Name: Asif Sayyad

# Neo4j

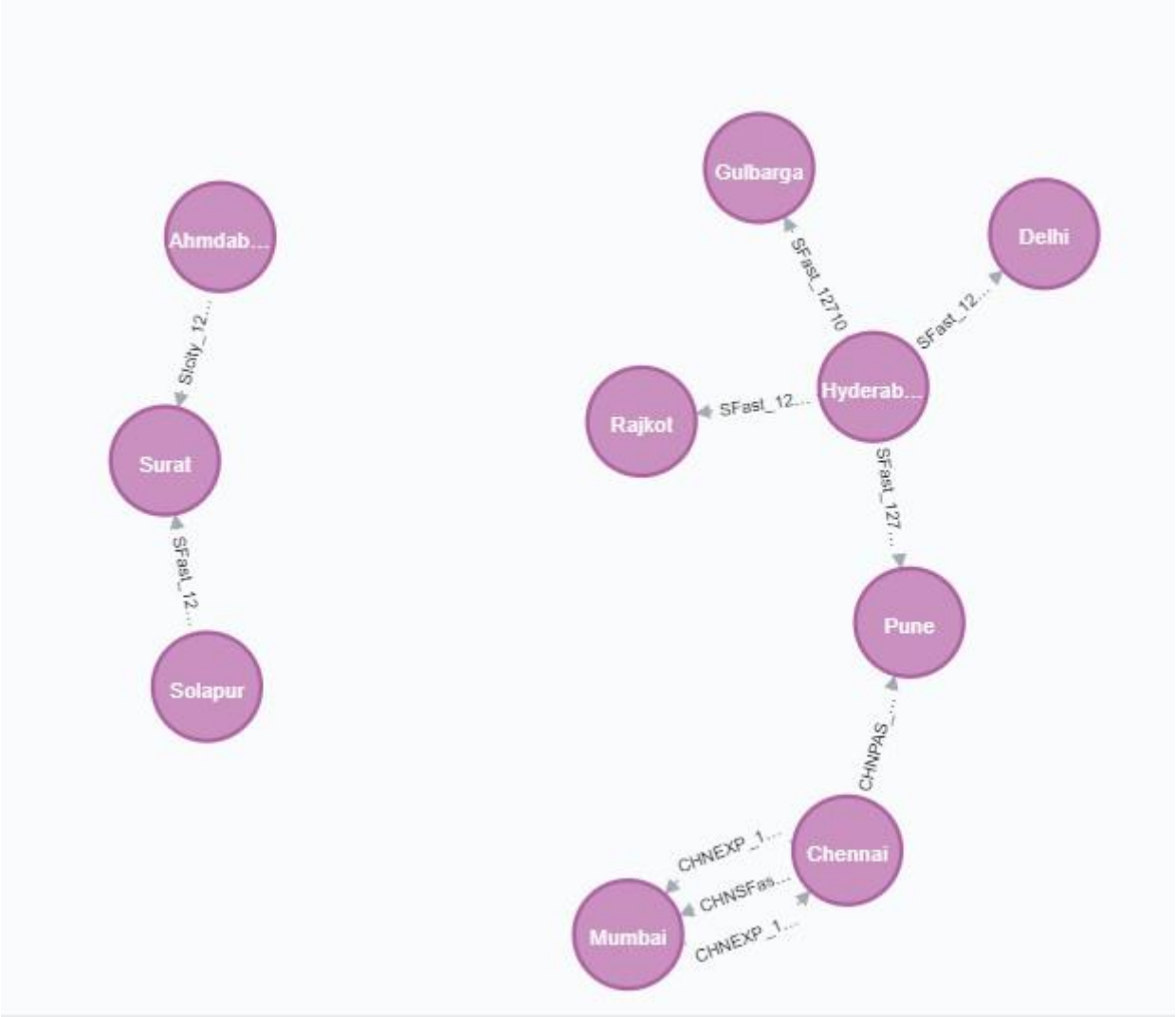
1. **Create 10 nodes**

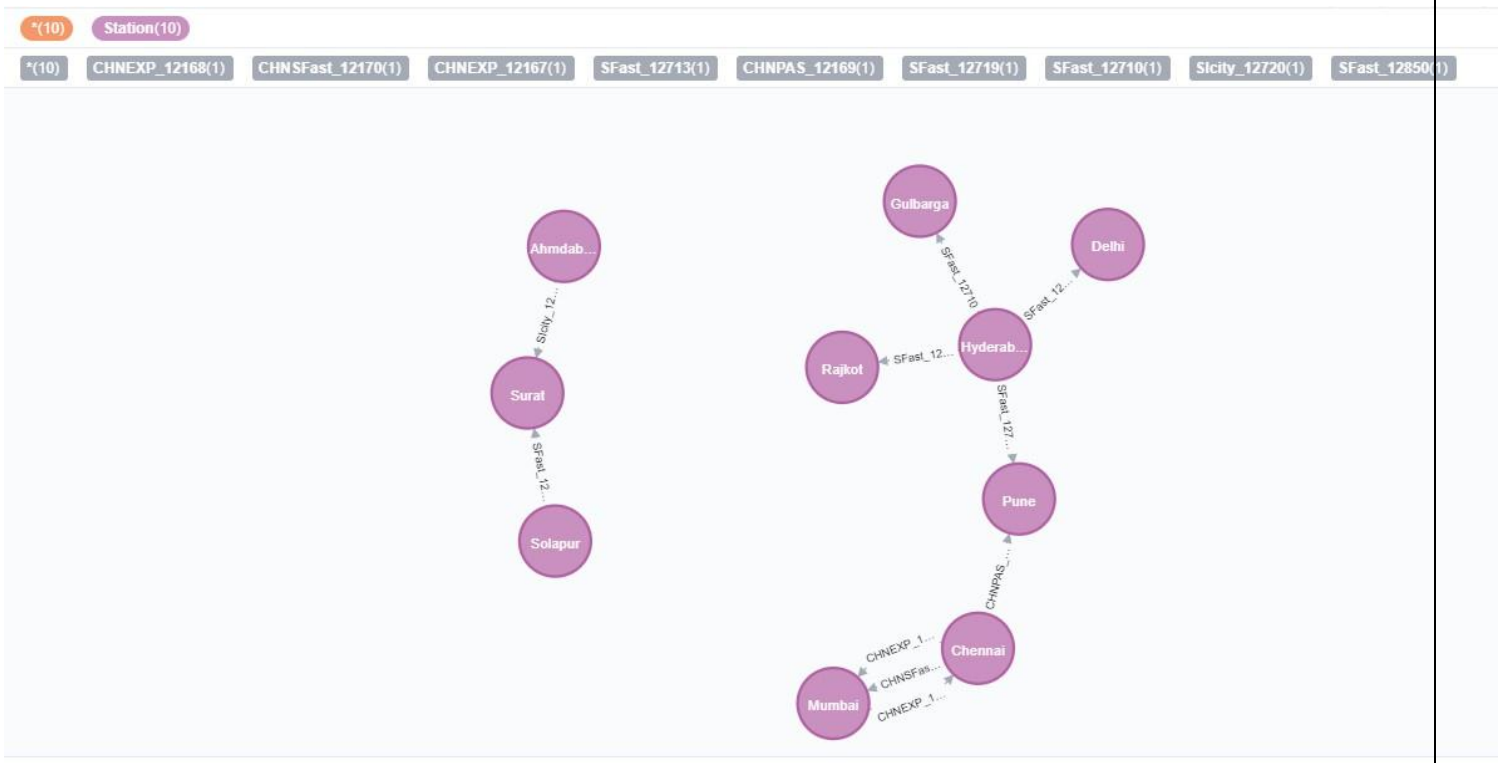
```
CREATE (Chennai:Station {name: "Chennai", stncode:"MAS"})
CREATE (Mumbai:Station {name: "Mumbai", stncode:"CST"})
CREATE (Pune:Station {name: "Pune", stncode:"PNE"})
CREATE (Solapur:Station {name: "Solapur", stncode:"SUR"})
CREATE (Delhi:Station {name: "Delhi", stncode:"DHY"})
CREATE (Hyderabad:Station {name: "Hyderabad", stncode:"HYD"})
CREATE (Gulbarga:Station {name: "Gulbarga", stncode:"GBA"})
CREATE (Ahmdabad:Station {name: "Ahmdabad", stncode:"AMD"})
CREATE (Surat:Station {name: "Surat", stncode:"SRT"})
CREATE (Rajkot:Station {name: "Rajkot", stncode:"RJKT"})
```

