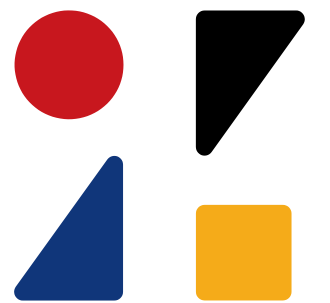


ログの収集と BigQueryを用いた分析

分析をもっと自由に

株式会社ZOZOテクノロジーズ
イノベーション推進部 エンジニア
武田修平

Copyright © ZOZO Technologies, Inc. All Rights Reserved.



ZOZO
Technologies

プロフィール



ZOZOテクノロジーズ
イノベーション推進部 エンジニア

武田修平 / sesta / せすた

Google Assistant 周り、Firebase、Flutter を触ってる

一児のパパで趣味は庭いじり

DroidKaigi で登壇が決まった

ついに個人でアプリを出したので好き勝手話せる

=> ファッション辞典

ZOZOTOWN



- 日本最大級のファッションショッピングサイト/アプリ
- 1,100以上のショップ、6,800以上のブランドの取り扱い
- 常時65万点以上の商品アイテム数と毎日平均3,100点以上の新着商品を掲載
- 即日配送サービス / ギフトラッピングサービス / ツケ払い など

<http://zozo.jp/>



WEAR

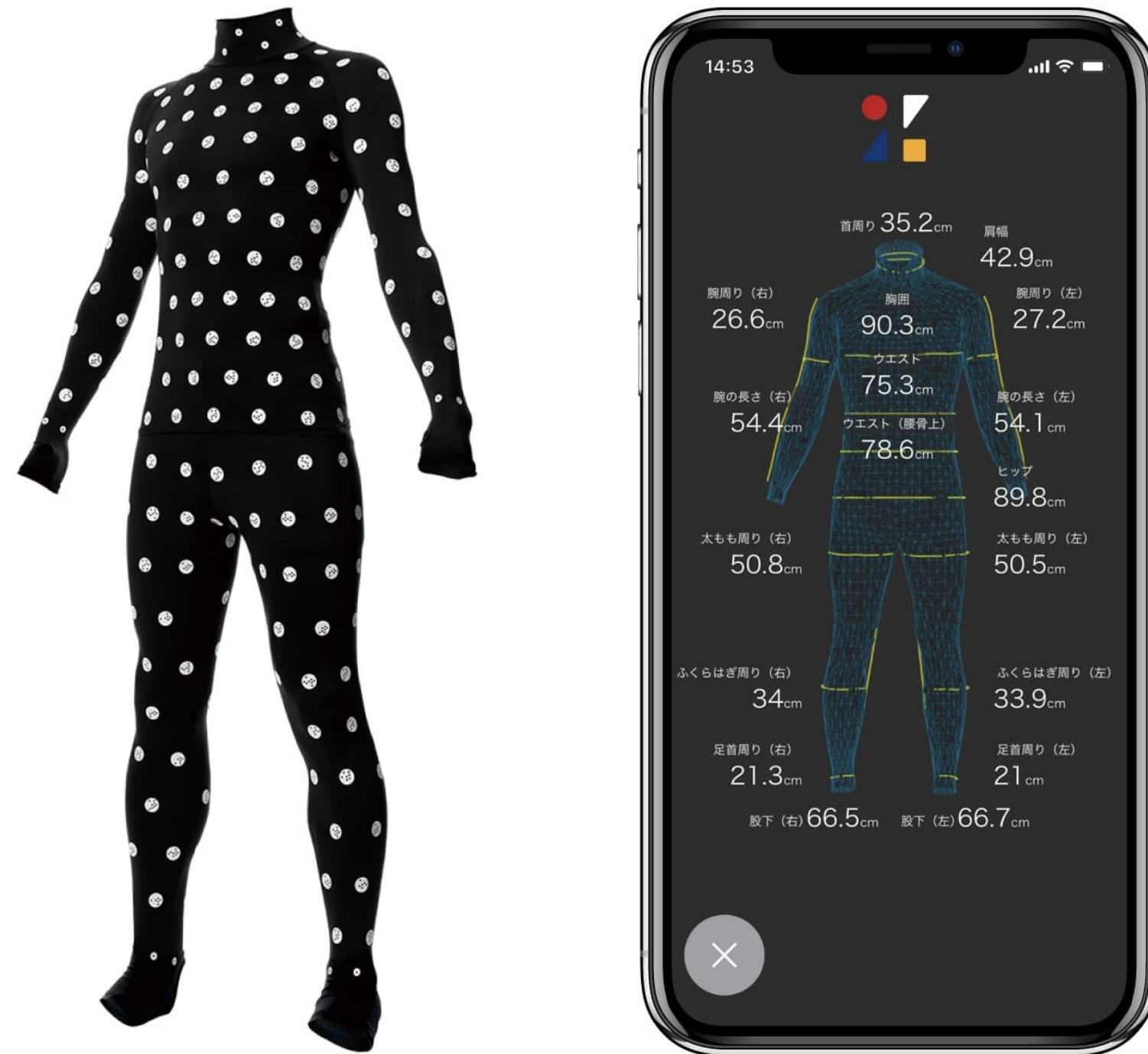


- 日本最大級のファッションコーディネートアプリ
- 1,100万ダウンロード突破、コーディネート投稿総数は800万件
- 全世界（App Store / Google playが利用可能な全ての国）でダウンロードが可能
- 10万人以上のフォロワーを持つユーザー（WEARISTA）も誕生

