

The Ultimate Data Observability Platform Evaluation Guide



**Getting started with data observability?
Here are the 10 things you need to succeed.**

MC

MONTE CARLO



Overview

What is Data Observability?

For the past decade or so, software engineers have leveraged targeted solutions like New Relic and DataDog to ensure high application uptime while keeping downtime to a minimum.

In data, we call this phenomena data downtime. Data downtime refers to periods of time when data is partial, erroneous, missing, or otherwise inaccurate, and it only multiplies as data systems become increasingly complex, supporting an endless ecosystem of sources and consumers.

The good news? By applying the same principles of software application observability and reliability to data, these issues can be identified, resolved and even prevented. Data Observability is an organization's ability to fully understand the health of the data in their system by eliminating data downtime via best practices of DevOps and software engineering.



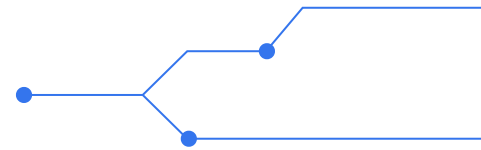
DATA OBSERVABILITY PILLARS

Freshness | Distribution | Volume | Schema | Lineage

Assessing data observability platforms

Data observability platforms are the newest layer in the modern data stack, helping teams monitor the health of critical data assets and pipelines while building organizational trust in data at scale.

Read on for the definitive checklist to evaluate data observability platforms across 10 key areas: end-to-end visibility, automated monitoring & alerting across freshness, volume, and schema, easily configurable distribution monitors, rapid ML-based detection & resolution, centralized root cause and impact analysis, quick time-to-value, and enterprise readiness.



1. End-to-End Pipeline Visibility

Your data observability platform should give you unprecedented visibility into your data ecosystem so you can proactively identify, root cause, and resolve data issues from source to orchestration and transformation tooling, to BI layer.

Your most critical asset? Automated field-level lineage.

Limited to no visibility into data pipelines upstream of a data assets makes troubleshooting manual and time consuming.

With data observability, data teams should be able to see the entire data lineage upstream and across warehouse projects of a BI report or materialized tables by simply searching in a centralized UI.

Lineage provides information to end users on dashboard staleness and other incidents due to downstream issues, and offers proactive troubleshooting to help address issues in a timely manner.

Field Lineage
Column-level lineage across data assets

Left Panel (Assets):

- ANALYTICS:PROD active_monitors
- ANALYTICS:PROD_DETECTORS metric_status_indicator
- ANALYTICS:PROD_INSIGHTS insight_custom_rules
- ANALYTICS:PROD_STAGE custom_rules
- ANALYTICS:PROD_DETECTORS numeric_status_indicator
- ANALYTICS:PROD table_metadata
- ANALYTICS:PROD recent_metrics

Center Panel (Selected Asset):

- ANALYTICS:PROD monitor_issues_and_suggestions (VARCHAR(30))

Right Panel (Lineage):

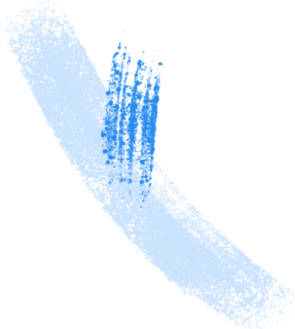
- ANALYTICS:DBT_APELTNOVICH monitor_issues_and_suggesti...
- ANALYTICS:PROD_SNAPSHOTS monitor_issues_and_suggesti...
- ANALYTICS:PROD_INSIGHTS to_monolith_monitor_issues_...
- ANALYTICS:PROD_INSIGHTS insight_monitor_issues_and_s...

SELECT clause lineage (1)

- monitor_status (VARCHAR(30))

Non-SELECT lineage (16)

- account_id (VARCHAR(6777216))
- tables (ARRAY)
- full_table_id (VARCHAR(6777216))
- project_name (VARCHAR(6777216))
- dataset_name (VARCHAR(6777216))
- created_on (TIMESTAMP_NTZ(0))



2. Quick time-to-value

Data teams should get immediate value out of data observability tools. Full stop.

They should be easy and quick to deploy, and fast to start detecting anomalies or other issues with your data, instead of a weeks-long process in manually configuring monitors. Even the most efficient data engineering team doesn't have time for that.

At a fundamental level, a data observability platform should monitor your data's health and alert your team when pipelines break or jobs don't run well before your downstream data consumers notice the issue in their dashboards or queries.

