



SNOWPARK: データクラウドに新たなデータ プログラマビリティをもたらす



TABLE OF CONTENTS

- 2** エグゼクティブサマリー
- 3** Snowparkを使用してより優れたパイプラインやデータモデルを構築する方法
- 5** Snowparkのデータエンジニアリングユースケース
- 8** Snowparkのデータサイエンスユースケース
- 11** Snowparkのデータガバナンスおよびセキュリティユースケース
- 12** 他のSnowflakeパートナーのユースケース
- 13** Snowparkを導入する
- 14** Snowflakeについて

エグゼクティブサマリー

Snowflakeは、データの世界を抜本的に見直し、信頼性、安全性、パフォーマンス、スケーラブルなクラウド向けデータ処理システムの設計方法を再考することにより、データクラウドの実現に向けた取り組みを開始しました。

その結果生まれたのが、Snowflake向けの新しい開発者フレームワークである「Snowpark」です(プレビュー提供中)。Snowparkでは、データエンジニア、データサイエンティスト、データ開発者が単一のプラットフォーム内で自分の好みの言語を使って自分の使い慣れた方法でコーディングを行い、パイプライン、MLワークフロー、データアプリをより迅速かつ安全に実行できます。

Snowpark APIは、ScalaやJavaなど開発者がよく使う言語に、緊密に統合されたDataFrame型プログラミングをもたらします(プレビュー提供中)。Snowpark UDFを使用すると、データユースケースの幅を簡単に広げ、Java UDF、JavaScript UDF、外部関数、Python UDF(近日公開予定)などをSnowflake内で実行できます。Snowparkは、複雑なデータパイプラインの構築を大幅に簡素化し、開発者がデータの移動なしにSnowflakeと直接やり取りできるよう設計されています。

Snowparkで実現可能なユースケースの例を以下に挙げます。

- 機械学習を使用し、トレーニング済みのモデルをJavaでホストすることでデータを増強する
- データ内の異常をスキャンする
- PIIを識別するためのルーチンを開発する
- 共有Javaライブラリをデプロイしてビジネスロジックを標準化する

Snowparkはデータプログラマビリティの進化における大きな一歩であり、Snowflakeプラットフォームの可能性を簡単に広げられるようにします。Snowparkのリリース以来、魅力的なユースケースがすでに数多く誕生しています。これには、Snowpark Acceleratedプログラムに参加している当社のパートナーからのユースケースも含まれます。

皆様はSnowparkで何を構築できますか。このeBookではいくつかのユースケース例をご紹介しますが、皆様はもっと多くのユースケースを構築できるでしょう。では始めましょう。

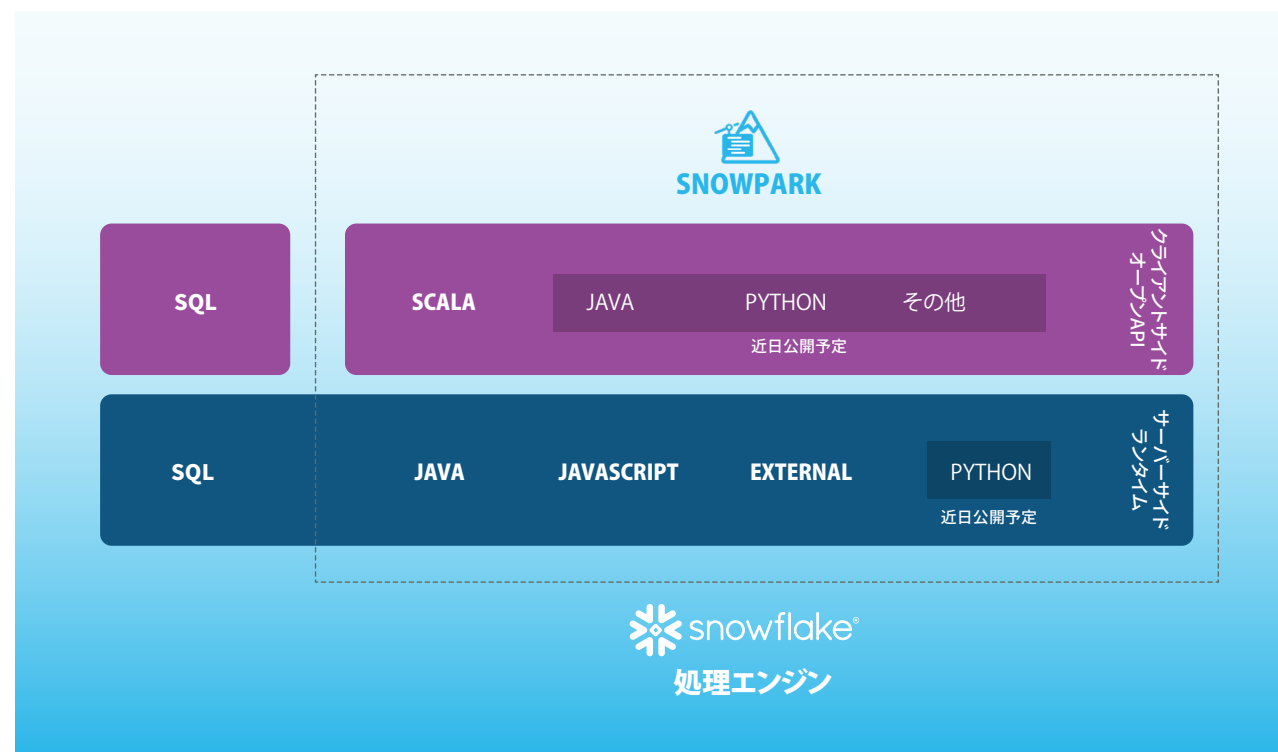


図1: Snowparkでは、開発者やエンジニアがデータの移動なしにSnowflakeと直接やり取りできます

SNOWFLAKEを使用してより優れたパイプラインやデータモデルを構築する方法

Snowparkの仕組みを理解するため、まずPII(個人を特定できる情報)を用いた一般的な例を見てみましょう。

効率的なクエリ、シームレスな変換

Snowpark APIでは、デベロッパーはSQL文字列を作成して渡すことなく、コードでDataFrameを使用してクエリを構築できます。以下に例を示します。

```
val sess = // get connection to Snowflake

val sales:DataFrame = sess.table("sales")
val line_items:DataFrame = sess.table("sales_details")

val query = sales.join(line_items, sales("id") === line_items("sid"))
    .groupBy(line_items("product_id"))
    .count()
```

