VINUNIVERSITY

Data Observ-ability & Lineage

AICB-P2T2 · Ngày 27 · Chương 6: Tổng Hợp

Giảng viên

VinUniversity · Phase 2 · Track 2 · Tuần 6

HÃY SUY NGHĨ...

"Pipeline chạy thành công nhưng data sai — làm sao bạn biết? ***Case study:*** *Một team phát hiện model accuracy giảm 15% — sau 3 ngày mới biết up-stream data bị schema change. Data observability phát hiện trong 60 giây."*

Giữ câu hỏi này trong đầu suốt buổi học hôm nay

Nội Dung Bài Học

1. Data Observability vs Pipeline Monitoring
2. Great Expectations: Suites & Checkpoints
3. Monte Carlo & Anomaly Detection
4. dbt Tests: Unit & Integration
5. SLO Engineering cho Data & AI
6. Incident Response cho Data Systems
7. Live Demo: Incident Detection
8. Labs: Data Observability Implementation

Mục Tiêu

Sau buổi học này, bạn sẽ:

1. Master data observability với Great Expectations + Monte Carlo
2. Thiết kế advanced Grafana dashboards cho data quality
3. Implement SLO engineering cho data & AI services
4. Xây dựng incident response workflow cho data incidents

> Data observability concepts (20 min) → Great Expectations (45 min) → Monte Carlo / dbt tests (30 min) → SLOs (30 min) → Demo & Labs

Deliverable Cuối Ngày

GE checkpoint suite + Monte Carlo-style anomaly detection + SLO dashboard

- Great Expectations Checkpoint chạy trong Airflow DAG
- Z-score anomaly detection script với Slack alert
- 3 SLOs cho data platform + Grafana dashboard với error budget panel
- Incident response runbook document

Pipeline Monitoring vs Data Observability

Infrastructure view:

- Is the job running?
- How long did it take?
- Did it error out?
- Resource usage (CPU, memory)

→ **"Pipeline succeeded"** nhưng data có thể sai!

Data quality view:

- Is data **correct**?
- Is data **fresh**?
- Is data **complete**?
- Is data **consistent**?

→ **Detect silent data issues** trước khi impact downstream

Data downtime: trung bình 10+ giờ/tuần cho data teams — observability giảm 80%.

Cost of bad data: Gartner estimate $15M/năm trung bình — đầu tư data quality có ROI rõ ràng.

Impact gì?;

Hầu hết teams ở Level 0–1. Mục tiêu khoá này: đạt Level 2 (anomaly detection) và hiểu roadmap lên Level 3–4.

auto-remediation; [-Stealth, thick, gray] (l0) – (l1); [-Stealth, thick, gray] (l1) – (l2); [-Stealth, thick, gray] (l2) – (l3); [-Stealth, thick, gray] (l3) – (l4);

Observability cho Unstructured Data

Traditional data quality checks (null rate, schema, value range) không áp dụng được cho unstructured data

- **Document embeddings drift:** cosine similarity giữa batch mới vs baseline giảm → model retrieval quality suy giảm
- **Image quality degradation:** resolution, blur score, aspect ratio distribution thay đổi giữa các batch
- **Text length distribution shifts:** trung bình token count thay đổi đột ngột → upstream data source có vấn đề

Approach: Monitor statistical distributions (KL divergence, KS test) trên derived features thay vì raw content.

Expectation Suite & Profiler

- Nhóm expectations cho một data asset
- Ví dụ: “users” suite kiểm tra email format, age range, null rates
- Versioned trong Git cùng pipeline code
- Reusable across environments (dev/staging/prod)

- Tự động generate expectations từ data sample
- `UserConfigurableProfiler` — chọn columns, rules
- Baseline expectations: not_null, unique, value ranges
- Review & refine — profiler là starting point, không phải final

Great Expectations: Code Example

```python
import great_expectations as gx

context = gx.get_context()

# Create suite
suite = context.add_expectation_suite(
    "users_suite"
)

# Add expectations
suite.add_expectation(
  gx.expectations
    .ExpectColumnValuesToNotBeNull(
      column="email"
    )
)
suite.add_expectation(
  gx.expectations
    .ExpectColumnValuesToBeBetween(
      column="age", min_value=0,
```

- Kết hợp suite + datasource + action list
- Chạy trong CI hoặc Airflow
- Actions on failure:
  - `SlackNotificationAction`
  - `StoreEvaluationParameters`
  - Block pipeline execution

Auto-generate HTML report — share với stakeholders, link trong Airflow task logs

Monte Carlo: ML-Based Data Observability

- Connect to warehouse → ML-based anomaly detection
- Tự động monitor >200 metrics
- Incident timeline: map lineage để tìm root cause trong phút
- Alerting: Slack, PagerDuty, email

- `pandas-profiling` + statistical tests
- Z-score: `abs(current - mean) / std > 3`
- Time-series: Prophet model dự đoán expected value
- Alert nếu actual > 3σ deviation
- Prometheus metrics export

