

Neo4j: Tanssikanta

Jun Fengari

Contents

Tanssikanta	2
Tietomalli	3
Asennusohjeet	3
Esimerkkidatan Lisääminen	4
Tulos	6
Käyttötapauksia ja Kyselyitä	7
Reflektointi	8
Haasteet	8
Jatkokehitys	8

Tanssikanta

Tanssikanta on graafitietokanta, jonka avulla käyttäjä voi tarkastella tanssijoiden suhteita eri tanssiryhmiin sekä näiden tanssiryhmien tulevia esityksiä, tapahtumia ja aikatauluja.

Tässä tietokannassa esiintyy useita many-to-many -suhteita, ja se on tyypillinen monisuhteinen käyttötapaus, joka soveltuu hyvin Neo4j-grafitietokannan hyödyntämiseen.

Tätä tietokantaa voidaan hyödyntää esimerkiksi sovelluksessa, jossa käyttäjä voi etsiä eri tanssijoita, seurueita, koreografioita ja tapahtumia sekä tarkastella niiden suhteita toisiinsa.

Tanssikannan suhteet:

- Tanssija -[TANSSII]-> Tanssilaji
- Tanssija -[KUULUU]-> Seurue
- Seurue -[ESITTÄÄ]-> Koreografia
- Koreografia -[EDUSTAA]-> Tanssilaji
- Seurue -[OSALLISTUU]-> Tapahtuma
- Tapahtuma -[SISÄLTÄÄ]-> Koreografia
- Tapahtuma -[PIDETÄÄN]-> Tapahtumapaikka

Suhteiden kardinalisuudet:

Tanssija: Oletetaan että tanssija voi kuulua kerrallaan vain yhteen seurueeseen.

Tanssija voi tanssia useita eri tanssilajeja.

Seurue: Seurueeseen kuuluu useita tanssijoita, ja seurueet voi esittää useata koreografiaa. Eri seurueet voivat esittää samaa koreografiaa. Seurueet voivat osallistua useampaan tapahtumaan.

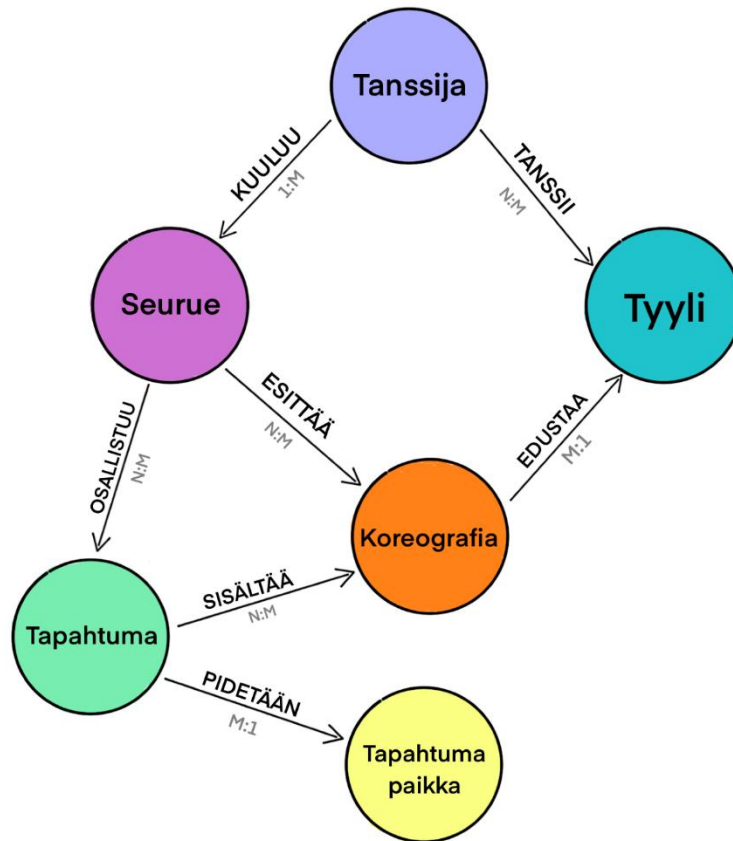
Koreografia: Yksinkertaisuuden vuoksi oletetaan, että koreografia edustaa vain yhtä tanssilajia.

