



# 【AWS Black Belt Online Seminar】 AWSで構築するデータレイク基盤の アーキテクチャ

アマゾン ウェブ サービス ジャパン株式会社  
ソリューションアーキテクト 丹羽勝久

2018.04.24

# 自己紹介

名前：

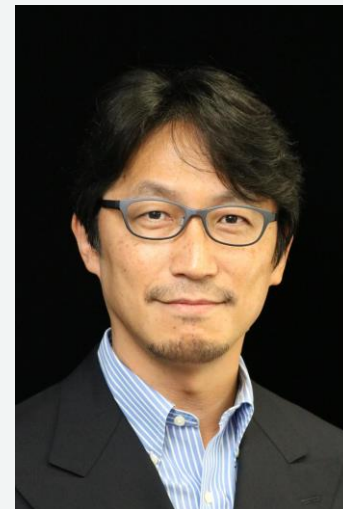
丹羽 勝久（にわ かつひさ）

所属：

アマゾンウェブサービスジャパン株式会社  
インダストリーソリューション部  
ソリューションアーキテクト

担当：

製造公益領域のお客様担当



# 内容についての注意点

- 本資料では2018年4月24日時点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が東京リージョンを使用する場合、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.

# Agenda

- データレイクとは何か？なぜ必要なのか？
- データレイクアーキテクチャ
- データレイクを構成するAWSのサービス
- データレイク活用事例

データレイクの話の前に….

# 「ビッグデータ」 について考えてみましょう

# ビッグデータとは？

3Vs: Volume(大容量), Velocity(高更新頻度), Variety(多様)

5Vs: **3Vs** + Value(高い価値), Veracity(高い正確性)



**Wikipediaより**) <https://ja.wikipedia.org/wiki/ビッグデータ>

ビッグデータ (英: big data) とは、一般的なデータ管理・処理ソフトウェアで扱うことが困難なほど巨大で複雑なデータの集合を表す用語である。

**Gartnerの定義**) <https://www.gartner.com/it-glossary/big-data/>

**Big data is high-volume, high-velocity and/or high-variety information assets that demand cost-effective, innovative forms of information processing that enable enhanced insight, decision making, and process automation.**

# ビッグデータ分析：論理的推論の2つの手法

## • 演繹法

- 一般論やルールに観察事項を加えて、必然的な結論を導く思考方法で三段論法とも言われる

例) 「人間はいつか死ぬ」 → 「ソクラテスは人間である」 → 「ソクラテスはいつか死ぬ」

## • 帰納法

- 帰納法は、多くの観察事項（事実）から類似点をまとめ上げることで結論を引き出すという論法

例)

- 「人であるソクラテスは死んだ」
  - 「人であるプラトンは死んだ」
  - 「人であるアリストテレスは死んだ」
- 「したがって人は全て死ぬ」

機械学習や高度統計分析の  
アプローチ

# ビッグデータ分析の分類

## トップダウンアプローチ (演繹的)

- ①分析の要件、目的を明確化  
→ 経営分析、財務分析、売上分析など
- ②データモデリング、ディメンジョン設計
- ③データ収集、変換、ロード (ETL)  
→ どのシステムからデータを収集するか?
- ④分析実行、レポート生成

**データウェアハウス、データマート**  
SQL主体で基本100%の正確性で  
事実把握 (Schema on Write)

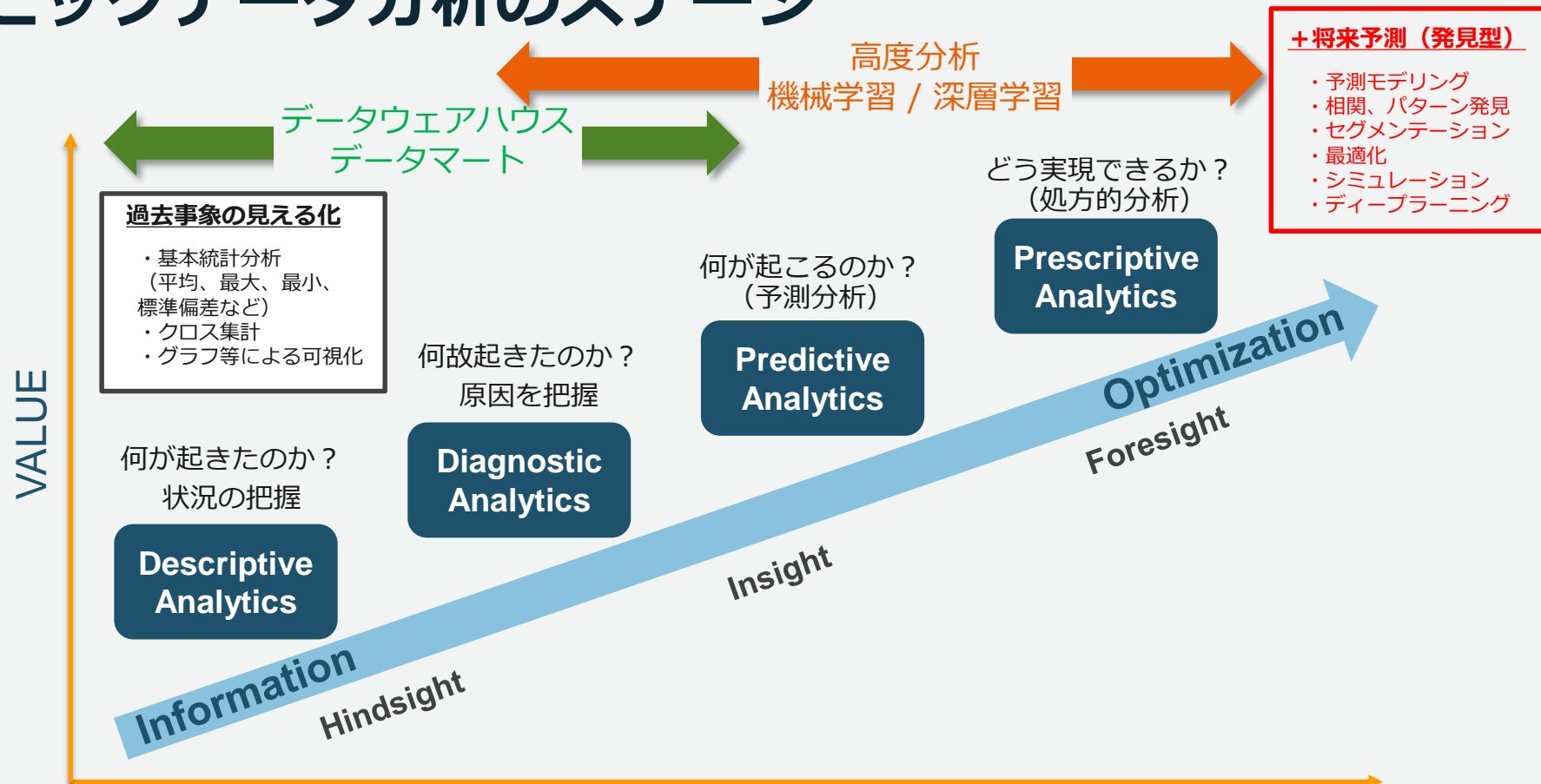
## 機械学習/深層学習など

Java, Scala, R, Pythonなどが主体  
予測分析、傾向分析、推論導出  
(Schema on Read)

- ⑤分析結果の可視化、レポート
- ④データ形式変換、分析の実行
- ③求められる分析に対する仮説化、理論化
- ②収集されているデータを観察、パターン化
- ①要件にかかわらず関連データ収集

## ボトムアップアプローチ (帰納的)

# ビッグデータ分析のステージ



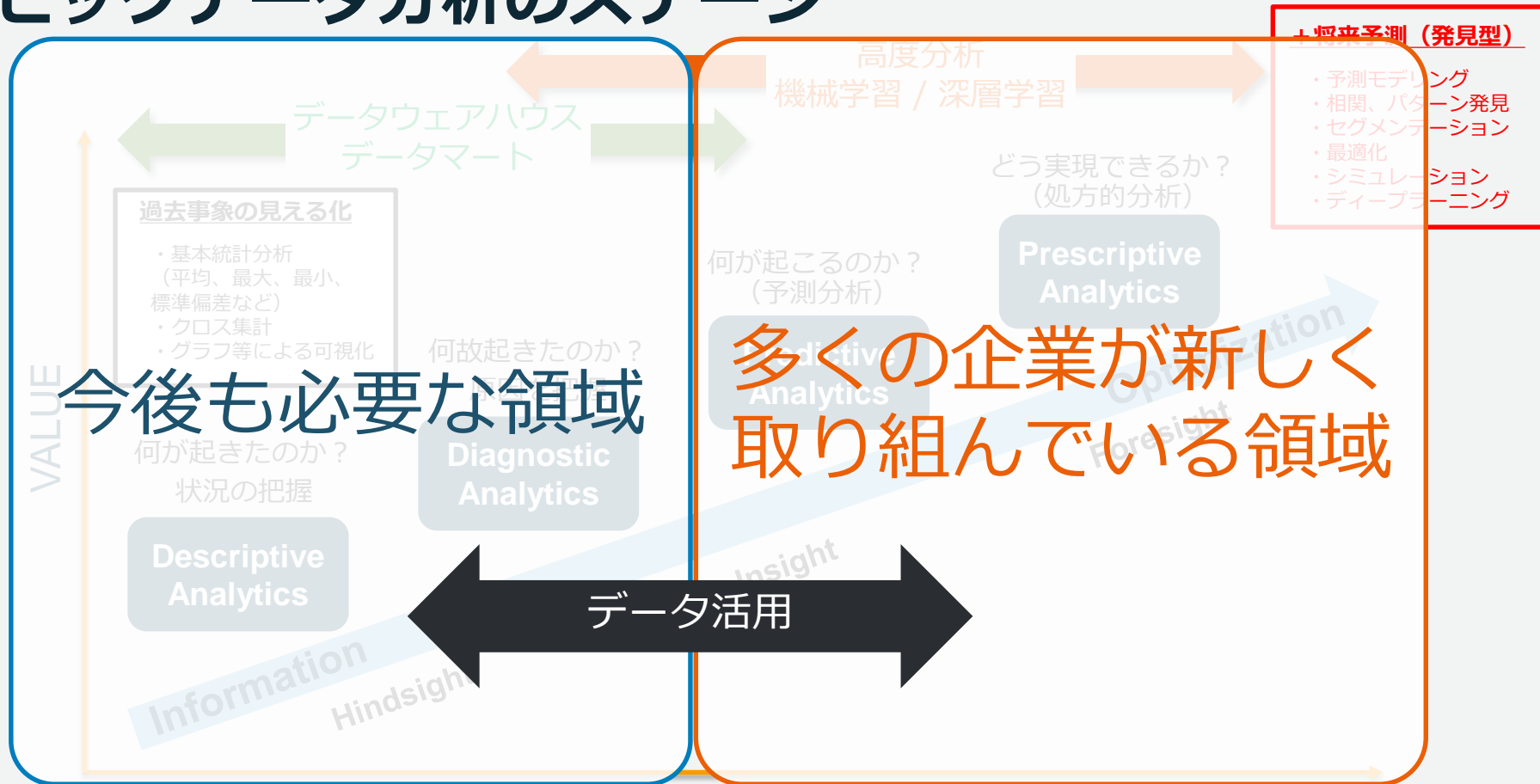
Source: Gartner

© 2018, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

DIFFICULTY



# ビッグデータ分析のステージ



Source: Gartner

© 2018, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

DIFFICULTY



# ビッグデータ基盤とは？

- データ活用サイクル全体を管理できる新しいツール群で構成
- 技術的にもコスト的にも実現可能な基盤である必要あり



クラウドで実現するビッグデータ基盤

収集

蓄積

処理／分析

可視化

# ビッグデータ基盤とは？

- データ活用サイクル全体を管理できる新しいツール群で構成
- 技術的にもコスト的にも実現可能な基盤である必要あり

AWS

クラウドで実現するビッグデータ基盤

収集

データ  
レイク

処理／分析

可視化

# 統制の取れていないビッグデータの取組みでは

## 複数のデータウェアハウス基盤



## 複数のHadoop系基盤や統計分析基盤



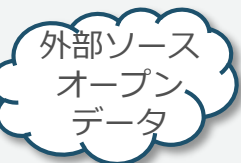
## 複数の分析可視化基盤 (BI)



- 経理・購買・人事 (ERP)
- CRM/顧客管理
- 生産情報、SCM、ロジ etc.



