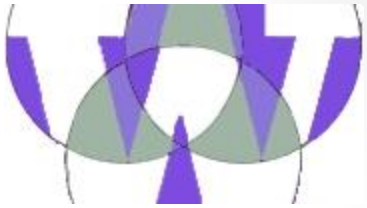
Introduction to Data Science

Lecture 10; December 14th, 2016

Ernst Henle
ErnstHe@UW.edu
Skype: ernst-henle

Agenda



- Announcements
  - Social Interactions on LinkedIn: Continue to discuss statistics, job prospects, etc.
- Review: Hadoop MapReduce Lab
- Break
- Introduction to Graph Data
- Quiz Graph Data (Measures)
- Graph Data Popularity
- Quiz Graph Data (Popularity)
- Graph Data Google Matrix
- Break
- SPARQL Intro
- Quiz SPARQL intro
- SPARQL Examples
- Time Permitting:
  - Nested Tables and Associations
  - Quiz Nested Tables
  - Connectivity of the Web
- Assignment. See assignment slides at the end of the deck. Complete all assignments items from all assignment slides. Complete by this Saturday at 11:57 PM.

MapReduce Review

MapReduce Review

- Changes
  - Package name: _____
  - WordMapper.java
    - ***extends Mapper<LongWritable, Text, ____, _____>***
    - ***context.write(new _____, new _____);***
  - SumReducer.java
    - ***extends Reducer<____, _____, ____, _____>***
    - ***public void reduce(____ key, Iterable<_____> values, Context context)***
    - ***context.write(key, new _____);***
  - WordCount.java
    - ***job.setOutputKeyClass(_____);***
    - ***job.setOutputValueClass(_____);***

MapReduce Review

Map input key-value types are typically given

- Changes
  - Package name: _____
  - WordMapper.java
    - *extends Mapper<LongWritable, Text, ____, _____>*
    - *context.write(new _____, new _____);*
  - SumReducer.java
    - *extends Reducer<____, _____, ____, _____>*
    - *public void reduce(____ key, Iterable<_____> values, Context context)*
    - *context.write(key, new _____);*
  - WordCount.java
    - *job.setOutputKeyClass(_____);*
    - *job.setOutputValueClass(_____);*

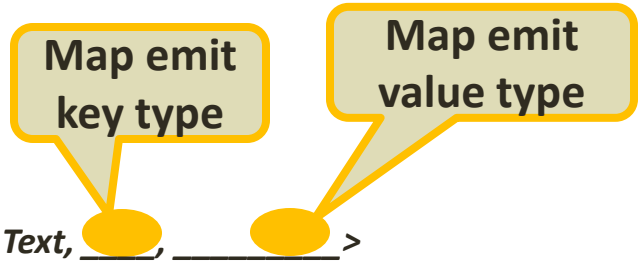
Map input key type

Map input value type

MapReduce Review

Map emit key-value types are chosen

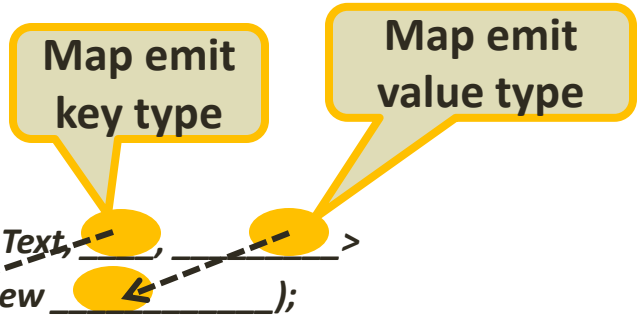
- Changes
  - Package name: _____
  - WordMapper.java
    - *extends Mapper<LongWritable, Text, _____, _____>*
    - *context.write(new _____, new _____);*
  - SumReducer.java
    - *extends Reducer<____, _____, ____, _____>*
    - *public void reduce(____ key, Iterable<_____> values, Context context)*
    - *context.write(key, new _____);*
  - WordCount.java
    - *job.setOutputKeyClass(_____);*
    - *job.setOutputValueClass(_____);*



MapReduce Review

Map emit key-value types need to be consistent

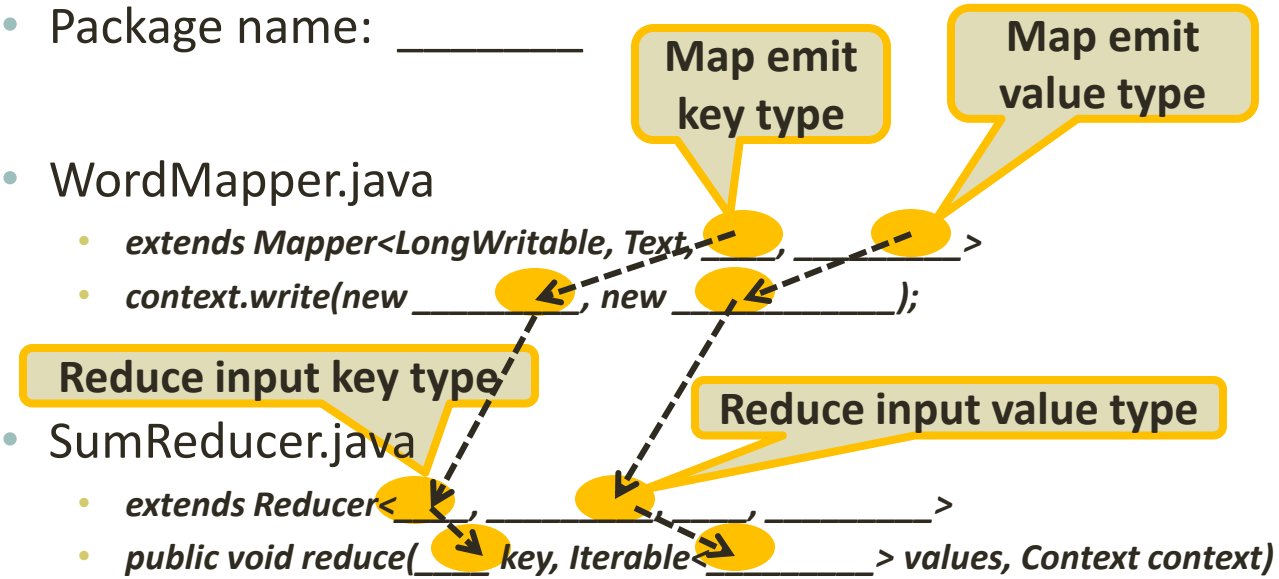
- Changes
  - Package name: _____
  - WordMapper.java
    - *extends Mapper<LongWritable, Text, _____, _____>*
    - *context.write(new _____, new _____);*
  - SumReducer.java
    - *extends Reducer<____, _____, ____, _____>*
    - *public void reduce(____ key, Iterable<_____> values, Context context)*
    - *context.write(key, new _____);*
  - WordCount.java
    - *job.setOutputKeyClass(_____);*
    - *job.setOutputValueClass(_____);*



MapReduce Review

Map emit key-value types are the same as Reduce input key-value types

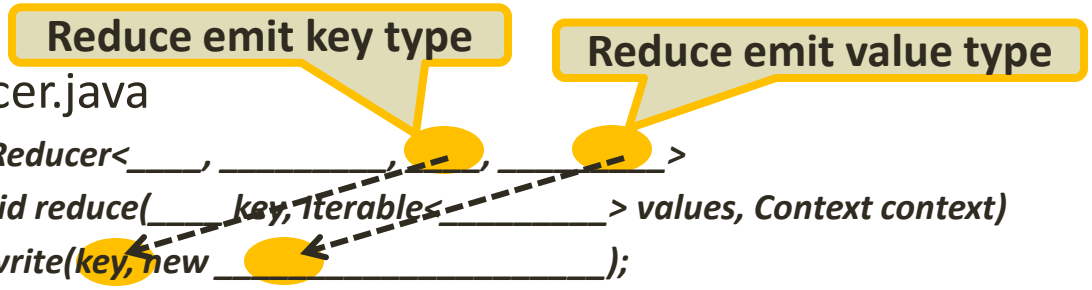
- Changes
  - Package name: _____
  - WordMapper.java
    - *extends Mapper<LongWritable, Text, _____, _____>*
    - *context.write(new _____, new _____);*
  - SumReducer.java
    - *extends Reducer<_____, _____, _____, _____>*
    - *public void reduce(_____ key, Iterable<_____> values, Context context)*
    - *context.write(key, new _____);*
  - WordCount.java
    - *job.setOutputKeyClass(_____);*
    - *job.setOutputValueClass(_____);*



MapReduce Review

Reduce emit key-value types need to be consistent

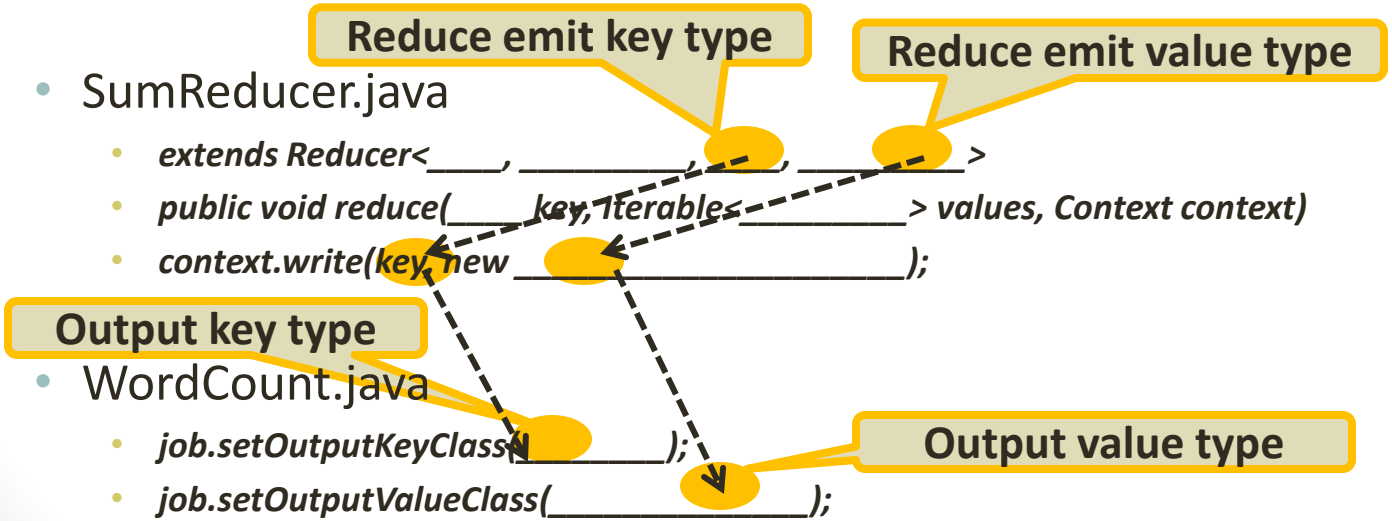
- Changes
  - Package name: _____
  - WordMapper.java
    - *extends Mapper<LongWritable, Text, ____, _____>*
    - *context.write(new _____, new _____);*
  - SumReducer.java
    - *extends Reducer<____, _____, ____, _____>*
    - *public void reduce(____ key, Iterable<_____> values, Context context)*
    - *context.write(key, new _____);*
  - WordCount.java
    - *job.setOutputKeyClass(_____);*
    - *job.setOutputValueClass(_____);*



MapReduce Review

Reduce emit key-value types are the same as Output key-value types

- Changes
  - Package name: _____
  - WordMapper.java
    - *extends Mapper<LongWritable, Text, ____, _____>*
    - *context.write(new _____, new _____);*
  - SumReducer.java
    - *extends Reducer<____, _____, ____, _____>*
    - *public void reduce(____ key, Iterable<_____> values, Context context)*
    - *context.write(key, new _____);*
  - WordCount.java
    - *job.setOutputKeyClass(_____);*
    - *job.setOutputValueClass(_____);*



MapReduce Review

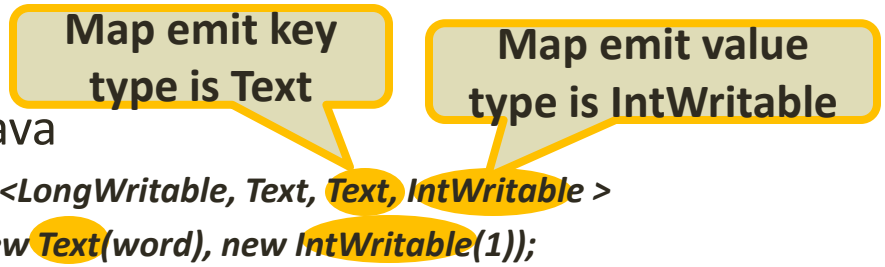
Original WordCount

- Changes
  - Package name: solution
  - WordMapper.java
    - ***extends Mapper<LongWritable, Text, Text, IntWritable >***
    - ***context.write(new Text(word), new IntWritable(1));***
  - SumReducer.java
    - ***extends Reducer<Text, IntWritable, Text, IntWritable>***
    - ***public void reduce(Text key, Iterable<IntWritable> values, Context context)***
    - ***context.write(key, new IntWritable(wordCount));***
  - WordCount.java
    - ***job.setOutputKeyClass(Text.class);***
    - ***job.setOutputValueClass(IntWritable.class);***

MapReduce Review

Original WordCount

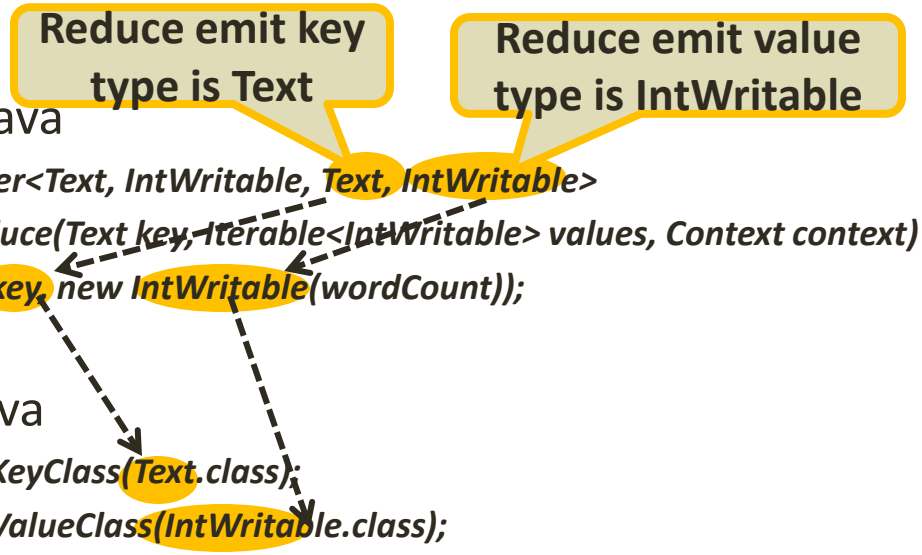
- Changes
  - Package name: solution
  - WordMapper.java
    - *extends Mapper<LongWritable, Text, Text, IntWritable >*
    - *context.write(new Text(word), new IntWritable(1));*
  - SumReducer.java
    - *extends Reducer<Text, IntWritable, Text, IntWritable>*
    - *public void reduce(Text key, Iterable<IntWritable> values, Context context)*
    - *context.write(key, new IntWritable(wordCount));*
  - WordCount.java
    - *job.setOutputKeyClass(Text.class);*
    - *job.setOutputValueClass(IntWritable.class);*



MapReduce Review

Original WordCount

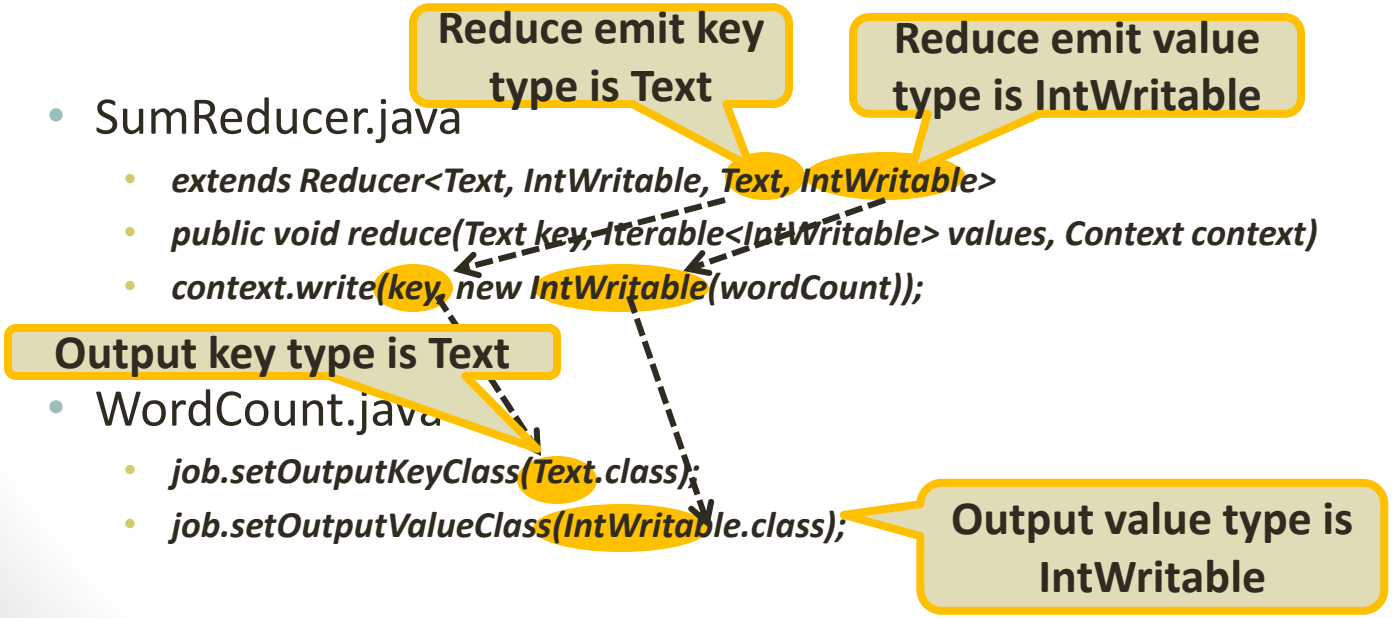
- Changes
  - Package name: solution
  - WordMapper.java
    - *extends Mapper<LongWritable, Text, Text, IntWritable >*
    - *context.write(new Text(word), new IntWritable(1));*
  - SumReducer.java
    - *extends Reducer<Text, IntWritable, Text, IntWritable>*
    - *public void reduce(Text key, Iterable<IntWritable> values, Context context)*
    - *context.write(key, new IntWritable(wordCount));*
  - WordCount.java
    - *job.setOutputKeyClass(Text.class);*
    - *job.setOutputValueClass(IntWritable.class);*



MapReduce Review

Original WordCount

- Changes
  - Package name: solution
  - WordMapper.java
    - *extends Mapper<LongWritable, Text, Text, IntWritable >*
    - *context.write(new Text(word), new IntWritable(1));*
  - SumReducer.java
    - *extends Reducer<Text, IntWritable, Text, IntWritable>*
    - *public void reduce(Text key, Iterable<IntWritable> values, Context context)*
    - *context.write(key, new IntWritable(wordCount));*
  - WordCount.java
    - *job.setOutputKeyClass(Text.class);*
    - *job.setOutputValueClass(IntWritable.class);*



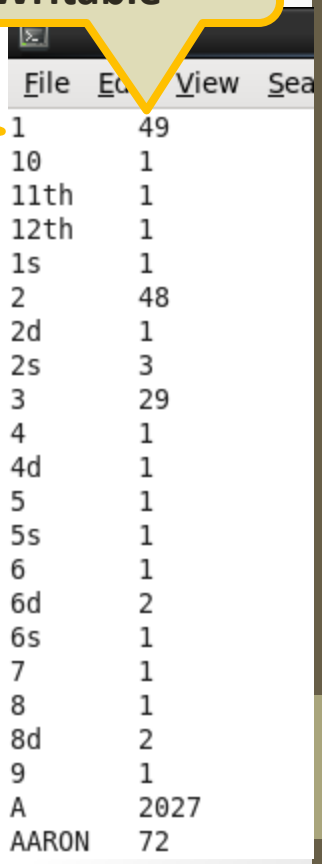
MapReduce Review

Original WordCount

- Changes
  - Package name: solution
  - WordMapper.java
    - *extends Mapper<LongWritable, Text, Text, IntWritable >*
    - *context.write(new Text(word), new IntWritable(1));*
  - SumReducer.java
    - *extends Reducer<Text, IntWritable, Text, IntWritable>*
    - *public void reduce(Text key, Iterable<IntWritable> values, Context context)*
    - *context.write(key, new IntWritable(wordCount));*
  - WordCount.java
    - *job.setOutputKeyClass(Text.class);*
    - *job.setOutputValueClass(IntWritable.class);*

Output value type is IntWritable

Output key type is Text



Output key type is Text

Output value type is IntWritable

MapReduce Review

Assignment: Length Count 1

- Changes
  - Package name: lengthcounts1
  - WordMapper.java
    - *extends Mapper<LongWritable, Text, Text, IntWritable >*
    - *context.write(new Text(String.valueOf(word.length())), new IntWritable(1));*
  - SumReducer.java
    - *extends Reducer<Text, IntWritable, Text, IntWritable>*
    - *public void reduce(Text key, Iterable<IntWritable> values, Context context)*
    - *context.write(key, new IntWritable(wordCount));*
  - WordCount.java
    - *job.setOutputKeyClass(Text.class);*
    - *job.setOutputValueClass(IntWritable.class);*

The only change to WordCount does not require any changes to types

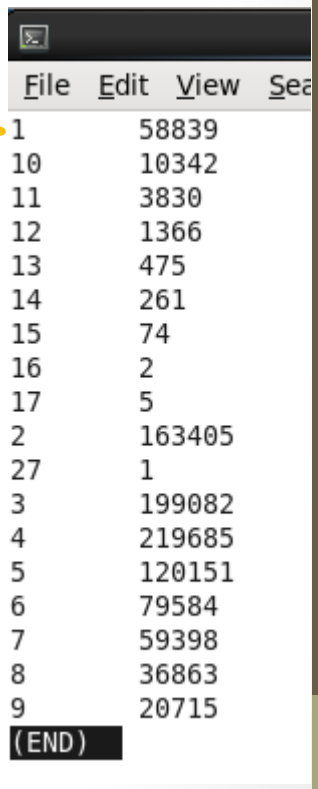
MapReduce Review

Assignment: Length Count 1

- Changes
  - Package name: lengthcounts1
  - WordMapper.java
    - *extends Mapper<LongWritable, Text, Text, IntWritable >*
    - *context.write(new Text(String.valueOf(word.length())), new IntWritable(1));*
  - SumReducer.java
    - *extends Reducer<Text, IntWritable, Text, IntWritable>*
    - *public void reduce(Text key, Iterable<IntWritable> values, Context context)*
    - *context.write(key, new IntWritable(wordCount));*
  - WordCount.java
    - *job.setOutputKeyClass(Text.class);*
    - *job.setOutputValueClass(IntWritable.class);*

Output key type is Text

Output key type is Text



MapReduce Review

Assignment: Length Count 2

- Changes
  - Package name: lengthcounts2
  - WordMapper.java
    - *extends Mapper<LongWritable, Text, , IntWritable >*
    - *context.write(new IntWritable(word.length()), new IntWritable(1));*
  - SumReducer.java
    - *extends Reducer< , IntWritable, , IntWritable>*
    - *public void reduce(key, Iterable<IntWritable> values, Context context)*
    - *context.write(key, new IntWritable(wordCount));*
  - WordCount.java
    - *job.setOutputKeyClass(class);*
    - *job.setOutputValueClass(IntWritable.class);*

The first change to WordCount changes the Map emit type to IntWritable

MapReduce Review

Assignment: Length Count 2

- Changes
  - Package name: lengthcounts2
  - WordMapper.java
    - *extends Mapper<LongWritable, Text, IntWritable, IntWritable >*
    - *context.write(new IntWritable(word.length()), new IntWritable(1));*
  - SumReducer.java
    - *extends Reducer< , IntWritable, , IntWritable>*
    - *public void reduce(key, Iterable<IntWritable> values, Context context)*
    - *context.write(key, new IntWritable(wordCount));*
  - WordCount.java
    - *job.setOutputKeyClass(class);*
    - *job.setOutputValueClass(IntWritable.class);*

Map emit key type needs to be IntWritable

MapReduce Review

Assignment: Length Count 2

- Changes
  - Package name: lengthcounts2
  - WordMapper.java
    - *extends Mapper<LongWritable, Text, IntWritable, IntWritable >*
    - *context.write(new IntWritable(word.length()), new IntWritable(1));*
  - SumReducer.java

    Reduce input key type needs to be IntWritable
    - *extends Reducer< IntWritable, IntWritable, , IntWritable>*
    - *public void reduce(IntWritable key, Iterable<IntWritable> values, Context context)*
    - *context.write(key, new IntWritable(wordCount));*
  - WordCount.java
    - *job.setOutputKeyClass(class);*
    - *job.setOutputValueClass(IntWritable.class);*

MapReduce Review

Assignment: Length Count 2

- Changes
  - Package name: lengthcounts2
  - WordMapper.java
    - *extends Mapper<LongWritable, Text, IntWritable, IntWritable >*
    - *context.write(new IntWritable(word.length()), new IntWritable(1));*
  - SumReducer.java
    - *extends Reducer< IntWritable, IntWritable, IntWritable, IntWritable>*
    - *public void reduce(IntWritable key, Iterable<IntWritable> values, Context context)*
    - *context.write(key, new IntWritable(wordCount));*
  - WordCount.java
    - *job.setOutputKeyClass(class);*
    - *job.setOutputValueClass(IntWritable.class);*

Reduce emit key type needs to be IntWritable

MapReduce Review

Assignment: Length Count 2

- Changes
  - Package name: lengthcounts2
  - WordMapper.java
    - *extends Mapper<LongWritable, Text, IntWritable, IntWritable >*
    - *context.write(new IntWritable(word.length()), new IntWritable(1));*
  - SumReducer.java
    - *extends Reducer< IntWritable, IntWritable, IntWritable, IntWritable>*
    - *public void reduce(IntWritable key, Iterable<IntWritable> values, Context context)*
    - *context.write(key, new IntWritable(wordCount));*
  - WordCount.java
    - *job.setOutputKeyClass(IntWritable.class);*
    - *job.setOutputValueClass(IntWritable.class);*

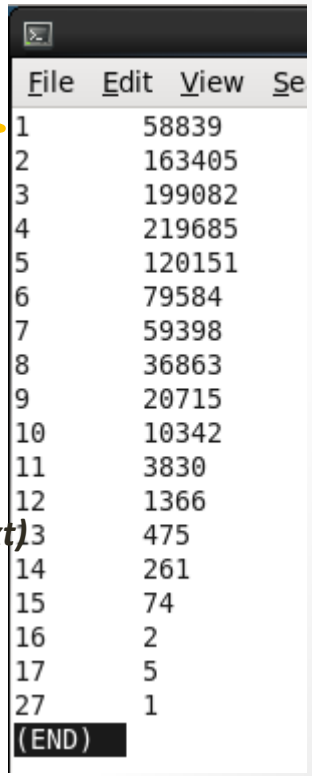
Output key type is IntWritable

MapReduce Review

Assignment: Length Count 2

- Changes
  - Package name: lengthcounts2
  - WordMapper.java
    - *extends Mapper<LongWritable, Text, IntWritable, IntWritable >*
    - *context.write(new IntWritable(word.length()), new IntWritable(1));*
  - SumReducer.java
    - *extends Reducer< IntWritable, IntWritable, IntWritable, IntWritable>*
    - *public void reduce(IntWritable key, Iterable<IntWritable> values, Context context)*
    - *context.write(key, new IntWritable(wordCount));*
  - WordCount.java
    - *job.setOutputKeyClass(IntWritable.class);*
    - *job.setOutputValueClass(IntWritable.class);*

