graph_NVD

```
CREATE SEQUENCE cve_graph_seq
    START WITH 1
    INCREMENT BY 1;

########## node: vuln_nvd  构建
DROP TABLE test.t_tmp_nvd_impact_exploit_score;
CREATE TABLE test.t_tmp_nvd_impact_exploit_score AS
WITH wt_cvss_score AS
(
SELECT dns.nvd_id , dns.cve_msg -> 'metrics' ->  'cvssMetricV2' -> 0  ->> 'impactScore' AS impact_score,
       dns.cve_msg -> 'metrics' ->  'cvssMetricV2' -> 0  ->> 'exploitabilityScore' AS exploitabilityScore,
       jsonb_build_object('source',dns.cve_msg -> 'metrics' ->  'cvssMetricV2' -> 0  ->> 'source' ,
              'type',dns.cve_msg -> 'metrics' ->  'cvssMetricV2' -> 0  ->> 'type' ) AS score_source_type
FROM  warehouse.dwd_nvd_source_f dns
--WHERE dns.nvd_id = 'CVE-1999-0095'
UNION all
SELECT dns.nvd_id , dns.cve_msg -> 'metrics' ->  'cvssMetricV30' -> 0  ->> 'impactScore' ,
       dns.cve_msg -> 'metrics' ->  'cvssMetricV30' -> 0  ->> 'exploitabilityScore' ,
       jsonb_build_object('source',dns.cve_msg -> 'metrics' ->  'cvssMetricV30' -> 0  ->> 'source' ,
              'type',dns.cve_msg -> 'metrics' ->  'cvssMetricV30' -> 0  ->> 'type' )
FROM  warehouse.dwd_nvd_source_f dns
--WHERE dns.nvd_id = 'CVE-1999-0095'
UNION ALL
SELECT dns.nvd_id , dns.cve_msg -> 'metrics' ->  'cvssMetricV31' -> 0  ->> 'impactScore' ,
       dns.cve_msg -> 'metrics' ->  'cvssMetricV31' -> 0  ->> 'exploitabilityScore' ,
       jsonb_build_object('source',dns.cve_msg -> 'metrics' ->  'cvssMetricV31' -> 0  ->> 'source' ,
              'type',dns.cve_msg -> 'metrics' ->  'cvssMetricV31' -> 0  ->> 'type' )
FROM  warehouse.dwd_nvd_source_f dns
--WHERE dns.nvd_id = 'CVE-1999-0095'
),
wt_impact AS
(
    SELECT im_tmp.nvd_id, jsonb_agg(im_tmp.impact_msg)  AS impact_msg_array
    FROM (SELECT wcs.nvd_id, jsonb_build_object( wcs.impact_score, jsonb_agg(DISTINCT score_source_type))  AS i
            FROM wt_cvss_score wcs
            WHERE wcs.impact_score IS NOT NULL
            GROUP BY wcs.nvd_id, wcs.impact_score
            )im_tmp
    GROUP BY im_tmp.nvd_id
),
wt_exploit_Score AS
(
    SELECT ex_tmp.nvd_id, jsonb_agg(ex_tmp.exploit_msg)  AS exploit_msg_array
    FROM (
        SELECT wcs.nvd_id, jsonb_build_object( wcs.exploitabilityScore, jsonb_agg(DISTINCT score_source_type) )
        FROM wt_cvss_score wcs
        WHERE wcs.exploitabilityScore IS NOT NULL
        GROUP BY wcs.nvd_id, wcs.exploitabilityScore
```

```
    )ex_tmp
    GROUP BY ex_tmp.nvd_id
)
SELECT t.nvd_id , wi.impact_msg_array, we.exploit_msg_array
FROM warehouse.dwd_nvd_source_f t
LEFT JOIN wt_impact wi
       ON t.nvd_id = wi.nvd_id
LEFT JOIN wt_exploit_Score we
       ON t.nvd_id = we.nvd_id;

ALTER SEQUENCE cve_graph_seq RESTART START WITH 1;
DROP TABLE IF EXISTS test.t_tmp_grahp_node_vuln_nvd ;
CREATE TABLE test.t_tmp_grahp_node_vuln_nvd AS
WITH wt_weakness AS
(
    SELECT wn.nvd_id, jsonb_agg(wn.weakness_msg) AS weakness_msg_array
    FROM
    (
        SELECT dns.nvd_id, jsonb_build_object('type', weaknesses.value ->> 'type' ,
                                              'cweId', weaknesses.value -> 'description' -> 0 ->> 'value' ,
                                              'description', NULL) AS weakness_msg
        FROM warehouse.dwd_nvd_source_f dns
