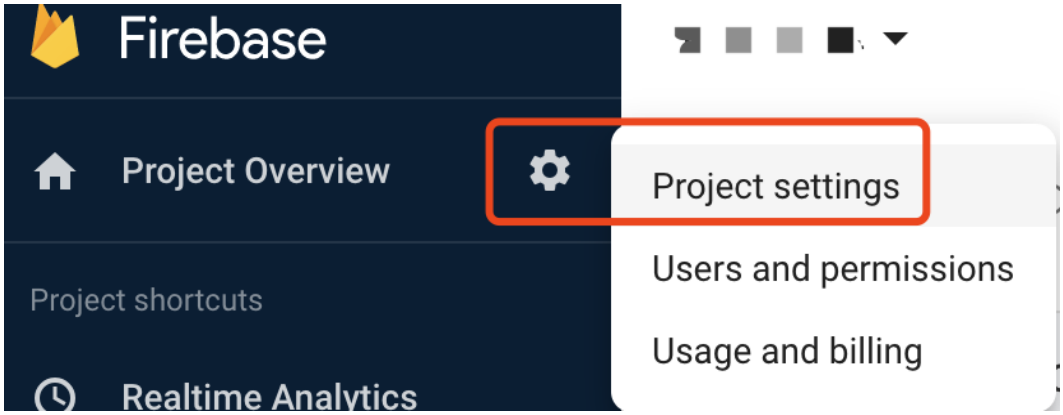
Firebase & Bigquery

Quick Start Manual

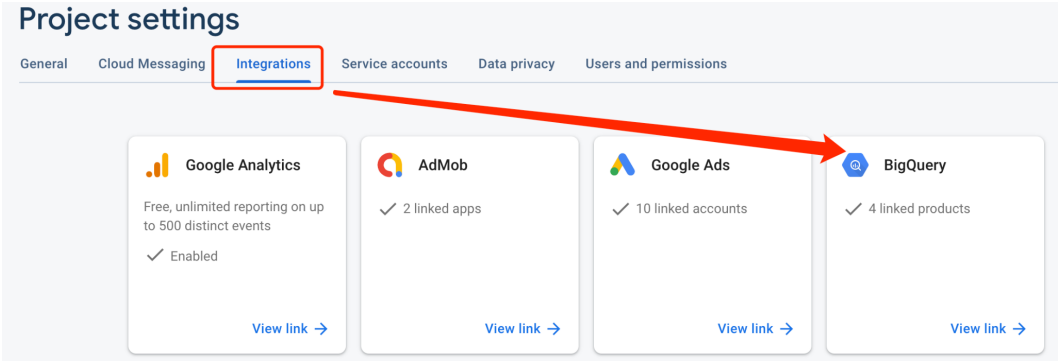
2023.04.25

Turn on the data integration

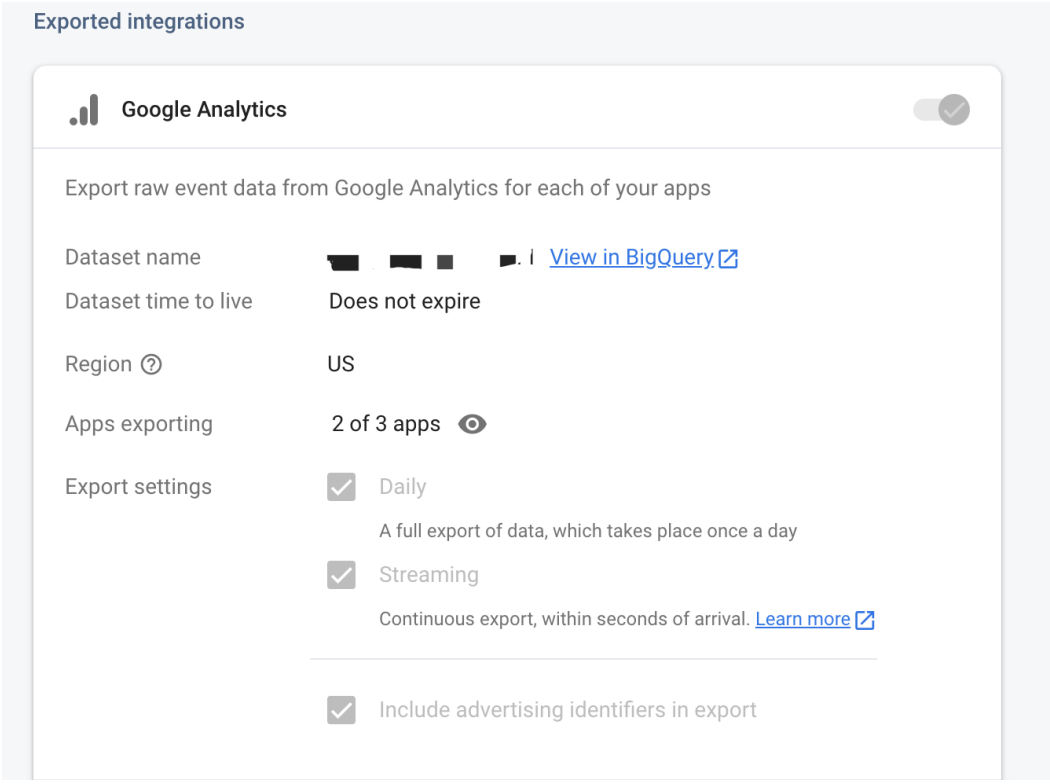
1. Go to the Firebase console.
2. Click on the **Project settings** icon (⚙️).



