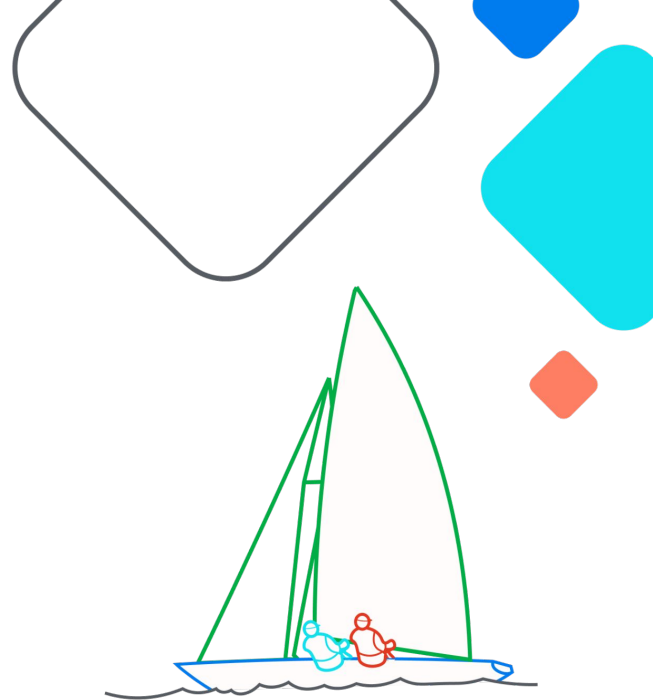


Elevating Data Quality: Great Expectations and Airflow at PepsiCo

Russell Lamb



 **Airflow Summit**

Let's flow together

September 19-21, 2023,
Toronto, Canada

Introduction

WHO AM I?



Name

Russell Lamb



Role

Retail Data Engineering Lead in PepsiCo's eCommerce



Mission

Deliver retailer data to drive value for eCommerce



Based in

Me: New York - Team: Asia, Europe, USA



Favorite PepsiCo Products

SodaStream + Cool Ranch Doritos



Why do I love Airflow?

Flexible, scalable, extensible



PepsiCo: Global Presence & Powerful Portfolio

BEVERAGES



FOODS & SNACKS



Performance



More than \$79 billion net revenue in 2021

Brands



23 billion-dollar brands

Scale



More than 200 countries & territories

Purpose



Creating Growth & Value

Consumers Get our Products via 3 Main Channels



Retail

Consumers buy our products to consume at home. Many types of Retailers, offering a wide variety of brands and flavors.



Foodservice

Consumers buy our products outside of a retail location to consume on site. Usually our brands are exclusive at that site.



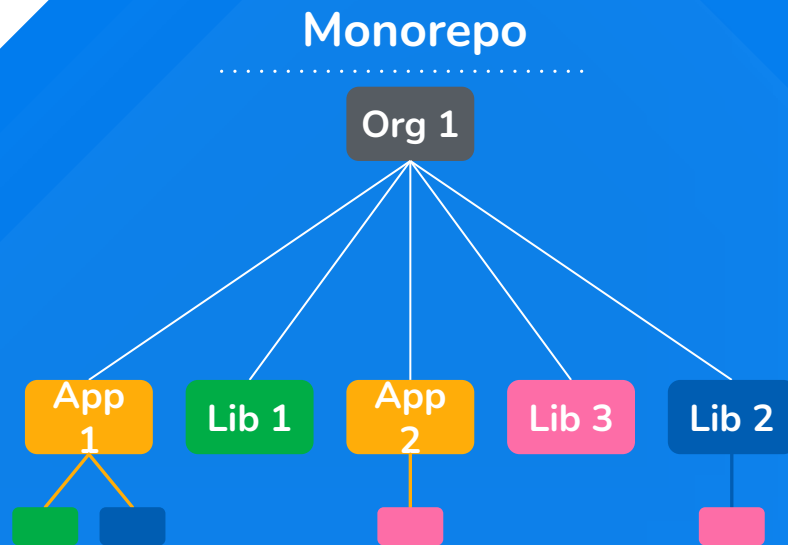
Ecommerce

Consumers buy our products via online merchants to have either delivered to their home or pick-up in store.