Snowpark APIはファーストクラスの言語コンストラクトを使用します。そのため、開発環境で型チェック、IntelliSense、エラー報告などのファーストクラスのサポートが提供されます。またこれらの操作をシームレスにSQLに変換し、Snowflakeのハイパフォーマンスでスケラブルなエンジン内で実行できるようにします。

しかし、Snowpark APIでできることはクエリの作成だけではありません。独自のカスタムロジックを使用することもできます。カスタムコードを使用してPIIをマスキングする例を以下に示します。

```
val maskPii = (s:String) => {
    // Custom PII detection logic.
}
```

Snowpark APIを利用すればこのコードを簡単にユーザー定義関数(UDF)として分類し、DataFrame操作に使用できます。

```
val maskPiiUdf = udf(maskPii)
sess.table("emails")
    .withColumn("body", maskPiiUdf(col("body")))
    .show()
```

Snowpark APIは、ロジックがデータの近くで実行されるように、ロジックをSnowflakeにプッシュします。

コードはSnowflakeウェアハウス内のセキュアなサンドボックス型JVM (Java仮想マシン)でホストされます。

自動化により手作業のコーディングが不要に

次に、Snowparkが一般的な操作をどのように簡素化するかを確認するため、Snowparkを使用してPII検出ロジックをテーブル内のすべての文字列に適用する例を見てみましょう。SQLでは、テーブルごとにクエリを手作業でコーディングするか、クエリを生成するコードを記述する必要があります。Snowparkでは、以下の汎用ルーチンを簡単に記述できます。

```
val maskTable = (df:DataFrame) => {
    df.select(df.schema.map(field =>
        if (field.dataType == StringType) maskPiiUdf(col(field.name))
        else col(field.name))
    )
}
```

その後、この汎用ルーチンを使用して、任意のテーブルにあるすべてのPIIを簡単にマスキングできます。

```
val maskedEmails = maskTable(sess.table("emails"))
```

この数行のコードを記述するだけで、Snowparkに堅牢なスキーマドリブンのクエリを動的に生成させることができます。

JAVA UDFを使用して複雑なロジックを構築

SQLは今でもSnowflakeの必須要素であり、SQLユーザーはJava UDFを使用してプラットフォームの新機能のメリットを最大限に活用できます。

Java UDFでは、シンプルな関数インターフェイスを公開する複雑なロジックを構築できます。

```
public class Sentiment
{
    public float score(String text)
    {
        // Your sentiment analysis logic here.
    }
}
```

これらの関数は、ソース制御、開発環境、デバッグツールなどの既存ツールセットと必要な任意のライブラリを用いて構築できます。Snowparkでは、GitHubなどのソースから有用なコードを取得してSnowflake内で使用できます。

コードをSQL内に取り込むプロセスは簡単です。1つもしくは複数のJARを作成してSnowflakeにロードし、関数を登録します。

```
create function sentiment(txt string) returns float
language java
imports = ('@jars/Sentiment.jar')
handler = 'Sentiment.score';
```

これで、SQLユーザーは他のユーザーが作成したロジックを他の関数と同様に使用できるようになります。

```
select id, sentiment(body)
from emails;
```

Java UDF1では既存のツールセットを用いて複雑なケースを構築できます。しかし、ユースケースがより基本的なものである場合もあります。そのため、この新機能にはシンプルなインライン定義も含まれています。

```
create or replace function reverse(s string) returns string
language java
handler = 'Reverse.reverse'
target_path = '@jars/Reverse.jar'
as
$$
public class Reverse
{
    public String reverse(String s)
    {
        return new StringBuilder(s).reverse().toString();
    }
}
$$;
```

SNOWPARKのデータエンジニアリング ユースケース

データエンジニアリングは、以下の2つの理由から複雑な作業となっています。第一に、データエンジニアリングでは一般に複数のシステムやソリューションを使用する必要があります。そのため、データパイプラインが柔軟性に欠けた過度に複雑なものになりがちです。その主な理由の1つは、データエンジニアリングが複数のチームによる共同作業であることです。データアナリストはSQLやGUIベースのツールを好む場合があります。一方、データサイエンティストは一般にノートブックとPythonを使用してデータを準備することを好みます。また、データエンジニアや開発者は複雑なコードやプログラミングコンストラクトに対処するために追加のツールを必要とします。データは通常、パイプラインを稼働させるために、これらのさまざまなシステム間を移動する必要があります。そのため、アーキテクチャが複雑になり、セキュリティやデータガバナンスが損なわれる可能性があります。

第二に、データ処理インフラストラクチャの管理と操作には通常、多大な手作業とメンテナンスのオーバーヘッドが伴います。その結果、データエンジニアは、多くの場合、あちこちでパイプラインを保守および修復することにほとんどの時間を費やすことになります。

Snowparkはこれらの問題を解決するように設計されています。詳細については、以下に示す当社のパートナーからのユースケースをお読みください。

ユースケース:TALENDを使用して信頼できるデータを提供する

分析機能の良し悪しはデータの品質によって決まります。データの品質を評価するにはどうすればよいのでしょうか。Snowflake内でTalend Data Inventory製品のコア機能であるData Trust Score™を使用して、データに対するヘルスチェックを実行できるようになりました。Trust Score™を使用すると、データの重大な問題を特定して診断できます。また、すべての主要なステークホルダーが、より信頼性の高い意思決定に向けた取り組みに乗り出すことができます。

