



# [AWS Black Belt Online Seminar]

## AWS Glue

-Glue Studioを使ったデータ変換のベストプラクティス-

Amazon Web Service Japan, K. K.  
Solutions Architect, Kazutaka Kubo

2021/03/30

AWS 公式 Webinar  
<https://amzn.to/JPWebinar>



過去資料

<https://amzn.to/JPArchive>



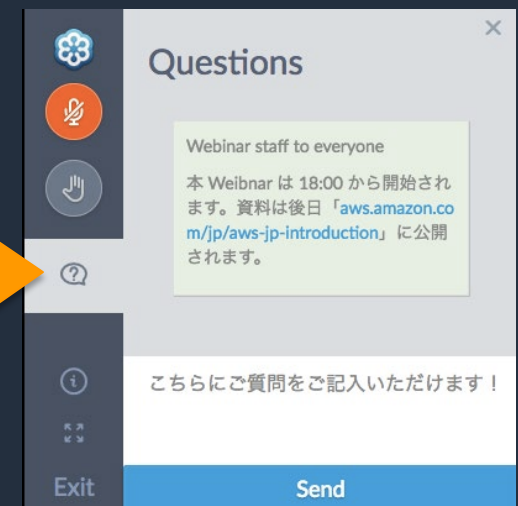
# AWS Black Belt Online Seminar とは

「サービス別」「ソリューション別」「業種別」のそれぞれのテーマに分かれて、アマゾンウェブサービス ジャパン株式会社が主催するオンラインセミナーシリーズです。

質問を投げることができます！

- 書き込んだ質問は、主催者にしか見えません
- 今後のロードマップに関するご質問は  
お答えできませんのでご了承下さい

- ① 吹き出しをクリック
- ② 質問を入力
- ③ Sendをクリック



 Twitter ハッシュタグは以下をご利用ください  
#awsblackbelt

# 内容についての注意点

- 本資料では2021年3月30日現在のサービス内容および価格についてご説明しています。最新の情報はAWS公式ウェブサイト(<http://aws.amazon.com>)にてご確認ください。
- 資料作成には十分注意しておりますが、資料内の価格とAWS公式ウェブサイト記載の価格に相違があった場合、AWS公式ウェブサイトの価格を優先とさせていただきます。
- 価格は税抜表記となっております。日本居住者のお客様には別途消費税をご請求させていただきます。
- AWS does not offer binding price quotes. AWS pricing is publicly available and is subject to change in accordance with the AWS Customer Agreement available at <http://aws.amazon.com/agreement/>. Any pricing information included in this document is provided only as an estimate of usage charges for AWS services based on certain information that you have provided. Monthly charges will be based on your actual use of AWS services, and may vary from the estimates provided.

# 自己紹介

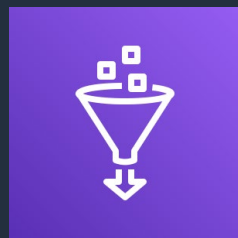
久保 和隆 (Kazutaka Kubo)

所属：西日本担当ソリューションアーキテクト

西日本のお客様に対してAWSの技術支援

好きなサービス：

AWS Glue



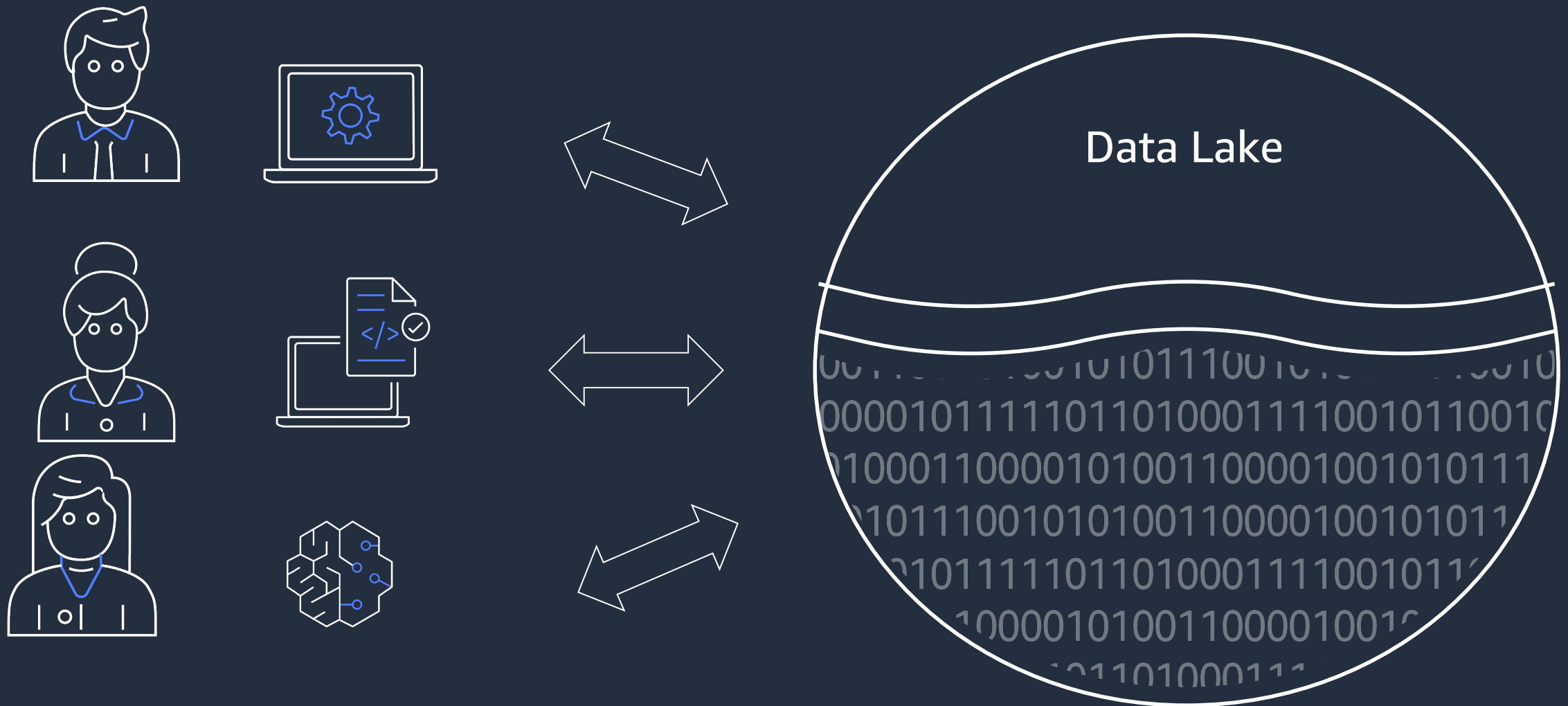
# 本日の内容

1. AWS Glueの位置付け
2. Glue Studioの概要
3. ユースケースとGlue Studioでの実装
4. その他の主要アップデート
5. まとめ

# AWS Glueの位置付け

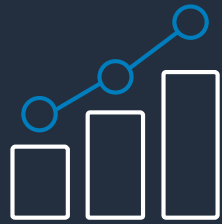
# データ分析の現状

- 多くの企業がデータレイクを構築し、有効活用を望んでいる。



# データ活用を阻む要因

- ・多くの企業でデータ活用が難航



日々増加するデータ



日々追加される  
データソース



日々追加される  
データフォーマット



様々な目的のユーザ

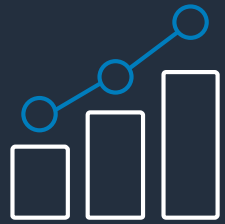


データを使いたい  
様々なアプリケーション



# データ活用を阻む要因

- ・多くの企業でデータ活用が難航



日々増加するデータ



日々追加される  
データソース



日々追加される  
データフォーマット



様々な目的のユーザ



データを使いたい  
様々なアプリケーション

使いやすい形に整形する「前処理」が、データ活用には重要

# 前処理における課題

## 取扱データの増加



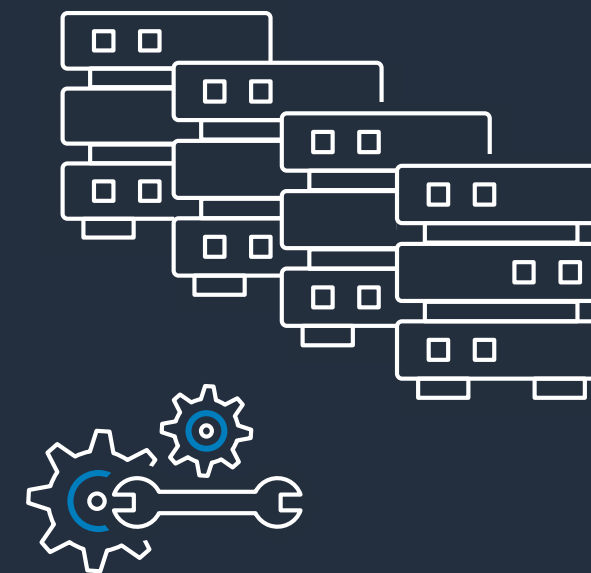
- 5年前と比較して、10倍以上のデータ量

## 目的に合わせたカスタマイズが必要



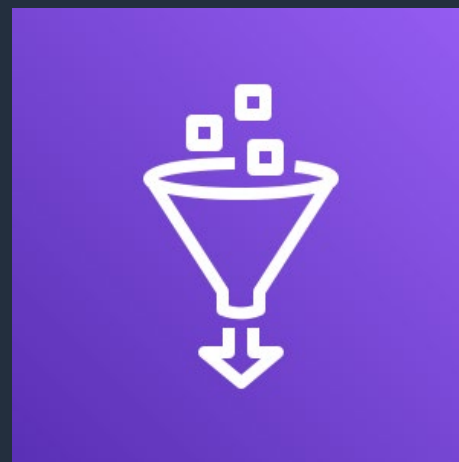
- ごみデータの排除
- フォーマット変換 等々

## 基盤の管理負荷



- サーバーのライフサイクル管理
- 監視管理

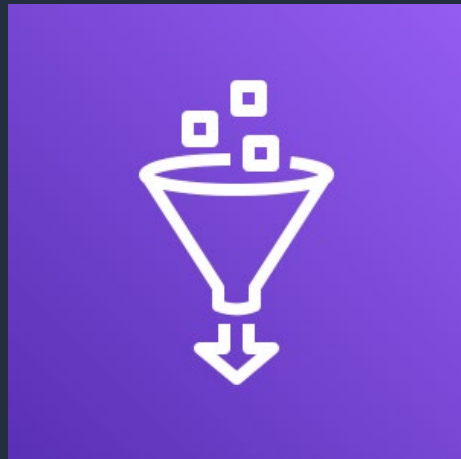
# AWS Glue



サービス間でデータを簡単に**移動**できるようにするための、  
**サーバーレスデータ統合サービス**

# AWS Glue の特徴

シンプルで拡張性に優れたサーバ不要のデータ統合



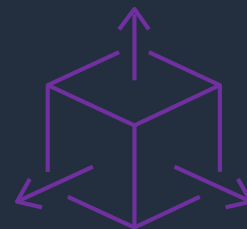
## 大量データの迅速な統合

データの準備を数ヶ月から数分に短縮



## 変換処理の自動化

何千ものETLジョブを簡単に実行、管理可能



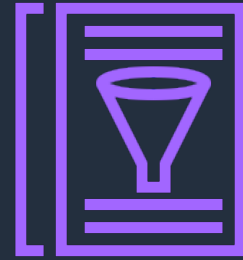
## サーバーレス

ジョブ実行で使われたリソースに対してのみ支払い

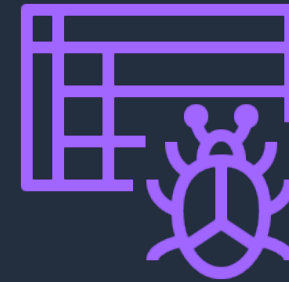
# AWS Glue の構成要素と周辺サービス



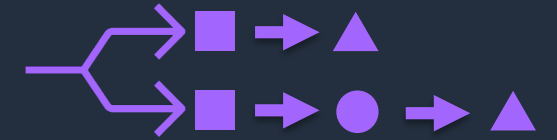
Extract, Transform, and Load (ETL) ジョブ



AWS Glue Data Catalog



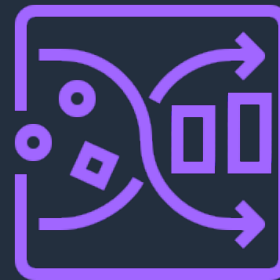
Crawler



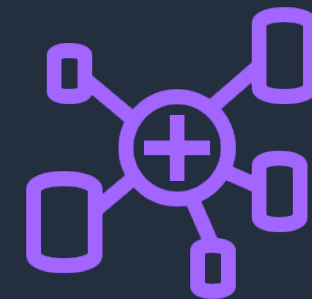
Workflow Management



AWS Glue Studio



AWS Glue DataBrew

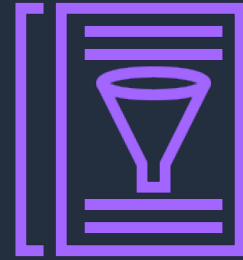


AWS Glue Elastic Views

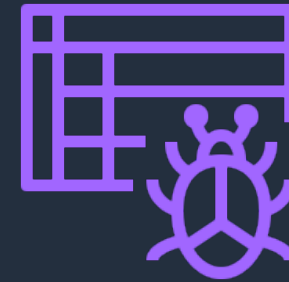
# AWS Glue の構成要素と周辺サービス



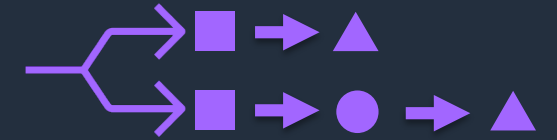
Extract, Transform, and Load (ETL) ジョブ



AWS Glue Data Catalog



Crawler

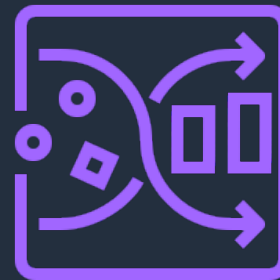


Workflow Management

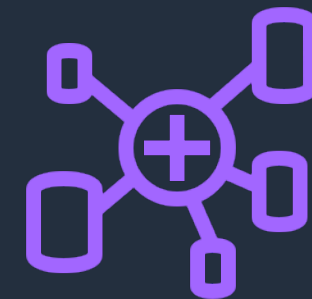
New



AWS Glue Studio



AWS Glue DataBrew

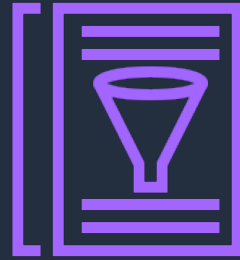


AWS Glue Elastic Views

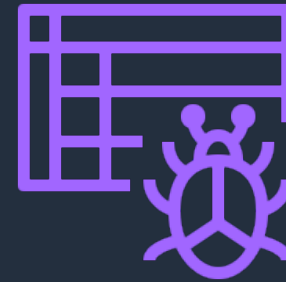
# AWS Glue の構成要素と周辺サービス



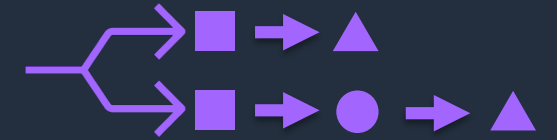
Extract, Transform, and Load (ETL) ジョブ



AWS Glue Data Catalog



Crawler



Workflow Management

フォーマット変換したり、データを結合したり、  
様々な処理を行うJOBを作成・管理・実行することが可能



AWS Glue Studio



AWS Glue DataBrew

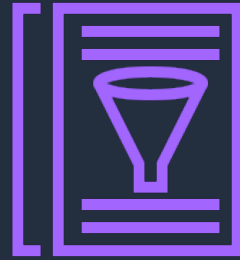


AWS Glue Elastic Views

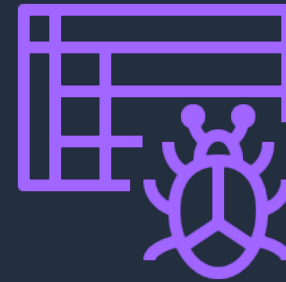
# AWS Glue の構成要素と周辺サービス



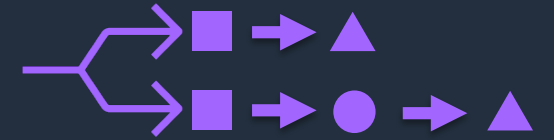
Extract, Transform, and Load (ETL) ジョブ



AWS Glue Data Catalog



Crawler



Workflow Management

Apache Hiveメタストア互換のメタデータリポジトリ  
データソースにどのようなデータが入っているのかをカタログ化して、保存しておくことが可能



AWS Glue Studio



AWS Glue DataBrew



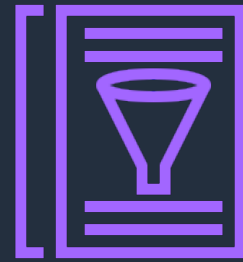
AWS Glue Elastic Views



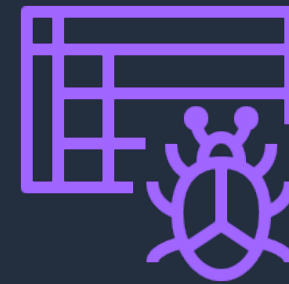
# AWS Glue の構成要素と周辺サービス



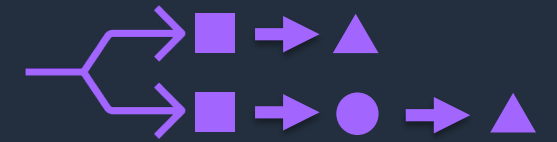
Extract, Transform, and Load (ETL) ジョブ



AWS Glue Data Catalog



Crawler



Workflow Management

New



AWS Glue Studio

Glueのデータカタログにメタデータを作成するプログラム  
分類子の優先度に従って、スキーマ情報を自動で判断し、スキーマを形成

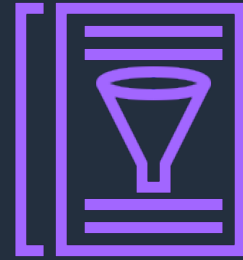
AWS Glue DataBrew

AWS Glue Elastic Views

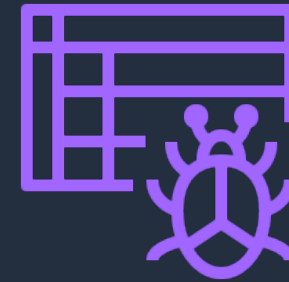
# AWS Glue の構成要素と周辺サービス



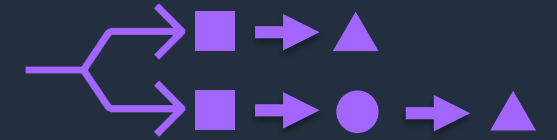
Extract, Transform, and Load (ETL) ジョブ



AWS Glue Data Catalog



Crawler



Workflow Management

New

クローラー、トリガー、ジョブのDAGを生成するワークフロー機能  
データソースのクロール/データカタログの生成 / JOBの実行、これら一連の処理を自動化



AWS Glue Studio

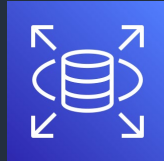


AWS Glue DataBrew

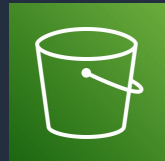


AWS Glue Elastic Views

# データレイク環境の構成要素



Amazon RDS



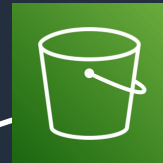
Other S3



On-premises data



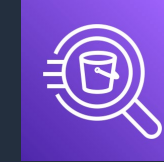
データソース



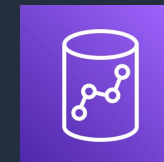
Amazon Simple Storage Service (S3)



