Kinesis Analyticsを使ってみた

2016/11/18

株式会社　NTTドコモ
サービスイノベーション部 ビッグデータ担当
高田　雅人

目次

- 自己紹介
- Kinesisの紹介
- デモンストレーション
- 現時点での評価

自己紹介

- **所属/名前**
  - NTTドコモ　R&Dイノベーション本部
    サービスイノベーション部　ビッグデータ担当
  - 高田　雅人(たかだ　まさと)

- **業務**
  - ビッグデータ処理基盤の調査・活用推進
    - ストリーム処理
  - NW分析

- **好きなAWSサービス**
  - Amazon Kinesis

きっかけ

- 社内のあるデータを収集し、リアルタイム処理を行う
- リアルタイム処理した結果を複数システムに送る必要がある

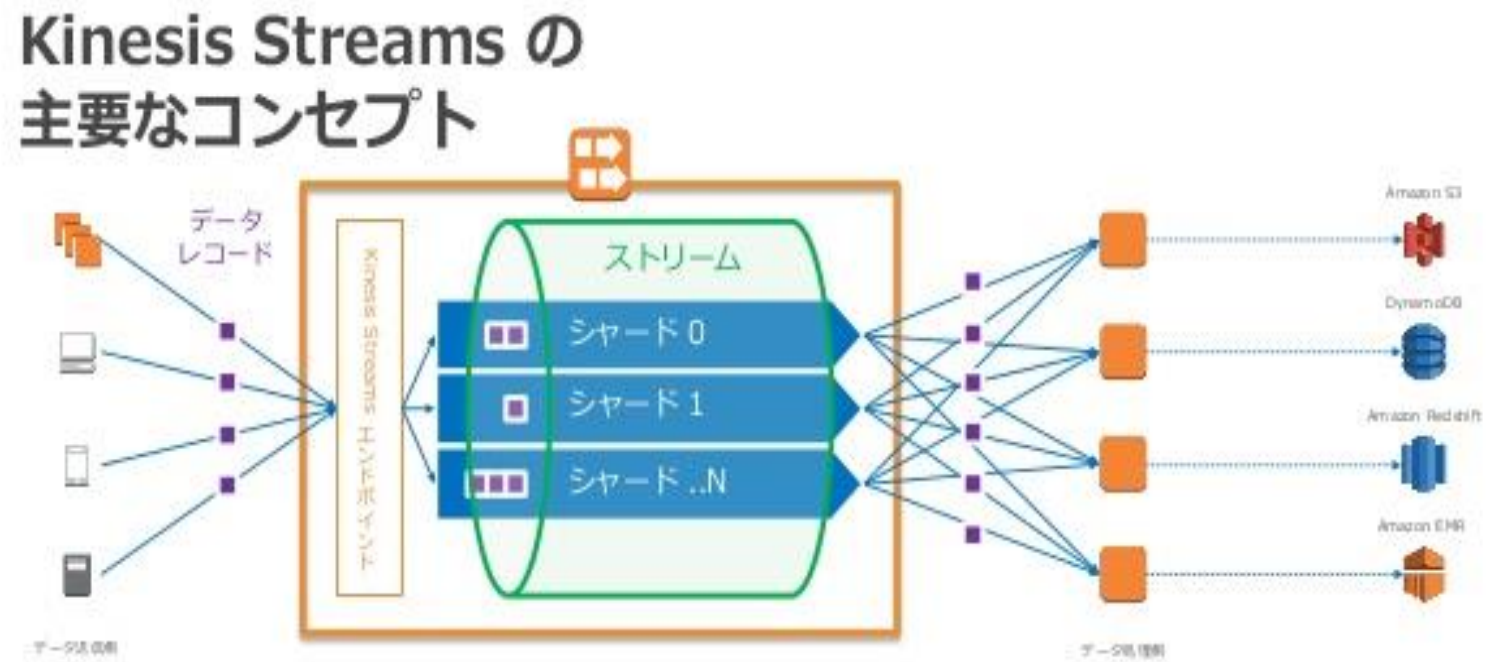
Amazon Kinesis

- **Kinesisは３つサービス**

資料参考：
http://www.slideshare.net/AmazonWebServicesJapan/amazon-kinesis-analytics

Kinesis Streams

- **一言でいえば、マネージドkafka**

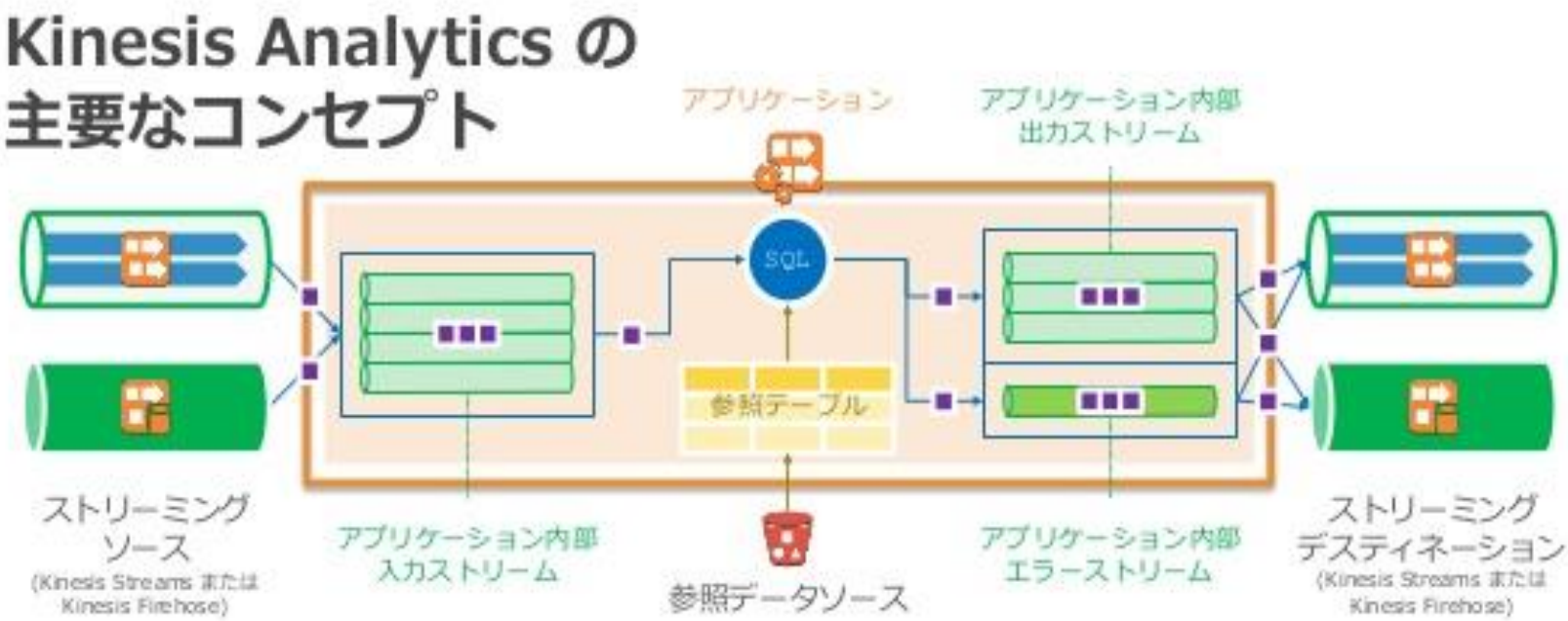


- データの種類や処理の用途に応じて「ストリーム」を作成。ストリームは 1 つ以上の「シャード」で構成
- 保存されるデータの単位を「データレコード」と呼び、保持期間はデフォルトで 24 時間／最長で 7 日間
- 1 データレコードの最大サイズは 1 MB