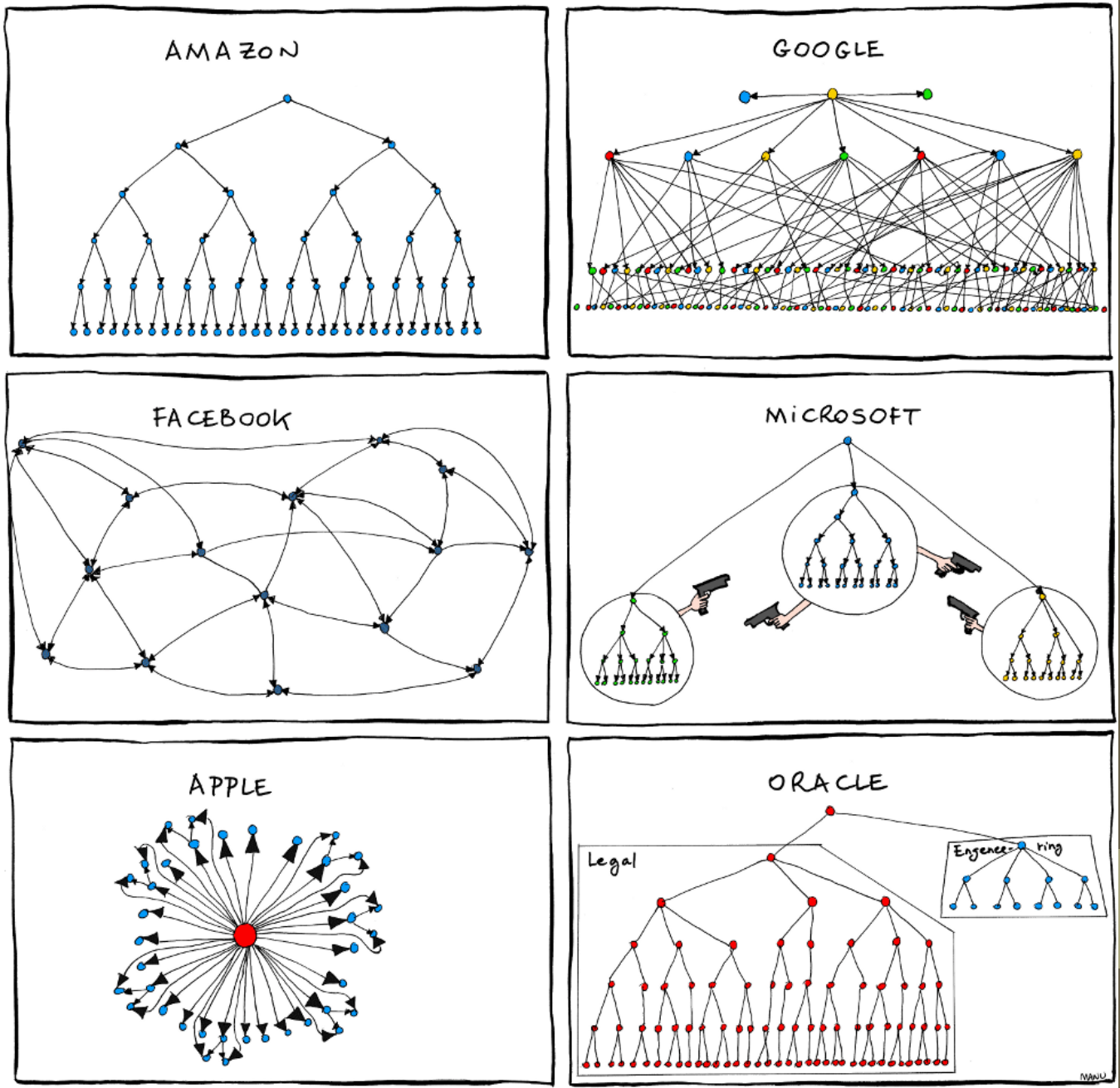
Output key type is IntWritable



Output key type is IntWritable

MapReduce Review

Break

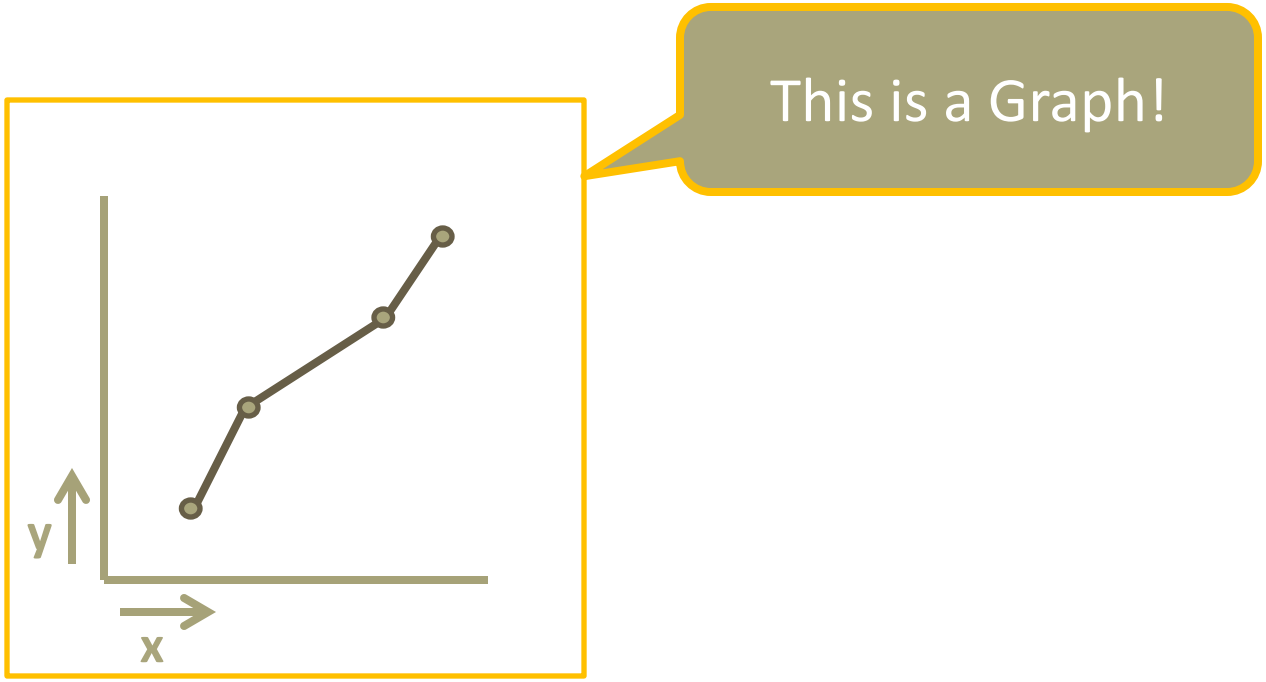

AMAZON
GOOGLE
FACEBOOK
MICROSOFT
APPLE
ORACLE
Legal
Engineering
MANU

Introduction to Graph Data

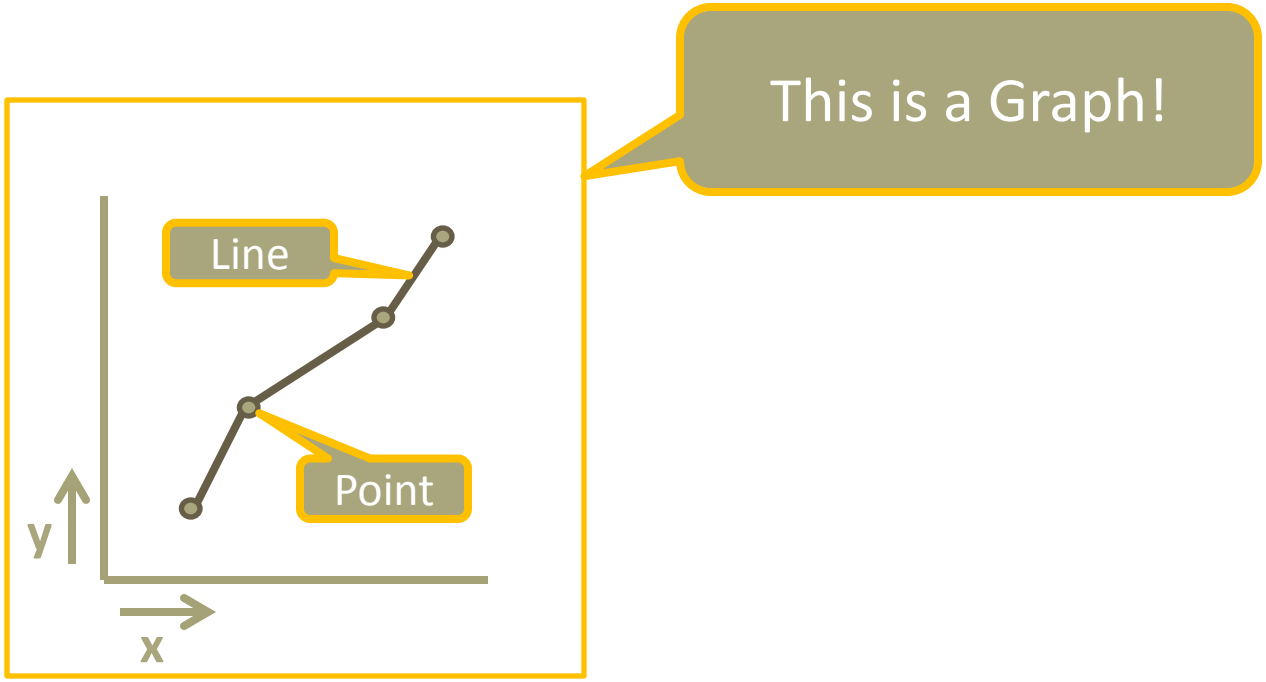
Graph Data: What is it? (0)

What is a Graph?

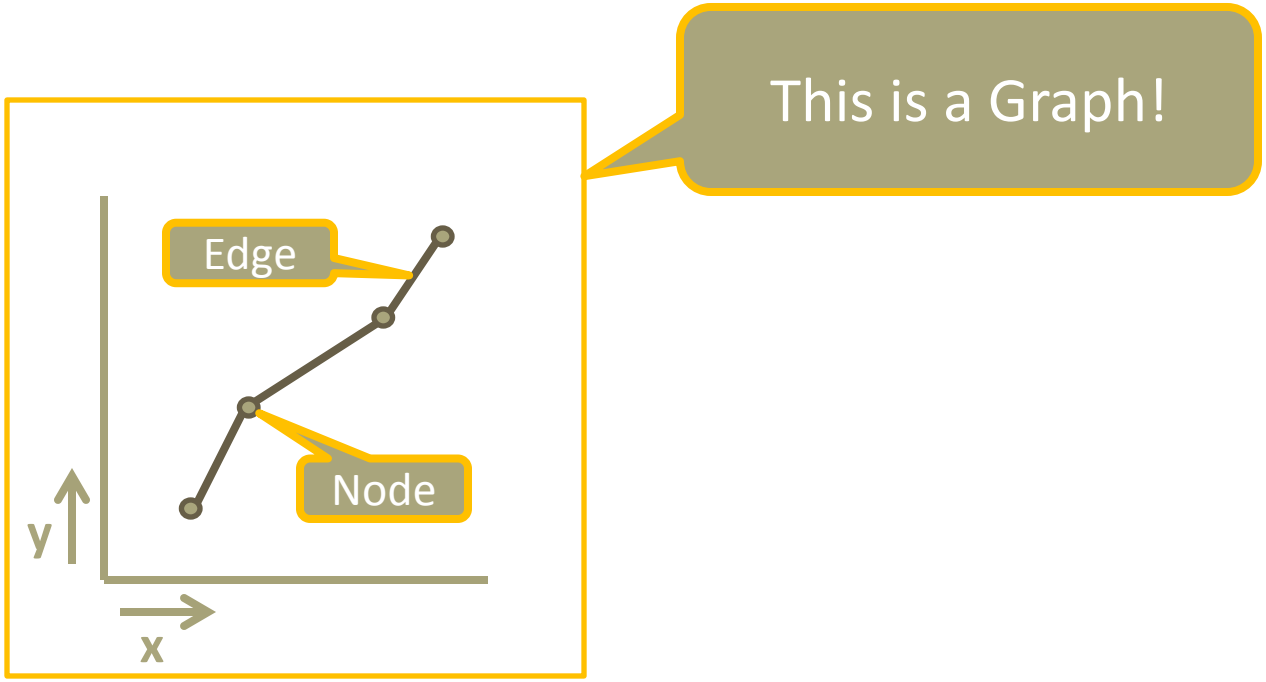
Graph Data: What is it? (1)



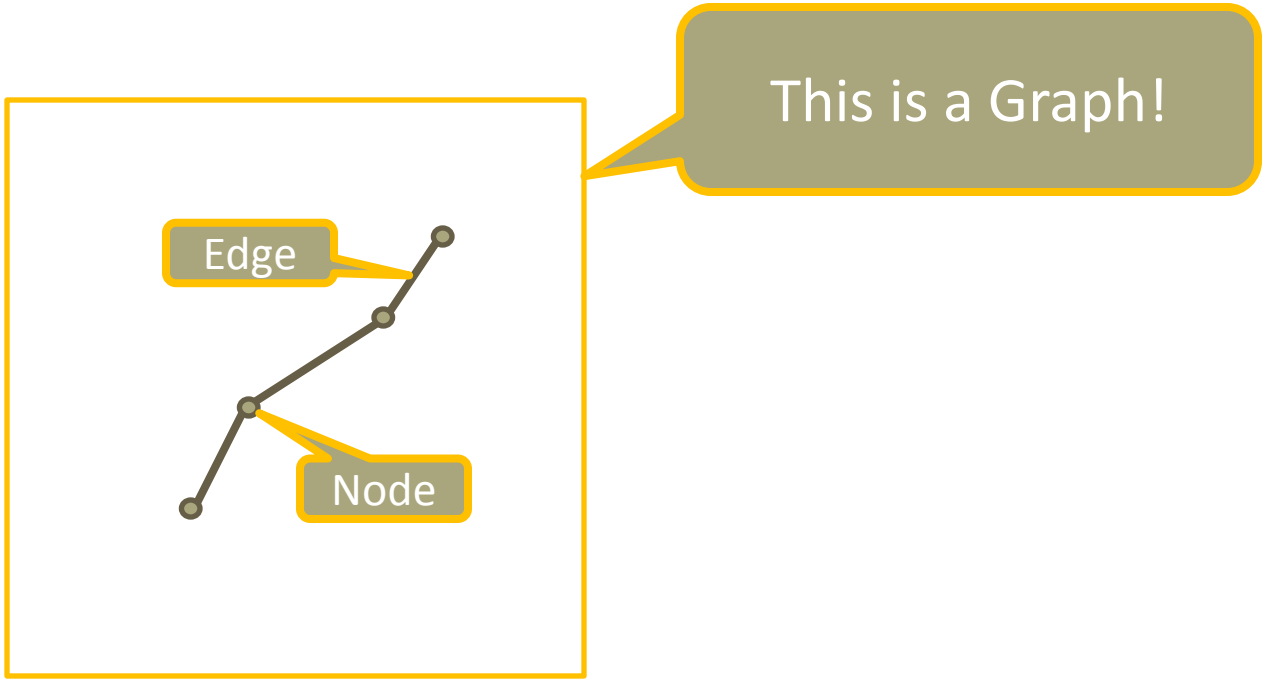
Graph Data: What is it? (2)



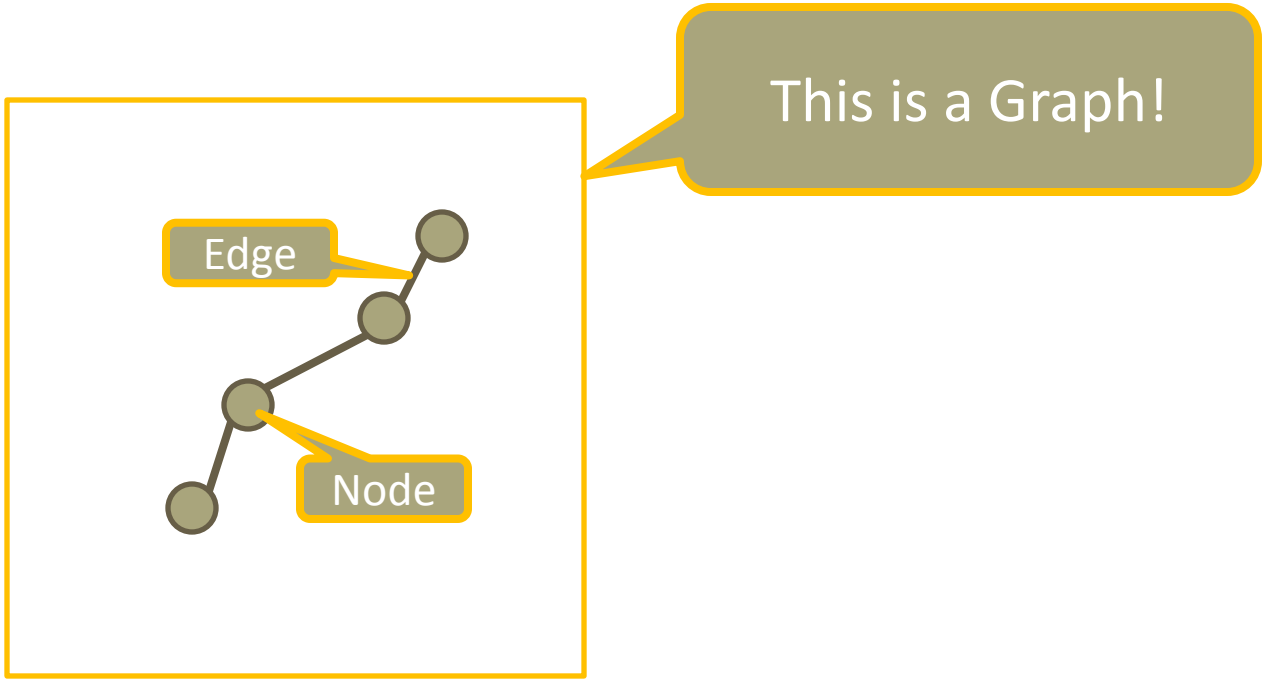
Graph Data: What is it? (3)



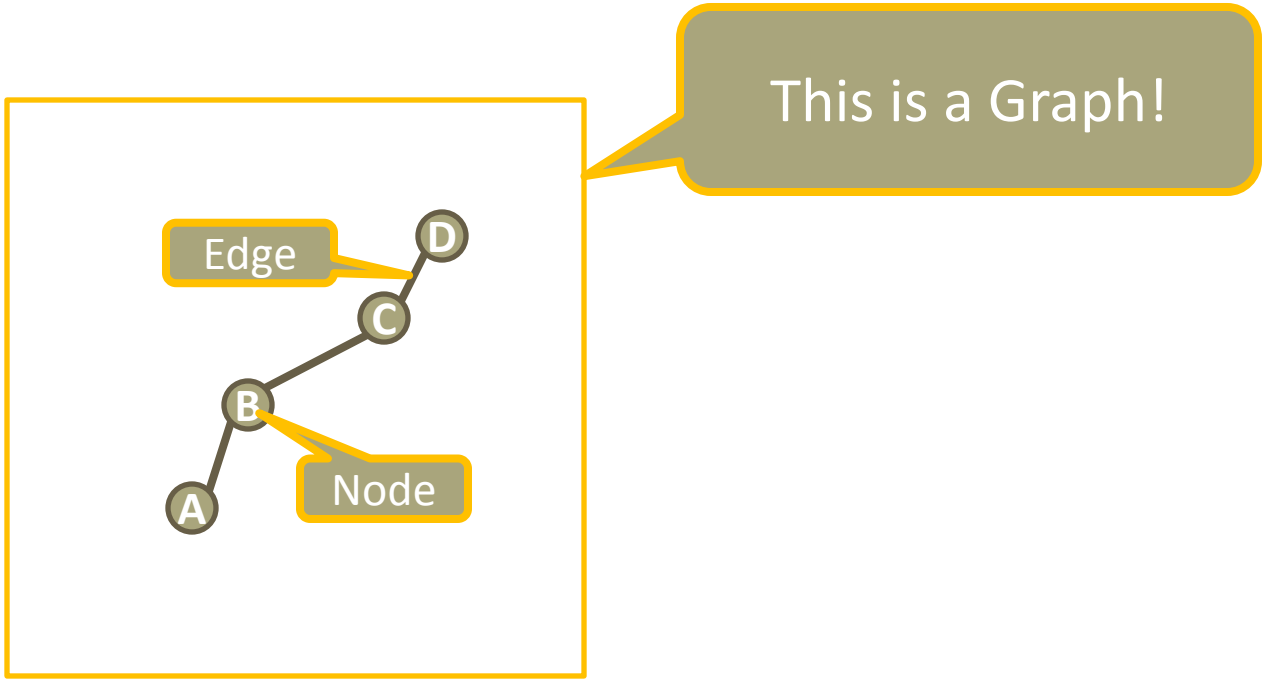
Graph Data: What is it? (4)



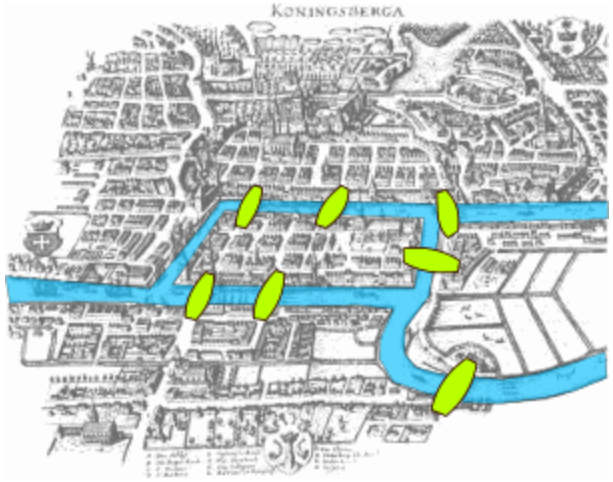
Graph Data: What is it? (5)



Graph Data: What is it? (6)



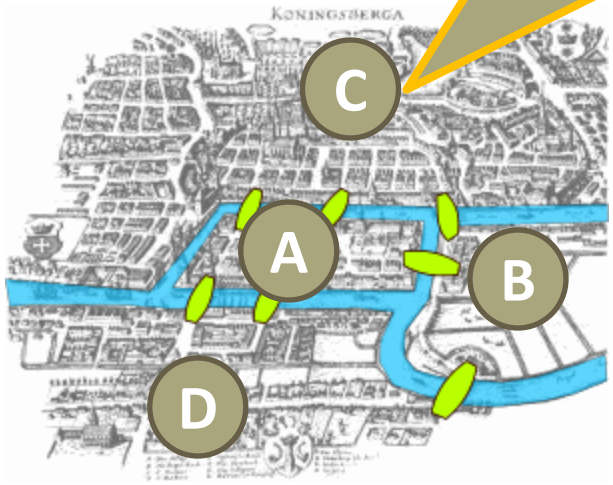
Graph Data: Euler's Seven Bridges of Königsberg (0)



"find a walk through the city that would cross each bridge once and only once"
http://en.wikipedia.org/wiki/Seven_Bridges_of_K%C3%B6nigsberg

Graph Data: Euler's Seven Bridges of Königsberg (1)

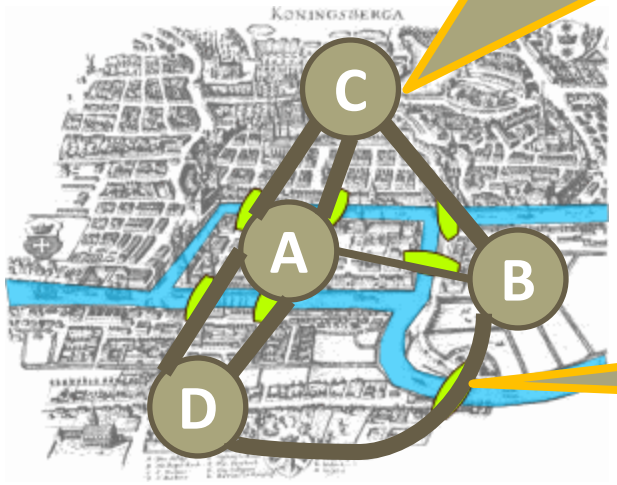
Each separated piece of land is a node



"find a walk through the city that would cross each bridge once and only once"
http://en.wikipedia.org/wiki/Seven_Bridges_of_K%C3%B6nigsberg

Graph Data: Euler's Seven Bridges of Königsberg (2)

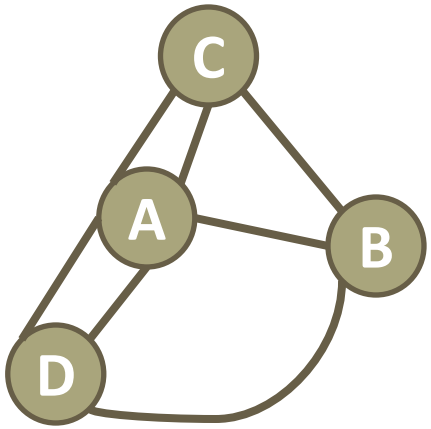
Each separated piece of land is a node



Each bridge is an edge

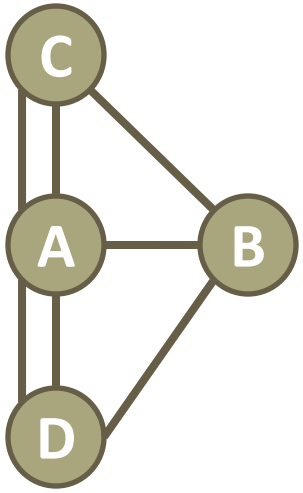
"find a walk through the city that would cross each bridge once and only once"
http://en.wikipedia.org/wiki/Seven_Bridges_of_K%C3%B6nigsberg

Graph Data: Euler's Seven Bridges of Königsberg (3)



"find a walk through the city that would cross each bridge once and only once"
http://en.wikipedia.org/wiki/Seven_Bridges_of_K%C3%B6nigsberg

Graph Data: Euler's Seven Bridges of Königsberg (4)

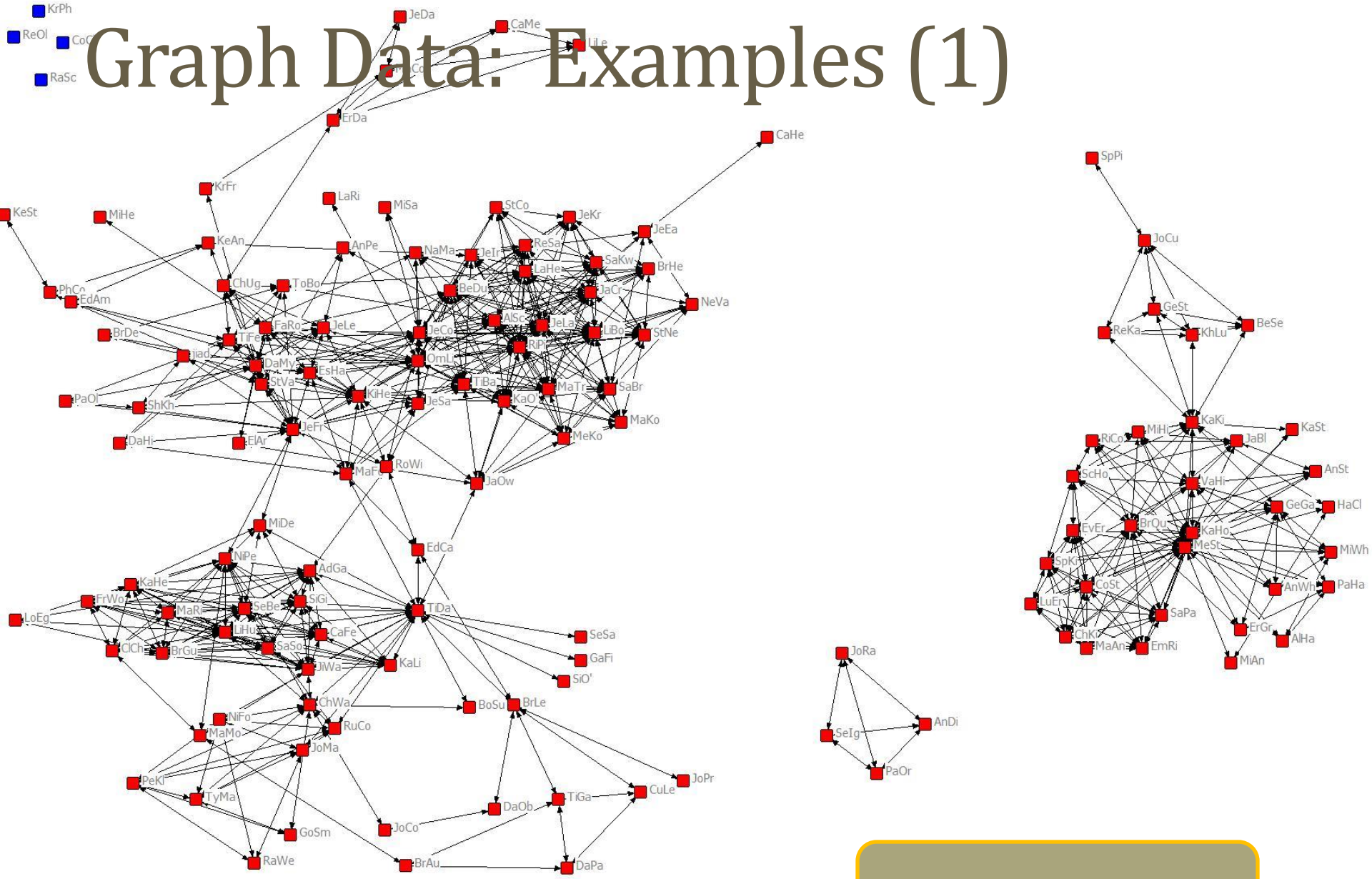


"find a walk through the city that would cross each bridge once and only once"
http://en.wikipedia.org/wiki/Seven_Bridges_of_K%C3%B6nigsberg

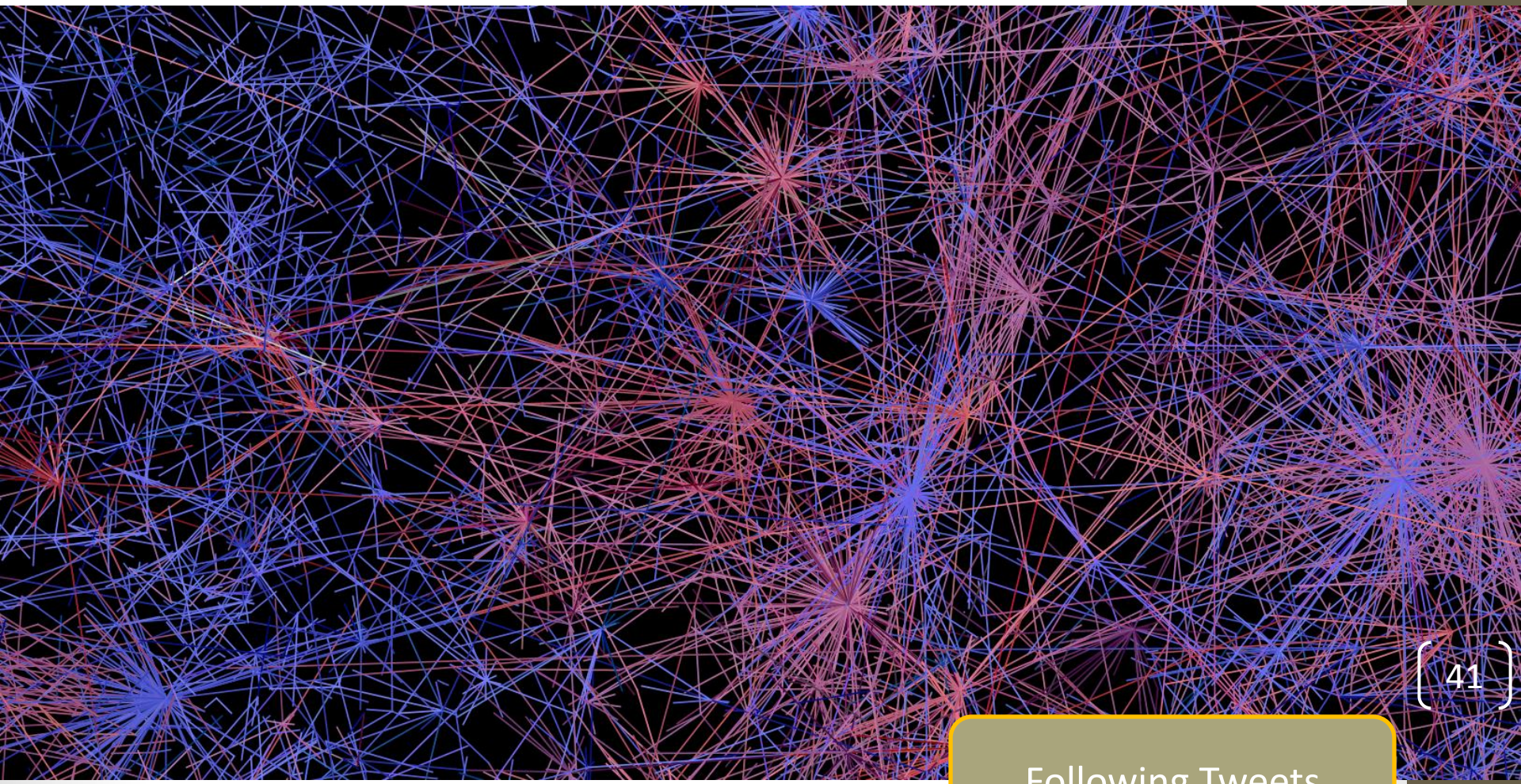
Graph Data: Examples (0)

