

EDA of Billion Dollar Retail Store Outlet

- Firebase Messaging Data

This is an EDA done on the Android logs of “Firebase Messaging” data for a Billion Dollar Retail Store Outlet.

Field name	Type	Description
event_timestamp	TIMESTAMP	Timestamp since Epoch when event started on client device (trace start, network start, etc). Timestamp since Epoch when event started on client device (trace start, network start, etc)
app_display_version	STRING	The display version of the application. VersionName for Android; CFBundleShortVersionString for iOS. E.g. "4.1.7". The display version of the application. VersionName for Android; CFBundleShortVersionString for iOS. E.g. "4.1.7".
app_build_version	STRING	The build version of the application. VersionCode for Android; CFBundleVersion for iOS. E.g. "1523456". The build version of the application. VersionCode for Android; CFBundleVersion for iOS. E.g. "1523456".
os_version	STRING	Android API level or iOS version. E.g. "26" (Android) or "11.4" (iOS). Android API level or iOS version. E.g. "26" (Android) or "11.4" (iOS).
device_name	STRING	Name of the client device. E.g. "Google Pixel". Name of the client device. E.g. "Google Pixel".
country	STRING	2-letter country code of the country from which the event took place. E.g. "US" or "ZZ" (unknown). 2-letter country code of the country from which the event took place. E.g. "US" or "ZZ" (unknown).
carrier	STRING	Carrier of the client device.
radio_type	STRING	Active radio type when the event took place. E.g. "WIFI". Active radio type when the event took place. E.g. "WIFI".
custom_attributes	RECORD	All custom attributes attached to this event. All custom attributes attached to this event.
event_type	STRING	Type of the event. Possible values: DURATION_TRACE (including app start, foreground, background, and all developer instrumented traces); SCREEN_TRACE (traces spanning the lifetime of screens); TRACE_METRIC (developer instrumented metrics that are associated with traces, previously known as counters); NETWORK_REQUEST. Type of the event. Possible values: DURATION_TRACE (including app start, foreground, background, and all developer instrumented traces); SCREEN_TRACE (traces spanning the lifetime of screens); TRACE_METRIC (developer instrumented metrics that are associated with traces, previously known as counters); NETWORK_REQUEST.
event_name	STRING	Name of the event. For DURATION_TRACE, the trace name. For SCREEN_TRACE: "_st_" followed by the trace name. For NETWORK_REQUEST: the network request url pattern. For TRACE_METRIC: the metric name. Name of the event. For DURATION_TRACE, the trace name. For SCREEN_TRACE: "_st_" followed by the trace name. For NETWORK_REQUEST: the network request url pattern. For TRACE_METRIC: the metric name.

parent_trace_name	STRING	Name of the parent trace that carries the trace metric. Only present for TRACE_METRIC. Name of the parent trace that carries the trace metric. Only present for TRACE_METRIC.
trace_info	RECORD	Only present for DURATION_TRACE, SCREEN_TRACE, TRACE_METRIC. Only present for DURATION_TRACE, SCREEN_TRACE, TRACE_METRIC.
duration_us	INTEGER	For DURATION_TRACE, SCREEN_TRACE: duration that this trace lasts. For TRACE_METRIC: duration that the parent trace lasts. Unit: microsecond. For DURATION_TRACE, SCREEN_TRACE: duration that this trace lasts. For TRACE_METRIC: duration that the parent trace lasts. Unit: microsecond.
screen_info	RECORD	Only present for SCREEN_TRACE. Only present for SCREEN_TRACE.
slow_frame_ratio	FLOAT	The ratio of slow frames for this screen trace, between 0 and 1. E.g. a value of 0.05 means 5% of the frames for this screen instance took more than 16ms to load. The ratio of slow frames for this screen trace, between 0 and 1. E.g. a value of 0.05 means 5% of the frames for this screen instance took more than 16ms to load.
frozen_frame_ratio	FLOAT	The ratio of frozen frames for this screen trace, between 0 and 1. E.g. a value of 0.05 means 5% of the frames for this screen instance took more than 700ms to load. The ratio of frozen frames for this screen trace, between 0 and 1. E.g. a value of 0.05 means 5% of the frames for this screen instance took more than 700ms to load.
metric_info	RECORD	Only present for TRACE_METRIC. Only present for TRACE_METRIC.
metric_value	INTEGER	Value of the trace metric.
network_info	RECORD	Only present for NETWORK_REQUEST. Only present for NETWORK_REQUEST.
response_code	INTEGER	HTTP response code for the network response. E.g. 200 or 404. HTTP response code for the network response. E.g. 200 or 404.
response_mime_type	STRING	MIME type of the network response. E.g. "text/html". MIME type of the network response. E.g. "text/html".
request_http_method	STRING	HTTP method of the network request. E.g. "GET" or "POST". HTTP method of the network request. E.g. "GET" or "POST".
request_payload_bytes	INTEGER	Size of the network request payload. Unit: byte. Size of the network request payload. Unit: byte.
response_payload_bytes	INTEGER	Size of the network response payload. Unit: byte. Size of the network response payload. Unit: byte.
request_completed_time_us	INTEGER	Microseconds after event_timestamp when network request sending is complete. Unit: microsecond. Microseconds after event_timestamp when network request sending is complete. Unit: microsecond.
response_initiated_time_us	INTEGER	Microseconds after event_timestamp when network response is initiated. Unit: microsecond. Microseconds after event_timestamp when network response is initiated. Unit: microsecond.
response_completed_time_us	INTEGER	Microseconds after event_timestamp when network response is completed. Unit: microsecond. Microseconds after event_timestamp when network response is completed. Unit: microsecond.