3. Click on the **Integrations** tab.



4. Click on the **BigQuery** card.
5. Click on the **Link** button.

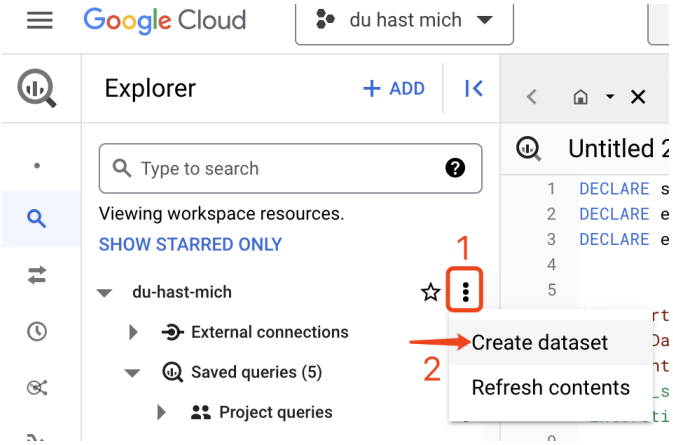


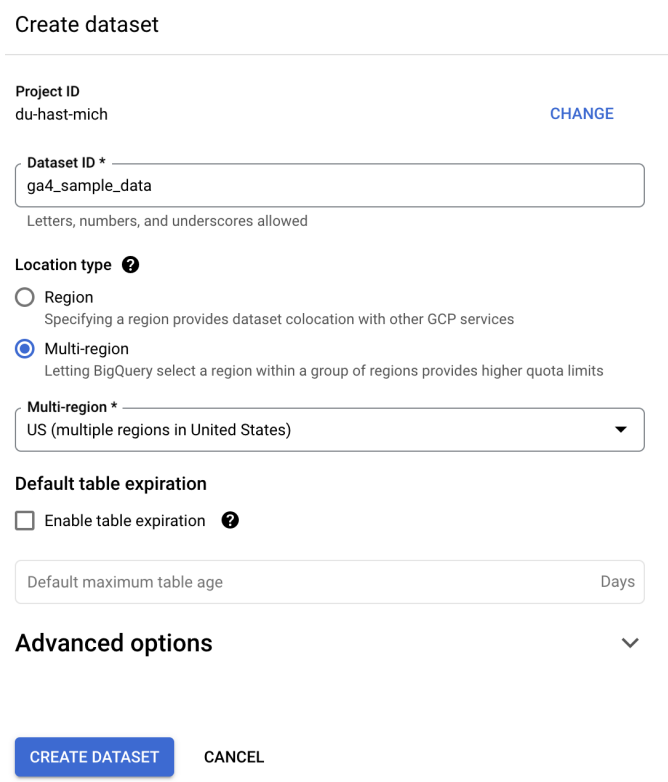
Analyze the data in Bigquery

Before we start, the full SQL code repository can be found here. Please note, scripts are querying a public dataset. You will need to change the targeting *table/dataset* to see your own data.

(Optional) Create a view pointing to sample/public dataset

1. Go to Bigquery console
2. Create a dataset, e.g. `ga4_sample_data`

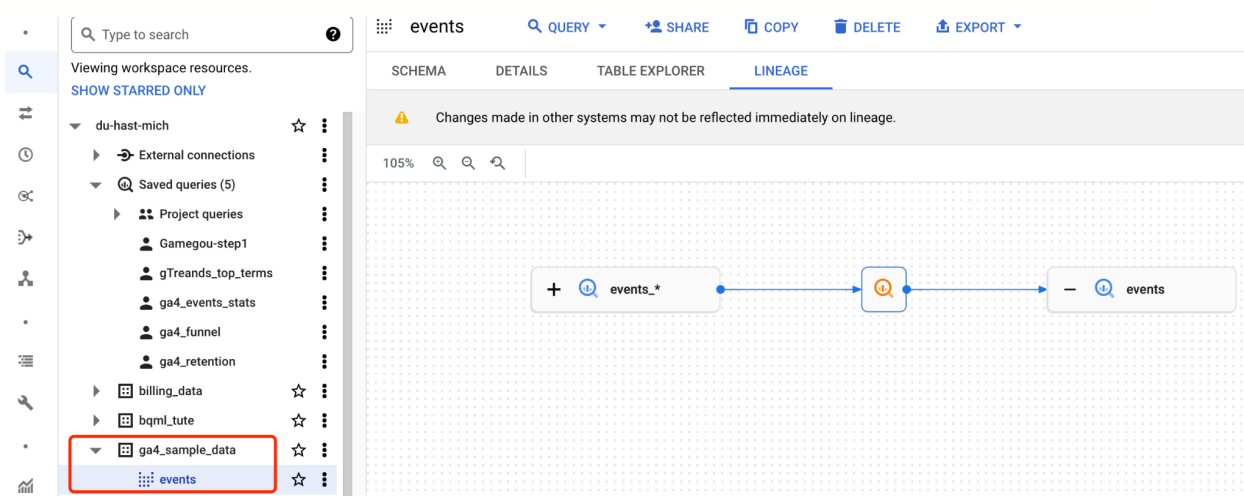




3. Create a View pointing to the sample/public dataset
   Run the following SQL in BQ workbench

```
CREATE VIEW ga4_sample_data.events AS
select * from `bigquery-public-data.ga4_obfuscated_sample_ecommerce.events_*`
```

Once done, you should be able to see the following view has been created



Explore user retention

```
WITH analytics_data AS (
 SELECT user_pseudo_id, event_timestamp, event_name,
   UNIX_MICROS(TIMESTAMP("2020-11-01 00:00:00", "+8:00")) AS start_day,
   3600*1000*1000*24*7 AS one_week_micros
 FROM `bigquery-public-data.ga4_obfuscated_sample_ecommerce.events_*`
)

SELECT week_0_cohort / week_0_cohort AS week_0_pct,
week_1_cohort / week_0_cohort AS week_1_pct,
week_2_cohort / week_0_cohort AS week_2_pct,
week_3_cohort / week_0_cohort AS week_3_pct
FROM (
 WITH week_3_users AS (
   SELECT DISTINCT user_pseudo_id
   FROM analytics_data
   WHERE event_timestamp BETWEEN start_day+(3*one_week_micros) AND
start_day+(4*one_week_micros)
 ),
 week_2_users AS (
   SELECT DISTINCT user_pseudo_id
   FROM analytics_data
   WHERE event_timestamp BETWEEN start_day+(2*one_week_micros) AND
start_day+(3*one_week_micros)
 ),
 week_1_users AS (
   SELECT DISTINCT user_pseudo_id
   FROM analytics_data
   WHERE event_timestamp BETWEEN start_day+(1*one_week_micros) AND
start_day+(2*one_week_micros)
 ),
 week_0_users AS (
   SELECT DISTINCT user_pseudo_id
   FROM analytics_data
   WHERE event_name = 'first_visit'
```

```
    AND event_timestamp BETWEEN start_day AND start_day+(1*one_week_micros)
 )
 SELECT
   (SELECT count(*)
    FROM week_0_users) AS week_0_cohort,
   (SELECT count(*)
    FROM week_1_users
    JOIN week_0_users USING (user_pseudo_id)) AS week_1_cohort,
   (SELECT count(*)
    FROM week_2_users
    JOIN week_0_users USING (user_pseudo_id)) AS week_2_cohort,
   (SELECT count(*)
    FROM week_3_users
    JOIN week_0_users USING (user_pseudo_id)) AS week_3_cohort
)
```

