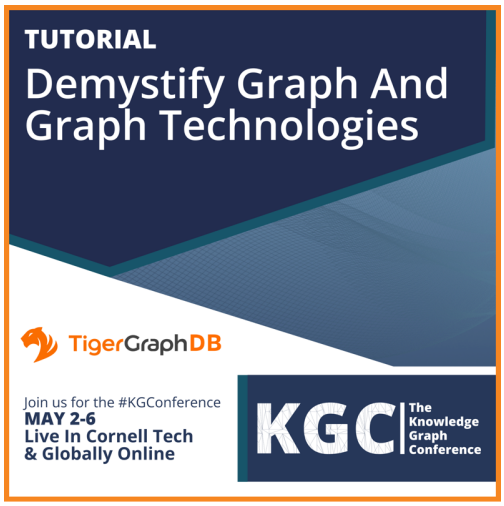


TigerGraph

A guide to Demystify Graph and Graph Technologies

Introduction

Data Technology Innovators may be struggling with graphs and practical adoption and reluctant to learn new technology due to; what a mentor coined as FUD (Fear, Uncertainty, and Doubt.). Fearing the unknown of learning new technologies. “How different is it to adopt Graph? Do I have to unlearn all my Relational or No-SQL database modeling and programming skills?” The feeling of Uncertainty, wondering how hard is it to learn or implement? How much time it will it take? Will it work? Perhaps a sense of Doubt. “What is the benefit?“, “Will it disappear?“, “Is it the future of data tech?” or “Would others adopt it?”

This session will demystify Graph and Graph Technologies by removing the FUD and, at the same time, showing people at any skill level how easy it is and how practical Graphs can be. We will demonstrate how to use Graph to solve complex problems that relational databases struggle with to meet today’s fast business processes.

To combat those challenges we face every day, we rely on a network of friends who themself are Solution Providers. Perhaps with a phone call, an email at 2 AM, or searching the internet for articles, blogs, or any hint of a hope to find a solution to impediment the best performant query. Their responses, blogs, and articles help us, while we support them by returning the favor; it’s a nurturing relationship. They help us deliver complex solutions using emergent technologies like Graph, ML, and AI on time, reducing defects, and meeting those unrealistic response times.

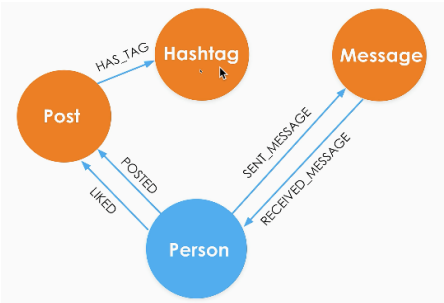
My favorite method of keeping up with emergent technologies like Graph is those workshops that show you where to get started and how simple it is to learn and implement new technologies by providing hands-on (aka Fingers to keyboard) training. We will “Help You, Help Us” by showing you how easy it is to get started in Graph Technologies in this session. We hope you can help us usher in this emergent technology we call Graph.

We all have a “What’s in it for me.” need to know. How is this going to help me advance in my career? How will this help me open new and exciting opportunities for me? Let me help you with the answers. Check out these links to learn why you need to know Graph Database Technologies.

- https://www.ziprecruiter.com/Salaries/Data-Scientist-Salary
- https://www.techrepublic.com/article/graphs-quantum-computing-and-their-future-roles-in-analytics/
- https://www.zdnet.com/article/why-graph-db-ai-may-be-the-future-of-data-management/

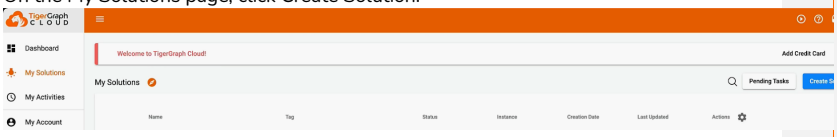
Session Goals:
After this session, you will:
1: Understand what Label Property Graph Database is.
2: Know how to design a Graph Schema.
3: Know-how Data is Mapped to a Graph Schema.
4: Learn how to load Data into a Graph Database.
5: Know how to explore a Graph Database.
6: Know how to write queries.

