

#### Пункт 4. Внесение данных в графовую БД (БД neo4j):

```
uri, username, password, db = 'neo4j://localhost:7687', 'neo4j', '*****', 'neo4j'

def create_event(tx, event):
    tx.run("""MERGE (p1:Person {name: $event.Person_1})
              MERGE (p2:Person {name: $event.Person_2})
              MERGE (p1)-[:RELATION {id: $event.id}]- (p2)
            """, event = event
    )

with GraphDatabase.driver(uri, auth = (username, password)) as driver:
    with driver.session(database=db) as session:

        for row in range(len(df)):
            event = df.iloc[row].to_dict()
            session.execute_write(create_event, event)
```

#### Пункт 4. Внесение данных в графовую БД (БД NebulaGraph):

```
config = Config()
config.max_connection_pool_size = 10
keys_list = {j:f'p_{i}' for i,j in enumerate(fio_list,1)} #id номера персоналей для
формирования vid

connection_pool = ConnectionPool()
ok = connection_pool.init([('127.0.0.1', 9669)], config)

with connection_pool.session_context('root', '*****') as session:
    session.execute('''CREATE SPACE IF NOT EXISTS task_7(partition_num=15,replica_factor=1,
                    vid_type=FIXED_STRING(32));
                    USE task_7;
                    CREATE TAG IF NOT EXISTS Person(name string);
                    CREATE EDGE IF NOT EXISTS event(id int);''')

    for row in range(len(df)):
        p1, p2, w = df.loc[row, 'Person_1'], df.loc[row, 'Person_2'], df.loc[row, 'id']
        p1_vid, p2_vid = keys_list[p1], keys_list[p2]
        session.execute(f'''INSERT VERTEX IF NOT EXISTS Person(name) VALUES "{p1_vid}":
                        ("{p1}"), "{p2_vid}":("{p2}");
                        INSERT EDGE IF NOT EXISTS event(id) VALUES "{p1_vid}"→
                        "{p2_vid}":({w});''')

connection_pool.close()
```

Пункт 5. Запросы к БД :

Выведем все узлы графа:

```
MATCH (p)
RETURN p
```


Neo4j	NebulaGraph																										
Определим количество узлов графа:																											
<pre>MATCH () RETURN COUNT(*)</pre> <table><tr><td>"COUNT(*)"</td></tr><tr><td>9899</td></tr></table>	"COUNT(*)"	9899	<pre>SUBMIT JOB STATS SHOW STATS</pre> <table><tr><td colspan="3">+-----+-----+-----+</td></tr><tr><td>  Type</td><td>  Name</td><td>  Count  </td></tr><tr><td colspan="3">+-----+-----+-----+</td></tr><tr><td>  "Tag"</td><td>  "Person"</td><td>  9899  </td></tr><tr><td>  "Edge"</td><td>  "event"</td><td>  5000  </td></tr><tr><td>  "Space"</td><td>  "vertices"</td><td>  9899  </td></tr><tr><td>  "Space"</td><td>  "edges"</td><td>  5000  </td></tr><tr><td colspan="3">+-----+-----+-----+</td></tr></table>	+-----+-----+-----+			Type	Name	Count	+-----+-----+-----+			"Tag"	"Person"	9899	"Edge"	"event"	5000	"Space"	"vertices"	9899	"Space"	"edges"	5000	+-----+-----+-----+		
"COUNT(*)"																											
9899																											
+-----+-----+-----+																											
Type	Name	Count																									
+-----+-----+-----+																											
"Tag"	"Person"	9899																									
"Edge"	"event"	5000																									
"Space"	"vertices"	9899																									
"Space"	"edges"	5000																									
+-----+-----+-----+																											

Определим количество отношений (событий) графа:

MATCH () - - > () RETURN COUNT(*)	SUBMIT JOB STATS SHOW STATS																	
<table><tr><td>"count(*)"</td></tr><tr><td>5000</td></tr></table>	"count(*)"	5000	<table><tr><td>Type</td><td>Name</td><td>Count</td></tr><tr><td>"Tag"</td><td>"Person"</td><td>9899</td></tr><tr><td>"Edge"</td><td>"event"</td><td>5000</td></tr><tr><td>"Space"</td><td>"vertices"</td><td>9899</td></tr><tr><td>"Space"</td><td>"edges"</td><td>5000</td></tr></table>	Type	Name	Count	"Tag"	"Person"	9899	"Edge"	"event"	5000	"Space"	"vertices"	9899	"Space"	"edges"	5000
"count(*)"																		
5000																		
Type	Name	Count																
"Tag"	"Person"	9899																
"Edge"	"event"	5000																
"Space"	"vertices"	9899																
"Space"	"edges"	5000																