TODOs:

1. Try to build a 7-day retention query
2. Explore your own data against different app versions and/or specific devices

Define your own closed funnels

```
SELECT count(distinct funnel_1) as funnel_1_total, count(distinct funnel_2) as
funnel_2_total from (
 SELECT
   IF (event_name = "session_start", user_pseudo_id, NULL) AS funnel_1,
   IF (event_name = "session_start" AND next_event = "purchase" AND next_timestamp -
event_timestamp < 20 * 60 * 1000 * 1000, user_pseudo_id, NULL) AS funnel_2
 FROM (
   SELECT event_name, user_pseudo_id , event_timestamp,
   LEAD(event_name, 1) OVER (PARTITION BY user_pseudo_id ORDER BY event_timestamp) AS
next_event,
   LEAD(event_timestamp, 1) OVER (PARTITION BY user_pseudo_id ORDER BY
event_timestamp) AS next_timestamp
   FROM `bigquery-public-data.ga4_obfuscated_sample_ecommerce.events_*`
   WHERE (event_name = "session_start" OR event_name = "purchase")
```

```
  ORDER BY 2,3
 )
)
```

TODOs:

1. Build a more complex funnel, e.g. more steps
2. Compare the same funnel against different criterias, e.g. geo-location, devices etc.

Visualized the data in Looker

Transform data in BigQuery

The reason for the conversion is that we need to convert the data exported from firebase into the schema needed for the looker ml model, either as a table or as a view, here is the example:

1. Create a sql file called "create_events_view.sql" and write the following sql (please remember to change the project id and dataset)

```
CREATE VIEW {project_id}.{dataset}.v_gaming_events AS
SELECT
GENERATE_UUID() as unique_event_id,
TIMESTAMP_MICROS(event_timestamp) as event,
event_name,
event_bundle_sequence_id,
safe_cast(user_pseudo_id as STRING) as user_id,
TIMESTAMP_MICROS(user_first_touch_timestamp) as user_first_seen,
platform as device_platform,
device.mobile_brand_name as device_brand,
device.mobile_model_name as device_model,
device.operating_system_version as device_os_version,
device.language as device_language,
geo.continent as continent,
geo.region as region,
geo.country as country,
app_info.install_source as install_source,
(event_timestamp=user_first_touch_timestamp) as is_first_seen,
ecommerce.purchase_revenue as iap_revenue,
user_ltv,
( select
```

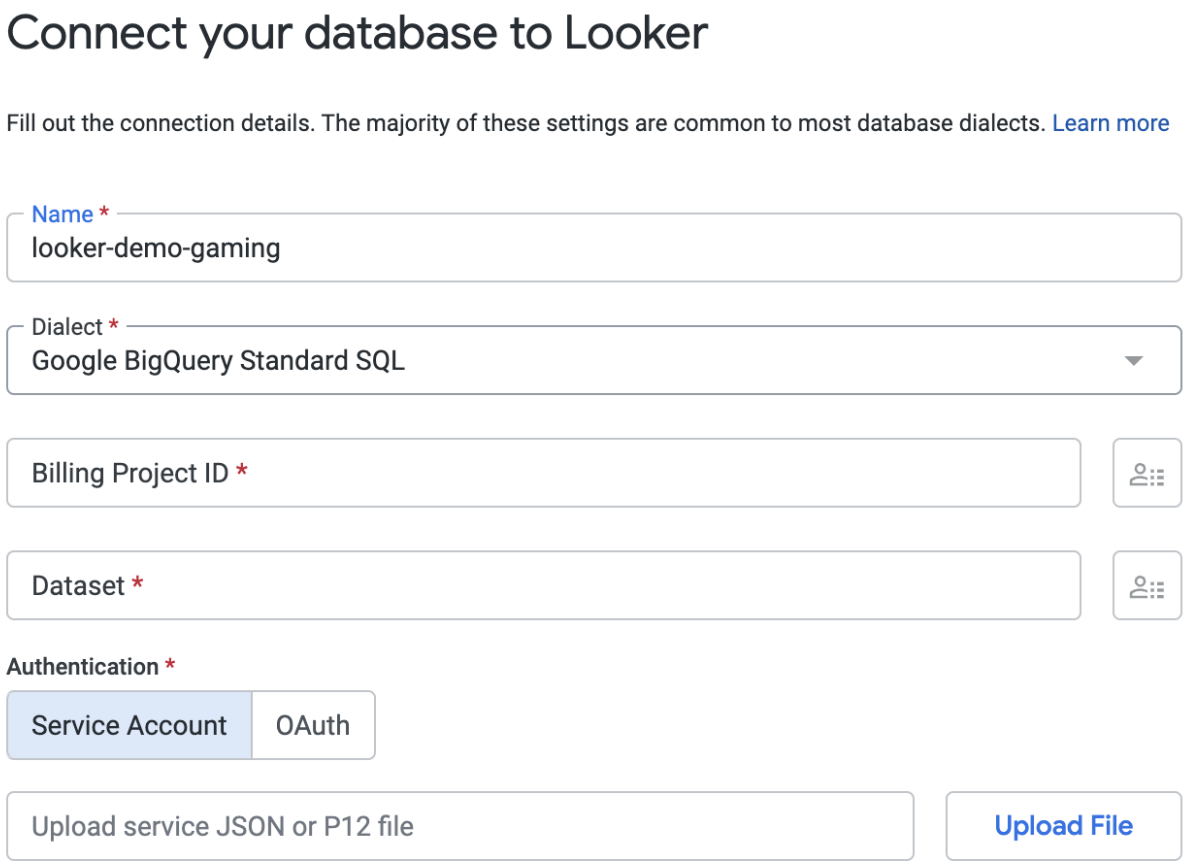
```
   x
 from UNNEST(ARRAY<STRUCT<x INT64, y STRING>>[(event_timestamp, event_name)])
 where y='session_start') as session_start,
(CASE
  WHEN event_name!='session_start' THEN null
  ELSE TIMESTAMP_MICROS(LEAD(event_timestamp, 1) OVER (
          PARTITION BY (select z
                        from UNNEST(ARRAY<STRUCT<x INT64, y STRING, z
STRING>>[(event_timestamp, event_name, user_pseudo_id)])
                        where y='session_start')  ORDER BY event_timestamp)
        )
  END
) as next_session_start,
( select value.float_value
 from unnest(event_params)
 where key='@ga_ad_revenue' limit 1) as ad_revenue,
( select value.float_value
 from unnest(event_params)
 where key='@ga_install_cost' limit 1) as install_cost,
( select value.int_value
 from unnest(event_params)
 where key='@ga_gems_earned' limit 1) as gems_earned,
( select value.string_value
 from unnest(event_params)
 where key='@ga_campaign_name' limit 1) as campaign_name,
( select value.string_value
 from unnest(event_params)
 where key='@ga_ad_network' limit 1) as ad_network,
( select value.string_value
 from unnest(event_params)
 where key='@ga_campaign_id' limit 1) as campaign_id,
( select value.string_value
 from unnest(event_params)
 where key='@ga_game_name' limit 1) as game_name,
( select value.string_value
 from unnest(event_params)
```

```
 where key='@ga_game_version' limit 1) as game_version,
( select value.int_value
 from unnest(event_params)
 where key='@ga_player_level' limit 1) as player_level,
( select value.int_value
 from unnest(event_params)
 where key='@ga_session_number' limit 1) as player_session_sequence,
( select safe_cast(value.int_value as STRING)
 from unnest(event_params)
 where key='@ga_session_id' limit 1) as unique_session_id
FROM
`bigquery-public-data.ga4_obfuscated_sample_ecommerce.events_*` as e
```

