Neo4j-何智鹏-2112233062

实验内容

1. Provide the commands used to load your data into your database（提供用于将数据加载到数据库中的命令）

C++ | 复制代码

```
LOAD CSV WITH HEADERS FROM "file:/apartment.csv" AS row CREATE (:apartment {name:
row.name, established: toInteger(row.established),beds:toInteger(row.beds)})

LOAD CSV WITH HEADERS FROM "file:/collegeS20.csv" AS row
CREATE(:college{name:row.name,established:toInteger(row.established),membership:toIn
teger(row.membership),beds:toInteger(row.beds),location:row.location})

LOAD CSV WITH HEADERS FROM "file:///distance.csv" AS row MATCH (a{name:row.from}),
(b{name:row.to}) MERGE (a)-[:DISTANCE {meters: toInteger(row.meters)}]->(b)

LOAD CSV WITH HEADERS FROM "file:///distance.csv" AS row MATCH (a{name:row.from})
OPTIONAL MATCH (b {name: row.to})merge (b)-[:DISTANCE {meters:
toInteger(row.meters)}]->(a)
```

2. What year was RGA established?（RGA成立于哪一年？）

	c.established
1	1999

3. Grouping the colleges by location, what is the average year that residential colleges were established? Order by location, descending. Return the location and the average year, rounded to the nearest integer.（按地点对学院进行分组，寄宿学院成立的平均年份是多少？按位置降序排列。返回位置和平均年份，四舍五入到最接近的整数。）

```
MATCH (c:college)
WITH c.location AS location, AVG(c.established) AS avg_established
RETURN location, round(avg_established) AS avg_year
ORDER BY location DESC
```

```
MATCH (c:college)
WITH c.location AS location, AVG(c.established) AS avg_established
RETURN location, round(avg_established) AS avg_year
ORDER BY location DESC
```

Table

	location	avg_year
1	"south"	1962.0
2	"north"	1983.0

Text

Code

4. What is the maximum number of beds in any residence?（最大床位是多少？）

```
neo4j$ MATCH (r:college) RETURN MAX(r.beds) AS max_beds
```

Table

	max_beds
1	324

Text

Code

5. What is/are the names of the residences with the maximum number of beds? Return the names in alphabetical order.（床位最多的宿舍名称是什么？按字母顺序返回姓名。）

```
MATCH (r:college)
WITH MAX(r.beds) AS max_beds
MATCH (r:college {beds: max_beds})
RETURN r.name AS college_name
ORDER BY college_name ASC
```

```
MATCH (r:college)
WITH MAX(r.beds) AS max_beds
MATCH (r:college {beds: max_beds})
RETURN r.name AS college_name
ORDER BY college_name ASC
```

Table | Text | Code

	college_name
1	"Duncan"
2	"McMurtry"

6. What college(s) has the smallest ratio of beds to members?（哪个学院的床位与会员比例最小？）

C++ | 复制代码

```
MATCH (c:college)
WITH c, toFloat(c.beds) / toFloat(c.membership) AS bed_member_ratio
ORDER BY bed_member_ratio ASC
LIMIT 1
RETURN c.name AS college_name, bed_member_ratio
```

```
MATCH (c:college)
WITH c, toFloat(c.beds) / toFloat(c.membership) AS bed_member_ratio
ORDER BY bed_member_ratio ASC
LIMIT 1
RETURN c.name AS college_name, bed_member_ratio
```

Table | Text

	college_name	bed_member_ratio
1	"Will Rice"	0.5925

7. The “degree” of a node in a network is the number of edges connected to the node. This is a simple metric of how important a node is within a network. Compute the degree of each node, and store it as a new property, named degree for each node. Provide your code to compute and store this value, as well as a separate query to return the degree for Will Rice college, using the new property.（网络中节点的“度”是连接到该节点的边的数量。这是一个简单的衡量节点在网络中的重要性的指标。计算每个节点的