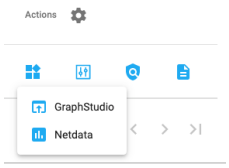
This guide is intended to provide a walkthrough for creating a new graph solution and exploring the various features and functionality in graph. In this session we will create a graph representing social media application with four nodes and five edges that connect them.



Introduction to Graph Using TG Cloud

For this session we will be using TG Cloud and GraphStudio to create a graph and explore the graph. It is completely free to use without the marketing spam.

LOG INTO TIGERGRAPH CLOUD & CREATE A SOLUTION	
STEP 1.	Navigate to https://tgcloud.io/ and select Login/Register.
STEP 2.	Select Sign Up (or login if you already have an account). You will need to confirm your email before you can create a solution.
STEP 3.	After logging in, you will be directed to the dashboard page, on the right side of the page, click My Solutions.
STEP 4.	On the My Solutions page, click Create Solution. 
STEP 5.	In the Instance Settings section set the following: ● TigerGraph Version: 3.5.0 or later ● Starter Kit: entity-resolution-mdm v3.5.0 or later Click Next. ● Select Platform: AWS ● Select an Instance Type: TG Free (this will disable the partition and replication factors) ● Select Region: Click your region ● Disk Size: 50 GB Click Next. ● Name your Solution: entity-resolution ● Tag your Solution: entity-resolution ● Set the Initial Password: tigergraph (lower case) ● Subdomain: add your initials er-[your initials] Click Next.
STEP 6.	Review your solution settings. Once confirmed click Submit. While your solution initializes it will appear in the Pending Tasks section of the My Solutions page. It will take a few minutes for your solution to create.

STEP 7.	In the row associated with the TigerGraph 101 solution, there is an Actions icon, click the Applications icon and select GraphStudio from the dropdown list. 

CREATE A GRAPH	
STEP 1.	Within GraphStudio, click Global View from the left side menu bar, and Create Graph and name it MySocial and click Create. Remember, no spaces in the case name.
DESIGN SCHEMA	
STEP 1.	Click Design Schema from the left side menu bar.
STEP 2.	Click the add vertex type button in the toolbar to create a new vertex type. In the add vertex type panel enter the following: • Vertex name: Person • Primary id: id • Primary id Type: String • Place a check in the box for As Attribute To add an Attribute, click the plus icon and add the following attributes: **1.** Attribute Name: email_address ○ Attribute Type: STRING **2.** Attribute Name: username ○ Attribute Type: STRING **3.** Attribute Name: full_name ○ Attribute Type: STRING **4.** Attribute Name: join_date Attribute Type: DATETIME
STEP 3.	Click the Add button to add the vertex type.

STEP 4.	Click the add vertex type button in the toolbar to create a new vertex type. In the add vertex type panel enter the following: • Vertex name: Post • Primary id: id • Primary id Type: String • Place a check in the box for As Attribute Click the + to add Attributes: **1.** Attribute Name: content ○ Attribute Type: STRING **2.** Attribute Name: posted_date ○ Attribute Type: DATETIME **3.** Attribute Name: deleted ○ Attribute Type: BOOL
STEP 5.	Click the Add button to add the vertex type.
STEP 6.	Click the add vertex type button in the toolbar to create a new vertex type. In the add vertex type panel enter the following: • Vertex name: Messages • Primary id: id • Primary id Type: String • Place a check in the box for As Attribute To add an Attribute, click the green add button and add the following attributes: **1.** Attribute Name: subject ○ Attribute Type: STRING **2.** Attribute Name: body ○ Attribute Type: STRING
STEP 7.	Click the Add button to add the vertex type.

STEP 8.	Click the add vertex type button in the toolbar to create a new vertex type. In the add vertex type panel enter the following: • Vertex name: Hashtag • Primary id: tag • Primary id Type: String • Place a check in the box for As Attribute
STEP 9.	Click the Add button to add the vertex type.