2. **Create links**

```
MERGE (Chennai)-[:CHNEXP_12167]-(Mumbai)
MERGE (Mumbai)-[:CHNEXP_12168]-(Chennai)
MERGE (Chennai)-[:CHNPAS_12169]-(Pune)
MERGE (Chennai)-[:CHNSFast_12170]-(Mumbai)
MERGE (Hyderabad)-[:SFast_12710]-(Gulbarga)
MERGE (Hyderabad)-[:SFast_12712]-(Rajkot)
MERGE (Hyderabad)-[:SFast_12713]-(Pune)
MERGE (Hyderabad)-[:SFast_12719]-(Delhi)
MERGE (Ahmdabad)-[:SIcity_12720]-(Surat)
MERGE (Solapur)-[:SFast_12850]-(Surat)
```

**OUTPUT:**


Ahmdab...
SFast_12...
Surat
SFast_12...
Solapur
Gulbarga
SFast_12710
Delhi
SFast_12...
Rajkot
SFast_12...
Hyderab...
SFast_127...
Pune
CHNPas...
CHNEXP_1...
Chennai
CHNSFas...
CHNEXP_1...
Mumbai

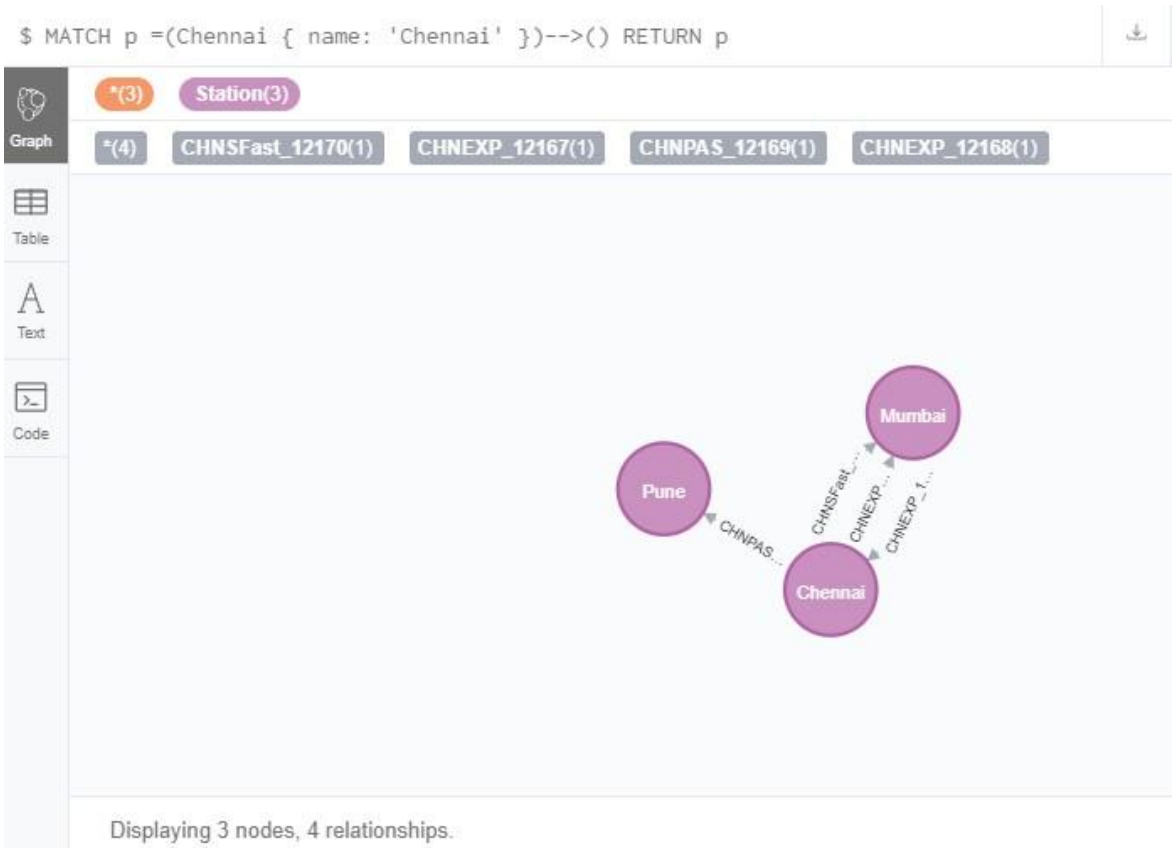



Displaying 10 nodes, 10 relationships.

