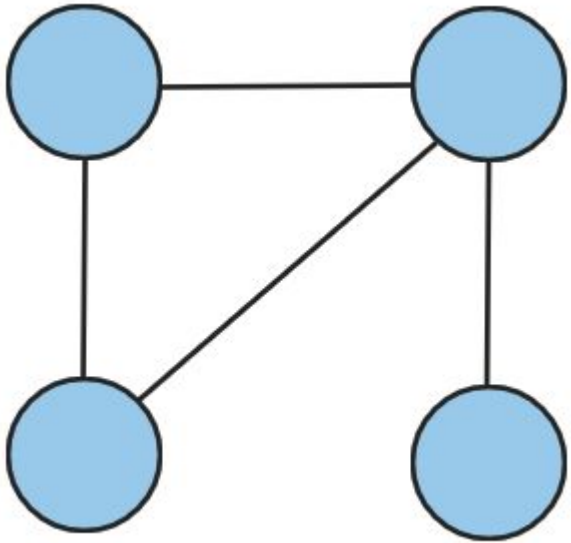
Графови бази от данни

Веселин
Камелия
Мартин
Михаела

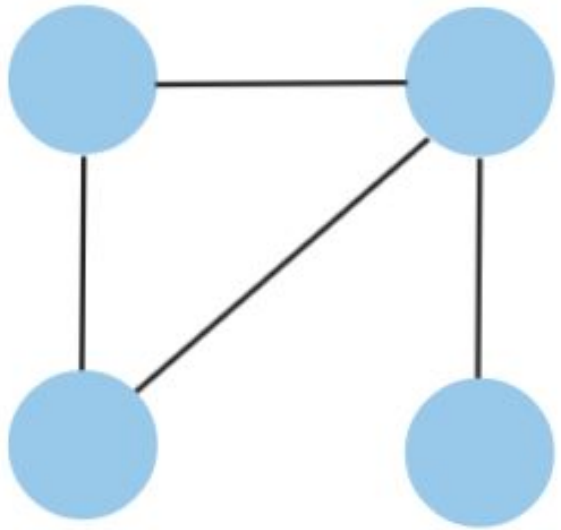
Обхождане



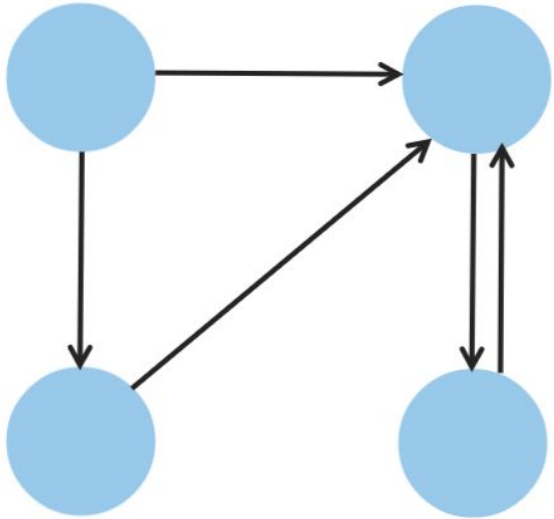
Графови модели

- Неориентиран граф
- Ориентиран граф
- Ориентиран мултиграф
- Ориентиран мултиграф с тегла
- Ориентиран мултиграф с етикети
- Ориентиран мултиграф със свойства
- Ориентиран мултиграф с етикети и свойства

