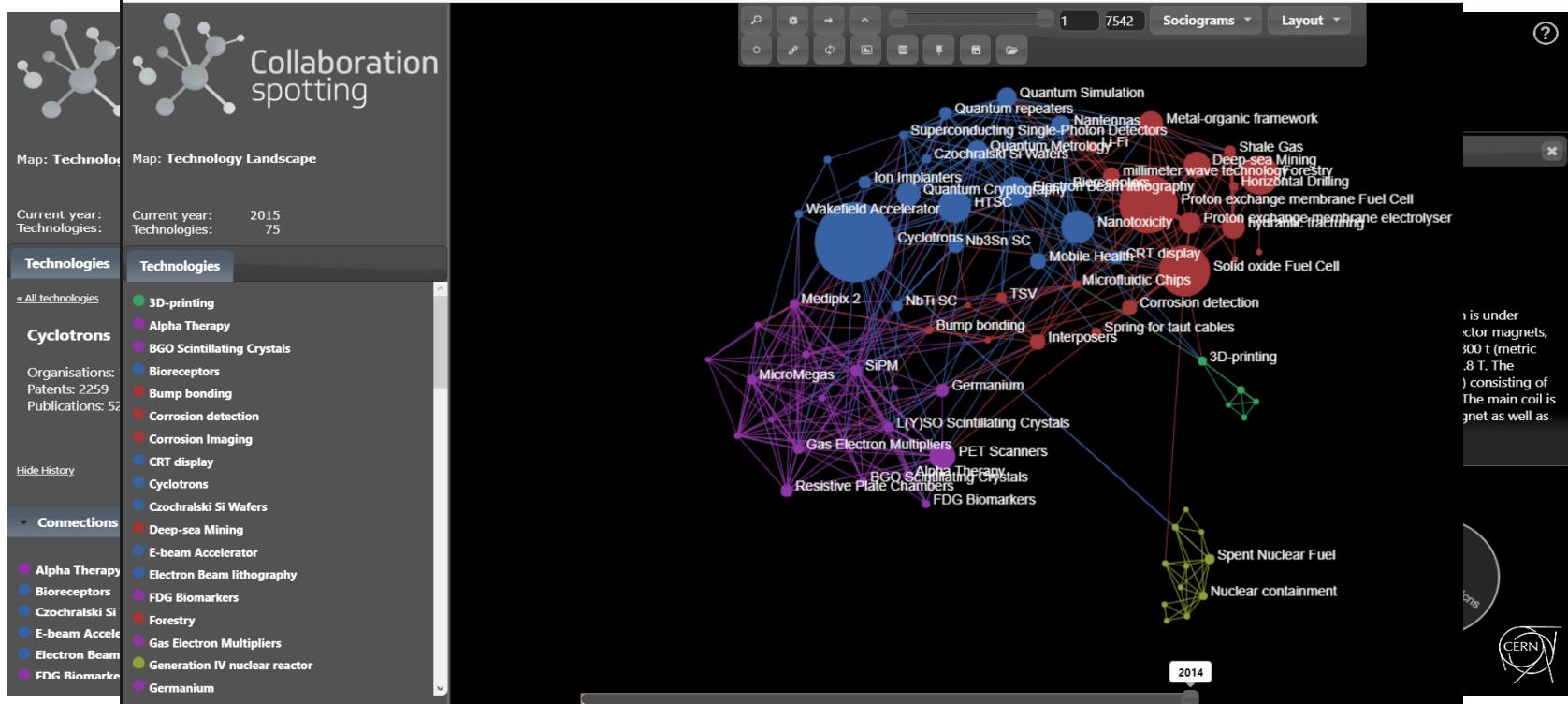


CERN Internship

Collaboration Mapping in Physics

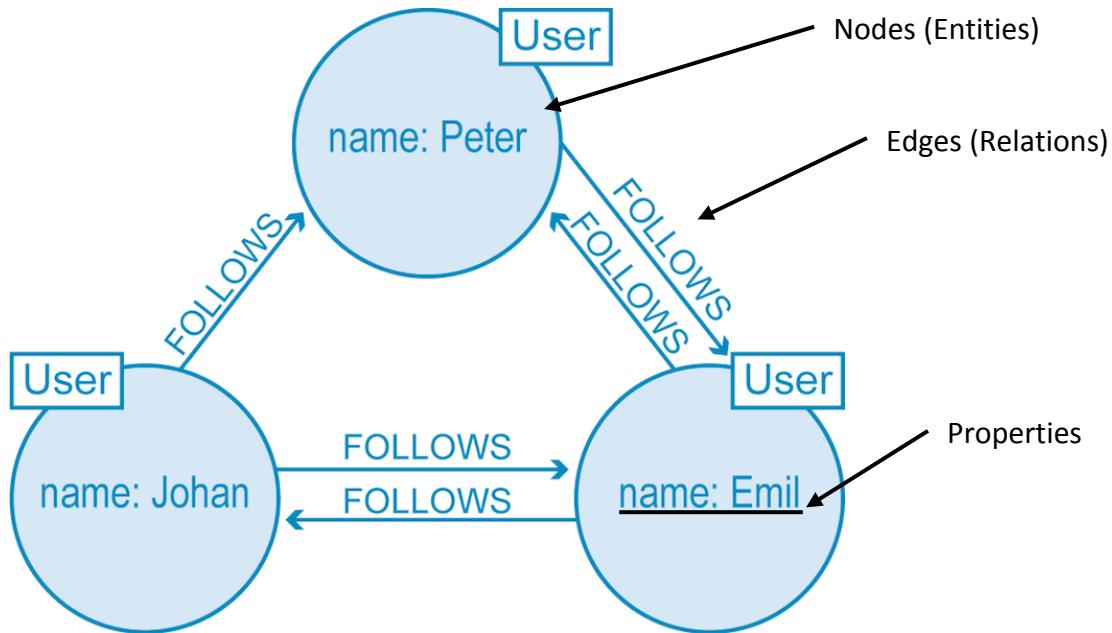


Collaboration Spotting