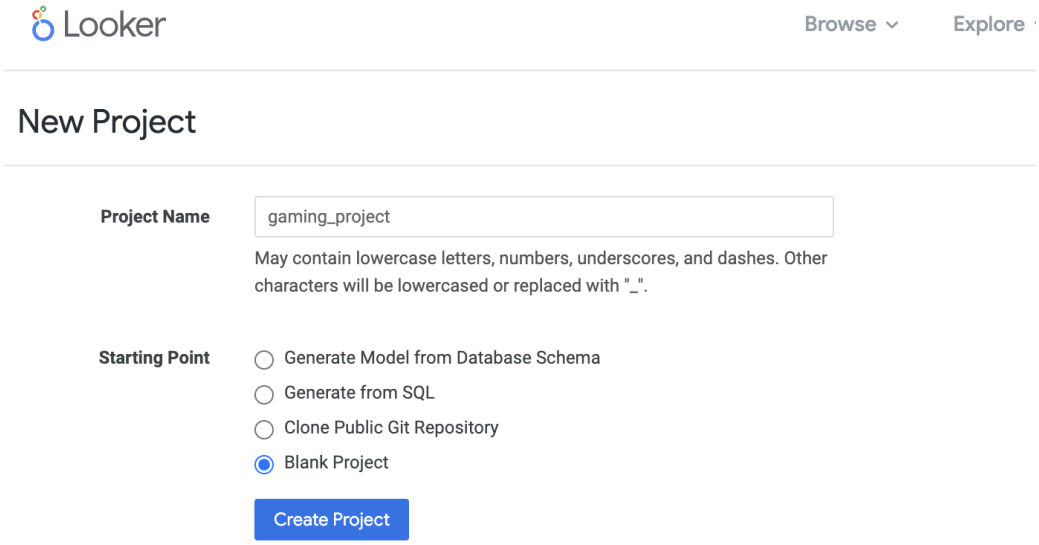
2. Create a view point for Looker
   Run the following command in console:
   ```
   bq query --use_legacy_sql=false \
    --parameter=ga_ad_revenue::ad_revenue \
    --parameter=ga_install_cost::install_cost \
    --parameter=ga_gems_earned::gems_earned \
    --parameter=ga_campaign_name::campaign_name \
    --parameter=ga_campaign_id::campaign_id \
    --parameter=ga_ad_network::link_url \
    --parameter=ga_game_name::game_name \
    --parameter=ga_game_version::game_version \
    --parameter=ga_player_level::player_level \
    --parameter=ga_session_id::ga_session_id \
    --parameter=ga_session_number::ga_session_number < create_events_view.sql
   ```
   In the SQL above, we'll need to adjust the following fields to match the actual event parameters being passed
3. (Optional) If your data is stored in multiple locations, you can change the way some fields are ingested, for example, querying the field “install cost” by multiple tables using a join table.

