Solution Architect Interview Questions Based on GSQL

Introduction

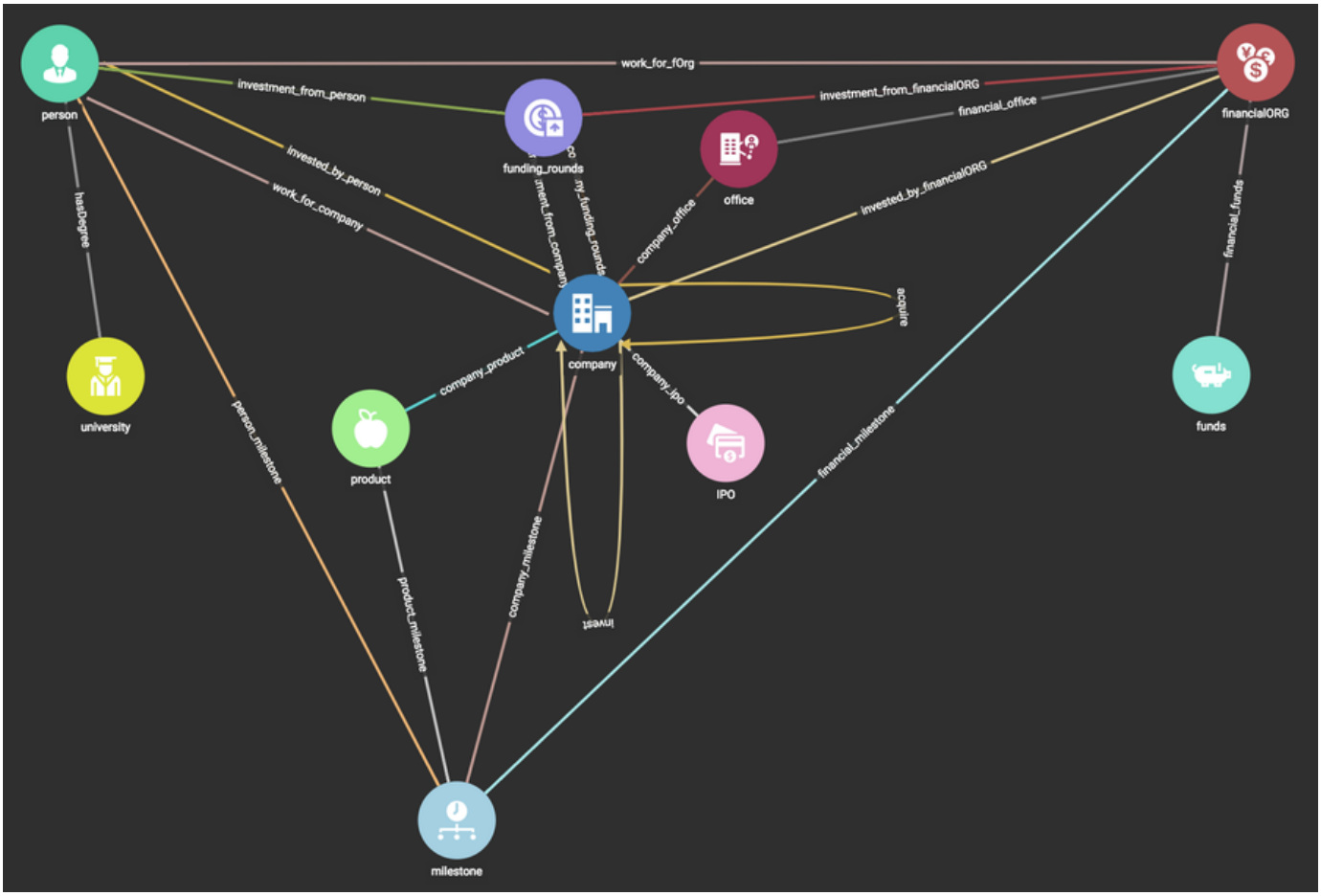
This article lists several GSQL query assignments on a real data set--crunchbase pre-2013 open source data. This data set describes funding history of companies. It records the relationship between the financial organizations, the investors, the entrepreneurs, and the companies.

When working on this exercise, please refer to our online GSQL tutorial at http://doc.tigergraph.com/start/

DISCLAIMER: Data is from Crunchbase 2013 Snapshot under the Creative Commons Attribution License [CC-BY].

Crunchbase Data Schema

Below is the Crunchbase data set schema, created by TigerGraph Graph Studio.