Kai Käfer, Johanna Eschment

Graph Databases



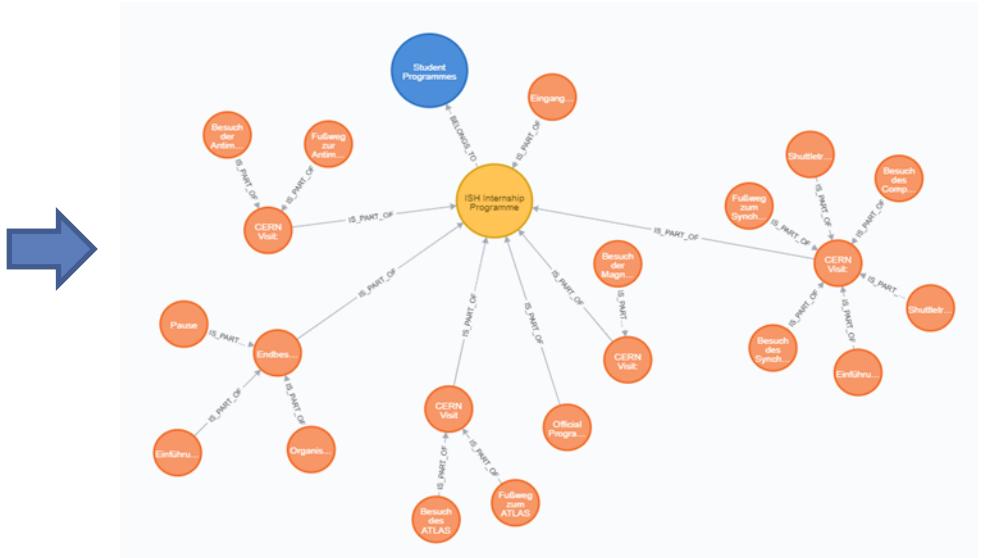
- Graph structured database
- Useful for heavily inter-connected data
- Requires a own query language