- ログファイル
- ストリームデータ
- レポート
- 監視カメラ映像 etc.



- デモグラフィック
- 経済指標・為替
- 気象情報
- 地理情報
- ソーシャル etc.

# 統制の取れていないビッグデータの取組みでは

複数のデータウェアハウス基盤



複数のHadoop系基盤や統計分析基盤



複数の分析可視化基盤 (BI)



- 経理・購買・人事 (ERP)
- CRM/顧客管理
- 生産情報、SCM、etc.



- ログファイル
- ストリームデータ
- レポート
- 監視カメラ映像 etc.



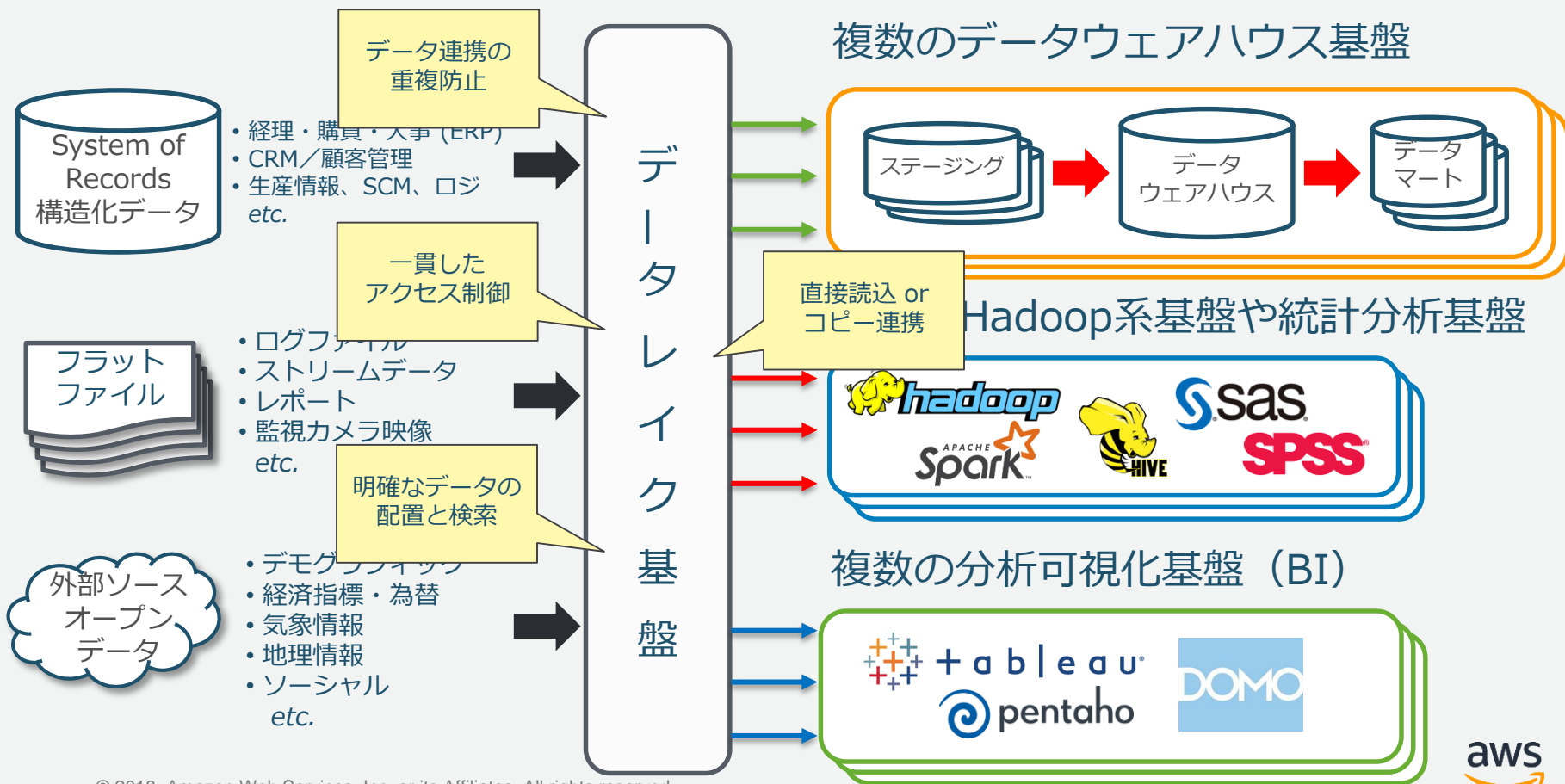
- デモグラフィック
- 経済指標・為替
- 気象情報
- 地理情報
- ソーシャル etc.

ストレージコスト増加

セキュリティの懸念

社内データ探索の負荷大

# データレイクを採用したビッグデータ処理



# データレイク基盤の求められる要件



## 求められる要件

### 解決したい課題

ストレージコスト増加

セキュリティの懸念

社内データ探索の負荷大

エクサバイトクラスの  
データ容量を**安全に格納**

**低コスト**であること

**セキュア**であること

**統制**がとれていること  
(データが**信頼**できる)

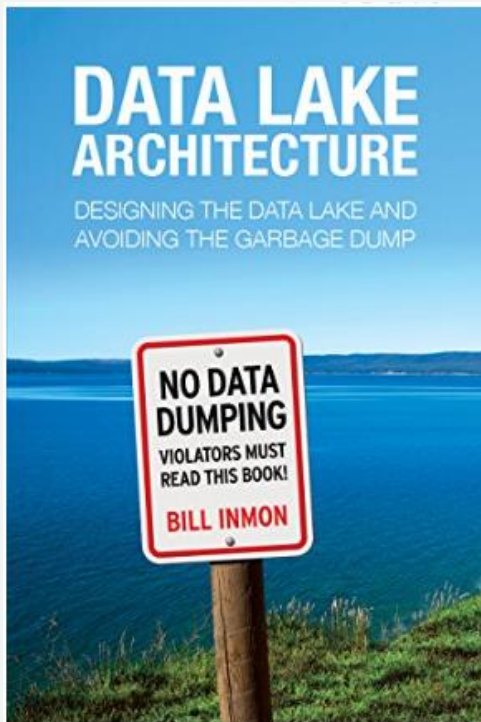
**高速／高性能**であること

該当オブジェクトを  
**容易に探索／再利用**

# Agenda

- データレイクとは何か？なぜ必要なのか？
- データレイクアーキテクチャ
- データレイクを構成するAWSのサービス
- データレイク活用事例

# ご参考) データレイク アーキテクチャ



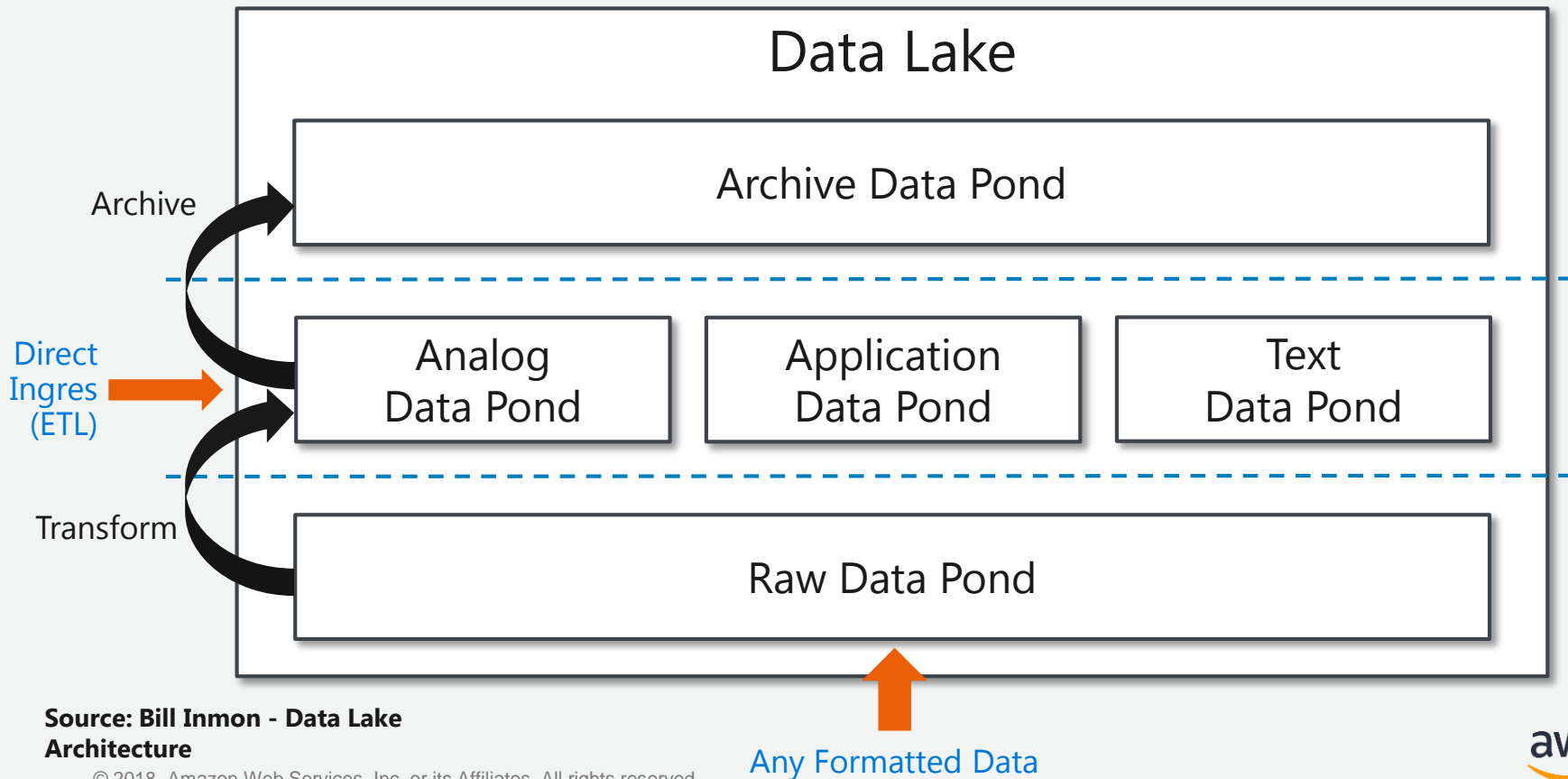
## Data Lake Architecture

Designing the Data Lake and Avoiding the Garbage Dump

著者 : Bill Inmon

データウェアハウスの提唱者とされていて、Ralph Kimballと並んで、データウェアハウスアーキテクチャの第一人者

# データレイクの論理アーキテクチャ



Source: Bill Inmon - Data Lake Architecture

© 2018, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Any Formatted Data



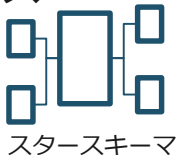
# クラウドベースのビッグデータ基盤 ~ データレイク

## クラウドベースのビッグデータ基盤

### データカタログ

### データウェアハウス

構造化データ  
定型分析用



アーカイブ

アーカイブ

ロード

変換

変換

アーカイブデータ  
Tier 2, DWHのアーカイブ  
Tier 3

半構造化データ  
分析用に加工(csv等)  
Tier 2

非構造化Rawデータ  
(ログ、文書、バイナリ)  
Tier 1

### データレイク

### データカタログ:

- データレイク、データウェアハウスなどのデータオブジェクトのカタログ

### データウェアハウス:

- 大量の構造化データを高速に集約/分析処理を実行

### データレイク: ※前ページの論理アーキテクチャ踏襲

第1層 (Tier 1)

