**CYPHER QUERIES**

Experiment

## Query to get Artifacts and its number of Releases

MATCH (a:Artifact)-[r:relationship\_AR]->(re:Release)

RETURN a.id AS Artifact, COUNT(re) AS ReleaseCount

ORDER BY ReleaseCount DESC LIMIT 200

// Results saved at Data/ReleaseCountperArtifact.csv

RQ1

## Query to get count of Releases per year

MATCH (a:Artifact)-[r:relationship\_AR]->(b:Release)

WITH b, datetime({epochMillis: b.timestamp}) AS releaseDate

RETURN releaseDate.year AS year, releaseDate.month AS month, COUNT(b) AS releaseCount

ORDER BY year, month

// Results saved at Data/evolution.xlsx

RQ1

## Query to get count of Artifacts per year

MATCH (a:Artifact)-[r:relationship\_AR]->(b:Release)

WITH a, b, MIN(b.timestamp) AS firstReleaseTimestamp

WITH a, b, datetime({epochMillis: firstReleaseTimestamp}) AS releaseDate

RETURN releaseDate.year AS year, releaseDate.month AS month, COUNT(DISTINCT a)

AS artifactCount

ORDER BY year, month

// Results saved at Data/ evolution.xlsx

RQ2 and RQ3

## Query to fetch Artifacts and Count of its Dependencies.

MATCH(a:Artifact)-[ra:relationship\_AR]->(r:Release)-[d:dependency]->(ar:Artifact)

RETURN a.id AS Artifact, COUNT(DISTINCT(ar)) AS DependenciesCount

// Results saved at Data/DependenciesCountperArtifact.csv

RQ2 and RQ3

## Query to fetch Artifacts and details its Dependencies.

MATCH (r:Release)-[d:dependency]->(ad:Artifact)

WHERE r.id =~ "<ARTIFACT\_ID>.\*"

RETURN  r.id AS releaseID, r.timestamp AS releaseUnixTime,

        r.version AS releaseVersion,

        datetime({epochMillis: r.timestamp}) AS releaseDate,

        COUNT(d) AS dependencyCount,

        COLLECT(ad.id) AS dependencyArtIDs

ORDER BY releaseDate, releaseVersion

// Results stored in Data/DependenciesDetails folder

RQ2, RQ3 and RQ4

## Query to fetch Artifacts and Count of Dependent releases and artifacts.

MATCH (a:Artifact)<-[d:dependency]-(r:Release)

WITH a, COUNT(DISTINCT r) AS DependentReleases, COLLECT(r) AS releases

UNWIND releases AS r

MATCH (r)<-[ra:relationship\_AR]-(ar:Artifact)

RETURN a.id AS Artifact, DependentReleases, COUNT(DISTINCT ar) AS DependentArtifacts

// Results saved in Data/DependentsCounts.csv

RQ4

## Query to fetch all Critical CVE and respective ArtifactIDs.

MATCH (av:AddedValue)<-[ad]-(r)<-[ra]-(a:Artifact)

WHERE av.value CONTAINS 'CRITICAL\\'

WITH a, split(x.value, '},') AS rows

UNWIND rows AS row

WITH a, row, split(row, ']') AS parts

UNWIND parts AS part

WITH a, part, split(part, ',') AS subparts

WITH a, part, subparts[1] AS severity, subparts[2] AS name

WHERE severity CONTAINS 'CRITICAL\\'

WITH DISTINCT split(name, '"')[3] AS CVE, a.id AS Artifact

// Results saved in Data/CriticalCVE.csv

RQ4

## Query to fetch Artifacts and Count of Dependent releases and artifacts.

MATCH (ap:Artifact {id: "<ARTIFACT\_ID>.\*"})<-[d:dependency]-(r:Release)

MATCH (ap)-[ar:relationship\_AR]->(rp:Release)

WHERE rp.version = d.targetVersion  // Match the appropriate release by version

RETURN ap.id AS artifactID,

       d.targetVersion AS targetVersion,

       COUNT(d) AS dependentCount,

       datetime({epochMillis: rp.timestamp}) AS releaseDate,

       rp.timestamp AS releaseTimestamp

ORDER BY releaseTimestamp

// Results stored in Data/Dependents folder