-- Improved Ethereum Activity Index with Z-Score Normalization

WITH eth\_prices AS (

SELECT

DATE\_TRUNC('month', hour) AS month,

AVG(price) AS avg\_eth\_price\_usd

FROM ETHEREUM.PRICE.EZ\_PRICES\_HOURLY

WHERE symbol = 'ETH'

AND hour >= '2023-01-01'

AND hour < '2025-09-01'

AND price BETWEEN 100 AND 10000

GROUP BY 1

),

-- Base activity metrics from transactions

base\_activity AS (

SELECT

DATE\_TRUNC('month', block\_timestamp) AS month,

COUNT(\*) AS total\_transactions,

COUNT(DISTINCT from\_address) AS unique\_users,

SUM(COALESCE(value, 0)) AS native\_eth\_volume,

COUNT(CASE WHEN tx\_succeeded = TRUE THEN 1 END) AS successful\_transactions

FROM ETHEREUM.CORE.FACT\_TRANSACTIONS

WHERE block\_timestamp >= '2023-01-01'

AND block\_timestamp < '2025-09-01'

GROUP BY 1

),

-- DeFi volumes for economic activity

defi\_volumes AS (

SELECT

DATE\_TRUNC('month', block\_timestamp) AS month,

SUM(COALESCE(amount\_in\_usd, 0)) AS dex\_volume\_usd,

COUNT(DISTINCT origin\_from\_address) AS dex\_users

FROM ETHEREUM.DEFI.EZ\_DEX\_SWAPS

WHERE block\_timestamp >= '2023-01-01'

AND block\_timestamp < '2025-09-01'

AND amount\_in\_usd BETWEEN 0.1 AND 1e9

AND amount\_in\_usd IS NOT NULL

GROUP BY 1

),

lending\_volumes AS (

-- Combine lending deposits and borrows

SELECT

month,

SUM(lending\_volume) AS total\_lending\_volume,

COUNT(DISTINCT user\_address) AS lending\_users

FROM (

SELECT

DATE\_TRUNC('month', block\_timestamp) AS month,

SUM(COALESCE(amount\_usd, 0)) AS lending\_volume,

origin\_from\_address AS user\_address

FROM ETHEREUM.DEFI.EZ\_LENDING\_DEPOSITS

WHERE block\_timestamp >= '2023-01-01'

AND block\_timestamp < '2025-09-01'

AND amount\_usd BETWEEN 0.1 AND 1e9

AND amount\_usd IS NOT NULL

GROUP BY 1, 3

UNION ALL

SELECT

DATE\_TRUNC('month', block\_timestamp) AS month,

SUM(COALESCE(amount\_usd, 0)) AS lending\_volume,

origin\_from\_address AS user\_address

FROM ETHEREUM.DEFI.EZ\_LENDING\_BORROWS

WHERE block\_timestamp >= '2023-01-01'

AND block\_timestamp < '2025-09-01'

AND amount\_usd BETWEEN 0.1 AND 1e9

AND amount\_usd IS NOT NULL

GROUP BY 1, 3

)

GROUP BY 1

),

-- Combine all metrics

combined\_metrics AS (

SELECT

ba.month,

ba.total\_transactions,

ba.unique\_users,

ba.successful\_transactions,

ba.native\_eth\_volume,

COALESCE(dv.dex\_volume\_usd, 0) AS dex\_volume\_usd,

COALESCE(dv.dex\_users, 0) AS dex\_users,

COALESCE(lv.total\_lending\_volume, 0) AS lending\_volume\_usd,

COALESCE(lv.lending\_users, 0) AS lending\_users,

-- Total economic activity (native + DeFi)

COALESCE(dv.dex\_volume\_usd, 0) + COALESCE(lv.total\_lending\_volume, 0) AS total\_defi\_volume\_usd

FROM base\_activity ba

LEFT JOIN defi\_volumes dv ON ba.month = dv.month

LEFT JOIN lending\_volumes lv ON ba.month = lv.month

),

-- Calculate statistics for z-score normalization

stats AS (

SELECT

AVG(total\_transactions) AS avg\_txns,

STDDEV(total\_transactions) AS std\_txns,

AVG(unique\_users) AS avg\_users,

STDDEV(unique\_users) AS std\_users,

AVG(total\_defi\_volume\_usd) AS avg\_volume,

STDDEV(total\_defi\_volume\_usd) AS std\_volume

FROM combined\_metrics

WHERE total\_transactions > 0 -- Exclude any null months

)

-- Final calculation with multiple index approaches

SELECT

p.month,

p.avg\_eth\_price\_usd,

cm.total\_transactions,

cm.successful\_transactions,

cm.unique\_users,

cm.dex\_volume\_usd / 1e9 AS dex\_volume\_billions,

cm.lending\_volume\_usd / 1e9 AS lending\_volume\_billions,

cm.total\_defi\_volume\_usd / 1e9 AS total\_defi\_volume\_billions,

-- Z-score normalized metrics

(cm.total\_transactions - s.avg\_txns) / NULLIF(s.std\_txns, 0) AS txn\_zscore,

(cm.unique\_users - s.avg\_users) / NULLIF(s.std\_users, 0) AS user\_zscore,

(cm.total\_defi\_volume\_usd - s.avg\_volume) / NULLIF(s.std\_volume, 0) AS volume\_zscore,

-- Composite Activity Index (z-score weighted)

(0.25 \* (cm.total\_transactions - s.avg\_txns) / NULLIF(s.std\_txns, 0) +

0.35 \* (cm.unique\_users - s.avg\_users) / NULLIF(s.std\_users, 0) +

0.4 \* (cm.total\_defi\_volume\_usd - s.avg\_volume) / NULLIF(s.std\_volume, 0)) AS activity\_index

FROM eth\_prices p

JOIN combined\_metrics cm ON p.month = cm.month

CROSS JOIN stats s

ORDER BY p.month