度，并将其存储为一个新属性，为每个节点命名为度。提供计算和存储该值的代码，以及使用新属性返回Will Rice学院学位的单独查询。）

C++ 复制代码

```
CALL apoc.periodic.iterate(
'MATCH (n) RETURN n',
'SET n.degree = apoc.node.degree(n)', {batchSize: 1000, parallel: true}
)
```

```
CALL apoc.periodic.iterate(
'MATCH (n) RETURN n',
'SET n.degree = apoc.node.degree(n)',
{batchSize: 1000, parallel: true}
)
```

Table Text Code

	batches	total	timeTaken	committedOperations	failedOperations	failedBatches	retries
1	1	13	0	13	0	0	0

8. What is the shortest distance between any two colleges? That is, the value of the meters property on the relationship connecting two colleges that is the smallest.（两所大学之间的最短距离是多少？）

C++ 复制代码

```
MATCH (c1:college)-[r:DISTANCE]->(c2:college)
RETURN MIN(r.meters) AS shortest_distance
```

```
MATCH (c1:college)-[r:DISTANCE]→(c2:college)
RETURN MIN(r.meters) AS shortest_distance
```

Table Text

	shortest_distance
1	47

9. What is the shortest distance between any three colleges, ignoring the direction of the DISTANCE relationships? Return the distance and the names of the colleges (in

any order). Consider colleges A, B, and C. The distance is the sum of the meters between college A and college B and the meters between college B and college C。（忽略distance关系的方向，三所大学之间的最短距离是多少？返回距离和学院名称（按任何顺序）。以学院A、B和C为例。距离是学院A和学院B之间的米数以及学院B和学院C之间的米的总和）

C++ | 复制代码

```
MATCH (a:college)-[r1:DISTANCE]-(b:college)-[r2:DISTANCE]-(c:college)
WHERE a <> b AND a <> c AND b <> c
WITH a, b, c, r1.meters + r2.meters AS distance
ORDER BY distance ASC
LIMIT 1
RETURN MIN(distance),a.name, b.name, c.name
```

```
MATCH (a:college)-[r1:DISTANCE]-(b:college)-[r2:DISTANCE]-
(c:college)
WHERE a <> b AND a <> c AND b <> c
WITH a, b, c, r1.meters + r2.meters AS distance
ORDER BY distance ASC
LIMIT 1
RETURN MIN(distance),a.name, b.name, c.name
```

Table | Text

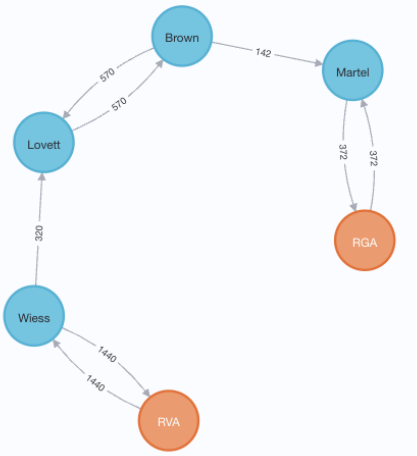
	MIN(distance)	a.name	b.name	c.name
1	166	"Lovett"	"Will Rice"	"Baker"

10. What is the shortest path from RGA to RVA? Consider paths of at most 20 nodes. Return the path (as a graph) and the length (the number edges) of the path. Include a screen shot of your path in your submission. （从RGA到RVA的最短路径是什么？考虑最多20个节点的路径。返回路径（以图形形式）和路径的长度（边数）。在提交的文件中包含路径的屏幕截图。）

C++ | 复制代码

```
MATCH path = shortestPath((a:apartment {name:"RGA"})-[*1..20]-(b:apartment
{name:"RVA"}))
RETURN path
```

```
MATCH path = shortestPath((a:apartment {name:"RGA"})-[*1..20]-(b:apartment {name:"RVA"})) RETURN path
```



11. By hand, compute the distance (sum of the meters for each relationship) along the shortest path you just found.（计算沿着刚刚找到的最短路径的距离（每个关系的米的总和））。

