Neo4j Aura Resume Import – Cypher Runbook

One-page reference for importing a JSON-LD resume graph from S3 into Neo4j Aura, promoting predicates into native relationships, and visualizing the resume subgraph.

1. One-time Constraint (Idempotent)

```
CREATE CONSTRAINT resource_uri IF NOT EXISTS
FOR (n:Resource) REQUIRE n.uri IS UNIQUE;
```

2. Create Nodes from @id

```
CALL apoc.load.json("<S3_JSONLD_URL>") YIELD value AS row
MERGE (:Resource {uri: row['@id']});
```

3. Pass A – Create Relationships from object @id (Aura-safe)

```
CALL apoc.load.json("<S3_JSONLD_URL>") YIELD value AS row
MATCH (n:Resource {uri: row['@id']})
UNWIND [k IN keys(row) WHERE NOT k IN ['@id','@type']] AS pred
WITH n, pred, row[pred] AS raw
WITH n, pred,
  CASE WHEN raw IS NULL THEN []
  WHEN raw IS :: LIST<ANY> THEN raw
  ELSE [raw] END AS objs
UNWIND objs AS obj
WITH n, obj, last(split(pred, '#')) AS relKey
WHERE obj IS :: MAP AND obj['@id'] IS NOT NULL
MERGE (m:Resource {uri: obj['@id']})
MERGE (n)-[:RELATES {type: relKey}]->(m);
```

4. Pass B – Literal Properties (typed yearsExperience)

```
CALL apoc.load.json("<S3_JSONLD_URL>") YIELD value AS row
MATCH (n:Resource {uri: row['@id']})
UNWIND [k IN keys(row) WHERE NOT k IN ['@id','@type']] AS pred
WITH n, pred, row[pred] AS raw
WITH n, pred,
  CASE WHEN raw IS NULL THEN []
  WHEN raw IS :: LIST<ANY> THEN raw
  ELSE [raw] END AS objs
UNWIND objs AS obj
WITH n, obj, last(split(pred, '#')) AS key
WHERE obj IS :: MAP AND obj['@value'] IS NOT NULL
FOREACH (_ IN CASE WHEN key = "yearsExperience" THEN [1] ELSE [] END |
  SET n.yearsExperience = toFloat(obj['@value']) )
FOREACH (_ IN CASE WHEN key <> "yearsExperience" THEN [1] ELSE [] END |
  SET n[key] = coalesce(n[key], []) + toString(obj['@value']) );
```

5. Promote RELATES → Native Relationships

```
MATCH (a)-[r:RELATES {type:"hasSkill"}]->(b) MERGE (a)-[:HAS_SKILL]->(b) DELETE r;
MATCH (a)-[r:RELATES {type:"hasJob"}]->(b) MERGE (a)-[:HAS_JOB]->(b) DELETE r;
MATCH (a)-[r:RELATES {type:"hasCertification"}]->(b) MERGE (a)-[:HAS_CERTIFICATION]->(b) DELETE r;
MATCH (a)-[r:RELATES {type:"usedTechnology"}]->(b) MERGE (a)-[:USED_TECHNOLOGY]->(b) DELETE r;
MATCH (a)-[r:RELATES {type:"exactMatch"}]->(b) MERGE (a)-[:EXACT_MATCH]->(b) DELETE r;
MATCH (a)-[r:RELATES {type:"broader"}]->(b) MERGE (a)-[:BROADER]->(b) DELETE r;
MATCH (a)-[r:RELATES {type:"narrower"}]->(b) MERGE (a)-[:NARROWER]->(b) DELETE r;
MATCH (a)-[r:RELATES {type:"http://schema.org/alumniOf"}]->(b) MERGE (a)-[:ALUMNI_OF]->(b) DELETE
r;
MATCH (a)-[r:RELATES {type:"http://schema.org/hiringOrganization"}]->(b) MERGE
(a)-[:HIRING_ORGANIZATION]->(b) DELETE r;
```

6. Visualization Queries

```
MATCH (p:Resource {uri:"<PERSON_URI>"})
OPTIONAL MATCH (p)-[r1]-(n1)
OPTIONAL MATCH (n1)-[r2]-(n2)
RETURN p,r1,n1,r2,n2;

MATCH (p:Resource)-[:HAS_SKILL]->(s:Resource) RETURN p,s;
MATCH (p:Resource)-[:HAS_JOB]->(j:Resource)
OPTIONAL MATCH (j)-[:HIRING_ORGANIZATION]->(o:Resource)
RETURN p,j,o;
```