AWSが支える Eightリコメンデーションエンジン の裏側

AWS Dev Day Tokyo 2017
Day4: 6/2

Eight

Sansan株式会社　Eight事業部
鈴木康寛

名刺を企業の資産に変える

Sansan

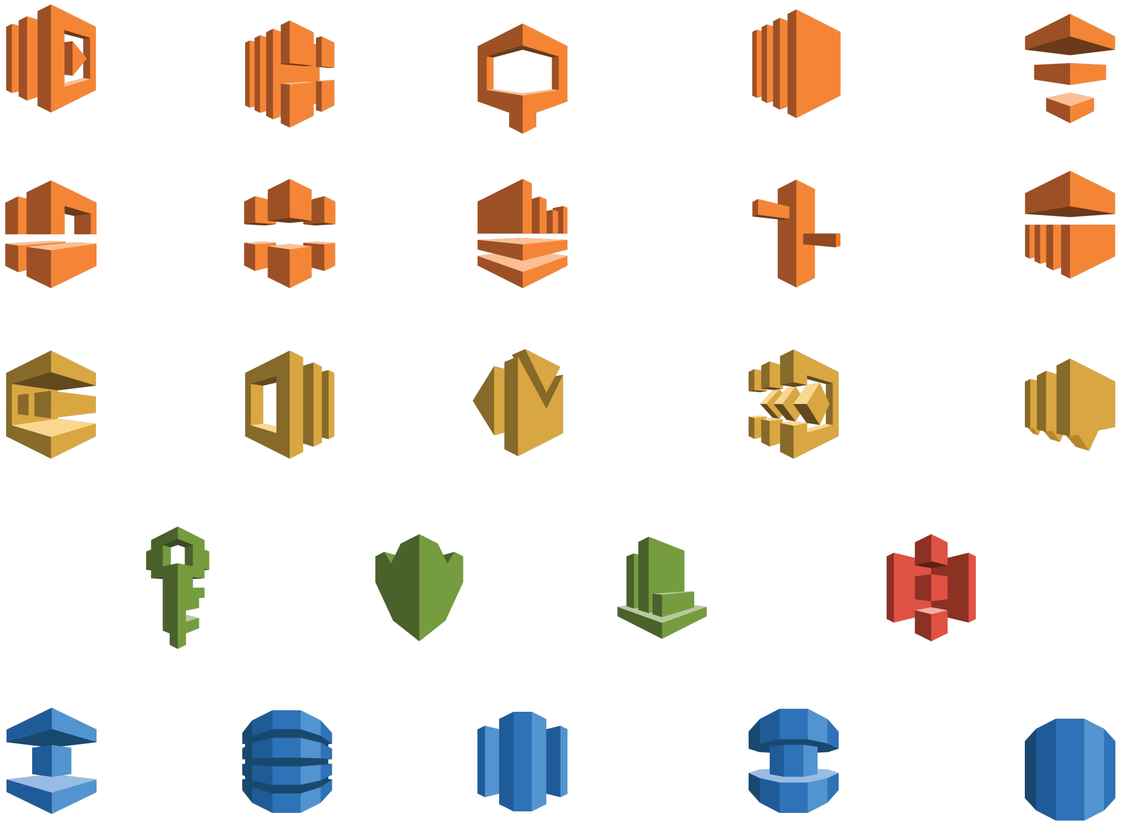
Eight

鈴木 康寛

リコメンデーションチーム
チームリーダ兼テクニカル・アーキテクト

- Rubyでサービス基盤やAPI作ったり
- 最近はPythonでLambda実装とか
- 海外旅行好き 🌴


powered by
amazon
web services



計20サービス以上

サーバーレス

×

ビッグデータ

アジェンダ

1. Eightにおけるリコメンデーション

2. リコメンドアーキテクチャ刷新

3. リコメンドデータ洗い替え

1. Eightにおける
リコメンデーション

個人向け名刺アプリ

Eight

「名刺をビジネスのつながり」に変える

名刺のデータ化

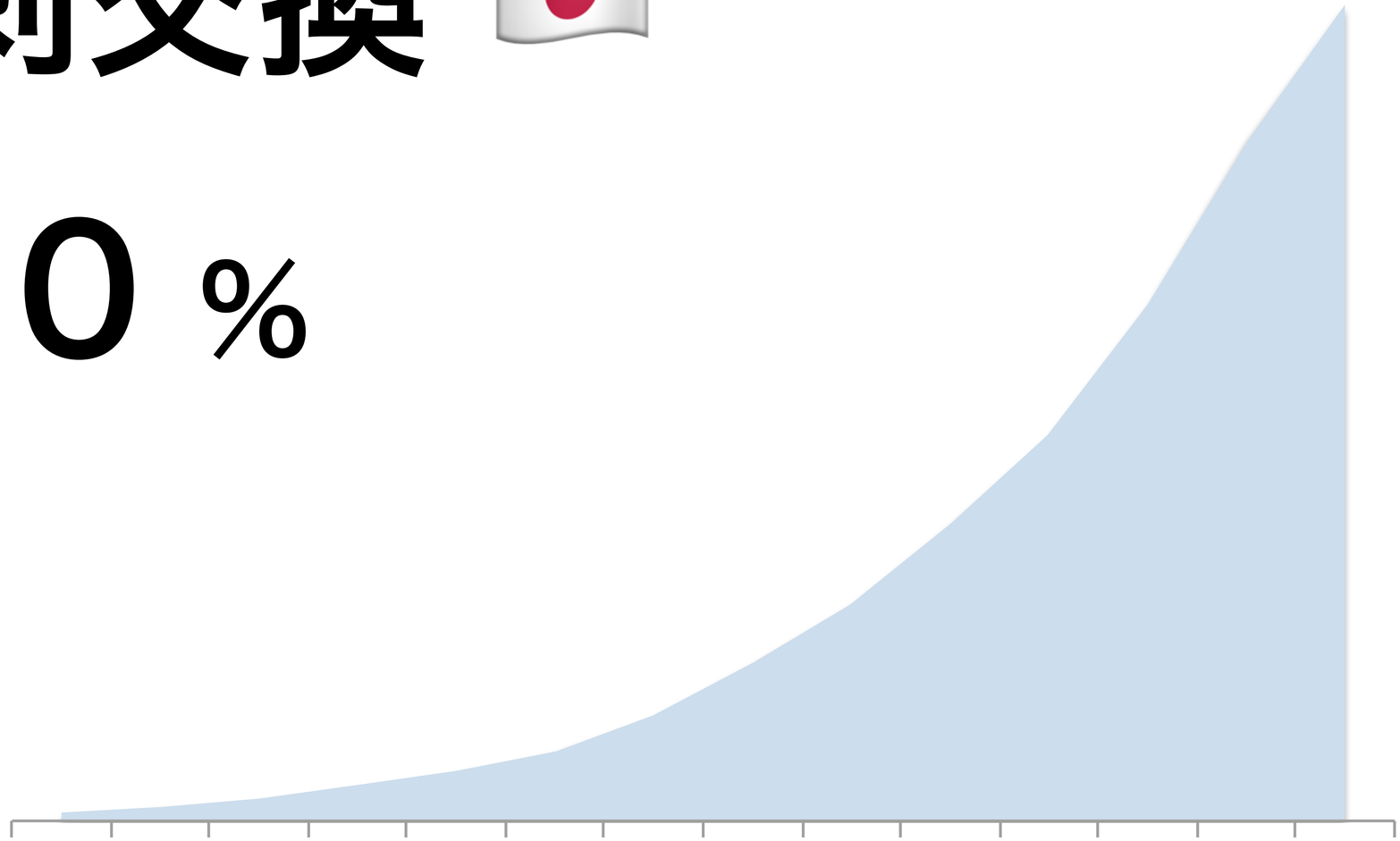

Eight
最近見た名刺
すべて
データ入力待ち
つながっている人
ラベルを作成する
オフ
株式会社千代田飲料
柏木泰助
代表取締役 CEO
名刺は枠に合わせて撮影してください。
フォルダやケースから出して撮影をお願いします。
1枚目 - おもて
片面撮影
両面撮影
名前
会社名
交換月
相沢翔太
株式会社電産広告
メディア事業部
部長
江口林太郎
神谷町ビルディング株式会社
営業本部 東京営業部
遠藤翼
株式会社LINKSUITE
代表取締役
柏木泰助
株式会社千代田飲料
代表取締役 CEO
菊池聡
MONY株式会社
営業企画部 営業推進G
詳細
メッセージ
柏木泰助
よろしくお願いします！
昨日はおめでとう中、ありがとうございました。
また、ご招待ありがとうございます。
送信

つながった相手とコミュニケーション

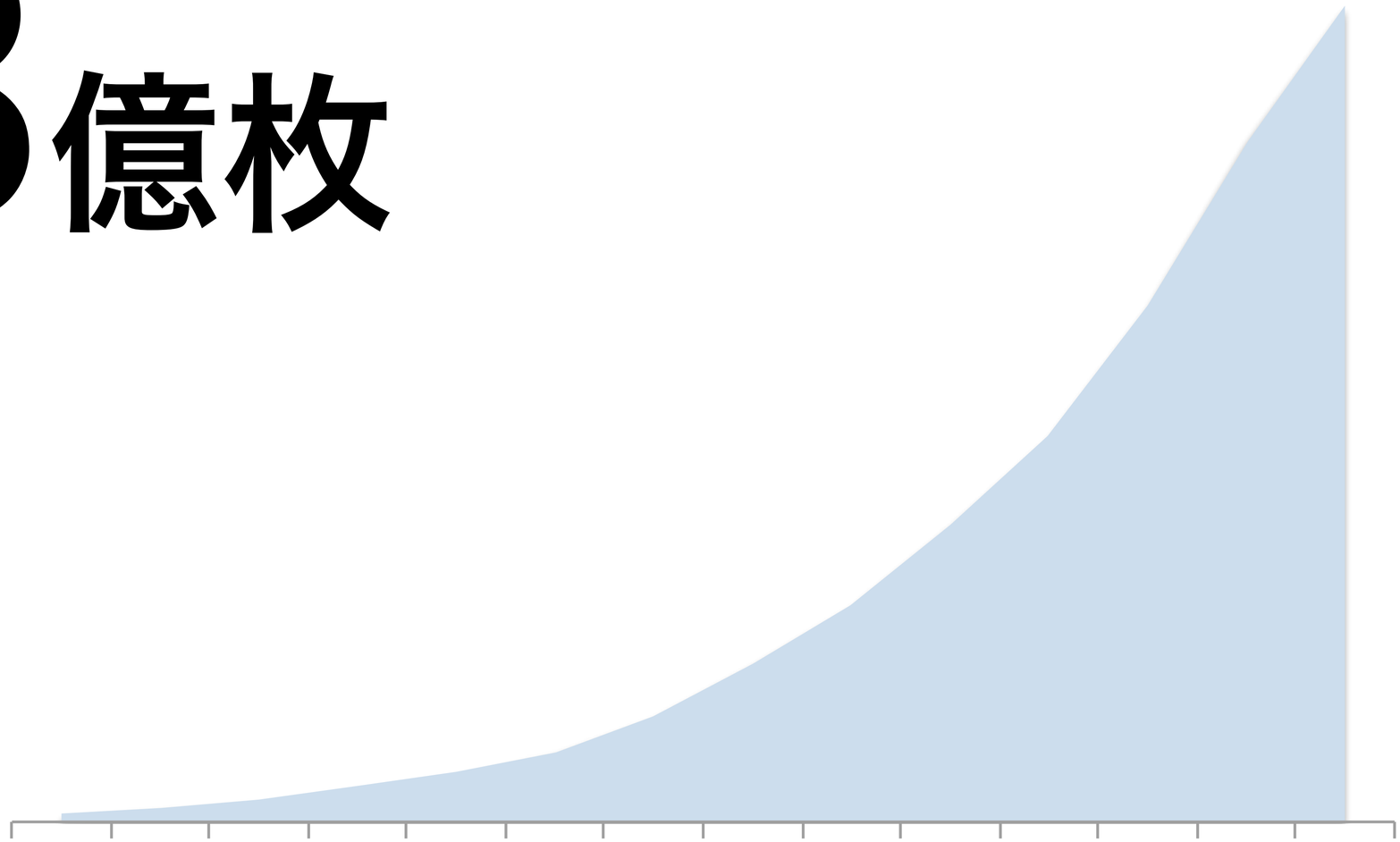
名刺交換

年間名刺交換 🇯🇵

約10％



約3億枚



ビッグデータ

名刺管理から
ビジネスネットワークへ

つながり

つながることで

- 名刺情報のアップデート
- フィード投稿閲覧
- メッセージ送受信

ネットワーク活性化

つながりを 加速 させる技術

リコメンデーション

利用者一人ひとりの嗜好に合わせ
コンテンツを推薦

Eight

名前、会社を検索

藤井洋太郎
Sansan株式会社

フィード
メッセージ
お知らせ

あなたのネットワーク 338
データ入力待ち 0
最近見た名刺 20
つながっている人 225

Business Network Lab
4時間前

「こうすれば大企業の新規事業はうまくいく」

\QUANTUM井上裕太が体現する
ての生き方

お知り合いですか？
名刺交換リクエストを送信して、追加しましょう。

山本啓治
Sansan株式会社
追加

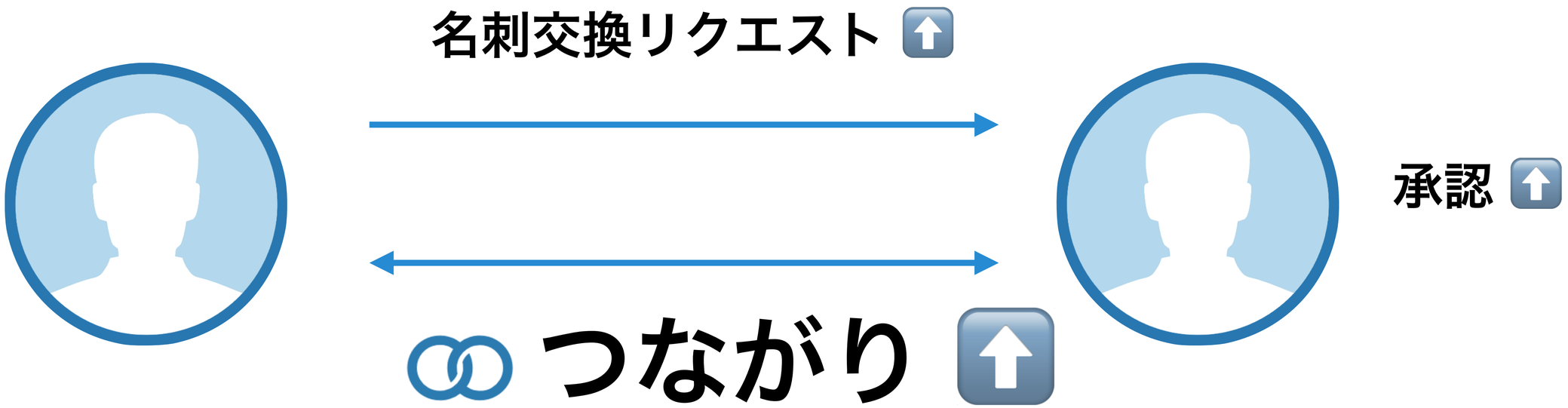
宮城新
Sansan株式会社
追加

小川泰正
Sansan株式会社
追加

酒居潤平
Sansan株式会社
追加

追加

人物 を推薦



「つながり」を 効率的に生み出す

リコメンデーション
エンジン

実は、

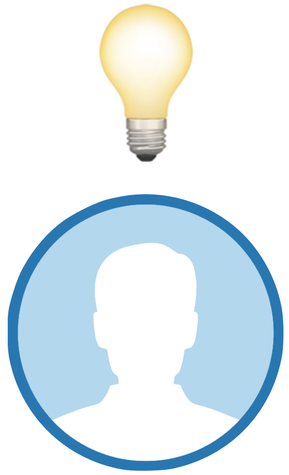
2年ほど前から導入

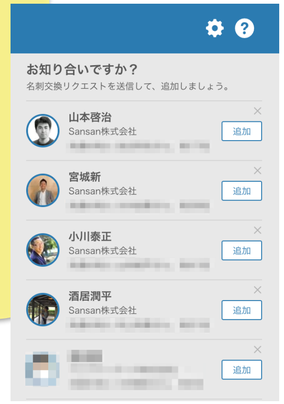
旧 リコメンデーションエンジン

誰が誰に何をしたか
(ユーザーアクティビティ)

つながった
名刺追加

生成





毎日(毎週)一度、バッチ処理で更新

しかし、

Redshift

- 無駄に複雑なクエリ

CloudSearch

- 全文検索として使ってなかった: _score
- 更新までのレイテンシー

つぎはぎ
アーキテクチャ


増えない
つながり

- パフォーマンス ⬇️ 😢
- オペレーションコスト ⬆️ 😢

最適なAWSサービス
選定の重要性

2. リコメンド
アーキテクチャ刷新

昨年 9月

刷新のチャンス到来 💡

リコメンドに最適な
アーキテクチャ設計とは

リコメンデーションの要

1. データ分析

2. アルゴリズム

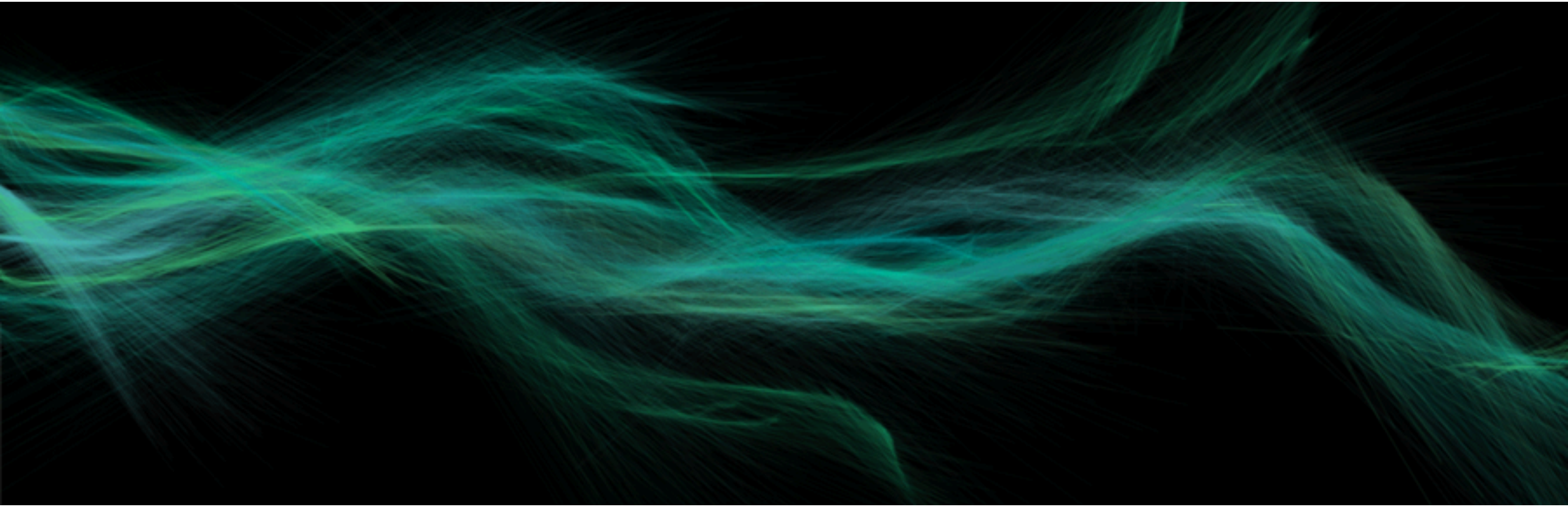
シニアデータサイエンティスト曰く

3. リアルタイムフィードバック

直近の出会いを大事に

データの鮮度が良いほど、
ユーザーはリクエスト承認をしやすくなる
= つながる

“ リアルタイム ”