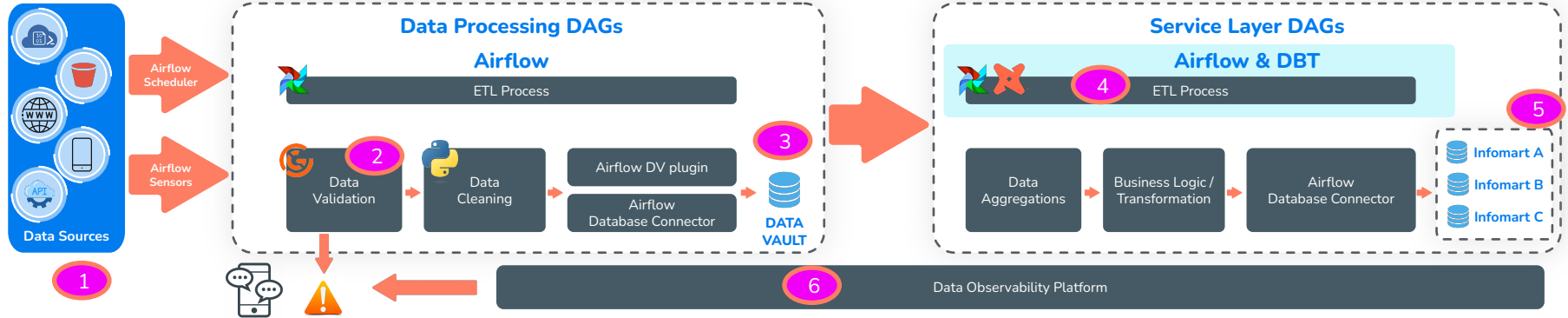


Airflow Usage

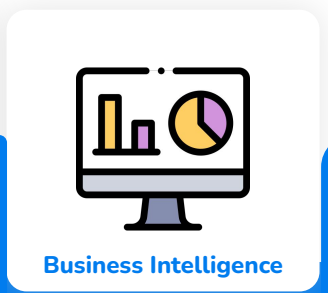
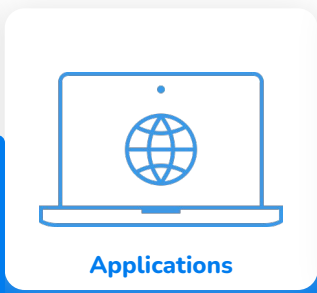
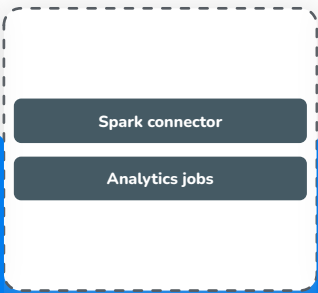
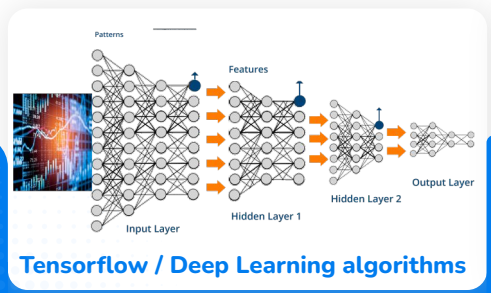
- **800+ active Production Airflow pipelines**
 - Self-hosted Airflow on managed Kubernetes
 - Multi-cloud: AWS and Azure
 - Dedicated data platform team in-house
- **50+ Data Sources**
 - Mostly customers, some 3rd party providers
 - Multiple data types & formats
- **Multi-team Airflow instance**
 - Tag Airflow jobs by team
 - Mono-repo: Shared code repository with several teams
- **Decentralized Development**
 - Data Vault Airflow plug-in
 - Distributed ownership by domain
 - Collaborative governance
 - Shared data models



