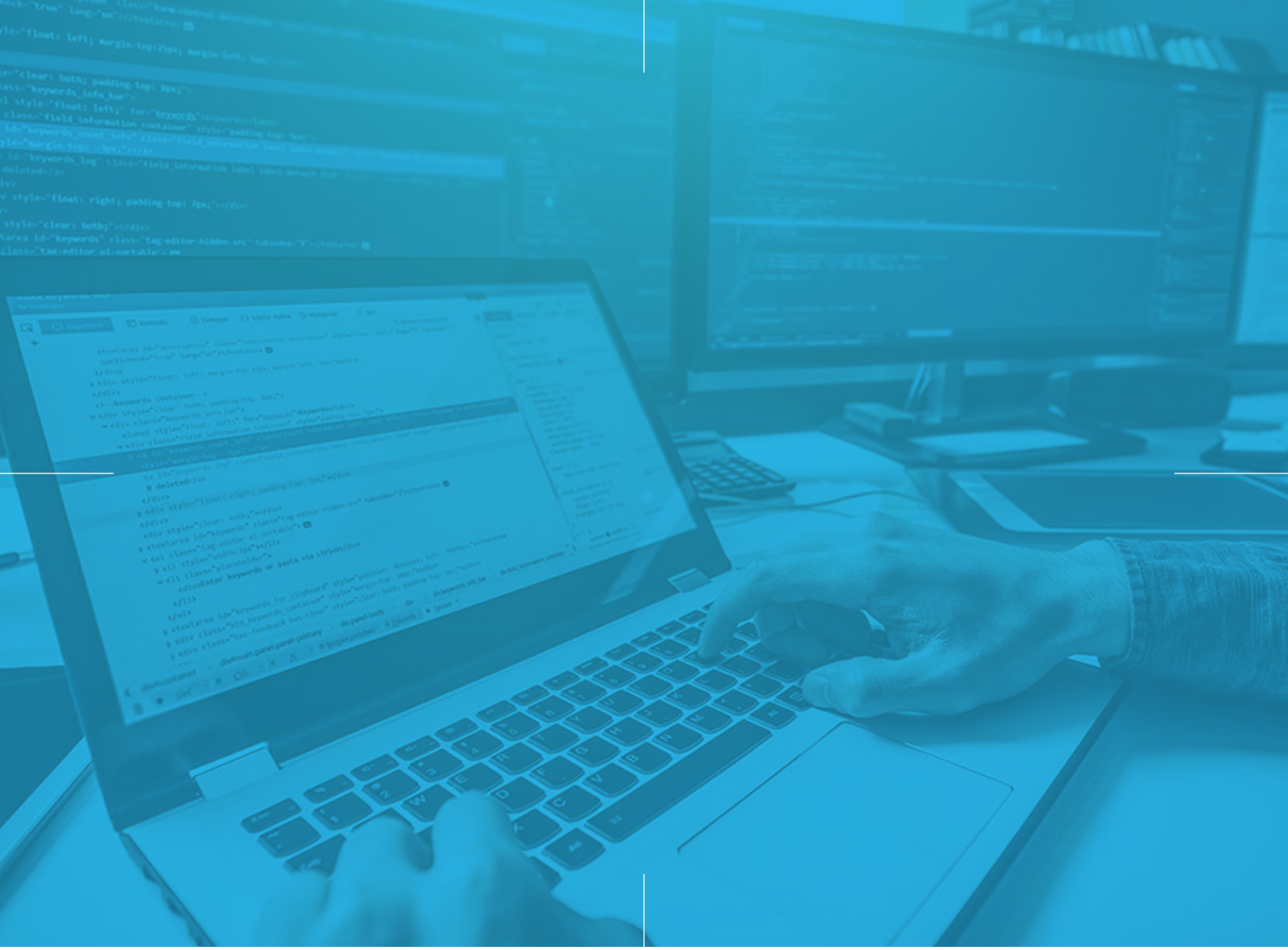

SNOWFLAKE FOR DATA ENGINEERING:

EASILY INGEST, TRANSFORM, AND DELIVER DATA FOR UP-TO-THE-MOMENT INSIGHT

snowflake

WHITE PAPER

DATA PIPELINE PERFORMANCE IS THE FOUNDATION OF A DATA-DRIVEN ORGANIZATION

Data pipelines are the lifeblood of the modern enterprise. Efficient data pipelines can be the difference between an architecture that delivers real value to the business or one that becomes a burden.

Enterprises receive data 24 hours a day, seven days a week, from the web, enterprise applications, and mobile and IoT devices. Your data pipeline has to load and process that data while data scientists and analysts are analyzing the data and downstream applications are processing it for further use.

Data engineers must collect, transform, and deliver data to different lines of business, while keeping up with the latest technology innovations to stay ahead of business demands. However, traditional legacy architectures create challenges every step of the way, making the task of the data engineer more complex.

HOW LEGACY ARCHITECTURES AFFECT DATA PIPELINES

Traditional data architectures were not built to meet the requirements of modern analytics and efficient data engineering. Figure 1 shows a typical data pipeline seen in many enterprises: a complex web of technologies stitched together to deliver data from its raw source to its intended destination.

Across the integration, transformation, aggregation, and delivery steps, organizations deploy multiple tools, with each connection point a possible point of failure. Governance, security, and silo issues are commonplace with this approach, as multiple copies of data eventually emerge to meet the varying needs of different parts of the business. In the end, organizations spend more time managing infrastructure and little time actually working with data.