<https://wear.jp/>

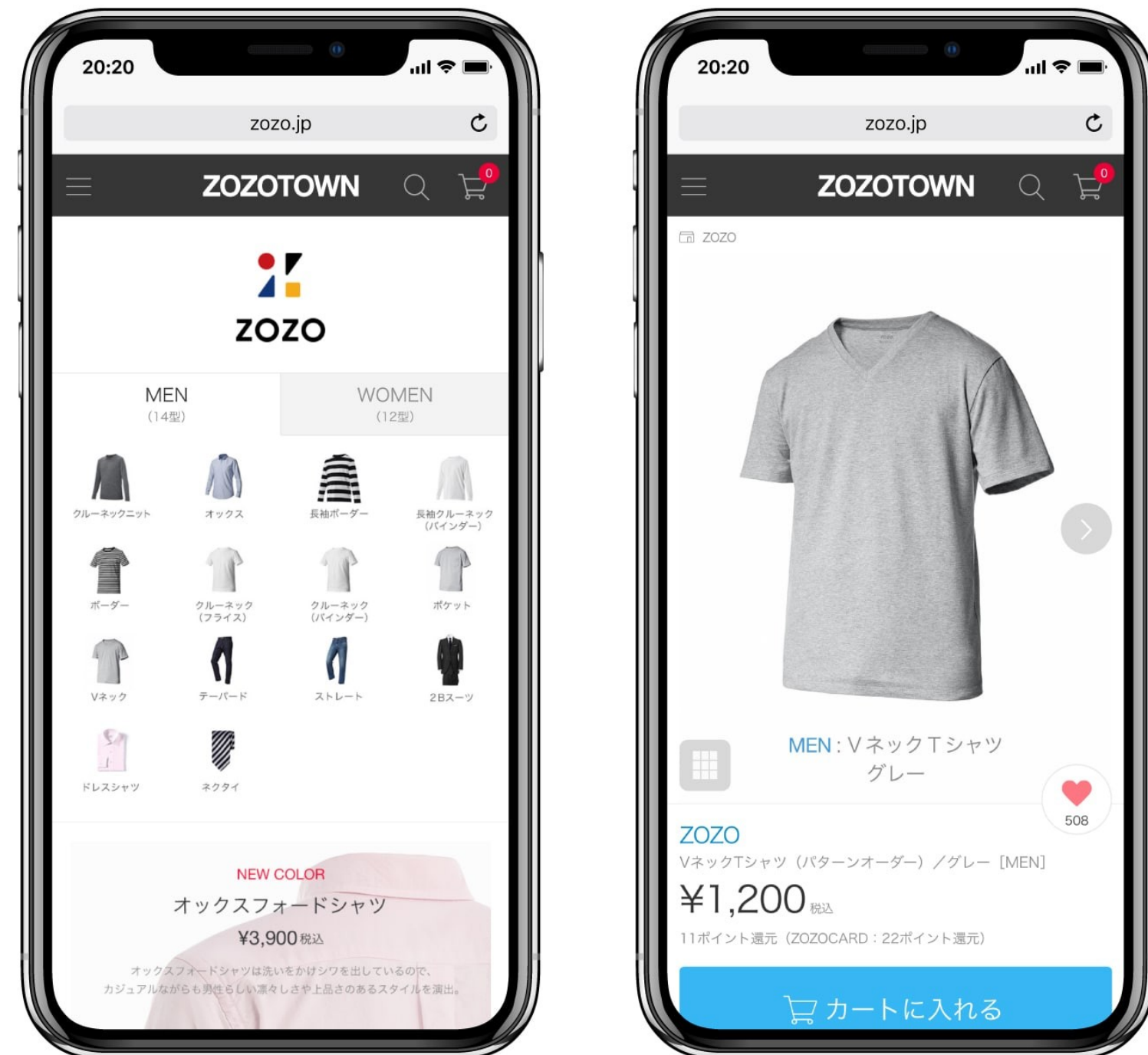


ZOZOSUIT



- 当社が独自に開発した採寸用ボディースーツ
- 全体に施されたドットマーカースマートフォンカメラで360度撮影することで、体型データを計測
- 計測した体型データは、瞬時に3Dモデル化され、ZOZOTOWNアプリに保存。3Dモデルはあらゆる角度に動かすことができ、体型を360度チェックすることが可能