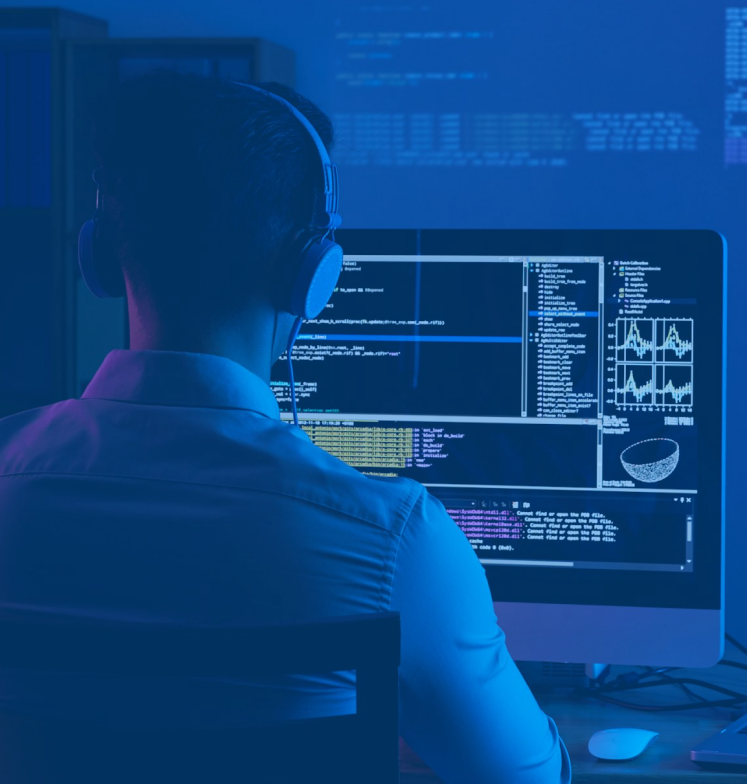
Modern Data Platform & Architecture



Data Warehouse



Data Quality – Challenges



Schema Changes

- Monitoring needed for consistency
- Data is 2nd and 3rd party



Bad Data Hard to Remove

- Requires changes to production
- Custom logic used once



Docs for Data

- Capturing the business context
- What does good data look like?



Errors

- Missing Data
- Duplicates

Our Solution

- **Great Expectations**

- Version 0.13, July 2020
- 300+ Expectation suites
- Soft fail vs Hard fail

- **Custom Code**

- Expectations Operator for Cloud Storage and Database
- Action class for Slack and Rendering
- Action for Data Docs Rendering

- **Why**

- More notification options, e.g. multi-channel
- Handling Authentication, especially on Azure

- **Many options “out of the box”**

- GreatExpectationsOperator
- Webhook for Slack
- Use existing data docs

committed on Jul 31, 2020 Verified

```
✓ great_expectations
  > checkpoints
  > expectations
  > plugins
  > uncommitted
  README.md
  __init__.py
  builder.py
  example_config_variables.yml
  example_validation.json
  failed_expectation_decorator.py
  great_expectations.yml
  operators.py
```


What is Great Expectations?

Great Expectations

- Python library
- Open Source

Components

- Web App: Data Docs
- Validation Engine
- Notifications via Webhook

Integrates with Airflow

- Validation step in pipeline
- Open Source Operator
- Custom Operator

Overview
Expectation Suite: default
Data asset: None
Status: ✔ Succeeded

Statistics

Evaluated Expectations	2
Successful Expectations	2
Unsuccessful Expectations	0
Success Percent	100%

Show more info...

passenger_count

Status	Expectation	Observed Value
✔	values must be greater than or equal to <input type="text"/> and less than or equal to <input type="text"/> .	0% unexpected

pickup_datetime

Status	Expectation	Observed Value
✔	values must never be null.	100% not null

```
class CloudStorageExpectationOperator(PepExpectationOperator):  
    """Airflow Operator for validating data stored in cloud storage (S3, ABS)."""  
    """Create a new CloudStorageExpectationOperator.  
  
    Use this operator to run Great Expectations suites against data stored  
    in a cloud storage provider (AWS S3, Azure blob storage).  
  
    data_source_name - Data source from great_expectations.yml  
    data_asset_name - Path to data within the bucket/container  
    expectation_suite_name - Expectation suite to run  
    hard_fail - If false, validation failures do not cause the validation  
                task to fail. If true, validation failures will throw an exception  
                that fails the task with no further retries.  
    evaluation_parameters - Runtime parameters to pass to the validation run  
    """
```