<pre>MATCH ()-[r:RELATION]-&gt;() WITH AVG(r.id) AS avgId MATCH ()-[r:RELATION]-&gt;() WHERE r.id&gt;avgId RETURN COUNT(*)</pre> <table><tr><td>"count(*)"</td></tr><tr><td>2519</td></tr></table>	"count(*)"	2519	<pre>MATCH ()-[e]-&gt;() \ WITH avg(e.id) AS avgId \ MATCH ()-[e]-&gt;() WHERE e.id&gt;avgId \ RETURN count(*)</pre> <table><tr><td>"count(*)"</td></tr><tr><td>2519</td></tr></table>	"count(*)"	2519
"count(*)"					
2519					
"count(*)"					
2519					

Выведем граф для определенного ФИО (Обелова Кристина Ильдаровна):

<pre>MATCH ans = ({name: "Обелова Кристина Ильдаровна"}) --() RETURN ans</pre> <pre>[{"name": "Обелова Кристина Ильдаровна"}, {"id": 791076}, {"name": "Божок Виталий Яковлев"}]</pre> 	<pre>MATCH ans=(:Person{name:"Обелова Кристина Ильдаровна"})-[:event@0 {id: 791076}]-&gt;(:Person{name:"Божок Виталий Яковлев"})</pre> <p>где p_xxx – индексы узлов БД.</p>
--	---

Пункт 6. Важный пункт (БД нео4j):

Определим узлы имеющие более одного отношения:

```
MATCH (p1)--(p2)
WITH p1, count(p2) AS n
WHERE n > 1
RETURN p1, n
```

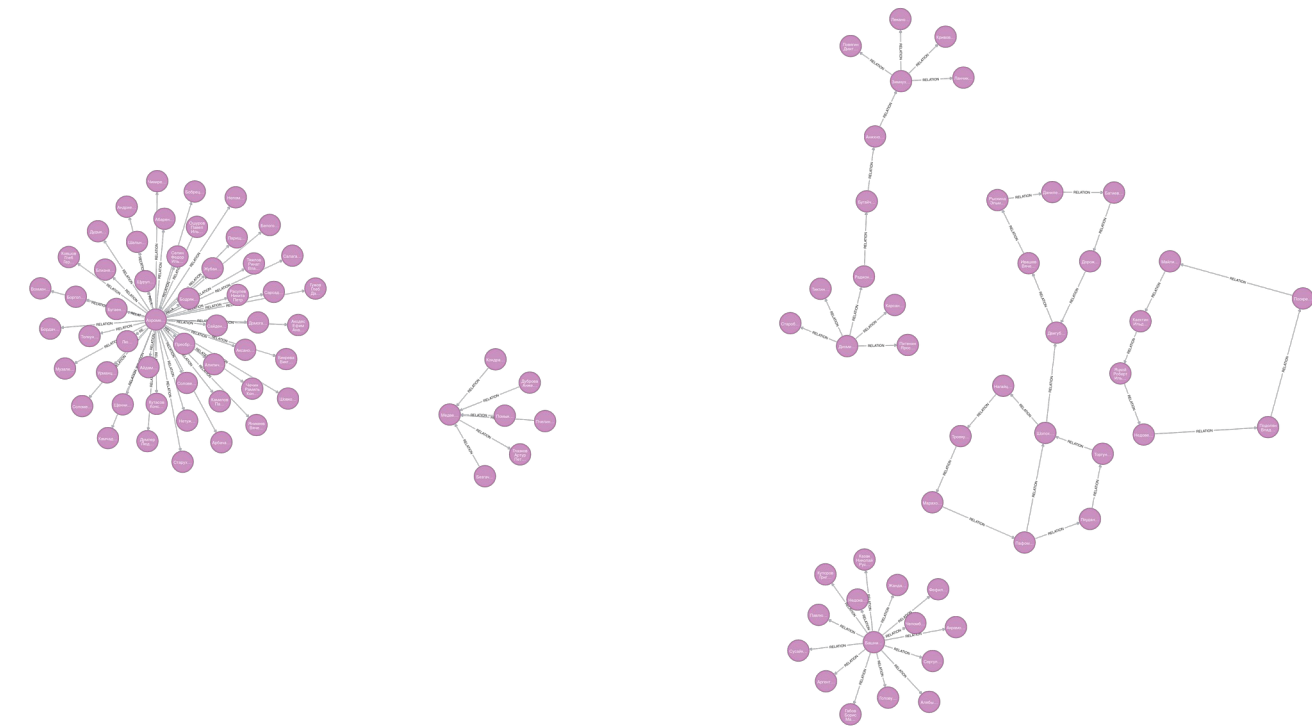
Результаты запроса:

где n это количество контрагентов  
p1 представитель социальной сети имеющий отношения с более чем одним контрагентом.