C++ | 复制代码

```
MATCH path = shortestPath((a:apartment {name:"RGA"})-[*1..10]-(b:apartment
{name:"RVA"})) RETURN reduce(totalDistance = 0, rel in relationships(path)
|
totalDistance + rel.meters) AS distance
```

```
$ MATCH path = shortes
```

distance
2844

12. Now, add a direct relationship from RGA to RVA with distance 3000m. Include your code here.（现在，添加从RGA到RVA的距离为3000m的直接关系。在此处包含您的代码。）

C++ | 复制代码

```
MATCH (start:apartment{name:"RGA"}), (end:apartment{name:"RVA"})
MERGE (start)-[:DISTANCE {distance: 3000}]-(end) 12
```

13. Rerun your query to find the shortest path and length. What is the length?（重新运行查询以查找最短路径和长度。长度是多少？）

最短路径为：RGA->Martel->Brown->Lovett->Wiess->RVA，距离为2844.

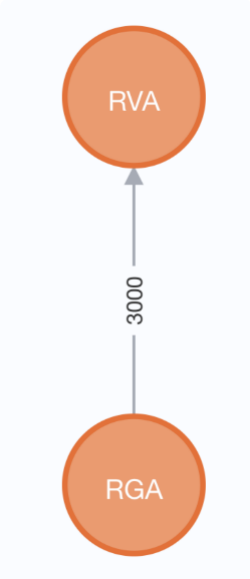
14. Are the results what you expected? Why or why not?（结果是你所期望的吗？为什么？）

是我期望的，因为新增的路径长度小于原先的长度。

15. Use the shortestPath algorithm https://neo4j.com/docs/graph algorithms/current/algorithms/shortest-path/ to calculate the shortestPath from RGA to RVA. Return the names of the nodes in the shortest path, and the cumulative cost (in meters). Show your code and results.（使用shortestPath算法以计算从RGA到RVA的最短路径。返回最短路径中节点的名称以及累计成本（以米为单位）。显示代码和结果。）

C++ | 复制代码

```
MATCH (startNode:apartment {name:"RGA"}), (endNode:apartment{name:"RVA"})
CALL apoc.algo.dijkstra(startNode, endNode, 'RELATIONSHIP_TYPE', 'distance'
)
YIELD path, weight
RETURN path, weight
```

