DZone

TREND REPORT

SEPTEMBER 2023

Data Pipelines

Investigating the Modern Data Stack

BROUGHT TO YOU IN PARTNERSHIP WITH

Prophecy

Welcome Letter

By ChatGPT, Assistant at OpenAI

In the era of data-driven decision-making, organizations are grappling with an unprecedented influx of information from a myriad of sources. The ability to harness the potential of this data, transforming it into actionable insights, has become a hallmark of successful enterprises.

At the heart of this data revolution lies the concept of **data pipelines**, which play a pivotal role in seamlessly processing, transforming, and delivering data to power various business functions.

The landscape of data pipelines has evolved significantly, particularly in tandem with the emergence of the modern data stack. Traditional data management approaches no longer suffice in the face of the ever-increasing volumes, velocities, and varieties of data. The modern data stack represents a comprehensive ecosystem of tools, technologies, and practices that address the challenges of data processing, analysis, and utilization in a rapidly changing business environment.

This Trend Report delves into the intricacies of data pipelines within the context of the modern data stack. We explore the fundamental concepts of data ETL, as well as the evolving paradigm of ELT, and how these approaches contribute to efficient data movement.

Additionally, we will investigate the significance of real-time data processing and analytics in the context of modern data pipelines, enabling organizations to make agile decisions based on up-to-the-moment insights.

Observability emerges as a critical aspect of data pipelines in the modern landscape. The ability to monitor, trace, and troubleshoot the flow of data through the pipeline ensures operational excellence, minimizes downtime, and enhances the quality of insights derived from the data.

Moreover, as AI and ML take center stage, data pipelines become the backbone that facilitates the seamless integration of these advanced technologies.

Cloud-based data engineering, a linchpin of the modern data stack, brings unparalleled scalability, flexibility, and cost effectiveness to data pipelines. The cloud empowers organizations to handle data of any scale while allowing them to dynamically adjust resources as needed.

As the data landscape continues to evolve, embracing the principles of modern data pipelines becomes a strategic imperative. DZone's 2023 *Data Pipelines* Trend Report serves as a guide to navigate the complexities of data pipelines within the context of the modern data stack, illuminating the path toward a more agile, efficient, and effective data-driven future.

Best regards,

ChatGPT

ChatGPT, Assistant at OpenAI
chat.openai.com

The AI model known as ChatGPT is a creation of OpenAI. Developed on the GPT-3.5 architecture, ChatGPT excels in offering diverse assistance, from informative responses to creative writing. With knowledge extending up to September 2021, it serves as a valuable resource across a wide range of subjects.

ORIGINAL RESEARCH

Key Research Findings

An Analysis of Results From DZone's 2023 Data Pipelines Survey

By G. Ryan Spain, Freelance Software Engineer, former Engineer & Editor at DZone

Data analytics is no longer just about using data to drive business decisions; we are entering a new era where cloud-based systems and tools are at the heart of data processing and analytics. Data-centric tools and techniques like warehouses and lakes, ETL/ELT, observability, real-time analytics, and so much more — these technologies are democratizing the data we collect. The proliferation of and growing emphasis on data democratization results in increased and nuanced ways in which data platforms can be used — and by extension, also empowers business users to make data-driven decisions with confidence.

In August 2023, DZone surveyed software developers, architects, and other IT professionals in order to understand the state of data pipelines.

Major research targets were:

- ETL/ELT methods and solutions
- Data pipeline and storage practices
- Impacts of data pipelines in the software profession

Methods: We created a survey and distributed it to a global audience of software professionals. Question formats included multiple choice, free response, and ranking. Survey links were distributed via email to an opt-in subscriber list, popups on DZone.com, the DZone Core Slack workspace, and various DZone social media channels. The survey was opened on August 11th and ended on August 29th; it recorded 98 complete and partial responses.

Demographics: Due to the limited response rate for our 2023 Data Pipelines survey, we've noted certain key audience details below in order to establish a more solid impression of the sample from which results have been derived:

- 21% of respondents described their primary role in their organization as "Technical Architect," 20% described their primary role as "Developer/Engineer," 13% described their primary role as "Consultant/Solutions Architect," and 10% described their primary role as "Developer Team Lead."
- 68% of respondents said they were currently developing "Web applications/Services (SaaS)," 47% said they were developing "Enterprise business applications," 14% said they were developing "Native mobile apps," and 12% said they were developing "Boxed software with updates over the web" and "High-risk software (bugs and failures can mean significant financial loss or loss of life)."
- "Python" (75%) was the most popular language ecosystem used at respondents' companies, followed by "Java" (59%), and "JavaScript (client-side)" (53%).
- Regarding responses on the primary language respondents use at work, 29% of respondents each said that "Java" or "Python" was the main language they used, followed very distantly by "C#" (9%) and "JavaScript (client-side)" (7%).
- On average, respondents said they had 18.39 years of experience as an IT professional, with a median of 20 years.
- 24% of respondents worked at organizations with fewer than 100 employees; 34% of respondents worked at organizations with 100-999 employees; and 40% of respondents worked at organizations with 1,000+ employees.

In this report, we review some of our key research findings. Many secondary findings of interest are not included here.

Research Target One: ETL/ELT Methods and Solutions

Motivations:

1. Extract, transform, and load processes (ETL or ELT, depending on the order of transformation and loading) allow datasets to have value, to be something more than meaningless rows and columns or nodes and edges, to provide practical utility to systems and users. There are a wide variety of methods that can be used to perform ETL, which can range in

complexity, resource intensity, and usefulness, and choosing an appropriate method often depends on factors such as the characteristics of the original dataset(s), available budget, and desired application of the transformed data. We wanted to understand how often and under what circumstances various approaches to ETL/ELT are being used.

2. As the value of usable data increases, and as more and more organizations look for ways to capitalize on this value, the number of tools and solutions designed to help with ETL, and the adoption of those tools, is bound to increase as well. We wanted to know more about plans for the adoption of ETL solutions and what tools were being used to handle ETL for dashboards and analysis.

PREVALENCE OF ETL/ELT APPROACHES

ETL and data utility are inextricably linked. To prepare these research findings, we manually imported response data as plain text, comma-separated values from our cloud-based survey solution, loaded the data onto a local storage drive, and transformed the data through ad hoc spreadsheet functions and file manipulation (this, of course, ignores many ETL processes undertaken by the survey tool itself in the collection and redistribution of response data).

This process works reasonably well for its use case, but it would be wildly inappropriate for use in, say, a dashboard visualizing real-time user activity or a cloud server adjusting resources based on load. To find out how often respondents were using different ETL approaches for their needs, we asked:

> *How often do you extract, transform, and load data using each of the following approaches? Rank from most frequently used (1) to least frequently used (9).*

Results:

Table 1

FREQUENCY RANKING OF ETL/ELT APPROACHES: 2023 vs. 2022

Approach	2023 (*n*=66)			2022 (*n*=228)			Rank Change
	Rank	Score	*n*=	Rank	Score	*n*=	
Ad hoc batch scripts run on a schedule	1	298	44	2	1,129	204	+1
Manual database imports/exports (csv, text, etc.)	2	291	46	1	1,137	213	-1
Reusable but manually run SQL scripts	3	257	42	3	1,117	214	0
Specialized ETL tool	4	251	42	6	1,026	205	+2
Specialized data pipelining tool	5	238	39	4	1,106	214	-1
Ad hoc batch scripts run on demand (manually initiated)	6	233	39	8	1,004	200	+2
Custom-built middleware	7	205	36	7	1,022	228	0
General-purpose integration tool	8	186	35	5	1,047	218	-3
Ad hoc streaming scripts	9	139	31	9	998	211	0

Additionally, we looked at results segmented by responses to the following demographic questions:

> *What is your primary programming language at work?*
>
> and
>
> *What is the size of your organization in terms of employees?*

SEE RESULTS ON NEXT PAGE

Table 2

FREQUENCY RANKING OF ETL/ELT APPROACHES: JAVA vs. PYTHON

Approach	Java (*n*=22)			Python (*n*=22)			Overall (*n*=66)	
	Score	Rank	*n*=	Score	Rank	*n*=	Score	Rank
Ad hoc batch scripts run on a schedule	96	1	14	74	5	12	298	1
Manual database imports/exports (csv, text, etc.)	64	4	11	71	6	13	291	2
Reusable but manually run SQL scripts	54	8	10	81	3	13	257	3
Specialized ETL tool	84	2	13	81	2	13	251	4
Specialized data pipelining tool	63	5	11	83	1	13	238	5
Ad hoc batch scripts run on demand (manually initiated)	62	6	10	75	4	12	233	6
Custom-built middleware	56	7	10	64	7	12	205	7
General-purpose integration tool	68	3	11	54	8	12	186	8
Ad hoc streaming scripts	48	9	10	37	9	9	139	9

Table 3

FREQUENCY RANKING OF ETL/ELT APPROACHES BY ORGANIZATION SIZE

Approach	1-99 (*n*=14)			100-999 (*n*=20)			1,000+ (*n*=29)			Overall (*n*=66)	
	Rank	Score	*n*=	Rank	Score	*n*=	Rank	Score	*n*=	Rank	Score
Ad hoc batch scripts run on a schedule	4	70	10	1	109	15	3	113	18	1	298
Manual database imports/exports (csv, text, etc.)	2	75	12	2	87	14	4	111	18	2	291
Reusable but manually run SQL scripts	1	81	13	4	79	12	6	90	16	3	257
Specialized ETL tool	9	38	10	3	83	13	1	129	18	4	251
Specialized data pipelining tool	7	41	9	7	69	10	2	126	19	5	238
Ad hoc batch scripts run on demand (manually initiated)	3	74	12	6	71	12	8	83	14	6	233
Custom-built middleware	6	41	9	5	78	11	7	83	15	7	205
General-purpose integration tool	8	40	9	9	33	7	5	96	17	8	186
Ad hoc streaming scripts	5	44	8	8	37	9	9	46	12	9	139

Observations:

1. The highest rated approaches to ETL/ELT with regards to frequency of use were "Ad hoc batch scripts run on a schedule," "Manual database imports/exports (csv, text, etc.)," and "Reusable but manually run SQL scripts." These results closely align with the responses we saw from 2022's Data Pipelines survey, in which these three approaches also comprised the top placements.

   This year, however, scheduled ad hoc batch scripts narrowly surpassed manual database imports and exports as the top ranked approach, whereas last year the opposite case was true.

2. For the most part, the overall rank of each approach either remained the same as its rank in 2022, or varied by only one place. The exceptions here were the usage of "Specialized ETL tool[s]" and "Ad hoc batch scripts run on demand (manually initiated)" — each of which with an overall rank two places higher than in 2022 — and "General-purpose integration tool[s]," which had an overall rank three places lower than in 2022.

   Generally, this indicates little to no growth towards more "mature" data pipelining models. Much like last year, these rankings imply that intentional, carefully constructed data pipeline systems tend to take a back seat to more free-form,

as-needed data wrangling. This is not necessarily a negative observation; while data utilization seems to be growing more important to businesses as time goes on, some organizations simply do not need the most sophisticated data solutions — or at least, the resource investment required for these solutions would outweigh any benefits.

As data value increases overall and barriers to entry of more mature data solutions diminish, adoption of more advanced or specialized approaches will almost certainly overtake the simpler, more manual ones, but that shift is not yet evident based solely on our year-over-year results.

3. The differences in ETL/ELT approaches based on business needs seemed much clearer when segmenting samples based on organizational demographic factors. For example, respondents who said their primary programming language at work was Java ranked their "General-purpose integration tool" usage frequency third, five places ahead of its eighth-place ranking overall. These Java-focused respondents also ranked "Reusable but manually run SQL scripts" in eighth place, compared to its overall ranking of third place.

   On the other hand, respondents who said their primary programming language at work was Python — a language frequently used for data pipelining and ETL/ELT — ranked "Specialized data pipelining tool" first, well ahead of its ranking of fifth place overall, while ranking "Ad hoc batch scripts run on a schedule" (fifth) and "Manual database imports/exports (csv, text, etc.)" (sixth) much lower than their overall ranks (first and second, respectively).

4. Organization size also impacted the frequency rankings of ETL/ELT approaches. Respondents at smaller organizations (< 100 employees) ranked their usage of "Ad hoc batch scripts run on a schedule" (fourth) and "Specialized ETL tool" (ninth) much lower than respondents overall (first and forth, respectively), while they ranked both "Ad hoc batch scripts run on demand (manually initiated)" (third) and "Ad hoc streaming scripts" (fifth) several places higher than respondents overall.

   Respondents at the largest organizations (1,000+ employees) unsurprisingly ranked specialized or third-party tools — "Specialized ETL tool[s]" (first), "Specialized data pipelining tool[s]" (second), and "General-purpose integration tool[s]" (fifth) — higher than respondents overall (fourth, fifth, and eighth, respectively). These respondents also ranked "Reusable but manually run SQL scripts" (sixth) lower than the approach's third-place overall ranking.

USE OF ETL SOLUTIONS

Advancements in data pipeline technology and advantages of more robust ETL approaches are likely to drive ETL solution adoption. In order to try to determine the current state of this adoption, we asked the following questions:

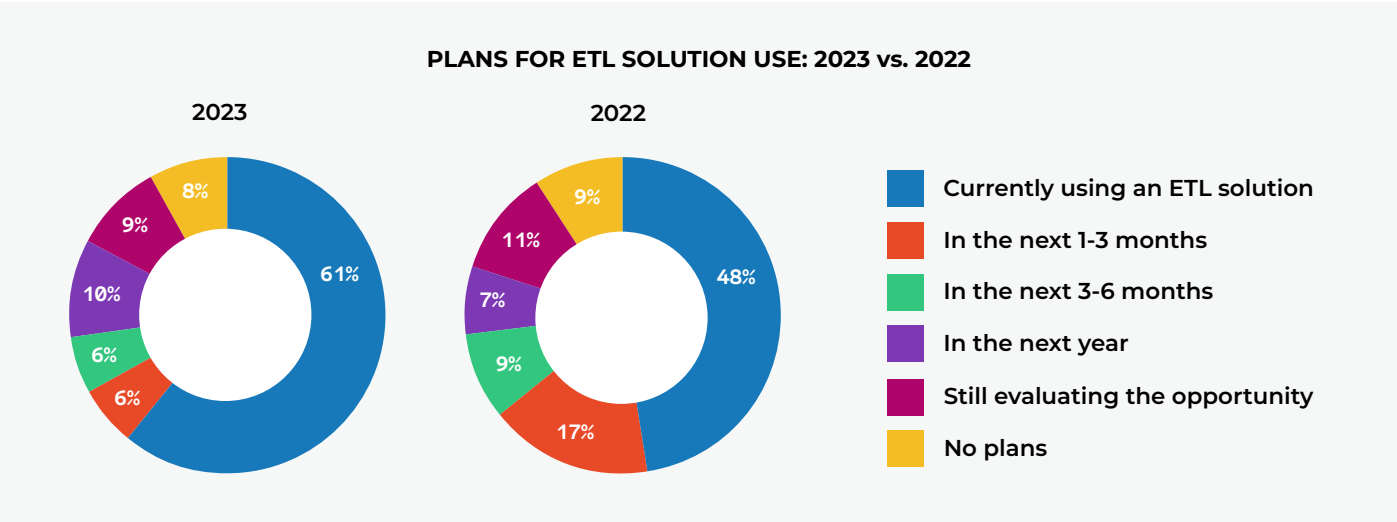
When are you planning to use an ETL solution?

and

Does your organization use a third-party ETL/ELT solution to collect data for dashboards and analyses?

Results (*n*=79 in 2023 and *n*=282 in 2022):

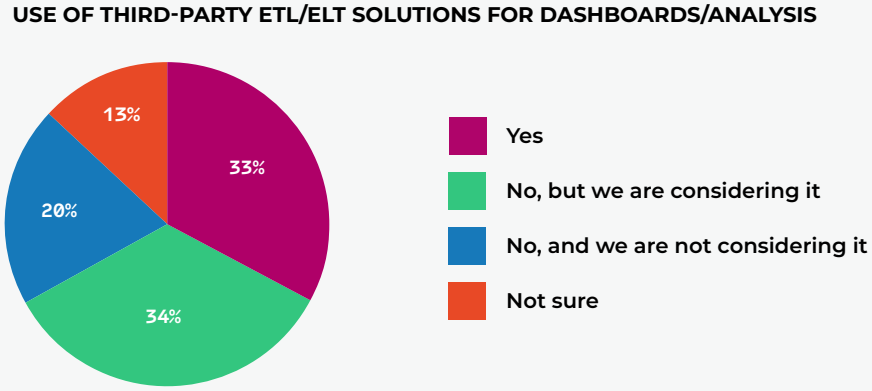
Figure 1



Results (*n*=79):

SEE FIGURE 2 ON NEXT PAGE

Figure 2

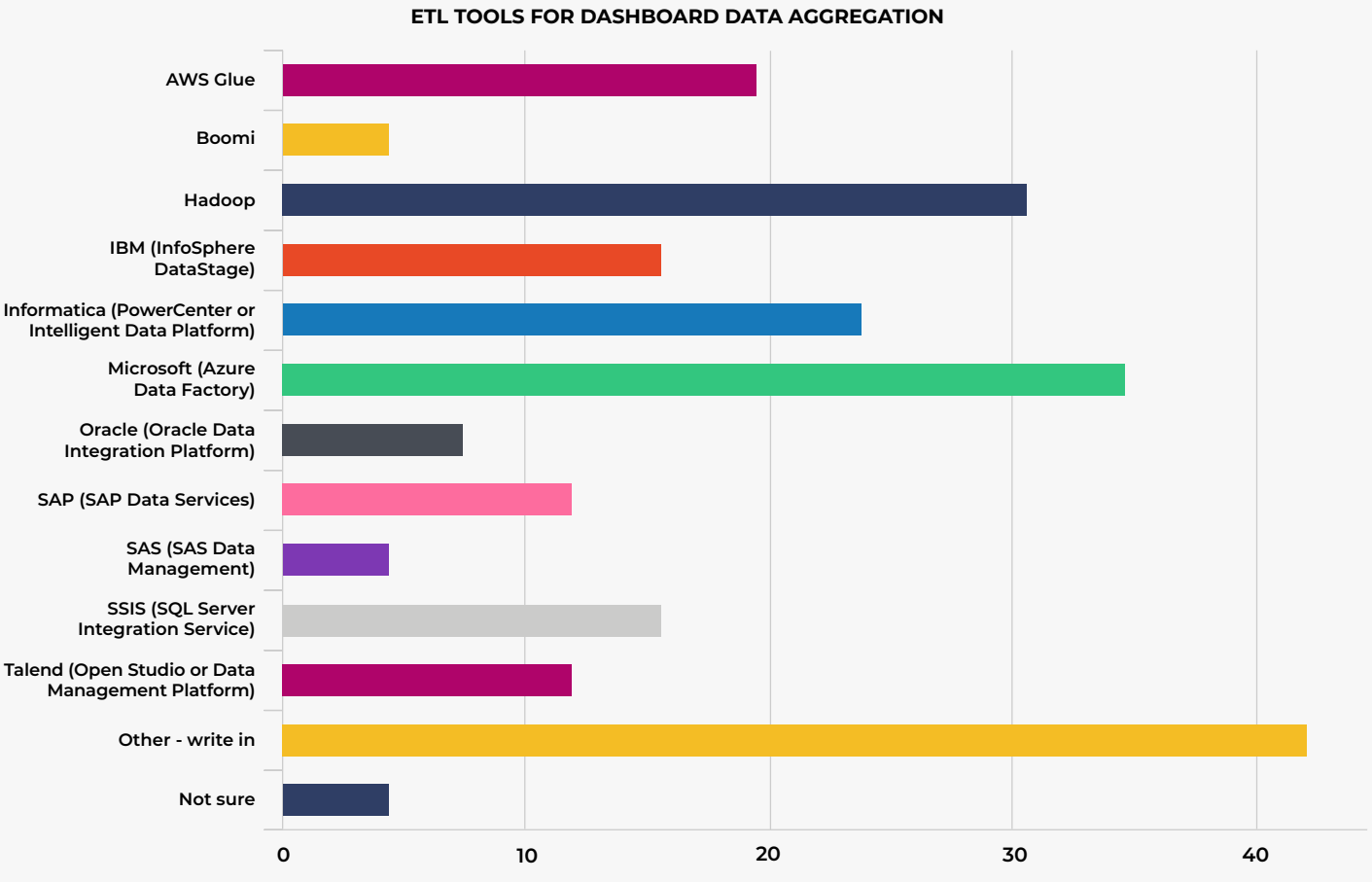


To find out more about which specific ETL tools were being used, we also asked respondents who answered "Yes" to the latter of the two questions above:

Which of the following ETL tools does your organization use to pull together data for your dashboards?