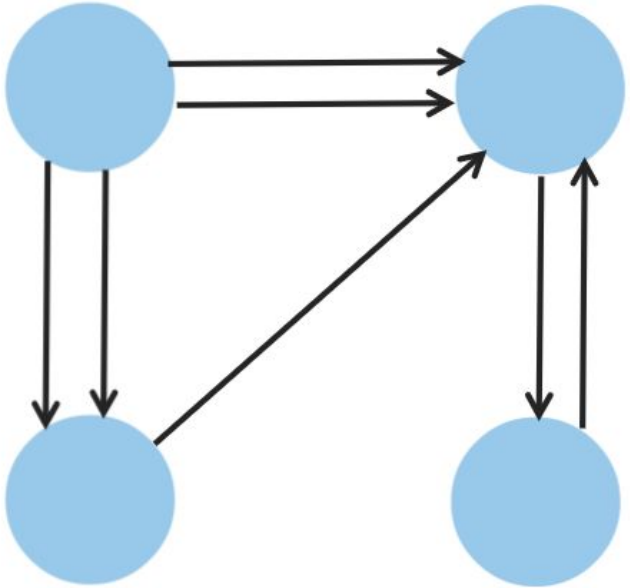
Неориентиран граф



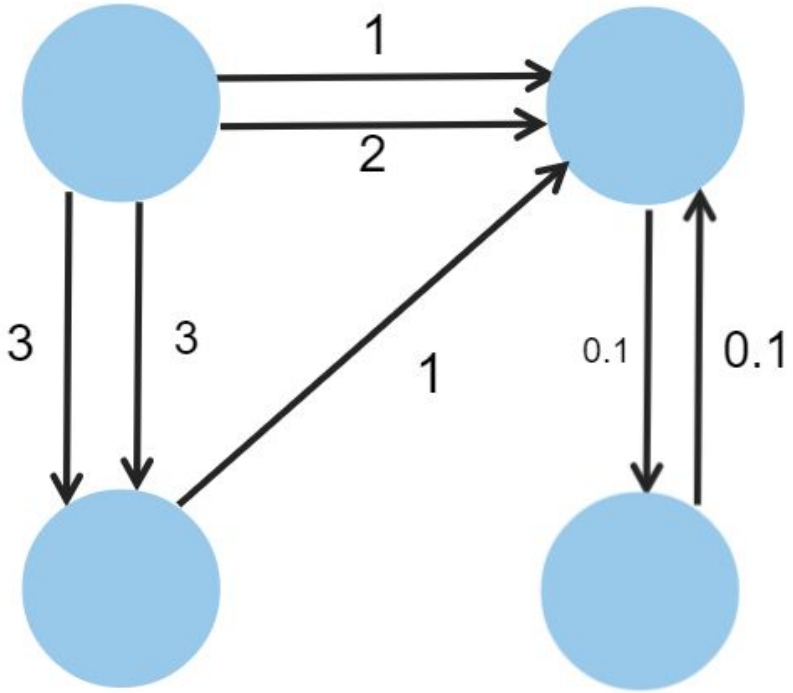
Ориентиран граф



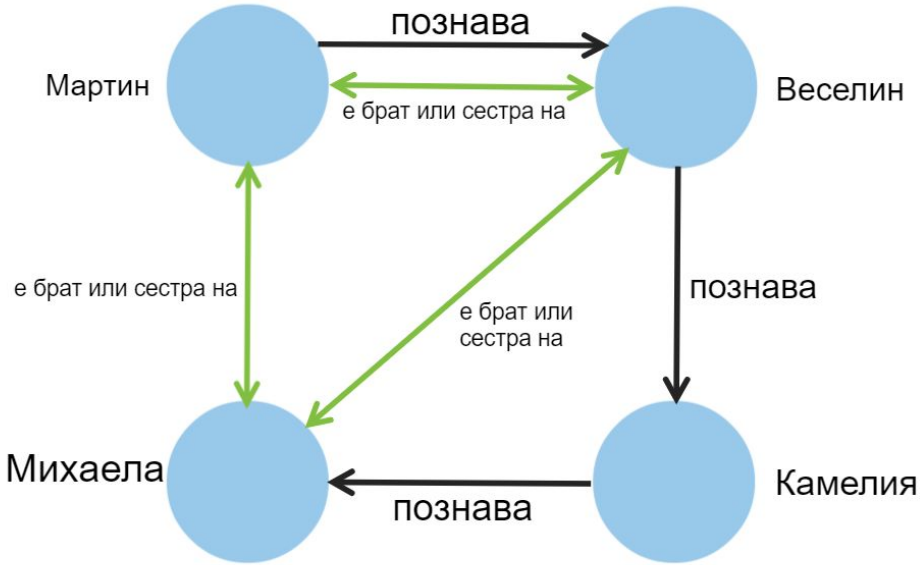
Ориентиран мултиграф



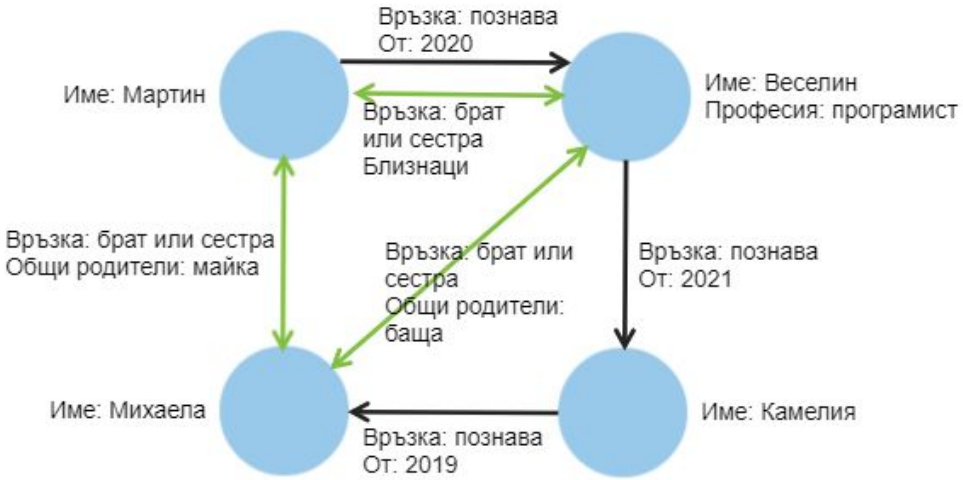
Ориентиран мултиграф с тегла