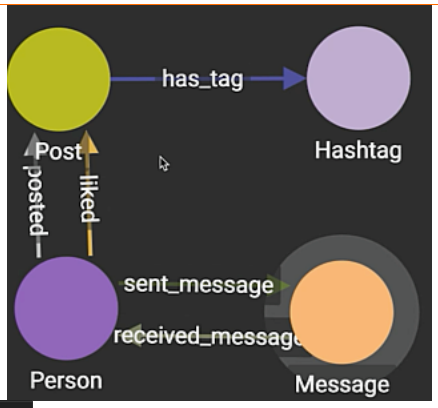
CREATE EDGES	
With our four nodes created, it is time to create the edges.	
STEP 1.	Click the add edge type button to add an edge type.
STEP 2.	Click the person vertex and post vertex (you will see a line created between the two vertexes).
STEP 3.	In the Add edge Type panel enter the following: • Edge type name: posted • Place a check in the box for Directed (this will also enable Reverse edge) The reverse edge option will allow you to see who the person was that posted a specific post. • Source vertex type: Person • Target vertex type: Post In the Attributes section click add and create the following: • Attribute Name: posted_at ○ Attribute Type: DATETIME
STEP 4.	Click the Add button to create the edge.
STEP 5.	Click the add edge type button to add an edge type.
STEP 6.	Click on the Post vertex and Hashtag vertex.
STEP 7.	In the Add edge Type panel enter the following: • Edge type name: has_tag • Place a check in the box for Directed (this will also enable Reverse edge) • Source vertex type: Post • Target vertex type: Hashtag There are no Attributes for this vertex because we are linking a post to a hashtag, there is no additional information we are storing so therefore there will be no attributes to create.
STEP 8.	Click the Add button to create the edge.
STEP 9.	Click the add edge type button to add an edge type.
STEP 10.	Click the Person vertex and Message vertex.
STEP 11.	In the Add edge Type panel enter the following:

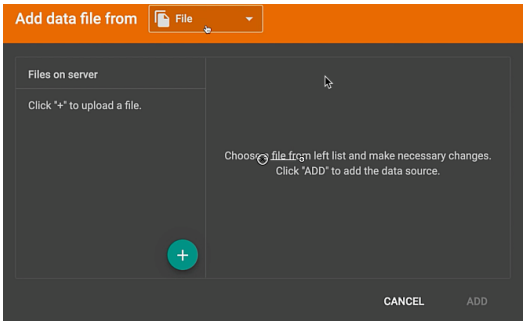
	• Edge type name: sent_message
	• Place a check in the box for Directed (this will also enable Reverse edge) This will link a person to the message that they sent.
	• Source vertex type: Person
	• Target vertex type: Message
	In the Attributes section click add + and create the following:
	• Attribute Name: to_user
	  ○ Attribute Type: STRING
	• Attribute Name: sent_time
	  ○ Attribute Type: DATETIME

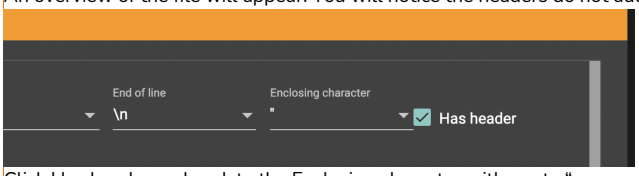
STEP 12.	Click the Add button ✓ to create the edge.
STEP 13.	Click the add edge type button → to add an edge type.
STEP 14.	Click on the Message vertex and Person vertex.

STEP 15.	In the Add edge Type panel enter the following:
	• Edge type name: received_message
	• Place a check in the box for Directed (this will also enable Reverse edge)
	• Source vertex type: Message
	• Target vertex type: Person
	In the Attributes section click add + and create the following:
	• Attribute Name: from_user
	  ○ Attribute Type: STRING
	• Attribute Name: read_time
	  ○ Attribute Type: DATETIME
	We will be able to see which user the message came from and tell what time the recipient read it at.

