SRA の技術情報誌
GS LetterNeo
Vol.162　2022 年 1 月

GraphDB を試してみました!

菊池　幸博

株式会社 SRA

アドバンストクラウドエンジニアリング事業部

はじめに

数年前に DB の検索速度の向上について調べていたところ、「それは GraphDB で解決できるよ」というアドバイスを頂きました。その時は GraphDB が何者か分からずに聞き流してしまいましたが、Neo4j の Online conference を切っ掛けに GraphDB に調べていくうちにパナマ文書の解析でも使われた事を知り、興味を持ち始めました。馴染んだ RDB と異なる世界が面白いですよ～♪

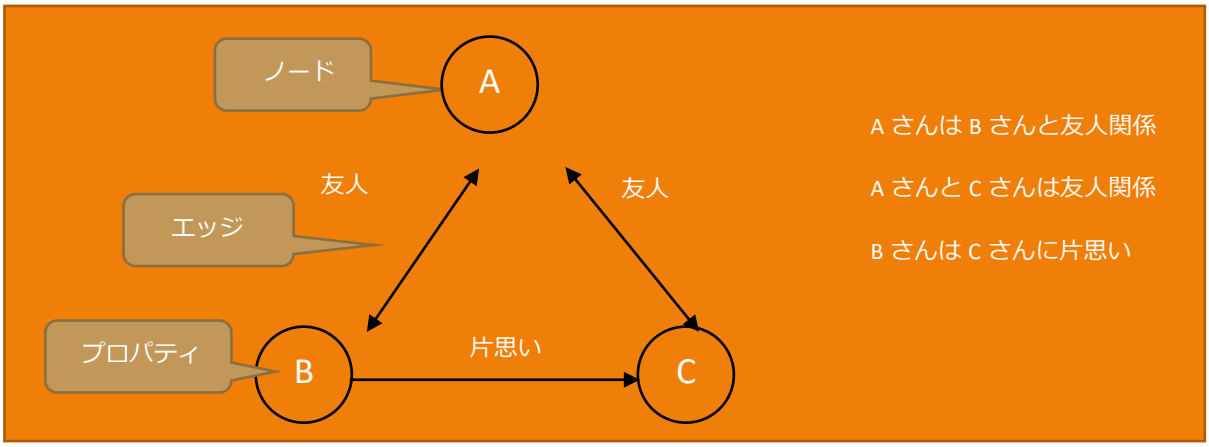
GraphDB 超入門

- ✓ GraphDB ってどんな DB？
- ✓ GraphDB を使ったサービス
- ✓ これからのこと
- ✓ Appendix

GraphDB ってどんな DB？

GraphDB をご存知でしょうか。
聞いた事は無くても実は Facebook や Yahoo!など私たちの身近なサービスに広く使用されています。
有名なところでは「パナマ文章」の解析に Neo4j という GraphDB が使用されています。
膨大なメールや数十万、数百万というレベルで存在する人間関係や企業との関係をグラフ化して、金や物の流れを把握するために使用されたという事です。
この興味深い DB について簡単な例を挙げて紹介したいと思います。

グラフという言葉で連想するのは、折れ線グラフや棒グラフではないでしょうか。
ここでは一旦そういうイメージをゼロにして下さいね。
物と物との関係性を「つながり方」を中心として抽象化した概念をグラフと呼びます。
例えば、S 社の社員 A さん、B さん、C さんの関係を以下のように表現します。



この様に人や物の関係性に注目してデータの関係性を構築していくのが GraphDB といわれるデータベースになります。
GraphDB におけるグラフは「ノード」、「エッジ」の要素でノードがあらわすもの同士の「関係性」を表現します。（この図ではノードのプロパティは名前だけ記述しています）
ぱっと見て A さん、B さん、C さんの関係性が分かると思います。
プロパティに性別や性格、身長体重などを記述すれば個人を表すデータとなり、職歴が記述されていれば履歴書となります。また、九星などの占星術のデータがあれば、相性などを表すことが可能です。
このように GraphDB は関係性を表すのに都合の良いデータベースです。
PostgreSQL や MySQL などの RDB と Neo4j や AWS Neptune、Azure Cosmos DB の簡単な比較を表にしてみました。