Our Goal: Graph Database of Indico

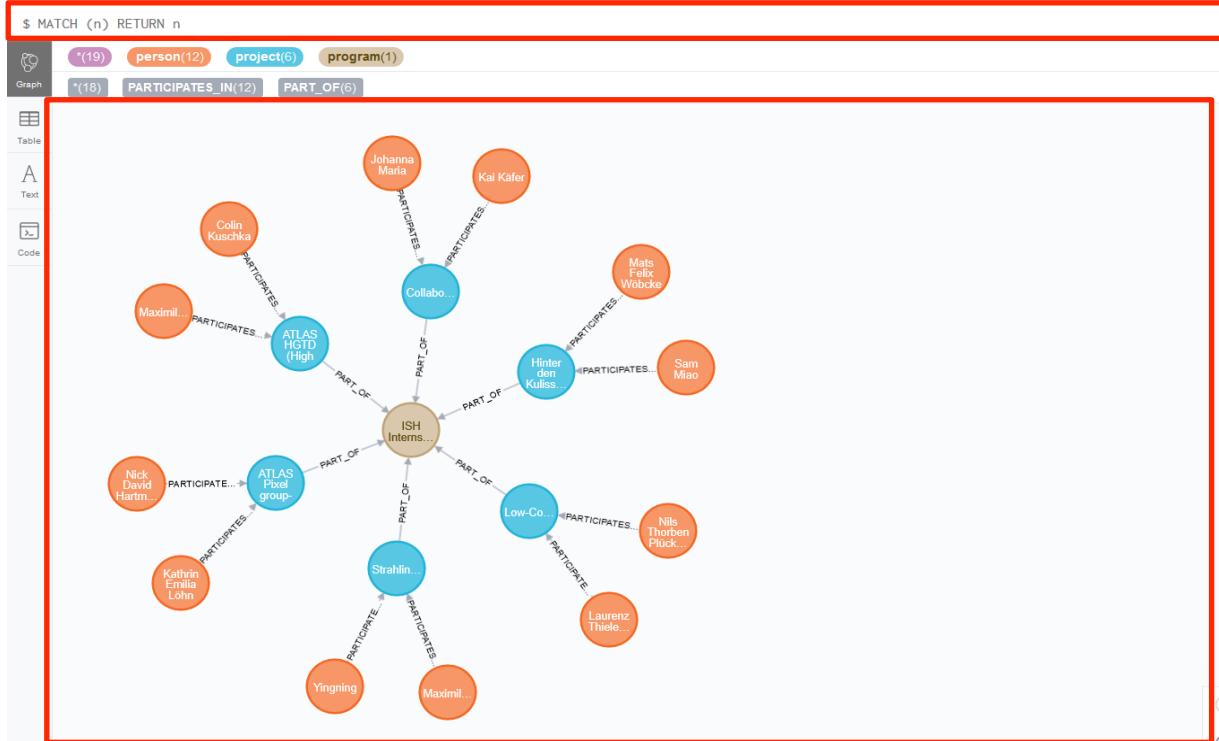


The screenshot shows the Indico interface with a top navigation bar for 'Public', 'Europa/Zürich', and 'J. Eschenmoser'. The main content area displays a timeline of events:

- FRIDAY, 13 APRIL**
 - 09:00 - 09:50 CERN Visit:** Treffpunkt: Rezeption Gebäude 33. Unbedingt: GEÖFFNETE SCHÜRE und KLEINE HÖHEN ABSATZE! Untersetzung: LEBENSMITTEL oder GETRÄNKE! Empfohlen: Minihupe von Kameras / Smartphones zum Fotografieren
 - 09:00 - 10:30 CERN Visit:** Besuch des ältesten Teilchenbeschleunigers am CERN - das Synchrozyklotron (SC)
- TUESDAY, 17 APRIL**
 - 09:00 - 10:30 CERN Visit:** Unbedingt: GEÖFFNETE SCHÜRE und KLEINE HÖHEN ABSATZE! Untersetzung: LEBENSMITTEL oder GETRÄNKE! Empfohlen: Minihupe von Kameras / Smartphones zum Fotografieren Treffpunkt: Rezeption Gebäude 33
- FRIDAY, 20 APRIL**
 - 09:00 - 13:10 Endbesprechung und Vorträge**
 - 09:00 Einführung** Speaker: Susanne Dührkoop (Westerdolls Gymnasium DE))
 - 09:10 High Performance Computing Simulationen für Mehrteilchen-Effekte in den Synchrotronen** Speakers: Daniel Gebhard, Philipp Apostel
 - High performance c...
 - High performance c...