"The increasing diversity of data, and the need to provide the right data to the right people at the right time, has created demand for the data engineering practice. Data and analytics leaders must integrate the data engineering discipline into their data management strategy."

Gartner Research
Data Engineering Is Critical to Driving Data and Analytics Success,
December 18, 2019 [1]

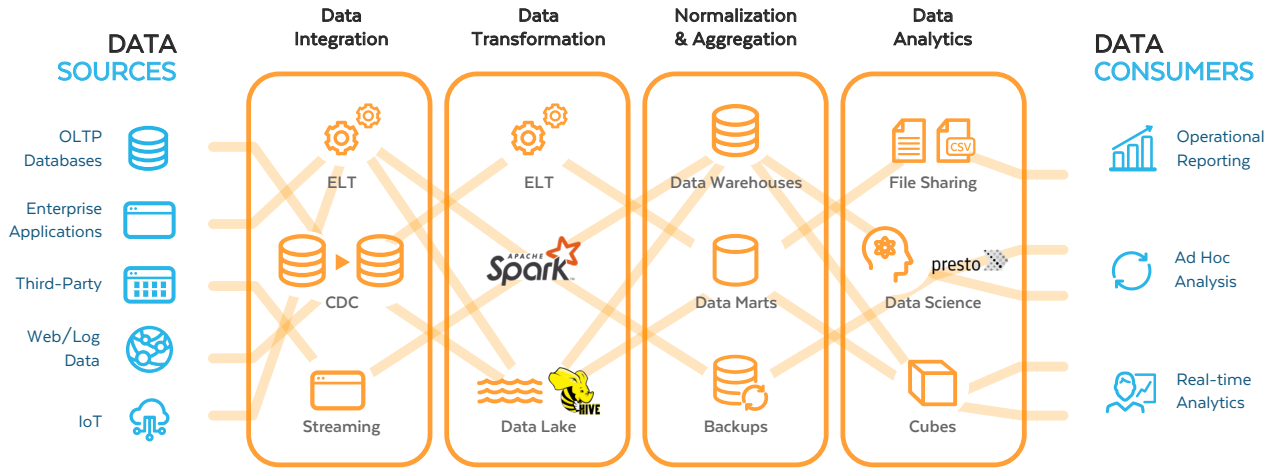

Figure 1: Traditional data architecture is complex, costly, and constrained.

For a data engineer, traditional legacy architectures can lead to the challenges summarized in Figure 2:

- Siloed, diverse data is integrated at different speeds
- Performance and reliability degrades due to resource contention
- Complex pipelines and architecture require excessive time and diverse skill sets to build and manage

Siloed, diverse data

With a legacy architecture, data often lands in different databases, data lakes, and data marts, as well as in the cloud and on premises. This data exists in many formats and is being generated at different speeds. Adding to the complexity, different applications have varying expectations for how data should be delivered. When data availability is delayed, applications and analysts work on stale data, causing decision makers to overlook key insights and forcing them to work on incomplete views of the business.

Reliability and performance degradation

Increased workloads in complex data pipelines result in reduced reliability and slow data transformation. When pipelines don't run as intended because resources are scarce, data engineers fight to fix

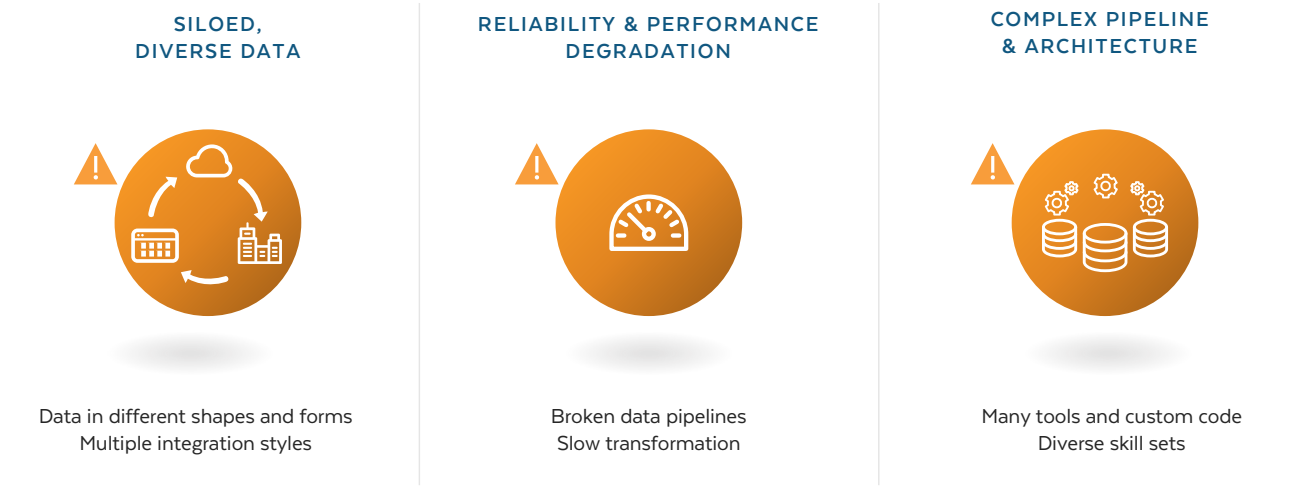

Figure 2: Data engineers face architecture and pipeline challenges.

the pipelines so their data consumers don't suffer. Dedicated load windows don't solve the problem—they can simply lead to slow, clunky pipelines and stale data.