データレイク



Amazon Athena



Amazon Redshift



Amazon SageMaker



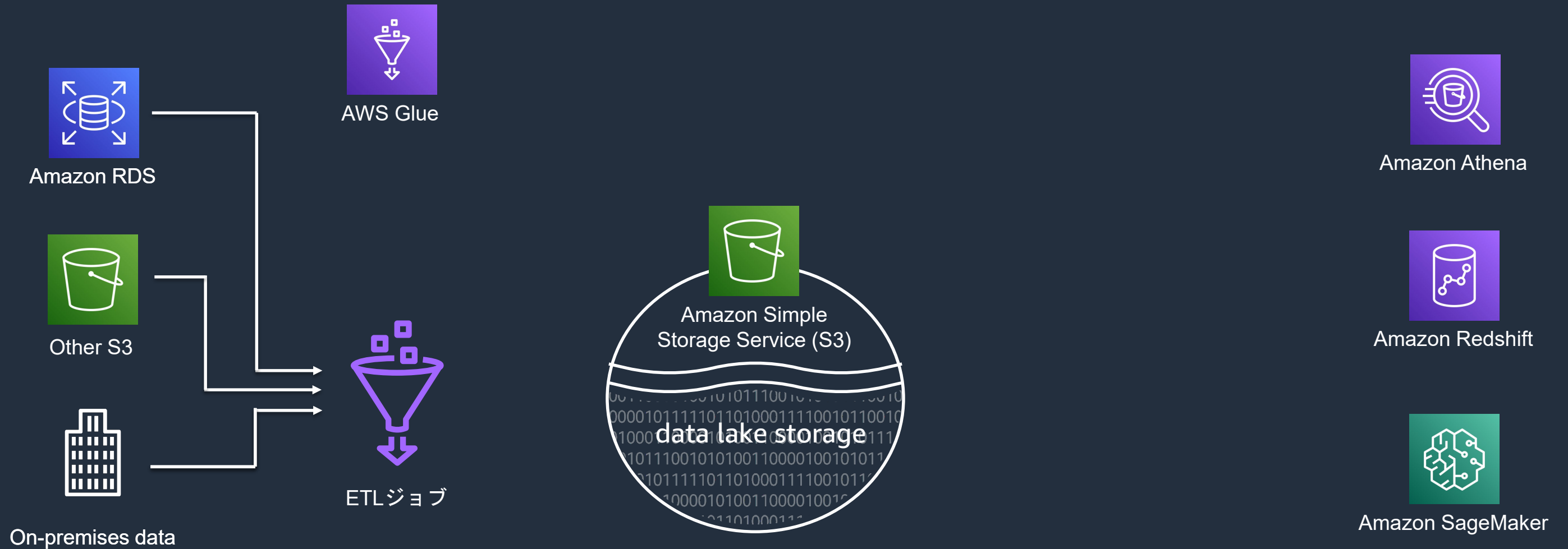
ターゲット

# データレイク環境におけるAWS Glueの役割



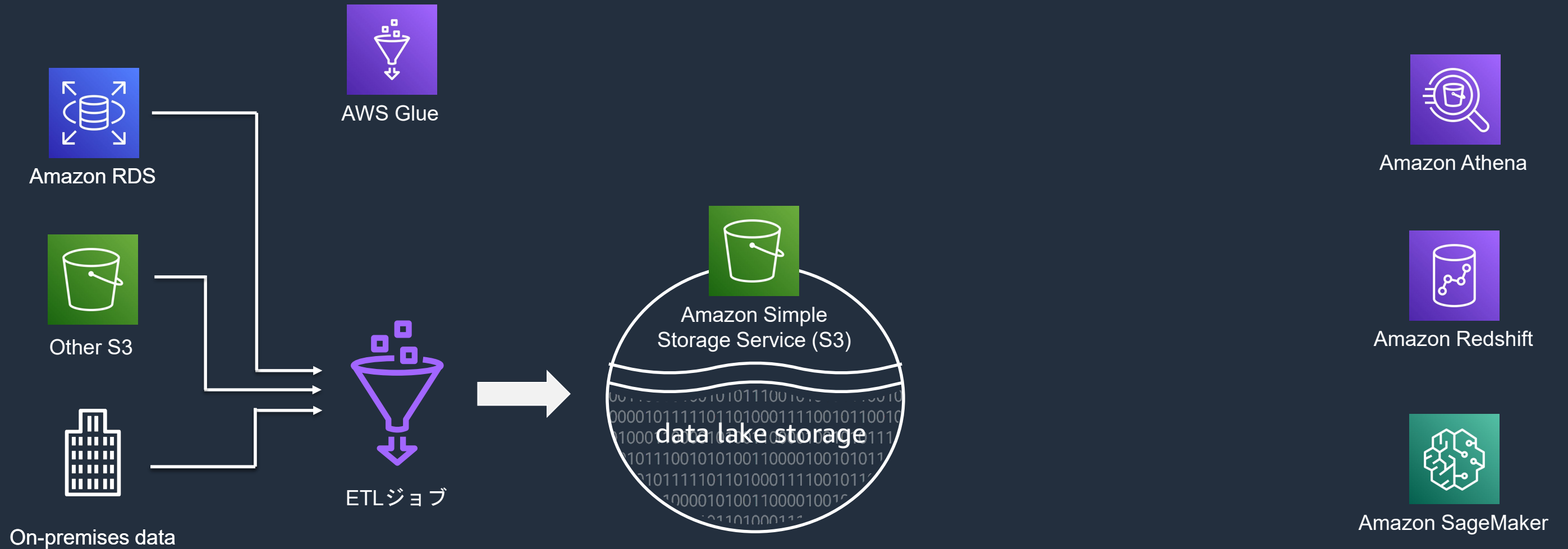
各要素の間を取り持つのがGlueの役割

# ETLジョブ



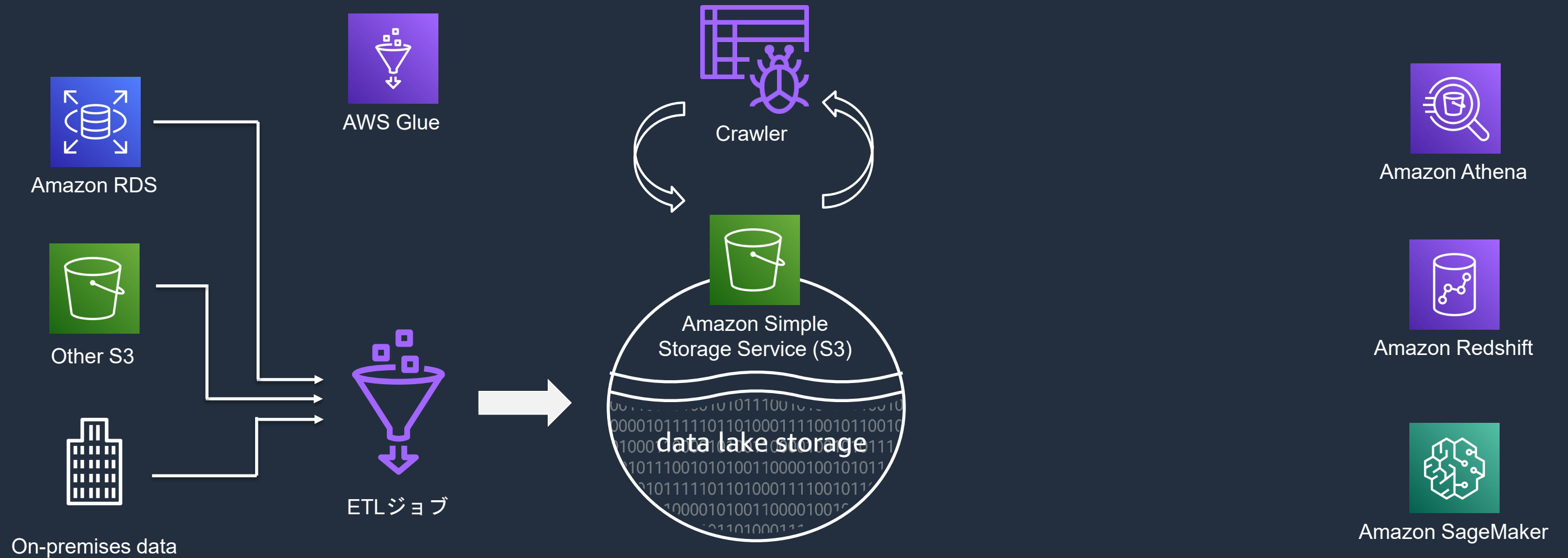
データソースからデータレイクストレージへデータを出力

# ETLジョブ



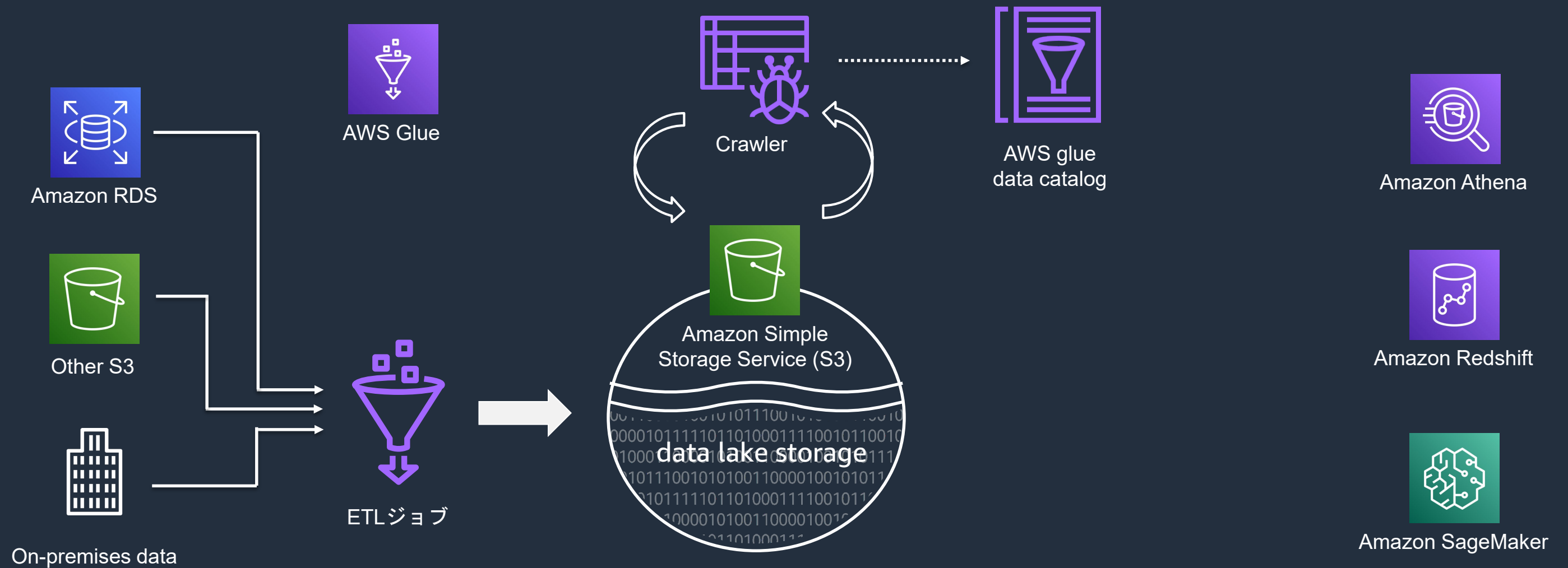
データソースからデータレイクストレージへデータを出力

# データのクローリング



データレイクをクローリングし、データカタログを出力

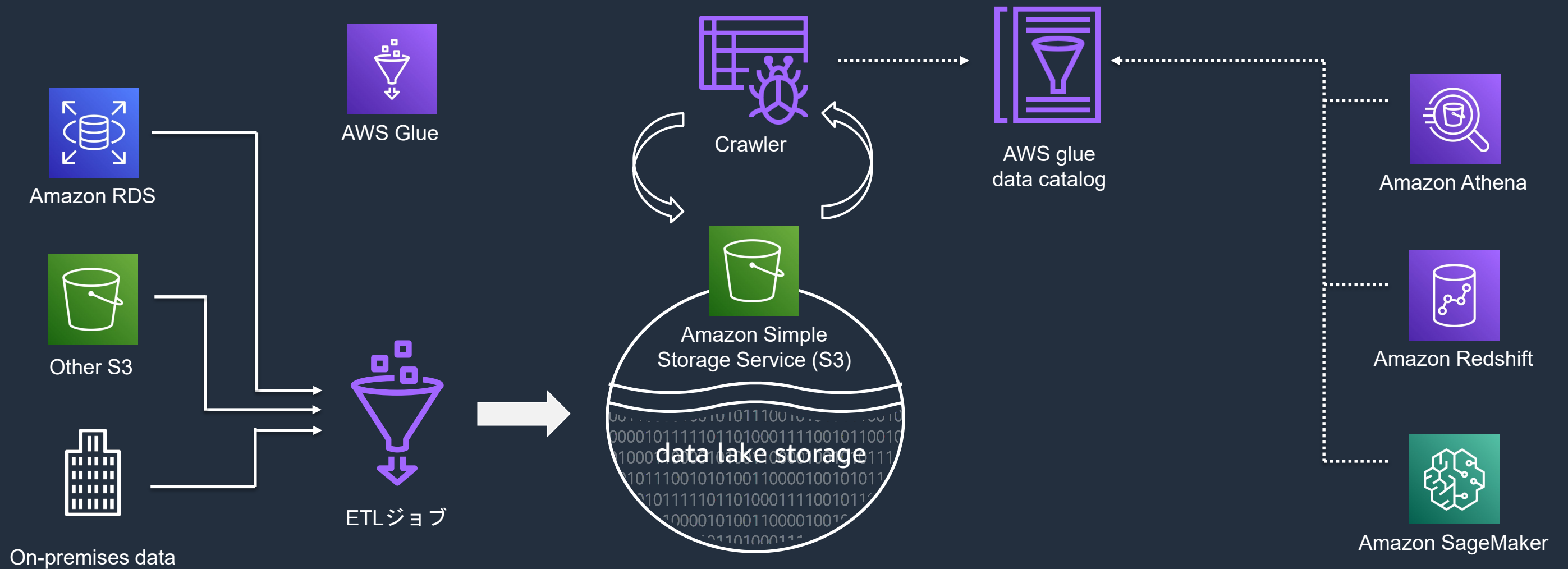
# データのクローリング



データレイクをクローリングし、データカタログを出力

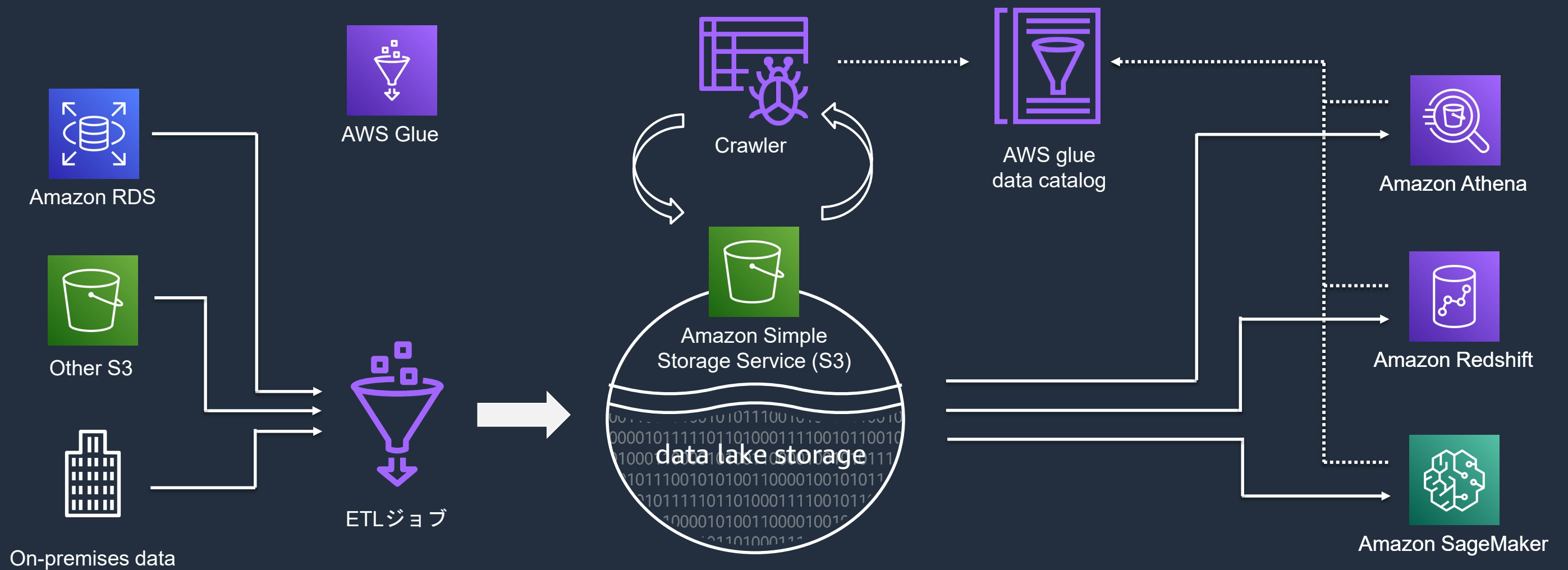


# カタログを使用したデータ参照



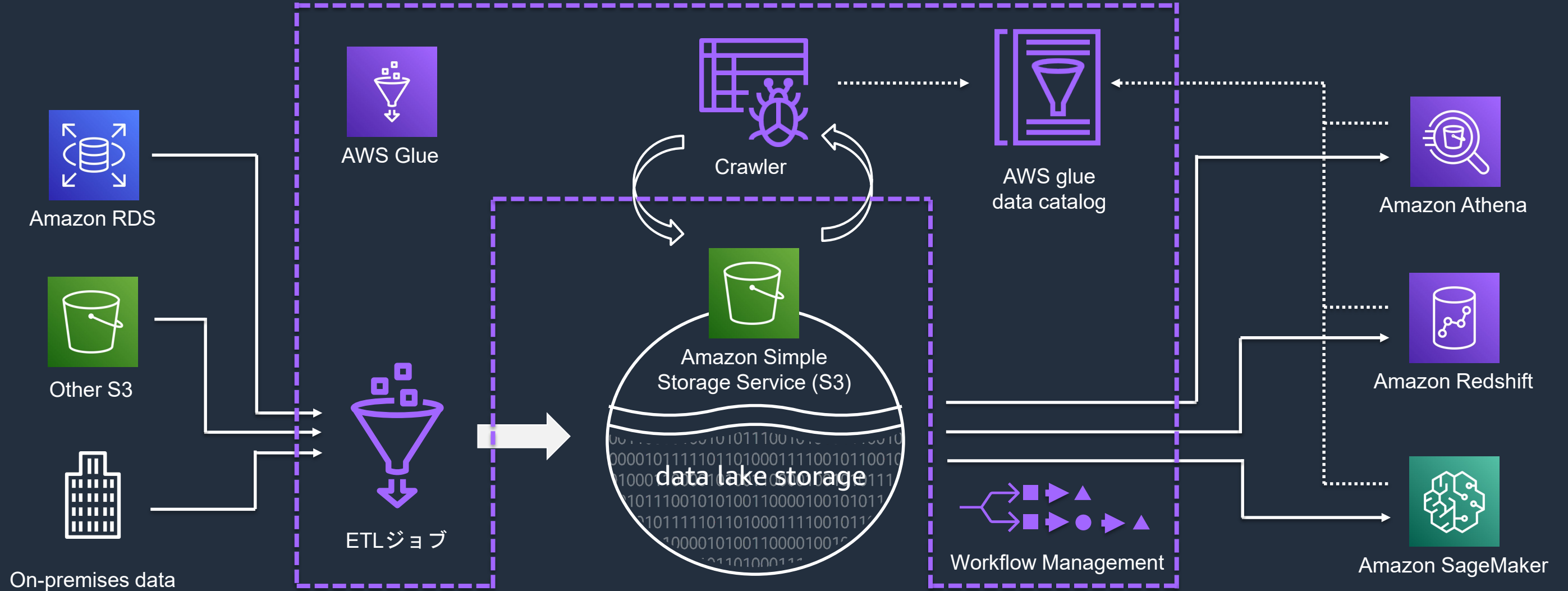
データカタログを参照し、必要データをロード

# カタログを使用したデータ参照



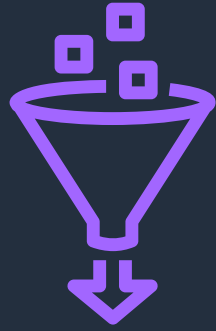
データカタログを参照し、必要データをロード

# ワークフロー管理

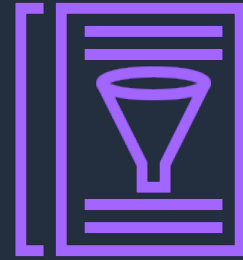


一連の処理をワークフロー化し、自動化可能

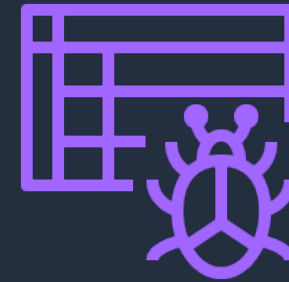
# AWS Glue の構成要素と周辺サービス (再掲)



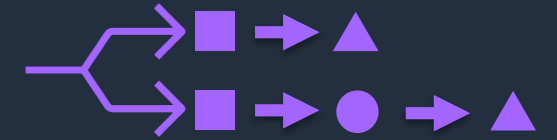
Extract, Transform, and Load (ETL) ジョブ



AWS Glue Data Catalog



Crawler

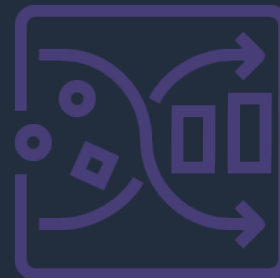


Workflow Management

New



AWS Glue Studio



AWS Glue DataBrew

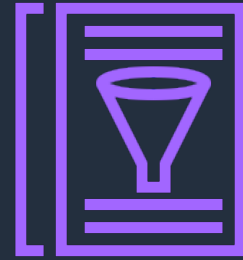


AWS Glue Elastic Views

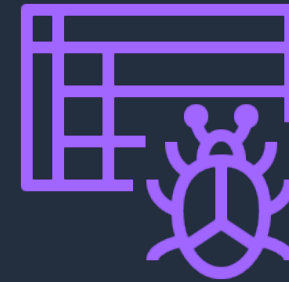
# AWS Glue の構成要素と周辺サービス (再掲)



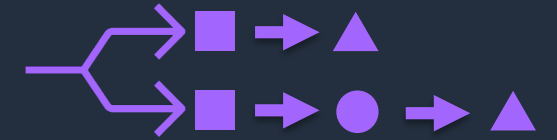
Extract, Transform, and Load (ETL) ジョブ



AWS Glue Data Catalog



Crawler

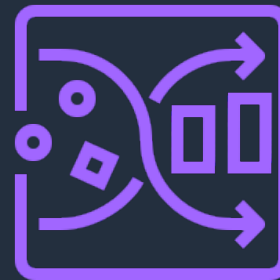


Workflow Management

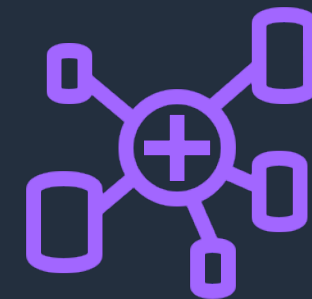
New



AWS Glue Studio



AWS Glue DataBrew



AWS Glue Elastic Views

# AWS Glue の構成要素と周辺サービス (再掲)



Extract, Transform, and Load (ETL) ジョブ



AWS Glue Data Catalog



Crawler



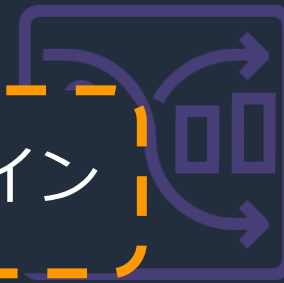
Workflow Management

New



AWS Glue Studio

本日のメイン



AWS Glue DataBrew



AWS Glue Elastic Views

# AWS Glue Studioの概要

# 代表的なデータ利用者

## ビジネスユーザー：

データ分析についての専門的な知見を持ってはいないが、蓄積したデータを活用する。組織で活動するあらゆる人が当てはまる。

## ETLデベロッパー：

自分の開発・運用しているプロダクトのためにデータを活用する。システム障害の原因追求、新機能がどう使われているかの分析するロール。

## データアナリスト：

データ分析自体が職務で、マーケティング施策の効果を売り上げデータをもとに検証する。データによる意思決定の支援を行うロール。

## データサイエンティスト：

より高度な手法を使い、複雑な意思決定の支援を実施する。業務システムに組み込んで使用するための機械学習モデルの開発を行うロール。



# 代表的なデータ利用者

## ビジネスユーザー：

データ分析についての専門的な知見を持ってはいないが、蓄積したデータを活用する。組織で活動するあらゆる人が当てはまる。

## ETLデベロッパー：

自分の開発・運用しているプロダクトのためにデータを活用する。システム障害の原因追求、新機能がどう使われているかの分析するロール。

## データアナリスト：

データ分析自体が職務で、マーケティング施策の効果を売り上げデータをもとに検証する。データによる意思決定の支援を行うロール。

## データサイエンティスト：

より高度な手法を使い、複雑な意思決定の支援を実施する。業務システムに組み込んで使用するための機械学習モデルの開発を行うロール。

# 従来のAWS Glue でのJOB実装

## コードベースのJOB作成インターフェース

ジョブ: kensho-glue-studio-sample-job    アクション ▼    保存    ジョブの実行    ダイアグラムの生成    ⓘ カーソルにテンプレートを挿入 ⓘ    ソース    ターゲット    ターゲット位置    変換    スピゴット    ?    ✕

データベース名 kensho-glue-studio-01  
テーブル名 kensho-glue-studio-jdb

↓

変換の名前 ApplyMapping

↓

変換の名前 SelectFields

↓

変換の名前 ResolveChoice

↓

データベース名 kensho-glue-studio-01  
テーブル名 kensho-glue-studio-bq

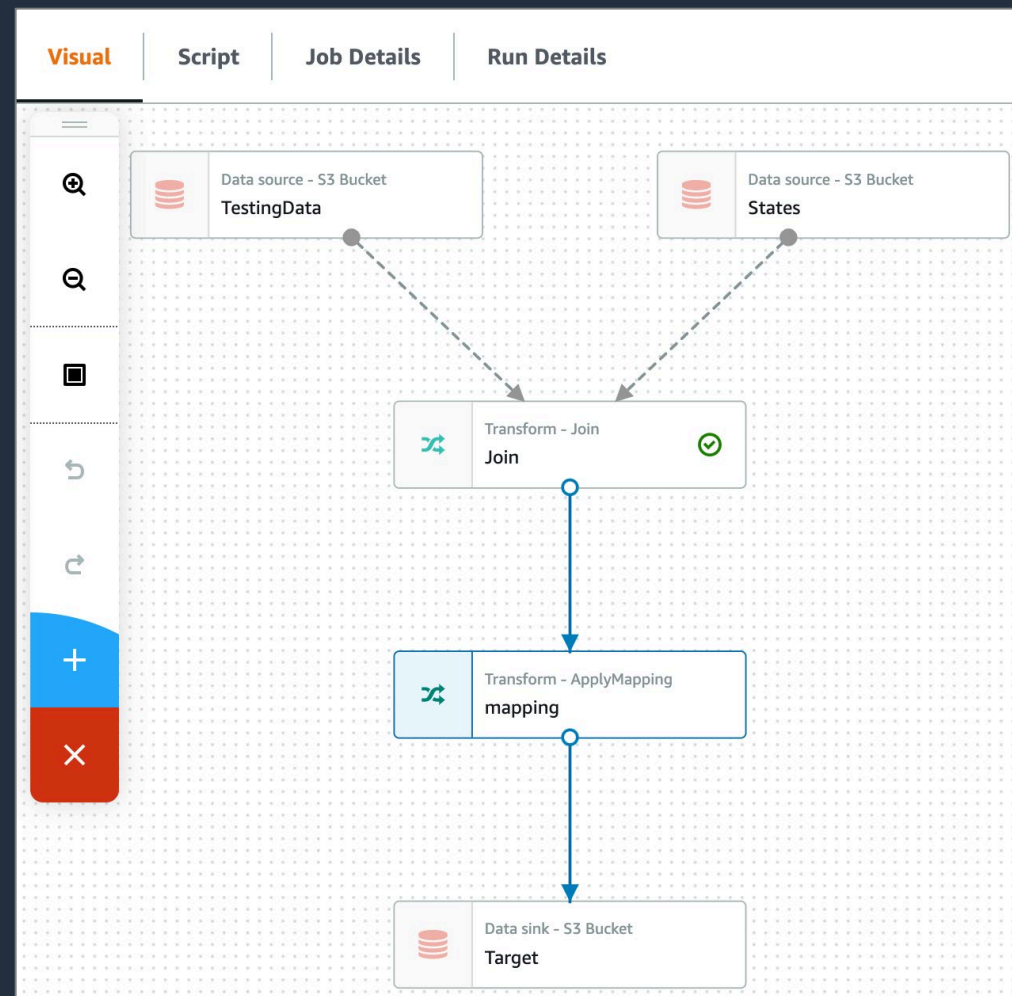
```
1 |import sys
2 |from awsglue.transforms import *
3 |from awsglue.utils import getResolvedOptions
4 |from pyspark.context import SparkContext
5 |from awsglue.context import GlueContext
6 |from awsglue.job import Job
7
8 |## @params: [JOB_NAME]
9 |args = getResolvedOptions(sys.argv, ['JOB_NAME'])
10
11 |sc = SparkContext()
12 |glueContext = GlueContext(sc)
13 |spark = glueContext.spark_session
14 |job = Job(glueContext)
15 |job.init(args['JOB_NAME'], args)
16 |## @type: DataSource
17 |## @args: [database = "kensho-glue-studio-01", table_name = "kensho-glue-studio-jdbc-01", transformation_ctx = "datasource0"]
18 |## @return: datasource0
19 |## @inputs: []
20 |datasource0 = glueContext.create_dynamic_frame.from_catalog(database = "kensho-glue-studio-01", table_name = "kensho-glue-studio-jdbc-01", transformation_ctx = "datasource0")
21 |## @type: ApplyMapping
22 |## @args: [mapping = [("location", "string", "product_title", "string"), ("利益", "long", "product_parent", "long"), ("売上日", "string", "product_id", "string")]]
23 |## @return: applymapping1
24 |## @inputs: [frame = datasource0]
25 |applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [("location", "string", "product_title", "string"), ("利益", "long", "product_parent", "long")])
26 |## @type: SelectFields
27 |## @args: [paths = ["marketplace", "customer_id", "review_id", "product_id", "product_parent", "product_title", "product_category", "star_rating", "helpful_votes"]]
28 |## @return: selectfields2
29 |## @inputs: [frame = applymapping1]
30 |selectfields2 = SelectFields.apply(frame = applymapping1, paths = ["marketplace", "customer_id", "review_id", "product_id", "product_parent", "product_title", "product_category", "star_rating", "helpful_votes"])
31 |## @type: ResolveChoice
32 |## @args: [choice = "MATCH_CATALOG", database = "kensho-glue-studio-01", table_name = "kensho-glue-studio-bq", transformation_ctx = "resolvechoice3"]
33 |## @return: resolvechoice3
34 |## @inputs: [frame = selectfields2]
35 |resolvechoice3 = ResolveChoice.apply(frame = selectfields2, choice = "MATCH_CATALOG", database = "kensho-glue-studio-01", table_name = "kensho-glue-studio-bq", transformation_ctx = "resolvechoice3")
36 |## @type: DataSink
37 |## @args: [database = "kensho-glue-studio-01", table_name = "kensho-glue-studio-bq", transformation_ctx = "datasink4"]
38 |## @return: datasink4
```

ログ    スキーマ

# AWS Glue Studio

NEW

ETL ジョブの作成、実行、監視を容易にする視覚的なインターフェース



ビジュアルオーサリング

コードを書くことなくETLジョブを作成可能

サーバーレス

ビッグデータ処理能力を利用可能

シンプルビュー

単一画面でジョブを管理可能

カスタマイズ可能

任意のコード(Python、Scala、Java)で ETL を高度化可能

<https://aws.amazon.com/jp/blogs/big-data/making-etl-easier-with-aws-glue-studio/>

# シンプルビューの詳細

NEW

The screenshot displays the AWS Glue Studio interface. On the left, the 'Jobs' sidebar shows a list of jobs with 'YYZ-Tickets-Job' selected and highlighted by an orange box. An orange arrow points from this box to the main workspace. The main workspace shows a workflow diagram with nodes: 'Data source - S3 bucket Tickets', 'Data source - S3 bucket Trials', 'Transform - ApplyMapping Ticket\_Mapping', 'Transform - ApplyMapping Trial\_Mapping', 'Transform - Join Join\_Ticket\_Trial' (highlighted with an orange box), 'Transform - Custom code Aggregate\_Tickets', 'Transform - SelectFromCol... Select\_Aggregated\_D...', and 'Data target - S3 bucket S3 bucket'. On the right, the 'Node properties' panel is open for the 'Join\_Ticket\_Trial' node, showing 'Join type' set to 'Inner join' and 'Join conditions' with 'ticket\_number' from 'Ticket\_Mapping' equal to 'parking\_ticket\_number' from 'Trial\_Mapping'. An orange arrow points from the 'Join\_Ticket\_Trial' node in the diagram to the 'Join conditions' section in the properties panel.