        LEFT JOIN jsonb_array_elements(dns.cve_msg -> 'weaknesses') weaknesses ON 1=1
    )wn
    GROUP BY wn.nvd_id
)
,nvd_vul_tmp AS
(
SELECT
       dns.cve_msg ->> 'id' AS id,
      -- null AS name,  在创建node的时候，给name属性值为空
       '[]' AS aliases,
       dns.cve_msg  ->> 'sourceIdentifier' AS source_Identifier,
       dns.cve_msg -> 'descriptions' -> 0 ->> 'value' AS descriptions_details,
       jsonb_build_object('cvssV4_0',  dns.cve_msg -> 'metrics' ->  'cvssMetricV40' -> 0  -> 'cvssData' )
        AS cvssMetricV40,
       jsonb_build_object('cvssV3_1',  dns.cve_msg -> 'metrics' ->  'cvssMetricV31' -> 0  -> 'cvssData' )
        AS cvssMetricV31,
       jsonb_build_object('cvssV3_0',  dns.cve_msg -> 'metrics' ->  'cvssMetricV30' -> 0  -> 'cvssData' )
        AS cvssMetricV30,
       jsonb_build_object('cvssV2_0',  dns.cve_msg -> 'metrics' ->  'cvssMetricV2' -> 0  -> 'cvssData'
                           || (('{"baseSeverity": "' || (dns.cve_msg -> 'metrics' ->  'cvssMetricV2' -> 0  ->>
                           || (',"userInteraction":"' || (dns.cve_msg -> 'metrics' ->  'cvssMetricV2' -> 0  ->
        AS cvssMetricV2,
       dns.cve_msg ->> 'lastModified' AS timeinfo_lastModified,
       dns.cve_msg ->> 'published' AS timeinfo_published,
       dns.cve_msg ->> 'vulnStatus' AS vulnStatus,
--     dns.cve_msg ->> 'metrics' AS metrics,
--     dns.cve_msg ->> 'configurations' AS configurations,
       'https://services.nvd.nist.gov/rest/json/cves/2.0?cveId=' || (dns.cve_msg ->> 'id')::varchar AS api_url
FROM warehouse.dwd_nvd_source_f dns
)
SELECT nextval('cve_graph_seq') AS seq, t.id, t.aliases, jsonb_build_object('discovery', NULL, 'identifier', t.
      jsonb_build_object('title', NULL, 'details', descriptions_details) AS description_msg,
      ww.weakness_msg_array,
      ( CASE WHEN (cvssMetricV40 -> 'cvssV4_0') = 'null' THEN '{}' ELSE cvssMetricV40 END
      || CASE WHEN (cvssMetricV31 -> 'cvssV3_1') = 'null' THEN '{}' ELSE cvssMetricV31 END
      || CASE WHEN (cvssMetricV30 -> 'cvssV3_0') = 'null' THEN '{}' ELSE cvssMetricV30 END
```

```
      || CASE WHEN (cvssMetricV2 -> 'cvssV2_0') = 'null' THEN '{}' ELSE cvssMetricV2 END )::jsonb AS severity,
      jsonb_build_object('published', t.timeinfo_published, 'lastModified', t.timeinfo_lastModified, 'datePubli
      jsonb_build_object('impact_info', jsonb_build_object('impacts',NULL, 'impactScore', ies.impact_msg_array)
                         'solution_info', NULL ,
                         'exploit_info', jsonb_build_object('exploitable', NULL, 'exploits', NULL, 'exploit_url
                         'PoC_info', jsonb_build_object('PoC_available', NULL, 'PoC', NULL, 'PoC_url', NULL),
                         'patch_info', jsonb_build_object('patch_available', NULL, 'patch_url', null),
                         'report_status', null
       ) AS vul_status, t.vulnStatus, t.api_url
FROM nvd_vul_tmp t
LEFT JOIN test.t_tmp_nvd_impact_exploit_score ies
       ON t.id = ies.nvd_id
LEFT JOIN wt_weakness ww
       ON t.id = ww.nvd_id ;


DELETE FROM test.dws_graph_node_vul WHERE vul_source = 'NVD';
ALTER SEQUENCE cve_graph_seq RESTART START WITH 1;
INSERT INTO test.dws_graph_node_vul
SELECT nextval('cve_graph_seq') AS seq, tmp.*