Complex pipelines and architecture

Data engineering often requires custom coding, a variety of tools, and diverse skill sets. Complex pipelines and data services built on systems stitched together mean more infrastructures to configure and manage, adding to the complexity and cost of building and maintaining those pipelines. Capacity planning, performance tuning, and concurrency handling make pipeline maintenance even more complex. The end result? A data engineer may spend more time building complex pipelines, APIs, and managing and maintaining infrastructure than delivering data to meet business SLAs.

SNOWFLAKE HELPS DATA ENGINEERS DELIVER UP-TO-THE-MOMENT INSIGHT

Snowflake streamlines data engineering, delivering performance and simplicity so you can focus on getting more value from data, rather than managing the pipelines and infrastructure used to deliver it. Snowflake's multi-cluster shared data architecture can allocate multiple independent, isolated clusters for processing, data loading, transformation, and analytics—while concurrently sharing the same data—without resource contention. You can process data exactly where it is without unnecessary data movement or transformation.

As shown in Figure 3, Snowflake for data engineering enables you to:

- Integrate your data, streaming or bulk, in a single platform
- Run data pipelines with instant scalability and no resource contention
- Enhance productivity with streamlined architecture and extensible data pipelines

Snowflake is ideal for data engineering because it:

- Handles both structured and semi-structured data types easily (and soon, it will support unstructured data)
- Ingests data in bulk or near real time
- Does automatic error handling and data deduplication upon loading
- Allows you to process your data directly in Snowflake, using its powerful, best-of-breed transformation engine
- Simplifies data pipelines and architecture by leveraging SQL and extensibility features (such as functions and stored procedures)
- Enables you to easily and securely collaborate and deliver data to your users through live data sharing
- Provides built-in support for common drivers (ODBC, JDBC, Python, Spark) and the ability to export to your data lake

See the following sections for more about how to use Snowflake for data ingestion, transformation, and delivery.

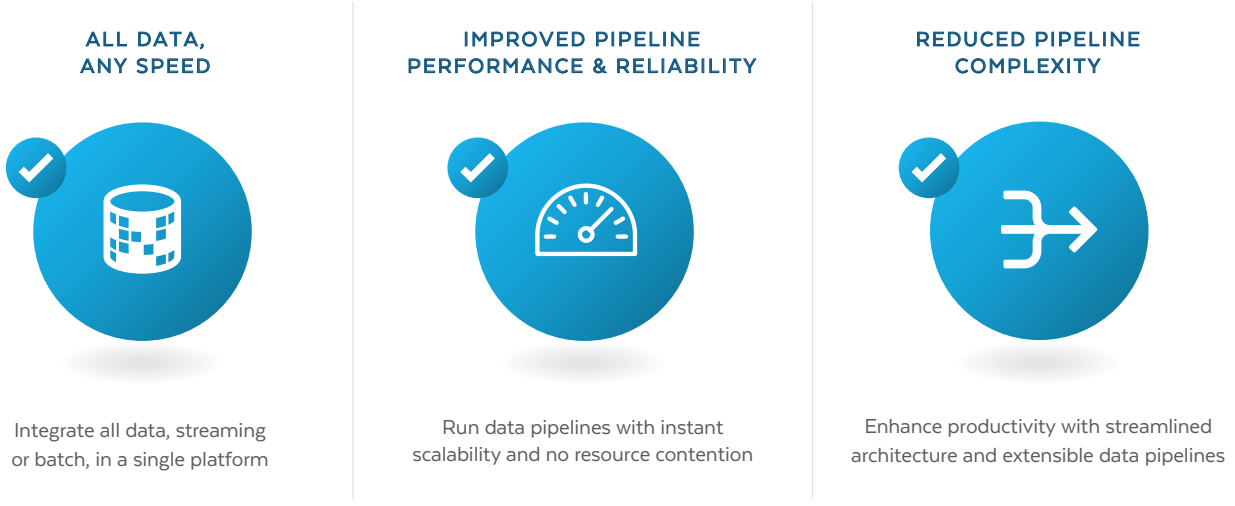

Figure 3: Snowflake streamlines data engineering so you can focus on working with your data.

INGEST ALL DATA, AT ANY SPEED

Data pipelines in Snowflake can be bulk or continuous, and processing can happen directly within Snowflake itself. Thanks to Snowflake's multi-cluster compute approach, these pipelines can handle complex transformations without impacting the performance of other workloads. All of your data can easily be ingested through streaming or batch and securely centralized, making both structured and semi-structured data available natively to your users.

As shown in the table below and explained in the following sections, Snowflake offers you several methods to ingest data.

USE THIS	TO DO THIS
Bulk loading	Load batches of data from files already available in cloud storage or copy data files from a local machine to Snowflake, and load the data into tables using the COPY command
Continuous loading with Snowpipe	Load small volumes of data (micro-batches) and incrementally make them available for analysis
Snowflake Connector for Kafka	Ingest event streams from Apache Kafka and load into a Snowflake table
Snowflake Data Marketplace	Ingest ready-to-query, live data published by others

For a comparison of how Snowflake processes bulk, continuous, and Apache Kafka data loading, see Figure 4 below.