<https://aws.amazon.com/jp/blogs/big-data/making-etl-easier-with-aws-glue-studio/>

# カスタマイズを可能にするノード

NEW

The screenshot displays the AWS Glue Studio interface for a job named "YYZ-Tickets-Job". The workflow is visualized as follows:

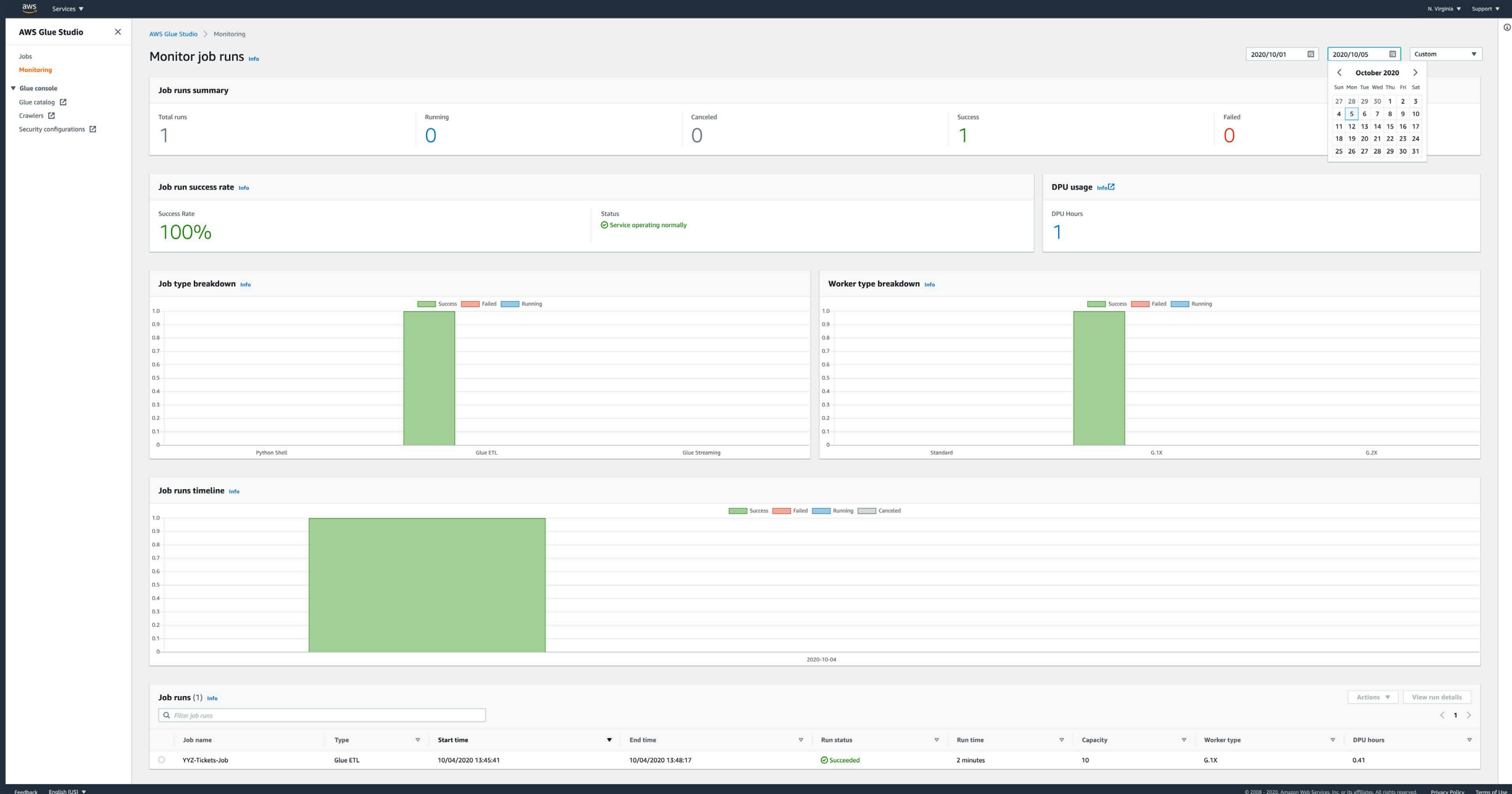
- Two data sources: "Data source - S3 bucket Tickets" and "Data source - S3 bucket Trials".
- Two transform nodes: "Transform - ApplyMapping Ticket\_Mapping" and "Transform - ApplyMapping Trial\_Mapping".
- A join node: "Transform - Join Join\_Ticket\_Trial".
- A custom code node: "Transform - Custom code Aggregate\_Tickets" (highlighted with an orange border).
- A select node: "Transform - SelectFromCol... Select\_Aggregated\_D...".
- A data target: "Data target - S3 bucket S3 bucket".

The "Aggregate\_Tickets" node is selected, and its properties are shown in the right-hand pane:

- Node properties: Transform, Output schema.
- Code block: `def Aggregate_Tickets (glueContext, dfc) -> DynamicFrameCollection: selected = dfc.select(list(dfc.keys())[0]).toDF() selected.createOrReplaceTempView("ticketcount") totals = spark.sql("select court_location as location, infraction_description as results = DynamicFrame.fromDF(totals, glueContext, "results") return DynamicFrameCollection({"results": results}, glueContext)`

# 視覚的なJOBモニタリング

NEW



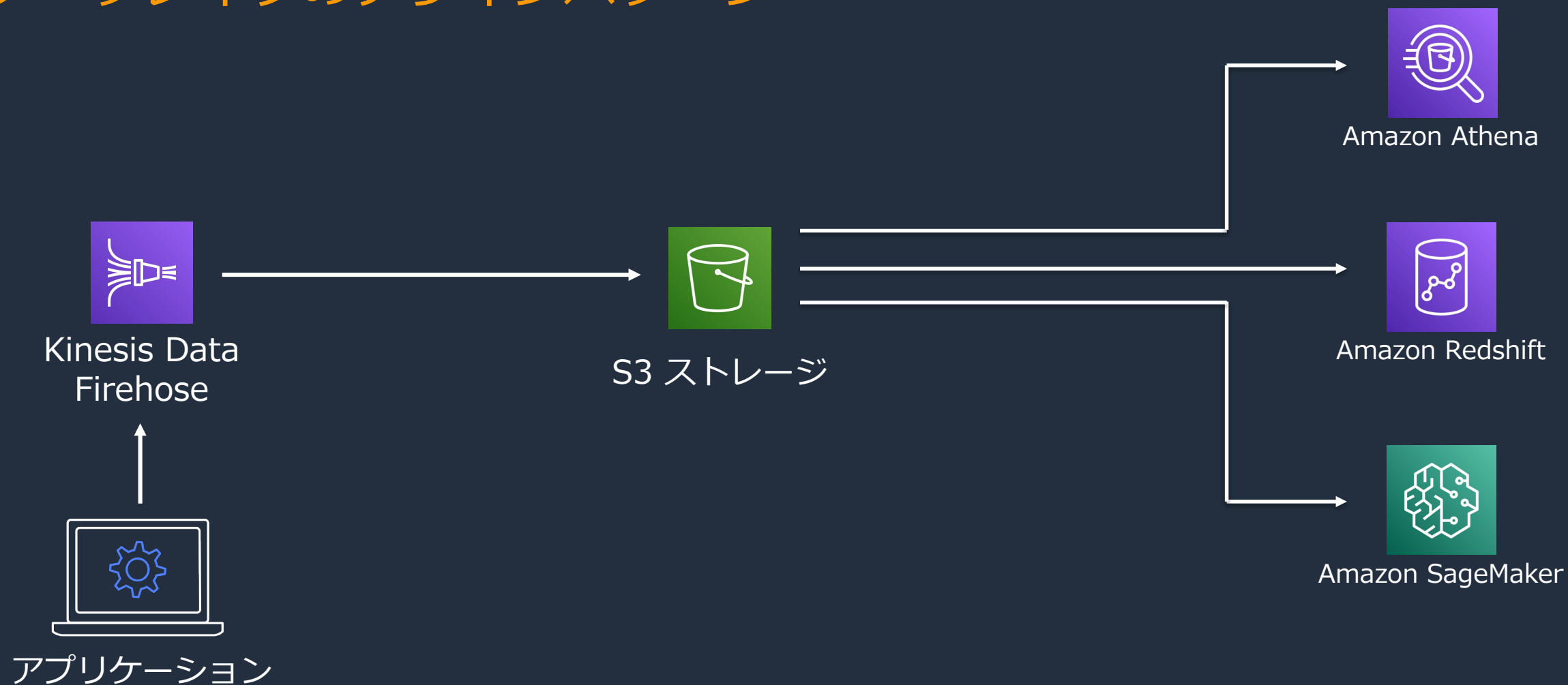
# ユースケースとGlue Studioでの実装

# ユースケース 1： ログストリーム/IoTセンサーデータの分析



# ログストリーム / IoTセンサーデータ分析の課題

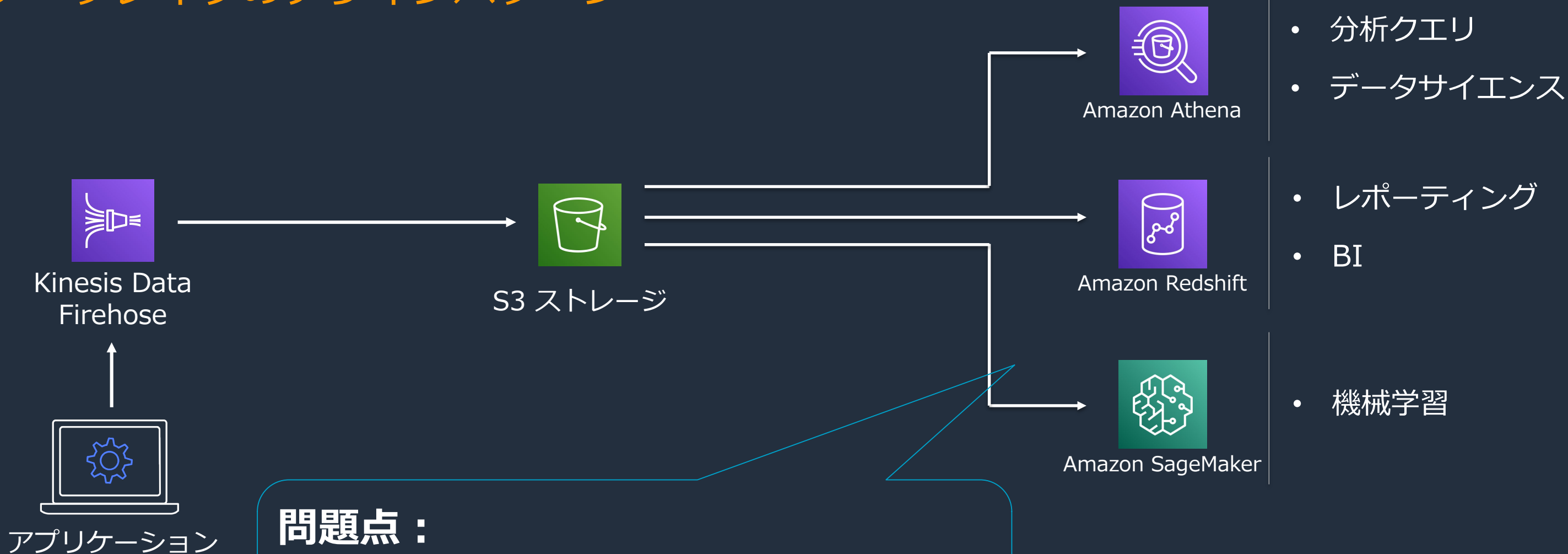
リアルタイムで発生するデータを分析するための  
データレイクのデザインパターン



- 分析クエリ
- データサイエンス
- レポートティング
- BI
- 機械学習

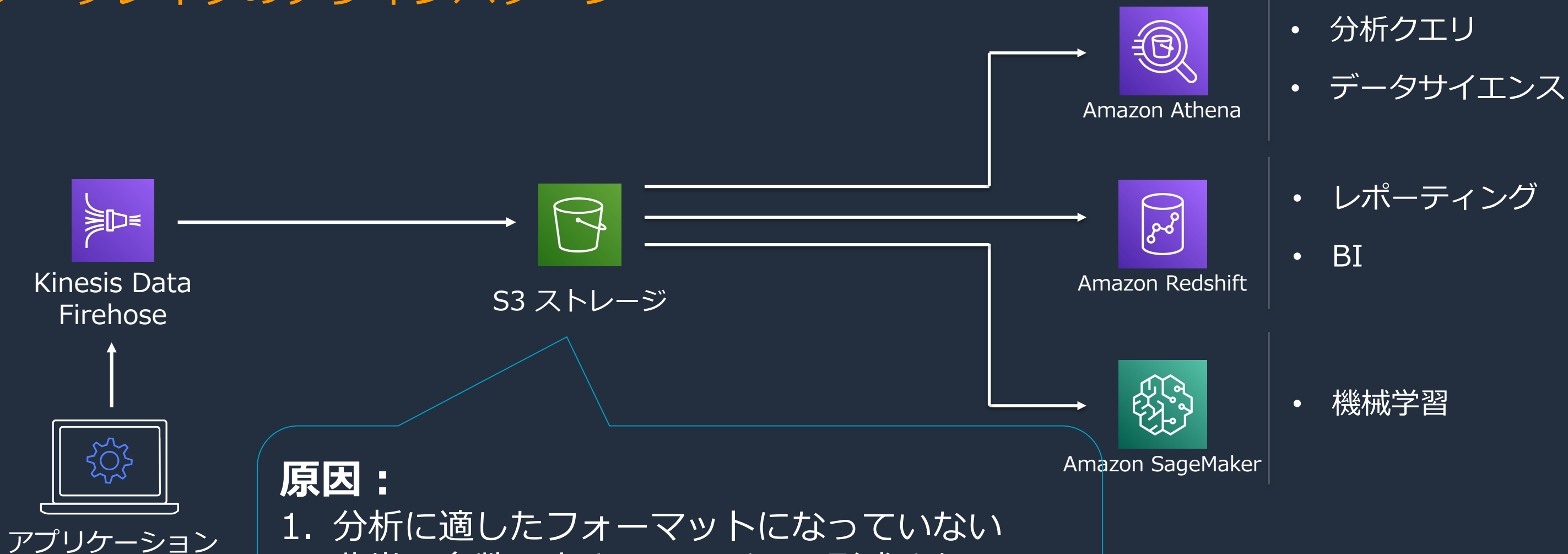
# ログストリーム / IoTセンサーデータ分析の課題

リアルタイムで発生するデータを分析するための  
データレイクのデザインパターン





# ログストリーム / IoTセンサーデータ分析の課題

リアルタイムで発生するデータを分析するための  
データレイクのデザインパターン



# 具体的なデータイメージ

細かい大量のファイルが定期的に出力

名前	▲	タイプ ▼	最終更新日時	▼	サイズ ▼	ストレージクラス
 <a href="#">kensho-glue-studio-01-1-2021-03-02-12-00-31-a12ae31b-c9d5-4fce-bc34-659908b2b585</a>		-	2021/03/02 09:01:33 PM JST		20.3 KB	スタンダード
 <a href="#">kensho-glue-studio-01-1-2021-03-02-12-01-34-bdb3957f-ee56-4516-acc0-05c1fbd44bab</a>		-	2021/03/02 09:02:36 PM JST		19.4 KB	スタンダード
 <a href="#">kensho-glue-studio-01-1-2021-03-02-12-02-35-1978af44-5535-4670-8f29-7807b07017de</a>		-	2021/03/02 09:03:37 PM JST		19.5 KB	スタンダード
 <a href="#">kensho-glue-studio-01-1-2021-03-02-12-03-37-0730996d-edf3-425b-85e0-c4e1b1800836</a>		-	2021/03/02 09:04:38 PM JST		19.3 KB	スタンダード
 <a href="#">kensho-glue-studio-01-1-2021-03-02-12-04-38-68b18a5e-6208-4294-832e-7cb700db41b9</a>		-	2021/03/02 09:05:40 PM JST		20.5 KB	スタンダード
 <a href="#">kensho-glue-studio-01-1-2021-03-02-12-05-40-15c8353e-655a-4cf2-b1e2-cb2df0c863e6</a>		-	2021/03/02 09:06:41 PM JST		18.1 KB	スタンダード
 <a href="#">kensho-glue-studio-01-1-2021-03-02-12-06-40-050373b7-f719-4d74-9c7e-b07bb2a81f14</a>		-	2021/03/02 09:07:43 PM JST		20.4 KB	スタンダード
 <a href="#">kensho-glue-studio-01-1-2021-03-02-12-07-42-876ccbae-9688-483c-9f5e-6a0d718dc92d</a>		-	2021/03/02 09:08:43 PM JST		18.2 KB	スタンダード
 <a href="#">kensho-glue-studio-01-1-2021-03-02-12-08-42-dfe50012-79f0-4bbf-9496-988212cfdb8e</a>		-	2021/03/02 09:09:45 PM JST		19.4 KB	スタンダード
 <a href="#">kensho-glue-studio-01-1-2021-03-02-12-09-43-03d4964f-93fd-443c-8c60-168664b38be2</a>		-	2021/03/02 09:10:45 PM JST		19.5 KB	スタンダード
 <a href="#">kensho-glue-studio-01-1-2021-03-02-12-10-45-507353e9-0dc3-4173-8bc5-9ed619046633</a>		-	2021/03/02 09:11:46 PM JST		19.3 KB	スタンダード
 <a href="#">kensho-glue-studio-01-1-2021-03-02-12-11-46-6d46bb54-37b0-4800-8dd8-a395a2ca07f4</a>		-	2021/03/02 09:12:47 PM JST		19.5 KB	スタンダード
 <a href="#">kensho-glue-studio-01-1-2021-03-02-12-12-47-3128179f-24a5-4282-9925-96623035a1ec</a>		-	2021/03/02 09:13:48 PM JST		19.3 KB	スタンダード

中身はJSON形式のテキストファイルかつ、様々なシステムからのアラートが混在

```
{"timestamp":"02/Mar/2021:12:00:29 +0000","alarmlevel":"WARNING","host":"prd-web","user":"SystemA","number":"1001","text":"This is WARNING"}
{"timestamp":"02/Mar/2021:12:00:29 +0000","alarmlevel":"WARNING","host":"prd-db","user":"SystemD","number":"1001","text":"This is WARNING"}
{"timestamp":"02/Mar/2021:12:00:29 +0000","alarmlevel":"WARNING","host":"prd-db","user":"SystemF","number":"1001","text":"This is WARNING"}
{"timestamp":"02/Mar/2021:12:00:29 +0000","alarmlevel":"WARNING","host":"prd-ap","user":"SystemA","number":"1001","text":"This is WARNING"}
```

# このデータに対して求められること

## 解決策：

1. フォーマットを列指向フォーマットに変換する
2. 大量ファイルをコンパクションする
3. 適切な単位でパーティショニングする

上記の変更を加えることで、より効率的な分析が可能

# 列指向フォーマットとは

カラム（列単位）でデータをまとめて保存するデータフォーマット

## メリット1) OLAP 系の分析クエリを効率的に実行できる

- 多くの分析クエリは、一度のクエリで一部のカラムしか使用しない

行指向

1	2	3	4	5	6

列指向

1	2	3	4	5	6

## メリット2) I/O の効率があがる

- 圧縮と同時に使うことで I/O 効率がさらに向上
- カラムごとに分けられてデータが並んでいる
- 同じカラムは、似たような中身のデータが続くため、圧縮効率がよくなる

行指向

1	2	3	4	5	6

列指向

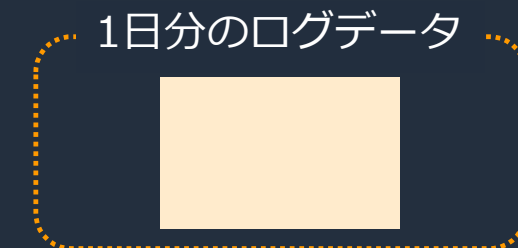
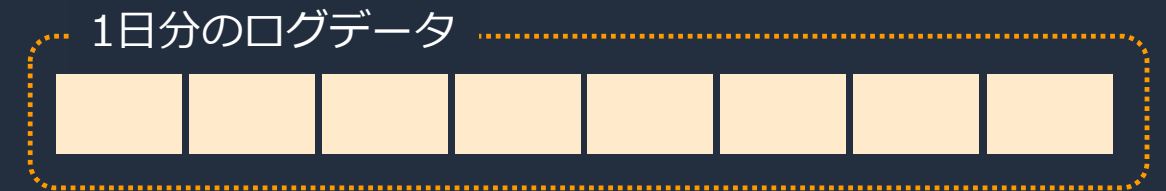
1	2	3	4	5	6

# コンパクションとは

サイズの小さい複数ファイルを分析のしやすい単位に集約

## ファイル数が多い場合の問題点

- Hadoop系の分散エンジンでは、読み込む必要のあるブロックごとにタスクが生成される。その為、データが少ないブロックへの処理ではオーバーヘッドが発生しパフォーマンスが低下する。