通常、外部アプリケーションを使用して大量のデータセットをプロファイリングする場合は、データが保存されているシステムとは異なるシステムで分析を実行できるようにデー



データのサンプルを抽出します。しかし、データをSnowflakeの外部に移動すると、データセキュリティとプライバシーのリスク、データの入出力に伴うコスト、拡張性の低いリソースによる処理の失敗、結果がデータのサンプルに基づいているために不正確になることなどの問題が発生します。

SnowflakeデータセットのTrust Score™を計算すると、これらの問題の多くが軽減されます。スコアを計算する処理が、

Snowflakeの内部でJava UDFを用いてネイティブに行われるためです。また、結果がサンプルデータだけではなくデータセット全体に基づいているため、精度も向上します。Java UDFは、SnowparkまたはSQLを使用して呼び出すことができます。Talendは両方のアプローチを使用しました。1つはプロトタイプフェーズ中に使用し、そしてもう1つは当社の製品であるTalend Data Inventoryで機能を有効化する

ときに使用しました。今すぐSnowflake Partner ConnectでTalendを試し、健全なデータの文化に向けた取り組みを開始しましょう。

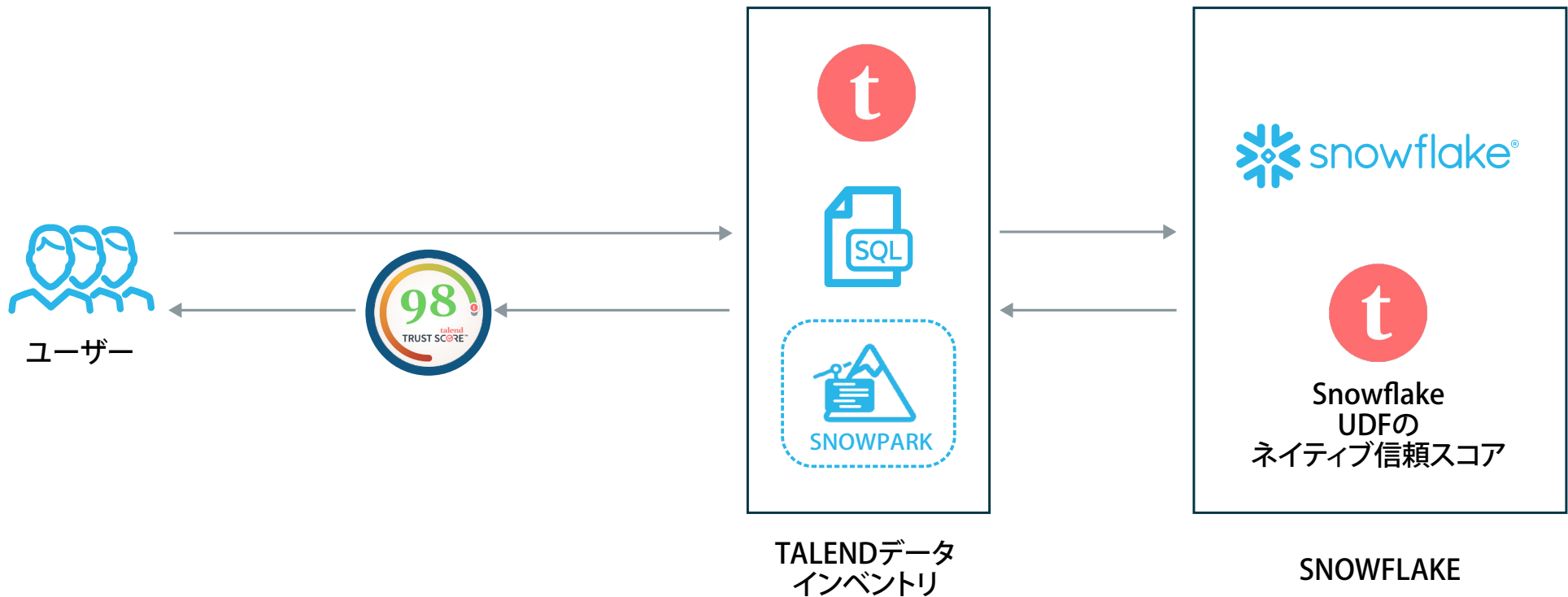


図2: SnowflakeとTalendを使用して信頼できるデータを提供する!

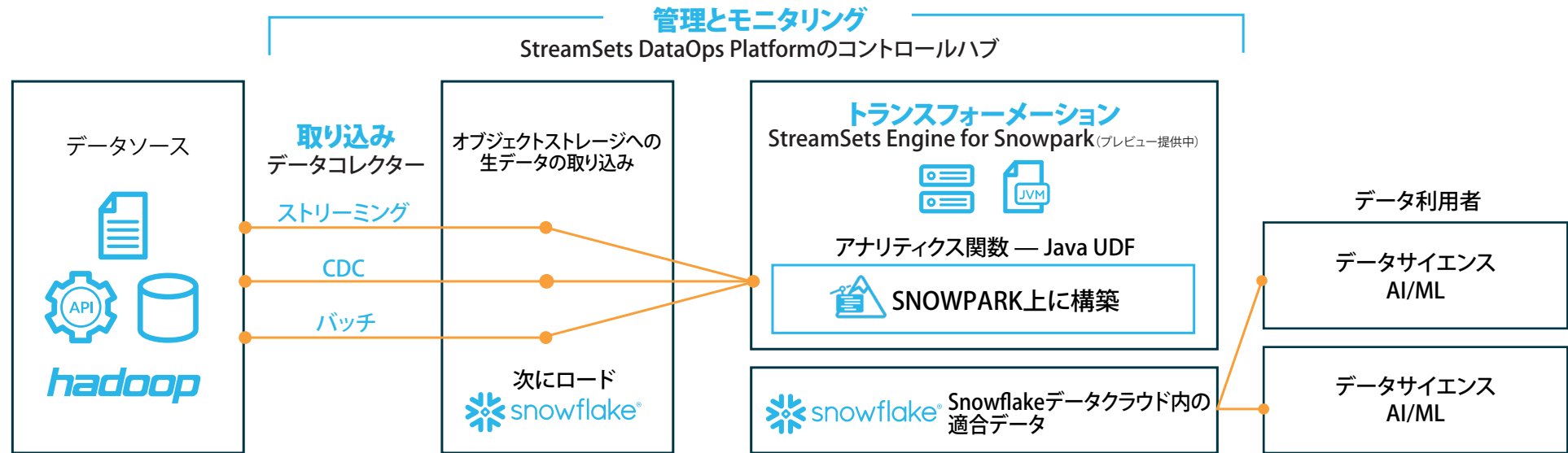


図3: StreamSets engine for Snowparkを使用してDataOpsを最適化する³⁾

ユースケース: STREAMSETS ENGINE FOR SNOWPARKを使用してDATAOPSを最適化する

StreamSets engine for Snowparkは、Snowparkを基盤に構築されており、Snowparkの多言語サポートの柔軟性および表現力とデータクラウドオペレーションのシンプルさを実現します。