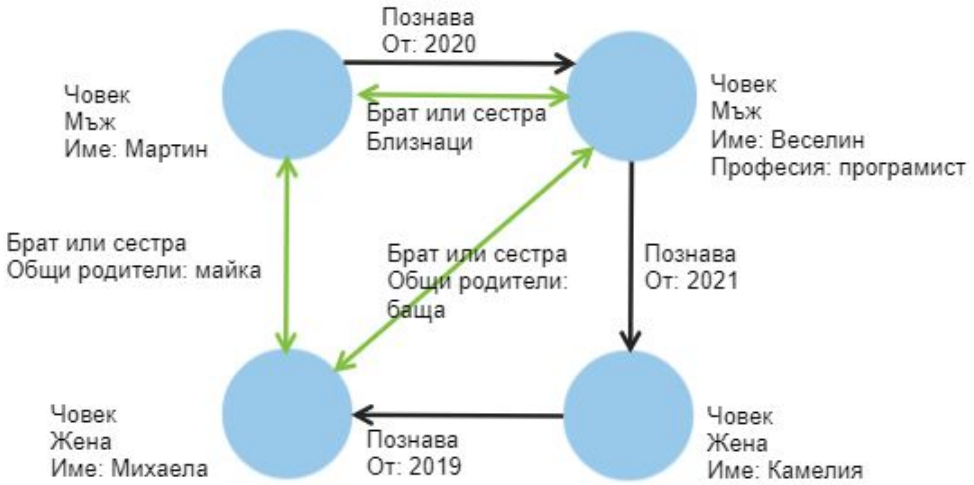
Ориентиран мултиграф с етикети



Ориентиран мултиграф със свойства



Ориентиран мултиграф с етикети и свойства



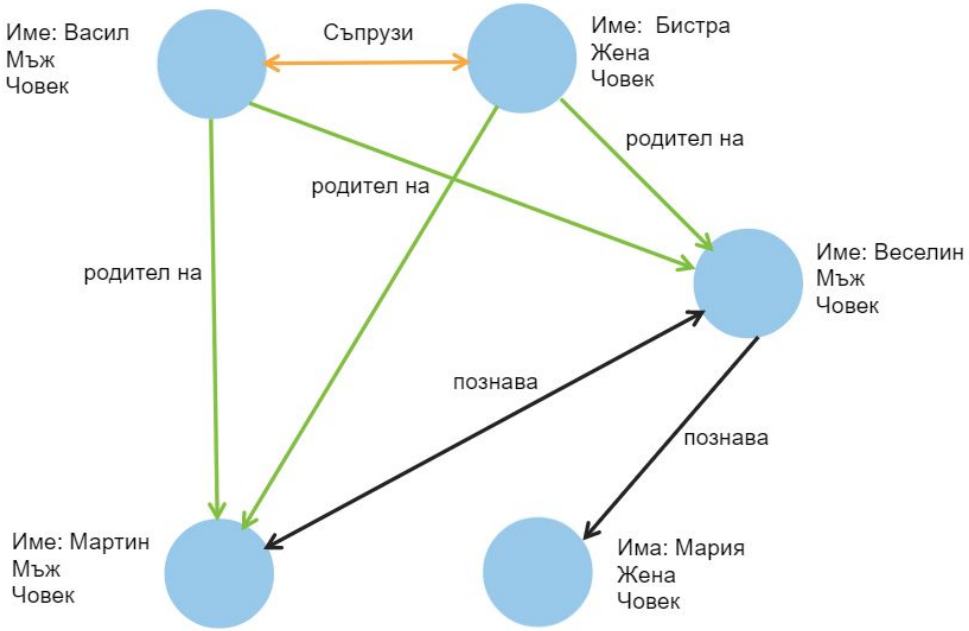
- Най-популярната система за управление на графова база от данни
- Отворен код
