# https://medium.com/larus-team/how-to-create-recommendation-engine-in-neo4j-7963e635c730

**Airbnb Neo4j queries**

LOAD CSV FROM 'file:///neo4J\_nodes.csv' AS row

WITH row, row[0] as id, row[1] as labels, row[2] as name, row[3] as start, row[4] as end, row[5] as type, toFloat(row[6]) as rating, row[7] as url,row[8] as picture\_url,row[9] as host\_identity\_verified,row[10] as accomodates,row[11] as bedrooms,row[12] as bathrooms,row[13] as beds,row[14] as price,toFloat(row[15]) as score

WHERE labels=":User"

CREATE (:User{id:id, name: name});

LOAD CSV FROM 'file:///neo4J\_nodes.csv' AS row

WITH row, row[0] as id, row[1] as labels, row[2] as name, row[3] as start, row[4] as end, row[5] as type, toFloat(row[6]) as rating, row[7] as url,row[8] as picture\_url,row[9] as host\_identity\_verified,row[10] as accomodates,row[11] as bedrooms,row[12] as bathrooms,row[13] as beds,row[14] as price,toFloat(row[15]) as score

WHERE labels=":Amenities"

CREATE (:Amenities{id:id, name: name});

LOAD CSV FROM 'file:///neo4J\_nodes.csv' AS row

WITH row, row[0] as id, row[1] as labels, row[2] as name, row[3] as start, row[4] as end, row[5] as type, toFloat(row[6]) as rating, row[7] as url,row[8] as picture\_url,row[9] as host\_identity\_verified,row[10] as accomodates,row[11] as bedrooms,row[12] as bathrooms,row[13] as beds,row[14] as price,toFloat(row[15]) as score

WHERE labels=":property\_type"

CREATE (:property\_type{id:id, name: name});

LOAD CSV FROM 'file:///neo4J\_nodes.csv' AS row

WITH row, row[0] as id, row[1] as labels, row[2] as name, row[3] as start, row[4] as end, row[5] as type, toFloat(row[6]) as rating, row[7] as url,row[8] as picture\_url,row[9] as host\_identity\_verified,row[10] as accomodates,row[11] as bedrooms,row[12] as bathrooms,row[13] as beds,row[14] as price,toFloat(row[15]) as score

WHERE labels=":room\_type"

CREATE (:room\_type{id:id, name: name});

LOAD CSV FROM 'file:///neo4J\_nodes.csv' AS row

WITH row, row[0] as id, row[1] as labels, row[2] as name, row[3] as start, row[4] as end, row[5] as type, toFloat(row[6]) as rating, row[7] as url,row[8] as picture\_url,row[9] as host\_identity\_verified,row[10] as accomodates,row[11] as bedrooms,row[12] as bathrooms,row[13] as beds,row[14] as price,toFloat(row[15]) as score

WHERE labels=":City"

CREATE (:City{id:id, name: name});

LOAD CSV FROM 'file:///neo4J\_nodes.csv' AS row

WITH row, row[0] as id, row[1] as labels, row[2] as name, row[3] as start, row[4] as end, row[5] as type, toFloat(row[6]) as rating, row[7] as url,row[8] as picture\_url,row[9] as host\_identity\_verified,row[10] as accomodates,row[11] as bedrooms,row[12] as bathrooms,row[13] as beds,row[14] as price,toFloat(row[15]) as score

WHERE labels=":State"

CREATE (:State{id:id, name: name});

LOAD CSV FROM 'file:///neo4J\_nodes.csv' AS row

WITH row, row[0] as id, row[1] as labels, row[2] as name, row[3] as start, row[4] as end, row[5] as type, toFloat(row[6]) as rating, row[7] as url,row[8] as picture\_url,row[9] as host\_identity\_verified,row[10] as accomodates,row[11] as bedrooms,row[12] as bathrooms,row[13] as beds,row[14] as price,toFloat(row[15]) as score

WHERE labels=":Country"

CREATE (:Country{id:id, name: name});

LOAD CSV FROM 'file:///neo4J\_nodes.csv' AS row

WITH row, row[0] as id, row[1] as labels, row[2] as name, row[3] as start, row[4] as end, row[5] as type, toFloat(row[6]) as rating, row[7] as url,row[8] as picture\_url,row[9] as host\_identity\_verified,row[10] as accomodates,row[11] as bedrooms,row[12] as bathrooms,row[13] as beds,row[14] as price,toFloat(row[15]) as score

WHERE labels=":Listing"

CREATE (:Listing{id:id, name: name, url:url,picture\_url:picture\_url,host\_identity\_verified:host\_identity\_verified,accomodates:accomodates,bedrooms:bedrooms,bathrooms:bathrooms,beds:beds,price:price,score:score});

LOAD CSV FROM 'file:///neo4J\_nodes.csv' AS row

WITH row, row[0] as id, row[1] as labels, row[2] as name, row[3] as start, row[4] as end, row[5] as type, toFloat(row[6]) as rating, row[7] as url,row[8] as picture\_url,row[9] as host\_identity\_verified,row[10] as accomodates,row[11] as bedrooms,row[12] as bathrooms,row[13] as beds,row[14] as price,toFloat(row[15]) as score

WHERE type="HAS\_AMENITY"

MATCH (l:Listing) WHERE l.id= start

MATCH (a:Amenities) WHERE a.id= end

CREATE (l)-[:HAS\_AMENITY]->(a);