データエンジニアは、StreamSets DataOps Platformを使用することで、SQLの枠を超えた強力なデータパイプラインを表現できます。

直感的なGUIでScalaまたはJavaを使用することで、コードを一切使用せずに済みます。また、必要に応じてコードを投入することもできます。StreamSets engine for Snowparkは、複雑なデータパイプラインを大規模にモニタリングおよびオーケストレーションするための組み込みの機能というStreamSets DataOps Platformのメリットを備えています。これらの機能はすべてクラウドに配置され、追加のハードウェアは必要ありません。

ユースケース: DATAOPS.LIVEによる開発ライフサイクルの管理

DataOps.liveとSnowflakeとのパートナーシップの拡大により、開発、テスト、および本番運用のライフサイクル全体にわたってSnowparkワークロードとその他すべてのオブジェクトをSnowflake内で完全に管理できるようになりました。DataOps.live Gitレポジトリには、通常の「あらゆるデータの信頼できる情報源」がすべて格納されます。これに加えて、このレポジトリは通常のソフトウェア開発要件をすべて満たします。DataOps.liveにより、他のソフトウェアプロジェクトと同じようにソフトウェアのブランチ化、バージョン管理、コンパイル、テスト、およびデプロイとアーチファクトの生成を行うことができます。

Snowpark APIを実行することは、実際にはSnowparkライブラリが使用されているSnowparkアプリケーションを実行することを意味します。たとえば、Scalaが使用されている場合は、その特定の言語用のランタイムツールと適切な

ライブラリ(さらにSnowparkライブラリ自体)を備えた環境が必要になります。

多くの場合、Snowparkアプリケーションは、高度なデータ操作、特にSQL内で達成できない操作を実行するために使用されますが、その結果は引き続きSnowflakeに格納されます。このような場合は、DataOps.liveのモデリングおよび変換エンジン内の自動データテストを使用してSnowparkアプリケーションの結果を検証できます。

SNOWPARKのデータサイエンスユースケース

Snowparkのデータプログラマビリティの進化により、柔軟性と拡張性が向上します。そのため、データサイエンティストは、Pythonなど自分の好きなプログラミング言語を用いて、機械学習ワークフローの一環としてデータにアクセスし、データを可視化および処理できます。これにより企業は、非構造化データやサードパーティデータを含むデータの価値の最大化を実現できます。

Snowflakeのデータサイエンスパートナーは、このような進化を活用し、より深く統合されたエクスペリエンスによって機械学習ワークフローを加速させる取り組みを積極的に進めています。Snowparkでは、データサイエンティストは自分の好きなノートブックで作業しながら、データが保存さ

れている場所に処理をプッシュダウンすることができます。Snowparkのデータサイエンスユースケースの例を以下にいくつか示します。

- 特徴量エンジニアリング
- MLモデル推論
- SQLによるSnowflake内でのエンドツーエンドのML

ユースケース: SNOWPARKでAIのトレーニングと推論を呼び出せるようにDATAFRAMEを準備する

Snowparkでは、Snowflake内のデータを、Snowflake環境内で実行されるScalaおよびJavaコードでDataFrameとして使用できます。

Snowparkは、複雑なデータパイプラインを簡単に構築できるようにするとともに、Jupyter、Dataiku、H2O.ai、Zepl (現在はDataRobotプラットフォームの一部) を使用したノートブックベースのプログラミングの一環としてコードを使用しているデータエンジニア、データサイエンティスト、デベロッパーを支援できるように設計されています。Snowparkでは、これらのデータ技術者が自分の好きな言語を使用し、DataFrameなどのなじみ深い概念に基づいて特徴量エンジニアリングを促進し、その後、これらのワークロードをSnowflake内で直接実行できます。たとえば、データサイエンティストはH2O.aiノートブックで新しい特徴量を定義し、その処理をSnowflakeで実行することで、Snowflakeのスケラビリティやパフォーマンスのメリットを享受できます。