- Платена за комерсиално ползване

Cypher query language

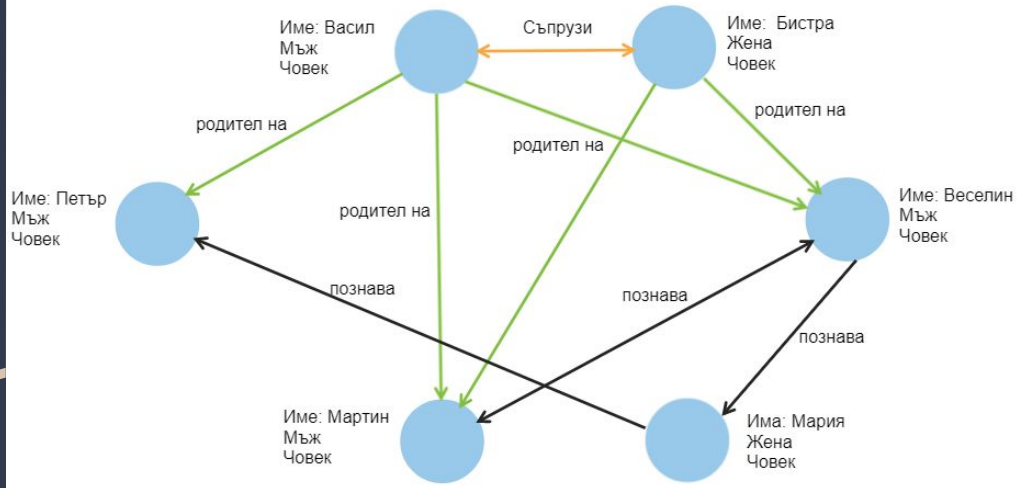
- Език за заявки към графови данни
- Създаден е специално за Neo4j
- Декларативен език
- Pattern matching
- Многоплатформен
- Изчерпателен
- Лесен за четене и писане