Results (*n*=26):

Figure 3



Observations:

1. More than half of respondents (61%) said they were "Currently using an ETL solution," significantly higher than the rate of respondents answering the same in 2022. This increase in response rate coincided with a significant decrease in respondents who said they planned to use an ETL solution "In the next 1-3 months" (6% in 2023 vs. 17% in 2022), while no

significant change was recorded for responses over this three-month period. This may indicate increased ETL solution adoption among individuals and organizations that already recognize these solutions' immediate benefit, whereas those who are "on the fence" about or see no benefit in ETL solutions continue to hold out.

2. About one-third of respondents (33%) said their organization was currently using a third-party ETL/ELT solution for dashboard/analysis data collection, while about another one-third (34%) said their organization was considering it. The remaining one-third of respondents (33%) were split between those who said their organization was not considering third-party ETL solutions for dashboards and analysis (20%) and those who said they were not sure (13%).
3. Of the respondents at organizations currently using third-party ETL/ELT solutions, the most popular tools used were "Microsoft (Azure Data Factory)" (35%), "Hadoop" (31%), "Informatica (PowerCenter or Intelligent Data Platform)" (23%), and "AWS Glue" (19%). Furthermore, 35% of respondents provided write-in options to the question. Overall, the spread of both provided and suggested (write-in) responses seems to demonstrate ample room still for front runners to emerge within the space.

Research Target Two: Data Pipeline and Storage Practices

Motivations:

1. Organizations have a lot of options when it comes to choosing solutions for their data pipeline and storage needs, and the pool of potential solutions continues to grow as organizations' needs vary more and more. We tried to find out more about what types of data pipeline and storage solutions different organizations were using.
2. As the value and abundance of data continues to rise, local storage or simple cloud servers are no longer enough to fulfill the data requirements of many organizations. Data warehouses, lakes, and lakehouses have evolved from these simpler storage solutions to keep up with the growth of data. We looked at how much organizations were using these more modern data storage options.
3. "Big data," as we know, doesn't describe only an increase in data volume in the contemporary software landscape — other "V" nouns may also bestow the adjective "big" to one's data; "velocity" of data is one such attribute. Real-time data ingestion and analysis has become a benefit, if not a necessity, to many organizations, which means that dealing with data velocity is growing ever important. We wanted to know how many organizations depend on real-time analytics.

TYPES OF DATA PIPELINE AND STORAGE SOLUTIONS

Data pipeline and storage solutions come in a variety of shapes and sizes, each with its own list of characteristics. We focused on three main types of pipeline/storage solution — on-premises, cloud-based, and open source — and asked respondents:

What type of data pipeline and storage solutions are established at your organization?

Results (*n*=76):

Figure 4

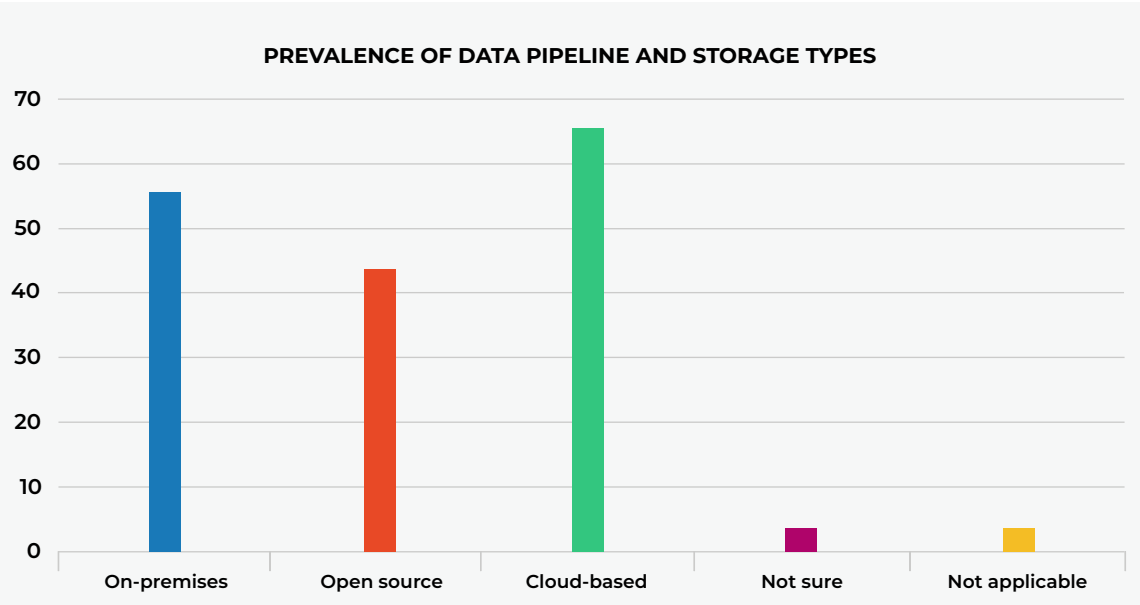


Table 4

PREVALENCE OF DATA PIPELINE AND STORAGE TYPES BY ORGANIZATION SIZE*				
Organization Size	**On-Premises**	**Cloud-Based**	**On-Prem and Cloud**	**Open Source**
1-99	50%	33%	17%	50%
100-999	58%	77%	35%	35%
1,000+	58%	74%	42%	45%
Overall	56%	65%	33%	43%

**Note: % of row*

Observations:

1. Over half of respondents said that their organizations used "Cloud-based" (66%) or "On-premises" (55%) data pipeline and storage solutions, with one-third of respondents saying their organization used both (33%). 43% of respondents said that their organization used open-source data pipeline and storage solutions.
2. Respondents at smaller organizations were slightly less likely to have on-premises data pipeline/storage solutions (50%) than mid-sized and large organizations (58% each), but much less likely to use cloud-based solutions — 33% at small companies vs. 77% at mid-sized and 74% at large companies — as well as moderately less likely to use both — 17% at small companies vs. 35% at mid-sized and 42% at large companies. On the other hand, small organizations were more likely to use open-source data pipeline and storage solutions (50%) than mid-sized (35%) and large (45%) organizations.
3. Respondents who reported that their organization used cloud-based data pipeline solutions were slightly more likely to say they also used open-source solutions (32%) than respondents who said their organizations used on-premises solutions (25%).

DATA WAREHOUSES, DATA LAKES, AND DATA LAKEHOUSES

The rise of big data has led to the need for novel data storage solutions. Data warehouses, data lakes, and data lakehouses have evolved from simple cloud storage in order to provide greater data capabilities with higher volume, velocity, and variety. To see how prevalent each of these storage types are, we asked:

Does your organization keep data in any of the following locations?

Results (*n*=74 and *n*=66, respectively):

Table 5

PREVALENCE OF DATA WAREHOUSES, LAKES, AND LAKEHOUSES				
Storage Location	**Yes**	**No**	**I don't know**	***n*=**
Data warehouse	68%	20%	12%	74
Data lake	46%	32%	22%	69
Data lakehouse	17%	55%	29%	66

Table 6

PREVALENCE OF DATA STORAGE LOCATION BY ORGANIZATION SIZE*			
Organization Size	**Warehouse**	**Lake**	**Lakehouse**
1-99	50%	28%	6%
100-999	73%	42%	8%
1,000+	71%	52%	26%
Overall	67%	43%	15%

**Note: % of row*

Observations:

1. Most respondents said that their organization kept data in a data warehouse (68%), and almost half of respondents said their organization kept data in a data lake (46%), but only 17% of respondents said that their organization kept data in a data lakehouse. Given that the concept of a data lakehouse is relatively new, this current adoption rate is unsurprising. Future analysis will be able to better show whether data lakehouse adoption rates increase to contend with other "big data'' storage solutions.
2. When segmenting these results by respondents' organization size, the following correlations emerged:
   - Data warehouse usage was reported significantly less by respondents at small organizations (50%) than those at mid-sized (73%) and large (71%) organizations.
   - Data lakes were used moderately more by mid-sized organizations (42%) than small organizations (28%), and moderately more by large organizations (52%) than mid-sized.
   - Data lakehouse usage was reported predominantly by respondents at large organizations (26%), well ahead of respondents at small (6%) and mid-sized (8%) companies.

   All of these observations are supported by the reasoning that larger corporations are more likely to need large data solutions, and that they are more likely to have the resources to dedicate (and incentive to dedicate resources) to cutting-edge data solutions.

IMPORTANCE OF REAL-TIME ANALYTICS

Real-time ingestion of non-local data is a relatively new phenomenon in software (or anywhere, for that matter), but the emergence of technologies capable of real-time analytics has been seemingly matched almost one to one with their importance to businesses. We wanted to get a sense of this importance, so we asked respondents:

> *How important are real-time analytics for your organization's operations?*

Results (*n*=82):

Figure 5

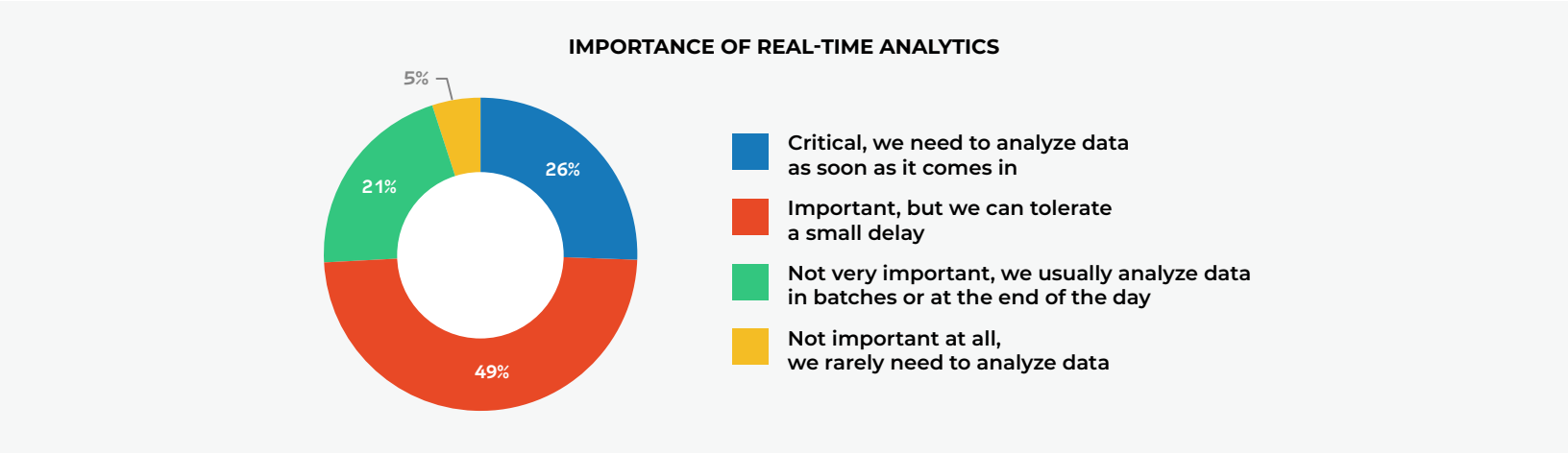


Table 7

IMPORTANCE OF REAL-TIME ANALYTICS vs. DATA STORAGE LOCATION TYPES*

	Warehouse	Lake	Lakehouse
Critical, we need to analyze data as soon as it comes in	43%	33%	24%
Important, but we can tolerate a small delay	78%	48%	10%
Not very important, we usually analyze data in batches or at the end of the day	53%	35%	12%
Not important at all, we rarely need to analyze data	25%	0%	0%
Overall	61%	39%	13%

**Note: % of row*

Observations:

1. A large majority of respondents indicated that their organization at least somewhat relied on real-time analytics:
   - Nearly half (49%) reported real-time analytics as being "Important, but we can tolerate a small delay."
   - About another quarter (26%) said real-time analytics are "Critical, we need to analyze data as soon as it comes in."
   - 21% described real-time analytics as "Not very important, we usually analyze data in batches or at the end of the day."
   - Only 5% claimed their organization considers real-time analytics "Not important at all, we rarely need to analyze data."
2. Respondents at organizations for which real-time analytics is considered "important" (but not "critical") were significantly more likely to report that their organizations use data warehouses (78% vs. 61% overall) and data lakes (48% vs. 39% overall) than those at other organizations, including respondents at organizations where real-time analytics are considered "critical." However, respondents who said their organizations consider real-time analytics as "critical" were about twice as likely as other respondents to report that their organizations use data lakehouses (24% vs. 13% overall), aligning with one of the purported benefits of data lakehouses — better support for real-time data ingestion.

Research Target Three: Impacts of Data Pipelines in the Software Profession

Motivations:

1. Learning new skills is a given in a software development job; the rapid rate of technological change and overall increased complexity of software systems make learning new languages, frameworks, or tools a requirement in the field. We wanted to see how much data pipelines specifically were necessitating the acquisition of new skills.
2. As is the case with many facets of software development, there are times in which it makes business sense to outsource data pipeline work to an external team. We wanted to look at how many organizations were using external partners to handle their data pipeline needs and how many were keeping data pipeline creation and maintenance in house.

LEARNING NEW SKILLS TO IMPLEMENT DATA PIPELINES

Given the escalation of importance put upon data collection and utilization in the recent past, it seems a reasonable notion that developers and other software professionals would need to augment their skills and learn new tools or languages to meet the growing demand of data pipeline expertise. To evaluate how reasonable that notion is, we asked:

Have you ever learned a new technology (language, library, platform) in order to implement a data pipeline?

Results (*n*=87 and *n*=74, respectively):

Figure 6

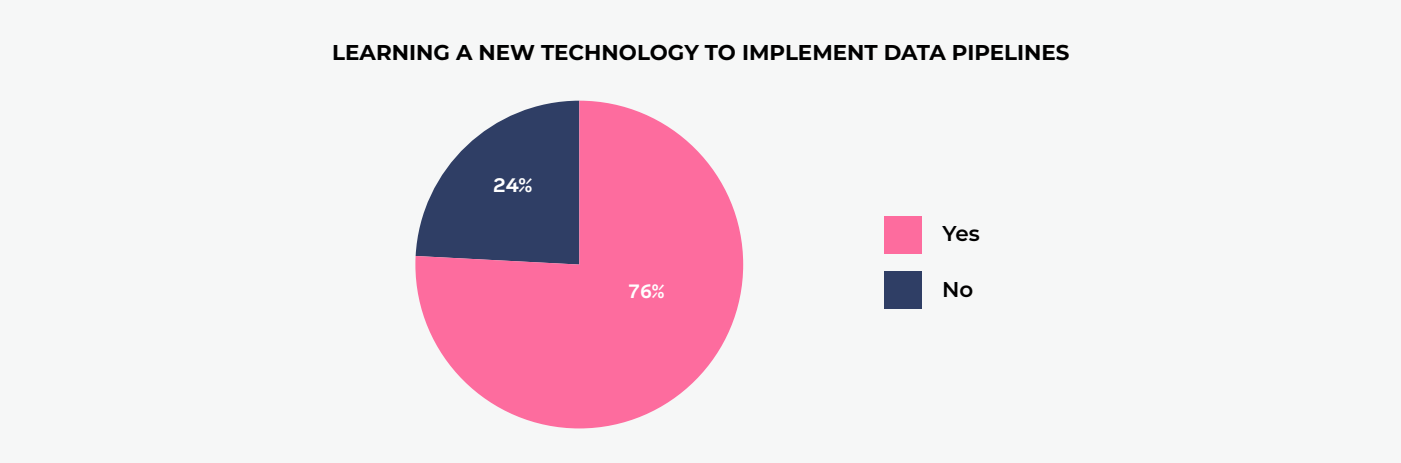


Table 8

NEW TECH LEARNED TO IMPLEMENT DATA PIPELINES BY ORGANIZATION SIZE*		
Organization Size	**Yes**	**No**
1-99	50%	28%
100-999	73%	42%
1,000+	71%	52%

**Note: % of row*

Observations:

1. Over three-quarters of respondents (76%) said that they had learned a new technology in order to implement a data pipeline, a significant increase over respondents who said the same in 2022 (60%).
2. Learning new skills for data pipelining was correlated to organization size. Respondents at organizations with 1,000+ employees were significantly more likely than others to say they had learned a new technology to implement data pipelines (87%), followed by respondents at organizations with 100-999 employees (81%). Respondents at organizations with < 100 employees were least likely to learn new skills for data pipeline implementation (65%).