Knowing what issues to monitor for can be overwhelming which is where a ML driven approach is critical. Instead of relying on manual tests to check for data quality issues, your data observability platform should intelligently surface when changes are noticed.

A tool that provides fast time to value will quickly address any questions regarding data freshness, volume, distribution, and schema changes over time, and provided domain data quality dashboards for technical and business users alike. The solution will also reduce time to monitor assets over time and provide a single pane for data governance needs.



Aug 11th 2022 11:00 EDT More than 0 rows returned Every 1 hour(s) 44 + 44

Anomalous rows distribution
(Based on sampled data)

Fields	Type	Min	P25th	Median	P75th	Max	Field Distribution
METRIC	STRING	-	-	-	-	-	total_views
DIMENSIONS	STRING	-	-	-	-	-	null
WHERE_CON_...	STRING	-	-	-	-	-	
MONITOR_U_...	STRING	-	-	-	-	-	
IS_BOOTSTR...	BOOLEAN	-	-	-	-	-	False
CONTEXT	STRING	-	-	-	-	-	
COUNTER	INTEGER	2	2	2	2	3	2
MEASUREME...	DATETIME	202...	-	-	-	202...	2022-08-11T11:34:39, 2022-08-11T11:34..., 2022-08-11T11..., 2022-08-11T11..., Others, 86.4%
TIMESTAMP	DATETL...	-	-	-	-	-	2022-08-11T11:34:39+00:00, 2022-08-11T11:34..., 2022-08-11T11..., 2022-08-11, Others, 86.4%
VALUE	FLOAT	1	3.75	12.5	84	6801	10, Others

Help

3. Automated monitoring for freshness, volume, and schema

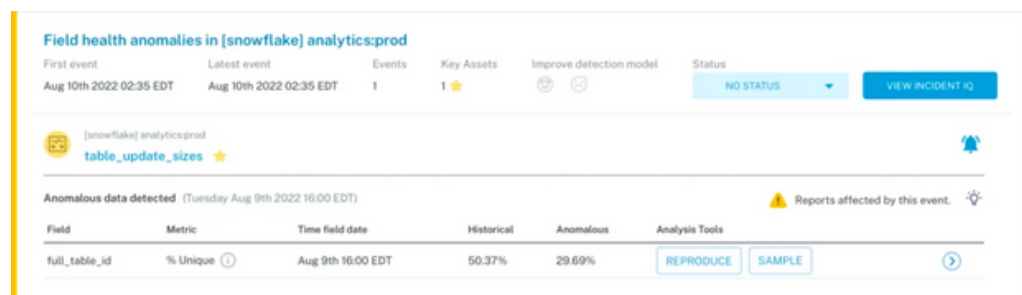
Data observability solutions should proactively (and automatically) alert you to freshness and volume anomalies, as well as schema changes that are then logged in a central catalog.

Your platform should use metadata such as query logs and schema change logs to automatically alert when changes break your data pipelines. By using this metadata instead of the data itself, your platform provides an additional layer of security and reduces costs by limiting data monitoring on field values requiring direct queries to only your most critical assets (more on that later).

This is table stakes for any data observability platform. Just as software engineers rely on ML-powered monitoring to alert them to any outages or downtime, data teams should be able to rely on monitoring to automatically notify them of any anomalies in data health rather than waiting for an internal consumer or customer to notice something's gone wrong.

Your tool must also speed time to resolution, not just time detection through root cause analysis. Your solution will provide automated downstream lineage and incident impact summary to support incident prioritization and communication.

Automated incident reporting should identify issues that do not quickly surface in any dashboards or buckets.



The screenshot displays a dashboard for monitoring field health anomalies in a Snowflake environment. The main heading is "Field health anomalies in [snowflake] analytics:prod". Below this, there are summary statistics: "First event" (Aug 10th 2022 02:35 EDT), "Latest event" (Aug 10th 2022 02:35 EDT), "Events" (1), and "Key Assets" (1). There are also buttons for "Improve detection model", "Status" (set to "NO STATUS"), and "VIEW INCIDENT IQ".