Примери с Neo4j

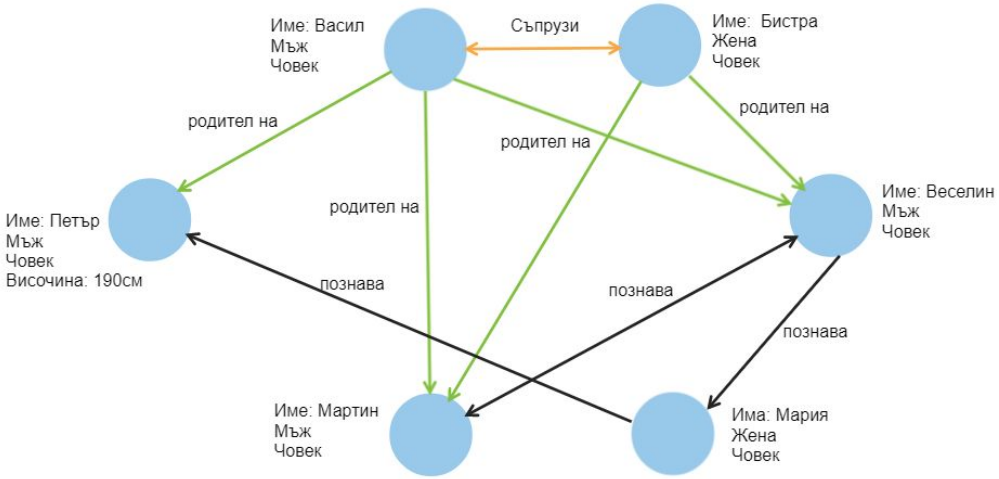
Първоначален граф



Добавяне на елемент



Промяна на елемент



Сравнение между различните типове бази данни

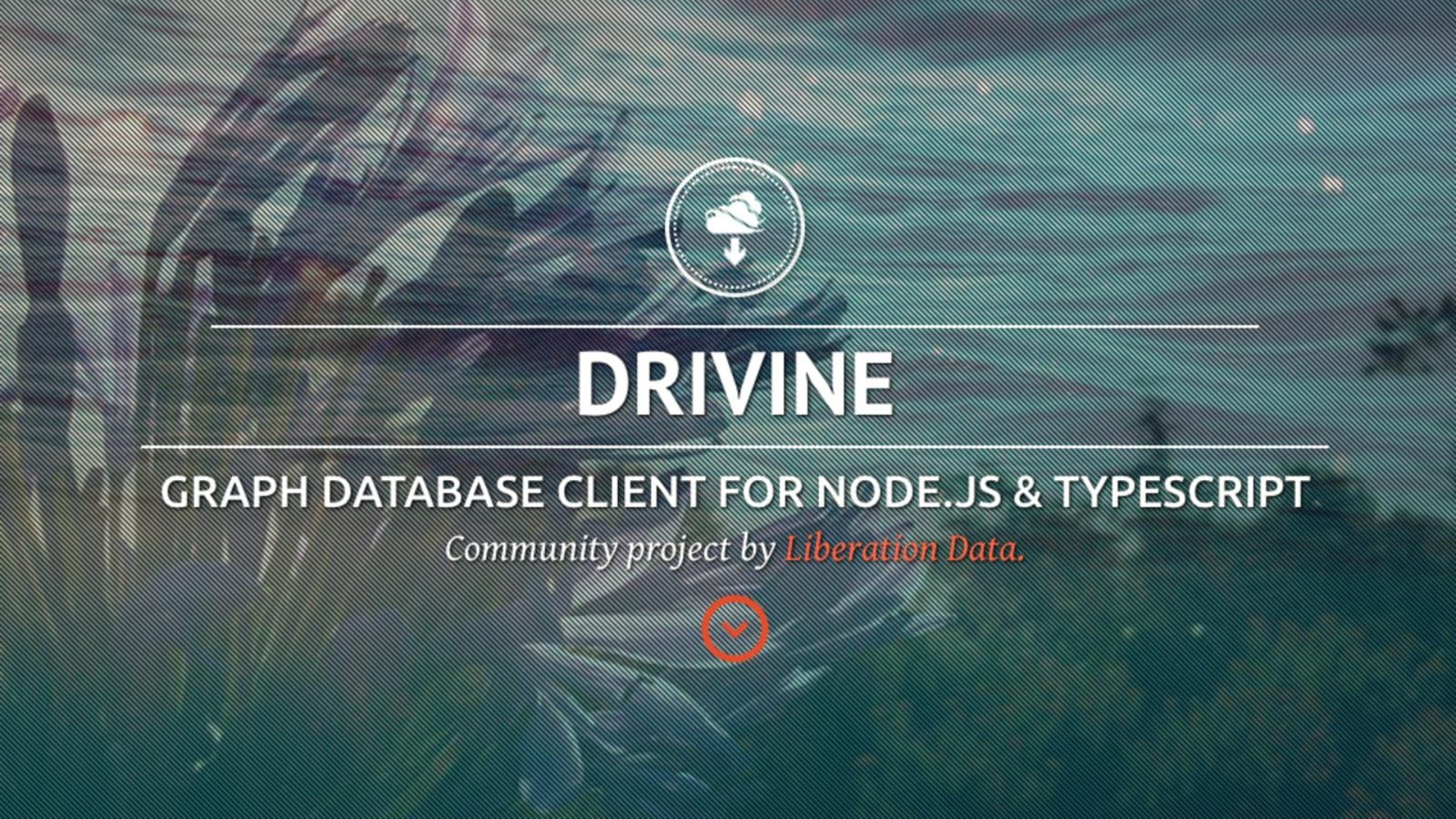
Графова	Релационна	Нерелационна
обработва сложни връзки, обработва големи количества от данни, използва възли, дъги, свойства	обработва сложни връзки, по-малко гъвкава, силно структурирана, използва таблици,	обработва неструктурирани/полуструктурирани данни, обработва големи количества данни, по-малко структурирана, по-трудно се изпращат заявки
социални мрежи, механизми за препоръки, системи за откриване на измами,	финансови системи, управление на инвентара и взаимоотношения с клиенти	анализи в реално време, управление на съдържанието, IoT системи

