BigQuery SQL集

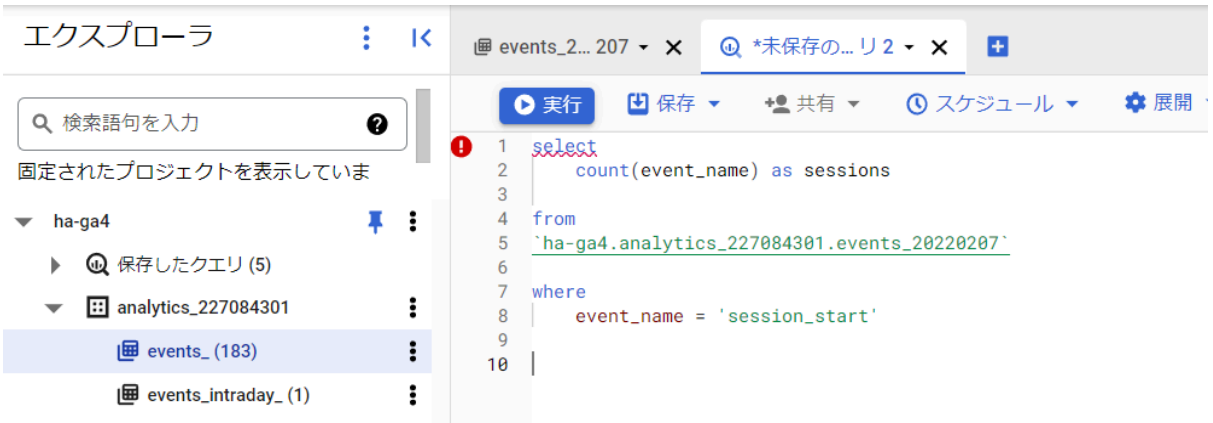
最初に

■本ドキュメントはSQLを覚えることを目的にしたドキュメントではなく、SQLを使ってGA4のBigQueryからよく使いそうなSQLをまとめたものになります。対象は今までSQLを使ってデータをだした事が無い方、昔一度試して挫折あるいはどうすればよいかわからなくなってしまった方が対象です。そのため「とりあえず使ってみる」事を意識しております。必要な限りコメントを入れて、わかりやすさ優先の記述となっています。そのため必ずしも効率的では無いことをご了承下さい。

利用にあたって、まずは以下2つを確認しておいてください。

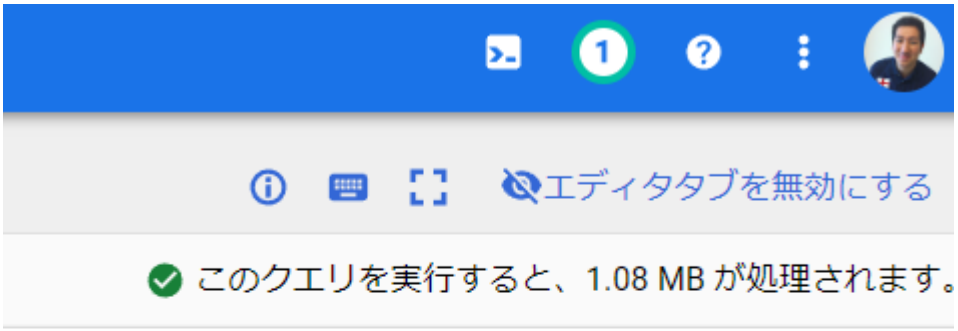
1）**from**でデータの範囲を指定する部分は、自分のデータセットを指定する

例えば以下のようにデータが格納されている場合



fromの部分は以下の通りとなります。
`ha-ga4.analytics_227084301.events_20220207`
events_以降は日付になります。全期間選びたい場合はevents_* と指定してください

2）クエリの実行画面で、実行前に処理されるサイズが表示されます。月間**1TB**まで無料で使えます。

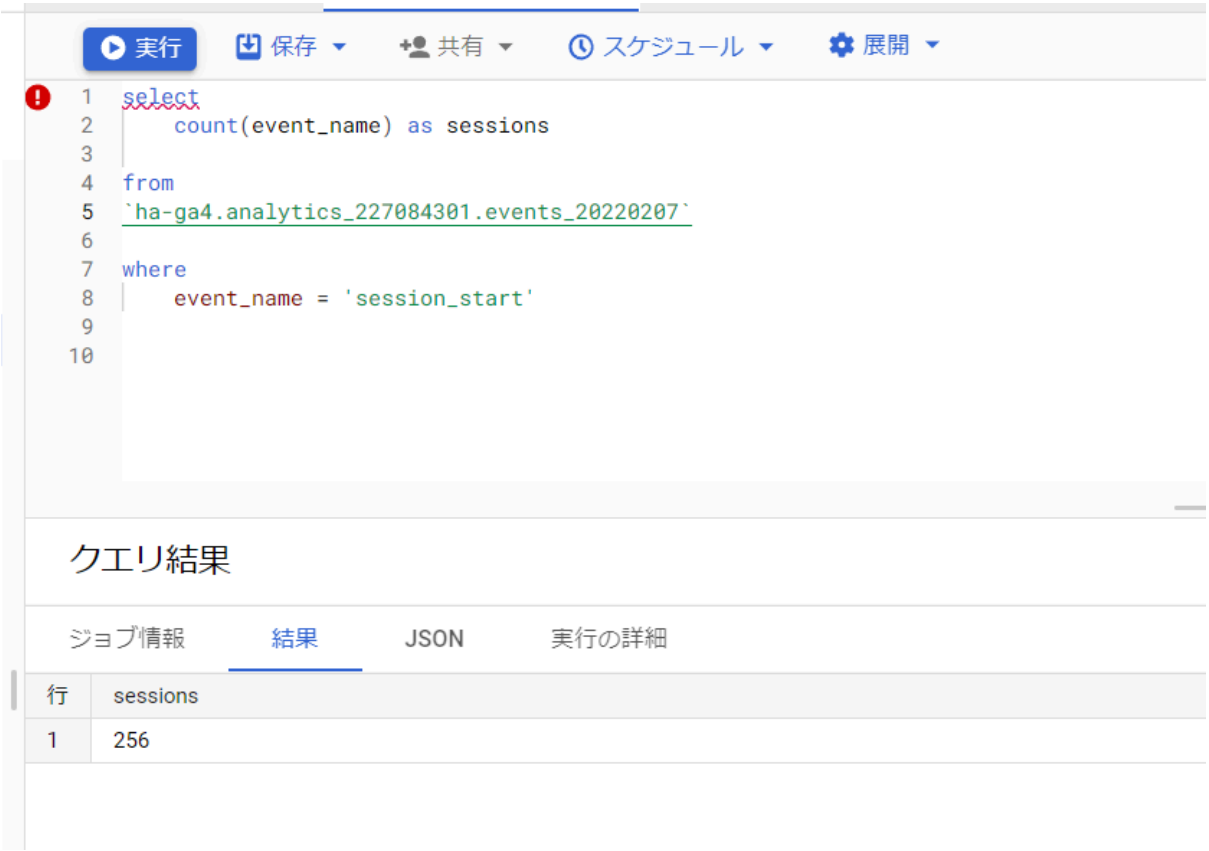


極端に大きなデータ量にならないように、項目や日付を絞ってBigQueryは利用しましょう。

1：特定のイベントをカウント

```
select
    count(event_name) as sessions  -- 後ほど指定するイベント名の列の見出しを「sessions」にする
from
`ha-ga4.analytics_227084301.events_20220207`      -- データの選択範囲。ここでは2022年2月7日のみを指定
where
    event_name = 'session_start'   -- イベント名がsession_startに合致するものだけを抽出
```

実行結果



2：日付範囲を指定し、日ごとの数値を出す

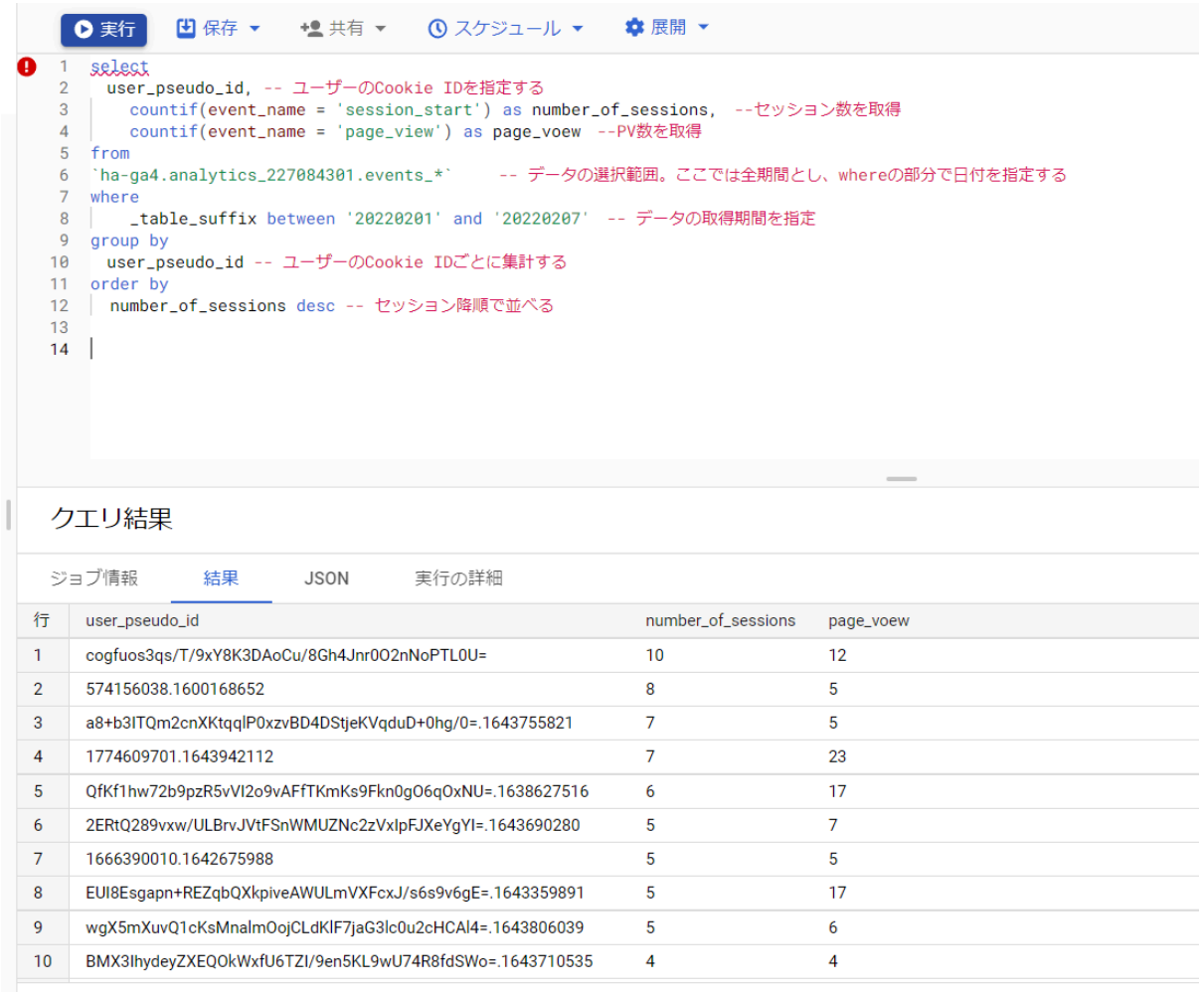
```
select
  date(timestamp_micros(event_timestamp),"Asia/Tokyo") as event_date, -- イベントの発生日付を選択
    count(event_name) as sessions  -- 後ほど指定するイベント名の列の見出しを「sessions」にする
from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
    event_name = 'session_start'   -- イベント名がsession_startに合致するものだけを抽出
    and _table_suffix between '20220201' and '20220207'  -- データの取得期間を指定
group by
  event_date   -- 日付ごとに集計する
order by
  event_date   -- 昇順で並び替える。降順で並び替えたい場合は  event_date desc と記載する
```

実行結果

3:ユーザーごとのセッション数やPVを降順で並べる

```
select
  user_pseudo_id, -- ユーザーのCookie IDを指定する
    countif(event_name = 'session_start') as number_of_sessions,  --セッション数を取得
    countif(event_name = 'page_view') as page_voew  --PV数を取得
from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
    _table_suffix between '20220201' and '20220207'  -- データの取得期間を指定
group by
  user_pseudo_id -- ユーザーのCookie IDごとに集計する
order by
  number_of_sessions desc -- セッション降順で並べる
```

実行結果



行	user_pseudo_id	number_of_sessions	page_voew
1	cogfuos3qs/T/9xY8K3DAoCu/8Gh4Jnr0O2nNoPTL0U=	10	12
2	574156038.1600168652	8	5
3	a8+b3ITQm2cnXKtqqlP0xzvBD4DStjeKVquD+0hg/0=.1643755821	7	5
4	1774609701.1643942112	7	23
5	QfKf1hw72b9pzR5vVI2o9vAFfTKmKs9Fkn0gO6qOxNU=.1638627516	6	17
6	2ERtQ289vxw/ULBrvJVtFSnWMUZNc2zVxlpFJXeYgYI=.1643690280	5	7
7	1666390010.1642675988	5	5
8	EUI8Esgapn+REZqbQXkpiveAWULmVXFcxJ/s6s9v6gE=.1643359891	5	17
9	wgX5mXuvQ1cKsMnalmOojCLdKlF7jaG3lc0u2cHCAl4=.1643806039	5	6
10	BMX3IhydeyZXEQOkWxfU6TZI/9en5KL9wU74R8fdSWo=.1643710535	4	4

4：日ごとのユーザー数をカウント

```
select
  date(timestamp_micros(event_timestamp),"Asia/Tokyo") as event_date, -- イベントの発生日付を選択
count(distinct user_pseudo_id) as users   -- CookieIDのユニークな数をカウントする
from
`ha-ga4.analytics_227084301.events_*`     -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20220201' and '20220205'   -- 日付の指定
group by
event_date
```

実行結果

日ごとのユニークなユーザー数になりますが、期間内のユニークなユーザー数を見たい場合は、2行目・8行目・9行目を消しましょう

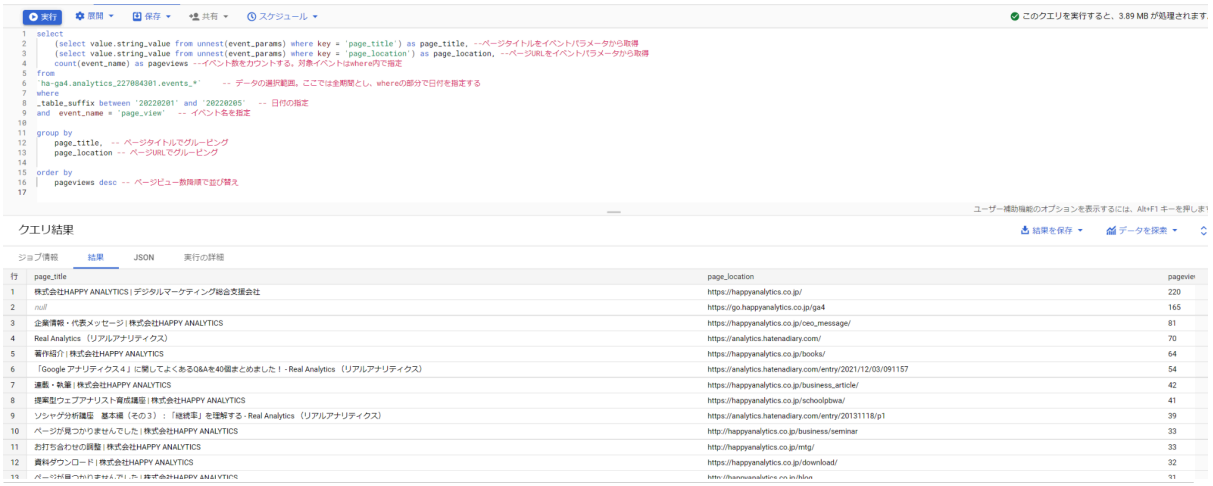
```
select
count(distinct user_pseudo_id) as users   -- CookieIDのユニークな数をカウントする
from
`ha-ga4.analytics_227084301.events_*`     -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20220201' and '20220205'   -- 日付の指定
```

5：ページごとのPV数を取得

```
select
    (select value.string_value from unnest(event_params) where key = 'page_title') as page_title, --ページタイトルをイベントパラメータから取得
    (select value.string_value from unnest(event_params) where key = 'page_location') as page_location, --ページURLをイベントパラメータから取得
    count(event_name) as pageviews --イベント数をカウントする。対象イベントはwhere内で指定
from
`ha-ga4.analytics_227084301.events_*`     -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20220201' and '20220205'   -- 日付の指定
and  event_name = 'page_view'   -- イベント名を指定
```

```
group by
    page_title,  -- ページタイトルでグルーピング
    page_location -- ページURLでグルーピング

order by
    pageviews desc -- ページビュー数降順で並び替え
```