- 非構造化データ/Rawデータ

第2層 (Tier 2)

- 半構造化データ (CSV, Parquetなど)
- Hadoop/Spark、機械学習などから直接利用される

第3層 (Tier 3)

- 半構造化形式のアーカイブデータ

# ビッグデータ基盤

AWS

## ガバナンスとセキュリティ

### 収集

- ETLツール

### 蓄積

アーカイブデータ  
Tier 3

半構造データ  
Tier 2

非構造Rawデータ  
Tier 1

データレイク

### 処理／分析

データウェアハウス

類似・相関分析

統計解析・分析

イメージ／音声／動画分析

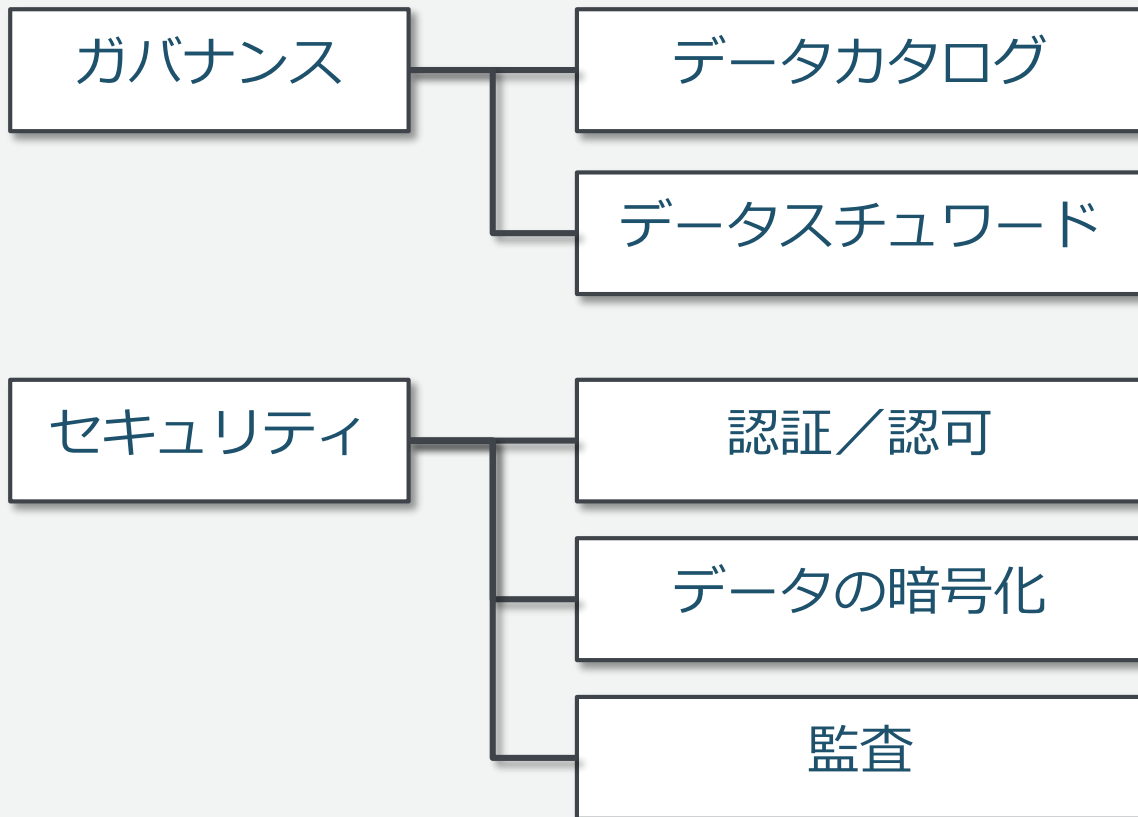
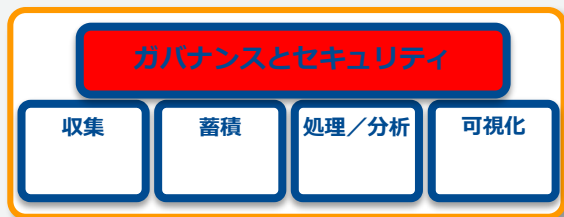
自然言語処理

機械学習

### 可視化

- BI ツール
- 定型レポート  
(帳票)
- ダッシュボード

# ガバナンスとセキュリティ



# ガバナンス：データレイクを「ゴミ溜め」にしないために

## データスチュワード



### 組織内のデータの統制を保つ特殊な任務を担当

- データの内容（コンテンツ）、メタデータを共に管理
- メタデータ内のデータエレメントの定義を明確にする
- データカタログ内のエレメント間で矛盾を発生させない
- データレイク、周辺データのアクセス権限管理を統括

## データレイク

- データの健全性を保つ
- データ連携の定義管理
- 利用者サポート

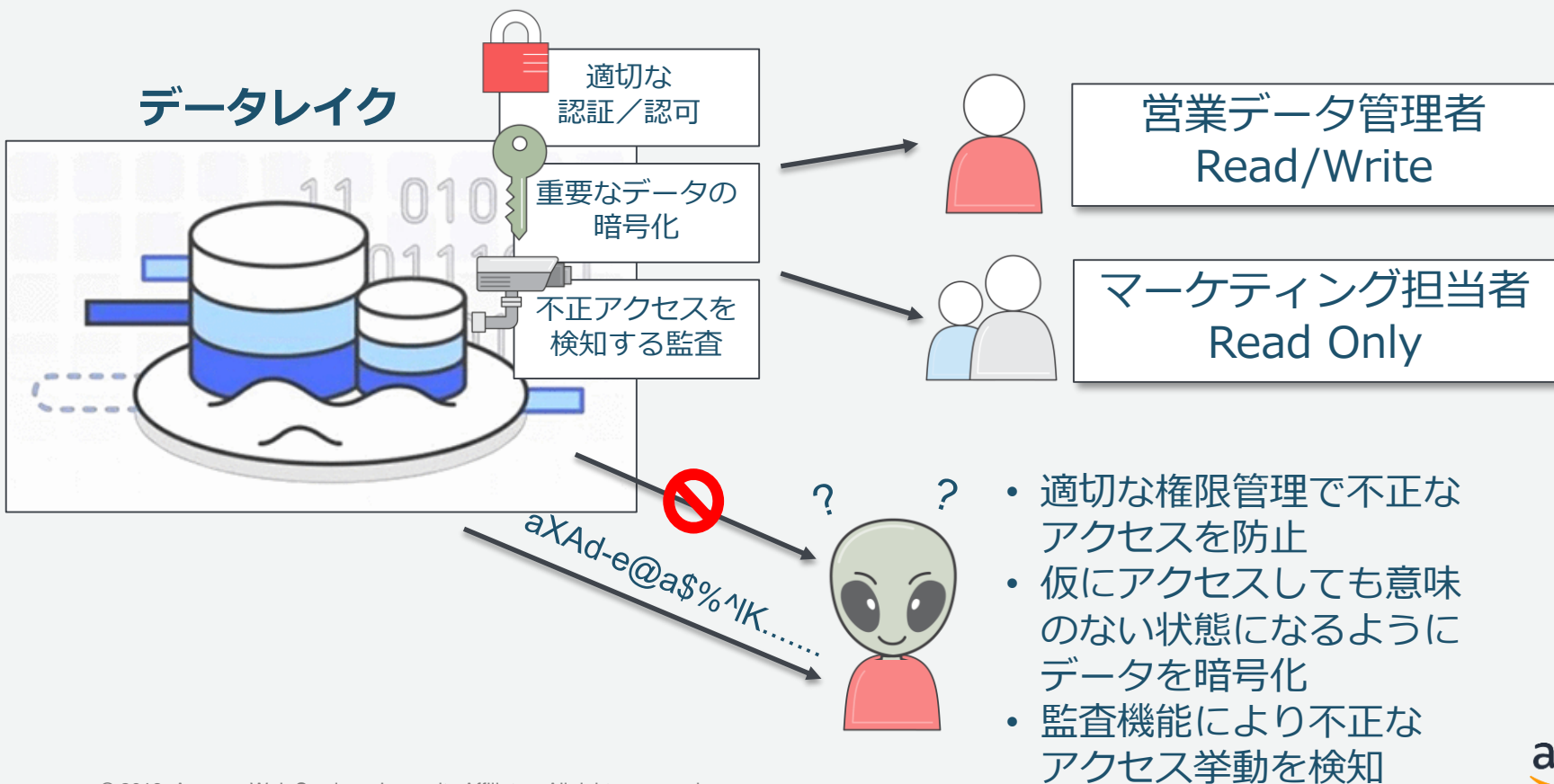


データカタログ管理  
(メタデータ管理)