   This seems to suggest that the increased resources and higher likelihood of needing more robust data pipelines at larger organizations outweighs the higher likelihood that large orgs would have dedicated teams for handling data pipelines. That is to say, either data pipeline implementation work has not grown to the point where dedicated staff is generally necessary, even at large organizations, or — seemingly more likely — the need for data pipeline skills and knowledge expands beyond teams dedicated to them and are either critical or significantly useful across departments.
3. Like last year, Python (15%) was the most commonly reported new technology learned to implement a data pipeline, with Apache Spark (11%) coming in second. This year, Apache Airflow (11%) was tied with Spark for second place. Apache Kafka (9%), Azure Data Factory (8%), and Microsoft SQL Server Integration Services (8%) were also mentioned frequently.*

**Note: This data was collected from open-ended write-in responses from respondents who reported they had learned a new technology to implement a data pipeline and, therefore, may be less accurate than results for questions where answer options were given explicitly.*

INTERNAL vs. EXTERNAL RESPONSIBILITY FOR DATA PIPELINES

The decision to build and manage data pipelines in house vs. externally depends on a variety of factors, and each option has its advantages and disadvantages. Internal responsibility may lead to more control over pipeline management at the expense of budget and staffing resources, for example. To see where organizations lean regarding this decision, we asked respondents:

> *Does your organization have an internal team or an external vendor/partner responsible for building and maintaining data pipelines?*

Results (*n*=82 and *n*=75, respectively):

Figure 7

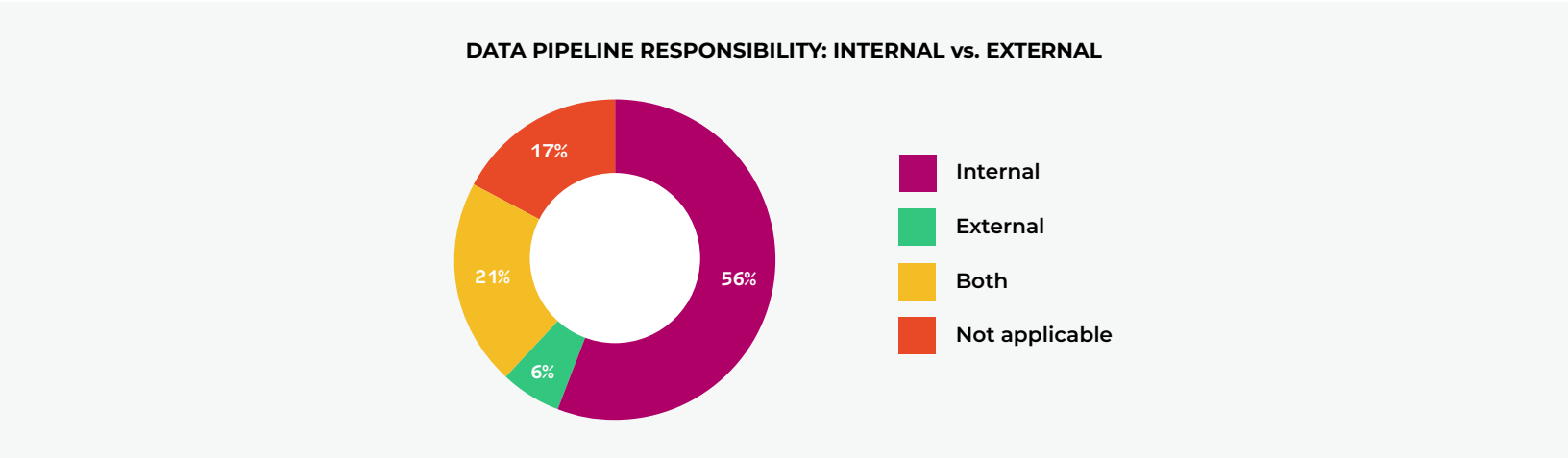


Table 9

INTERNAL vs. EXTERNAL RESPONSIBILITY FOR DATA PIPELINES BY ORGANIZATION SIZE*

Organization Size	Internal	External	Both	Not Applicable
1-99	39%	11%	11%	39%
100-999	65%	0%	23%	12%
1,000+	65%	6%	26%	3%
Overall	59%	5%	21%	15%

**Note: % of row*

Observations:

1. Very few respondents (5%) indicated that their organization relied solely on external vendors or partners to manage building and maintaining their data pipelines. Most reported that their data pipelines were managed internally (56%) or — to a lesser extent — managed by a combination of both internal teams and external partners (21%).
2. Respondents at small organizations were significantly less likely to report internal responsibility for data pipelines (39%) than mid-sized (65%) or large (65%) organizations, and less likely to report combined internal/external responsibility (11% for small organizations vs. 23% for mid-sized and 26% for large organizations). Respondents at small organizations were also much more likely to indicate that their organization had no internal or external teams building or managing data pipelines by answering "Not applicable" (39%), compared to mid-sized (12%) and large (3%) organizations.

    Finally, respondents at small organizations were more likely to report their company using external vendors or partners to build or maintain their data pipelines (11%) than larger organizations (0% for mid-sized organizations and 6% for large organizations), even though this rate still accounts for relatively few respondents overall.

Future Research

Our analysis here only touched the surface of the available data, and we will look to refine and expand our Data Pipelines survey as we produce further Trend Reports. Some of the topics we didn't get to in this report, but were incorporated in our survey, include:

- Explicit data pipeline modeling
- Data literacy
- Manual implementations of sorting and aggregation algorithms
- Concerns about slowly changing data
- Preferences for data isomorphism
- Democratization of data engineering

Please contact publications@dzone.com if you would like to discuss any of our findings or supplementary data.

G. Ryan Spain, Freelance Software Engineer, former Engineer & Editor at DZone
@grspain on DZone, GitHub, GitLab | gryanspain.com

G. Ryan Spain lives on a beautiful two-acre farm in McCalla, Alabama with his lovely wife and adorable dog. He is a polyglot software engineer with an MFA in poetry, a die-hard Emacs fan and Linux user, a lover of The Legend of Zelda, a journeyman data scientist, and a home cooking enthusiast. When he isn't programming, he can often be found playing Animal Crossing: New Horizons with a glass of red wine or a cold beer.

PARTNER CASE STUDY

Case Study: Texas Rangers

How Low-Code Data Engineering and Analytics Deliver a Competitive Edge

Challenge

Major League Baseball has become highly reliant on data and analytics for a competitive edge; unfortunately for the Texas Rangers baseball club, their data journey was holding them back. Their baseball operations team went from a legacy on-premises architecture that was impossible to scale to a cloud data warehouse with massive infrastructure costs, maintenance needs, and data staleness. Neither architecture could unify their siloed, disparate data sources, resulting in a non-cohesive array of data products.

Further, the operations team's tooling required familiarity coding with Apache Spark, for which there was limited internal experience. Their existing tooling also did not offer collaborative capabilities, which meant much of the analytics work was done in a vacuum. This delayed delivery of data products, such as data pipelines, trend reports, and visualizations, severely impacting player and team performance analytics.

Solution

The Texas Rangers baseball operations team looked to Prophecy as their data engineering solution on top of their Databricks Lakehouse. This new data architecture and the decision to implement Prophecy has provided massive benefits for the club:

- Prophecy, a visual, low-code solution, enabled them to quickly and easily develop, deploy, and manage their data pipelines at scale.
- The baseball operations team can seamlessly connect to any data source of their choice and rapidly onboard and incorporate new player data to further their analytics needs.
- Even though the data team did not have deep experience in coding with Spark, they can still deliver high-quality data products without the need for Spark expertise.

With the Texas Rangers now exploring advanced AI and large language model (LLM) use cases, Prophecy can fully support the ambitions of the baseball operations team through the Prophecy Generative AI platform.

Results

Thanks to their deployment of Prophecy, the results have been staggering. The time savings of deploying new data pipelines decreased from requiring multiple sprints, taking as much as two weeks, to a single sprint where they can move their data pipelines into production within two days. This improved efficiency allows their analysts to create reports and services — and glean new statistics and metrics for evaluating players — faster than ever before, resulting in:

- **3x** more analysts and developers being able to create production-ready ETL pipelines
- A **7x** increase in velocity of producing new data products
- **10x** saved working days per month for the baseball operations team

COMPANY
Texas Rangers

COMPANY SIZE
234 employees

INDUSTRY
Media and Entertainment

PRODUCTS USED
AWS, Databricks Lakehouse, Apache Airflow, Statcast, Tableau, Prophecy

PRIMARY OUTCOME
Rapid migration of legacy ETL workloads to the cloud, while enabling the visual development of high quality data products by their entire data team, regardless of skill set.

"Prophecy has created a lot of opportunity, allowing us to increase the velocity and impact of our data products while maintaining a high level of quality within our engineering team."

— **Alexander Booth**,
Assistant Director of R&D,
Texas Rangers

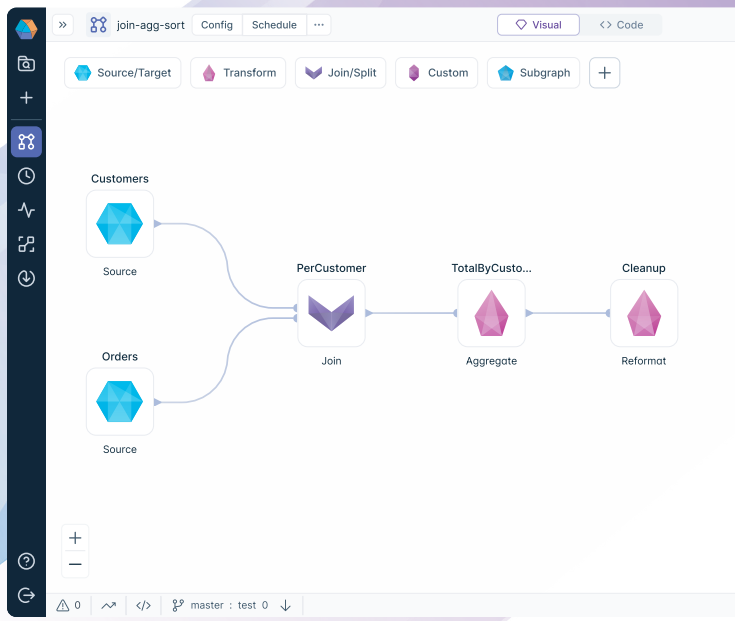
CREATED IN PARTNERSHIP WITH

Prophecy

Prophecy

Low-code data transformation

Enable all data users to transform raw data into reliable, analytics-ready data using visual pipelines.



- **Low-code for all** - Empower every data user in your business to transform data like expert data engineers.
- **Trusted data** - Enjoy visual development that's based on software best practices and ensures data quality control for all data users.
- **Open and extensible** - Turn your data pipelines into open-source Spark or SQL code where you can add new visual components to standardize your operations.

Develop

Build low-code data pipelines that generate clean Spark or SQL code

Manage

Gain control with a metadata catalog, data quality, search and data lineage

Deploy

Reliably move to production with versions, tests, CI, CD and scheduling

Simplifying data transformation for all data teams.

Visit us at **prophecy.io** to start a free trial today

CONTRIBUTOR INSIGHTS

A Guide to Data-Driven Design and Architecture

Key Principles, Patterns, and Considerations

By Boris Zaikin, Lead Solution Architect at CloudAstro GmbH

Data-driven design is a game changer. It uses real data to shape designs, ensuring products match user needs and deliver user-friendly experiences. This approach fosters constant improvement through data feedback and informed decision-making for better results. In this article, we will explore the importance of data-driven design patterns and principles, and we will look at an example of how the data-driven approach works with artificial intelligence (AI) and machine learning (ML) model development.

Importance of the Data-Driven Design

Data-driven design is crucial as it uses real data to inform design decisions. This approach ensures that designs are tailored to user needs, resulting in more effective and user-friendly products. It also enables continuous improvement through data feedback and supports informed decision-making for better outcomes.

Data-driven design includes the following:

- **Data visualization** – Aids designers in comprehending trends, patterns, and issues, thus leading to effective design solutions.
- **User-centricity** – Data-driven design begins with understanding users deeply. Gathering data about user behavior, preferences, and challenges enables designers to create solutions that precisely meet user needs.
- **Iterative process** – Design choices are continuously improved through data feedback. This iterative method ensures designs adapt and align with user expectations as time goes on.
- **Measurable outcomes** – Data-driven design targets measurable achievements, like enhanced user engagement, conversion rates, and satisfaction.

This is a theory, but let's reinforce it with good examples of products based on data-driven design:

- **Netflix** uses data-driven design to predict what content their customers will enjoy. They analyze daily plays, subscriber ratings, and searches, ensuring their offerings match user preferences and trends.
- **Uber** uses data-driven design by collecting and analyzing vast amounts of data from rides, locations, and user behavior. This helps them optimize routes, estimate fares, and enhance user experiences. Uber continually improves its services by leveraging data insights based on real-world usage patterns.
- **Waze** uses data-driven design by analyzing real-time GPS data from drivers to provide accurate traffic updates and optimal route recommendations. This data-driven approach ensures users have the most up-to-date and efficient navigation experience based on the current road conditions and user behavior.

Common Data-Driven Architectural Principles and Patterns

Before we jump into data-driven architectural patterns, let's reveal what data-driven architecture and its fundamental principles are.

DATA-DRIVEN ARCHITECTURAL PRINCIPLES

Data-driven architecture involves designing and organizing systems, applications, and infrastructure with a central focus on data as a core element. Within this architectural framework, decisions concerning system design, scalability, processes, and interactions are guided by insights and requirements derived from data.

Fundamental principles of data-driven architecture include:

- **Data-centric design** – Data is at the core of design decisions, influencing how components interact, how data is processed, and how insights are extracted.

- **Real-time processing** – Data-driven architectures often involve real-time or near real-time data processing to enable quick insights and actions.
- **Integration of AI and ML** – The architecture may incorporate AI and ML components to extract deeper insights from data.
- **Event-driven approach** – Event-driven architecture, where components communicate through events, is often used to manage data flows and interactions.

DATA-DRIVEN ARCHITECTURAL PATTERNS

Now that we know the key principles, let's look into data-driven architecture patterns. Distributed data architecture patterns include the data lakehouse, data mesh, data fabric, and data cloud.

DATA LAKEHOUSE

Data lakehouse allows organizations to store, manage, and analyze large volumes of structured and unstructured data in one unified platform. Data lakehouse architecture provides the scalability and flexibility of data lakes, the data processing capabilities, and the query performance of data warehouses. This concept is perfectly implemented in Delta Lake. Delta Lake is an extension of Apache Spark that adds reliability and performance optimizations to data lakes.

DATA MESH

The data mesh pattern treats data like a product and sets up a system where different teams can easily manage their data areas. The data mesh concept is similar to how microservices work in development. Each part operates on its own, but they all collaborate to make the whole product or service of the organization. Companies usually use conceptual data modeling to define their domains while working toward this goal.

DATA FABRIC

Data fabric is an approach that creates a unified, interconnected system for managing and sharing data across an organization. It integrates data from various sources, making it easily accessible and usable while ensuring consistency and security. A good example of a solution that implements data fabric is Apache NiFi. It is an easy-to-use data integration and data flow tool that enables the automation of data movement between different systems.

DATA CLOUD

Data cloud provides a single and adaptable way to access and use data from different sources, boosting teamwork and informed choices. These solutions offer tools for combining, processing, and analyzing data, empowering businesses to leverage their data's potential, no matter where it's stored. Presto exemplifies an open-source solution for building a data cloud ecosystem. Serving as a distributed SQL query engine, it empowers users to retrieve information from diverse data sources such as cloud storage systems, relational databases, and beyond.

Now we know what data-driven design is, including its concepts and patterns. Let's have a look at the pros and cons of this approach.

Pros and Cons of Data-Driven Design

It's important to know the strong and weak areas of the particular approach, as it allows us to choose the most appropriate approach for our architecture and product. Here, I gathered some pros and cons of data-driven architecture:

Table 1

PROS AND CONS OF DATA-DRIVEN DESIGN	
Pros	**Cons**
Personalized experiences: Data-driven architecture supports personalized user experiences by tailoring services and content based on individual preferences.	**Privacy concerns**: Handling large amounts of data raises privacy and security concerns, requiring robust measures to protect sensitive information.
Better customer understanding: Data-driven architecture provides deeper insights into customer needs and behaviors, allowing businesses to enhance customer engagement.	**Complex implementation**: Implementing data-driven architecture can be complex and resource-intensive, demanding specialized skills and technologies.
Informed decision-making: Data-driven architecture enables informed and data-backed decision-making, leading to more accurate and effective choices.	**Dependency on data availability**: The effectiveness of data-driven decisions relies on the availability and accuracy of data, leading to potential challenges during data downtimes.

Data-Driven Approach in ML Model Development and AI

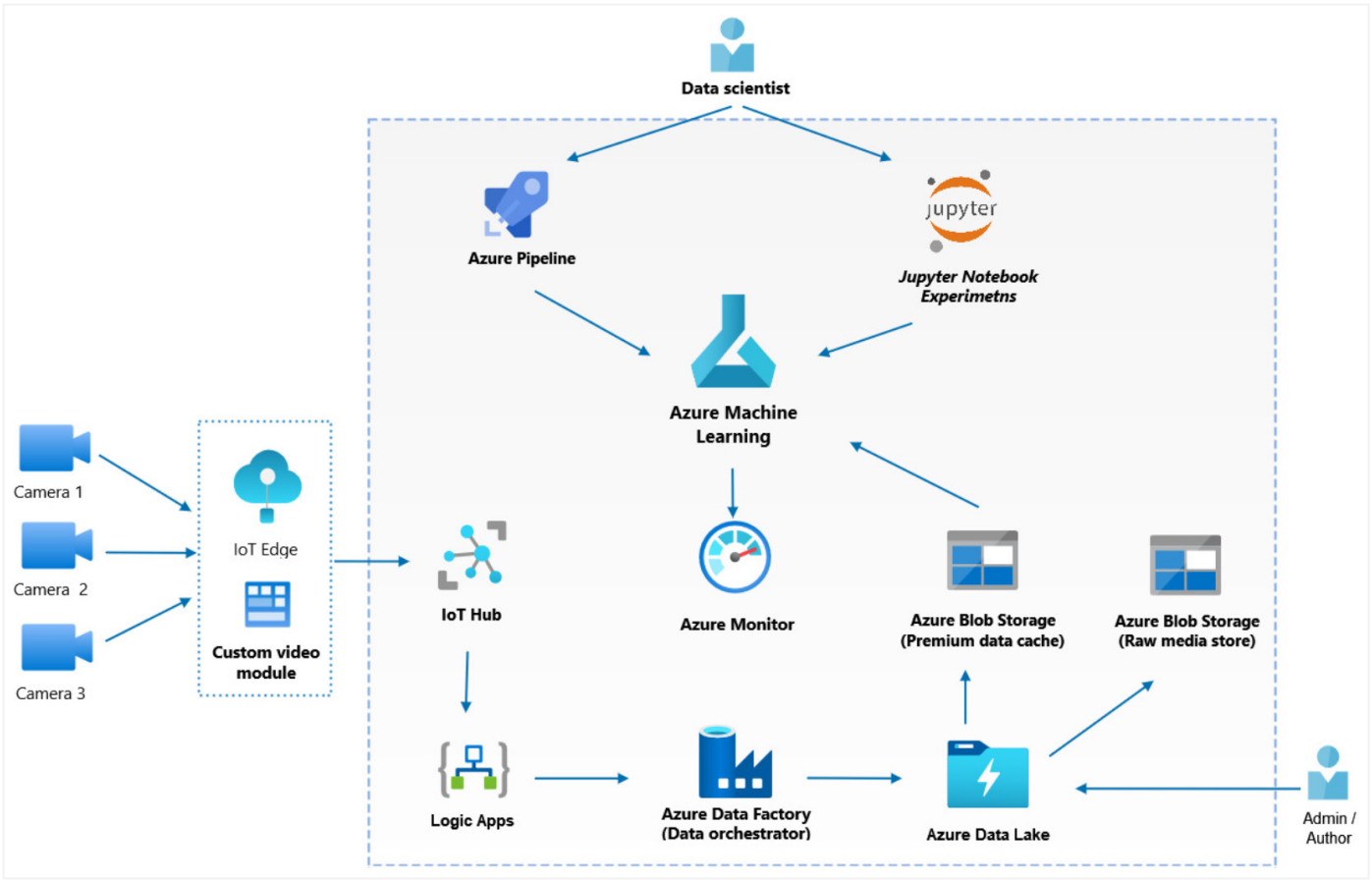
A data-driven approach in ML model development involves placing a strong emphasis on the quality, quantity, and diversity of the data used to train, validate, and fine-tune ML models. A data-driven approach involves understanding the problem domain, identifying potential data sources, and gathering sufficient data to cover different scenarios. Data-driven decisions help determine the optimal hyperparameters for a model, leading to improved performance and generalization.