Заявка, която извлича всички публикации и коментари, създадени от приятели на потребител

```
SELECT p.post_id, p.content, c.comment_id, c.content
FROM Posts p
JOIN Users u ON p.user_id = u.user_id
JOIN Friends f ON u.user_id = f.user_id_1 OR u.user_id = f.user_id_2
JOIN Posts p2 ON f.user_id_1 = p2.user_id OR f.user_id_2 = p2.user_id
JOIN Comments c ON p2.post_id = c.post_id
WHERE u.user_id = <user_id>
```

```
MATCH (u:User)-[:FRIENDS_WITH]-(f:User)-[:CREATED]->(p:Post)-[:HAS_COMMENT]->(c:Comment)
WHERE u.user_id = <user_id>
RETURN u.name, f.name, p.content, c.content
```

```
db.posts.aggregate([{
    $lookup: {
      from: "users",
      localField: "user_id",
      foreignField: "user_id",
      as: "user"
    }
  },{
    $unwind: "$user"
  },{
    $lookup: {
      from: "friends",
      let: { user_id: "$user.user_id" },
      pipeline: [{
          $match: {
            $expr: {
              $or: [
                { $eq: ["$user_id_1", "$$user_id"] },
                { $eq: ["$user_id_2", "$$user_id"] }
              ]
            }
          }
        }],
      as: "friend"
    }
  },{
    $unwind: "$friend"
  },{
    $lookup: {
      from: "comments",
      localField: "post_id",
      foreignField: "post_id",
      as: "comments"
    }
  },{
    $unwind: "$comments"
  },{
    $project: {
      _id: 0,
      "user.name": 1,
      "friend.name": 1,
      "content": 1,
      "comments.content": 1
    }
  }
])
```


DRIVINE
GRAPH DATABASE CLIENT FOR NODE.JS & TYPESCRIPT
Community project by Liberation Data.

Благодарим за вниманието