Neo4j



Command line for the Neo4j Cypher language

Illustration of the graph



Cypher Language

```
$ CREATE (johanna:person{name:"Johanna", id:"1234"})  
  
$ MATCH (n:person{name:"Johanna"}),(m:person{name:"Kai"}) CREATE (n)-[:KNOWS]->(m)  
  
1 MATCH (nicole:Actor {name: 'Nicole Kidman'})-[:ACTED_IN]->(movie:Movie)  
2 WHERE movie.year < $yearParameter  
3 RETURN movie
```



Neo4j with Java

```
private final Driver driver;

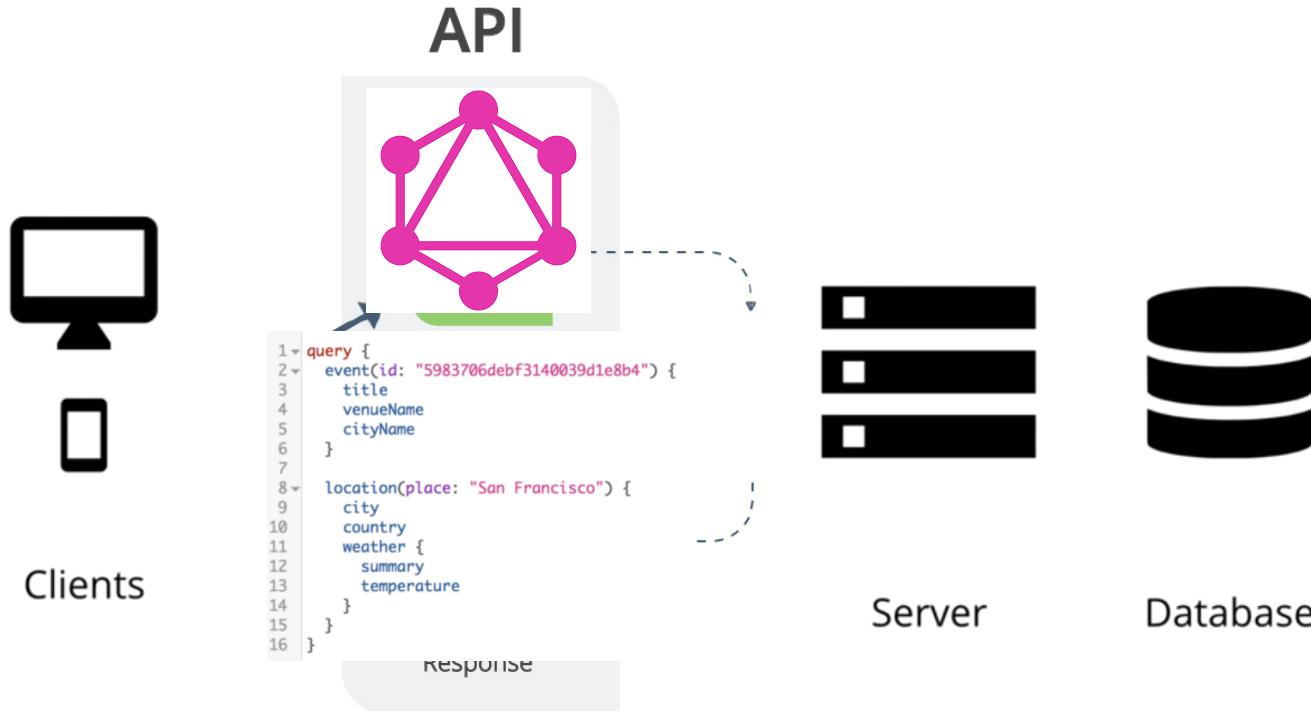
public DatabaseEx( String uri, String user, String password )
{
    driver = GraphDatabase.driver( uri, AuthTokens.basic( user, password ) );
}

public void createNode(final String Label, final String Name)
{
    try ( Session session = driver.session() )
    {