FROM
(
SELECT  DISTINCT  id ,  aliases , source_msg, description_msg ,weakness_msg_array, severity, time_info,  vul_st
)tmp ;
bash gen_graph_data.sh "NVD" "node" "vul"
bash neo4j_vul_node_load.sh "NVD" "0"


ALTER SEQUENCE cve_graph_seq RESTART START WITH 1;

DROP TABLE IF EXISTS test.t_tmp_cve_graph_cwe_nodes ;
CREATE TABLE test.t_tmp_cve_graph_cwe_nodes AS
SELECT nextval('cve_graph_seq') AS seq,
       tmp.*
FROM
(
SELECT DISTINCT dns.nvd_id,
       weaknesses.value ->> 'type' AS cwe_type,
       weaknesses.value ->> 'source'AS cwe_source,
       weaknesses.value -> 'description' -> 0 ->> 'value' AS cwe_id
FROM warehouse.dwd_nvd_source_f dns ,
     jsonb_array_elements(dns.cve_msg -> 'weaknesses') weaknesses
--     ,jsonb_array_elements(weaknesses.value -> 'description') des
)tmp ;

SELECT *FROM test.t_tmp_cve_graph_cwe_nodes LIMIT 10

DELETE FROM test.dws_graph_node_cwe WHERE vul_source = 'NVD';
ALTER SEQUENCE cve_graph_seq RESTART START WITH 1;
INSERT INTO test.dws_graph_node_cwe
  SELECT nextval('cve_graph_seq') AS seq, tmp.*
  FROM (
    SELECT   DISTINCT cwe_id, 'NVD'   FROM   test.t_tmp_cve_graph_cwe_nodes
  )tmp;
bash gen_graph_data.sh "NVD" "node" "cwe"
bash neo4j_cwe_node_load.sh "NVD" "0"

```

```

DELETE FROM test.dws_graph_relationships_cwe WHERE vul_source = 'NVD';
ALTER SEQUENCE cve_graph_seq RESTART START WITH 1;
INSERT INTO test.dws_graph_relationships_cwe
  SELECT nextval('cve_graph_seq') AS seq, tmp.*
  FROM (
    SELECT   DISTINCT nvd_id, cwe_id , cwe_type, cwe_source , 'NVD' AS vul_source  FROM   test.t_tmp_cve_graph_
  )tmp;
bash gen_graph_data.sh "NVD" "relationships" "cwe"
bash neo4j_relationships_cwe_load.sh "NVD" "0"

--\copy (SELECT DISTINCT cwe_id FROM test.t_tmp_cve_graph_cwe_nodes) to 'cwe_nodes.csv' with  (delimiter ',', F
--
--LOAD CSV WITH HEADERS FROM 'file:///cwe_nodes.csv' AS row
--CREATE (n:cwe {id: row.cwe_id})
--
--LOAD CSV WITH HEADERS FROM 'file:///cwe_nodes.csv' AS row
--MERGE (cwe:CWE {id: row.cwe_id})
--SET cwe.description = row.description,
--    cwe.severity = row.severity
--RETURN cwe;
--
--

--\copy (SELECT DISTINCT nvd_id AS r_start, cwe_id AS r_end,  cwe_type , cwe_source FROM test.t_tmp_cve_graph_c
--\copy (SELECT DISTINCT nvd_id AS r_start, cwe_id AS r_end,  cwe_type , cwe_source FROM test.t_tmp_cve_graph_c
--\copy (SELECT DISTINCT nvd_id AS r_start, cwe_id AS r_end,  cwe_type , cwe_source FROM test.t_tmp_cve_graph_c
--

--LOAD CSV WITH HEADERS FROM 'file:///cwe_ref.csv' AS row
--             MATCH (cve:Vuln_NVD {id: row.r_start})
--             MATCH (cwe:CWE {id: row.r_end})
--             MERGE (cve)-[:HAS_CWE{cweId: row.r_end, type: row.cwe_type, description: row.cwe_source}]->(cwe)

--https://lists.apache.org/thread.html/r0276683d8e1e07153fc8642618830ac0ade85b9ae0dc7b07f63bb8fc%40%3Ccvs.httpd
ALTER SEQUENCE cve_graph_seq RESTART START WITH 1;
DROP TABLE test.t_tmp_cve_graph_ref_nodes ;
CREATE TABLE test.t_tmp_cve_graph_ref_nodes AS
SELECT nextval('cve_graph_seq') AS seq,
       tmp.*
FROM
(
SELECT  DISTINCT dns.nvd_id,
       refs.value ->> 'url' AS ref_url,
       refs.value ->> 'source' AS ref_source,
       refs.value ->> 'tags' AS ref_tags
FROM warehouse.dwd_nvd_source_f dns ,