- Graphs represent many things like Associations and Networks.
- Graphs represent the world wide web
- Graphs represent Facebook networks
- Graphs represent metabolic pathways

Graph Data: Examples (1)

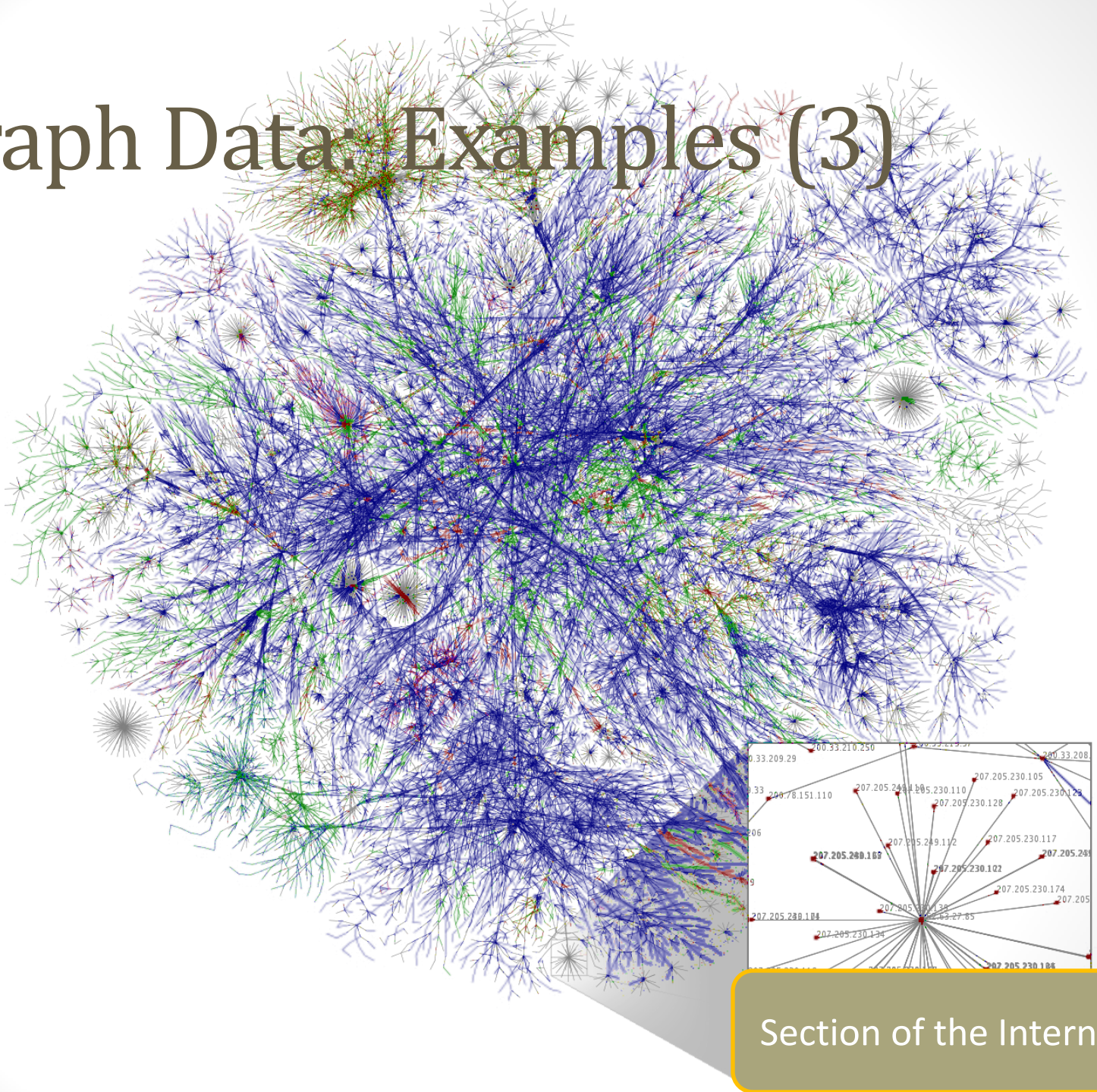


Graph Data: Examples (2)

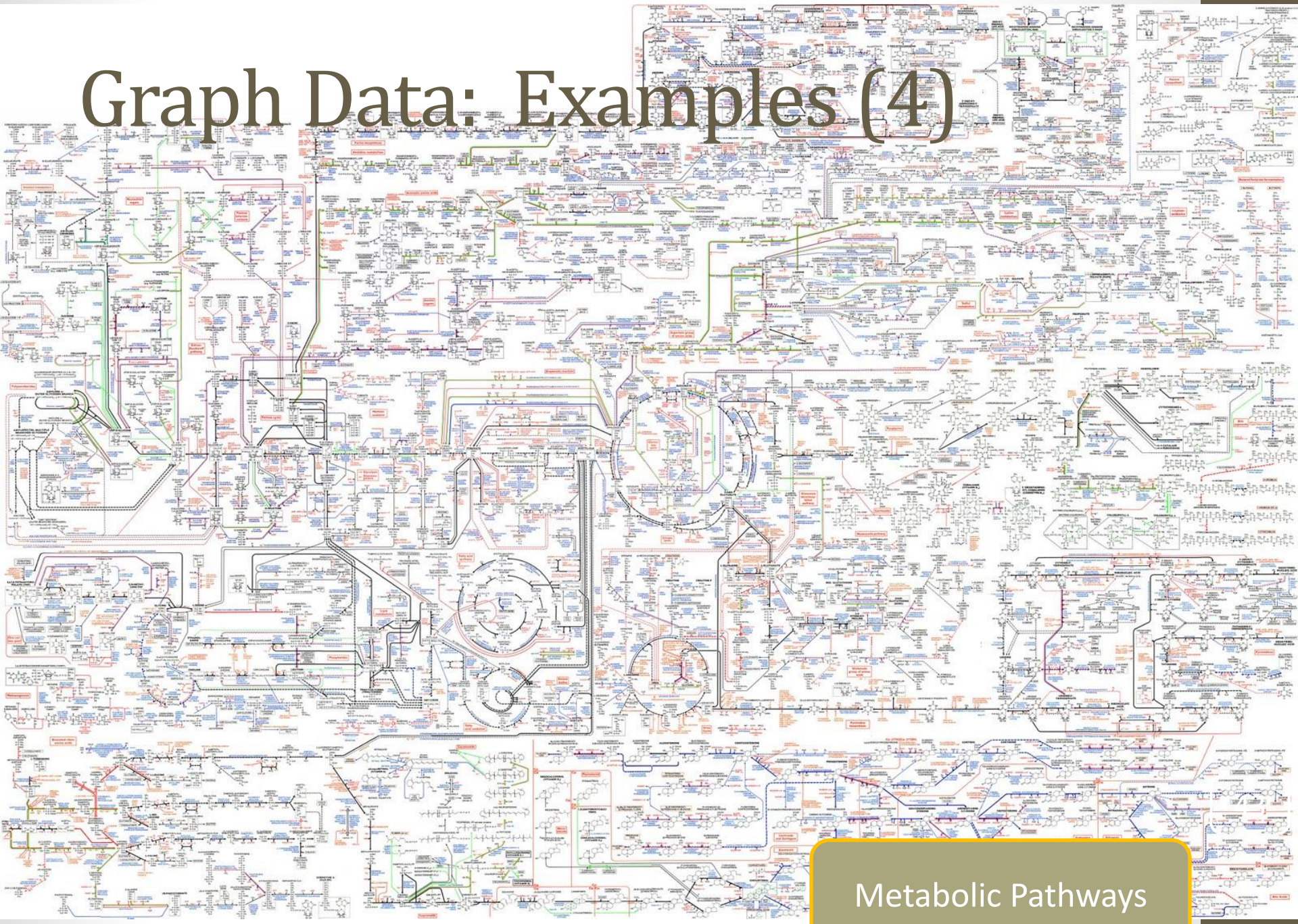


Following Tweets

Graph Data: Examples (3)

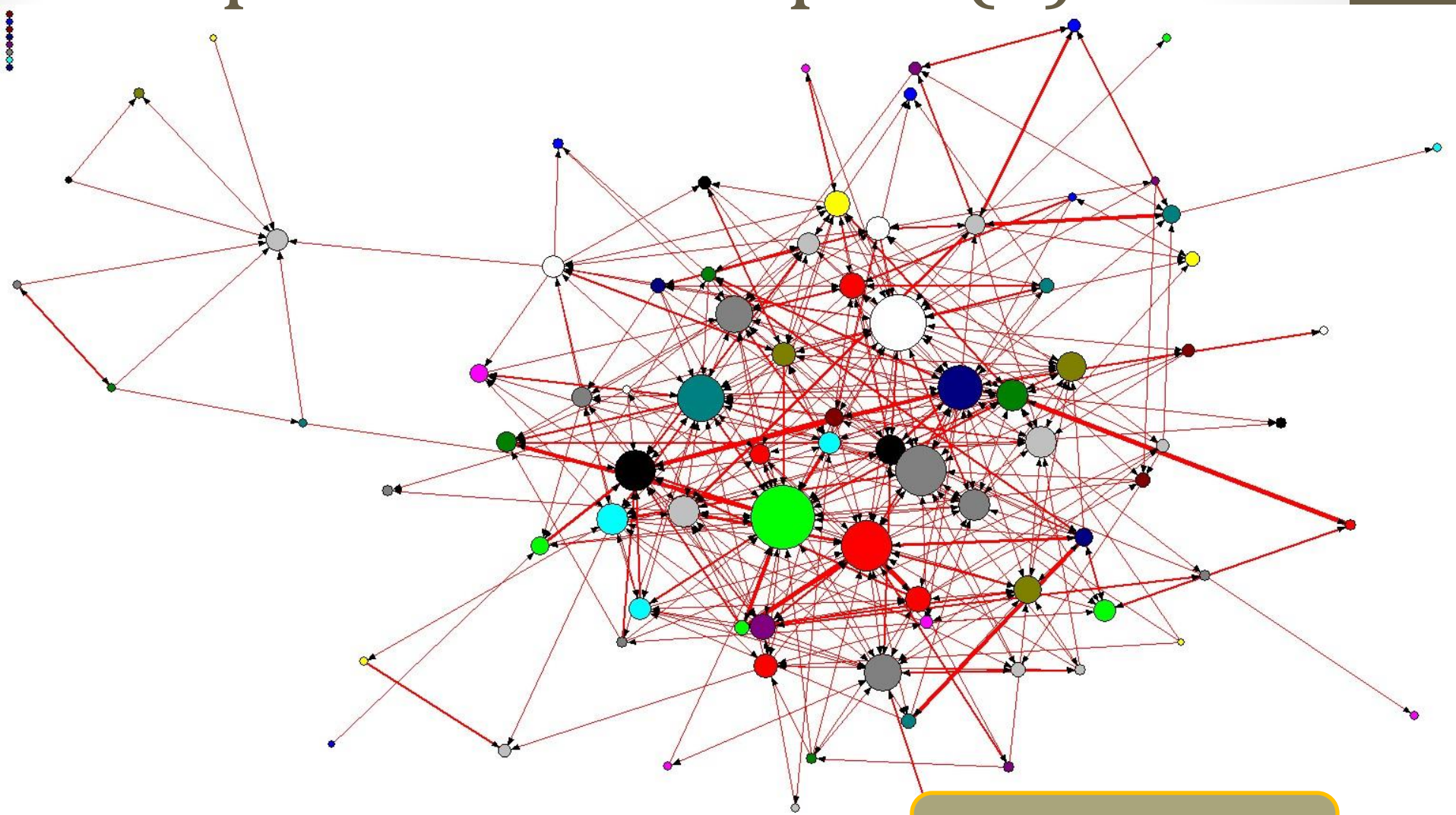


Graph Data: Examples (4)



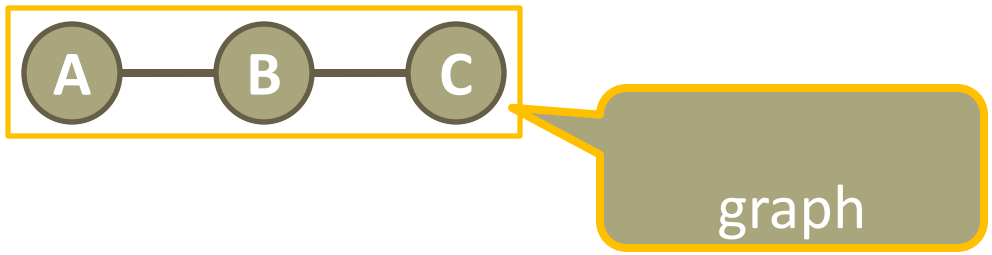
Metabolic Pathways

Graph Data: Examples (5)

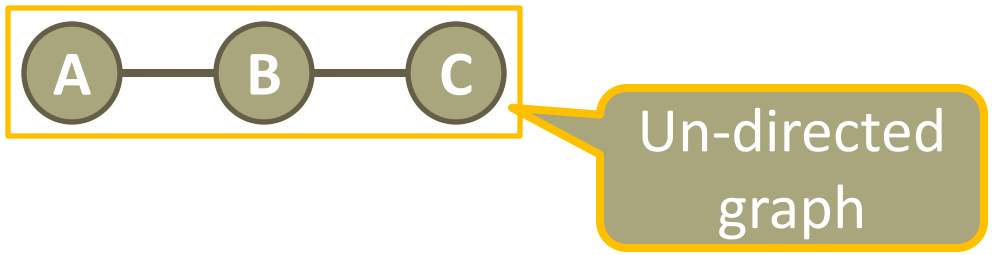


Ranked Web Pages

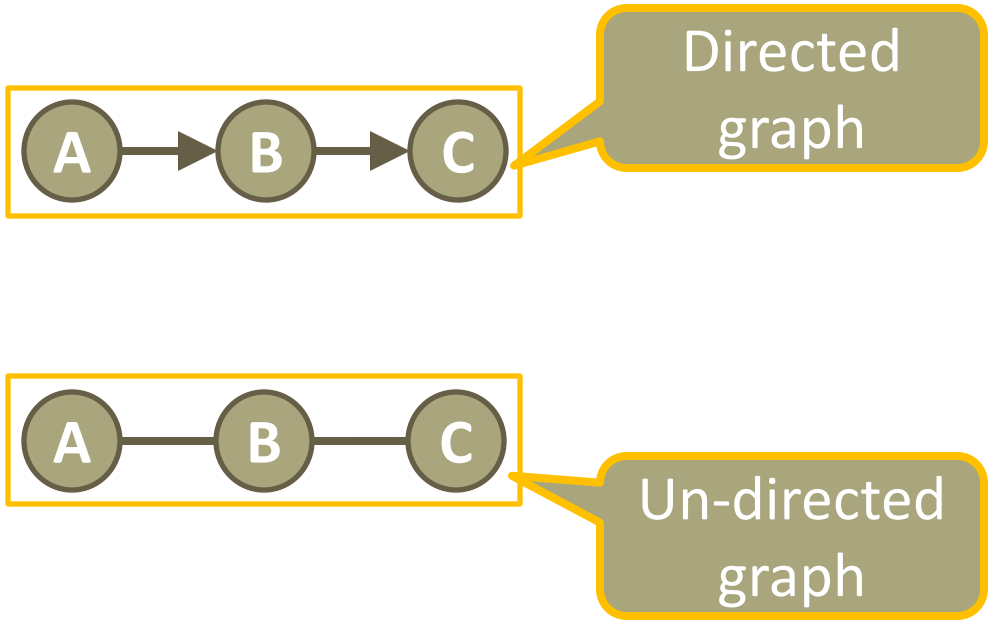
Graph Data: Directed and Un-directed Graphs (0)



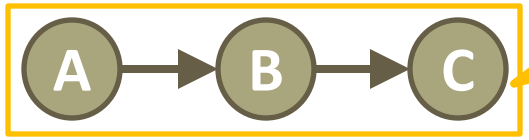
Graph Data: Directed and Un-directed Graphs (1)



Graph Data: Directed and Un-directed Graphs (2)

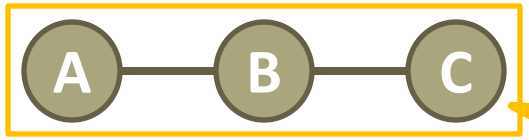


Graph Data: Directed and Un-directed Graphs (3)



Directed graph

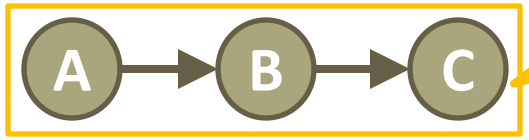
"The link goes from A to B"



Un-directed graph

"The link is between A and B"

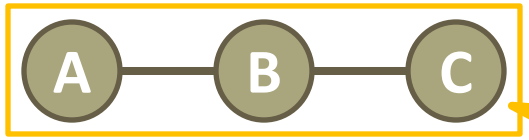
Graph Data: Directed and Un-directed Graphs (4)



Directed graph

"The link goes from A to B"

Example of directed graphs: World Wide Web

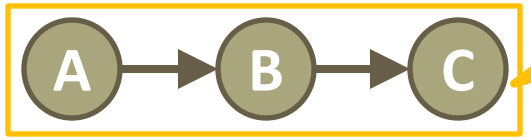


Un-directed graph

"The link is between A and B"

Examples of un-directed graphs: Facebook, LinkedIn

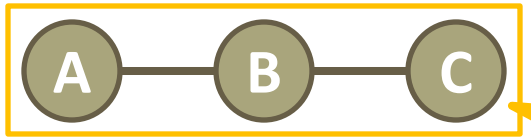
Graph Data: Directed and Un-directed Graphs (5)

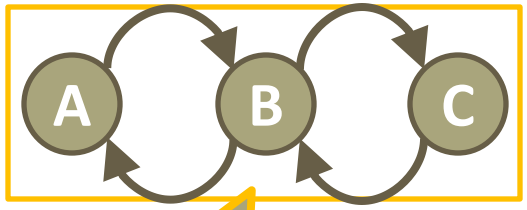


Directed graph

"The link goes from A to B"

Example of directed graphs: World Wide Web





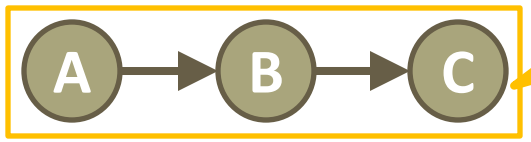
Un-directed graph

"The link is between A and B"

Examples of un-directed graphs: Facebook, LinkedIn

graph with bi-directional links

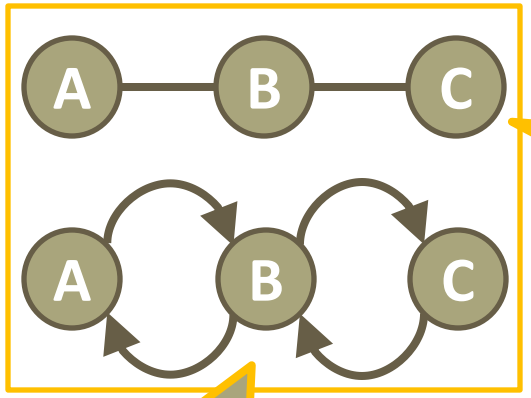
Graph Data: Directed and Un-directed Graphs (6)



Directed graph

"The link goes from A to B"

Example of directed graphs: World Wide Web



Un-directed graph

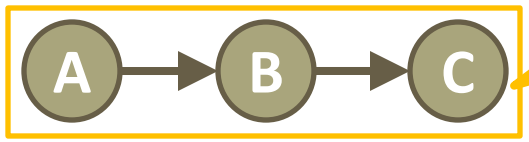
"The link is between A and B"

Examples of un-directed graphs: Facebook, LinkedIn

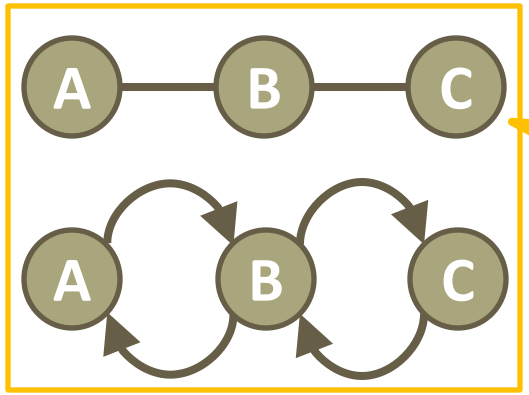
graph with bi-directional links

A graph with bi-directional links is like an undirected graph

Graph Data: Formalize as Rectangular Data (0)



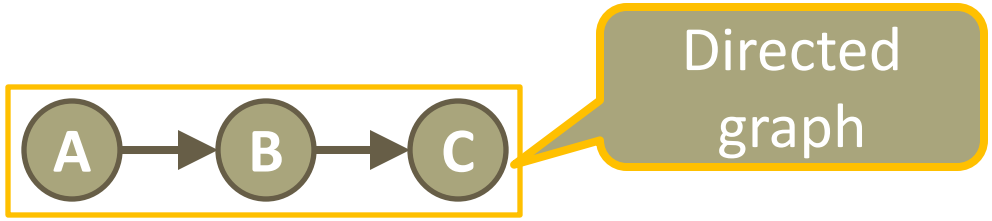
Directed graph



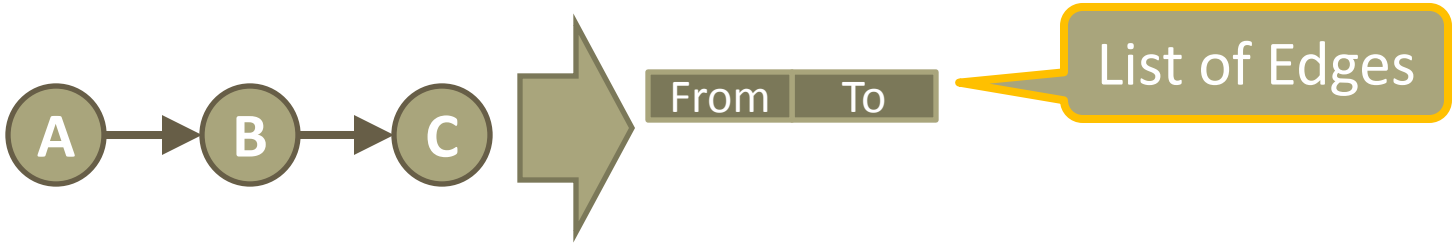
Un-directed graph

- How can we perform operations on graphs?
- How can we formalize graphs as rectangular data?

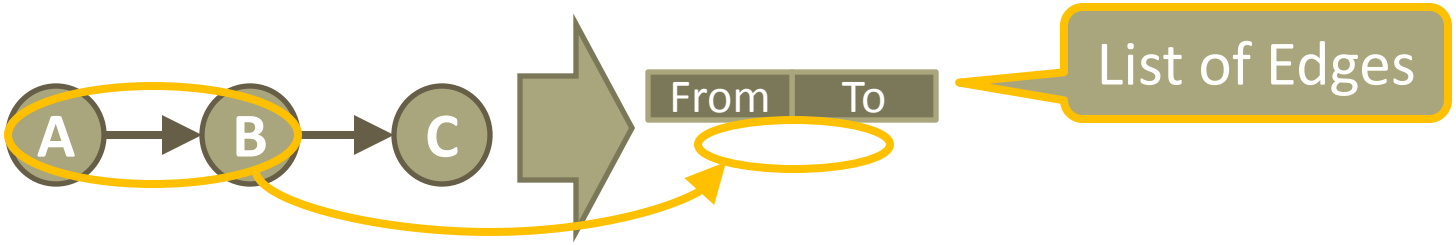
Graph Data: Formalize as Rectangular Data (1)



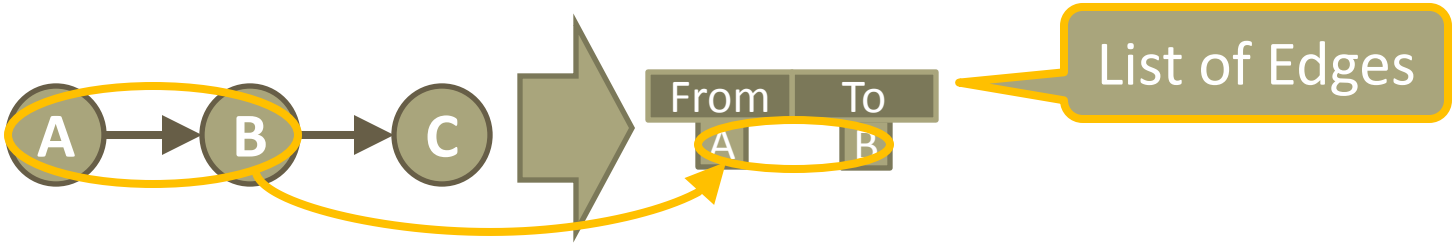
Graph Data: Formalize as Rectangular Data (2)



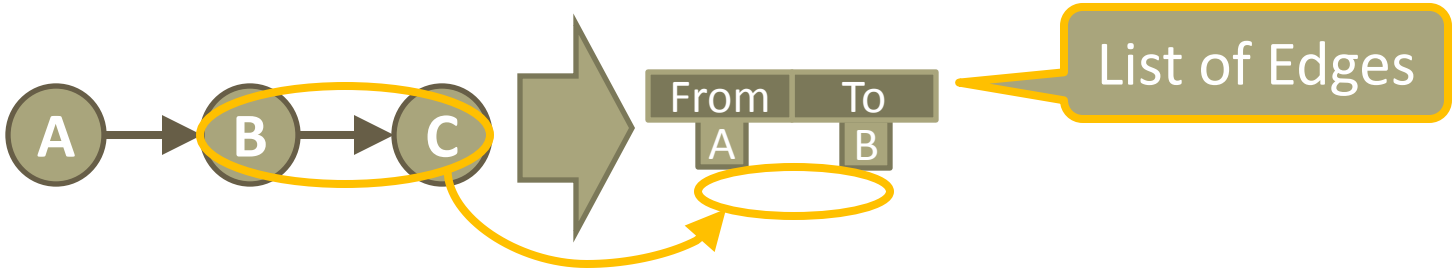
Graph Data: Formalize as Rectangular Data (3)



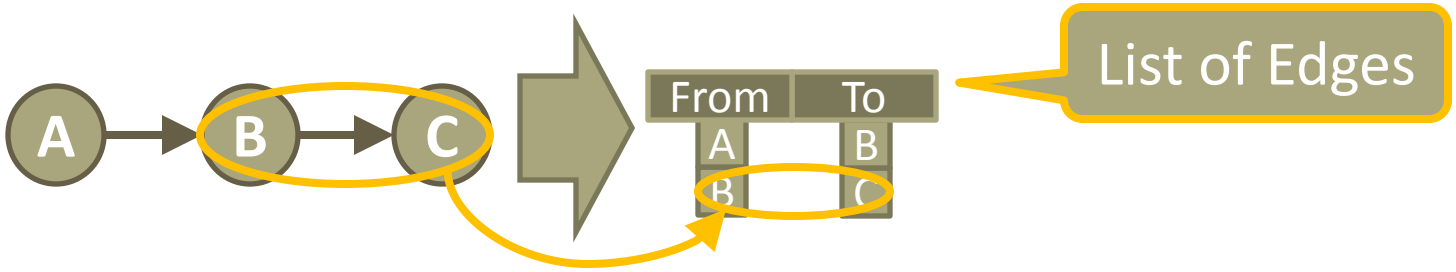
Graph Data: Formalize as Rectangular Data (4)



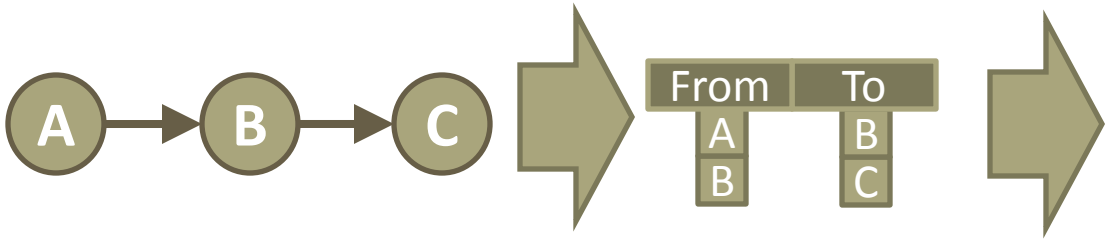
Graph Data: Formalize as Rectangular Data (5)



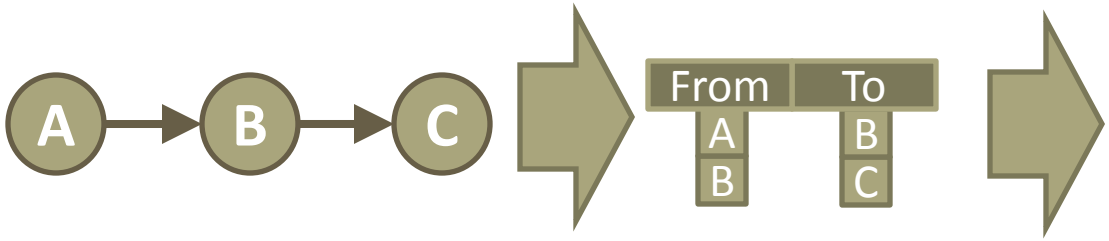
Graph Data: Formalize as Rectangular Data (6)



Graph Data: Formalize as Rectangular Data (7)



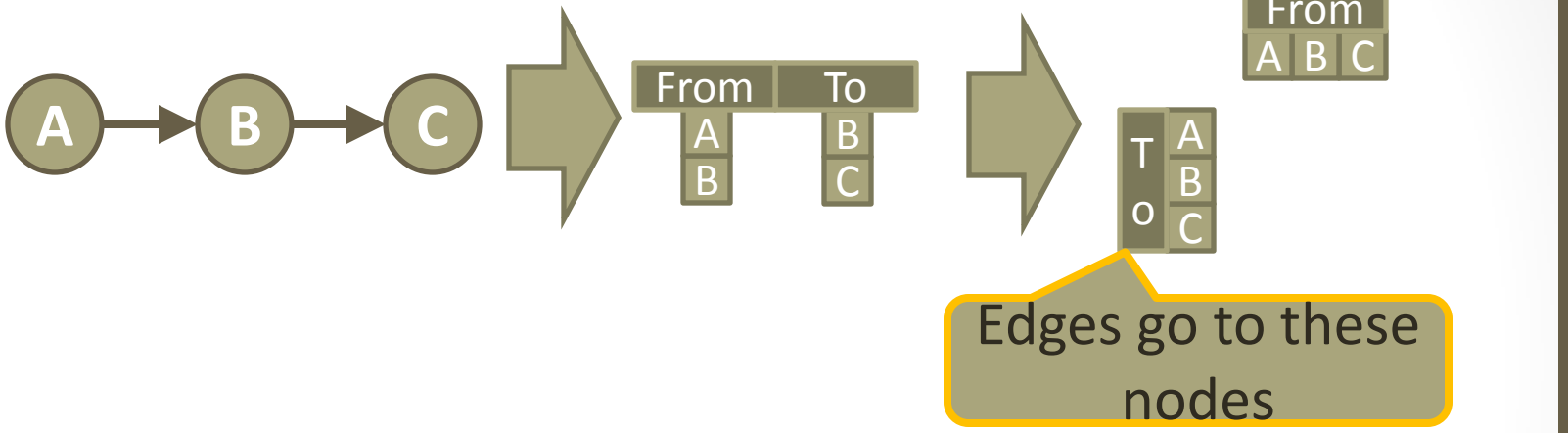
Graph Data: Formalize as Rectangular Data (8)



Edges start from these nodes

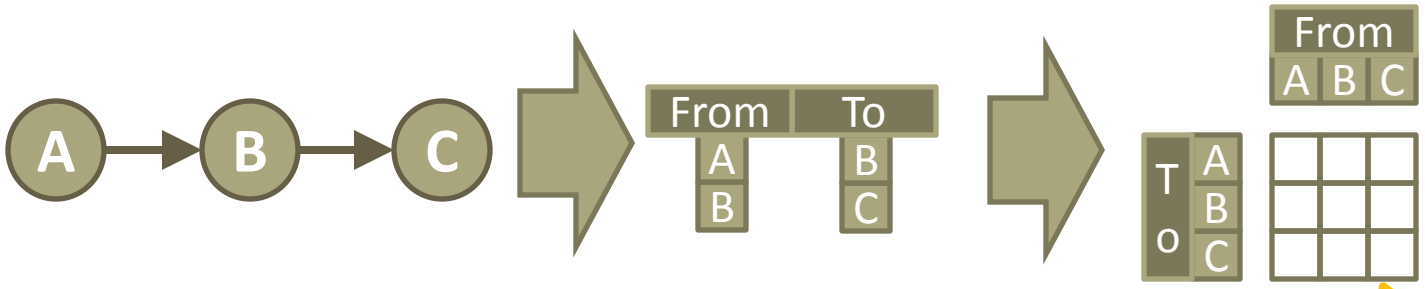
From
A
B
C

Graph Data: Formalize as Rectangular Data (9)



Every node could emanate and receive links.