アクセス権限管理

# セキュリティ：データレイクから重要な情報を漏洩させない



# Agenda

- データレイクとは何か？なぜ必要なのか？
- データレイクアーキテクチャ
- データレイクを構成するAWSのサービス
- データレイク活用事例

# AWSサービス：ビッグデータ領域

## セキュリティとガバナンス



データカタログ



## 収集



## 保存



## 分析



## 可視化



収集

保存

分析

可視化



Amazon  
S3

# Amazon S3

ビッグデータを安全に格納する  
データレイク基盤の中心

# Amazon S3によるデータレイク実現のメリット

## • 大容量

- 上限無し：サイジング不要
- 様々な種類のデータも格納可能

## • 強固なセキュリティ

- IAMを利用した高い認証・認可レベル
- 暗号化設定可能

## • 安定した性能

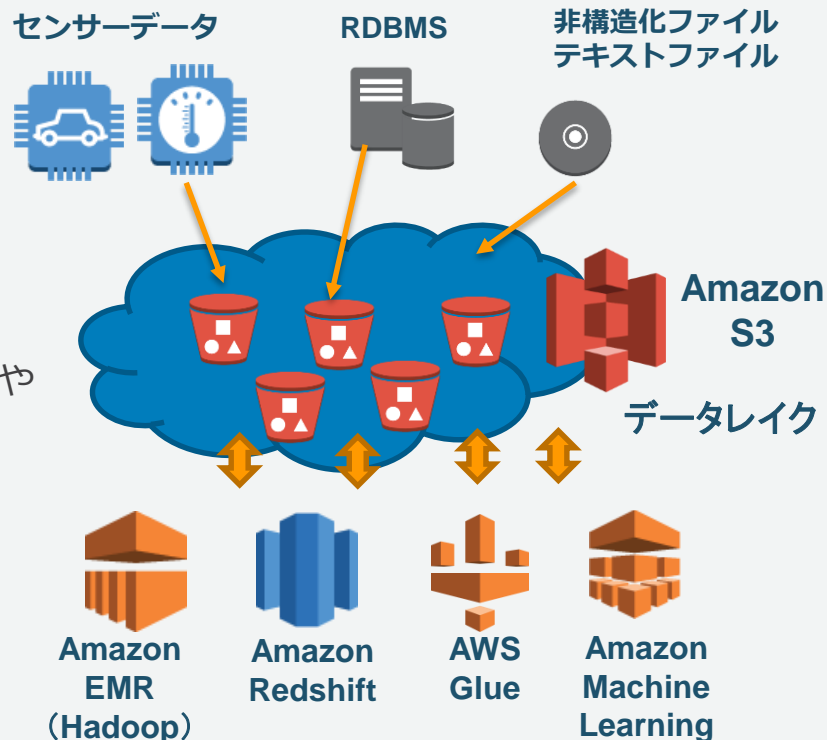
- データ容量に依存しない性能（サーバ台数、媒体本数やRAID、RAIDコントローラを考慮する必要がない）

## • 高信頼性

- 高い耐久性：99.999999999% ※複数AZにコピー
- バージョニング、リージョン間コピー設定可能

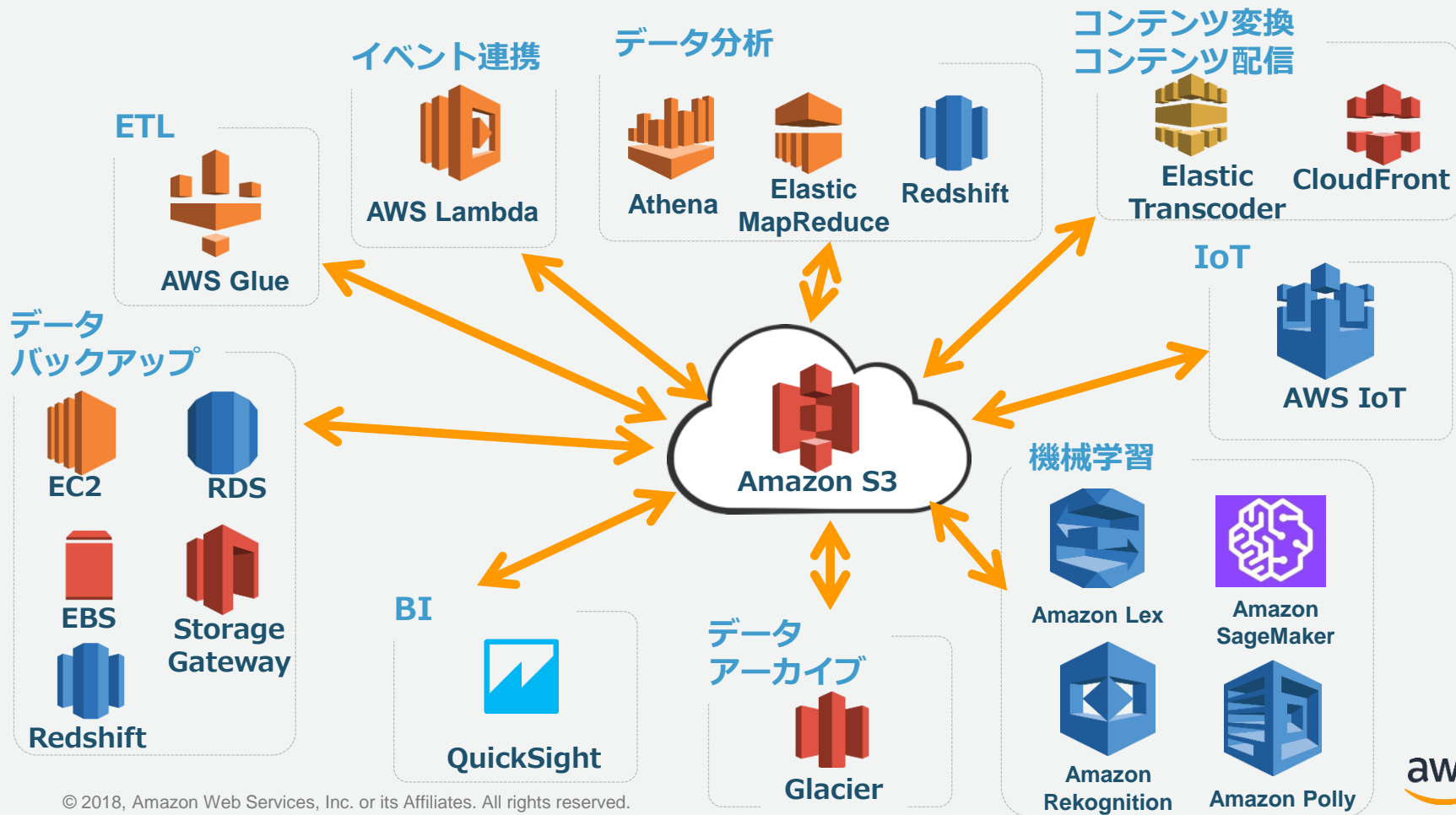
## • 安価：例)10TBの保存で約2.1万円/月\*\*

- \$0.025/GB/月\*（スタンダード）
- \$0.019/GB/月\*（標準-低頻度アクセス）



\* 費用は2018年4月時点での東京リージョンでの価格です  
\*\* 1USドル = 110円で、標準-低頻度アクセスでの試算

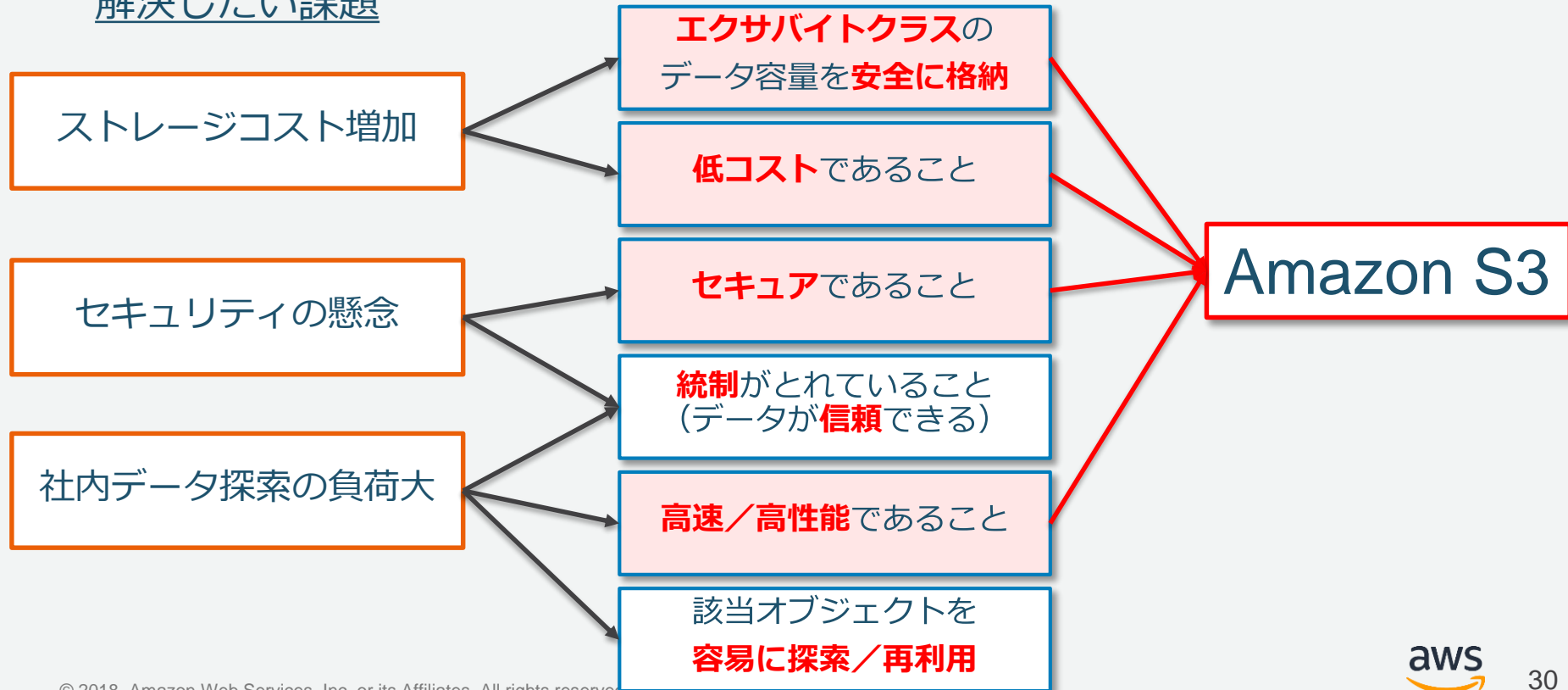
# Amazon S3 : 様々なAWSのサービスと連携



# データレイク基盤の求められる要件の多くをS3でカバー

## 求められる要件

### 解決したい課題



# AWS Glue

データレイク基盤、分析基盤への  
ETL、プリプロセス処理を行い  
連携したデータをカタログ化する

セキュリティとガバナンス



収集

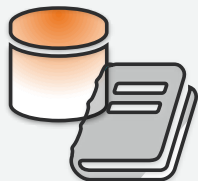


保存

分析

可視化

# AWS Glueの構成要素



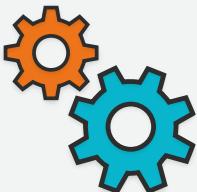
データカタログ

- Hiveメタストア互換のデータソース用メタデータリポジトリ
- テーブル、データ型、パーティションフォーマットを推測するためにソースを**クロールしてカタログ自動生成**
- **Redshift Spectrum, Athena, EMRのカタログとしても利用・連携可能**