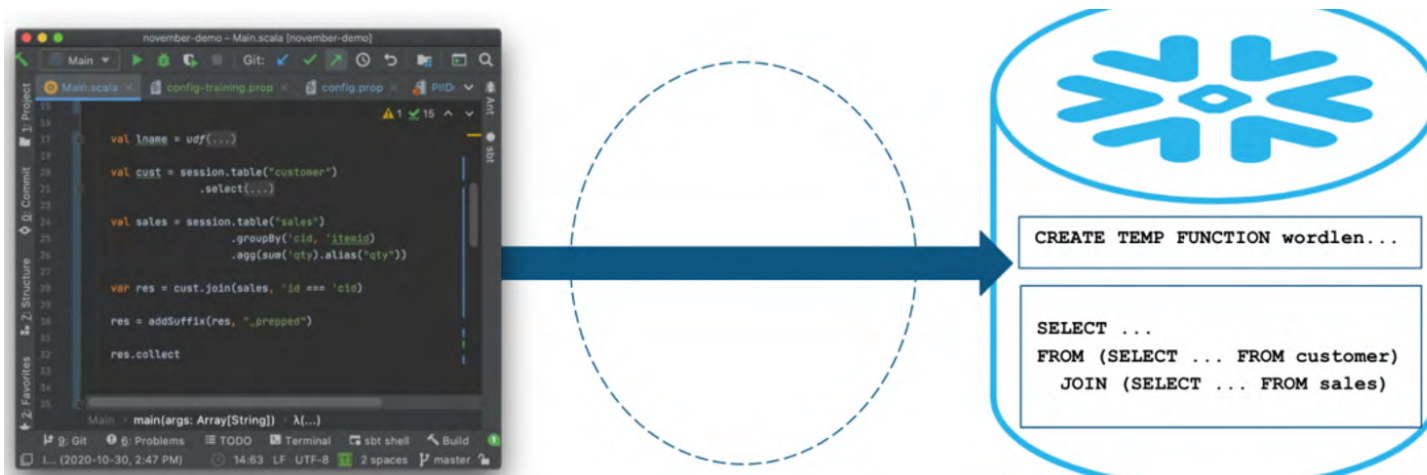


図4: H2O.aiノートブックでDriverless AIトレーニングを呼び出せるようにDataFrameを準備する⁷

ユースケース: DATAIKUでデータ準備関数と予測モデルスコアリングを拡張する

Snowparkが登場する前、一部のデータ準備関数と予測モデルスコアリングはDataikuなどのエンジンで実行されました。これらはSQLで表現できなかったためです。これを行うにはデータをSnowflakeから出し入れする必要があり、その結果としてパフォーマンスに悪影響が及びました。今ではDataikuのパイプラインを完全にSnowflake内で稼働させることにより、Snowflakeのパフォーマンスやガバナンスのメリットを最大限に享受することができるようになりました。現在、顧客は、SnowflakeでDataikuパイプラインを完全に稼働させることで、Snowflakeのパフォーマンスとガバナンスのメリットを最大限に活用できるようになりました。

Dataikuでデータ準備パイプラインを構築したいと考えているノーコードユーザーとローコードユーザーは、Java UDFを使用してSnowflakeエンジンのメリットを活用できます。Java UDFを使用すると、Javaで表現されたワークロードをSnowflake内のJVMで実行できます。

DataikuのコアエンジンはJavaであるため、Dataikuでデータ準備関数を再パッケージ化し、Snowflake内にプッシュダウンして実行すると便利です。これにより、任意の数のユーザー、ジョブ、またはデータに対応できるよう拡張することが容易になると同時に、アーキテクチャが簡素化されます。

ユースケース: SNOWFLAKE内のモデル推論を活用して本番運用への道筋を簡素化する

モデル推論を大規模に実行する場合、オペレーションチームは通常、モデルをデプロイするため、管理された「信頼できる情報源」から別の環境に本番データを移動する必要があります。大量のデータを移動すると、コストとセキュリティの両面に悪影響が及びます。そのため、モデルを本番環境で運用できなくなることがよくあります。