Create new project in Looker

1. Create a new connection to connect your dataset in BigQuery.



2. Create a new project, enter a name for it and select Blank Project.

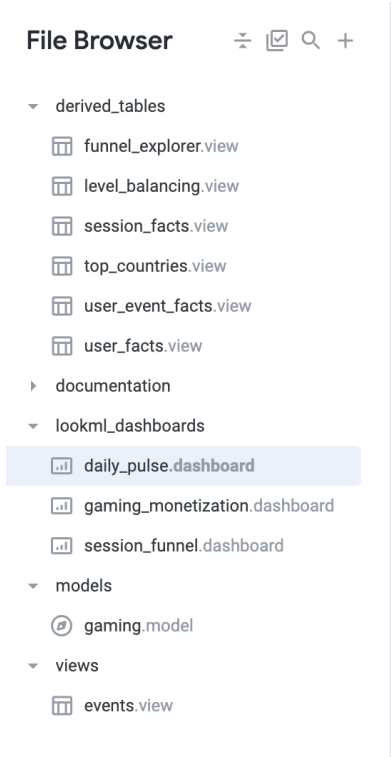


Import model file

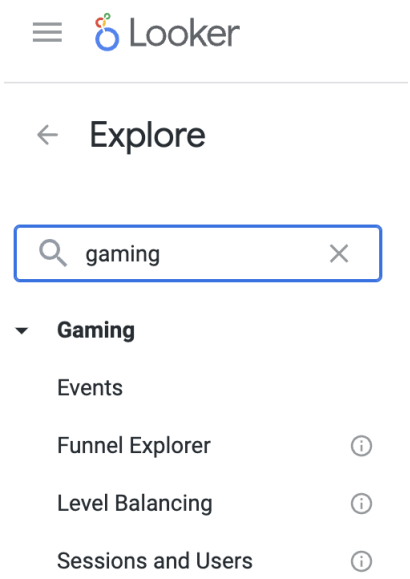
Import the following files in the folders as shown below:

- models:
  gaming.model (need to change the “connection”)
- views:
  events.view (need to change the “sql_table_name”)
- derived_tables:
  funnel_explorer.view
  level_balancing.view
  session_facts.view
  top_countries.view
  user_event_facts.view
  user_facts.view
- lookml_dashboards:
  daily_pulse.dashboard
  gaming_monetization.dashboard
  session_funnel.dashboard

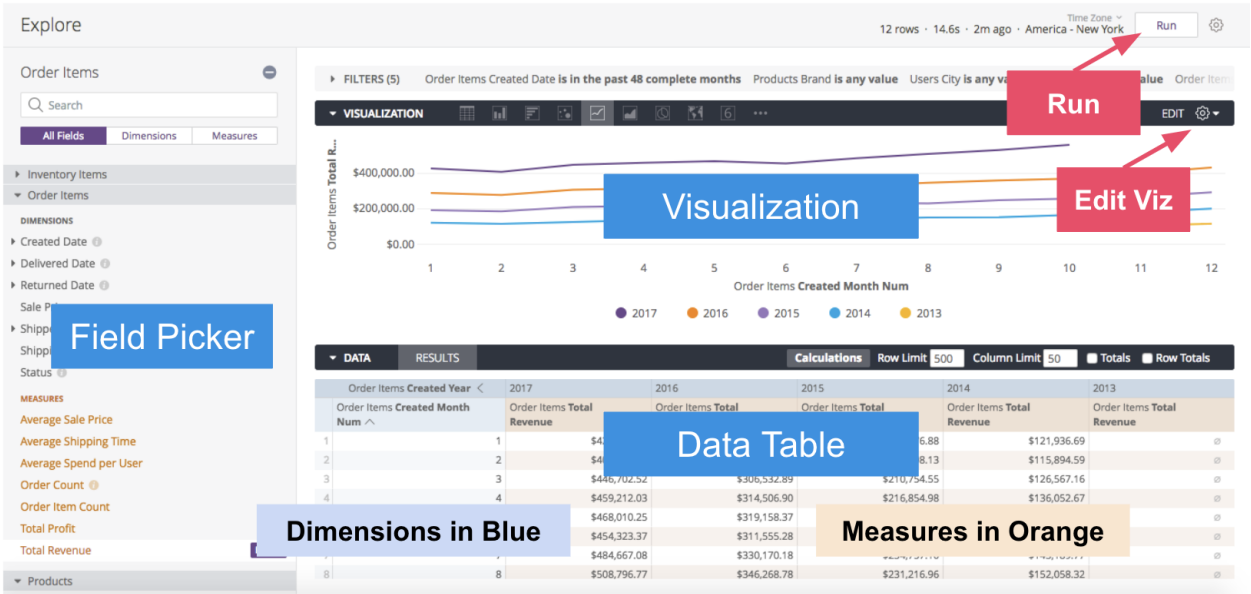
The full code repository can be found here:
https://gist.github.com/ping-coder/3ec7277d0c2714c8cf034b851755d7ea