Vertices: 694252

person: 227830

(PRIMARY_ID id STRING, fullname STRING, normalized_name STRING, firstname STRING, lastname STRING, birthplace STRING, affiliation_name STRING, permalink STRING, status STRING, domain STRING, homepage_url STRING, twitter_username STRING, logo_url STRING, overview STRING, tag_list STRING, first_milestone_at DATETIME, last_milestone_at DATETIME, created_by STRING, created_at DATETIME, updated_at DATETIME)

university: 21068

(PRIMARY_ID id STRING, name STRING)

company: 197842

(PRIMARY_ID id STRING, name STRING, normalized_name STRING, permalink STRING, category_code STRING, status STRING, founded_at DATETIME, closed_at DATETIME, domain STRING, homepage_url STRING, twitter_username STRING, logo_url STRING, short_description STRING, description STRING, overview STRING, tag_list STRING, country STRING, state STRING, city STRING, region STRING, first_investment_at DATETIME, last_investment_at DATETIME, first_funding_at DATETIME, last_funding_at DATETIME, funding_rounds UINT, funding_total_usd DOUBLE, first_milestone_at DATETIME, last_milestone_at DATETIME, relationships UINT, created_by STRING, created_at DATETIME, update_at DATETIME)

product: 27781

(PRIMARY_ID id STRING, name STRING, normalized_name STRING, permalink STRING, status STRING default "operating", founded_at DATETIME, closed_at DATETIME, domain STRING, homepage_url STRING, twitter_username STRING, logo_url STRING, overview STRING, tag_list STRING, created_at DATETIME, updates_at DATETIME)

financialORG: 11806

(PRIMARY_ID id STRING, name STRING, normalized_name STRING, permalink STRING, category_code STRING, status STRING default "operating", founded_at DATETIME, closed_at DATETIME, domain STRING, homepage_url STRING, twitter_username STRING, logo_url STRING, short_description STRING, description STRING, overview STRING, tag_list STRING, country_code STRING, state_code STRING, city STRING, region STRING, first_investment_at DATETIME, last_investment_at DATETIME, first_funding_at DATETIME, last_funding_at DATETIME, funding_rounds INT, funding_total_usd DOUBLE, first_milestone_at DATETIME, last_milestone_at DATETIME, relationships INT, created_by STRING, created_at DATETIME, updated_at DATETIME)

funds: 1564

(PRIMARY_ID id UINT, name STRING, funded_at DATETIME, raised_amount DOUBLE, raised_currency_code STRING, source_url STRING, source_description STRING, created_at DATETIME, updated_at DATETIME)

milestone: 39456

(PRIMARY_ID id UINT, milestone_at DATETIME, milestone_code STRING, description STRING, source_url STRING, source_description STRING, created_at DATETIME, updated_at DATETIME)

office: 112718

(PRIMARY_ID id UINT, description STRING, region STRING, address1 STRING, address2 STRING, city STRING, zip_code STRING, state_code STRING, country_code STRING, latitude DOUBLE, longitude DOUBLE)

IPO: 1259

(PRIMARY_ID id UINT, valuation_amount DOUBLE, valuation_currency_code STRING, raised_amount DOUBLE, raised_currency_code STRING, public_at DATETIME, stock_symbol STRING, source_url STRING, source_description STRING, created_at DATETIME, updated_at DATETIME)

funding_rounds: 52928

(PRIMARY_ID id UINT, funded_at DATETIME, funding_round_type STRING, funding_round_code STRING, raised_amount_usd DOUBLE, raised_amount DOUBLE, raised_currency_code STRING, pre_money_valuation_usd DOUBLE, pre_money_valuation DOUBLE, pre_money_currency_code STRING, post_money_valuation_usd DOUBLE, post_money_valuation DOUBLE, post_money_currency_code STRING, is_first_round INT, is_last_round INT, source_url STRING, source_description STRING, created_by STRING, created_at DATETIME, updated_at DATETIME)

Edges: 1776084

hasDegree: 205272

(from person, to university, degree_type STRING, subject STRING, graduated_at DATETIME, created_at DATETIME, updated_at DATETIME)

work_for_company: 720500

(from person, to company, start_at DATETIME, end_at DATETIME, is_past INT, sequence INT default "0", title STRING, created_at DATETIME, updated_at DATETIME)

work_for_fOrg: 69928

(from person, to fiancialORG, start_at DATETIME, end_at DATETIME, is_past INT, sequence INT default "0", title STRING, created_at DATETIME, updated_at DATETIME)

investment_from_person: 26510

(from person, to funding_rounds)

investment_from_company: 12402

(from company, to funding_rounds)

investment_from_financialORG: 122688

(from fiancialORG, to funding_rounds)

company_funding_rounds: 105856

(from company, to funding_rounds)

acquire (with reverse_edge="acquired_by"): 9536 (9536)