    jsonb_array_elements(dns.cve_msg -> 'references') refs
)tmp ;

```

```
SELECT  t.*
FROM test.t_tmp_cve_graph_ref_nodes t LIMIT 10

DELETE FROM test.dws_graph_node_refs WHERE vul_source = 'NVD';
ALTER SEQUENCE cve_graph_seq RESTART START WITH 1;
INSERT INTO test.dws_graph_node_refs
  SELECT nextval('cve_graph_seq') AS seq, tmp.*
  FROM (
    SELECT   DISTINCT ref_url, 'NVD' AS source  FROM   test.t_tmp_cve_graph_ref_nodes
  )tmp;
bash gen_graph_data.sh "NVD" "node" "refs"
bash neo4j_refs_node_load.sh "NVD" "0"


DELETE FROM test.dws_graph_relationships_refs WHERE vul_source = 'NVD';
ALTER SEQUENCE cve_graph_seq RESTART START WITH 1;
INSERT INTO test.dws_graph_relationships_refs
  SELECT nextval('cve_graph_seq') AS seq, tmp.*
  FROM (
    SELECT   DISTINCT nvd_id, ref_url , ref_tags,  ref_source , 'NVD' AS vul_source  FROM   test.t_tmp_cve_grap
  )tmp;
bash gen_graph_data.sh "NVD" "relationships" "refs"
bash neo4j_relationships_refs_load.sh "NVD" "0"

--\copy (SELECT DISTINCT ref_url FROM test.t_tmp_cve_graph_ref_nodes) to 'ref_nodes.csv' with  (delimiter ',',
--LOAD CSV WITH HEADERS FROM 'file:///ref_nodes.csv' AS row
--MERGE (ref:references {url: row.ref_url});
--
--create index for (n:references) on (n.url);
--

--\copy (SELECT  DISTINCT nvd_id AS r_start, ref_url AS r_end , ref_source , ref_tags FROM test.t_tmp_cve_graph
--\copy (SELECT DISTINCT nvd_id AS r_start, ref_url AS r_end , ref_source , ref_tags FROM test.t_tmp_cve_graph_
--\copy (SELECT DISTINCT nvd_id AS r_start, ref_url AS r_end , ref_source , ref_tags FROM test.t_tmp_cve_graph_
--\copy (SELECT DISTINCT nvd_id AS r_start, ref_url AS r_end , ref_source , ref_tags FROM test.t_tmp_cve_graph_
--\copy (SELECT DISTINCT nvd_id AS r_start, ref_url AS r_end , ref_source , ref_tags FROM test.t_tmp_cve_graph_
--\copy (SELECT DISTINCT nvd_id AS r_start, ref_url AS r_end , ref_source , ref_tags FROM test.t_tmp_cve_graph_
--\copy (SELECT DISTINCT nvd_id AS r_start, ref_url AS r_end , ref_source , ref_tags FROM test.t_tmp_cve_graph_
--\copy (SELECT DISTINCT nvd_id AS r_start, ref_url AS r_end , ref_source , ref_tags FROM test.t_tmp_cve_graph_
--\copy (SELECT DISTINCT nvd_id AS r_start, ref_url AS r_end , ref_source , ref_tags FROM test.t_tmp_cve_graph_
--\copy (SELECT DISTINCT nvd_id AS r_start, ref_url AS r_end , ref_source , ref_tags FROM test.t_tmp_cve_graph_
--

--LOAD CSV WITH HEADERS FROM 'file:///ref_10w_9.csv' AS row
--             MATCH (cve:Vuln_NVD {id: row.r_start})
--             MATCH (ref:references {url: row.r_end})
--             MERGE (cve)-[:HAS_REFERENCE{description: row.ref_source,tags: COALESCE(row.ref_tags, '')}]->(ref




DROP TABLE test.t_tmp_grahp_node_affected_msg;
CREATE TABLE test.t_tmp_grahp_node_affected_msg AS
WITH wt_cpe_match AS
(
    SELECT replace(cve_cpeMatch.value ->> 'criteria',  E'\\:', '|+|') AS cpe_value,
           cve_cpeMatch.value AS cpe_match_val,
```

```
           dnsf.nvd_id ,
           (cve_cpeMatch.value ->> 'criteria')::varchar AS cpe_source_val
           -- ,ORDINALITY AS cve_nodes_index
    FROM warehouse.dwd_nvd_source_f dnsf ,
    jsonb_array_elements(dnsf.cve_msg -> 'configurations') cve_conf, --WITH ORDINALITY cve_conf,