<http://zozo.jp/zozosuit/>



○ 「ZOZOSUIT」で計測した体型データをもとに、一人ひとりの体型に合った「あなたサイズ」のアイテム

○ 「究極のフィット感」を実現したベーシックアイテムを提供
グローバルサイト「ZOZO.com」で海外展開

○ アイテム：Tシャツ、デニムパンツ、シャツ、ビジネススーツ、ネクタイ、ボーダーTシャツ、長袖クルーネックTシャツ など

<http://zozo.jp/pb/>

今回する話

- ・ 個人で開発した「ファッション辞典」の話です

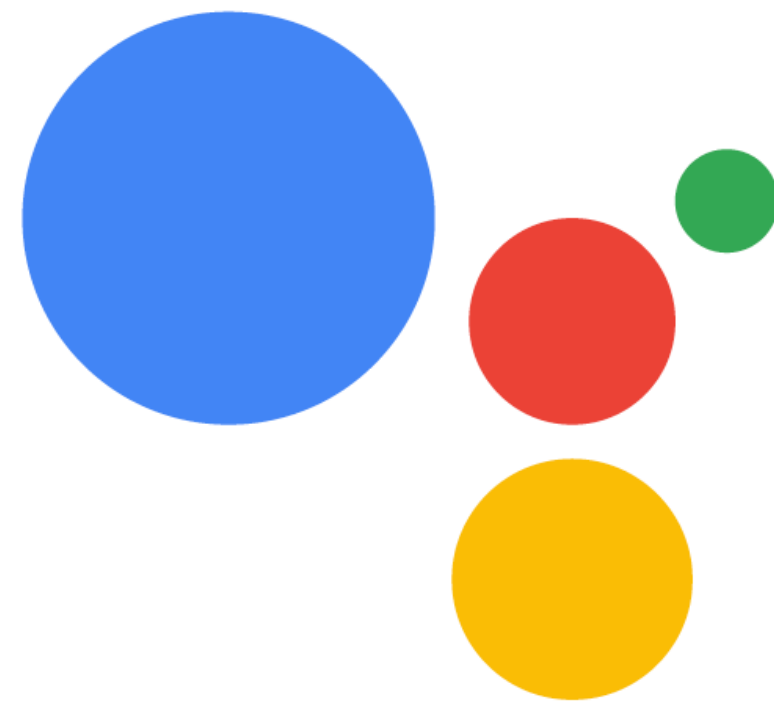


分析

分析していますか？

どうやって
分析していますか？

分析のよくある手段



Actions on Google の
Analytics 機能



Dialogflow の
Analytics や History 機能

AoG や Dialogflow でわかること

- ・ アプリが何回呼び出されたか？
- ・ エラー率はどれぐらいか？
- ・ インテントの呼び出される順番は？

👍 アプリ全体の様子 👍

AoG: Actions on google

AoG や Dialogflow でわからないこと

- ・多く使われている言葉は？
- ・話しかけられている時間帯は？
- ・利用頻度の高いユーザーの特徴は？

👉 一部の側面の様子 👉

要は
自由に分析したい

自由に分析とは

- ・好きなデータだけ抜き出したい
 - ・発話内容
 - ・時間帯
- ・集計の仕方をいじりたい
 - ・全体の平均
 - ・時間帯ごとの合計
 - ・ユーザーベースの中央値

=> **BigQuery**

自由に分析するまでの手順

1. 生データにアクセス
2. データをBigQueryに同期
3. BigQueryを存分に叩く

1. 生データにアクセス

生データのある場所

- Stackdriver Logging


The screenshot displays the Stackdriver Logging web interface. On the left is a sidebar with a menu containing 'Stackdriver Logging', 'ログ' (Logs), 'ログベースの指標' (Log-based metrics), 'エクスポート' (Export), and 'ログの取り込み' (Log ingestion). The main content area has a top bar with buttons for '指標を作成' (Create metric), 'エクスポートを作成' (Create export), and a refresh button. Below this is a search bar labeled 'ラベルまたはテキスト検索でフィルタ' (Filter by label or text). The main content area shows a filter for 'Google アシスタントの操作' (Google Assistant operations), a dropdown for 'すべてのログ' (All logs), a 'すべて' (All) button, and a time range selector set to '過去 1 時間' (Last 1 hour). A message states '過去 1 時間 から 11:35 まで (JST) のログを表示しています' (Showing logs from the last 1 hour until 11:35 (JST)). At the bottom, there is a yellow bar with a downward arrow and a button labeled '古いログを読み込む' (Load old logs).