リコメンデーションエンジン

開発期間

2ヶ月

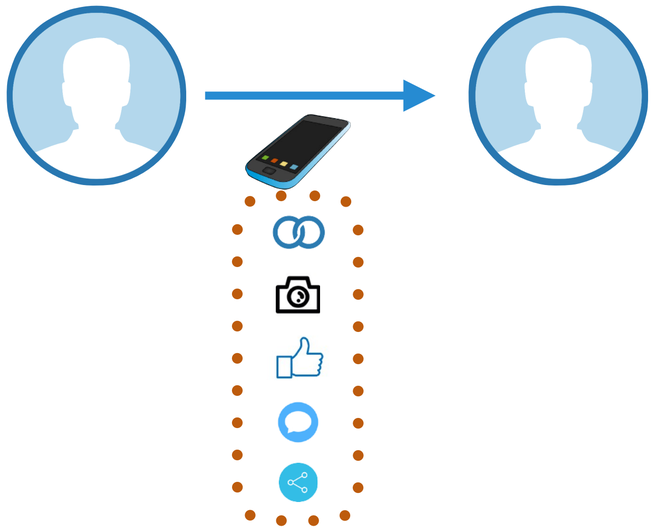
条件

- 2ヶ月できることにフォーカス
- スケーラビリティ

→リアルタイム性を担保し、膨大なデータを処理できること

レーティングデータ

誰が誰に何をしたか
(ユーザーアクティビティ)



生ログ

1レコードに情報をまとめる

誰が	Aさん	Xさん
誰に	Bさん	Yさん
つながり状態	つながっている	…
いいね回数	5回	…
メッセージ回数	2回	…
投稿シェア回数	1回	…
FBつながり	なし	…

リアルタイム性を担保するために
中間データを持つ

構想

- ログ部分は既存のRedshiftのデータを活用
- 中間データ更新にKinesis + Lambdaが使えそう
- 知見があるDynamoDBをストレージに
- シャード数/キャパシティを上げることでスケール

これって巷で話題の 🤔

サーバーレス
アーキテクチャ

激動の2ヶ月を経て

○　○　○ (色々省略)

新
リコメンデーション
エンジン
誕生 

利用AWSサービス

ストリーム

Kinesis Streams

DynamoDB Streams

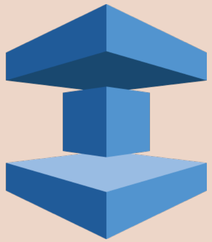
コンシューマ

Lambda

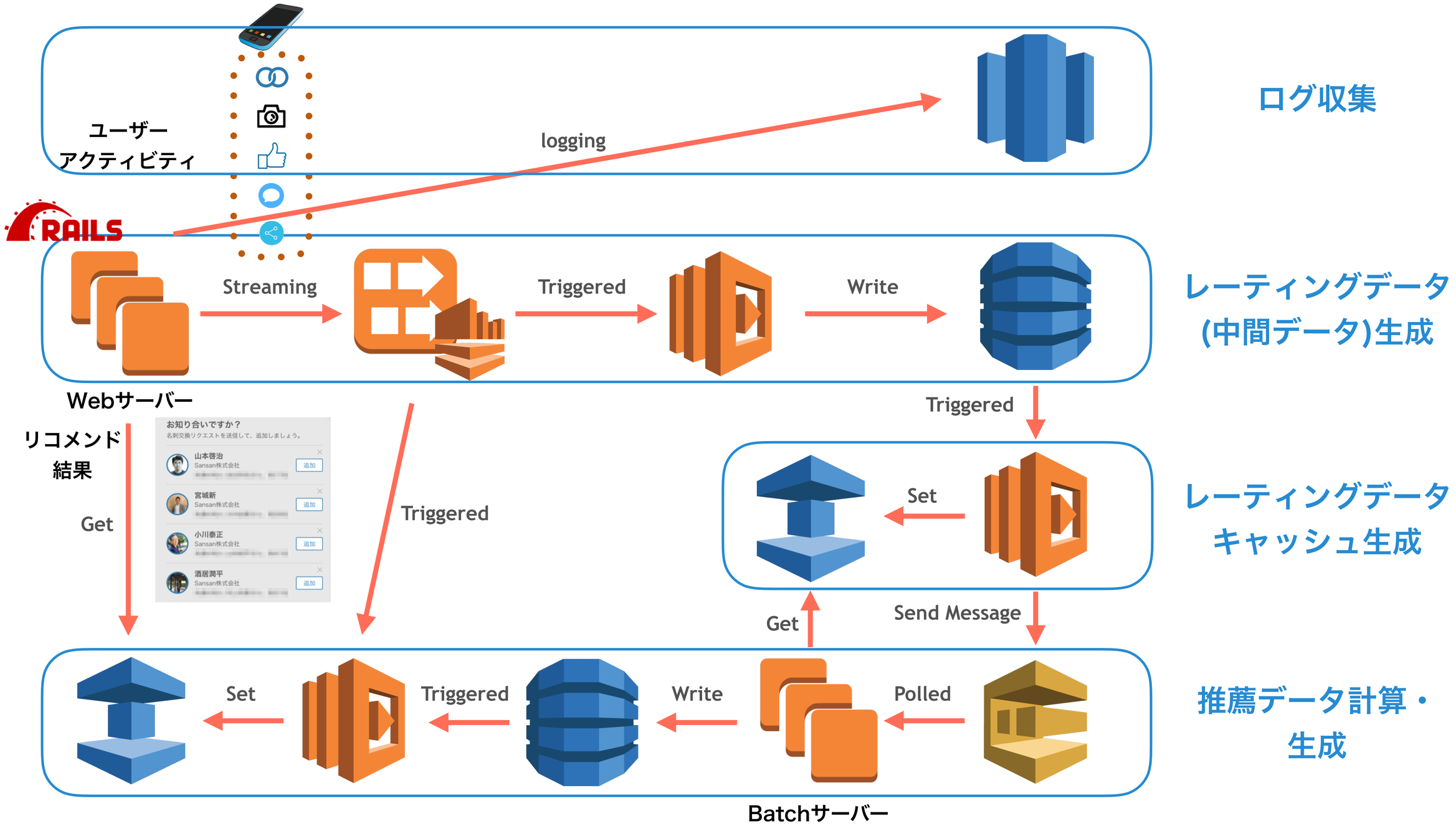
ストレージ

Redshift

DynamoDB



ElastiCache


ユーザー
アクティビティ
logging
ログ収集
RAILS
Streaming
Triggered
Write
レーティングデータ
(中間データ)生成
Webサーバー
Triggered
リコメンド
結果
Get
お知り合いですか？
名刺交換リクエストを送信して、追加しましょう。
山本啓治
Sansan株式会社
追加
宮城新
Sansan株式会社
追加
小川泰正
Sansan株式会社
追加
酒居潤平
Sansan株式会社
追加
Triggered
Set
レーティングデータ
キャッシュ生成
Get
Send Message
Set
Triggered
Write
Polled
推薦データ計算・
生成
Batchサーバー

(ほぼ)

サーバーレス
アーキテクチャ

リコメンド結果が
リアルタイムで生成される

ほど、

甘くはなかった！！😢


サーバーレスは
夢のツールではなかった

サーバレスに対する誤解

- パフォーマンス ⬆️
- オペレーションコスト ⬇️

サーバレスに対する誤解

- パフォーマンス 
- オペレーションコスト 

パフォーマンス

パフォーマンス問題

1分間のアクティビティが

1時間たっても処理できない 😢

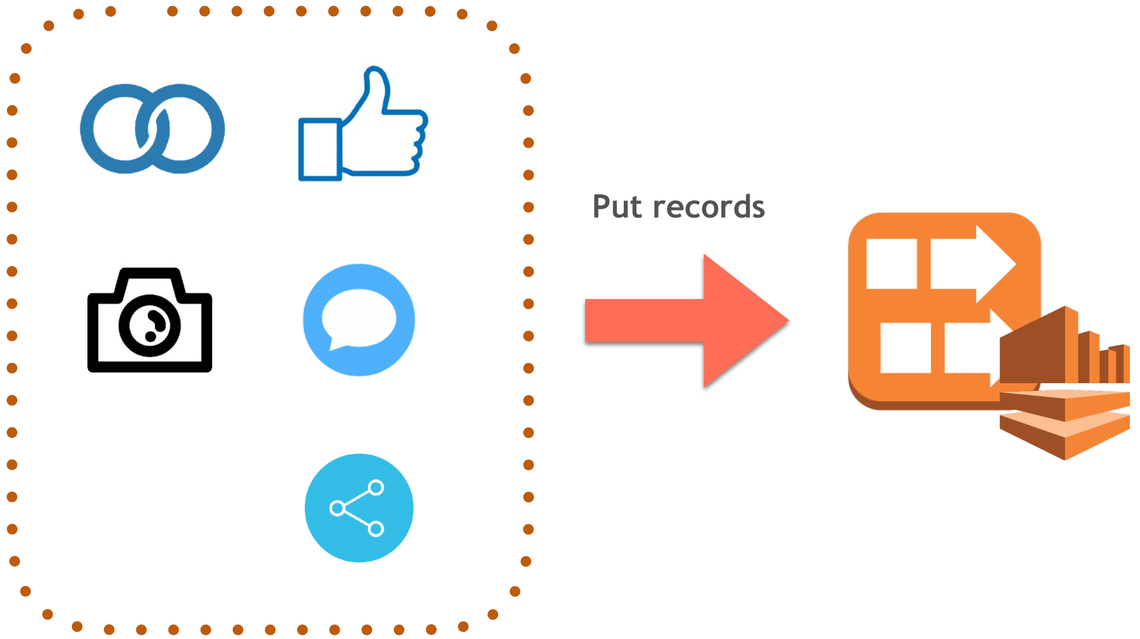
どうしたか？

解決方法

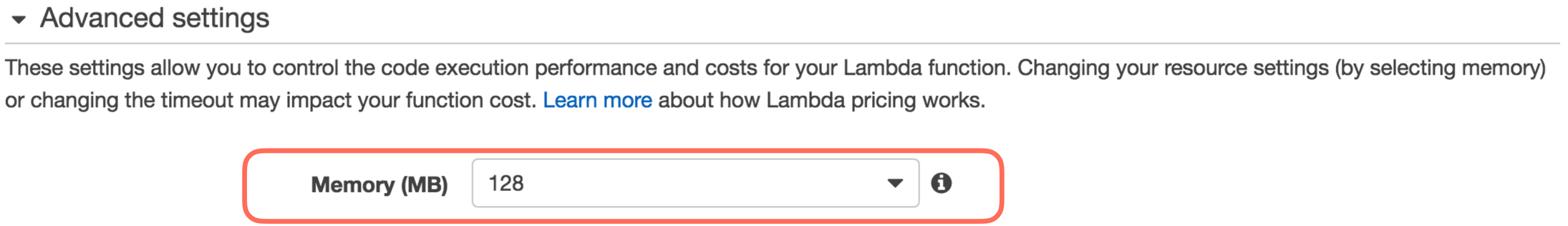
 Kinesis Streams	Put Recordsでリクエストをまとめる
 Lambda	メモリ設定でリソースを増やす
 DynamoDB	Batch writeでまとめて書き込み

