Big Data

Graph Databases

von
Benjamin Ellmer (S2210455012)

Mobile Computing Master
FH Hagenberg

May 2, 2023

Installation

Start Neo4j

```
git clone https://github.com/Digital-Media/neo4j.git
docker compose -f neo4j/docker-compose.yml up -d
```

Start Postgres

```
docker run --name postgres-big-data-ex4 -e
   POSTGRES_PASSWORD=geheim -d postgres:14
```

Step 1 - Translation

Drop, create and select schema:

```
DROP SCHEMA IF EXISTS graph_demos CASCADE;
CREATE SCHEMA IF NOT EXISTS graph_demos;
SET search_path TO graph_demos;
```

Create folks table:

```
CREATE TABLE IF NOT EXISTS folks (
    id bigint NOT NULL,
    name varchar(100) NOT NULL,
    father bigint NULL,
    mother bigint NULL,
    PRIMARY KEY (id),
    CONSTRAINT father_fk FOREIGN KEY (father) REFERENCES
       folks(id),
    CONSTRAINT mother_fk FOREIGN KEY (mother) REFERENCES
       folks(id)
);
```

Insert folks:

```
INSERT INTO folks (id, name, father, mother) VALUES
(100, 'Alex', 20, 30),
(20, 'Dad', 10, null),
(30, 'Mom', null, null),
(10, 'Grandpa Bill', null, null),
(98, 'Sister Amy', 20, 30);
```

Create vertices table:

```
CREATE TABLE vertices (
    vertex_id bigint NOT NULL,
    alias varchar (255),
    label varchar (255),
    name varchar (255),
    type varchar (255),
    properties jsonb,
    PRIMARY KEY (vertex_id)
);
```

Insert vertices:

```
INSERT INTO vertices (vertex_id, alias, label, name, type)
   VALUES
(1, 'NAmerica', 'Location', 'North America', 'continent'),
(2, 'Europe', 'Location', 'Europe', 'continent'),
(3, 'USA', 'Location', 'United States', 'country'),
(4, 'UK', 'Location', 'United Kingdom', 'country'),
(5, 'England', 'Location', 'England', 'country'),
(6, 'Austria', 'Location', 'Österreich', 'country'),
(7, 'Idaho', 'Location', 'Idaho', 'state'),
(8, 'London', 'Location', 'London', 'city'),
(9, 'UpperAustria', 'Location', 'Oberösterreich', '
   Bundesland'),
(10, 'Waldviertel', 'Location', 'Waldviertel', 'Viertel'),
(11, 'Grein', 'Location', 'Grein', 'city'),
(12, 'Andrea', 'Person', 'Andrea', 'person'),
(13, 'Bert', 'Person', 'Bert', 'person'),
(14, 'Christian', 'Person', 'Christian', 'person');
```

Create edges table:

```
CREATE TABLE edges (
    edge_id bigint NOT NULL,
    tail_vertex bigint REFERENCES vertices (vertex_id),
    head_vertex bigint REFERENCES vertices (vertex_id),
    label varchar(255),
    properties jsonb,
    PRIMARY KEY (edge_id),
    CONSTRAINT tail_vertex_fk FOREIGN KEY (tail_vertex)
       REFERENCES vertices(vertex_id),
    CONSTRAINT head_vertex_fk FOREIGN KEY (head_vertex)
       REFERENCES vertices(vertex_id)
);
```

Insert edges:

```
INSERT INTO edges (edge_id, tail_vertex, head_vertex, label
   ) VALUES
(1, 3, 1, 'within'),
(2, 4, 2, 'within'),
(3, 5, 4, 'within'),
(4, 6, 2, 'within'),
(5, 7, 3, 'within'),
(6, 8, 5, 'within'),
(7, 9, 6, 'within'),
(8, 10, 9, 'within'),
(9, 11, 10, 'within'),
(10, 12, 7, 'born_in'),
(11, 12, 8, 'lives_in'),
(12, 13, 11, 'born_in'),
(13, 13, 8, 'lives_in'),
(14, 14, 8, 'born_in'),
(15, 12, 13, 'married'),
(16, 13, 12, 'married');
```