Michael Sheroubi

Data Generation Report

19 November 2019

Data Generation Script

Language: Python 3

Libraries: Faker, random

Dependencies: common-verbs.txt common-nouns.txt common-interests.txt

Breakdown:
The data generation is divided up into different functions, each responsible for generating data for a specific table or node. Some cypher relationships are generated alongside the nodes they connect to ensure consistency between SQL Foreign keys and Cypher relationships. Each function can take a set of parameters; (n) is the number of tuples the function should create, (x, y, z) each holds the number of one of (Users, Groups, Events, Interests, Posts) to make sure that a relationship or foreign key does not reference a node that does not exist. Each table/node is indexed by an incremental integer.
The script creates three files; sql_file, cypher_node_file, cypher_rel_file. The sql_file contains the INSERTS for every table in order to watch for dependencies. The cypher_node_file contains the CREATE node statements for every node, and the cypher_rel_file contains the MATCH CREATE statements that match the two nodes to connect, then creates a relationship between them.
The final function header *generateAllData* takes in a number for each table and generates the data accordingly.

Known Issues: Can generate duplicates.

Systems Information

Relational Model

DBMS: SQL SERVER Management Studio

Host: localhost

Related Files: sql_data.sql

Graph Model

DBMS: Neo4J Browser

Host: localhost

Related Files: cypher_node_data.cql cypher_rel_data.cql

TEST CASE

Parameters: 25 Users, 5 Groups, 10 Events, 10 Interests, 50 Posts, 50 Comments, 50 Group Memberships, 50 Friendships, 100 hasInterests

Raw Output: cypher_node_data.cql cypher_rel_data.cql sql_data.sql

Sample Images: sample01_images.zip
This contains screenshots of relations from the relational model and a couple from the graph model. Visualizing the graph model with this many nodes and relationships gets messy.

Note: sendsMessage and isAttending data not included. (Forgot to call them in the generateAllData, the issue has been fixed.

NOTES

- **Data loads/insertion is exponentially faster in SQL Server than Neo4j**
- **Creating 100 nodes in Neo4j using browser took about 2 minutes**
- **Creating 365 relationships in Neo4j using its browser took 50+ minutes (55:11)**

- **Long create query execution time in cypher might be from using neo4j browser**
- **Querying and visualizing the data in Neo4j browser takes a few seconds**

KNOWN ISSUES

1. **Duplicates can be generated,** but they get caught by SQL Server but not by neo4j. This is more of an issue now since the number of nodes is very small; 25 Users, 5 Groups, 10 Events, 10 Interests, 50 Posts, 50 Comments, 50 Group Memberships, 50 Friendships, 100 hasInterests