Before

After



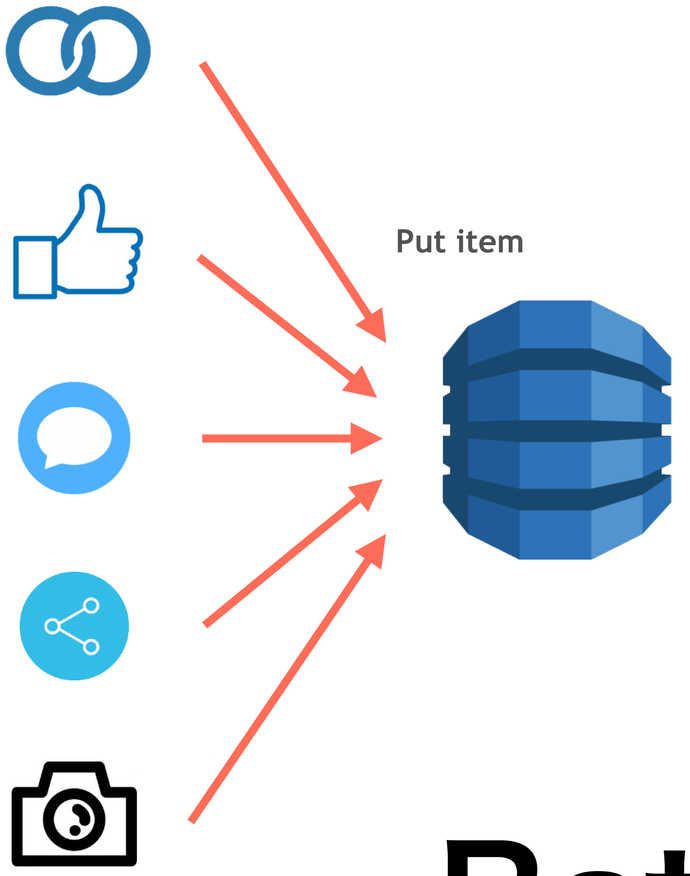
Put recordsでまとめてリクエスト



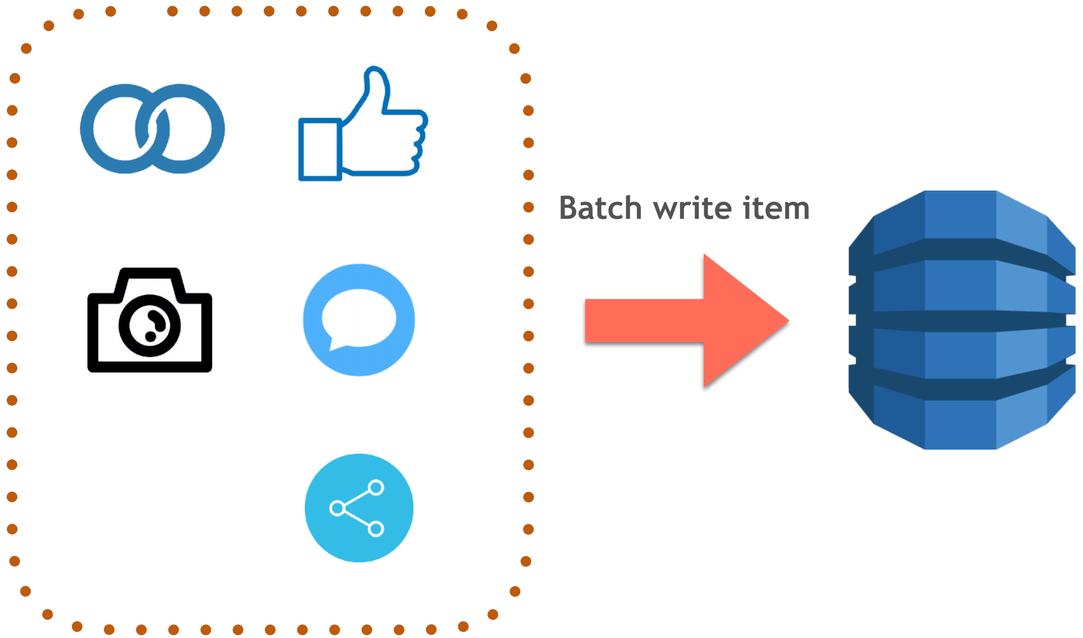
128MB → 1024MB

メモリ設定はコンピューティングリソースの設定

Before



After

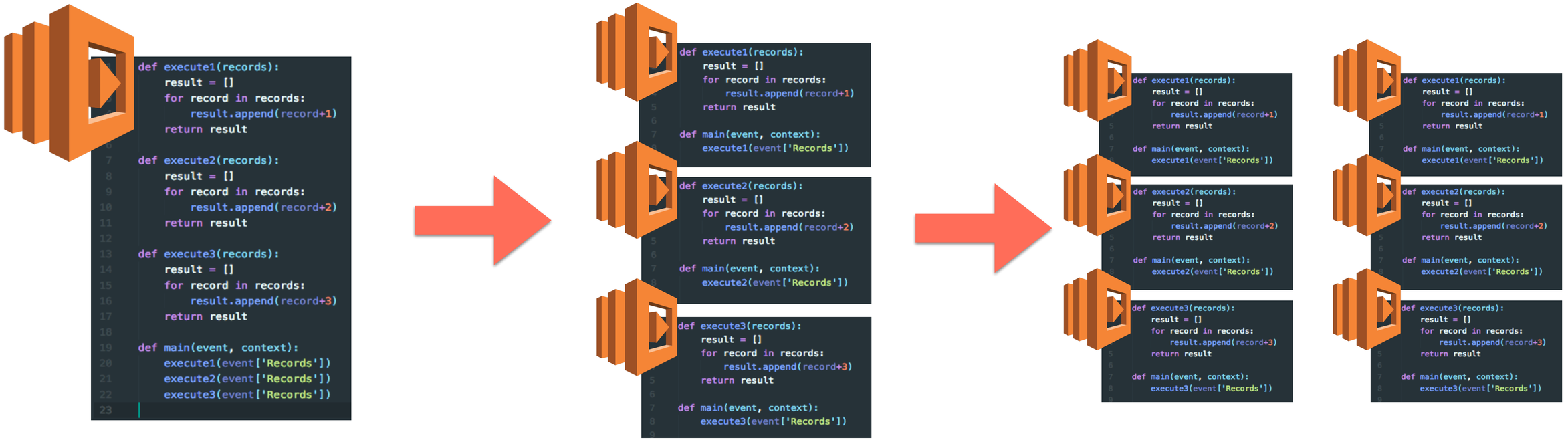


Batch writeで一括保存

パフォーマンス改善 ✨

オペレーション
コスト

Function数の増加問題



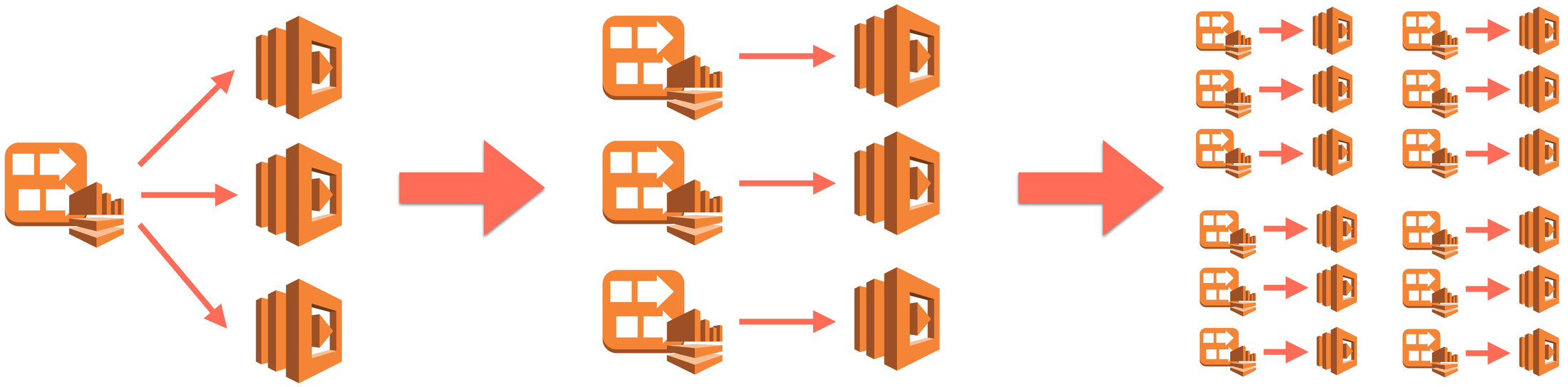
処理の単純化したい

Function数の増加

管理しきれない

ソースコード管理とFunction数の
トレードオフを解決する

ストリーム - コンシューマ 問題



ReadProvisioned
ThroughputExceeded が
発生しやすくなる

ストリーム数の増加

管理しきれない &
コスト増

ストリームとコンシューマのバランスをとる

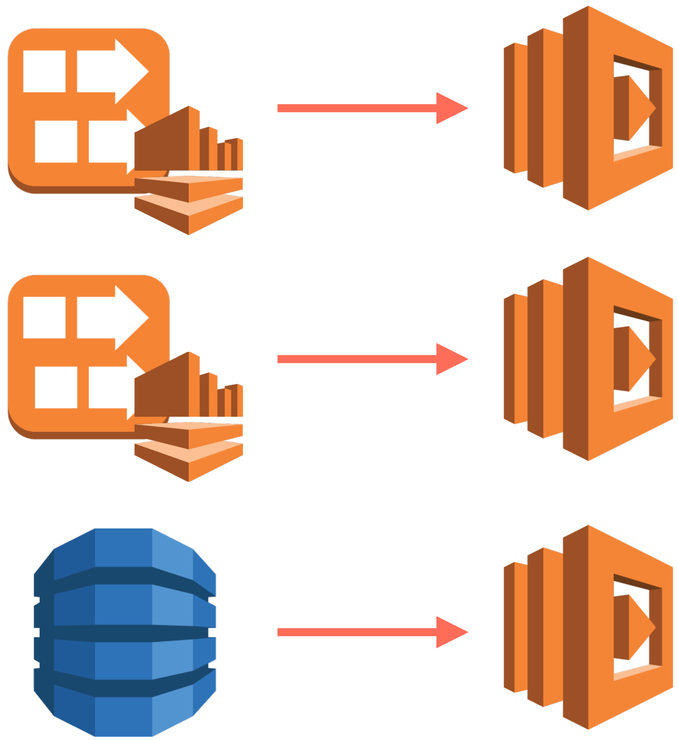
どうしたか？

解決方法

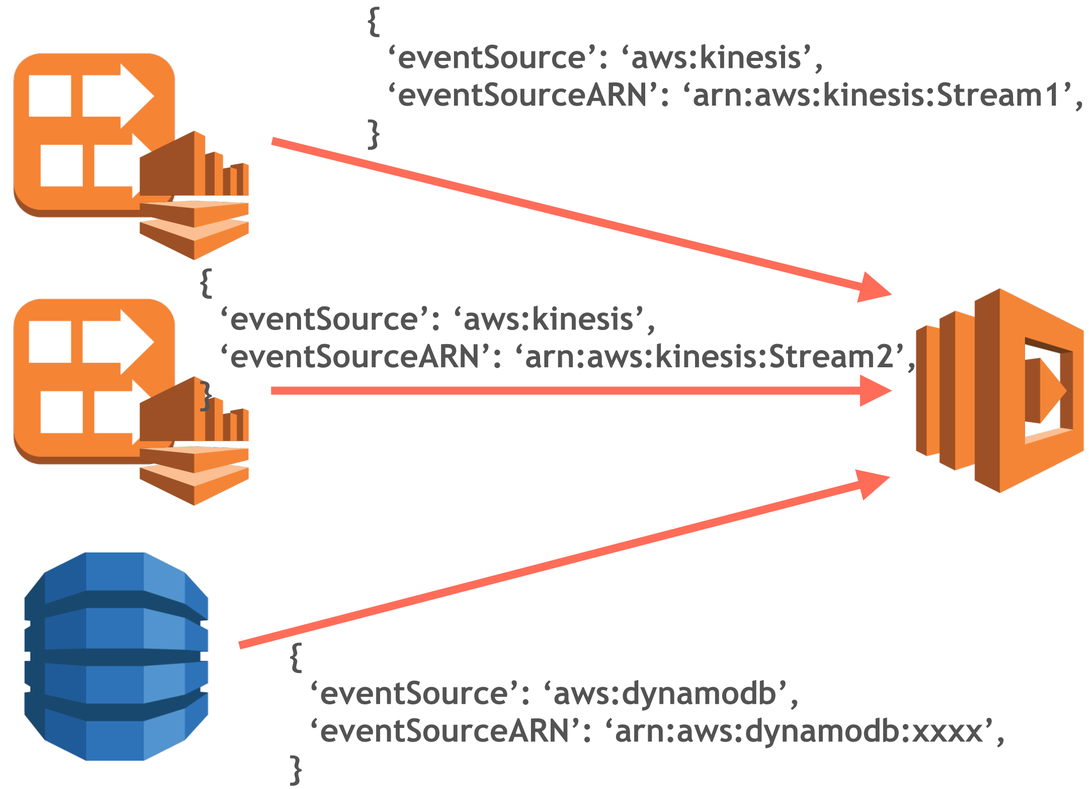
 Lambda	ストリームの判別により、 Lambdaを削減する
 Kinesis Streams	格納するパラメータを工夫して、 StreamとLambdaを削減する

ストリームの判別 ✨

異なるストリームから
同じような処理をさせたいケース



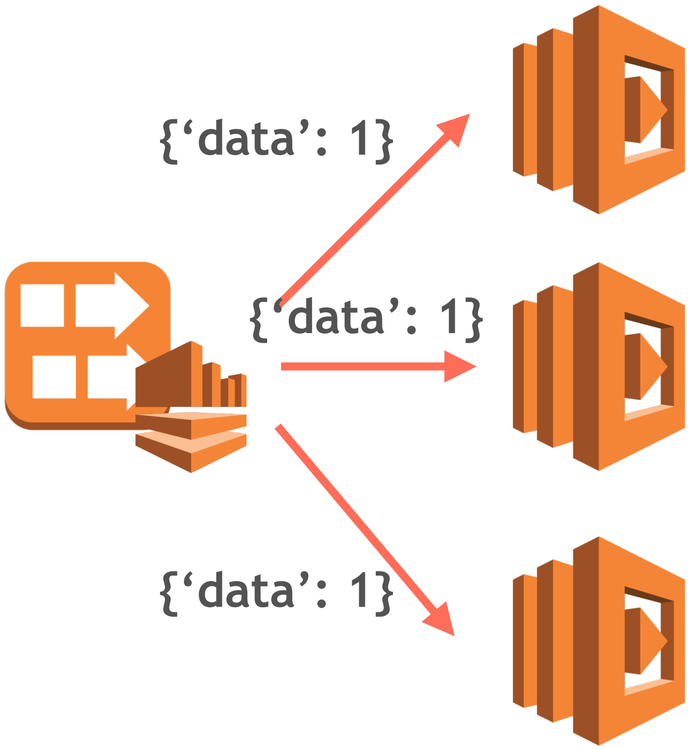
Lambdaをまとめ、ストリームを判別



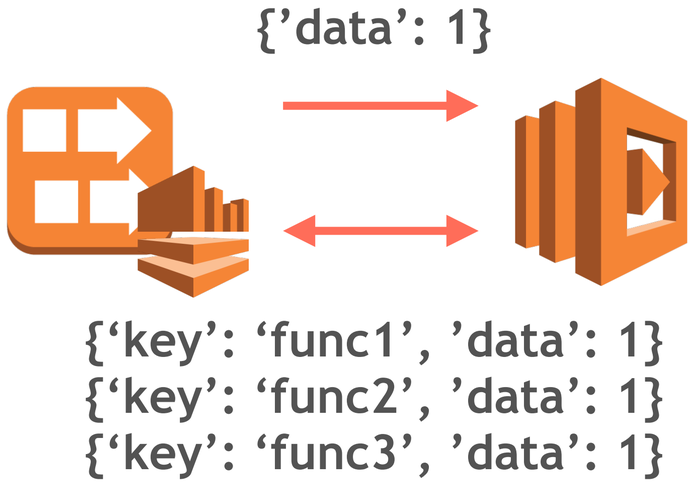
どのストリームかを判別し、処理を分岐

パラメータを工夫 ✨

同じパラメータで
異なる処理をさせたいケース



Streamを書き戻す
(分身の術)



パラメータにより処理を分岐

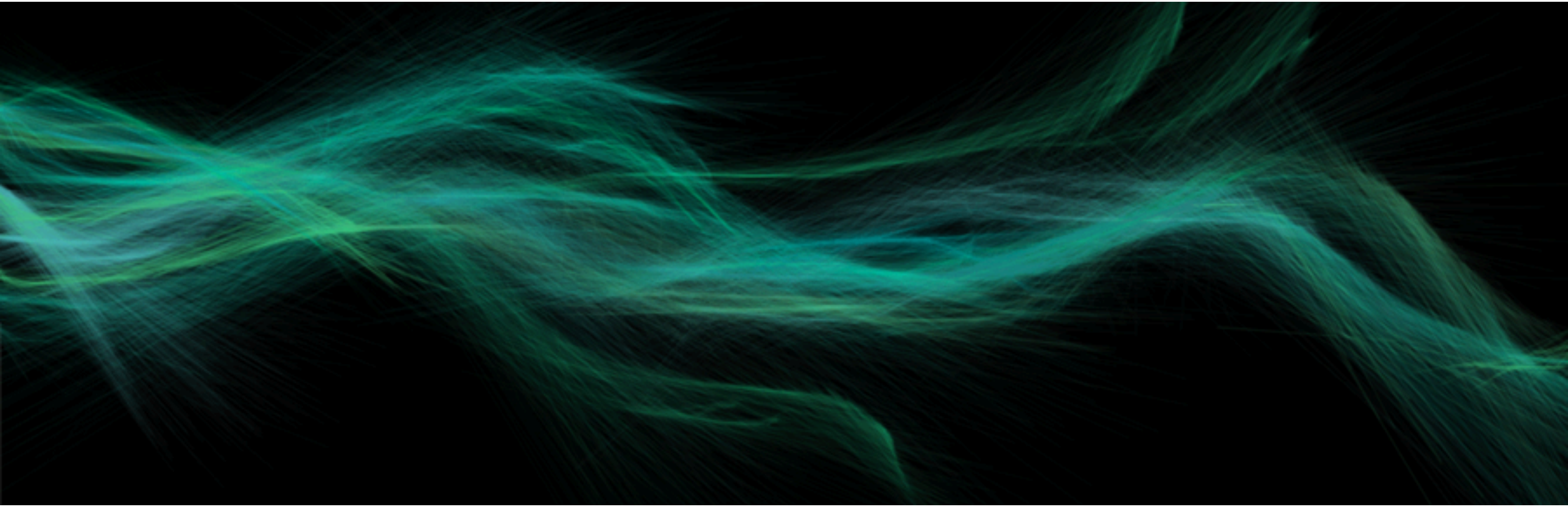
最小限で
管理しやすく✨

- パフォーマンス ⬆️
- オペレーションコスト ⬇️

サーバーレス化 

リアルタイム化 

“ リアルタイム ”



リコメンデーションエンジン

3. リコメンドデータ洗い替え

残る課題

レーティングデータの陳腐化

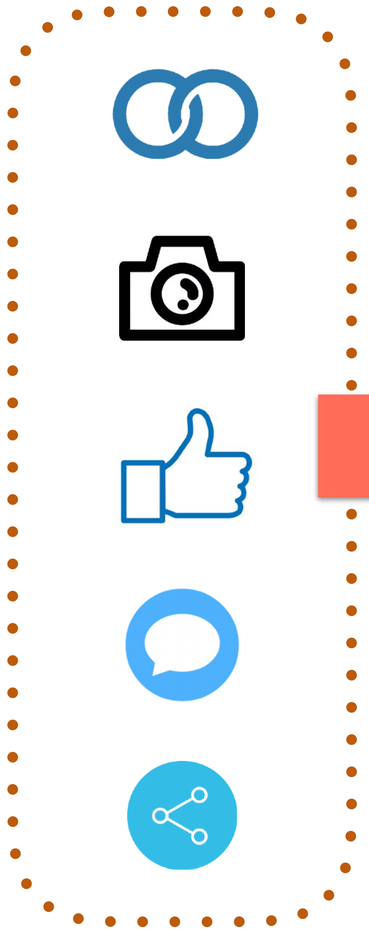
原因

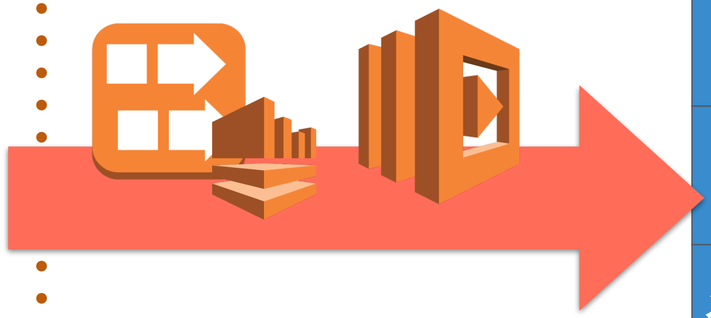
1. Eight本サービス側のデータ構造の変更
2. データ不整合の発生
3. リコメンドアルゴリズムの変更