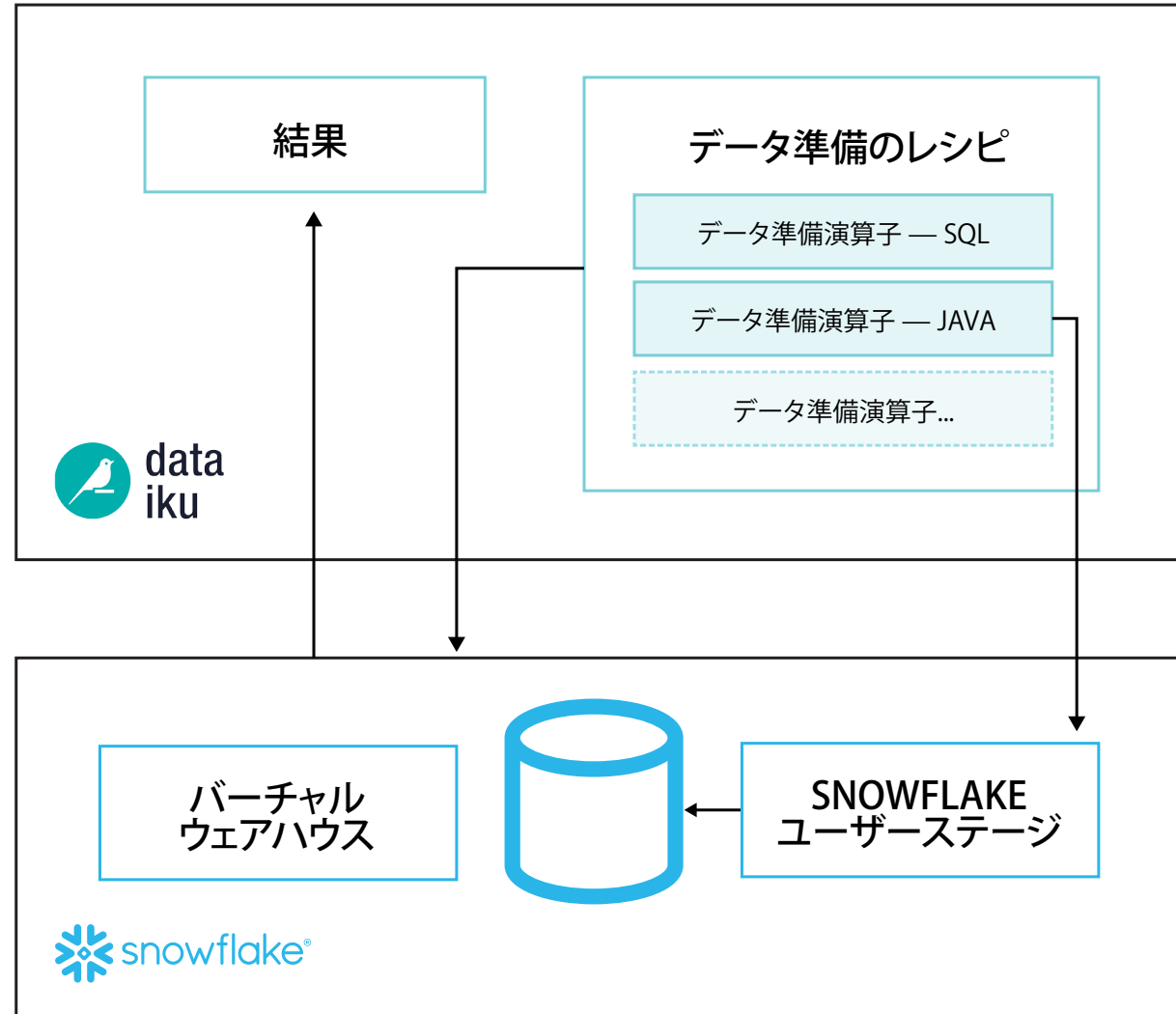


図5: Dataikuでデータ準備関数と予測モデルスコアリングを拡張する⁸

データをその保存場所から移動することに伴う複雑さを解決するため、MLの開発およびオペレーションチームは、Java UDFを使用してモデルをデータの保存場所に移動するアプローチに移行できます。Java UDFでは、Javaで表現されたトレーニング済みモデルを使用して、Snowflake内で推論を実行できます。

SnowflakeのデータサイエンスパートナーであるDataRobotとH2O.aiは、モデルをエクスポートしてSnowflake内にデプロイすることで、スケーラブルなバッチ志向のモデル推論を実行する機能と、外部関数を呼び出してリアルタイムの推論を実行する機能をすでに導入しています。

AI CloudであるDataRobotは、お使いのデータクラウド向けに継続的なAIライフサイクル開発・管理サービスを提供します。DataRobotを使用すると、Snowflakeのデータをシームレスに結び付けて活用し、そのデータをAI用に変換して準備したり、Snowflake内で特徴量検出を自動的に実行したり、AIモデルを構築してからそれらのモデルをJARファイルとして本番環境にエクスポートし、Snowflake内にデプロイしてスケーラブルなモデル推論を実行したりできます。

DataRobot AI Cloudには、スコアリングコードの生成などの柔軟なデプロイメントオプションが用意されています。DataRobotにより、SnowflakeにサポートライブラリがあるすべてのモデルをSnowflake Java UDF内に直接デプロイできるようになりました。このデプロイメントオプションにより、以前はSnowflakeの外部で実行されていた従来のモデルスコアリング手法を超える高いスコアリングスピードが実現します。

また、DataRobotユーザーは、サービスおよび予測データをDataRobot AI Cloudに再び取り込んで、モデルの経時的傾向を分析できます。さらに、DataRobotでは、Snowflake内にデプロイされた、DataRobot以外で作成されたモデルをモニタリングすることで、複数のAIイニシアチブにまたがって利用可能な「信頼できる唯一のAIソース」を維持できます。

H2O.aiは、あるユースケースで、同社が所有する、債務不履行を予測するためのデータセットと、LendingClubが公開しているデータセットおよびSnowflakeデータマーケットプレイスから得られた人口統計データを組み合わせました。これにより、Snowflake環境内でモデルを拡張できるようになったため、モデルの精度が向上するとともに、スコアリング(推論時間)が減少しました。H2O.aiは、同社のデータとSnowflakeデータマーケットプレイスからのサードパーティデータに関するモデルのトレーニングを終えると、データが保存されている場所にモデルをインポートできるようになりました。これにより、データが保存されている場所にモデルが移動されたため、デプロイメントが容易になったとともに、推論の効率性が向上しました。

最後に、Snowflake内でモデルを実行すると、現在使用中のMLプラットフォームをモデルモニタリングに使用し続けることができます。たとえば、DataRobotユーザーは、サービスおよび予測データをDataRobot MLOpsに再び取り込んで、モデルの経時的傾向を分析できます。

また、Dataikuは同じ技術を用いて機械学習モデルをパッケージ化し、Snowflakeにデプロイすることで、データをSnowflakeの外に移動することなく予測スコアリングを実行できるようにしています。

```

Portable predictions allow DataRobot models to be deployed on a variety of external infrastructures. These models can report monitoring statistics back to this deployment using the Monitoring Agent.

1. Download Scoring Code

Portable predictions allow DataRobot to be deployed directly into Snowflake as User-Defined Functions.

To get started, first download the model's Scoring Code and follow the instructions to configure the model in Snowflake.

Download Scoring Code (.jar)

2. Configure and execute installation script

This script uploads the JAR to a Snowflake stage and creates a User-Defined Function for making predictions using Scoring Code.
Update the script template with the desired parameters for the Snowflake warehouse, database, schema and JAR file location: Copy to clipboard

1 -- Replace with the warehouse to use
2 USE WAREHOUSE my_warehouse;
3 -- Replace with the database to use
4 USE DATABASE my_database;
5
6 -- Replace with the schema you want to use
7 CREATE SCHEMA IF NOT EXISTS scoring_code_udf_schema;
8 USE SCHEMA scoring_code_udf_schema;
9
10 -- Update this path to match the Scoring Code jar's location
11 PUT 'file:///path/to/6081ee9727a78f2a2247cafa-6076150513693237b74e6997.jar' '@~/jars/' AUTO_COMPRESS=FALSE;
12
13 -- Create the UDF
14 CREATE OR REPLACE FUNCTION datarobot_deployment_6081ee9727a78f2a2247cafa(RowValue Object)
15 RETURNS OBJECT
16 LANGUAGE JAVA
17 IMPORTS=('@~/jars/6081ee9727a78f2a2247cafa-6076150513693237b74e6997.jar')