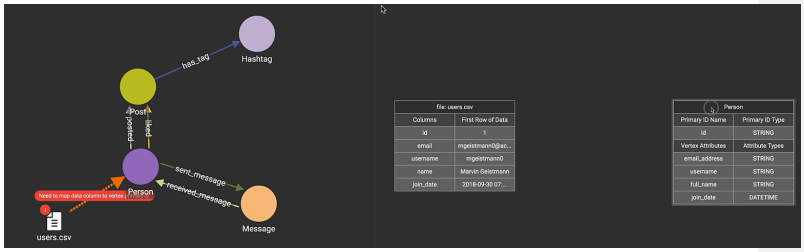
STEP 16.	Click the Add button ✓ to create the edge.
STEP 17.	Click the add edge type button → to add an edge type.
STEP 18.	Click on the Person vertex and Post vertex.
STEP 19.	In the Add edge Type panel enter the following: • Edge type name: liked • Place a check in the box for Directed (this will also enable Reverse edge) • Source vertex type: Person • Target vertex type: Post In the Attributes section click add + and create the following: • Attribute Name: like_time ○ Attribute Type: DATETIME • Attribute Name: read_time Attribute Type: DATETIME

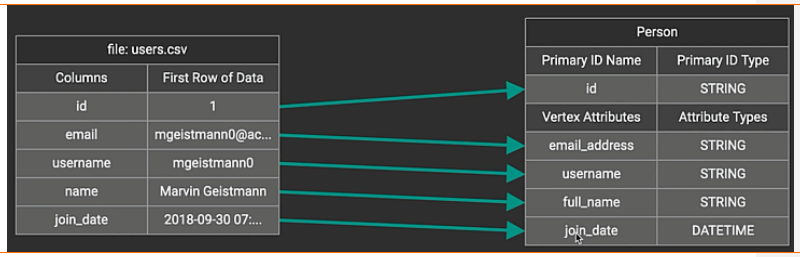
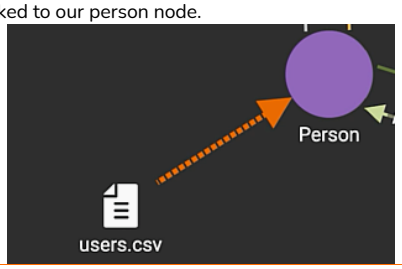
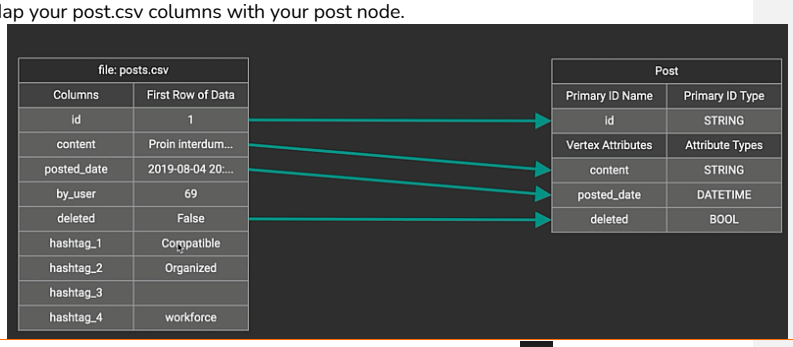
	 Click the Add button to create the edge.
STEP 20.	With our schema completed we can now load this into our graph by clicking the Publish Schema button in the toolbar.