(from company, to company, term_code STRING, price_amount DOUBLE, price_currency_code STRING, acquired_at DATETIME, source_url STRING, source_description STRING, created_at DATETIME, updated_at DATETIME)

company_ipo: 2508

(from company, to IPO)

company_product: 55428

(from company, to product)

company_office: 204270

(from company, to office)

financial_office: 21166

(from fiancialORG, to office)

financial_funds: 3128

(from fiancialORG, to funds)

person_milestone: 11944

(from person, to milestone)

company_milestone: 57414

(from company, to milestone)

product_milestone: 3606

(from product, to milestone)

financial_milestone: 5948

(from fiancialORG, to milestone)

invest (with reverse_edge="invested_by_company): 5768 (5768)

(from company, to company)

invested_by_person: 24944

(from person, to company)

invested_by_financialORG: 91964

(from fiancialORG, to company)

Business Queries

Below are some questions you can write using GSQL following the tuorial

http://doc.tigergraph.com/GSQL-Tutorial-and-Demo-Examples.html

and developer guide

http://doc.tigergraph.com/dev/index.html

Note: in essense, GSQL is a vertex flow language. The computed vertex set is always on the left of "=" in a query block. For a query block (SELECT-FROM), we always drive from the source vertex of an edge. And the source must be some vertex set you have computed in the previous query blocks or a seed block (That is, the source can only have those vertex set names appearing on the left of a "=" in the previous query blocks). The target set is passive following the source set based on the schema.

Below are one positive and one negative example to clarify this.

```
//Correct example
CREATE QUERY ABC () FOR GRAPH Crunchbase {
  A = {Company.*} ;  //<---- this is a seed block
  B = SELECT C
      FROM A-(E1:e)->:C; //<----starts from A set, which has been
defined above by a seed block.
  D = SELECT p
      FROM B-(E2:e)->Person:p; //<----starts from B set, which has been
computed above.
}
================================================================================
//Negative example
CREATE QUERY ABC () FOR GRAPH Crunchbase {
  A = {Company.*} ;  //<---- this is a seed block
  B = SELECT C
      FROM A-(E1:e)->:C; //<----starts from A set, which has been
defined above by a seed block.
  D = SELECT p
      FROM Person:p-(E2:e)->B; //<----WRONG!!! we cannot drive from
Person, since it's not a computed vertex set.
}
```

DDL

Please write the DDL to create a graph named "crunchbase_pre_2013".

Loading Job

Please write a loading job using GSQL to load a csv file to vertex types, **University, Person,** and edge type **hasDegree.**

http://doc.tigergraph.com/GSQL-Language-Reference-Part-1---Defining-Graphs-and-Loading-Data.html#GSQLLanguageReferencePart1-DefiningGraphsandLoadingDatav1.0-CreatingaLoadingJob

Assuming the csv file has the following columns.

file1.csv: (person_first_name, person_last_name, attended_university_name, degree, subject, graduation_date).

Note

- Please use the default value for vertex/edge type loading if the csv file does not provide those info.
- Please create an online loading job which contains the mappings only.
- Please write the GSQL command to invoke the online loading job with the file name "file1.csv"

Query 1

Problem: Find the top 1 university which has the largest number of alumni who invested or worked for a company that went to IPO in year YYYY?

Input :

year: yyyy (DATETIME)

Output:

university name (STRING)

Query 2

Problem: Find the those companies (say the company name is ABC) that was acquired by a public company (IPOed, say the company name is XYZ), and the company ABC has investors who are employees of the public company XYZ. Note: an investor can invest a company directly via the "invested_by_person" edge. Or, indirectly via the "investment_from_person" edge, and the edge connect to a fund which invest in the targeted company.

Input: N/A

Output:

company names.

Query 3

Problem: This query is to help people find the top k promising startups which have the most number of "succesful" employees within a round cutoff and sector filtering.
E.g. If top k is 3, round cutoff is "c", and indurstry sector is "software", we find the top 3 startups which are before "d" round stage, and have the biggest count of succesful employees work for them. Here, "succesful" is defined as those people who have worked for any startup during the time it went to IPO.

Input
roundStage, k, sector.

E.g.
- roundStage: angel, k: 5, sector: mobile
- roundStage: c, k: 3, sector: games_video

Output:

company names.