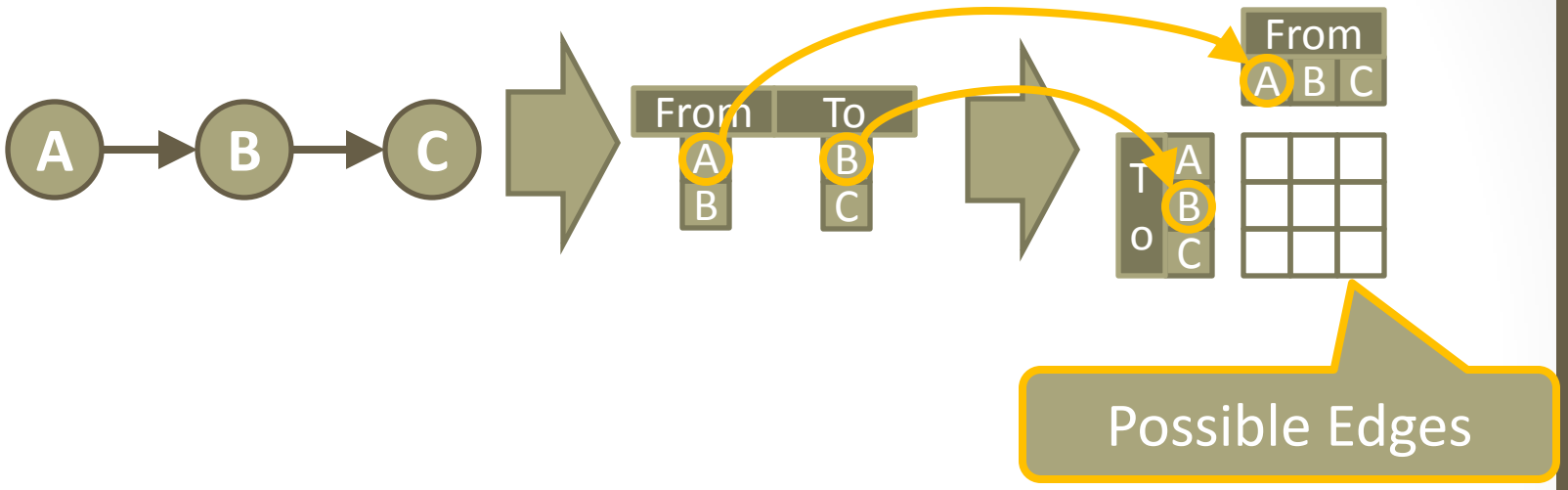
Graph Data: Formalize as Rectangular Data (10)



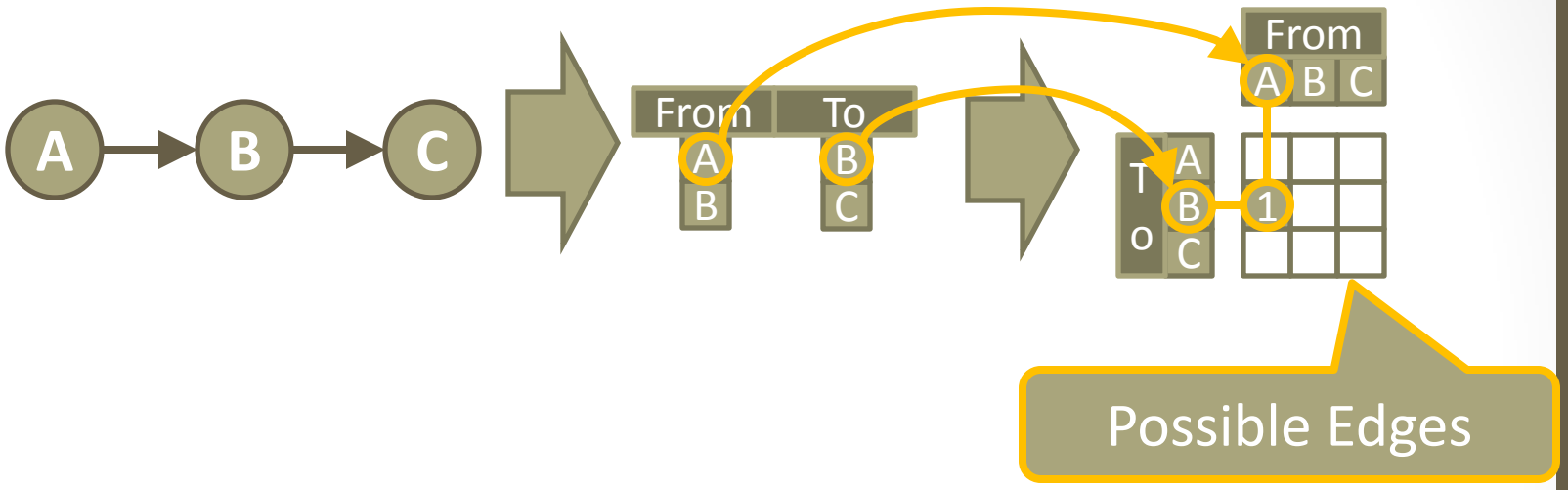
Possible Edges

Every node could emanate and receive links. Therefore the matrix is square.

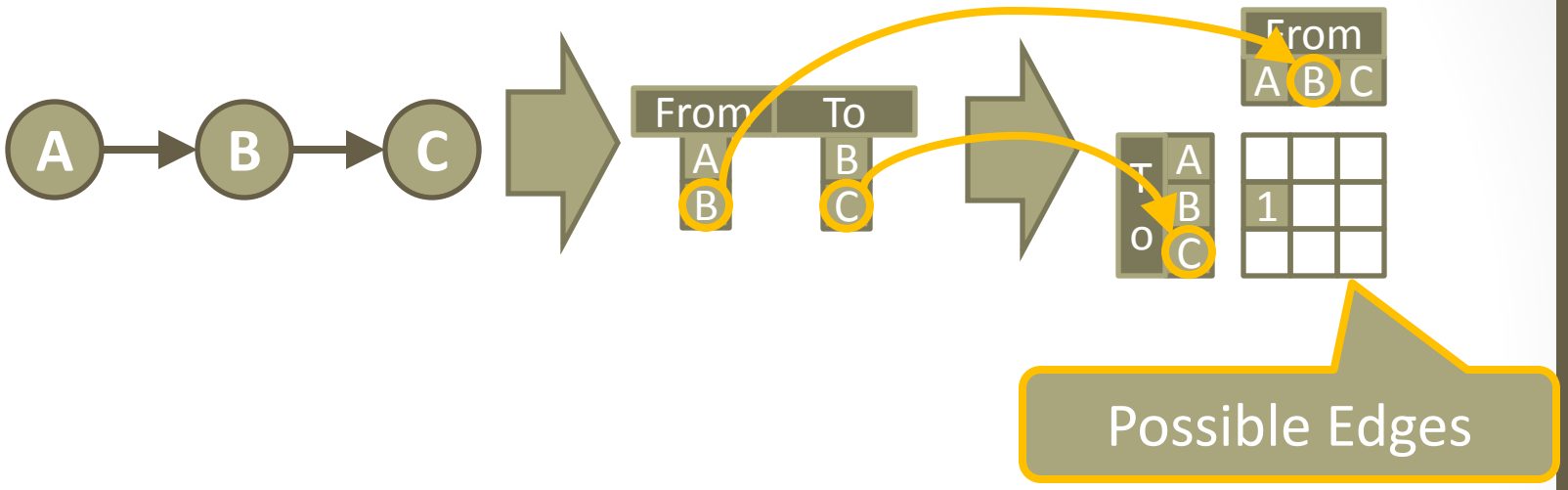
Graph Data: Formalize as Rectangular Data (11)



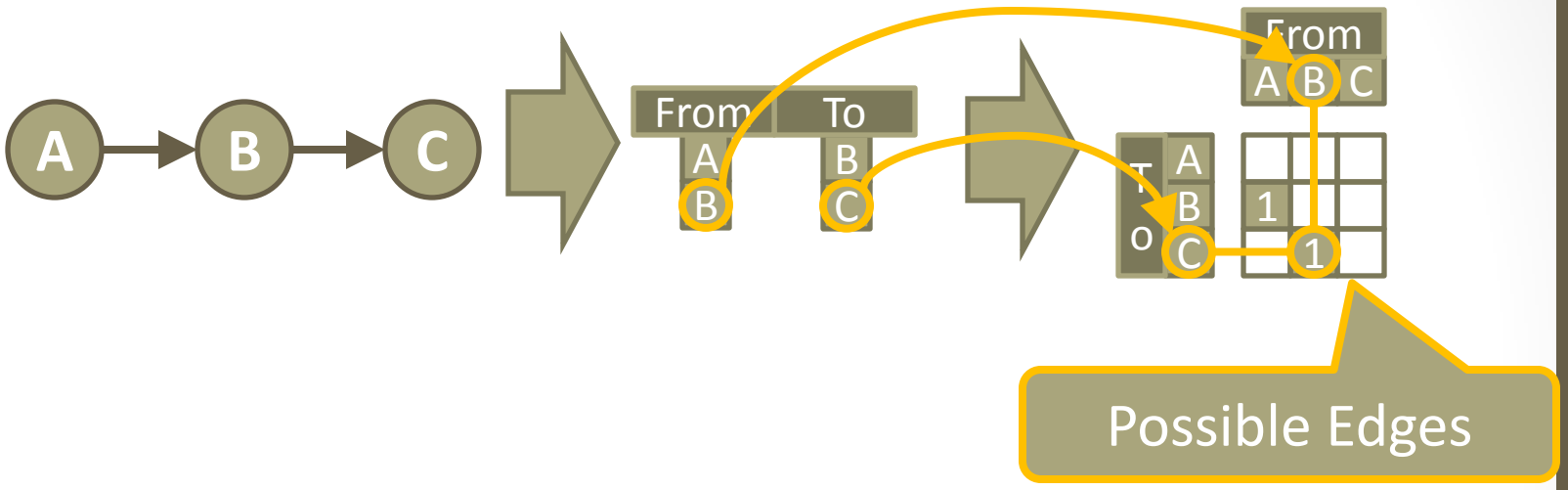
Graph Data: Formalize as Rectangular Data (12)



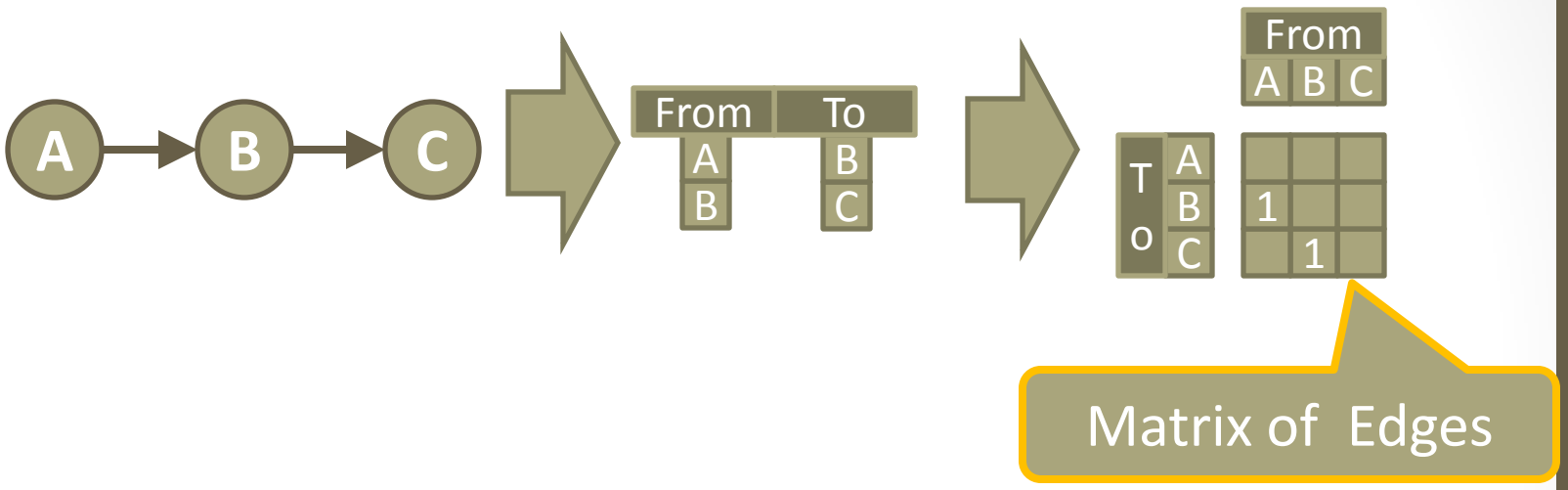
Graph Data: Formalize as Rectangular Data (13)



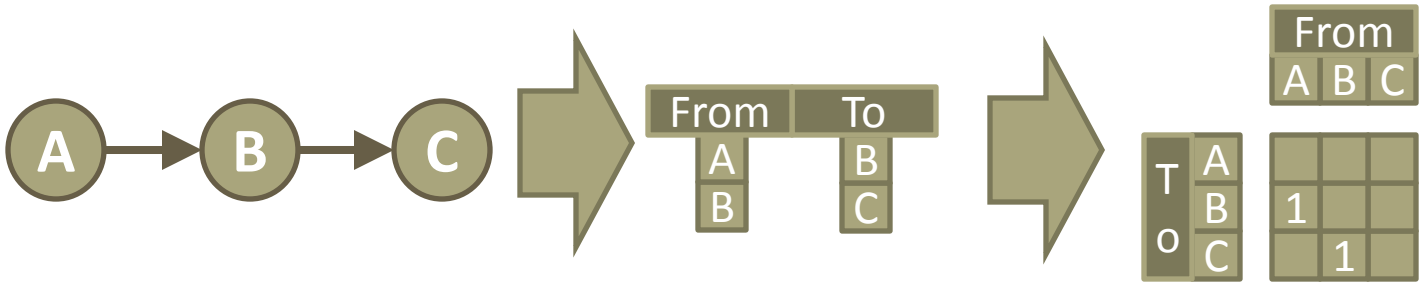
Graph Data: Formalize as Rectangular Data (14)

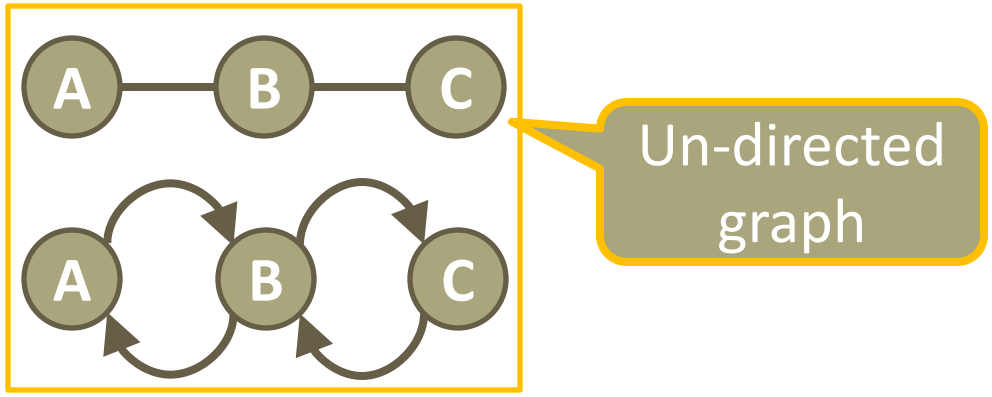


Graph Data: Formalize as Rectangular Data (15)

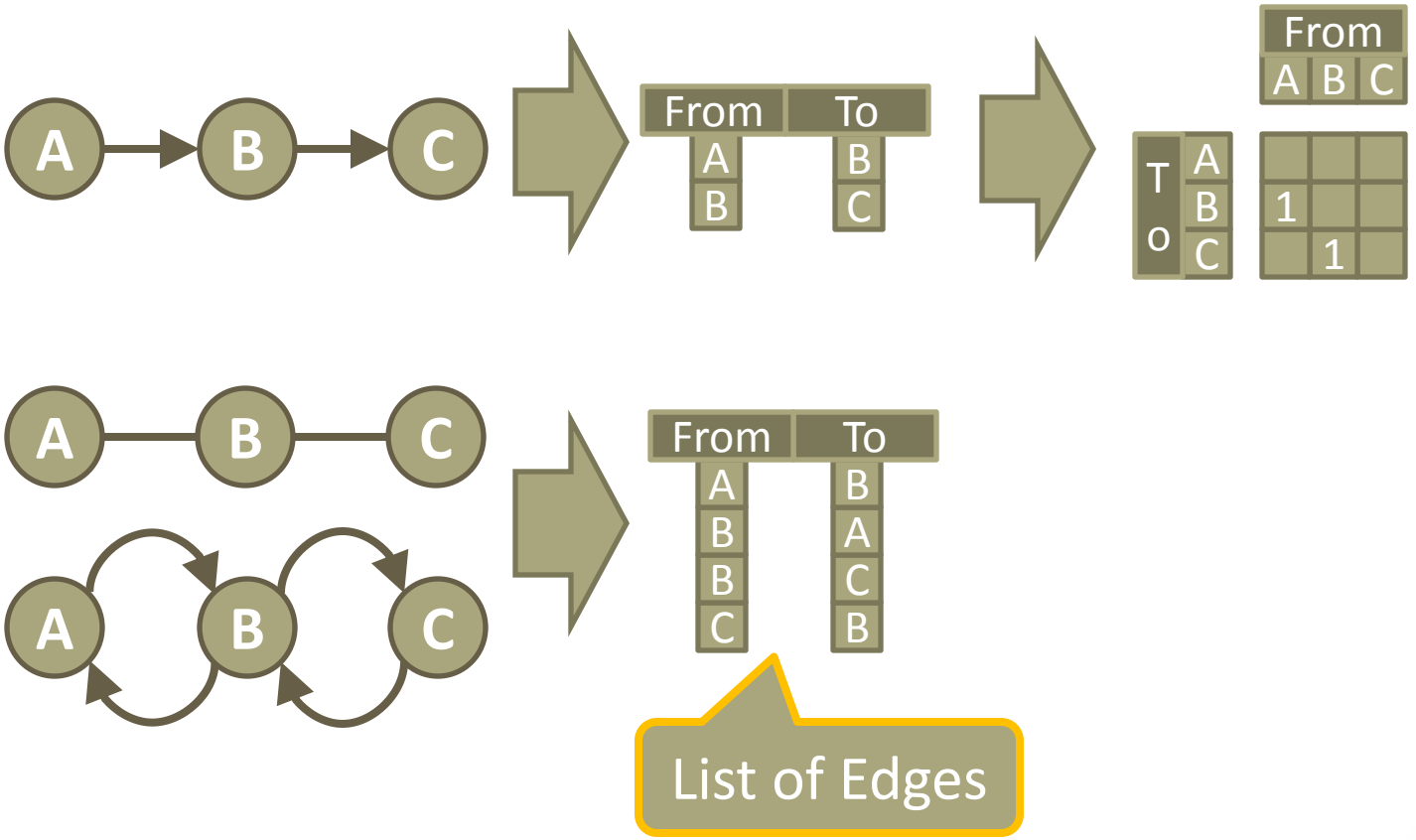


Graph Data: Formalize as Rectangular Data (16)

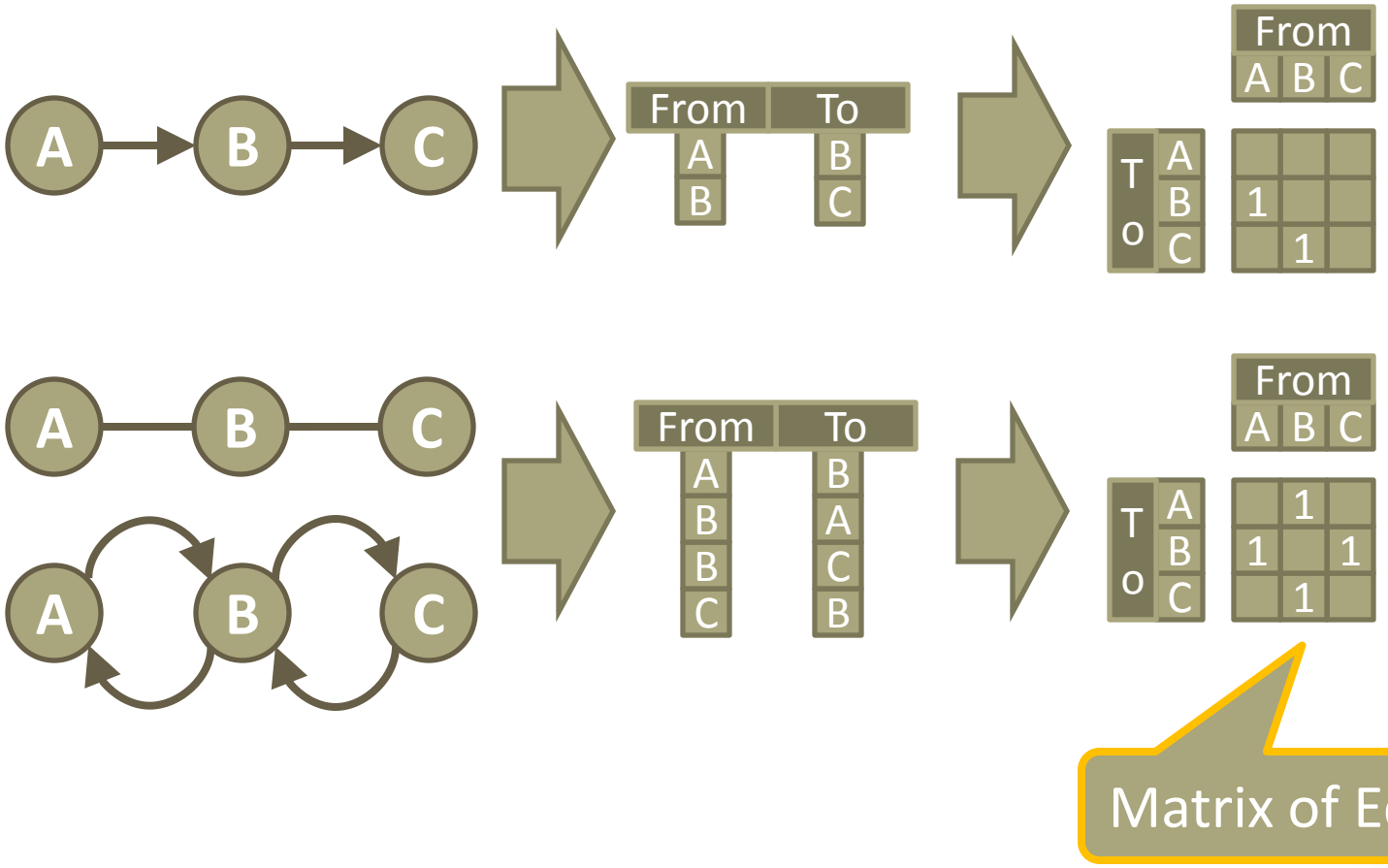




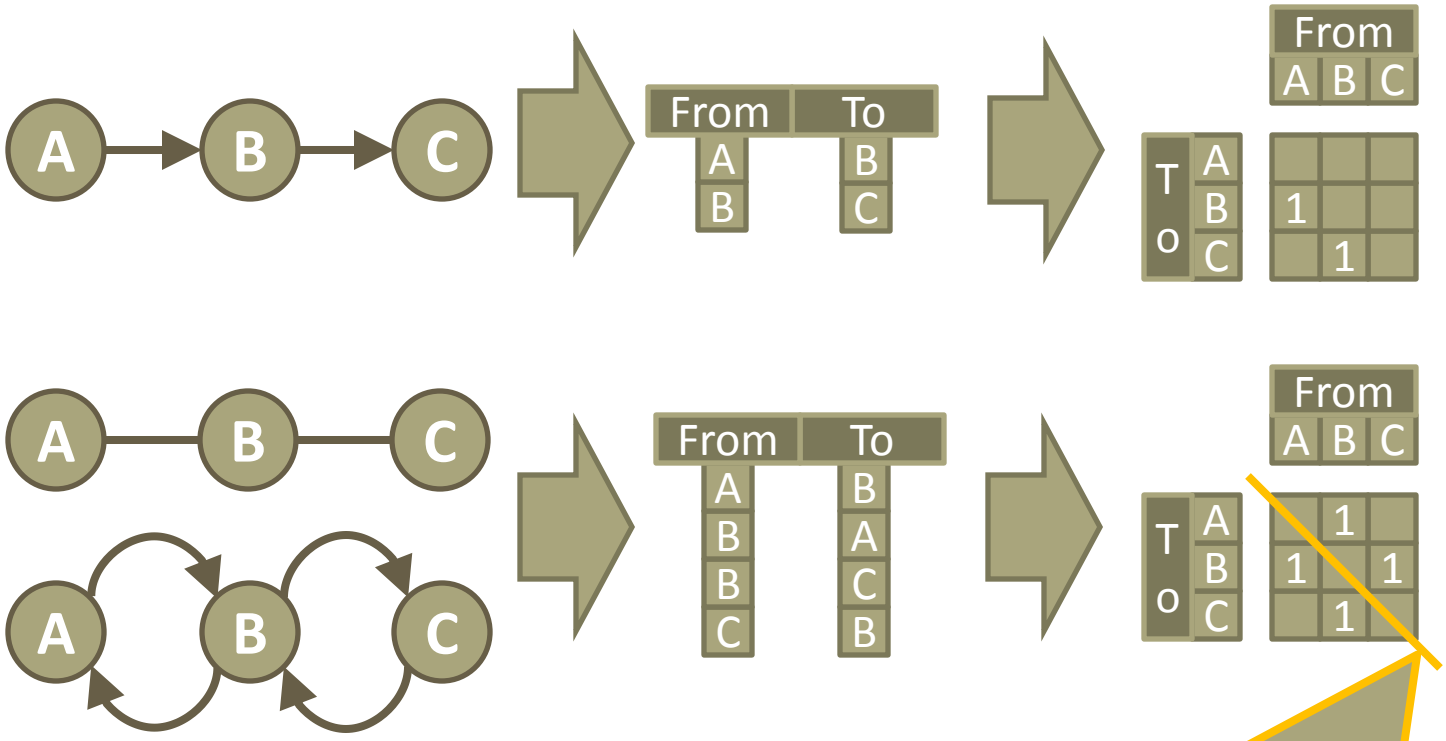
Graph Data: Formalize as Rectangular Data (17)