実際に発生 😱

今まで頑張って蓄積したものは無に還る

ALTER TABLE
とかないので



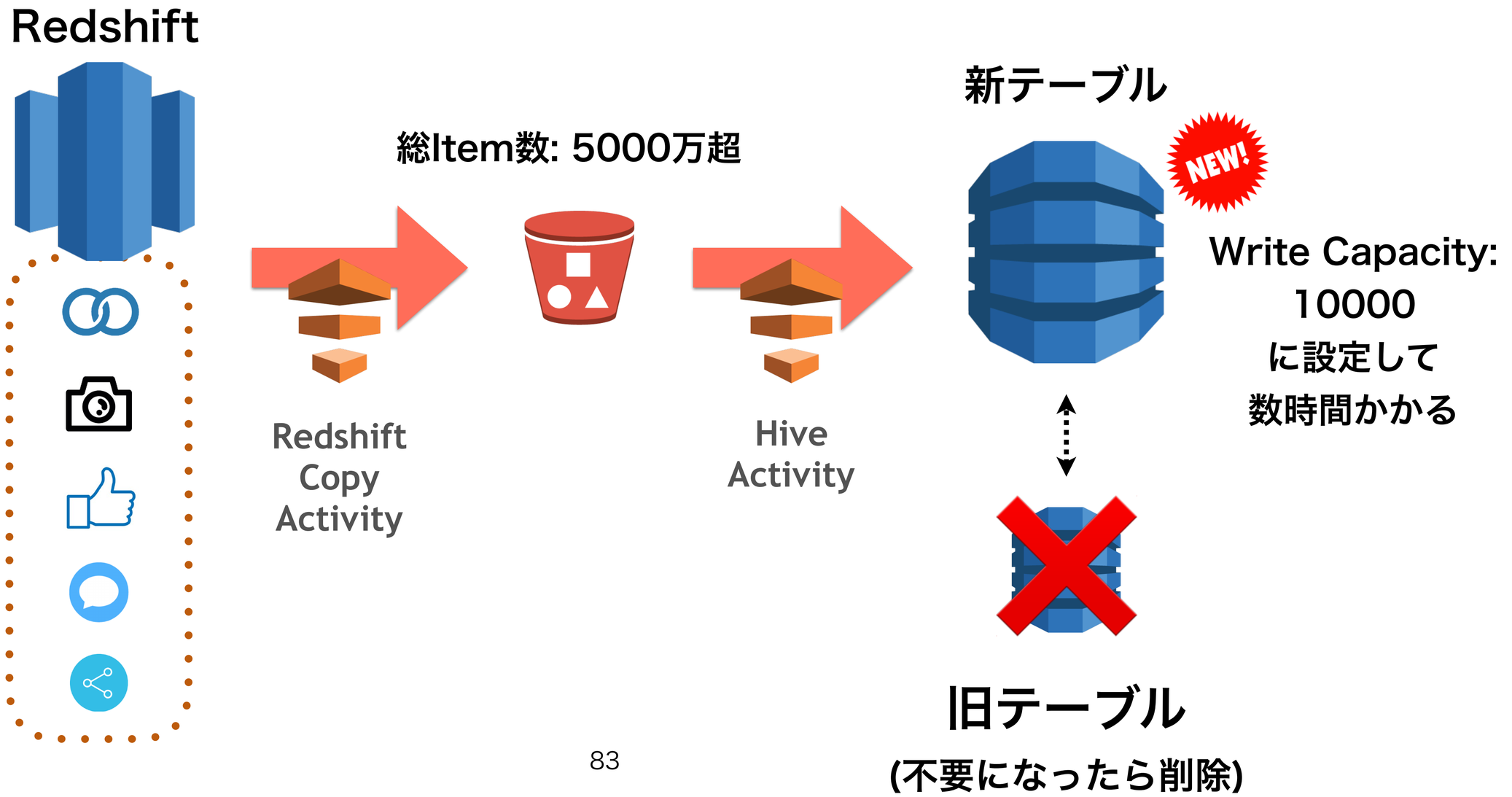


誰が	Aさん	Xさん
誰に	Bさん	さん
つながり	がって	…
いいね回数		…
メッセージ回数		…
投稿シ	1回	
FBつながり	なし	…

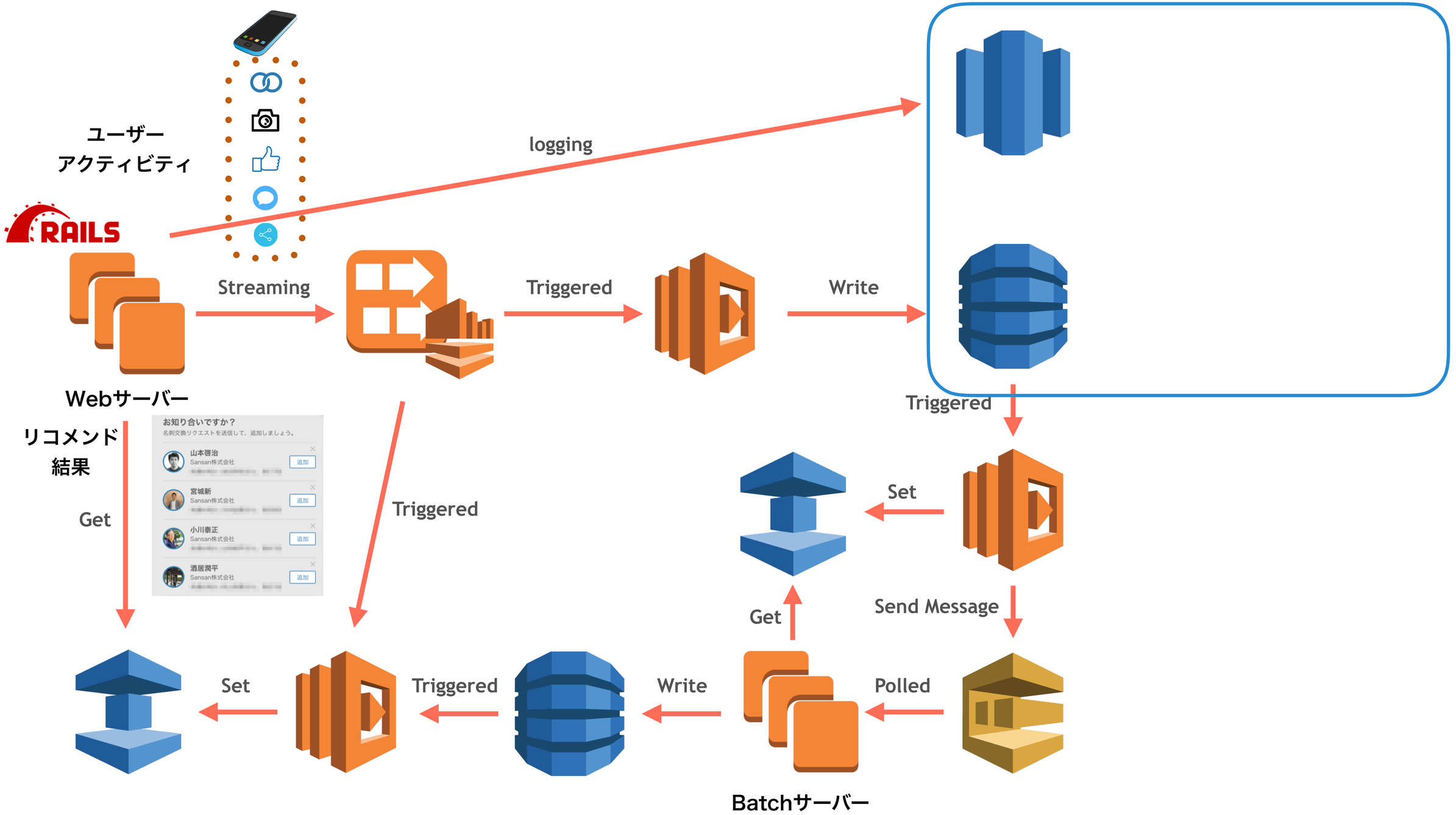
どうしたか？

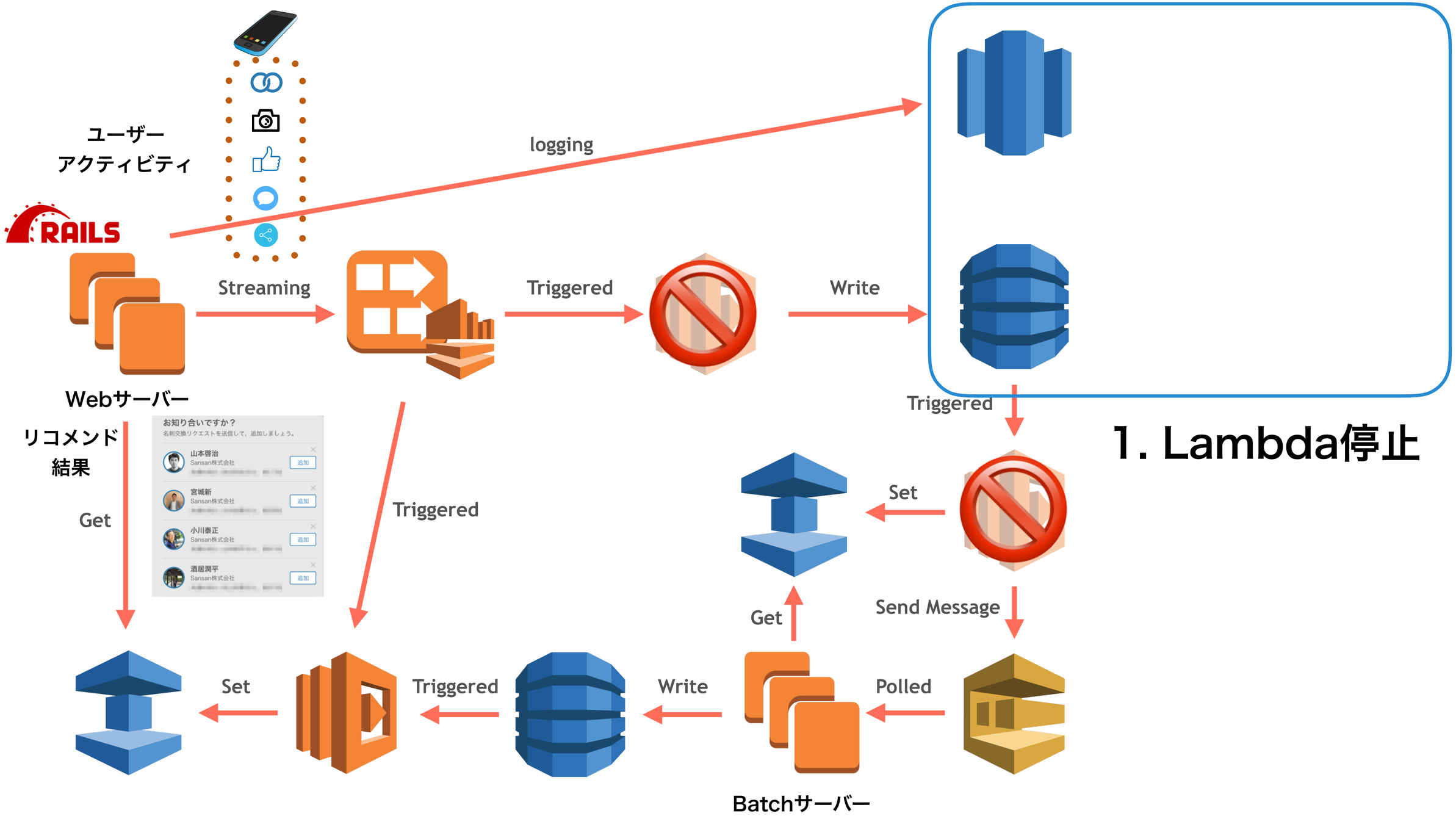
全過去ログを集計して
レーティングデータを
作り直す✨

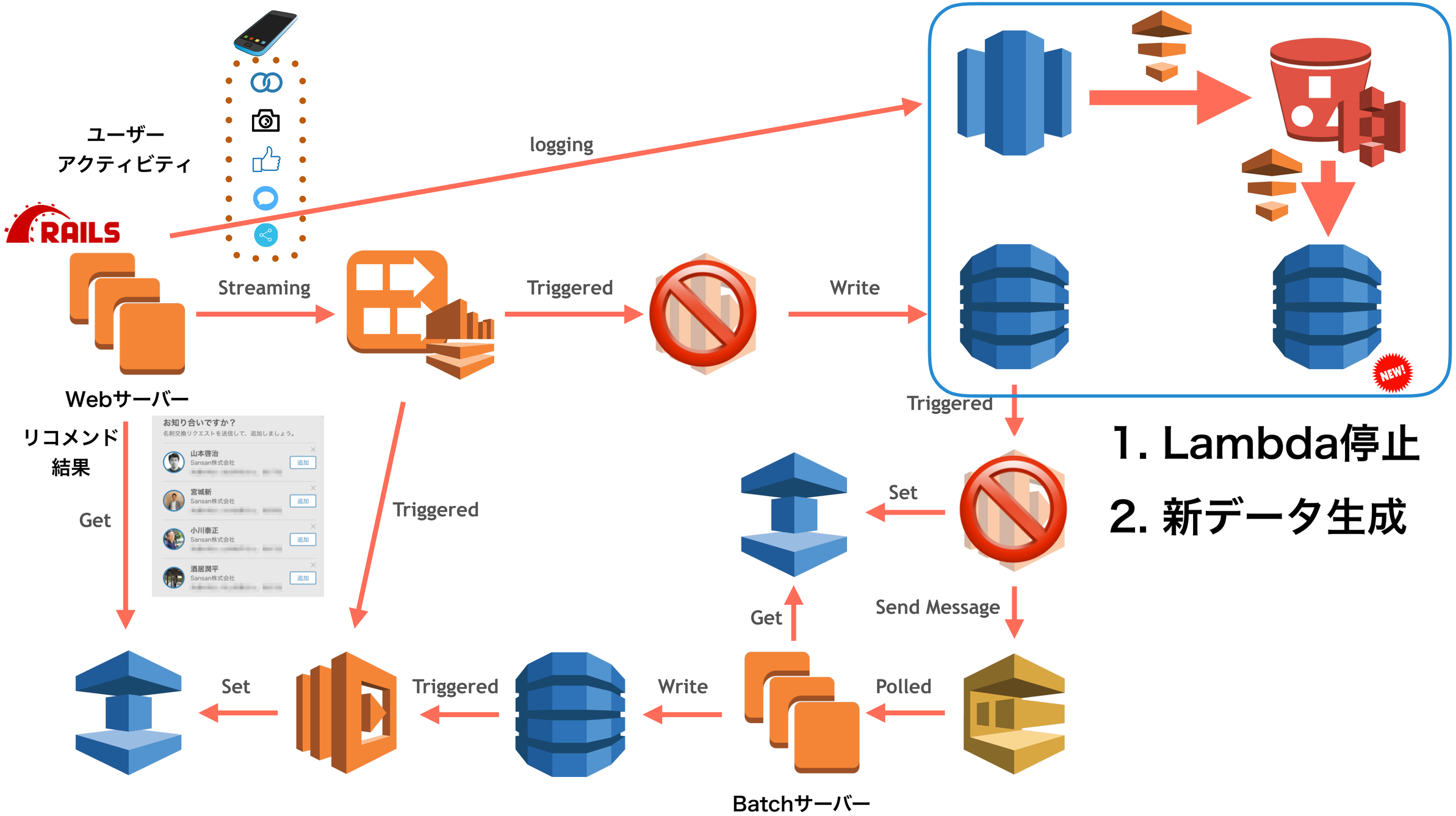
Data Pipelineを利用

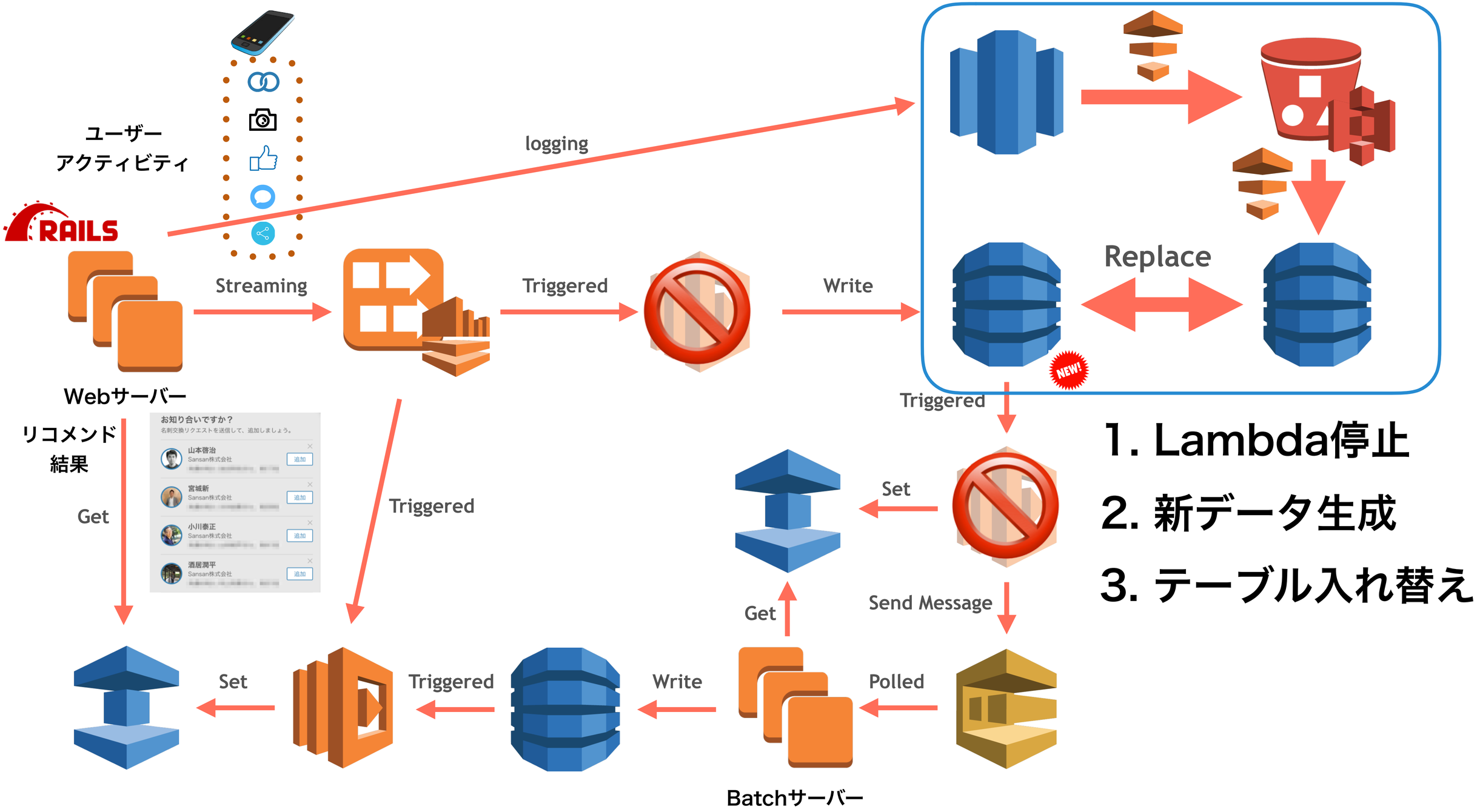


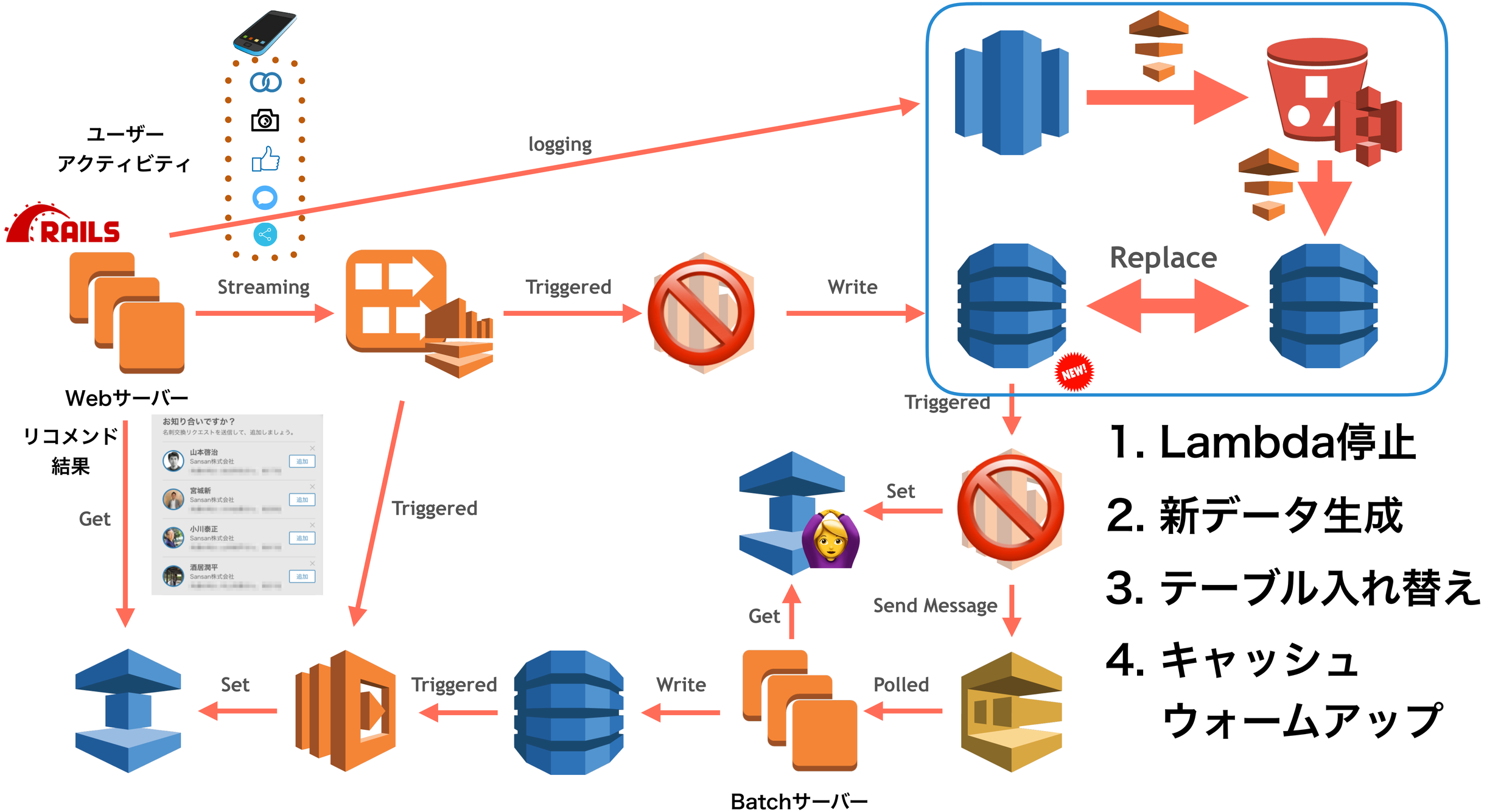
システム全体で考えると

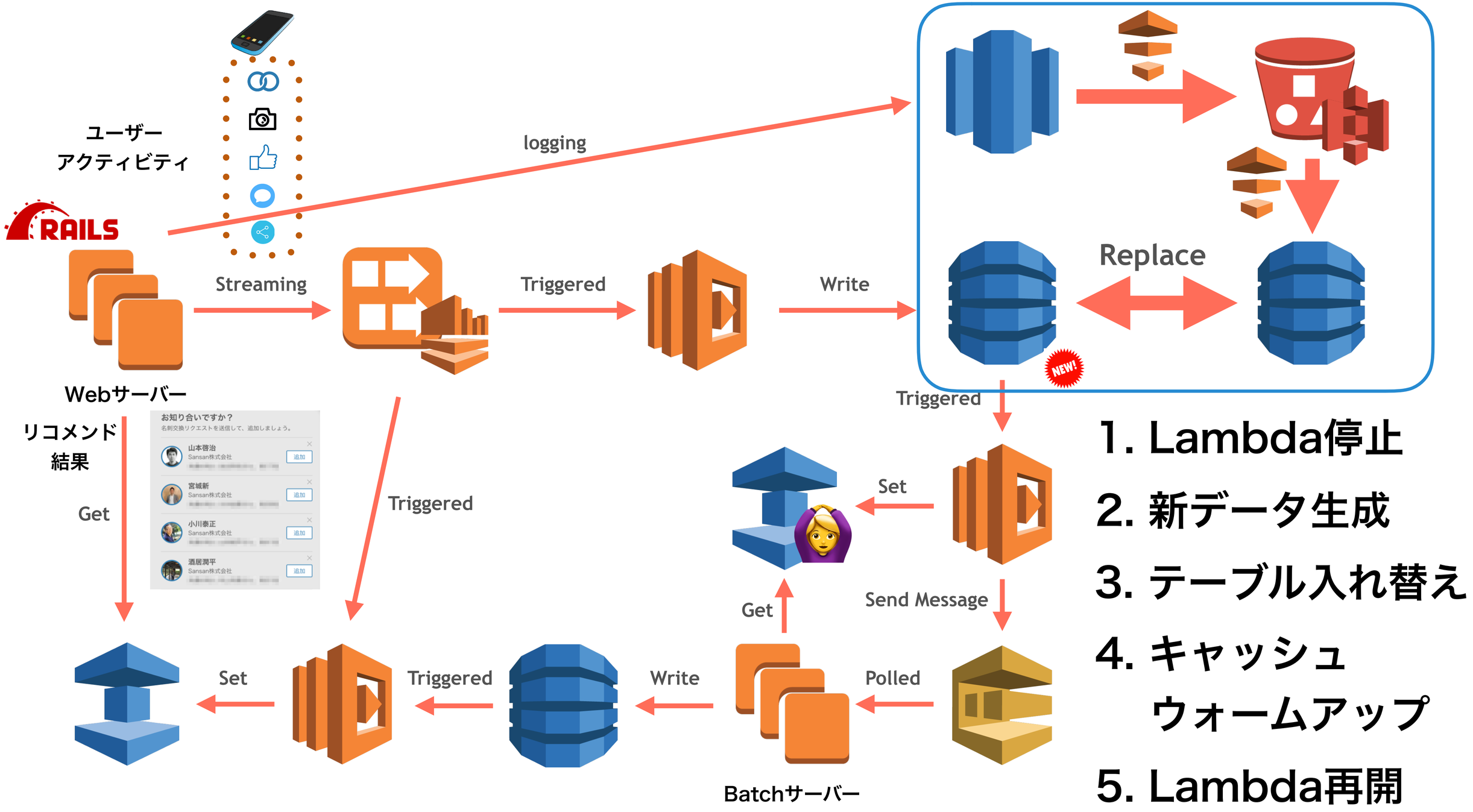

ユーザー
アクティビティ
logging
RAILS
Streaming
Triggered
Write
Webサーバー
Triggered
リコメンド
結果
Get
お知り合いですか？
名刺交換リクエストを送信して、追加しましょう。
山本啓治
Sansan株式会社
追加
宮城新
Sansan株式会社
追加
小川泰正
Sansan株式会社
追加
酒居潤平
Sansan株式会社
追加
Triggered
Set
Get
Send Message
Set
Triggered
Write
Polled
Batchサーバー


ユーザー
アクティビティ
logging
RAILS
Streaming
Triggered
Write
Webサーバー
Triggered
1. Lambda停止
リコメンド
結果
Get
お知り合いですか？
名刺交換リクエストを送信して、追加しましょう。
山本啓治
Sansan株式会社
追加
宮城新
Sansan株式会社
追加
小川泰正
Sansan株式会社
追加
酒居潤平
Sansan株式会社
追加
Triggered
Set
Get
Send Message
Set
Triggered
Write
Polled
Batchサーバー

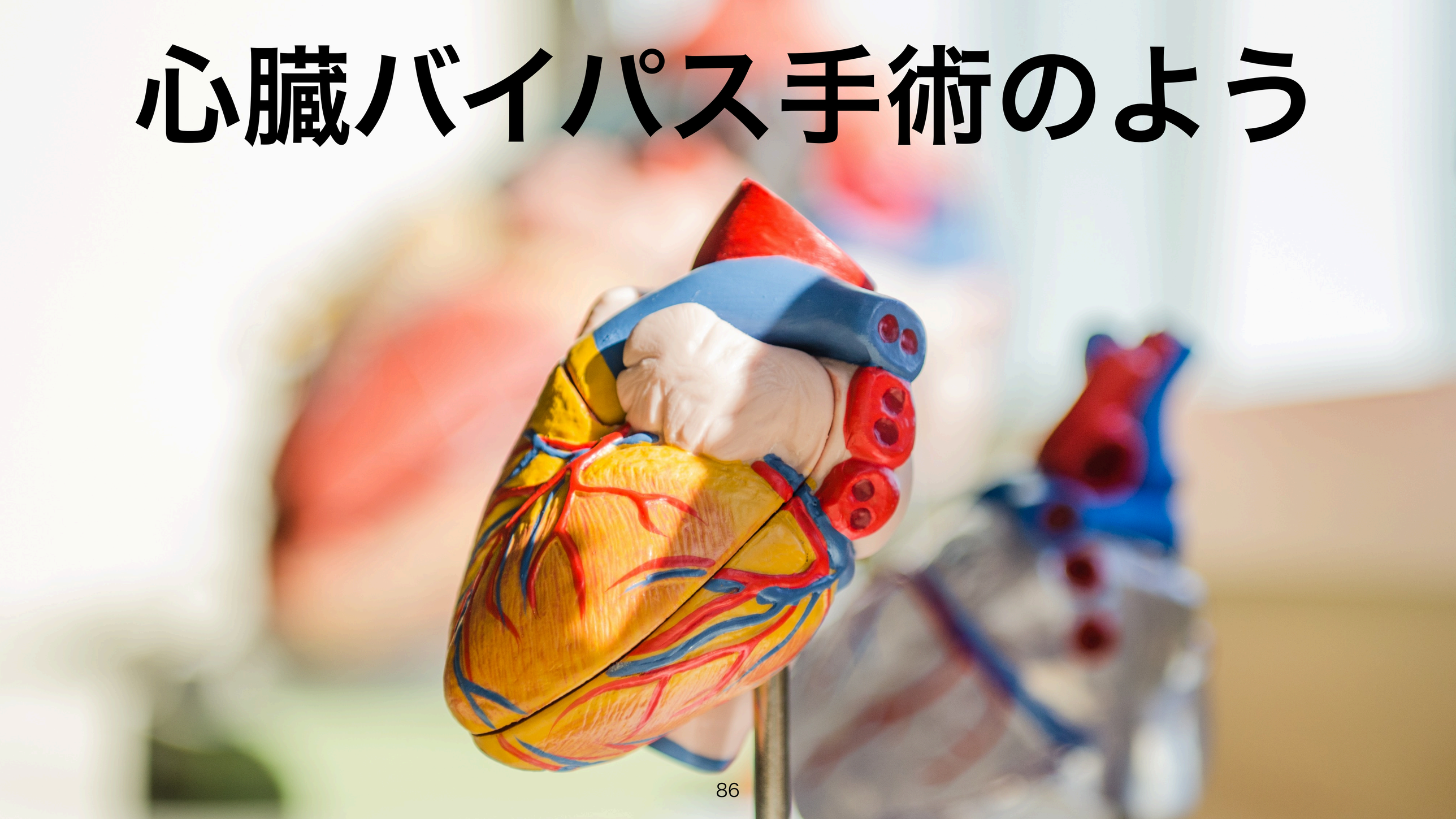

ユーザー
アクティビティ
logging
RAILS
Streaming
Triggered
Write
Webサーバー
NEW!
Triggered
1. Lambda停止
2. 新データ生成