	GraphDB	RDB
特徴	ノード、エッジ、プロパティで定義され、データは相関図として表示される。 大量データの検索速度が速い。	全てのデータが表の中にあり、外部キーなどにより関係性を定義する。 テーブル間の様子は ER 図で表現。 Join で結合されたデータは処理速度が遅くなる傾向がある。
主なソフトウエア・サービス	AWS Neptune Azure Cosmos DB Neo4j	PostgreSQL Oracle MySQL MSSQL
アクセス言語	Gremline Cypher	SQL

GraphDB を使ったサービス

では、実際にどのようなサービスで GraphDB が使われているのでしょうか。
Facebook では「知り合いかも」や「おすすめのグループ」が関係性を辿って表示していますし、Amazon では「おすすめ」で表示されるものが参照履歴や購入履歴から辿って表示される GraphDB を使ったサービスの例になります。そして瞬時に表示されるスピードが GraphDB の持ち味の一つとなっています。

GraphDB の紹介のため架空のプロジェクトを表現してみました。

プロジェクト管理の中で誰がどのサービスを担当して、どれくらいの工数がかかっているかを最終目標に、あるプロジェクトの担当と人の関係を Neo4j の Desktop 環境に登録しています。
今回は工数部分を省いた内容です。（言語は Cypher を使用しています）
登録内容は Appendix に載せていますのでご参照ください。

●プロジェクトの担当
　プロジェクトのメンバーを以下のコマンドで登録していきます。
大量データを投入する場合は CSV 形式のデータも可能ですが、今回は以下の様に登録しました。
最初にプロジェクトを構成するメンバーを登録します。
登録内容は Appendix の「Person」を参照して下さい。
（本記事の中の人名やサービスは本記事用に作成したものです）

・登録コマンド

```
CREATE (:Person {name:”名前”,authority:”得意分野”,duties:”職掌”})
```

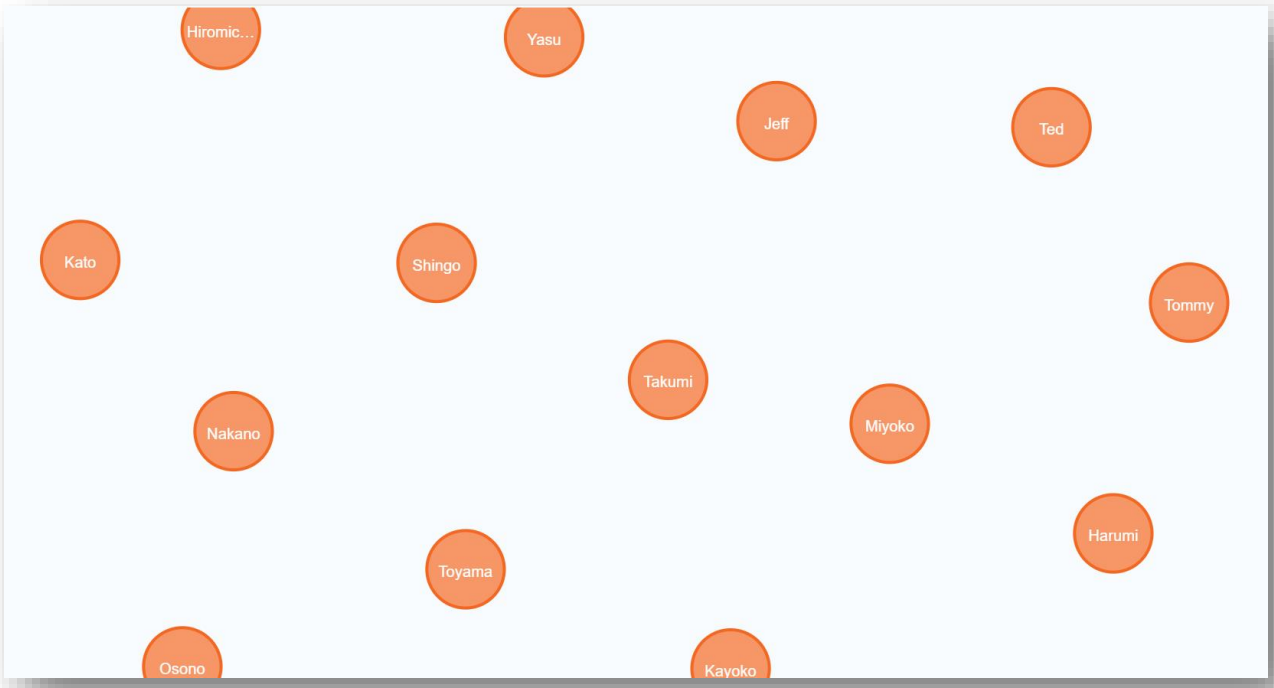
```
CREATE
        (:Person{name:"Taro",authority: "ADC", duties: "Manager"}),
        (:Person{name: "Jiro", authority: "Mail", duties: "Chief"}),
        (:Person{name:"Sugirin",authority: "Cloud", duties: "Manager"}),
        (:Person{name: "Takumi", duties: "Member"}),
                                                    :
```

