-- (step 1) Read consumer embeddings

WITH consumer\_embeddings AS (

{CONSUMER\_EMBEDDINGS\_SQL}

),

-- (step 1) Read tweet embeddings

tweet\_embeddings AS (

{TWEET\_EMBEDDINGS\_SQL}

),

-- (step 1) Compute tweet embeddings norms (we will use this to compute cosine sims later)

tweet\_embeddings\_norm AS (

SELECT tweetId, SUM(tweetScore \* tweetScore) AS norm

FROM tweet\_embeddings

GROUP BY tweetId

HAVING norm > 0.0

),

-- (step 2) Get top N clusters for each consumer embedding. N = 25 in prod

consumer\_embeddings\_top\_n\_clusters AS (

SELECT userId, ARRAY\_AGG(STRUCT(clusterId, userScore) ORDER BY userScore DESC LIMIT {TOP\_N\_CLUSTER\_PER\_SOURCE\_EMBEDDING}) AS topClustersWithScores

FROM consumer\_embeddings

GROUP BY userId

),

-- (step 2) Get top M tweets for each cluster id. M = 100 in prod

clusters\_top\_m\_tweets AS (

SELECT clusterId, ARRAY\_AGG(STRUCT(tweetId, tweetScore) ORDER BY tweetScore DESC LIMIT {TOP\_M\_TWEETS\_PER\_CLUSTER}) AS tweets

FROM tweet\_embeddings

GROUP BY clusterId

),

-- (step 3) Join the results, get top M \* N tweets for each user

user\_top\_mn\_tweets AS (

SELECT userId, consumer\_embedding\_cluster\_score\_pairs.userScore AS userScore, clusters\_top\_m\_tweets.clusterId AS clusterId, clusters\_top\_m\_tweets.tweets AS tweets

FROM (

SELECT userId, clusterId, userScore

FROM consumer\_embeddings\_top\_n\_clusters, UNNEST(topClustersWithScores)

) AS consumer\_embedding\_cluster\_score\_pairs

JOIN clusters\_top\_m\_tweets ON consumer\_embedding\_cluster\_score\_pairs.clusterId = clusters\_top\_m\_tweets.clusterId

),

-- (step 4) Compute the dot product between each user and tweet embedding pair

user\_tweet\_embedding\_dot\_product AS (

SELECT userId,

tweetId,

SUM(userScore \* tweetScore) AS dotProductScore

FROM user\_top\_mn\_tweets, UNNEST(tweets) AS tweets

GROUP BY userId, tweetId

),

-- (step 5) Compute similarity scores: dot product, cosine sim, log-cosine sim

user\_tweet\_embedding\_similarity\_scores AS (

SELECT userId,

user\_tweet\_embedding\_dot\_product.tweetId AS tweetId,

dotProductScore,

SAFE\_DIVIDE(dotProductScore, SQRT(tweet\_embeddings\_norm.norm)) AS cosineSimilarityScore,

SAFE\_DIVIDE(dotProductScore, LN(1+tweet\_embeddings\_norm.norm)) AS logCosineSimilarityScore,

FROM user\_tweet\_embedding\_dot\_product

JOIN tweet\_embeddings\_norm ON user\_tweet\_embedding\_dot\_product.tweetId = tweet\_embeddings\_norm.tweetId

),

-- (step 6) Get final top K tweets per user. K = 150 in prod

results AS (

SELECT userId, ARRAY\_AGG(STRUCT(tweetId, dotProductScore, cosineSimilarityScore, logCosineSimilarityScore)

ORDER BY logCosineSimilarityScore DESC LIMIT {TOP\_K\_TWEETS\_PER\_USER\_REQUEST}) AS tweets

FROM user\_tweet\_embedding\_similarity\_scores

GROUP BY userId

)

SELECT \*

FROM results