Let's look at the example of the data-driven architecture based on AI/ML model development. The architecture represents the factory alerting system. The factory has cameras that shoot short video clips and photos and send them for analysis to our system. Our system has to react quickly if there is an incident.

Below, we share an example of data-driven architecture using Azure Machine Learning, Data Lake, and Data Factory. This is only an example, and there are a multitude of tools out there that can leverage data-driven design patterns.

- The IoT Edge custom module captures real-time video streams, divides them into frames, and forwards results and metadata to Azure IoT Hub.
- The Azure Logic App watches IoT Hub for incident messages, sending SMS and email alerts, relaying video fragments, and inferencing results to Azure Data Factory. It orchestrates the process by fetching raw video files from Azure Logic App, splitting them into frames, converting inferencing results to labels, and uploading data to Azure Blob Storage (the ML data repository).
- Azure Machine Learning begins model training, validating data from the ML data store, and copying required datasets to premium blob storage. Using the dataset cached in premium storage, Azure Machine Learning trains, validates model performance, scores against the new model, and registers it in the Azure Machine Learning registry.
- Once the new ML inferencing module is ready, Azure Pipelines deploys the module container from Container Registry to the IoT Edge module within IoT Hub, updating the IoT Edge device with the updated ML inferencing module.

Figure 1: Smart alerting system with data-driven architecture



Conclusion

In this article, we dove into data-driven design concepts and explored how they merge with AI and ML model development. Data-driven design uses insights to shape designs for better user experiences, employing iterative processes, data visualization, and measurable outcomes. We've seen real-world examples like Netflix using data to predict content preferences and Uber optimizing routes via user data. Data-driven architecture, encompassing patterns like data lakehouse and data mesh, orchestrates data-driven solutions. Lastly, our factory alerting system example showcases how AI, ML, and data orchestrate an efficient incident response. A data-driven approach empowers innovation, intelligent decisions, and seamless user experiences in the tech landscape.

Boris Zaikin, Lead Solution Architect at CloudAstro GmbH

@borisza on DZone | @boris-zaikin on LinkedIn | boriszaikin.com

I'm a certified senior software and cloud architect with solid experience designing and developing complex solutions based on the Azure, Google, and AWS clouds. I have expertise in building distributed systems and frameworks based on Kubernetes and Azure Service Fabric. My areas of interest include enterprise cloud solutions, edge computing, high-load applications, multitenant distributed systems, and IoT solutions.

CONTRIBUTOR INSIGHTS

Unlocking Data Insights and Architecture

Data Warehouses, Lakes, and Lakehouses

By Ted Gooch, Staff Software Engineer at Stripe, Inc.

Data management is an ever changing landscape, but throughout its history, a few use cases have driven most of the value and hence the majority of innovation. The following is a list of the key features enabled by effective data management:

- Informed decision-making
- Regulatory compliance
- Improved efficiency
- Data quality and security
- Competitive advantage

As data volume within organizations has scaled ever larger, the underlying technologies have had to evolve and adapt to keep up with the ever increasing demand imposed by such growth. Traditionally, the majority of data was consolidated into a centrally managed platform known as a data warehouse. However, over the last decade, new technologies and data strategies have emerged in an attempt to provide more cost effective solutions. Two new paradigms have emerged as alternatives to the traditional data warehouse stack: the data lake and the data lakehouse.

This article will outline what each of these data management strategies entails and how they map to various selection criteria such as cost, data volume, data integration, security and compliance, ease of use, and a number of other pivotal requirements.

Data Warehouse vs. Data Lake vs. Data Lakehouse

Data warehouses played a crucial role in data-driven organizations for years, supporting business intelligence and historical data analysis. However, as data volumes grew, their integrated storage couldn't scale cost-effectively. This led to the emergence of data lakes, shifting focus to scalable object storage over highly optimized solutions. Data lakes enabled storing vast data amounts, including unstructured or semi-structured data. However, ingestion efficiency and integration with traditional tools posed challenges.

In 2019, the term "data lakehouse" was introduced to bridge the gap between data warehouses and data lakes. The goal is a unified platform for structured and unstructured data, fostering collaboration among data professionals. The below table summarizes the main decision points and how each architecture addresses (or doesn't) that item:

Table 1

DATA MANAGEMENT ARCHITECTURE FEATURE COMPARISON

Criteria	Data Warehouse	Data Lake	Data Lakehouse
Data type support	Primarily structured	Diverse (structured, semi-structured, unstructured)	Diverse (structured, semi-structured, unstructured)
Schema enforcement	Enforced schema	Schema-on-read	Structured and flexible
Data processing	High-performance SQL	Flexibility for exploration, ad hoc analysis	Both high-performance SQL and exploration
Data integration	Structured ETL	Supports batch and real-time ingestion	Supports batch and real-time ingestion
Data storage	Structured, columnar	Raw and native format	Raw and structured format
Data quality and governance	Strong governance	Requires careful management	Supports governance with flexibility
Use cases	Structured analytics, complex reporting	Data exploration, machine learning, raw data processing	Combines structured analytics and data exploration

DATA MANAGEMENT ARCHITECTURE FEATURE COMPARISON			
Criteria	**Data Warehouse**	**Data Lake**	**Data Lakehouse**
Query performance	High-speed, low latency	Varied, depending on tools and tuning	High-performance with flexibility
Historical analysis	Yes	Yes	Yes
Scalability	Limited for very large data	Scales horizontally	Scales for data growth
Cost effectiveness	Can be expensive	Cost-effective for storing raw data	Balances cost and performance
Regulatory compliance	Often supported	Requires implementation	Supports compliance measures
Vendor ecosystem	Well-established	Varied and expanding	Evolving and expanding
User profiles	Data analysts, business intelligence	Data engineers and scientists, analysts	Data engineers and scientists, analysts
Real-time analytics	Possible but limited	Varies depending on tools	Supports real-time analytics
Schema evolution	Requires schema changes	Flexible with schema evolution	Supports both schema changes and structure
Data exploration	Limited capability	Flexible for exploration	Supports both analytics and exploration
Hybrid architecture	Can be integrated with data lakes	Can be combined with data warehouses	Combines elements of both

DATA WAREHOUSE

Data warehouses excel at processing structured data with a well-defined schema. With these restrictions, a data warehouse can offer highly efficient querying capabilities. Furthermore, they have strong integration with business intelligence tooling, and have robust integrated support for data quality and governance. The following table gives an overview of data warehouse aspects and how they may benefit or detract from a given use case:

Table 2

DATA WAREHOUSE ASPECT COVERAGE		
Aspect	**Benefits**	**Weaknesses**
Structured data	Efficient storage and management	Limited support for unstructured data
Optimized queries	High-performance querying	Expensive
Data consistency	Enforced data consistency	Inflexible schema

BENEFITS OF USING A DATA WAREHOUSE

Data warehouses provide several key advantages:

- Excel in efficiently storing and managing structured data, making complex analytics accessible through predefined schemas that enhance user-friendliness
- Offer high-performance querying capabilities, enabling the execution of complex analytical tasks and scaling to maintain query speed as data volumes expand
- Prioritize data consistency by enforcing structured schemas and implementing robust data governance measures, ensuring data integrity and reliability, making them a reliable single source of truth for decision-making within organizations

LIMITATIONS OF USING A DATA WAREHOUSE

The weaknesses of a data warehouse revolve around cost, inflexible schema, and limited support for unstructured data. Implementing and maintaining a data warehouse can be expensive, with substantial initial setup and ongoing operational costs. Its reliance on a predefined schema makes it less adaptable to changes in data structure or the inclusion of new data sources, potentially hindering agility. Additionally, data warehouses are primarily designed for structured data, which limits their ability to efficiently handle unstructured or semi-structured data, potentially missing out on valuable insights from diverse data sources.

DATA LAKE

The data lake architecture evolved as a response to the rising costs of operating a data warehouse. A primary goal of this design was to lower the bar, in terms of cost, for storing vast amounts of data. Although data lakes provide a low price point for

storage, they lack some of the integrations and features that have been developed in data warehouses over the years. Below are some of the trade-offs to consider when building a data lake:

Table 3

DATA LAKE ASPECT COVERAGE		
Aspect	**Benefits**	**Limitations**
Scalability	Highly scalable, handles massive data volumes	Data quality concerns
Cost-effectiveness	Cost-effective for storing raw data	Complexity in data processing
Storage of raw and unstructured data	Accommodates diverse data types	Potential data silos

BENEFITS OF USING A DATA LAKE

A data lake architecture offers distinct advantages for organizations seeking to harness their data effectively:

- Provides exceptional scalability, effortlessly accommodating massive data volumes as businesses grow
- Proves highly cost-effective, offering a budget-friendly solution for storing raw data in its native format
- Excels at storage, allowing organizations to effortlessly ingest and manage diverse data types, including unstructured and semi-structured data

This versatility enables businesses to leverage their entire data ecosystem, promoting innovation and data-driven decision-making while keeping costs in check.

LIMITATIONS OF USING A DATA LAKE

Despite its strengths, a data lake architecture is not without its challenges. It often introduces complexity in data processing, as the flexibility it offers can lead to difficulties in data organization, quality assurance, and integration. Moreover, there is a risk of potential data silos within a data lake, where data may become fragmented and less accessible, hindering the ability to derive valuable insights. Data discovery becomes a concern. To maximize the benefits of a data lake, organizations must carefully plan their data governance and integration strategies to mitigate these challenges effectively.

DATA LAKEHOUSE

The data lakehouse paradigm seeks to balance the benefits and trade-offs of a data warehouse and a data lake. This is accomplished by providing an integrated solution on top of what was traditionally data lake components. The goal is to provide the scalability, flexibility, and cost benefits of a data lake while still offering the performance, data governance, and user-friendliness of a data warehouse.

Table 4

DATA LAKEHOUSE ASPECT COVERAGE		
Aspect	**Benefits**	**Limitations**
Hybrid architecture	Combines data warehouse and data lake capabilities	Architectural complexity
Cost-to-performance flexibility	Offers cost-effective scalability with high performance	Potential performance issues
Real-time analytics	Supports real-time analytics	Evolving technology landscape

BENEFITS OF USING A DATA LAKEHOUSE

A data lakehouse architecture presents a compelling solution for organizations aiming to unlock the full potential of their data. By seamlessly combining the robust features of a data warehouse and the flexibility of a data lake, it offers a comprehensive data management ecosystem. One of its standout advantages lies in its cost-to-performance flexibility, allowing businesses to balance their data storage and processing needs efficiently, optimizing both cost-effectiveness and performance.

Additionally, the data lakehouse empowers organizations with real-time analytics capabilities, enabling them to make data-driven decisions and respond swiftly to changing trends and opportunities. This amalgamation of features positions the data lakehouse as a versatile and powerful solution for modern data management and analytics needs.

LIMITATIONS OF USING A DATA LAKEHOUSE

A data lakehouse does come with certain limitations. One key concern is architectural complexity, as the integration of these diverse features can lead to intricate data management structures, requiring thorough planning and management. Potential

performance issues may arise due to the combination of features, and organizations must carefully optimize their data processing to prevent bottlenecks.

Additionally, the ever-evolving technology landscape means that staying up-to-date with the latest advancements and best practices is essential for maximizing the benefits of a data lakehouse. Despite these limitations, its capacity to provide a comprehensive data solution often outweighs these challenges for organizations seeking to harness the full potential of their data assets.

The Future of Data Storage

The future of data management and storage is poised to undergo transformative changes driven by evolving trends. One of the pivotal developments is the growing emphasis on interoperability between existing data architectures, including data warehouses, data lakes, and data lakehouses. Organizations are recognizing the need to seamlessly integrate these technologies to harness the full spectrum of their data assets efficiently. Simultaneously, data governance and data quality are becoming paramount concerns, driven by the exponential growth of data volumes and the increasing importance of compliance and data accuracy.

As organizations navigate this landscape, they are likely to adopt comprehensive data governance strategies, leveraging automation and AI-powered tools to enhance data quality, traceability, and privacy. Overall, the future of data management and storage will revolve around achieving a harmonious synergy between diverse data architectures, underpinned by robust data governance practices to ensure the reliability and integrity of data assets in an ever-evolving digital ecosystem.

EVOLVING TECHNOLOGIES

Machine learning and AI technologies will play a pivotal role in automating data processing, analysis, and decision-making, enabling organizations to derive deeper insights from their data assets. Moreover, the rise of edge computing and the Internet of Things (IoT) will necessitate real-time data management capabilities, prompting the adoption of cloud-native solutions and distributed data architectures. As data privacy and security concerns grow, robust data governance frameworks will become imperative, ensuring that organizations maintain compliance with evolving regulations while safeguarding sensitive data.

Collaboration across departments and data-driven cultures will be pivotal, with data democratization empowering a broader range of employees to harness data for informed decision-making. In this dynamic landscape, the ability to adapt swiftly to emerging technologies and data management trends will be the cornerstone of success in the data-driven future.

HYBRID SOLUTIONS

Hybrid solutions in data management architecture overcome limitations of different storage types. Such hybrid solutions are becoming more popular, and are starting to precipitate fully new designs. A model that exemplifies this concept involves not just the separation of compute and storage, as often seen in data lakes, but also a distinct storage platform integrated separately from the compute layer. This has played out most visibly in the emergence of open table formats such as Iceberg, Hudi, and Delta Lake.

Conclusion

The decision between a data warehouse, data lake, or data lakehouse involves a complex set of trade-offs. Data warehouses excel in structured analytics but may lack flexibility for diverse data types. Data lakes offer versatility but require careful data governance. The emerging data lakehouse concept seeks to balance these trade-offs by combining features of both, offering a unified platform; however, this choice is not one-size-fits-all. Organizations must weigh their specific business needs and adapt their data management strategies accordingly, considering factors such as data type diversity, scalability, cost, and the evolving technology landscape. The key lies in making informed decisions that align with current and future data requirements and recognizing the importance of ongoing adaptation in the dynamic world of data management.

Ted Gooch, Staff Software Engineer at Stripe, Inc.
@tgooch44 on DZone | @tedgooch on X (Twitter)

Ted is a seasoned data professional with 15+ years of work in the data space, and he has worked with many organizations to improve infrastructure and best practices around data. While at Netflix, he worked on the initial implementation of Iceberg and is currently a committer on the Iceberg open-source project. In his current role, he is helping Stripe build the next generation of data tooling with an emphasis on security, compliance, and user experience.

CONTRIBUTOR INSIGHTS

The Evolution of Data Pipelines

ETL, ELT, and the Rise of Reverse ETL

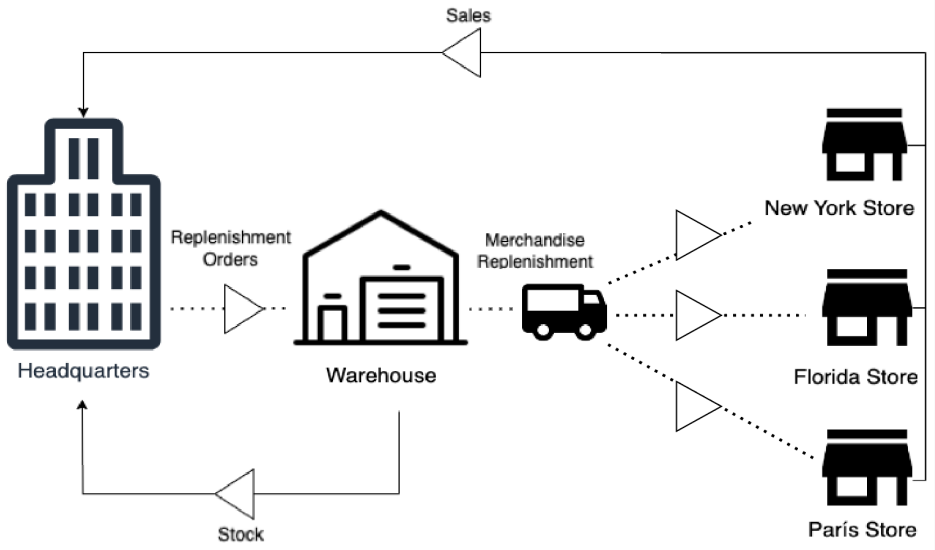
By Miguel García Lorenzo, VP of Engineering at Nextail

Originally, the term "data pipeline" was focused primarily on the movement of data from one point to another, like a technical mechanism to ensure data flows from transactional databases to destinations such as data warehouses or to aggregate this data for analysis. Fast forward to the present day, data pipelines are no longer seen as IT operations but as a core component of a business's transformation model.

New cloud-based data orchestrators are an example of evolution that allows integrating data pipelines seamlessly with business processes, and they have made it easier for businesses to set up, monitor, and scale their data operations. At the same time, data repositories are evolving to support both operational and analytical workloads on the same engine.

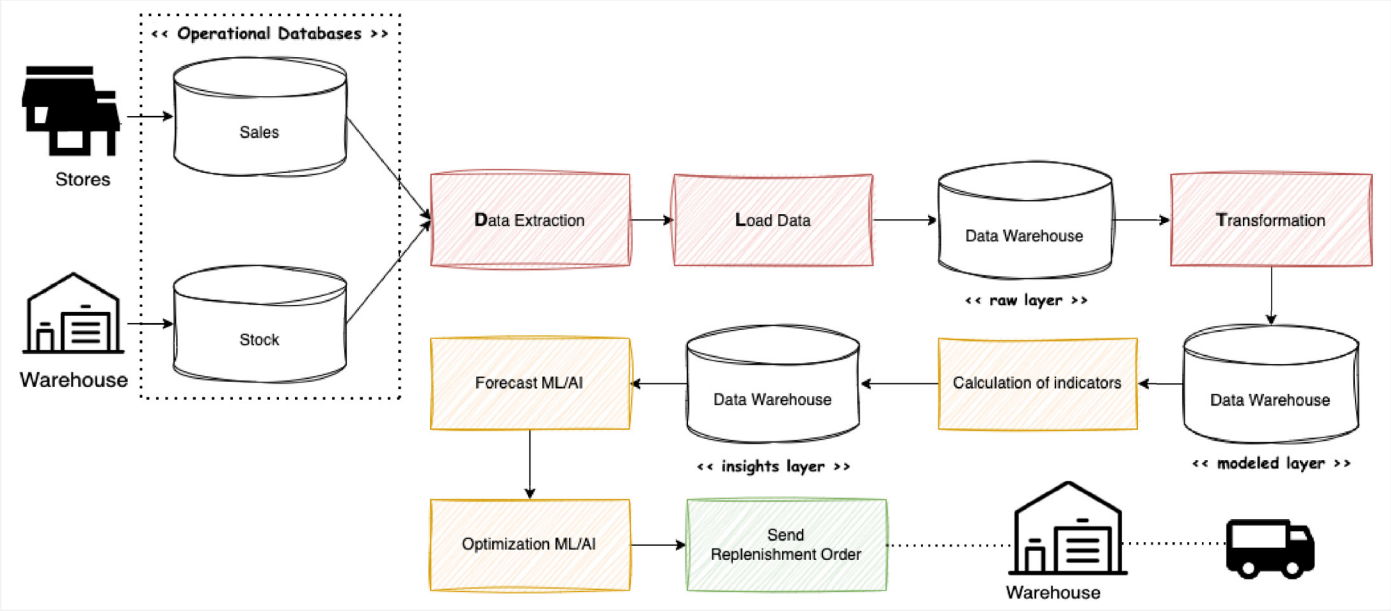
Consider a replenishment process for a retail company, a mission-critical business process, in Figure 1.