## コンパクション後のメリット

- データ分析前にファイルをひとまとめにしておくことで、オーバーヘッドが減少し分析時のパフォーマンスが向上する。

# パーティショニングとは

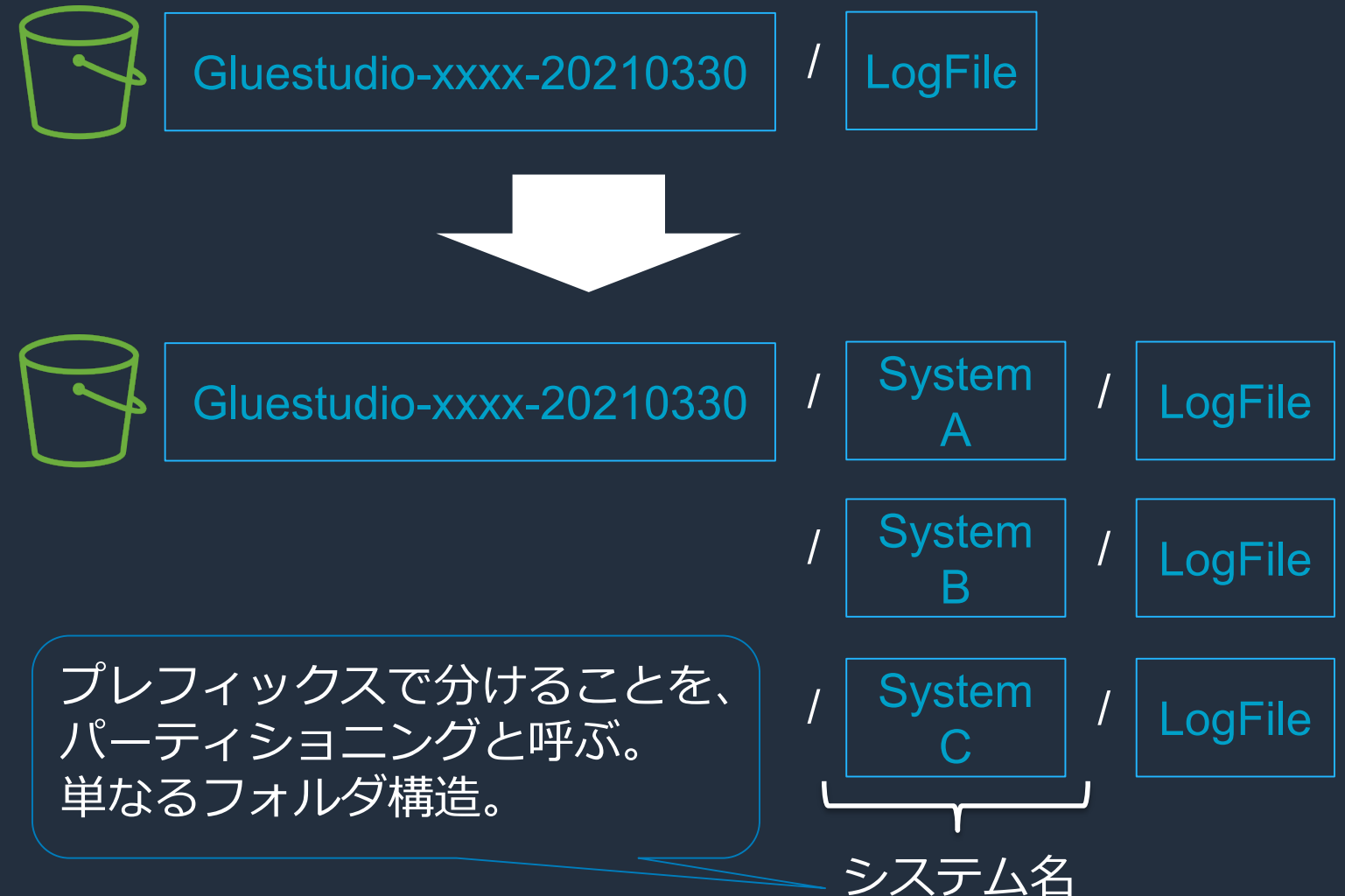
分析によく使う単位でファイル进行分析して格納

## パーティショニングをしない場合の問題点

- 特定のシステムのログを検索したい場合、全てのLogFileに対してスキャンが掛かる。その為、読み込むデータ量が多くなり、パフォーマンスに影響がある。

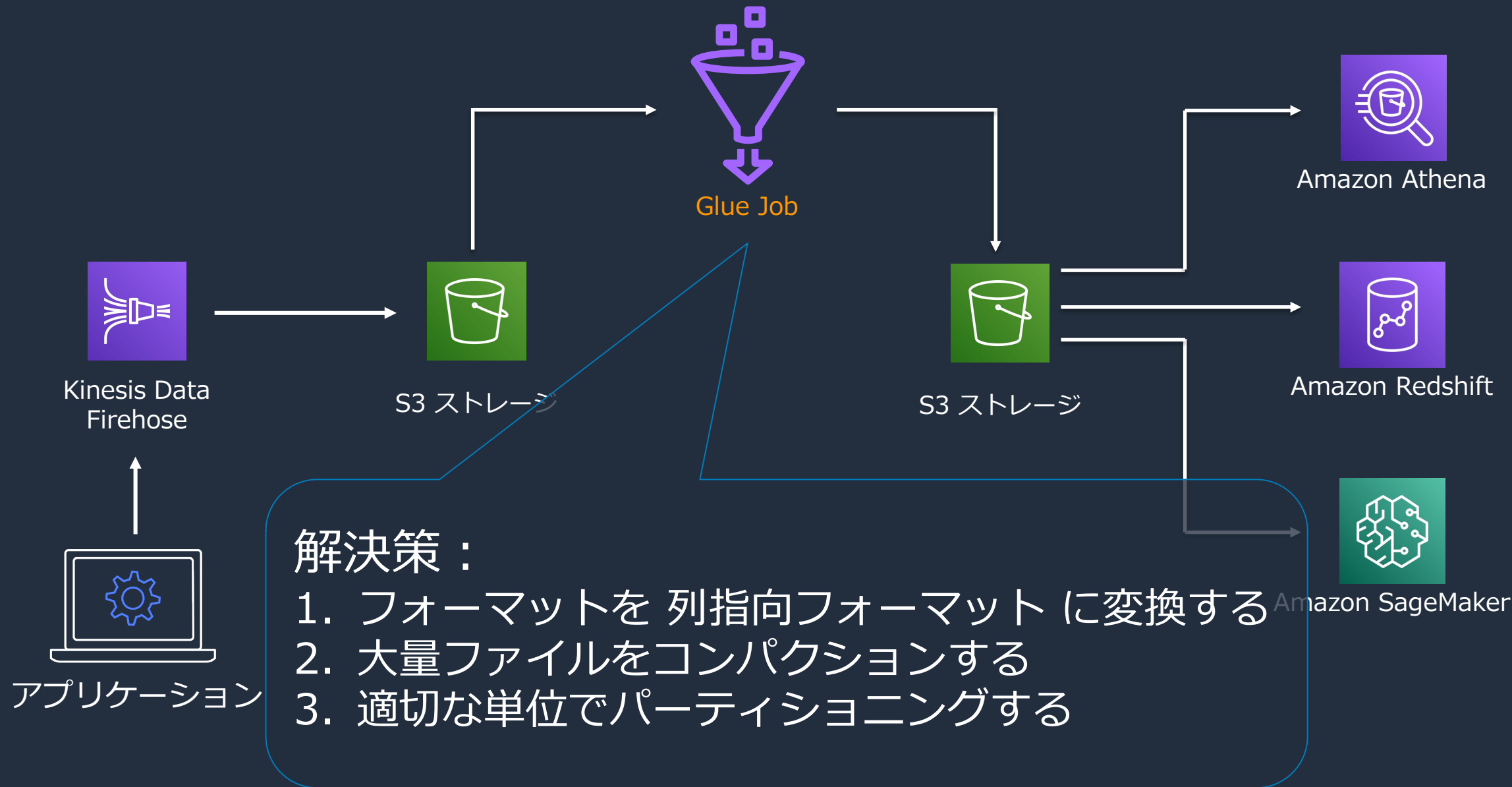
## パーティショニング後のメリット

- 検索条件にシステム名を含めることで、特定のファイルにのみアクセスする。不要データへのアクセスが減る為、パフォーマンスが向上する。





# ログストリーム / IoTセンサーデータ分析の解決策



- 分析クエリ
- データサイエンス
- レポーティング
- BI
- 機械学習

# AWS Glue Studioを使用したETL JOBの作成

- マネージメントコンソールより、AWS Glue を選択
- AWS Glue コンソールの左ペインより、AWS Glue Studioを選択
- 表示されたAWS Glue Studioの画面にて Create and manage Jobs を選択




Analytics

## AWS Glue Studio


Visually create job flows and monitor their performance

AWS Glue Studio is an easy-to-use graphical interface for creating, running, and monitoring AWS Glue extract, transform, and load (ETL) jobs.

Getting started



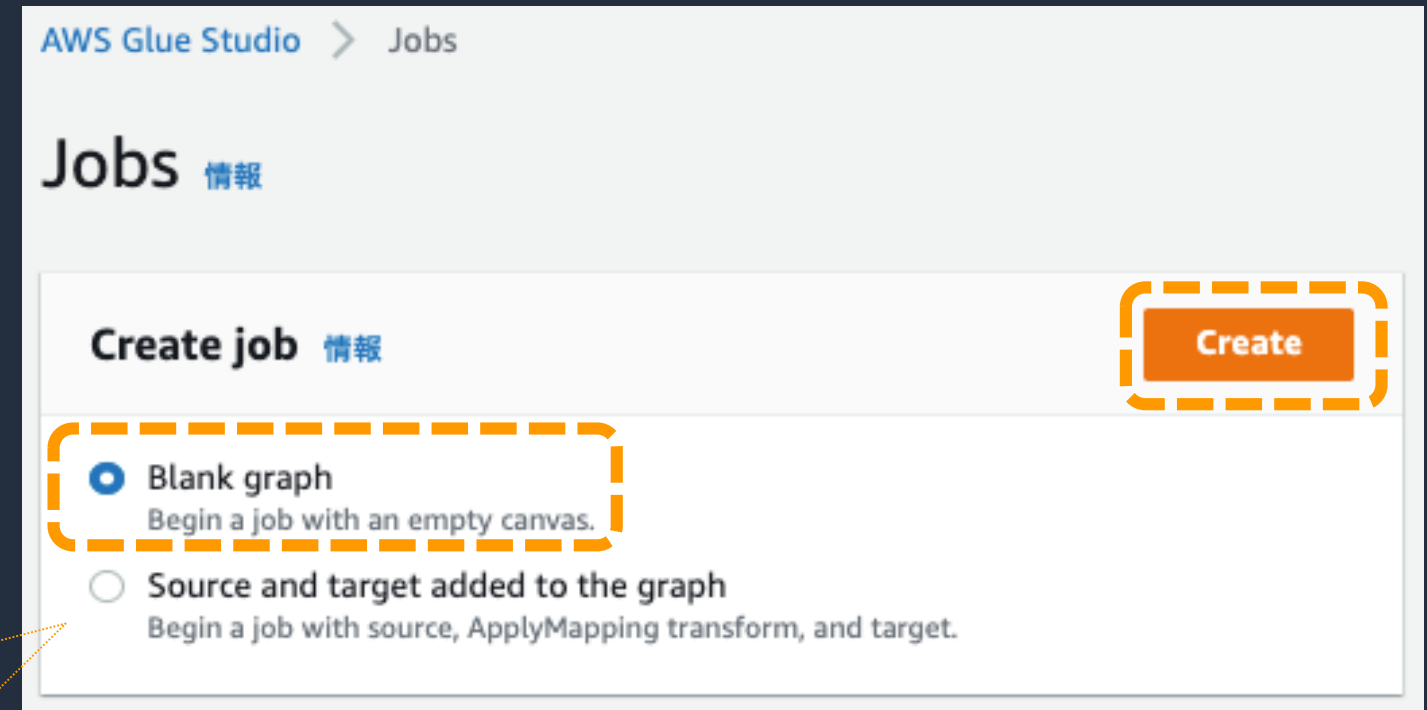
**Create and manage jobs**  
Visually author, run, view, and edit your AWS Glue jobs.



**Monitor job runs**  
Diagnose, debug, and check the status of your AWS Glue jobs.

# JOBの作成

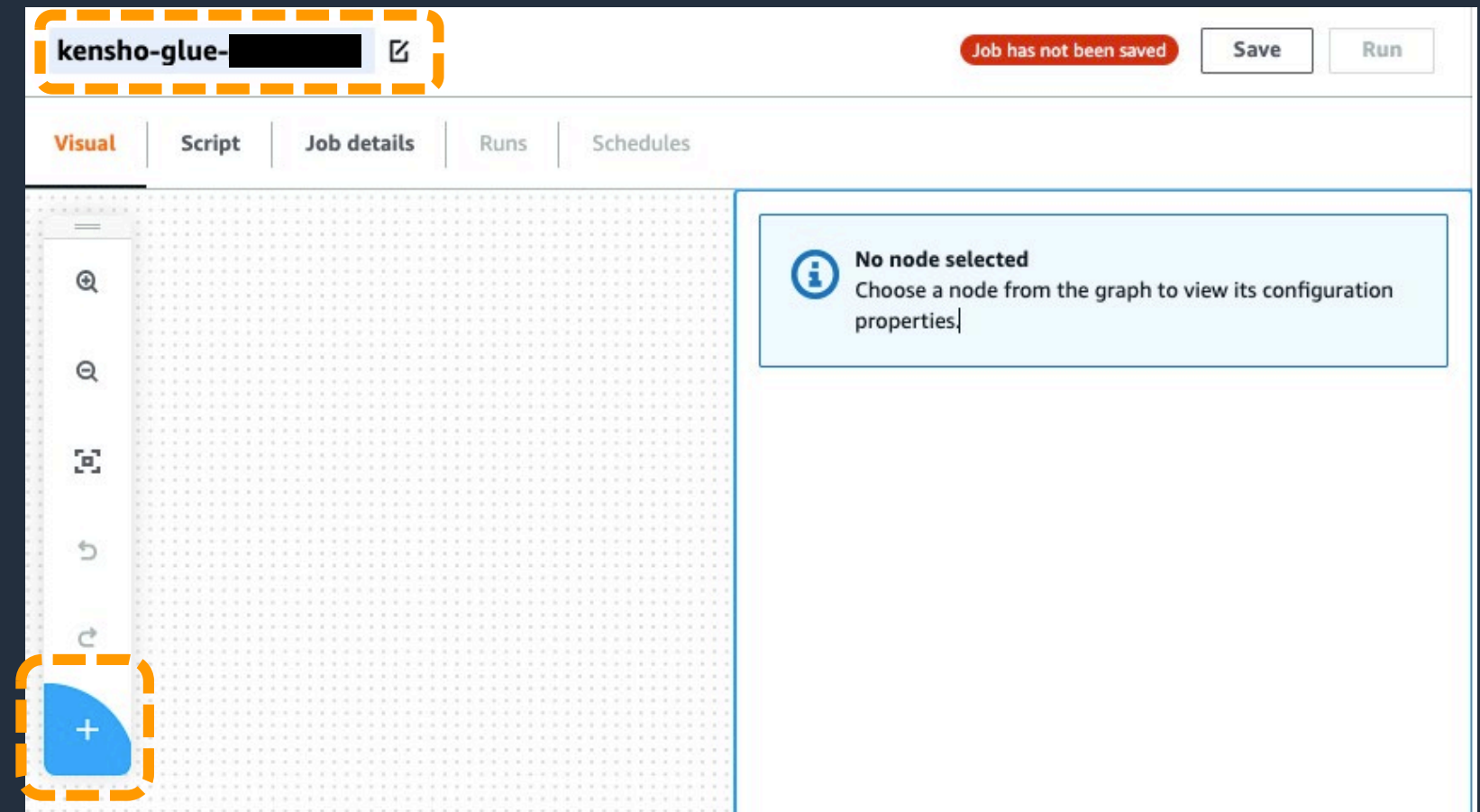
- Create job の項目にて、Blank graphを選択しCreateボタンを押下



Source and target added to the graph  
指定したデータソースとターゲットノードがあらかじめ作成された状態から開始する。どちらを選んでも問題ない。

# データソースの指定①

- JobのEdit画面が表示
- 左上にJob名を入力
- 画面左下の“+”を押下し、ノードを表示



# データソースの指定②

- Node typeにS3を選択

選択可能なデータソースは、Kinesis, Kafka, JDBC, Redshift 等々。  
custom connectorを使用することで追加可能。

The screenshot shows the AWS Glue console interface for a job named 'kensho-glue-'. The 'Visual' tab is active, displaying a workflow canvas with a 'Data source - S3 bucket' node. The right-hand panel is open to 'Data source properties - S3'. The 'Node type' dropdown is set to 'S3', and the 'Data source' dropdown is also set to 'S3'. A dashed orange box highlights the 'Node type' and 'Data source' sections. The 'Output schema' section is empty. The top right of the console shows a 'Job has not been saved' warning and 'Save' and 'Run' buttons. The bottom navigation bar includes a search icon, a plus sign, and a minus sign.

# データソースの指定③

- Data Catalog tableを選択
- DatabaseとTableに、Glue のクローラによって生成されたデータカタログを選択

S3 locationを選択し、バケットを直接指定してもOK

The screenshot shows the AWS Glue console interface for configuring a job. The job name is 'kensho-glue-...'. The 'Visual' tab is selected. The 'Data source properties - S3' tab is active, showing the following configuration:

- S3 source type:** Data Catalog table (selected), S3 location (unselected). The S3 location option includes the instruction: "Choose a file or folder in an S3 bucket."
- Database:** kensho-glue-... (selected)
- Table:** applog (selected)
- Partition predicate - optional:** Enter a Boolean expression supported by Spark SQL, using only partition columns.

A callout box points to the 'S3 location' option, indicating that selecting an S3 location and specifying the bucket directly is also acceptable.

# データの整形①

- S3 bucket ノードを選択した状態で、画面左下の“+”を押下
- Node typeにApplyMappingを選択

ApplyMappingは、入力データの列の修正・削除を行うためのノード。

他にもフィールドの分割や、ソースデータの結合等、様々な変換ノードが用意されている。

The screenshot displays the AWS Glue console interface for a job named 'kensho-glue-'. The 'Visual' tab is active, showing a workflow with two nodes: 'Data source - S3 bucket' and 'Transform - ApplyMapping'. A red dashed box highlights the '+' button in the bottom-left corner of the visual editor. A red dashed box also highlights the 'Node properties' panel on the right, which is currently showing the 'ApplyMapping' node type selected in the 'Node type' dropdown. The 'Node properties' panel includes fields for 'Name' (ApplyMapping), 'Node type' (ApplyMapping), and a search bar. Below the search bar, the 'AWS Glue Data Catalog' table is visible. The 'Transform' section shows the 'ApplyMapping' node selected, with a description: 'Map fields to new names and types of your choice.' The 'Output schema' tab is also visible, showing the 'SelectFields' node type.

# データの整形②

- Target key列に入力されている文字列を修正
- Drop列のボックスにチェック

ここでは以下の2点を実施している。  
1. user列の名前をsystem列に変更している。  
2. partition\_0列を削除している。

The screenshot shows the AWS Glue console interface for a job named 'kensho-glue-'. The 'Transform' tab is selected, showing a 'Transform - ApplyMapping' node. The 'Apply mapping' table is visible, with the following data:

Source key	Target key	Data type	Drop
timestamp	timestamp	date	<input type="checkbox"/>
alarmlevel	alarmlevel	string	<input type="checkbox"/>
host	host	string	<input type="checkbox"/>
user	system	string	<input type="checkbox"/>
number	number	string	<input type="checkbox"/>
text	text	string	<input type="checkbox"/>
partition_0			<input checked="" type="checkbox"/>



# データファイルのコンパクション①

- ApplyMapping ノードを選択した状態で、画面左下の“+”を押下
- Node typeにCustom transformを選択

Custom transformは、コードを自由に記述することが可能なノード。  
Python, Scala, Javaの、任意のコードを使用可能。

The screenshot displays the AWS Glue console interface for a job named 'kensho-glue-'. The 'Visual' tab is active, showing a workflow with two nodes: 'Transform - ApplyMapping' and 'Transform - Custom code'. The 'Custom code' node is selected, and its configuration is shown in the right-hand pane. The 'Node type' is set to 'Custom transform'. The 'Node parents' dropdown is set to 'ApplyMapping'. The 'ApplyMapping' node is highlighted in the 'Node parents' list. The 'Node properties' pane also shows the 'Name' as 'Custom Transform'.

# データファイルのコンパクション②

- コードブロックに下記記載の、Pythonコードを入力

```
selected = dfc.select(list(dfc.keys())[0]).toDF()
reprep = selected.repartition(5)
results = DynamicFrame.fromDF(reprep, glueContext, "results")
return DynamicFrameCollection({"results": results}, glueContext)
```



このコードブロックは、DynamicFrameオブジェクトを受け取っている。受け取ったオブジェクトに対し任意の処理を記述可能。

The screenshot shows the AWS Glue console interface. At the top, there are three tabs: 'Node properties', 'Transform', and 'Output schema'. The 'Transform' tab is selected. Below the tabs, there is a 'Code block' section with a sub-tab '情報' (Info). The text 'Enter a custom script to add to your job.' is displayed. The code editor shows a Python function definition:

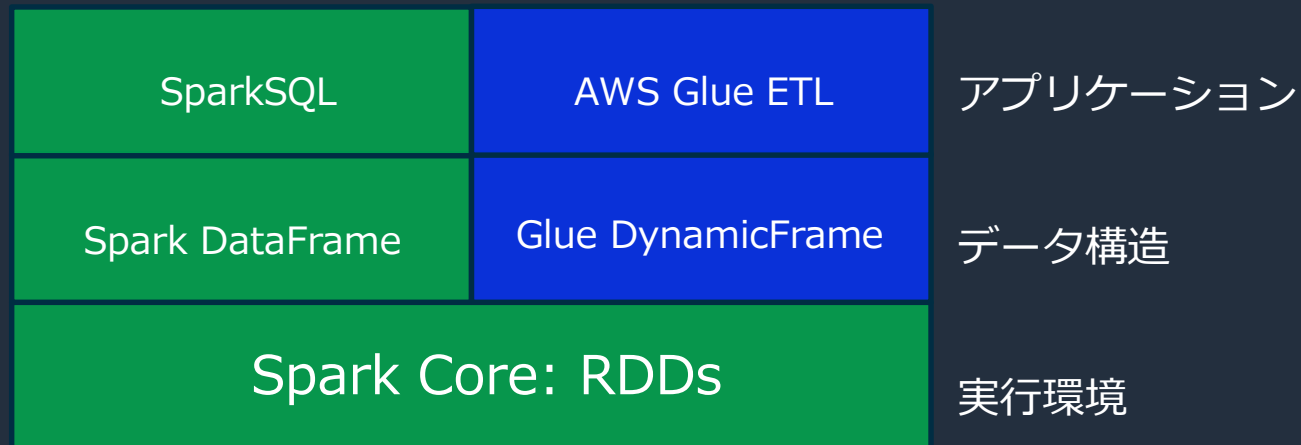
```
1 def MyTransform (glueContext, dfc) -> DynamicFrameCollection:
2
3
4
5
6 |
7
```

# DynamicFrameとは

## SparkSQL DataFrameと似たGlue特有の抽象化の概念

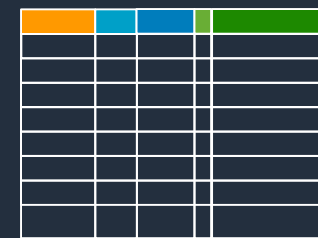
- SparkSQL DataFrameとの違いはETLに特化しているかどうか  
(DynamicFrameはスキーマの不一致を明示的にエンコードする“Schema on the Fly”を採用)
- 複数の型の可能性を残して、後で決定できるようにする (Choice型)
- DynamicFrameはデータ全体を表し、DynamicRecordはデータ 1 行を指す
- DataFrameとDynamicFrame間でそれぞれ変換することができる(fromDF関数・toDF関数)
- Pythonライブラリ PandasのDataFrameとは異なるので注意

### アーキテクチャ : SparkおよびGlueライブラリ



### データ構造イメージ

#### SparkSQL DataFrame



構造テーブルに類似

#### DynamicFrame



半構造テーブルに類似

# コードの内容

Node properties

**Transform**

Output schema



Code block [情報](#)

Enter a custom script to add to your job.

```
1 def MyTransform (glueContext, dfc) -> DynamicFrameCollection:  
2     selected = dfc.select(list(dfc.keys())[0]).toDF()  
3     reprep = selected.repartition(5)  
4     results = DynamicFrame.fromDF(reprep, glueContext, "results")  
5     return DynamicFrameCollection({"results": results}, glueContext)  
6
```

上記のコードは受け取ったDynamicFrameオブジェクトに以下の内容を実施している。

1. 受け取ったDynamicFrameオブジェクトを、toDF()関数を使用し、SparkSQL DataFrameに変換
2. DataFrameの分割数を変更し、5分割に変更
3. 変更後のDataFrameをfromDF()関数を使用し、DynamicFrameオブジェクトに変換
4. DynamicFrameオブジェクトをリターン

# データファイルのコンパクション③

- Custom Transform ノードを選択した状態で、画面左下の“+”を押下  
(自動生成の場合この手順は不要)
- Node typeにSelectFromCollectionを選択

SelectFromCollectionは、Custom Transform ノードで実行したコードからオブジェクトを受け取るためのノード。  
Custom Transformノードとセットで使用する。

The screenshot shows the AWS Glue console interface. At the top, there's a header with 'kensho-glue-...' and 'Last Saved at 2021/3/3 13:35:12'. Below the header are tabs for 'Visual', 'Script', 'Job details', 'Runs', and 'Schedules'. The main workspace shows a workflow with two nodes: 'Transform - Custom code Custom Transform' and 'Transform - SelectFromCol... SelectFromCollection'. A blue arrow points from the first node to the second. On the left, a vertical toolbar contains a search icon, a plus sign, and a minus sign. On the right, the 'Node properties' panel is open, showing the configuration for the selected 'SelectFromCollection' node. The 'Name' field is 'SelectFromCollection'. The 'Node type' dropdown is set to 'SelectFromCollection'. The 'Node parents' dropdown is set to 'Select parents'. Below these are buttons for 'Custom Transform' and 'CustomCode - Transform'.

# フォーマット変換①

- SelectFromCollection ノードを選択した状態で、画面左下の“+”を押下
- Node typeにS3を選択

Node typeには、Data targetのS3とData sourceのS3があるので注意する。

The screenshot shows the AWS Glue console interface. At the top, the job name is 'kensho-glue-'. The 'Visual' tab is selected, displaying a workflow diagram with two nodes: 'Transform - SelectFromCollection' and 'Data target - S3 bucket'. The 'Node properties' panel on the right is open, showing the 'Node type' dropdown menu with 'S3' selected. A callout box highlights the 'Node type' dropdown.

# フォーマット変換②

- FormatをGlue Parquetを選択
- Compression TypeにSnappyを選択

Parquet とは  
Parquetは分析に適した列指向フォーマット

Glue Parquet とは  
よりGlueに最適化されたParquetフォーマット  
通常のParquetと変わらないが、出力ファイルのスキーマを動的に計算し、高速に“Parquet”ファイルに書き込むことが可能

The screenshot shows the 'Data target properties - S3' configuration page in the AWS Glue console. The 'Format' dropdown is set to 'Glue Parquet' and the 'Compression Type' dropdown is set to 'Snappy'. The 'S3 Target Location' field contains 's3://kensho-glue-...' and the 'Data Catalog update options' section has 'Do not update the Data Catalog' selected. The 'Partition (0)' dropdown is set to 'system'.

# 出力先の指定

- S3 Target Locationに出力先となる、S3バケットを指定
- Data Catalog update optionsにて、Do not update the Data Catalogを選択

## Data Catalog update options について

### Do not change table definition:

ジョブでDataCatalogの更新を行わない設定。

### Update schema and add new partitions :

スキーマの変更または新しいパーティションの追加時にジョブでData Catalogを更新する設定。

### Keep existing schema and add new partitions :

新しいパーティションを追加する目的にのみ、ジョブでDataCatalogを更新する設定。

Node properties | **Data target properties - S3** | Output schema

Format  
Glue Parquet

Compression Type  
Snappy

**S3 Target Location**  
Choose an S3 location in the format s3://bucket/prefix/object/ with a trailing slash (/).  
s3://kensho-glue-.../ View Browse S3

**Data Catalog update options** 情報  
Choose how you want to update the Data Catalog table's schema and partitions.

- Do not update the Data Catalog
- Create a table in the Data Catalog and on subsequent runs, update the schema and add new partitions
- Create a table in the Data Catalog and on subsequent runs, keep existing schema and add new partitions

Partition keys - optional  
Add partition keys.

Partition (0)  
system

Add a partition key

You can add 49 more partition keys.



# パーティショニング

- Partition keys にて、パーティショニングに使用するカラムを選択

Partitionに使用するカラムは複数選択が可能。  
複数選択することで階層構造でファイルが出力される。

The screenshot shows the AWS Glue console configuration for a job's data target properties. The 'Data target properties - S3' tab is active. The 'Format' is set to 'Glue Parquet' and 'Compression Type' is 'Snappy'. The 'S3 Target Location' is 's3://kensho-glue-.../'. Under 'Data Catalog update options', 'Do not update the Data Catalog' is selected. The 'Partition keys - optional' section is highlighted with a dashed orange box. It shows 'Partition (0)' with a dropdown menu containing 'system' and a trash icon. Below the dropdown is an 'Add a partition key' button. At the bottom of the highlighted area, it says 'You can add 49 more partition keys.'

