PORTFOLIO · SQL · BIGQUERY



SQL Sample
YouTube *Trending Analysis*

What makes a YouTube video popular?
Six problems, from data-quality checks to engagement metrics, all written and run in BigQuery.

PREPARED BY

Divya Nagpal

DATASET

Trending YouTube Video Statistics and Comments (Kaggle)

DATE

September, 2026

ENGINE

Google BigQuery

CONTACT

divyanagpal1308@gmail.com

Project *Overview*

Problem Statement: What makes a YouTube video popular?

The data covers videos that made YouTube's US Trending list each day in 2017. It comes in two files, one with video stats and one with comments, connected through video_id. I start by checking the grain and the join keys, then look at channels, categories, conversation and how long videos stay on Trending.

2,364
unique trending videos

691,374
comments joined

43
trending days in 2017

6
SQL problems solved

Data *Sources*

01 Dataset	Trending YouTube Video Statistics and Comments, a public dataset on Kaggle (kaggle.com/datasets/datasnaek/youtube).
02 Tables	USvideos.csv is loaded as youtube_trending.us_videos and UScomments.csv as youtube_trending.us_comments. I also built a category lookup (youtube_trending.categories) and a one-row-per-video table (youtube_trending.videos_dedup)
03 Environment	Google BigQuery, run under my own account (divyanagpal1308@gmail.com).
04 Limitations	It's about six weeks of 2017 US data, and YouTube retired the Trending page in July 2025, so today's patterns may differ. Comments are a sample of roughly 305 per video, and every result describes trending videos only.

FIELD REFERENCE

Video file

us_videos · one row per video per trending day

video_id	join key shared with the comments file
title	
channel_title	
category_id	numeric, mapped via the US category list
tags	pipe-separated, [none] if empty
views	
likes	
dislikes	
comment_total	comment count on the video
thumbnail_link	
date	formatted as [day].[month]

Comments file

us_comments · about 305 sampled comments per video

video_id	join key shared with the video file
comment_text	
likes	likes on the comment
replies	replies to the comment

Data Quality Checks

Avoid *double-counting*

A video shows up once for every day it trended, so a raw SUM would count the same video several times. Before measuring anything, I check what one row represents and build a table with exactly one row per video.

1a Check the grain: how many rows, videos and days?

QUERY

```sql
SELECT
  COUNT(*) AS total_rows,
  COUNT(DISTINCT video_id) AS unique_videos,
  COUNT(DISTINCT date) AS trending_days
FROM youtube_trending.us_videos;
```

OUTPUT

Query results — Open in

Job information | Results | Visualization | JSON

Row	total_rows	unique_videos	trending_days
1	7998	2364	43

1b Keep one row per video, using its latest snapshot

Counts only grow over time, so the row with the most views is always the most recent day.

QUERY

```sql
CREATE OR REPLACE TABLE youtube_trending.videos_dedup AS
SELECT * EXCEPT (rn)
FROM (
  SELECT
    *,
    ROW_NUMBER() OVER (PARTITION BY video_id ORDER BY views DESC) AS rn
  FROM youtube_trending.us_videos
)
WHERE rn = 1;
```

1c Confirm the new table has exactly one row per video

QUERY

```sql
SELECT
  COUNT(*) AS total_rows,
  COUNT(DISTINCT video_id) AS unique_videos
FROM youtube_trending.videos_dedup;
```

OUTPUT

Query results — Open in

Job information | Results | Visualization | JSON

Row	total_rows	unique_videos
1	2364	2364

SOLUTION

7,998
raw rows across 43 trending days

2,364
unique videos, one row each in videos_dedup

3.4x
rows per video, the overcount this removes

The raw table has 7,998 rows but only 2,364 unique videos across 43 days, so each video appears about 3.4 times. Summing it as-is would inflate views and comments by roughly 3.4x. The new videos_dedup table has exactly 2,364 rows, one per video, and it powers everything that follows.

Problem 2

Do a few big channels *dominate?*

If Trending were a closed club, a handful of channels would take most of the spots. I rank channels by trending videos, then measure how much of the list the top 10 actually hold.

2a Rank the top 10 channels by trending videos

HAVING drops channels with fewer than five trending videos.