Graph Data: Formalize as Rectangular Data (18)

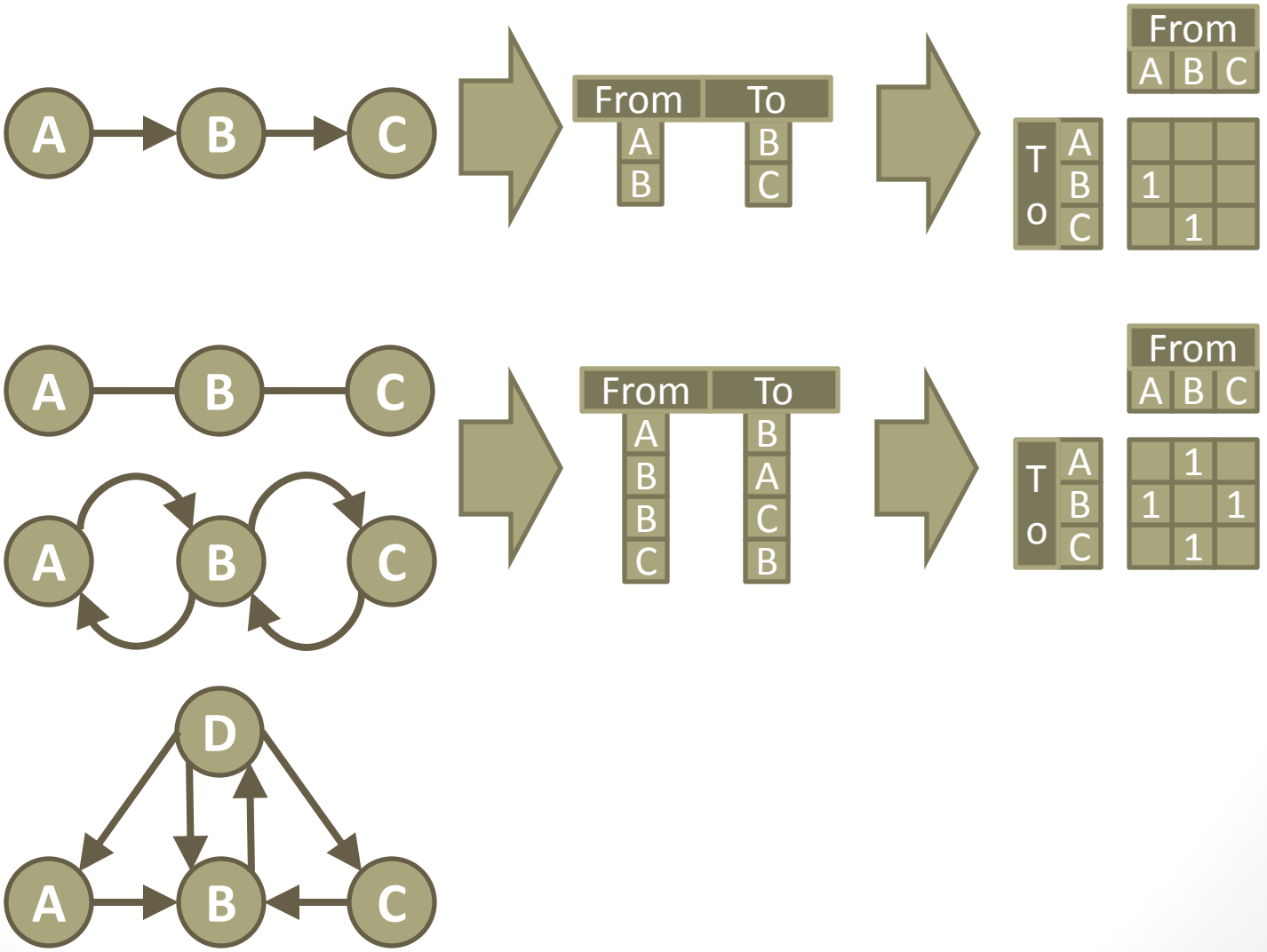


Graph Data: Formalize as Rectangular Data (19)

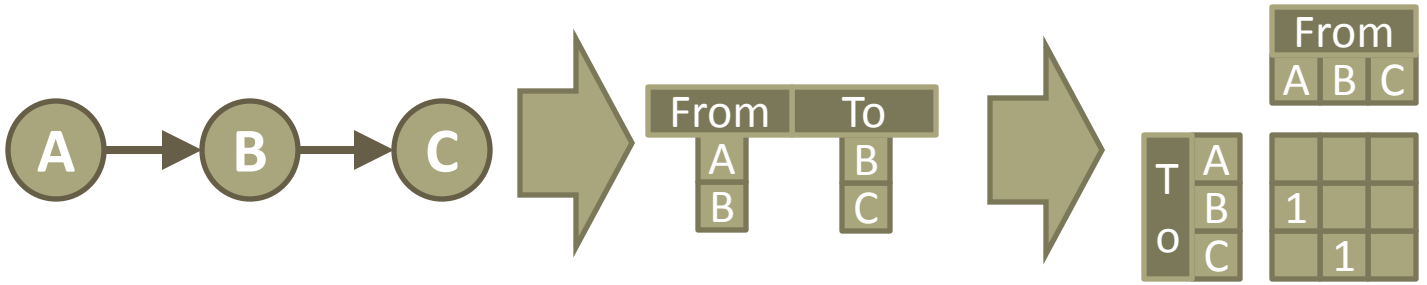


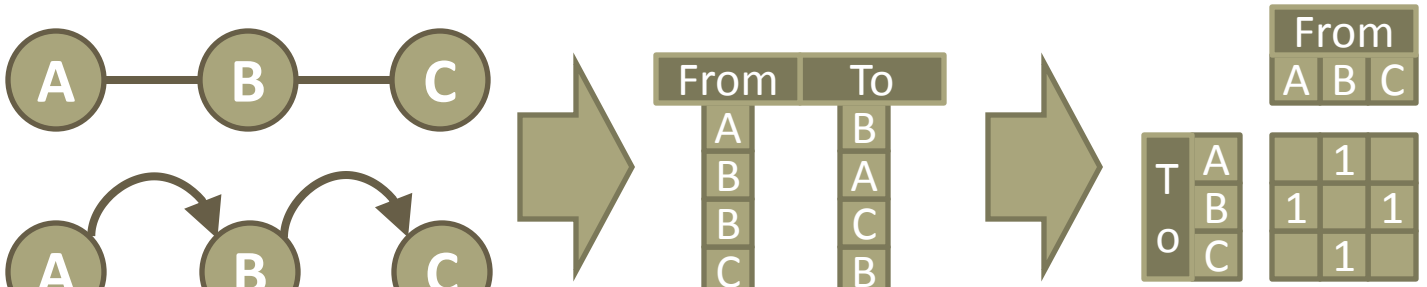
Note: undirected or bi-directional graphs Have symmetric matrices

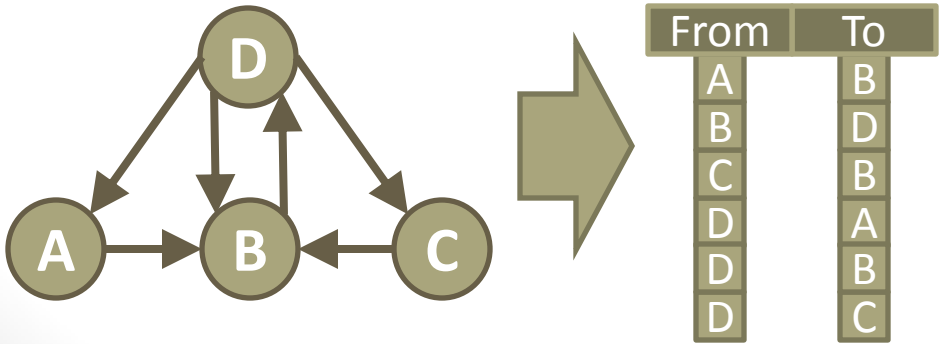
Graph Data: Formalize as Rectangular Data (20)



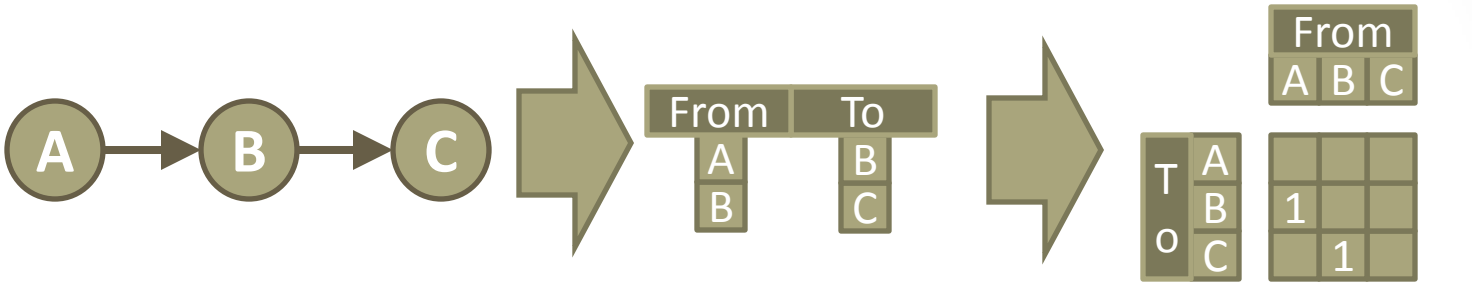
Graph Data: Formalize as Rectangular Data (21)

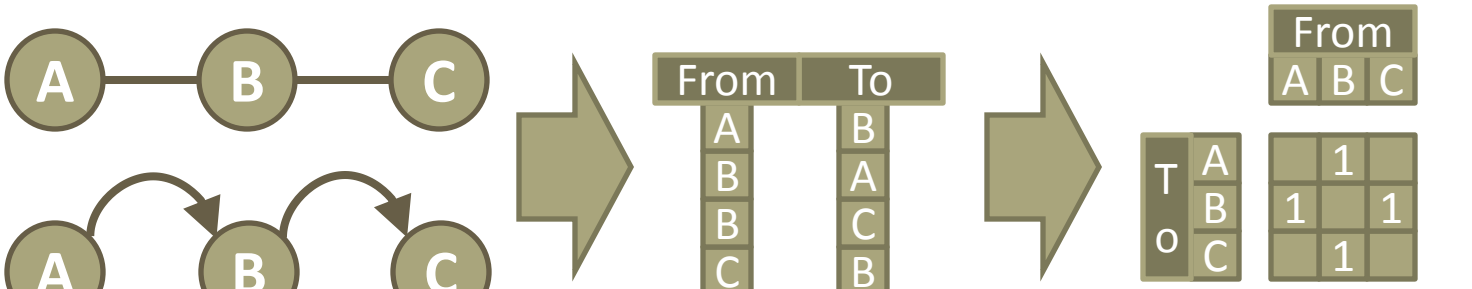


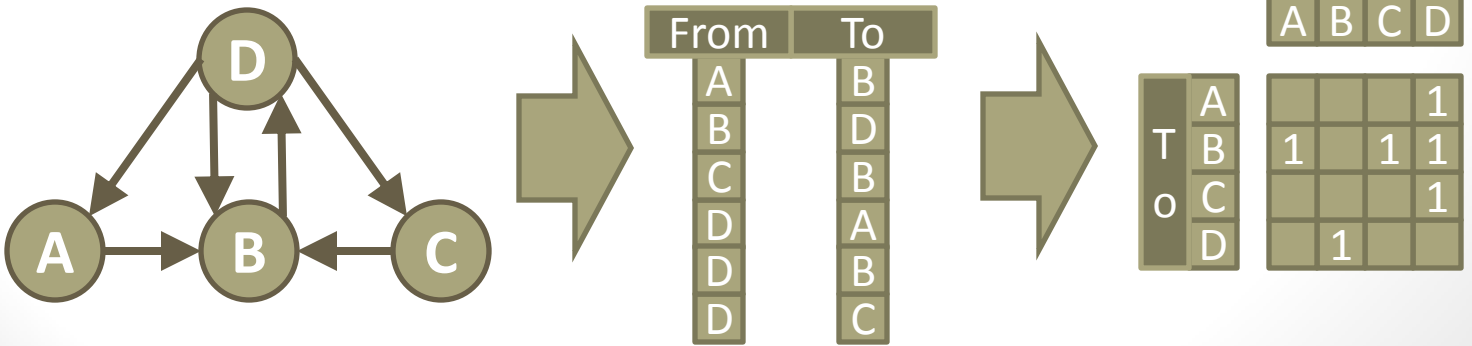




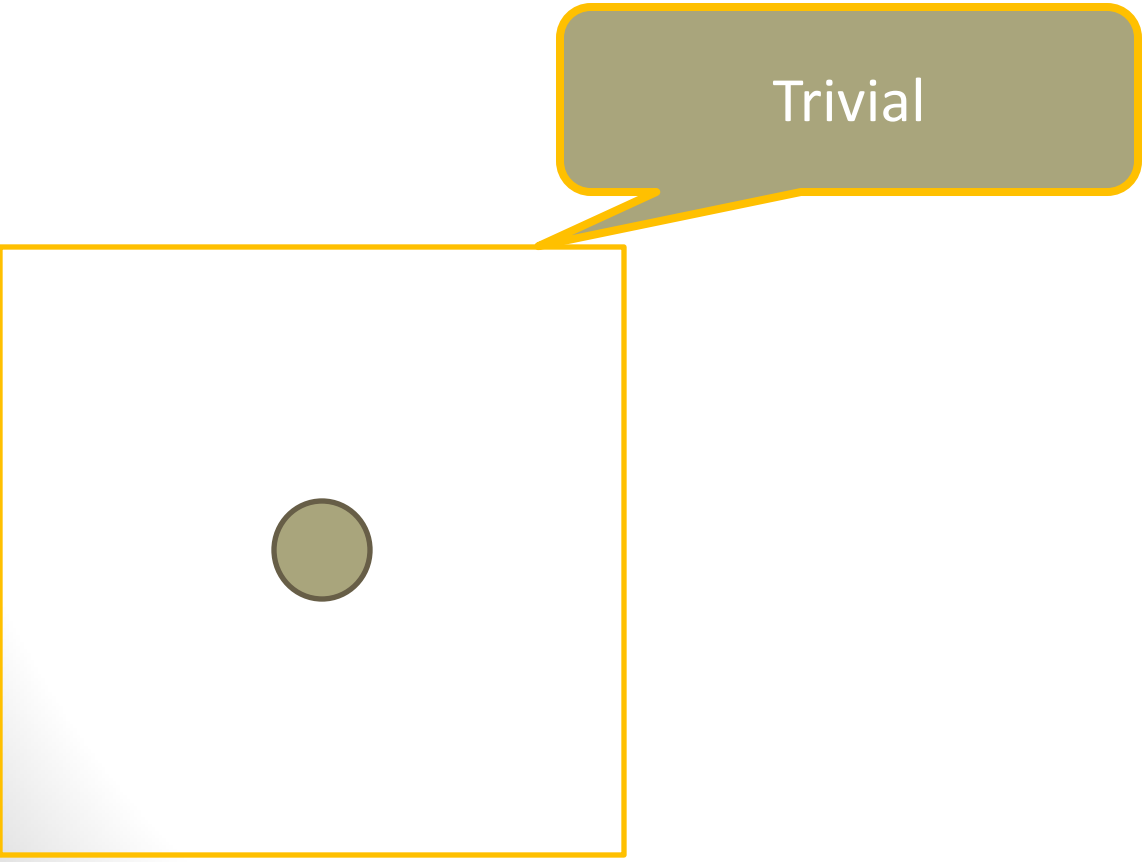
Graph Data: Formalize as Rectangular Data (22)



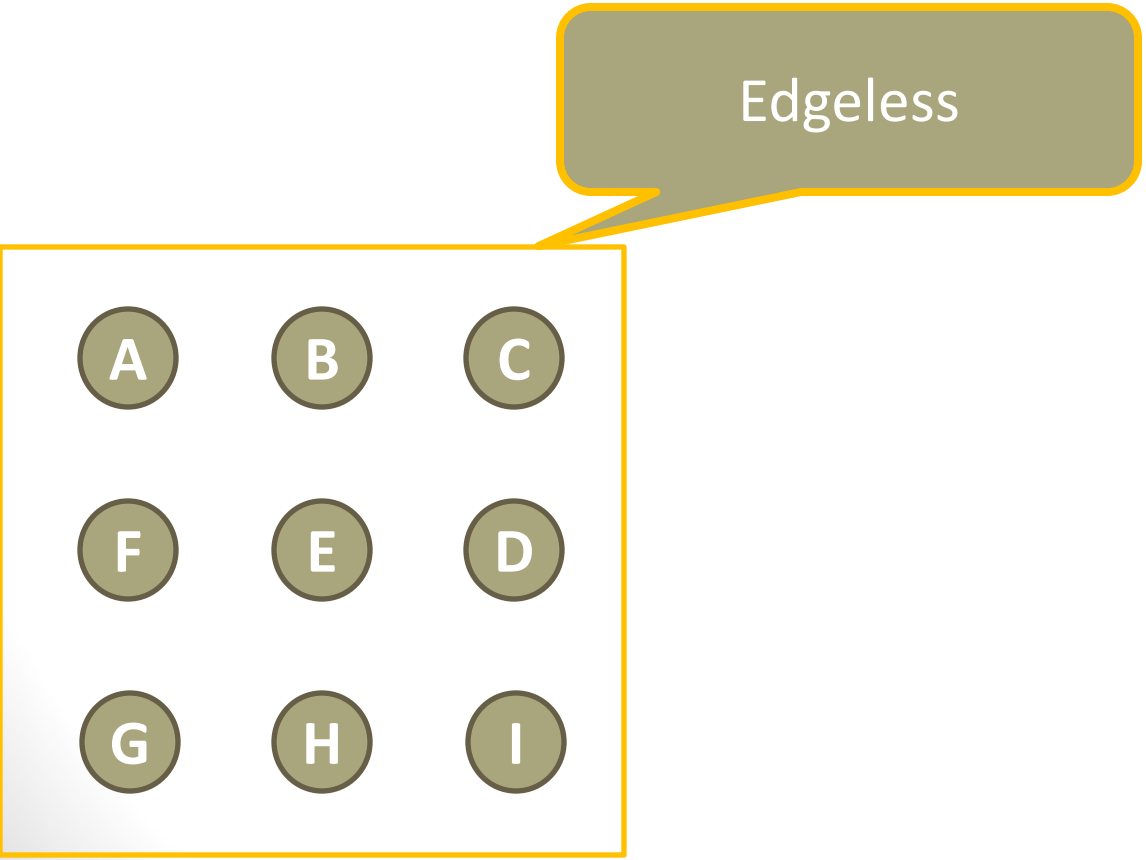




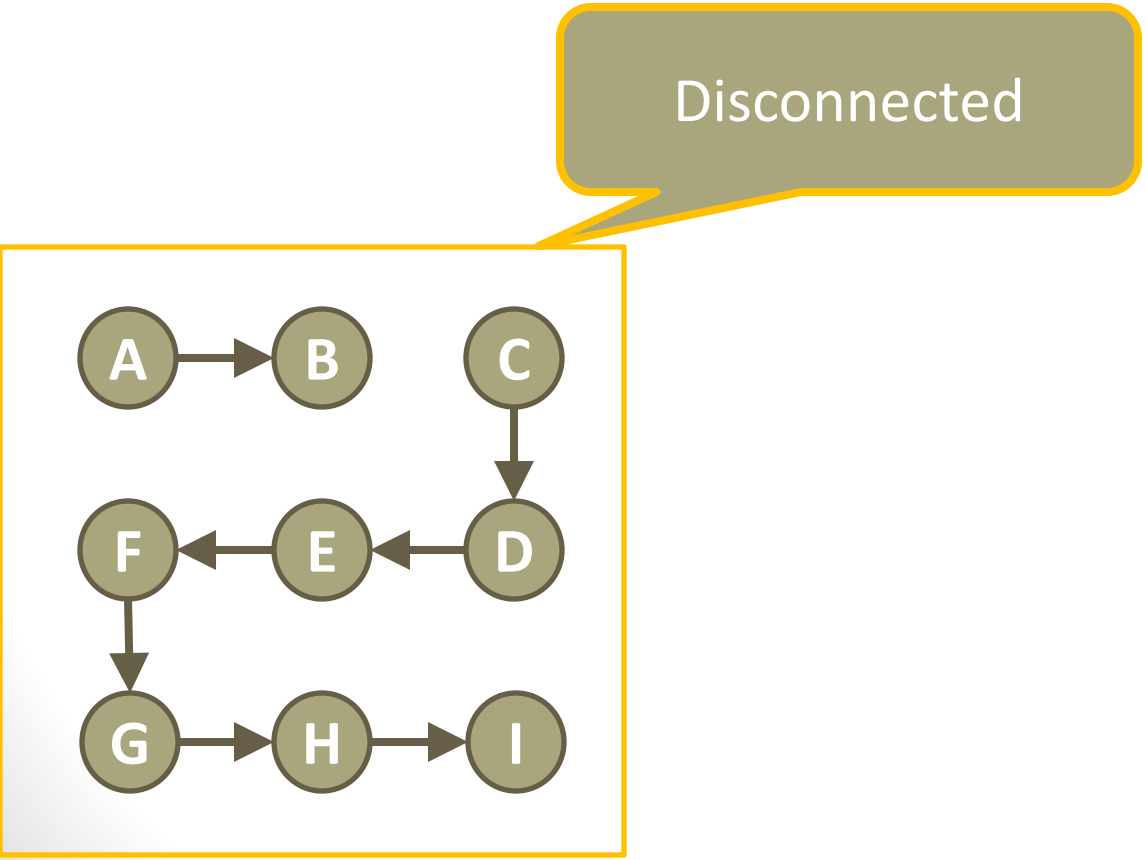
Graph Data: Connectedness and Density (0)



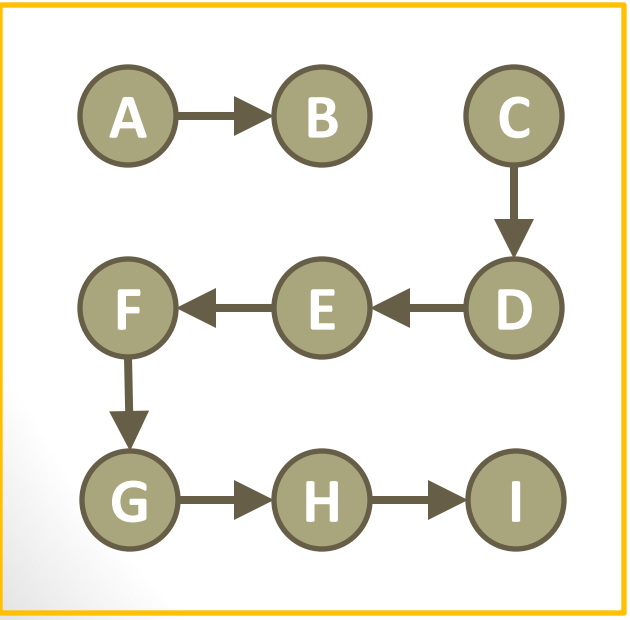
Graph Data: Connectedness and Density (1)

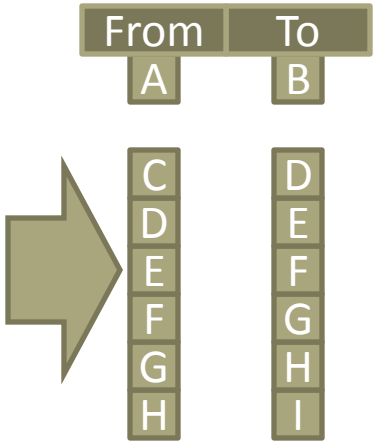


Graph Data: Connectedness and Density (2)

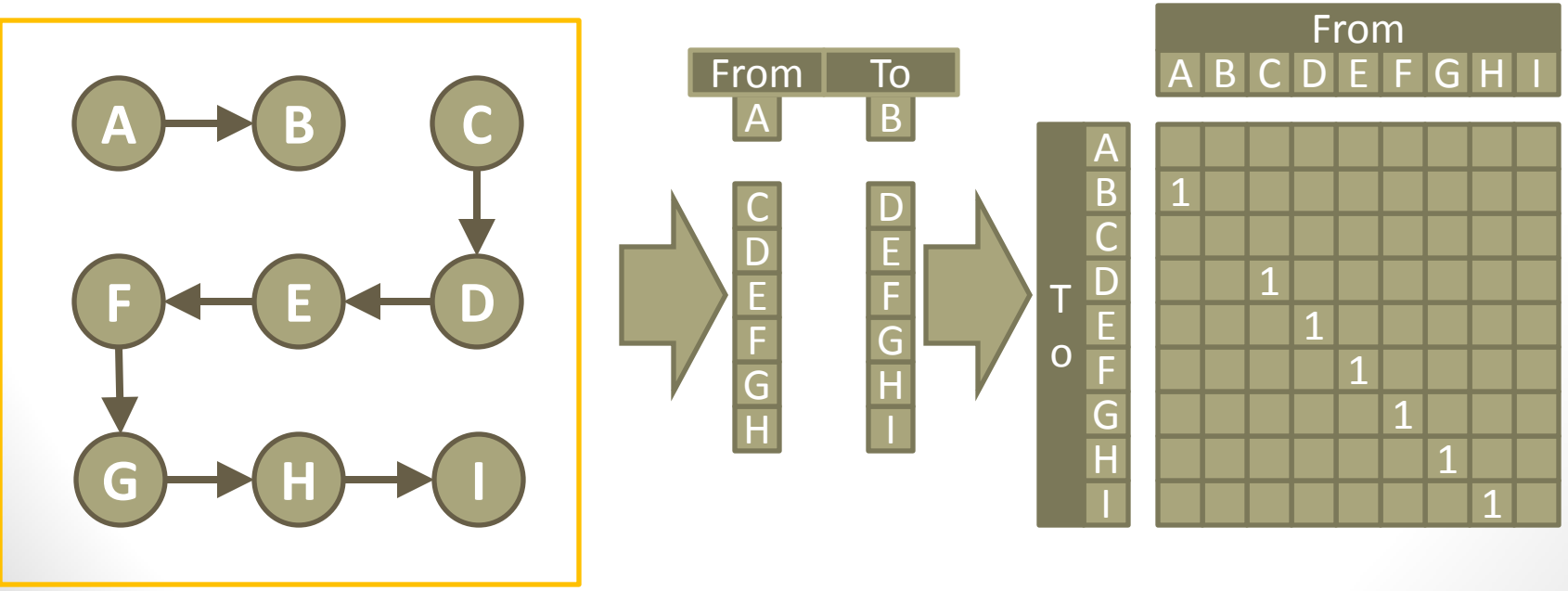


Graph Data: Connectedness and Density (3)

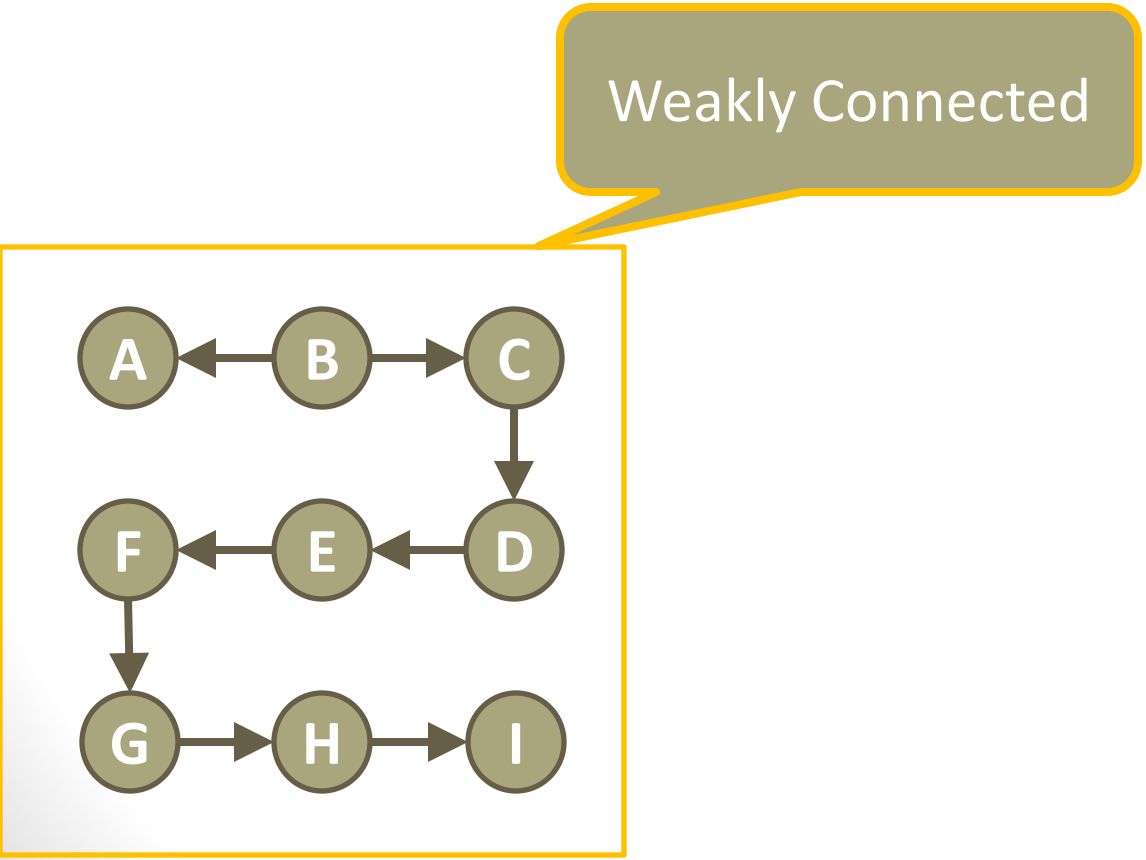




Graph Data: Connectedness and Density (4)