リコメンド
結果
Get
お知り合いですか？
名刺交換リクエストを送信して、追加しましょう。
山本啓治
Sansan株式会社
追加
宮城新
Sansan株式会社
追加
小川泰正
Sansan株式会社
追加
酒居潤平
Sansan株式会社
追加
Triggered
Set
Get
Send Message
Set
Triggered
Write
Polled
Batchサーバー


ユーザー
アクティビティ
logging
RAILS
Streaming
Triggered
Write
Replace
NEW!
Webサーバー
Triggered
リコメンド
結果
Get
お知り合いですか？
名刺交換リクエストを送信して、追加しましょう。
山本啓治
Sansan株式会社
追加
宮城新
Sansan株式会社
追加
小川泰正
Sansan株式会社
追加
酒居潤平
Sansan株式会社
追加
Triggered
Set
Get
Send Message
Set
Triggered
Write
Polled
Batchサーバー
1. Lambda停止
2. 新データ生成
3. テーブル入れ替え


ユーザー
アクティビティ
logging
RAILS
Streaming
Triggered
Write
Replace
NEW!
Webサーバー
Triggered
リコメンド
結果
Get
お知り合いですか？
名刺交換リクエストを送信して、追加しましょう。
山本啓治
Sansan株式会社
追加
宮城新
Sansan株式会社
追加
小川泰正
Sansan株式会社
追加
酒居潤平
Sansan株式会社
追加
Triggered
Set
Send Message
Get
Polled
Set
Triggered
Write
Batchサーバー
1. Lambda停止
2. 新データ生成
3. テーブル入れ替え
4. キャッシュ
ウォームアップ


ユーザー
アクティビティ
logging
RAILS
Streaming
Triggered
Write
Replace
NEW!
Webサーバー
Triggered
リコメンド
結果
Get
お知り合いですか？
名刺交換リクエストを送信して、追加しましょう。
山本啓治
Sansan株式会社
追加
宮城新
Sansan株式会社
追加
小川泰正
Sansan株式会社
追加
酒居潤平
Sansan株式会社
追加
Triggered
Set
Send Message
Get
Polled
Set
Triggered
Write
Batchサーバー
1. Lambda停止
2. 新データ生成
3. テーブル入れ替え
4. キャッシュ
ウォームアップ
5. Lambda再開