実行結果



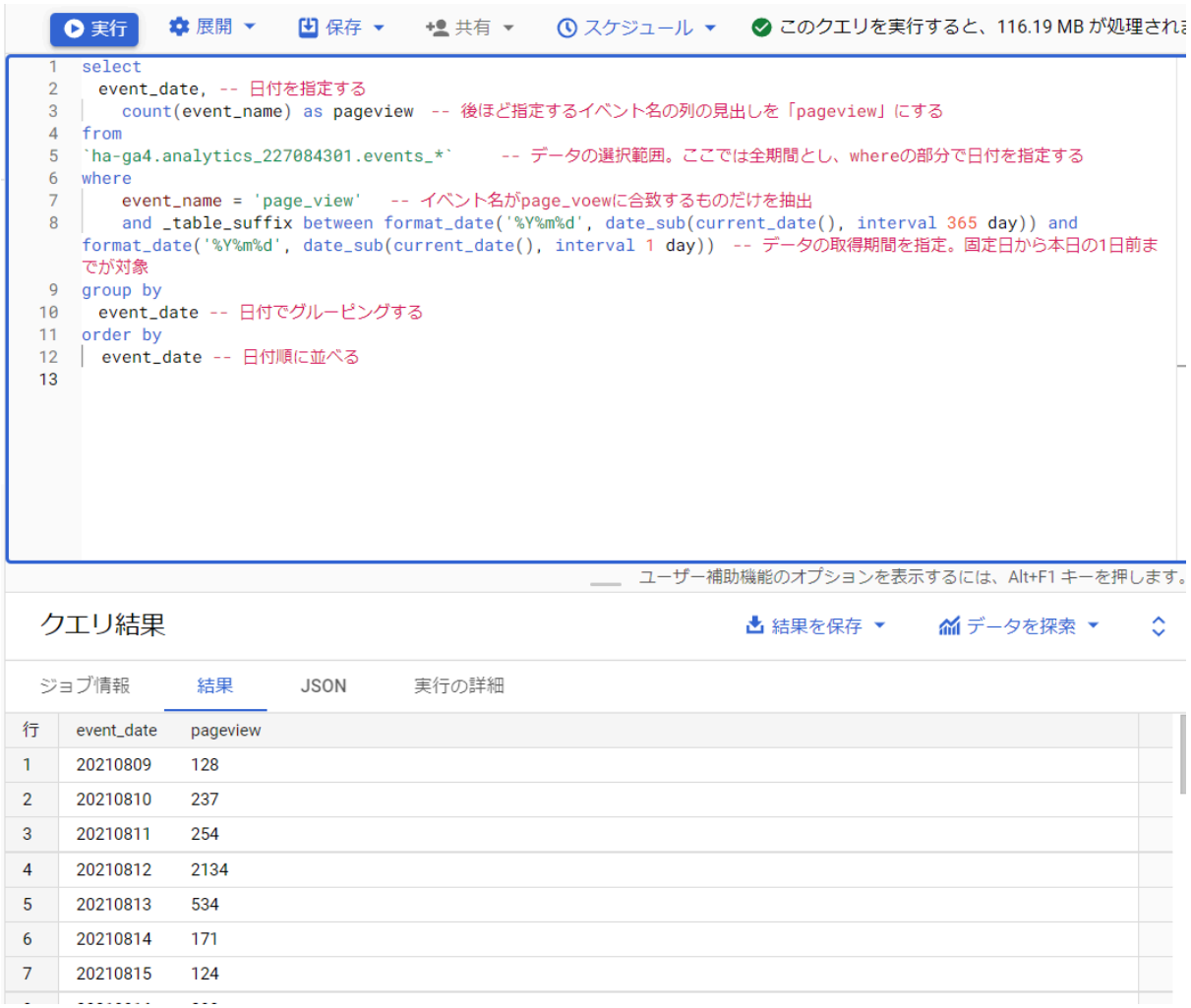
クエリ結果

行	page_title	page_location	pageview
1	株式会社HAPPY ANALYTICS | デジタルマーケティング総合支援会社	https://happyanalytics.co.jp/	220
2	null	https://go.happyanalytics.co.jp/ga4	165
3	企業情報・代表メッセージ | 株式会社HAPPY ANALYTICS	https://happyanalytics.co.jp/ceo_message/	81
4	Real Analytics （リアルアナリティクス）	https://analytics.hatenadiary.com/	70
5	著作紹介 | 株式会社HAPPY ANALYTICS	https://happyanalytics.co.jp/books/	64
6	「Google アナリティクス４」に関してよくあるQ&Aを40個まとめました！ - Real Analytics （リアルアナリティクス）	https://analytics.hatenadiary.com/entry/2021/12/03/091157	54
7	連載・執筆 | 株式会社HAPPY ANALYTICS	https://happyanalytics.co.jp/business_article/	42
8	提案型ウェブアナリスト育成講座 | 株式会社HAPPY ANALYTICS	https://happyanalytics.co.jp/schoolpbwa/	41
9	ソシャゲ分析講座　基本編（その３）：「継続率」を理解する - Real Analytics （リアルアナリティクス）	https://analytics.hatenadiary.com/entry/20131118/p1	39
10	ページが見つかりませんでした | 株式会社HAPPY ANALYTICS	http://happyanalytics.co.jp/business/seminar	33
11	お打ち合わせの調整 | 株式会社HAPPY ANALYTICS	http://happyanalytics.co.jp/mtg/	33
12	資料ダウンロード | 株式会社HAPPY ANALYTICS	https://happyanalytics.co.jp/download/	32

6：相対的な日付指定を行う

```
select
  date(timestamp_micros(event_timestamp),"Asia/Tokyo") as event_date, -- イベントの発生日付を選択
    count(event_name) as pageview  -- 後ほど指定するイベント名の列の見出しを「pageview」にする
from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
    event_name = 'page_view'   -- イベント名がpage_voewに合致するものだけを抽出
    and _table_suffix between format_date('%Y%m%d', date_sub(current_date(), interval 365 day)) and format_date('%Y%m%d', date_sub(current_date(), interval 1 day))  -- データの取得期間を指定。本日の365日前から本日の1日前までが対象
group by
  event_date -- 日付でグルーピングする
order by
  event_date -- 日付順に並べる
```

実行結果（こちらのデータセットの期間の都合上、日付の開始日が違いますが記述はあっています）

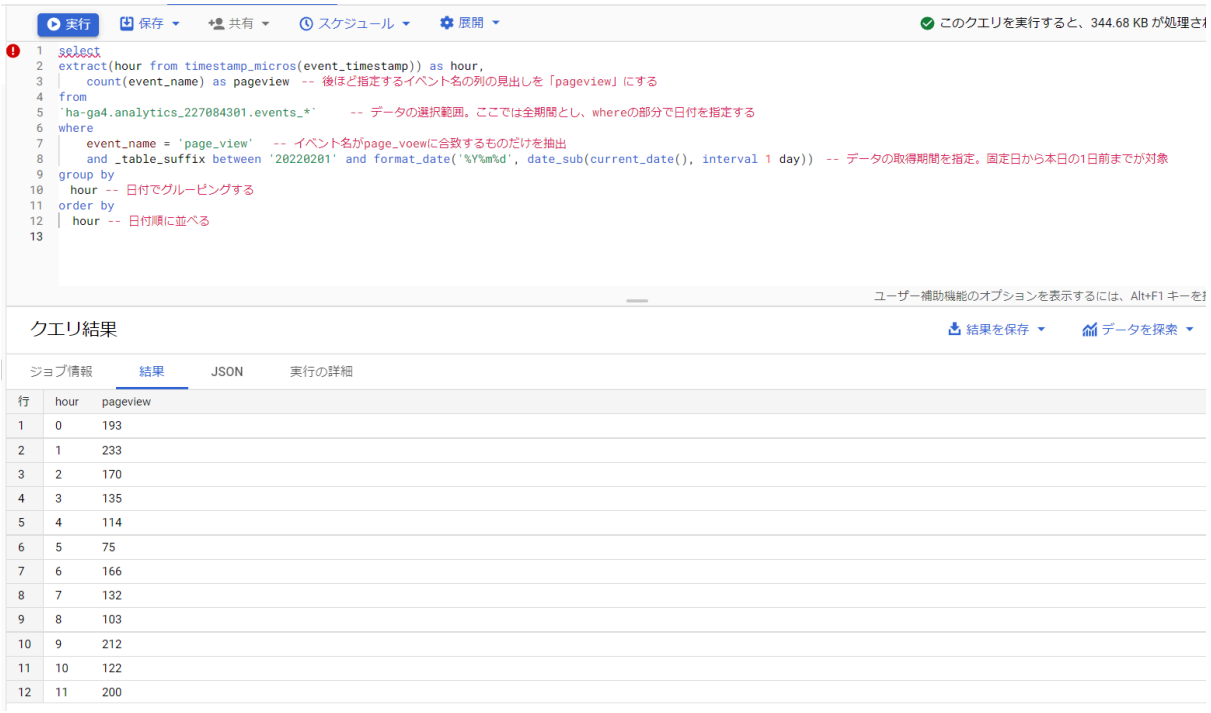


7：時間単位のデータを出す

```
select
extract(hour from timestamp_micros(event_timestamp)at time zone "Asia/Tokyo") as hour,  -- 「時」を抽出する
    count(event_name) as pageview  -- 後ほど指定するイベント名の列の見出しを「pageview」にする
from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
    event_name = 'page_view'   -- イベント名がpage_voewに合致するものだけを抽出
    and _table_suffix between '20220201' and format_date('%Y%m%d', date_sub(current_date(), interval 1 day))  -- データの取得期間を指定。固定日から本日の1日前までが対象
group by
  hour -- 時でグルーピングする
```

```
order by
  hour -- 時間順に並べる
```

実行結果



「hour」と指定している部分を全て以下に変えることで、違うグルーピングが出来ます

名称	意味
date	日付 yyyy-mm-dd形式
minute	分（1～60の値）
year	年
quarter	四半期
month	月
dayofyear	1月1日を「1」とした時に何日目か
day	日のみ(1～31の値）
week	週（1～53の値）
second	秒

複数の列を指定すれば掛け合わせが可能です。

```
select
```

```
extract(hour from timestamp_micros(event_timestamp)at time zone "Asia/Tokyo") as hour,  -- 「時」を抽出する
extract(minute from timestamp_micros(event_timestamp)at time zone "Asia/Tokyo") as minute,  -- 「分」を抽出する
    count(event_name) as pageview  -- 後ほど指定するイベント名の列の見出しを「pageview」にする
from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
    event_name = 'page_view'   -- イベント名がpage_voewに合致するものだけを抽出
    and _table_suffix between '20220201' and format_date('%Y%m%d', date_sub(current_date(), interval 1 day))  -- データの取得期間を指定。固定日から本日の1日前までが対象
group by
  hour, -- 時でグルーピングする
  minute -- 分でグルーピングする
order by
  hour, -- 時で並べる
  minute -- 分で並べる
```

実行結果

8：初回訪問日ごとのユーザー数を出す

```
select
    extract(date from timestamp_micros(user_first_touch_timestamp)at time zone "Asia/Tokyo") as day_first_touch,   -- user_first_touch_timestampという初回訪問時間が記録されている列から日付を抽出する
    count(distinct user_pseudo_id) as users   -- 該当日付のユニークなユーザー数をカウント
```

```
from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20220201' and '20220205'   -- 日付の指定
group by
 day_first_touch   -- 日付でグルーピングする
having
 day_first_touch is not null     -- タイムスタンプがnullではないという条件を満たす事で空白を除外
order by
 day_first_touch  -- 日付で並び替え
```

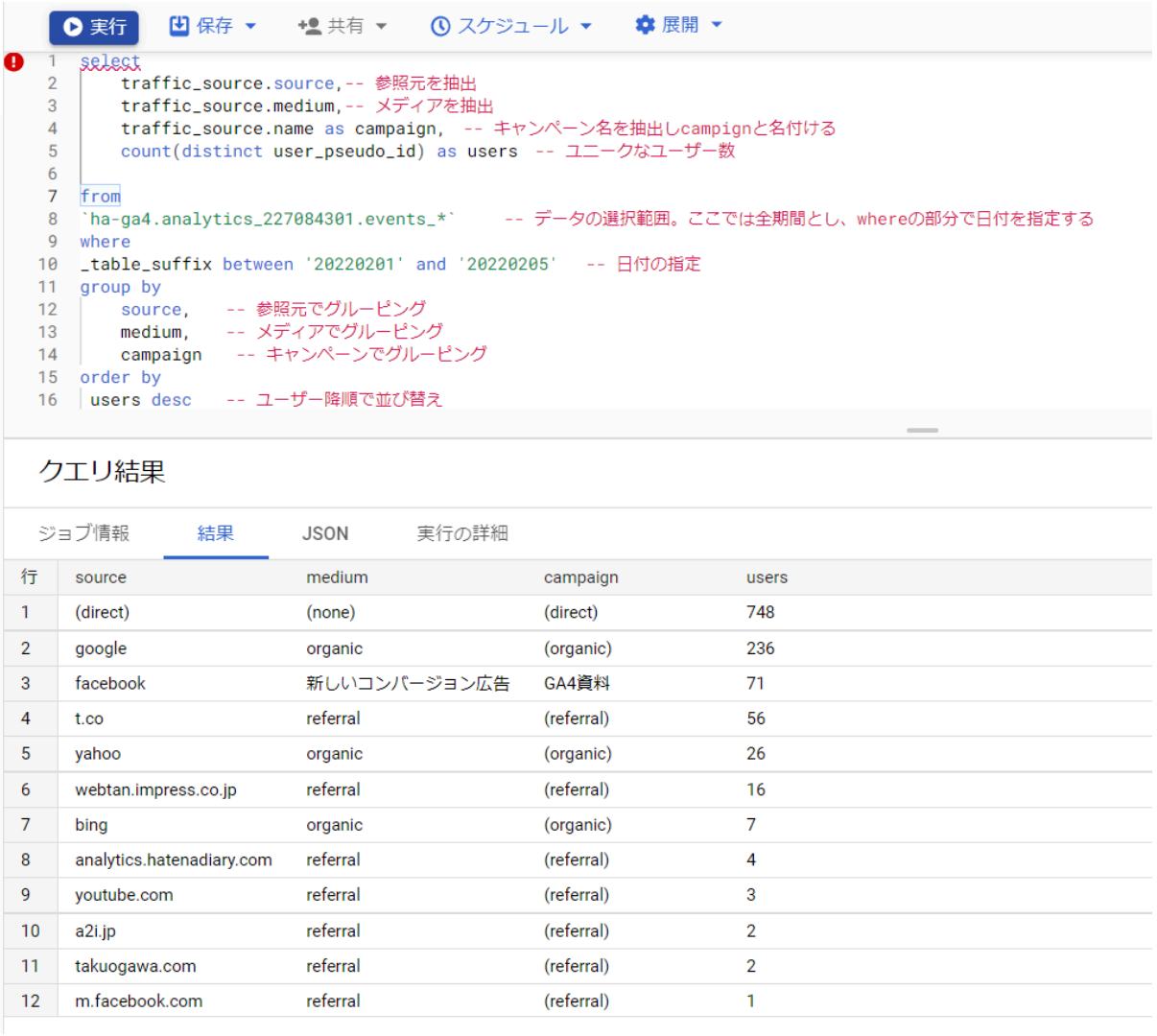
実行結果

9:流入元ごとのユーザー数を出す

```
select
    traffic_source.source,-- 参照元を抽出
    traffic_source.medium,-- メディアを抽出
    traffic_source.name as campaign,  -- キャンペーン名を抽出しcampignと名付ける
    count(distinct user_pseudo_id) as users  -- ユニークなユーザー数

from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20220201' and '20220205'   -- 日付の指定
```

```
group by
    source,    -- 参照元でグルーピング
    medium,    -- メディアでグルーピング
    campaign    -- キャンペーンでグルーピング
order by
 users desc    -- ユーザー降順で並び替え
```

実行結果



実行 保存 共有 スケジュール 展開

```
1  select
2      traffic_source.source,-- 参照元を抽出
3      traffic_source.medium,-- メディアを抽出
4      traffic_source.name as campaign,  -- キャンペーン名を抽出しcampignと名付ける
5      count(distinct user_pseudo_id) as users  -- ユニークなユーザー数
6
7  from
8  `ha-ga4.analytics_227084301.events_*`     -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
9  where
10 _table_suffix between '20220201' and '20220205'   -- 日付の指定
11 group by
12     source,    -- 参照元でグルーピング
13     medium,    -- メディアでグルーピング
14     campaign    -- キャンペーンでグルーピング
15 order by
16  users desc    -- ユーザー降順で並び替え
```

クエリ結果

ジョブ情報　結果　JSON　実行の詳細

行	source	medium	campaign	users
1	(direct)	(none)	(direct)	748
2	google	organic	(organic)	236
3	facebook	新しいコンバージョン広告	GA4資料	71
4	t.co	referral	(referral)	56
5	yahoo	organic	(organic)	26
6	webtan.impress.co.jp	referral	(referral)	16
7	bing	organic	(organic)	7
8	analytics.hatenadiary.com	referral	(referral)	4
9	youtube.com	referral	(referral)	3
10	a2i.jp	referral	(referral)	2
11	takuogawa.com	referral	(referral)	2
12	m.facebook.com	referral	(referral)	1

10：参照元 / メディアのように項目を繋げる

```
select
    concat(traffic_source.source, " / ", traffic_source.medium) as source_medium,
-- concat関数で２つのsourceとmediumを「/」でつなぐ
    count(event_name) as sessions  -- 後ほど指定するイベント名の列の見出しを「sessions」にする
from
```

```
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20220201' and '20220205'    -- 日付の指定
and event_name = 'session_start'    -- イベント名がsession_startに合致するものだけを抽出
group by
 source_medium    -- source_mediumでグルーピング
order by
 sessions desc    -- ユーザー降順で並び替え
```

実行結果

実行 保存 共有 スケジュール 展開 このクエリ

```
1  select
2      concat(traffic_source.source, " / ", traffic_source.medium) as source_medium,  -- concat関数で２つのsourceとmediumを「/」でつなぐ
3      count(event_name) as sessions  -- 後ほど指定するイベント名の列の見出しを「sessions」にする
4  from
5  `ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
6  where
7  _table_suffix between '20220201' and '20220205'    -- 日付の指定
8  and event_name = 'session_start'    -- イベント名がsession_startに合致するものだけを抽出
9  group by
10  source_medium    -- source_mediumでグルーピング
11 order by
12  sessions desc    -- ユーザー降順で並び替え
```