QUERY

```sql
SELECT
  channel_title,
  COUNT(*) AS trending_videos,
  SUM(views) AS total_views
FROM youtube_trending.videos_dedup
GROUP BY channel_title
HAVING COUNT(*) >= 5
ORDER BY trending_videos DESC
LIMIT 10;
```

OUTPUT

Query results — Open in — Job information | Results | Visualization | JSON

Row	channel_title	trending_videos	total_views
1	The Tonight Show Starring Jim...	21	20043002
2	Vox	20	9964792
3	Jimmy Kimmel Live	18	35707763
4	Refinery29	17	363018
5	TheEllenShow	16	15623489
6	NFL	15	11510806
7	Netflix	14	21141783
8	The Late Show with Stephen Co...	13	14546137
9	Late Night with Seth Meyers	13	12917774
10	ESPN	13	6369437

2b Measure the top 10's share of all trending videos

QUERY

```sql
WITH channel_counts AS (
  SELECT channel_title, COUNT(*) AS trending_videos
  FROM youtube_trending.videos_dedup
  GROUP BY channel_title
),
top10 AS (
  SELECT * FROM channel_counts
  ORDER BY trending_videos DESC
  LIMIT 10
)
SELECT
  (SELECT SUM(trending_videos) FROM top10) AS top10_videos,
  (SELECT SUM(trending_videos) FROM channel_counts) AS all_videos,
  ROUND(100 * (SELECT SUM(trending_videos) FROM top10)
  / (SELECT SUM(trending_videos) FROM channel_counts), 1) AS top10_share_pct;
```

OUTPUT

Query results — Open in — Job information | Results | Visualization | JSON

Row	top10_videos	all_videos	top10_share_pct
1	160	2364	6.8

SOLUTION

6.8%
of trending videos held by the top 10 channels

160
of 2,364 videos come from those 10 channels

17
Refinery29 videos trended with just 363K views combined

Not really. The top 10 channels hold just 6.8% of trending videos (160 of 2,364). Five of them are late-night or talk shows. Refinery29 trended 17 times with only 363K views combined, which shows Trending isn't simply a views leaderboard.

Problem 3

Which categories drive *engagement?*

category_id is just a number, so I first map it to category names. Then I compare categories two ways: average views and engagement rate.

Metric Engagement rate = (likes + dislikes + comments) / views × 1,000. SAFE_DIVIDE keeps zero-view rows from throwing errors.

3a Build a lookup table for category names

Built from the US category list, since category IDs vary by region.

QUERY

```sql
CREATE OR REPLACE TABLE youtube_trending.categories AS
SELECT * FROM UNNEST([
  STRUCT(1 AS category_id, 'Film & Animation' AS category_name),
  (2, 'Autos & Vehicles'),
  (10, 'Music'),
  (15, 'Pets & Animals'),
  (17, 'Sports'),
  (19, 'Travel & Events'),
  (20, 'Gaming'),
  (22, 'People & Blogs'),
  (23, 'Comedy'),
  (24, 'Entertainment'),
  (25, 'News & Politics'),
  (26, 'Howto & Style'),
  (27, 'Education'),
  (28, 'Science & Technology'),
  (29, 'Nonprofits & Activism'),
  (43, 'Shows')]);
```

3b Compare views and engagement by category

The LEFT JOIN keeps every video, even if its category has no match.

QUERY

```sql
SELECT
  COALESCE(c.category_name, 'Other') AS category,
  COUNT(*) AS videos,
  ROUND(AVG(v.views)) AS avg_views,
  ROUND(1000 * SAFE_DIVIDE(
    SUM(v.likes + v.dislikes + v.comment_total),
    SUM(v.views)), 1) AS engagement_per_1k_views
FROM youtube_trending.videos_dedup AS v
LEFT JOIN youtube_trending.categories AS c
  ON v.category_id = c.category_id
GROUP BY category
HAVING COUNT(*) >= 20
ORDER BY engagement_per_1k_views DESC;
```

OUTPUT

Job information | Results | Visualization | JSON | Execution details

Row	category	videos	avg_views	engagement_per...
1	Howto & Style	262	556027.0	59.6
2	Music	335	1444205.0	53.1
3	Comedy	214	1317201.0	48.0
4	People & Blogs	256	892992.0	44.2
5	Education	94	550585.0	35.5
6	Gaming	28	777213.0	33.2
7	Entertainment	477	1059428.0	32.3
8	Science & Technology	151	861564.0	28.4
9	Pets & Animals	38	633906.0	28.3
10	Film & Animation	101	966547.0	26.9
11	Autos & Vehicles	37	622728.0	22.9
12	Sports	149	618075.0	21.7
13	News & Politics	198	533577.0	12.6