The dashboard identifies an anomaly for the table "[snowflake] analytics:prod table_update_sizes". A section titled "Anomalous data detected" (Tuesday Aug 9th 2022 16:00 EDT) shows a table with the following data:

Field	Metric	Time field date	Historical	Anomalous	Analysis Tools
full_table_id	% Unique	Aug 9th 16:00 EDT	50.37%	29.69%	REPRODUCE SAMPLE

4. Deep data quality monitors for field-health and dimension tracking

While automation is critical for freshness, volume, and schema-related incidents, your organization knows your data assets best. As a result, your data observability platform should include the flexibility for data engineers, data analysts, and data scientists to set their own thresholds to monitor data according to business needs or as new pipelines and projects are introduced.

User-defined, machine learning-powered monitors are best deployed using the most well-defined, understood logic to catch anomalies that are frequent or severe. It is typically expressed through a SQL statement.

A strong data observability solution will proactively alert you to field-level anomalies, such as an increase in the number of dimensions, a change in the % of unique values, or whether a value is within historical range, through Field Health and Dimension tracking monitors in a timely manner, with no-code configuration.

A Warning About Deep Monitoring

When you only deploy user-defined and machine learning monitors on your most important tables:

- You miss or are delayed in detecting and resolving anomalies more evident upstream.
- Alerting on a key table, oftentimes dozens of processing steps removed from the root cause, will involve the wrong person and give them little context on the source of the issue or how to solve it.
- Without understanding the dependencies in the system, your team will waste time trying to determine where to focus their monitoring attention. Maintaining that as the environment and data consumption patterns change can also be tedious and error prone.

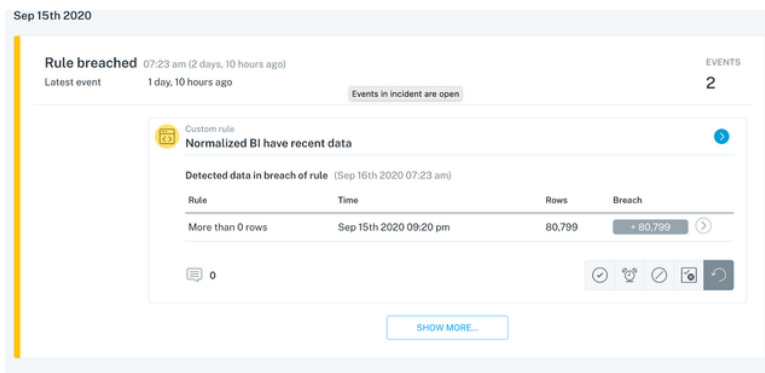
5. Ability to create custom data quality monitors

While automation saves manual labor, your data organization knows your data assets best. Most data teams need an approach that leverages the best of both worlds: using machine learning to identify abnormalities in your data based on historical behavior, as well as the ability to set rules unique to the specs of your data. Unlike ad hoc queries coded into modeling workflows or SQL wrappers, such monitoring doesn't stop at "field T in table R has values lower than S today."

As a result, your data observability platform should include the flexibility for data engineers, data analysts, and data scientists to set their own thresholds to monitor data according to business needs or as new pipelines and projects are introduced.

With such a feature, teams can write custom SQL to check for anomalies specific to your business. Given the flexibility of SQL, users often use these rules to get notified when thresholds for null values, referential integrity, empty strings, etc. are breached. Even better if you can write new monitors as code, instead of having to configure new rules via UI.

A strong data observability solution will also allow you to use a single numeric value returned by the custom SQL query as the metric instead of the row count of the query result. For example, if you want to trigger an incident when the sum of a specific column drops below a given value, you might want to use a value-based threshold.



The screenshot displays a notification for a "Rule breached" event. The event occurred at 07:23 am on Sep 16th, 2020, 10 hours ago. The latest event was 1 day, 10 hours ago. The notification indicates that 2 events in the incident are open. The rule being breached is a custom rule named "Normalized BI have recent data". The detected data in breach of the rule is as follows:

Rule	Time	Rows	Breach
More than 0 rows	Sep 15th 2020 09:20 pm	80,799	+ 80,799

At the bottom of the notification, there is a "SHOW MORE..." button.

6. Centralized alert routing

Data observability should apply across your entire organization, standardizing and centralizing data quality monitoring to ensure every team is speaking the same language and operating from the same playbook.

Alerts are only helpful if the proper team (i.e., the data owner) receives them. Your data observability platform should enable routing based on assignments and incident types.

Alerts should be meaningful communications, not white noise. Not only should your data observability platform support intelligent routing of alerts to the right team members—ideally, those alerts should be delivered via their preferred channels (think Slack, email, or SMS text message).



