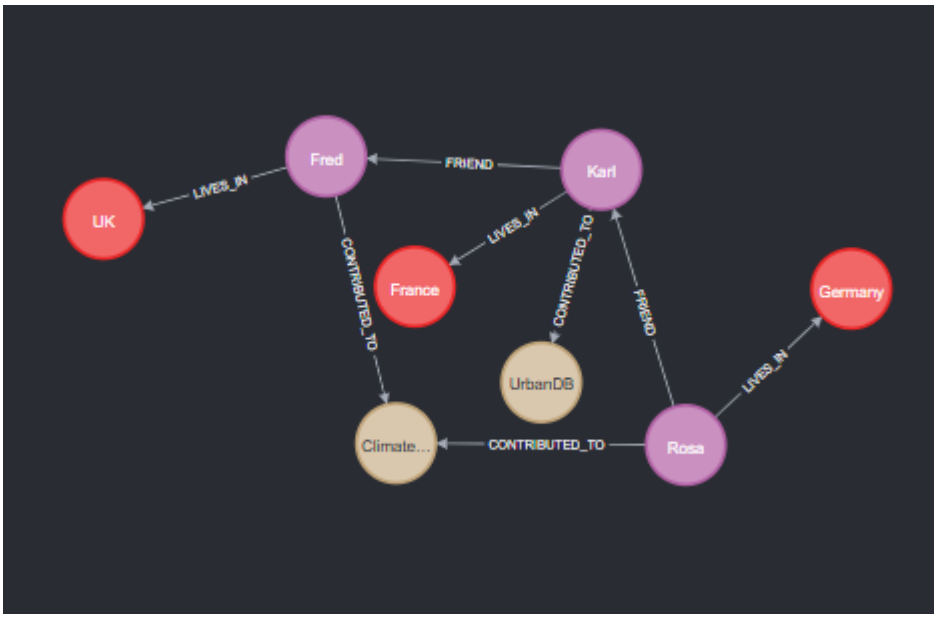
23SA63 – BIGDATA & MODERN DATABASE SYSTEMS

PACKAGE

GRAPH DATABASE – Neo4j

```
// Create Person nodes
CREATE (rosa:Person {name: "Rosa", gender: "F"})
CREATE (karl:Person {name: "Karl", gender: "M"})
CREATE (fred:Person {name: "Fred", gender: "M"})

// Create Place nodes
CREATE (berlin:Place {city: "Berlin", country: "Germany", lives_since: 2020})
CREATE (paris:Place {city: "Paris", country: "France", lives_since: 1980})
CREATE (london:Place {city: "London", country: "UK"})

// Create FRIEND relationships
CREATE (rosa)-[:FRIEND]->(karl)
CREATE (karl)-[:FRIEND]->(fred)

// Create LIVES_IN relationships
CREATE (rosa)-[:LIVES_IN]->(berlin)
CREATE (karl)-[:LIVES_IN]->(paris)
CREATE (fred)-[:LIVES_IN]->(london)

// Create Project nodes and contributed_to relationships
CREATE (proj1:Project {name: "ClimateGraph"})
CREATE (proj2:Project {name: "UrbanDB"})

CREATE (rosa)-[:CONTRIBUTED_TO {DBs_used: ["Neo4j", "MongoDB"]}]->(proj1)
CREATE (karl)-[:CONTRIBUTED_TO {DBs_used: ["PostgreSQL"]}]->(proj2)
CREATE (fred)-[:CONTRIBUTED_TO {DBs_used: ["Neo4j", "Redis"]}]->(proj1)
```



MATCH (p:Person {name: "Rosa"})

SET p.dob = date("1990-05-12")

Set 1 property, completed after 3 ms.

MATCH (n)

SET n.years_experience = 5

Set 11 properties, completed after 2 ms.

MATCH (p:Person)-[:LIVES_IN]->(pl:Place {city: "Berlin"})

RETURN p.name, p.gender

p.name	p.gender
"Rosa"	"F"

MATCH (rosa:Person {name: "Rosa"})-[:FRIEND]->(:Person)-[:FRIEND]->(fof:Person)

```
WHERE fof.name <> "Rosa"
RETURN DISTINCT fof.name, fof.gender
```

fof.name	fof.gender
"Fred"	"M"

```
MATCH (p:Person)
WHERE p.name STARTS WITH "Ka"
RETURN p.name
```

p.name
"Karl"

```
MATCH (p:Person)
WHERE p.name CONTAINS "os"
RETURN p.name
```

p.name
"Karl"

```
MATCH (p:Person)
WHERE NOT p.name ENDS WITH "d"
RETURN p.name
```

p.name
"Tom Hanks"
"Robert Zemeckis"
"Rosa"
"Karl"