"p1"	"n"
{"name": "Ахромеева Алина Ивановна"}	50
{"name": "Башнина Антонина Глебовна"}	14
{"name": "Медведева Дарья Алексеевна"}	6
{"name": "Пафомова Кира Вадимовна"}	3
{"name": "Ляуданский Валентин Владиславович"}	2
{"name": "Зимнухова Карина Даниловна"}	5
{"name": "Недовесков Владимир Иванович"}	2
{"name": "Подольян Владислав Денисович"}	2
{"name": "Шолохов Игорь Робертович"}	4
{"name": "Двигубская Валентина Геннадьевна"}	3
{"name": "Даниленко Владимир Семенович"}	2
{"name": "Батиевская Ангелина Романовна"}	2
{"name": "Майлина Гульнара Ивановна"}	2
{"name": "Каехтин Ильдар Эдуардович"}	2
{"name": "Анихнова Тамара Руслановна"}	2
{"name": "Ивашев Вячеслав Игоревич"}	2
{"name": "Диомидов Игорь Ильдарович"}	5
{"name": "Рыскина Эльмира Ивановна"}	2
{"name": "Нагайцева Анжелика Яновна"}	2
{"name": "Троекуров Глеб Ефимович"}	2
{"name": "Радионова Тамара Ярославовна"}	2
{"name": "Бугайчук Роман Эдуардович"}	2
{"name": "Дорожкин Анатолий Егорович"}	2
{"name": "Яцкой Роберт Ильдарович"}	2
{"name": "Мараховская Дарья Романовна"}	2
{"name": "Торгунаков Роман Кириллович"}	2

{ "name": "Поскребышев Яков Дмитриевич" }	2
---	---

# Графическое отражение социальных взаимодействий:



## Вывод:

1. 27 из 9899 представителей социальной сети взаимодействуют более чем с одним контрагентом (0.27%)
2. Представленные взаимоотношения можно отнести к двум структурам:
  - 2.1. Древовидная с единым корневым узлом. Если соотнести это с трудовыми отношениями то это связь руководитель – подчиненный ("вертикальное взаимодействие").
  - 2.2. Замкнутый граф описывает в терминах трудовых отношений горизонтальное взаимодействие;

Пункт 6. Важный пункт (Nebula Graph):

Определим узлы имеющие более одного отношения:

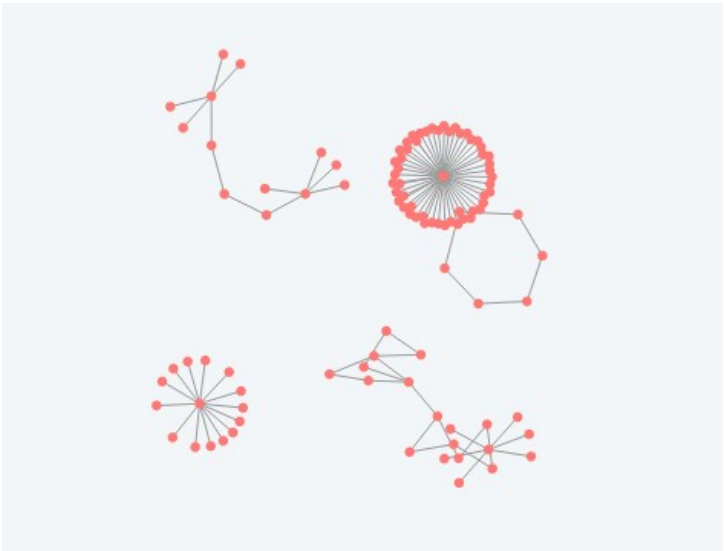
```
MATCH (p1)--(p2) \
WITH p1,count(p2) as n WHERE n>1 \
RETURN p1, n ORDER BY n DESC;
```

```
(root@nebula) [task_7]> MATCH (p1)--(p2) WITH p1,count(p2) as n WHERE n>1 RETURN p1, n ORDER BY n DESC;
```