        session.writeTransaction( new TransactionWork<String>()
        {
            @Override
            public String execute( Transaction tx )
            {
                tx.run( "CREATE ("+Name+":"+Label+" {name: '"+Name+"'})" );
                return null;
            }
        });
    }
}

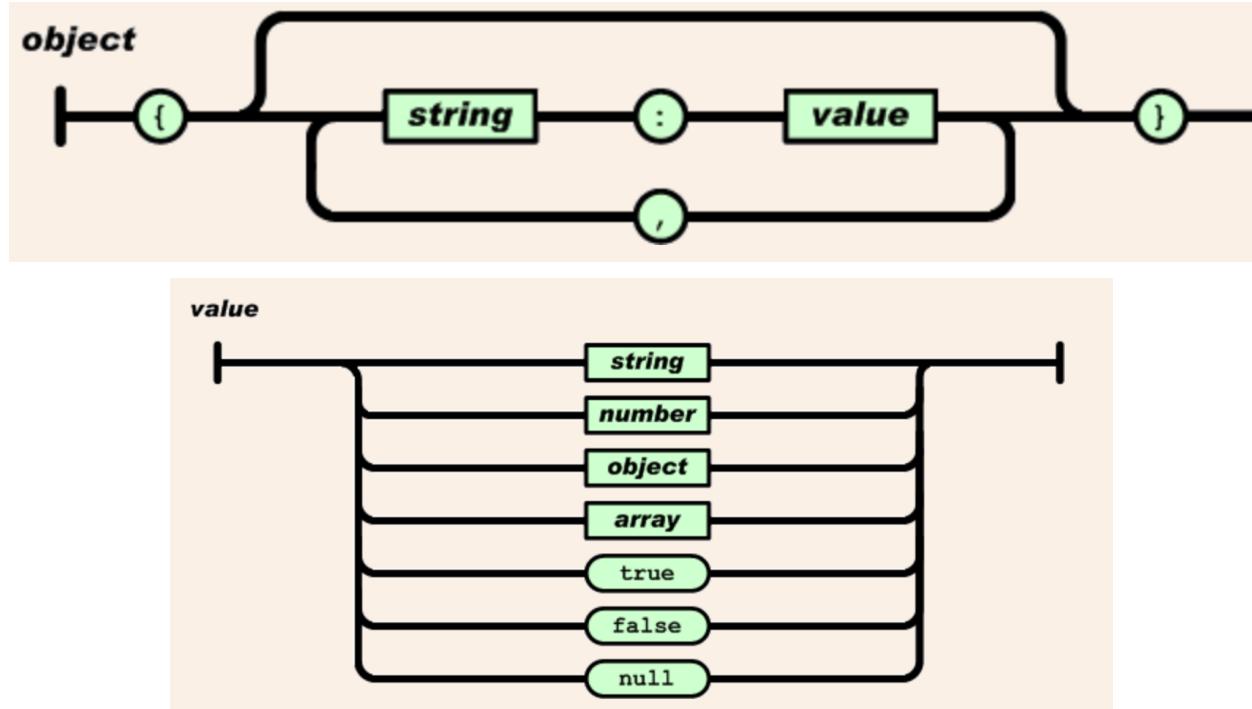
public static void main( String... args ) throws Exception
{
    try ( Database greeter = new Database( "bolt://localhost:11008", "neo4j", "234" ) )
    {
    }
}
```

>Login to your database in Neo4j

Cypher command



GraphQL: Based on the Data Type JSON





GraphQL: Schema

Nodes as types:

```
1 type Person{  
2   name: String!  
3   id: ID!  
4   born: int  
5   from_Person: [Person] @relation(name: "friends", direction: "BOTH")  
6 }
```