Figure 1: Retail business replenishment process



The figure is a clear example of where the new data pipeline approach is having a transformational impact on the business. Companies are evolving from corporate management applications to new data pipelines that include artificial intelligence (AI) capabilities to create greater business impact. Figure 2 demonstrates an example of this.

Figure 2: Retail replenishment data pipeline



We no longer see a data process based on data movement, but rather we see a business process that includes machine learning (ML) models or integration with distribution systems. It will be exciting to see how data pipelines will evolve with the emergence of the new generative AI.

Data Pipeline Patterns

All data pipeline patterns are composed of the following stages, although each one of them has a workflow and use cases that make them different:

- **E**xtract – To retrieve data from the source system without modifying it. Data can be extracted from several sources such as databases, files, APIs, streams, or more.
- **T**ransform – To convert the extracted data into final structures that are designed for analysis or reporting. The data transformed is stored in an intermediate staging area.
- **L**oad – To load the transformed data into the final target database.

Currently and after the evolution of data pipelines, these activities are known as data ingestion and determine their pattern as we will see below. Here are some additional activities and components that are now part and parcel of modern data pipelines:

- **Data cleaning** is a crucial step in the data pipeline process that involves identifying and correcting inconsistencies and inaccuracies in datasets, such as removing duplicate records or handling missing values.
- **Data validation** ensures the data being collected or processed is accurate, reliable, and meets the specified criteria or business rules. This includes whether the data is of the correct type, falls within a specified range, or that all required data is present and not missing.
- **Data enrichment** improves the quality, depth, and value of the dataset by adding relevant supplementary information that was not originally present with additional information from external sources.
- **Machine learning** can help enhance various stages of the pipeline from data collection to data cleaning, transformation, and analysis, thus making it more efficient and effective.

EXTRACT, TRANSFORM, LOAD

Extract, transform, load (ETL) is a fundamental process pattern in data warehousing that involves moving data from the source systems to a centralized repository, usually a data warehouse. In ETL, all the load related to the transformation and storage of the raw data is executed in a layer previous to the target system.

Figure 3: ETL data pipeline

The workflow is as follows:

1. Data is extracted from the source system.
2. Data is transformed into the desired format in an intermediate staging area.
3. Transformed data is loaded into the data warehouse.

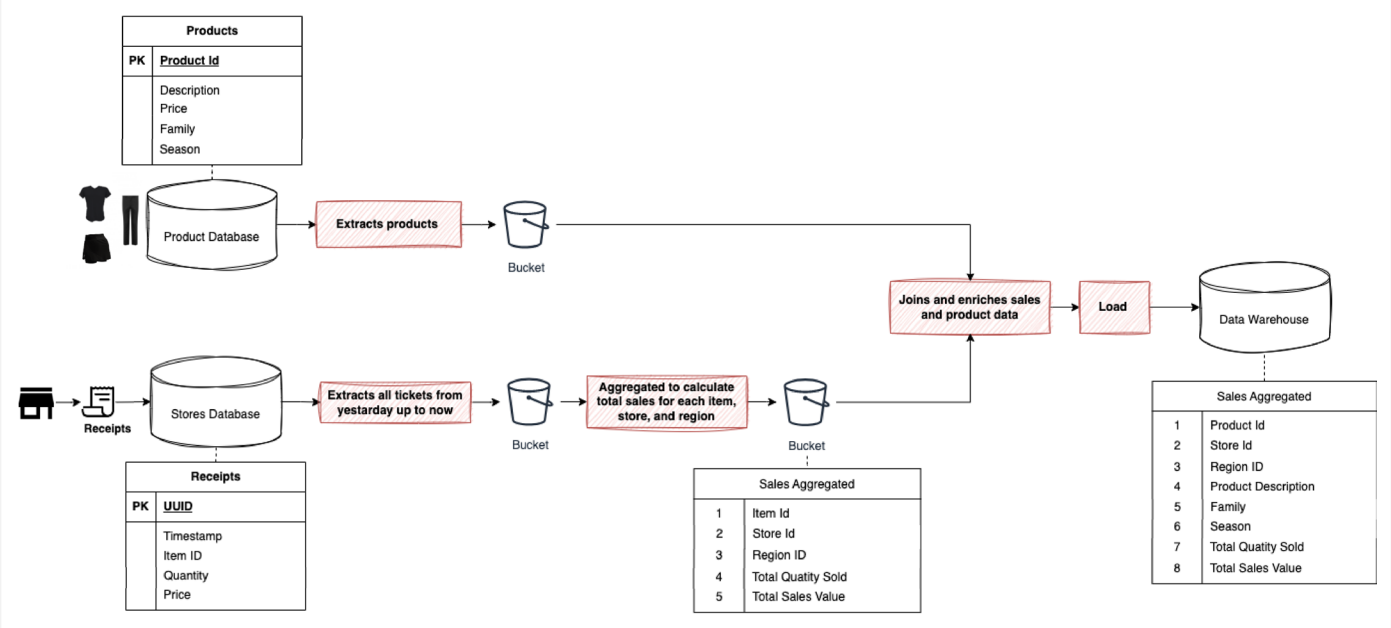
When to use this pattern:

- **Target system performance** – If the target database, usually a data warehouse, has limited resources and poor scalability, we want to minimize the impact on performance.
- **Target system capacity** – When the target systems have limited storage capacity or the GB price is very high, we are interested in transforming and storing the raw data in a cheaper layer.
- **Pre-defined structure** – When the structure of the target system is already defined.

ETL USE CASE

This example is the classic ETL for an on-premise system where, because of the data warehouse computational and storage capacity, we are neither interested in storing the raw data nor in executing the transformation in the data warehouse itself. It is more economical and efficient when it involves an on-premises solution that is not highly scalable or at a very high cost.

Figure 4: ETL sales insights data pipeline

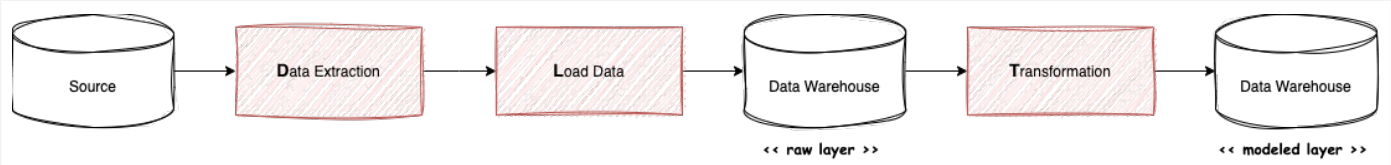


EXTRACT, LOAD, TRANSFORM

Modern cloud-based data warehouses and data lakes, such as Snowflake or BigQuery, are highly scalable and are optimized for in-house processing that allows for handling large-scale transformations more efficiently in terms of performance and cheaper in terms of cost.

In extract, load, transform (ELT), the data retrieved in the source systems is loaded directly without transformation into a raw layer of the target system. Finally, the following transformations are performed. This pattern is probably the most widely used in modern data stack architectures.

Figure 5: ELT data pipeline



The workflow is as follows:

1. Data is extracted from the source system.
2. Data is loaded directly into the data warehouse.
3. Transformation occurs within the data warehouse itself.

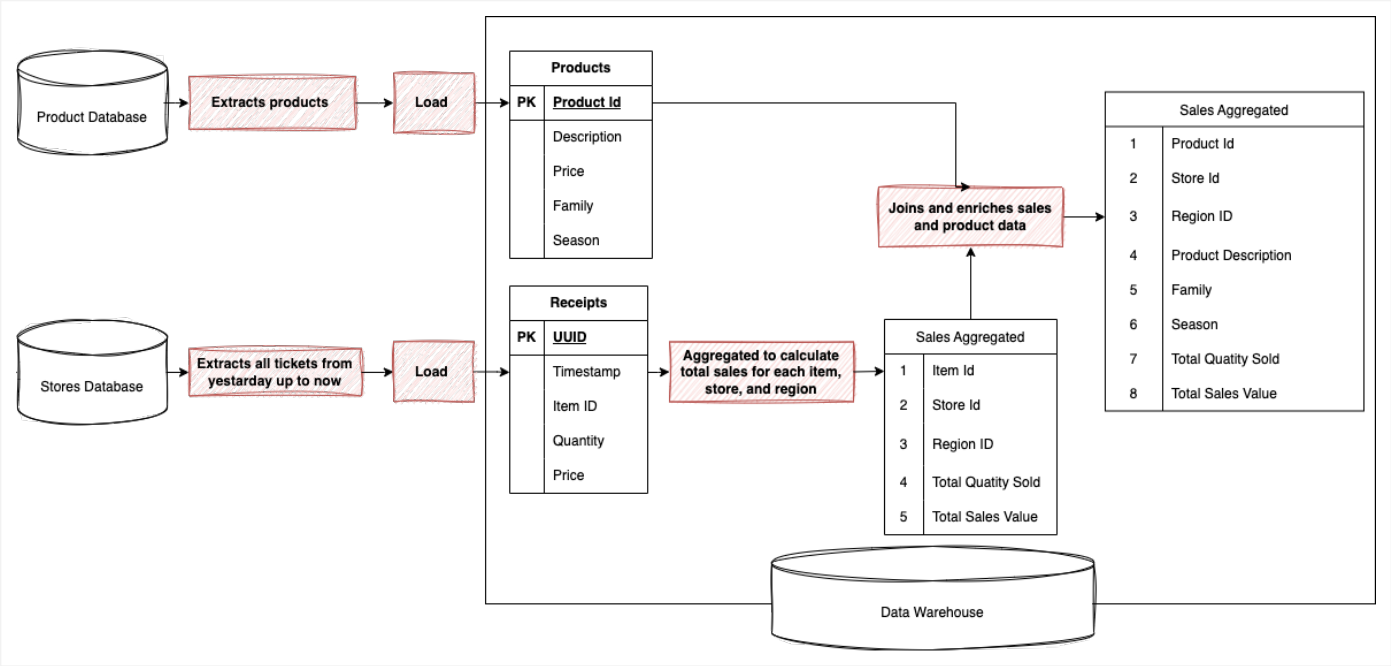
When to use this pattern:

- **Cloud-based modern warehouse** – Modern data warehouses are optimized for in-house processing and can handle large-scale transformations efficiently.
- **Data volume and velocity** – When handling large amounts of data or near real-time processing.

ELT USE CASE

New cloud-based data warehouse and data lake solutions are high-performant and highly scalable. In these cases, data repositories are better suited for work than external processes. The transformation process can take advantage of new features and run data transformation queries inside the data warehouse faster and at a lower cost.

Figure 6: ELT sales insights data pipeline



REVERSE ETL

Reverse ETL is a new data pattern that has grown significantly in recent years and has become fundamental for businesses. It is composed of the same stages as a traditional ETL, but functionally, it does just the opposite. It takes data from the data warehouse or data lake and loads it into the operational system.

Nowadays, analytical solutions are generating information with a differential value for businesses and also in a very agile manner. Bringing this information back into operational systems allows it to be actionable across other parts of the business in a more efficient and probably higher impact way.

Figure 7: Reverse ETL data pipeline

The workflow is as follows:

1. Data is extracted from the data warehouse.
2. Data is transformed into the desired format in an intermediate staging area.
3. Data is loaded directly into the operational system.

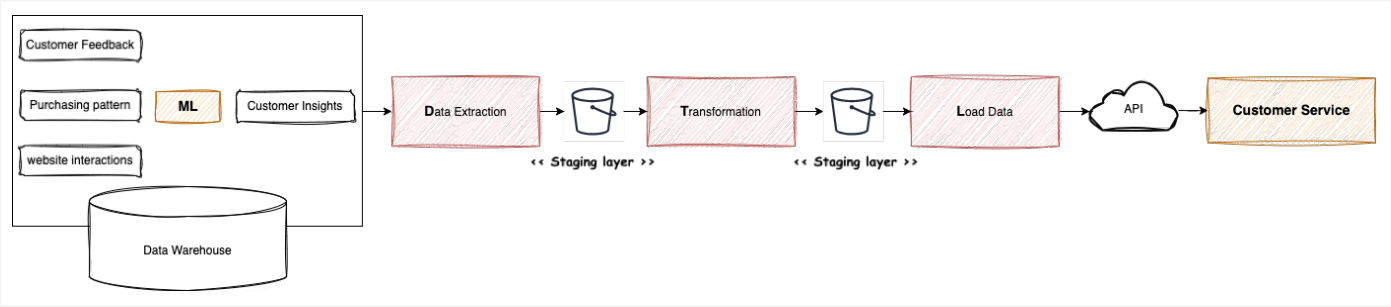
When to use this pattern:

- **Operational use of analytical data** – to send back insights to operational systems to drive business processes
- **Near real-time business decisions** – to send back insights to systems that can trigger near real-time decisions

REVERSE ETL USE CASE

One of the most important things in e-commerce is to be able to predict what items your customers are interested in; this type of analysis requires different sources of information, both historical and real-time. The data warehouse contains historical and real-time data on customer behavior, transactions, website interactions, marketing campaigns, and customer support interactions. The reverse ETL process enables e-commerce to operationalize the insights gained from its data analysis and take targeted actions to enhance the shopping experience and increase sales.

Figure 8: Reverse ETL customer insights data pipeline



THE RISE OF REAL-TIME DATA PROCESSING

As businesses become more data-driven, there's an increasing need to have actionable information as quickly as possible. This evolution has driven the transition from batch processing to real-time processing that allows processing the data immediately as it arrives. The advent of new technological tools and platforms that are capable of handling real-time data processing such as Apache Kafka, Apache Pulsar, or Apache Flink have made it possible to build real-time data pipelines.

Real-time analytics became crucial for scenarios like fraud detection, IoT, edge computing, recommendation engines, and monitoring systems. Combined with AI, it allows businesses to make automatic, on-the-fly decisions.

THE INTEGRATION OF AI AND ADVANCED ANALYTICS

Advancements in AI, particularly in areas like generative AI and large language models (LLMs), will transform data pipeline landscape capabilities such as enrichment, data quality, data cleansing, anomaly detection, and transformation automation. Data pipelines will evolve exponentially in the coming years, becoming a fundamental part of the digital transformation, and companies that know how to take advantage of these capabilities will undoubtedly be in a much better position. Some of the activities where generative AI will be fundamental and will change the value and way of working with data pipelines include:

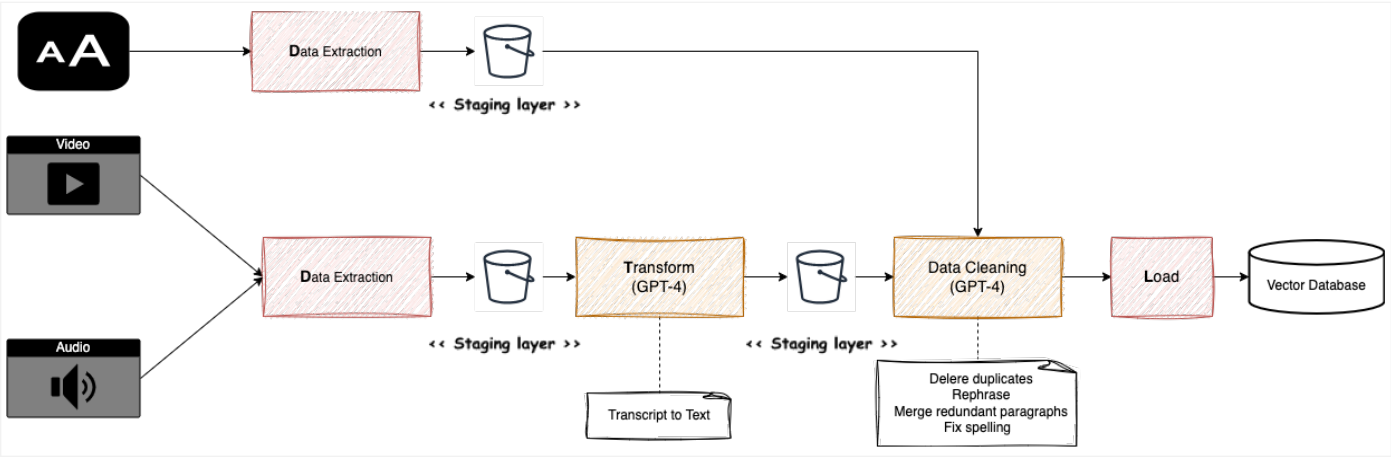
- Data cleaning and transformation
- Anomaly detection
- Enhanced data privacy
- Real-time processing

AI AND ADVANCED ANALYTICS USE CASE

E-commerce platforms receive many support requests from customers every day, including a wide variety of questions and responses written in different conversational styles. Increasing the efficiency of the chatbot is so important to improve the customer experience, and many companies decide to implement a chatbot using a GPT model. In this case, we need to provide all information from questions, answers, and technical product documentation that is available in different formats.

The new generative AI and LLM models not only allow us to provide innovative solutions for interacting with humans, but also to increase the capabilities of our data pipelines, such as data cleaning or transcriptions. Training the GPT model requires clean and preprocessed text data. This involves removing any personally identifiable information, correcting spelling, correcting grammar mistakes, or removing any irrelevant information. A GPT model can be trained to perform these tasks automatically.

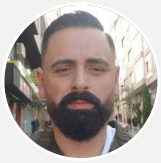
Figure 9: ETL data pipeline with AI for chatbot content ingestion



Conclusion

Data pipelines have evolved a lot in the last few years, initially with the advent of streaming platforms and later with the explosion of the cloud and new data solutions. This evolution means that every day they have a greater impact on business value, moving from a data movement solution to a key element in the business transformation. The explosive growth of generative AI solutions in the last year has opened up an exciting path, as they have a significant impact on all stages of the data pipelines; therefore, the near future is undoubtedly linked to AI.

Such a disruptive evolution requires the adaptation of organizations, teams, and engineers to enable them to use the full potential of technology. The data engineer role must evolve to acquire more business and machine learning skills. This is a new digital transformation, and perhaps the most exciting and complex movement in recent decades.



Miguel García Lorenzo, VP of Engineering at Nextail
@miguelglor on DZone | @mgarlorenzo on LinkedIn

Miguel is VP of Engineering at Nextail. He has 10+ years of experience in the data space, leading teams and building high-performance solutions. A book lover and advocate of platform design as a service and data as a product.

CONTRIBUTOR INSIGHTS

Real-Time Analytics

All Data, Any Data, Any Scale, at Any Time

By Timothy Spann, Principal Developer Advocate at Cloudera

We live in an era of rapid data generation from countless sources, including sensors, databases, cloud, devices, and more. To keep up, we require real-time analytics (RTA), which provides the immediacy that every user of data today expects and is based on stream processing. Stream processing is used to query a continuous stream of data and immediately process events in that stream. Ideally, your stream is processing events in subseconds as they occur.

RTA enables organizations to process and analyze data in real-time, providing valuable insights and driving timely decision-making. The key factor is that we must process each event as it arrives, and time matters. Unlike traditional batch processing, where data is analyzed in predetermined intervals, RTA provides an immediate and continuous understanding of events and data as they arrive.

Why is this important for data pipelines? Let's discuss.