The screenshot displays a dashboard for 'Anomalous rows distribution' based on sampled data. It shows a table with columns for Fields, Type, Min, P25th, Median, P75th, Max, and Field Distribution. The 'total_views' metric is highlighted with a dark blue bar. Other metrics like 'DIMENSIONS', 'WHERE_CON...', 'MONITOR_U...', 'IS_BOOTSTR...', 'CONTEXT', 'COUNTER', 'MEASUREME...', 'TIMESTAMP', and 'VALUE' are also listed with their respective distributions.

Fields	Type	Min	P25th	Median	P75th	Max	Field Distribution
METRIC	STRING	-	-	-	-	-	total_views
DIMENSIONS	STRING	-	-	-	-	-	null
WHERE_CON...	STRING	-	-	-	-	-	
MONITOR_U...	STRING	-	-	-	-	-	
IS_BOOTSTR...	BOOLEAN	-	-	-	-	-	False
CONTEXT	STRING	-	-	-	-	-	
COUNTER	INTEGER	2	2	2	2	3	2
MEASUREME...	DATETIME	202...	-	-	-	202...	2022-08-11T11:34:39, 2022-08-11T11:34..., 2022-08-11T11..., 2022-08-11T11..., Others
TIMESTAMP	DATETL...	-	-	-	-	-	2022-08-11T11:34:39+00:00, 2022-08-11T11:34..., 2022-08-11T11..., 2022-08-11T..., Others, 86.4%, Others
VALUE	FLOAT	1	3.75	12.5	84	6801	1.0, Others

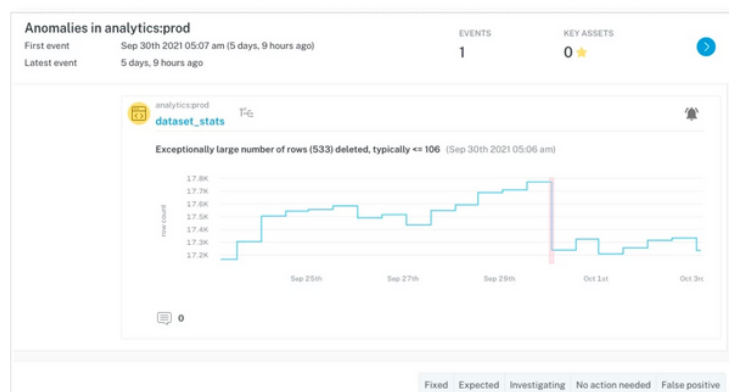
7. Root cause and impact analysis in a single UI

Data breaks. When it happens, it's critical that your team can assess the incident timeline, view exactly when the freshness, distribution, or volume anomaly occurred, access query and orchestration error logs, and trace the issue upstream and downstream—all in one place, and across your modern data stack.

Identifying the root cause of issues is time consuming and often requires input from multiple stakeholders. The best data observability solutions will incorporate automated downstream lineage and incident impact summary to support incident prioritization and communication.

Faster time-to-resolution is enabled with automated lineage, impact radius assessment, workflow tools, and the context your teams need—all in one platform.

As soon as your data team is notified of an incident, they can immediately triage by understanding who and which reports were impacted.

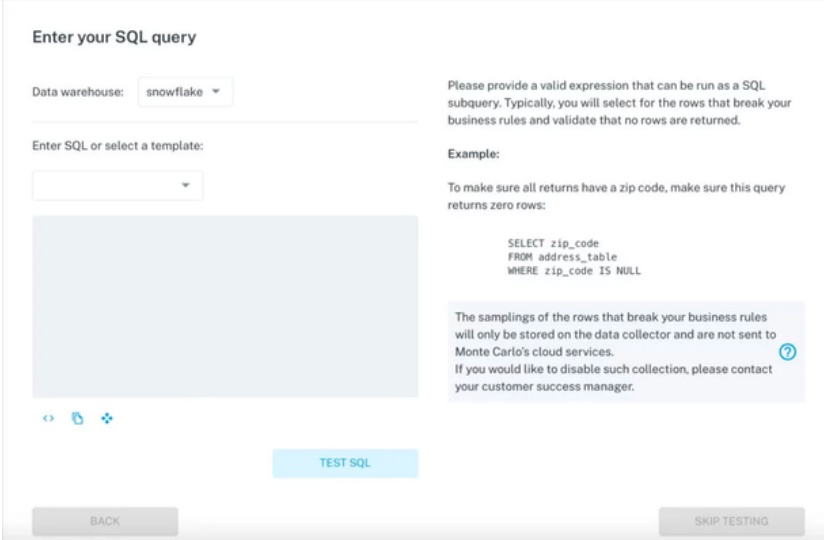


8. Ability to easily configure and manage monitors as part of existing CI/CD process

Your platform should also enable a code-based solution that allows customers to rapidly scale custom distribution monitors, all from the comfort of their IDE – in other words, monitors as code.