## Cypher Queries:

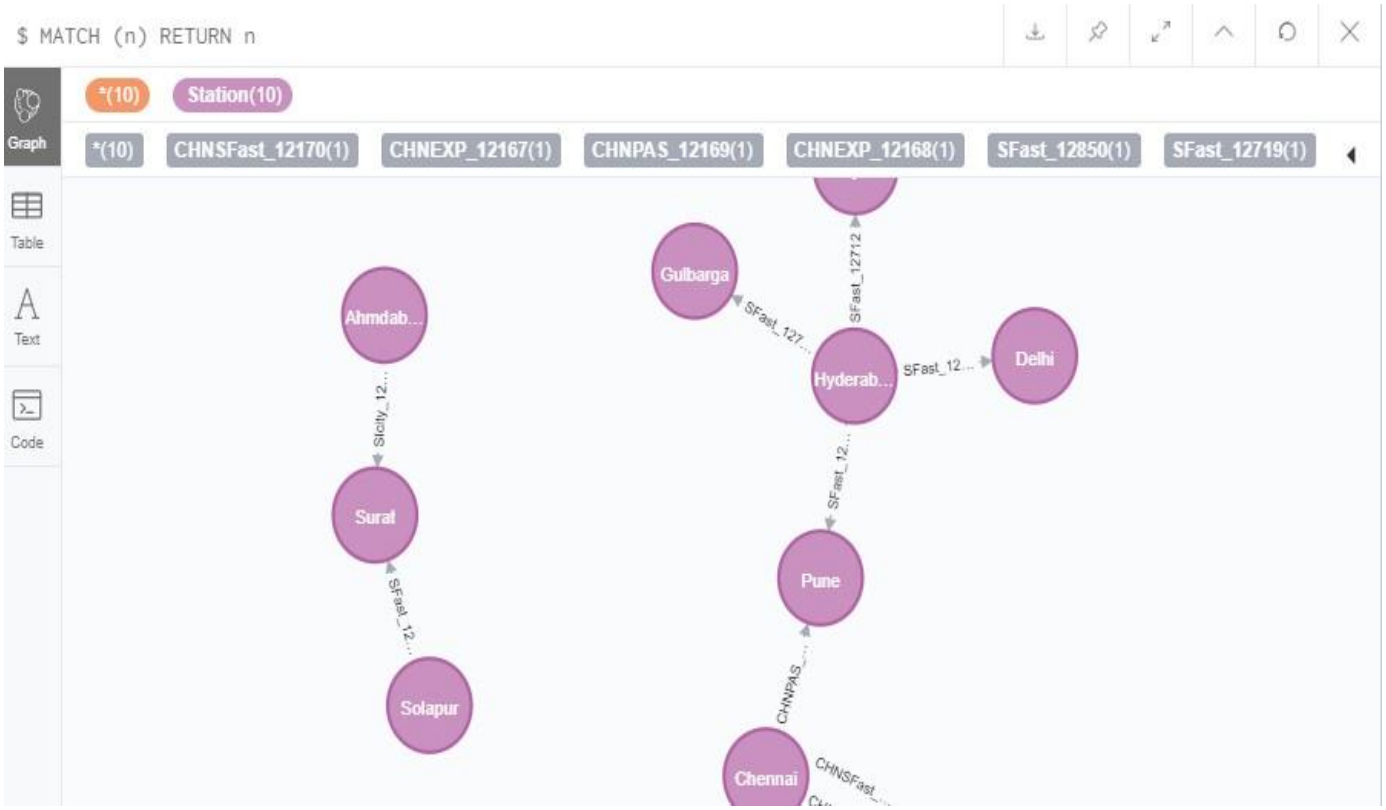
### 1) Train having source or destination as Chennai