Big Query Console

<https://console.cloud.google.com/bigquery?inv=1&inv=AbyErg&project=order-fulfillment---prod>

Big Query Documentation

https://cloud.google.com/bigquery/docs/reference/standard-sql/date_functions#extract

How to setup data in BigQuery:

- All Firebase data is Stored in Google Cloud infrastructure.
- Firebase retains performance data for 30 days.
- **Go to Firebase Console** - <https://console.firebase.google.com>
- Go back to Firebase → **Project Settings** → **Integrations**
- Find **BigQuery** and click "**Link**".
- After linking, Firebase will begin exporting daily **Analytics event data** into BigQuery automatically.
- Go to BigQuery Console - <https://console.cloud.google.com/bigquery>
- Required Permissions for Linking Firebase to BigQuery - **Firebase Admin** or **Owner** role on the Firebase project for linking services.
- In Google Cloud Console, go to **IAM & Admin** → Search your user email. Confirm if you have both Firebase Admin and BigQuery Admin/Data Editor roles.
- Make sure **BigQuery API** is enabled in Google Cloud Console.
- Go to analytics.google.com and select the GA4 property connected to your app.
- Use Project ID or property ID in Firebase Analytics or GA4 to **SEARCH** in Bigquery after you integrate Bigquery in both of them. You will find Project ID in settings.
- They are like plugins that must be enabled in Integrations tab on firebase and Admin -> product Links -> Bigquery Links in GA4.
- Wait at least 24 hrs after events are fired from Frontend.


Select a resource

No organization ▼

Search projects and folders

Q |

RecentStarredAll

Name	Type	ID
✓  No organization ?	Organization	0

Explorer

+ Add data



Search BigQuery resources



Show starred only

- ▼ [redacted]t--prod ☆ ⋮
 - ▶ 📁 Repositories ⋮
 - ▶ 🔍 Queries ⋮
 - ▶ 📖 Notebooks ⋮
 - ▶ 📊 Data canvases ⋮
 - ▶ 🛠 Data preparations ⋮
 - ▶ 📡 Pipelines ⋮
 - ▶ 🔌 External connections ⋮
- ▼ 🏠 firebase_messaging ☆ ⋮
 - 🏠 data ☆ ⋮
- ▼ 🏠 firebase_performance ☆ ⋮
 - 🏠 com_[redacted]. ☆ ⋮
 - 🏠 com_[redacted]. ☆ ⋮

Note: Only the first 10 rows are shown in the screenshot. To see the full picture you must execute the query.

Total records

```
SELECT
  COUNT(*) AS `count`
FROM
  `project---prod.firebase_messaging.data`
```

236659 records

Data date range

```
SELECT
  MIN(EXTRACT(DATE FROM event_timestamp)) AS `min_date`,
  MAX(EXTRACT(DATE FROM event_timestamp)) AS `max_date`
FROM
  `project---prod.firebase_messaging.data`
```

Row	min_date	max_date
1	2025-03-28	2025-05-25

This is 2 months data starting from 28th March 2025 to 25th May 2025.