LOAD CSV FROM 'file:///neo4J\_nodes.csv' AS row

WITH row, row[0] as id, row[1] as labels, row[2] as name, row[3] as start, row[4] as end, row[5] as type, toFloat(row[6]) as rating, row[7] as url,row[8] as picture\_url,row[9] as host\_identity\_verified,row[10] as accomodates,row[11] as bedrooms,row[12] as bathrooms,row[13] as beds,row[14] as price,toFloat(row[15]) as score

WHERE type="HAS\_PROPERTY\_TYPE"

MATCH (l:Listing) WHERE l.id= start

MATCH (p:property\_type) WHERE p.id= end

CREATE (l)-[:HAS\_PROPERTY\_TYPE]->(p);

LOAD CSV FROM 'file:///neo4J\_nodes.csv' AS row

WITH row, row[0] as id, row[1] as labels, row[2] as name, row[3] as start, row[4] as end, row[5] as type, toFloat(row[6]) as rating, row[7] as url,row[8] as picture\_url,row[9] as host\_identity\_verified,row[10] as accomodates,row[11] as bedrooms,row[12] as bathrooms,row[13] as beds,row[14] as price,toFloat(row[15]) as score

WHERE type="HAS\_ROOM\_TYPE"

MATCH (l:Listing) WHERE l.id= start

MATCH (r:room\_type) WHERE r.id= end

CREATE (l)-[:HAS\_ROOM\_TYPE]->(r);

LOAD CSV FROM 'file:///neo4J\_nodes.csv' AS row

WITH row, row[0] as id, row[1] as labels, row[2] as name, row[3] as start, row[4] as end, row[5] as type, toFloat(row[6]) as rating, row[7] as url,row[8] as picture\_url,row[9] as host\_identity\_verified,row[10] as accomodates,row[11] as bedrooms,row[12] as bathrooms,row[13] as beds,row[14] as price,toFloat(row[15]) as score

WHERE type="IN\_CITY"

MATCH (l:Listing) WHERE l.id= start

MATCH (c:City) WHERE c.id= end

CREATE (l)-[:IN\_CITY]->(c);

LOAD CSV FROM 'file:///neo4J\_nodes.csv' AS row

WITH row, row[0] as id, row[1] as labels, row[2] as name, row[3] as start, row[4] as end, row[5] as type, toFloat(row[6]) as rating, row[7] as url,row[8] as picture\_url,row[9] as host\_identity\_verified,row[10] as accomodates,row[11] as bedrooms,row[12] as bathrooms,row[13] as beds,row[14] as price,toFloat(row[15]) as score

WHERE type="IN\_COUNTRY"

MATCH (l:Listing) WHERE l.id= start

MATCH (c:Country) WHERE c.id= end

CREATE (l)-[:IN\_COUNTRY]->(c);

LOAD CSV FROM 'file:///neo4J\_nodes.csv' AS row

WITH row, row[0] as id, row[1] as labels, row[2] as name, row[3] as start, row[4] as end, row[5] as type, toFloat(row[6]) as rating, row[7] as url,row[8] as picture\_url,row[9] as host\_identity\_verified,row[10] as accomodates,row[11] as bedrooms,row[12] as bathrooms,row[13] as beds,row[14] as price,toFloat(row[15]) as score

WHERE type="IN\_STATE"

MATCH (l:Listing) WHERE l.id= start

MATCH (s:State) WHERE s.id= end

CREATE (l)-[:IN\_STATE]->(s);

**LOAD CSV FROM 'file:///neo4J\_nodes.csv' AS row**

WITH row, row[0] as id, row[1] as labels, row[2] as name, row[3] as start, row[4] as end, row[5] as type, toFloat(row[6]) as rating, row[7] as url,row[8] as picture\_url,row[9] as host\_identity\_verified,row[10] as accomodates,row[11] as bedrooms,row[12] as bathrooms,row[13] as beds,row[14] as price,toFloat(row[15]) as score

WHERE type="RATED"

MATCH (u:User) WHERE u.id= start

MATCH (l:Listing) WHERE l.id= end

CREATE (u)-[:RATED{rating: rating}]->(l);

Recommendations

……………………………

MATCH(u:User{name:'10952'})-[:RATED]->(s:Listing)-[:HAS\_AMENITY]->(c:Amenities)<-[:HAS\_AMENITY]-(z:Listing)

WHERE NOT EXISTS ((u)-[:RATED]->(z))

WITH s, z, COUNT(c) AS intersection

MATCH (s)-[:HAS\_AMENITY]->(sc:Amenities)

WITH s, z, intersection, COLLECT(sc.name) AS s1

MATCH (z)-[:HAS\_AMENITY]->(zc:Amenities)

WITH s, z, s1, intersection, COLLECT(zc.name) AS s2

WITH s, z, intersection, s1+[x IN s2 WHERE NOT x IN s1] AS union, s1, s2

RETURN s.name as UserListing, z.name as Recommendate, s1 as UserListingAmenities, s2 as RecommendateListingAmenities, ((1.0\*intersection)/SIZE(union)) AS jaccard ORDER BY jaccard DESC;

MATCH (p1:User {name:"10952"})-[x:RATED]->(m:Listing)<-[y:RATED]-(p2:User) WITH COUNT(m) AS numbershows, SUM(x.rating \* y.rating) AS xyDotProduct,SQRT(REDUCE(xDot = 0.0, a IN COLLECT(x.rating) | xDot + a^2)) AS xLength,SQRT(REDUCE(yDot = 0.0, b IN COLLECT(y.rating) | yDot + b^2)) AS yLength,p1, p2 RETURN p1.name as SelectedUser, p2.name as SimilarUser, xyDotProduct / (xLength \* yLength) AS sim ORDER BY sim DESC;

MATCH (:User)-[r:RATED]-(:Listing)

DELETE r;