



SPARK TO SNOWFLAKE MIGRATION GUIDE

Migration Strategies and Best Practices



TABLE OF CONTENTS

- 2** Why Migrate?
- 3** Your Motivation to Migrate
- 5** Snowflake with Snowpark
- 8** Is Your Workload the Right Fit?
 - 8** - What workloads and processes can you migrate with minimal effort?
 - 9** - What new workloads would you like to add that would deploy more easily in Snowflake?
 - 10** - Which processes have issues today and would benefit from re-engineering?
 - 11** - What workloads are outdated and require a complete overhaul, or can really take advantage of the platform?
- 13** Migration Plan
- 16** The Value of Snowflake over Spark
- 17** Need Help?
- 18** About Snowflake

WHY MIGRATE?

In the last few years, many data pipelines have been developed using technologies such as Hive, Hadoop, and Apache Spark.

While the exponential growth of data presents new opportunities for organizations, current infrastructures make ingesting, transforming, and analyzing vast amounts of data at scale challenging.¹

Developers have noted a variety of challenges:

- 1. Complexity:** Infrastructure is complicated to build, maintain, and troubleshoot. Implementations rely on many disparate tools to process and prepare data. Pipeline reliability is hard to achieve, and this leads to performance bottlenecks and downtime. Large efforts and investments might be needed to correct issues.
- 2. Usability and cost:** Data teams must manage these complex deployments, and the specialized experts organizations need to hire can be hard to find and expensive to retain.
- 3. Many data sources:** Data scientists have to spend time finding and pulling data from many sources, and finding ways to share it.
- 4. Scalability issues:** Lag time for adding and/or removing capacity.
- 5. Availability risks:** Setting up a multi zone (AZ) solution requires detailed knowledge of the underlying components.
- 6. Security:** Too many places for setting up and keeping credentials.

WHAT IS SPARK?

Apache Spark™ is a multi-language engine for executing data engineering, data science, and machine learning on single-node machines or clusters.² It provides a mechanism for collecting, shaping, and processing unstructured information—even for large data sets.

Spark has been a useful tool for compute-intensive data engineering and analytical tasks. It enables users to write code that scales out over a cluster, but it does not remove the complexities related to management of it. The Spark abstractions do not hide the inherent complexities of using distributed resources like memory and compute. Significant effort is devoted to infrastructure instead of focus on data.



YOUR MOTIVATION TO MIGRATE

Near-zero maintenance: Snowflake reduces complexity with built-in performance, so there is no infrastructure to tweak, no tuning required, nor any hurdles like system upgrades, security patches, and capacity planning.

Modern architecture that meets the business users' needs: With a traditional data architecture approach (see Figure 1), customers often run separate layers of cloud storage, data warehouses, and a processing framework such as Spark. As a result, lots of data movement, unnecessary data pipelines, and data silos are created, which can limit full accessibility. The data warehouse, cloud storage, and Spark system all have different security domains and controls. Each end point is also a potential governance and security weak spot. All this complexity creates multiple potential points of failure and workload bottlenecks that require highly skilled staff to manually manage and maintain.

Faster access to data: Snowflake's elastic, near-unlimited scale and speed enables data professionals to have fast access to all current and historical data at any time.

Data sharing: Snowflake enables direct, governed, and secure data sharing in near real time, so enterprises can forge one-to-one, one-to-many, and many-to-many data sharing relationships.

