USE-CASE 3

Jessey who is traveler missed the carpool pickup rejects payment refund and request to assign and carpooling based on her new location.

Step -1 Loads the data from CSV file

LOAD CSV WITH HEADERS FROM

'file:///MOCK\_DATA.csv' AS line

Create (u:User {username:line.Username, first\_name: line.first\_name,lastname: line.last\_name,gender: line.Gender,

TripType:line.TripType,NumOfPasngrs:line.NumOfSeats,CityName:line.HomeCity,State:line.State,Country:line.Country,latitude:line.latitude,longitude:line.longitude})

MERGE (l:Location {CityName:line.DestinationCity,State:line.DestinationState,

Country:line.DestinationState,latitude:line.DestinationLat,longitude:line.DestinationLong})

Step-2 Create relationship between each user and other nodes

MERGE (u)-[:Distance {Distance:0}]->(l)

Step-3 Created Indexes

CREATE INDEX ON :User(HomeCity);

Then we set the distance property between all cities to be a point made from the latitude and longitude

Step-4

MATCH (n:User)-[t:Distance ]->(m:Location)

WITH

point({ longitude: toInteger(n.longitude), latitude: toInteger(n.latitude) }) AS p1,

point({ longitude: toInteger(m.longitude), latitude: toInteger(m.latitude)}) AS p2,t

SET t.Distance=round(distance(p1,p2), 2)

RETURN round(distance(p1,p2), 2) as Dist,t

Let’s find neighboring users

STEP-5 Now let’s find Union City, CA and the neighboring cities within 10 kilometers:

MATCH (c:User)-[t:Distance ]->(s:Location)

WHERE c.CityName<>s.CityName and t.Distance>120000

RETURN c.username,c.CityName as FromCity,s.CityName as ToCity,t.Distance as Distance

Graphical user interface, text, application, email

Description automatically generated