Data Validation at Scale

Managing Data Quality in Complex Data Pipelines

DB / DC 2023, Munich, Germany
December 05, 2023
Wolle

Material Available at https://wolle.science
(Direct Link to Slides: https://wolle.science/data-validation)
Data Validation at Scale: Managing Data Quality in Complex Data Pipelines

Wolle "Wolle" Wingerath

Material Available at https://wolle.science

(Direct Link to Slides: https://wolle.science/data-validation)

Video of a similar presentation!
Research:
• Stream Processing
• Real-Time Databases
• NoSQL & Cloud Systems
• ...

Practice:
• Web Caching
• Big Data Analytics
• Anger Management
• ...

Slides: wolle.science/data-validation
1. The Importance of Data Validation
   Where is data validation integrated into data science pipelines and what is its impact?

2. Data Quality & Constraints
   What dimensions of data quality are there and how can they be ensured?

3. Scalability-Related Challenges
   Why is data validation difficult in data-intensive domains?
Running Example: Web Performance Analysis Pipeline

Collection → Ingestion → Analytics → Reporting

- Tracking (RUM)
- Performance Dashboard
- QA Dashboard
- Real-Time Alerting
- Ad-hoc SQL Interface
- Custom Reporting

- F. Gessert, W. Wingerath, Batching Was Yesterday: Real-Time Tracking & Analysis For 100+ Million Visitors, Flink Forward (2021)

Running Example: Web Performance Analysis Pipeline

Collection  Ingestion  Analytics  Reporting

Tracking (RUM)  Flink

SQL Interface

Performance Dashboard  QA Dashboard  Real-Time Alerting  Ad-hoc SQL Interface  Custom Reporting

Running Example: Web Performance Analysis Pipeline

Collection  Ingestion  Analytics  Reporting

Tracking (RUM)  Flink

SQL Interface

Performance Dashboard  QA Dashboard  Real-Time Alerting  Ad-hoc SQL Interface  Custom Reporting

Running **Example**: Web Performance Analysis Pipeline

1. Tracking (RUM)
2. Cloud
3. Database
4. Analysis
5. Insight
Running **Example**: Web Performance Analysis Pipeline

Collection → Ingestion → Processing & Storage → Analysis, Reporting & ML

- Collection
- Ingestion
- Processing & Storage
- Analysis, Reporting & ML

- Dashboarding
- Data Warehousing
- Machine Learning
• **Goal**: verify that data in the pipeline is in an acceptable state for downstream processing, e.g.
  
  - External reporting (statistics, visualizations & dashboarding)
  - Internal reporting (debugging, product optimizations)
  - Decision-making (analytics, machine learning)

• **Data validation can be integrated in and between all stages**
There are different **dimensions of data quality**, especially:

- **Completeness**: Do we have all the data we need to assess page load performance?
- **Consistency**: Does data have a valid format and does it comply with business semantics?
- **Accuracy**: Do data items represent their corresponding real-world entities well?
- **Uniqueness**: Are duplicate records known and are all unique attributes actually distinct?
There are different dimensions of data quality, especially:

- **Completeness**: Do we have all the data we need to assess page load performance?
- **Consistency**: Does data have a valid format and does it comply with business semantics?
- **Accuracy**: Do data items represent their corresponding real-world entities well?
- **Uniqueness**: Are duplicate records known and are all unique attributes actually distinct?
Unload Beacon Reliability as an Example Challenge

- Is beacon loss a problem at all?
- **When** is beacon loss a problem?
  - For which beacon types? For which beacon strategies?
- **Where** is beacon loss a problem?
  - For which browsers? For which device types?
Unload Beacon Reliability by Strategy

```javascript
1. addEventListener("unload", (event) => {
2.   navigator.sendBeacon(url, JSON.stringify(data));
3. });
```

available on all platforms

```javascript
1. addEventListener("beforeunload", (event) => {
2.   navigator.sendBeacon(url, JSON.stringify(data));
3. });
```

experimental feature
(only available as origin trial in Chrome)

```javascript
1. addEventListener("pagehide", (event) => {
2.   navigator.sendBeacon(url, JSON.stringify(data));
3. });
```

```javascript
1. var beacon = new window.PendingPostBeacon(
2.   url,
3.   {
4.     timeout: 60000,
5.     backgroundTimeout: 0
6.   });
7. beacon.setData(JSON.stringify(data));
```