p1	n
("p_1974" :Person{name: "Ахромеева Алина Ивановна"})	50
("p_7206" :Person{name: "Башнина Антонина Глебовна"})	14
("p_6735" :Person{name: "Медведева Дарья Алексеевна"})	6
("p_1511" :Person{name: "Диомидов Игорь Ильдарович"})	5
("p_2888" :Person{name: "Зимнухова Карина Даниловна"})	5
("p_9451" :Person{name: "Шолохов Игорь Робертович"})	4
("p_319" :Person{name: "Пафомова Кира Вадимовна"})	3
("p_5281" :Person{name: "Двигубская Валентина Геннадьевна"})	3
("p_1684" :Person{name: "Подольн Владислав Денисович"})	2
("p_1316" :Person{name: "Ляуданский Валентин Владиславович"})	2
("p_8803" :Person{name: "Яцкой Роберт Ильдарович"})	2
("p_1110" :Person{name: "Недовесков Владимир Иванович"})	2
("p_7521" :Person{name: "Дорожкин Анатолий Егорович"})	2
("p_5609" :Person{name: "Бугайчук Роман Эдуардович"})	2
("p_796" :Person{name: "Даниленко Владимир Семенович"})	2
("p_1570" :Person{name: "Торгунаков Роман Кириллович"})	2
("p_5406" :Person{name: "Мараховская Дарья Романовна"})	2
("p_7344" :Person{name: "Майлина Гульнара Ивановна"})	2
("p_5394" :Person{name: "Рыскина Эльмира Ивановна"})	2
("p_6006" :Person{name: "Нагайцева Анжелика Яновна"})	2
("p_7499" :Person{name: "Ивашев Вячеслав Игоревич"})	2
("p_5848" :Person{name: "Батиевская Ангелина Романовна"})	2
("p_9159" :Person{name: "Каехтин Ильдар Эдуардович"})	2
("p_5615" :Person{name: "Анихнова Тамара Руслановна"})	2
("p_2738" :Person{name: "Поскребышев Яков Дмитриевич"})	2
("p_3544" :Person{name: "Радионова Тамара Ярославовна"})	2
("p_7491" :Person{name: "Троекуров Глеб Ефимович"})	2

```
MATCH ans=(p1)--(p2) \
WITH p1, count(p2) as n WHERE n>1 \
MATCH ans=(p1)--()
RETURN ans;
```

Графическое отражение социальных взаимодействий:



### Пункт 7. REST запрос (БД neo4j):

```
d = {"statements": [{"statement": "MATCH ans = ({name: $props.name})--() RETURN ans",
                        "parameters": {"props": {"name": fio}},
                    ]}}

url = "http://localhost:7474/db/neo4j/tx/commit"

headers = {'Accept': 'application/json',
           'Content-Type': 'application/json',
           'Authorization': 'Basic ' + base64.b64encode(f'{username}:{password}'.encode('utf-8')).decode()}

r = requests.post(url, data=json.dumps(d), headers=headers)

result = []
for event in json.loads(r.text)['results'][0]['data'][0]['row']:
    _a = dict(zip(['person_1', 'id', 'person_2'], (list(k.values())[0] for k in event)))
    result.append(json.dumps(_a))
result

[{'person_1': "Обелова Кристина Ильдаровна", "id": 791076, "person_2": "Божок Виталий Яковлев"}]
```

### Пункт 7. Запрос (БД Nebula Graph):

```
connection_pool = ConnectionPool()
ok = connection_pool.init([('127.0.0.1', 9669)], config)

with connection_pool.session_context('root', '*****') as session:

    session.execute('USE task_7;')
    r = session.execute('LOOKUP ON Person
                        WHERE Person.name=="Медведева Дарья Алексеевна"
                        YIELD id(vertex) AS p | GO FROM $.p OVER event BIDIRECT
                        YIELD properties($).name AS person_1, properties(edge).id AS id,
                        properties($^).name AS person_2')

connection_pool.close()

result = []
for row in range(r.row_size()):
    _dict = dict.fromkeys(r.keys(), None)
    for k in r.keys():
        if k in ['id']:
            _dict[k] = r.column_values(k)[row].as_int()
        else:
            _dict[k] = r.column_values(k)[row].as_string()
    result.append(json.dumps(_dict))

result

[{'person_1': 'Кондратьев Борис Германович', 'id': 87253, 'person_2': 'Медведева Дарья Алексеевна'},
 {'person_1': 'Помыкалова Тамара Федоровна', 'id': 196243, 'person_2': 'Медведева Дарья Алексеевна'},
 {'person_1': 'Пчелинцев Артур Глебович', 'id': 580478, 'person_2': 'Медведева Дарья Алексеевна'},
 {'person_1': 'Дуброва Анжелика Григорьевна', 'id': 327044, 'person_2': 'Медведева Дарья Алексеевна'},
 {'person_1': 'Безгачий Денис Ефимович', 'id': 109281, 'person_2': 'Медведева Дарья Алексеевна'},
 {'person_1': 'Глазков Артур Петрович', 'id': 173973, 'person_2': 'Медведева Дарья Алексеевна'}]
```