Three-Tiered Approach to Quality

Inbound
Data

Transforming
Data

Warehoused
Data



Great
Expectations



Data Build
Tool



Data
Observability

Why Data Quality is Important

- Investing in data quality systems allows us to focus on building new capabilities
- Gartner: Bad data costs companies an average of \$12.9 Million per year ¹
- IBM: Bad data costs the US \$3.1 trillion ²

Save Time and Resources

Confidence in Data

- Sales decisions
- Marketing Spend
- Inventory replenishment

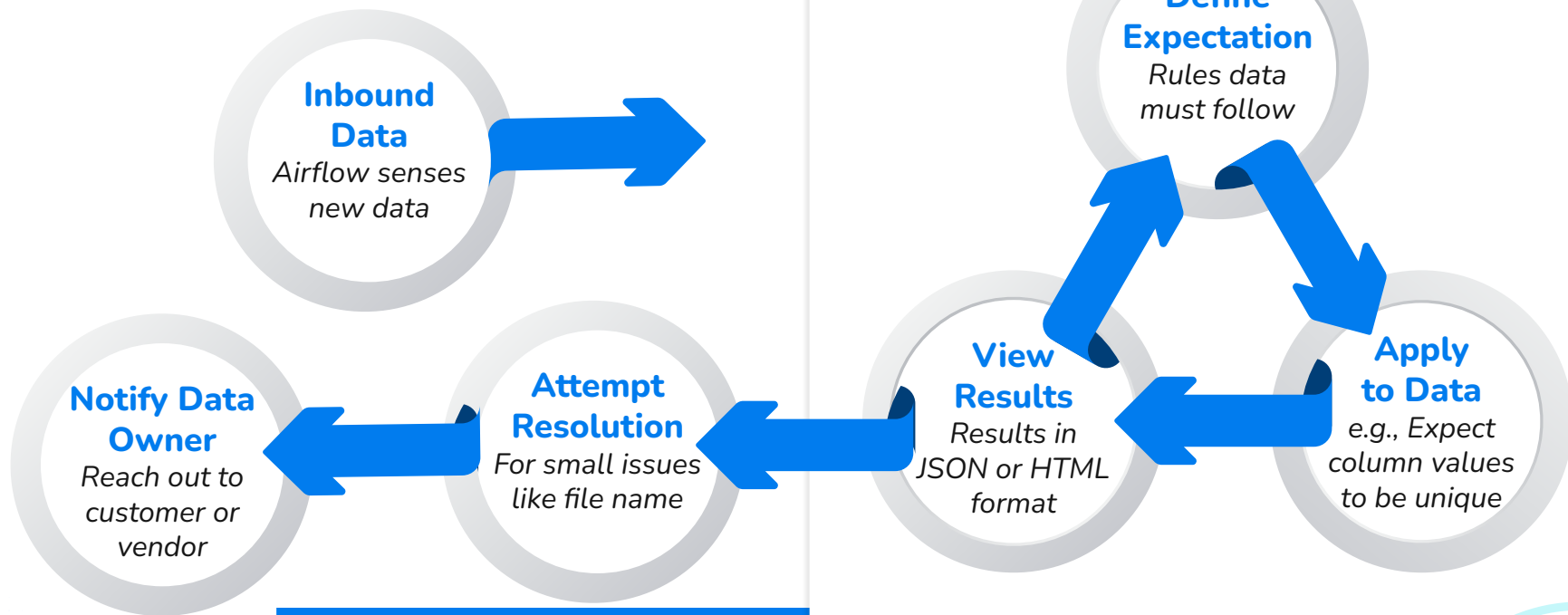
Variety of Providers, Sources, and Use Cases

- Logic of each must be documented
- Documentation should change with code, not in a separate document to maintain
- Good documentation allows engineers to build off the efforts of their peers

1. Tech Target – data quality - <https://www.techtarget.com/searchdatamanagement/definition/data-quality>

2. Harvard Business Review Bad Data Costs the U.S. \$3 Trillion Per Year- <https://hbr.org/2016/09/bad-data-costs-the-u-s-3-trillion-per-year>

