

AWSとGCPのいいとこどりでつくる 分析基盤のきほん

@chie8842
chie hayashida



DevFest
Tokyo 2017

自己紹介

Chie Hayashida

Twitter: @chie8842

GitHub: chie8842

Retty.Inc

Software Engineer



#GCP #AWS #機械学習 #Python #Scala #Clojure #DB #vim
#焼肉 #ピアノ #テニス #スノボ

クラウドフル活用で
大規模分析基盤を超短期間で
構築した事例を共有します。

今日話す範囲

進め方
そのものの
話

アプリレイヤの話

基盤レイヤの話



この話をします。

分析基盤構築の背景

- Retty入社初日

とりあえず、分析基盤つくって。
2ヶ月で！

え、分析基盤？

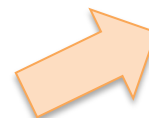
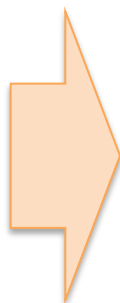
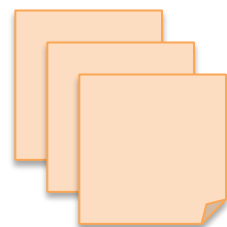
マネージャ

わたし（入社初日、
肩書き：データ
サイエンティスト）

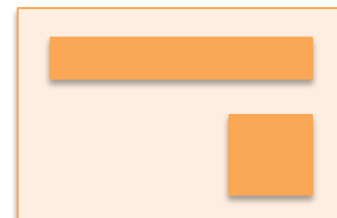
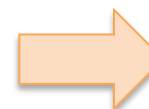


そもそも分析基盤とは？

- データを蓄積・活用するための基盤



施策の評価



アドテク



レコメンド

もともとあった分析基盤の課題①

■ DWHのテーブル設計の問題

例

- 不要なログがログ全体の9割
- 適切なデータ型が使われていない
- jsonオブジェクトがテキスト形式で入っている

分析しづらい
(アドホック分析の度に
複雑な正規表現抽出)

ストレージ容量逼迫

クエリ実行時に過大なサーバ
リソースが必要



もともとあった分析基盤の課題②

■ マスタデータは別のDBにある

- マスタデータと突合して分析したい場合は別の環境にデータを移す必要がある
- joinしたいカラム同士でデータ型が異なる

分析者ごとに環境構築

データ転送コスト








もともとあった分析基盤の課題③



■ログ増大に伴うパフォーマンスボトルネック

- 日次バッチが終わらない
- 気軽にアドホック分析できない

→クエリを投げる際はSlackに報告する運用

  10:35 AM
調査のためクエリを投げます。(2-3個ほど)
 さんに確認済みです。
宜しくお願いします。

  信任 11:51 AM
アクセス調査のために幾つかクエリ投げます。
5分で終わらなかったクエリはすぐ止めます。

  🏠 12:13 PM
データ取得のためクエリを投げます。10分以内には収まると思います。

RAI 1

現状整理

- ログサイズ：1日に数十GB（gz.json状態）
→けっこうでかい。これからも増える
- 正規化されていないログ
 - 単純なデータ転送や正規表現抽出ですまない
 - SessionizeもETLでやる
- サービス側の機能追加に伴う要件変更が予想される