Property

Edge



GraphQL: Queries and Mutations

```
1 ▶ mutation{  
2     CreateUser(id: "123", name: "Kai", user_type: FACEBOOK){  
3         name  
4     }  
5 }  
6
```



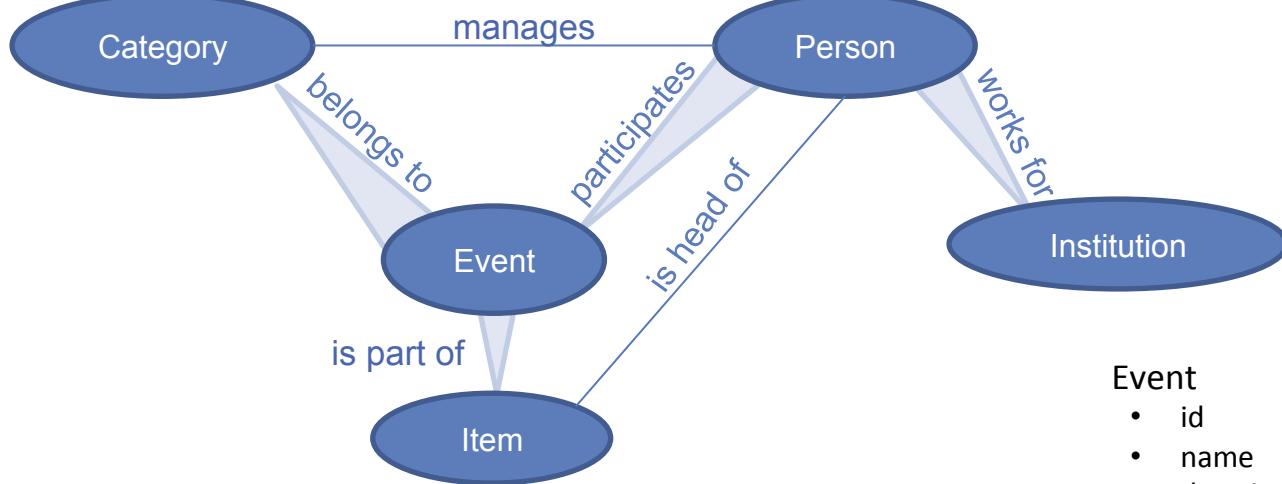
```
▼ {  
  ▼   "data": {  
    ▼     "CreateUser": {  
        "name": "Kai"  
    }  
  }  
}
```

```
1 ▶ query{  
2     User(name: "Kai"){  
3         id  
4     }  
5 }  
6
```



```
▼ {  
  ▼   "data": {  
    ▼     "User": [  
        {  
            "id": "123"  
        }  
    ]  
  }  
}
```

Our own Schema of Indico



Item	<ul style="list-style-type: none"> • id • name • description • starts • ends • file <ul style="list-style-type: none"> • id • title • url 	Person	<ul style="list-style-type: none"> • id • name
Event	<ul style="list-style-type: none"> • id • name • description • startDate • endDate 	Institution	<ul style="list-style-type: none"> • id • Name
Category	<ul style="list-style-type: none"> • id • name 		