How RTA Is Relevant to Data Pipelines

RTA needs to be fed by a continuous flow of data from various sources. Data pipelines are responsible for ingesting data from sources — like sensors, logs, databases, and APIs — and delivering the data to an RTA platform as a stream. In the case of a real-time streaming data pipeline fed from a tool such as Apache NiFi, we receive each chunk of data as an event from Apache Kafka. There are many data pipelines that are batched by being fed in large chunks at regular intervals or on demand. These batches yield bursts of data when translated to a stream, and the bursts can cause a backup, like going from local roads to a superhighway.

A tool like NiFi that lets you transition batches to streams without code is ideal. There are several architecture decisions based on how to keep batches and streams flowing and available to the same consumers and often having the same sources. An important part is that we need a real-time lakehouse based on Apache Iceberg or Apache Paimon to permanently store large data that will be needed later for LLM, data training, and lookups.

STREAM PROCESSING AND RTA

RTA is built on stream processing, which includes frameworks like Apache Flink, Apache Kafka Streams, RisingWave, and Faust Streaming. Stream processing is often done by writing code in Java, Scala, or Python. The most exciting form of stream processing is done with SQL, and it can be often done without implicit compiling or deploying in an IDE.

The main feature of stream processing is ingesting a continuous event stream to filter, enrich, join, transform, convert, route, and analyze data as it arrives in near real time. Real time has a lot of definitions, but for stream processing in most contexts, let us assume time means subsecond.

The Importance of Real-Time Analytics

RTA is important for timely decision making since the faster data arrives and is processed, the faster we can act. The immediate usage of data streams can make new types of applications possible, such as detecting fraud while it occurs. With data arriving constantly from thousands of devices, RTA is critical for receiving, processing, handling, and potentially alerting on this data.

RTA allows for predictive analytics on real-world machinery that could be alerting to upcoming failures based on current and historical data and machine learning (ML) analysis. RTA enables processing not just for each event as it arrives, but also for time windows of data, frequently in one-, five-, or 15-minute chunks.

BATCH vs. REAL TIME FOR DATA PIPELINES

I often want to make every data pipeline a real-time one for the previously mentioned reasons. But there are a number of reasons batches make sense for certain use cases. Batch is good for collecting data at predefined times and chunks. This often occurs when dealing with mainframes, legacy systems, and old data warehouses.

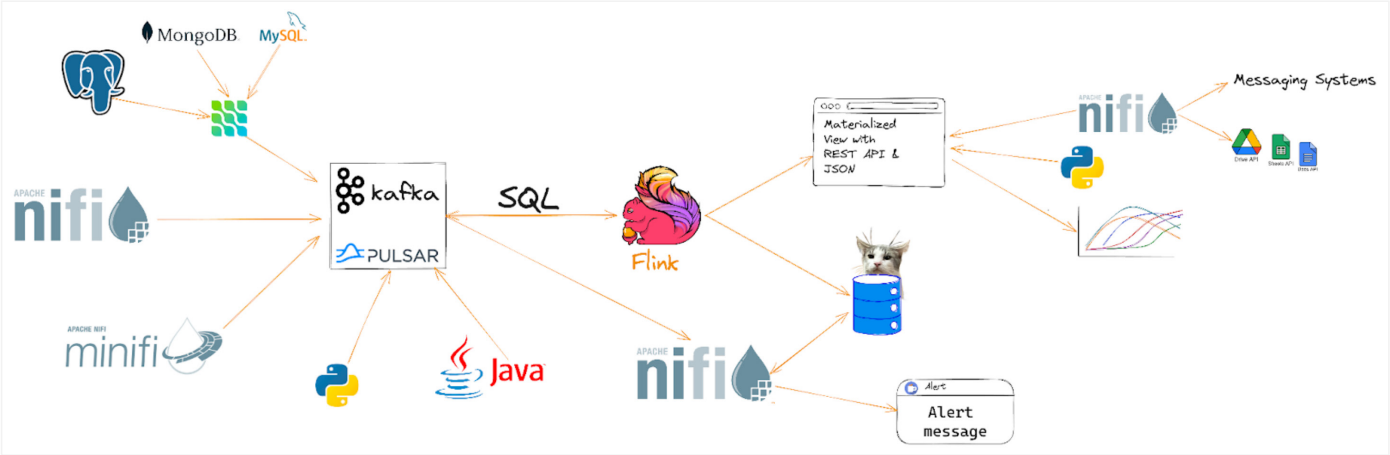
These batches tend to be easier to scale and more predictable. This can allow for more affordable data processing at slower, timed intervals. Batch workloads can be run at cheaper times on lesser hardware and allow for longer time to process. You can optimize and break up batches to use less resources (in exchange for time).

Real time lowers latency to seconds or less at the cost of being more expensive to scale, less predictable, burstable, and using many resources in a short period of time. Real time lends itself better to things like Kubernetes with the ability to scale on demand or automatically.

If you need results immediately, then you cannot choose batch processing for your pipeline. One great feature of a tool like Apache Nifi is that converting from batch to stream or stream to batch is merely choosing a continuous interval or setting a cron schedule.

For many applications, you may need a combination of batch and real-time processing on your data, aka the famous Lambda architecture. For a variation of this, we send each event immediately to Apache Kafka, while we set an interval before we push chunks of data to Apache Iceberg. We often set chunks to be 250 megabytes, 500,000 records, or five minutes of data — whichever comes first.

Figure 1: Real-time architecture layout



Implementing a Real-Time Analytics Pipeline

One of the great features of RTA pipelines is that if you plan your architecture, each piece is straightforward to implement. We will be using the most popular open-source frameworks to build a pipeline and highlight alternatives available in the community. The most important thing is to determine your data sources, time and data size requirements, user requirements, and platform.

The first implementation step is to choose how you are ingesting data sources. I suggest utilizing Apache NiFi, which is a great way to start the ingestion process and explore that data as it arrives. This is also when you start building data schemas, determine the enrichments and transformations needed, and tour intermediate and final data formats.

Often, there will be multiple data destinations, including real-time OLAP datastores like Apache Pinot, vector databases like Milvus, analytic data lake tables like Apache Iceberg, and column-oriented data stores like Apache Kudu. For many use cases, we often write a raw copy of the incoming data into storage like Apache Ozone or Hadoop Distributed File System (HDFS). This may be used later for validation, reloads, or training ML models.

Once the data meets the minimum clean format and has a basic schema, we stream it into Apache Kafka or Pulsar for stream processing. We usually choose JSON or Apache Avro as the format for this data.

Now that our data is streaming into one or more topics within Kafka, we can start consuming it with our real-time stream processing and analytics tools. There are a number of ways to create RTA applications with Flink, but I recommend starting with Flink SQL. This allows you to build streaming applications with just SQL. You will be able to join streams, do windowing, and handle multiple sources and sinks.

When Flink SQL has completed its flows, we often sink it to another Kafka topic, database table, or other datastore. For most architectures, I sink it to Kafka.

EXPLORING RTA USE CASES

There are many interesting use cases out there for RTA. I will highlight a couple that I have worked on, but this is not an exhaustive list. You can do much more; your data dreams are the only limit.

MONITORING IOT DEVICES AND SYSTEMS

A use case that I love due to personal experience is monitoring IoT devices and systems. This use case is constantly expanding as every real-world object is adding sensors, networks, cameras, GPUs, computers, and even edge AI systems. One of the main difficulties with IoT is that often, existing systems are not designed to stream data and may not make sensor endpoints available at all or in non-standard formats. This has been challenging in the past but is improving.

Often, we add additional devices alongside built-in systems so we can control the data types, speed, and operation. This is enabled by adding a small smart agent such as Apache MiNiFi.

REAL-TIME FRAUD DETECTION

This is probably one of the most quintessential use cases for protecting people from bad transactions, and most major credit cards have been implementing this for some time. With real time, we can see not just every transaction and all the metadata surrounding them, but we can see trends over windows of time. The ability of stream processing to discover duplicate, anomalous, and overly frequent charges in unusually small windows of event time helps detect fraud.

Stream processing makes it possible to compare new data against existing data profiles of people and credit cards. This is also supercharged by real-time pipelines continuously training ML models on new data so that they recognize typical and atypical patterns for every credit card and its user.

RTA Data Pipeline Guiding Principles

Designing an RTA data pipeline with Apache NiFi, Apache Kafka, and Apache Flink involves careful consideration of the data sources, processing logic, and how each component fits into the overall architecture. Here's a step-by-step approach to help you decide how to process different data sources:

1. **Discover your data sources** such as databases, APIs, files, logs, messages, documents, images, videos, and sensors.
2. **Start collecting** them with NiFi to see what the data looks like, how large the stream is, and if you need to use a different tool. This is where you can do basic cleanup, transformations, enrichment, and field definitions. Then you can route to Kafka for additional processing. For simple use cases, you may have a NiFi stream to a datastore directly.
3. Before arriving in Kafka, **you must plan** out your topics, schemas of data, and run some tests to see the amount of data.
4. The next step is where you **determine which RTA you need**. This could be as simple as detecting alert conditions; joining data streams like weather and transit on latitude and longitude; transforming data further; or making computations, aggregations, and anything you can do with advanced SQL using Flink SQL.
5. Often, you will **want this data stored for future usage**. Flink SQL can store in an underlying streaming table format in Apache Paimon or in an open lakehouse format like Apache Iceberg.
6. Finally, to **move this to production**, you need to run on an enterprise data platform that provides things like monitoring, fault tolerance, auto-scaling, data security, data governance, and control of the underlying hybrid cloud environment.

Remember that the specific implementation details will depend on your use case, data volume, processing requirements, and technical constraints. It's crucial to thoroughly understand the capabilities and limitations of each framework to design an efficient and robust RTA pipeline.

Conclusion

All data pipelines need to embrace real-time data streams to harness the value of data as it is created, and it's also time to embrace both real-time analytics and batch processing as the two processes converge. It is critical to be ready for the future, which will push more data, more speed, more requirements, and more challenges, including:

- As edge computing and edge AI continue to grow, they will produce ever increasing amounts of data and require faster processing.
- New frameworks, updated cloud computing architectures, more integrated AI/ML workloads, and new paradigms are bound to intercept with RTA and data pipelines.

The next couple of years should be very exciting.

SEE ADDITIONAL RESOURCES ON NEXT PAGE

Additional resources:

- "Harnessing the Power of NiFi" by Timothy Spann
- "Real-Time Stream Processing With Hazelcast and StreamNative" by Timothy Spann and Fawaz Ghali, PhD
- *Getting Started With Apache Iceberg* Refcard by Ted Gooch
- *Apache Kafka Patterns and Anti-Patterns* Refcard by Abhishek Gupta
- *Apache Kafka Essentials* Refcard by Sudip Sengupta
- "Flink + Iceberg," Alibaba Cloud, Community Blog
- GitHub RTA examples

Timothy Spann, Principal Developer Advocate at Cloudera
@bunkertor on DZone | @timothyspann on LinkedIn | @paasdev on Twitter | datainmotion.dev

Tim Spann is a principal developer advocate in Data In Motion for Cloudera. He works with Apache NiFi, Apache Kafka, Apache Pulsar, Apache Flink, Flink SQL, Apache Pinot, Trino, Apache Iceberg, DeltaLake, Apache Spark, Big Data, IoT, Cloud, and deep learning. Tim has over 10 years of experience with the IoT, big data, distributed computing, messaging, streaming technologies, and Java programming.

CONTRIBUTOR INSIGHTS

Data Observability: Better Insights Through Reliable Data Practices

By Joana da Silva Carvalho, Site Reliability Engineer at Virtuoso.qa

Organizations today rely on data to make decisions, innovate, and stay competitive. That data must be reliable and trustworthy to be useful. Many organizations are adopting a data observability culture that safeguards their data accuracy and health throughout its lifecycle. This culture involves putting in motion a series of practices that enable you and your organization to proactively identify and address issues, prevent potential disruptions, and optimize their data ecosystems. When you embrace data observability, you protect your valuable data assets and maximize their effectiveness.

Understanding Data Observability

As Yuval Noah Harari puts it, data is an incredibly valuable asset today. As such, organizations must ensure that their data is accurate and dependable. This is where data observability comes in, but what is data observability exactly?

> *"In a world deluged by irrelevant information, clarity is power."*
>
> —Yuval Noah Harari, *21 Lessons for the 21st Century, 2018*

Data observability is the means to ensure our data's health and accuracy, which means understanding how data is collected, stored, processed, and used, plus being able to discover and fix issues in real time. By doing so, we can optimize our system's effectiveness and reliability by identifying and addressing discrepancies while ensuring compliance with regulations like GDPR or CCPA. We can gather valuable insights that prevent errors from recurring in the future by taking such proactive measures.

WHY IS DATA OBSERVABILITY CRITICAL?

Data reliability is vital. We live in an era where data underpins crucial decision-making processes, so we must safeguard it against inaccuracies and inconsistencies to ensure our information is trustworthy and precise. Data observability allows organizations to proactively identify and address issues before they can spread downstream, preventing potential disruptions and costly errors.

One of the advantages of practicing data observability is that it'll ensure your data is reliable and trustworthy. This means continuously monitoring your data to avoid making decisions based on incomplete or incorrect information, giving you more confidence.

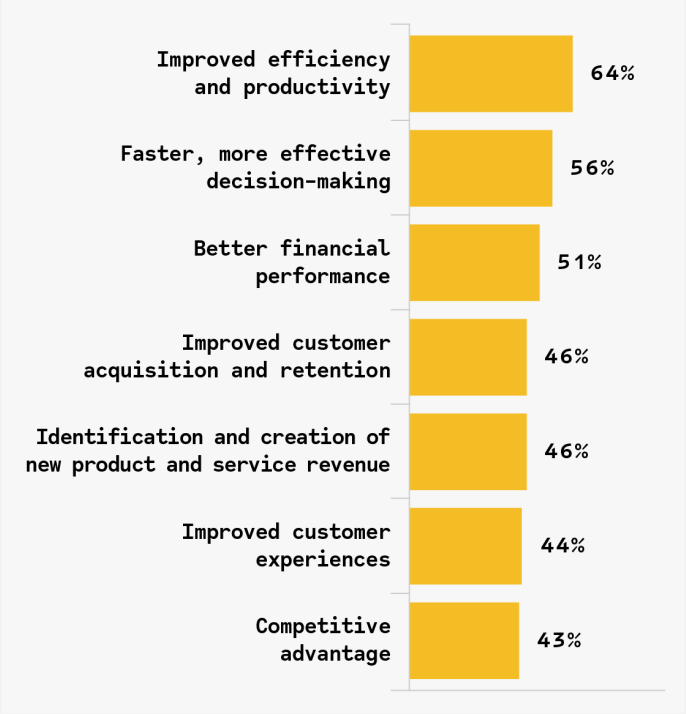
Analyzing your technology stack can also help you find inefficiencies and areas where resources are underutilized, saving you money. But incorporating automation tools into your data observability process is the cherry on top of the proverbial cake, making everything more efficient and streamlined.

Data observability is a long-run approach to safeguarding the integrity of your data so that you can confidently harness its power, whether it's for informed decision-making, regulatory compliance, or operational efficiency.

ADVANTAGES AND DISADVANTAGES OF DATA OBSERVABILITY

When making decisions based on data, it's essential to be quick. But what if the data isn't dependable? That's where data observability comes in. However, like any tool, it has its advantages and disadvantages.

Figure 1: The benefits of companies using analytics



Data source: The Global State of Enterprise Analytics, 2020, MicroStrategy

Table 1

IMPLEMENTING DATA OBSERVABILITY: ADVANTAGES AND DISADVANTAGES	
Advantages	**Disadvantages**
Trustworthy insights for intelligent decisions: Data observability provides decision-makers with reliable insights, ensuring well-informed choices in business strategy, product development, and resource allocation.	**Resource-intensive setup**: Implementing data observability demands time and resources to set up tools and processes, but the long-term benefits justify the initial costs.
Real-time issue prevention: Data observability acts as a vigilant guardian for your data, instantly detecting issues and averting potential emergencies, thus saving time and resources while maintaining data reliability.	**Computational overhead from continuous monitoring**: Balancing real-time monitoring with computational resources is essential to optimize observability.
Enhanced team alignment through shared insights: Data observability fosters collaboration by offering a unified platform for teams to gather, analyze, and act on data insights, facilitating effective communication and problem-solving.	**Training requirements for effective tool usage**: Data observability tools require skill, necessitating ongoing training investments to harness their full potential.
Accurate data for sustainable planning: Data observability establishes the foundation for sustainable growth by providing dependable data that's essential for long-term planning, including forecasting and risk assessment.	**Privacy compliance challenges**: Maintaining data observability while adhering to strict privacy regulations like GDPR and CCPA can be intricate, requiring a delicate balance between data visibility and privacy compliance.
Resource savings: Data observability allows you to improve how resources are allocated by identifying areas where your technology stack is inefficient or underutilized. As a result, you can save costs and prevent over-provisioning resources, leading to a more efficient and cost-effective data ecosystem.	**Integration complexities**: Integrating data observability into existing data infrastructure may pose challenges due to compatibility issues and legacy systems, potentially necessitating investments in specific technologies and external expertise for seamless integration.

To sum up, data observability has both advantages and disadvantages, such as providing reliable data, detecting real-time problems, and enhancing teamwork. However, it requires significant time, resources, and training while respecting data privacy. Despite these challenges, organizations that adopt data observability are *better prepared to succeed in today's data-driven world and beyond*.

Cultivating a Data-First Culture

Data plays a crucial role in today's fast-paced and competitive business environment. It enables informed decision-making and drives innovation. To achieve this, it's essential to cultivate an environment that values data. This culture should prioritize accuracy, dependability, and consistent monitoring throughout the data's lifecycle.

To ensure effective data observability, strong leadership is essential. Leaders should prioritize data from the top down, allocate necessary resources, and set a clear vision for a data-driven culture. This leadership fosters team collaboration and alignment, encouraging them to work together towards the same objectives. When teams collaborate in a supportive work environment, critical data is properly managed and utilized for the organization's benefit.

Technical teams and business users must work together to create a culture that values data. Technical teams build the foundation of data infrastructure while business users access data to make decisions. Collaboration between these teams leads to valuable insights that drive business growth.

By leveraging data observability, organizations can make informed decisions, address issues quickly, and optimize their data ecosystem for the benefit of all stakeholders.