Validation Lifecycle





Running in Production

Slack Alert notifies Data team of failed validation



Data analyst clicks link to Great Expectations “Data Docs” site



Data Docs contains readable results of the evaluation



Check the Airflow logs to verify the error

APP 4:17 PM
Batch Validation Status: Failed ❌
Expectation suite name: [redacted]
Summary: 4 of 7 expectations were met
DataDocs can be found here: [redacted]

Status	Expectation
❌	Must have at least these columns (in any order): <code>asin</code> , <code>program</code> , <code>date_shipped</code> , <code>fulfillment_center</code> , <code>item_name</code> , <code>merchant_brand_name</code> , <code>shipped_units</code> , <code>shipped_cogs</code> , <code>shipped_ops</code>
	Values for given compound columns must be unique together: <code>asin</code> , <code>program</code> , <code>date_shipped</code> , <code>fulfillment_center</code>
	<code>expect_compound_columns_to_be_unique</code> raised an exception: <code>KeyError: '['program'] not in index"</code>

```
"result": {
  "observed_value": [
    "asin",
    "date_shipped",
    "fulfillment_center",
    "item_name",
    "merchant_brand_name",
    "shipped_cogs",
    "shipped_ops",
    "shipped_units"
  ],
  "details": {
    "mismatched": {
      "missing": [
        "program"
      ]
    }
  }
}
```

Benefits of Great Expectations



Document Assumptions

- Gives clarity to non-Subject matter experts
- Which columns can be null?
- What's a valid value?



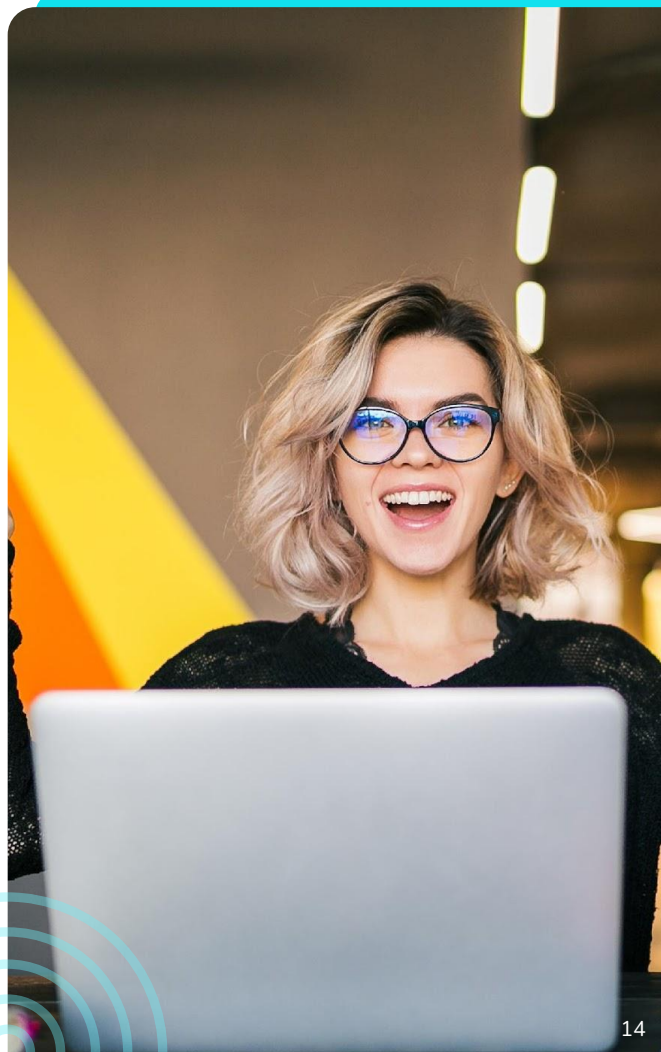
Streamline Support

- Find the reason for failure quickly
- Embed Great Expectations “Data docs” links in Airflow log & Slack



Keep Bad Data Out

- Stop pipelines before loading data



Questions?



Find me on LinkedIn

www.linkedin.com/in/russellamb