    jsonb_array_elements(cve_conf.value -> 'nodes') cve_nodes ,
    jsonb_array_elements(cve_nodes.value -> 'cpeMatch') cve_cpeMatch
)
,
wt_ecosystem_map AS (
    select 'node.js'  as key ,    'npm' as value union all
    select  'python'  as key ,  'PyPI' as value union all
    select  'ruby'  as key ,  'RubyGems' as value  union all
    select  'java'  as key ,  'Maven' as value union all
    select  'go'  as key ,  'Go' as value  union all
    select  'nuget'  as key ,  'NuGet' as value  union all
    select  'php'  as key ,  'Packagist' as value  union all
    select  'rust'  as key ,  'crates.io' as value  union all
    select  'android'  as key ,  'Android' as value  union all
    select  'debian'  as key ,  'Debian' as value  union all
    select  'linux'  as key ,  'Linux' as value union all
    select  'alpine'  as key ,  'Alpine' as value union all
    select  'git'  as key ,  'GIT' as value  union all
    select  'oss-fuzz'  as key ,  'OSS-Fuzz'
),
wt_cpe_parse AS
(
    SELECT cm.cpe_value,
           split_part(cm.cpe_value, ':', 3) AS part,
           REPLACE(split_part(cm.cpe_value, ':', 4), '|+|', ':') AS vendor,
           REPLACE(split_part(cm.cpe_value, ':', 4), '|+|', '\\:') AS vendor_source,
           replace(split_part(cm.cpe_value, ':', 5), '|+|', ':') AS product,
           replace(split_part(cm.cpe_value, ':', 5), '|+|', '\\:') AS product_source,
--         split_part(cm.cpe_value, ':', 5) AS product,
           split_part(cm.cpe_value, ':', 6) AS version,
           split_part(cm.cpe_value, ':', 7) AS update,
           split_part(cm.cpe_value, ':', 8) AS edition,
           split_part(cm.cpe_value, ':', 9) AS language,
           split_part(cm.cpe_value, ':', 10) AS sw_edition,
           split_part(cm.cpe_value, ':', 11) AS target_sw,
           split_part(cm.cpe_value, ':', 12) AS target_hw,
           split_part(cm.cpe_value, ':', 13) AS other,
           cm.nvd_id ,
           cm.cpe_source_val,
           cm.cpe_match_val
           -- ,cm.cve_nodes_index
    FROM wt_cpe_match cm
)
,
wt_cpe_process AS
(
    SELECT cp1.cpe_match_val,
            (CASE WHEN cp1.versionStartExcluding IS NOT NULL THEN '(' || cp1.versionStartIncluding
                  WHEN cp1.versionStartIncluding IS NOT NULL THEN '[' || cp1.versionStartIncluding
                  ELSE '(0' END )
         || (CASE WHEN cp1.versionEndExcluding IS NOT NULL THEN  ',' || cp1.versionEndExcluding || ')'
                  WHEN cp1.versionEndIncluding IS NOT NULL THEN  ',' || cp1.versionEndIncluding || ']' ELSE ',-
         cp1.package_name,
```

```
         cp1.part || ':' || cp1.vendor_source || ':' || cp1.package_name AS unique_package,
         cp1.version_op,
         cp1.matchCriteriaId,
         cp1.nvd_id,
         cp1.vulnerable,
         cp1.vendor,
         cp1.cpe_source_val,
         -- , cp1.cve_nodes_index
         cp1.ecosystem
    FROM
    (
        SELECT CASE WHEN cp.target_sw <> '*' THEN cp.product_source || '-' || cp.target_sw ELSE cp.product END
               CASE WHEN cp.target_sw <> '*' AND em.KEY IS NOT NULL THEN  em.value ELSE '*' END AS ecosystem,
               cp.VERSION || (CASE WHEN cp.VERSION <> '*' AND cp.UPDATE <> '*' THEN '.' || cp.UPDATE ELSE '' EN