ユーザー補助機能のオプ

クエリ結果　結果を

ジョブ情報　結果　JSON　実行の詳細

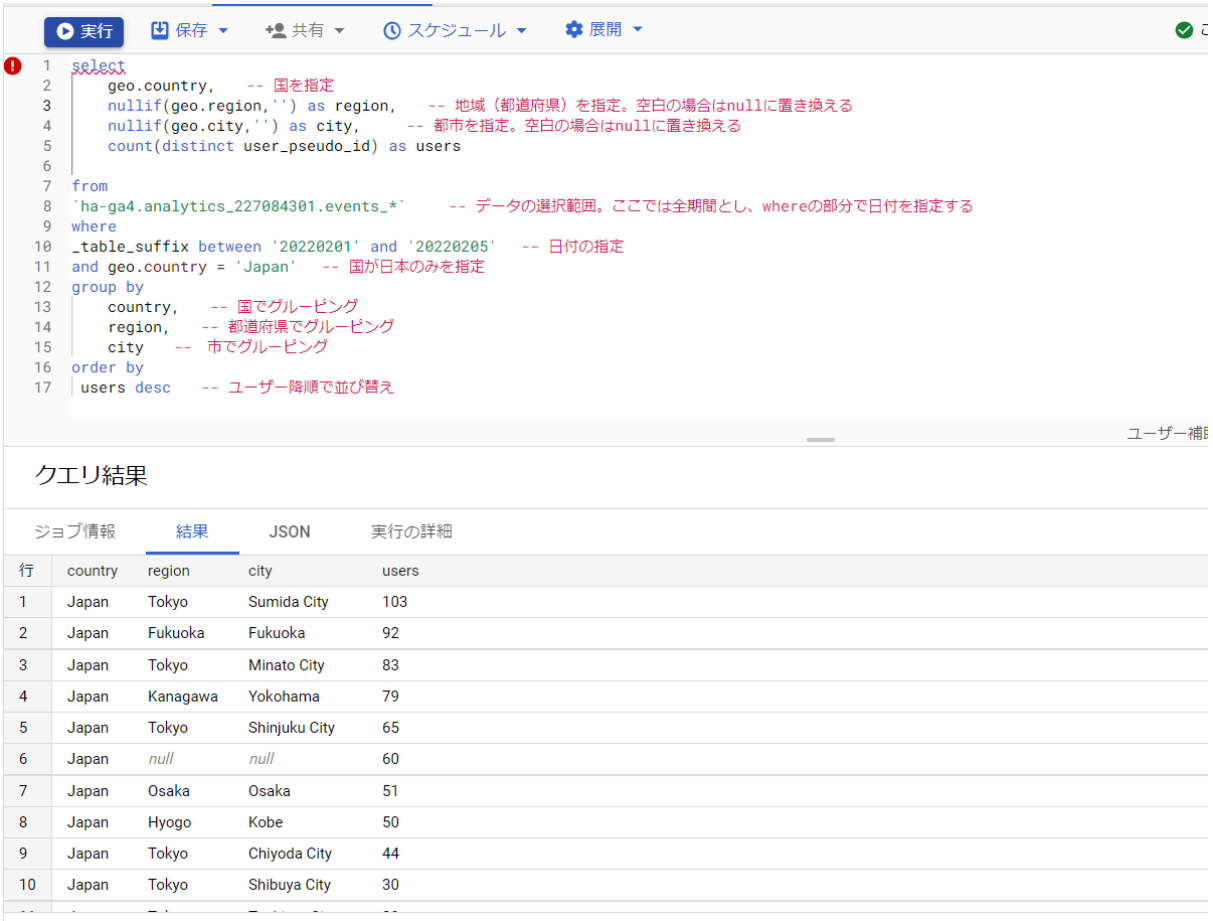
行	source_medium	sessions
1	(direct) / (none)	829
2	google / organic	306
3	facebook / 新しいコンバージョン広告	81
4	t.co / referral	62
5	yahoo / organic	27
6	webtan.impress.co.jp / referral	24
7	bing / organic	7
8	analytics.hatenadiary.com / referral	4
9	youtube.com / referral	3
10	takuogawa.com / referral	3
11	a2i.jp / referral	2

11：都道府県別のユーザー数を出す

```
select
    geo.country,    -- 国を指定
    nullif(geo.region,'') as region,    -- 地域（都道府県）を指定。空白の場合はnullに置き換える
    nullif(geo.city,'') as city,      -- 都市を指定。空白の場合はnullに置き換える
    count(distinct user_pseudo_id) as users

from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
```

```
where
_table_suffix between '20220201' and '20220205'   -- 日付の指定
and geo.country = 'Japan'   -- 国が日本のみを指定
group by
    country,    -- 国でグルーピング
    region,    -- 都道府県でグルーピング
    city    --  市でグルーピング
order by
 users desc    -- ユーザー降順で並び替え
```

実行結果



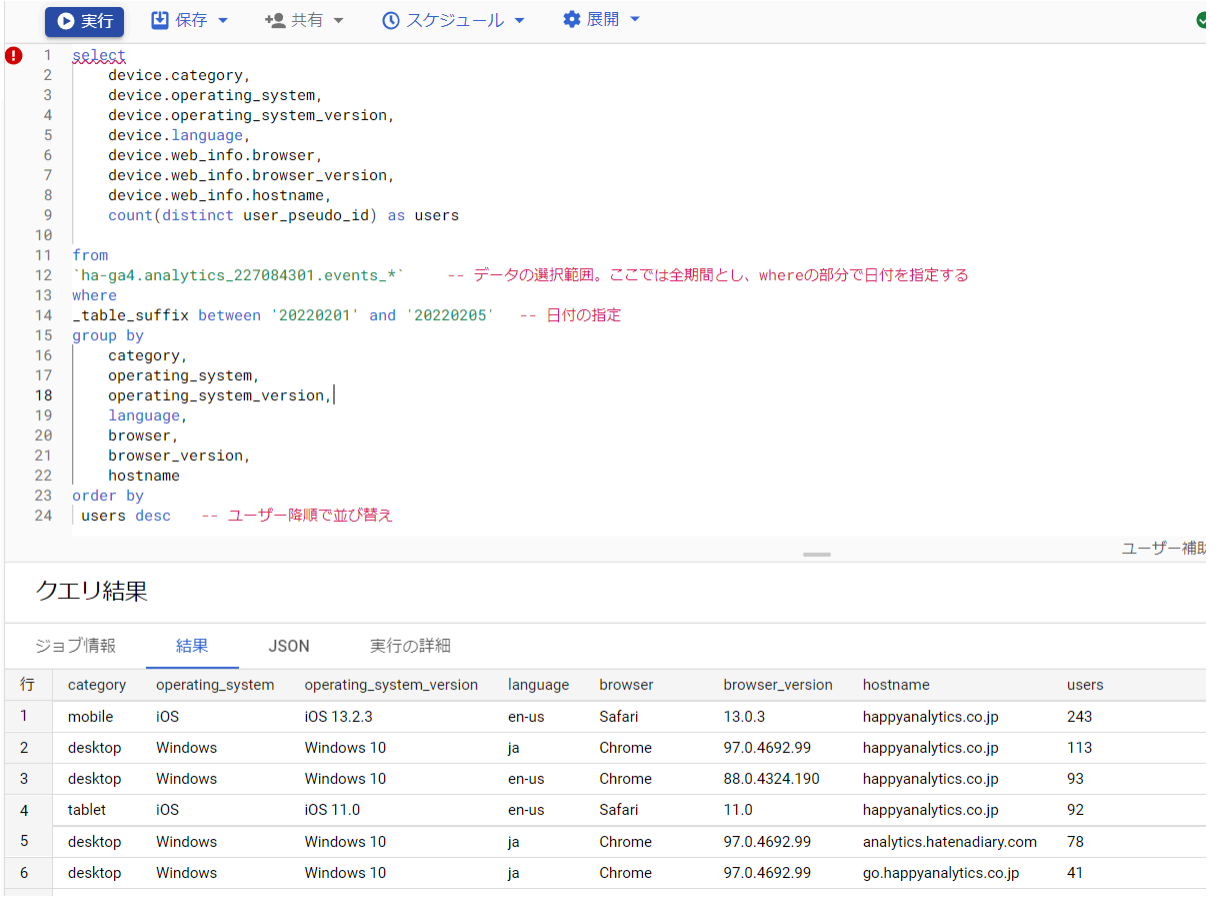
行	country	region	city	users
1	Japan	Tokyo	Sumida City	103
2	Japan	Fukuoka	Fukuoka	92
3	Japan	Tokyo	Minato City	83
4	Japan	Kanagawa	Yokohama	79
5	Japan	Tokyo	Shinjuku City	65
6	Japan	*null*	*null*	60
7	Japan	Osaka	Osaka	51
8	Japan	Hyogo	Kobe	50
9	Japan	Tokyo	Chiyoda City	44
10	Japan	Tokyo	Shibuya City	30

12：デバイスやブラウザ別の情報を取得

```
select
    device.category,
    device.operating_system,
    device.operating_system_version,
    device.language,
    device.web_info.browser,
    device.web_info.browser_version,
```

```
    device.web_info.hostname,
    count(distinct user_pseudo_id) as users

from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20220201' and '20220205'   -- 日付の指定
group by
    category,
    operating_system,
    operating_system_version,
    language,
    browser,
    browser_version,
    hostname
order by
 users desc    -- ユーザー降順で並び替え
```

実行結果　※利用時には上記から必要な項目だけを出すと良いでしょう



行	category	operating_system	operating_system_version	language	browser	browser_version	hostname	users
1	mobile	iOS	iOS 13.2.3	en-us	Safari	13.0.3	happyanalytics.co.jp	243
2	desktop	Windows	Windows 10	ja	Chrome	97.0.4692.99	happyanalytics.co.jp	113
3	desktop	Windows	Windows 10	en-us	Chrome	88.0.4324.190	happyanalytics.co.jp	93
4	tablet	iOS	iOS 11.0	en-us	Safari	11.0	happyanalytics.co.jp	92
5	desktop	Windows	Windows 10	ja	Chrome	97.0.4692.99	analytics.hatenadiary.com	78
6	desktop	Windows	Windows 10	ja	Chrome	97.0.4692.99	go.happyanalytics.co.jp	41

13：パラメータの値のユニーク数を取得

```
select
    count(distinct (select value.int_value from unnest(event_params) where key = 'ga_session_id')) as session_id_count -- ga_session_idパラメータのユニーク数(distinct)をカウントする(count)。unnestでパラメータ値を展開する必要があります
from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する

where
_table_suffix between '20220201' and '20220205'  -- 日付の指定
```

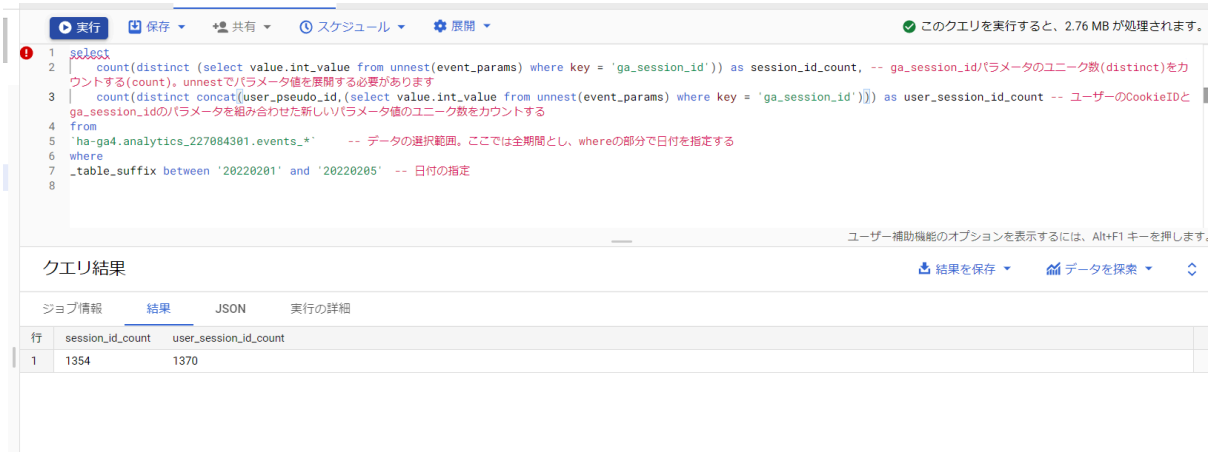
ここではga_session_id（セッションに割り当てられるパラメータ値）をカウントしています。

実行結果

参考：session_idは訪問した時間を元に生成されています。そのため同時にアクセスがあった場合、同じsession_idの値が2人のユーザーに紐づいてしまいます。そこで厳密にセッションを出す場合は、user_pseudo_idとsession_idを組み合わせた上でのユニークな件数を見る必要があります。そのための記述と実行結果は以下の通りとなります。

```
select
    count(distinct (select value.int_value from unnest(event_params) where key = 'ga_session_id')) as session_id_count, -- ga_session_idパラメータのユニーク数(distinct)をカウントする(count)。unnestでパラメータ値を展開する必要があります
    count(distinct concat(user_pseudo_id,"-",(select value.int_value from unnest(event_params) where key = 'ga_session_id'))) as user_session_id_count -- ユーザーのCookieIDとga_session_idのパラメータを組み合わせた新しいパラメータ値のユニーク数をカウントする
from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20220201' and '20220205'  -- 日付の指定
```

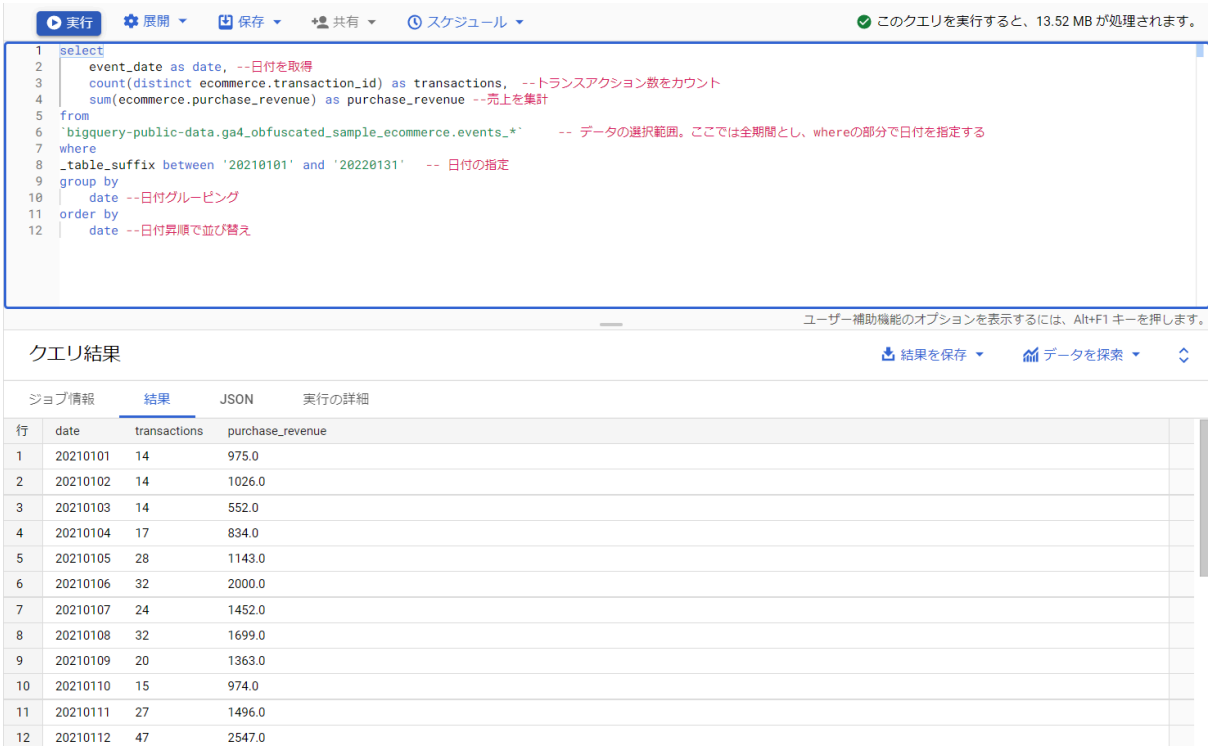
実行結果（2つの値が違う事が分かります。後者の方がより厳密なセッション数です）



14：eコマースの日別購入数と売上

```
select
  date(timestamp_micros(event_timestamp),"Asia/Tokyo") as date, -- イベントの発生日付を選択
    count(distinct ecommerce.transaction_id) as transactions,  --トランスアクション数をカウント
    sum(ecommerce.purchase_revenue) as purchase_revenue --売上を集計
from
`bigquery-public-data.ga4_obfuscated_sample_ecommerce.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20210101' and '20220131'   -- 日付の指定
group by
    date --日付グルーピング
order by
    date --日付昇順で並び替え
```