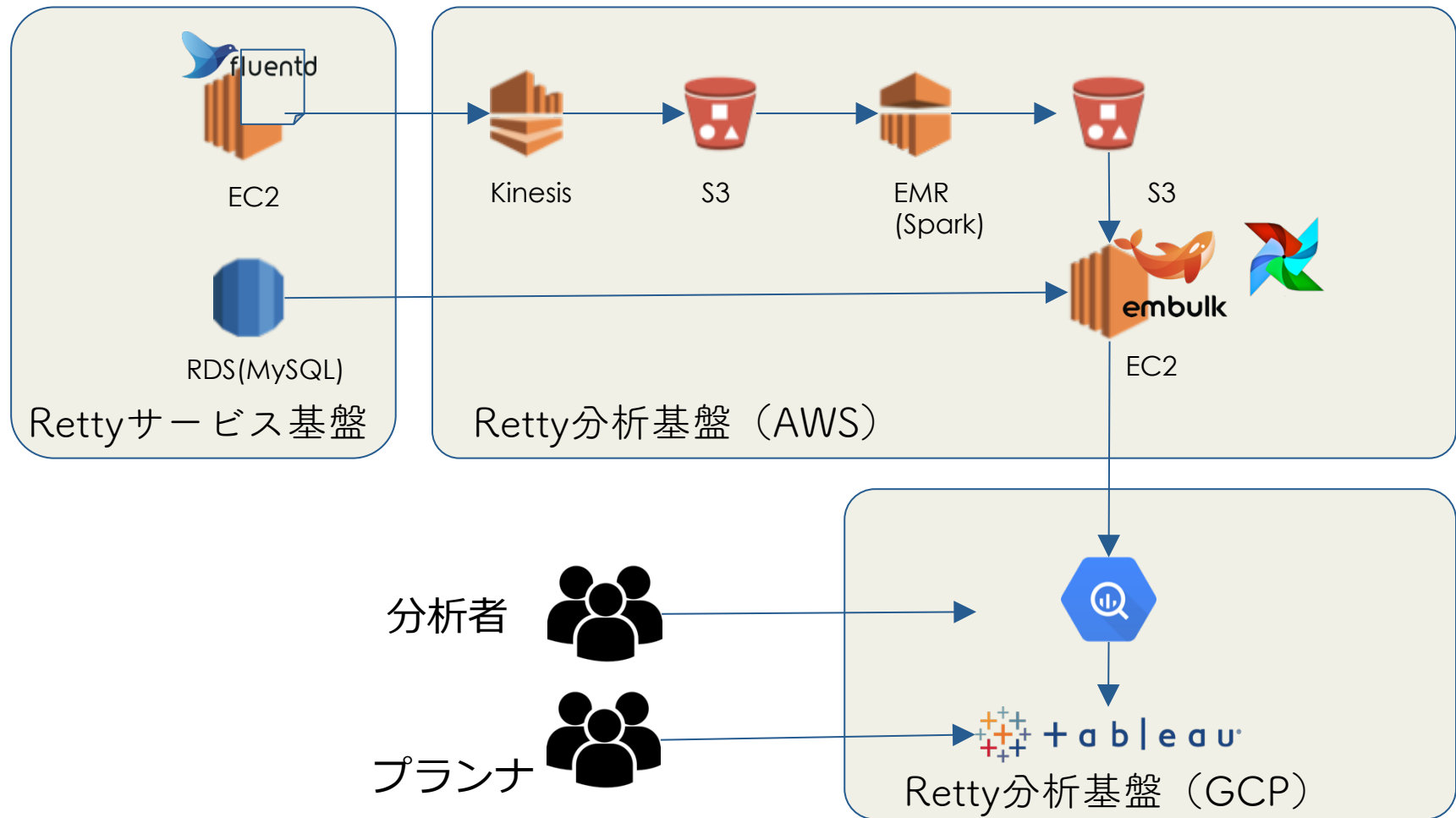
新しい分析基盤に求められるもの

- 分析者にとって使いやすい
 - SQLやそれに準ずるクエリ言語が利用できる
 - レスポンスやスループット
- 追加開発・運用がしやすい
 - 列変更等が柔軟にできる
 - 複雑なETL処理に柔軟に対応できる
- コスト（イニシャル/ランニング）が現実的である
- スケーラブルである
 - 分析対象データの種類やサイズが増えても対応できる