Tapahtuma voi sisältää useita koreografioita, ja tapahtumaa pidetään vain yhdessä tapahtumapaikassa. Useat seurueet voivat osallistua samaan tapahtumaan.

Tanssija	<- many-to-many ->	Tanssilaji
Tanssija	<- many-to-one ->	Seurue
Seurue	<- many-to-many ->	Koreografia
Seurue	<- many-to-many ->	Tapahtuma
Koreografia	<- many-to-one ->	Tanssilaji
Tapahtuma	<- many-to-many ->	Koreografia
Tapahtuma	<- many-to-one ->	Tapahtumapaikka

Tietomalli

Tietomallissa on käytetty tanssilajin (Laji) sijaan Tyyli, ja Paikan sijaan Tapahtumapaikka. Tietokantatoteutuksessa selkeytin nimityksiä.



Asennusohjeet

1. Käy <https://github.com/JunFengari/Tanssikanta-Neo4j> sivuilla, ja kloonaa repositorio haluamaasi hakemistoon seuraavalla komennolla:
`git clone https://github.com/JunFengari/Tanssikanta-Neo4j.git`
2. Käynnistä hakemistossa Neo4j Docker kontin komennolla:
`docker compose up -d`
3. Kun kontti on käynnistynyt, avaa linkki <http://localhost:7474/browser/>
4. Avaa hakemistossa olevan `tanssikannan_luonti.cypher` -tiedoston, ja kopio sen koko sisältö Neo4jn komentoriville ja suorita komennot. Tämän jälkeen tietokanta pitäisi näkyä.
Voit hahmoittamisen avuksi laittaa eri värejä solmujen labelleille, kuten on esitetty alempana tulokuvissa. Valmiit kyselyt löytyy `kyselyt.cypher` tiedostosta.

Esimerkkidatan Lisääminen

Huomio! Tässä on versio tietokannan alustamisen komennoista dokumentointia varten, mutta käytäthän cypher tiedostojen versioita formatointiongelmien välttämiseksi. Sama pätee kyselyihin.

```
// Poistetaan kaikki aiemmat tiedot Neo4jstä.
```

```
MATCH (n)
```

```
DETACH DELETE n;
```

```
// -----
```

```
// TANSSIJAT
```

```
CREATE
```

```
(stella:Tanssija {nimi: "Stella", ika: 22, kokemusVuodet: 10}),  
(roberto:Tanssija {nimi: "Roberto", ika: 28, kokemusVuodet: 15}),  
(monique:Tanssija {nimi: "Monique", ika: 32, kokemusVuodet: 5}),  
(stefan:Tanssija {nimi: "Stefan", ika: 44, kokemusVuodet: 25}),  
(rebecca:Tanssija {nimi: "Rebecca", ika: 15, kokemusVuodet: 3}),  
(alessandra:Tanssija {nimi: "Alessandra", ika: 34, kokemusVuodet: 20}),
```

```
// -----
```

```
// SEURUEET
```

```
(vividty:Seurue
```

```
{nimi: "Vividty", perustettu: 1965, kotipaikka: "Helsinki"}),
```

```
(joutsenet:Seurue
```

```
{nimi: "Joutsenet", perustettu: 1865, kotipaikka: "Tampere"}),
```

```
(sambacha:Seurue {nimi: "Sambacha", perustettu: 2021, kotipaikka: "Joensuu"}),
```

```
// -----
```

```
// LAJIT
```

```
(bachata:Laji {nimi: "Bachata", alkupera: "Dominican Republic"}),
```

```
(baletti:Laji {nimi: "Baletti", alkupera: "Italia"}),
```

```
(valssi:Laji {nimi: "Valssi", alkupera: "Itävalta"}),
```

```
// -----
```

```
// KOREOGRAFIA
```

```
(swanlake:Koreografia {nimi: "Swan Lake"}),
```

```
(viennesewaltz:Koreografia {nimi: "Viennese Waltz"}),
```

```
(modernabachata:Koreografia {nimi: "Moderna Bachata"}),
```

```
(nutcracker:Koreografia {nimi: "NutCracker"}),
```

```
// -----
```

```
// TAPAHTUMA
```

```
(balettiviikot:Tapahtuma
```

```
{nimi: "Baletti Viikot", ajankohta: date("2026-03-25"}),
```

```
(tanssimania:Tapahtuma {nimi: "TanssiMania", ajankohta: date("2027-06-03"}),
```

```
(dancefest:Tapahtuma {nimi: "DanceFest", ajankohta: date("2026-09-15"}),
```

```
// -----
```

```
// PAIKKA
```

```
(kansallisteatteri:Paikka {nimi: "Kansallisteatteri", kaupunki: "Helsinki"}),
```

```
(oopperatalo:Paikka {nimi: "Ooppera Talo", kaupunki: "Helsinki"}),
```

```
(tampereenteatteri:Paikka {nimi: "Tampereen Teatteri", kaupunki: "Tampere"}),
```