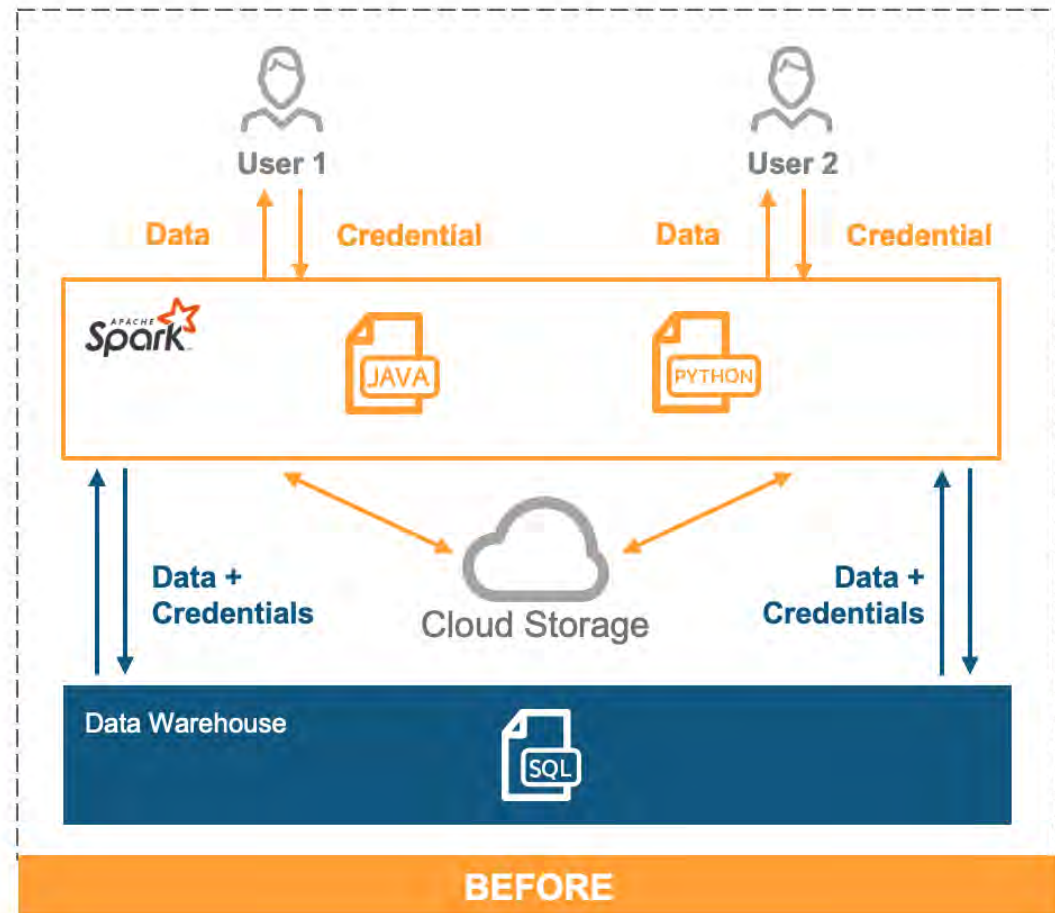


Figure 1: Before: Complex data pipelines with traditional approach

Complete SQL database with the programming language of your choice: Snowflake supports the tools millions of business users already know how to use today. It also comes with Snowpark, a developer framework that allows data engineers and data scientists to code in Scala, Java, or Python. Learn more about Snowpark in the “Snowflake with Snowpark” section of this guide.

Cost: With Snowflake, you avoid the complexity and usability challenges of traditional data architectures, which can increase the total cost of ownership (TCO) due to higher operating expenses. Performance issues associated with traditional technology can also increase costs due to longer query completion times and the need for larger cluster sizes. And limited elasticity further

compounds costs in terms of clusters that continue running after inactivity (due to the lack of ease to turn clusters on or off) and the recommended use of paid-for spare capacity. Last but not least, the creation of data silos can further reduce business value.

Governance capabilities: Snowflake provides a single source of truth where you can know your data (object tagging), protect your data (anonymization, access policies, masking), and track data access all from just one platform.

SNOWFLAKE WITH SNOWPARK

Snowpark brings a new developer experience that allows you to write Snowflake code in your preferred way and execute it directly within Snowflake.

Snowpark makes data programmability natural and easy. Organizations can use their current SQL skills and also incorporate Scala, Java, or Python, and familiar DataFrame constructs to build powerful and efficient pipelines, machine learning (ML) workflows, and data applications. This means that Spark users can code in the same way, but execute faster and more securely with Snowflake (see Figure 2).

Organizations can create user-defined functions (UDFs) with their preferred libraries and custom logic for things like data augmentation, transformation, ML scoring, and business logic—then automatically push to the Snowflake engine to do the heavy lifting. Additionally, Snowpark enables organizations to take advantage of the Snowflake Data Cloud, which was designed natively in the cloud as an easy-to-use, connected, cost efficient, and governed platform.

Snowflake fully leverages the benefits of cloud elasticity across data storage, processing, and

sharing, and fundamentally changes how different data workloads can be approached with strong collaboration across teams.

Snowflake's architecture provides the flexibility for customers to enable data lake, data warehouse, or data mesh architectures, depending on their needs. With Snowflake's consumption-based pricing, customers pay only for what they use. And, most importantly, Snowflake allows customers to tap into a network of data and services that are easily

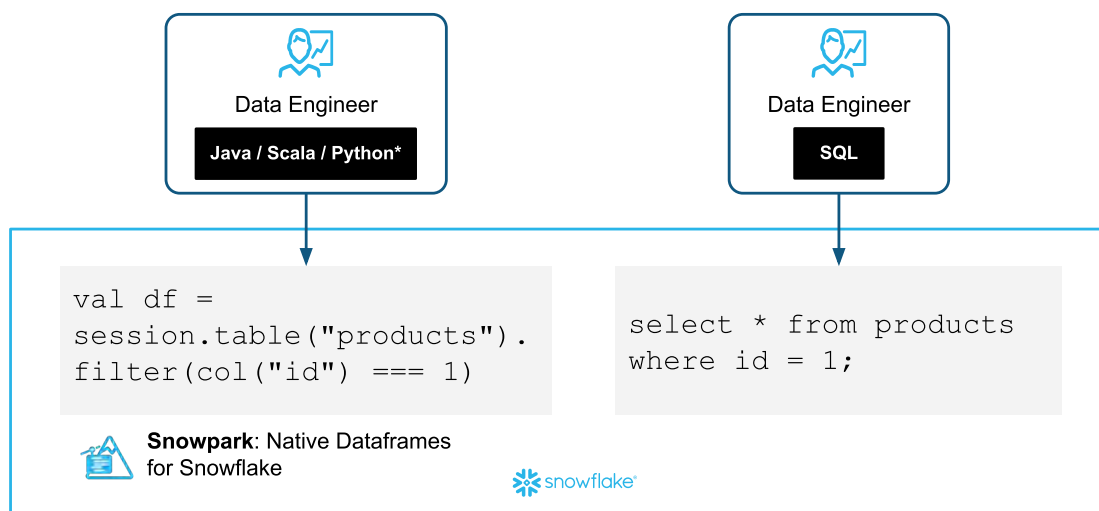


Figure 2: Snowpark enables users to code in Java, Scala, or Python, and execute it directly within Snow lake.

accessible, using data sharing capabilities without the need to build unnecessary data pipelines or traditional ways to access this data.