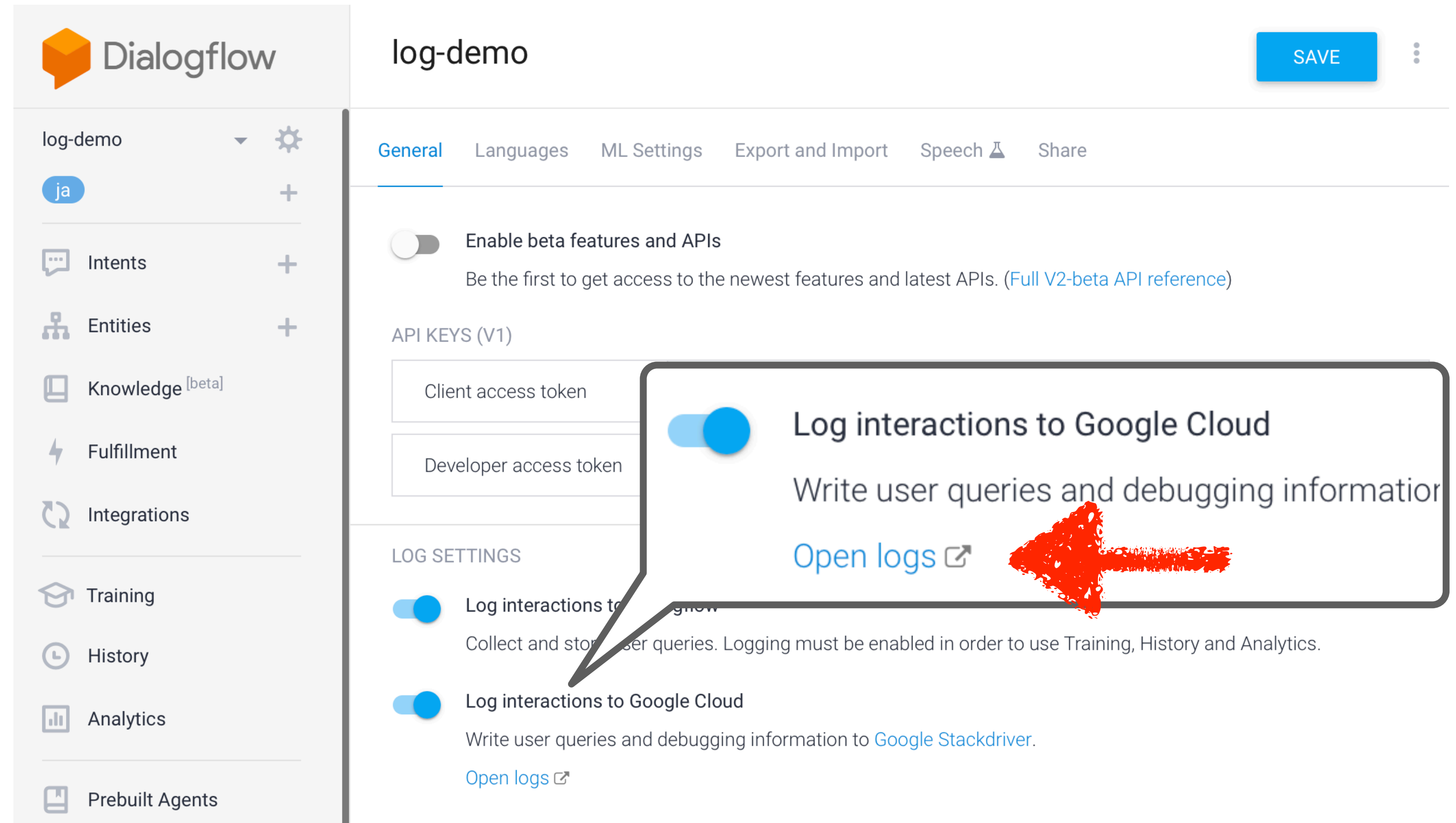
Stackdriver Logging とは

- Google Cloud の管理ツールの1つで、ログの格納や検索ができる



Stackdriver Logging に行く方法


- Dialogflow
 - => Setting()
 - => General



Dialogflow

log-demo

SAVE

General Languages ML Settings Export and Import Speech  Share


☐ Enable beta features and APIs
Be the first to get access to the newest features and latest APIs. ([Full V2-beta API reference](#))


API KEYS (V1)

Client access token

Developer access token

LOG SETTINGS

☒ Log interactions to Google Cloud
Write user queries and debugging information to [Google Stackdriver](#).
[Open logs](#) 

☒ Log interactions to Google Cloud
Write user queries and debugging information to [Google Stackdriver](#).
[Open logs](#) 

生データの様子

- 「Googleアシスタントの操作」
で絞ると出てきます

過去1時間 から 11:41 まで (JST) のログを表示しています ログをタ

↓ 古いログを読み込む

▶ 2018-11-21 10:46:03.074 JST Sending request with post data: {"user":{"userId": [REDACTED]}

▶ 2018-11-21 10:46:03.179 JST Received response from agent with body: HTTP/1.1 200 OK Server: r

▼ 2018-11-21 10:55:55.559 JST Sending request with post data: {"user":