実行結果



15：eコマースの商品ごとの購入数と売上

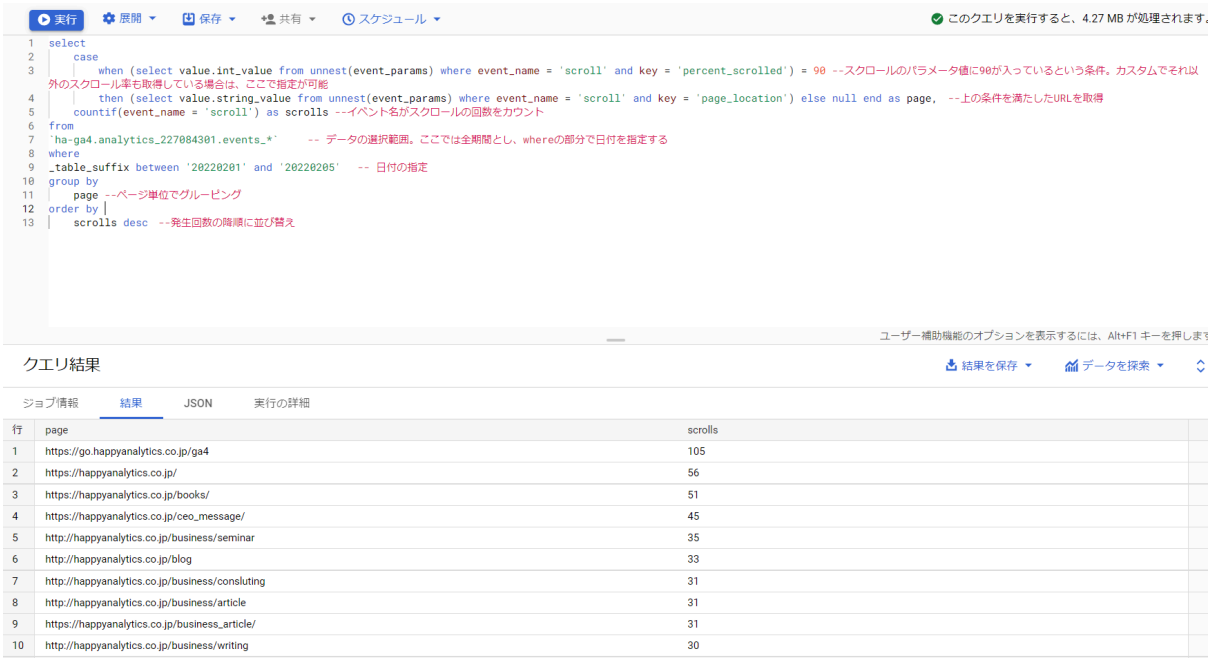
```
select
    items.item_name as item_name,
    sum(items.quantity) as items,
    sum(items.item_revenue) as item_revenue
from
`ha-ga4.analytics_227084301.events_*`,      -- データの選択範囲。ここでは全期間とし、where の部分で日付を指定する
 unnest(items) as items
where
_table_suffix between '20220101' and '20220207'   -- 日付の指定

and event_name = 'purchase'

group by
item_name
```

実行結果

16:ページごとのスクロール率の集計

```
select
    case
        when (select value.int_value from unnest(event_params) where event_name = 'scroll' and key = 'percent_scrolled') = 90 --スクロールのパラメータ値に90が入っているという条件。カスタムでそれ以外のスクロール率も取得している場合は、ここで指定が可能
        then (select value.string_value from unnest(event_params) where event_name = 'scroll' and key = 'page_location') else null end as page,  --上の条件を満たしたURLを取得
    countif(event_name = 'scroll') as scrolls --イベント名がスクロールの回数をカウント
from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20220201' and '20220205'   -- 日付の指定
group by
    page --ページ単位でグルーピング
order by
    scrolls desc  --発生回数の降順に並び替え
```

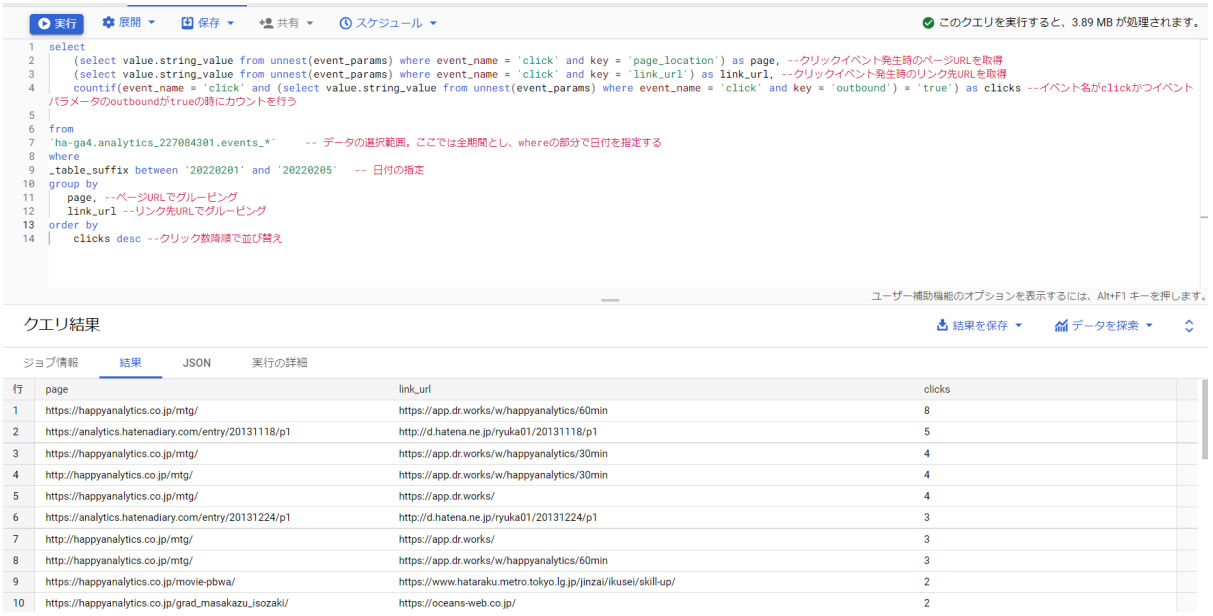
実行結果



17：外部リンクのクリック回数を取得

```
select
    (select value.string_value from unnest(event_params) where event_name = 'click' and key = 'page_location') as page, --クリックイベント発生時のページURLを取得
    (select value.string_value from unnest(event_params) where event_name = 'click' and key = 'link_url') as link_url, --クリックイベント発生時のリンク先URLを取得
    countif(event_name = 'click' and (select value.string_value from unnest(event_params) where event_name = 'click' and key = 'outbound') = 'true') as clicks --イベント名がclickかつイベントパラメータのoutboundがtrueの時にカウントを行う

from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20220201' and '20220205'   -- 日付の指定
group by
   page, --ページURLでグルーピング
   link_url --リンク先URLでグルーピング
order by
    clicks desc --クリック数降順で並び替え
```

実行結果



18：サイト内検索キーワードを取得

```
select
    (select value.string_value from unnest(event_params) where event_name = 'view_search_results' and key = 'search_term') as search_term, --イベントパラメータsearch_termを取得する
    countif(event_name = 'view_search_results') as searches --検索結果の回数を取得する

from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20220101' and '20220205'   -- 日付の指定
group by
    search_term --検索キーワードでグルーピングする
order by
    searches desc --検索回数の降順でに並び替え
```

実行結果

クエリ結果

ジョブ情報　結果　JSON　実行の詳細

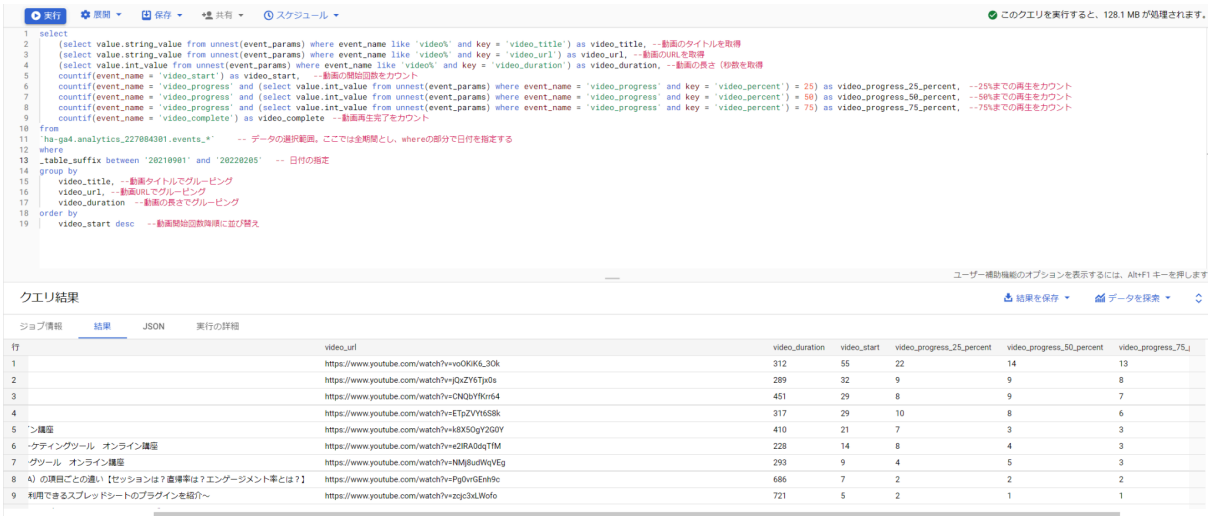
行	search_term	searches
1	*null*	5
2	アクセス解析やTwitter	3
3	データぽー経っる	1
4	データポータル	1
5	アクセス解析やツイッター	1
6	週次	1
7	ハピアナニュース	1
8	GA4	1
9	マニュアル	1

19：動画エンゲージメントの情報を取得

```
select
    (select value.string_value from unnest(event_params) where event_name like
'video%' and key = 'video_title') as video_title, --動画のタイトルを取得
    (select value.string_value from unnest(event_params) where event_name like
'video%' and key = 'video_url') as video_url, --動画のURLを取得
    (select value.int_value from unnest(event_params) where event_name like
'video%' and key = 'video_duration') as video_duration, --動画の長さ（秒数を取得
    countif(event_name = 'video_start') as video_start,   --動画の開始回数をカウント
    countif(event_name = 'video_progress' and (select value.int_value from
unnest(event_params) where event_name = 'video_progress' and key = 'video_percent')
= 25) as video_progress_25_percent,  --25%までの再生をカウント
    countif(event_name = 'video_progress' and (select value.int_value from
unnest(event_params) where event_name = 'video_progress' and key = 'video_percent')
= 50) as video_progress_50_percent,  --50%までの再生をカウント
    countif(event_name = 'video_progress' and (select value.int_value from
unnest(event_params) where event_name = 'video_progress' and key = 'video_percent')
= 75) as video_progress_75_percent,  --75%までの再生をカウント
    countif(event_name = 'video_complete') as video_complete  --動画再生完了をカウント
from
`ha-ga4.analytics_227084301.events_*`     -- データの選択範囲。ここでは全期間とし、whereの
部分で日付を指定する
where
_table_suffix between '20210901' and '20220205'   -- 日付の指定
group by
    video_title, --動画タイトルでグルーピング
    video_url, --動画URLでグルーピング
```

```
    video_duration   --動画の長さでグルーピング
order by
    video_start desc    --動画開始回数降順に並び替え
```

実行結果



```
select
    (select value.string_value from unnest(event_params) where event_name like 'video%' and key = 'video_title') as video_title, --動画のタイトルを取得
    (select value.string_value from unnest(event_params) where event_name like 'video%' and key = 'video_url') as video_url, --動画のURLを取得
    (select value.int_value from unnest(event_params) where event_name like 'video%' and key = 'video_duration') as video_duration, --動画の長さ（秒数を取得
    countif(event_name = 'video_start') as video_start,   --動画の開始回数をカウント
    countif(event_name = 'video_progress' and (select value.int_value from unnest(event_params) where event_name = 'video_progress' and key = 'video_percent') = 25) as video_progress_25_percent,  --25%までの再生をカウント
    countif(event_name = 'video_progress' and (select value.int_value from unnest(event_params) where event_name = 'video_progress' and key = 'video_percent') = 50) as video_progress_50_percent,  --50%までの再生をカウント
    countif(event_name = 'video_progress' and (select value.int_value from unnest(event_params) where event_name = 'video_progress' and key = 'video_percent') = 75) as video_progress_75_percent,  --75%までの再生をカウント
    countif(event_name = 'video_complete') as video_complete  --動画再生完了をカウント
from
`ha-ga4.analytics_227084301.events_*`    -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20210901' and '20220205'   -- 日付の指定
group by
    video_title, --動画タイトルでグルーピング
    video_url, --動画URLでグルーピング
    video_duration  --動画の長さでグルーピング
order by
    video_start desc   --動画開始回数降順に並び替え
```

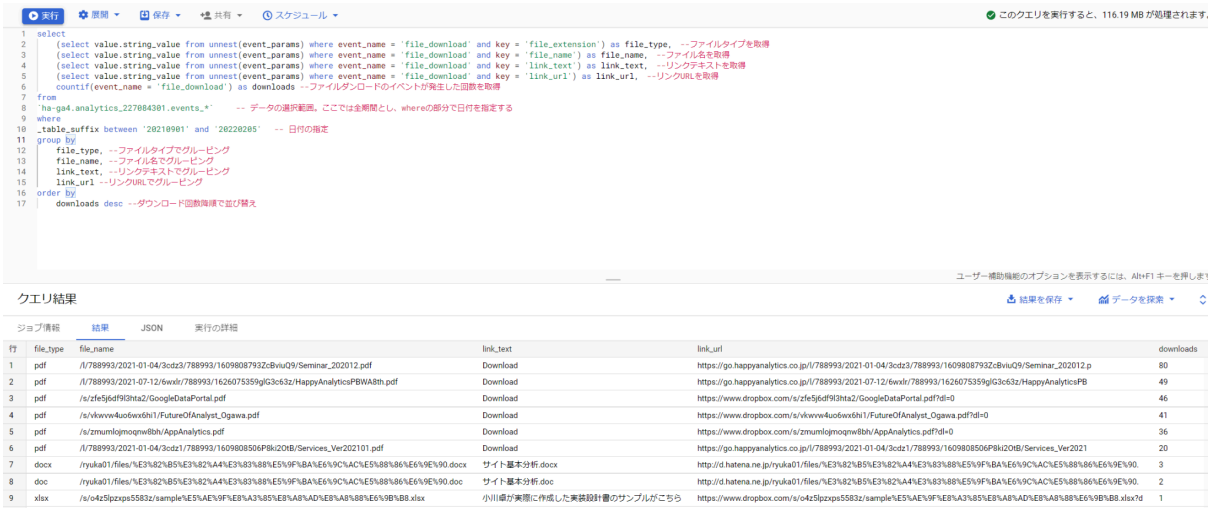
クエリ結果

行	video_url	video_duration	video_start	video_progress_25_percent	video_progress_50_percent	video_progress_75_
1	https://www.youtube.com/watch?v=voOKiK6_3Ok	312	55	22	14	13
2	https://www.youtube.com/watch?v=jQxZY6TjxOs	289	32	9	9	8
3	https://www.youtube.com/watch?v=CNQbYfKrn64	451	29	8	9	7
4	https://www.youtube.com/watch?v=ETpZVYt6S8k	317	29	10	8	6
5	ン講座	https://www.youtube.com/watch?v=k8X5OgY2G0Y	410	21	7	3
6	ケティングツール　オンライン講座	https://www.youtube.com/watch?v=e2IRA0dqTfM	228	14	8	4
7	グツール　オンライン講座	https://www.youtube.com/watch?v=NMj8udWqVEg	293	9	4	5
8	4）の項目ごとの違い【セッションは？直帰率は？エンゲージメント率とは？】	https://www.youtube.com/watch?v=PgOvrGEnh9c	686	7	2	2
9	利用できるスプレッドシートのプラグインを紹介～	https://www.youtube.com/watch?v=zcjc3xLWofo	721	5	2	1

20：ファイルダウンロードの情報を取得

```
select
    (select value.string_value from unnest(event_params) where event_name = 'file_download' and key = 'file_extension') as file_type,  --ファイルタイプを取得
    (select value.string_value from unnest(event_params) where event_name = 'file_download' and key = 'file_name') as file_name,  --ファイル名を取得
    (select value.string_value from unnest(event_params) where event_name = 'file_download' and key = 'link_text') as link_text,  --リンクテキストを取得
    (select value.string_value from unnest(event_params) where event_name = 'file_download' and key = 'link_url') as link_url,  --リンクURLを取得
    countif(event_name = 'file_download') as downloads --ファイルダンロードのイベントが発生した回数を取得
from
`ha-ga4.analytics_227084301.events_*`    -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20210901' and '20220205'   -- 日付の指定
group by
    file_type, --ファイルタイプでグルーピング
    file_name, --ファイル名でグルーピング
    link_text, --リンクテキストでグルーピング
    link_url --リンクURLでグルーピング
order by
    downloads desc --ダウンロード回数降順で並び替え
```

実行結果



```
select
    (select value.string_value from unnest(event_params) where event_name = 'file_download' and key = 'file_extension') as file_type, --ファイルタイプを取得
    (select value.string_value from unnest(event_params) where event_name = 'file_download' and key = 'file_name') as file_name, --ファイル名を取得
    (select value.string_value from unnest(event_params) where event_name = 'file_download' and key = 'link_text') as link_text, --リンクテキストを取得
    (select value.string_value from unnest(event_params) where event_name = 'file_download' and key = 'link_url') as link_url, --リンクURLを取得
    countif(event_name = 'file_download') as downloads --ファイルダンロードのイベントが発生した回数を取得
from
`ha-ga4.analytics_227084301.events_*`    -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20210901' and '20220205'  -- 日付の指定
group by
    file_type, --ファイルタイプでグルーピング
    file_name, --ファイル名でグルーピング
    link_text, --リンクテキストでグルーピング
    link_url --リンクURLでグルーピング
order by
    downloads desc --ダウンロード回数降順で並び替え
```