- データ送信側のキャパシティは 1 シャードあたり秒間 1 MB もしくは 1,000 PUT レコード
- データ処理側のキャパシティは 1 シャードあたり秒間 2 MB もしくは 5 回の読み取りトランザクション
- ストリーム内のシャード数を増減することでスループットをコントロール

資料参考：
http://www.slideshare.net/AmazonWebServicesJapan/amazon-kinesis-analytics

Kinesis Analytics

- **PipelineDBに良く似ている**



- 分析単位に「アプリケーション」を作成し、入力／出力となる「ストリーミングソース／デスティネーション」を設定
- ストリーミングソース／デスティネーションはアプリケーション内部の「入力／出力ストリーム」に対応
- SQL でアプリケーション内部の入力ストリームを分析し、結果を出力ストリームへ出力
- アプリケーション内部ストリームの最大行サイズは 50 KB／参照データソースの最大サイズは 1 GB
- クエリーの複雑さとデータのスループットに応じて処理能力 (KPU – Kinesis Processing Units) を自動伸縮
- 米国東部（バージニア北部）／米国西部（オレゴン）／欧州（アイルランド）リージョンで利用可能

Analyticsのデモを行います

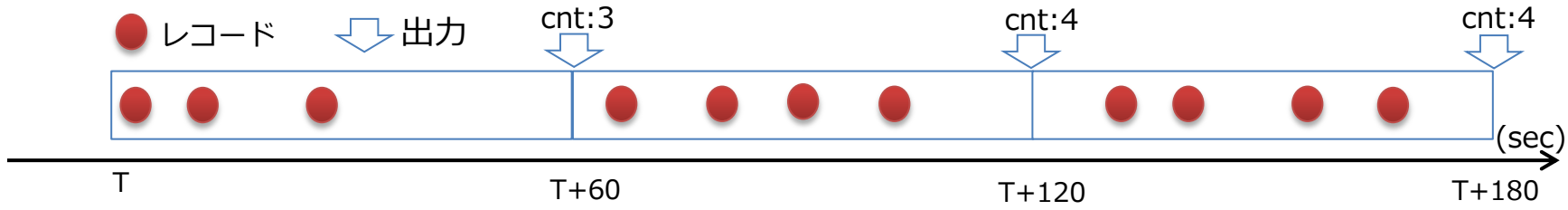
資料参考：
http://www.slideshare.net/AmazonWebServicesJapan/amazon-kinesis-analytics