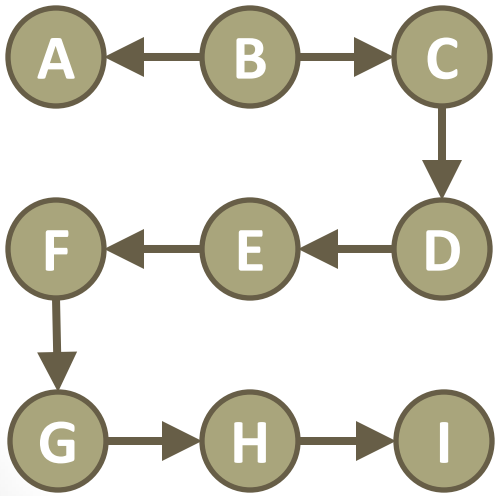


Graph Data: Connectedness and Density (5)



Graph Data: Connectedness and Density (6)

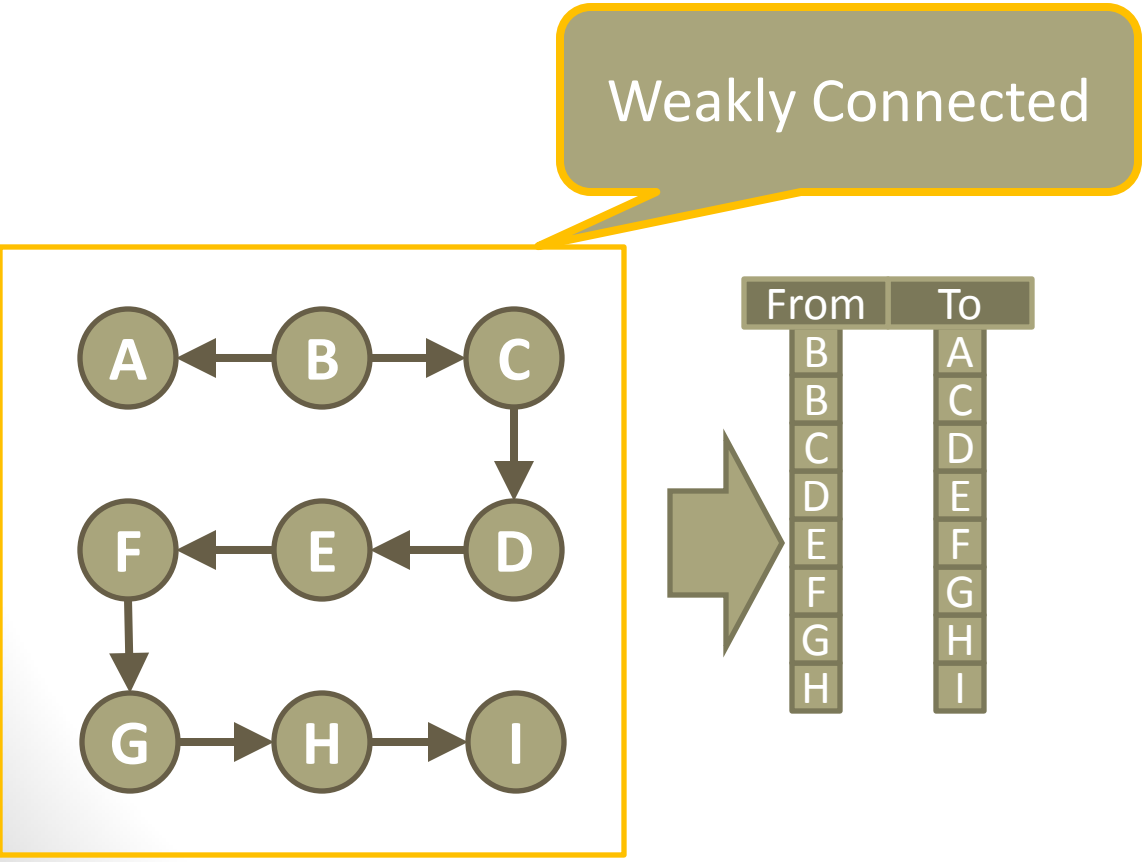
Weakly Connected



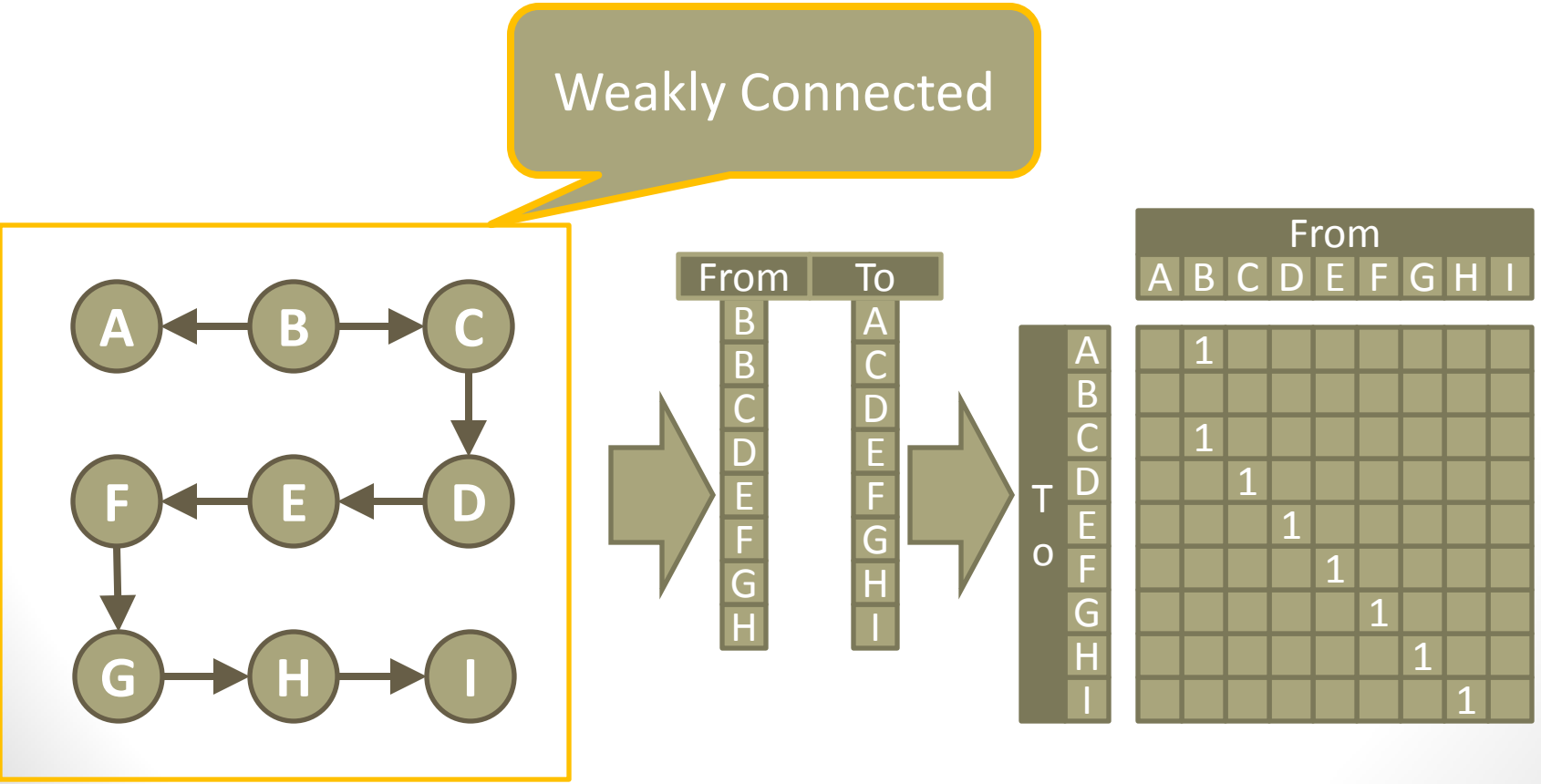
Weakly Connected:
For every pair of vertices, a path exists along the graph's edges regardless of the edge's direction. There exists at least one pair of vertices (u, v), where a path does not exist along the graph's directed edges from u to v an does not exist from v to u.

In this case: A path does not exist from A to C and does not exist from C to A.

Graph Data: Connectedness and Density (7)

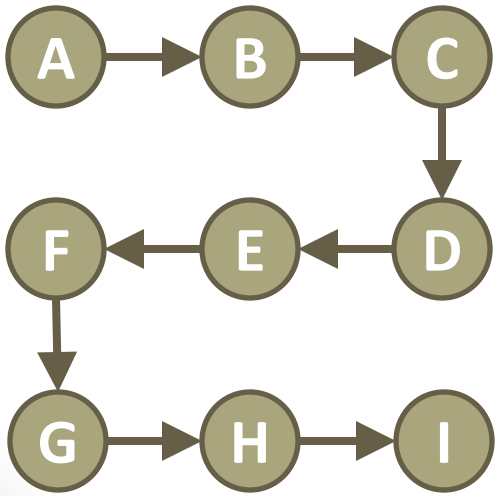


Graph Data: Connectedness and Density (8)



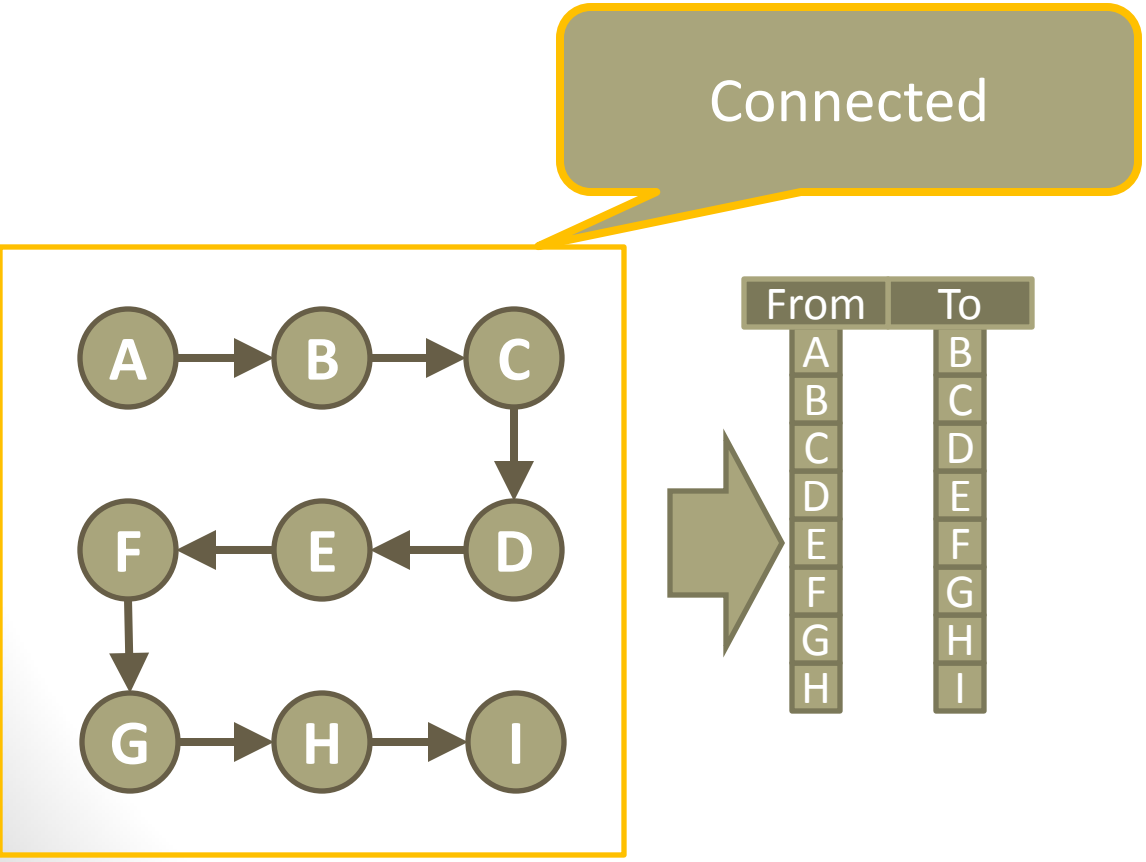
Graph Data: Connectedness and Density (9)

Connected

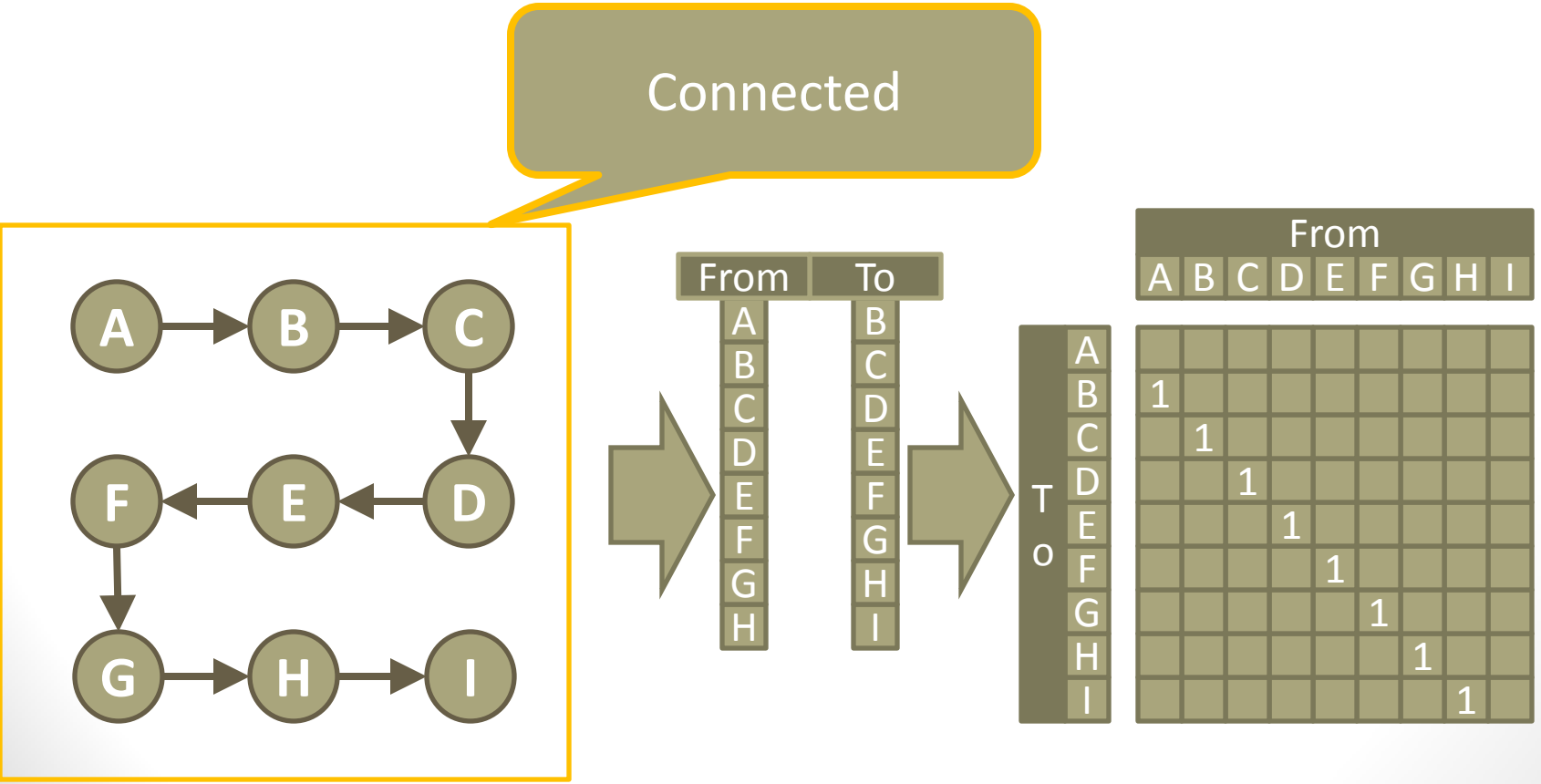


Connected:
For every pair of vertices (u, v), a path exists along the graph's directed edges from u to v or from v to u. There exists at least one pair of vertices (u, v), where a path does not exist along the graph's directed edges from either u to v or from v to u.

Graph Data: Connectedness and Density (10)

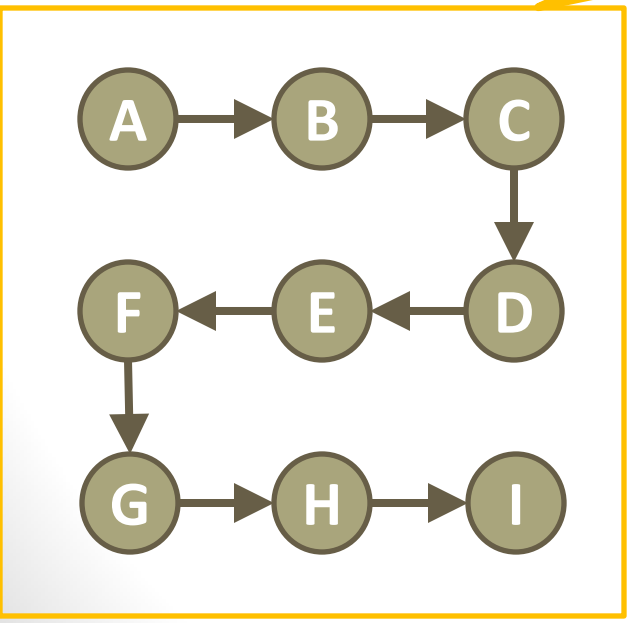


Graph Data: Connectedness and Density (11)



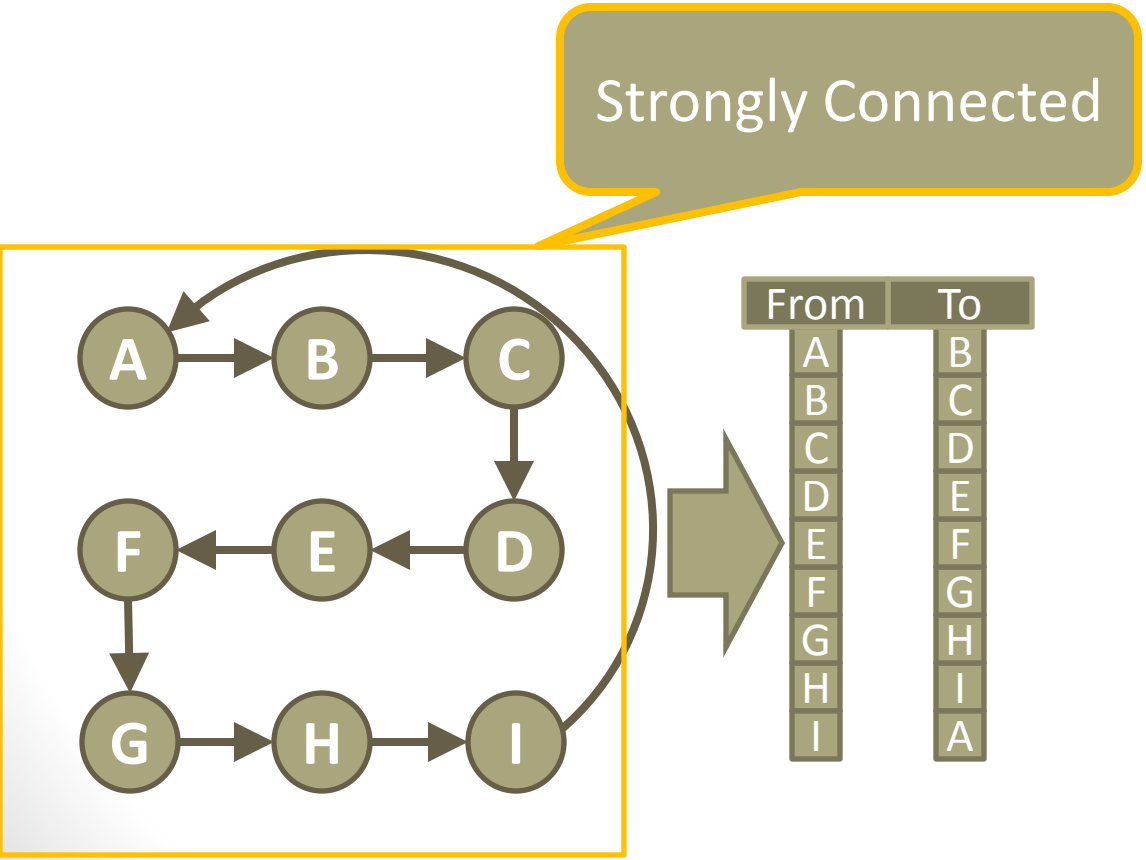
Graph Data: Connectedness and Density (12)

Connected

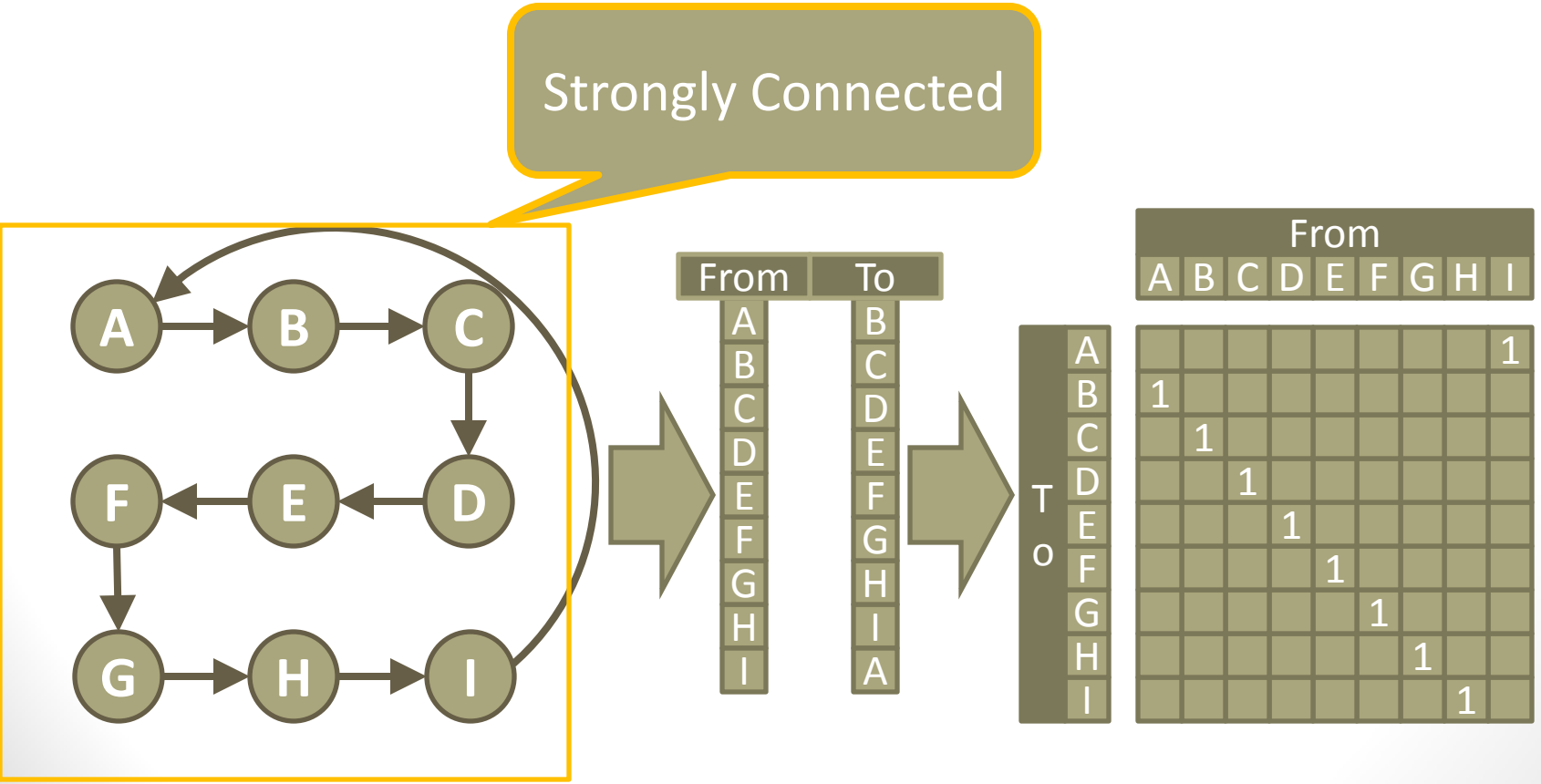


Connected:
A path exists along the graph's directed edges from every node to every node

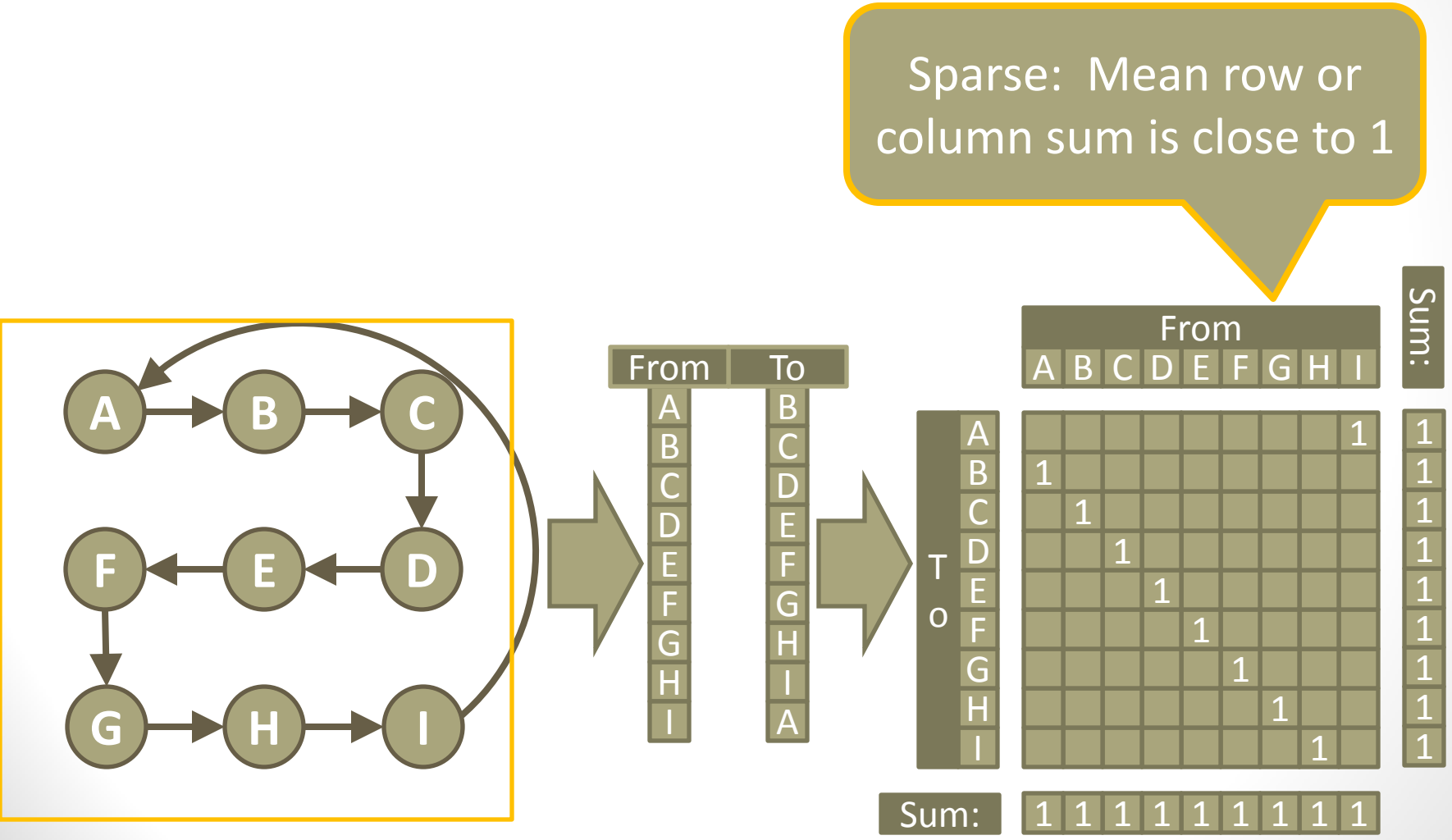
Graph Data: Connectedness and Density (13)



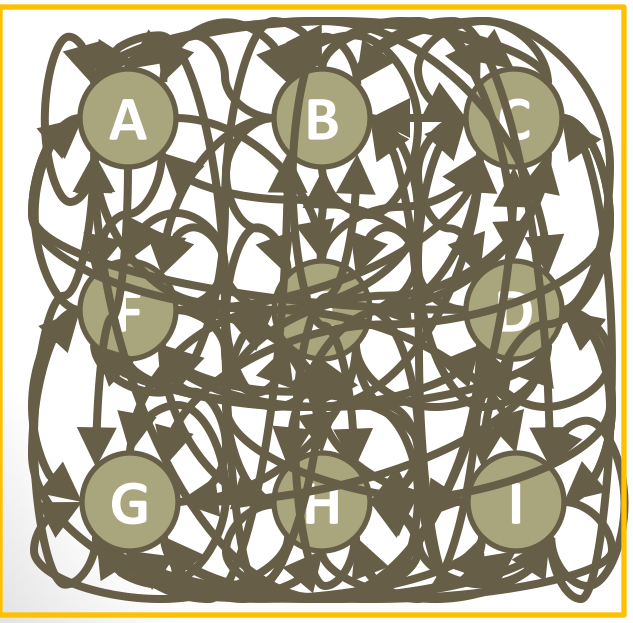
Graph Data: Connectedness and Density (14)