Wolfram Wingerath (Munich, December 05, 2023)

Slides: wolle.science/data-validation
Unload Beacon Reliability

1. `addEventListener("unload", (event) => {
2.   navigator.sendBeacon(url, JSON.stringify(data));
3. })`;

Available on all platforms currently only available as origin trial in Chrome

1. `addEventListener("beforeunload", (event) => {
2.   navigator.sendBeacon(url, JSON.stringify(data));
3. })`;

1. `addEventListener("pagehide", (event) => {
2.   navigator.sendBeacon(url, JSON.stringify(data));
3. })`;

1. `if (document.visibilityState == "hidden") {
2.   navigator.sendBeacon(url, JSON.stringify(data));
3. }
4. })`;

1. `beacon.setData(JSON.stringify(data));`

Sophie Ferrlein. Data Viz for Engineers: Optimizing Insight & Decision Making Through Visualization. code.talks (2023)
Unload Beacon Reliability by Strategy & Device

<table>
<thead>
<tr>
<th>Strategy</th>
<th>MOBILE</th>
<th>DESKTOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>PendingPostBeacon</td>
<td>97%</td>
<td>99%</td>
</tr>
<tr>
<td>visibilitychange</td>
<td>90%</td>
<td>91%</td>
</tr>
<tr>
<td>pagehide</td>
<td>81%</td>
<td>88%</td>
</tr>
<tr>
<td>unload</td>
<td>78%</td>
<td>87%</td>
</tr>
<tr>
<td>beforeunload</td>
<td>28%</td>
<td>73%</td>
</tr>
</tbody>
</table>

Based on 52 million page views on a globally operating e-commerce site measured by Speed Kit’s real user-monitoring | August 2023

Desktop data may be overrepresented!

## Unload Beacon Reliability by Strategy & Browser

<table>
<thead>
<tr>
<th>Pending PostBeacon</th>
<th>Chrome</th>
<th>Safari</th>
<th>Firefox</th>
</tr>
</thead>
<tbody>
<tr>
<td>visibility change</td>
<td>98%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>pagehide</td>
<td>92%</td>
<td>89%</td>
<td>95%</td>
</tr>
<tr>
<td>unload</td>
<td>88%</td>
<td>80%</td>
<td>91%</td>
</tr>
<tr>
<td>before unload</td>
<td>86%</td>
<td>80%</td>
<td>90%</td>
</tr>
<tr>
<td></td>
<td>83%</td>
<td>3%</td>
<td>91%</td>
</tr>
</tbody>
</table>

Based on 52 million page views on a globally operating e-commerce site measured by Speed Kit's real user monitoring | August 2023

---

Unload Beacon Reliability: The Ideal Combo Strategy

Based on 52 million page views on a globally operating e-commerce site measured by Speed Kit’s real user monitoring | August 2023

Can be used with Backward-Forward Cache?