登録した内容を表示してみます。
ラベルを指定してマッチした結果を表示します。
ここではラベルに Person を指定します。

```
MATCH (n:ラベル) RETURN n
```

コマンドは以下になります。

```
MATCH (n:Person) RETURN n
```



この段階ではノードがバラバラに表示されています。
次に人と人の関係を登録します。
登録内容は Appendix の「人と人の関係」を参照ください。
Taro さんは Jiro さんの上司(BOSS)と登録します。
この他チームのリーダーという意味で「Chief」という関係も登録します。

MATCH 文で登録されている人を検索して、関係を定義します。

```
MATCH (人名 1 :Person{name:"人名 1 "}),
      (人名 2 :Person{name:"人名 2 "})

CREATE (人名 1 )-[:BOSS]->(人名 2 )
```

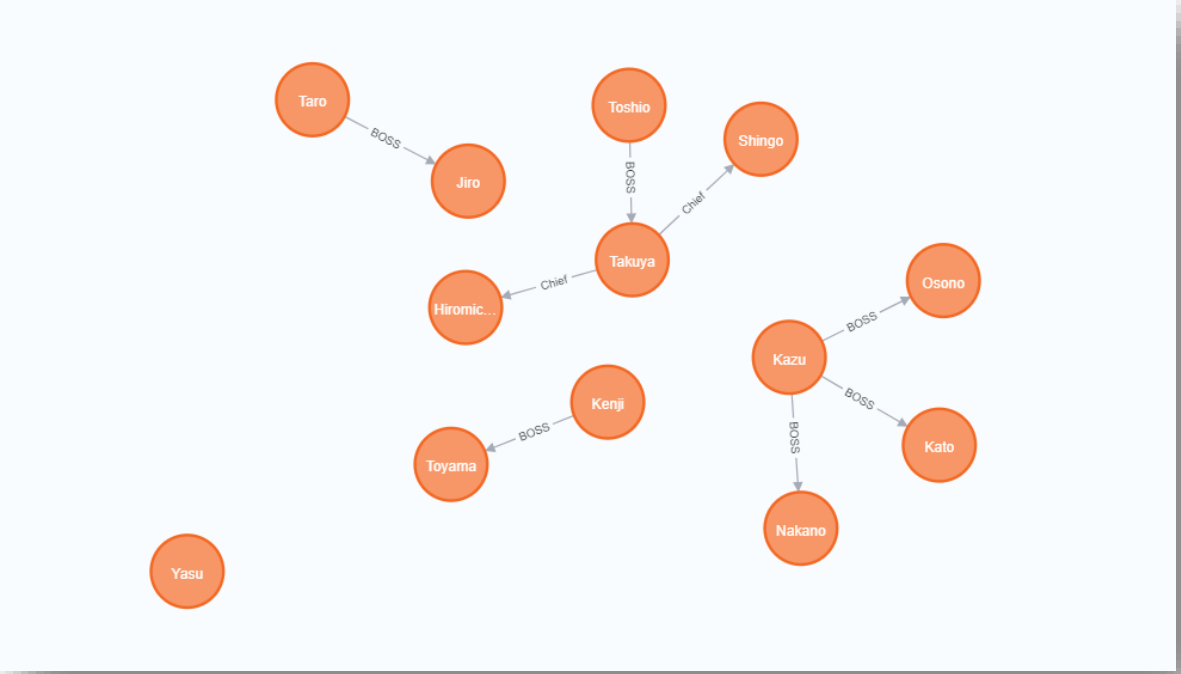
```
MATCH (Taro:Person{name:"Taro"}),
      (Jiro:Person{name:"Jiro"})

CREATE (Taro)-[:BOSS]->(Jiro)
```