// -----

// SUHTEET

```
(stella)-[:KUULUU]->(vividty),
(roberto)-[:KUULUU]->(joutsenet),
(monique)-[:KUULUU]->(sambacha),
(stefan)-[:KUULUU]->(joutsenet),
(rebecca)-[:KUULUU]->(vividty),
(alessandra)-[:KUULUU]->(joutsenet),
(stella)-[:TANSSII]->(baletti),
(roberto)-[:TANSSII]->(baletti),
(monique)-[:TANSSII]->(bachata),
(stefan)-[:TANSSII]->(baletti),
(rebecca)-[:TANSSII]->(baletti),
(rebecca)-[:TANSSII]->(valssi),
(alessandra)-[:TANSSII]->(baletti),
(vividty)-[:ESITTAA]->(viennesewaltz),
(joutsenet)-[:ESITTAA]->(nutcracker),
(joutsenet)-[:ESITTAA]->(swanlake),
(sambacha)-[:ESITTAA]->(modernabachata),
(swanlake)-[:EDUSTAA]->(baletti),
(nutcracker)-[:EDUSTAA]->(baletti),
(modernabachata)-[:EDUSTAA]->(bachata),
(viennesewaltz)-[:EDUSTAA]->(valssi),
(balettiiviikot)-[:SISALTAA]->(swanlake),
(balettiiviikot)-[:SISALTAA]->(nutcracker),
(tanssimania)-[:SISALTAA]->(modernabachata),
(dancefest)-[:SISALTAA]->(viennesewaltz),
(balettiiviikot)-[:PIDETAAN]->(oopperatalo),
(tanssimania)-[:PIDETAAN]->(tampereenteatteri),
(dancefest)-[:PIDETAAN]->(kansallisteatteri),
(vividty)-[:OSALLISTUU]->(dancefest),
(joutsenet)-[:OSALLISTUU]->(balettiiviikot),
(sambacha)-[:OSALLISTUU]->(tanssimania),
(sambacha)-[:OSALLISTUU]->(dancefest);
```