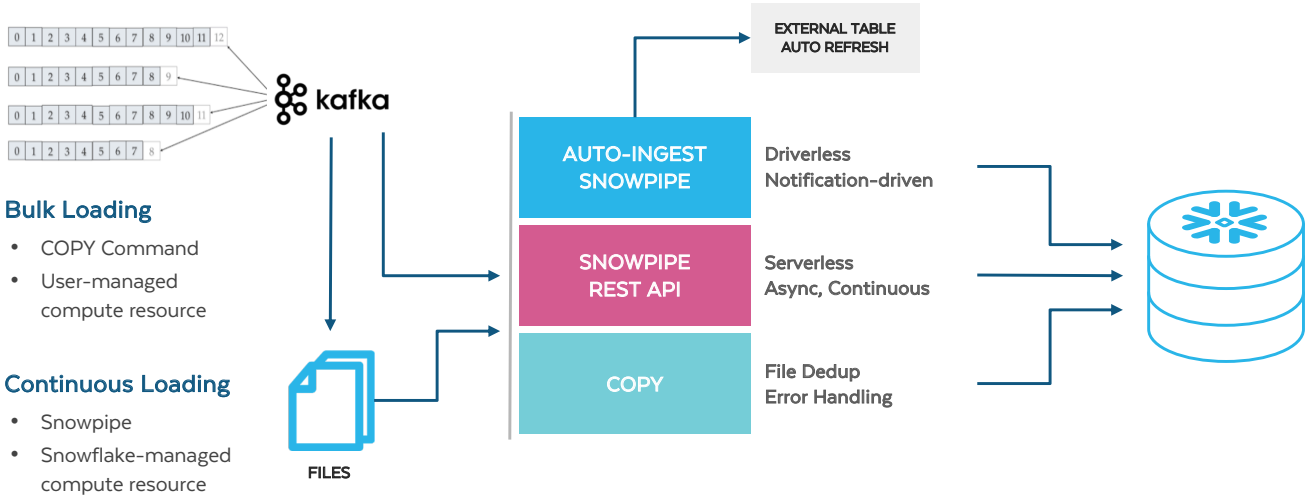

Figure 4: Snowflake gives data engineers ultimate flexibility in how to load their data.

Bulk data loading using the COPY command

Depending on the volume of data you have to load and the frequency (or infrequency) of loading it, you may decide to use the COPY command for bulk data loading. This option enables you to load batches of data from files already available in cloud storage, or copy (stage) data files from a local machine to an internal Snowflake cloud storage location before loading the data into tables using the COPY command.

Bulk loading relies on user-provided virtual warehouses, which you specify in the COPY statement. You must size the warehouse appropriately to accommodate expected loads.

Snowflake supports simple transformations while loading data into a table using the COPY command. Options include:

- Column reordering
- Column omission
- Casts
- Truncating text strings that exceed the target column length

Continuous data loading using Snowpipe

As data architectures evolve, so do the requirements for how data is loaded and processed within a data warehouse. Rapid and immediate access to live data, and the ability to analyze it in the moment, are key requirements so the data-driven enterprise can streamline operations, better serve customers, and uncover new market opportunities.

Snowpipe is a serverless tool that enables you to automatically load data from files as soon as they're available in a stage. Designed to load small volumes of data—micro-batches—and incrementally make them available for analysis, Snowpipe uses notifications from services such as Amazon Simple Queue Service (SQS) to listen for files and load them as they land. Snowpipe loads data within minutes after files are added to a stage and submitted for ingestion, ensuring that users have the latest results as soon as the raw data is available.

Because data loading happens continuously as new data arrives in an AWS, Azure, or Google object store, you no longer need scripts and scheduling tools. In addition, data pipelines can leverage Snowpipe to continuously load micro-batches of data into staging tables for transformation and optimization, using automated tasks and the CDC information in streams.

Figure 5 details the benefits of using Snowpipe for data ingestion.

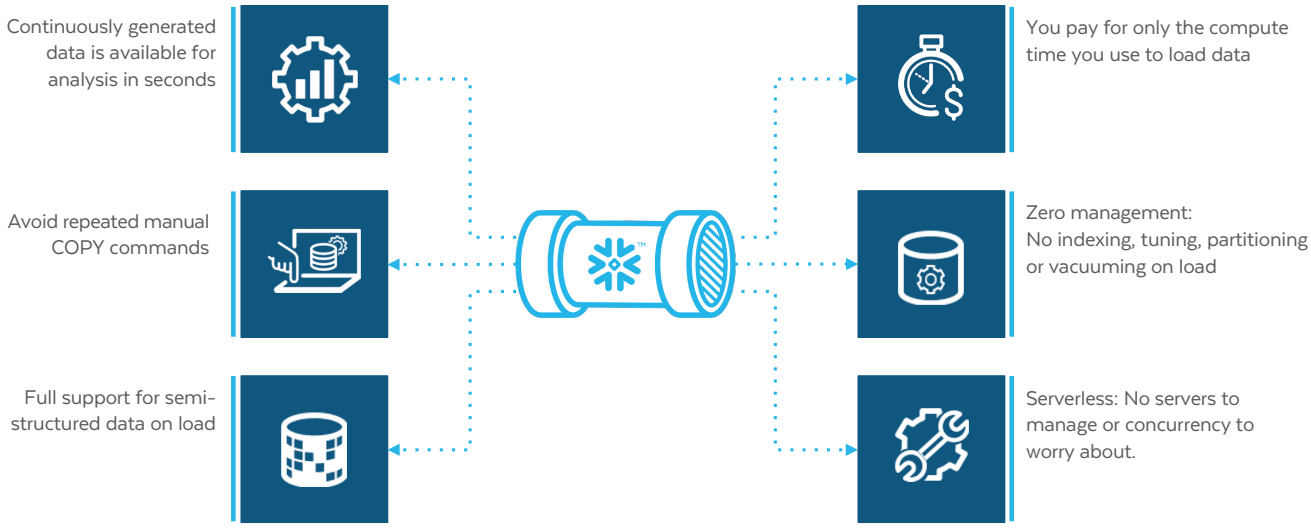

Figure 5: Snowpipe simplifies the process of moving files from object stores into Snowflake.