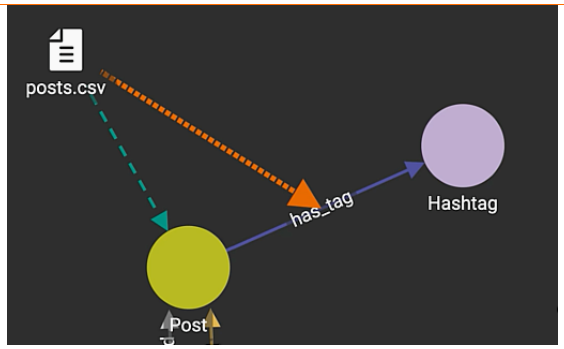
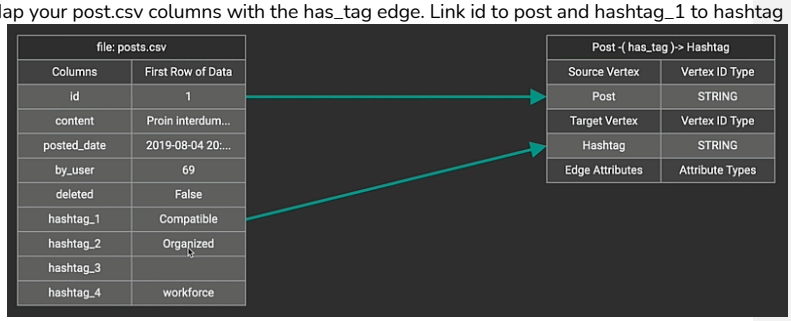
MAP DATA TO GRAPH	
After you have created a graph schema, the next step is to map your data to the schema.	
STEP 1.	Click the Map Data To Graph option from the left side menu bar.
STEP 2.	In the toolbar click Add data file.  Browse to the location you saved your student data and select all the files: likes.csv, messages.csv, posts.csv and users.csv. Click Open
STEP 3.	When successfully added you will see the files appear in the dialog box. Click on the users.csv file.

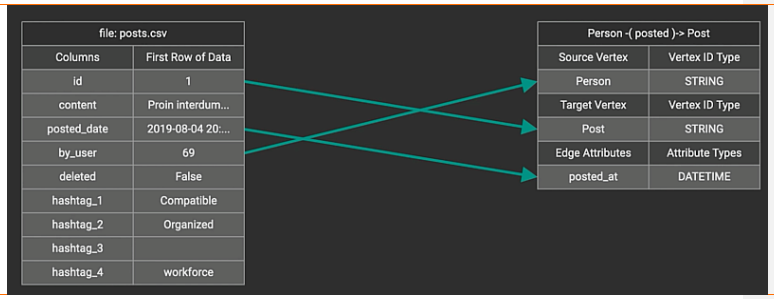
STEP 4.	An overview of the file will appear. You will notice the headers do not automatically register.  Click Has header and update the Enclosing character with quote “
STEP 5.	Click Add.

Your working panel will update with the user.csv file as a node and can be connected to the data. Let’s walk through this process.

STEP 6.	In the toolbar click map data file to vertex or edge  .
STEP 7.	Click your users.csv data file and person.  Two tables will appear in your working panel. The columns from the user.csv file will display in the users table and the person table will appear with the attributes from the node.
STEP 8.	Map the rows. Click the id row from the users table and click on the id row from the person table. An arrow will indicate that they are linked. Continue this process until all rows are mapped to the appropriate row in the corresponding table.

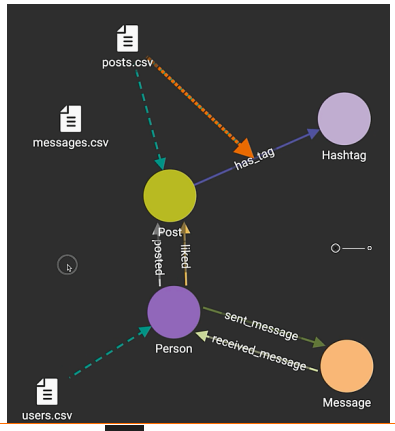
	
STEP 9.	Our user data is now linked to our person node. 
STEP 10.	Let's bring in more data. Click Add Data File.
STEP 11.	Click the posts.csv file.
STEP 12.	Click Has header and update the Enclosing character with quote " and click Add.
STEP 13.	In the toolbar click map data file to vertex or edge.
STEP 14.	Click your posts.csv data file and post.
STEP 15.	Map your post.csv columns with your post node. 
STEP 16.	Now, let's map our tags. Click map data file to vertex or edge. Map the post.csv to your has_tag edge.