デモ内容

- **Step1. コンソール画面で構築**
- **Step2. 基本的なSQLクエリ**
  - **CREATE STREAMS**
  - **WHERE**
  - **GROUP BY**
- **Step3. windowを使用したSQLクエリ**

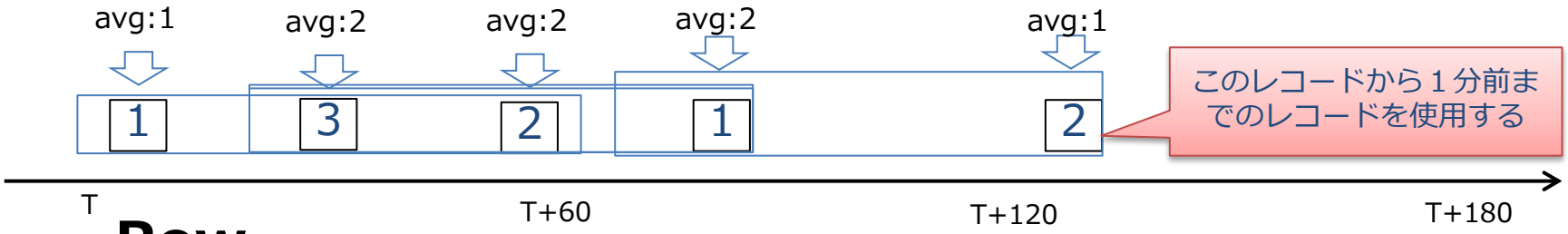
- **Tumbling window**

```
SELECT  count(*)  as cnt
FROM “source “
GROUP BY item,Floor(“source”.rowtime TO minute )
```



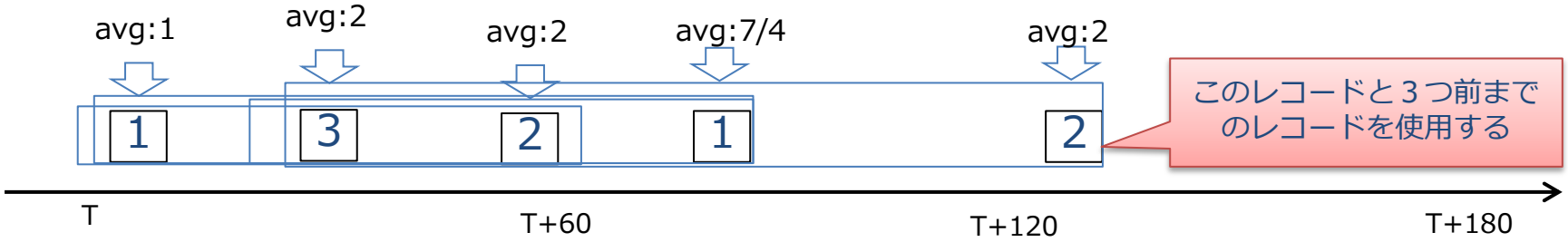
- **Sliding Window**
  - **Time**

```
SELECT  avg(value)  over stw as avg  FROM “source “
WINDOW stw  as (partition by item  range interval ’60’ second preceding)
```



  - **Row**

```
SELECT avg(value)  over srw as cnt  FROM “source “
WINDOW srw as(partition by item ROW 3 preceding)
```



現時点でのAnalyticsへの評価

- **Good**
  - 構築は非常に楽。テンプレートがあるので参考になる
  - 標準SQLのため簡単※ただし列指定は注意！とりあえず「“」で挟む
  - TOP-K等、いくつかの関数も用意されており、便利そう

- **Bad**
  - 参考になるのがドキュメントのみ
  - コンソール画面でのスキーマ定義が面倒
    - 特にAddcolumnsを押すと、一番上に配置されるのが・・・
  - コンソール画面からReferenceデータを確認できない。
    - 列定義をわざわざCLIを見直す
  - 稼動時のKPU数が確認できず、オートスケーリングのため、料金の予測ができない。
  - StreamsのGetShardIteratorオプションがLATEST固定？なため、Streams側にいちいちデータを投入しないといけない

所感

- SQLライクで馴染みやすいが、WINDOWの概念には当初、戸惑った
- Streams、Analyticsともにフルマネージドという点にはかなり魅力的
  - Kafkaで言うと、zookeeperの管理が無くなる
- 似たようなことをEC2で検証したことがあるが値段が格段に安い。ただし、逆に性能面が心配