Event stream loading using Snowflake Connector for Kafka

To reap the value of data, you need to collect it, store it, and analyze it. Apache Kafka is the system of choice for the reliable collection, transmission, and delivery of event data.

Snowflake Connector for Kafka makes it simple to configure a Kafka Connect cluster to deliver JSON and Avro events into a Snowflake table. The connector continuously loads records from one or more Apache Kafka topics into an internal Snowflake stage and then into a staging table using Snowpipe (see Figure 6). Once events are in Snowflake, you can use data pipeline features to further process your data, integrate it with other business data, and refine it for use in your analyses.

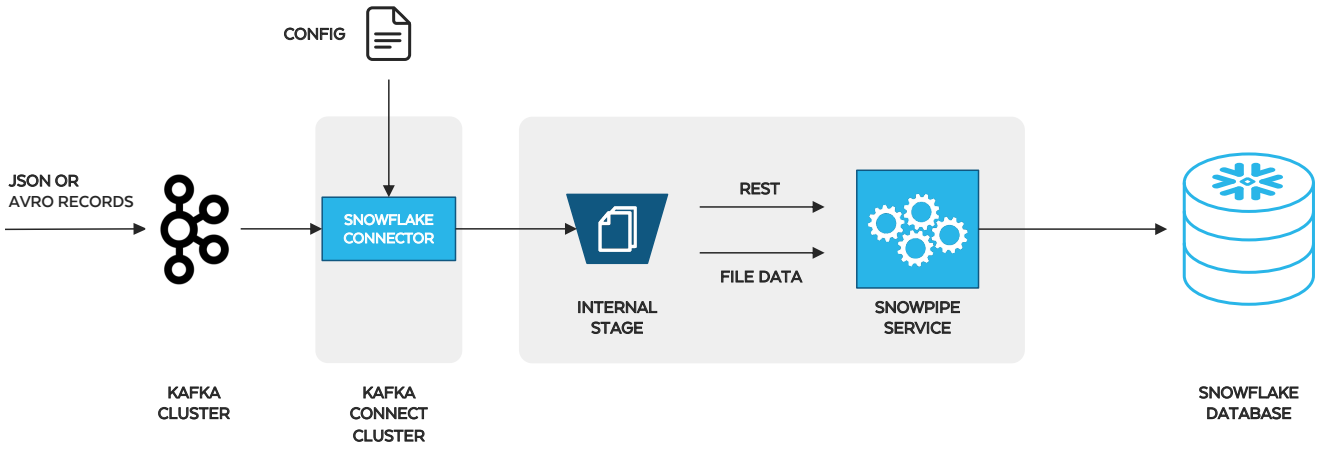

Figure 6: Snowflake Connector for Kafka automatically delivers JSON and Avro events into a Snowflake table.

Live data loading from Snowflake Data Marketplace

Snowflake Data Marketplace uses Snowflake Secure Data Sharing to connect data providers with consumers. Data science, analytics, and engineering teams can discover and access a variety of third-party data and have those data sets available directly in their Snowflake account to query without transformation and join with their own data.

Anyone who desires data-driven decision-making can use Snowflake Data Marketplace to access live and ready-to-query data. This data can come from your ecosystem of business partners and customers, or from potentially thousands of data and data service providers. No FTP or APIs are needed.

As a data consumer, you can:

- Securely access live and governed shared data sets, and receive automatic updates in near real time
- Avoid the risk and hassle of having to copy and move stale data
- Combine new data sets with your existing data in Snowflake to derive new business insights
- Have data sets available instantly and updated continually for users
- Receive frictionless access to raw data products from vendors
- Eliminate the costs of building and maintaining various APIs and data pipelines to load and update data
- Use the BI tools of your choice
- Discover and test third-party data sources

TRANSFORM DATA WITHIN SNOWFLAKE TO HARNESS THE POWER OF YOUR DATA

Snowflake allows for both ETL and ELT data loading. But bringing more of your data transformations into Snowflake has many advantages:

- Reliable and simplified data processing via Snowflake's powerful processing engine
- Instant scalability and no resource contention for your data workloads
- Native support for semi-structured data
- Automatic change data capture capabilities
- Stored procedures
- External functions
- Java UDFs to execute Java code within Snowflake
- Snowpark API for data programmability
- Support for unstructured data

Powerful processing engine lets you transform, prepare, and enrich data where it lives

Snowflake's Data Cloud offers a powerful processing engine—decoupled but deeply integrated with your data—to quickly and reliably process a near-unlimited volume of data with your language of choice. As shown in Figure 7, this unique processing engine lets you:

- **Eliminate redundant data processing.** Process data once and make it available for all of your analytic use cases; make prepared data sets easily accessible, internally and outside your organization; improve productivity with instant access to live, curated, ready-to-query data, for you to enrich as needed.
- **Build how you want with minimal complexity.** Develop flexible data pipelines with support for the most popular programming languages, such as Scala and Java(in preview) ; write code in your IDE of choice and execute data processing in Snowflake with pushdown capabilities; minimize the complexity of having to manage additional environments to run non-SQL pipelines.
- **Execute with speed and reliability.** Enable efficient data processing, with automatic micro-partitioning and data clustering; instantly support any number of users or jobs with auto-scaling policies on multi-clustered workloads; automatically scale, suspend, and resume workloads to meet your processing demands in the most cost-effective way.