Last but not least, Snowflake's intelligent infrastructure interfaces with data without your teams having to worry about infrastructure installation or maintenance. When you migrate to Snowflake, you get out of the box:

- **Security:** Security configuration based on industry best practices, plus continuous vulnerability testing. For cloud computing, your data assets need to be protected by the appropriate security settings and ongoing education and efforts meant to deflect a constant stream of new security threats. Snowflake provides a secure platform from the ground up with a robust RBAC system ready to be leveraged.
- **Ease of use:** Simplified architecture with one copy of data and a single source of truth.
- **Data sharing:** Easily share data securely within your organization or externally with your customers via direct shares or Snowflake Data Marketplace.
- **Zero-Copy Cloning:** Create multiple "copies" of tables, schemas, or databases without actually copying the data. This saves time to copy and reduces data storage costs.
- **Near-instant Scalability:** Scale compute and storage independent of one another, and isolate compute power for jobs that need their own dedicated warehouse.
- **Availability:** Simple Multi-AZ support, plus replication and failover across all regions and clouds.

SNOWFLAKE DATA MARKETPLACE

Snowflake Data Marketplace provides data teams with direct access to hundreds of ready-to-query data sets and data services to inform their models. This includes a wide range of open and commercial data across multiple sectors and categories, including public health, weather, location, demographics, and others.

Consider the needs of an organization using AI/ML to predict aluminum prices on a daily basis. To do that, they would need to subscribe to Bloomberg to get that data daily. In the process, they learn that corn prices are impacting aluminum prices. That's external data that would further improve the data team's predictions. In this case, their data scientists need broader, on-demand access to data. Snowflake Data Marketplace provides this with access to over 240 third-party data providers and data service providers (as of January 31, 2022).

Snowflake Data Marketplace reduces the costs and effort associated with traditional ETL processes of third-party data ingestion and transformation thanks to direct, secure, and governed access from your Snowflake account. It provides faster access to fresher data through automatic real-time updates, removing the need to build unnecessary data pipelines.

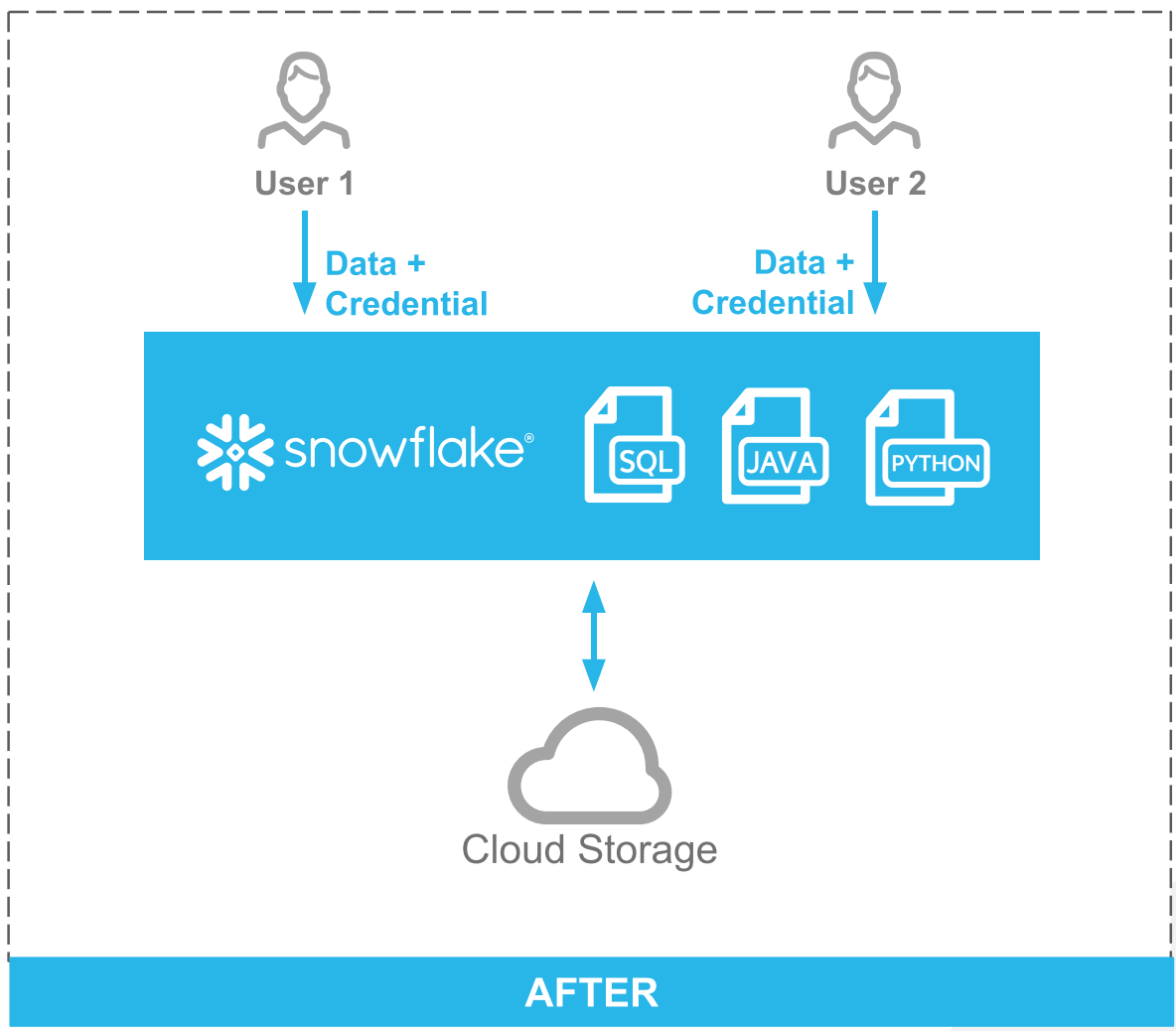


Figure 3: Simplified architecture with Snowflake

MIGRATE WITH CONFIDENCE

Taking advantage of these elements of the Data Cloud is now easier than ever with the support provided by Snowflake for Snowpark. The following sections will guide you through the migration process and focus on Spark to Snowpark migrations, but you don't have to undertake the entire process on your own.

