next

What’s Next

- Geospatial: Druid & Superset
- Streaming platform: Kinesis/KCL -> Kafka/Flink
- Further scale Presto / Hive
- Query Federation Proxy
  - Kill “bad” query
  - Forward query to right cluster
  - Convert long Presto query to hive (investigating)
- Better logging / schema service
- Move more queries off Redshift
- Open source: Airflow / Superset

Thank you!

Shenghu Yang