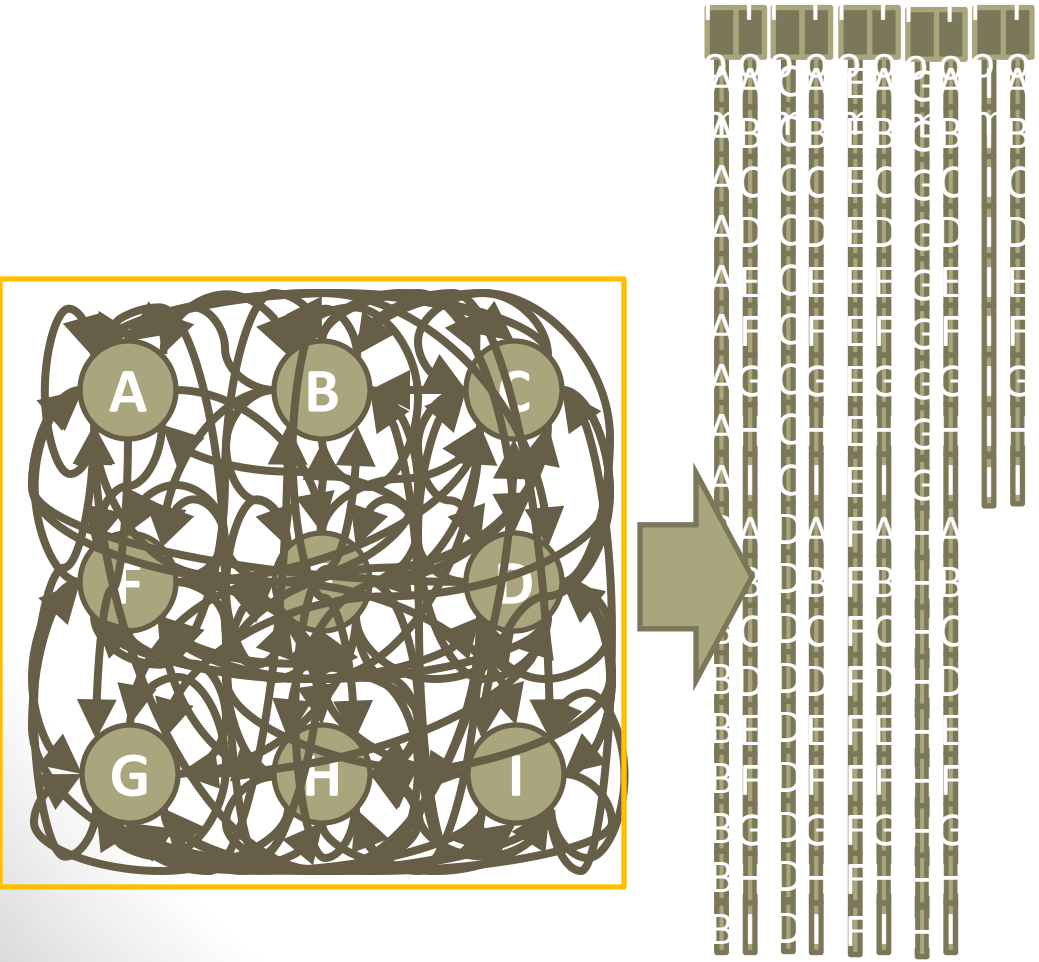
Graph Data: Connectedness and Density (15)



Graph Data: Connectedness and Density (16)



Graph Data: Connectedness and Density (17)



Graph Data: Connectedness and Density (18)

From	To
A	A
A	B
A	C
A	D
A	E
A	F
A	G
A	H
A	I
B	A
B	B
B	C
B	D
B	E
B	F
B	G
B	H
B	I

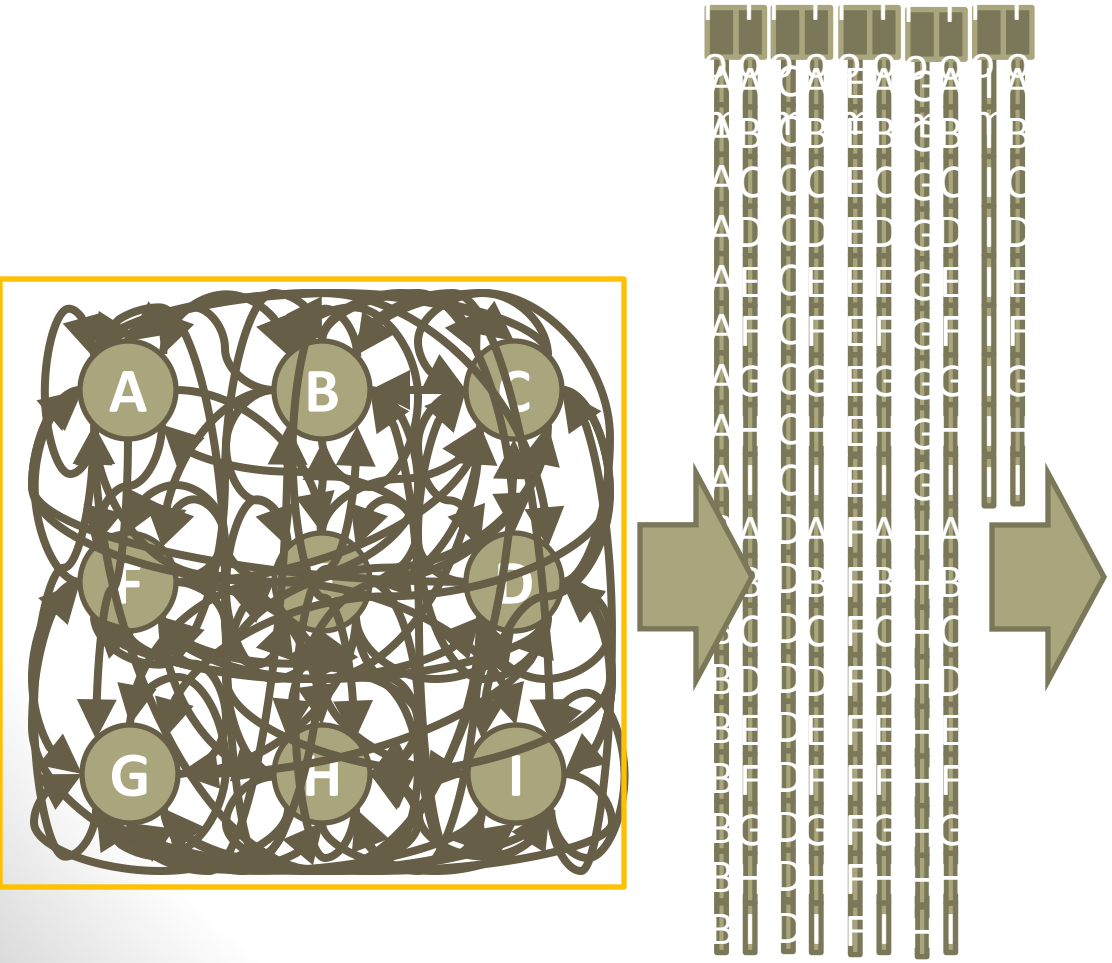
From	To
C	A
C	B
C	C
C	D
C	E
C	F
C	G
C	H
C	I
D	A
D	B
D	C
D	D
D	E
D	F
D	G
D	H
D	I

From	To
E	A
E	B
E	C
E	D
E	E
E	F
E	G
E	H
E	I
F	A
F	B
F	C
F	D
F	E
F	F
F	G
F	H
F	I

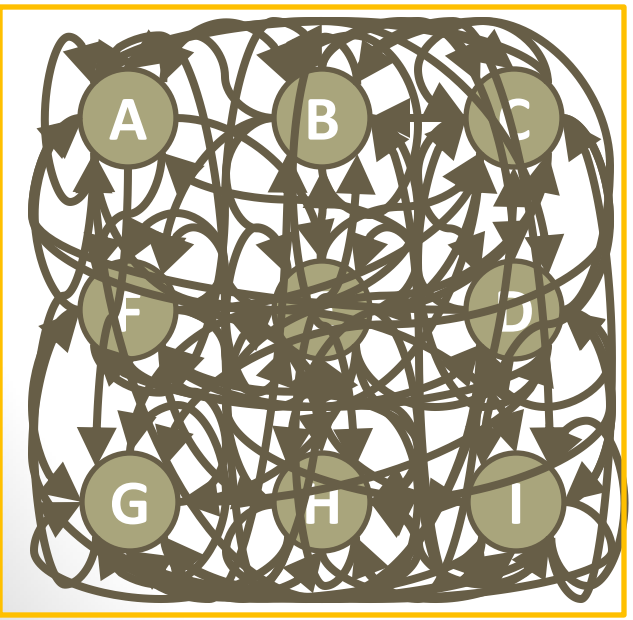
From	To
G	A
G	B
G	C
G	D
G	E
G	F
G	G
G	H
G	I
H	A
H	B
H	C
H	D
H	E
H	F
H	G
H	H
H	I

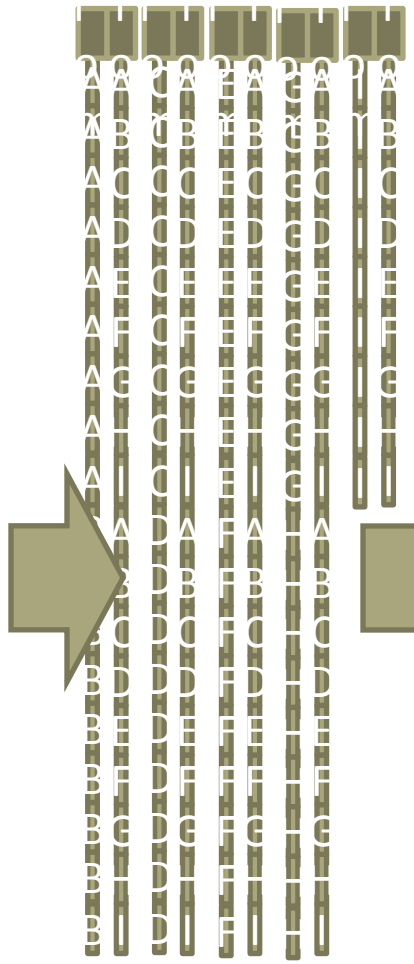
From	To
I	A
I	B
I	C
I	D
I	E
I	F
I	G
I	H
I	I

Graph Data: Connectedness and Density (19)



Graph Data: Connectedness and Density (20)

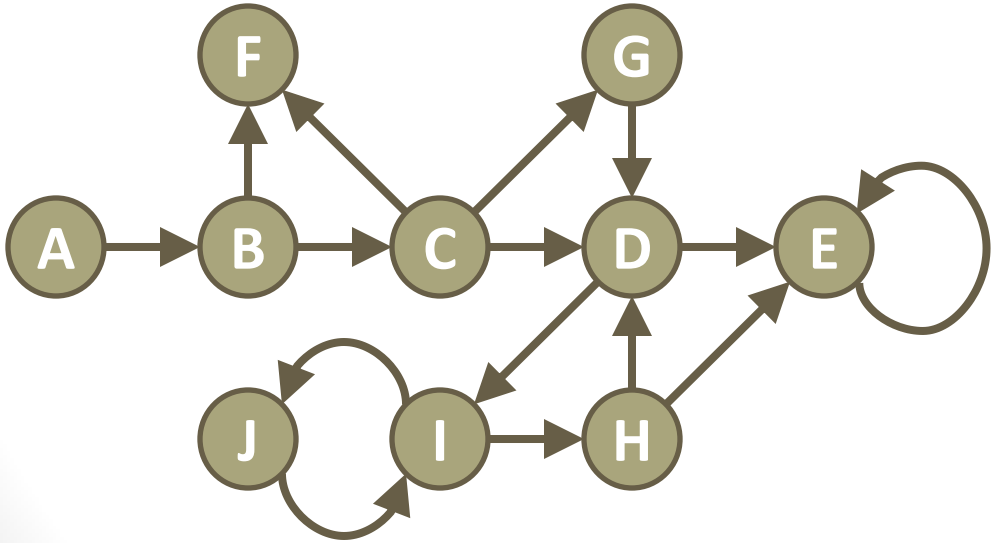




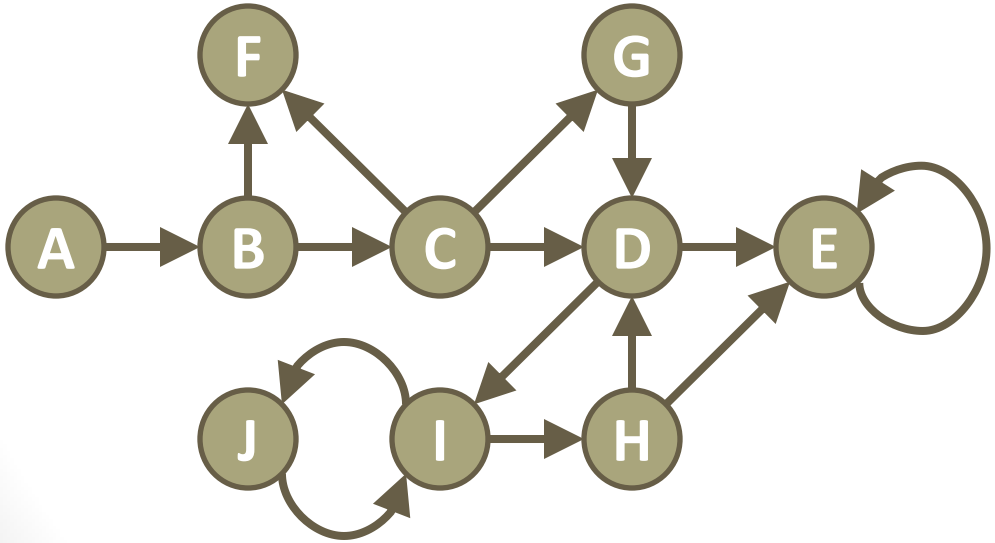
Dense: Mean row or column sum is close to number of nodes

To \ From	A	B	C	D	E	F	G	H	I	Sum:
A	1	1	1	1	1	1	1	1	1	9
B	1	1	1	1	1	1	1	1	1	9
C	1	1	1	1	1	1	1	1	1	9
D	1	1	1	1	1	1	1	1	1	9
E	1	1	1	1	1	1	1	1	1	9
F	1	1	1	1	1	1	1	1	1	9
G	1	1	1	1	1	1	1	1	1	9
H	1	1	1	1	1	1	1	1	1	9
I	1	1	1	1	1	1	1	1	1	9
Sum:	9	9	9	9	9	9	9	9	9	

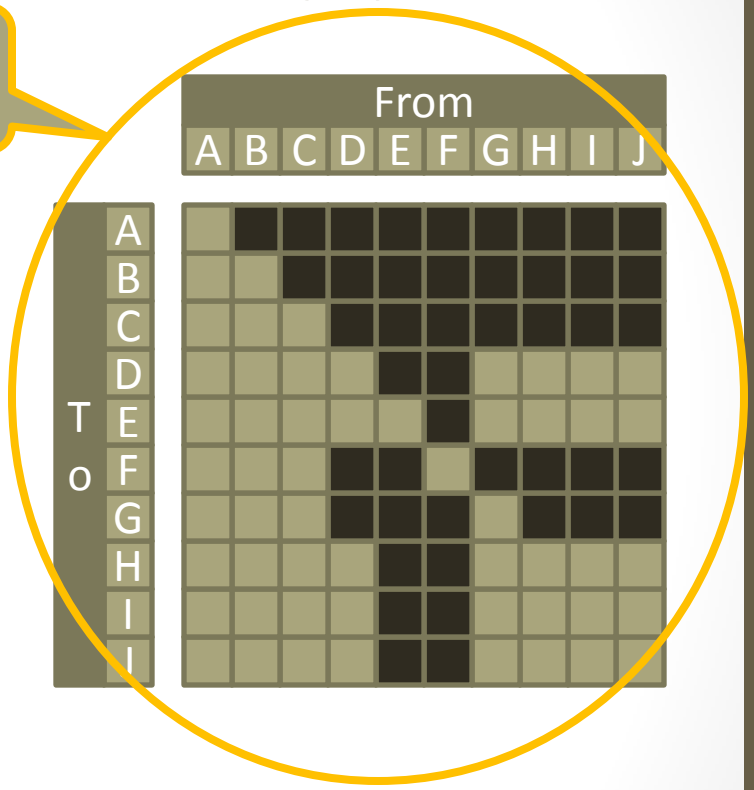
Graph Data: Allowed Paths, Distance, and Diameter (0)



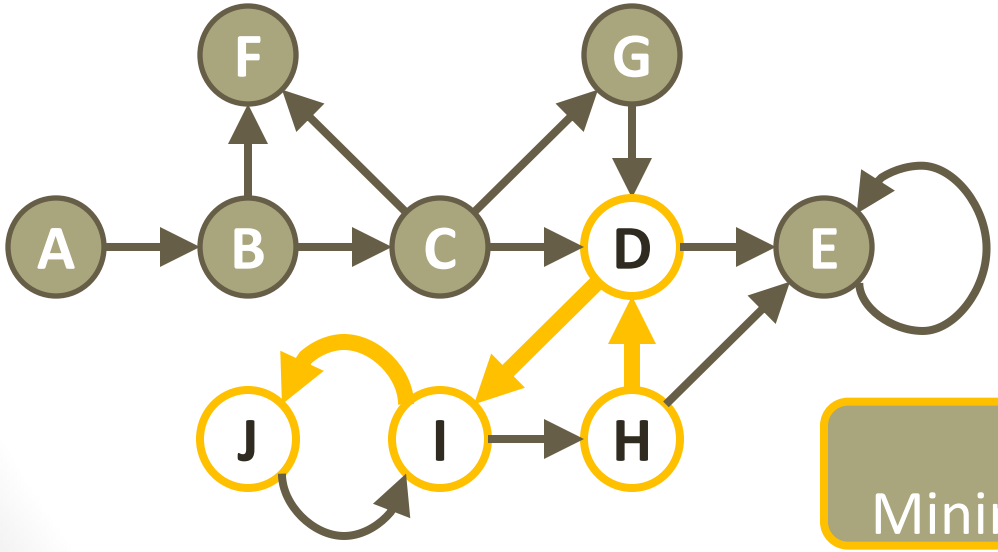
Graph Data: Allowed Paths, Distance, and Diameter (1)

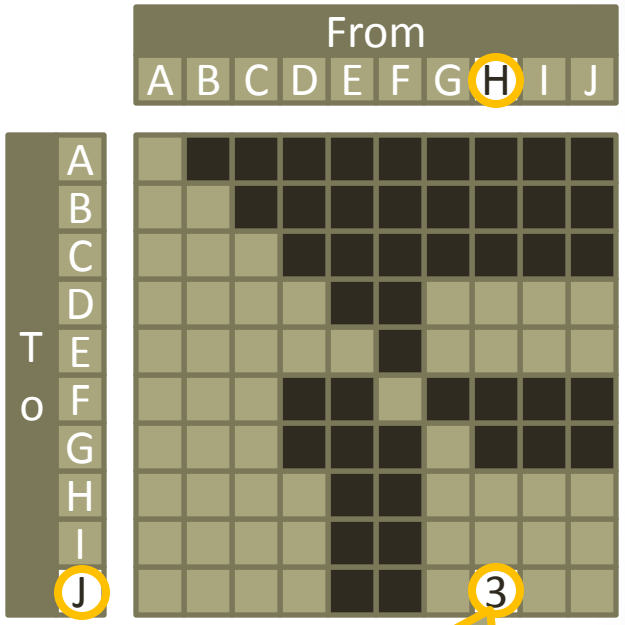


Allowed Paths



Graph Data: Allowed Paths, Distance, and Diameter (2)

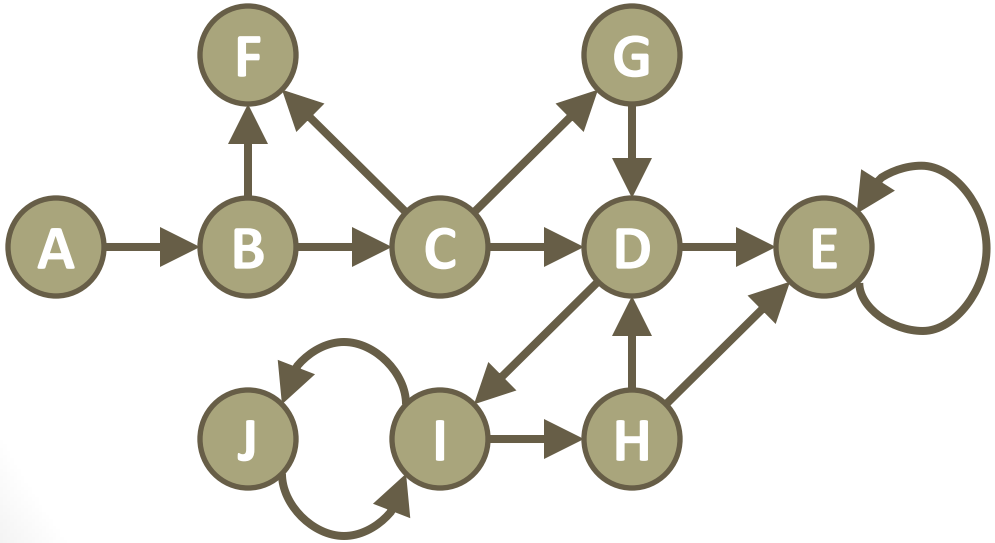




Path Distance:
Minimum Distance from H to J

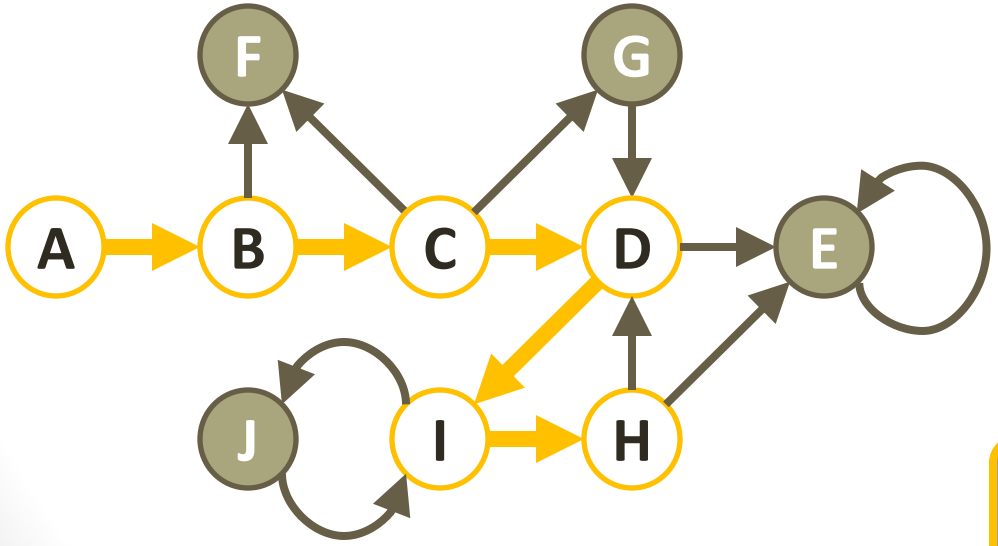
Graph Data: Allowed Paths, Distance, and Diameter (3)

Distance Matrix:
Minimum distance between any two vertices along a directed edge



To \ From	A	B	C	D	E	F	G	H	I	J
A	0									
B	1	0								
C	2	1	0							
D	3	2	1	0			1	1	2	3
E	4	3	2	1	0		2	1	2	3
F	2	1	1			0				
G	4	2	1				0			
H	5	4	3	2			3	0	1	2
I	4	3	2	1			2	2	0	1
J	5	4	3	2			3	3	1	0

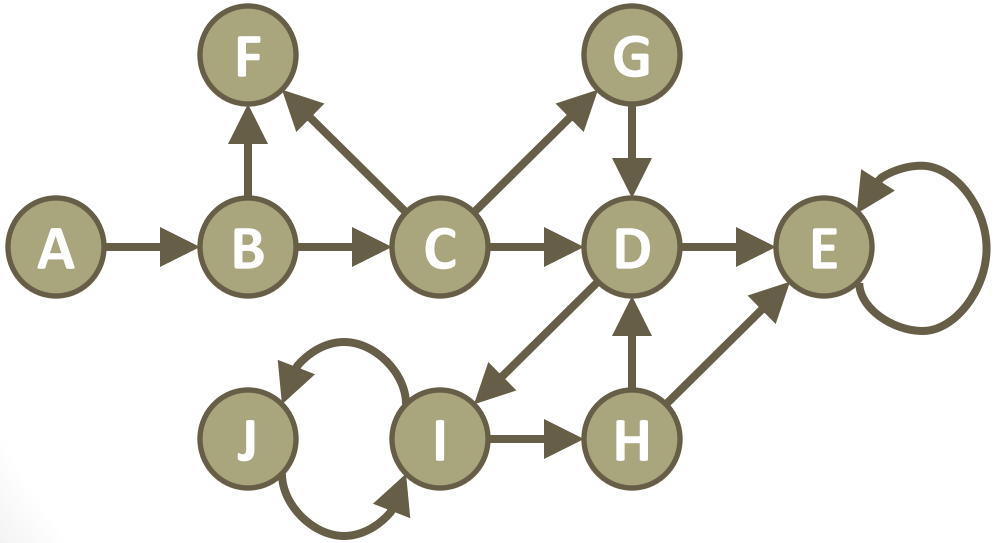
Graph Data: Allowed Paths, Distance, and Diameter (4)



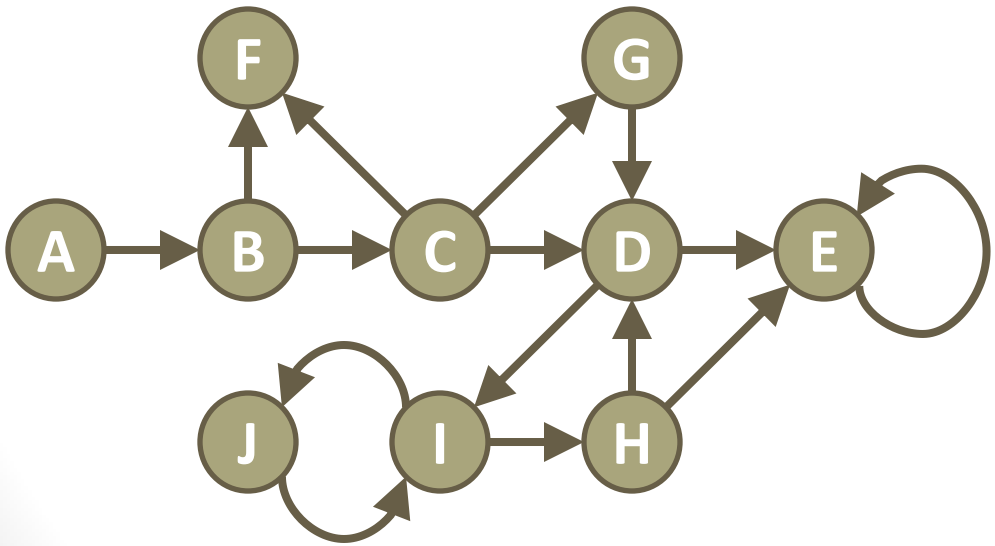
To \ From	A	B	C	D	E	F	G	H	I	J
A	0									
B	1	0								
C	2	1	0							
D	3	2	1	0			1	1	2	3
E	4	3	2	1	0		2	1	2	3
F	2	1	1			0				
G	4	2	1				0			
H	5	4	3	2			3	0	1	2
I	4	3	2	1			2	2	0	1
J	5	4	3	2			3	3	1	0

Diameter:
Max Path Distance

Graph Data: Order and Size (0)



Graph Data: Order and Size (1)

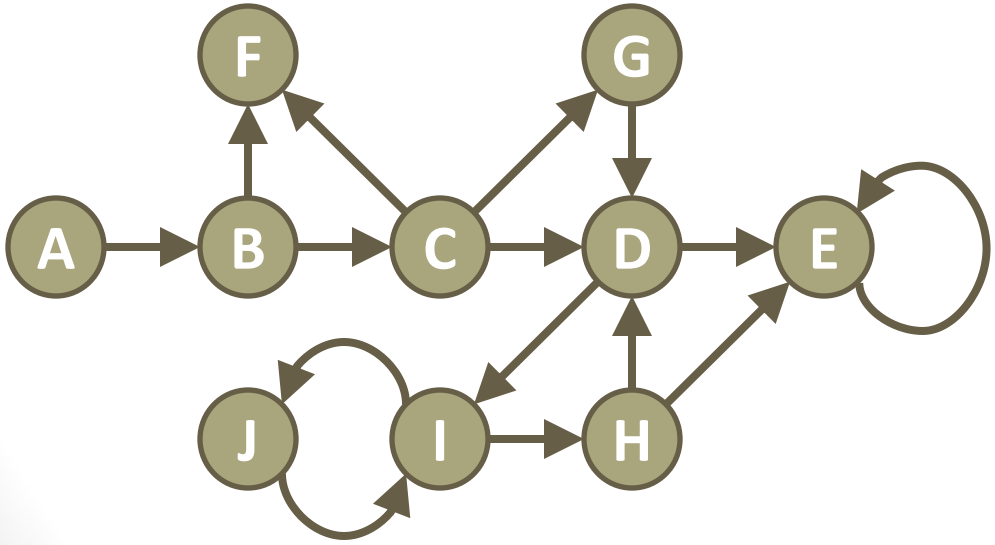


Matrix of Edges

To \ From	A	B	C	D	E	F	G	H	I	J
A										
B	1									
C		1								
D			1				1	1		
E				1	1			1		
F		1	1							
G			1							
H									1	
I				1						1
J									1	

Graph Data: Order and Size (2)

Order of Graph: 10

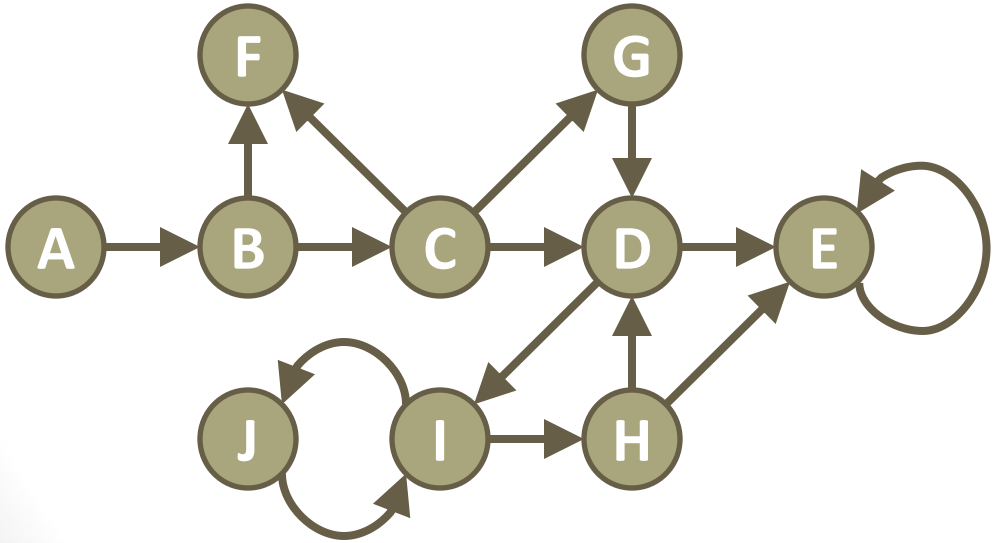


To \ From	A	B	C	D	E	F	G	H	I	J
A										
B	1									
C		1								
D			1				1	1		
E				1	1			1		
F		1	1							
G			1							
H									1	
I				1						1
J									1	