Figure 7: Snowflake's processing engine offers reliable data processing made simple.

Multi-cluster compute architecture provides elasticity and scale

Data pipelines in Snowflake can be batch or continuous, and processing can happen directly within Snowflake itself. Thanks to Snowflake's multi-cluster compute approach, these pipelines can handle complex transformations without impacting the performance of other workloads.

How does this work? In Snowflake's multi-cluster, shared data architecture, compute and storage resources are physically separate, but they are logically part of a single, integrated data warehouse system. This unique multi-cluster architecture enables you to support as many separate workloads as you need. Each workload has its own compute engine and can scale up or down as required, with no need to pre-allocate resources or interrupt other processes.

With this unparalleled architecture, Snowflake can easily support multiple disparate workloads. Because of the separation of compute and storage, you can spin up separate virtual warehouses of different sizes to run your ELT processes or to support BI report users, data scientists, and data miners—all with zero resource contention.

Native support for semi-structured data simplifies transformation architecture

Semi-structured formats such as JSON, XML, and Avro are not strongly typed. You typically see a hierarchy of nodes. Each node has a name and a value—either a single constant value, an array of name/value pairs, or a nested object. Traditional databases haven't been able to deal with that varying structure, so developers resort to ETL tooling, and design complex transformations that flatten the data into traditional row/column structures.

Snowflake has a special data type just for semi-structured data, called a VARIANT. Here's an example of how the ingestion process for semi-structured data works: The virtual warehouse writes the entire JSON or XML document, in its original state, into the VARIANT column. If you query that column in SQL, you get the same JSON or XML document back.

At the same time, the global services layer collects specific metadata about the contents of the document—node names, hierarchies, array structures—and writes that data into the metadata store. Snowflake also internally organizes the data into columns so you can reference it directly in SQL. This columnar data is written into the micro-partition files, but you don't see actual column names in the table definition (note that there are no column names for any of the JSON nodes). You reference nodes from the VARIANT column with a simple dot notation in SQL. The process is highly performant and eliminates the need to write explicit transformations.

CDC capabilities keep track of what's changing

You can build continuous data pipelines with Snowpipe and then use Snowflake Streams and Tasks to schedule data integration jobs and capture change data.

- A *stream* is a Snowflake object type that provides CDC capabilities to track the delta of changes in a table, including inserts and DML changes, so action can be taken using the changed data. A table stream allows you to query a table and consume a set of changes to a table, at the row level, between two transactional points in time.
- A *task* is a Snowflake object type that defines a recurring schedule for executing SQL statements, including statements that call stored procedures. You can chain tasks together for successive execution to support more-complex, periodic processing.

In a continuous data pipeline, tasks may optionally use streams to provide a convenient way to continuously process new or changed data. A task can verify whether a stream contains changed data for a table and either consume the changed data or skip the current run if no changed data exists.

After a transformation is done, you can unload the transformed data back to your data lake using the Data Lake Export feature. Snowflake Streams and Tasks make it easy to schedule data integration jobs and capture change data. So you don't need to load all the data every time; you can simply process changed data.

Stored procedures automate frequent tasks

A *procedure* automates a frequently performed task that requires multiple SQL statements. You can create a procedure once and have it execute many times.

To create a stored procedure, use JavaScript and, in most cases, SQL. JavaScript provides the control structures (branching and looping), while SQL is executed by calling functions in a JavaScript API.

Stored procedures offer:

- Procedural logic (branching and looping), which straight SQL does not support
- Error handling
- The ability to dynamically create a SQL statement and execute it
- The ability to write code that executes with the privileges of the role that owns the procedure, rather than with the privileges of the role that runs the procedure

Snowflake external functions extend data pipelines

Many data transformations are extremely complex and difficult to build. For example, they may be built using other languages or frameworks, or they may leverage third-party code and external services. In the past, you would have to manage different services and systems, often in different data environments with complex architectures. In contrast, Snowflake allows extensible pipelines where you can define external functions and also leverage third-party services.

Users can write and call their own remote services, or they can call remote services written by third parties. These remote services can be written using any HTTP server stack, including cloud serverless compute services such as AWS Lambda.

Below are some examples of when you might want to use an external function:

- Using a geocoding service to augment addresses with coordinates and political regions
- Using a third-party service to perform sentiment analysis of messages
- Scoring customers using a custom machine learning model
- Using custom logic to extract email addresses from emails
- Fetching live stock prices from a remote service

The table below summarizes the usage and benefits of external functions.

EXTERNAL FUNCTIONS: Use custom or third-party services or transformation logic in your data pipelines to transform and augment your data	
Usage	• Limited capability to make complex data transformation in pipelines • Requirement to perform these activities outside of Snowflake
Benefits	• **Developers** can expose web services as functions inside Snowflake • **Users** can access this functionality as if it were built into Snowflake • **Administrators** retain control over where data in their system can go

Java user-defined functions (UDFs) execute JVM inside Snowflake

Java UDFs, now in preview, allow you to write a Java method and call it as though it were a SQL function. This lets you transform and augment your data using custom logic while executing the jobs next to your data, improving performance with no need to manage a separate service.