MATCH (p:Person)-[:LIVES_IN]->(pl:Place {city: "Berlin"})

WHERE p.name IN ["Rosa", "Karl"]

RETURN p.name

p.name
"Rosa"

MATCH (p:Person)-[:LIVES_IN]->(pl:Place {city: "Berlin"})

MATCH (p)-[:FRIEND]->(f1)-[:FRIEND]->(f2)

RETURN DISTINCT f1.name AS Friend, f2.name AS FriendOfFriend

Friend	FriendOfFriend
"Karl"	"Fred"

MATCH (p:Person)-[:LIVES_IN]->(pl:Place)

WITH pl.city AS city, COUNT(p) AS population

RETURN city, population

ORDER BY population DESC

city	population
"Berlin"	1
"Paris"	1
"London"	1

MATCH (p:Person)

WHERE p.years_experience >= 9 AND p.years_experience <= 12

RETURN p.name, p.years_experience

(no changes, no records)

MATCH (p:Person)-[r:CONTRIBUTED_TO]->(:Project)

WHERE "Neo4j" IN r.DBs_used

RETURN DISTINCT p.name

p.name
"Rosa"
"Fred"

MATCH (p:Person)-[r:CONTRIBUTED_TO]->(:Project)

WHERE "MongoDB" IN r.DBs_used

MATCH (p)<-[:FRIEND]-(f:Person)

RETURN DISTINCT f.name

(no changes, no records)

```
MATCH (rosa:Person {name: "Rosa"})-[:FRIEND]->(f:Person)
WHERE NOT (f)-[:CONTRIBUTED_TO]->(:Project)
RETURN f.name
```

(no changes, no records)

```
MATCH (:Person)-[:CONTRIBUTED_TO]->(p:Project {name: "project A"})
RETURN COUNT(*) AS member_count
```

member_count
0

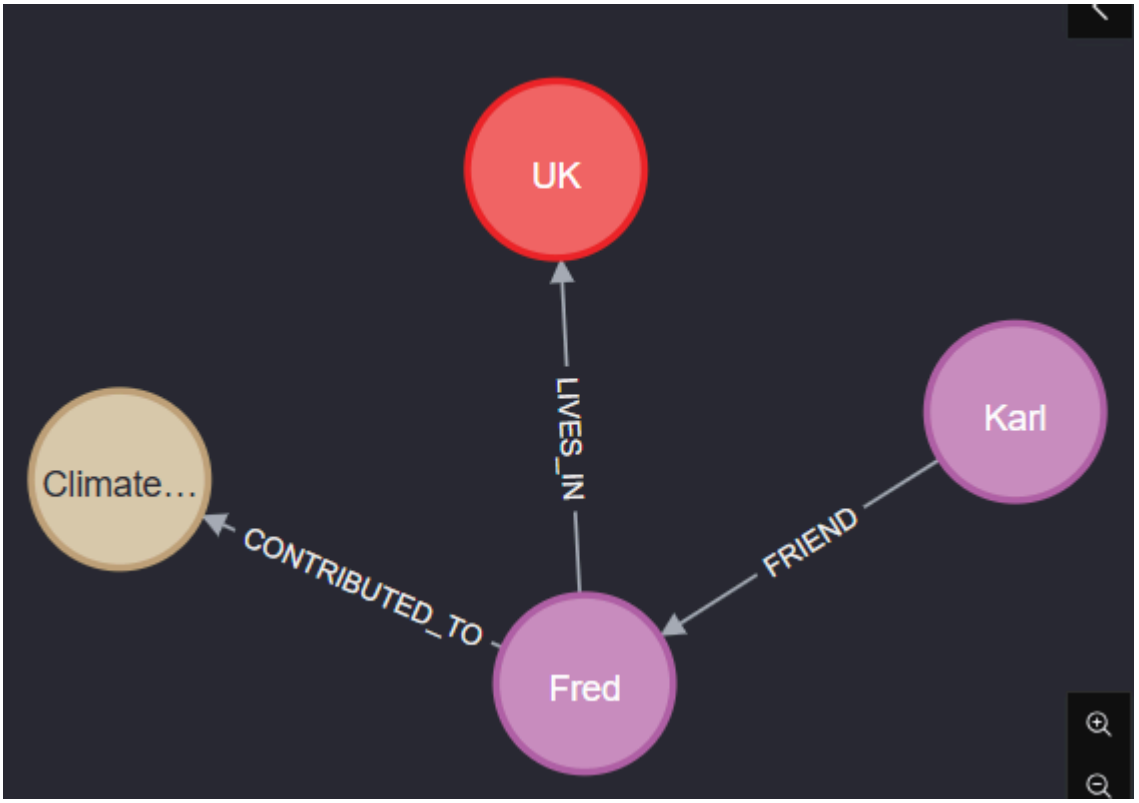
```
MATCH (:Person {name: "Rosa"})-[:FRIEND]->(f:Person)
RETURN f.name
ORDER BY f.name DESC
```