Topic frequency

```
SELECT
  (
    CASE
      WHEN REGEXP_EXTRACT(topic, r'^\w+[-]\d+[-](.*)') IS NULL THEN
        "NA"
      ELSE REGEXP_EXTRACT(topic, r'^\w+[-]\d+[-](.*)')
    END
  ) AS `topic`,
  COUNT(*) AS `count`,
  ROUND(
```

```

        (COUNT(1) * 100) / (SELECT COUNT(*) AS `total` FROM
`project---prod.firebase_messaging.data`), 2
    ) AS `percent`
FROM
    `project---prod.firebase_messaging.data`
GROUP BY 1
ORDER BY 2 DESC

```

Row	topic	count	percent
1	 order	212113	89.63
2	NA	24546	10.37

10% of messages sent seem to be going no where.

Topic frequency by Store

```

SELECT
    topic,
    COUNT(*) AS `count`,
    ROUND(
        (COUNT(1) * 100) / (SELECT COUNT(*) AS `total` FROM
`project---prod.firebase_messaging.data`), 2
    ) AS `percent`
FROM
    `project---prod.firebase_messaging.data`
GROUP BY 1
ORDER BY 2 DESC

```

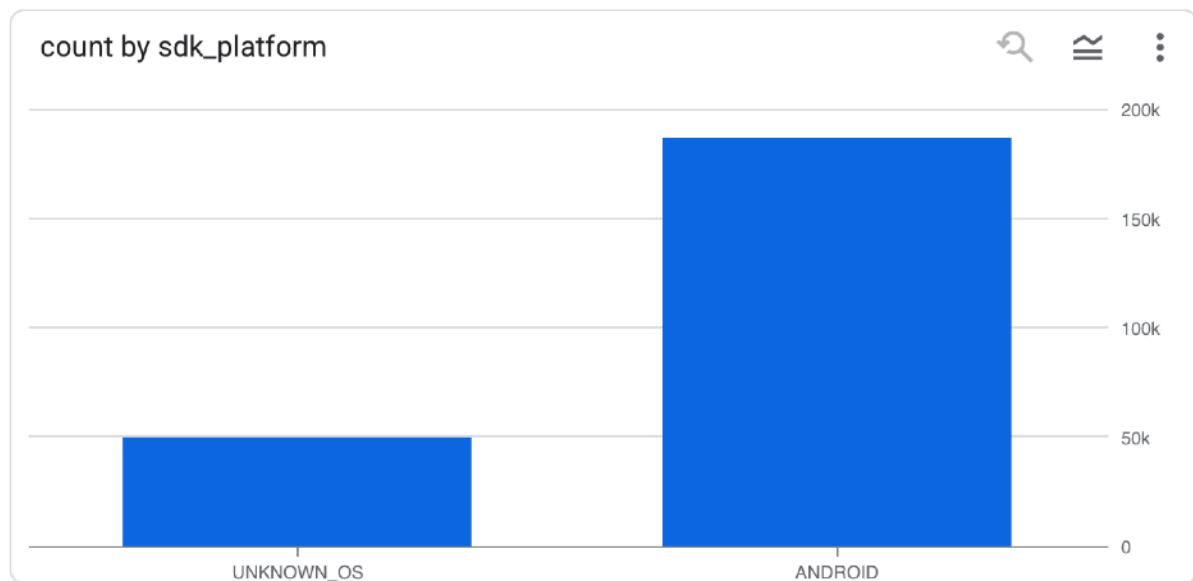
Row	topic ▼	count ▼	percent ▼
1		24546	10.37
2		1246	0.53
3		1147	0.48
4		1120	0.47
5		1010	0.43
6		928	0.39
7		901	0.38
8		893	0.38
9		872	0.37
10		858	0.36

Interestingly 10 percent of events were sent to no store. Firebase messaging is FREE is that is not much of an issue but I think we might incur minor BE cost based on the way we set it up.

SDK Platform

```
SELECT
  sdk_platform,
  COUNT(*) AS `count`,
  ROUND(
    (COUNT(1) * 100) / (SELECT COUNT(*) AS `total` FROM `project---prod.firebase_messaging.data`), 2
  ) AS `percent`
FROM
  `project---prod.firebase_messaging.data`
GROUP BY 1
```

Row	sdk_platform ▼	count ▼	percent ▼
1	UNKNOWN_OS	49693	21.0
2	ANDROID	186966	79.0