心臓バイパス手術のよう



しかも 

対象は3テーブル存在

レーティングデータ1
Item数: 約6500万

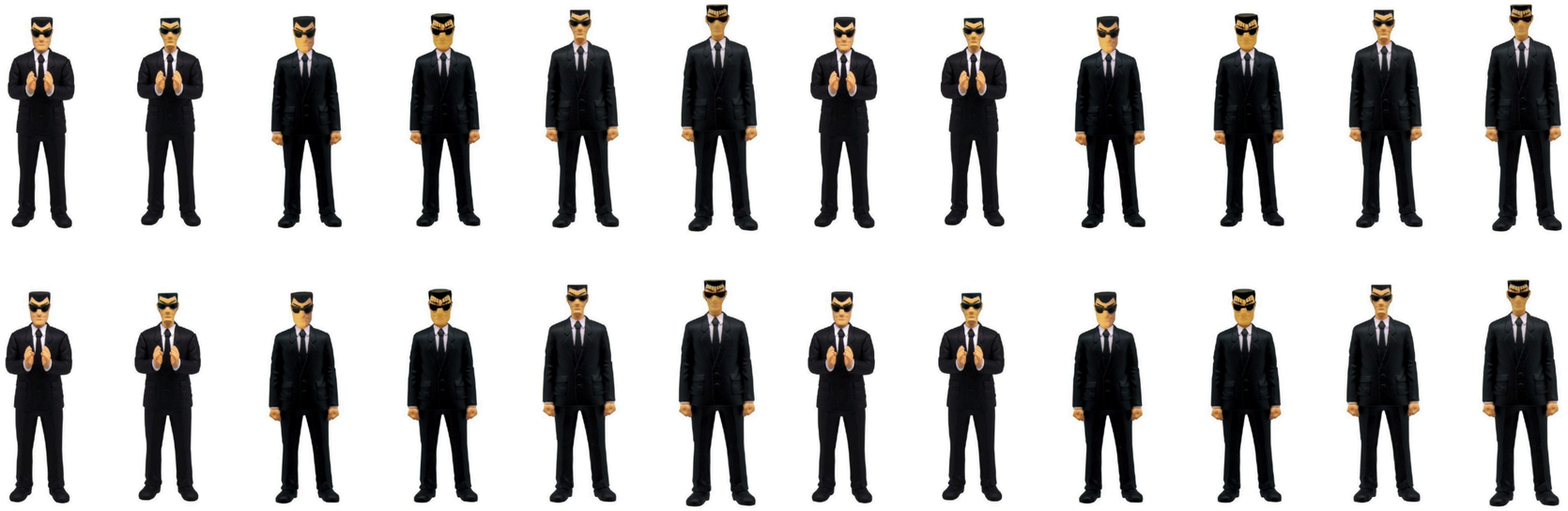
レーティングデータ2
Item数: 約5000万

レーティングデータ3
Item数: 約250万

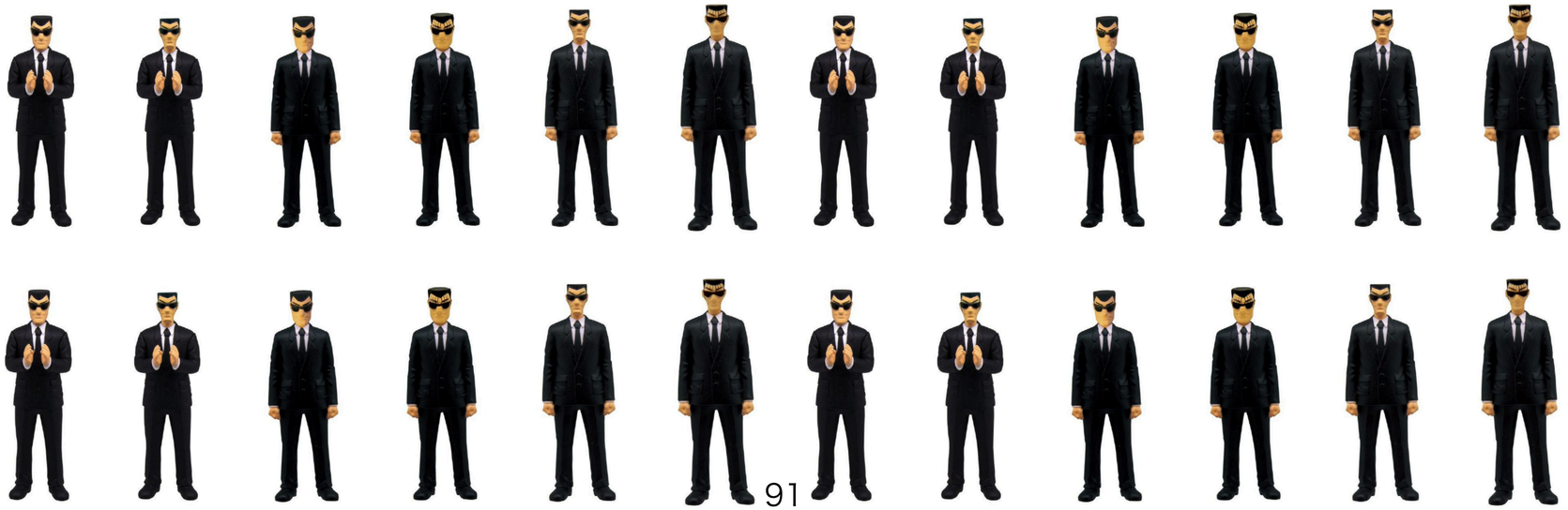
指定したものだけ洗い替えたい


こんなの
手で
やってられない

ダウンタイム無しで行うには
ワークフロー管理が必須



サーバーレスで絶対完結したい



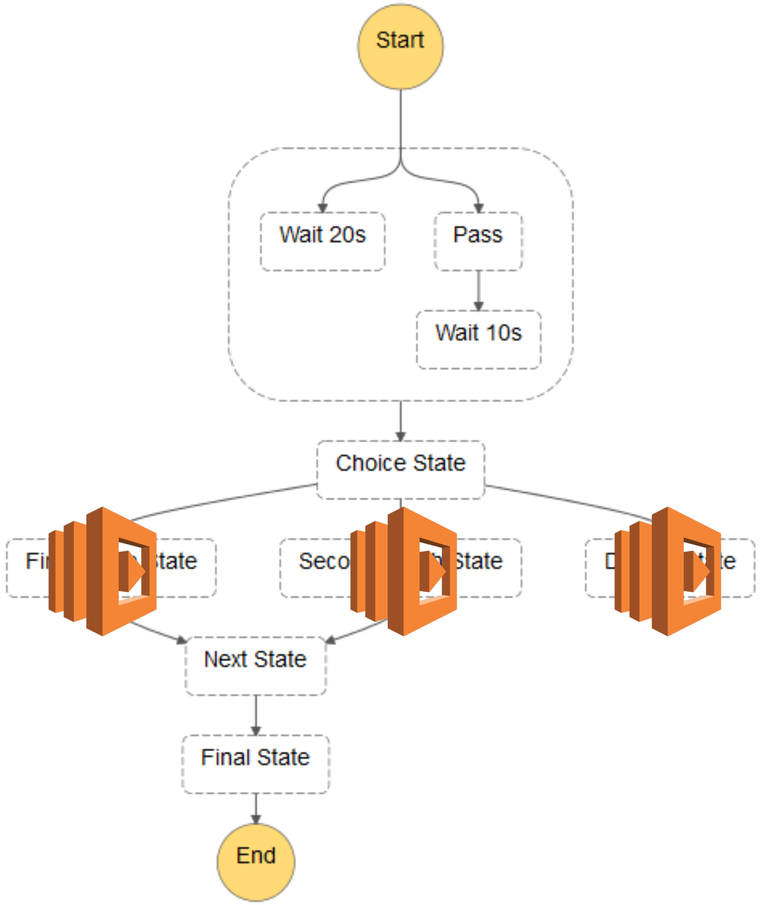
昨年 12月

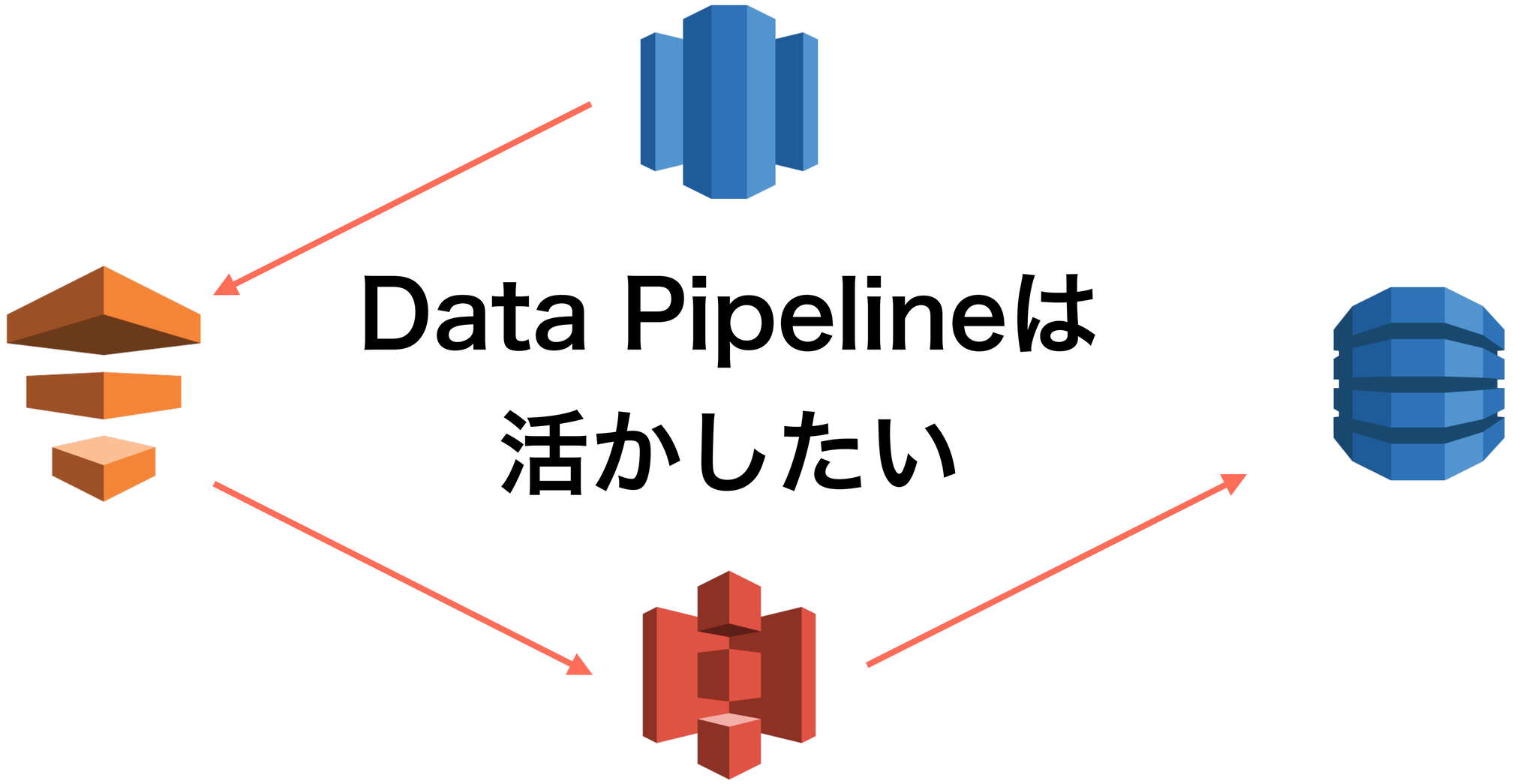
奇跡的なタイミングで

神アップデート！！

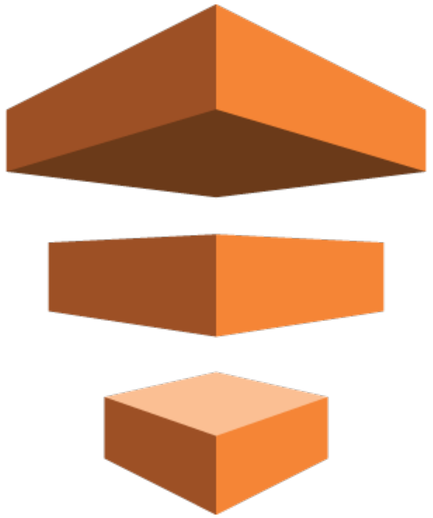
Step Functions ✨

複数のLambda処理をまとめて
ワークフロー管理できるようにするサービス




Data Pipelineは
活かしたい

サーバーレスでデータ洗い替え



Data Pipeline

On

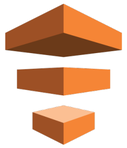
Step Functions

課題

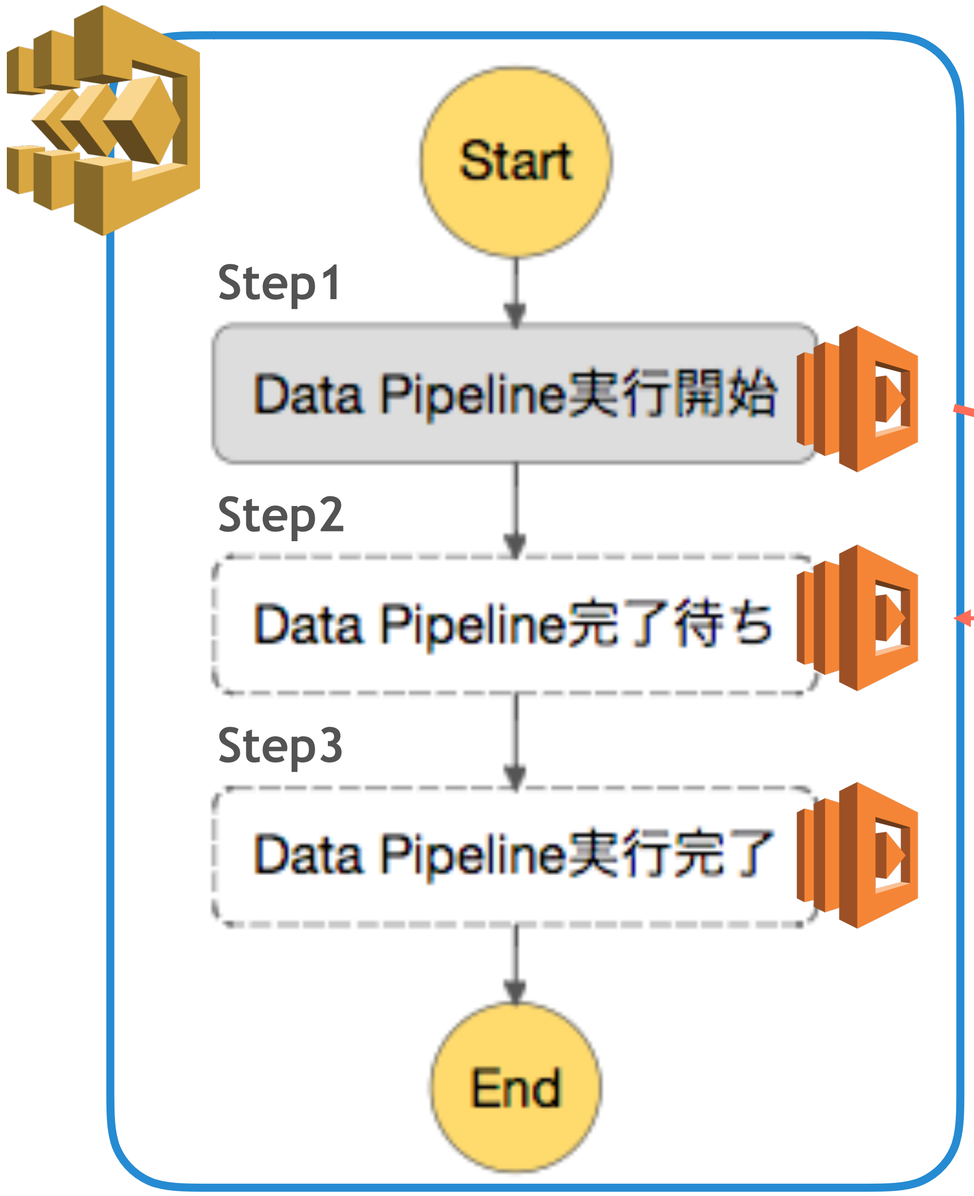
- Step FunctionsからData Pipelineの状態管理
- LambdaのTimeout問題
- 洗い替え対象テーブルの指定方法
- キャッシュのウォームアップ問題
- ダウンタイム無しで洗い替え

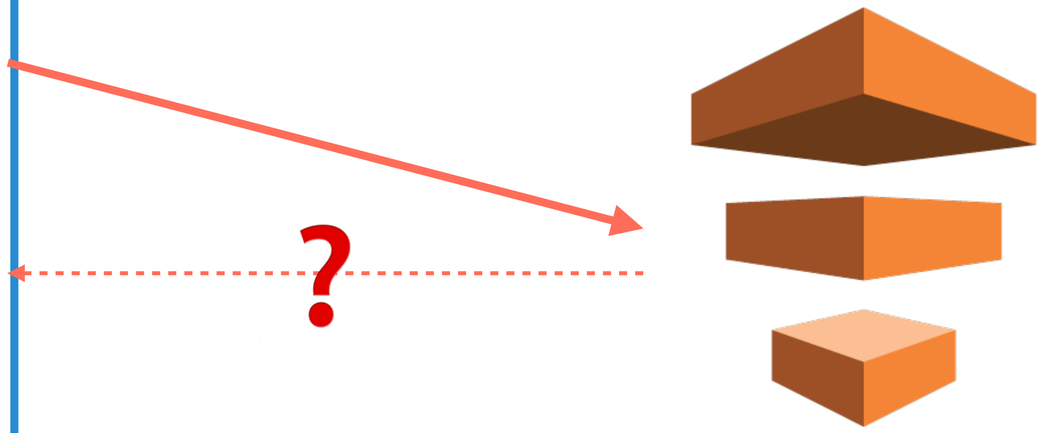
どうしたか？

解決方法

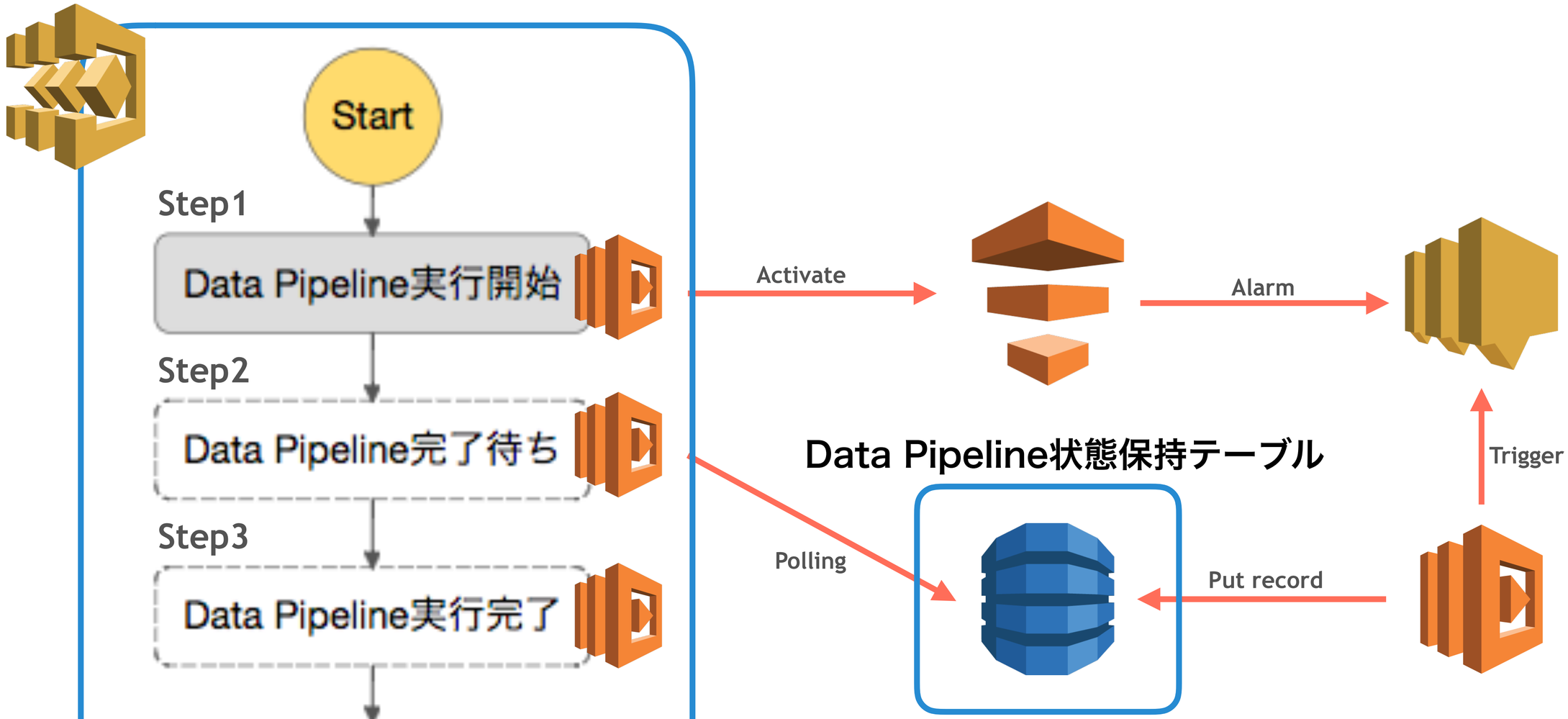
Data Pipelineの状態管理	 DataPipelineの状態を保持
LambdaのTimeout問題	 Retryを活用
洗い替え対象テーブルの 指定方法	 Parallel & Choiceを活用
キャッシュの ウォームアップ問題	 +  +  で高速にウォームアップ
ダウンタイム無しで 洗い替え	 At Timestamp指定でトリガー設定