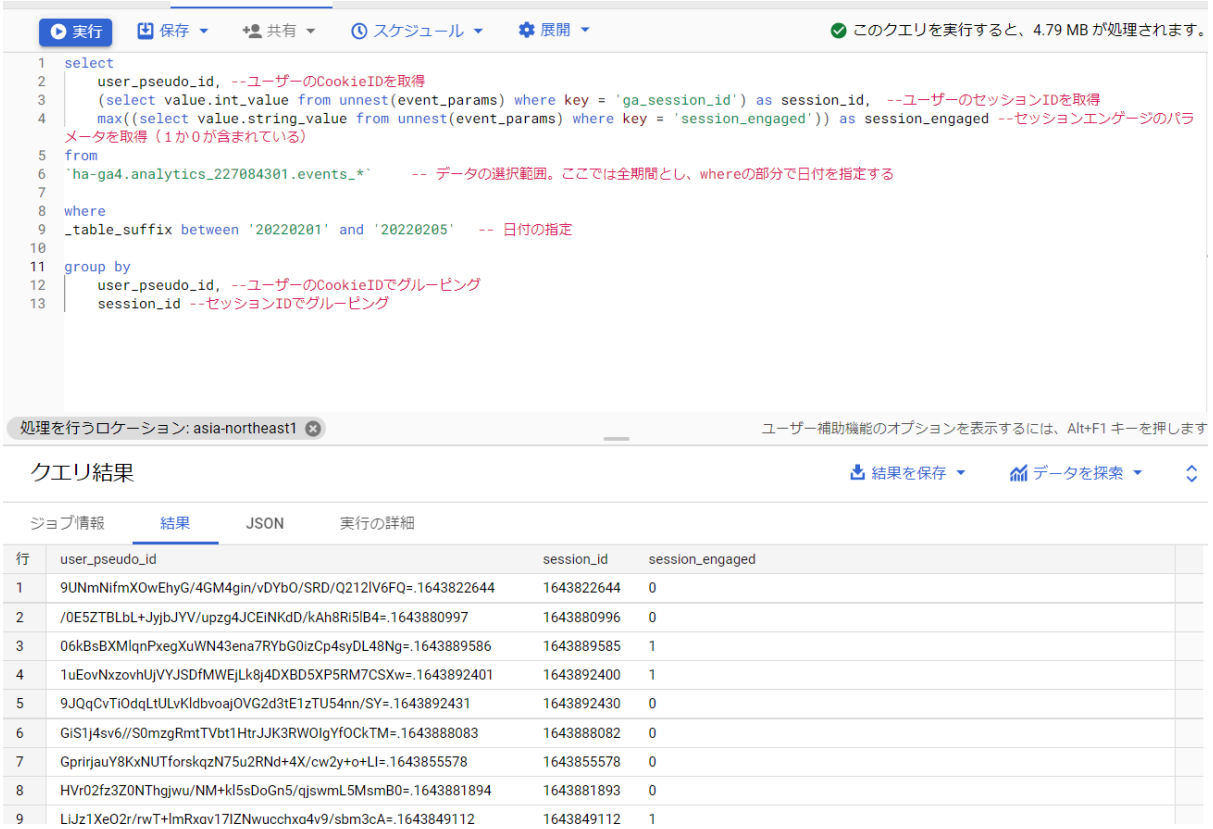
クエリ結果

行	file_type	file_name	link_text	link_url	downloads
1	pdf	/l/788993/2021-01-04/3cdz3/788993/1609808793ZcBviuQ9/Seminar_202012.pdf	Download	https://go.happyanalytics.co.jp/l/788993/2021-01-04/3cdz3/788993/1609808793ZcBviuQ9/Seminar_202012.p	80
2	pdf	/l/788993/2021-07-12/6wxlr/788993/1626075359glG3c63z/HappyAnalyticsPBWA8th.pdf	Download	https://go.happyanalytics.co.jp/l/788993/2021-07-12/6wxlr/788993/1626075359glG3c63z/HappyAnalyticsPB	49
3	pdf	/s/zfe5j6df9l3hta2/GoogleDataPortal.pdf	Download	https://www.dropbox.com/s/zfe5j6df9l3hta2/GoogleDataPortal.pdf?dl=0	46
4	pdf	/s/vkwvw4uo6wx6hi1/FutureOfAnalyst_Ogawa.pdf	Download	https://www.dropbox.com/s/vkwvw4uo6wx6hi1/FutureOfAnalyst_Ogawa.pdf?dl=0	41
5	pdf	/s/zmumlojmoqnw8bh/AppAnalytics.pdf	Download	https://www.dropbox.com/s/zmumlojmoqnw8bh/AppAnalytics.pdf?dl=0	36
6	pdf	/l/788993/2021-01-04/3cdz1/788993/1609808506P8ki2OtB/Services_Ver202101.pdf	Download	https://go.happyanalytics.co.jp/l/788993/2021-01-04/3cdz1/788993/1609808506P8ki2OtB/Services_Ver2021	20
7	docx	/ryuka01/files/%E3%82%B5%E3%82%A4%E3%83%88%E5%9F%BA%E6%9C%AC%E5%88%86%E6%9E%90.docx	サイト基本分析.docx	http://d.hatena.ne.jp/ryuka01/files/%E3%82%B5%E3%82%A4%E3%83%88%E5%9F%BA%E6%9C%AC%E5%88%86%E6%9E%90.	3
8	doc	/ryuka01/files/%E3%82%B5%E3%82%A4%E3%83%88%E5%9F%BA%E6%9C%AC%E5%88%86%E6%9E%90.doc	サイト基本分析.doc	http://d.hatena.ne.jp/ryuka01/files/%E3%82%B5%E3%82%A4%E3%83%88%E5%9F%BA%E6%9C%AC%E5%88%86%E6%9E%90.	2
9	xlsx	/s/o4z5lpzxps5583z/sample%E5%AE%9F%E8%A3%85%E8%A8%AD%E8%A8%88%E6%9B%B8.xlsx	小川卓が実際に作成した実装設計書のサンプルがこちら	https://www.dropbox.com/s/o4z5lpzxps5583z/sample%E5%AE%9F%E8%A3%85%E8%A8%AD%E8%A8%88%E6%9B%B8.xlsx?d	1

21：セッション後のエンゲージを確認

```
select
    user_pseudo_id, --ユーザーのCookieIDを取得
    (select value.int_value from unnest(event_params) where key = 'ga_session_id')
as session_id,  --ユーザーのセッションIDを取得
    max((select value.string_value from unnest(event_params) where key =
'session_engaged')) as session_engaged --セッションエンゲージのパラメータを取得（1か0が含まれている）
from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する

where
_table_suffix between '20220201' and '20220205'   -- 日付の指定

group by
    user_pseudo_id, --ユーザーのCookieIDでグルーピング
    session_id --セッションIDでグルーピング
```

実行結果



22：エンゲージしたセッション数を取得

```
with predata as （ --14行目までのselect文に名称をつける。ここではpredata。「(」を忘れないように
select
    user_pseudo_id, --ユーザーのCookieIDを取得
    (select value.int_value from unnest(event_params) where key = 'ga_session_id') as session_id,  --ユーザーのセッションIDを取得
    max((select value.string_value from unnest(event_params) where key = 'session_engaged')) as session_engaged --セッションエンゲージのパラメータを取得（1か0が含まれている）
from
`ha-ga4.analytics_227084301.events_*`     -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する

where
_table_suffix between '20220201' and '20220205'   -- 日付の指定

group by
    user_pseudo_id, --ユーザーのCookieIDでグルーピング
    session_id) --セッションIDでグルーピング。クエリの最後でpredataを閉じるため「)」を忘れないように

select
```

```
    count(distinct concat(user_pseudo_id,"-",session_id)) as sessions,  --CookieIDとセッションIDをつなげユニーク数をカウントし、厳密なセッション数を取得
    count(distinct case when session_engaged = '1' then concat(user_pseudo_id,"-",session_id) else null end) as engaged_sessions --session_engagedの値が1の時に、CookieIDとセッションIDをつなげたユニーク数をカウントし、エンゲージメントセッション数を算出。1以外の場合はnullに変換
from
    predata --上記の2つのcountをpredataのクエリ結果から取得
```

実行結果

```
with predata as ( --14行目までのselect文に名称をつける。ここではpredata。「(」を忘れないように
select
    user_pseudo_id, --ユーザーのCookieIDを取得
    (select value.int_value from unnest(event_params) where key = 'ga_session_id') as session_id,  --ユーザーのセッションIDを取得
    max((select value.string_value from unnest(event_params) where key = 'session_engaged')) as session_engaged --セッションエンゲージのパラメータを取得（1か0が含まれている）
from
`ha-ga4.analytics_227084301.events_*`     -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する

where
_table_suffix between '20220201' and '20220205'   -- 日付の指定

group by
    user_pseudo_id, --ユーザーのCookieIDでグルーピング
    session_id) --セッションIDでグルーピング。クエリの最後でpredataを閉じるため「)」を忘れないように

select
    count(distinct concat(user_pseudo_id,session_id)) as sessions,  --CookieIDとセッションIDをつなげユニーク数をカウントし、厳密なセッション数を取得
    count(distinct case when session_engaged = '1' then concat(user_pseudo_id,session_id) else null end) as engaged_sessions --session_engagedの値が1の時に、CookieIDとセッションIDをつなげたユニーク数をカウントし、エンゲージメントセッション数を算出。1以外の場合はnullに変換
from
    predata --上記の2つのcountをpredataのクエリ結果から取得
```

ユーザー補助機能のオプションを表示するには、Alt+F1 キーを押します。

クエリ結果　結果を保存　データを探索

ジョブ情報　結果　JSON　実行の詳細

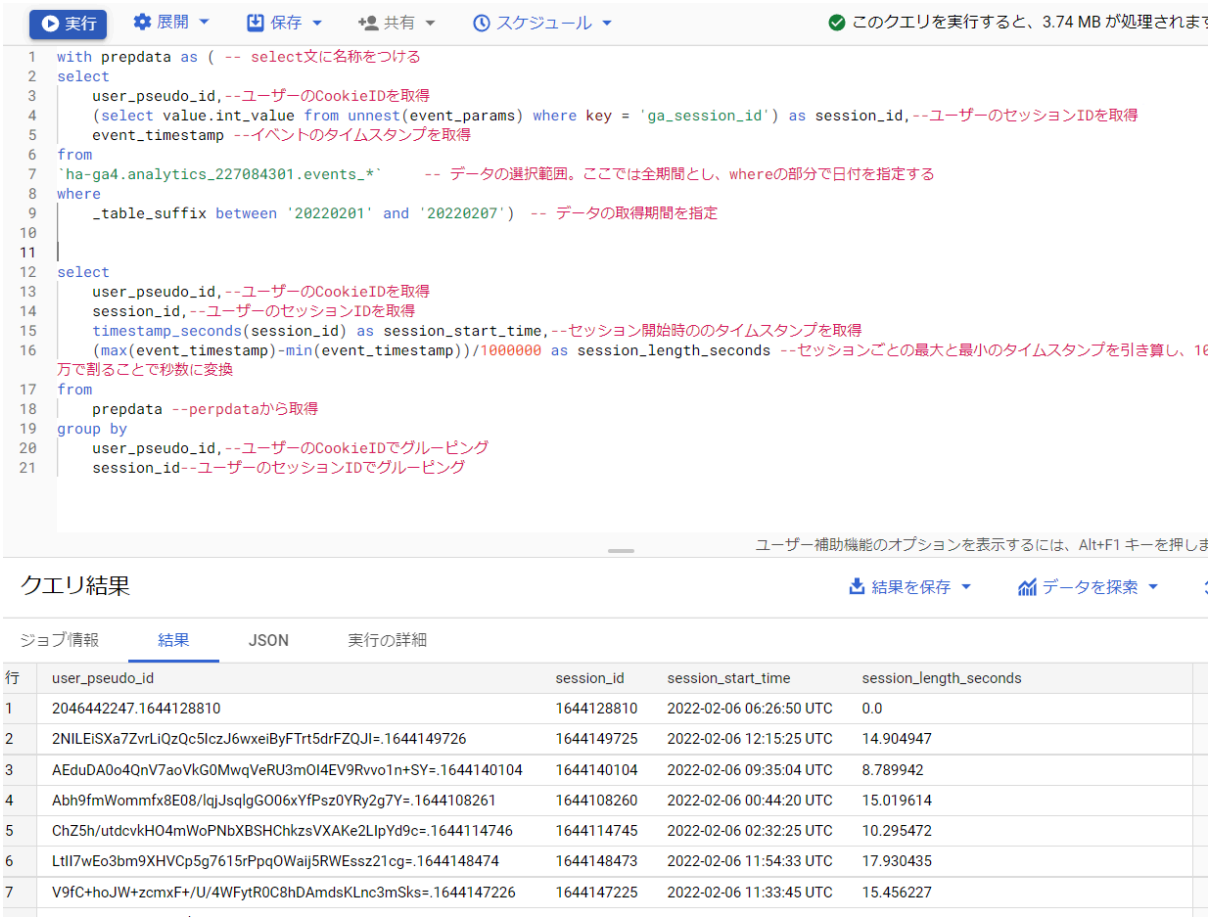
行	sessions	engaged_sessions
1	1370	696

23：セッションごとの滞在時間を取得

```
with prepdata as ( -- select文に名称をつける
select
    user_pseudo_id,--ユーザーのCookieIDを取得
    (select value.int_value from unnest(event_params) where key = 'ga_session_id') as session_id,--ユーザーのセッションIDを取得
    event_timestamp --イベントのタイムスタンプを取得
from
`ha-ga4.analytics_227084301.events_*`     -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
    _table_suffix between '20220201' and '20220207')  -- データの取得期間を指定


select
    user_pseudo_id,--ユーザーのCookieIDを取得
    session_id,--ユーザーのセッションIDを取得
    timestamp_seconds(session_id) as session_start_time,--セッション開始時ののタイムスタンプを取得
    (max(event_timestamp)-min(event_timestamp))/1000000 as session_length_seconds --セッションごとの最大と最小のタイムスタンプを引き算し、100万で割ることで秒数に変換
```

```
from
    prepdata --perpdataから取得
group by
    user_pseudo_id,--ユーザーのCookieIDでグルーピング
    session_id--ユーザーのセッションIDでグルーピング
```

実行結果



実行 | 展開 | 保存 | 共有 | スケジュール | このクエリを実行すると、3.74 MB が処理されます

```
1  with prepdata as ( -- select文に名称をつける
2  select
3      user_pseudo_id,--ユーザーのCookieIDを取得
4      (select value.int_value from unnest(event_params) where key = 'ga_session_id') as session_id,--ユーザーのセッションIDを取得
5      event_timestamp --イベントのタイムスタンプを取得
6  from
7  `ha-ga4.analytics_227084301.events_*`     -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
8  where
9      _table_suffix between '20220201' and '20220207')  -- データの取得期間を指定
10
11
12 select
13     user_pseudo_id,--ユーザーのCookieIDを取得
14     session_id,--ユーザーのセッションIDを取得
15     timestamp_seconds(session_id) as session_start_time,--セッション開始時ののタイムスタンプを取得
16     (max(event_timestamp)-min(event_timestamp))/1000000 as session_length_seconds --セッションごとの最大と最小のタイムスタンプを引き算し、16
   万で割ることで秒数に変換
17 from
18     prepdata --perpdataから取得
19 group by
20     user_pseudo_id,--ユーザーのCookieIDでグルーピング
21     session_id--ユーザーのセッションIDでグルーピング
```

ユーザー補助機能のオプションを表示するには、Alt+F1 キーを押しま

クエリ結果　結果を保存　データを探索

ジョブ情報　結果　JSON　実行の詳細

行	user_pseudo_id	session_id	session_start_time	session_length_seconds
1	2046442247.1644128810	1644128810	2022-02-06 06:26:50 UTC	0.0
2	2NILEiSXa7ZvrLiQzQc5IczJ6wxeiByFTrt5drFZQJI=.1644149726	1644149725	2022-02-06 12:15:25 UTC	14.904947
3	AEduDA0o4QnV7aoVkG0MwqVeRU3mOI4EV9Rvvo1n+SY=.1644140104	1644140104	2022-02-06 09:35:04 UTC	8.789942
4	Abh9fmWommfx8E08/lqjJsqlgGO06xYfPsz0YRy2g7Y=.1644108261	1644108260	2022-02-06 00:44:20 UTC	15.019614
5	ChZ5h/utdcvkHO4mWoPNbXBSHChkzsVXAKe2LlpYd9c=.1644114746	1644114745	2022-02-06 02:32:25 UTC	10.295472
6	LtIl7wEo3bm9XHVCp5g7615rPpqOWaij5RWEssz21cg=.1644148474	1644148473	2022-02-06 11:54:33 UTC	17.930435
7	V9fC+hoJW+zcmxF+/U/4WFytR0C8hDAmdsKLnc3mSks=.1644147226	1644147225	2022-02-06 11:33:45 UTC	15.456227

24：流入元ごとのセッション数を取得

```
with predata as ( --select文に名称をつける。ここではpredata。「(」を忘れないように
select
    user_pseudo_id, --ユーザーのCookieIDを取得
    (select value.int_value from unnest(event_params) where key = 'ga_session_id')
as session_id,  --ユーザーのセッションIDを取得
    first_value((select value.string_value from unnest(event_params) where key =
'session_engaged')) as session_engaged, --セッションエンゲージのパラメータ取得
    first_value((select value.string_value from unnest(event_params) where key =
'medium')) as medium, --イベントパラメータからメディアを取得。session_startのイベントには流
入元は記録されていないため、該当セッションID内に含まれているメディアを取得してくる
    first_value((select value.string_value from unnest(event_params) where key =
'source')) as source --イベントパラメータから参照元を取得。session_startのイベントには流入元
は記録されていないため、該当セッションID内に含まれているメディアを取得してくる
```

```
from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20220201' and '20220205'   -- 日付の指定
group by
    user_pseudo_id, --ユーザーのCookieIDでグルーピング
    session_id) --セッションIDでグルーピング。クエリの最後でpredataを閉じるため「)」を忘れないように

select
    concat(ifnull(source,'(direct)'),' / ',ifnull(medium,'(none)')) as session_source_medium, --セッションとメディアを「/」でつなげる。参照元に値が入っていない場合は参照元を(direct)に変換、メディアに値が入っていない場合はメディアを(none)に変換
    count(distinct concat(user_pseudo_id,"-",session_id)) as sessions,  --CookieIDとセッションIDをつなげユニーク数をカウントし、厳密なセッション数を取得
    count(distinct case when session_engaged = '1' then concat(user_pseudo_id,"-",session_id) else null end) as engaged_sessions --session_engagedの値が1の時に、CookieIDとセッションIDをつなげたユニーク数をカウントし、エンゲージメントセッション数を算出。1以外の場合はnullに変換

from
    predata --上記の2つのcountをpredataのクエリ結果から取得
group by
 session_source_medium   --参照元メディアでグルーピング
order by
 sessions desc   --セッション数、降順で並び替え
```