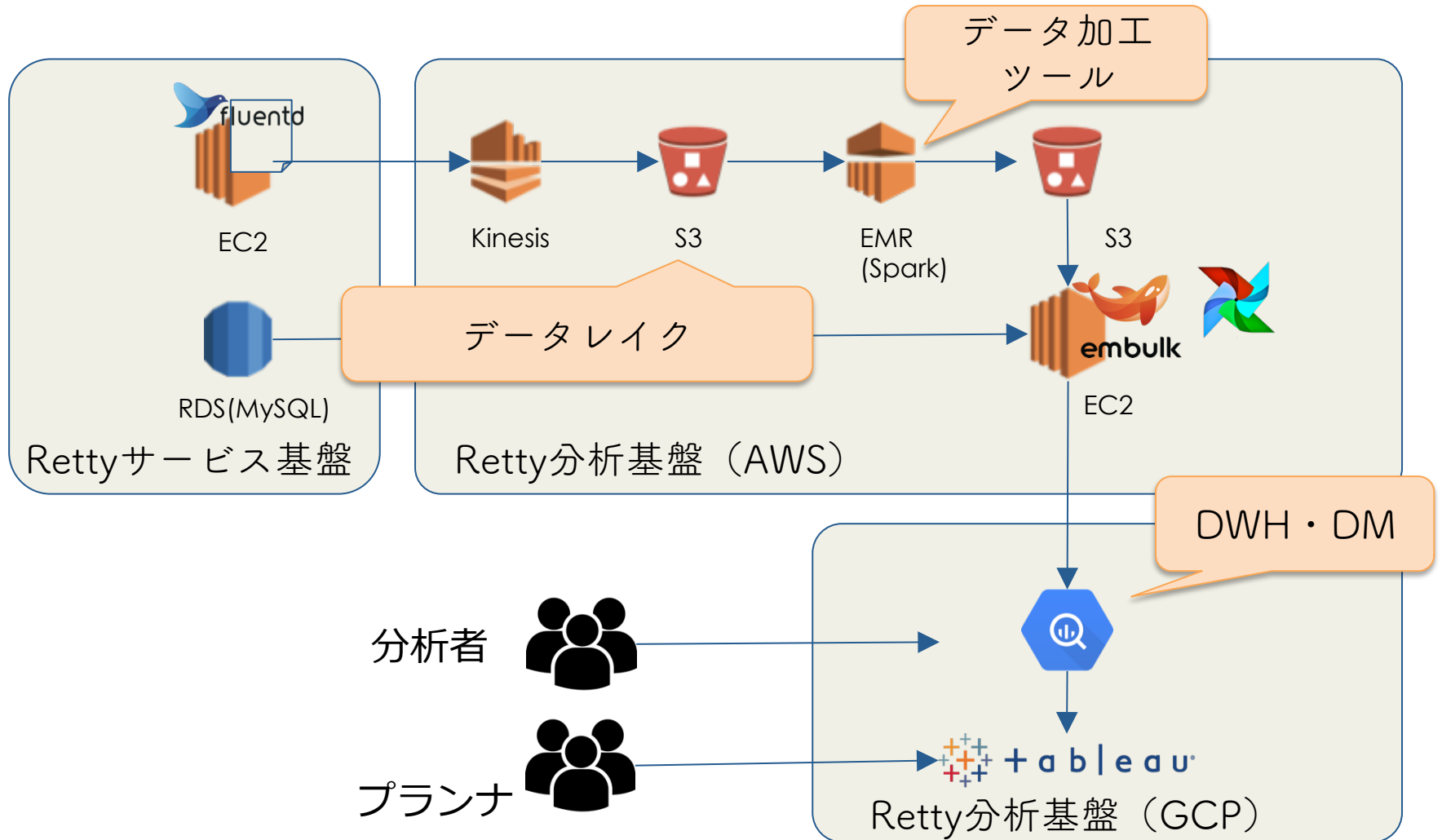


AWSとGCPのいいとこどりした分析基盤

つくった分析基盤



つくった分析基盤



データレイク：S3

- 非構造化データの保存
- サービスの動いている環境（AWS）に近い場所にデータを保持するほうが都合がよい
 - ネットワーク転送コスト
 - 管理しやすさ
- 同じバケット内でもプレフィックスやタグを利用した柔軟なライフサイクルの運用
- Kinesis Firehoseを利用することでかんたんに日時ごとにディレクトリを分けて保存できる



DWH・DM：BigQuery

- 分析者にとって使いやすい
 - StandardSQLが利用できる
 - UDFやWindow関数も使える
 - スプレッドシートやpandas dataframeとの連携
- 後のテーブル設計変更がしやすい
 - テーブルへの列追加ができる
- 安定したレイテンシとスループット
- メンテナンスフリー
- 時間課金でなくクエリ課金
- RedShiftやAthenaを使う場合と比べて、AWSからGCPへのデータ転送が発生するが、運用コストの削減で相殺できる範囲だった