	 Connecting post to hashtag and mapping to tag will not give the results we are looking for, because it will create a hashtag node for each hashtag but there is nothing that relates this hashtag back to the post that has the hashtags. This is because our post node does not have any fields for storing that hashtag information.
STEP 17.	Map your post.csv columns with the has_tag edge. Link id to post and hashtag_1 to hashtag  This will take each post, load data into each post node, create a vertex for each entry in the .csv file and attempt to create an edge from each vertex that it created to any hashtags it might contain.
We have four hashtag fields, and we only have one field for hashtag in our edge. Because you can only map one column per linkage to the source vertex, we will need to create multiple mappings.	
STEP 18.	Click map data file to vertex or edge and map the post.csv to your has_tag edge.
STEP 19.	Link id to post and hashtag_2 to hashtag.
STEP 20.	Repeat this process until all 4 hashtags are linked.
STEP 21.	Let's map a post to an edge. Click map data file to vertex or edge and map the post.csv to the posted edge.
STEP 22.	Link the following: • id to Post • by_user to Person • posted_date to posted_at



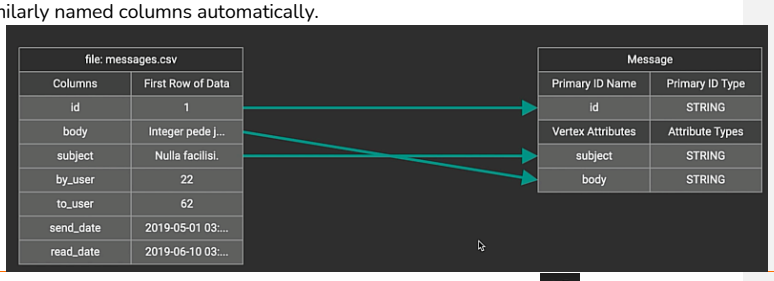
STEP 23. Next, let's link our messages. Click Add data file.

STEP 24. Click the messages.csv file and select Has header. Ensure that " is select for the Enclosing character and click Add.

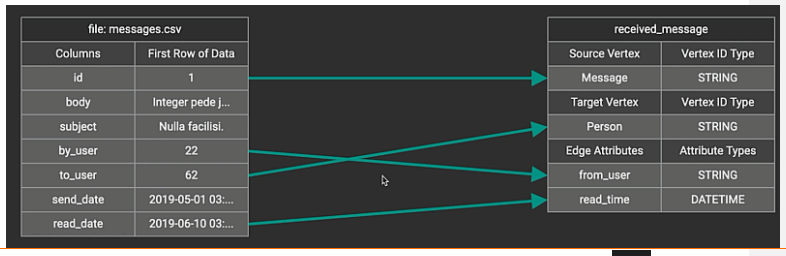
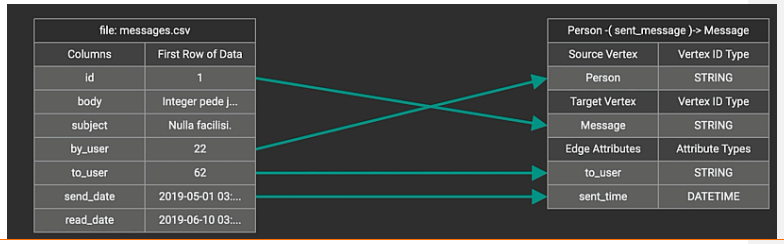


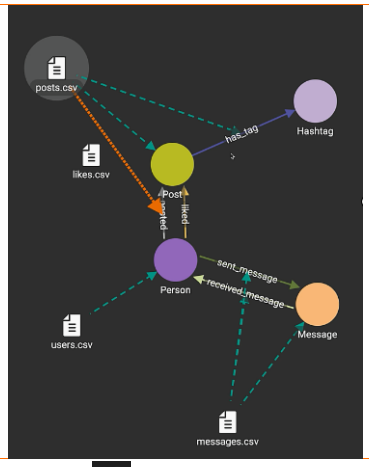
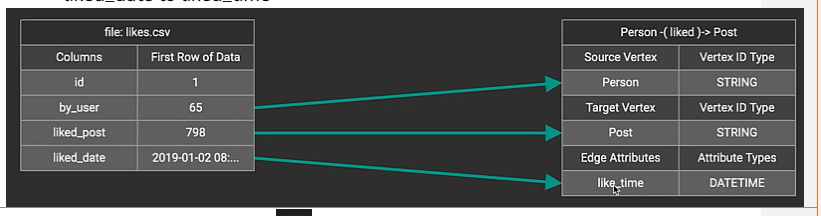
STEP 25. Click map data file to vertex or edge and map the messages.csv to Message node.