Snowflake's partner network is available to support you, and tools such as Mobilize.Net SnowConvert can help you accelerate your migration project.

IS YOUR WORKLOAD THE RIGHT FIT?

This should be the first question to ask when considering migrating workloads from Spark to Snowflake. From a top-level view, it is important to understand the engineering or data science that composes your workload. Classifying your Spark jobs can help to properly determine the pieces of your workload that are a best fit to an upgrade with Snowflake.

To help you get started with the best workloads to target, ask these four questions about your workloads and data:

1. WHAT WORKLOADS AND PROCESSES CAN YOU MIGRATE WITH MINIMAL EFFORT?

Data pipelines

With Spark, you have to set up different data pipelines and spend time finding and pulling data from different sources. Migrating to Snowpark removes some of these challenges.

Snowflake customers have the flexibility to move data to Snowflake to fully take advantage of the Data Cloud and streamlined architecture, or keep their object store but also access data directly from Snowflake via External Tables. Snowflake can work with files from your cloud storage right where they are.

In most cases, moving that data into Snowflake tables will provide greater flexibility and performance. For example, getting slices of data based on date or company is more performant with a table than with a Parquet file, with no maintenance overhead. Migration tools can speed the conversion of the DDL definitions.

If the data is already in Snowflake, then you can just focus on what transformations are needed by your data. Accessing data between Snowflake warehouses is straightforward and another good example of the power of Snowflake Secure Data Sharing.

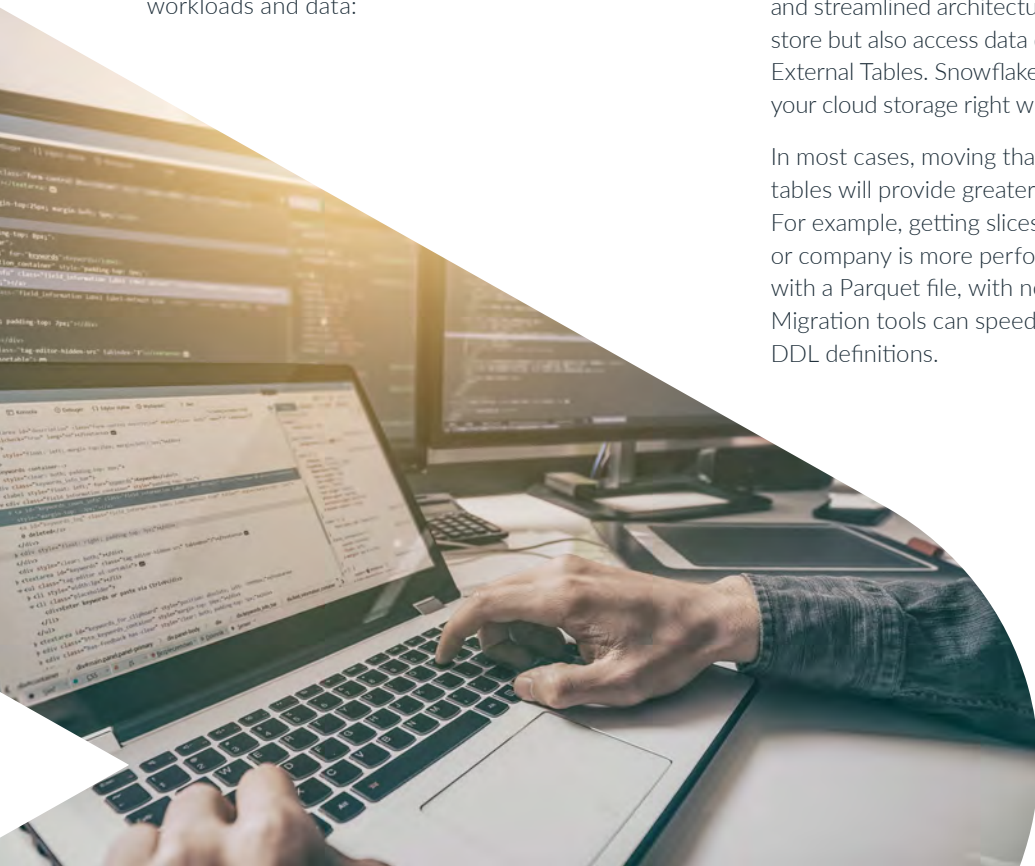
For on-premises data, there are many connectors available in order to migrate it, or it can be uploaded to cloud storage.

Extract transform load (ETL) or extract load transform (ELT) tasks

The Snowpark DataFrame API simplifies this process. With just a few lines it is possible to take multiple data tables and transform that data by deduplicating it, combining it, and loading it into other tables or views.

You can also filter data sets, aggregate them by adding columns from other data sets, join multiple data, save those results, and then use a subset of them on another calculation step—all with whatever language you love. You can do it in SQL, you can use your Python and Pandas knowledge, your core Java knowledge, or you can leverage functional Scala data programming.

You can also take these pipelines written in your preferred language and operationalize them in Snowflake with Tasks and Stored Procedures.