f.name
"Karl"

```
MATCH (karl:Person {name: "Karl"})-[:FRIEND]->(f:Person)-[r:CONTRIBUTED_TO]->(:Project)
WHERE "Neo4j" IN r.DBs_used
RETURN DISTINCT f.name
```

f.name
"Fred"

```
MATCH (fred:Person {name: "Fred"})-[r]-(n)
RETURN fred, r, n
```



MATCH (p:Person)

RETURN DISTINCT p.name

p.name
"Tom Hanks"
"Robert Zemeckis"
"Rosa"
"Karl"
"Fred"

// UNION DISTINCT

```
MATCH (p:Person)
RETURN p.name AS label
UNION
MATCH (pl:Place)
RETURN pl.city AS label
```

label
"Tom Hanks"
"Robert Zemeckis"
"Rosa"
"Karl"
"Fred"
"Berlin"

"Paris"
"London"

```
CALL {
  MATCH (p:Person)
  RETURN p.name AS label
  UNION ALL
  MATCH (pl:Place)
  RETURN pl.city AS label
}
RETURN label, COUNT(*) AS occurrences
ORDER BY occurrences DESC
```

	label	occurrences
1	"Tom Hanks"	1
2	"Robert Zemeckis"	1
3	"Rosa"	1
4	"Karl"	1
5	"Fred"	1
6	"Berlin"	1
7	"Paris"	1
8	"London"	1

```
MATCH (p:Person)
SET p.age = 44,
    p.email = p.name + "@gmail.com"
```

Set 10 properties, completed after 3 ms.

```
MATCH (:Person {name: "Rosa"})-[:FRIEND]->(f:Person)
WHERE f.age > 30
RETURN f.name, f.age
```

f.name	f.age
"Karl"	44

MATCH (p:Person)-[:LIVES_IN]->(pl:Place {city: "Berlin"})

WHERE p.name STARTS WITH "R"

RETURN p.name

p.name
"Rosa"

MATCH (p:Person)

WHERE p.email =~ ".*@gmail\\.com"

RETURN p.name, p.email

	p.name	p.email
1	"Tom Hanks"	"Tom Hanks@gmail.com"
2	"Robert Zemeckis"	"Robert Zemeckis@gmail.com"
3	"Rosa"	"Rosa@gmail.com"
4	"Karl"	"Karl@gmail.com"
5	"Fred"	"Fred@gmail.com"

MATCH (p:Person)-[:FRIEND]->(f:Person)

WITH p, COUNT(f) AS friend_count

WHERE friend_count > 1

```
RETURN p.name, friend_count
```

(no changes, no records)

```
MATCH (p:Person)-[:LIVES_IN]->(pl:Place {city: "London"})
RETURN DISTINCT p.name
```

p.name
"Fred"

```
MATCH (p:Person)
RETURN
  COUNT(p) AS total_persons,
  MAX(p.age) AS max_age,
  MIN(p.age) AS min_age,
  percentileCont(p.age, 0.5) AS median_age,
  percentileDisc(p.age, 0.75) AS percentile_75,
  SUM(p.years_experience) AS total_experience,
  STDEV(p.years_experience) AS experience_stdev
```

total_persons	max_age	min_age	median_age	percentile_75	total_experience	experience_stdev
5	44	44	44	44	25	0.0

```
MATCH (n)
WHERE n.age > 50
SET n.category = 'senior'
WITH n
WHERE n.age >= 30 AND n.age <= 50
SET n.category = 'middle age'
WITH n
WHERE n.age < 30
```