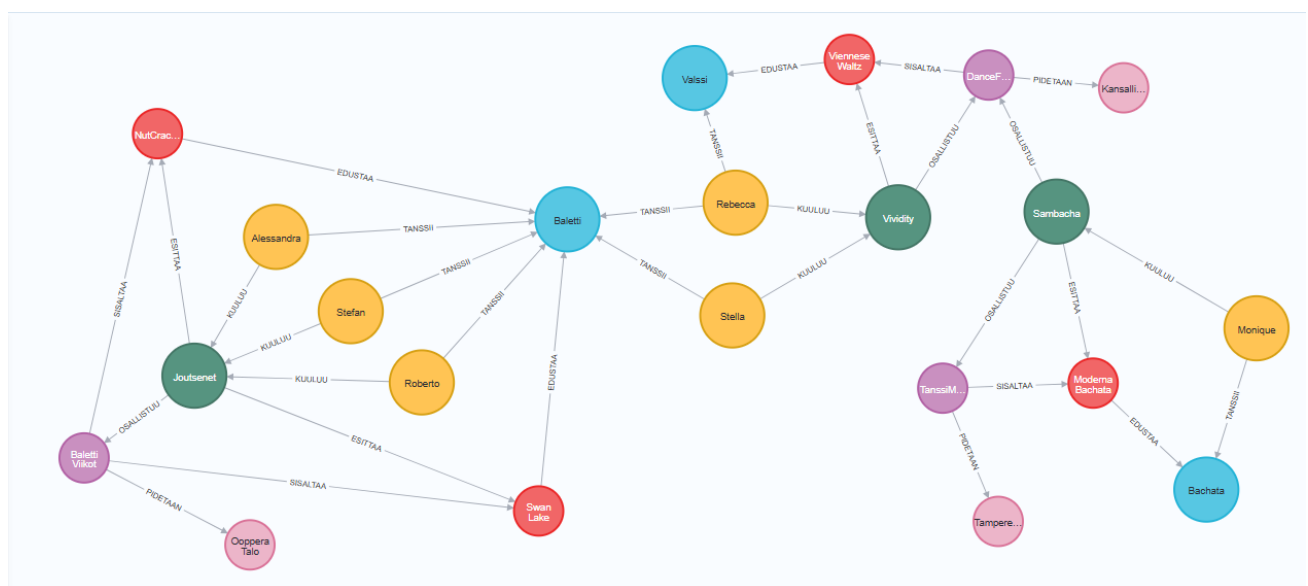
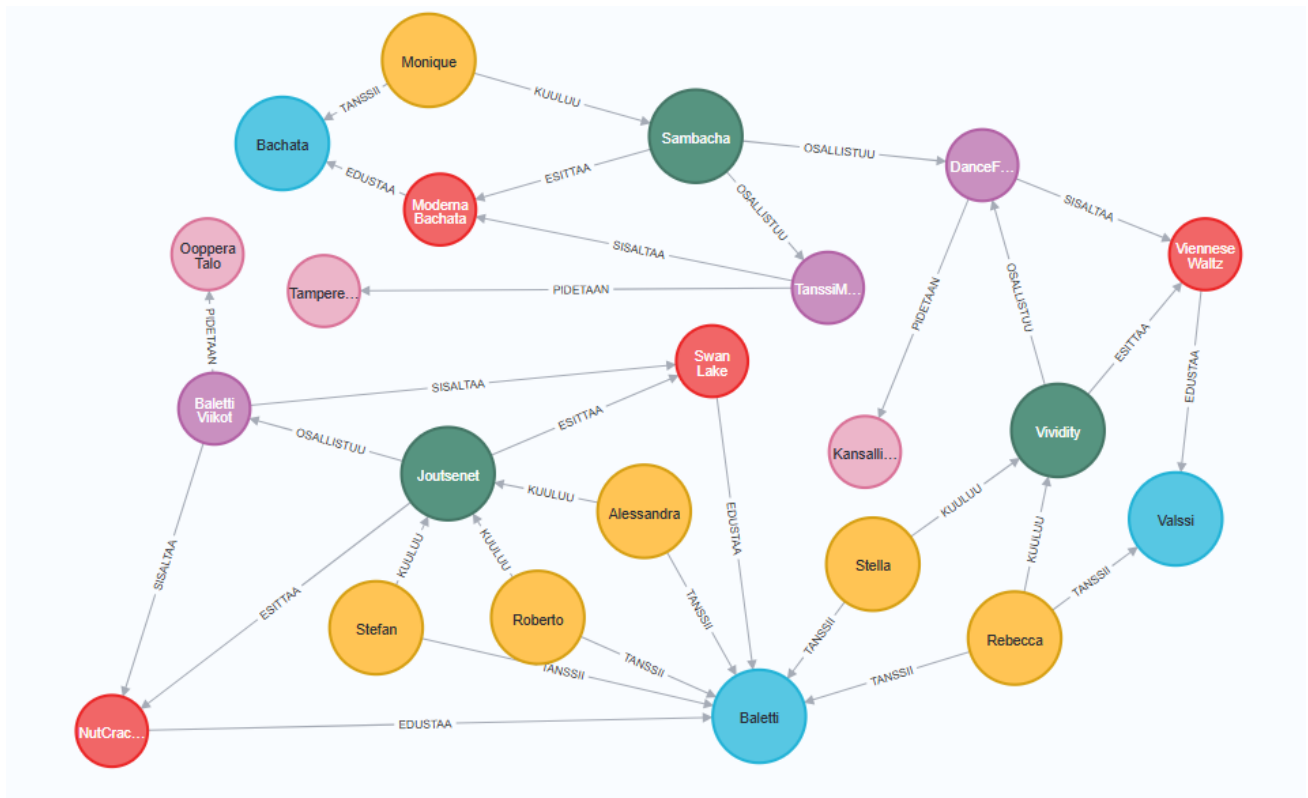
// -----