- visibilitychange + pagehide + beforeunload + unload
  - 91.7%
  - ✗

- visibilitychange + pagehide + beforeunload
  - 91.7%
  - ✗

- visibilitychange + pagehide
  - 91.3%
  - ✓

Dimensions of Data Quality: **Consistency**

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Pageload ID</th>
<th>Browser</th>
<th>LCP (Performance)</th>
<th>Session ID</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>09:05:04.578</td>
<td>37ab08</td>
<td>Edge</td>
<td>&quot;670ms&quot;</td>
<td>123</td>
<td>null</td>
</tr>
<tr>
<td>13:26:48.139</td>
<td>9cddf7</td>
<td>Firefox</td>
<td>692 654</td>
<td>456</td>
<td>abc.de/red</td>
</tr>
<tr>
<td>13:28:23.857</td>
<td>0b577a</td>
<td>Firefox</td>
<td>0.256</td>
<td>456</td>
<td>abc.de/blue</td>
</tr>
<tr>
<td>13:29:17.468</td>
<td>faf55e</td>
<td>Edge</td>
<td>1.598</td>
<td>456</td>
<td>abc.de/sold</td>
</tr>
<tr>
<td>20:45:38.941</td>
<td>faf55e</td>
<td>null</td>
<td>null</td>
<td>null</td>
<td>abc.de/sold</td>
</tr>
</tbody>
</table>

**Note:** Browser values should be unified for all records in the same session!

- There are different **dimensions of data quality**, especially:
  - **Completeness:** Do we have all the data we need to assess page load performance?
  - **Consistency:** Does data have a valid format and does it comply with business semantics?
  - **Accuracy:** Do data items represent their corresponding real-world entities well?
  - **Uniqueness:** Are duplicate records known and are all unique attributes actually distinct?
There are different **dimensions of data quality**, especially:

- **Completeness**: Do we have all the data we need to assess page load performance?
- **Consistency**: *Does data have a valid format and does it comply with business semantics?*
- **Accuracy**: Do data items represent their corresponding real-world entities well?
- **Uniqueness**: Are duplicate records known and are all unique attributes actually distinct?

**Note**: Browser values should be unified for all records in the same session! → value may be replaced with majority vote (another reasonable option: replace old values with latest one)

- Timestamp | Pageload ID | Browser | LCP (Performance) | Session ID | URL
- --- | --- | --- | --- | --- | ---
- 09:05:04.578 | 37ab08 | Edge | "670ms" | 123 | null
- 13:26:48.139 | 9cddf7 | Firefox | 692 654 | 456 | abc.de/red
- 13:28:23.857 | 0b577a | Firefox | 0.256 | 456 | abc.de/blue
- 13:29:17.468 | faf55e | Firefox | 1.598 | 456 | abc.de/sold
- 20:45:38.941 | faf55e | null | null | null | abc.de/sold

Dimensions of Data Quality: **Consistency**
There are different **dimensions of data quality**, especially:

- **Completeness**: Do we have all the data we need to assess page load performance?
- **Consistency**: Does data have a valid format and does it comply with business semantics?
- **Accuracy**: Do data items represent their corresponding real-world entities well?
- **Uniqueness**: Are duplicate records known and are all unique attributes actually distinct?

### Note
All values represent milliseconds, but formats differ depending on browser.
There are different dimensions of data quality, especially:

- **Completeness**: Do we have all the data we need to assess page load performance?
- **Consistency**: Does data have a valid format and does it comply with business semantics?
- **Accuracy**: Do data items represent their corresponding real-world entities well?
- **Uniqueness**: Are duplicate records known and are all unique attributes actually distinct?

**Note**: All values represent milliseconds, but formats differ depending on browser. → values may be converted to integer.
There are different dimensions of data quality, especially:

- **Completeness**: Do we have all the data we need to assess page load performance?
- **Consistency**: Does data have a valid format and does it comply with business semantics?
- **Accuracy**: Do data items represent their corresponding real-world entities well?
- **Uniqueness**: Are duplicate records known and are all unique attributes actually distinct?

**Note**: Despite being in the right format, one value does not represent a reasonable timer value.

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Pageload ID</th>
<th>Browser</th>
<th>LCP (Performance)</th>
<th>Session ID</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>09:05:04.578</td>
<td>37ab08</td>
<td>Edge</td>
<td><strong>670</strong></td>
<td>123</td>
<td>null</td>
</tr>
<tr>
<td>13:26:48.139</td>
<td>9cddf7</td>
<td>Firefox</td>
<td><strong>692 654</strong></td>
<td>456</td>
<td>abc.de/red</td>
</tr>
<tr>
<td>13:28:23.857</td>
<td>0b577a</td>
<td>Firefox</td>
<td>256</td>
<td>456</td>
<td>abc.de/blue</td>
</tr>
<tr>
<td>13:29:17.468</td>
<td>faf55e</td>
<td>Firefox</td>
<td>1598</td>
<td>456</td>
<td>abc.de/sold</td>
</tr>
<tr>
<td>20:45:38.941</td>
<td>faf55e</td>
<td>null</td>
<td>null</td>
<td>null</td>
<td>abc.de/sold</td>
</tr>
</tbody>
</table>
There are different dimensions of data quality, especially:

- **Completeness**: Do we have all the data we need to assess page load performance?
- **Consistency**: Does data have a valid format and does it comply with business semantics?
- **Accuracy**: Do data items represent their corresponding real-world entities well?
- **Uniqueness**: Are duplicate records known and are all unique attributes actually distinct?

### Table of Data Quality Dimensions

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Pageload ID</th>
<th>Browser</th>
<th>LCP (Performance)</th>
<th>Session ID</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>09:05:04.578</td>
<td>37ab08</td>
<td>Edge</td>
<td>670</td>
<td>123</td>
<td>null</td>
</tr>
<tr>
<td>13:26:48.139</td>
<td>9cddf7</td>
<td>Firefox</td>
<td>null</td>
<td>456</td>
<td>abc.de/red</td>
</tr>
<tr>
<td>13:28:23.857</td>
<td>0b577a</td>
<td>Firefox</td>
<td>256</td>
<td>456</td>
<td>abc.de/blue</td>
</tr>
<tr>
<td>13:29:17.468</td>
<td>faf55e</td>
<td>Firefox</td>
<td>1598</td>
<td>456</td>
<td>abc.de/sold</td>
</tr>
<tr>
<td>20:45:38.941</td>
<td>faf55e</td>
<td>null</td>
<td>null</td>
<td>null</td>
<td>abc.de/sold</td>
</tr>
</tbody>
</table>

**Note**: Despite being in the right format, one value does not represent a reasonable timer value. → broken value may be removed.
There are different dimensions of data quality, especially:

- **Completeness**: Do we have all the data we need to assess page load performance?
- **Consistency**: Does data have a valid format and does it comply with business semantics?
- **Accuracy**: Do data items represent their corresponding real-world entities well?
- **Uniqueness**: Are duplicate records known and are all unique attributes actually distinct?

---

**Note**: The ID field should be unique, but two different records share the same value!
There are different **dimensions of data quality**, especially:

- **Completeness**: Do we have all the data we need to assess page load performance?
- **Consistency**: Does data have a valid format and does it comply with business semantics?
- **Accuracy**: Do data items represent their corresponding real-world entities well?
- **Uniqueness**: Are duplicate records known and are all unique attributes actually distinct?
There are different **dimensions of data quality**, especially:

- **Completeness**: Do we have all the data we need to assess page load performance?
- **Consistency**: Does data have a valid format and does it comply with business semantics?
- **Accuracy**: Do data items represent their corresponding real-world entities well?
- **Uniqueness**: Are duplicate records known and are all unique attributes actually distinct?

---

**Note**: The ID field should be unique, but two different records share the same value! → merge duplicates into a single record
• **Constraints** are rules, conditions, or limits that data must adhere to

• **Type Checks** represent expectations on the data format, e.g.
  - *Value range* for numerical data (e.g. [0,MAX_INTEGER) for load timers)
  - *Format or pattern* for string-valued data (e.g. ISO 8601 for timestamps)
  - *Structure* for complex attributes (e.g. required keys for JSON objects)

• **Complex conditions** can further describe complex semantics such as
  - Cross-field or cross-record relationships (e.g. same browser within sessions)
  - Referential integrity between records in different collections
  - Custom constraints for domain semantics
import pandas as pd
import pandera as pa
from pandera import Column, DataFrameSchema, Check