2. WHAT NEW WORKLOADS WOULD YOU LIKE TO ADD THAT WOULD DEPLOY MORE EASILY IN SNOWFLAKE?

With Snowflake, you may be able to address challenges that include:

Standardizing business logic

Your business has code that implements valuable reusable logic either in Java/Python/Scala for tasks like data format handling, or credit scoring algorithm (see Figure 4).

Data shaping

A common task in your pipelines will be to process the data and shape it into a common format for consumption. Let's review some use cases:

- Translation from a binary format or data extraction with a trained model:** A UDF can read a value in a column from a particular binary format (possibly one that's encrypted) and apply a custom algorithm. You can then upload files to your stage and read them from your UDF. The UDF can also leverage a previously trained model and apply that to the input file to aggregate data, for use cases such as sentiment analysis from a document. Once the UDF has been deployed, you can do this operation even from a simple SQL query.
- JSON normalization:** Unstructured information from different sources may require some work to shape it, so that it can then be unified for further processing. XML or JSON files can be processed to match supported formats that will be consumed further down your pipeline.

- Custom XML processing:** This step is used for performing analysis on large or complex XML data, or for normalization of documents with different schemas. And you can use any of the Python packages for XML processing, for example.

Data wrangling

Data wrangling—also called data cleaning, data remediation, or data munging—refers to a variety of processes designed to transform raw data into more readily usable formats. These may include data cleansing tasks as well as data validation.

- Cleanse and correct records:** Apply checks to make sure the values are within expected ranges. These critical operations are used to detect and correct any corrupt or inaccurate records from a record set, table, or database.
 - Remove outliers**—excluding the outlying data point with clear cause documented.

- Correct spelling errors**—for example, by using a Python spellchecker.
- Bot detection**—When an application gathers metrics to track user behavior, usually bots need to be detected so they do not pollute that data meant for training over human behavior.
- Data validation:** The typical goal of validation is making sure that data is good before moving on to model training. These steps include:
 - Data lookup**—making sure that the data on a column is within a known set of allowed values; using a value as a key to retrieve additional information.
 - Common validations**—checking that the value is not null, or mapping nulls to a proper value.



Figure 4: Deploy common Java / Python libraries for all users.

Data augmentation

With Snowpark, you can bring custom functions, including trained ML models, as Java or Python UDFs to augment incoming data inside Snowflake (see Figure 5).

Examples of more data augmentation use cases include:

- **Language detection:** Determine the language for a text snippet.
- **IP geolocation:** Mapping an IP or MAC address to the real-world geographic location of an internet-connected device. For example, to get the country for your visitors.
- **Network analysis:** This can be used to understand the relationships on social networks or the structure of a natural process, or for complex matters like the analysis of biological systems of organisms.
- **Data Binning:** This process sorts a number of more or less continuous values into a smaller number of “bin” groupings. For example, if you have data about a group of people, you might want to arrange their ages into a smaller number of age intervals.

3. WHICH PROCESSES HAVE ISSUES TODAY AND WOULD BENEFIT FROM RE-ENGINEERING?

Data security

- **PII data from customer information.** A UDF can be applied to the content of a column in a data set to detect or remove all personally identifiable information (PII). For example, once a sensitive column such as credit card numbers or SSNs is identified, a Java UDF can be applied to encrypt regulated data at rest inside Snowflake (see Figure 6)



Figure 5: Augmenting data with Java or Python UDFs inside Snowflake.

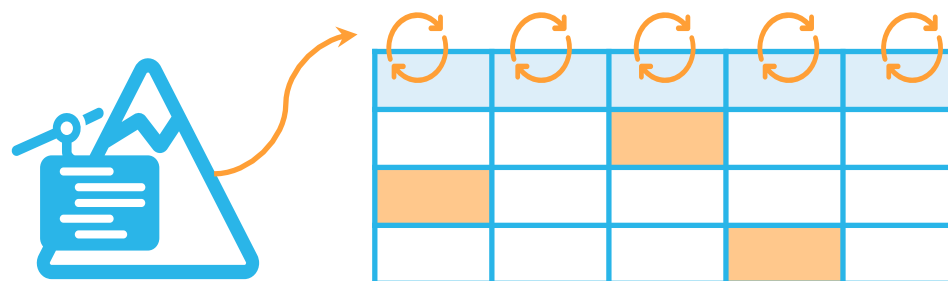


Figure 6: Use Snowpark UDFs to iterate over all DataFrame columns, to scan for anomalies or PII.

- **Data masking, filtering, and row-level encryption can be leveraged.** This is relevant to data flows subject to GDPR, HIPAA, CCPA, and many others (see Figure 7).

Publishing data

Publishing data in Snowflake is easy. After you transform your data, you can save it to a file in cloud storage, or even better, save again into Snowflake and then use Data Share to provide data to other consumers with RBAC and proper security and governance.

With integration with your cloud storage, Secure Data Sharing, Zero-Copy Cloning, and a set of features that include data clean room functionality, Snowflake enables you to make data available within a single governed platform.

4. WHAT WORKLOADS ARE OUTDATED AND REQUIRE A COMPLETE OVERHAUL, OR CAN REALLY TAKE ADVANTAGE OF THE PLATFORM?

Unstructured data management

Snowflake enables you to store, access, process, and share unstructured data, as well as metadata, with fine-grained governance of files, and retrieve any data set with simple SQL commands (see Figure 8).