With Java UDF:

- Developers can build functionality into Snowflake using the popular Java language and libraries
- Users can access the functionality as if it were built into Snowflake
- Administrators can rest assured that data never leaves Snowflake

Snowpark API expands data programmability

Snowpark API, now in preview, is a new developer experience that enables you to write Snowflake code in your preferred language, and execute it directly within Snowflake. As shown in Figure 8, with Snowpark you can:

- Code in a familiar language that suits your project and preference. Snowflake now supports Scala, and will support Java and other languages in the future.
- Easily complete and debug data pipelines. You can also create new functions or register existing ones by bringing in custom and third-party libraries.
- Execute within Snowflake without setting up additional systems or spinning up clusters to execute your workloads.

Snowpark API is designed and optimized to leverage the best of the Snowflake processing engine—performance, reliability, and scalability with near-zero maintenance. It gives you the ability to write code directly with Snowflake in a way that's deeply integrated into the languages you use most, which opens up the data programmability of Snowflake to developers and allows all users to benefit from Snowflake's powerful processing engine. Users of supported languages can write code that is deeply integrated into their development experience. Rather than writing opaque SQL, they can write code in a familiar language and reap the benefits of their favorite IDE to give them intellisense, code completion, and type checking. These languages can make many scenarios much easier to express and debug than doing them in SQL.

Snowpark API works by pushing their operations into Snowflake's engine; there's no other system to build out or maintain. That pushdown capability also makes full use of Snowflake's upcoming Java support to enable pushdown of custom functionality into Snowflake—including pushdown of third-party or custom libraries. Pushdown enables data transformation without ever having to move it out of Snowflake, and allows developers to seamlessly extend Snowflake with any necessary functionality Snowflake may not natively support.

Support for unstructured data expands platform capabilities

The amount of unstructured data—data from audio, video, PDFs, imaging data, and more—is rapidly increasing, and Snowflake now supports unstructured data in preview. When this feature is generally available, data engineers will be able to

Developers • Data engineers • Data scientists

A new developer experience that allows you to write Snowflake code in your preferred way and execute it directly within Snowflake

Efficient and powerful pipelines
Easily complete and debug data pipelines with familiar constructs and bring in third-party libraries.

Choice of language
Write in your language with your preferred tool.

One system to manage
Eliminate the need to have other processing systems and run directly on Snowflake

Figure 8: Snowpark's data programmability enables data engineers to greatly expand what they can do with their data pipelines.

orchestrate pipeline executions of unstructured data. Unstructured data management in Snowflake means you can avoid accessing and managing multiple systems, and you can deploy fine-grained governance over files, unstructured data, and its metadata. It also opens the possibility of new revenue opportunities by giving you expanded business use cases.

DELIVER DATA TO YOUR USERS

Once data is transformed, data engineers need to make sure it is delivered appropriately to users within and outside of the organization.

Share and collaborate in the Data Cloud

Companies need to share data, whether internally with other business units or subsidiaries, or externally with vendors, partners, suppliers, and customers. Shared data results in increased innovation, better data-based decision-making, and new insights into predictive trends. But issues inherent in data collaboration such as risk, security, and privacy, can hamper effective data sharing.

Traditional methods of sharing data (such as APIs, ETL, and FTP) involve moving and copying data. This is costly, not secure, and results in delayed, out-of-sync data.

Snowflake's unique multi-cluster, shared data architecture enables companies to take advantage of Snowflake Secure Data Sharing, which allows the sharing of data generally without copying or moving it. Secure Data Sharing is unique in that it supports global sharing across clouds/regions; it can share data, logic, and services; and it offers a secure way to share data, including the ability to revoke access.

As shown in Figure 9 on the next page, you can use Secure Data Sharing to:

- Access data and services from more than 150 providers or to market and deliver your products to customers through Snowflake Data Marketplace
- Share data across your business ecosystem through Snowflake's Direct Share and Data Exchange features and through Snowflake reader accounts

Secure Data Sharing is the technology foundation for Snowflake Data Marketplace. Snowflake Data Marketplace gives data scientists, business intelligence and analytics professionals, and everyone