実行結果

25：エンゲージしたユーザー数を取得

```
with predata as ( --14行目までのselect文に名称をつける。ここではpredata。「(」を忘れないように
select
    user_pseudo_id, --ユーザーのCookieIDを取得
    sum((select value.int_value from unnest(event_params) where key = 'engagement_time_msec')) as engagement_time_msec --ユーザーのエンゲージメント時間（画面がユーザーの前面に出ていた時間（ミリ秒）を取得。

from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する

where
_table_suffix between '20220201' and '20220205'   -- 日付の指定

group by
    user_pseudo_id) --ユーザーのCookieIDでグルーピング。クエリの最後でpredataを閉じるため「)」を忘れないように

select
    count(distinct user_pseudo_id) as users, --ユーザーのCookieIDユニーク数をユーザー数として取得
```

```
    count(distinct case when engagement_time_msec > 0 then user_pseudo_id else null
end) as active_users   --エンゲージメント時間が0ミリ秒より大きい場合はアクティブユーザーとす
る。0の場合はnullとし、アクティブユーザーのユーザーCookieiDのユニーク数を取得

from
    predata --上記の2つのcountをpredataのクエリ結果から取得
```

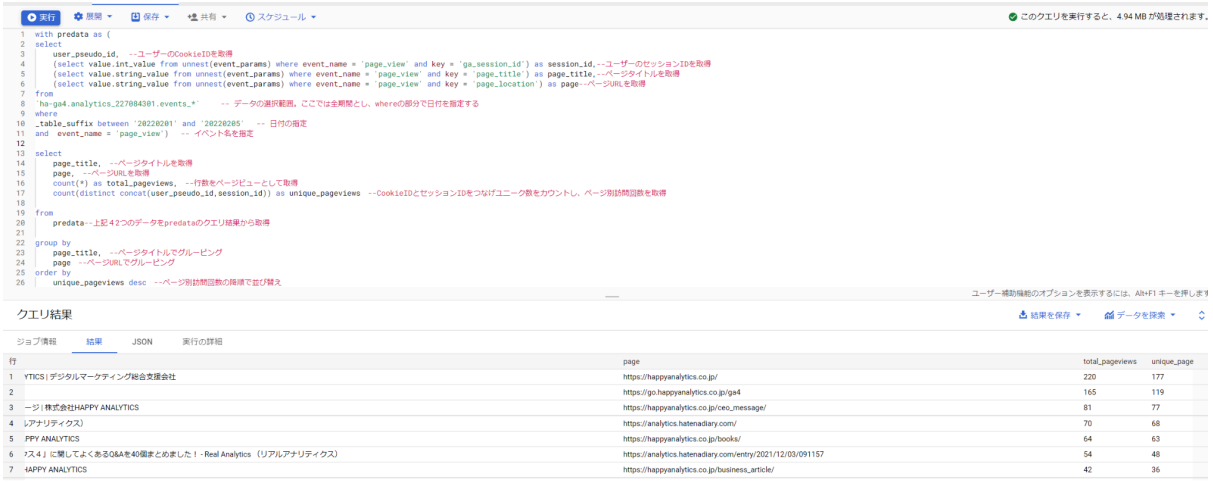
実行結果

26：ページ別の訪問回数を取得

```
with predata as (
select
    user_pseudo_id,  --ユーザーのCookieIDを取得
    (select value.int_value from unnest(event_params) where event_name =
'page_view' and key = 'ga_session_id') as session_id,--ユーザーのセッションIDを取得
    (select value.string_value from unnest(event_params) where event_name =
'page_view' and key = 'page_title') as page_title,--ページタイトルを取得
    (select value.string_value from unnest(event_params) where event_name =
'page_view' and key = 'page_location') as page--ページURLを取得
from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの
部分で日付を指定する
where
_table_suffix between '20220201' and '20220205')   -- 日付の指定

select
    page_title,  --ページタイトルを取得
    page,  --ページURLを取得
    count(*) as total_pageviews,  --行数をページビューとして取得
```

```
    count(distinct concat(user_pseudo_id,"-",session_id)) as unique_pageviews
--CookieIDとセッションIDをつなげユニーク数をカウントし、ページ別訪問回数を取得

from
    predata--上記４2つのデータをpredataのクエリ結果から取得

group by
    page_title,  --ページタイトルでグルーピング
    page  --ページURLでグルーピング
order by
    unique_pageviews desc  --ページ別訪問回数の降順で並び替え
```

実行結果



クエリ結果

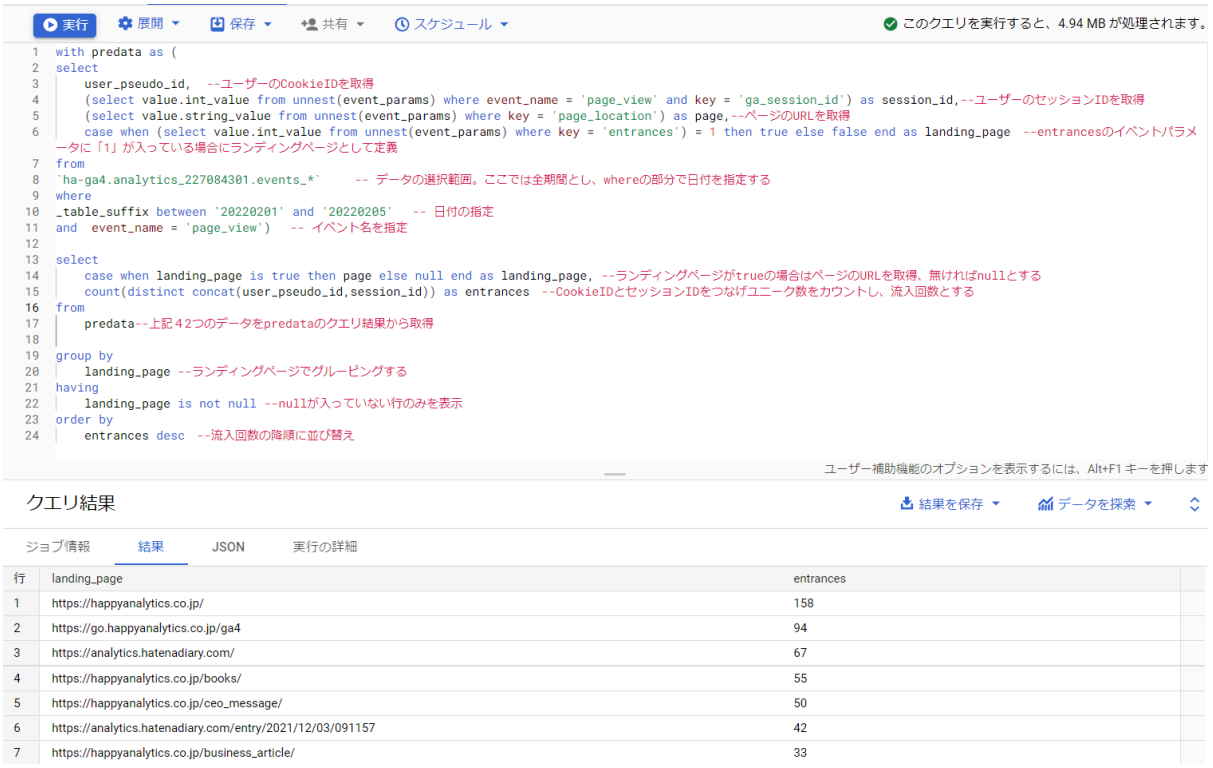
行	page	total_pageviews	unique_page
1	https://happyanalytics.co.jp/	220	177
2	https://go.happyanalytics.co.jp/ga4	165	119
3	https://happyanalytics.co.jp/ceo_message/	81	77
4	https://analytics.hatenadiary.com/	70	68
5	https://happyanalytics.co.jp/books/	64	63
6	https://analytics.hatenadiary.com/entry/2021/12/03/091157	54	48
7	https://happyanalytics.co.jp/business_article/	42	36

27：ランディングページごとの流入回数を取得

```
with predata as (
select
user_pseudo_id, --ユーザーのCookieIDを取得
(select value.int_value from unnest(event_params) where event_name = 'page_view'
and key = 'ga_session_id') as session_id,--ユーザーのセッションIDを取得
case
when (select value.int_value from unnest(event_params) where key = 'entrances') = 1
then (select value.string_value from unnest(event_params) where key =
'page_location')
end as landing_page -- entrancesのパラメータに「1が入ってる場合」page_locationの値を
landing_pageとして扱う
from
`ha-ga4.analytics_227084301.events_*` -- データの選択範囲。ここでは全期間とし、whereの部分
で日付を指定する
where
_table_suffix between '20220201' and '20220205' -- 日付の指定
and event_name = 'page_view'
) -- イベント名を指定
```

```
select
landing_page,
count(distinct concat(user_pseudo_id, "-", session_id)) as entrances --CookieIDとセッションIDをつなげユニーク数をカウントし、流入回数とする
from
predata--上記2つのデータをpredataのクエリ結果から取得
where
landing_page is not null --nullが入っていない行のみを表示
group by
landing_page --ランディングページでグルーピングする
order by
entrances desc --流入回数の降順に並び替え
```

実行結果



実行 | 展開 | 保存 | 共有 | スケジュール | このクエリを実行すると、4.94 MB が処理されます。

```
with predata as (
select
    user_pseudo_id,  --ユーザーのCookieIDを取得
    (select value.int_value from unnest(event_params) where event_name = 'page_view' and key = 'ga_session_id') as session_id,--ユーザーのセッションIDを取得
    (select value.string_value from unnest(event_params) where key = 'page_location') as page,--ページのURLを取得
    case when (select value.int_value from unnest(event_params) where key = 'entrances') = 1 then true else false end as landing_page  --entrancesのイベントパラメータに「1」が入っている場合にランディングページとして定義
from
`ha-ga4.analytics_227084301.events_*`     -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20220201' and '20220205'   -- 日付の指定
and  event_name = 'page_view')   -- イベント名を指定

select
    case when landing_page is true then page else null end as landing_page, --ランディングページがtrueの場合はページのURLを取得、無ければnullとする
    count(distinct concat(user_pseudo_id,session_id)) as entrances  --CookieIDとセッションIDをつなげユニーク数をカウントし、流入回数とする
from
    predata--上記42つのデータをpredataのクエリ結果から取得

group by
    landing_page --ランディングページでグルーピングする
having
    landing_page is not null --nullが入っていない行のみを表示
order by
    entrances desc  --流入回数の降順に並び替え
```

ユーザー補助機能のオプションを表示するには、Alt+F1 キーを押します

クエリ結果　結果を保存　データを探索

ジョブ情報　結果　JSON　実行の詳細

行	landing_page	entrances
1	https://happyanalytics.co.jp/	158
2	https://go.happyanalytics.co.jp/ga4	94
3	https://analytics.hatenadiary.com/	67
4	https://happyanalytics.co.jp/books/	55
5	https://happyanalytics.co.jp/ceo_message/	50
6	https://analytics.hatenadiary.com/entry/2021/12/03/091157	42
7	https://happyanalytics.co.jp/business_article/	33

28：ページごとの流入回数、離脱数、直帰数を取得

```
with predata as (
select
    event_timestamp,
    user_pseudo_id,  --ユーザーのCookieIDを取得
```

```
    (select value.int_value from unnest(event_params) where event_name =
'page_view' and key = 'ga_session_id') as session_id,--ユーザーのセッションIDを取得
    (SELECT value.string_value FROM UNNEST(event_params) WHERE key =
'page_location') AS page_location  --ページURLを取得

from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの
部分で日付を指定する
where
_table_suffix between '20220201' and '20220205'   -- 日付の指定
and  event_name = 'page_view'),   -- イベント名を指定

predata2 as (
  SELECT
    page_location,  -- ページURLを取得
    case
when row_number() over(partition by session_id order by event_timestamp) = 1 then 1
else 0 end as entrance,
    case
      when row_number() over(partition by user_pseudo_id, session_id order by
event_timestamp desc) = 1 then 1 else 0
    end as exit,  --該当セッションIDで一番最後のpage_viewのタイムスタンプの場合は離脱ページと
して１を割り当てる
    case
      when count(1) over(partition by session_id) = 1 then 1 else 0
    end as bounce  --該当セッションIDでpage_viewイベントが１つしか存在しない場合は直帰として、
1を割り当てる。GA4の直帰（非エンゲージメント）とは違うでの注意
from predata)

select
  page_location as page_location,  --ページURLを取得
  sum (entrance) as landing,  --ランディングページを合計
  sum (exit) as exit,  --離脱ページを合計
  sum(bounce) as bounce    --直帰ページを合計
from
predata2 --predata2からデータを取得
group by  page_location --URLでグルーピング
order by landing desc --流入回数降順で並び替え
```

実行結果

実行　展開　保存　共有　スケジュール　このクエリを実行すると、4.37 MB が処理されます。

```
with predata as (
select
    event_timestamp,
    user_pseudo_id,  --ユーザーのCookieIDを取得
    (select value.int_value from unnest(event_params) where event_name = 'page_view' and key = 'ga_session_id') as session_id,--ユーザーのセッションIDを取得
    (SELECT value.string_value FROM UNNEST(event_params) WHERE key = 'page_location') AS page_location,  --ページURLを取得
    case when (select value.int_value from unnest(event_params) where key = 'entrances') = 1 then 1 else 0 end as landing_page  --entrancesのイベントパラメータに「1」が入っている場合にランディングページとして定義

from
`ha-ga4.analytics_227084301.events_*`     -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20220201' and '20220205'   -- 日付の指定
and  event_name = 'page_view'),   -- イベント名を指定

predata2 as (
  SELECT
    page_location,  -- ページURLを取得
    landing_page,  -- ページURLを取得
    case
      when row_number() over(partition by  session_id order by  event_timestamp desc) = 1 then 1 else 0
    end as is_exit,  --該当セッションIDで一番最後のpage_viewのタイムスタンプの場合は離脱ページとして１を割り当てる
    case
      when count(1) over(partition by session_id) = 1 then 1 else 0
    end as is_bounce  --該当セッションIDでpage_viewイベントが１つしか存在しない場合は直帰として、1を割り当てる
from predata)

select
  predata2.page_location as page_location,  --ページURLを取得
  sum (predata2.landing_page) as landing,  --ランディングページを合計
  sum (is_exit) as exit,  --離脱ページを合計
  sum(is_bounce) as bounce    --直帰ページを合計
from
predata2 --predata2からデータを取得
group by  page_location --URLでグルーピング
order by landing desc --流入回数降順で並び替え
```

ユーザー補助機能のオプションを表示するには、Alt+F1 キーを押します

クエリ結果　結果を保存　データを探索

ジョブ情報　結果　JSON　実行の詳細

行	page_location	landing	exit	bounce
1	https://happyanalytics.co.jp/	158	109	90
2	https://go.happyanalytics.co.jp/ga4	94	110	67
3	https://analytics.hatenadiary.com/	67	59	58
4	https://happyanalytics.co.jp/books/	55	57	53

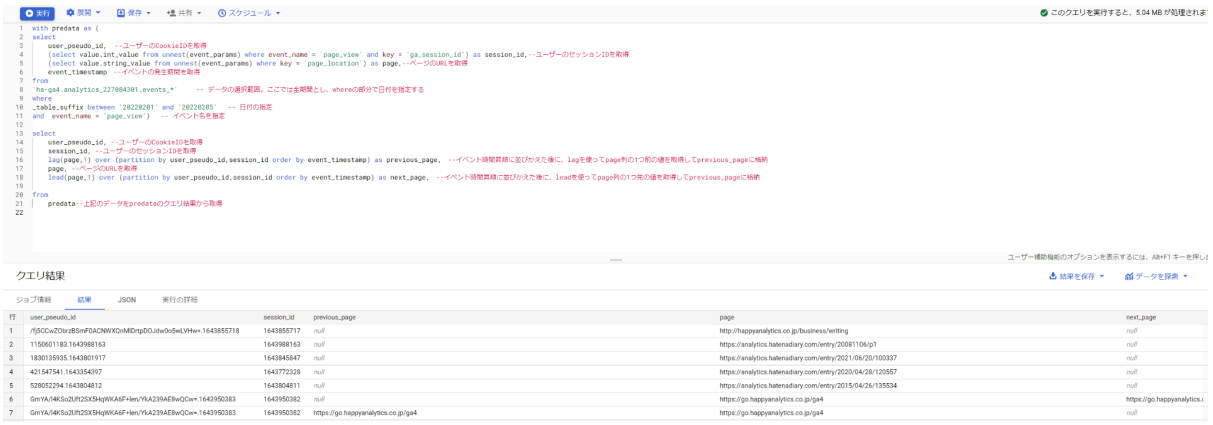
※注：ここで取得している直帰率は1ページだけ見て離脱したセッション数を表示しています。
GA4内での直帰の定義は非エンゲージメントとなるため、クエリ結果とは数値があいません。

29：閲覧ページの1つ前と1つ後をまとめて取得

```
with predata as (
select
    user_pseudo_id,  --ユーザーのCookieIDを取得
    (select value.int_value from unnest(event_params) where event_name = 'page_view' and key = 'ga_session_id') as session_id,--ユーザーのセッションIDを取得
    (select value.string_value from unnest(event_params) where key = 'page_location') as page,--ページのURLを取得
    event_timestamp  --イベントの発生時間を取得
from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20220201' and '20220205'   -- 日付の指定
```

```
and  event_name = 'page_view')   -- イベント名を指定

select
    user_pseudo_id, --ユーザーのCookieIDを取得
    session_id, --ユーザーのセッションIDを取得
    lag(page,1) over (partition by user_pseudo_id,session_id order by
event_timestamp) as previous_page,  --イベント時間昇順に並びかえた後に、lagを使ってpage列の
1つ前の値を取得してprevious_pageに格納
    page, --ページのURLを取得
    lead(page,1) over (partition by user_pseudo_id,session_id order by
event_timestamp) as next_page,  --イベント時間昇順に並びかえた後に、leadを使ってpage列の1つ
先の値を取得してprevious_pageに格納

from
    predata--上記のデータをpredataのクエリ結果から取得
```