STEP 26. Let's try another method for mapping. In the toolbar click auto mapping to map all similarly named columns automatically.

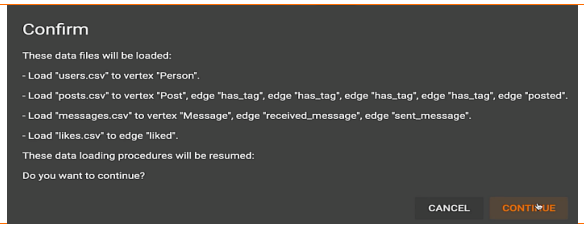


STEP 27. Now, let's create our edges. Click map data file to vertex or edge and map the messages.csv to the received_messages edge.

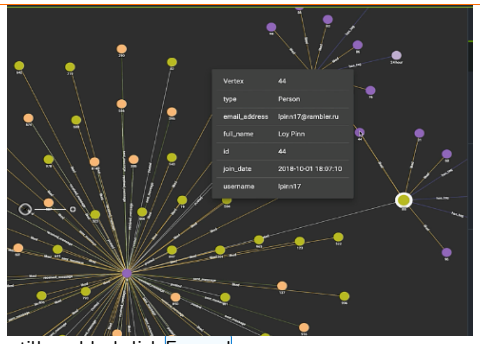
STEP 28.	Link the following: ● id to Messages ● to_user to Person ● by_user to from_user ● read_date to read_time 
STEP 29.	Create mapping for sent messages. Click map data file to vertex or edge  and map the messages.csv to the sent_messages edge.
STEP 30.	Link the following: ● id to Messages ● by_user to Person ● to_user to to_user ● send_date to sent_time 
STEP 31.	Lastly, add the likes.csv file. Click Add data file.
STEP 32.	Click the likes.csv file and select Has header and click Add.

	
STEP 33.	Click map data file to vertex or edge and map the likes.csv to liked.
STEP 34.	Link the following: • by_user to Person • liked_post to Post • liked_date to liked_time 
STEP 35.	Click the publish schema button to publish the data loading procedure to the TigerGraph system. It takes about 2 to 3 seconds for publishing each data file mapping.

LOAD DATA	
After mapping data files to the graph schema, you can start loading data.	
STEP 1.	Click Load Data on the left side menu bar.
STEP 2.	Click on the start loading button in the toolbar.
STEP 3.	Verify you are loading all the data sources and click Continue.



EXPLORE GRAPH	
After data has been loaded, the Explore Graph page allows you to search for vertices in a graph, to discover nearby vertices which satisfy conditions of your choice, and to find the paths between vertices.	
STEP 1.	Click Explore Graph from the left side menu bar.
STEP 2.	In the Search vertices section, uncheck All from the Pick vertices by vertex types area.
STEP 3.	Place a check in the box for Person and set the Search vertices by attribute should be the default, 5. Click Pick Vertices.
STEP 4.	Hover over the people and you can see the attributes for each vertex.
STEP 5.	Double click a person. This will expand any edges associated with that person.
STEP 6.	In the bottom right corner click the layout button force to change the view. Generally, force is the best to use.
STEP 7.	Hover over message and you can see who the message was sent to.
STEP 8.	Locate a Post and double click it. You can see all the users that liked it as well as the hashtags associated with that post.
STEP 9.	To reset your view, navigate to the change layout button and click Force.
STEP 10.	Click the Expand from vertices tab and uncheck All from the expand through edge types section.
STEP 11.	Place a check in the box for has_tag from the expand through edge types section.
STEP 12.	Select All from the expand towards vertex types section and click Expand.
STEP 13.	You will see that we expanded on the hashtags but not the people that liked them. Back in the expand through edge types section uncheck has_tag and check liked and click Expand. Now we can see the group of users who have liked that post.
STEP 14.	Reset your view by clicking the change layout button and click Force.
STEP 15.	We can see this user has liked both posts. Click on the person (id 44). Let's see if this user has liked any other posts by user 9.