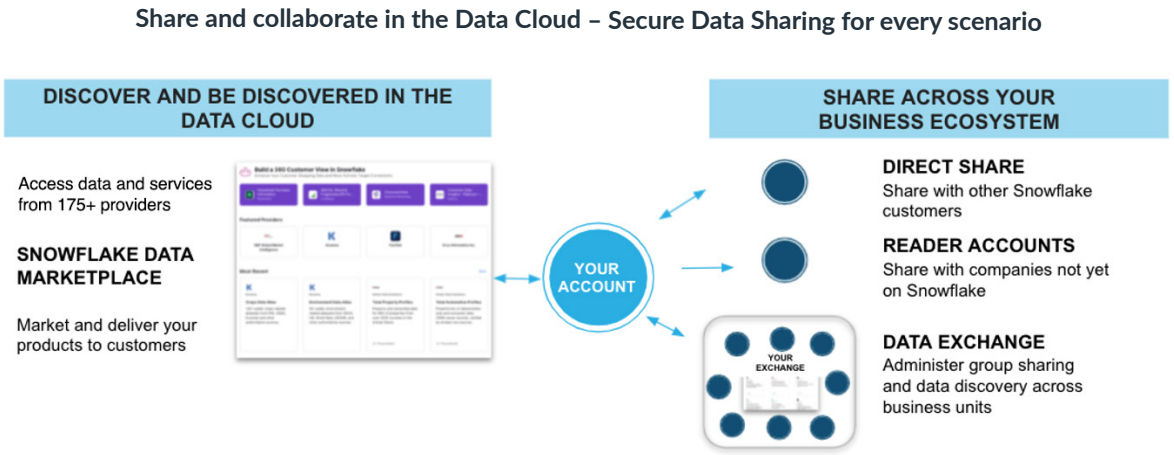

Figure 9: Secure Data Sharing ensures safe and instant collaboration

who desires data-driven decision-making, access to live and ready-to-query data from your ecosystem of business partners and customers—as well as from potentially thousands of data providers and data service providers.

With Snowflake Data Marketplace, companies can establish a private data hub and open access to employees, business units, partners, customers, and others. Snowflake's platform makes it generally unnecessary to move, copy, or transfer any data. Companies can also leverage public data sets from Snowflake Data Marketplace and combine them with their own data to analyze and potentially share within their company data hubs.

Snowflake Data Marketplace offers these benefits:

- Source data faster and more easily—Avoid the risk and hassle of having to copy and move stale data. Instead, securely access live and governed shared data sets, and receive automatic updates in near real time.
- Reduce analytics costs—Virtually eliminate the costs and effort associated with the traditional ETL processes of data ingestion and transformation thanks to direct, secure, and governed access from your Snowflake account to live and ready-to-query shared data.
- Monetize your own data—Create new revenue streams by joining Snowflake Data Marketplace as a data provider to market your own governed data assets to potentially thousands of Snowflake data consumers.

Learn more about Snowflake Data Marketplace here. See this ebook for best practices for leveraging third-party data in your analytics.

Leverage Snowflake's robust partner ecosystem

Snowflake offers a rich ecosystem of partners with highly integrated capabilities to meet your different data needs. These certified partnerships and integrations help you leverage Snowflake's flexibility, performance, and ease of use to deliver more-meaningful data insights. The Snowflake Partner Network includes partners who deliver technology, services, cloud storage, and third-party data.

The easiest way to get started is to connect from Snowflake Partner Connect. With Snowflake Partner Connect, you can create trial accounts with selected Snowflake business partners and integrate these accounts with Snowflake. This feature provides a convenient option for trying additional tools and services, and then adopting the ones that best meet your business needs.

For example, you can start loading data more rapidly by using Snowflake Partner Connect. It simplifies the onboarding process through pre-built integrations with Snowflake's technology partners. You can automatically provision and configure partner applications and start loading data into Snowflake in minutes for immediate analysis.

Export data back to your data lake

Depending on how you design your architecture, you can use external tables to access your cloud data lake directly. Or you can use Snowpipe, Snowflake's serverless ingestion service, to ingest data into Snowflake automatically. Snowflake Streams and Tasks features make it easy to schedule data integration jobs and capture change data. You don't need to load all the data every time; you can simply process changed data. Once transformation is complete, you can unload the data back to your data lake, automatically partitioned by a column of your choice, with Snowflake's Data Lake Export feature.

The export capability enables you to process large amounts of raw data in Snowflake and then easily export it back to your data lake (supporting Parquet, CSV, and JSON).

Connect to common drivers

Snowflake provides common drivers such as Python, Spark, JDBC, and ODBC drivers so you can easily bring your data to downstream applications and meet different analytical needs.

- The Snowflake Connector for Python provides an interface for developing Python applications that can connect to Snowflake and perform all standard operations. It provides a programming alternative to developing applications in Java or C/C++ using the Snowflake JDBC or ODBC drivers.
- The Snowflake Connector for Spark brings Snowflake into the Apache Spark ecosystem, enabling Spark to read data from, and write data to, Snowflake. From Spark's perspective, Snowflake looks similar to other Spark data sources (PostgreSQL, HDFS, S3, and so on). The connector supports bidirectional data movement between a Snowflake cluster and a Spark cluster. You can use the Spark connector to perform operations such as reading and writing. For example, you can use it to populate a Spark DataFrame and write a Spark DataFrame to a Snowflake table.
- The Snowflake JDBC Driver supports core JDBC functionality. You can use the driver with most client tools and applications that support JDBC for connecting to a database server. JDBC makes it possible to establish a connection with a data source, send queries and update statements, and process the results.
- The Snowflake ODBC Driver enables you to access Snowflake directly from any applications that support ODBC connectivity. You can use the ODBC driver with clients such as SQL clients.

ABOUT SNOWFLAKE

Snowflake delivers the Data Cloud—a global network where thousands of organizations mobilize data with near-unlimited scale, concurrency, and performance. Inside the Data Cloud, organizations unite their siloed data, easily discover and securely share governed data, and execute diverse analytic workloads. Wherever data or users live, Snowflake delivers a single and seamless experience across multiple public clouds. Snowflake's platform is the engine that powers and provides access to the Data Cloud, creating a solution for data warehousing, data lakes, data engineering, data science, data application development, and data sharing. Join Snowflake customers, partners, and data providers already taking their businesses to new frontiers in the Data Cloud. **Snowflake.com**.

ENDNOTES

[1] gtnr.it/2W2C1r5
[2] mck.co/3iRNGBK