Figure 2: Data generated, gathered, copied, and consumed

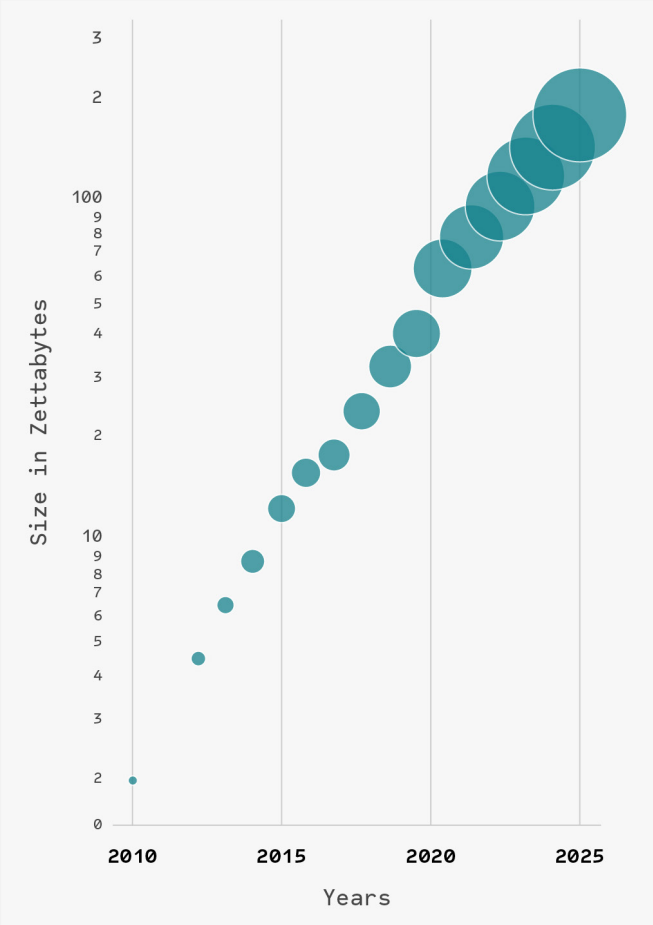


Figure 2 data source: Data and Analytics Leadership Annual Executive Survey 2023, NewVantage Partners

NURTURING DATA LITERACY AND ACCOUNTABILITY

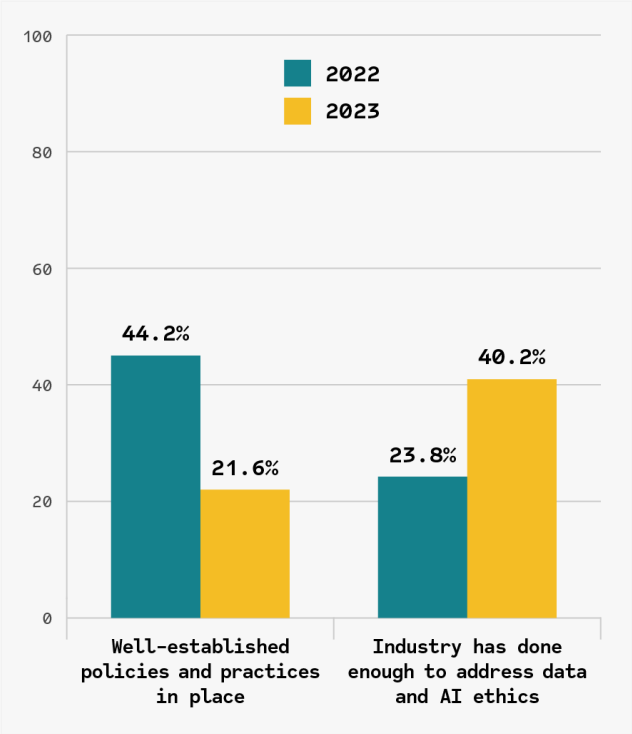
Promoting data literacy and accountability is not only about improving efficiency but also an ethical consideration. Assigning both ownership and accountability for data management empowers people to make informed decisions based on data insights, strengthens transparency, and upholds principles of responsibility and integrity, ensuring accuracy, security, and compliance with privacy regulations.

A data-literate workforce is a safeguard, identifying instances where data may be misused or manipulated for unethical purposes.

OVERCOMING RESISTANCE TO CHANGE

Incorporating observability practices is often a considerable challenge, and facing resistance from team members is not uncommon. However, you should confront these concerns and communicate clearly to promote a smooth transition. You can encourage adopting data-driven practices by highlighting the long-term advantages of better data quality and observability, which might inspire your coworkers to welcome changes. Showcasing real-life cases of positive outcomes, like higher revenue and customer satisfaction, can also help make a case.

Figure 3: The state of data responsibility and data ethics



Data source: Amount of data created, consumed, and stored 2010-2020, with forecasts to 2025, 2023, Statistica

Implementing Data Observability Techniques

You can keep your data pipelines reliable and at a high quality by implementing data observability. This implementation involves using different techniques and features that will allow you to monitor and analyze your data. Those processes include data profiling, anomaly detection, lineage, and quality checks. These tools will give you a holistic view of your data pipelines, allowing you to monitor its health and quickly identify any issues or inconsistencies that could affect its performance.

ESSENTIAL TECHNIQUES FOR SUCCESSFUL IMPLEMENTATION

To ensure the smooth operation of pipelines, you must establish a proper system for monitoring, troubleshooting, and maintaining data. Employing effective strategies can help achieve this goal. Let's review some key techniques to consider.

CONNECTIVITY AND INTEGRATION

For optimal data observability, your tools must integrate smoothly with your existing data stack. This integration should not require major modifications to your pipelines, data warehouses, or processing frameworks. This approach allows for an easy deployment of the tools without disrupting your current workflows.

DATA MONITORING AT REST

Observability tools should be able to monitor data while it's at rest without needing to extract it from the current storage location. This method ensures that the monitoring process doesn't affect the speed of your data pipelines and is cost effective. Moreover, this approach makes your data safer as it doesn't require extraction.

AUTOMATED ANOMALY DETECTION

Automated anomaly detection is an important component of data observability. Through machine learning models, patterns and behaviors in data are identified; this enables alerts to be sent when unexpected deviations occur, reducing the number of false positives and alleviating the workload of data engineers who would otherwise have to manage complex monitoring rules.

DYNAMIC RESOURCE IDENTIFICATION

Data observability tools give you complete visibility into your data ecosystem. These tools should automatically detect important resources, dependencies, and invariants. They should be flexible enough to adapt to changes in your data environment, giving you insights into vital components without constant manual updates and making data observability extensive and easy to configure.

COMPREHENSIVE CONTEXTUAL INFORMATION

For effective troubleshooting and communication, data observability needs to provide comprehensive contextual information. This information should cover data assets, dependencies, and reasons behind any data gaps or issues. Having the full context will allow data teams to identify and resolve any reliability concerns quickly.

PREVENTATIVE MEASURES

Data observability implements monitoring data assets and offers preventive measures to avoid potential issues. With insights into data and suggesting responsible alterations or revisions, you can proactively address problems before they affect data pipelines. This approach leads to greater efficiency and time savings in the long run. If you need to keep tabs on data, it can be tough to ensure everything is covered. Only using batch and stream processing frameworks isn't enough. That's why it's often best to use a tool specifically made for this purpose.

You could use a data platform, add it to your existing data warehouse, or opt for open-source tools. Each of these options has its own advantages and disadvantages:

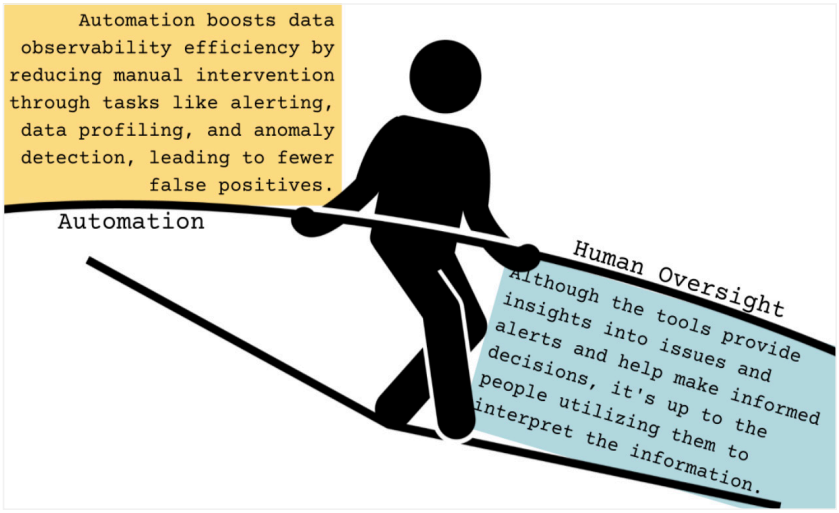
- **Use a data platform** – Data platforms are designed to manage all of your organization's data in one place and grant access to that data through APIs instead of via the platform itself.
  - There are many benefits to using a data platform, including speed, easy access to all your organization's information, flexible deployment options, and increased security. Additionally, many platforms include built-in capabilities for data observability, so you can ensure your databases perform well without having to implement an additional solution.
- **Build data observability into your existing platform** – If your organization only uses one application or tool to manage its data, this approach is probably the best for you, provided it includes an observability function.

Incorporating data observability into your current setup is a must-have if you manage complex data stored in multiple sources, thus improving the reliability of your data flow cycle.

BALANCING AUTOMATION AND HUMAN OVERSIGHT

While automation is a key component of data observability, it's important to strike a balance between automation and human oversight. While automation can help with routine tasks, human expertise is necessary for critical decisions and ensuring data quality. Implementing data observability techniques involves seamless integration, automated anomaly detection, dynamic resource identification, and comprehensive contextual information. Balancing automation and human oversight is important for efficient and effective data observability, resulting in more reliable data pipelines and improved decision-making capabilities.

Figure 4: Balancing automation and human oversight



Conclusion

In conclusion, data observability empowers organizations to thrive in a world where data fuels decision-making by ensuring data's accuracy, reliability, and trustworthiness. We can start by cultivating a culture that values data integrity, collaboration between technical and business teams, and a commitment to nurturing data literacy and accountability. You will also need a strong data observability framework to monitor your data pipelines effectively. This includes a set of techniques that will help identify issues early and optimize your data ecosystems.

But automated processes aren't enough, and we must balance our reliance on automation with human oversight, recognizing that while automation streamlines routine tasks, **human expertise remains invaluable** for critical decisions and maintaining data quality. With data observability, data integrity is safeguarded, and its full potential is unlocked — leading to innovation, efficiency, and success.

Joana da Silva Carvalho, Site Reliability Engineer at Virtuoso.qa
@radra on DZone

Joana has been a performance engineer for the last 10 years. She analyzed root causes from user interaction to bare metal, performance tuning, and new technology evaluation. Her goal is to create solutions to empower the development teams to own performance investigation, visualization, and reporting so that they can, in a self-sufficient manner, own the quality of their services.

CONTRIBUTOR INSIGHTS

Future-Proofing Data Architecture for Better Data Quality

Best Practices for Building Robust Data Engineering Systems to Deliver High-Quality Data

By Xinran Waibel, Founder of Data Engineer Things

Data quality is an undetachable part of data engineering. Because any data insight can only be as good as its input data, building robust and resilient data systems that consistently deliver high-quality data is the data engineering team's holiest responsibility. Achieving and maintaining adequate data quality is no easy task. It requires data engineers to design data systems with data quality in mind. In the hybrid world of data at rest and data in motion, engineering data quality could be significantly different for batch and event streaming systems.

This article will cover key components in data engineering systems that are critical for delivering high-quality data:

- **Monitoring data quality** – Given any data pipeline, how to measure the correctness of the output data, and how to ensure the output is correct not only today but also in the foreseeable future.
- **Data recovery and backfill** – In case of application failures or data quality violations, how to perform data recovery to minimize impact on downstream users.
- **Preventing data quality regressions** – When data sources undergo changes or when adding new features to existing data applications, how to prevent unexpected regression.

Monitoring Data Quality

As the business evolves, the data also evolves. Measuring data quality is never a one-time task, and it is important to continuously monitor the quality of data in data pipelines to catch any regressions at the earliest stage possible. The very first step of monitoring data quality is defining data quality metrics based on the business use cases.

DEFINING DATA QUALITY

Defining data quality is to set expectations for the output data and measure the deviation in the actual data from the established expectations in the form of quantitative metrics. When defining data quality metrics, the very first thing data engineers should consider is, "What truth does the data represent?" For example, the output table should contain all advertisement impression events that happened on the retail website. The data quality metrics should be designed to ensure the data system accurately captures that truth.

In order to accurately measure the data quality of a data system, data engineers need to track not only the baseline application health and performance metrics (such as job failures, completion timestamp, processing latency, and consumer lag) but also customized metrics based on the business use cases the data system serves. Therefore, data engineers need to have a deep understanding of the downstream use cases and the underlying business problems.

As the business model determines the nature of the data, business context allows data engineers to grasp the meanings of the data, traffic patterns, and potential edge cases.

While every data system serves a different business use case, some common patterns in data quality metrics can be found in Table 1.

Table 1

METRICS FOR MEASURING DATA QUALITY IN A DATA PIPELINE

Type	Limitations
Application health	The number of jobs succeeded or running (for streaming) should be N.
SLA/latency	The job completion time should be by 8 a.m. PST daily. The max event processing latency should be < 2 seconds (for streaming).
Schema	Column `account_id` should be `INT` type and can't be `NULL`.
Column values	Column `account_id` must be positive integers. Column `account_type` can only have the values: `FREE`, `STANDARD`, or `MAX`.
Comparison with history	The total number of confirmed orders on any date should be within +20%/-20% of the daily average of the last 30 days.
Comparison with other datasets	The number of shipped orders should correlate to the number of confirmed orders.

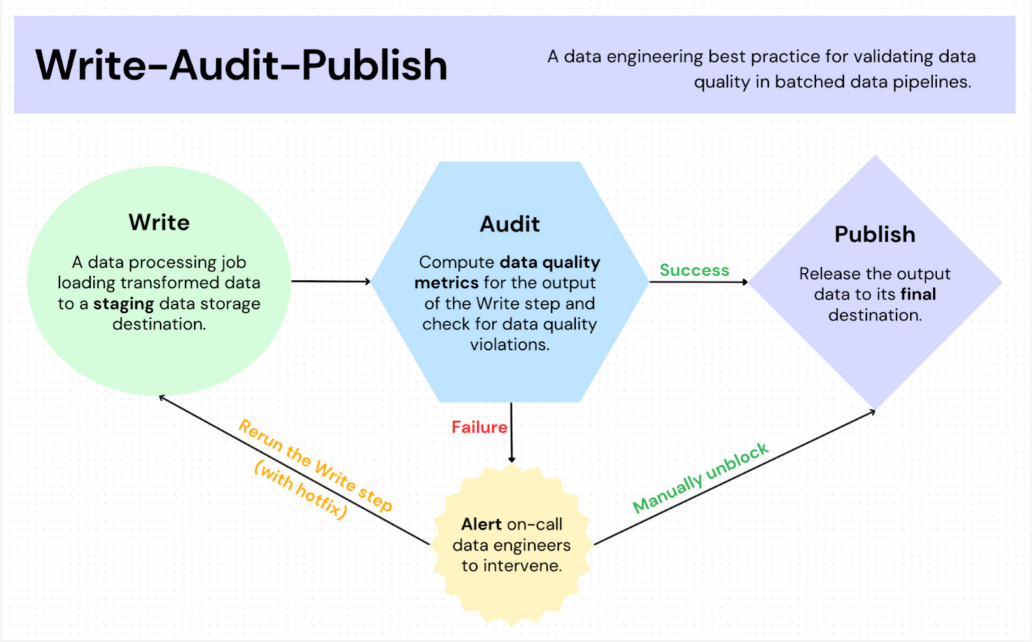
IMPLEMENTING DATA QUALITY MONITORS

Once a list of data quality metrics is defined, these metrics should be captured as part of the data system and metric monitors should be automated as much as possible. In case of any data quality violations, the on-call data engineers should be alerted to investigate further. In the current data world, data engineering teams often own a mixed bag of batched and streaming data applications, and the implementation of data quality metrics can be different for batched vs. streaming systems.

BATCHED SYSTEMS

The Write-Audit-Publish (WAP) pattern is a data engineering best practice widely used to monitor data quality in batched data pipelines. It emphasizes the importance of always evaluating data quality before releasing the data to downstream users.

Figure 1: Write-Audit-Publish pattern in batched data pipeline design



STREAMING SYSTEMS

Unfortunately, the WAP pattern is not applicable to data streams because event streaming applications have to process data nonstop, and pausing production streaming jobs to troubleshoot data quality issues would be unacceptable. In a Lambda architecture, the output of event streaming systems is also stored in lakehouse storage (e.g., an Apache Iceberg or Apache Hudi table) for batched usage. As a result, it is also common for data engineers to implement WAP-based batched data quality monitors on the lakehouse table.

To monitor data quality in near real-time, one option is to implement data quality checks as real-time queries on the output, such as an Apache Kafka topic or an Apache Druid datasource. For large-scale output, sampling is typically applied to improve the query efficiency of aggregated metrics. Helper frameworks such as Schema Registry can also be useful for ensuring output events have a compatible as-expected schema.

Another option is to capture data quality metrics in an event-by-event manner as part of the application logic and log the results in a time series data store. This option introduces additional side output but allows more visibility into intermediate data stages/operations and easier troubleshooting. For example, assuming the application logic decides to drop events that have invalid `account_id`, `account_type`, or `order_id`, if an upstream system release introduces a large number of events with invalid `account_id`, the output-based data quality metrics will show a decline in the total number output events. However, it would be difficult to identify what filter logic or column is the root cause without metrics or logs on intermediate data stages/operations.

Data Recovery and Backfill

Every data pipeline will fail at some point. Some of the common failure causes include:

- **Incompatible source data updates** (e.g., critical columns were removed from source tables)
- **Source or sink data systems failures** (e.g., sink databases became unavailable)
- **Altered truth in data** (e.g., data processing logic became outdated after a new product release)
- **Human errors** (e.g., a new build introduces new edge-case errors left unhandled)

Therefore, all data systems should be able to be backfilled at all times in order to minimize the impact of potential failures on downstream business use cases. In addition, in event streaming systems, the ability to backfill is also required for bootstrapping large stateful stream processing jobs.

The data storage and processing frameworks used in batched and streaming architectures are usually different, and so are the challenges that lie behind supporting backfill.

BATCHED SYSTEMS

The storage solutions for batched systems, such as AWS S3 and GCP Cloud Storage, are relatively inexpensive and source data retention is usually not a limiting factor in backfill. Batched data are often written and read by event-time partitions, and data processing jobs are scheduled to run at certain intervals and have clear start and completion timestamps.

The main technical challenge in backfilling batched data pipelines is data lineage: what jobs updated/read which partitions at what timestamp. Clear data lineage enables data engineers to easily identify downstream jobs impacted by problematic data partitions. Modern lakehouse table formats such as Apache Iceberg provide queryable table-level changelogs and history snapshots, which allow users to revert any table to a specific version in case a recent data update contaminated the table. The less queryable data lineage metadata, the more manual work is required for impact estimation and data recovery.

STREAMING SYSTEMS

The source data used in streaming systems, such as Apache Kafka topics, often have limited retention due to the high cost of low-latency storage. For instance, for web-scale data streams, data retention is often set to several hours to keep costs reasonable. As troubleshooting failures can take data engineers hours if not days, the source data could have already expired before backfill. As a result, data retention is often a challenge in event streaming backfill.

Below are the common backfill methodologies for event streaming systems:

Table 2

METHODS FOR BACKFILLING STREAMING DATA SYSTEMS

Method	Description
Replaying source streams	Reprocess source data from the problematic time period before those events expire in source systems (e.g., Apache Kafka). Tiered storage can help reduce stream retention cost.
Lambda architecture	Maintain a parallel batched data application (e.g., Apache Spark) for backfill, reading source data from a lakehouse storage with long retention.
Kappa architecture	The event streaming application is capable of streaming data from both data streams (for production) and lakehouse storage (for backfill)
Unified batch and streaming	Data processing frameworks, such as Apache Beam, support both streaming (for production) and batch mode (for backfill).