登録した内容を表示してみます。
実行するコマンドは以下になります。

```
MATCH (n:Person) RETURN n
```

表示された内容には上司(BOSS)とリーダー(Chief)の関係が表示されています。
関係性を定義していない人は誰ともつながらない形で表示されています。



次にこのプロジェクトで運用しているサービスを登録します。
登録内容は Appendix の「Service」を参照して下さい。
コマンドは以下になります。

```
CREATE (:Services {name:"サービス名"})
```

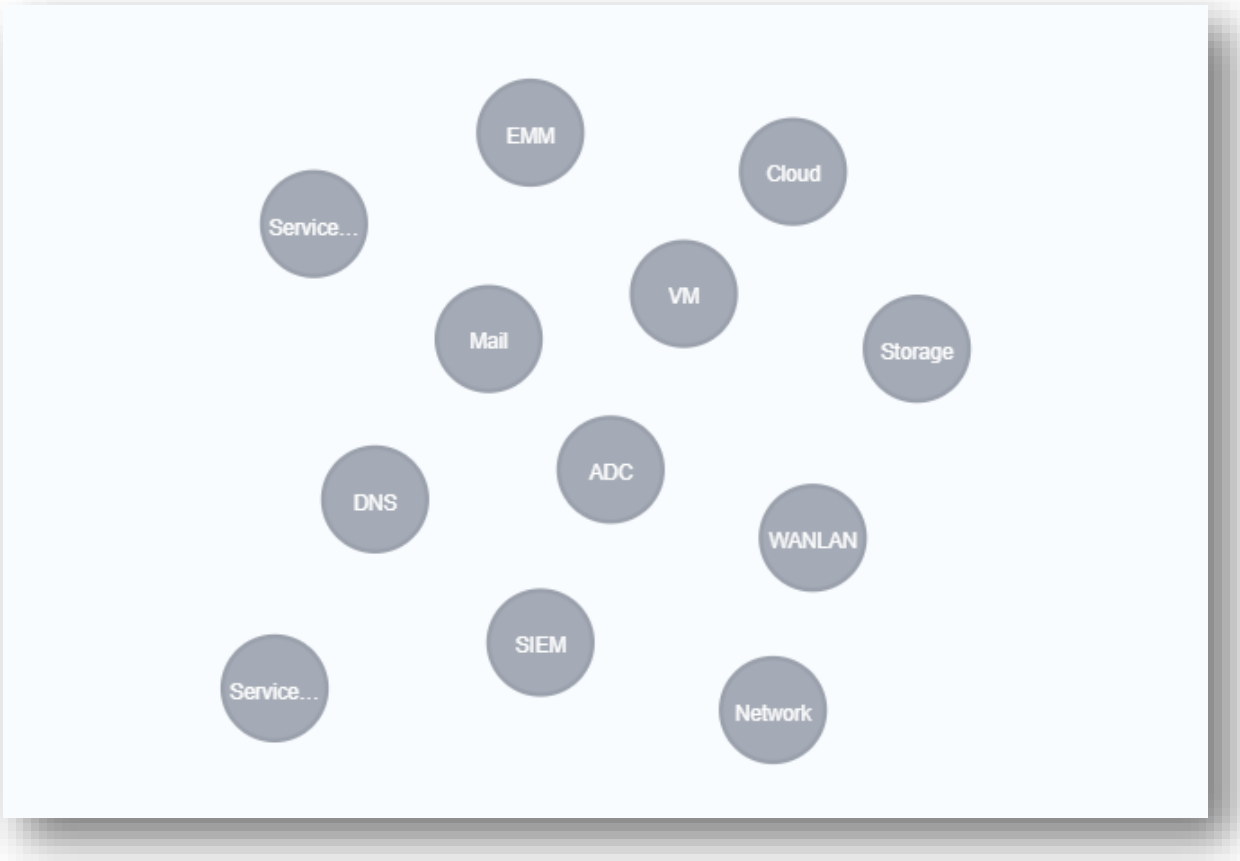
実行するコマンドは以下になります。

```
CREATE (:Services {name:"DNS"})
CREATE (:Services {name:'Mail'})
CREATE (:Services {name:"ADC"})
                    :
```