                          || (CASE WHEN cp.VERSION <> '*' AND cp.edition <> '*' THEN '.' || cp.edition ELSE ''
               cp.cpe_match_val ->> 'vulnerable' AS vulnerable,
               cp.cpe_match_val ->> 'versionStartExcluding' AS versionStartExcluding,
               cp.cpe_match_val ->> 'versionStartIncluding' AS versionStartIncluding,
               cp.cpe_match_val ->> 'versionEndExcluding' AS versionEndExcluding,
               cp.cpe_match_val ->> 'versionEndIncluding' AS versionEndIncluding,
               cp.cpe_match_val ->> 'matchCriteriaId' AS matchCriteriaId,
               cp.nvd_id,
               cp.cpe_match_val,
               cp.vendor
               ,cp.cpe_source_val
               ,cp.part, cp.vendor_source
               -- ,cp.cve_nodes_index
        FROM wt_cpe_parse cp
        LEFT JOIN wt_ecosystem_map em
               ON cp.target_sw = em.key
    )cp1
)
, wt_cve_affect_unnest AS
(
    SELECT  tmp.nvd_id, tmp.unique_package, tmp.package_name, tmp.vendor , tmp.vulnerable, tmp.ecosystem,
           array_to_string(array_agg(tmp.version_events), ',') AS cve_affect
           ,sum(tmp.version_flag) AS version_flag_sum
           ,'{' || array_to_string( array_agg(tmp.cpe_source_val_list) , ',') || '}' AS cpe_source_val_list
--        json_build_object('version_events',   tmp.version_events , 'vulnerable', tmp.vulnerable) AS json_affe
--        tmp.cve_nodes_index
--    SELECT          tmp.nvd_id, tmp.unique_package, tmp.package_name, tmp.vendor , tmp.cve_nodes_index
    FROM
    (
        SELECT cp.nvd_id , cp.unique_package, cp.package_name,  cp.vendor,  '{' || array_to_string(array_agg(DI
               ,array_to_string(array_agg(cp.cpe_source_val),',') AS cpe_source_val_list, 1 AS version_flag, cp
        FROM wt_cpe_process cp
        WHERE  cp.version_ranges = '(0,-)'-- <> '(0,,-)'
        GROUP BY cp.nvd_id , cp.unique_package, cp.package_name,  cp.vendor, cp.vulnerable, cp.ecosystem
        UNION ALL
        SELECT cp2.nvd_id , cp2.unique_package, cp2.package_name,  cp2.vendor,  array_to_string(array_agg(DISTI
              ,array_to_string(array_agg(cp2.cpe_source_val),',') AS cpe_source_val_list, 2 AS version_flag, cp
        FROM wt_cpe_process cp2
        WHERE cp2.version_ranges <>  '(0,-)'
        GROUP BY cp2.nvd_id , cp2.unique_package, cp2.package_name,  cp2.vendor, cp2.vulnerable, cp2.ecosystem
    )tmp
--  WHERE tmp.nvd_id = 'CVE-2019-12767'
    GROUP BY tmp.nvd_id, tmp.unique_package, tmp.package_name, tmp.vendor , tmp.vulnerable, tmp.ecosystem
--  HAVING count(DISTINCT tmp.vulnerable) > 1
```

```
)
SELECT t1.nvd_id, t1.unique_package, t1.vendor, t1.package_name, t1.ecosystem, jsonb_build_object('true', t1.cv
       t1.cpe_source_val_list, split_part(t1.unique_package, ':', 1) AS cpe_part , t1.version_flag_sum,  CASE W
FROM (SELECT *FROM  wt_cve_affect_unnest cau WHERE cau.vulnerable = 'true')t1
LEFT JOIN (SELECT *FROM  wt_cve_affect_unnest cau2 WHERE cau2.vulnerable = 'false')t2
      ON t1.nvd_id = t2.nvd_id
      AND t1.unique_package = t2.unique_package
UNION ALL
SELECT t1.nvd_id, t1.unique_package, t1.vendor, t1.package_name, t1.ecosystem,  jsonb_build_object('true', null