# JOBの詳細設定①

- Job detailsタブを選択
- IAM Role に必要な権限を持ったロールを選択
- Glue version にて、Glue 2.0が  
選択されていることを確認

## 必要な権限について

ユースケース1で必要となる権限は以下。

- AWS Glue サービスのアクセス許可
- Amazon CloudWatch のアクセス許可
- データソースとデータターゲットへのアクセス許可

これらの権限が付与されたロールを選択する。

Visual | Script | **Job details** | Runs | Schedules

### Basic properties 情報

Name  
kensho-glue-██████████

Description - *optional*

Descriptions can be up to 2048 characters long.

**IAM Role**  
Role assumed by the job with permission to access your data stores. Ensure that this role has permission to your Amazon S3 sources, targets, temporary directory, scripts, and any libraries used by the job.

**AWSGlueStudioRole**  
No description available.

**Type**  
The type of ETL job. This is set automatically based on the types of data sources you have selected.

Spark

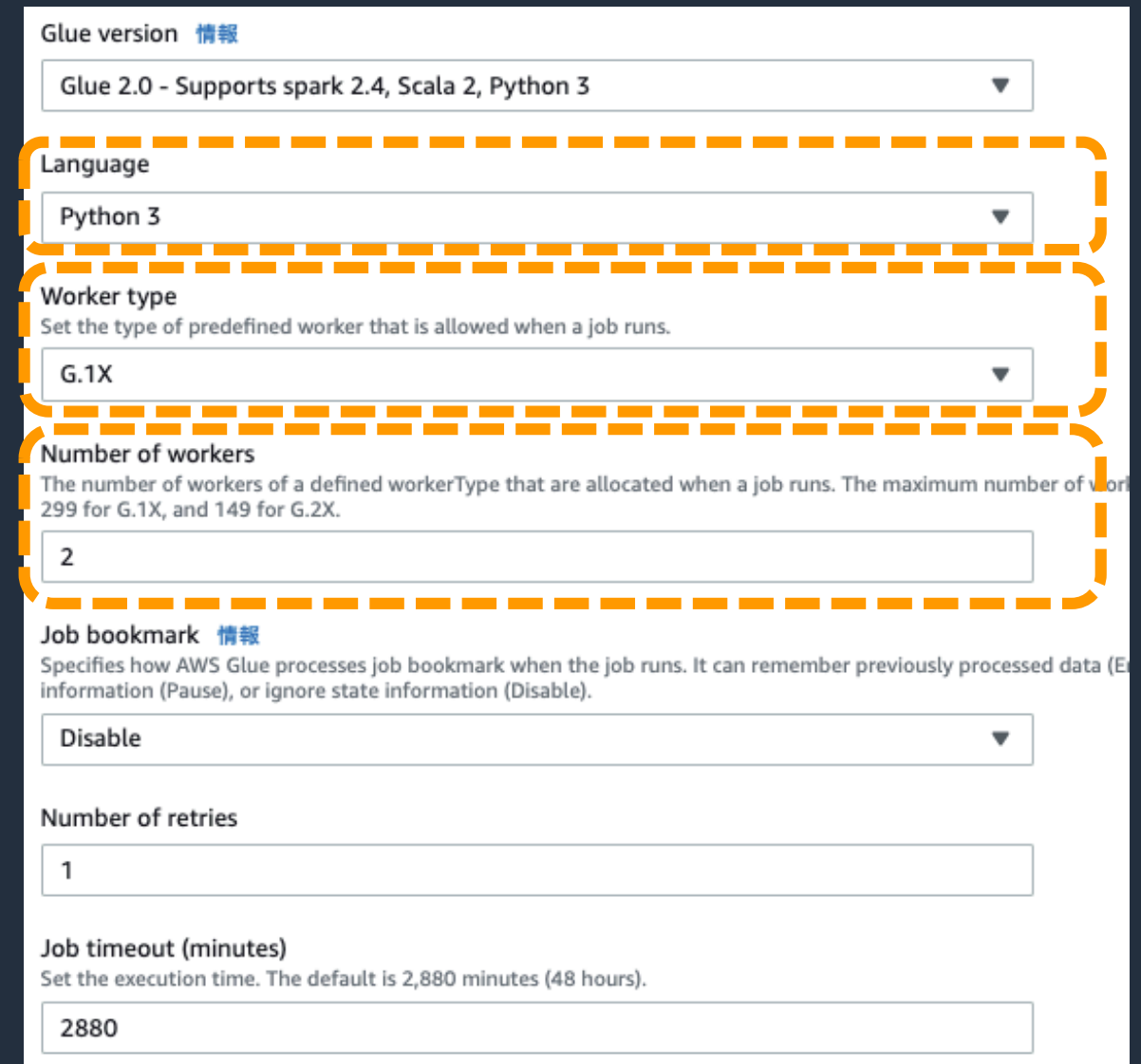
**Glue version 情報**

**Glue 2.0 - Supports spark 2.4, Scala 2, Python 3**

# JOBの詳細設定②

- Language にPython 3が選択されていることを確認
- Worker type にてG.1Xを選択
- Number of workersをデフォルトの10から2に変更

Worker typeの詳細については次のスライドをご参照



Glue version 情報

Glue 2.0 - Supports spark 2.4, Scala 2, Python 3

Language

Python 3

Worker type 情報

Set the type of predefined worker that is allowed when a job runs.

G.1X

Number of workers

The number of workers of a defined workerType that are allocated when a job runs. The maximum number of workers is 299 for G.1X, and 149 for G.2X.

2

Job bookmark 情報

Specifies how AWS Glue processes job bookmark when the job runs. It can remember previously processed data (Enable), ignore state information (Pause), or ignore state information (Disable).

Disable

Number of retries

1

Job timeout (minutes)

Set the execution time. The default is 2,880 minutes (48 hours).

2880

# Worker Type

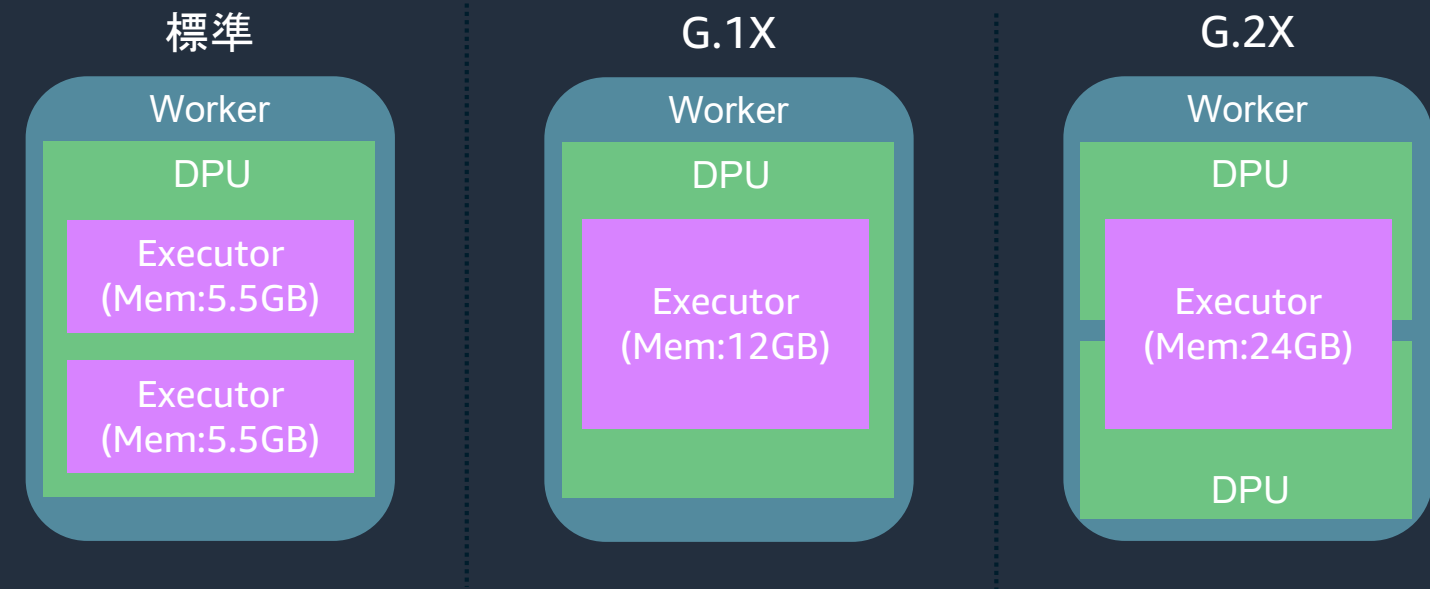
ジョブ実行環境にメモリ大量使用ワークロード向けのWorker Typeを選択可能

- ジョブ実行時に割り当てる処理能力をDPU(Data Processing Unit)という  
1DPU = 4vCPU、16GBメモリ
- Glue Studioでは、G.1xとG.2Xが選択可能

## Worker Type一覧

Worker Type	DPU数 /1Worker	Executor数 /1Worker	メモリ数 /1Executor
標準	1	2	5.5GB
G.1X	1	1	12GB
G.2X	2	1	24GB

## Worker Type構成イメージ



参考URL : Spark Components(<https://spark.apache.org/docs/latest/cluster-overview.html>)

# JOBの詳細設定③

- Job bookmark にてDisableを選択
- Number of retriesをデフォルトの3から1に変更

Job bookmarkについて  
ジョブを実行した状態情報を保持することで、古いデータを再処理しないようにする機能。デフォルトのDisableにすることでJobの実行のたびにデータセット全体に対して処理をすることが可能。

Glue version 情報

Glue 2.0 - Supports spark 2.4, Scala 2, Python 3

Language

Python 3

Worker type

Set the type of predefined worker that is allowed when a job runs.

G.1X

Number of workers

The number of workers of a defined workerType that are allocated when a job runs. The maximum number of workers is 299 for G.1X, and 149 for G.2X.

2

Job bookmark 情報

Specifies how AWS Glue processes job bookmark when the job runs. It can remember previously processed data (Enable), pause state information (Pause), or ignore state information (Disable).

Disable

Number of retries

1

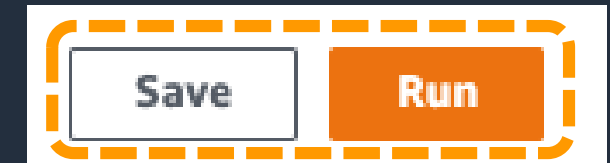
Job timeout (minutes)

Set the execution time. The default is 2,880 minutes (48 hours).

2880

# 結果のモニタリング

- 画面右上のSaveを押下した後、Runボタンを押下しJobを実行
- Runs タブを選択
- Run statusが、Succeededになったことを確認



A screenshot of the AWS Glue console interface for a job named 'kensho-glue'. The 'Runs' tab is selected. The page shows 'Recent job runs (1) 情報'. A specific run is detailed for '2021年3月03日 13:42'. The 'Run status' is 'Succeeded', which is highlighted with a dashed orange box. Other details include 'Id: jr\_cee0779b463611c5f4b556a9f0de8c8e5f33e8b00ac5b89f9b4e8627bb53e07f', 'Recent attempt: -', and 'Glue version: 2.0'. On the right, there are links for 'Logs', 'Output Logs', and 'Error logs', each with a 'Cloudwatch' link icon.

## トラブル対応について

処理が中断した場合や正常終了しなかった場合、Cloudwatchにログが出力されているのでそこで確認が可能。

# 実行結果

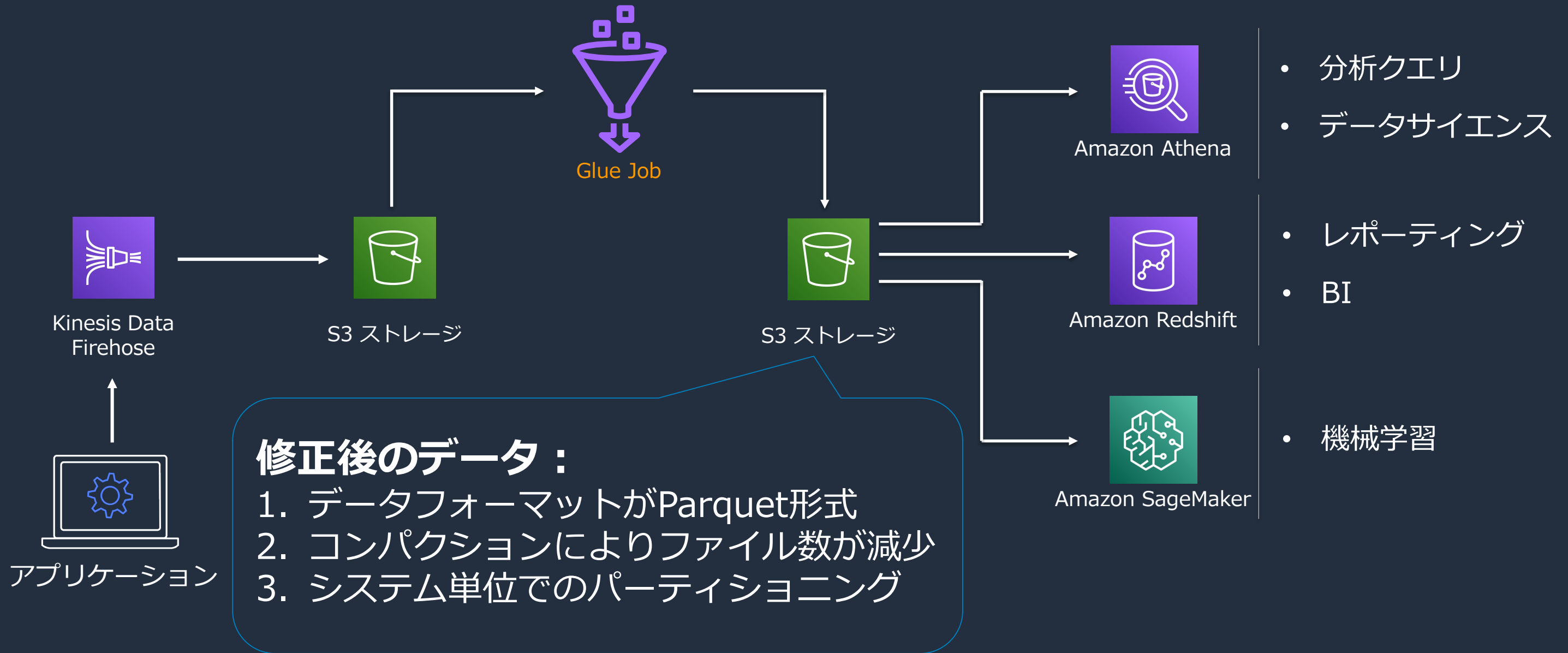
- 出力バケット直下にシステム名でプレフィックスが作成されていることを確認

<input type="checkbox"/>	名前 ▲	タイプ ▼	最終更新日時 ▼
<input type="checkbox"/>	system=SystemA/	フォルダ	-
<input type="checkbox"/>	system=SystemB/	フォルダ	-
<input type="checkbox"/>	system=SystemC/	フォルダ	-
<input type="checkbox"/>	system=SystemD/	フォルダ	-
<input type="checkbox"/>	system=SystemE/	フォルダ	-
<input type="checkbox"/>	system=SystemF/	フォルダ	-

- 各プレフィックス毎にParquet形式の5つのファイルが出力されていることを確認

名前 ▲	タイプ ▼	最終更新日時 ▼
<a href="#">run-DataSink0-1-part-block-0-0-r-00000-snappy.parquet</a>	parquet	2021/03/03 01:43:31 PM JST
<a href="#">run-DataSink0-1-part-block-0-0-r-00001-snappy.parquet</a>	parquet	2021/03/03 01:43:31 PM JST
<a href="#">run-DataSink0-1-part-block-0-0-r-00002-snappy.parquet</a>	parquet	2021/03/03 01:43:31 PM JST
<a href="#">run-DataSink0-1-part-block-0-0-r-00003-snappy.parquet</a>	parquet	2021/03/03 01:43:31 PM JST
<a href="#">run-DataSink0-1-part-block-0-0-r-00004-snappy.parquet</a>	parquet	2021/03/03 01:43:31 PM JST

# ログストリーム / IoTセンサーデータ分析の解決策

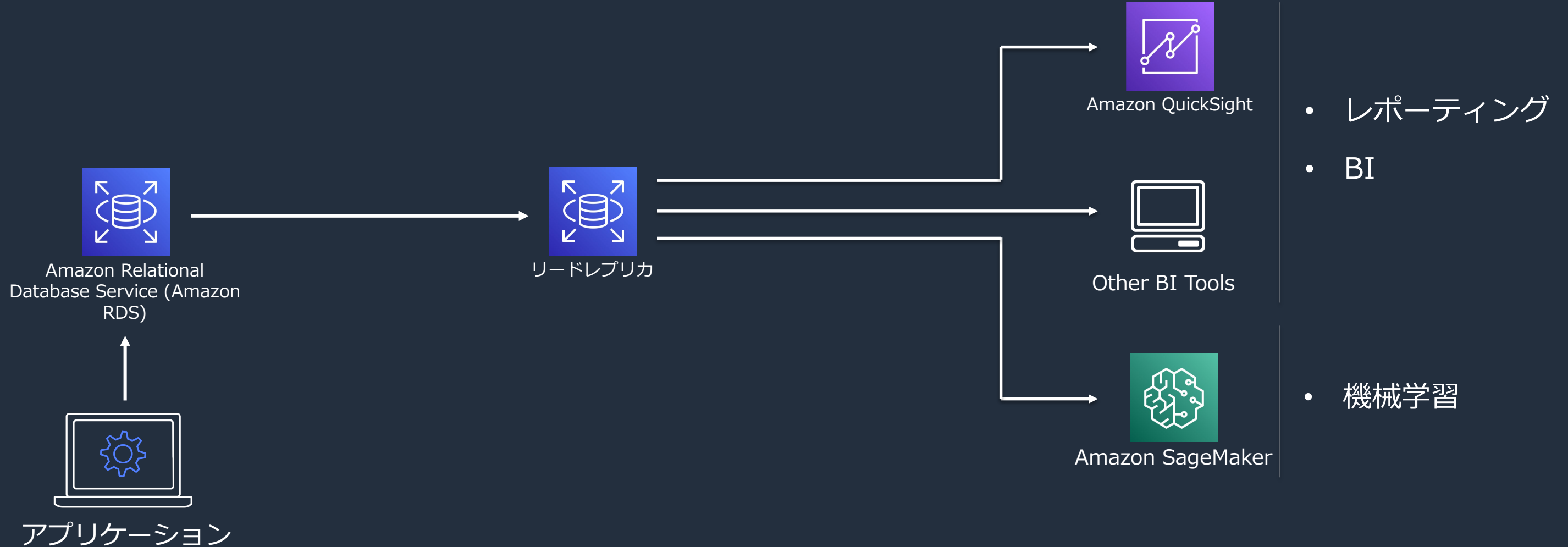




# ユースケース 2: データベースレプリカへの処理をオフロード

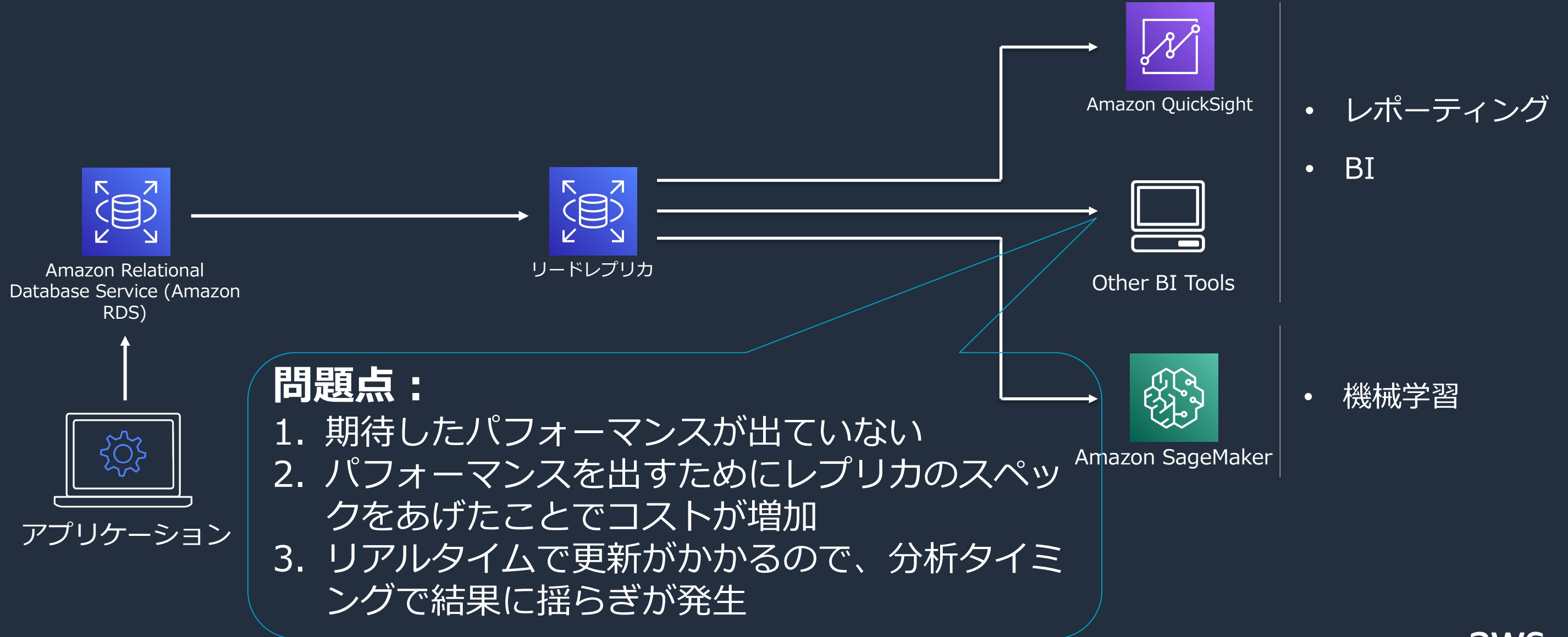
# RDSにあるデータを直接分析している場合の課題

分析用のデータベースレプリカにかかる負荷をデータレイクへオフロード



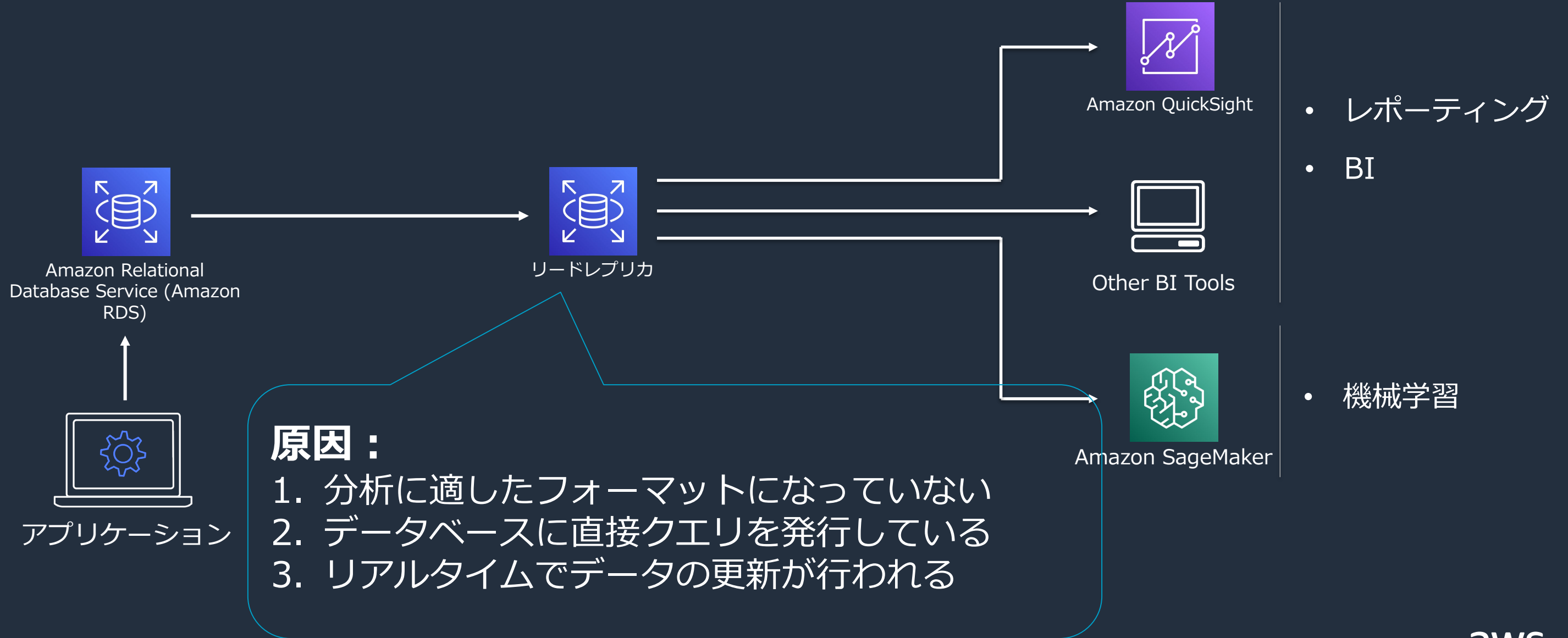
# RDSにあるデータを直接分析している場合の課題

分析用のデータベースレプリカにかかる負荷をデータレイクへオフロード

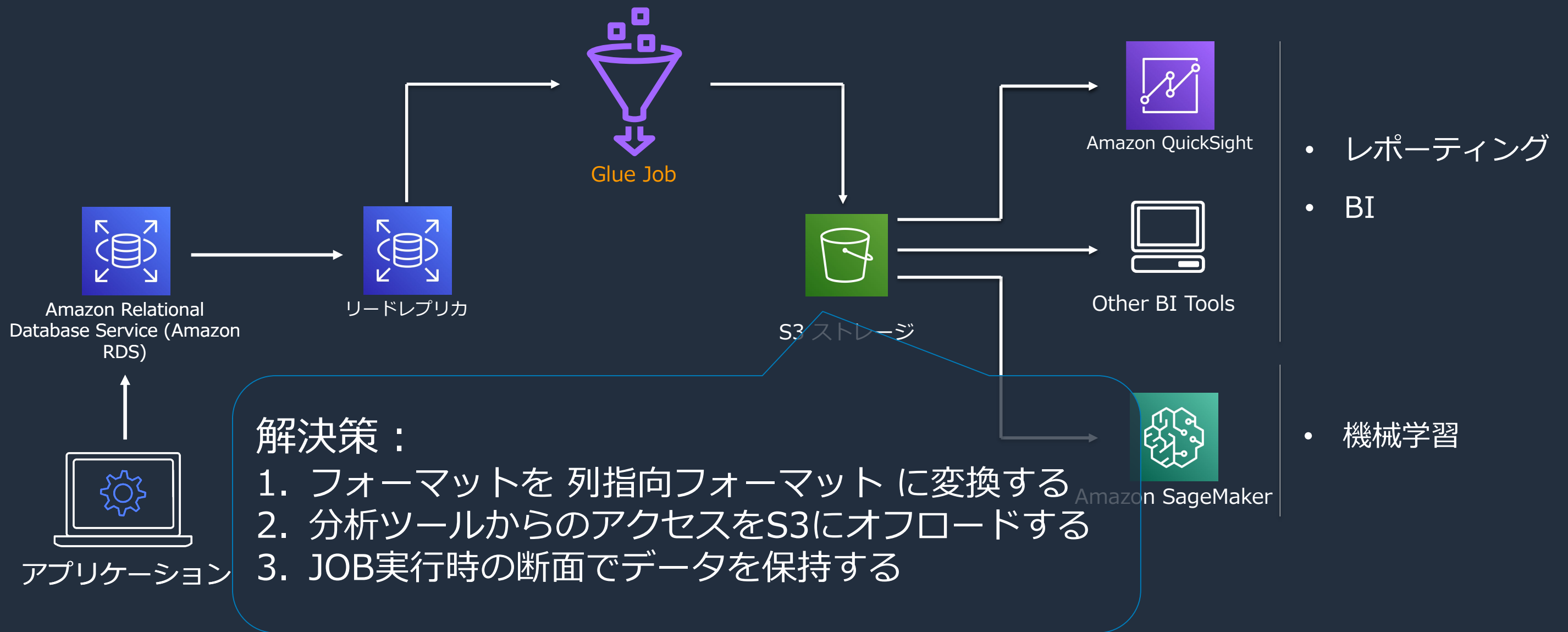


# RDSにあるデータを直接分析している場合の課題

分析用のデータベースレプリカにかかる負荷をデータレイクへオフロード



# RDSにあるデータを直接分析している場合の解決策



# AWS Glue Studioを使用したETL JOBの作成

- マネージメントコンソールより、AWS Glue を選択
- AWS Glue コンソールの左ペインより、AWS Glue Studioを選択
- 表示されたAWS Glue Studioの画面にて Create and manage Jobs を選択



Analytics

## AWS Glue Studio

Visually create job flows and monitor their performance

AWS Glue Studio is an easy-to-use graphical interface for creating, running, and monitoring AWS Glue extract, transform, and load (ETL) jobs.