すべて展

2018-11-21 10:46:03.074 JST Sending request with post data: {"user":{"userId": [REDACTED]}

2018-11-21 10:46:03.179 JST Received response from agent with body: HTTP/1.1 200

timestamp: "2018-11-21T01:55:55.559935827Z"

2. データをBigQueryに同期

同期の方法

- ・ Loggingのエクスポートを使う

 Stackdriver Logging	エクスポート  エクスポートを作成  削除						
 ログ	ログのエクスポート						
 ログベースの指標	 シンクをフィルタ 						
 エクスポート	<table><thead><tr><th><input checked="" type="checkbox"/> シンク名</th><th>出力先</th><th>書き込み ID</th></tr></thead><tbody><tr><td><input checked="" type="checkbox"/> log_demo</td><td>bigquery.googleapis.com/projects/log_demo/datasets/log_demo_request</td><td></td></tr></tbody></table>	<input checked="" type="checkbox"/> シンク名	出力先	書き込み ID	<input checked="" type="checkbox"/> log_demo	bigquery.googleapis.com/projects/log_demo/datasets/log_demo_request	
<input checked="" type="checkbox"/> シンク名	出力先	書き込み ID					
<input checked="" type="checkbox"/> log_demo	bigquery.googleapis.com/projects/log_demo/datasets/log_demo_request						
 ログの取り込み							

同期の設定の様子

指標を作成

↑

↺

▶

1 resource.type="assistant_action"

2 NOT "Crawler"

3 "Sending request"