登録した内容を表示してみます。
実行するコマンドは以下になります。

```
MATCH (n:Services) RETURN n
```

登録したサービスだけが表示されています。



さて、いよいよ人と担当を関係づけていきます。
「A さんが xx サービスの担当」という形です。
担当は複数定義できますので、一人の人から複数のエッジが出ている事もあります。
登録内容は Appendix の「Person」にある「担当」という項目になります。
コマンドは以下になります。

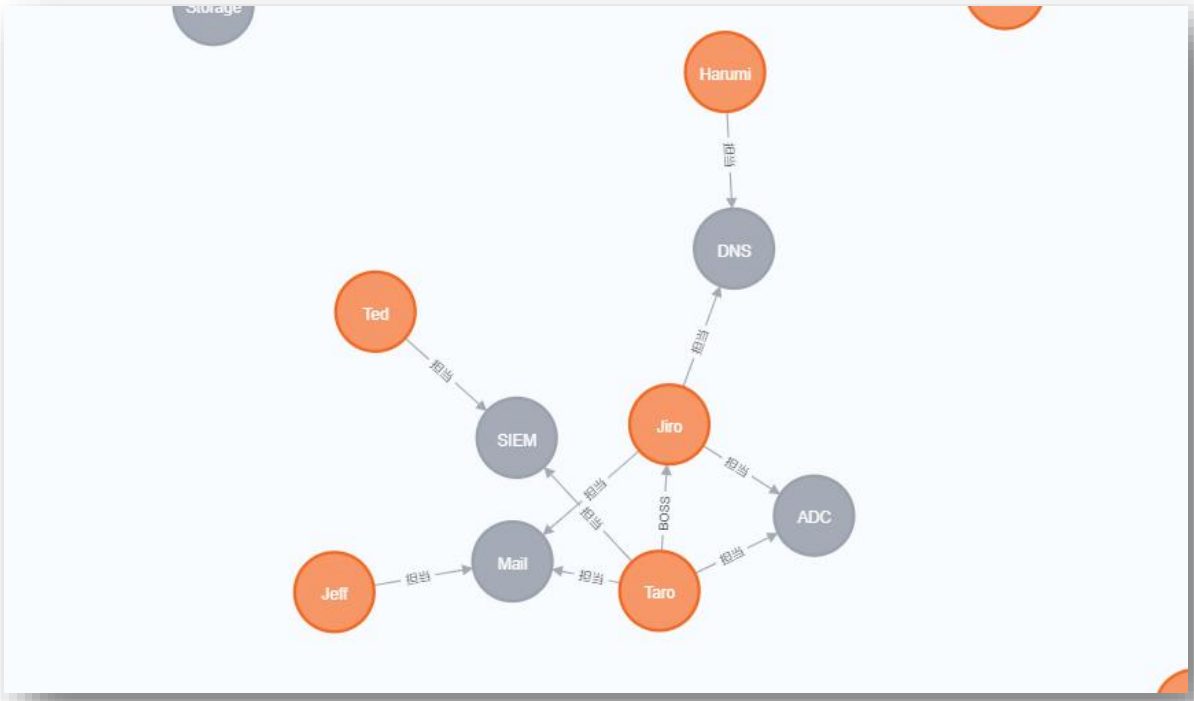
```
MATCH (人名:Person{name:"人名"}),
(SIEM:Services {name:"サービス名"})
CREATE (人名)-[:担当]->(サービス名)
```

```
MATCH (Taro:Person{name:"Taro"}),
(SIEM:Services {name:"ADC"})
CREATE (Taro)-[:担当]->(ADC)
                    :
```