SOLUTION

59.6
interactions per 1K views for Howto & Style, the leader

4.7x
Howto & Style vs News & Politics (12.6)

1.44M
average views for Music, the reach leader

Howto & Style leads with 59.6 interactions per 1,000 views, 4.7x News & Politics (12.6). Views tell a different story: Music has the highest average views (1.44M), while Howto & Style sits near the bottom (556K). Which content looks most popular depends on the metric you pick.

Problem 4

Which categories spark *conversation?*

Likes and views show reaction, not discussion. To measure real conversation, I bring in the comments file and compare replies per comment by category.

4a Check how well the two files match before joining

QUERY

```sql
SELECT
  COUNT(*) AS total_comments,
  COUNT(DISTINCT video_id) AS videos_with_comments,
  COUNTIF(video_id IN (SELECT video_id FROM youtube_trending.videos_dedup))
    AS comments_matching_videos
FROM youtube_trending.us_comments;
```

OUTPUT

Job information | Results | Visualization | JSON

Row	total_comments	videos_with_com...	comments_matc...
1	691374	2266	691374

4b Pre-aggregate comments, then join and compare

Rolling comments up to one row per video first stops each video from being duplicated about 305 times.

QUERY

```sql
WITH comment_stats AS (
  SELECT
    video_id,
    AVG(likes) AS avg_comment_likes,
    SAFE_DIVIDE(SUM(replies), COUNT(*)) AS replies_per_comment
  FROM youtube_trending.us_comments
  GROUP BY video_id
)
SELECT
  COALESCE(c.category_name, 'Other') AS category,
  COUNT(*) AS videos,
  ROUND(AVG(cs.avg_comment_likes), 1) AS avg_likes_per_comment,
  ROUND(AVG(cs.replies_per_comment), 2) AS avg_replies_per_comment
FROM youtube_trending.videos_dedup AS v
JOIN comment_stats AS cs
  ON v.video_id = cs.video_id
LEFT JOIN youtube_trending.categories AS c
  ON v.category_id = c.category_id
GROUP BY category
HAVING COUNT(*) >= 20
ORDER BY avg_replies_per_comment DESC;
```

OUTPUT

Job information | Results | Visualization | JSON | Execution details

Row	category	videos	avg_likes_per_co...	avg_replies_per_c...
1	Education	89	4.3	1.4
2	Autos & Vehicles	35	1.5	0.81
3	People & Blogs	245	7.3	0.73
4	Sports	139	2.5	0.7
5	Science & Technology	144	4.2	0.65
6	News & Politics	175	2.1	0.59
7	Film & Animation	97	14.6	0.56
8	Howto & Style	260	8.3	0.54
9	Comedy	211	8.1	0.47
10	Entertainment	458	4.3	0.4
11	Gaming	27	2.2	0.35
12	Pets & Animals	37	1.8	0.32
13	Music	329	1.8	0.16

SOLUTION

100%
of 691,374 comments matched a video

96%
of videos have comments (2,266 of 2,364)

9x
more replies per comment for Education than Music

Every one of the 691,374 comments matched a video, and 2,266 of 2,364 videos (96%) have comments. Conversation ranks categories differently again. Music leads on reach but sparks the least discussion (0.16 replies per comment), while Education sparks about 9x more (1.40). News & Politics had the lowest engagement rate in Problem 3, yet sits mid-table here: fewer people react, but the ones who do tend to reply.

Problem 5

Does engagement keep a video *trending?*

If engagement drives staying power, videos that trend longer should show higher engagement. I count each video's days on Trending, group them into buckets and compare.

5 **Bucket videos by days on Trending**

This one uses the raw daily table on purpose, since every day is needed to count days on Trending.