SET n.category = 'junior'

```
(no changes, no records)
```

:param people => [

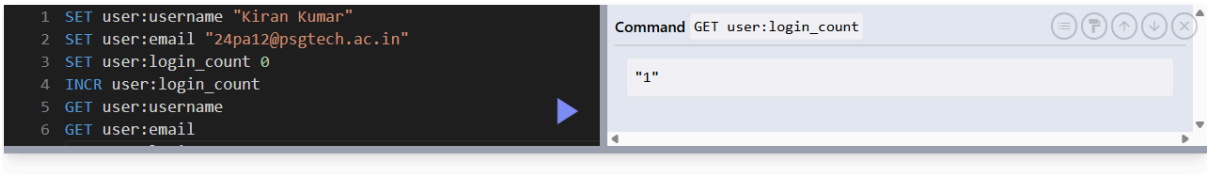
  {name: "Tom", age: 54},

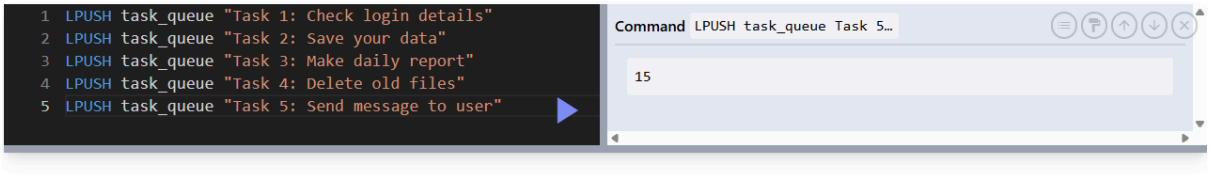
  {name: "Alex", age: 44},

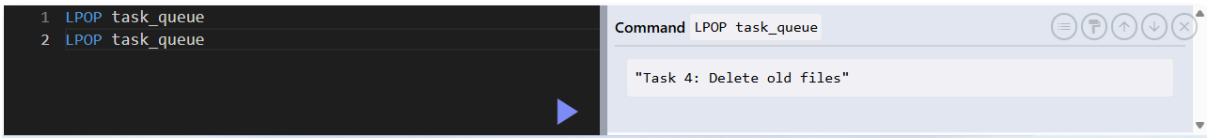
  {name: "Susan", age: 20}

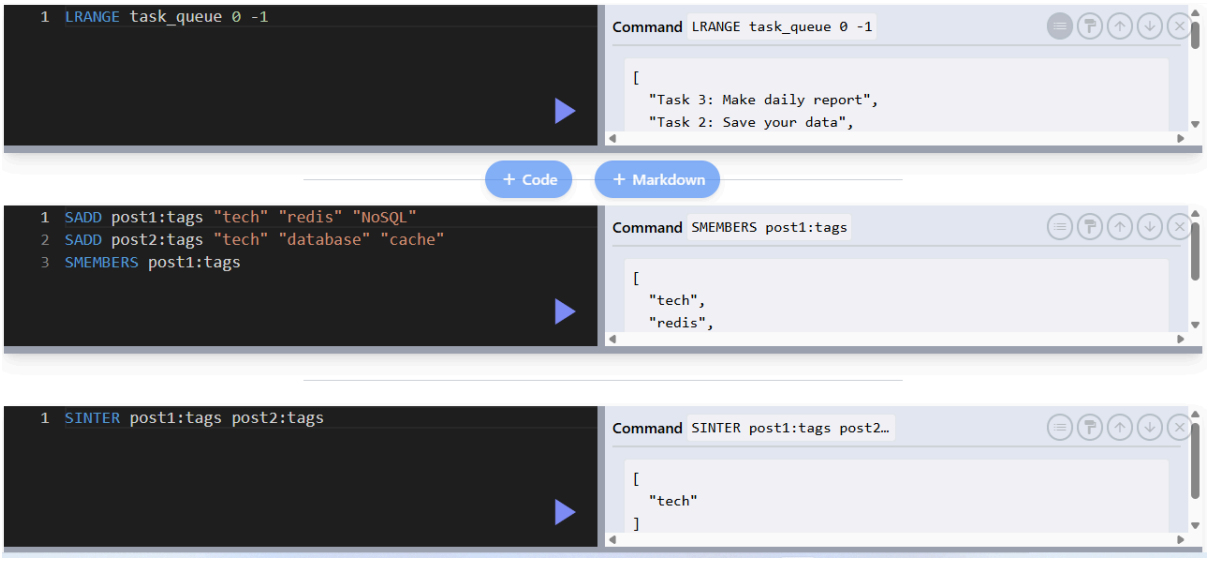
]