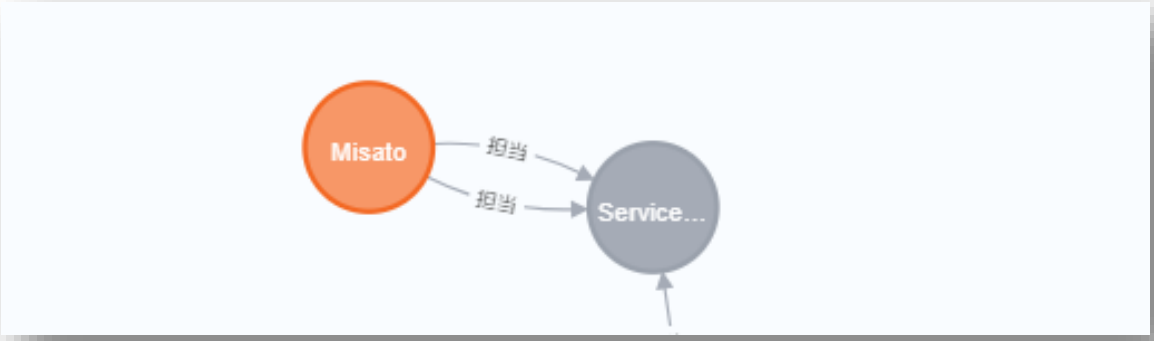
登録された内容を表示してみます。
実行するコマンドは以下になります。

```
MATCH P=()-[:担当]->() RETURN p
```

ここでは一部だけの表示になりますが、こんな感じで人の関係と担当が表現できます。
オレンジ色が担当者でグレーがサービス名になっています。
Taro は ADC、Mail、SIEM、Jiro は ADC、Mail、DNS を担当しています。
また、Taro は Jiro の上司（BOSS）になります。



ここで注意すべきことがあります。
「関係性を表現しているエッジ（丸と丸の間の線）は同じ内容を登録できてしまう」のです。
以下をご覧ください。
Misato さんから ServiceNow へ「担当」という線が二本出ています。
これは同じコマンドを二度実行した結果こうなってしまいました。
データ登録の際には二重登録にはくれぐれもお気を付けください。



さて、応用問題です。
運用サービスで良くあるのが、「ADC の設定で相談したいけど、誰に聞いたらいいの？」というような
ADC を得意としている人が分からない場面が出て来ます。
そういう時はこんなコマンドで探しましょう。

```
MATCH  (n:Person {authority:'ADC'}) RETURN n
```

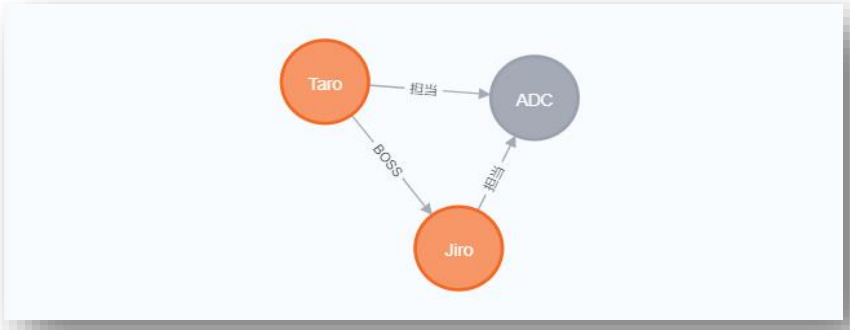
以下の様に Taro さんが表示されました。
Appendix の「Person」で Taro さんを見ると authority の欄に「ADC」とあります。