QUERY

```sql
WITH per_video AS (
  SELECT
    video_id,
    COUNT(DISTINCT date) AS days_trending,
    MAX(views) AS views,
    MAX(likes + dislikes + comment_total) AS interactions
  FROM youtube_trending.us_videos
  GROUP BY video_id
)
SELECT
  CASE
    WHEN days_trending = 1 THEN '1 day'
    WHEN days_trending <= 3 THEN '2-3 days'
    WHEN days_trending <= 6 THEN '4-6 days'
    ELSE '7+ days'
  END AS trending_bucket,
  COUNT(*) AS videos,
  ROUND(1000 * SAFE_DIVIDE(SUM(interactions), SUM(views)), 1)
    AS engagement_per_1k_views
FROM per_video
GROUP BY trending_bucket
ORDER BY trending_bucket;
```

OUTPUT

Job information | Results | Visualization | JSON | Exe

Row	trending_bucket	videos	engagement_per...
1	1 day	356	36.4
2	2-3 days	860	40.4
3	4-6 days	1115	41.2
4	7+ days	33	34.6

SOLUTION

34.6 to 41.2
interactions per 1K views across all buckets

84%
of videos trend for 2 to 6 days

33
videos in the 7+ day bucket, too few to read into

No. Engagement barely moves with time on Trending, ranging from 34.6 to 41.2 interactions per 1,000 views. Most videos (84%) trend for 2 to 6 days. The 7+ day bucket has only 33 videos, too few to read much into. The buckets add up to 2,364, so no videos were lost along the way.

Problem 6

Do channels trend *repeatedly?*

Problem 2 showed no small group dominates. The next question is whether repeat creators drive Trending instead, so I measure how many channels trend more than once and how much of the list they supply.

6 Measure repeat channels and their share of videos

QUERY

```sql
WITH channel_videos AS (
  SELECT channel_title, COUNT(*) AS trending_videos
  FROM youtube_trending.videos_dedup
  GROUP BY channel_title
)
SELECT
  COUNT(*) AS channels,
  COUNTIF(trending_videos >= 2) AS repeat_channels,
  ROUND(100 * COUNTIF(trending_videos >= 2) / COUNT(*), 1)
    AS repeat_rate_pct,
  ROUND(100 * SUM(IF(trending_videos >= 2, trending_videos, 0))
    / SUM(trending_videos), 1) AS pct_videos_from_repeat_channels
FROM channel_videos;
```

OUTPUT

Job information | Results | Visualization | JSON | Execution

Row	channels	repeat_channels	repeat_rate_pct	pct_videos_from_...
1	1227	396	32.3	64.8

SOLUTION

32.3%
of channels trended more than once (396 of 1,227)

64.8%
of trending videos come from those repeat channels

6.8%
held by the top 10 channels (Problem 2)

Yes. About a third of channels (32.3%, or 396 of 1,227) trended more than once, and they supplied 64.8% of trending videos. Combined with the top 10's modest 6.8% (Problem 2), Trending runs on a broad base of consistent creators rather than a few giants.

Key *Takeaways*

Popularity on Trending isn't one number. Reach, engagement and conversation each reward different content, and checking the data first kept every metric honest.

01 **Popularity isn't one number.** Problems 3, 4

Music wins on reach, Howto & Style on engagement and Education on conversation. Looking at views alone misses most of the story.

02 **Consistent creators fill Trending.** Problems 2, 6

The top 10 channels hold just 6.8%, while the 32% of channels that trend more than once supply 65% of videos.

03 **Engagement doesn't predict staying power.** Problem 5

It stays nearly flat across time on Trending, so early view growth is the next thing worth testing.

04 **Checking the data first paid off.** Problem 1

Catching the 3.4x double-count up front kept every metric in this analysis accurate.

SQL Toolkit

The techniques behind all six problems, in BigQuery SQL.

01 · Data quality

Trust the grain first

- ROW_NUMBER() dedup
- Grain checks
- COUNT(DISTINCT)
- Key match rate
- Total reconciliation

02 · Joins & aggregation

Join without fan-out

- CTEs (WITH)
- LEFT vs INNER JOIN
- Pre-aggregation
- Scalar subqueries
- GROUP BY · HAVING

03 · Metrics & logic

Define metrics safely

- SAFE_DIVIDE
- COUNTIF
- CASE buckets
- UNNEST(STRUCT)
- COALESCE

- ✓ Grain checked before aggregating Problem 1
- ✓ Join keys verified before joining Problem 4
- ✓ Comments pre-aggregated to stop fan-out Problem 4
- ✓ Division guarded with SAFE_DIVIDE Problems 3 to 5
- ✓ Small groups filtered with HAVING Problems 2 to 4
- ✓ Bucket totals reconciled to 2,364 Problem 5