### Getting started

#### Create and manage jobs

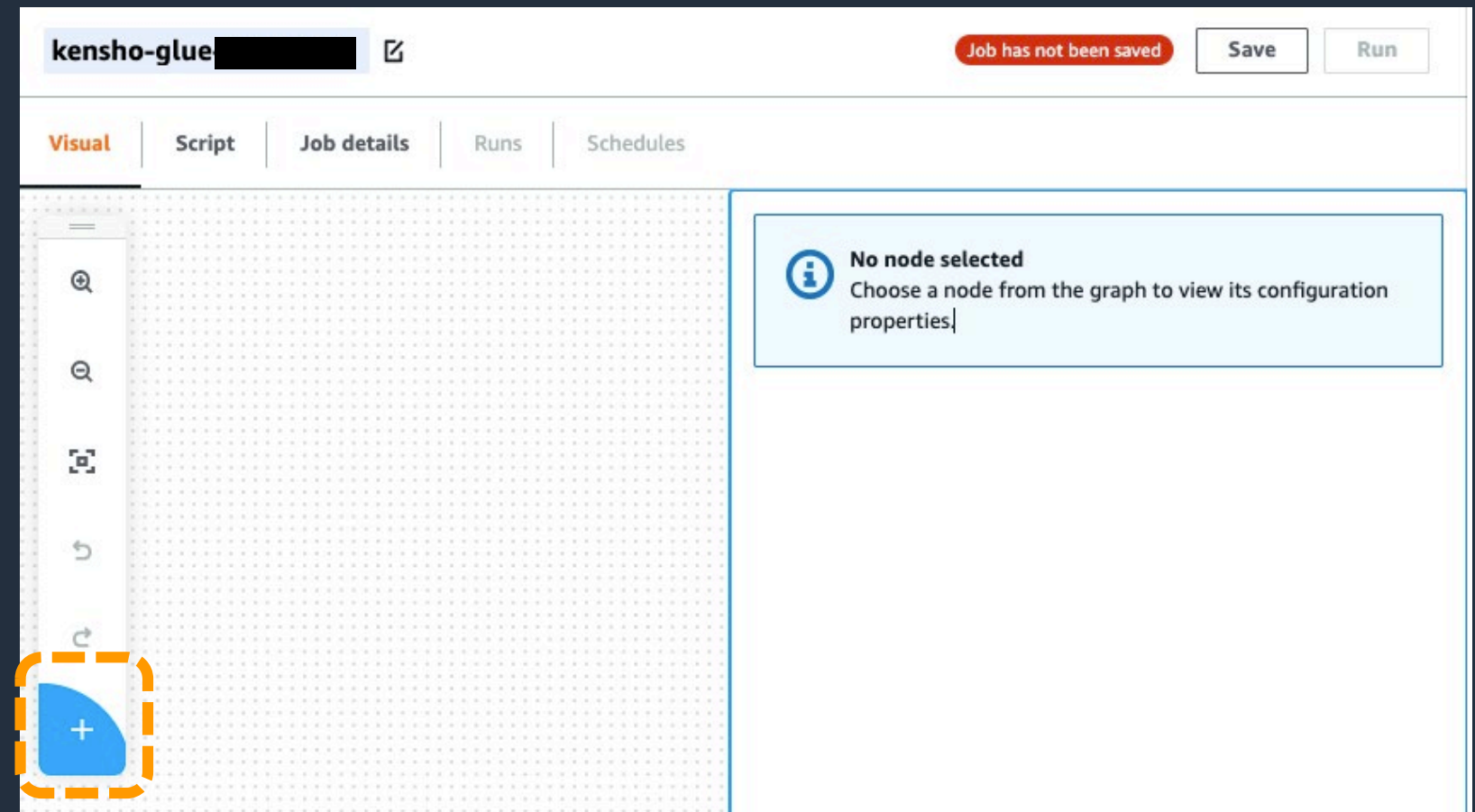
Visually author, run, view, and edit your AWS Glue jobs.

#### Monitor job runs

Diagnose, debug, and check the status of your AWS Glue jobs.

# データソースの指定①

- JobのEdit画面が表示
- 左上にJob名を入力
- 画面左下の“+”を押下し、ノードを表示



# データソースの指定②

- Node typeにJDBCを選択

The screenshot displays the AWS Glue console interface. At the top, there are tabs for 'Visual', 'Script', 'Job details', 'Runs', and 'Schedules'. The 'Visual' tab is active. On the left, a vertical toolbar contains icons for search, zoom, and other actions. The main workspace shows a 'Data source - JDBC' node with a 'JDBC Connection' label and a green checkmark. On the right, the 'Node properties' panel is open, showing the 'Data source properties - JDBC' tab. The 'Name' field is set to 'JDBC Connection'. The 'Node type' dropdown menu is highlighted with a dashed orange border and is set to 'JDBC', with the description 'Glue Data Catalog table with JDBC as the data source.' visible below it.



# データソースの指定③

- DatabaseとTableに、Glue のクローラによって生成されたデータカタログを選択

## DBとの接続情報について

データカタログにはDBへの接続情報も含まれている。その為、データカタログを指定することでDBへのJDBC接続が可能となる。

The screenshot shows the 'Data source properties - JDBC' configuration page in the AWS Glue console. It features three tabs: 'Node properties', 'Data source properties - JDBC' (which is active), and 'Output schema'. The 'Database' field is a dropdown menu with 'kensho-glue-studio-01' selected, and a refresh button to its right. The 'Table' field is a dropdown menu with 'postgres\_public\_cities' selected. A dashed orange border highlights the 'Database' and 'Table' sections.

# 外部テーブルの結合① (追加機能のご紹介)

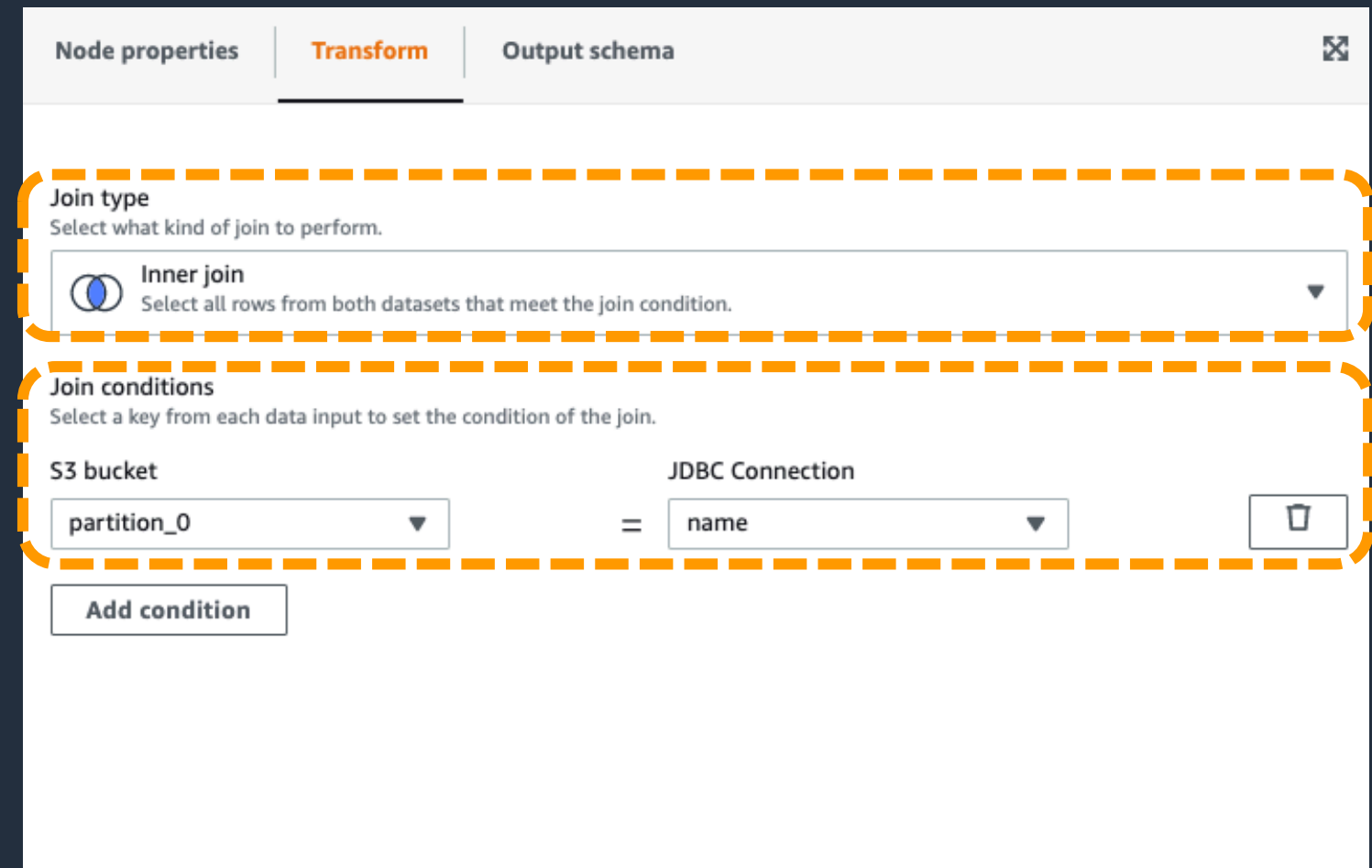
- Node typeにJoinを選択
- Node parentsに結合したい、別のデータソースを選択

結合処理は、JDBCとS3など異なるタイプのデータソース間でも結合が可能。

The screenshot displays the AWS Glue console interface. At the top, there are tabs for 'Visual 1', 'Script', 'Job details', 'Runs', and 'Schedules'. The main workspace shows a job graph with two data sources: 'Data source - S3 bucket' and 'Data source - JDBC', both marked with a green checkmark. Arrows from these sources point to a 'Transform - Join' node, which has a red warning triangle. On the left, a vertical toolbar contains icons for search, zoom, and a red '+' button. On the right, the 'Node properties' panel is open for the 'Join' node. The 'Node type' dropdown is set to 'Join', and the 'Node parents' dropdown is set to 'Select parents'. Below the dropdowns, two data sources are listed as selected parents: 'JDBC Connection RDS - DataSource' and 'S3 bucket S3 - DataSource'. A dashed orange box highlights the 'Node type' and 'Node parents' sections. A red '+' button is also visible at the bottom left of the console.

# 外部テーブルの結合②（追加機能のご紹介）

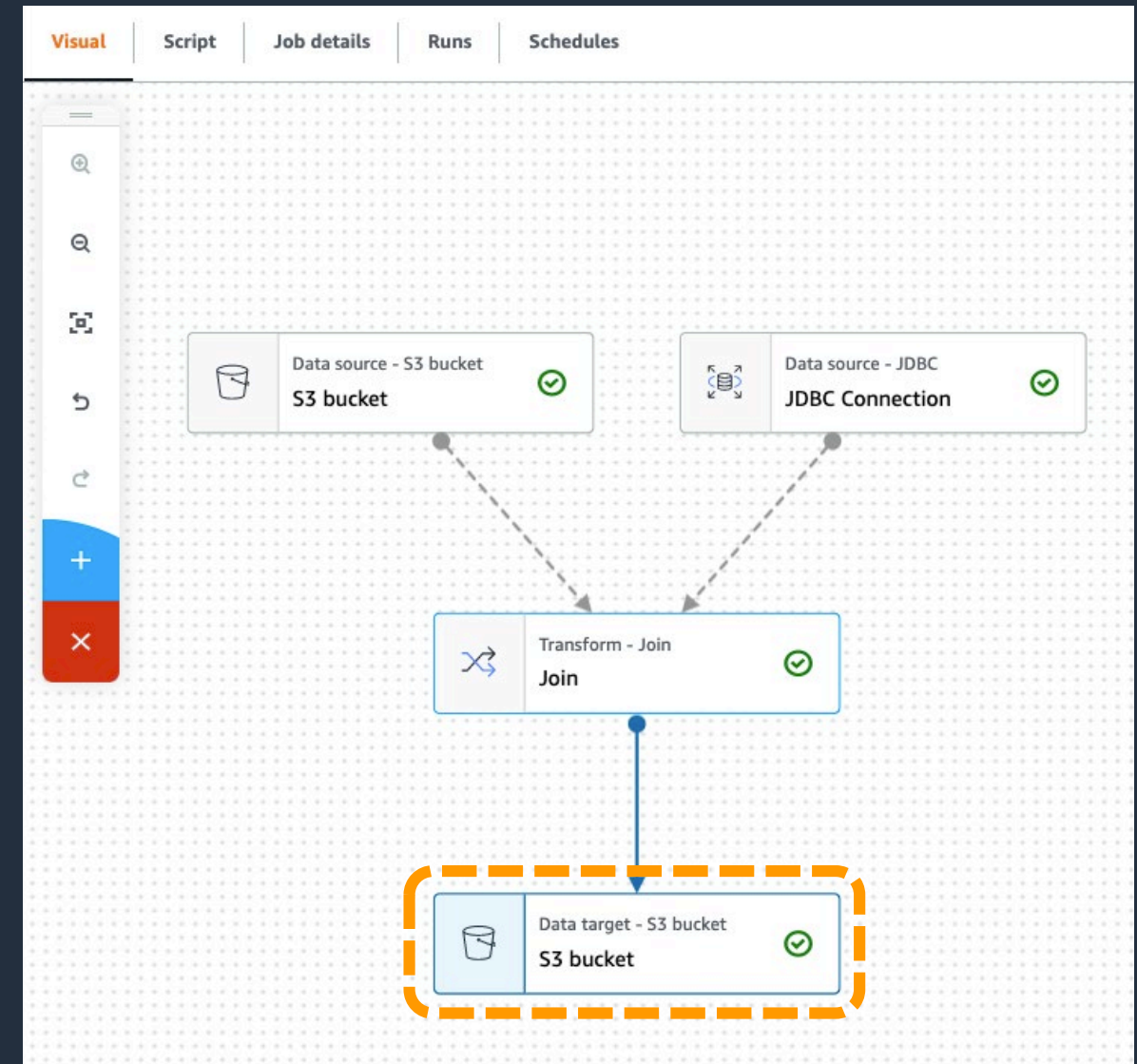
- Join type に Inner join を選択
- Join conditions に結合したいカラムを選択



The screenshot displays the configuration interface for a join operation in the AWS Glue console. It features three tabs: 'Node properties', 'Transform', and 'Output schema'. The 'Transform' tab is active. The 'Join type' section is highlighted with a dashed orange border and shows 'Inner join' selected, with the description 'Select all rows from both datasets that meet the join condition.' Below this, the 'Join conditions' section is also highlighted with a dashed orange border and shows a condition 'partition\_0 = name' where 'partition\_0' is selected from the 'S3 bucket' dropdown and 'name' is selected from the 'JDBC Connection' dropdown. An 'Add condition' button is located below the condition list.

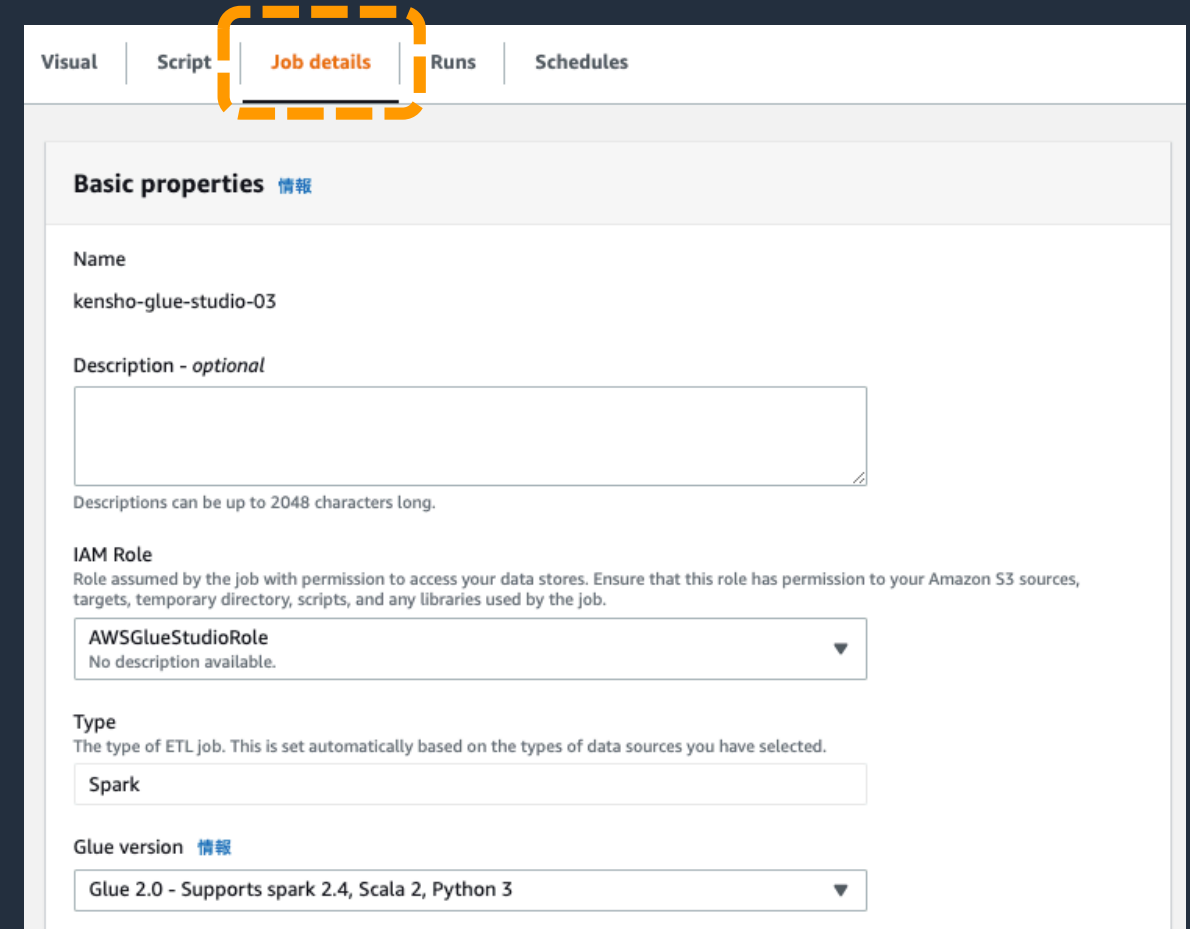
# ターゲットの選択

- Node typeにS3を選択
- その他の設定はユースケース1と同様の内容を実施



# JOBの詳細設定

- Job details タブを選択し、**ユースケース1**と同様の内容を設定
- Save ボタンを押下し、RunボタンでJobを実行



Visual | Script | **Job details** | Runs | Schedules

### Basic properties 情報

Name  
kensho-glue-studio-03

Description - *optional*  
  
Descriptions can be up to 2048 characters long.

IAM Role  
Role assumed by the job with permission to access your data stores. Ensure that this role has permission to your Amazon S3 sources, targets, temporary directory, scripts, and any libraries used by the job.  
AWSGlueStudioRole  
No description available.