# Define the schema
schema = DataFrameSchema(
    {
        "Timestamp": Column(pa.DateTime),
        "Pageload ID": Column(pa.String,
            Check(lambda x: x.str.len() == 8)),
        "Browser": Column(pa.String,
            Check(lambda x: x.isin(["Chrome", "Edge", "Firefox"]))),
        "LCP": Column(pa.Int,
            Check(lambda x: (x >= 0) & (x <= 600000))),
        "Session ID": Column(pa.Int),
    }
)
Example: Declarative Constraints With Pandera (2/3)

```python
# valid data item
good = {
    "Timestamp": [pd.Timestamp("2023-05-10 13:26:48.139")],
    "PageLoad ID": ["9cddf7"],
    "Browser": ["Firefox"],
    "LCP": [256],
    "Session ID": [456],
}

# invalid data item
bad = {
    "Timestamp": [pd.Timestamp("2023-05-10 09:05:04.578")],
    "PageLoad ID": ["37ab08"],
    "Browser": ["Edge"],
    "LCP": [692654], # timer value out of bounds
    "Session ID": [123],
}
```
for record in [good, bad]:
    record_id = record['Pageload ID'][0]
    try:
        validated = schema(pd.DataFrame(record))
        print(f"nValidation passed for record {record_id}!"")
    except pa.errors.SchemaError as e:
        print(f"nValidation FAILED for record {record_id}:")
        print(e)
One Month in Data Errors at Baqend: April 2023

Fundamental Challenge: Scalability

Data Errors by Type:
- Aggregation (66.89%)
- Attribute (32.83%)
- Internal (0.16%)

Data Errors by Cause:
- MissingValue (63.91%)
- BlankValue (11.92%)
- ValueOutOfRange (8.60%)
- AmbiguousValue (7.98%)
- StringLengthExceeded (4.30%)
- Other (2.46%)
Manual constraint definition is often infeasible, because of ...

- ... inherent data complexity (often hundreds of attributes)
- ... aggregation, derived storage, and evolving schemas
- ... a plethora of other data stores to integrate!

→ Automation is necessary!
Computing validation metrics from scratch periodically can be infeasible, because of:

- ... strict timing requirements
- ... efficiency or cost reasons
- ... data privacy reasons

→ Incremental computation can be the only option!
Specifying generalized constraints can be difficult in large deployments, because of ...

- temporal fluctuations (e.g. throughout the day, on black Friday, or during holidays)
- multi-tenancy (e.g. different data patterns by customer timezone or domain)

→ Elasticity & Multi-Tenancy requirements can be challenging!
Deployments can break things (and cause sudden load spikes for validation)

- **Availability**
  Can this take down your data pipeline?
- **Attribution**
  Who or what is causing the problem?
- **Responsibility**
  Can you fix the problem?
So How Do You Handle All This?

- **Advanced Techniques**
  - Inferring constraints
  - Adapting to schema changes
  - Incremental computation of complex measures

- **Tooling & Frameworks**
  - Validation libraries such as Great Expectations, Pandera, TFDV, or Deequ
  - Preprocessing and validation with Apache Spark and Apache Flink

- **Further Challenges**
  - Handling distribution (validation per partitioning, avoiding skew, ...)
  - Efficiency and performance (load distribution, approximation, ...)
  - Operational challenges (anomaly detection, fixing, load shedding, ...)

Wolfram Wingerath (Munich, December 05, 2023)

Slides: wolle.science/data-validation
Data Validation at Scale: Summary

- **Data Quality** can be measured along dimensions such as completeness, consistency, accuracy, and uniqueness.
- **Constraints** specify expectations about the data and can be used to enforce them.
- **Data Validation** is the process of ensuring high data quality for processes like analysis, modeling, and decision-making.
- Data Validation **Challenges at Scale** include:
  - **Complexity**: schemas are often too complex to define constraints manually.
  - **Volatility**: data varies throughout the day, by season, or by customer.
  - **Continuity**: incremental processing is required when computation from scratch is infeasible.
Thanks! **Questions?**

Material Available at https://wolle.science