ジョブオーサリング

- ETL処理のためのPythonコード(PySpark)、またはScalaを生成
- 任意のIDE（統合開発環境）ツールでコードを作成可能



オーケストレーション

- **分散処理**でジョブを実行
- **サーバレス** - 利用したリソース分だけの支払い

# AWS Glue データカタログ (Data catalog)

表のメタデータをHiveメタストアで管理

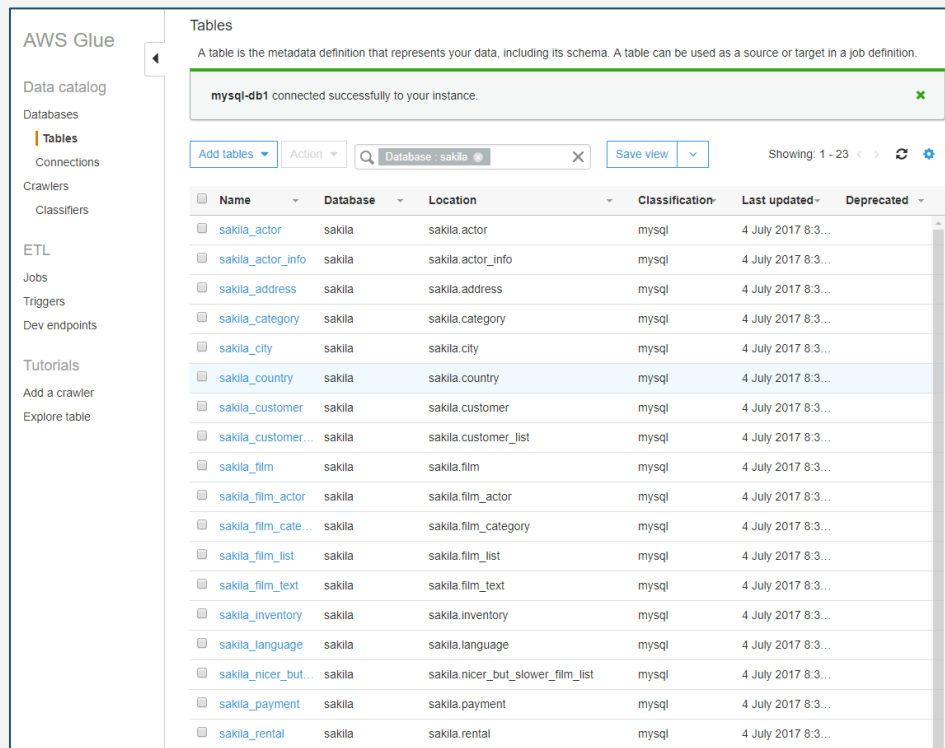
## メタデータ

- 列・プロパティ・型
- データロケーション (URI)
- 更新情報 等

クローラーによる自動チェックと登録

- Hiveパーティションを認識し登録を自動化

```
/mydata  
/year=2017  
/month=11/...
```



The screenshot displays the AWS Glue Data Catalog interface. On the left is a navigation sidebar with options like 'Data catalog', 'Databases', 'Tables', 'Connections', 'Crawlers', 'Classifiers', 'ETL', 'Jobs', 'Triggers', 'Dev endpoints', 'Tutorials', 'Add a crawler', and 'Explore table'. The main area shows a 'Tables' view for the 'sakila' database. A message at the top indicates 'mysql-db1 connected successfully to your instance.' Below this is a search bar with 'Database: sakila' and a 'Save view' button. The table below lists various tables with columns for Name, Database, Location, Classification, Last updated, and Deprecated. The table 'sakila\_country' is highlighted.

Name	Database	Location	Classification	Last updated	Deprecated
sakila_actor	sakila	sakila.actor	mysql	4 July 2017 8:3...	
sakila_actor_info	sakila	sakila.actor_info	mysql	4 July 2017 8:3...	
sakila_address	sakila	sakila.address	mysql	4 July 2017 8:3...	
sakila_category	sakila	sakila.category	mysql	4 July 2017 8:3...	
sakila_city	sakila	sakila.city	mysql	4 July 2017 8:3...	
sakila_country	sakila	sakila.country	mysql	4 July 2017 8:3...	
sakila_customer	sakila	sakila.customer	mysql	4 July 2017 8:3...	
sakila_customer...	sakila	sakila.customer_list	mysql	4 July 2017 8:3...	
sakila_film	sakila	sakila.film	mysql	4 July 2017 8:3...	
sakila_film_actor	sakila	sakila.film_actor	mysql	4 July 2017 8:3...	
sakila_film_cate...	sakila	sakila.film_category	mysql	4 July 2017 8:3...	
sakila_film_list	sakila	sakila.film_list	mysql	4 July 2017 8:3...	
sakila_film_text	sakila	sakila.film_text	mysql	4 July 2017 8:3...	
sakila_inventory	sakila	sakila.inventory	mysql	4 July 2017 8:3...	
sakila_language	sakila	sakila.language	mysql	4 July 2017 8:3...	
sakila_nicer_but...	sakila	sakila.nicer_but_slower_film_list	mysql	4 July 2017 8:3...	
sakila_payment	sakila	sakila.payment	mysql	4 July 2017 8:3...	
sakila_rental	sakila	sakila.rental	mysql	4 July 2017 8:3...	

# データカタログのメタデータ内容の例

テーブルの  
主要情報

テーブルの  
プロパティ

テーブル  
スキーマ

The screenshot shows the AWS Glue console interface for a table named 'ny\_pub'. The left sidebar contains navigation options like 'Data catalog', 'Databases', 'Tables', 'Connections', etc. The main area displays the table's metadata, including its name, description, database, classification, location, and last updated date. Below this, there are sections for 'Properties' and 'Schema'.

**Table Metadata:**

- Name: ny\_pub
- Description: nytaxianalysis
- Database: nytaxianalysis
- Classification: parquet
- Location: s3://.../canonical/NY-Pub/
- Connection: No
- Deprecated: No
- Last updated: Tue Jul 25 09:49:20 GMT-400 2017

**Properties:**

- sizeKey: 32032431007
- objectCount: 19
- UPDATED\_BY\_CRAWLER: RidesCrawler
- CrawlerSchemaSerializerVersion: 1.0
- recordCount: 2870781820
- averageRecordSize: 11
- CrawlerSchemaDeserializerVersion: 1.0
- compressionType: none
- typeOfData: file

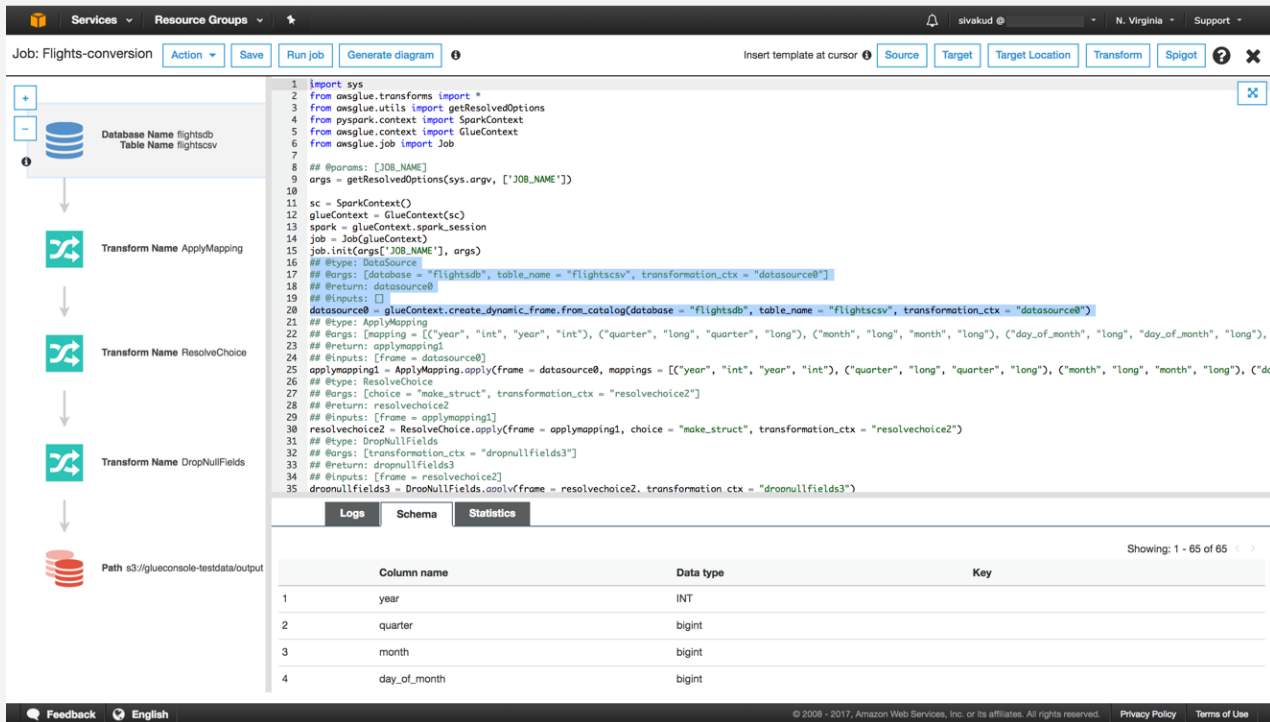
**Schema:**

	Column name	Data type	Key
1	vendorid	string	
2	pickup_datetime	bigint	
3	dropoff_datetime	bigint	
4	ratecode	int	
5	passenger_count	int	
6	trip_distance	double	
7	fare_amount	double	
8	total_amount	double	
9	payment_type	int	
10	year	string	Partition (0)
11	month	string	Partition (1)
12	type	string	Partition (2)

テーブル  
パーティション



# ジョブの生成 : Glue ETLによるデータ変換の自動化



```
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 = "flightsdb", table_name = "flightscsv", transformation_ctx = "datasource0"]
18 ## @return: DataSource0
19 ## @inputs: []
20 datasource0 = glueContext.create_dynamic_frame_from_catalog(database = "flightsdb", table_name = "flightscsv", transformation_ctx = "datasource0")
21 ## @type: ApplyMapping
22 ## @args: [mapping = [{"year", "int", "year", "int"}, {"quarter", "long", "quarter", "long"}, {"month", "long", "month", "long"}, {"day_of_month", "long", "day_of_month", "long"}]]
23 ## @return: applymapping1
24 ## @inputs: [frame = datasource0]
25 applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [{"year", "int", "year", "int"}, {"quarter", "long", "quarter", "long"}, {"month", "long", "month", "long"}, {"day_of_month", "long", "day_of_month", "long"}])
26 ## @type: ResolveChoice
27 ## @args: [choice = "make_struct", transformation_ctx = "resolvechoice2"]
28 ## @return: resolvechoice2
29 ## @inputs: [frame = applymapping1]
30 resolvechoice2 = ResolveChoice.apply(frame = applymapping1, choice = "make_struct", transformation_ctx = "resolvechoice2")
31 ## @type: DropNullFields
32 ## @args: [transformation_ctx = "dropnullfields3"]
33 ## @return: dropnullfields3
34 ## @inputs: [frame = resolvechoice2]
35 dropnullfields3 = DropNullFields.apply(frame = resolvechoice2, transformation_ctx = "dropnullfields3")
```