	name	authority	duties	担当
20	Taro	ADC	Manager	ADC,Mail,SIEM

では次に ADC を担当している人を全て表示します。
コマンドは以下になります。

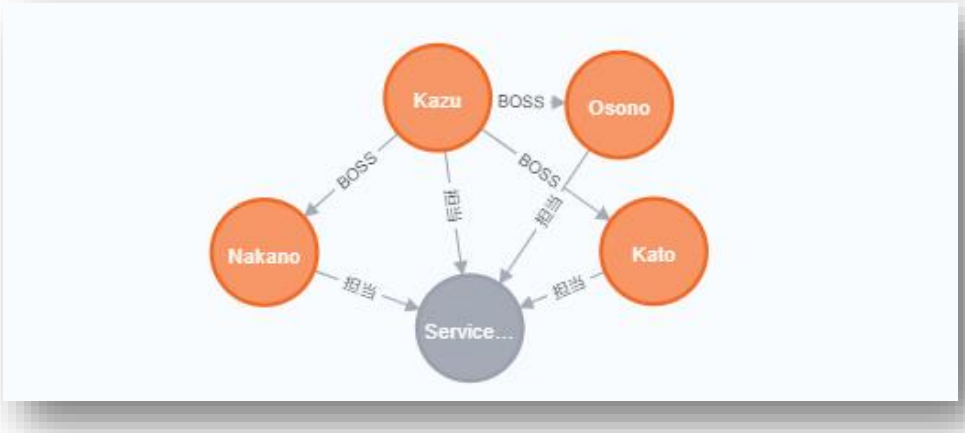
```
MATCH (p:Person)-[:担当]->(n:Services) WHERE n.name='ADC'  RETURN p, n
```

Taro さんと Jiro さんが上司と部下の関係にあり、それぞれ ADC を担当していることが分かります。



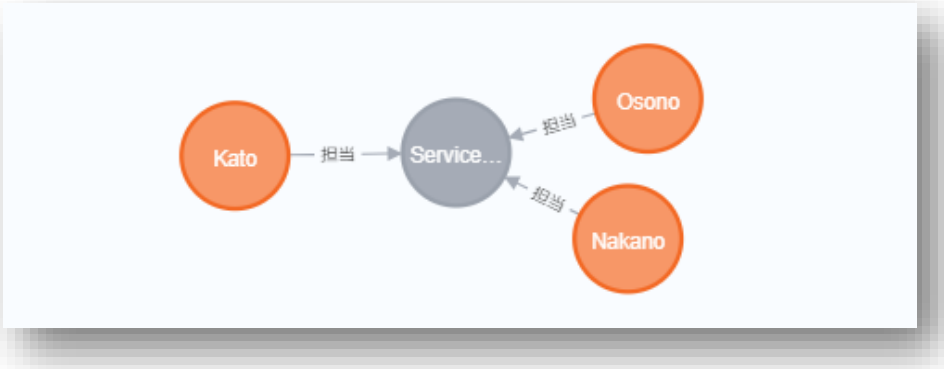
同様に ServiceDesk の担当者を表示してみます。コマンドは以下の通りです。

```
MATCH (p:Person)-[:担当]->(n:Services) WHERE n.name='ServiceDesk'  RETURN p, n
```



もちろん、業務担当だけの表示も可能です。
その場合は次のように NOT を使って filter します。
「NOT (p:Person)-[:BOSS]->()」で BOSS という関係を非表示にしています。

```
MATCH (p:Person)-[:担当]->(n:Services)
WHERE n.name='ServiceDesk' AND NOT (k:Person)-[:BOSS]->() RETURN p, n,k
```