Our own Schema of Indico

```
33 + type Event implements Root{  
34   id: ID!  
35   name: String!  
36   created: DateTime!  
37   updated: DateTime  
38   description: String  
39   startDate: DateTime  
40   endDate: DateTime  
41   to_Category: [Category] @relation(name: "BELONGS_TO", direction: "OUT")  
42   from_Person: [Person] @relation(name: "PARTICIPATES", direction: "IN")  
43   from_Item: [Item] @relation(name: "IS_PART_OF", direction: "IN")  
44 }  
45
```

```
39   startDate: DateTime  
40   endDate: DateTime  
41   to_Category: [Category] @relation(name: "BELONGS_TO", direction: "OUT")  
42   from_Person: [Person] @relation(name: "PARTICIPATES", direction: "IN")  
43   from_Item: [Item] @relation(name: "IS_PART_OF", direction: "IN")  
44 }  
45  
46 + type Category implements Brick{  
47   id: ID!  
48   name: String!  
49   from_Event: [Event] @relation(name: "BELONGS_TO", direction: "IN")  
50   from_Person: [Person] @relation(name: "MANAGES", direction: "IN")  
51   to_Category: [Category] @relation(name: "IS_PART_OF", direction: "IN")  
52   from_Category: [Category] @relation(name: "IS_PART_OF", direction: "OUT")  
53 }  
54
```

```
46 + type Category implements Brick{  
47   id: ID!  
48   name: String!  
49   from_Event: [Event] @relation(name: "BELONGS_TO", direction: "IN")  
50   (T_OF", direction: "IN")  
51   PART_OF", direction: "OUT")  
52   S", direction: "OUT")  
53   direction: "OUT")  
54   ", direction: "OUT")  
55   WORKS_FOR", direction: "OUT")  
56  
57   OF", direction: "IN")  
58   , direction: "OUT")  
59   Direction: "IN")  
60   , direction: "OUT")  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84   id: ID!  
85   name: String!  
86   country: String  
87   address: String  
88   building: String  
89   room: String  
90   from_Item: [Item] @relation(name: "TAKES_PLACE", direction: "IN")  
91 }  
92  
93  
94 + type Institution implements Brick{  
95   id: ID!  
96   name: String!  
97   from_Person: [Person] @relation(name: "WORKS_FOR", direction: "IN")  
98 }
```



Getting the Information from Indico

URL to get a JSON Object from the Indico API:

`http://my.indico.server/export/WHAT/ID.json?PARAMS&ak=KEY×tamp=TS&signature=SIG`

```
public static String build_indico_request(String path, String params, String api_key, String secret_key) {  
    String string;  
    string = "/export/" + path + "?ak=" + api_key + "&" + params + "&timestamp=";  
    long ut1 = Instant.now().getEpochSecond();  
    string = string + ut1;  
    string = "https://indico.cern.ch" + string + "&signature=" + HMAC.hmacDigest(string, secret_key, "HmacSHA1");  
    return string;  
}
```

```
def getUrl(type, id):  
    path = type + "/" + str(id) + ".json"  
    string = "/export/" + path + "?ak=" + key + "&" + param + "&timestamp=" + str(int(time.time()))  
    sha = hmac.new(s_key, string.encode(), hashlib.sha1).hexdigest()  
    url = "https://indico.cern.ch" + string + "&signature=" + sha  
    return url
```



Getting the Information from Indico

Event:

```
  ▶ startDate:
    date: "2019-03-25"
    tz: "Europe/Zurich"
    time: "08:00:00"
  _type: "Conference"
  hasAnyProtection: false
  ▶ endDate:
    date: "2019-04-05"
    tz: "Europe/Zurich"
    time: "21:00:00"
  description: "<h1>Praktikumsplätze</h1></table>\n\n<p>&nbsp;</p>"
```

roomMapURL: null

creator: {}

material: []

visibility: {}

roomFullscreen: ""

references: []

address: ""

timezone: "Europe/Zurich"

creationDate: {}

id: "776816"

category: "Student Programmes"

room: ""

title: "ISH Internship Programme"

url: "<https://indico.cern.ch/event/776816/>"

note: {}

chairs: [-]

location: "CERN"

_fossil: "conferenceMetadata"

type: "meeting"

categoryId: 5885

Item:

```
  ▶ startDate: {...}
    sessionSlotId: 323143
    code: ""
  ▶ endDate: {...}
  attachments: {...}
  color: "#dfe555"
  slotTitle: ""
  conferenceId: 776816
  sessionCode: ""
  entryType: "Session"
  ▶ entries: {...}
  duration: 240
  conveners: []
  id: "s3599403"
  inheritRoom: true
  room: "6-2-024 - BE Auditorium Meyrin"
  title: "Endbesprechung und Vorträge"
  url: "/event/776816/sessions/294763/"
  description: ""
  friendlyId: 22
  isPoster: false
  sessionId: 294763
  location: "CERN"
  uniqueId: "s3599403"
  inheritLoc: true
  contribDuration: 240
  ▶ pdf: "/event/776816/sessions/2...3/session-timetable.pdf"
  textColor: "#202020"
```