```

図6: JARをアップロードして関連するUDFを作成することで、Snowflakeで直接推論を実行するDataRobot生成スクリプト⁹

SNOWPARKのデータガバナンスおよびセキュリティユースケース

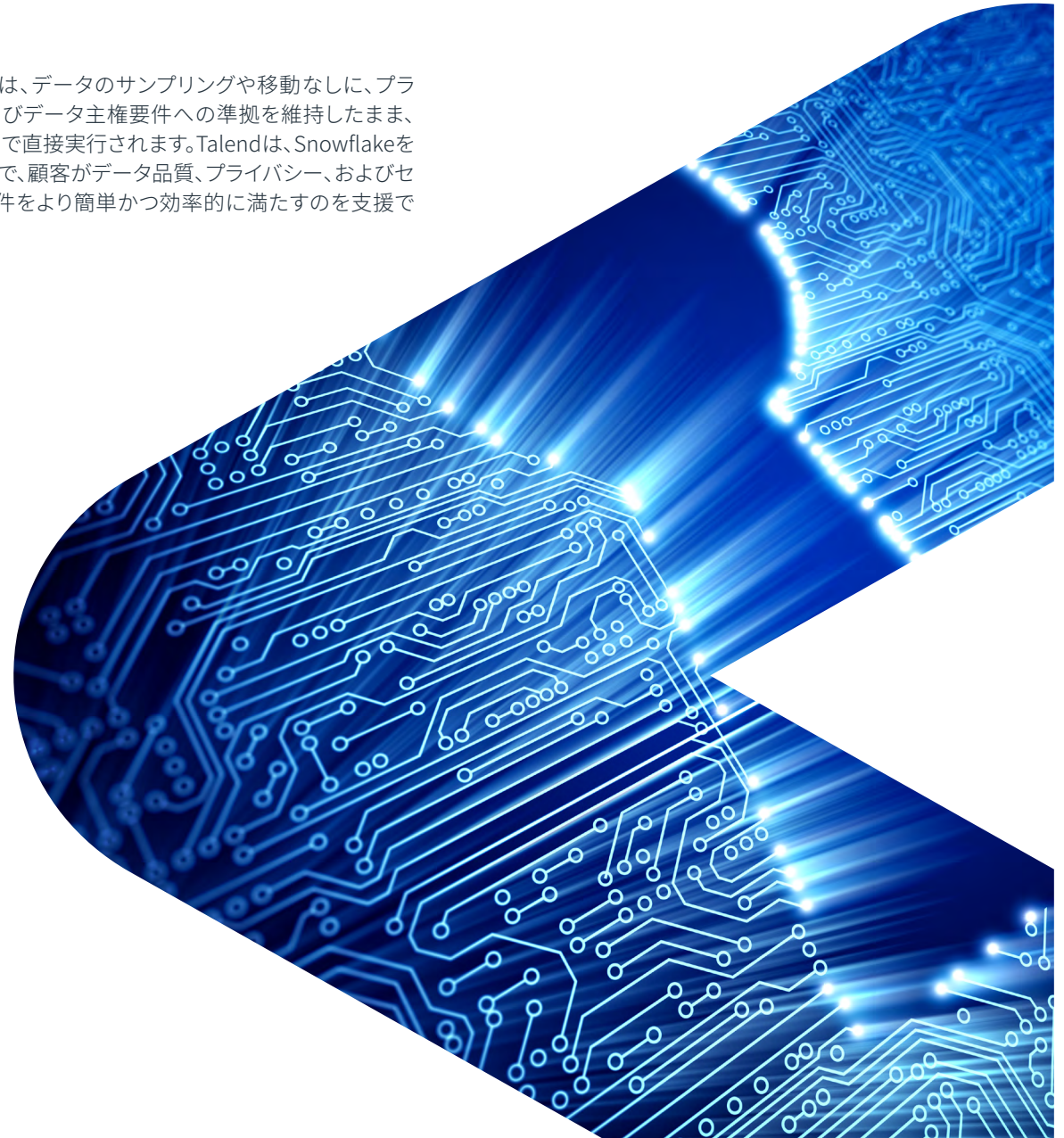
Snowparkは、データガバナンスの取り組みをサポートするソリューションの構築に利用できます。構築可能なソリューションの例を以下にいくつか示します。

- **データ品質プロファイラー:**組織はSnowparkを使用することで、データの品質を把握し、向上させることができます。Snowparkには、データをスキャンしてその完全性を確認し、異常が見つかった場合は、データ品質に関する潜在的な問題のフラグを立てる機能があります。
- **データ分類:**Snowflakeの分類機能によりPIIが自動的に検出されますが、Snowparkを使用すると、組織にとって重要なその他の種類のデータを分類するロジックを構築できます。
- **フォーマット維持型マスキング機能:**Snowparkを使用すると、Snowflakeの動的データマスキング機能に対するよりカスタマイズされたニーズをサポートできます。たとえば、クレジットカード番号をマスキングまたは置換するカスタム関数を作成できます。

ユースケース:TALENDを使用して信頼できるデータを提供する

前述のデータエンジニアリングユースケースのいくつかは、データガバナンスに非常に近いものです。Snowparkにより、パートナーはSnowflake内で直接ソフトウェアを実行してデータ移動の必要性を減らすことで、顧客のデータガバナンス態勢を簡素化できます。たとえば、SnowflakeとTalendの顧客は、Talend Trust Assessorを使用することで、Snowflakeのすべてのデータに対するヘルスチェックを1クリックで即座に実行できます。Snowparkでは、このへ

ルスチェックは、データのサンプリングや移動なしに、プライバシーおよびデータ主権要件への準拠を維持したまま、Snowflake内で直接実行されます。Talendは、Snowflakeを使用することで、顧客がデータ品質、プライバシー、およびセキュリティ要件をより簡単かつ効率的に満たすのを支援できます。



他のSNOWFLAKEパートナーの ユースケース

LTI

LTI Mosaic⁵は、Snowparkを使用することで、以下のSnowflake機能のシンプル化と拡張に成功しています。

- DataOps (データへのDevOps原則の適用)
- MLOps (機械学習オペレーション)
- ModelOps (分析ライフサイクルを通じてモデルを迅速かつ反復的に移動させる包括的手法)

Mosaicの顧客は、Snowparkを活用することで、機械学習モデルをデプロイしたり、モデル推論をプッシュしたり、Snowflakeにプッシュして処理できるコードを記述したりできます。

phData

phDataの顧客は、Snowflakeデータに対して自然言語処理(NLP)や画像認識などの機会学習タスクを実行する必要があります。これまで、これはSQLのみでは実行できない複雑なプロセスでした。現在、phDataの顧客は、この複雑なパイプラインをSnowflake内で管理し、コストと複雑さを大幅に軽減できるようになりました。⁶