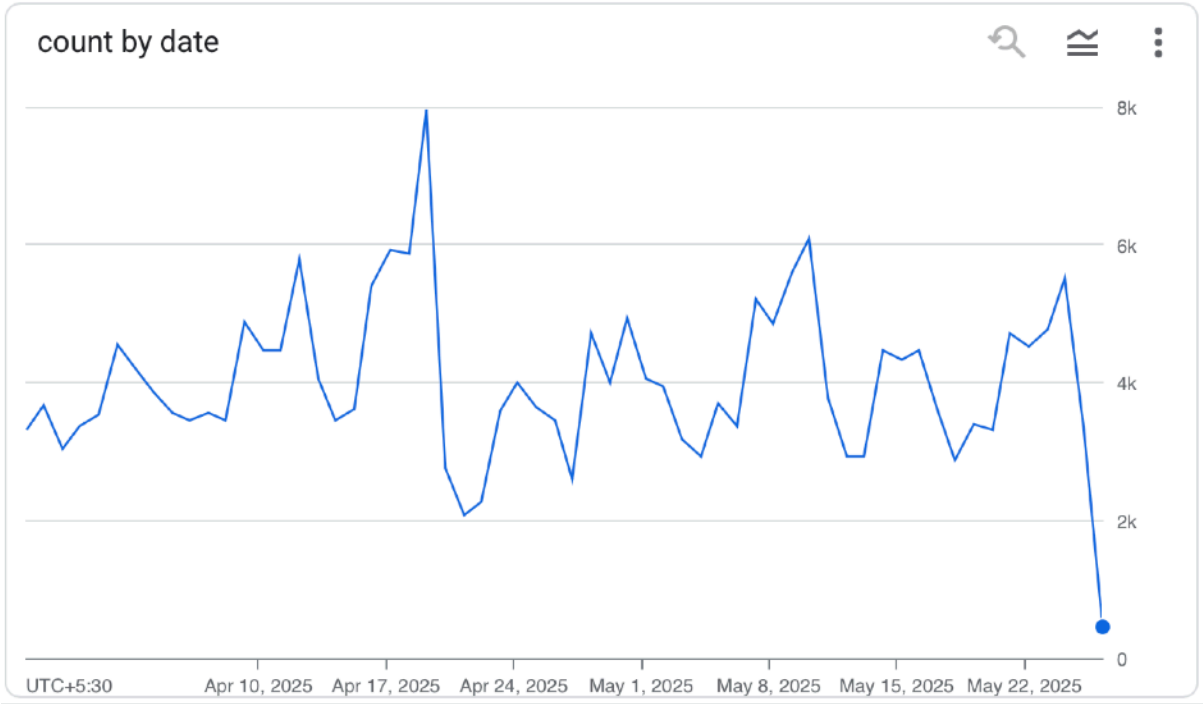


It looks like 21% of the messages are being sent or received by unknown OS.

Message frequency by day

```
SELECT
  EXTRACT(DATE FROM event_timestamp) AS `date`,
  COUNT(1) AS `count`
FROM
  `project---prod.firebase_messaging.data`
GROUP BY 1
ORDER BY 1
```


Row	date ▼	count ▼
1	2025-03-28	3323
2	2025-03-29	3668
3	2025-03-30	3045
4	2025-03-31	3380
5	2025-04-01	3528
6	2025-04-02	4559
7	2025-04-03	4220
8	2025-04-04	3863
9	2025-04-05	3559
10	2025-04-06	3448