	 With the liked edge still enabled click Expand.
STEP 16.	We can see the user does like more posts shared by user 9. Reset your view by clicking the change layout button and click Force. We have a path from user 9 through the posts to user 44.
STEP 17.	To find paths between two people and do not want to just click around, let's use our path finding algorithms to see if we can find any paths between user 28 and 9. Click the Find paths  option.
STEP 18.	In the Choose starting vertex section set the following: • Vertex Type: Person • Vertex id: 9
STEP 19.	In the Choose destination vertex section set the following: • Vertex Type: Person • Vertex id: 28
STEP 20.	In the Paths going through vertex types check All and click Find Paths. We have a post that user 9 liked and user 28 liked. User 44 has also liked this post.
STEP 21.	This shows the one shortest path, that was set by default in the Configuration section. Click Show all shortest paths and Find paths.
STEP 22.	We can now see that there are a couple more edges as well as messages.
STEP 23.	Reset your view by clicking the change layout button and click Force.

This completes the hands-on portion of this session.

Commented [1]: Did I get this right?

TigerGraph

Additional Resources

1. Why Graph Databases?
2. Native Parallel Graphs (eBook)
3. TigerGraph Solutions Page (use cases)

BENCHMARKS

4. Benchmarking Graph Analytic Systems (TigerGraph's report)
5. LDBC Benchmark PDF (3rd party report)
    a. Globe Newswire article

DOCUMENTATION

6. TigerGraph Technical Documentation
7. Architecture and System Overview
8. A list of webinars, videos are available on the link here, that includes our Graph GuruWebinars, Tutorial videos.

PROJECTS AND USER REPOSITORIES

9. TigerGraph on Github
10. TigerGraph User Community
11. pyTigerGraph

TIGERGRAPH CLOUD

12. TigerGraph Cloud Portal (tgcloud.io)
13. Getting started with TigerGraph Cloud
14. TigerGraph 101

INTRO TO TIGERGRAPH

15. GraphStudio (videos)
16. GraphStudio UI Guide (documentation)

SYSTEM ARCHITECTURE

17. Troubleshooting Guide (documentation)
18. System Security (documentation)
19. Graph Gurus 23: Best Practices to Model Your Data Using a Graph Database

GSLQ BASICS

20. GSQL 101 (documentation)
21. GSQL Select Statement
22. Graph Gurus 31: GSQL Writing Best Practices - Part 1 Thinking In GSQL
23. TigerGraph's Graph Query Language, GSQL (webinar)

ADVANCED QUERY WRITING

24. GSQL Writing Best Practices - Part 2 Design Optimal Traversal Plan
25. GSQL Writing Best Practices - Part 3 Memory Optimization
26. GSQL Writing Best Practices - Part 4 Parallelization and Preprocessing
27. GSQL Writing Best Practices - Part 5 Data Structure
28. Schema design best practices 1
29. GSQL Graph Algorithm Library (documentation)

DATA SCIENCE CAPABILITIES

30. Using Graph Algorithms for Advanced Analytics Part 1 - Shortest Paths
31. Using Graph Algorithms for Advanced Analytics Part 2 - Centrality
32. Using Graph Algorithms for Advanced Analytics Part 3 - Community Detection
33. Using Graph Algorithms for Advanced Analytics Part 4 – Similarity
34. Using Graph Algorithms for Advanced Analytics Part 5 Classification

TigerGraph is a platform for advanced analytics and machine learning on connected data. Built on the industry's first and only distributed native graph database, TigerGraph's proven technology supports advanced analytics and machine learning applications.
training@tigergraph.com