Preventing Data Quality Regressions

Let's say a data pipeline has a comprehensive collection of data quality metrics implemented and a data recovery mechanism to ensure that reasonable historical data can be backfilled at any time. What could go wrong from here? Without prevention mechanisms, the data engineering team can only react passively to data quality issues, finding themselves busy putting out the same fire over and over again. To truly future-proof the data pipeline, data engineers must proactively **establish programmatic data contracts** to prevent data quality regression at the root.

Data quality issues can either come from upstream systems or the application logic maintained by data engineers. For both cases, data contracts should be implemented programmatically, such as unit tests and/or integration tests to stop any contract-breaking changes from going into production.

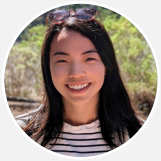
For example, let's say that a data engineering team owns a data pipeline that consumes advertisement impression logs for an online retail store. The expectations of the impression data logging should be implemented as unit and/or regression tests in the client-side logging test suite since it is owned by the client and data engineering teams. The advertisement impression logs are stored in a Kafka topic, and the expectation on the data schema is maintained in a Schema Registry to ensure the events have compatible data schemas for both producers and consumers.

As the main logic of the data pipeline is attributing advertisement click events to impression events, the data engineering team developed unit tests with mocked client-side logs and dependent services to validate the core attribution logic and integration tests to verify that all components of the data system together produce the correct final output.

Conclusion

Data quality should be the first priority of every data pipeline and the data architecture should be designed with data quality in mind. The first step of building robust and resilient data systems is defining a set of data quality metrics based on the business use cases. Data quality metrics should be captured as part of the data system and monitored continuously, and the data should be able to be backfilled at all times to minimize potential impact to downstream users in case of data quality issues. The implementation of data quality monitors and backfill methods can be different for batched vs. event streaming systems. Last but not least, data engineers should establish programmatic data contracts as code to proactively prevent data quality regressions.

Only when the data engineering systems are future-proofed to deliver qualitative data, data-driven business decisions can be made with confidence.



Xinran Waibel, Founder of Data Engineer Things
@xinran.waibel on DZone | @xinranwaibel on LinkedIn | @xinran_waibel on X (Twitter)

Xinran is a senior data engineer with over seven years of industry experience in designing and developing data systems. She is an active blog writer and the founder of Data Engineer Things, an online community dedicated to data engineering. Xinran has worked at Netflix, Confluent, and Target, where she leveraged modern data stacks to enable data-driven decision-making.

ADDITIONAL RESOURCES

Diving Deeper Into Data Pipelines

DZONE EVENT STREAMS

How to Go Pipeline-Free With Real-Time Analytics [Fireside Chat]

Real-time analytics frequently necessitates extensive data pipelines to achieve satisfactory performance, but these pipelines add complexity that eat away at your time and money. This talk explores the concept of "pipeline-free" real-time analytics, a novel approach to these problems that promises simplicity and improved efficiency.

How to Overcome Data Governance Challenges and Drive Data Health [Webinar]

In today's data-driven world, organizations face increasing data governance challenges in managing and utilizing their data assets effectively. With the sheer volume of data being generated, it is essential to maintain data health by ensuring that data is accurate, accessible, and secure — but it isn't always easy. Check out this webinar!

TREND REPORTS

Data Pipelines: Ingestion, Warehousing, and Processing

In DZone's 2022 *Data Pipelines* Trend Report, we review the key components of a data pipeline; explore the differences between ETL, ELT, and reverse ETL; propose solutions to common data pipeline design challenges; dive into engineered decision intelligence; and provide an assessment on the best approaches to modernize testing practices with data synthesis.

Database Systems: Architecture Advancements and Data Storage [...]

Today, we are in a new era of the "Modern Database," where databases must both store data and ensure that data is prepped and primed securely for insights and analytics, integrity and quality, and microservices and cloud-based architectures. In this 2023 *Database Systems* Trend Report, we explore DB trends, assess strategies and challenges, and provide forward-looking assessments of commonly used technologies.

REFCARDS

Getting Started With Real-Time Analytics

Real-time analytics is necessary for any business that needs to make decisions in hours, minutes, or seconds. Implementing real-time analytics requires processing high volumes of input data and matching it with existing data in minutes, seconds, or even less time.

This Refcard aims to acquaint readers with real-time analytics, where it is used, how it works, and the challenges involved.

Apache Kafka Patterns and Anti-Patterns

Kafka has an ecosystem of open-source components that, when combined, help store, process, and integrate data streams with other parts of your system in a secure, reliable, and scalable manner.

This Refcard dives into select patterns and anti-patterns, covering topics such as reliable messaging, scalability, error handling, and more.

MULTIMEDIA

Data Engineer Things [blog]

Focused on providing resources for data engineers of any level, Data Engineer Things is a great blog to stop by for assistance on or inspiration for your latest project. Covering everything from architecture and AI to career paths, you're sure to find fantastic tidbits of information.

DataFramed [podcast]

This weekly podcast explores the way AI is changing the way we work with data and how we understand the world. Meant for engineers and developers of any level, episodes include educational discussions and insightful conversations with industry experts that span from technical topics to job advice.

@dataengineeringvideos [YouTube Channel]

Gowtham's channel is dedicated to all things data engineering, including data modeling, ETL/ELT, data warehousing, and more. Learn about the hottest tools and latest industry news with videos averaging 10-15 minutes, some of which are available in both English and Tamil!

ADDITIONAL RESOURCES

Solutions Directory

This directory contains tools for analytics, BI, cloud, integration, storage, and more to manage data from end to end. It provides free trial data and product category information gathered from vendor websites and project pages. Solutions are selected for inclusion based on several impartial criteria, including solution maturity, technical innovativeness, relevance, and data availability.

DZONE'S 2023 DATA PIPELINES SOLUTIONS DIRECTORY

	Company	Product	Purpose	Availability	Website
2023 PARTNERS	CelerData	CelerData Cloud	Analytical database	Free tier	celerdata.com/celerdata-cloud
	Decodable	Decodable	Platform as a Service	Trial period	decodable.co/product
	Informatica	Intelligent Data Management Cloud™	AI-powered comprehensive data integration and management	By request	informatica.com/platform
	Prophecy	Prophecy	Low-code data transformation	Trial period	prophecy.io/low-code-platform
	Company	**Product**	**Purpose**	**Availability**	**Website**
	1010data	Insights Platform	End-to-end data management, analytics	By request	1010data.com
	Actian	DataConnect	Low-code data integration	By request	actian.com/data-integration/dataconnect
		DataFlow	Extract, analyze, transform, and load data		actian.com/data-integration/dataflow
	Ahana	Ahana Cloud	SaaS for PrestoDB	Free tier	ahana.io/ahana-cloud
	Alation	Alation	Data intelligence	By request	alation.com
	Alluxio	Alluxio	Data orchestration	Open source	alluxio.io
	Altair	Grid Engine	Distributed resource management and optimization	Trial period	altair.com/grid-engine
		Monarch	Self-service data prepartion		altair.com/monarch
		Panopticon	Data visualization, streaming analytics	By request	altair.com/panopticon
		Unlimited	Data analytics appliance		altair.com/altair-unlimited-data-analytics
	Alteryx	Analytics Cloud	Automated data analytics at scale	Trial period	alteryx.com/products/alteryx-cloud
	Apache Software Foundation	Avro	Data serialization system	Open source	avro.apache.org
		Flink	Stream- and batch-processing framework		flink.apache.org
		Hive	Distributed, fault-tolerant data warehouse		hive.apache.org
		Iceberg	Open table format for analytic datasets		iceberg.apache.org
		Kafka	Distributed event streaming		kafka.apache.org
		Kudu	Distributed data storage engine		kudu.apache.org
		NiFi	Automate data flow between systems		nifi.apache.org
		Ozone	Scalable, distributed storage for analytics, data, cloud-native apps		ozone.apache.org
		Paimon	Streaming data lake		paimon.apache.org
		Pinot	Real-time distributed OLAP datastore		pinot.apache.org
		Pulsar	All-in-one messaging and streaming		pulsar.apache.org
		Superset	Modern data exploration, visualization		superset.apache.org

DZONE'S 2023 DATA PIPELINES SOLUTIONS DIRECTORY				
Company	**Product**	**Purpose**	**Availability**	**Website**
Apple	Core Data	ORM	Free	developer.apple.com/documentation/coredata
AtScale	AtScale	Semantic layer tool	By request	atscale.com
AWS	Amazon Athena	Analyze petabyte-scale data	By request	aws.amazon.com/athena
	Amazon EMR	Big data platform		aws.amazon.com/emr
	Amazon Kinesis	Process and analyze real-time streaming data		aws.amazon.com/kinesis
	Amazon Redshift	Cloud data warehouse	Trial period	aws.amazon.com/redshift
	Glue	Serverless data integration	Free tier	aws.amazon.com/glue
	Lake Formation	Data lake governance	By request	aws.amazon.com/lake-formation
Bitam	Artus	BI for data visualization	By request	bitam.com/artus
Board	Board	Intelligent planning	By request	board.com
CDAP	CDAP	Data analytics	Sandbox	cdap.io
Cloudera	Cloudera Data Platform	Hybrid data management and analytics	By request	cloudera.com/products/cloudera-data-platform
Coginiti	Coginiti	AI-enabled enterprise data workspace	Trial period	coginiti.co
Confluent	Confluent Cloud	Cloud-native service for Kafka	Trial period	confluent.io/confluent-cloud
	Confluent Platform	Enterprise-grade Kafka distribution		confluent.io/product/confluent-platform
CyberGRX	CyberGRX	Data analytics for 3P risk management	By request	cybergrx.com
DAS42	DAS42	End-to-end data consulting and implementation	By request	das42.com
data.world	data.world	Data catalog	By request	data.world
Databricks	Databricks	Data lakehouse	Trial period	databricks.com
Datameer	Datameer	Snowflake analytics business platform	Trial period	datameer.com
DataRobot	DataRobot	Data science democratization AI platform	By request	datarobot.com
Datio	Datio	Data analytics consulting for banking	By request	datio.com
Delphix	DevOps Data Platform	API-first multi-cloud data platform	By request	delphix.com
Delta Lake	Delta Lake	Storage framework for building lakehouse architectures	Open source	delta.io
Digdag	Digdag	Build, run, schedule, and monitor complex task pipelines	Open source	digdag.io
Domo	Domo Data Experience Platform	BI and data visualization	Trial period	domo.com
Dremio	Dremio Cloud	Open data lakehouse	Free tier	dremio.com
Embulk	Embulk	Pluggable bulk data loader	Open source	embulk.org
EngineRoom by MoreSteam	EngineRoom	Web-based data analytics	Trial period	moresteam.com/engineroom
Eureka Security	Eureka	Cloud data security posture management	By request	eureka.security
Exaptive	Cognitive City	Collective intelligence tool	Trial period	exaptive.com/cognitive-city
	Studio	Data visualization		exaptive.com/studio

DZONE'S 2023 DATA PIPELINES SOLUTIONS DIRECTORY

Company	Product	Purpose	Availability	Website
Exasol	Exasol	In-memory database for analytics	Trial period	exasol.com
FICO	FICO Platform	Applied intelligence	By request	fico.com/en/fico-platform
Fivetran	Fivetran	Automated data movement	Free tier	fivetran.com
GoodData	GoodData	Cloud BI and analytics	Trial period	gooddata.com
Google Cloud	Big Query	Enterprise data warehouse	Trial period	cloud.google.com/bigquery
	Looker	BI		cloud.google.com/looker
Hazelcast	Hazelcast Platform	Continuous data processing	By request	hazelcast.com/products/hazelcast-platform
	Viridian	Managed cloud service	Free tier	hazelcast.com/products/viridian
Hitachi	Lumada	Digital solutions, services, and tech for data insights	By request	hitachi.com/products/it/lumada/global
Hitachi Vantara	Content Intelligence	Intelligent data discovery and transformation	By request	hitachivantara.com/en-us/products/storage-platforms/file-object-storage/content-intelligence
	Content Platform	Object storage for hybrid cloud		hitachivantara.com/en-us/products/storage-platforms/file-object-storage/content-platform
	Pentaho Data Platform	Intelligent DataOps		hitachivantara.com/en-us/products/pentaho-platform
IBM	Analytics Engine	Provision, manage, and run Hadoop and Spark clusters	Trial period	ibm.com/cloud/analytics-engine
	InfoSphere Information Server	Data integration	By request	ibm.com/information-server
	Streams	Real-time analytics		ibm.com/cloud/streaming-analytics
Incorta	Incorta	Open data delivery	Trial period	incorta.com
Indium Software	ibriX	Databricks AI platform	By request	indiumsoftware.com/ibrix
	iDAF	Data assurance framework		indiumsoftware.com/idaf
	Indium Software	Digital engineering services		indiumsoftware.com
Infor	Data Lake	Enterprise data capture repository	By request	infor.com/products/data-lake
Kyligence	Enterprise	Sub-second standard SQL query responses based on Apache Kylin	By request	kyligence.io/kyligence-enterprise
	Zen	Low-code metrics	Trial period	kyligence.io/zen
Kyvos Insights	Kyvos	Cloud-native analytics acceleration	Trial period	kyvosinsights.com
Microsoft	Power BI	Data visualization	Trial period	powerbi.microsoft.com/en-us
Microsoft Azure	Databricks	Open data lakehouse	Trial period	azure.microsoft.com/en-us/products/databricks
	Stream Analytics	Serverless real-time analytics from cloud to edge		azure.microsoft.com/en-us/products/stream-analytics
	Synapse Analytics	Enterprise analytics service		azure.microsoft.com/en-us/products/synapse-analytics
MicroStrategy	ONE	Analytics	By request	microstrategy.com/en/enterprise-analytics
Neo4j	Graph Data Science	Graph database and analytics	Free tier	neo4j.com/product/graph-data-science
Nordcloud	Nordcloud	Cloud strategy, app development, and managed services	By request	nordcloud.com

DZONE'S 2023 DATA PIPELINES SOLUTIONS DIRECTORY

Company	Product	Purpose	Availability	Website
OpenText	Contivo	Data mapping for enterprise integration teams	By request	opentext.com/products/data-transformation
	Lens	End-to-end integrated data flow visibility		opentext.com/products/lens-data-visibility
	Magellan BI & Reporting	Embedded self-service reports and dashboards		opentext.com/products/magellan-bi-and-reporting
	Magellan Data Discovery	Access, blend, explore, and analyze data without IT		opentext.com/products/magellan-data-discovery
Oracle	Oracle Analytics Platform	Analytics cloud and server	Trial period	oracle.com/business-analytics/analytics-platform
Panorama Education	Panorama	Data visualization for education	By request	panoramaed.com/product-overview
Pepperdata	Capacity Optimizer Classic	Resource utilization and cluster efficiency	By request	pepperdata.com/capacity-optimizer-classic
PKWARE	PK Protect® Data Protection Platform	End-to-end data proptection and privacy	By request	pkware.com/products/pk-protect
Presto Foundation	Presto	SQL query engine	Open source	prestodb.io
Progress Software	DataDirect	Cloud and on-prem data connectivity	Trial period	progress.com/datadirect-connectors
	MarkLogic Data Platform	Simplify complex data and achieve data agility	By request	marklogic.com/product/platform
Promethium	Augmented Self Service Analytics	Self-service analytics and data management	By request	promethium.ai/augmented-analytics
	Data Fabric	Distributed data and environment management	Trial period	promethium.ai/product-overview
Pyramid Analytics	Pyramid Decision Intelligence Platform	Integrated, no-code, AI-driven data	By request	pyramidanalytics.com/decision-intelligence-platform
Qlik	Data Integration	DataOps for analytics	Trial period	qlik.com/us/products/qlik-data-integration
	Sense	Cloud analytics		qlik.com/us/products/qlik-sense
RapidMiner	RapidMiner Platform	Enterprise data science	By request	rapidminer.com
Redpoint Global	rg1	Configurable, integrated customer data	By request	redpointglobal.com
Rockset	Rockset	Real-time search and analytics database	Trial period	rockset.com
Rudderstack	Rudderstack	End-to-end customer data toolset	Free tier	rudderstack.com
SAP	HANA Cloud	DBaaS for intelligent data apps and analytics	Trial period	sap.com/products/technology-platform/hana
SAS	SAS Viya	AI and analytics	Trial period	sas.com/en_us/software/viya
Sisense	Fusion Platform	AI-driven data and embedded analytics	By request	sisense.com/platform
Snowflake	Snowflake	Data cloud	Trial period	snowflake.com
SolarWinds	Loggly	Log management	Trial period	loggly.com
Splunk	Cloud Platform	Cloud-powered insights for petabyte-scale data analytics	Trial period	splunk.com/en_us/products/splunk-cloud-platform
	Splunk Enterprise	Data search, analysis, and visualization		splunk.com/en_us/products/splunk-enterprise

DZONE'S 2023 DATA PIPELINES SOLUTIONS DIRECTORY				
Company	**Product**	**Purpose**	**Availability**	**Website**
SQream	Panoply	Managed ELT and cloud data warehouse	Trial period	panoply.io
	SQream Blue	Data lakehouse, SQL data preparation	By request	sqream.com/product/blue
	SQreamDB	Petabyte-scale data warehouse and SQL database		sqream.com/product/sqreamdb
Starburst Data	Starburst Galaxy	Fully managed data lake analytics	Trial period	starburst.io/platform/starburst-galaxy
StreamSets	StreamSets	Data pipeline development, management	Trial period	streamsets.com
Tableau Software	Tableau Platform	End-to-end data and analytics	Trial period	tableau.com/products/our-platform
Talend	Data Fabric	Unified, reliable, accessible data	Trial period	talend.com/products/data-fabric
	Stitch	Fully managed data pipeline for analytics		stitchdata.com
TARGIT	Decision Suite	BI and analytics tool	By request	targit.com/targit-decision-suite
teX.ai	teX.ai	Text analytics accelerator	By request	tex-ai.com
ThoughtSpot	ThoughtSpot Analytics	AI-powered analytics for data stacks	Trial period	thoughtspot.com/product
	ThoughtSpot Everywhere	Embedded interactive analytics		thoughtspot.com/everywhere
TIBCO	EBX® Software	Data governance, managment, sharing	By request	tibco.com/products/tibco-ebx-software
	Spotfire®	Data visualization and analytics	Trial period	tibco.com/products/tibco-spotfire
	Streaming	Enterprise-grade and cloud-ready streaming analytics		tibco.com/products/tibco-streaming
Treasure Data	Customer Data Platform	Customer data management	By request	treasuredata.com/product
Trino	Trino	Big data distributed SQL query engine	Open source	trino.io
Unravel	Unravel	Data observability, intelligence, automation	Free tier	unraveldata.com
VMware	Tanzu Gemfire	In-memory data and compute grid	Free tier	tanzu.vmware.com/gemfire
	Tanzu Hub	Multi-cloud management	Trial period	vmware.com/products/aria-hub-powered-by-aria-graph
Yellowfin	Yellowfin Analytics Platform	Enterprise and embedded analytics	By request	yellowfinbi.com/suite
Zaloni	Arena	Automated data governance, observability	By request	zaloni.com

DZone

3343 Perimeter Hill Dr, Suite 100
Nashville, TN 37211
888.678.0399 | 919.678.0300

At DZone, we foster a collaborative environment that empowers developers and tech professionals to share knowledge, build skills, and solve problems through content, code, and community. We thoughtfully — and with intention — challenge the status quo and value diverse perspectives so that, as one, we can inspire positive change through technology.