Explore data

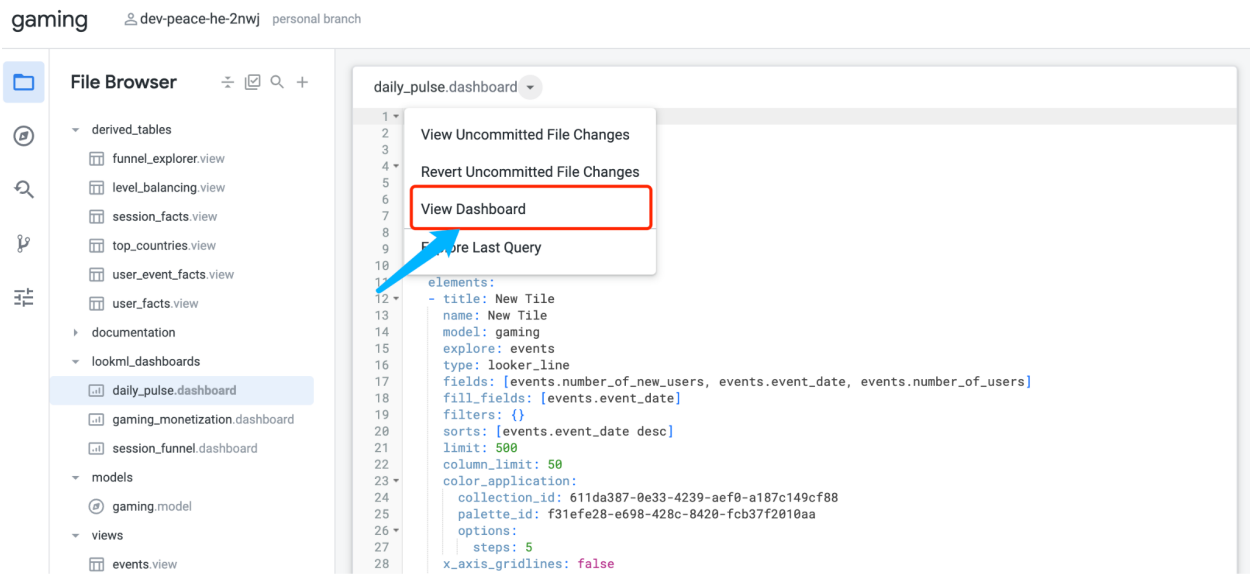


1. Go to Looker Console
2. Click on Explore menu
3. Go to Gaming Events Explore

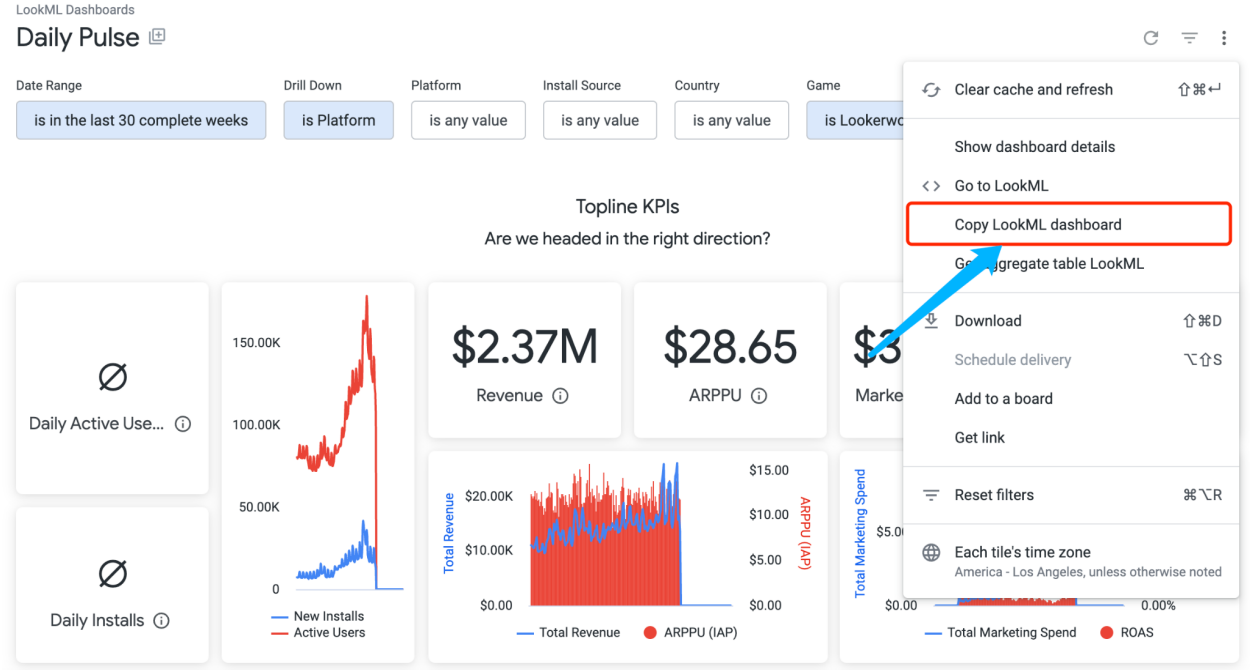


Create and edit dashboard

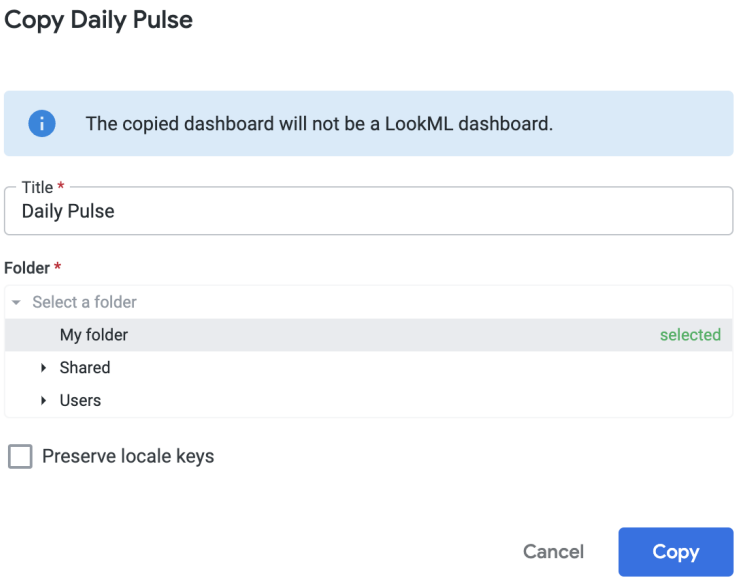
1. Goto the console of LookML, select daily_pulse.dashboard and click on “View Dashboard”



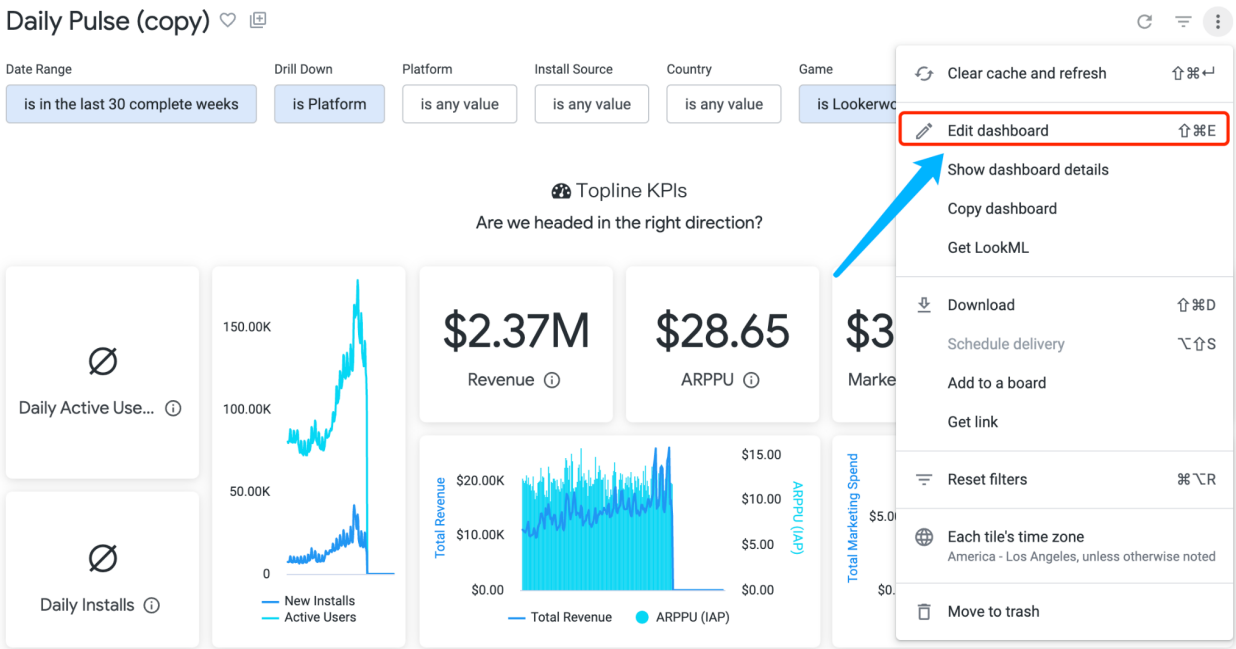
2. Goto the dashboard of Daily Pulse, and click on “Copy lookML dashboard”