MATCH p =(Chennai { name: 'Chennai' })-->() RETURN p



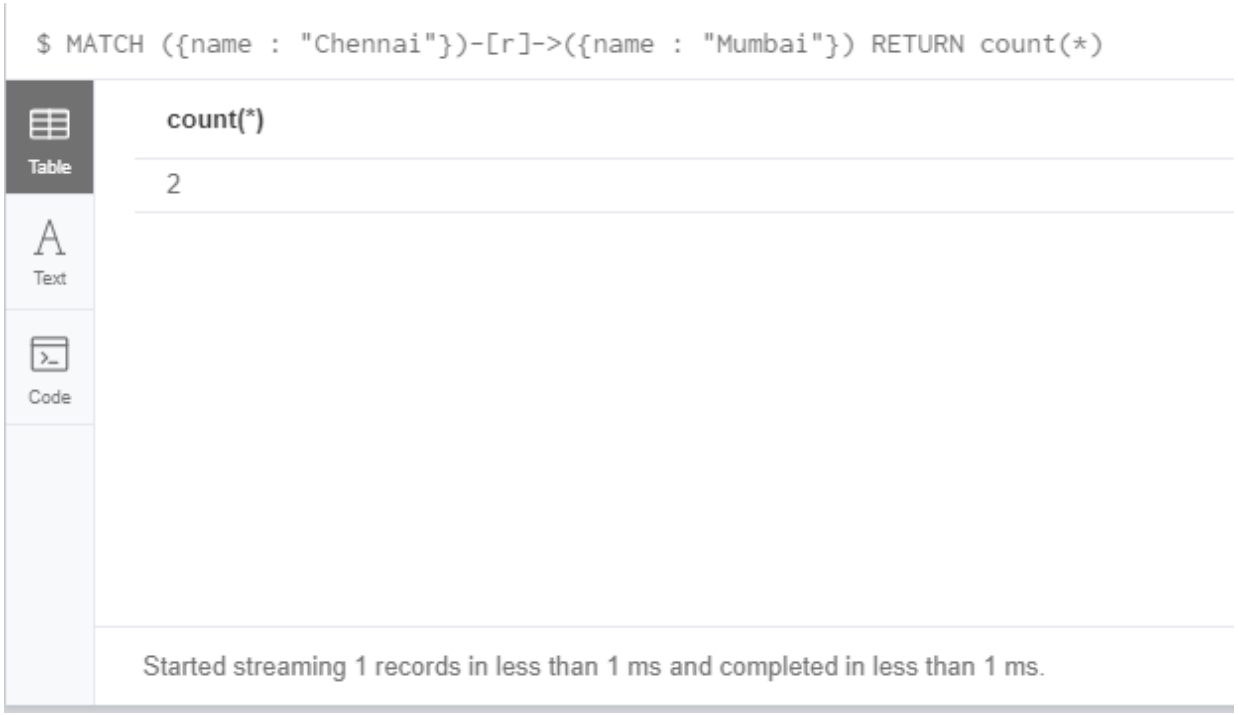
**2) Find train connectivity among all stations**

MATCH (n) RETURN n



**3) Count number of trains between Chennai to Mumbai**

```
MATCH ({name : "Chennai"})-[r]->({name : "Mumbai"})
RETURN count(*)
```



**4) Find nodes and nodeid**

MATCH (n:Station) RETURN n.name,ID(n) LImit 5

```
$ MATCH (n:Station) RETURN n.name,ID(n) LImit 5
```