SNOWPARKを導入する

Snowparkの導入を検討されている方は、当社のドキュメンテーションとステップバイステップクイックスタートガイドをご覧ください。私たちは、皆様がこれから構築する素晴らしいソリューションに期待しています。

寄稿者：

Carlos Bouloy

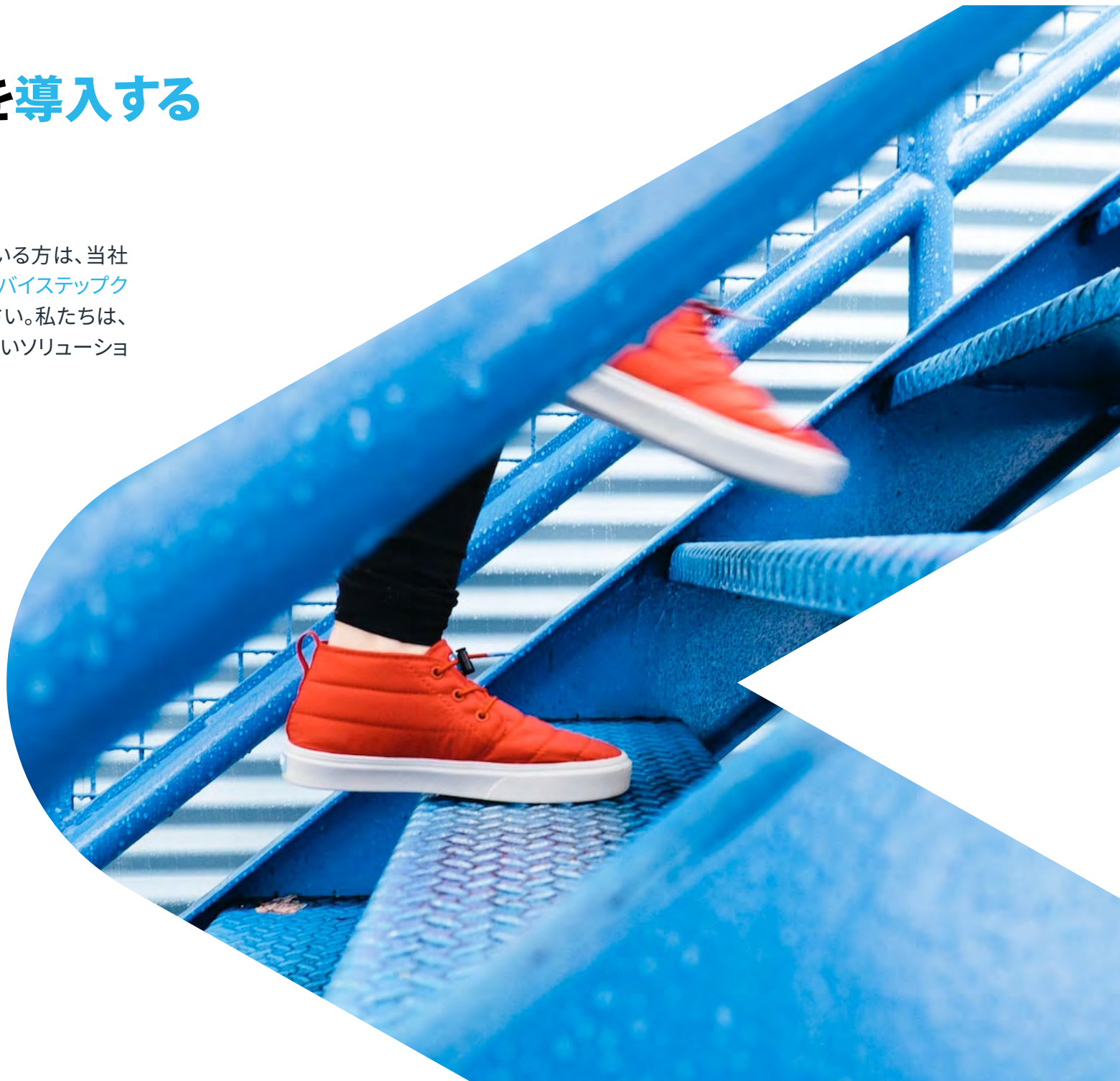
Frank Pacione

Julian Forero

Miles Adkins

Paul Gantz

Shiyi Gu





SNOWFLAKEについて

Snowflakeは、Snowflakeのデータクラウドを用い、あらゆる組織が自らのデータを活用できるようにします。お客様には、データクラウドを利用してサイロ化されたデータを統合し、データを発見してセキュアに共有し、多様な分析ワークロードを実行していただけます。データやユーザーがどこに存在するかに関係なく、Snowflakeは複数のクラウドと地域にまたがり単一のデータ体験を提供します。多くの業界から何千ものお客様（2021年10月31日時点で、2021年Fortune 500社のうちの223社を含む）が、Snowflakeデータクラウドを自社のビジネスの向上のために活用しています。詳しくは、[snowflake.com](https://www.snowflake.com)をご覧ください。



©2021 Snowflake Inc. All rights reserved. Snowflake、Snowflakeのロゴ、本書で使用されているその他すべてのSnowflakeの製品、機能、サービスの名称は、米国その他の国におけるSnowflake Inc.の登録商標または商標です。本書で言及または使用されているその他すべてのブランド名またはロゴは、識別目的でのみ使用されており、各所有者の商標である可能性があります。Snowflakeが、必ずしもかかる商標所有者と関係を持ち、または出資や支援を受けているわけではありません。

参考

1 bit.ly/2XAxZGU
2 bit.ly/3nZCICA
3 bit.ly/3tWMAbw
4 bit.ly/3Avdr0V
5 bit.ly/3nO8VY1
6 bit.ly/3zyQpEZ

7 bit.ly/3tLY1mh
8 bit.ly/39bbue0
9 bit.ly/3tNTkZh
10 bit.ly/3zcZzqH
11 bit.ly/3tLY1mh
12 bit.ly/3kgmugA