Connection between Python and graphQL

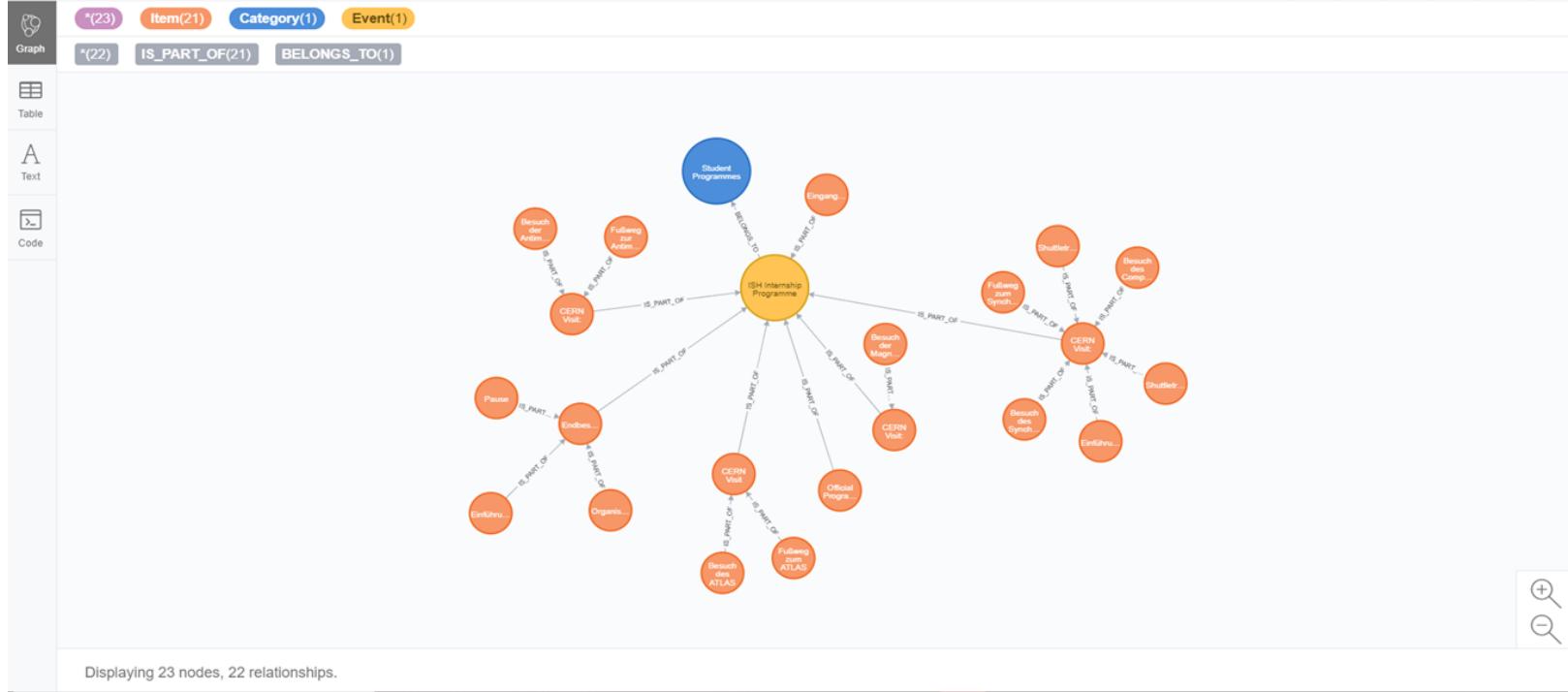
```
# Imports the GraphQL client
from graphqlclient import GraphQLClient

# Creates the GraphQL client
client = GraphQLClient('http://collspotting-symbiotics-dev.cern.ch:4001/browser/')

# Executes query or mutation
result = client.execute('''
query {
  User {
    name
  }
}
''')

# Prints results
print(result)
```

Creating an Event



Creating all Events of One Category