With monitors as code, data engineers can configure monitors via a YAML config file and apply those monitors easily as part of the build process or within their CI/CD process. Here's why it's a must-have for your data observability tool:

1. It's maintained in source control so changes are properly tracked and approved.
2. It can be automated (e.g. you can easily create 100 monitors without repetitive UI clicking, you can enforce certain standards, etc.)
3. It naturally fits into the data engineering workflow (which consists of writing code, tests, and now, monitors.)
4. It makes creating new monitors consistent and predictable, and less error-prone (e.g. you won't get someone accidentally deleting/updating monitors.)



The screenshot shows a web interface for configuring a monitor. It features a 'Data warehouse' dropdown menu set to 'snowflake'. Below it is a section for 'Enter SQL or select a template' with a dropdown menu and a large text area for the SQL query. To the right, there is an 'Example' section with a sample SQL query:

```
SELECT zip_code
FROM address_table
WHERE zip_code IS NULL
```

. Below the example, there is a note: 'The samplings of the rows that break your business rules will only be stored on the data collector and are not sent to Monte Carlo's cloud services. If you would like to disable such collection, please contact your customer success manager.' At the bottom, there are three buttons: 'BACK', 'TEST SQL', and 'SKIP TESTING'.

9. Enterprise readiness

To support the growing demand for data democratization and decentralized data ownership while meeting your company's strict compliance needs, your data observability platform should:

1. Offer SOC-2 Type II certification
2. Never extract or store individual records, PII, or other sensitive information outside of your environment
3. Allow you to comply with HIPAA, PCI, GDPR, CCPA, FINRA, and other compliance frameworks that you are subjected to
4. Allow easy and simple deployment with little to no ongoing operational overhead and frequent automatic upgrades
5. Provide a robust customer support experience that is quick, considerate of user needs and diverse tech stacks, and provides extensive documentation.

Another critical element of being enterprise ready is having ample support from your vendor, on-demand enablement sessions, and proactive product & engineering leadership that frequently sources new feature suggestions and improvements based on your team's needs.



10. Minimal setup effort & maintenance

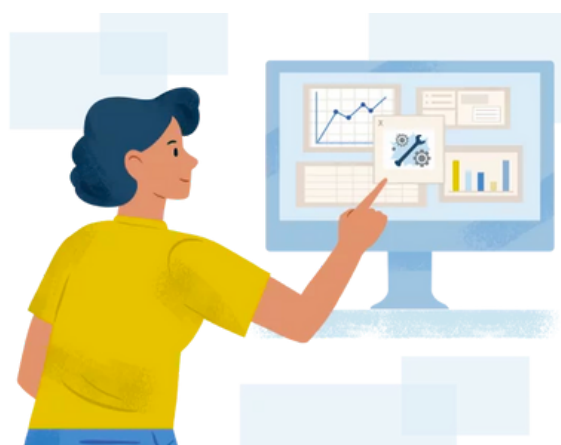
It should be easy - and quick - to deploy and spin up data observability.

A robust platform can be deployed in minutes, and immediately notifies specific data team owners as soon as data incidents occur. The best solutions should take ~20-30 mins to connect to your data warehouse and BI tool.

Data observability tools should be deployed using no-code implementation and pre-built integrations across your cloud-native data stack, allowing for setup in 20 minutes with complete field lineage mapping within 24 hours.

Machine learning-powered monitors are automatically deployed across 100% of your production tables to evaluate freshness, volume, and schema changes within 2 weeks—without configuring or thresholding.

Once deployed, your data observability platform should be easy to use and maintain, with centralized root cause analysis, impact analysis, and monitor configuration, as well as capabilities that help you understand which fields, tables, and queries are outdated, unused, or inefficient.



Interested in more reliable data?



[Stay up-to-date with all things data on the Data Downtime Blog](#)



[Register for IMPACT 2022: The Data Observability Summit](#)



[Request a demo of Monte Carlo](#)



MONTE CARLO