Data science

Snowpark also offers advantages across a number of data science processes:

- **Feature engineering:** This task is very domain-specific, but it usually involves a combination of data transformations, data binning, and some custom logic to add or remove the information to the data

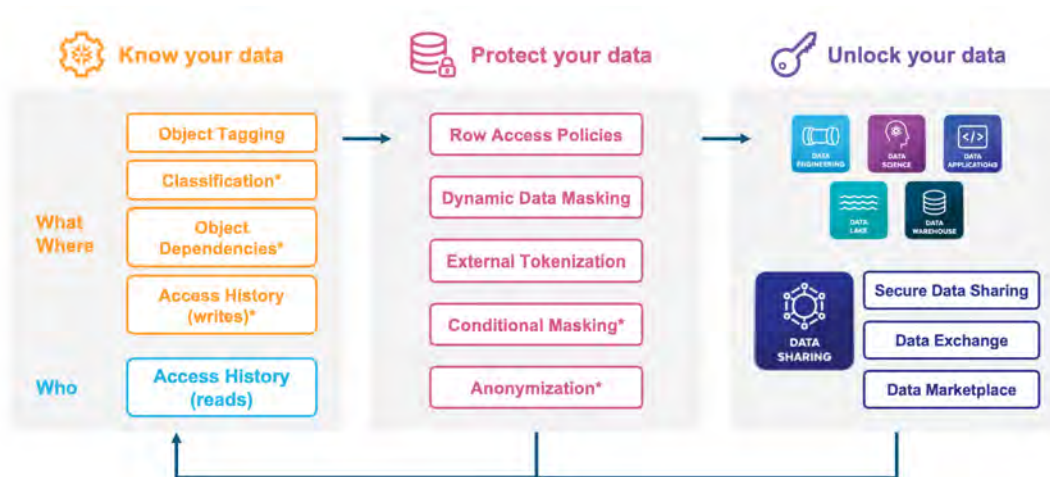


Figure 7: Use data masking, filtering, and row-level encryption for data flows to enable compliance with regulations such as GDPR, HIPAA, CCPA, and others. (*Private Preview features)

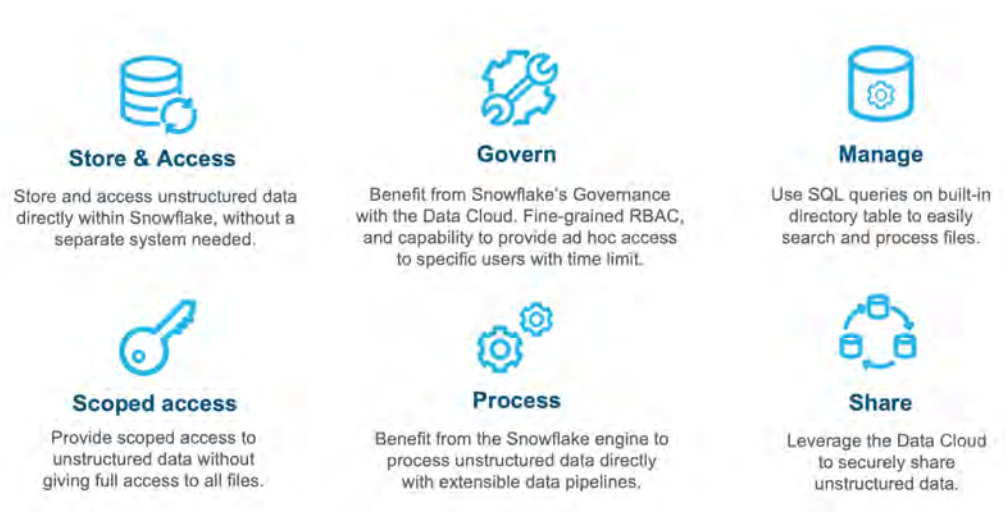


Figure 8: Store, access, process, and share unstructured data in Snowflake.

set. Sometimes feature engineering is a step in the process and sometimes it is the end goal itself. The refined data set with the added features is usually published so it can be used by other processes. These other processes may be other business intelligence (BI) tools, or can be the input for model training algorithms.

- **Data augmentation:** Snowflake's Data Marketplace provides a convenient source of additional data sets that can be used to augment your own data. For example, by enriching the data with better geographic data, and time-series personal and commercial context, a quantifiable improvement in the accuracy of the predictive decision-making can be achieved faster and at scale. With faster, better insights, organizations can increase revenue, streamline operations, and reduce risks.
- **Experimental snapshots:** Zero-Copy Cloning for snapshotting experimental data allows easy setup for model training experiments.
- **Model training:** Snowpark supports model training use cases that do not require distributed training or GPU infrastructure, using pre-installed open source libraries available as part of our product integration with Anaconda. For more extensive training support, Snowflake users can leverage prebuilt integrations to leading ML platforms, or the Snowflake Connector for Python fast data transfer between Snowflake and your ML training environment.
- **Operationalize trained models (MLOps):** Snowpark streamlines the path from experiment to production with support for objects, code, and other artifacts within a UDF. Machine learning models, whether trained in Snowflake or in an external platform, can be one of these artifacts. Bringing the model to Snowflake streamlines the integration between data pipelines and models to accelerate the path to model inference in production.



MIGRATION PLAN

A successful migration project requires a well-designed plan that identifies all components to consider for your migration. This section provides a checklist of information to gather before you start the actual migration.