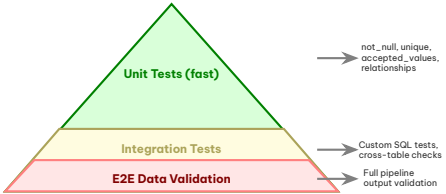
Z-Score Anomaly Detection: Implementation

```python
import numpy as np
from datetime import datetime

def detect_anomaly(current_value, history, threshold=3):
    """Detect anomaly using Z-score method."""
    mean = np.mean(history)
    std = np.std(history)
    if std == 0:
        return False, 0.0
    z_score = abs(current_value - mean) / std
    is_anomaly = z_score > threshold
    return is_anomaly, z_score

# Example: monitor daily row count
daily_counts = [10200, 10150, 10300, 10180, 10250]
today_count = 5100  # 50% drop!
anomaly, score = detect_anomaly(today_count, daily_counts)
# anomaly=True, score=7.2 -> ALERT!
```

dbt Test Pyramid



dbt Tests: Built-in & Custom

- `not_null` — column không chứa NULL
- `unique` — column giá trị duy nhất
- `accepted_values` — chỉ chấp nhận danh sách
- `relationships` — FK tồn tại trong bảng khác
- Free, fast, essential — chạy mỗi `dbt test`

- Custom tests: viết SQL trong `tests/`
- `dbt-expectations`: port Great Expectations vào dbt
- **Elementary**: open-source data observability trên dbt
- Anomaly detection trên dbt models
- Dashboard cho data quality trends

Ví dụ: SLO = 99.5% freshness <60 min → Error budget = 0.5% = 3.6 giờ/tháng cho phép data stale.

Error Budget Burn Rate & Alerting

- **Fast burn alert:** burn 2% budget/hour → P0 alert, cần response ngay
- **Slow burn alert:** burn 5% budget/6h → P1 alert, investigate within shift
- **SLO dashboard:** Google SRE Workbook patterns — Grafana template available
- **Cultural impact:** SLOs buộc team prioritize reliability over features
- **Action khi burn out:** freeze feature deployments, focus reliability fixes

```
freshness_slo: <60 min, 99.5% | null_rate_slo: <0.1%, 99.9% | schema_drift_slo: 0 violations, 100%
```

Incident Severity & Response

Severity	Description	Response Time	Example
P0	Production down	5 min	Pipeline halted, no data flowing
P1	Data incorrect	30 min	Wrong values in serving tables
P2	Degraded quality	2h	SLO breach, slow freshness
P3	Minor issue	Next business day	Documentation gap

- PagerDuty + Rundeck: auto-run diagnostic scripts
- Runbook automation: pre-written recovery steps
- Slack war room: real-time coordination

- Blameless: focus systemic fixes
- 5 Whys analysis
- Action items with owners & deadlines

Chaos Engineering cho Data Pipelines

- Inject failures vào data pipeline
- Netflix Chaos Monkey cho data
- Kill Airflow worker mid-task
- Corrupt upstream data source
- Simulate network partition

- Quarterly drill
- Simulate real data incident
- Practice full cycle:
  - Detection → Diagnosis
  - Remediation → Verify
  - Post-mortem → Improve

Live Demo: Data Incident Detection & Resolution

1. **Demo 1:** Inject schema change vào upstream data → GE checkpoint fails → Slack alert trong 60 giây
2. **Demo 2:** Inject volume anomaly (10% of normal) → Z-score detection → PagerDuty alert
3. **Demo 3:** dbt test failure → lineage graph identify upstream source of corruption
4. **Demo 4:** SLO dashboard — show error budget consumption, burn rate alert kích hoạt
5. **Resolution flow:** alert → runbook → auto-diagnostic → root cause → fix → verify

Mục tiêu: Data Observability Implementation

Deliverable:

- Setup Great Expectations project, create Suite với Profiler cho sample dataset
- Build Checkpoint integrate với Airflow DAG — block pipeline on validation failure
- Implement Z-score anomaly detection cho 5 key metrics, Slack alert khi anomaly
- Define 3 SLOs cho data platform, build Grafana SLO dashboard

Thời gian: 2.5h

Tổng kết — Key Takeaways

Những ý chính cần nhớ sau buổi học hôm nay

1. Data observability ≠ pipeline monitoring — cần cả hai, focus khác nhau. Pipeline succeeded không có nghĩa data đúng.
2. SLOs buộc team prioritize reliability over features — cultural shift quan trọng hơn tooling.
3. Automated anomaly detection phải có human review — false positives cần training models over time.

Tiếp theo & Bài tập

Ngày 28: Integration Workshop — Full Platform Demo

"Tích hợp toàn bộ infrastructure stack, demo end-to-end platform, hoàn thành Milestone 3"

- Hoàn thành Lab 27: Data Observability Implementation
- Review toàn bộ components từ N16–N27
- Chuẩn bị Milestone 3 demo script

Hỏi & Đáp

Câu hỏi nào về data observability, Great Expectations, SLOs, hay incident response?

VINUNIVERSITY

Cảm ơn!

AICB-P2T2 · Ngày 27

Data Observability & Lineage

`lms.vinuni.edu.vn` · Slide & template trên LMS