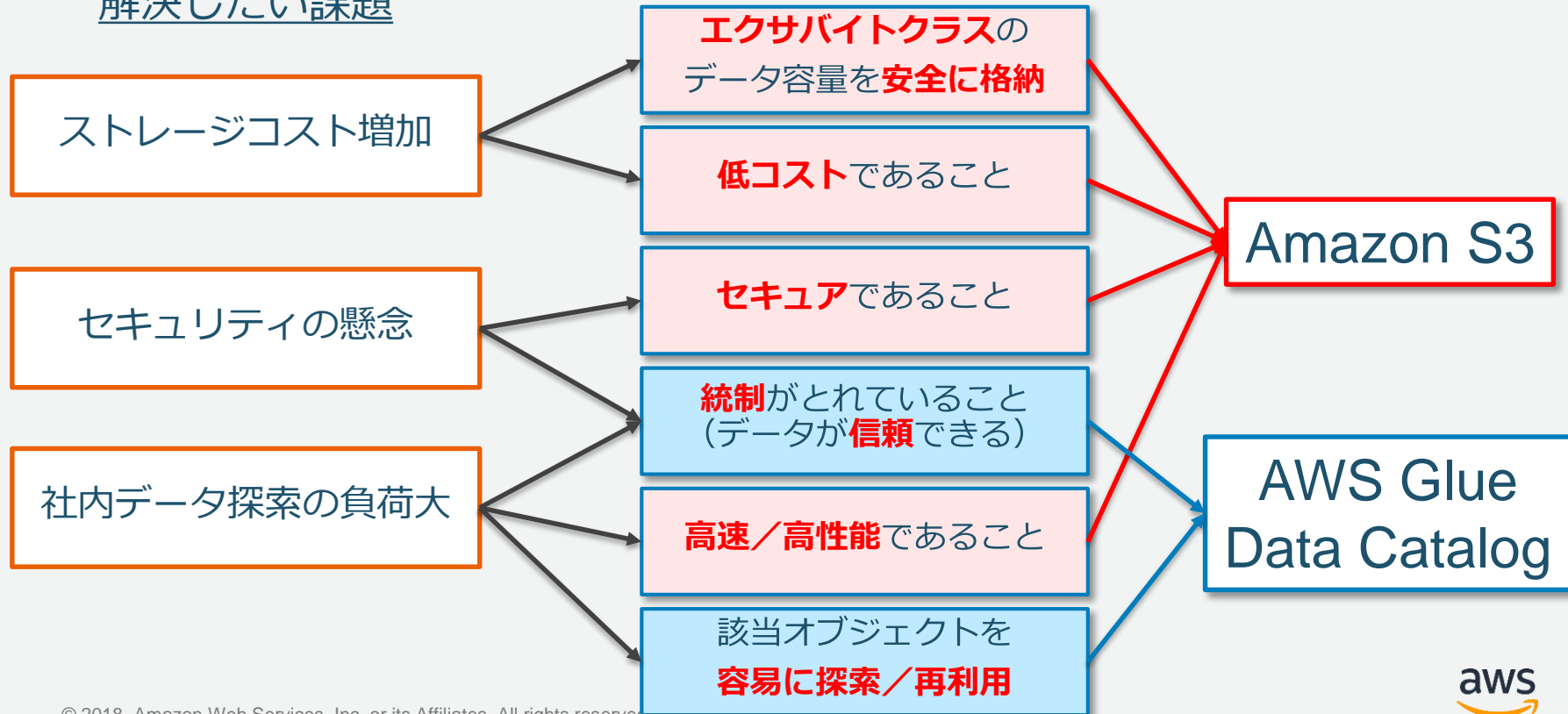
Column name	Data type	Key
1 year	INT	
2 quarter	bigint	
3 month	bigint	
4 day_of_month	bigint	

- サーバ管理不要（サーバレス）
- リポジトリとクローラー機能
- ジョブの雛形が生成され、独自の変換処理をPySpark、またはScalaで記述可能

# データレイク基盤の求められる要件の多くを S3 + Glueで実現

## 求められる要件

### 解決したい課題



収集

保存

分析

可視化



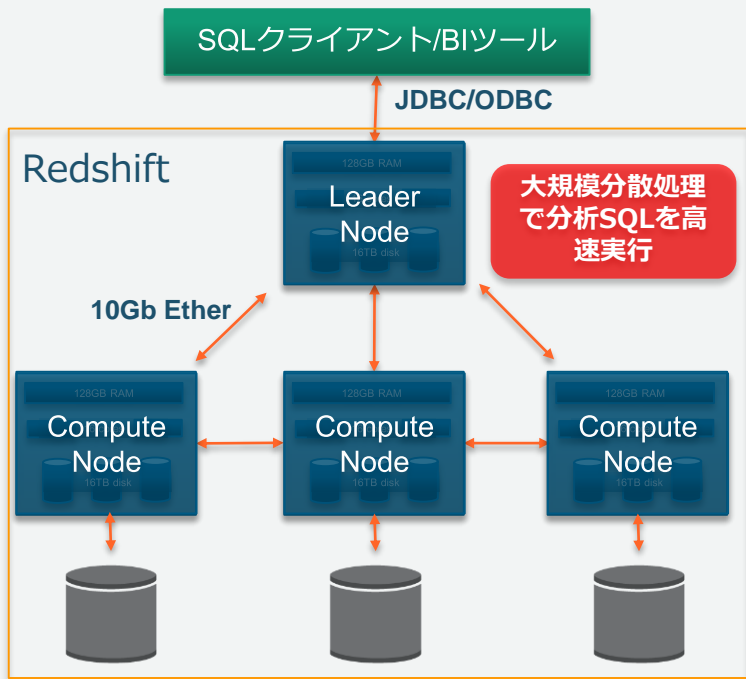
Amazon  
Redshift

# Amazon Redshift

大量の構造化データを  
高速に分析・集計する  
データウェアハウス基盤

# Amazon Redshift

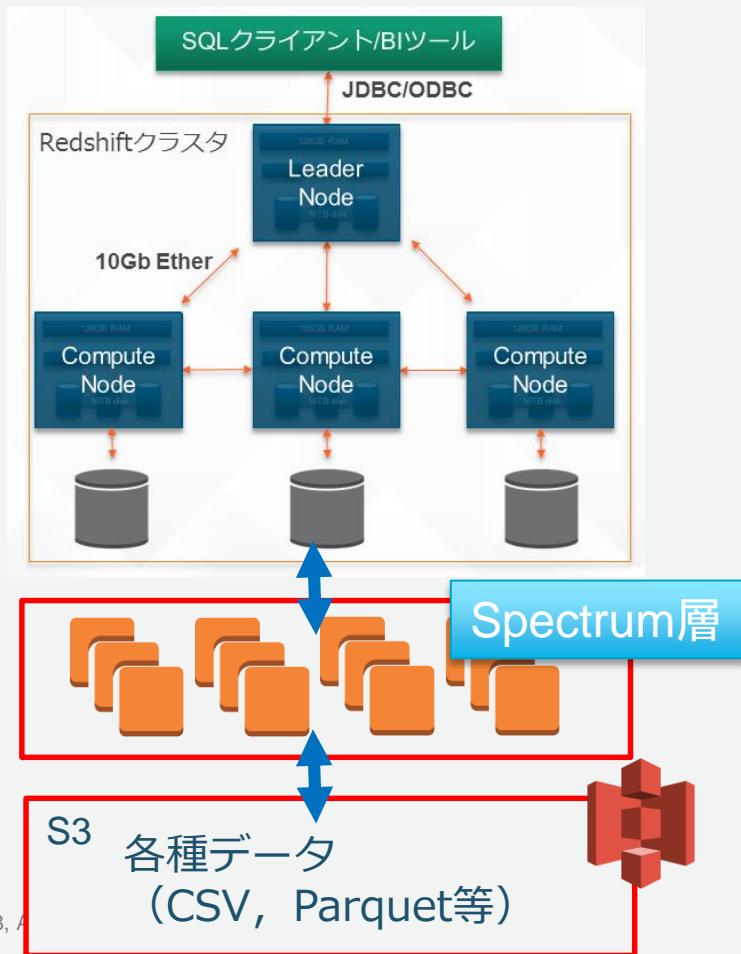
## フルマネージドのデータウェアハウスサービス



### 特徴

- データサイズ：最大2PBまで拡張可能
- 超並列(MPP)、カラムナ型DBエンジンによる高速SQL処理
- スケールアウト可能。最大128台
- PostgreSQLとの互換性
- 使った分だけの利用料金。従来のデータウェアハウスの1/10のコストで実現
- 管理機能がビルトイン

# Redshift Spectrum



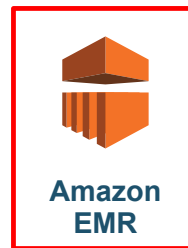
- Redshiftが拡張され、S3上に置いたファイルを外部テーブルとして直接参照可能に
- Spectrum層により、S3上のファイルを高速処理
- ローカルディスク上のデータと組み合わせてSQLでクエリ可能
- 多様なファイルフォーマットに対応
  - CSV, TSV, Parquet, SequenceFile, RCFile, ORC, RegexSerDe 等

収集

保存

分析

可視化



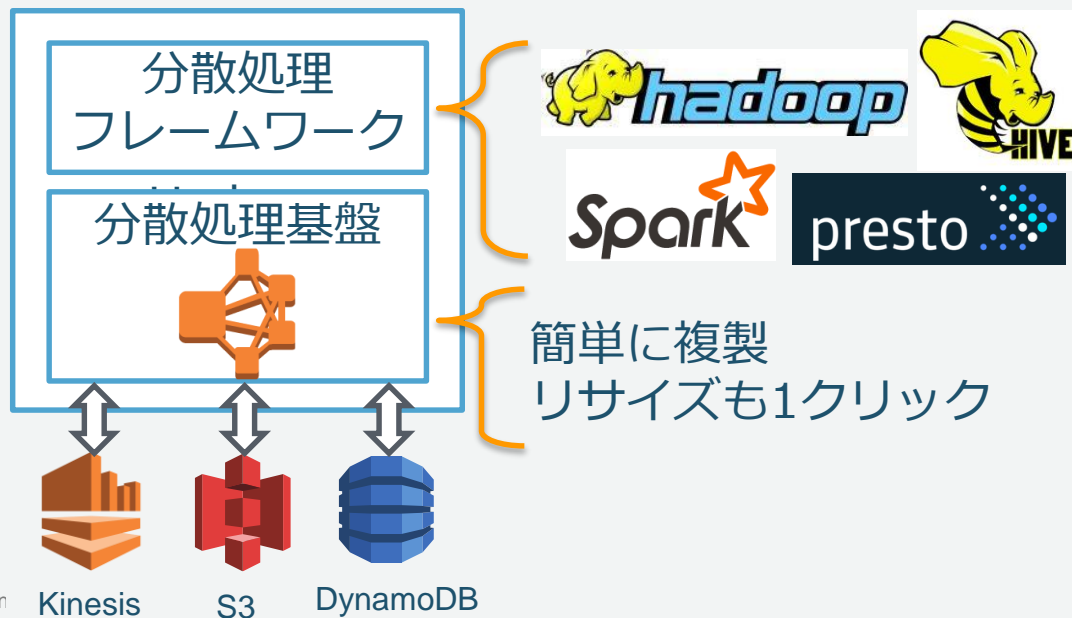
# Amazon EMR

データレイク基盤にアクセスして  
ログ分析、データ変換、機械学習、  
シミュレーションなどを高速実行する

# Amazon Elastic MapReduce (EMR)

Hadoop/Sparkなどの大規模分散処理環境を数クリックで構築

- 簡単スタート：数クリックでセットアップ完了
- 低コスト：従量課金、必要な時間だけクラスターを稼働

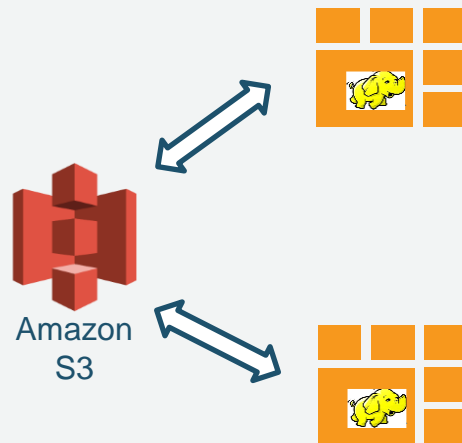




# EMRFS: S3をHDFSの様に扱う

“s3://” と指定するだけでHDFSと同様にS3にアクセス

- 計算ノードとストレージを分離できる
  - コスト面でもメリット大
- クラスターのシャットダウンが可能
  - クラスターを消してもデータをロストしない
- 複数クラスター間でデータ共有が簡単
- データの高い耐久性 (S3)



データレイクに直接並列でアクセスすることが可能

セキュリティとガバナンス

収集

保存

分析

可視化



Amazon  
Athena

# Amazon Athena

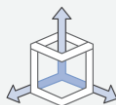
データレイクに直接アクセスし  
SQLによってインタラクティブに  
検索分析実行する



# Athena の特徴



サーバレスでインフラ管理の必要なし



大規模データに対しても高速なクエリ



事前のデータロードなしに S3 に直接クエリ



スキャンしたデータに対しての従量課金



JDBC 経由で BI ツールから直接クエリ

# マネジメントコンソールから簡単にアクセス

- 利用開始から数分でクエリを投げられる
- クエリごとのスキャンデータ量を確認可能
- Glue データカタログでテーブルスキーマやパーティションを確認

History

Search for name, query, etc.

