From Cron to Apache Airflow

A Startup Story

Adam Boscarino, 2020-07-13
Who am I?

- Data Engineer at Devoted Health
- Previously worked at DigitalOcean, Fitbit, Carbonite
- Airflow user for ~4 years
- GitHub & Twitter: ajbosco
What is Devoted Health?

- A next generation Medicare Advantage health insurer in the United States
- Founded in 2018, first enrolled members in 2019
- Offers a Clinical Services solution (Devoted Medical Group)
- Built on homegrown Devoted Tech Platform

“TO DRAMATICALLY IMPROVE HEALTHCARE FOR SENIORS IN THE UNITED STATES -- CARING FOR EVERYONE LIKE THEY ARE MEMBERS OF OUR OWN FAMILY”
### Devoted Health Data Platform, January 2019

<table>
<thead>
<tr>
<th>Source Data</th>
<th>Workflows</th>
<th>Storage/Data Lake</th>
<th>Data Warehouse</th>
<th>Reporting/BI</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Source Data" /></td>
<td><code>[^~]$ crontab</code></td>
<td><img src="image2.png" alt="Storage/Data Lake" /></td>
<td><img src="image3.png" alt="Data Warehouse" /></td>
<td><img src="image4.png" alt="Reporting/BI" /></td>
</tr>
</tbody>
</table>

- **Source Data**: Visual representation of data sources.
- **Workflows**: `[^~]$ crontab` represents a command-line interface for setting up scheduled tasks.
- **Storage/Data Lake**: ![Storage/Data Lake](image2.png) showing Amazon S3.
- **Data Warehouse**: ![Data Warehouse](image3.png) showing Amazon Redshift.
- **Reporting/BI**: ![Reporting/BI](image4.png) with Periscope Data by Sisense.
Devoted Health Data Platform - Successes

- It did its job
- Successfully launched new health plan
- Supported key business operations and workstreams
- Powered all internal reporting
Devoted Health Data Platform - Problems

- No task dependencies
- Undetected system downtime
- Onboarding new developers
- Environment parity
- Unsure of data quality
Devoted Health Data Platform, May 2019

Source Data

Workflows

Storage/Data Lake

Amazon S3

Data Warehouse

Data Warehouse

Reporting/BI

NEW!

NEW!

Periscope Data
by Sisense
Devoted Health Data Platform - Problems

- No task dependencies
- Undetected system downtime
- Onboarding new developers
- Environment parity
- Unsure of data quality
Solution: Deploying Airflow
Apache Airflow Deployment

- **Kubernetes**
  - Orchestrates Airflow services
  - Kubernetes Executor
- **Helm**
  - Kubernetes Package Manager
  - Describes Kubernetes resources
  - Official Helm Chart
- **Terraform**
  - Infrastructure as Code
  - Used to deploy Helm chart to Kubernetes clusters
DAG Deployment

- DAGs are stored on AWS EFS
  - Mounted to each Airflow pod in Kubernetes
- DAGs are pushed from GitHub to AWS EFS via CircleCI
  - No manual intervention
  - Many deployments every single day
Devoted Health Data Platform - Problems

- No task dependencies
- **Undetected system downtime**
- Onboarding new developers
- Environment parity
- Unsure of data quality
Solution: Monitoring Airflow
Monitoring Airflow

- Kubernetes Liveness & Health Checks
  - Monitor /health endpoint
  - Monitor Scheduler health
  - Restart services if in bad state

- Datadog Monitors
  - Alert on-call engineer via OpsGenie and Slack
  - Airflow is not running
  - No DAGs have completed in last 2 hours
  - CPU/Memory Usage has spiked
Monitoring DAGs

- OpsGenie alerts sent to DAG Owner (and Slack)
- DAG owners are responsible for resolving non-infrastructure failures
- Alerting is “built-in” to every DAG
Devoted Health Data Platform - Problems

- No dependency management
- Undetected system downtime
- **Onboarding new developers**
- Environment parity
- Unsure of data quality
Solution: YAML DAG Builder
YAML DAG Builder

- Internal library to simplify and standardize DAG development.
- Abstraction on top of Airflow.
- Developers only write a DDL query and transformation in SQL or Python.
- No prior Airflow knowledge required.
- Similar structure in all DAGs makes switching teams less painful and simplifies debugging DAGs.
- Data Engineer team can bolt on additional features (alerting, monitoring, testing, etc.)
Devoted Health Data Platform - Problems

- No task dependencies
- Undetected system downtime
- Onboarding new developers
- Environment parity
- Unsure of data quality
Solution: devflow
devflow

- Internal tool that wraps `kubectl`, `Helm`, and `Terraform`.
- Every developer gets their own Airflow deployment on Kubernetes.
- We develop on the same stack that we run production.
- Developers do not need to know anything about the infrastructure being used.
Devoted Health Data Platform - Problems

- No task dependencies
- Undetected system downtime
- Onboarding new developers
- Environment parity
- Unsure of data quality
Solution: Testing & Validation
Testing DAGs

- **Unit Tests**
  - Used on Python transformations and core library code

- **Integration Tests**
  - Used for SQL tasks
  - Internal framework built on pytest
  - Executed against Snowflake using a test database
  - Mock tables are created and populated
Data Validation Framework

- Data validation is executed at DAG run-time
- DAGs are stopped if validation fails to prevent reporting on bad data
- Started with Check Operator
- Added internal Operators
  - Runs multiple checks with one task
  - Save invalid records to table
  - Send check values to Datadog
- Checks range from primary key validation to custom business logic

```python
table_a_pk_validation:
  config_type: ValidateTask
  validation_type: 'uniqueness'
  validation_table: table_a
  validation_columns:
    - id
deps:
      - table_a_populated

table_a_count_validation:
  config_type: ValidateTask
  validation_type: 'custom'
  validation: 'validations/table_a_count_validation.sql'
  operation: '='
  pass_value: 0
deps:
  - table_a_populated

table_c_quality_checks:
  config_type: QualityCheckTask
  description: 'Runs all data quality checks for table C.'
  quality_checks: 'validations/table_c_validation.py'
deps:
  - table_c_populated
```
Mission Accomplished!
Current Issues & Future Work

- Improve SQL testing!
  - Explore tools like dbt and dataform
  - Remove need for end user to know pytest
- Improve DAG Builder
  - Make standard use cases easier
- SQL Linting/Formatting
  - Enforce best practices programmatically
- KEDA Autoscaler
  - Improve task spin-up speed
Questions?