// Lopulta muiden kommentojen jälkeen näytetään kaikki tulokset

MATCH (n)

RETURN n;

Tulos



Käyttötapauksia ja Kyselyitä

Kyselyt löytyy kyselyt.cypher tiedostosta.

1. Näytä kaikki seurueet jotka esiintyvät Tampereella.

```
MATCH
(s:Seurue)-[:OSALLISTUU]->
(t:Tapahtuma)-[:PIDETAAN]->
(p:Paikka {kaupunki: "Tampere"})
RETURN DISTINCT s.nimi AS TampereellaEsiintyvät;
```

tulos: Sambacha on ainut seurue, joka esiintyy Tampereella.

2. Etsi kaikki balettia edustavat koreografiat.

```
MATCH (k:Koreografia)-[:EDUSTAA]->(laji:Laji {nimi: "Baletti"})
RETURN k.nimi AS Koreografia;
```

tulos: Swan Lake ja NutCracker ovat balettia edustavia koreografioita.

3. Mitkä tanssijat kuuluu Joutsen seureeseen?

```
MATCH (t:Tanssija)-[:KUULUU]->(s:Seurue {nimi: "Joutsenet"})
RETURN t.nimi AS JoutsenenTanssijat;
```

tulos: Roberto, Stefan ja Alessandra kuuluvat Joutsen seureeseen.

4. Sinun suosikki tanssija on Roberto, missä voit nähdä hänet esiintymässä?

```
MATCH
(rob:Tanssija {nimi: "Roberto"})-[:KUULUU]->
(s:Seurue)-[:OSALLISTUU]->
(tap:Tapahtuma)-[:PIDETAAN]->
(p:Paikka)
RETURN tap.nimi AS Tapahtuma, p.nimi AS Paikka, p.kaupunki AS Kaupunki;
```

tulos: Roberto esiintyy Baletti Viikoilla.

5. Missä tapahtumissa esitetään Bachataa?

```
MATCH
(tap:Tapahtuma)-[:SISALTAA]->
(k:Koreografia)-[:EDUSTAA]->
(laji:Laji {nimi: "Bachata"})
RETURN DISTINCT tap.nimi AS Tapahtuma;
```

tulos: Tanssimania.

6. Kuinka monessa tapahtumassa jokainen ryhmä esiintyy?

```
MATCH (s:Seurue)-[:OSALLISTUU]->(tap:Tapahtuma)
WITH s, COUNT(tap) AS tapahtumienMaara
RETURN s.nimi AS Seurue, tapahtumienMaara;
```

tulos: Vividity 1, Joutsenet 1, Sambacha 2.

7. Päivitä Rebeccan tanssikokemusta 6 vuoteen

```
MATCH (rebu:Tanssija {nimi: "Rebecca"})
SET rebu.kokemusVuodet = 6
RETURN rebu;
```

8. Stefan jää eläkkeelle, poistetaan hänet tietokannasta.

```
MATCH (t:Tanssija {nimi: "Stefan"})
DETACH DELETE t;
```

Reflektointi

Tässä osiossa pohdin tietokannassa esiintyviä haasteita ja jatkokehityksen ideoita. Kerron myös lopussa projektin aikana tekoälyn käytöstä ja käytetystä ajasta.