これからのこと

いかがだったでしょうか。
簡単ではありますが GraphDB の一端をお見せできたかと思います。
ノードやノード間に関係性を追記する事で様々な関係を表現が可能になります。
GraphDB の特徴として関係性の検索が高速であるため、facebook や Yahoo!、eBay などの商用サービスに埋め込まれたコンテンツでも瞬時に表示することが可能です。
今後の予定として私が所属しているプロジェクトでサービス毎の作業時間を投入して作業の偏り補正や、配員計画に役立てようと思っています。
インフラ情報を入力して、各機器がどういうつながりを持っているかを明確にする事で、障害発生時に影響範囲の把握も容易になるはずです。
どの DB にも言える事ですが、実際にお客様へ提案する際には用途と目的により得意な分野で使う事が肝心です。
SRA のアドバンスクラウドエンジニアリング事業部はクラウドを扱う部署ですが、インフラ、ミドルウエアに興味を持つ人が増えて、切磋琢磨できるようになれば良いな、と思っています。今後は構築から運用まで出来る人材がますます重宝されると思いますので、本稿が少しでも興味の対象になればと思います。

以上

Appendix

架空のプロジェクトを想定して各ラベルの内容を記述しています。
Person にある「担当」は関係性になりますが、分かりやすいように表に記述しました。

◆Person

	name	authority	duties	担当
1	Harumi		Member	DNS
2	Hiromichi		Member	Network
3	Jeff		Member	Mail
4	Jiro	Mail	Chief	Mail,ADC,DNS
5	Kato		Member	ServiceDesk
6	Kayoko		Member	VM
7	Kazu	ServiceDesk	Manager	ServiceDesk
8	Kenji	VM	Manager	VM,EMM
9	Koji		Member	EMM
10	Kyoko	EMM	Chief	EMM,VM
11	Misato		Member	ServiceNow
12	Miyoko		Member	VM
13	Nakano		Member	ServiceDesk
14	Osamu		Member	Storage
15	Osono		Chief	ServiceDesk
16	Shingo		Member	Network
17	Sugirin	Cloud	Manager	Cloud
18	Takumi		Member	Cloud
19	Takuya	Network	Chief	Network
20	Taro	ADC	Manager	ADC,Mail,SIEM
21	Ted		Member	SIEM
22	Tommy		Member	ADC
23	Toshio	ServiceNow	Manager	ServiceNow,Network
24	Toyama		Chief	EMM
25	Yasu		Member	Storage

◆Service

No	Service
1	Mail
2	ServiceDesk
3	VM
4	EMM
5	Cloud
6	Network
7	ADC
8	ServiceNow
9	DNS
10	SIEM
11	WANLAN

◆人と人の関係

	Person1	Relation->	Person2
	Taro	BOSS	Jiro
	Kazu	BOSS	Osono
	Kazu	BOSS	Nakano
	Kazu	BOSS	Kato
	Toshio	BOSS	Takuya
	Kenji	BOSS	Toyama
	Takuya	Chief	Hiromichi
	Takuya	Chief	Shingo

◆参考文献

・https://neo4j.com/

・はじめての GraphDB Amazon Neptune
https://dev.classmethod.jp/articles/first-time-graphdb-amazon-neptune/

・パナマ文書解析の技術的側面
https://medium.com/@c_z/%E3%83%91%E3%83%8A%E3%83%9E%E6%96%87%E6%9B%B8-%E8%A7%A3%E6%9E%90%E3%81%AE%E6%8A%80%E8%A1%93%E7%9A%84%E5%81%B4%E9%9D%A2-d10201bbe195

・The Neo4j Cypher Manual v4.4

https://neo4j.com/docs/cypher-manual/current/

GSLetterNeo Vol.162
2022 年 1 月 20 日発行
発行者　株式会社 SRA 先端技術研究所

編集者　熊澤努　方 学芳
バックナンバー https://www.sra.co.jp/public/sra/gsletter/
お問い合わせ　gsneo@sra.co.jp

株式会社SRA
〒171-8513 東京都豊島区南池袋 2-32-8

夢を。Yawaraka Innovation
やわらかいのべーしょん