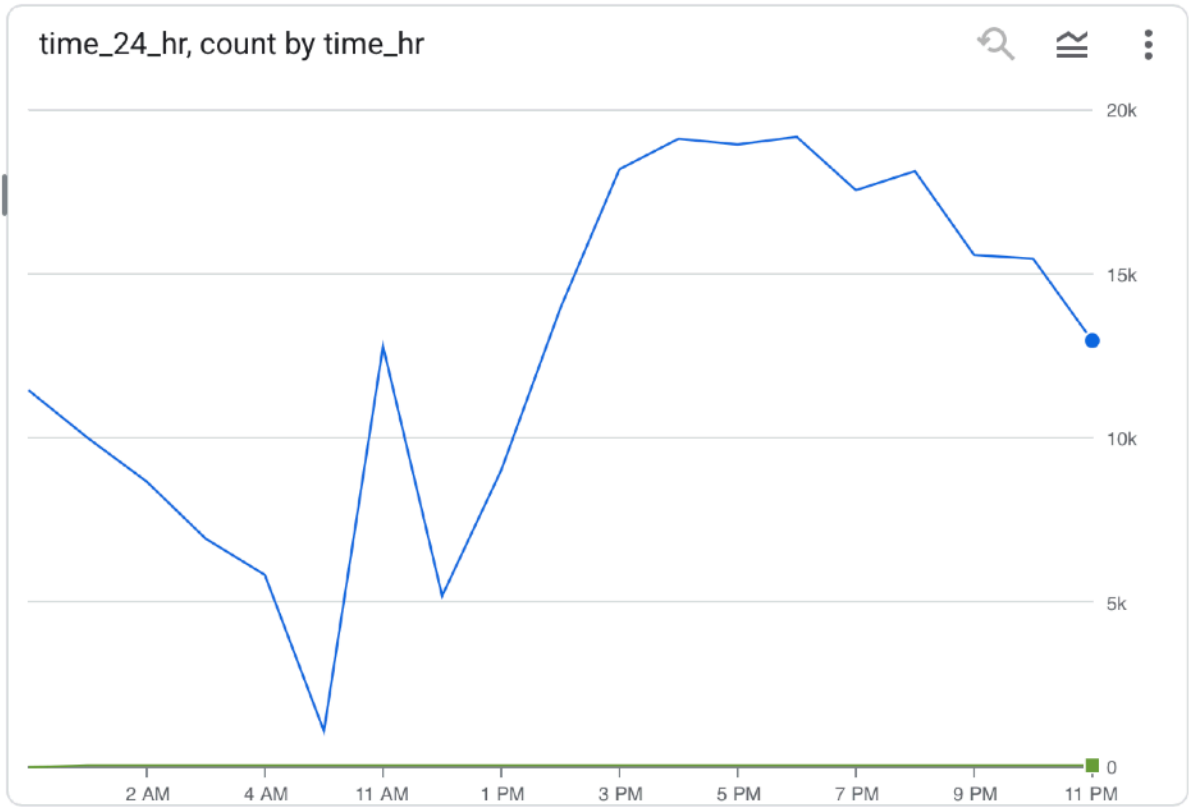


Peak days between 18th to 20th of April.

Message frequency by week

```
-- %l:%M %p
-- %I = zero-padded 12-hour (e.g., 07)
-- %l = non-padded 12-hour (e.g., 7)
-- %p = AM/PM
-- %M = minutes (always zero-padded)
SELECT
  FORMAT_DATETIME('%l %p', DATETIME(event_timestamp)) AS `time_hr`,
  EXTRACT(HOUR FROM event_timestamp) AS `time_24_hr`,
  COUNT(*) AS `count`
FROM
  `project---prod.firebaseio.messaging.data`
GROUP BY 1, 2
ORDER BY 2
```

Row	time_hr	time_24_hr	count
1	12 AM	0	11424
2	1 AM	1	10004
3	2 AM	2	8659
4	3 AM	3	6902
5	4 AM	4	5853
6	5 AM	5	1062
7	11 AM	11	12778
8	12 PM	12	5195
9	1 PM	13	9010
10	2 PM	14	13951