実行結果 previous_pageがnullの場合はランディングページ、next_pageがnullの場合は離脱

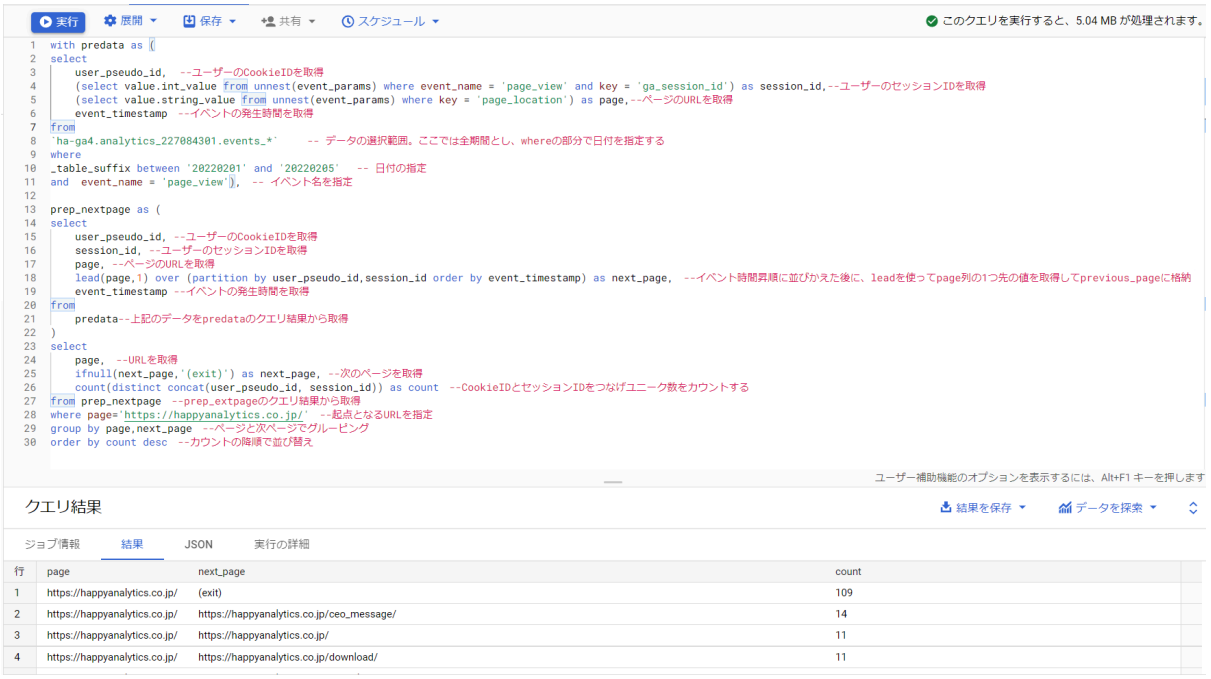


30：指定ページの1つ次に見たページを取得

```
with predata as (
select
    user_pseudo_id,  --ユーザーのCookieIDを取得
    (select value.int_value from unnest(event_params) where event_name =
'page_view' and key = 'ga_session_id') as session_id,--ユーザーのセッションIDを取得
    (select value.string_value from unnest(event_params) where key =
'page_location') as page,--ページのURLを取得
    event_timestamp  --イベントの発生時間を取得
from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの
部分で日付を指定する
where
_table_suffix between '20220201' and '20220205'   -- 日付の指定
and  event_name = 'page_view'),  -- イベント名を指定

prep_nextpage as (
select
```

```
    user_pseudo_id, --ユーザーのCookieIDを取得
    session_id, --ユーザーのセッションIDを取得
    page, --ページのURLを取得
    lead(page,1) over (partition by user_pseudo_id,session_id order by
event_timestamp) as next_page, --イベント時間昇順に並びかえた後に、leadを使ってpage列の1つ
先の値を取得してnext_pageに格納
    event_timestamp --イベントの発生時間を取得
from
    predata--上記のデータをpredataのクエリ結果から取得
)
select
    page, --URLを取得
    ifnull(next_page,'(exit)') as next_page, --次のページを取得
    count(distinct concat(user_pseudo_id,"-", session_id)) as count --CookieIDと
セッションIDをつなげユニーク数をカウントする
from prep_nextpage --prep_nextpageのクエリ結果から取得
where page='https://happyanalytics.co.jp/' --起点となるURLを指定
group by page,next_page --ページと次ページでグルーピング
order by count desc --カウントの降順で並び替え
```

実行結果

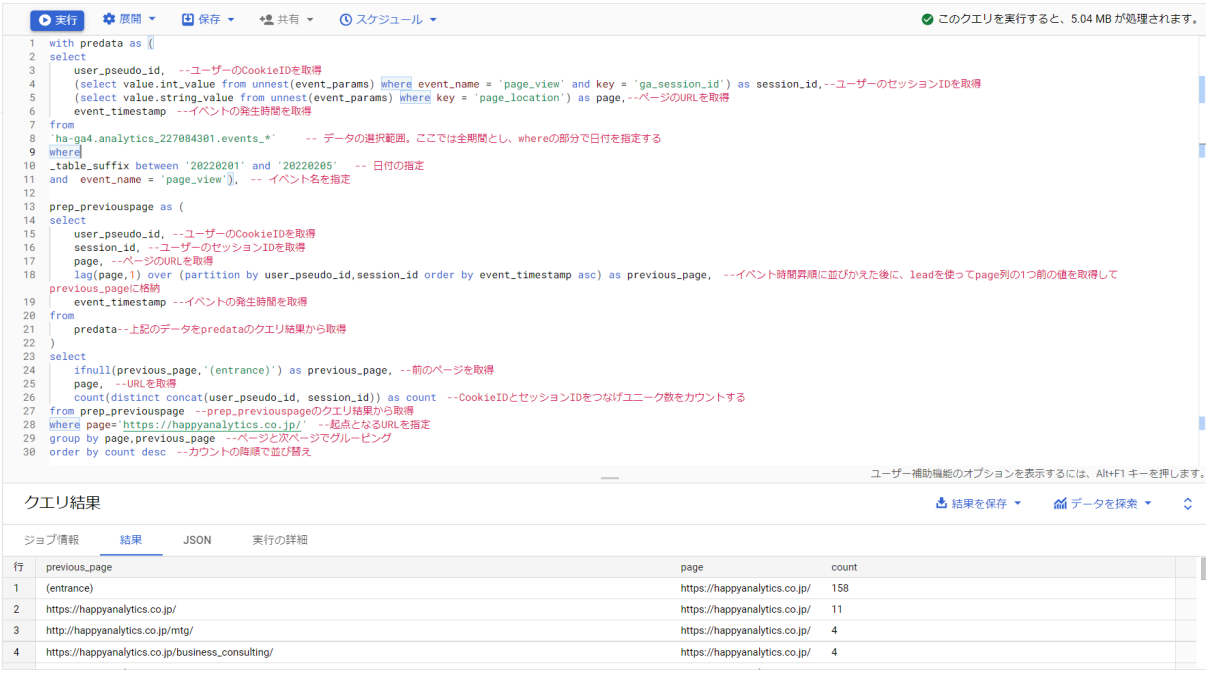


31：指定ページの1つ前に見たページを取得

```
with predata as (
select
    user_pseudo_id, --ユーザーのCookieIDを取得
    (select value.int_value from unnest(event_params) where event_name =
'page_view' and key = 'ga_session_id') as session_id,--ユーザーのセッションIDを取得
    (select value.string_value from unnest(event_params) where key =
'page_location') as page,--ページのURLを取得
```

```
    event_timestamp  --イベントの発生時間を取得
from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20220201' and '20220205'   -- 日付の指定
and  event_name = 'page_view'),  -- イベント名を指定

prep_previouspage as (
select
    user_pseudo_id, --ユーザーのCookieIDを取得
    session_id, --ユーザーのセッションIDを取得
    page, --ページのURLを取得
    lag(page,1) over (partition by user_pseudo_id,session_id order by event_timestamp asc) as previous_page,  --イベント時間昇順に並びかえた後に、leadを使ってpage列の1つ前の値を取得してprevious_pageに格納
    event_timestamp --イベントの発生時間を取得
from
    predata--上記のデータをpredataのクエリ結果から取得
)
select
    ifnull(previous_page,'(entrance)') as previous_page, --前のページを取得
    page,  --URLを取得
    count(distinct concat(user_pseudo_id,"-", session_id)) as count  --CookieIDとセッションIDをつなげユニーク数をカウントする
from prep_previouspage  --prep_previouspageのクエリ結果から取得
where page='https://happyanalytics.co.jp/'  --起点となるURLを指定
group by page,previous_page  --ページと前ページでグルーピング
order by count desc  --カウントの降順で並び替え
```

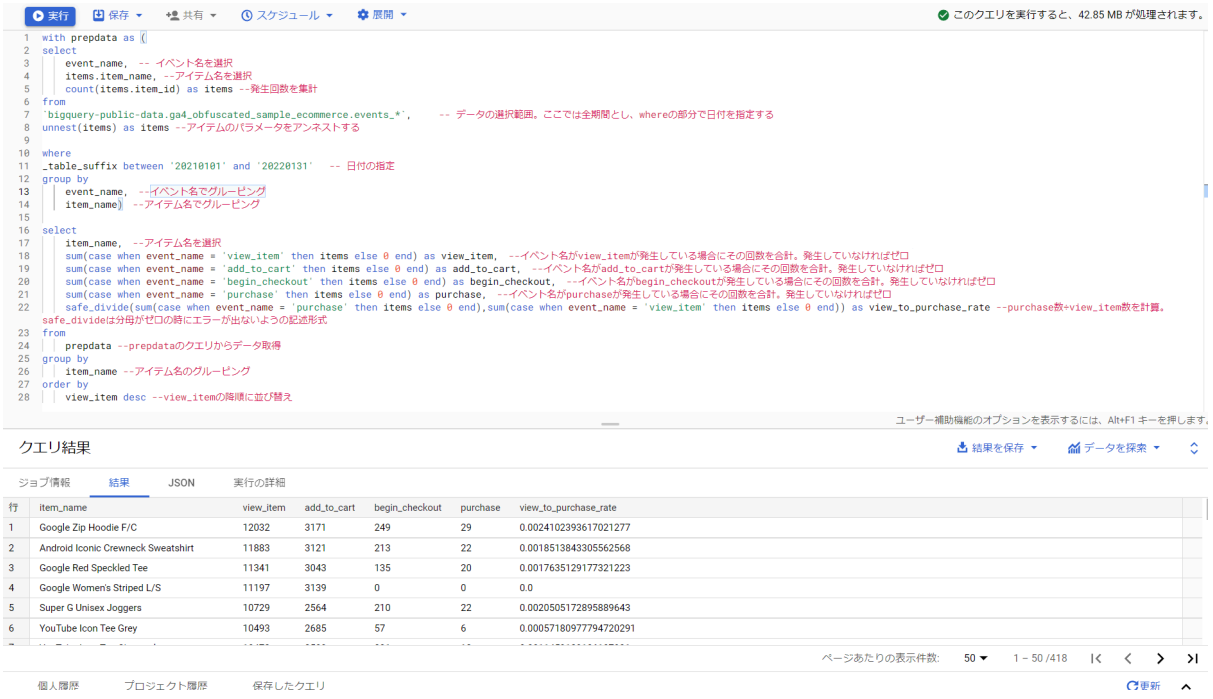
実行結果



32：アイテムごとのeコマースイベント発生回数とCVR

```
with prepdata as (
select
    event_name,  -- イベント名を選択
    items.item_name, --アイテム名を選択
    count(items.item_id) as items --発生回数を集計
from
`bigquery-public-data.ga4_obfuscated_sample_ecommerce.events_*`,      -- データの選択範囲。ここではwhereの部分で日付を指定する
unnest(items) as items --アイテムのパラメータをアンネストする

where
_table_suffix between '20210101' and '20220131'   -- 日付の指定
group by
    event_name,  --イベント名でグルーピング
    item_name)  --アイテム名でグルーピング

select
    item_name,  --アイテム名を選択
    sum(case when event_name = 'view_item' then items else 0 end) as view_item,  --イベント名がview_itemが発生している場合にその回数を合計。発生していなければゼロ
    sum(case when event_name = 'add_to_cart' then items else 0 end) as add_to_cart,  --イベント名がadd_to_cartが発生している場合にその回数を合計。発生していなければゼロ
    sum(case when event_name = 'begin_checkout' then items else 0 end) as begin_checkout,  --イベント名がbegin_checkoutが発生している場合にその回数を合計。発生していなければゼロ
    sum(case when event_name = 'purchase' then items else 0 end) as purchase,  --イベント名がpurchaseが発生している場合にその回数を合計。発生していなければゼロ
    safe_divide(sum(case when event_name = 'purchase' then items else 0 end),sum(case when event_name = 'view_item' then items else 0 end)) as view_to_purchase_rate --purchase数÷view_item数を計算。safe_divideは分母がゼロの時にエラーが出ないようの記述形式
from
    prepdata --prepdataのクエリからデータ取得
group by
    item_name --アイテム名のグルーピング
order by
    view_item desc --view_itemの降順に並び替え
```

実行結果

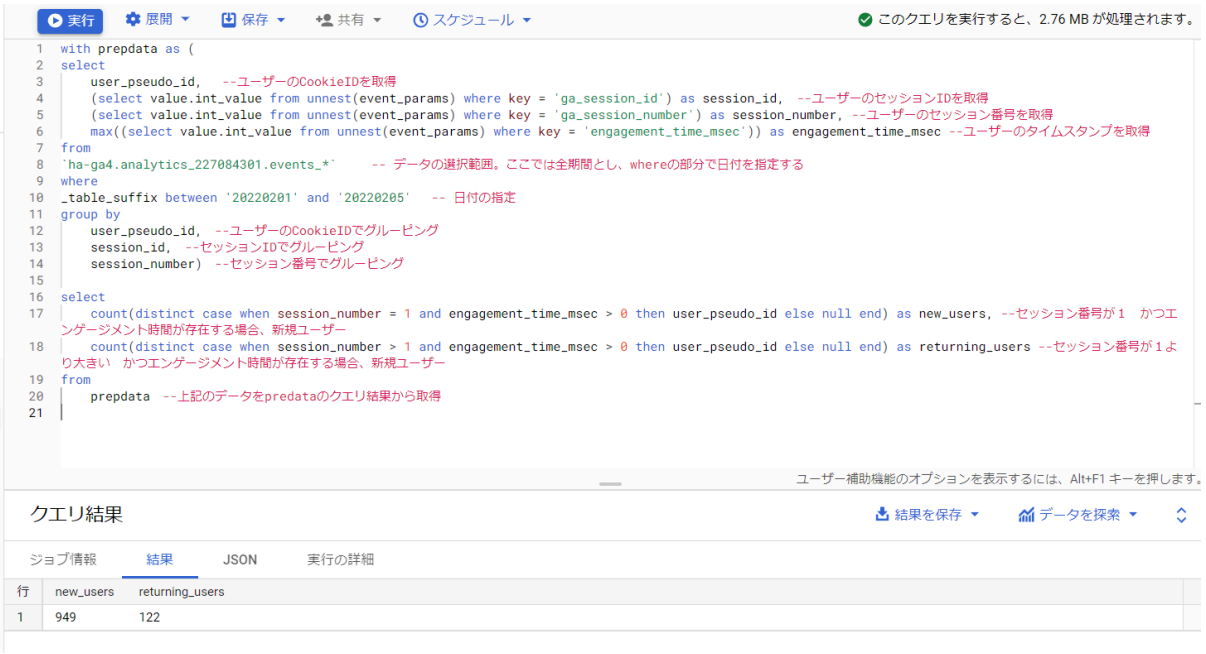


33：新規とリピートユーザーの取得

```
with prepdata as (
select
    user_pseudo_id,   --ユーザーのCookieIDを取得
    (select value.int_value from unnest(event_params) where key = 'ga_session_id')
as session_id,  --ユーザーのセッションIDを取得
    (select value.int_value from unnest(event_params) where key =
'ga_session_number') as session_number, --ユーザーのセッション番号を取得
    max((select value.int_value from unnest(event_params) where key =
'engagement_time_msec')) as engagement_time_msec --ユーザーのタイムスタンプを取得
from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの
部分で日付を指定する
where
_table_suffix between '20220201' and '20220205'   -- 日付の指定

group by
    user_pseudo_id,  --ユーザーのCookieIDでグルーピング
    session_id,  --セッションIDでグルーピング
    session_number)  --セッション番号でグルーピング

select
    count(distinct case when session_number = 1 and engagement_time_msec > 0 then
user_pseudo_id else null end) as new_users, --セッション番号が１　かつエンゲージメント時間
が存在する場合、新規ユーザー
    count(distinct case when session_number > 1 and engagement_time_msec > 0 then
user_pseudo_id else null end) as returning_users --セッション番号が１より大きい　かつエン
ゲージメント時間が存在する場合、リピートユーザー
```

```
from
    prepdata  --上記のデータをpredataのクエリ結果から取得
```