Haasteet

Projektin kasvaessa tulisi vastaan seuraavat haasteet:

1. Tietokannassa on looginen haaste.
Nykyinen struktuuri ei takaa tai kerro, mitä koreografiaa seurue esittää tapahtumassa. Kuvitellaan, että tapahtumassa on yhteensä 10 koreografiaa ja seurue esittää vaikka 2 näistä, mutta emme voi välttämättä tietää, mitä niistä he esittävät.
Tämän voisi korjata lisäämällä uuden solmun, esimerkiksi "esitys", joka tulee seurueen ja tapahtuman väliin. Päätin olla käyttämättä tätä, sillä se monimutkaistaa tietokannan struktuuria tämän projektin puitteiden ulkopuolelle.
2. Tällä hetkellä tietokanta toimii tietyillä oletuksilla, esimerkiksi että tanssijat kuuluvat vain yhteen ryhmään ja että kaikki seureen tanssijat osallistuu kaikkiin esityksiin.
Haasteita saattaa nousta struktuurin monimutkaistumisen kautta, esimerkiksi jos tarvitaan tapahtumapaikoille vuosittain uusiutuvia tapahtumia, esityksiin osallistuu vierastähtiä, useammat yhtymät esiintyvät yhdessä tai tekevät yhteistyötä, tai koreografiat ovat moniulotteisempia kuin pelkästään yhteen lajiin kuuluvia.
Struktuurin kasvaessa myös useammat suhteet monimutkaistuvat, mikä voi johtaa ylläpito-ongelmiin.
3. Dense nodes -ilmiö voi myös tulla ongelmaksi. Tässä struktuurissa ilmenee jo nyt monta N:M-suhdetta, ja tämän kasvaessa esimerkiksi isot tanssifestarit, tai sadat baletin harrastajat voivat aiheuttaa dense nodeja.

Jatkokehitys

Kehityskohteina tähän projektiin voisi esimerkiksi olla rajoitusten lisääminen tietokannan luontivaiheessa, esimerkiksi:

// Varmistetaan, että koreografia teoksen nimi on uniikki

```
CREATE CONSTRAINT koreo_nimi_uniikki IF NOT EXISTS FOR (kor:Koreografia)  
REQUIRE kor.nimi IS UNIQUE
```

Sovelluksen käyttötarkoituksen mukaan voidaan myös lisätä indeksejä.

Omissa kyselyissä käytin useammiten Tanssija.nimi, Laji.nimi ja myös kaupungin nimeä. Kaikille useammin käytetyille hauille voidaan luoda indeksejä esimerkiksi tällä tavalla:


```
CREATE INDEX tanssija_nimi_index IF NOT EXISTS  
FOR (t:Tanssija)  
ON (t.nimi);
```

Tämä parantaa tehokkuutta ja skaalautuvuutta isommissa tietokannoissa.

Jatkokehityksen mielessä tietokantaan voi lisätä kaiken muun mainitun lisäksi myös lisätietoja jokaiseen solmuun ja myös suhteeseen. Esimerkiksi seureen ja tapahtuman väliselle OSALLISTUU-suhteelle voidaan lisätä esityksen ajankohta.

Yhteenveto

Tätä tietokantamallia voisi hyödyntää erityisesti pienemmissä sovelluksissa. Projektin laajentuessa tietokanta kuitenkin vaatisi tarkempaa rakenteellista suunnittelua skaalautuvuuden varmistamiseksi sekä sovelluksen päätarkoituksen tehokkaaksi tukemiseksi.

Käytin tekoälyä tämän projektin aikana ideointiin, loogisten ongelmien ratkomiseen, haasteiden pohdintaan, ja syntaksien korjaamiseen. Projektiin kului yhteensä noin 8 tuntia työskentelyaikaa.