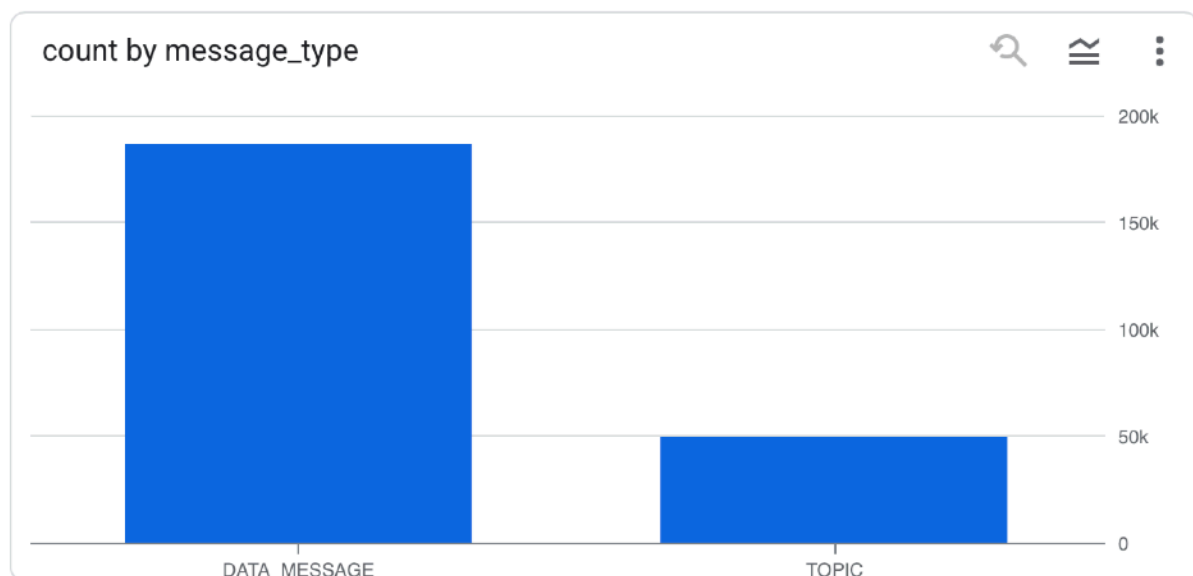


Peak timings are between 3PM to 8PM.

Message Type

```
SELECT
  message_type,
  COUNT(*) AS `count`,
  ROUND(
    (COUNT(1) * 100) / (SELECT COUNT(*) AS `total` FROM `project---prod.firebase_messaging.data`), 2
  ) AS `percent`
FROM
  `project---prod.firebase_messaging.data`
GROUP BY 1
ORDER BY 2 DESC
```

Row	message_type	count	percent
1	DATA_MESSAGE	186966	79.0
2	TOPIC	49693	21.0



Priority of messages

```
SELECT
  priority,
  ROUND(
    (COUNT(1) * 100) / (SELECT COUNT(*) AS `total` FROM
`project---prod.firebaseio.data`), 2
  ) AS `percent`
FROM
  `project---prod.firebaseio.data`
GROUP BY 1
```

Row	priority	percent
1	5	100.0

In Firebase Cloud Messaging (FCM) for Android, **message priorities** determine how your messages are handled by the system, particularly when the device is in **Doze mode** or **App Standby**.

Priority Levels in FCM

FCM supports **two levels of priority** for downstream (device-targeted) messages:

1. High Priority (value: "high")

- **Use case:** Time-sensitive messages (e.g., chat messages, urgent alerts).
- **Behavior:**
- Wake the device if it is asleep.
- Bypass battery optimizations like Doze mode and App Standby (subject to Android limits).

- Delivers messages immediately, if possible.

2. Normal Priority (value: "normal")

- **Use case:** Non-urgent background data sync or UI updates.
- **Behavior:**
- Delivers messages based on device conditions (e.g., only when the device is active).
- Does **not** wake the device.
- Messages may be delayed in Doze mode.

What About "Priority 5"?

There is **no official "priority 5"** in FCM. However:

- Internally, **FCM may represent priorities with integer values**, where:
- "high" priority might be mapped to 10
- "normal" priority might be mapped to 5

So, if you're seeing **priority 5** in logs or internal data, it likely refers to **normal priority** messages.

Summary

Priority (string)	Internal Value	Use Case	Behavior
"high"	Likely 10	Urgent alerts	Wakes device, bypasses Doze (mostly)
"normal"	Likely 5	Background sync	May delay in Doze mode

If you need a message to be delivered immediately (like chat notifications), set the priority to "high". If you're seeing priority 5, it likely means the message was sent with "normal" priority.

This might explain why there seems to be complaints about not receiving notifications.

Device recently active

```
SELECT
  device_recently_active,
  COUNT(*) AS `count`,
  ROUND(
    (COUNT(1) * 100) / (SELECT COUNT(*) AS `total` FROM
`project---prod.firebase_messaging.data`), 2
  ) AS `percent`
FROM
  `project---prod.firebase_messaging.data`
GROUP BY 1
ORDER BY 2 DESC
```

Row	device_recently_active ▼	count ▼	percent ▼
1	false	236659	100.0

I am not sure what this means but it looks like the messages were sent when the devices were inactive or the store associates are not using at that time.

Active Stores

In the past 2 months only these stores were active as per the messaging data.

```
WITH
  `cte1` AS (
    SELECT
      CAST(REGEXP_EXTRACT(topic, r'^\w+[-](\d+).') AS INT) AS
    `store`
    FROM
      `project---prod.firebase_messaging.data`
    GROUP BY 1
    ORDER BY 1
  )
SELECT
  STRING_AGG(CAST(store AS STRING), ', ') AS `stores`
FROM
  `cte1`
```

1, 2, 3, 4, 6, 7, 9, 10, 11, ...

Total of 1162 stores.