Data Pipelineの
状態管理



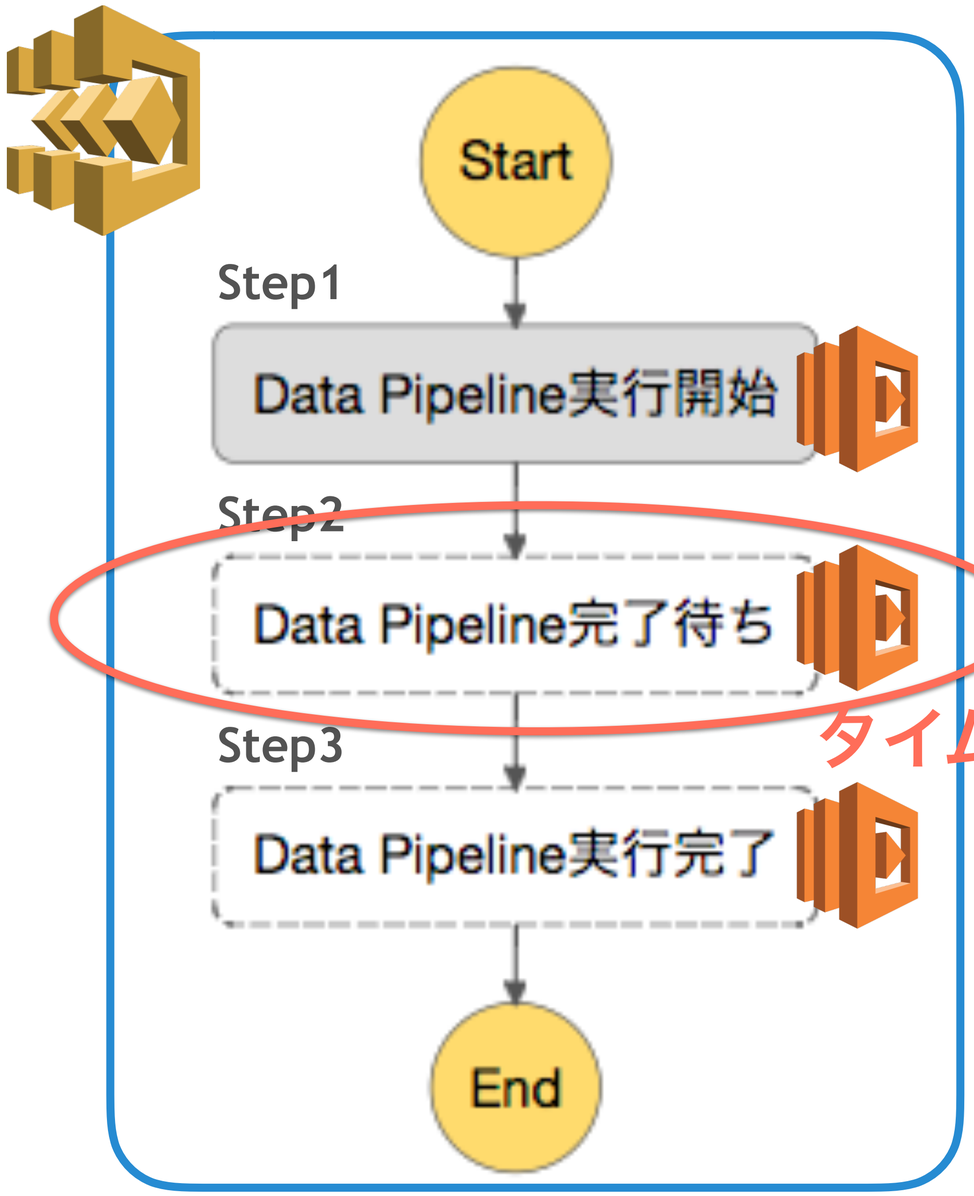


Step Functions
内部の状態しか知らない😢

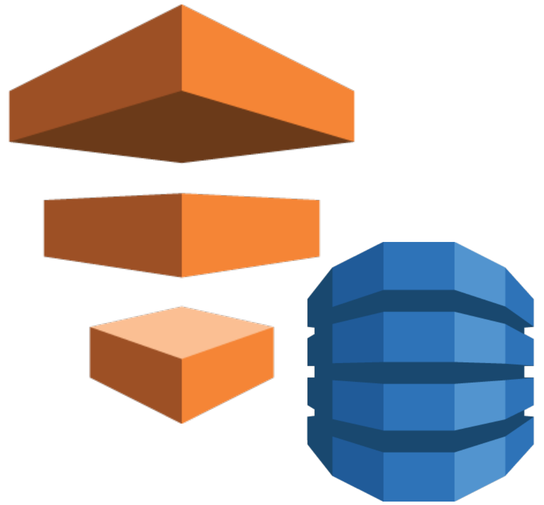


DynamoDBに
Data Pipelineの完了状態を持たせる

Lambdaの
タイムアウト問題

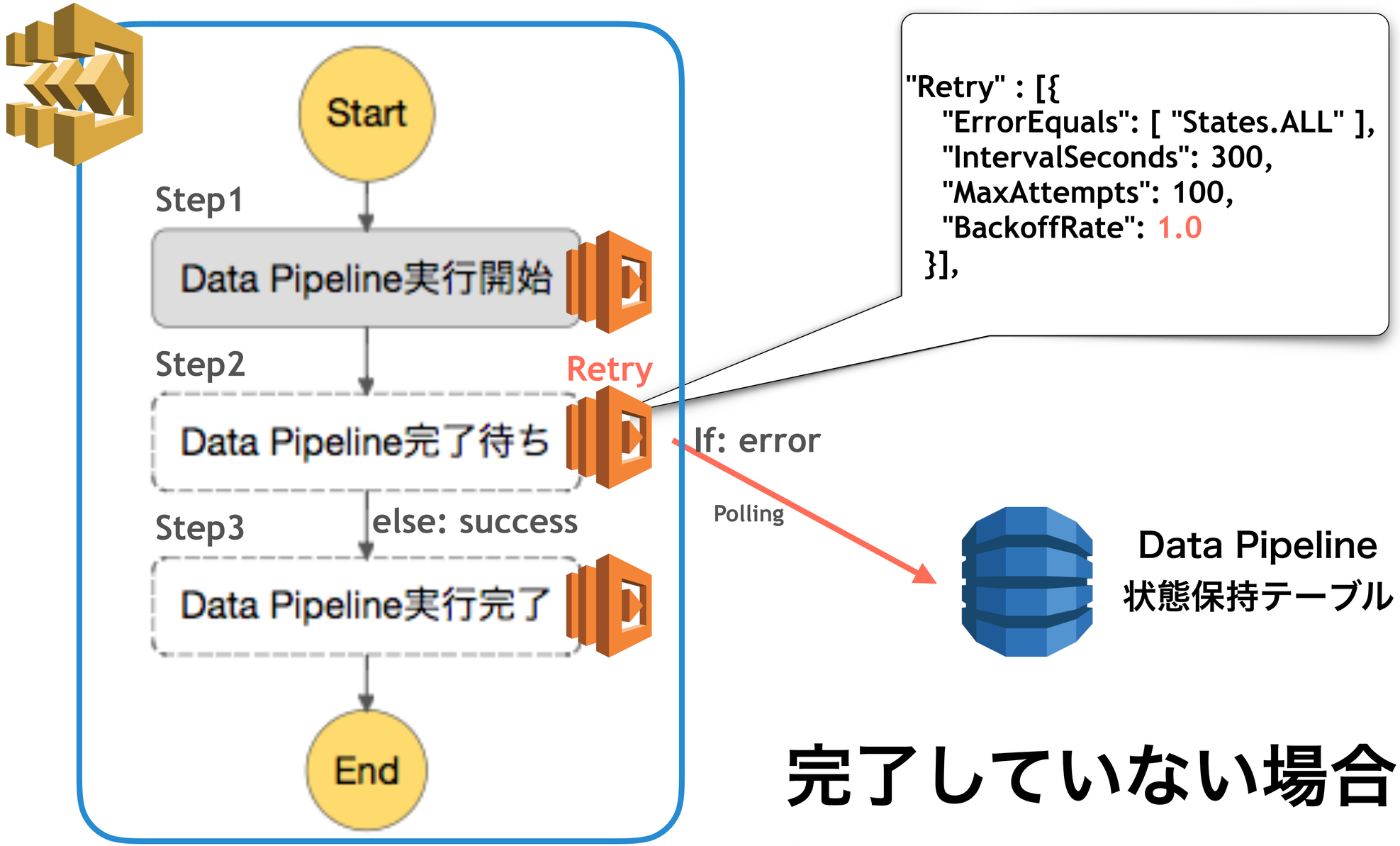


タイムアウト発生



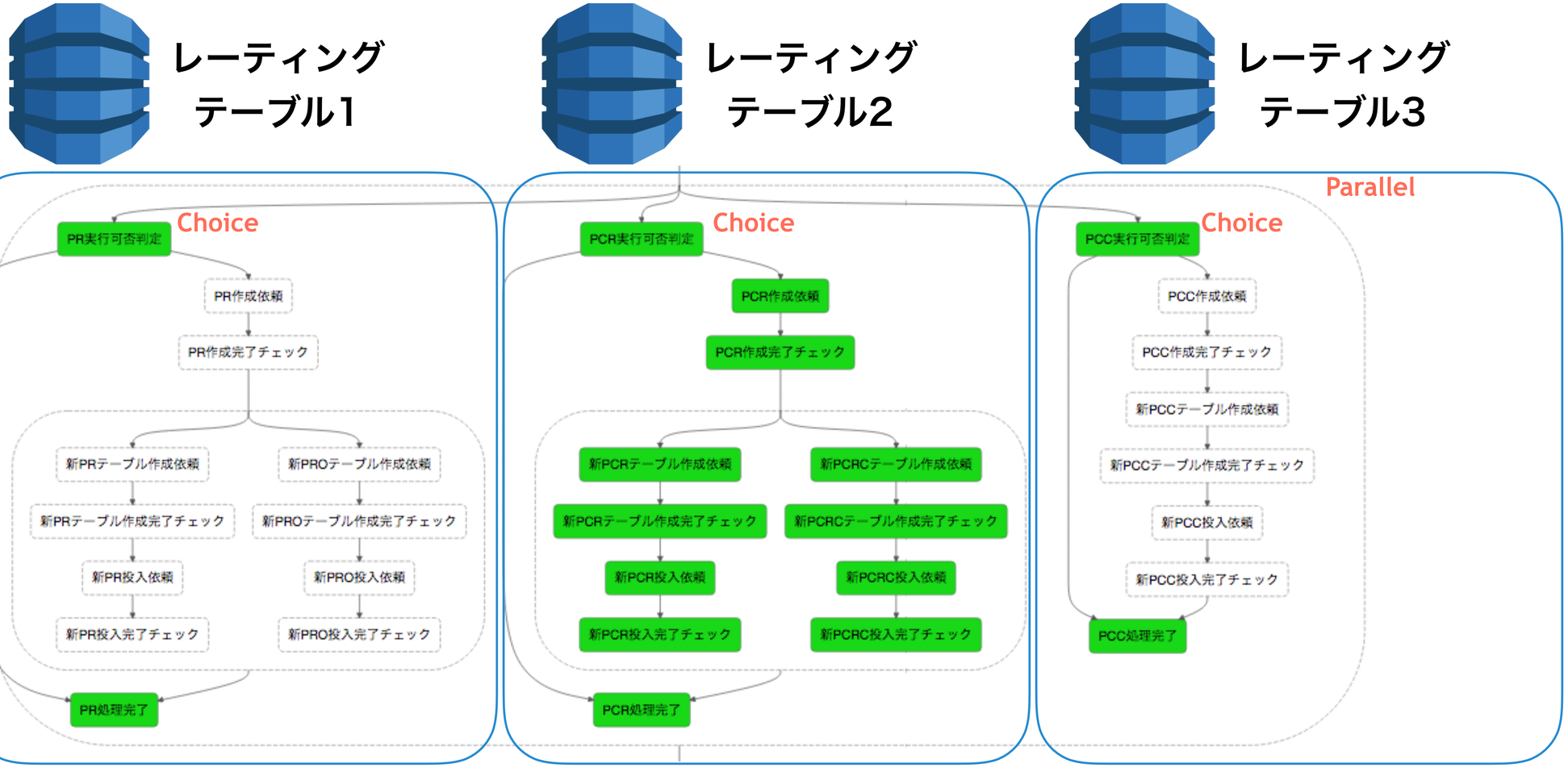
データのインポートに

数時間かかる😢

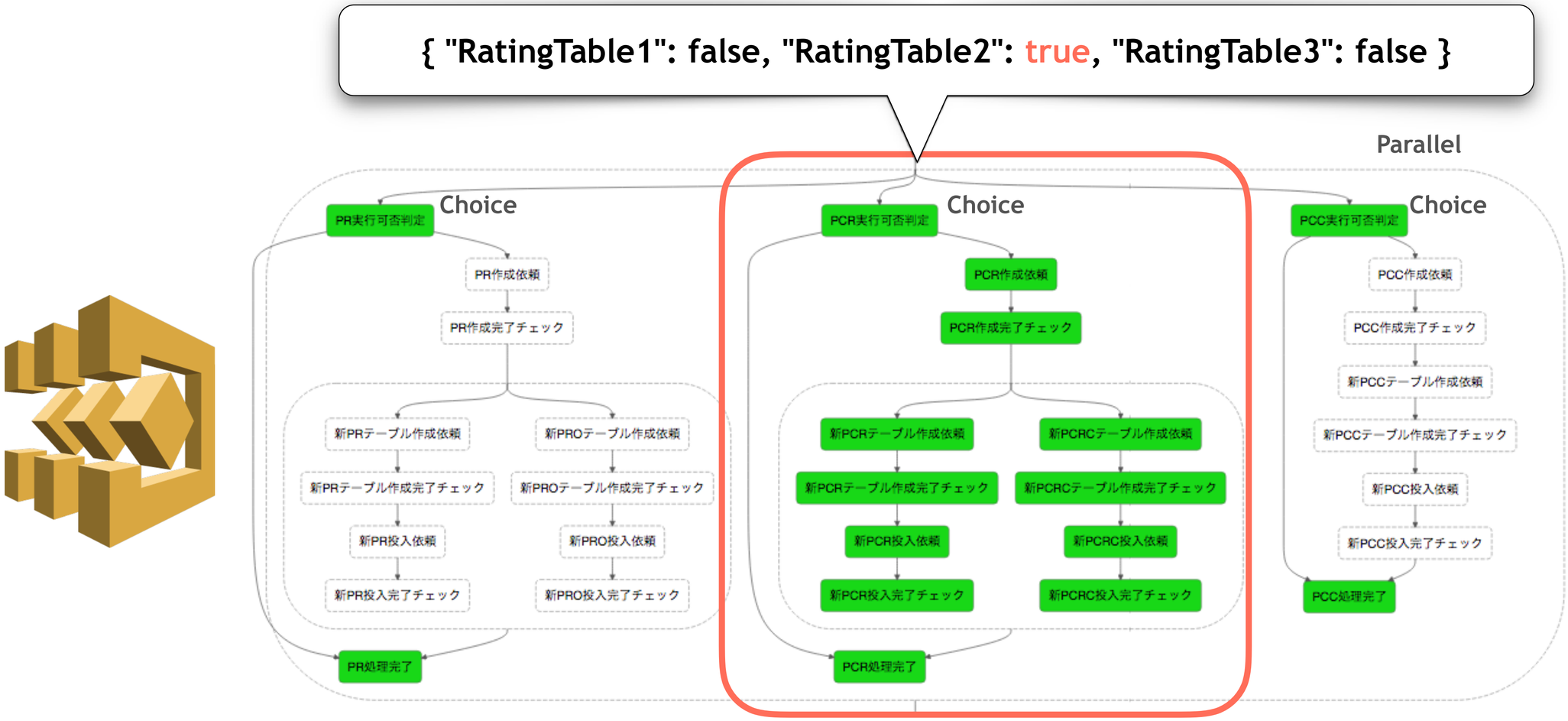


完了していない場合は
エラー扱いとし、定間隔でリトライ

洗い替え対象の指定方法

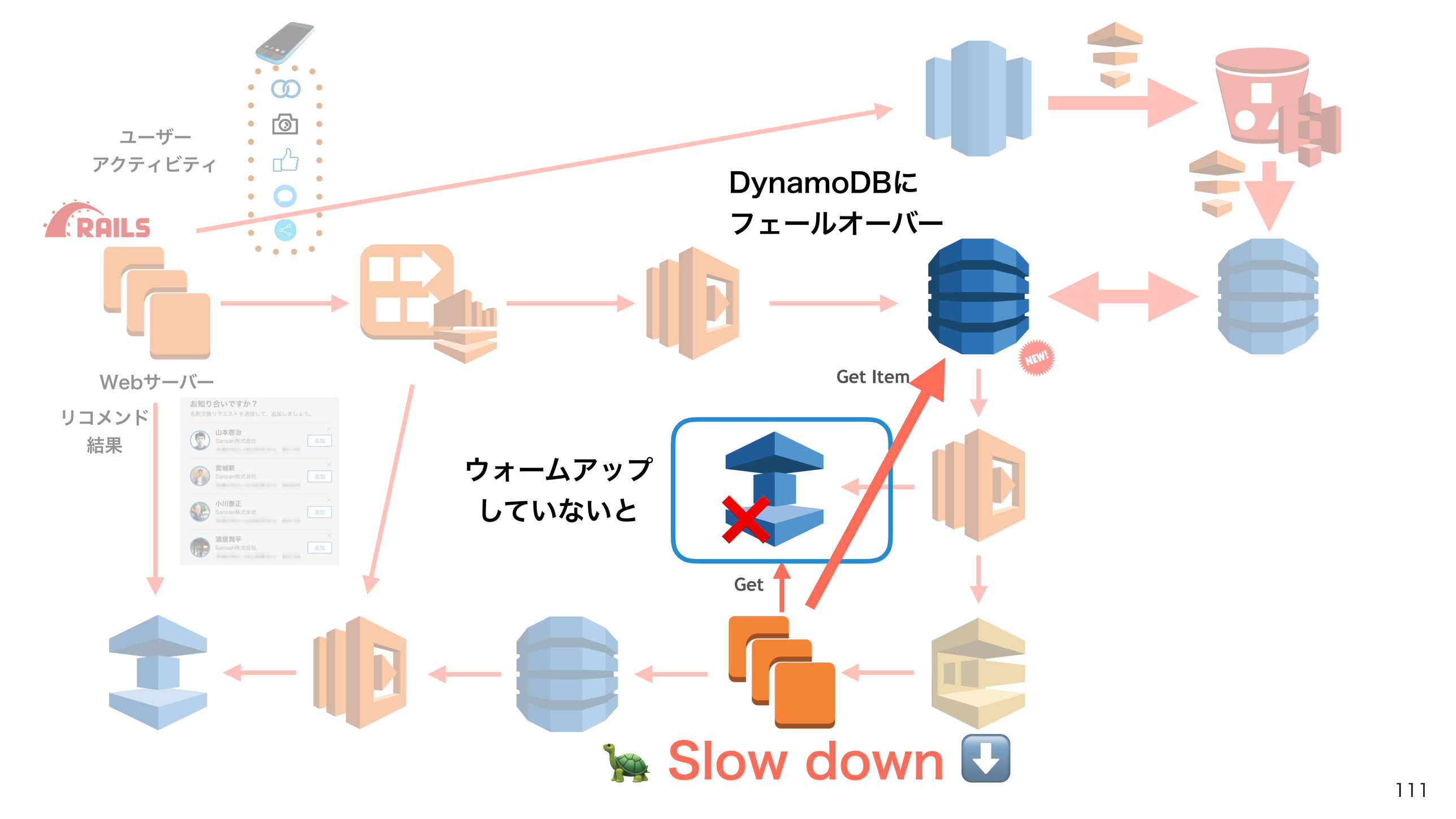


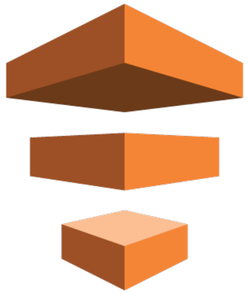
Parallelを使用し3並列に
Choiceで処理を分岐



入力するJSONを工夫し、指定テーブルのみ洗い替え

キャッシュの
ウォームアップ問題


ユーザー
アクティビティ
RAILS
Webサーバー
リコメンド
結果
お知り合いですか？
名刺交換リクエストを送信して、追加しましょう。
山本啓治
Sansan株式会社
追加
宮城新
Sansan株式会社
追加
小川泰正
Sansan株式会社
追加
酒居潤平
Sansan株式会社
追加
DynamoDBに
フェールオーバー
NEW!
Get Item
ウォームアップ
していないと
Get
Slow down