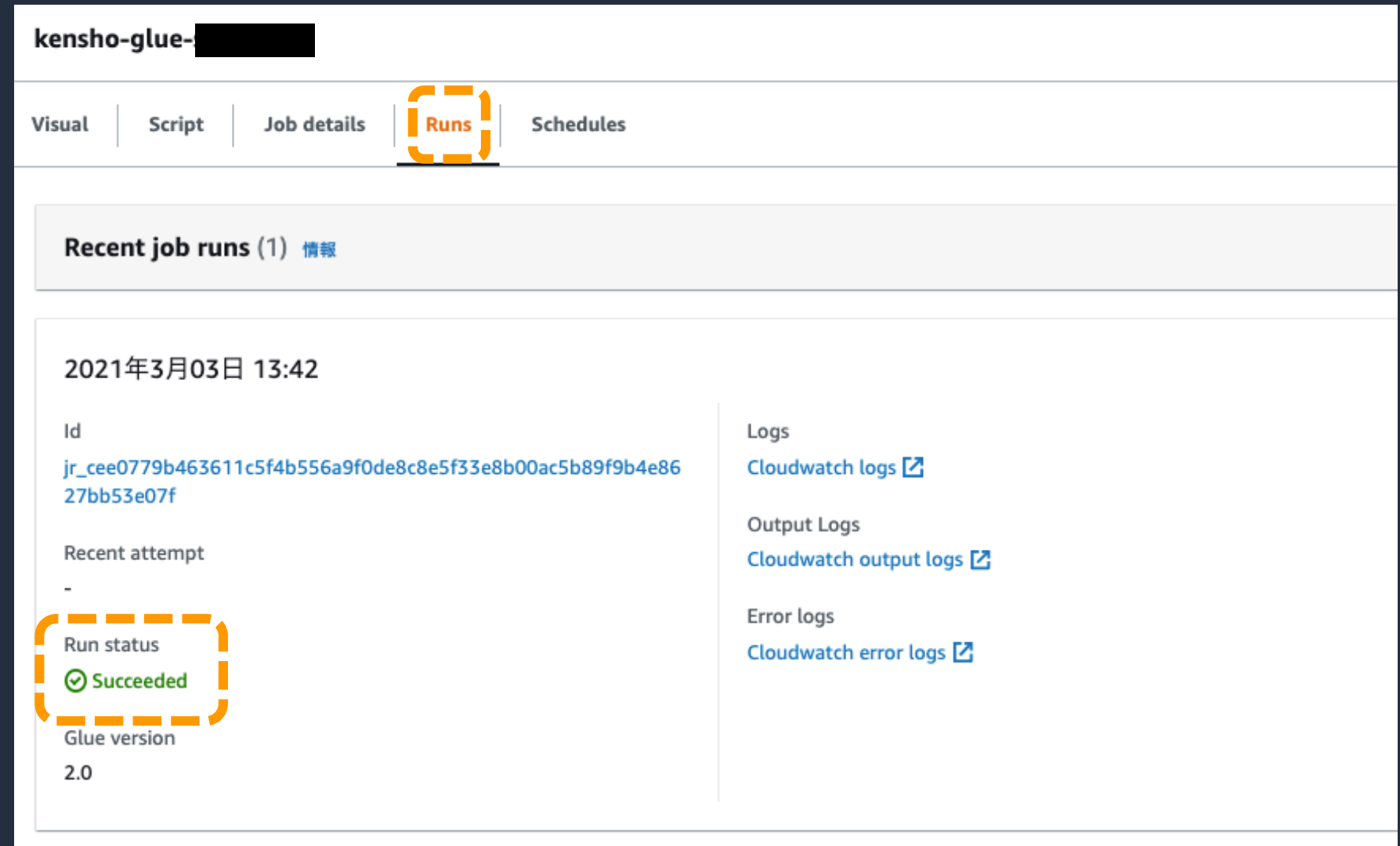
Type  
The type of ETL job. This is set automatically based on the types of data sources you have selected.  
Spark

Glue version 情報  
Glue 2.0 - Supports spark 2.4, Scala 2, Python 3



# 結果のモニタリング

- Runs タブを選択
- Run statusが、Succeededになったことを確認



The screenshot displays the AWS Glue console interface for a job named 'kensho-glue-'. The 'Runs' tab is selected and highlighted with a dashed orange box. The console shows the following details for a recent job run:

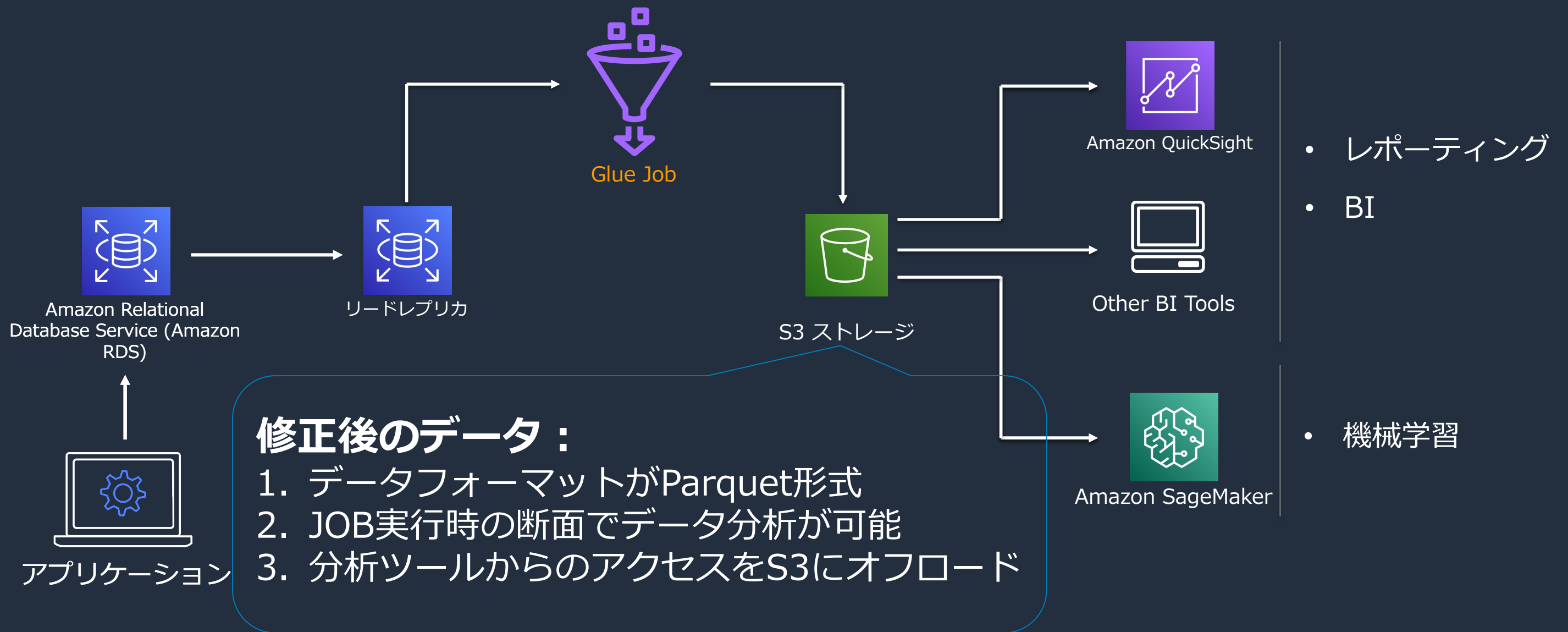
- Recent job runs (1) 情報**
- 2021年3月03日 13:42**
- Id:** jr\_cee0779b463611c5f4b556a9f0de8c8e5f33e8b00ac5b89f9b4e8627bb53e07f
- Recent attempt:** -
- Run status:** Succeeded (indicated by a green checkmark and a dashed orange box)
- Glue version:** 2.0
- Logs:** Cloudwatch logs, Cloudwatch output logs, and Cloudwatch error logs (all with external link icons)

# 実行結果

- 出力バケットにファイルが作成されていることを確認

名前 ▲	タイプ ▼
 <a href="#">run-DataSink0-1-part-block-0-0-r-00000-snappy.parquet</a>	parquet
 <a href="#">run-DataSink0-1-part-block-0-0-r-00001-snappy.parquet</a>	parquet
 <a href="#">run-DataSink0-1-part-block-0-0-r-00002-snappy.parquet</a>	parquet
 <a href="#">run-DataSink0-1-part-block-0-0-r-00003-snappy.parquet</a>	parquet
 <a href="#">run-DataSink0-1-part-block-0-0-r-00004-snappy.parquet</a>	parquet
 <a href="#">run-DataSink0-2-part-block-0-0-r-00000-snappy.parquet</a>	parquet
 <a href="#">run-DataSink0-2-part-block-0-0-r-00001-snappy.parquet</a>	parquet
 <a href="#">run-DataSink0-2-part-block-0-0-r-00002-snappy.parquet</a>	parquet
 <a href="#">run-DataSink0-2-part-block-0-0-r-00003-snappy.parquet</a>	parquet

# RDSにあるデータを直接分析している場合の解決策

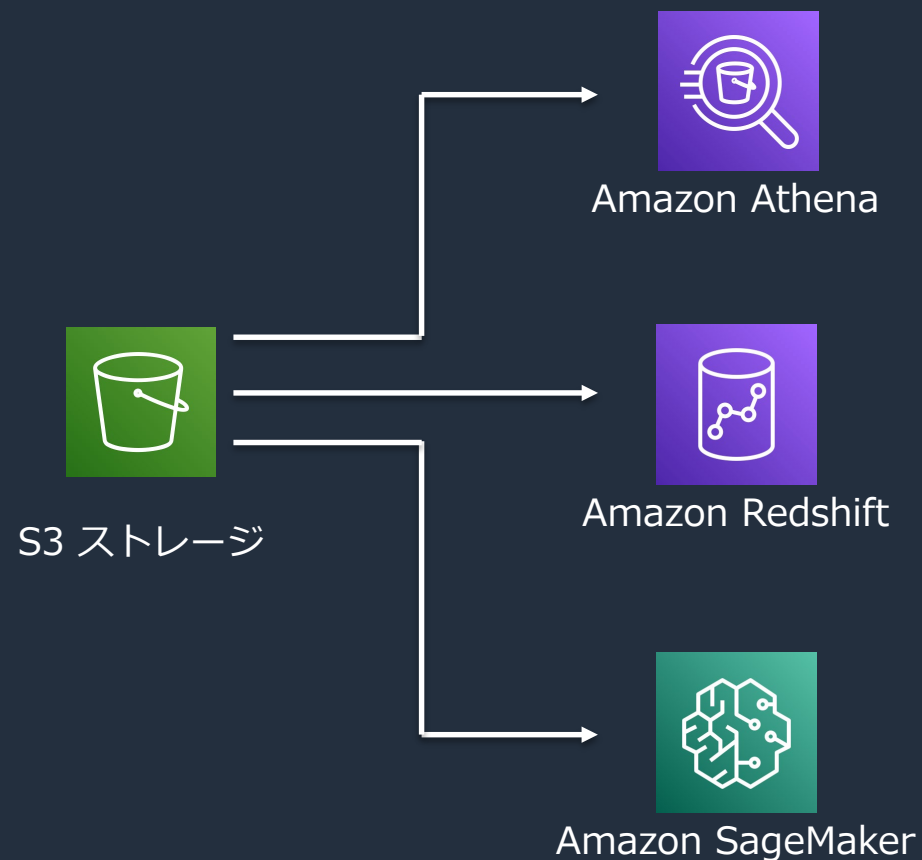
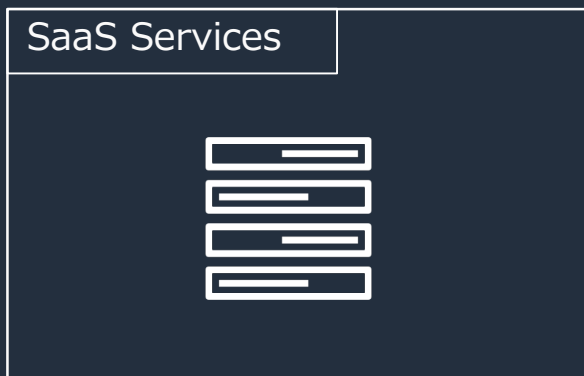




# ユースケース 3： 他クラウド/オンプレミス におけるデータの分析

# 他クラウド/オンプレミス にあるデータを分析する場合の問題点

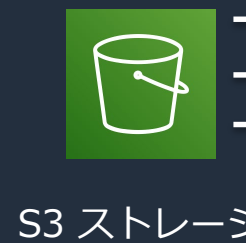
ネイティブでサポートされていないデータソースからのデータ移行



- 分析クエリ
- データサイエンス
- レポーティング
- BI
- 機械学習

# 他クラウド/オンプレミス にあるデータを分析する場合の問題点

ネイティブでサポートされていないデータソースからのデータ移行



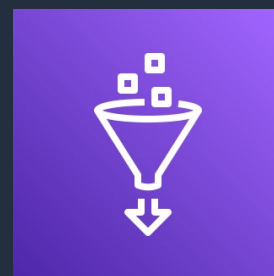
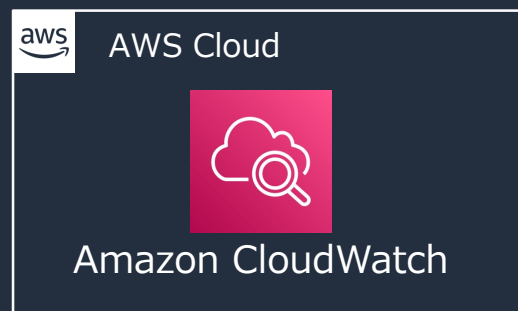
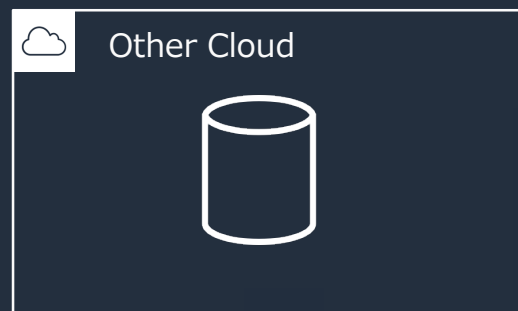
**問題点：**  
データが格納されていないため  
クエリをかけることができない

- 分析クエリ
- データサイエンス
- レポーティング
- BI
- 機械学習

# AWS Glue Custom Connectors

NEW

ネイティブにサポートされていないデータストアとの接続



AWS Glue

任意のデータソースからデータを転送可能

SaaS アプリケーション等から Amazon S3にデータを転送

多くのコネクタを利用可能

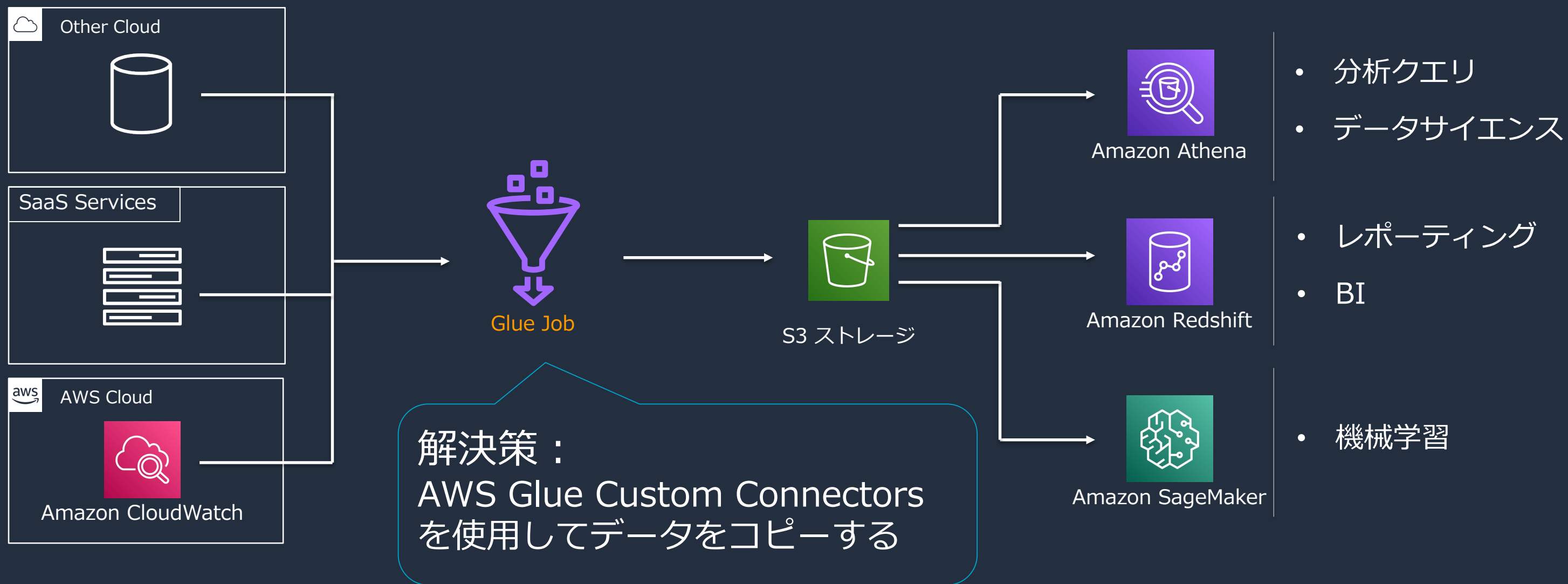
Salesforce、SAP、Snowflake、Google BigQuery 等とも連携

AWS Marketplaceとの統合

開発したカスタムコネクタを AWS Marketplace で共有可能

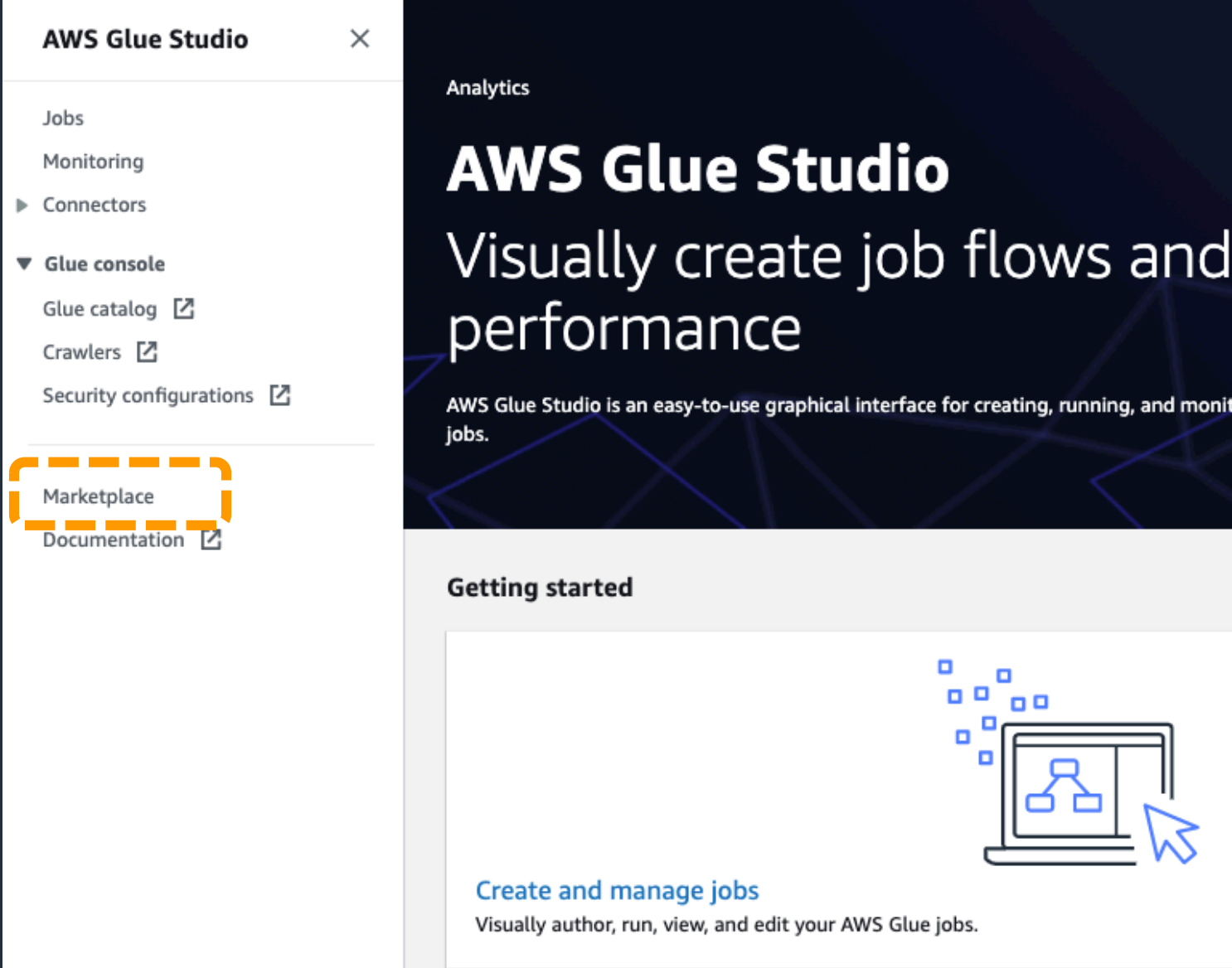
<https://aws.amazon.com/jp/about-aws/whats-new/2020/12/aws-glue-launches-aws-glue-custom-connectors/>

# 他クラウド/オンプレミス にあるデータを分析する場合の解決策



# AWS Glue Custom Connectors の使用

- AWS Glue Studioのホーム画面の左にある Marketplaceを選択



**AWS Glue Studio**

Analytics

## AWS Glue Studio

Visually create job flows and performance

AWS Glue Studio is an easy-to-use graphical interface for creating, running, and monitoring jobs.

### Getting started

**Create and manage jobs**  
Visually author, run, view, and edit your AWS Glue jobs.

# カスタムコネクタのサブスクライブ①

- Topにある検索バーにCloudWatchと入力

- 検索結果に表示される以下を選択

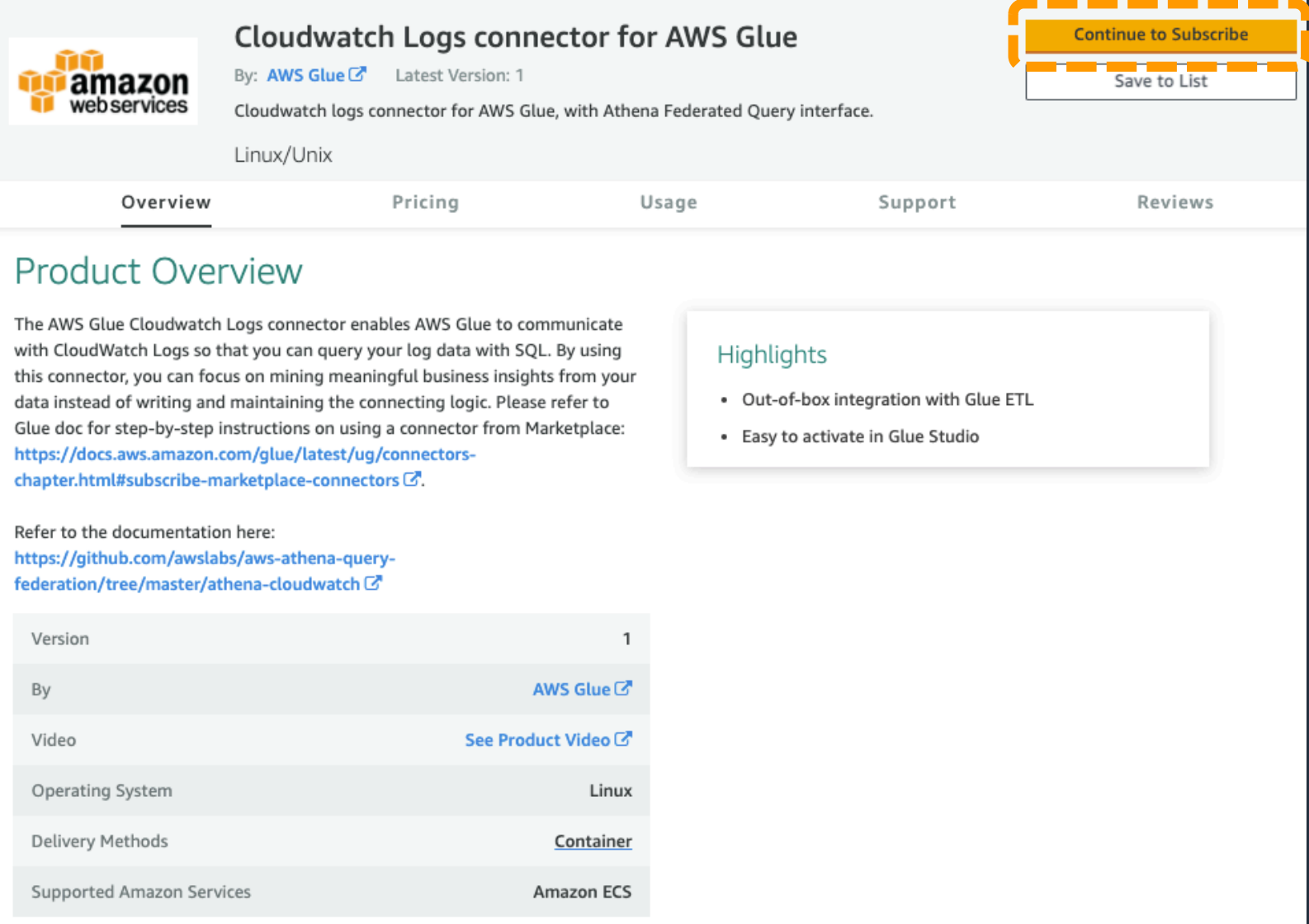
Cloudwatch Logs connector for AWS Glue

The screenshot shows the 'Search AWS Glue Studio products' interface. A search bar at the top contains the text 'CloudWatch' and is highlighted with a dashed orange border. Below the search bar, it indicates 'All products (61 results) showing 1 - 10'. The search results are listed as follows:

- aws** **AWS Glue Connector for Google BigQuery** [↗](#)  
By [Amazon Web Services](#) [↗](#) | Ver 2.12.0  
★★★★★ 1 AWS review [↗](#)  
The AWS Glue Connector for Google BigQuery simplifies the process of connecting AWS Glue to BigQuery data, facilitating cloud ETL processes for operational reporting, ba
- amazon web services** **AWS Glue Connector for Apache Hudi** [↗](#)  
By [AWS Glue](#) [↗](#) | Ver 0.5.3  
The AWS Glue Connector for Apache Hudi simplifies the process to create and update on Read (MOR) storage types.
- aws** **Elasticsearch Spark connector for AWS Glue** [↗](#)  
By [AWS Glue](#) [↗](#) | Ver 7.8.0  
This product helps you read and write data from Elasticsearch using Apache Spark. By writing and maintaining the connecting logic. For more details, please refer to Glue E
- amazon web services** **Cloudwatch Logs connector for AWS Glue** [↗](#)  
By [AWS Glue](#) [↗](#) | Ver 1  
The AWS Glue Cloudwatch Logs connector enables AWS Glue to communicate with CloudWatch Logs and mine meaningful business insights from your data instead of writing and maintainin

# カスタムコネクタのサブスクリプション②

- カスタムコネクタの内容を確認し、画面右上のContinue to Subscribe を選択



**amazon web services**

## Cloudwatch Logs connector for AWS Glue

By: [AWS Glue](#) Latest Version: 1

Cloudwatch logs connector for AWS Glue, with Athena Federated Query interface.

Linux/Unix

[Continue to Subscribe](#)  
[Save to List](#)

[Overview](#) [Pricing](#) [Usage](#) [Support](#) [Reviews](#)

### Product Overview

The AWS Glue Cloudwatch Logs connector enables AWS Glue to communicate with CloudWatch Logs so that you can query your log data with SQL. By using this connector, you can focus on mining meaningful business insights from your data instead of writing and maintaining the connecting logic. Please refer to Glue doc for step-by-step instructions on using a connector from Marketplace: <https://docs.aws.amazon.com/glue/latest/ug/connectors-chapter.html#subscribe-marketplace-connectors>.

Refer to the documentation here: <https://github.com/aws-labs/aws-athena-query-federation/tree/master/athena-cloudwatch>

Version	1
By	<a href="#">AWS Glue</a>
Video	<a href="#">See Product Video</a>
Operating System	Linux
Delivery Methods	<a href="#">Container</a>
Supported Amazon Services	Amazon ECS

### Highlights

- Out-of-box integration with Glue ETL
- Easy to activate in Glue Studio



# カスタムコネクタのサブスクライブ③

- ソフトウェアのバージョン等の選択後に、出力されるUsage Instructionsを押下
- 出力されるポップアップにて、以下を選択  
**Activate the Glue connector in AWS Glue Studio**
- Custom connector の設定画面へ自動遷移

amazon web services Cloudwatch Logs connector for AWS Glue

< Product Detail Subscribe Configure

Launch this software

Review your configuration and choose how you wish to launch the software.

**Usage Instructions for 1**

Please subscribe to the product from AWS Marketplace and **Activate the Glue connector from AWS Glue Studio**

**Configuration Details**

Fulfillment Option	Activate in AWS Glue Studio
Software Version	1
Supported Amazon Services	Amazon ECS

**Usage Instructions**

**Container Images**

This product has 1 container image. Use the following options to deploy the product.