「Control+Space キー」で候補の

フィルタを送信

過去 1 時間

現在の位置に移動

過去 1 時間 から 16:04 まで (JST) のログを表示し

↓ 過去 1 時間内に、現在のフィルタと一致する古いエントリがありま

▶ λ 2018-11-21 15:36:27.944 JST Sending request with post

▶ λ 2018-11-21 15:36:43.559 JST Sending request with post

↑

新しいログを読み込む

× エクスポートの編集

シンク名

log_demo_request

シンクサービス

BigQuery

シンクのエクスポート先

log_demo_request

ログシンクを作成すると、今後一致するログは選択したエクスポート先にエクスポートされます。

シンクを作成

キャンセル

必要なログだけにフィルター

1. アシスタント関係のログに絞る

2. クローラーを除外

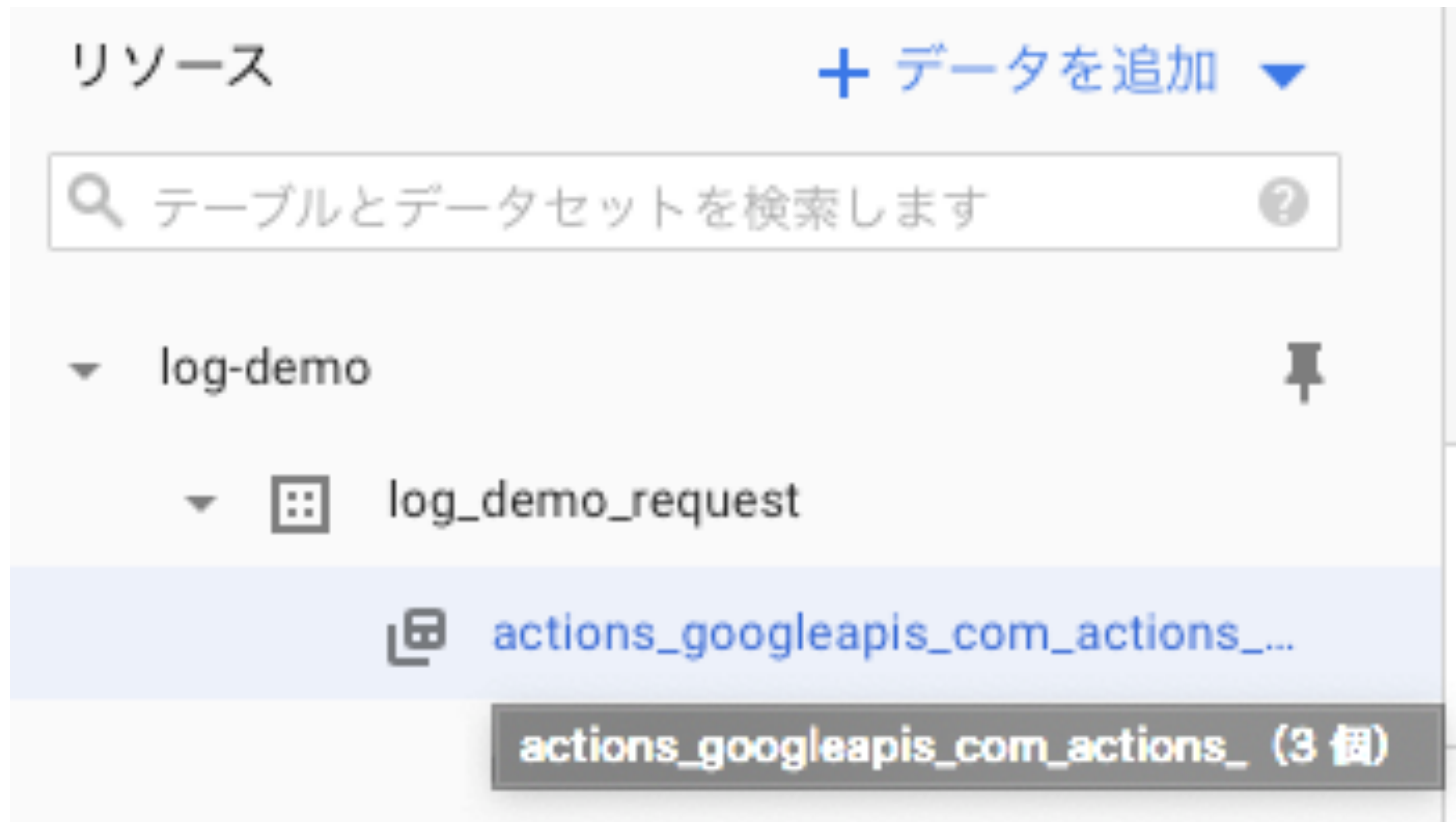
- ・ アプリの稼動状態を確認してる
リクエストが定期的にくるため

1	resource.type="assistant_action"
2	NOT "Crawler"
3	"Sending request"

3. Dialogflowに送っているログに絞る

- ・ ユーザーの発話内容が入っているのはこっちなので

🎉 シンク結果 🎉



3. BigQueryを存分に叩く

好きに叩きましょう

- ・「テーブルをクエリ」を押せばテンプレが出るのでそれをいい感じにいじる

```
1 SELECT *
2 FROM `log-demo.log_demo_request.actions_googleapis_com_actions_20181121`
3 LIMIT 1000
```

クエリを実行 クエリを保存 ビューを保存 展開

actions...tions_ 2018-11-21 🔍 テーブルをクエリ

例1) 発話された言葉ランキング

```
WITH formatted_json as (  
  SELECT  