Query submitted time	Query	Encryption type	State	Run time(s)	Data scanned	Action
2017/08/25 15:16:47 UTC+9	CREATE EXTERNAL TABLE IF NOT EXISTS awslogs.cloudfront_raw ( date STRING, time STRING, xEdgeLo...	N/A	SUCCEEDED	1.17	0KB	Download results
2017/08/25 15:15:47 UTC+9	drop table cloudfront_raw	N/A	SUCCEEDED	23.79	0KB	Download results
2017/08/25 00:10:18 UTC+9	SELECT "date", "time", "xedgeLocation", "scbytes", "cip", "csmethod", "cshost", "csuristen", "scstat..."	N/A	SUCCEEDED	53.34	15.62MB	Download results
2017/08/25 00:05:17 UTC+9	SELECT "timestamp", "version", "account", "interfaceId", "sourceAddress", "destinationAddress", "sou..."	N/A	SUCCEEDED	3.52	2.28MB	Download results
2017/08/25 00:00:49 UTC+9	SELECT "timestamp", "xEdgeRegion", "xContentSetting", "xEdgeLocation", "scBytes", "cip", "csMethod", ...	N/A	SUCCEEDED	14.12	15.03MB	Download results
2017/08/25 00:00:29 UTC+9	SELECT "type", "timestamp", "elb", "clientHost", "clientPort", "targetHost", "targetPort", "request..."	N/A	FAILED	2.5	66.81KB	Error details
2017/08/24 00:12:49 UTC+9	SELECT "date", "time", "xedgeLocation", "scbytes", "cip", "csmethod", "cshost", "csuristen", "scstat..."	N/A	SUCCEEDED	52.45	15.72MB	Download results
2017/08/24 00:10:21 UTC+9	SELECT "date", "time", "xedgeLocation", "scbytes", "cip", "csmethod", "cshost", "csuristen", "scstat..."	N/A	SUCCEEDED	58.25	15.72MB	Download results
2017/08/24 00:05:10 UTC+9	SELECT "timestamp", "version", "account", "interfaceId", "sourceAddress", "destinationAddress", "sou..."	N/A	SUCCEEDED	3.04	2.28MB	Download results
2017/08/24 00:00:50 UTC+9	SELECT "timestamp", "xEdgeRegion", "xContentSetting", "xEdgeLocation", "scBytes", "cip", "csMethod", ...	N/A	SUCCEEDED	14.1	15.03MB	Download results
2017/08/24 00:00:30 UTC+9	SELECT "type", "timestamp", "elb", "clientHost", "clientPort", "targetHost", "targetPort", "request..."	N/A	FAILED	2.67	231.24KB	Error details

Tables > cloudfront\_raw

Last updated: 25 Aug 2017 Table Version (Current)

[Edit table](#) [Delete table](#) [View partitions](#) [Compare versions](#)

Name	cloudfront_raw
Description	
Database	awslogs
Classification	Unknown
Location	s3://aws-cloudfront-logs-athena-666254511816/raw
Connection	
Deprecated	No
Last updated	Fri Aug 25 15:16:48 GMT+900 2017
Input format	org.apache.hadoop.mapred.TextInputFormat
Output format	org.apache.hadoop.hive.qi.io.HiveIgnoreKeyTextOutputFormat
Serde serialization lib	org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
Serde parameters	field.delim serialization.format
Table properties	EXTERNAL TRUE transient_lastDdlTime 1503641808

Schema

Showing: 1 - 27 of 27

	Column name	Data type	Key
1	date	string	
2	time	string	
3	xedgeLocation	string	



# Amazon Athena

S3に保存したファイルを直接クエリー。管理は不要

S3上に保存したファイル  
にSQLを実行可能に

PrestoベースでANSI SQL  
対応

サーバ管理は不要で  
ファイルを置いてSQLを  
書くだけ

```
1 select
2   class
3   , sex
4   , total_cnt
5   , survived_cnt
6   , round(cast(survived_cnt as double)/cast(total_cnt as double), 2) as survival_rate
7 from (
8   select
9     class
10    , sex
11    , count(survived) as total_cnt
12    , sum(survived) as survived_cnt
13 from
14   titanic_db.titanic
15 where
16   class != '+'
17 group by
18   class
19   , sex
20 )
21 order by survival_rate desc;
```

Run Query Save As Format Query New Query (Run time: 1.9 seconds, Data scanned: 25.43KB)

Results

	class	sex	total_cnt	survived_cnt
1	1st	female	143	134
2	2nd	female	107	94
3	3rd	female	212	80
4	1st	male	179	59

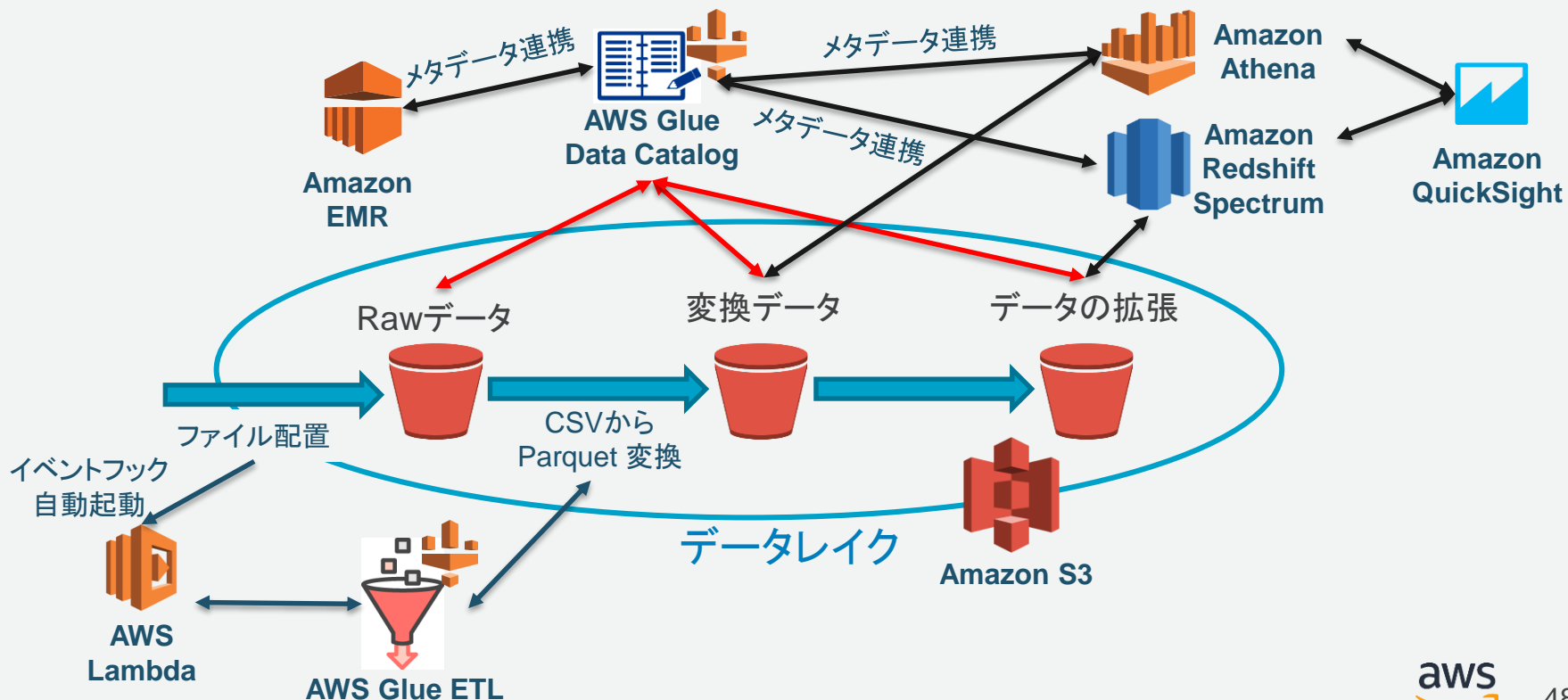
# データ分析プラットフォーム比較

	Amazon Redshift	Amazon EMR	Amazon Athena
分類	大規模データ処理に特化したマネージドRDBMS	Hadoop/Spark等分散処理フレームワークのマネージド環境	SQL特化のSQL処理環境（Prestoベース）。サーバ管理が不要
処理系	SQL（PostgreSQLと互換性）	アプリケーションに依存：Hive、Spark等 Presto等でSQLでの分析も可能	SQL（Prestoベース）
スケールアウト	可能（手動） ※Spectrum層は自動	可能（手動）	<b>可能（自動）</b>
ストレージの種類	<b>高速なローカルストレージ+S3上データへのクエリ</b>	HDFS、もしくは <b>EMRFSでS3上のデータを直接アクセス（ロードなし）</b>	<b>S3上のデータを直接アクセス</b>
ノードとストレージの分離	ノードとストレージは同時に増加・削減。 <b>S3上のデータはノードと関係なく増減可能</b>	<b>ストレージに影響を与えずにノード増減が可能（EMRFSの場合）</b>	<b>ストレージとノードは分離され、ユーザはノード管理自体不要</b>
最大処理サイズ	2PB+S3（上限無し）	S3（上限無し）※EMRFSの場合	S3（上限無し）
運用管理	<b>運用管理に必要な機能がビルトイン</b>	分散処理系+OSSアプリ環境を容易に導入可能	<b>運用が不要</b>
分析ツール、ETL等	<b>JDBC/ODBC + SQL プッシュバック等BIアプリがネイティブレベルで対応</b>	JDBC/ODBC経由もしくはHiveメタストア経由で多くの環境がサポート	JDBC経由で多くの環境がサポート。 API呼び出しも可能