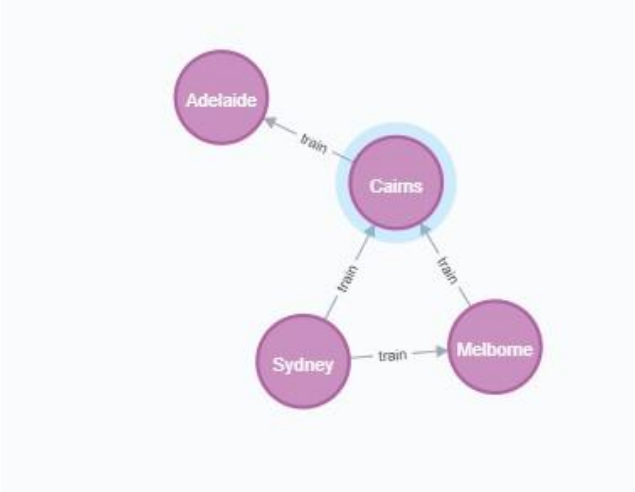
Table | Text | Code

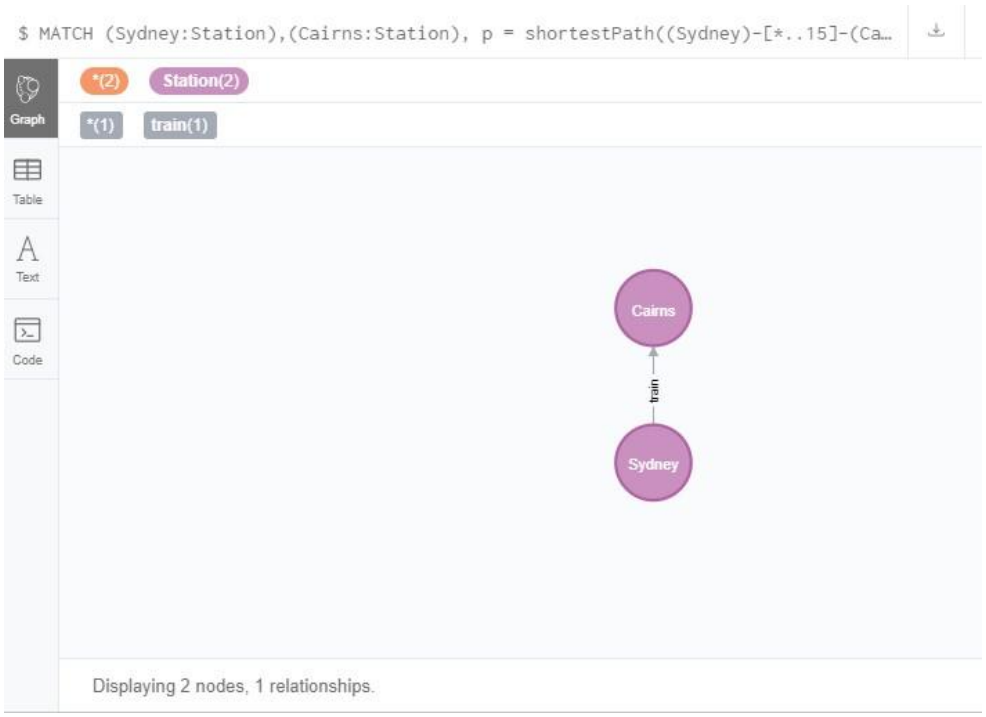
| n.name | ID(n) |
|---|---|
| "Sydney" | 37 |
| "Melborne" | 38 |
| " Cairns " | 39 |
| " Adelaide " | 40 |
| "Chennai" | 54 |

Started streaming 5 records in less than 1 ms and completed in less than 1 ms.

**4) Find Shortest path among two stations( that's we will prefer direct train instead of break journey)**

```
CREATE (Sydney:Station {name: "Sydney", stncode:"SYD"})
CREATE (Melborne:Station {name: "Melborne", stncode:"MLB"})
CREATE (Cairns:Station {name: " Cairns ", stncode:"CAI"})
CREATE (Adelaide:Station {name: " Adelaide ", stncode:"ADE"})
MERGE (Sydney)-[:train]-( Melborne)
MERGE (Melborne)-[:train]-(Cairns)
MERGE (Cairns)-[:train]-( Adelaide)
MERGE (Melborne)-[:train]-( Sydney)
MERGE (Sydney)-[:train]-( Cairns)
MATCH (Sydney:Station),(Cairns:Station),
p = shortestPath((Sydney)-[*..15]-(Cairns))
WHERE id(Sydney) = 42 AND id(Cairns) = 44
RETURN p
```






**5) City that don't have train route**

MATCH (Pondey:Station) WHERE not ((Pondey)--()) RETURN Pondey;