    SUBSTR(textPayload, 33, LENGTH(textPayload) - 33) as json, # json以外の部分を切り取る  
    timestamp,  
    resource.labels.action_id as action  
  FROM `log-demo.log_demo_request.actions_googleapis_com_actions_*`  
  WHERE  
    _TABLE_SUFFIX >= '20181101'  
    AND _TABLE_SUFFIX <= '20181107'  
)  
data as (  
  SELECT  
    JSON_EXTRACT_SCALAR(json, "$.conversation.conversationId") as conversationId, # スキル呼び出しごとのID  
    JSON_EXTRACT_SCALAR(json, "$.user.userId") as userId, # ユーザーID、アカウント連携してない場合null  
    JSON_EXTRACT_SCALAR(json, "$.inputs[0].rawInputs[0].query") as query, # 発話内容  
    JSON_EXTRACT_SCALAR(json, "$.isInSandbox") as isSandbox, # 開発アカウントかどうか  
    action, # Actions on Googleで認識できたアクション名  
    FORMAT_TIMESTAMP('%Y-%m-%d %H:%M:%S', timestamp, 'Asia/Tokyo') as date # 発話した時間  
  FROM formatted_json  
)  
  
SELECT query, COUNT (*) as count  
FROM data  
WHERE isSandbox IS NULL  
AND query IS NOT NULL  
GROUP BY query  
ORDER BY count DESC
```