# データレイクと各サービスの連動イメージ

AWSブログ: AWS GlueとAmazon S3を使用してデータレイクの基礎を構築する

<https://aws.amazon.com/jp/blogs/news/build-a-data-lake-foundation-with-aws-glue-and-amazon-s3/>



# Agenda

- データレイクとは何か？なぜ必要なのか？
- データレイクアーキテクチャ
- データレイクを構成するAWSのサービス
- データレイク活用事例

# Amazon.comの事例

本章は以下で公開されているre:invent 2017発表資料を元に作成しています

"A Look Under the Hood – How Amazon.com Uses AWS Services for Analytics at Massive Scale" (ABD329)

<https://www.youtube.com/watch?v=IWK96BNx3Qk>

<https://www.slideshare.net/AmazonWebServices/a-look-under-the-hood-how-amazoncom-uses-aws-services-for-analytics-at-massive-scale-abd329-reinvent-2017>



# Amazon.comのデータウェアハウス

多数の系列会社・部門で  
分析・レポートに利用される  
データウェアハウス

多様で膨大なデータ

- コアデータ：圧縮後で5PB+
- 全ストレージ：35PB+
- 100億行+ロード/日
- 20万クエリ/日



# データウェアハウスの利用状況と特長

データウェアハウス利用：2,300チーム

チームごとの表の数

- 最小1～最大598
- 平均：49

どんなチームも部分的に必要な  
データを活用するため、  
全部入のDWHは無駄が多い

分析環境にはスケーラビリティが  
必須要件

## “Datamarts”

Number of Teams using the DW: ~2300

Number of Tables Used per Team:

- Max: 598
- Min 1
- Average: 49

Ad-Hoc (any data any time) can be achieved via  
EMR can access the Data in Andes Directly  
Redshift can load data into the Redshift file  
system, or it can use the Spectrum Feature to  
directly access the Data in Andes

An Architecture that **Scales with the Business**



Amazon Internal Team (132 Tables)

**AWS**  
**re:Invent**

© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



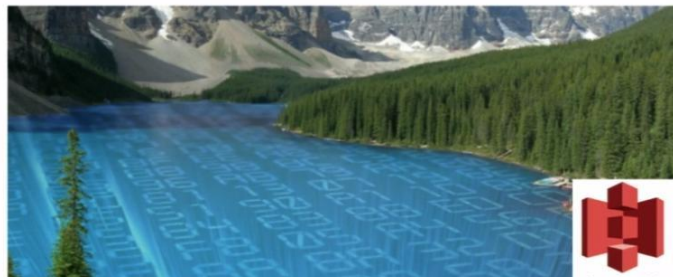
# Amazonのデータレイクプロジェクト "ANDES"

- アマゾンのデータを集めておく「THE（正式な）」場所  
"THE" Place for Data at Amazon
- S3がデータレイクの核

## ポイント)

- データ提供元のチームはデータレイクにデータを配置し、分析チームにアクセス権限を付与
- EMR (Hadoop系処理) は、Andesから 並列で直接データレイクにアクセス可能
- RedshiftはAndesから 並列でアクセスしてデータをロード、またはSpectrumにより 並列で直接アクセス可能

The Amazon "Data Lake" – Project Name "Andes"



The Goal: "THE" Place for Data at Amazon

- Source teams (**Data Producers**) put their Public Data there to give access to Analytic teams (**Data Consumers**) and to share private data within their team
- EMR Can Directly Access the Data in Parallel from Andes
- Redshift can load the data in Parallel from Andes, or it Can Directly Access the Data in Parallel with Spectrum

**aws**  
**re:Invent**

© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



# 課題と解決： 調整からセルフサービスへ

## [課題]

- どこにデータがあるか分からない
  - ✓ どこに有るか**問い合わせる**必要がある
- データを利用可能にするまでが大変
  - 自分達の環境にコピーしてもらうには、データ転送やETL等の仕組みの**調整**と構築するための**申請（予算化）**が必要
- データ鮮度が不明
  - データの更新頻度の調整に**開発**が必要



## [解決]

- **利用者が**検索出来るようにする
- **利用者が**必要なデータを自分の領域に「登録 (Subscribe)」出来るようにする
- **利用者が**更新頻度の調整等を設定・変更出来るようにする

# セルフサービスを実現するための仕組み

- 収集と後続処理が分離され、収集側がデータレイクにデータを置くだけで良い構成
- セルフサービスを促進するために「発見」「登録」層を導入



# データレイクポータルによる「発見」の実現

検索機能

## Table Subscriptions - The Vision

部署やデータタイプ  
による分類

The screenshot displays the Amazon Table Subscriptions interface. At the top, there is a search bar with the text 'Table Subscriptions' and a magnifying glass icon. Below the search bar, there are navigation links for 'Departments', 'Browsing History', 'My Subscriptions', 'My Clusters', and 'Help'. The main content area shows a grid of subscription cards. Each card includes the subscription name, the creator (e.g., 'by BOOKER'), the update frequency, storage size, and load time. For example, one card is 'D\_MP\_ASI' with 765 GB and a 6-minute load time. Another is 'O\_VENDI' with 156 GB and a 5 GB daily update. A sidebar on the left lists various categories for filtering, such as 'Daily Aem Activity', 'Remedy', 'Associate Shipment Analysis', etc. At the bottom left, there is an 'Avg. Customer Review' section with star ratings and 'Up' indicators.

各種情報を提供

- 登録者情報
- 詳細情報（登録者が記入）
- スキーマ
- サポートレベル
- 評価（今後の実装）

※この図はアイデア検討時のモックアップであり、実際とは異なります

AWS re:Invent

© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws

# 「登録」によるセルフサービスのデータ取得

## 関係者との調整や開発不要

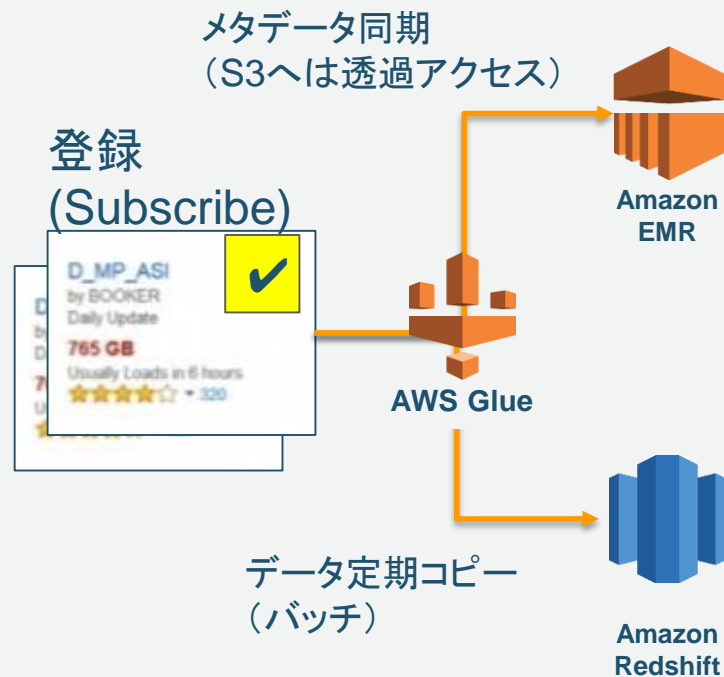
欲しいデータを発見したら、登録  
(Subscribe) する



登録時にはデータの行き先を指定  
(自部門のRedshiftやEMR等)



自動的にコピー処理やメタデータ  
同期処理が構築され、定期的に  
フレッシュなデータが供給される



# 「登録」モデルの実現とその効果

ユーザ調整が可能な設計

- 必要なデータ範囲や頻度を設定可能
- 独自のクエリを登録して、データ連携のタイミングで自動実行

自動的なバリデーション（表行数チェック、スキーマチェック等）

誰が何を使っているか把握できるため、データの削除や変更時にも**影響範囲が把握できる**という効果あり

Where

ID = ABC and ...

Timing

▼ Once a day : 3AM

User Query

INSERT INTO ..  
SELECT ...

When ...

▼ After data copy

# 本章のまとめ: Amazon.comのデータレイク

## 徹底したセルフサービス化でデータ活用を促進



- 小売の経験を活かしたカタログ機能
  - ✓ メタ情報定義、利用者の評価など
- データレイクからの連携処理を手続き型での処理で自動化

## Andesの現在の状態 :

- データレイクに20,000表があり同期されている
- データ連携先はRedshiftとEMRで計900+クラスター


### Andes – Current State

- We have the data!
  - 20k+ Tables maintained in Andes – All Active Tables have been Sourced from the Enterprise Data Warehouse
  - Many teams are adding new data sets!
- Have Onboarded 900+ Redshift and EMR systems to Subscriptions
  - 20,000+ tables being synchronized
- Usage off the Legacy DW
  - Three years (2014-2016) to grow from 0 to 100k Jobs each Day
  - In 2017, has grown from 100k to 300k Jobs each Day



Amazon.com  
Big Data  
Technologies

**aws re:Invent**  
© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



# まとめ

# 本セッション全体のまとめ

- データレイクは、複数のデータウェアハウス、高度分析環境（機械学習など）が存在する場合の無駄、重複を最小化
- データレイクをゴミ溜めにしないためには
  - ✓ データスチュワード / データカタログ / セキュリティ
- AWSのサービスで構築するデータレイク
  - ✓ 代表的なサービス : S3 / Glue / Redshift / Athena / EMR
- Amazon.comの事例
  - ✓ Discover / Subscription を採用


# 参考資料

- Amazon S3
  - <http://aws.amazon.com/jp/s3/>
- AWS Glue
  - <http://aws.amazon.com/jp/glue/>
- Amazon EMR
  - <http://aws.amazon.com/jp/emr/>
- Amazon Athena
  - <http://aws.amazon.com/jp/athena/>
- Amazon Web Services ブログ : 「AWS Glue と Amazon S3 を使用してデータレイクの基礎を構築する」
  - <http://aws.amazon.com/jp/blogs/news/build-a-data-lake-foundation-with-aws-glue-and-amazon-s3/>

# オンラインセミナー資料の配置場所

## AWS クラウドサービス活用資料集

- <https://aws.amazon.com/jp/aws-jp-introduction/>

			
<b>サービス別資料</b>	<b>ソリューション別資料</b>	<b>業種別資料</b>	<b>その他の資料</b>
無料オンラインセミナー「Black Belt Online Seminar」のサービスカット資料他、AWSのTechメンバーによる各サービスの解説資料がご覧いただけます。	無料オンラインセミナー「Black Belt Online Seminar」のソリューションカット資料他、特定のソリューションについてのAWS活用方法がご覧いただけます。	無料オンラインセミナー「Black Belt Online Seminar」のインダストリーカット資料他、特定の業界のユースケースがご覧いただけます。	イベントに関する資料やアップデート情報などをご覧いただけます。

## Amazon Web Services ブログ

- 最新の情報、セミナー中のQ&A等が掲載されています。
- <https://aws.amazon.com/jp/blogs/news/>

# 公式Twitter/Facebook AWSの最新情報をお届けします



@awscloud\_jp



検索

もしくは

<http://on.fb.me/1vR8yWm>

最新技術情報、イベント情報、お役立ち情報、  
お得なキャンペーン情報などを日々更新しています！