3. Type the title, and click on “Copy”



4. Goto My folder, select the dashboard named “Daily Pulse”, and click on “Edit dashboard”



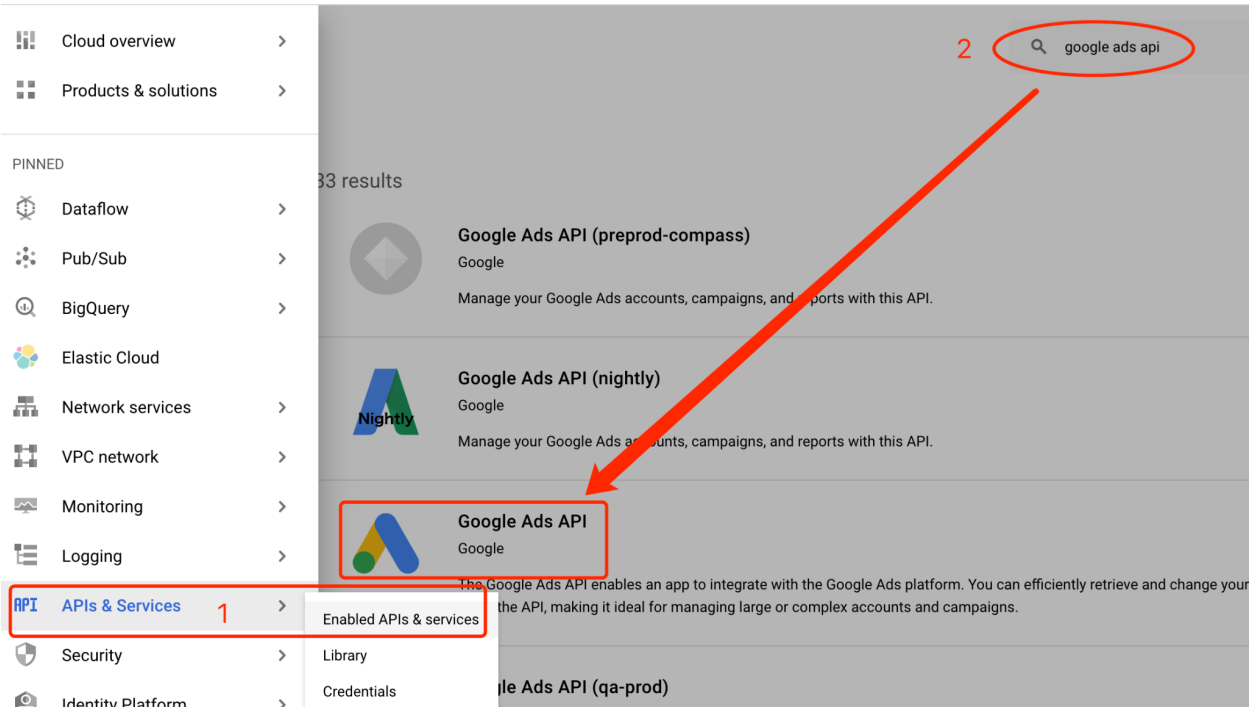
Finally, the dashboard tests are complete, and we need to finish configuring the git in the lookml page, and commit and publish to production.