行	query	count
1	ファッション辞典につないで	11
2	ファッション辞典を開いて	6
3	ファッション辞典でダッフルコート調べて	3
4	キュロット	3
5	香氏	2
6	ファッション	2
7	スカート	2
8	パンツ	2
9	声 サイコロ	1
10	シャツ	1

例2) 時間帯ごとのスキル呼び出し回数

```
WITH formatted_json as (  
  SELECT  
    SUBSTR(textPayload, 33, LENGTH(textPayload) - 33) as json, # json以外の部分を切り取る  
    timestamp,  
    resource.labels.action_id as action  
  FROM `log-demo.log_demo_request.actions_googleapis_com_actions_*`  
  WHERE  
    _TABLE_SUFFIX >= '20181101'  
    AND _TABLE_SUFFIX <= '20181107'  
)  
data as (  
  SELECT  
    JSON_EXTRACT_SCALAR(json, "$.conversation.conversationId") as conversationId, # スキル呼び出しごとのID  
    JSON_EXTRACT_SCALAR(json, "$.user.userId") as userId, # ユーザーID、アカウント連携してない場合null  
    JSON_EXTRACT_SCALAR(json, "$.inputs[0].rawInputs[0].query") as query, # 発話内容  
    JSON_EXTRACT_SCALAR(json, "$.isInSandbox") as isSandbox, # 開発アカウントかどうか  
    action, # Actions on Googleで認識できたアクション名  
    FORMAT_TIMESTAMP('%H', timestamp, 'Asia/Tokyo') as hour # 発話した時間  
  FROM formatted_json  
)  
  
SELECT hour, COUNT (DISTINCT conversationId) as count  
FROM data  
WHERE isSandbox IS NULL  
  AND query IS NOT NULL  
GROUP BY hour  
ORDER BY hour
```

行	hour	count	
1	00	239	
2	01	122	
3	02	31	
4	03	57	
5	04	18	
6	06	18	
7	07	70	
8	08	161	
9	09	317	
10	10	356	

※ count は迫力を出すために盛っています

好き勝手分析できるようになったのか

- ・好きなデータだけ抜き出したい => JSONの中にあるものならいける
 - ・発話内容
 - ・時間帯
- ・集計の仕方をいじりたい => StandardSQL がすごいので大丈夫
 - ・全体の平均
 - ・時間帯ごとの合計
 - ・ユーザーベースの中央値

Plus Ultra

- さらに高みへ -

BigQueryで広がる可能性

- ・ 他のデータと組み合わせる
 - ・ Firestoreに保存してるユーザーの性別
 - ・ Google Analytics で取った行動履歴
- ・ 色々なエクスポート
 - ・ Google スプレッドシート、Google Data Studio
 - ・ Re:dash

できそうなこと

- ・ 形態素解析
 - ・ どこかしらに MeCab なりを嚙ませればいけるはず
- ・ 会話のプロセスの分析
 - ・ Actions on Google との組み合わせが良いかも

まとめ

- ・ 把握したいことに合わせて AoG なのか Dialogflow なのか
または、BigQuery なのかを選ぶようにしたい
- ・ 思ったより簡単に集計できるようになった
 - ・ Googleに体が満たされていく感覚
- ・ プロダクトを作りに興味がある方は声をかけてください
 - ・ **一緒に世界をカッコよくしましょう**

END