       t1.cpe_source_val_list, split_part(t1.unique_package, ':', 1) AS cpe_part, NULL AS version_flag,  '{}' A
FROM (SELECT *FROM  wt_cve_affect_unnest cau WHERE cau.vulnerable = 'false')t1;


ALTER SEQUENCE cve_graph_seq RESTART START WITH 1;
CREATE TABLE test.t_tmp_node_component AS
SELECT nextval('cve_graph_seq') AS seq, nvd_id , package_name AS component_name, vendor, package_name, ecosyste
       NULL AS collectionURL, NULL AS defaultStatus,
       split_part(unique_package,':',1) AS platform, jsonb_build_object('version_range', t.package_events ->> '
       jsonb_build_object('version_range', t.package_events ->> 'false', 'version_list', t.packageversions ->>
FROM test.t_tmp_grahp_node_affected_msg t ;


SELECT *FROM test.t_tmp_node_component LIMIT 10

DELETE FROM test.dws_graph_node_affected_component WHERE vul_source = 'NVD';
ALTER SEQUENCE cve_graph_seq RESTART START WITH 1;
INSERT INTO test.dws_graph_node_affected_component
  SELECT nextval('cve_graph_seq') AS seq, tmp.*
  FROM (
    SELECT   DISTINCT component_name , vendor, package_name, ecosystem, 'NVD' AS vul_source   FROM   test.t_tmp
  )tmp;
bash gen_graph_data.sh "NVD" "node" "affected_component"
bash neo4j_affected_component_node_load.sh "NVD" "0"



DELETE FROM test.dws_graph_relationships_affected_components WHERE vul_source = 'NVD';
ALTER SEQUENCE cve_graph_seq RESTART START WITH 1;
INSERT INTO test.dws_graph_relationships_affected_components
  SELECT nextval('cve_graph_seq') AS seq, tmp.*
  FROM (
    SELECT   DISTINCT nvd_id, component_name, vendor, package_name, ecosystem, '' repo_url, '' AS platform,
             '' AS collectionurl, '' AS defaultstatus, affect_msg, unaffect_msg, 'NVD' AS vul_source   FROM   t
  )tmp;
bash gen_graph_data.sh "NVD" "relationships" "affected_components"
bash neo4j_relationships_affected_components.sh "NVD" "0"

--\copy (SELECT DISTINCT package_name AS component_name, vendor, package_name, ecosystem FROM test.t_tmp_node_c
--LOAD CSV WITH HEADERS FROM 'file:///affected_component_nodes.csv' AS row
--merge (n:affected_components {component_name: row.component_name, ecosystem: row.ecosystem, package_name:row.

--SELECT *FROM test.t_tmp_node_component LIMIT 100


--\copy (SELECT DISTINCT vendor FROM test.t_tmp_grahp_node_affected_msg) to 'vendor_nodes.csv' with  (delimiter


--SELECT *FROM test.t_tmp_grahp_node_affected_msg LIMIT 10
```

```
--
--
--
--create index for (n:Package) on (n.unique_name);
--
--
--LOAD CSV WITH HEADERS FROM 'file:///vendor_nodes.csv' AS row
--CREATE (n:Vendor {name: row.vendor})
--create index for (n:Vendor) on (n.name);


--SELECT *FROM test.t_tmp_node_component LIMIT 10
--
--\copy (SELECT   * FROM test.t_tmp_node_component WHERE seq <= 100000) to 'r_affected_10w_1.csv' with  (delimi
----\copy (SELECT DISTINCT   unique_package AS r_start, vendor AS r_end FROM test.t_tmp_grahp_node_affected_msg
--
--LOAD CSV WITH HEADERS FROM 'file:///r_affected_10w_11.csv' AS row
--           MATCH (n_nvd:Vuln_NVD {id: row.nvd_id})
--             MATCH (np:affected_components {component_name: row.component_name, ecosystem: row.ecosystem, pac
--             MERGE (n_nvd)-[:AFFECTS {repo_url: COALESCE(row.repo_url,''),  platform: row. platform, collecti

```