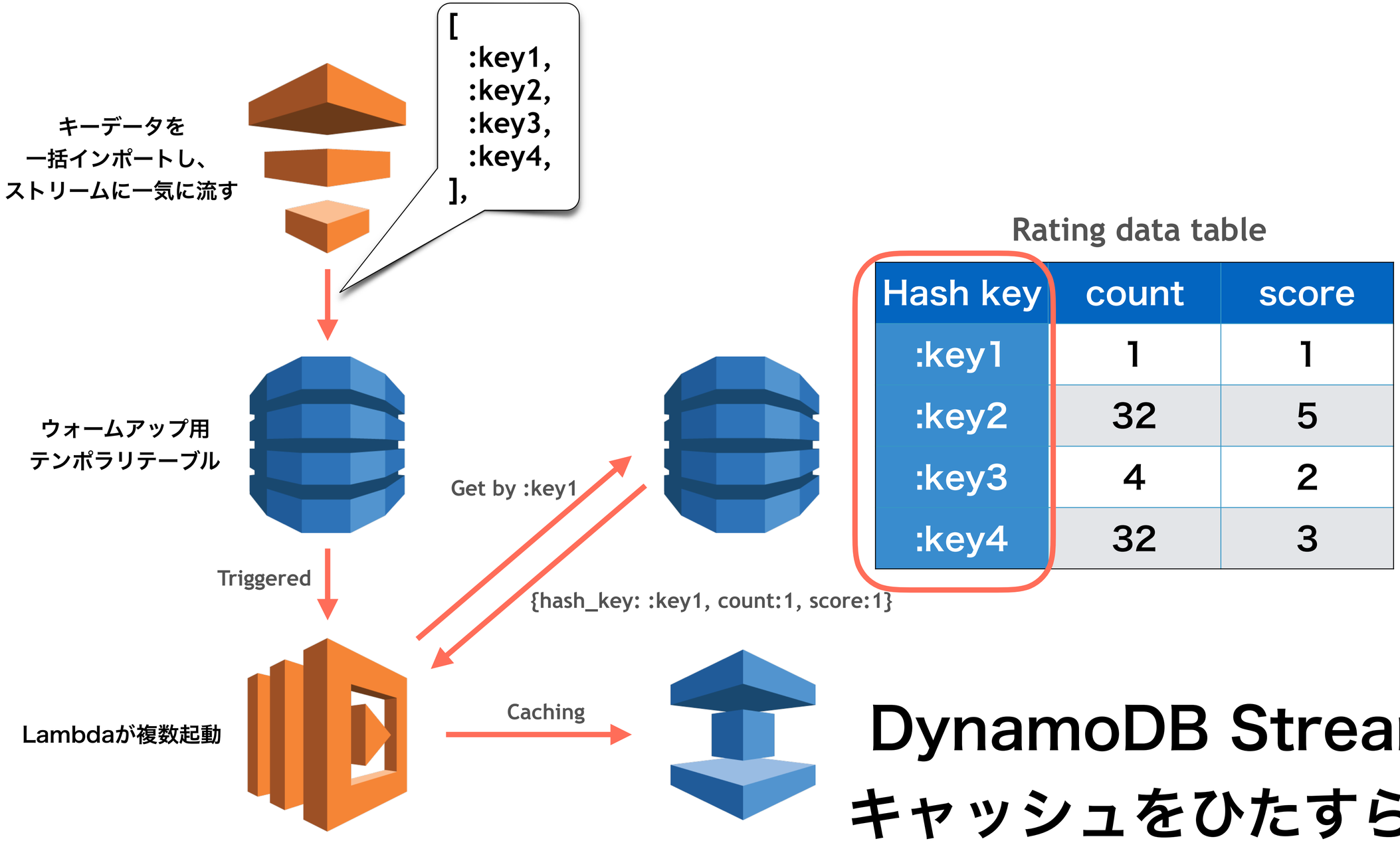
Data Pipeline

+

DynamoDB Stream

+

Lambda ✨



Rating data table

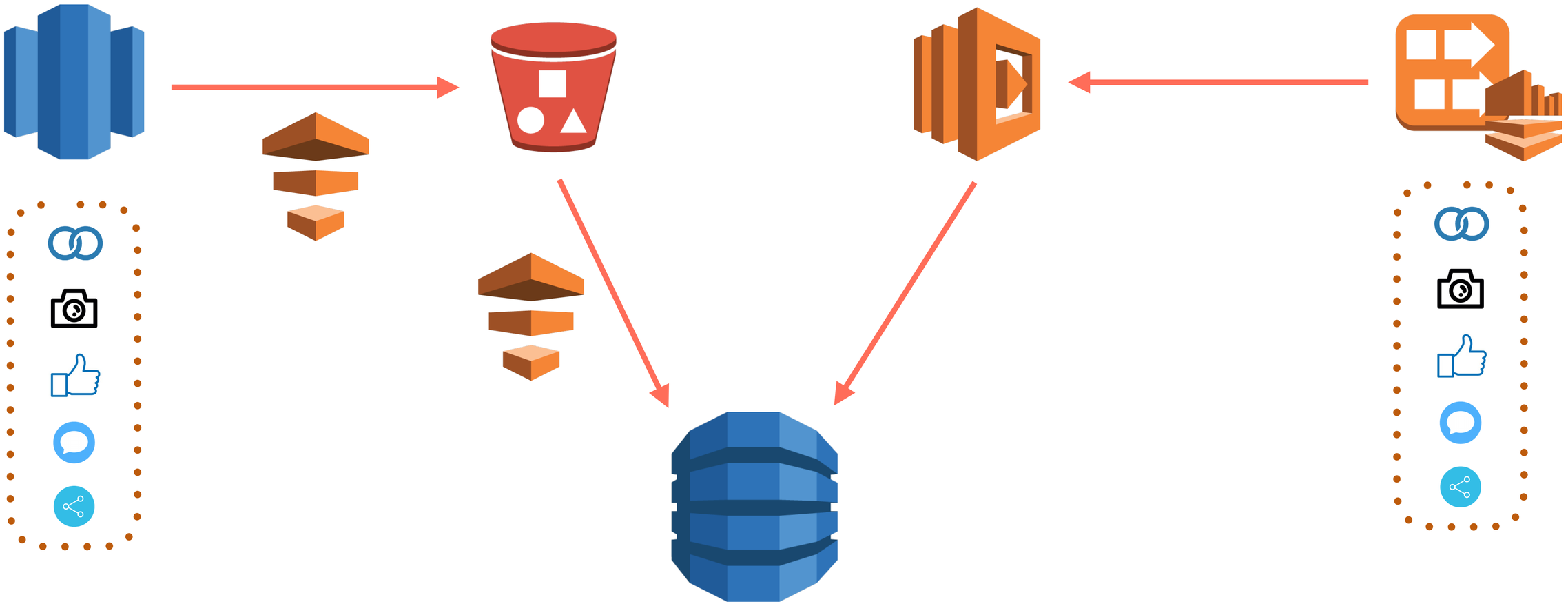
Hash key	count	score
:key1	1	1
:key2	32	5
:key3	4	2
:key4	32	3

DynamoDB Stream経由で
キャッシュをひたすら高速生成

ダウンタイム無し
で洗い替え

Lambda停止前

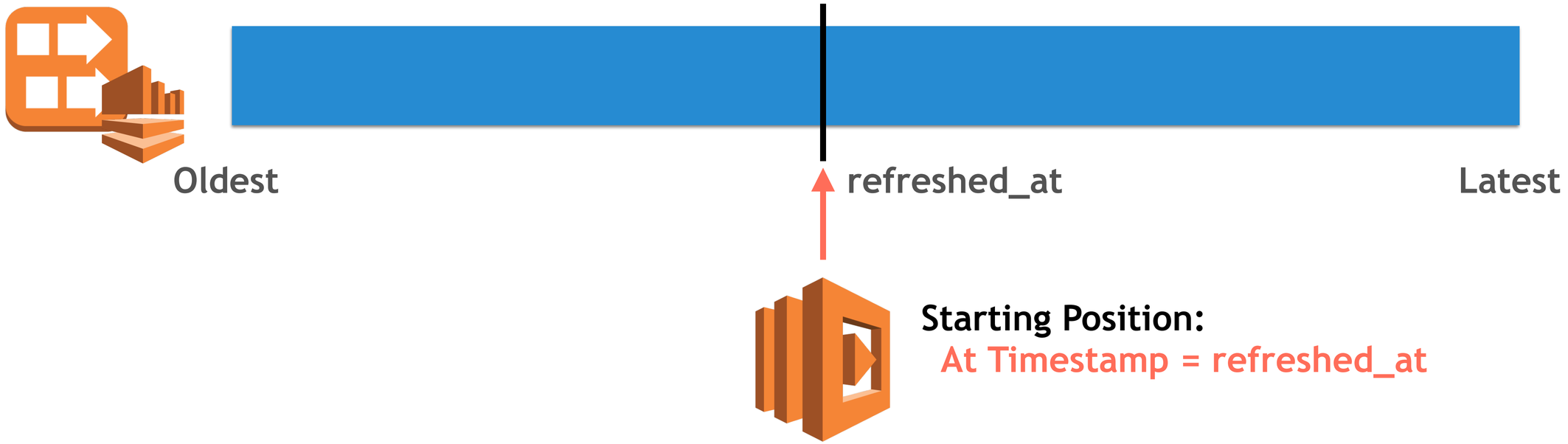
Lambda再開後



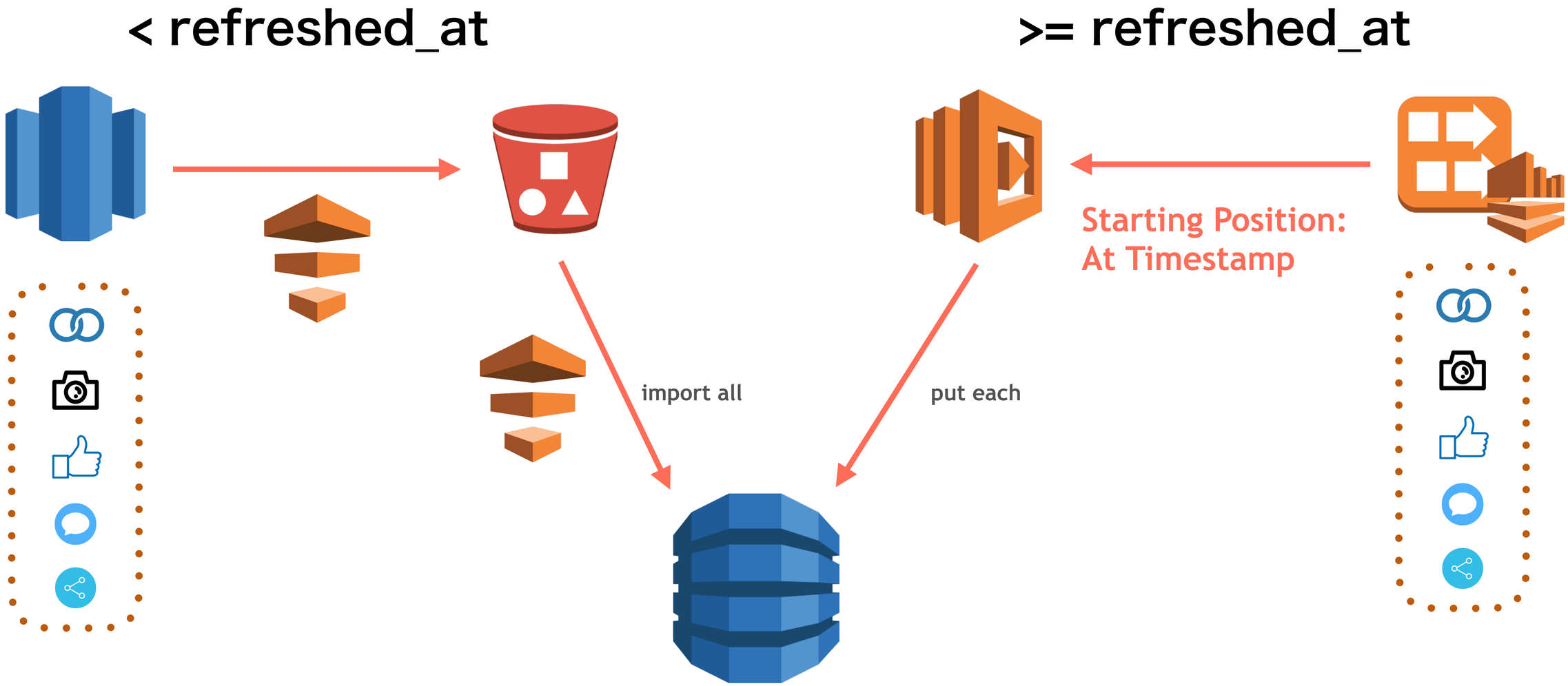
全過去ログからData Pipelineで
レーティングデータを一括生成

ストリームからLambdaで
レーティングデータを都度生成

At Timestampでトリガー設定



指定時刻(refreshed_at)よりも新しいものを処理



データー括生成とLambda処理の足並みを揃える


ついに
全自動化 😆

事前準備

- Lambda停止

レーティングデータ洗い替え

- 新データー括生成
- DynamoDBテーブル作成
- データインポート
- テーブル切り替え

キャッシュウォームアップ

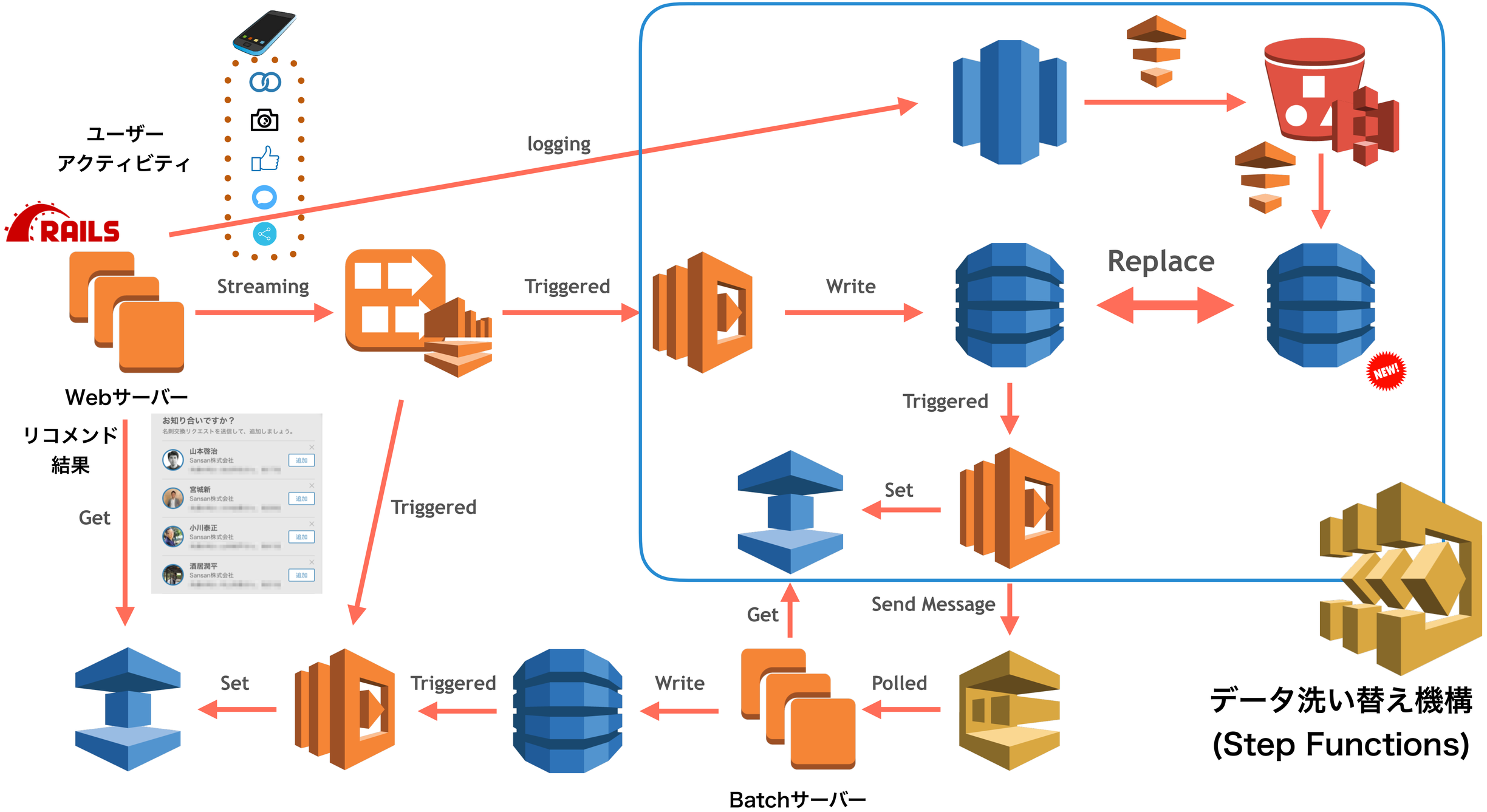
- ハッシュキーテーブル作成
- ハッシュキーインポート
- ウォームアップ実行

事後処理

- Lambda再開
- 不要リソース削除

- 計9500万件
- 約9時間

アーキテクチャの
全貌が 😆


ユーザー
アクティビティ
logging
RAILS
Streaming
Triggered
Write
Replace
NEW!
Webサーバー
Triggered
リコメンド
結果
Get
お知り合いですか？
名刺交換リクエストを送信して、追加しましょう。
山本啓治
Sansan株式会社
追加
宮城新
Sansan株式会社
追加
小川泰正
Sansan株式会社
追加
酒居潤平
Sansan株式会社
追加
Triggered
Set
Send Message
Get
Set
Triggered
Write
Polled
Batchサーバー
データ洗い替え機構
(Step Functions)

まとめ



サーバーレスの可能性

サーバーレスの利点

1. 気軽さ

2. スケーラビリティ

3. 柔軟性

正しく使うことの価値


Eight