実行結果



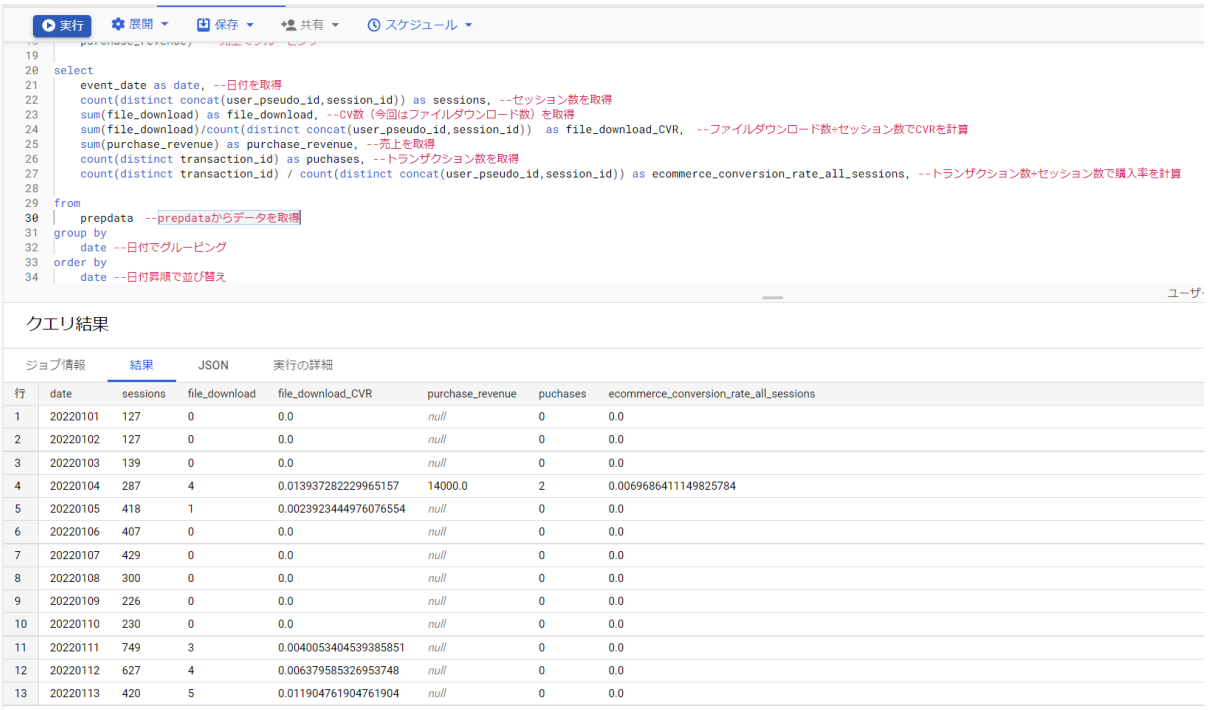
エンゲージメント時間を条件として使わずに新規・リピートユーザーを出したい場合は、タイムスタンプ部分及びengagemnt_time_msecの条件を外してください

34：コンバージョン率や購入率を取得

```
with prepdata as (
select
    user_pseudo_id, --ユーザーのCookieIDを取得
    (select value.int_value from unnest(event_params) where key = 'ga_session_id')
as session_id, --セッションIDを取得
    event_date, --イベントの発生日付を取得
    countif(event_name = 'file_download') as file_download,  --CVを見たいイベント名を指定。ここではfile_download
    ecommerce.transaction_id, --トランザクションIDを取得
    ecommerce.purchase_revenue --売上を取得
from
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
where
_table_suffix between '20220101' and '20220207'   -- 日付の指定
group by
    user_pseudo_id, --ユーザーのCookieIDでグルーピング
    session_id, --セッションIDでグルーピング
    event_date, --日付でグルーピング
    transaction_id, --トランザクションIDでグルーピング
    purchase_revenue)  --売上でグルーピング
```

```
select
    event_date as date, --日付を取得
    count(distinct concat(user_pseudo_id,"-",session_id)) as sessions, --セッション数を取得
    sum(file_download) as file_download, --CV数（今回はファイルダウンロード数）を取得
    sum(file_download)/count(distinct concat(user_pseudo_id,"-",session_id))  as file_download_CVR,  --ファイルダウンロード数÷セッション数でCVRを計算
    sum(purchase_revenue) as purchase_revenue, --売上を取得
    count(distinct transaction_id) as puchases, --トランザクション数を取得
    count(distinct transaction_id) / count(distinct concat(user_pseudo_id,"-",session_id)) as ecommerce_conversion_rate_all_sessions, --トランザクション数÷セッション数で購入率を計算

from
    prepdata  --prepdataからデータを取得
group by
    date --日付でグルーピング
order by
    date --日付昇順で並び替え
```

実行結果



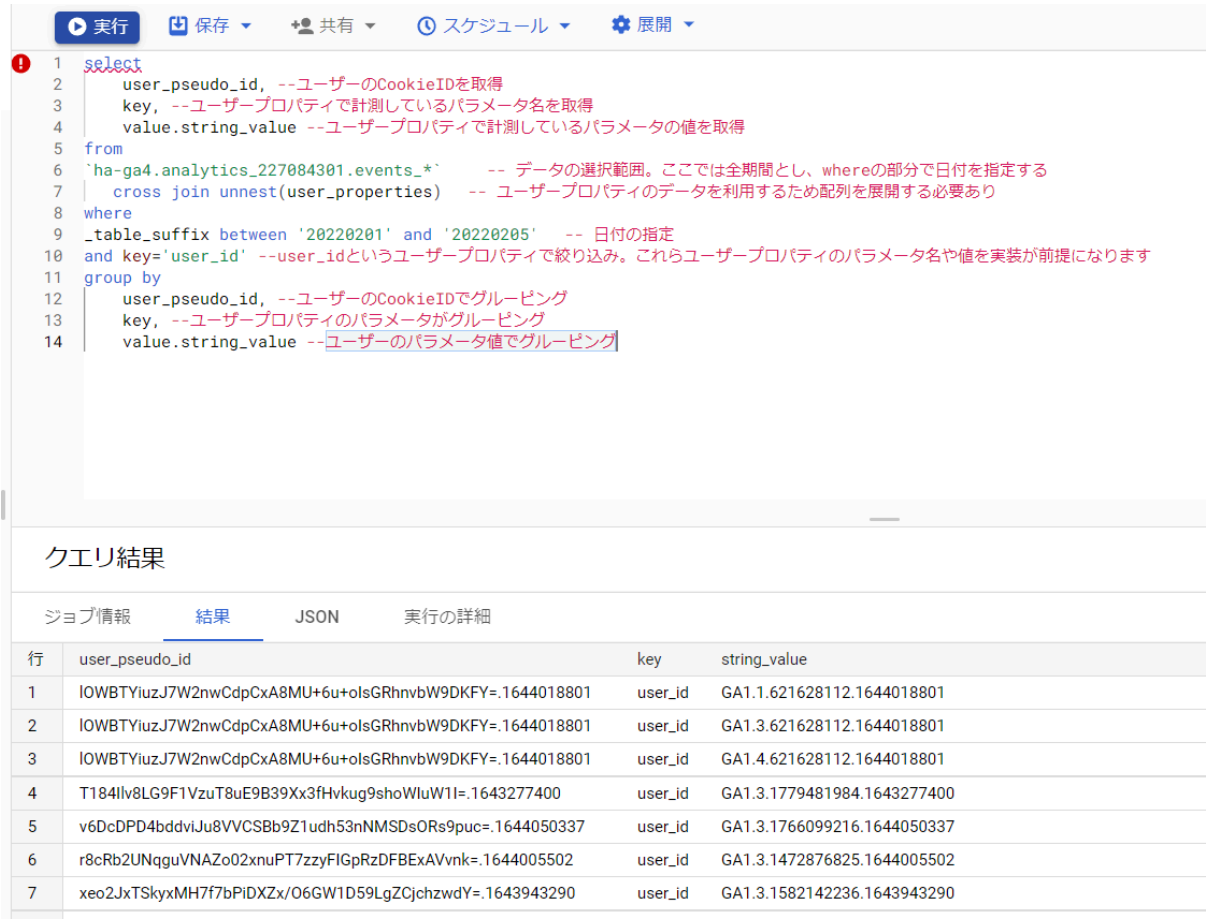
行	date	sessions	file_download	file_download_CVR	purchase_revenue	puchases	ecommerce_conversion_rate_all_sessions
1	20220101	127	0	0.0	null	0	0.0
2	20220102	127	0	0.0	null	0	0.0
3	20220103	139	0	0.0	null	0	0.0
4	20220104	287	4	0.013937282229965157	14000.0	2	0.0069684111149825784
5	20220105	418	1	0.0023923444976076554	null	0	0.0
6	20220106	407	0	0.0	null	0	0.0
7	20220107	429	0	0.0	null	0	0.0
8	20220108	300	0	0.0	null	0	0.0
9	20220109	226	0	0.0	null	0	0.0
10	20220110	230	0	0.0	null	0	0.0
11	20220111	749	3	0.0040053404539385851	null	0	0.0
12	20220112	627	4	0.006379585326953748	null	0	0.0
13	20220113	420	5	0.011904761904761904	null	0	0.0

35：ユーザープロパティのデータを取得

```
select
    user_pseudo_id, --ユーザーのCookieIDを取得
    key, --ユーザープロパティで計測しているパラメータ名を取得
    value.string_value --ユーザープロパティで計測しているパラメータの値を取得
from
```

```
`ha-ga4.analytics_227084301.events_*`      -- データの選択範囲。ここでは全期間とし、whereの部分で日付を指定する
   cross join unnest(user_properties)    -- ユーザープロパティのデータを利用するため配列を展開する必要あり
where
_table_suffix between '20220201' and '20220205'    -- 日付の指定
and key='user_id' --user_idというユーザープロパティで絞り込み。これらユーザープロパティのパラメータ名や値を実装が前提になります
group by
    user_pseudo_id, --ユーザーのCookieIDでグルーピング
    key, --ユーザープロパティのパラメータがグルーピング
    value.string_value --ユーザーのパラメータ値でグルーピング
```

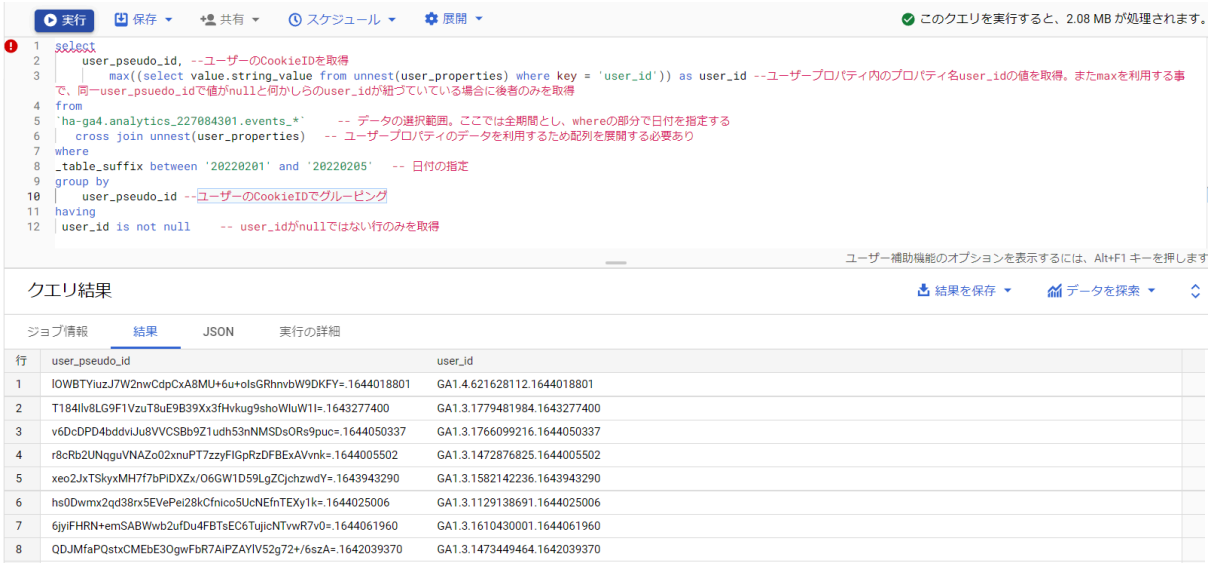
実行結果



36：ユーザープロパティの値をユーザーごとにグルーピング

```
select distinct
user_pseudo_id, --ユーザーのCookieIDを取得
(select value.string_value from unnest(user_properties) where key = 'user_id') as
user_id --ユーザープロパティ内のプロパティ名user_idの値を取得。またmaxを利用する事で、同一
user_psuedo_idで値がnullと何かしらのuser_idが紐づていている場合に後者のみを取得
from
`ha-ga4.analytics_227084301.events_*` -- データの選択範囲。ここでは全期間とし、whereの部分
で日付を指定する
cross join unnest(user_properties) -- ユーザープロパティのデータを利用するため配列を展開する
必要あり
where
_table_suffix between '20220201' and '20220205' -- 日付の指定
and user_id is not null -- user_idがnullではない行のみを取得
```

実行結果

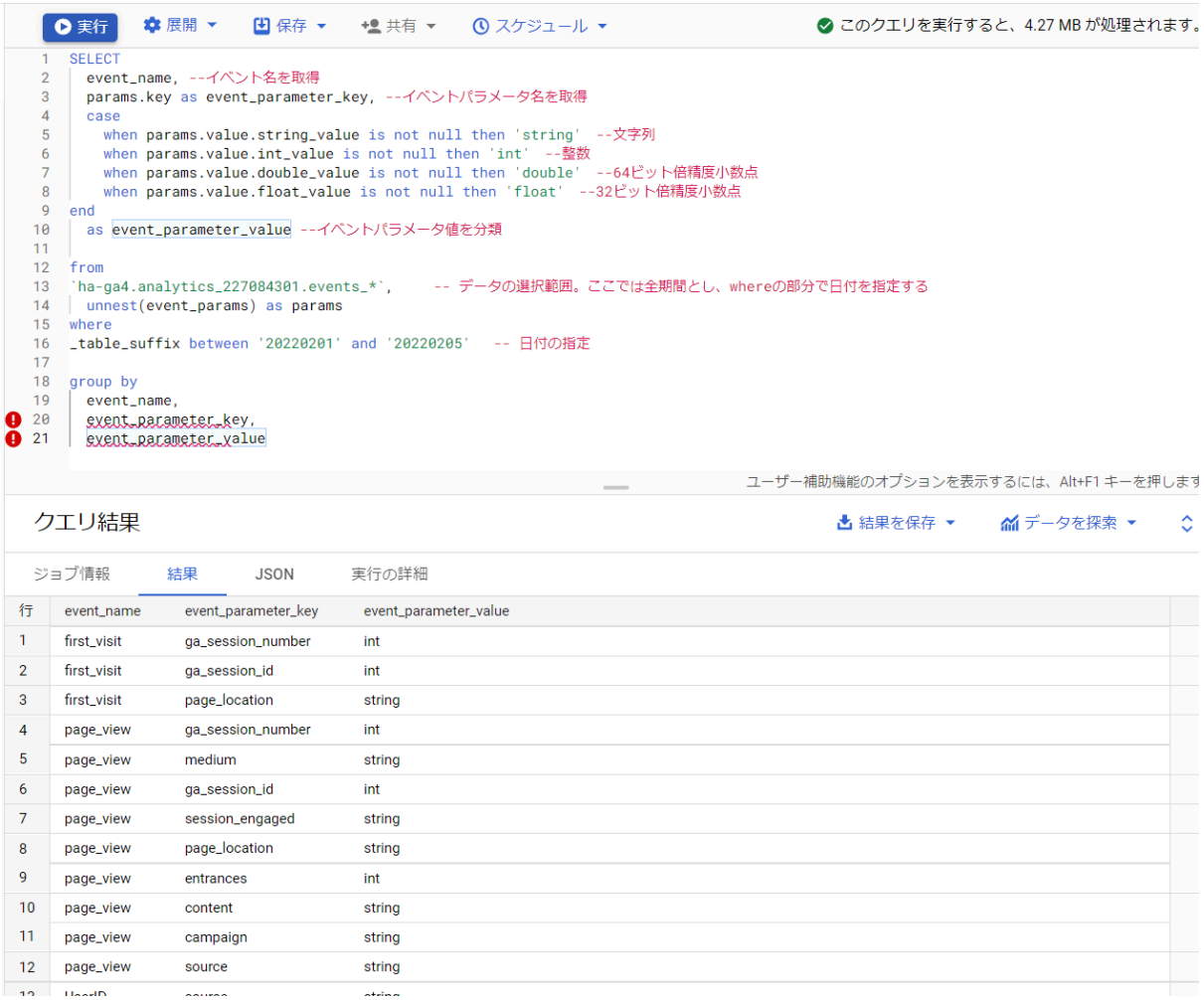


37：取得イベント名とパラメータ名の一覧表示

```
SELECT
  event_name, --イベント名を取得
  params.key as event_parameter_key, --イベントパラメータ名を取得
  case
    when params.value.string_value is not null then 'string'  --文字列
    when params.value.int_value is not null then 'int'  --整数
    when params.value.double_value is not null then 'double'  --64ビット倍精度小数点
    when params.value.float_value is not null then 'float'  --32ビット倍精度小数点
end
  as event_parameter_value --イベントパラメータ値を分類

from
```

```
`ha-ga4.analytics_227084301.events_*`,      -- データの選択範囲。ここでは全期間とし、where の部分で日付を指定する
  unnest(event_params) as params
where
_table_suffix between '20220201' and '20220205'    -- 日付の指定

group by
  event_name,
  event_parameter_key,
  event_parameter_value
```

実行結果



行	event_name	event_parameter_key	event_parameter_value
1	first_visit	ga_session_number	int
2	first_visit	ga_session_id	int
3	first_visit	page_location	string
4	page_view	ga_session_number	int
5	page_view	medium	string
6	page_view	ga_session_id	int
7	page_view	session_engaged	string
8	page_view	page_location	string
9	page_view	entrances	int
10	page_view	content	string
11	page_view	campaign	string
12	page_view	source	string