Deployment template

Please see usage instruction above

View container image details

# コネクタの作成

- Name欄に任意の名前を入力
- Connection access は今回は空欄

## 認証情報について

他クラウドやSaaSへの接続を行う際には認証情報はあらかじめ、AWS Secrets Managerに格納する必要がある。今回は、CloudWatchへの接続のため入力不要。

AWS Glue Studio > Connectors > Create connection

### Create connection 情報

**Connection properties** 情報

Connector  
AthenaCloudwatchLogs  
Athena connector for AWS Cloudwatch logs

Name  
Enter a unique name for your connection.  
kensho-glue-studio-  
Names can contain letters (A-Z), numbers (0-9), hyphens (-), or underscores (\_), and must be less than 256 characters long.

Description - optional  
Descriptions can be up to 2048 characters long.

**Connection access** 情報

AWS Secret - optional 情報  
Choose a secret from [AWS Secrets Manager](#). [AWS secrets eliminate hardcoding sensitive information.](#)  
Choose a secret to use

# データソースの指定①

- Node type にAthenaCloudwatchLogsが、追加されていることを確認

The screenshot displays the AWS Glue console interface. On the left, a card titled 'Data source - Connection AthenaCloudwatchLogs' is shown. On the right, the 'Node type' dropdown menu is open, showing a search bar and a list of options. The option 'AthenaCloudwatchLogs' is selected and highlighted with a dashed orange border. Below the node type, the 'Data target' section is visible, showing options like 'S3', 'AWS Glue Data Catalog', and 'AWS Glue Connector for Google BigQuery'.

# データソースの指定②

- Connectionに先程作成したコネクターを選択
- CloudWatchを確認し取得対象となるLogStreamsとLogGroupsを確認
- Table nameにLogStreamsを入力し、Schema nameにLogGroupsを入力

LogGroup内のすべてのLogStreamsで構成される特別な「all\_log\_streams (すべてのログストリーム)」などでもログを指定可能。

※詳細は以下リンク先ご参照。

The screenshot shows the 'Data source properties - Connector' configuration page in the AWS Glue console. It features three tabs: 'Node properties', 'Data source properties - Connector' (selected), and 'Output schema'. The 'Connection' section has a dropdown menu with 'AthenaConnectorTest01' selected and a refresh button. The 'Table name' section has a text input field containing a long alphanumeric string. The 'Schema name' section has a text input field containing '/aws-glue/jobs/error'. Below the schema name field is an 'Add schema' button. The 'Schema' section below has a description and an 'Add schema' button.

<https://github.com/aws-labs/aws-athena-query-federation/blob/master/athena-cloudwatch/README.md>

# ターゲットの指定

- Connection source ノードを選択した状態で、画面左下の“+”を押下
- Node typeにS3を選択し、**ユースケース1**と同様のターゲット設定を実施



# JOBの詳細設定

- Job detailsタブを選択
- IAM Role に必要な権限を持ったロールを選択
- その他はユースケース1と同じよう設定

必要な権限について  
ユースケース3にて追加で必要となる権限は以下。

- AWSの管理ポリシーである  
AmazonEC2ContainerRegistryReadOnly権限
- glue:GetJobおよびglue:GetJobsアクセス許可
- 取得対象となるCloud Watch Logsへのアクセス権限

これらの権限をユースケース1のロールに追加で付与した上でロールを選択する。

Visual | Script | **Job details** | Runs | Schedules

**Basic properties** 情報

Name  
kensho-glue-██████████

Description - optional

Descriptions can be up to 2048 characters long.

**IAM Role**  
Role assumed by the job with permission to access your data stores. Ensure that this role has permission to your Amazon S3 sources, targets, temporary directory, scripts, and any libraries used by the job.

AWSGlueStudioRole  
No description available.

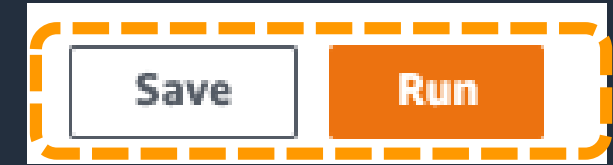
Type  
The type of ETL job. This is set automatically based on the types of data sources you have selected.

Spark

**Glue version** 情報  
Glue 2.0 - Supports spark 2.4, Scala 2, Python 3

# 結果のモニタリング

- 画面右上のSaveを押下した後、Runボタンを押下しJobを実行
- Runs タブを選択
- Run statusが、Succeededになったことを確認

A screenshot of the AWS Glue console interface for a job named 'kensho-glue-'. The 'Runs' tab is selected and highlighted with a dashed orange border. The page shows a list of recent job runs with one entry for '2021年3月03日 13:42'. The details for this run are shown in a table-like format. The 'Run status' is 'Succeeded', which is highlighted with a dashed orange border. Other details include the job ID, recent attempt, and Glue version (2.0). On the right side, there are links for 'Logs', 'Output Logs', and 'Error logs', each with a 'Cloudwatch' link icon.

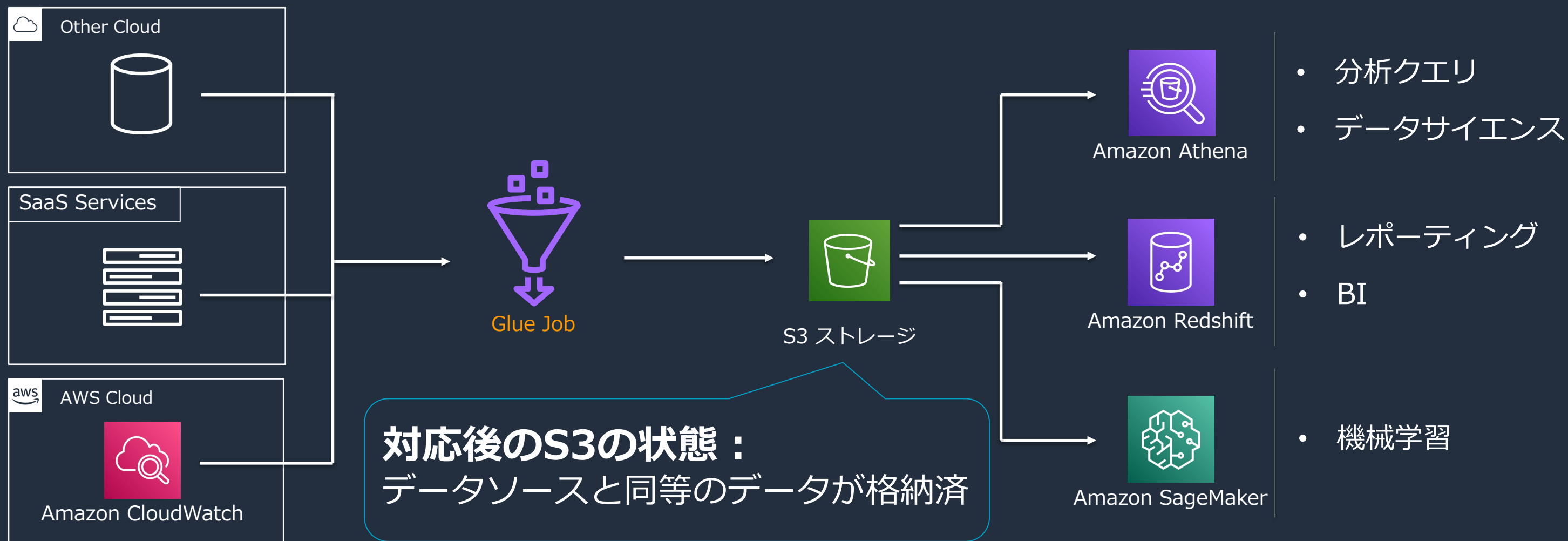
Recent job runs (1) <a href="#">情報</a>	
2021年3月03日 13:42	
Id	<a href="#">jr_cee0779b463611c5f4b556a9f0de8c8e5f33e8b00ac5b89f9b4e8627bb53e07f</a>
Recent attempt	-
Run status	<span>✔ Succeeded</span>
Glue version	2.0

Logs  
[Cloudwatch logs](#)

Output Logs  
[Cloudwatch output logs](#)

Error logs  
[Cloudwatch error logs](#)

# 他クラウド/オンプレミス にあるデータを分析する場合の解決策





# その他の主要アップデート

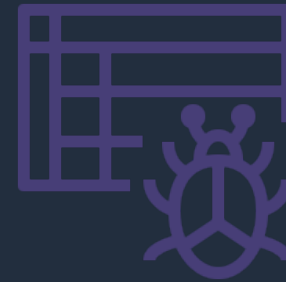
# AWS Glue の構成要素と周辺サービス (再掲)



Extract, Transform, and Load (ETL) ジョブ



AWS Glue Data Catalog



Crawler

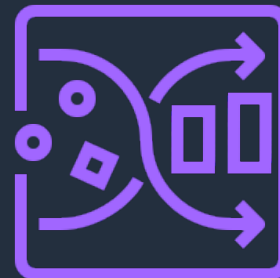


Workflow Management

New



AWS Glue Studio



AWS Glue DataBrew

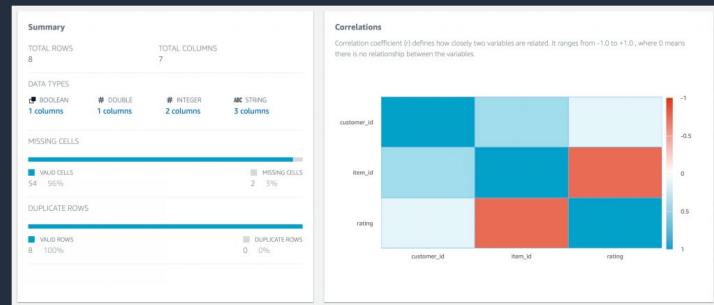
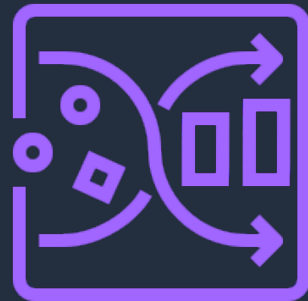


AWS Glue Elastic Views

# AWS Glue DataBrew

NEW

## 分析および機械学習のためのビジュアルライズツール



## データ準備タスクの自動化

データ準備にかかる時間を最大80%短縮

## ノンコードでのデータ変換

250を超える既成の組込関数を選択して処理を実施

**Comments**  
Dataset: Comments | Sample: First n sample (8 rows) | Create job

Viewing 7 columns | 8 rows

#	item_id	ABC category	#	rating
1	2345	Electronics;Computer	2	5.0
2	5432	Home;Furniture	2	1.0
3	3245	Electronics;Photography	3	3.0
4	2345	Electronics;Computer	2	4.0
5	4536	Home;Kitchen	5	5.0
6	5432	Home;Furniture	1	1.0
7	4536	Home;Kitchen	3	3.0
8	3245	Electronics;Photography	4	4.0

**Recipe (0)**  
Comments-recipe  
Version 0.1

Build your recipe  
Start applying transformation steps to your data. All your data preparation steps will be tracked in the recipe.

Add step

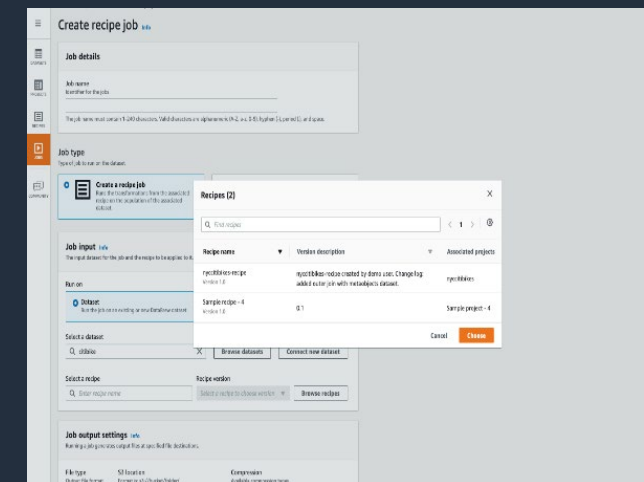
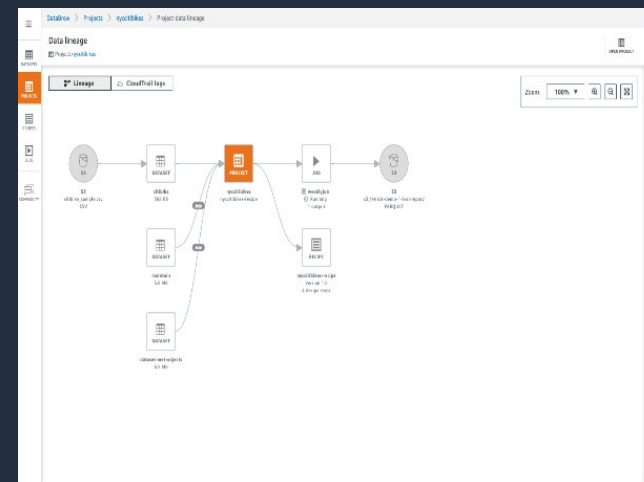
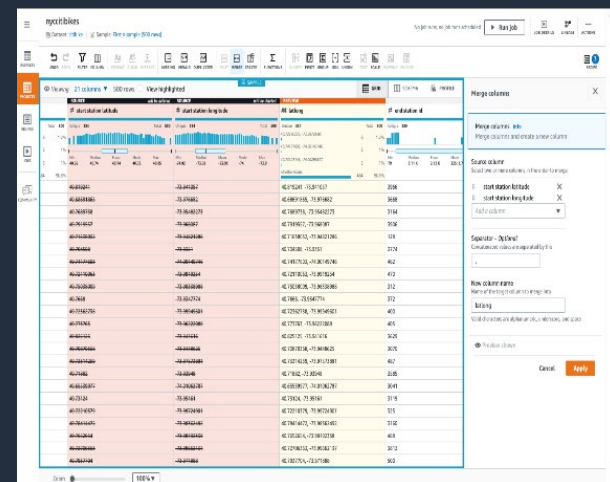
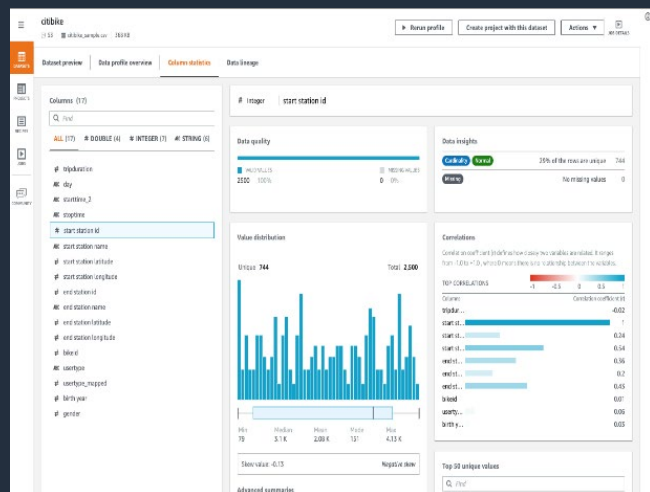
## サーバレス

インフラを管理することなく、テラバイト規模のデータを変換

# AWS Glue DataBrew の特徴

NEW

TB~PBクラスのデータに対して、クレンジングと正規化を直接実行



## データ品質の理解

データパターンを理解し異常を検出するためにプロファイリングを行い、データの品質を評価

## データのクリーンアップと正規化

250種類以上の変換処理から選択して、データの視覚化、クリーンアップ、正規化を実施

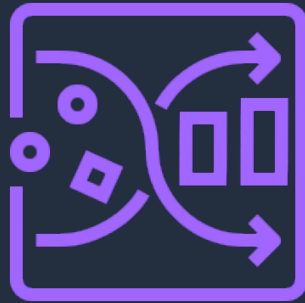
## データリネージの視覚化

さまざまなデータソースと変換手順を視覚化してトラッキング

## 自動化

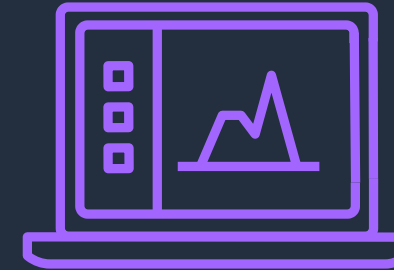
保存された変換手順を新しい入力データに適用

# AWS Glue DataBrew と AWS Glue Studioの違い



## AWS Glue DataBrew

- GUI でデータを可視化しながらデータ変換したい
- Data Brew で利用可能な 250種類以上の変換処理を活用したい
- **データアナリスト & データサイエンティスト向け**



## AWS Glue Studio

- プログラムコードでデータ変換したい
- GUI でデータ変換を記述しつつ、必要に応じてカスタムコードを加えたい
- カスタムコネクタでさまざまなデータソースと接続したい
- ストリーミング処理を活用したい
- **ETLデベロッパー向け**

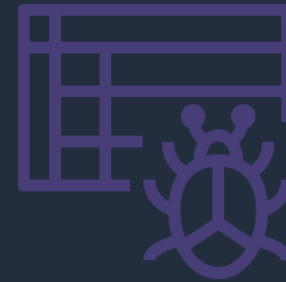
# AWS Glue の構成要素と周辺サービス (再掲)



Extract, Transform, and Load (ETL) ジョブ



AWS Glue Data Catalog



Crawler



Workflow Management

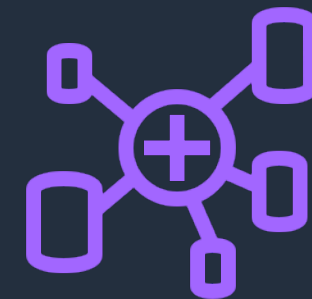
New



AWS Glue Studio



AWS Glue DataBrew

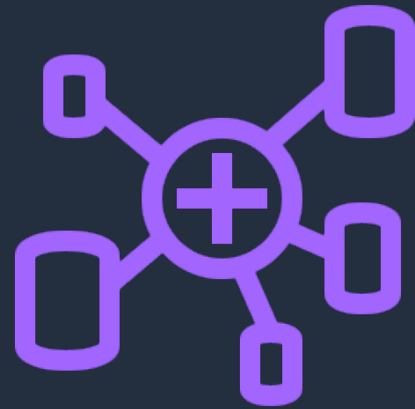


AWS Glue Elastic Views

# AWS Glue Elastic Views (Preview)

NEW

複数のデータソースにまたがるマテリアライズドビューを作成



## SQLを用いてターゲットDBを指定

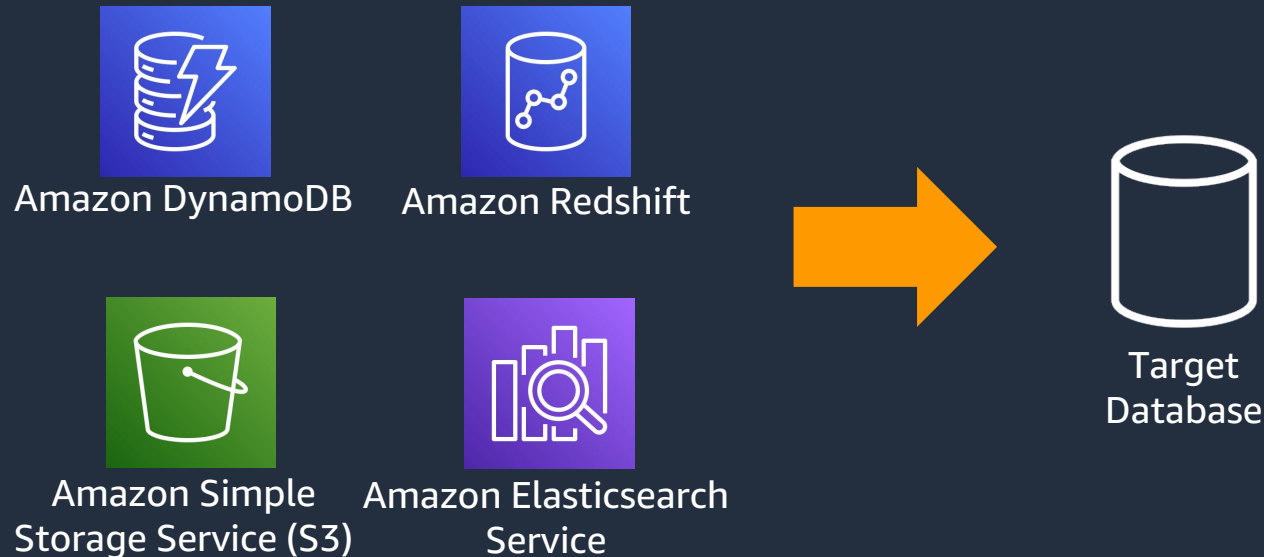
ターゲットデータベースに対して自動的にデータを複製

## データ変更に対応

データソースをモニタし、データの変更が発生したら迅速で反映

## サーバーレス

インフラ管理は不要。キャパシティを自動的に制御



# プレビュー時にサポートされるsourceとtarget

NEW

	Source	Target
Amazon DynamoDB	✓	
Amazon Elasticsearch Service		✓
Amazon S3		✓
Amazon Redshift		✓



# previewで利用可能なユースケース

NEW

- DynamoDB内のデータの検索インデックスをElasticsearch内に作成



- DynamoDB内の業務データをDataLakeに統合



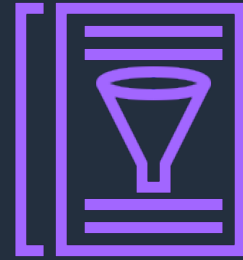
- DynamoDB内のデータをRedshiftで分析



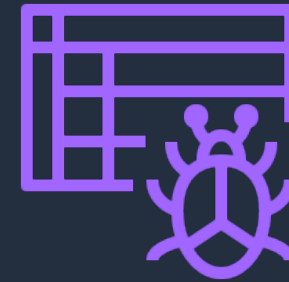
# AWS Glue の構成要素と周辺サービス (再掲)



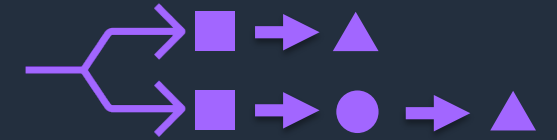
Extract, Transform, and Load (ETL) ジョブ



AWS Glue Data Catalog



Crawler

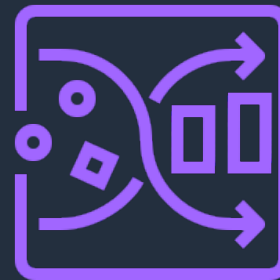


Workflow Management

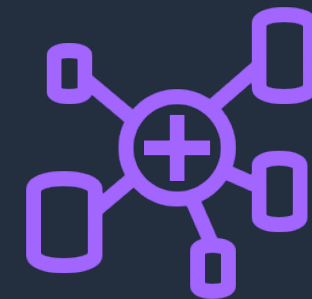
New



AWS Glue Studio



AWS Glue DataBrew



AWS Elastic Views

# まとめ

# まとめ

1. AWS Glueとは、  
サービス間でデータを簡単に**移動**するための、**サーバーレスデータ統合サービス**
2. データの前処理が必要な場合、  
**AWS Glue Studio**を使用することで**グラフィカル**にETLJobを作成可能
3. データがAWS以外に格納されている場合でも、  
**AWS Glue Custom Connectors**を使用することでデータの移動が可能

# 参考情報

AWS Glue ホームページ

<https://aws.amazon.com/jp/glue/>

AWS Glue 開発者ガイド（公式ドキュメント）

<https://aws.amazon.com/jp/documentation/glue/>

AWS Glue custom connectors の設定例

<https://aws.amazon.com/jp/blogs/big-data/migrating-data-from-google-bigquery-to-amazon-s3-using-aws-glue-custom-connectors/>

AWS Glueの料金

<https://aws.amazon.com/jp/glue/pricing/>

AWS Glueのサービス制限

[https://docs.aws.amazon.com/ja\\_jp/general/latest/gr/aws\\_service\\_limits.html#limits\\_glue](https://docs.aws.amazon.com/ja_jp/general/latest/gr/aws_service_limits.html#limits_glue)

# Q&A

お答えできなかったご質問については

AWS Japan Blog <https://aws.amazon.com/jp/blogs/news/> に  
後日掲載します。

# AWS の日本語資料の場所「AWS 資料」で検索



日本担当チームへお問い合わせ サポート 日本語 ▼ アカウント ▼

コンソールにサインイン

製品 ソリューション 料金 ドキュメント 学習 パートナー AWS Marketplace その他 🔍

## AWS クラウドサービス活用資料集トップ

アマゾン ウェブ サービス (AWS) は安全なクラウドサービスプラットフォームで、ビジネスのスケールと成長をサポートする処理能力、データベースストレージ、およびその他多種多様な機能を提供します。お客様は必要なサービスを選択し、必要な分だけご利用いただけます。それらを活用するために役立つ日本語資料、動画コンテンツを多数ご提供しております。(本サイトは主に、AWS Webinar で使用した資料およびオンデマンドセミナー情報を掲載しています。)

[AWS Webinar お申込 »](#)

[AWS 初心者向け »](#)

[業種・ソリューション別資料 »](#)

[サービス別資料 »](#)

<https://amzn.to/JPArchive>



# AWS Well-Architected 個別技術相談会

毎週“W-A個別技術相談会”を実施中

- AWSのソリューションアーキテクト(SA)に  
対策などを相談することも可能

- **申込みはイベント告知サイトから**  
(<https://aws.amazon.com/jp/about-aws/events/>)

**AWS イベント**

**で[検索]**





# 4月以降のBlack Belt Online Seminarについて

ライブ配信によるBlack Belt Online Seminarは3月一杯で終了し、  
今後はオンデマンドによる定期配信に変更いたします。

今後もコンテンツを拡充して行きますので、楽しみにお待ちしております。

オンデマンドでの配信スケジュールは、AWS Blog, AWSニュースレターでお知らせいたします（5月17日週に再開を予定しています）



# ご視聴ありがとうございました

AWS 公式 Webinar  
<https://amzn.to/JPWebinar>



過去資料

<https://amzn.to/JPArchive>