DWH製品比較

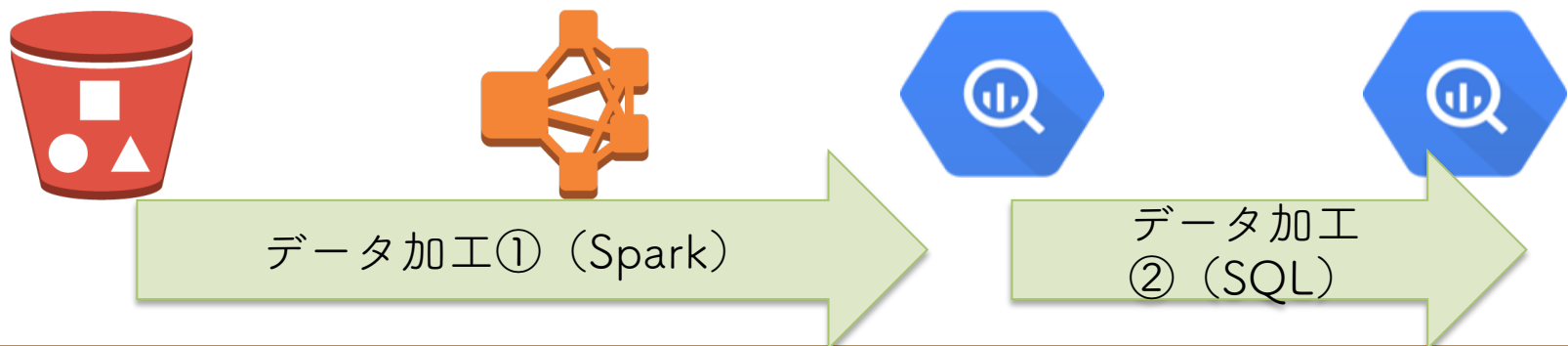
	RedShift	Athena	BigQuery
NW転送コスト	リージョン内	リージョン間転送 (東京→バージニア)	インターネット 越しの転送
課金方式	稼働時間課金	クエリ課金	クエリ課金
UDF	○	×	○
カラム変更	○	×	△
基盤運用	必要	必要	ほとんどなし
クエリ チューニング	必要	必要	ほとんどなし
クエリ言語	standardSQL	presto	standardSQL

2017/05時点

青文字は今回の要件に適していることを示す

EMR(Spark): データ加工

- サービス側のログ設計の関係で、以下が必要だった。
 - 不要なログ出力が全体の9割を占めるため、BigQueryへ転送する前にフィルタ処理
 - SQLで表現できない非構造化データに対する複雑なETL処理
- ログが増大してもクラスター台数を増やすことでスケールできる
- SQLで済ませられるものはBigQuery上で加工



使われる分析基盤構築のコツ

- 早く作っても長く使えないものを作っても意味がない
- DWHの場合、基盤部分は「作って壊して」が簡単にはすまない。
- 基盤部分は慎重に決めた

5月

6月

要件ヒアリング、製品・技術選定、PoC

ETLスクリプト作成・
環境構築

こっちに時間かけた。

ちゃんと使われる分析基盤ができた！

さいごに

- クラウドフル活用で分析基盤を超短期間で作れる！
 - でも1人でやるのはつらかった。色々な意味で。
- クラウドも他の技術も、一つにこだわらず柔軟に活用するの大事！
- 今回触れなかった技術選定の詳しい部分やアプリケーションレイヤーの話とかを10/20のX-Tech JAWSで発表する予定なので、興味ある方は是非。

用語

- データレイク
 - 加工前の生ログを保存する場所
- DWH
 - 分析しやすいように加工されたデータを格納するデータベース
- DM
 - 分析用途に応じて集計後のデータなどを格納するなど、サンドボックス的につかうためのデータベース
- データ加工ツール
 - ログを分析しやすい形に整形するツール
- ワークフローエンジン
 - 一連のデータ処理のフローを管理するツール