IDENTIFY AND ASSESS

The first step is to collect key application information, including conducting a file inventory, building a list of technologies already in use, and documenting API usage. To get started, here is a list of recommended information to collect:

- **Complete file inventory**
- **List of technologies—is the workload written in Python, Java, Scala?**
- **Presence of:**
 - Spark UDFs
 - Spark custom aggregate functions
 - Spark resilient distributed data sets (RDDs) and partitioning APIs
 - Additional libraries (open source, third party)
 - Initial list of entry points
- **Usage of Streaming**
- **Usage of SparkML or other model training-related technologies**
- **Collecting import usage**
- **Detailing API usage**

PHASE	OBJECTIVE	OUTCOME
Identify and assess	Determine application size and dependencies	Application inventory: <ul style="list-style-type: none"> • File list/sizing • List of jobs/tasks • Data source dependencies • Flat files, CSV, Parquet, Avro, delta, JDBC to other databases • Read/write status • Benchmarks • Latency (batch vs. real time) • Data/compute time • CPU/memory usage
Project overview	Identify potential risks and establish project timeline	Scope/project timeline
Application migration	Create a new codebase from the original application	Remove, replace, or re-map
Migration testing	Validate target application functional equivalence	Benchmarks validating target data and performance within tolerances

- **Directing Hive or Spark SQL commands and metrics on SparkSQL**
- **Highlighting files with long DataFrame chains**
- **Listing identified data source patterns**
- **Tracking Maven dependencies and compatibility**

All this information helps you to assess the size of the task at hand. However, this process itself can be complex and requires some manual effort. To help customers jumpstart their migration project, partner solutions such as the offering from Mobilize.Net may be helpful. Snowflake has worked with Mobilize.Net to build SnowConvert for Spark, a migration tool that will let you determine if most of your code leverages Snowpark-compatible DataFrame API, and identifies which files have a lot of unsupported features. The tool will determine your readiness score so you can determine how easy it might be to move to Snowpark.

The generated assessment also provides links to documents with best practices and reference architecture information that will guide you to re-engineer any elements of the code that require adjustments before moving to Snowpark.

Data sources inventory: This step assesses all structured, semi-structured, and unstructured formats, including CSV, Parquet, flat files, Avro, and Delta. It also tracks data sources for on-premises (e.g., SQL server, Oracle, Teradata, Greenplum), cloud database (e.g., Redshift, BigQuery), and cloud storage (e.g., Google, Amazon, Azure). For each data source, it is important to determine specific tables/files, data metrics, usage level, sync needs, and whether the data source is read or write.

It is important to determine the current data lake structure. Typical data lakes can have some challenges, such as a rigid architecture or slow data movement, which can have an effect on your pipeline stability, so you might consider moving this data directly into Snowflake tables.

Third-party vendor software or API dependencies:

This identifies if there are any third-party tools or APIs currently being used to enrich or transform the databases in question.

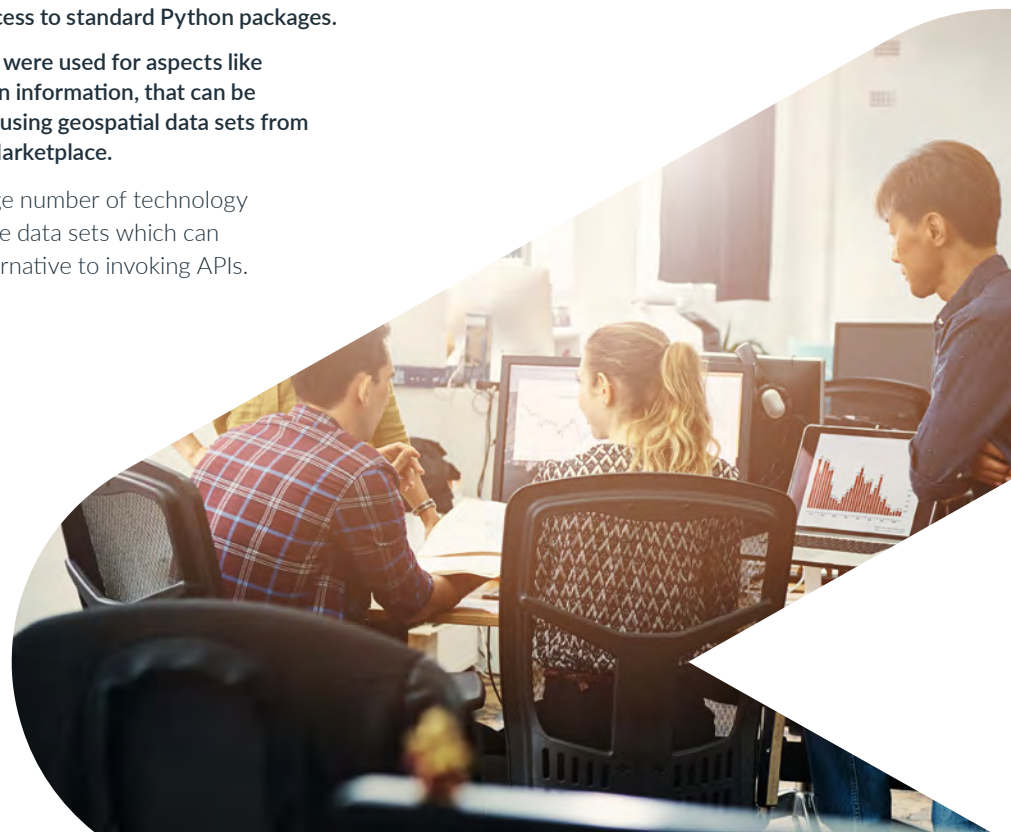
- **For Java and Scala workloads, most of the standard libraries can be uploaded to Snowflake and used without concerns.**
- **For Python, Snowflake now leverages Anaconda to provide secure access to standard Python packages.**
- **If third-party APIs were used for aspects like getting geolocation information, that can be easily replaced by using geospatial data sets from Snowflake Data Marketplace.**

Snowflake has a large number of technology partners that provide data sets which can provide a better alternative to invoking APIs.