Turn on Ads related APIs from Cloud Console

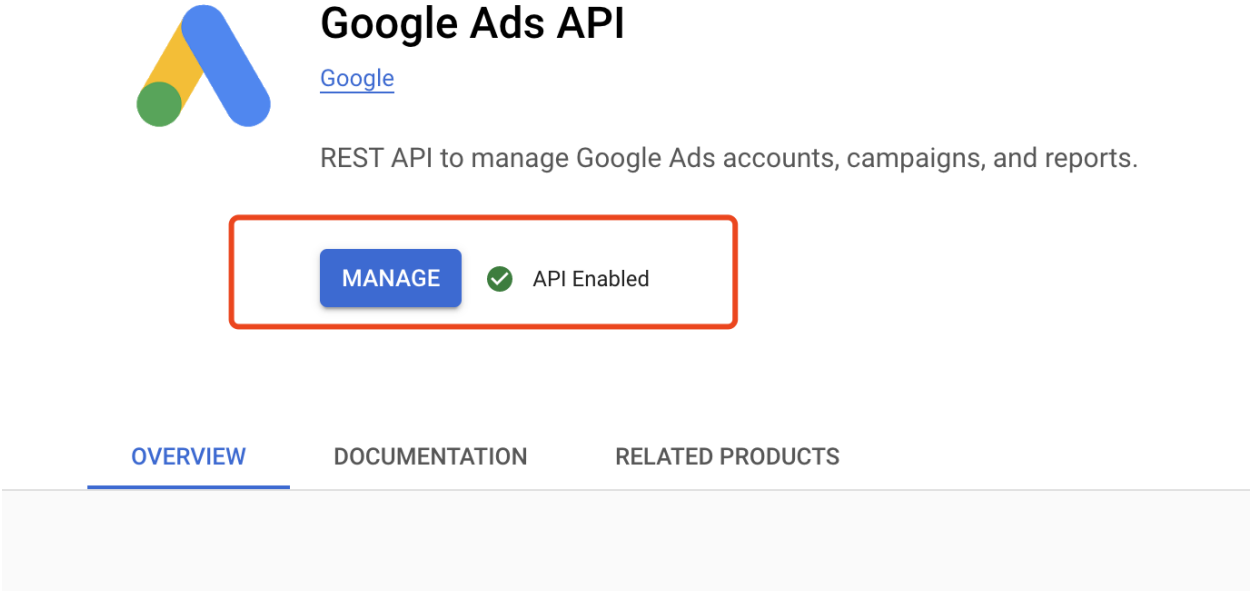
Google Ads API

1. Go to [cloud console]
2. From the left navigation menu select API & Services

3. Search “Google Ads API” from the mid-top text box

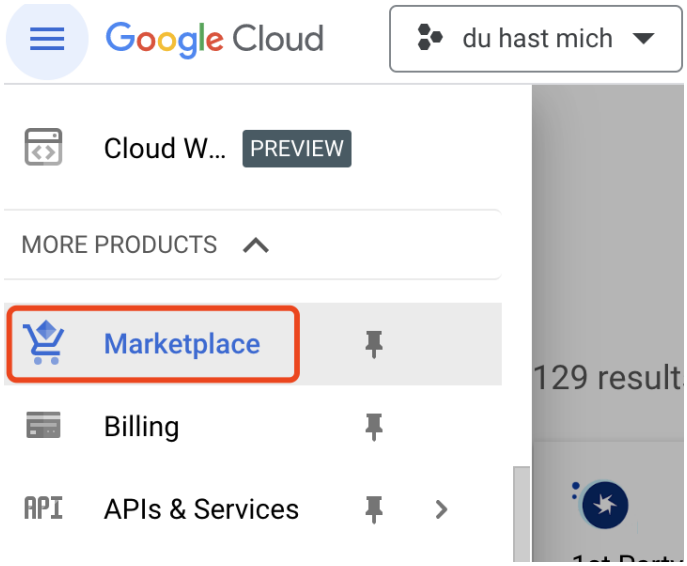


4. Click the “Enable” button to turn on the API

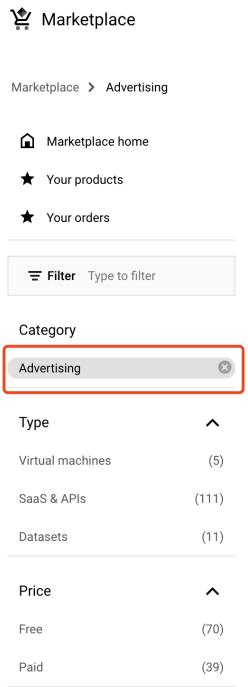


Full list of APIs

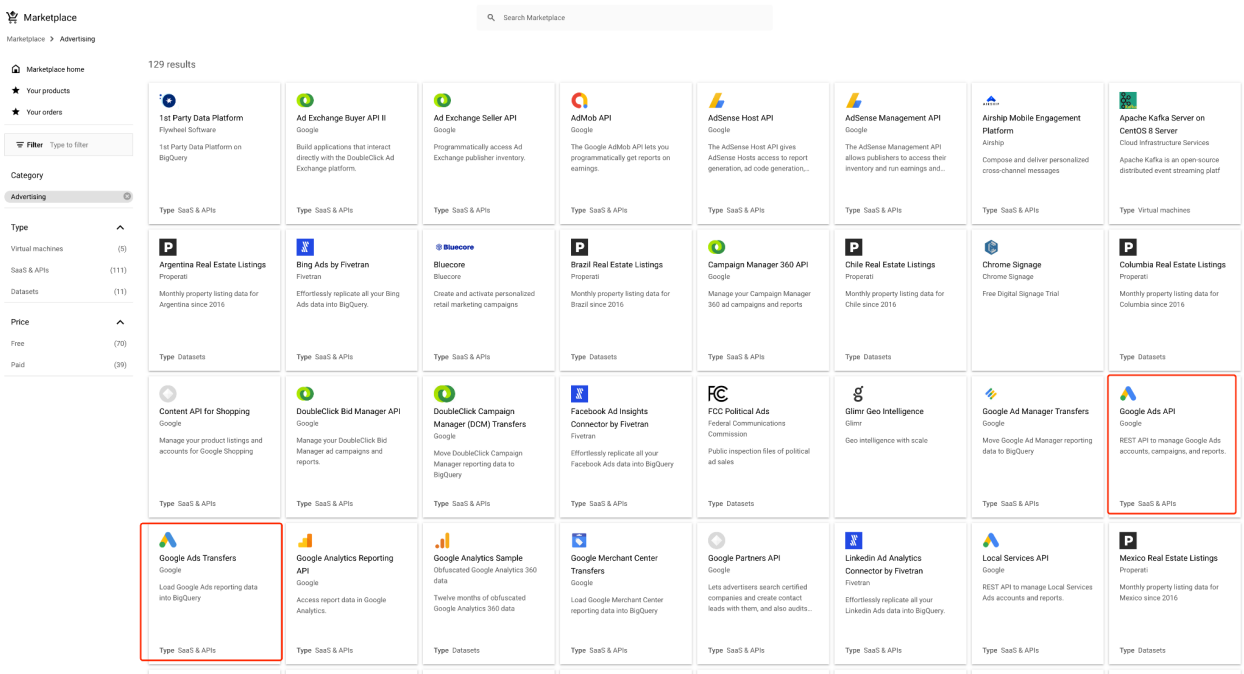
1. From the navigation menu select “Marketplace”



2. Set the filter to “Advertising”



3. A full list should be showing up



Appendix

[Link Google Ads to Firebase]
[Service Account for Accessing Cloud Services/APIs]