REDIS – Redis Cloud



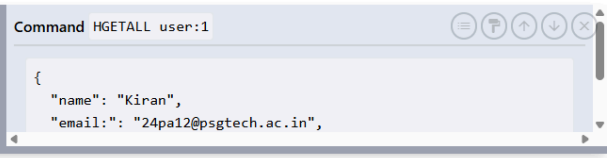






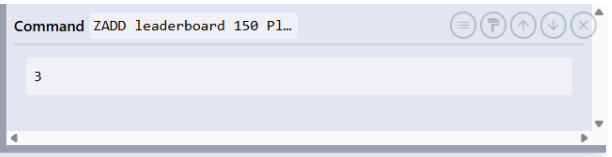
```
SUNION post1:tags post2:tags
```


```
HSET user:1 name "Kiran" email: "24pa12@psgtech.ac.in"
age "21"
HGETALL user:1
```




```
HSET user:1 email "newmail@example.com"
HDEL user:1 age
HLEN user:1
```


```
ZADD leaderboard 150 "Player1" 200 "Player2" 170
"Player3"
```



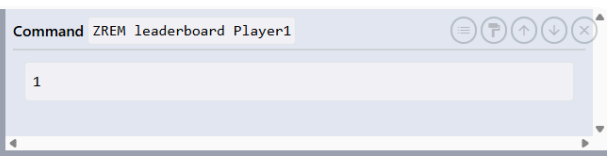

```
ZREVRANGE leaderboard 0 2 WITHSCORES
```


```
ZINCRBY leaderboard 20 "Player1"
```


```
ZREVRANGE leaderboard 0 -1 WITHSCORES
```


```
ZRANGE leaderboard 0 0 WITHSCORES
```


```
ZREM leaderboard "Player1"
```



```
SUBSCRIBE chat:room1
```

Command `SUBSCRIBE chat:room1`

CHANNEL	MESSAGE	TIME AGO
chat:room1	"Hello from user A"	2m

```
PUBLISH chat:room1 "Hello from user A"
```

Command `PUBLISH chat:room1 Hell…`

```
1
```

```
XADD chat:room1 * sender "UserA" message "Hello everyone"
XREAD COUNT 5 STREAMS chat:room1 0
```

Command `XREAD COUNT 5 STREAMS c…`

```
[
  [
    "chat:room1",
```

```
SUBSCRIBE chat:room2
```

Command `SUBSCRIBE chat:room2`

CHANNEL	MESSAGE	TIME AGO
chat:room2	"Hi from user B"	2m

```
PUBLISH chat:room2 "Hi from user B"
```

Command `PUBLISH chat:room2 Hi f…`

```
1
```

```
XADD chat:room2 * sender "UserB" message "Welcome to room 2"
```

Command `XADD chat:room2 * sende…`

```
"1761124227854-0"
```

```
XREAD COUNT 5 STREAMS chat:room2 0
```

Command `XREAD COUNT 5 STREAMS c…`

```
[
  [
    "chat:room2",
```

+ Code + Markdown

```
PFADD users:asia "user1" "user2" "user3"
```

Command `PFADD users:asia user1 …`

```
0
```

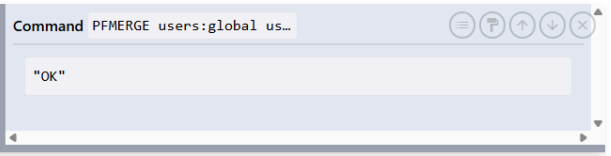
```
PFADD users:europe "user4" "user5"
```

Command `PFADD users:europe user…`

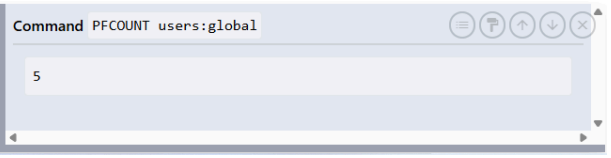
```
0
```

```
PFADD users:america "user1" "user5"
```

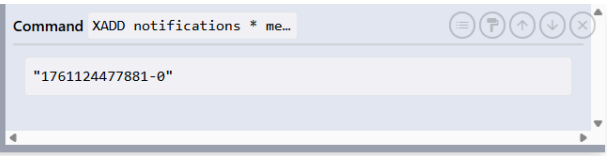

```
PFMERGE users:global users:asia users:europe
users:america
```



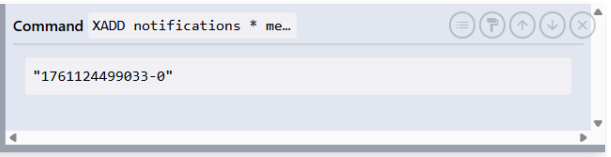

```
PFCOUNT users:global
```




```
XADD notifications * message_id "msg1" content "promo
1" recipient "UserA"
```




```
XADD notifications * message_id "msg2" content "Alert"
recipient "UserB"
```




```
XADD notifications * message_id "msg3" content "Update"
recipient "UserC"
```