PROJECT OVERVIEW

It is important to determine a project timeline and start identifying potential risks. An ideal situation would be to consume all data from within the Snowflake Data Cloud. This will provide the best performance while at the same time improving both security and governance.

Some replacement alternatives can be considered at this point, along with the possible cost benefits. For example, replacing Apache Parquet files will introduce performance benefits and make some filtering and grouping tasks easier.



Another key aspect for a successful migration is test data. And it is also important to determine the level of acceptable error tolerance (e.g., acceptable error margins introduced because of rounding). You should also consider that some data sets might have slight differences in row counts (e.g., rounding differences can cause some rows not to be filtered, returning more rows).

APPLICATION MIGRATION

This phase is where the original application source code is taken and transitioned to work with Snowpark. If you are not migrating your whole application, the recommendation is to create a slice from the original code with only the subset that you want to upgrade to Snowpark.

When it comes to actual migration of an application from Spark to Snowpark, there are three possible approaches:

- **Removing:** Elements that provide functionality already provided in Snowflake, such as end-to-end encryption and data masking.
- **Replacing:** Some applications or specific subset capabilities may provide limited value post-migration. It may be more cost-effective to replace these functionalities with those that are already provided by the Snowflake Data Cloud.
- **Remapping:** This approach can be applied to small and medium workloads. It refers to adapting Spark mapping patterns to Snowpark patterns. Snowpark supports both Python and Scala languages and it also provides a very compatible DataFrame API.

Tools such as Mobilize.Net SnowConvert for Spark-Scala or Mobilize.Net SnowConvert for Spark-Python will accelerate the migration planning process. These tools can take original source code and create a new folder with the new remapped project. They will adjust the namespaces or Python imports. They can check all usages of Spark APIs and map them to equivalent Snowpark APIs. If needed, the tools will also introduce helpers to reduce the number of changes for any areas that have a different behavior between Spark and Snowpark.

These tools can also help to track areas that need re-architecting or replacement. Many partition operations can be removed and others can be mapped to Snowflake window functions, which provide support for partitioning and ordering.

These tools will also introduce warnings on patterns that might require rewriting, with links to the Mobilize.Net SnowConvert for Spark-Scala or Mobilize.Net SnowConvert for Spark-Python knowledge base. This provides pattern libraries, sample code snippets, and reference source code to guide you on best practices for common use cases.

MIGRATION TESTING

This is perhaps the most important phase in migrating to Snowpark. We recommend planning migration testing in two parts:

- **Smoke tests:** These tests can be performed with a small subset of data. The idea is that the workload can be run from start to end to validate that the general logic is equivalent, and to have a basic verification of the applied changes.
- **User acceptance testing:** These tests are more complex because they require running the migrated workload with an environment close to production. This is so that it can be benchmarked to determine critical success factors, such as if target data row counts or workload total execution times are within tolerances.

THE VALUE OF SNOWFLAKE OVER SPARK

As the volume and velocity of data continues to expand, data scientists, analytics experts and BI professionals, and data engineers need tools that are purpose-built for modern workloads—designed for simultaneous performance, scalability, value, and security. To help you set goals and measure the success of your migration project, here are typical metrics to track:

METRICS	CURRENT STATE	FUTURE STATE
Assumption of data validation testing (with very minimal differences)	Capture data set based on the legacy run	Match the data set with converted Snowpark results
Performance <ul style="list-style-type: none"> • Processing time & frequency • Optimization, simplifying data pipeline • Data availability 	Capture current benchmark	All processing handled in Snowflake, providing improved performance
Cost savings <ul style="list-style-type: none"> • Run time • Support & maintenance operational licensing costs 	Capture current spending based on Spark infrastructure and people cost	Cost savings for the simplified architecture

NEED HELP MIGRATING?

Snowflake is available to accelerate your migration, structure and optimize your planning and implementation activities, and apply customer best practices to help meet your technology and business objectives.

[Snowflake Professional Services](#) deploys a powerful combination of data architecture expertise and advanced technical knowledge of the platform to deliver high-performing data strategies, proofs of concept, and migration projects.

Our technical solution partners such as [Mobilize.Net](#) also have extensive experience performing proofs of concept and platform migrations. They

offer services ranging from high-level architectural recommendations to tools to automate and accelerate the migration process.

Whether your organization is fully staffed for a platform migration or you need additional expertise, Snowflake and our solution partners have the skills and tools to help accelerate your journey to the Snowflake Data Cloud, so you can reap the benefits more quickly.





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](https://www.snowflake.com).



ABOUT MOBILIZE.NET

Mobilize.Net builds the world's highest fidelity source code translation technology. Millions of developers have used Mobilize.Net technology to successfully modernize billions of lines of code. Mobilize.Net solutions enable customers to reduce risk, cost, and time while moving applications to the platforms businesses demand today. Mobilize.Net SnowConvert migration technology is Snowflake's chosen solution for supporting Snowflake Professional Services' migration efforts.

The privately held Mobilize.Net is based in Bellevue, Washington. Find out more at www.mobilize.net.

©2022 Snowflake Inc. All rights reserved. Snowflake, the Snowflake logo, and all other Snowflake product, feature and service names mentioned herein are registered trademarks or trademarks of Snowflake Inc. in the United States and other countries. All other brand names or logos mentioned or used herein are for identification purposes only and may be the trademarks of their respective holder(s). Snowflake may not be associated with, or be sponsored or endorsed by, any such holder(s).

CITATION

¹ bit.ly/38QnHYD