

Data Pipeline HealthCheck for Correctness, Performance, & Cost Efficiency

Shivnath Babu CTO/Cofounder @ Unravel Adjunct Professor @ Duke University

TRUSTED BY





♥aetna"

Adobe 🗘 TIAA 🗹 Deutsche Bank Humana.

About the speaker



Shivnath Babu

Cofounder/CTO at Unravel

Adjunct Professor of Computer Science at Duke University Focusing on manageability of data pipelines and the modern data stack

Recipient of US National Science Foundation CAREER Award, IBM Faculty Award, HP Labs Innovation Research Award

Unravel radically simplifies DataOps & has strong adoption across platforms & industries

uncover

 Brings together information about all your apps, clusters, resource utilization, users, & datasets in a single place

DBS Isbank 24.51

understand

- Creates end-to-end view of data pipelines to easily track & understand issues
- Tracks & reports on usage across environments

Checks for & alerts on anomalous behavior

unravel

- Uses AI/ML to troubleshoot & optimize apps to meet desired performance & cost needs
- Spots & fixes inefficient usage
- Ensures efficiency, quality, & performance of all apps in development & production

A citibank WELLS

Every company is now a data company (or trying to become one!)



unravel





Data pipelines are complex

How can you keep them healthy?

Keeping data pipelines healthy



Aspects of Health

Correctness

Performance

Cost

Unhealthy Behavior

Wrong result, Pipeline failure Missed SLA, Growing lag/backlog Cost overrun, Going over budget

unravel

Rest of this talk

- HealthCheck for Data Pipeline Correctness
- HealthCheck for Data Pipeline Performance
- HealthCheck for Data Pipeline Cost
- Demo

HealthCheck for Data Pipeline Correctness



HealthCheck for data pipeline correctness

Example Checks at Data-level

- Daily partitions of Table "SignupsAndSubs" should have at least 1000 records
- "customerPinNumber" should not be NULL
- Feature "age" should be normally distributed

Example Checks at App-level

- At least one of the evaluation rules should fire
- Only nodes marked valid in the decision tree should be reached
- Stage S2 should not start before Stage S1 has finished

HealthCheck for data pipeline correctness

User-specified Checks

- Lot of context and application semantics are involved in defining correctness
- One man's food is another man's poison
- Best-practice: Tools like *Great Expectations* make it easy to define checks

Automatic Checks

- False negatives can arise if the user didn't specify all the checks
- Changes and anomalies can be captured automatically
- Balancing false positives Vs. false negatives remains an art today

HealthCheck for data pipeline correctness

Check Execution and Automated Actions

- Crucial to execute checks at the right time and in the right order
- Best-practice: Design checks as a first-class citizen in the pipeline

Tracking, Troubleshooting, and Tuning when Checks Fail

- Must capture failed checks in the context of the pipeline execution since lineage is important for root cause analysis
- Capturing history of pipeline runs is key to understand "what changed"

HealthCheck for Data Pipeline Performance



HealthCheck for data pipeline performance

Example Checks for Batch Pipelines

- Pipeline should finish by 6:00 AM PST
- Data in dashboard should not be older than 10 mins

Example Checks for Streaming Pipelines

- Latency of processing messages from Kafka topic should less than 10ms
- Lag should not exceed 10000 messages

HealthCheck for data pipeline performance

User-specified Checks

- Less context & application semantics needed compared to correctness checks
- Best-practice: Define end-to-end pipeline SLAs
- Best-practice: In Airflow, specify the maximum time a task can take

Automatic Checks

- Often, SLAs are implicit and may not be specified
- Building appropriate baselines and detecting deviations are critical

HealthCheck for data pipeline performance

Check Execution and Automated Actions

- Early warnings are key, it may be too late once end-to-end SLA is missed
- Best-practice: Keep pipeline stages short and frequent

Tracking, Troubleshooting, and Tuning when Checks Fail

- Single-pane-of-glass is very important since pipelines can be complex, with many moving parts
- Unrelated apps may affect each other due to multi-tenancy
- Automated insights to remediate the problems are invaluable

HealthCheck for Data Pipeline Cost



HealthCheck for data pipeline cost

Example Checks

- Cost of any one run of the "BI-report" pipeline should not exceed \$100
- Budget for the pipelines generating the "probable_churn" table is \$1M/month

HealthCheck for data pipeline cost

User-specified Checks

- Less context & application semantics needed compared to correctness checks
- More and more Data leaders are being compelled to create these checks

Automatic Checks

• Huge opportunities for automatic cost-inefficiency checks due to the complexity of data pipelines and the black-box nature of the cloud

HealthCheck for data pipeline cost

Check Execution and Automated Actions

- Early warnings are key since cost incurred will not be refunded
- Budget overruns can have severe consequences

Tracking, Troubleshooting, and Tuning when Checks Fail

- Single-pane-of-glass and detailed cost breakdown are must-haves
- Automated insights to remediate the problems are critical

Let us see it in action



Sign up for a free trial!

https://unraveldata.com/saas-free-trial

shivnath@unraveldata.com

Migrating pipelines to the cloud? Check out our talk: Lessons Learned while Migrating Data Pipelines from Enterprise Schedulers to Airflow