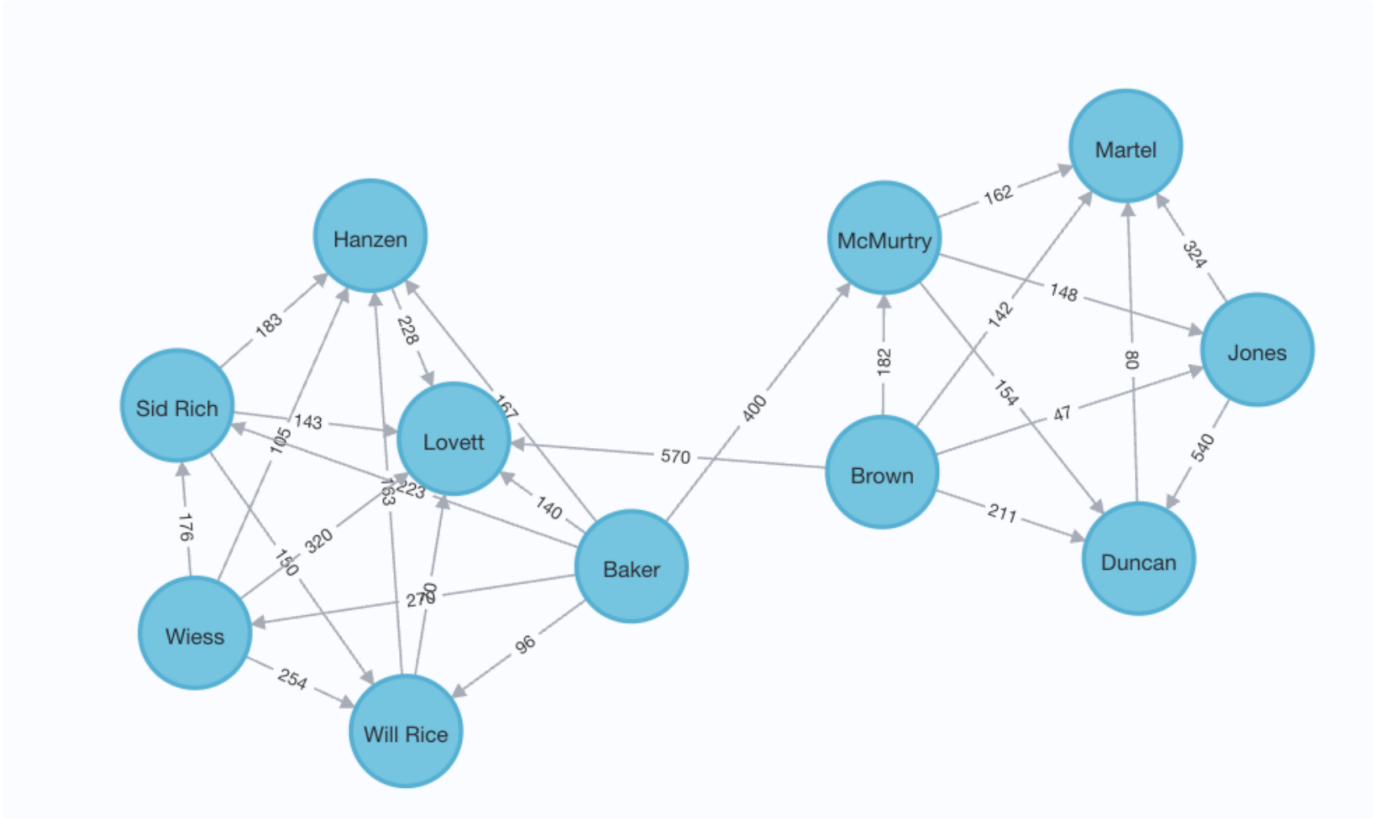


16. The file distanceMore.csv contains additional distances between residences. Load these edges into your graph now. View the graph and rearrange the nodes so it is easy to differentiate between the north and south campuses. Submit a picture of

your graph. You should notice 2 clusters of colleges, with a few connections between the colleges。（文件distanceMore.csv包含住宿之间的附加距离。现在将这些边加载到图形中。查看图表并重新排列节点，以便很容易区分南北校区。提交图形的图片。你应该注意到两组学院，它们之间有一些联系）

```
C++ | 复制代码

1  LOAD CSV WITH HEADERS FROM "file:///distanceMore.csv" AS row
2  MATCH (a{name:row.from}), (b{name:row.to}) MERGE (a)-[:DISTANCE {meters:
3  toInteger(row.meters)}]-(b)
4  MATCH (n:college)
5  RETURN n
6  ORDER BY n.location
```



17. List the four Colleges with the highest betweenness centrality values (ignoring the direction of relationships and using the DISTANCE relationship property METERS)? Use the algorithm described here https://neo4j.com/docs/graph-data-science/1.1/algorithms/betweenness-centrality/ Show your code and results.（列出介数中心值最高的四所学院。）

C++ | 复制代码

```
CALL gds.graph.project(
    'myGraph',
    'college', {
      distance : {
        type : 'DISTANCE',
        orientation : 'NATURAL',
        properties : 'meters'
      }
    });
CALL gds.betweenness.stream('myGraph', {relationshipWeightProperty : 'meters'})
YIELD nodeId, score
RETURN gds.util.asNode(nodeId).name AS college, score
ORDER BY score DESC
LIMIT 4
```

college	score
"McMurtry"	38.0
"Baker"	38.0
"Brown"	12.0
"Will Rice"	12.0