Order of Graph: Number of Nodes

Graph Data: Order and Size (3)

Order of Graph: 10
Size of Graph: 15



To \ From	A	B	C	D	E	F	G	H	I	J
A										
B	1									
C		1								
D			1				1	1		
E				1	1			1		
F		1	1							
G			1							
H									1	
I				1						1
J									1	

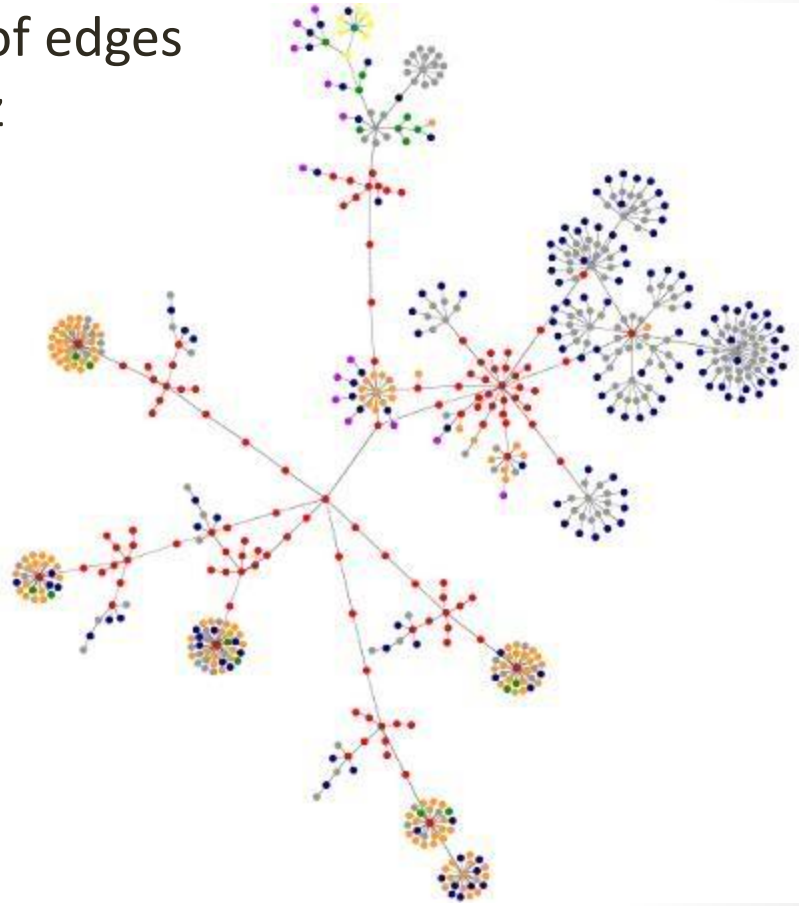
Size of Graph:
Number of Edges

Graph Data: Links (0)

- Graphs In General:
  - http://en.wikipedia.org/wiki/Graph_theory
  - https://en.wikipedia.org/wiki/Graph_(discrete_mathematics)
  - https://en.wikipedia.org/wiki/Graph_(abstract_data_type)
  - https://en.wikipedia.org/wiki/List_of_graph_theory_topics
- Graph Diameter
  - http://people.hofstra.edu/geotrans/eng/methods/diameter1.html
- Distance
  - http://en.wikipedia.org/wiki/Distance_(graph_theory)
- Resistance
  - http://en.wikipedia.org/wiki/Resistance_distance
- Centrality
  - http://en.wikipedia.org/wiki/Betweenness#Betweenness_centrality
- Connectivity
  - http://en.wikipedia.org/wiki/Connectivity_(graph_theory)
- Hypergraph: One edge can connect may nodes
  - http://en.wikipedia.org/wiki/Hypergraph
- Degree Distribution: Histogram of in-degrees, out-degrees, in-degrees normalized by out-degrees
- Popularity (Google Matrix)
- Density: #edges/(#nodes^2-#nodes)
- Clustering Coefficient
  - http://en.wikipedia.org/wiki/Clustering_coefficient

Quiz Graph Data (Measures)

- The presented slide contains a list of edges that describes the graph in the quiz



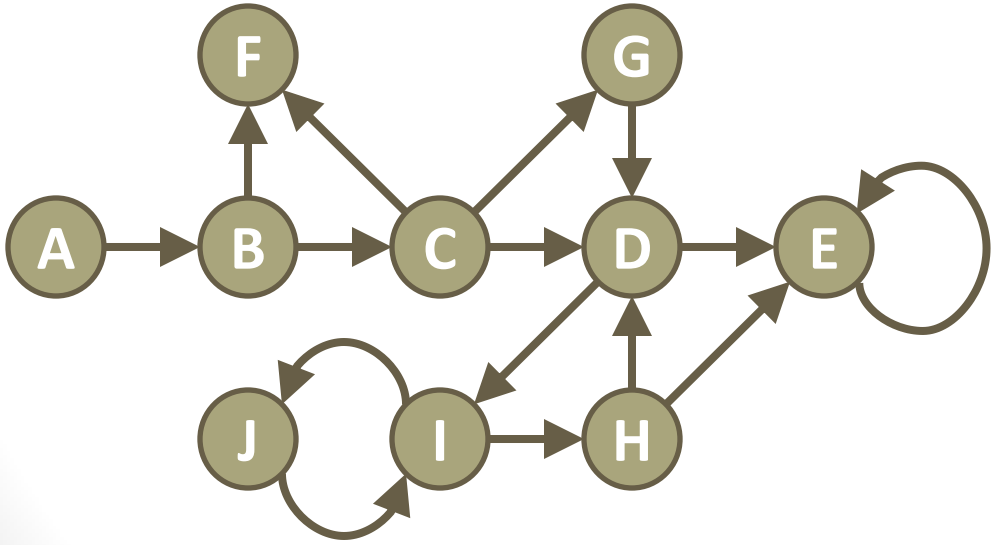
Introduction to Graph Data

Graph Data: Google Matrix

Google Matrix

- http://en.wikipedia.org/wiki/Google_matrix

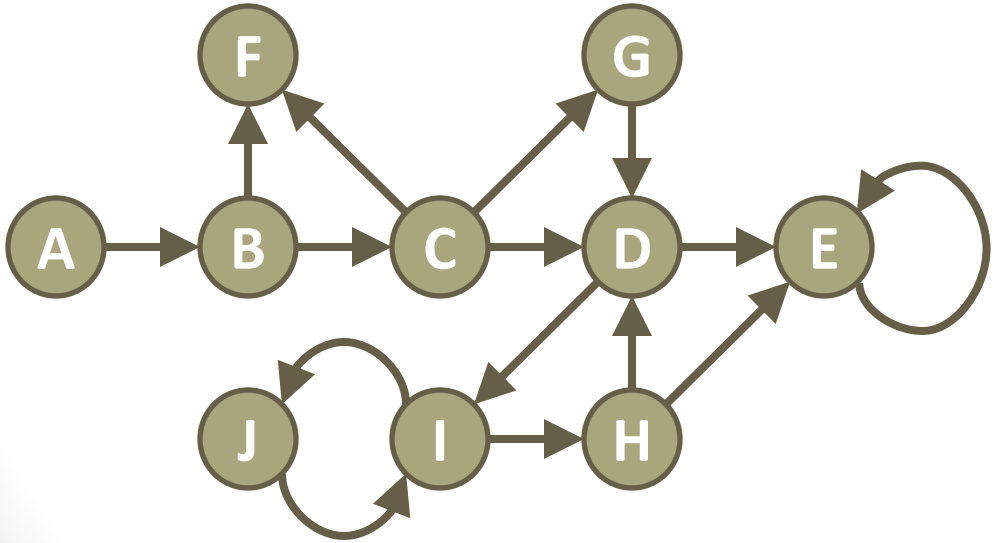
Graph Data: Popularity (0)



To \ From	A	B	C	D	E	F	G	H	I	J
A										
B	1									
C		1								
D			1				1	1		
E				1	1			1		
F		1	1							
G			1							
H									1	
I				1						1
J									1	

Graph Data: Popularity (1)

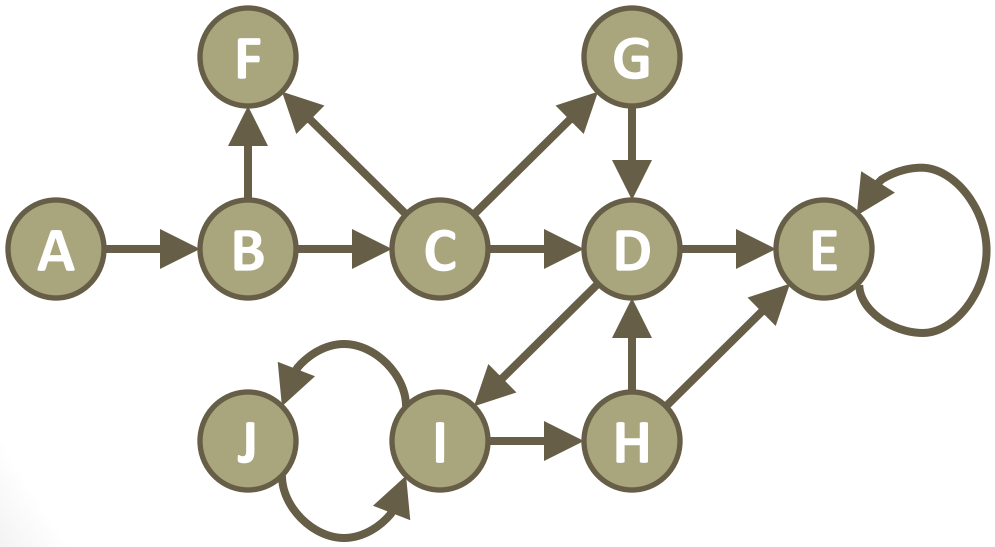
In-degree of a node: Popularity



To \ From	A	B	C	D	E	F	G	H	I	J	Sum:
A											0
B	1										1
C		1									1
D			1				1	1			3
E				1	1			1			3
F		1	1								2
G			1								1
H									1		1
I				1						1	2
J									1		1

Graph Data: Popularity (2)

In-degree of a node: Popularity



To \ From	A	B	C	D	E	F	G	H	I	J	Sum:
A											0
B	1										1
C		1									1
D			1				1	1			3
E				1	1			1			3
F		1	1								2
G			1								1
H									1		1
I				1						1	2
J									1		1
Sum:	1	2	3	2	1	0	1	2	2	1	

Out-degree of a node: Gregariousness

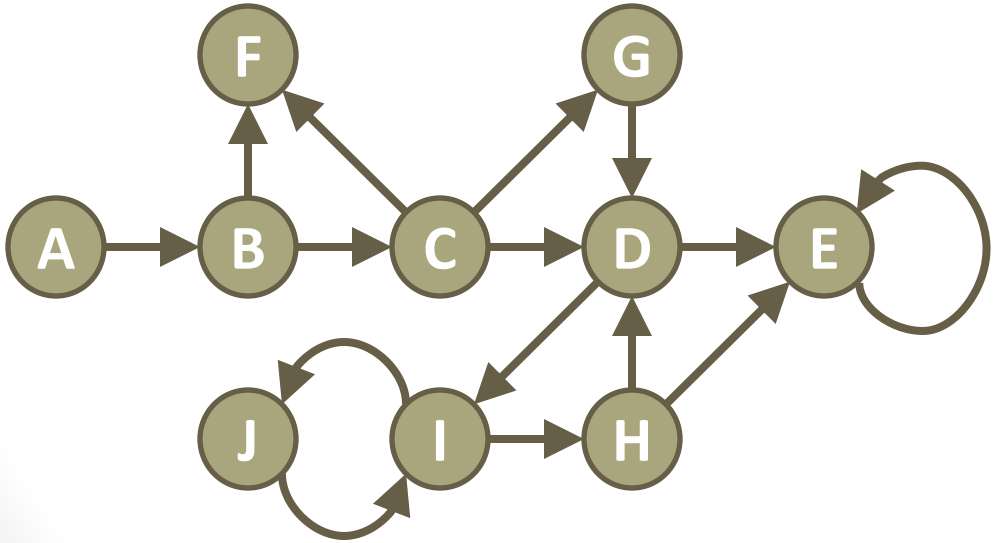
Quiz Graph Data (Popularity)

- Graph Data (Popularity)

- You need to view the projected slide to answer the questions

Graph Data: Popularity (3)

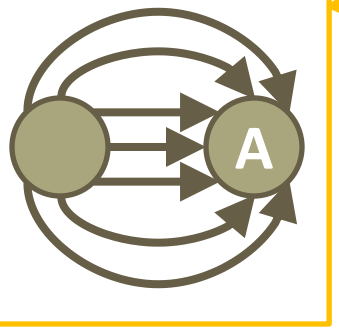
But, not every link is equally important. Otherwise, you could create a website with a thousand links that all point to your site.

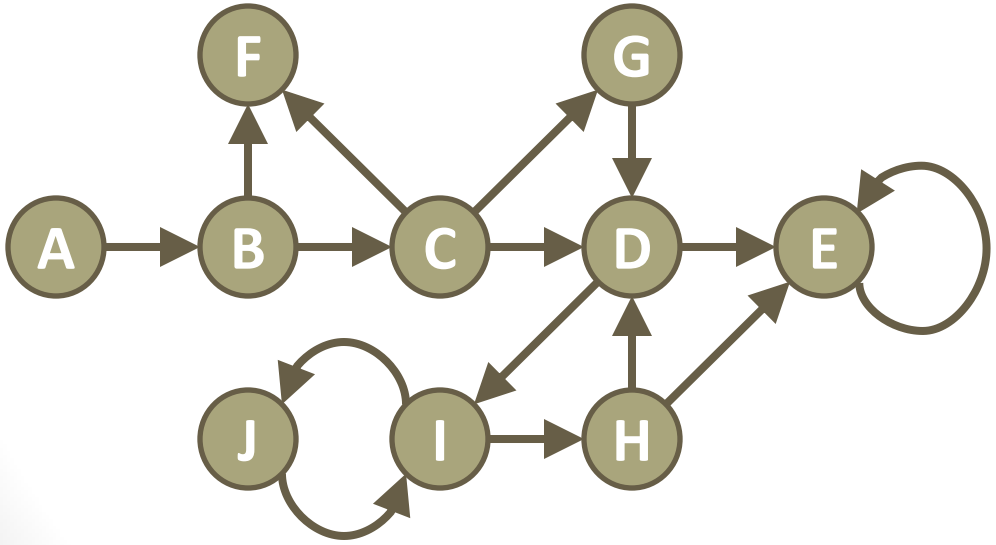


To \ From	A	B	C	D	E	F	G	H	I	J	Sum:
A											0
B	1										1
C		1									1
D			1				1	1			3
E				1	1			1			3
F		1	1								2
G			1								1
H									1		1
I				1						1	2
J									1		1
Sum:	1	2	3	2	1	0	1	2	2	1	

Graph Data: Popularity (4)

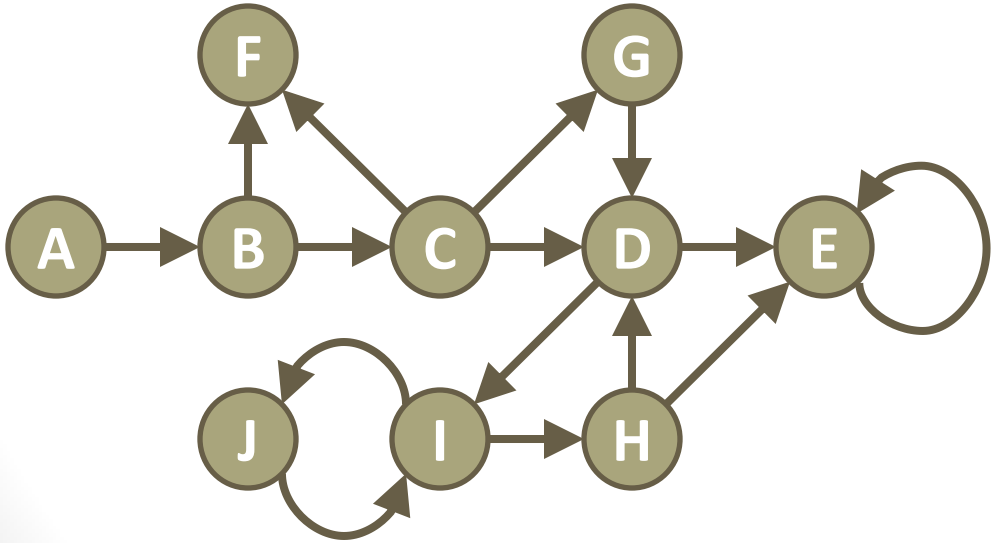
One very gregarious node could make another node very popular





To \ From	A	B	C	D	E	F	G	H	I	J	Sum:
A											0
B	1										1
C		1									1
D			1				1	1			3
E				1	1			1			3
F		1	1								2
G			1								1
H									1		1
I				1						1	2
J									1		1
Sum:	1	2	3	2	1	0	1	2	2	1	

Graph Data: Popularity (5)

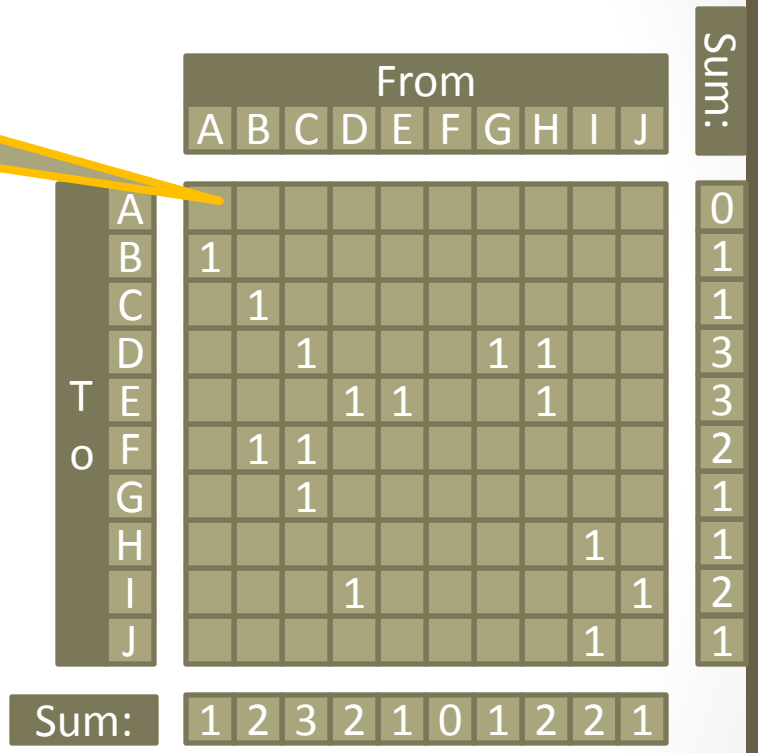


To \ From	A	B	C	D	E	F	G	H	I	J	Sum:
A											0
B	1										1
C		1									1
D			1				1	1			3
E				1	1			1			3
F		1	1								2
G			1								1
H									1		1
I				1						1	2
J									1		1
Sum:	1	2	3	2	1	0	1	2	2	1	

The amount of popularity that a single node can send out needs to be tempered by Gregariousness

Graph Data: Popularity (6)

This matrix needs some adjustments



Graph Data: Popularity (7)

To \ From	A	B	C	D	E	F	G	H	I	J	sum:
A											0
B	1										1
C		1									1
D			1				1	1			3
E				1	1			1			3
F		1	1								2
G			1								1
H									1		1
I				1						1	2
J									1		1
sum:	1	2	3	2	1	0	1	2	2	1	

Graph Data: Popularity (8)

To \ From	A	B	C	D	E	F	G	H	I	J	sum:
A											0
B	1										1.00
C		1/2									0.50
D			1/3				1	1/2			1.83
E				1/2	1			1/2			2.00
F		1/2	1/3								0.83
G			1/3								0.33
H									1/2		0.50
I				1/2						1	1.50
J									1/2		0.50
sum:	1	1	1	1	1	0	1	1	1	1	

Graph Data: Popularity (9)

To \ From	A	B	C	D	E	F	G	H	I	J	sum:	Popularity:
A											0	0
B	1										1.00	0.111
C		1/2									0.50	0.056
D			1/3				1	1/2			1.83	0.204
E				1/2	1			1/2			2.00	0.222
F		1/2	1/3								0.83	0.092
G			1/3								0.33	0.037
H									1/2		0.50	0.056
I				1/2						1	1.50	0.167
J									1/2		0.50	0.056
sum:	1	1	1	1	1	0	1	1	1	1		

Graph Data: Popularity (10)

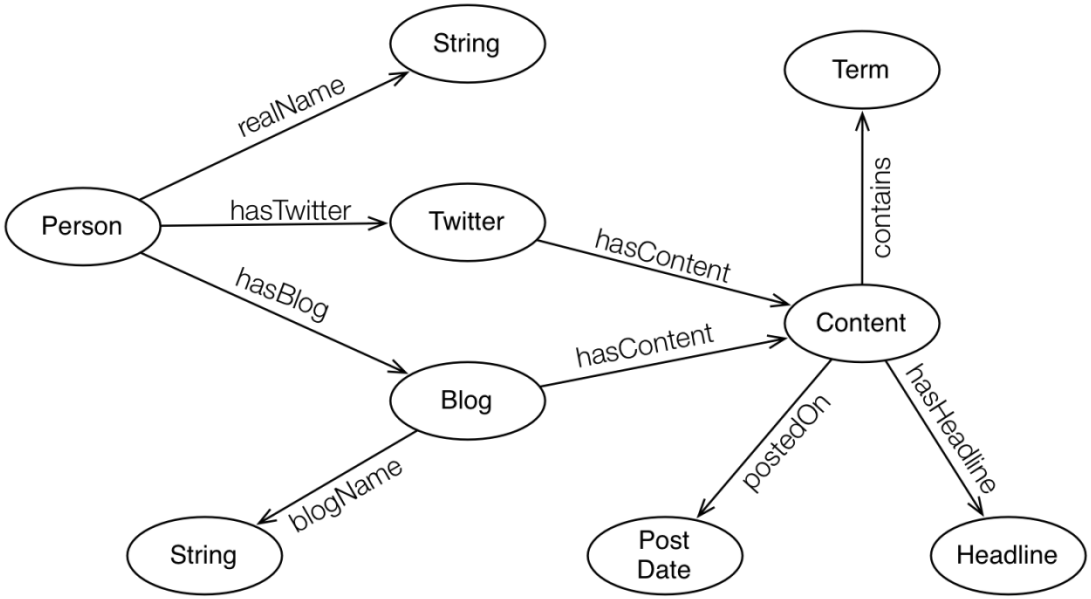
- Rules of a Popularity Network
  - Rule 0: Popularity transmits from one node to a downstream connected node.
  - Rule 1: Popularity transmissions from multiple nodes to a single node are additive.
  - Rule 2: A more popular node transmits more popularity.
  - Rule 3: A node splits up its popularity among its recipients.

- Continue with **EigenVectorOfGraphMatrix.R**

- Solve (or Iterate until convergence)
  - Modified Popularity is the Popularity divided by Gregariousness (Out-Degree)
  - Popularity of a website is the sum of the Modified Popularities that point to the website.

Graph Data: Google Matrix

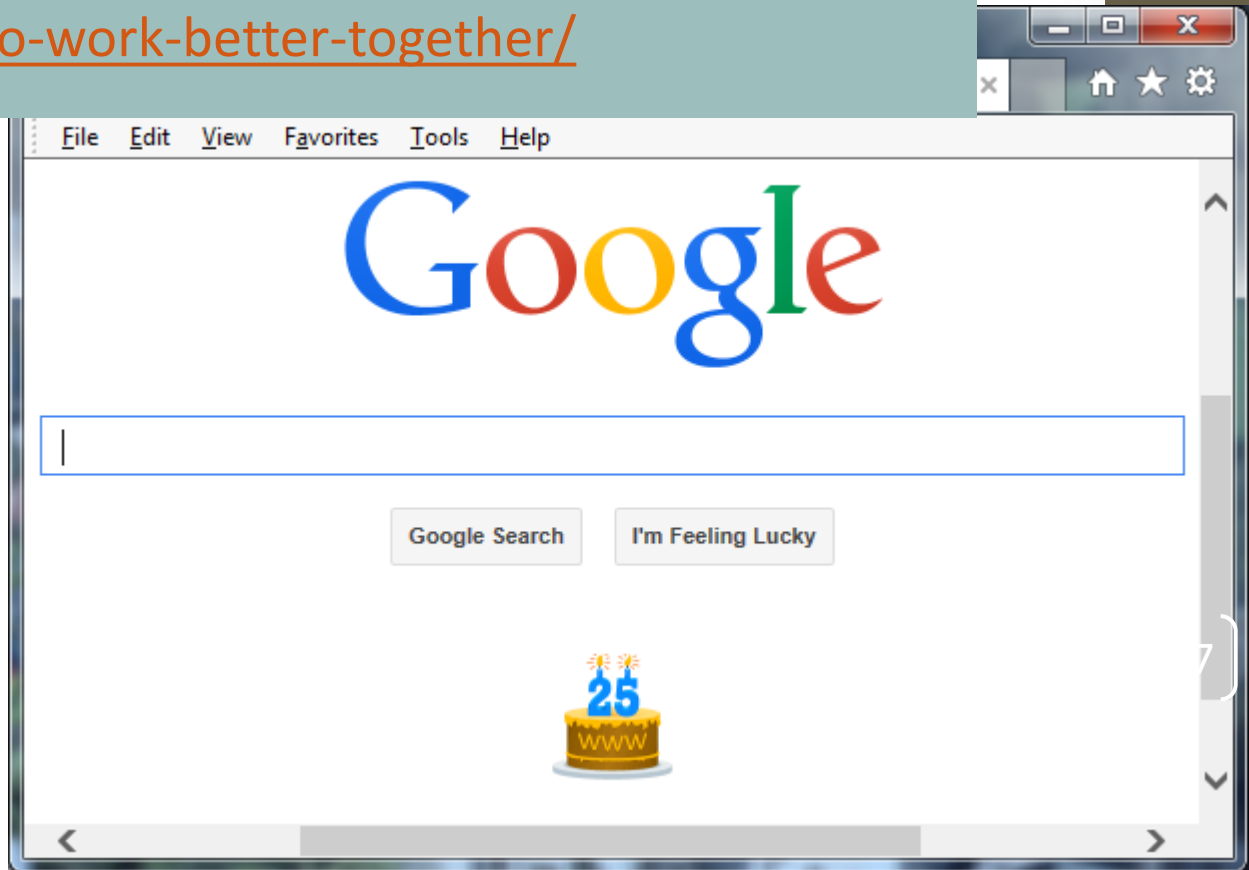
Break


String
realName
Term
Person
hasTwitter
Twitter
hasContent
contains
hasBlog
Content
hasContent
Blog
postedOn
hasHeadline
blogName
String
Post
Date
Headline

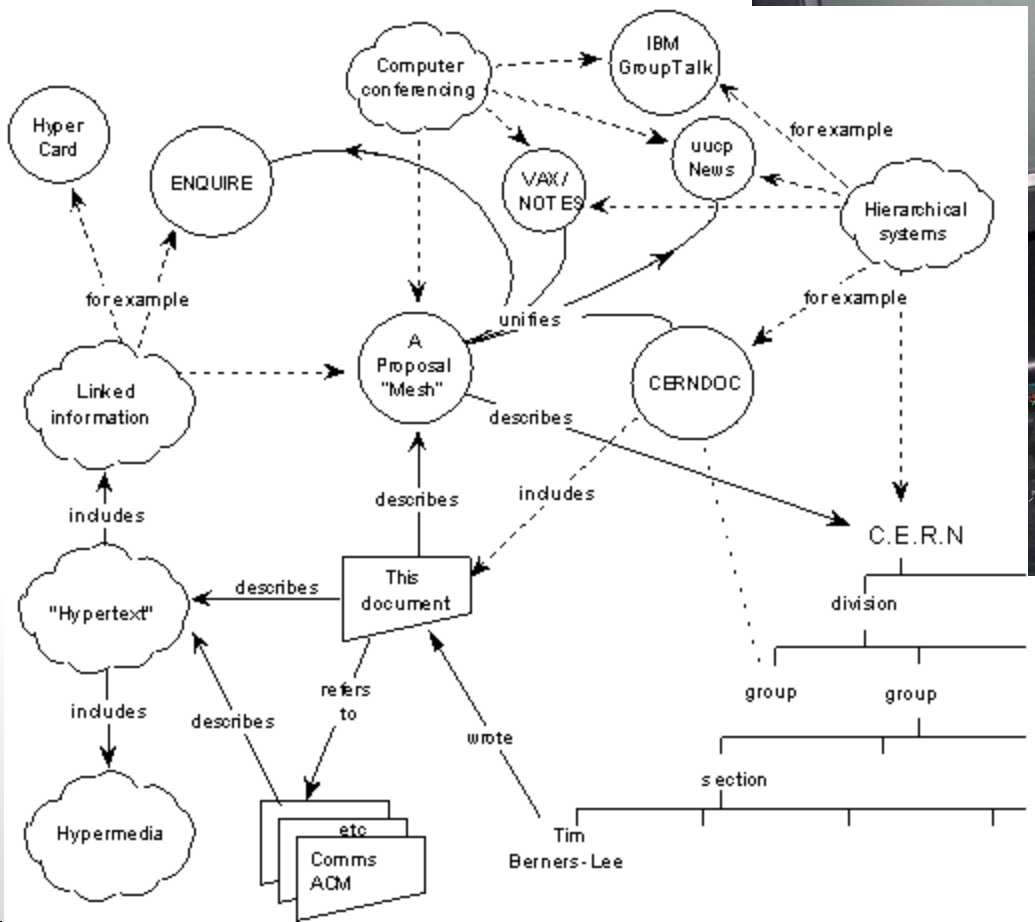
Introduction to SPARQL

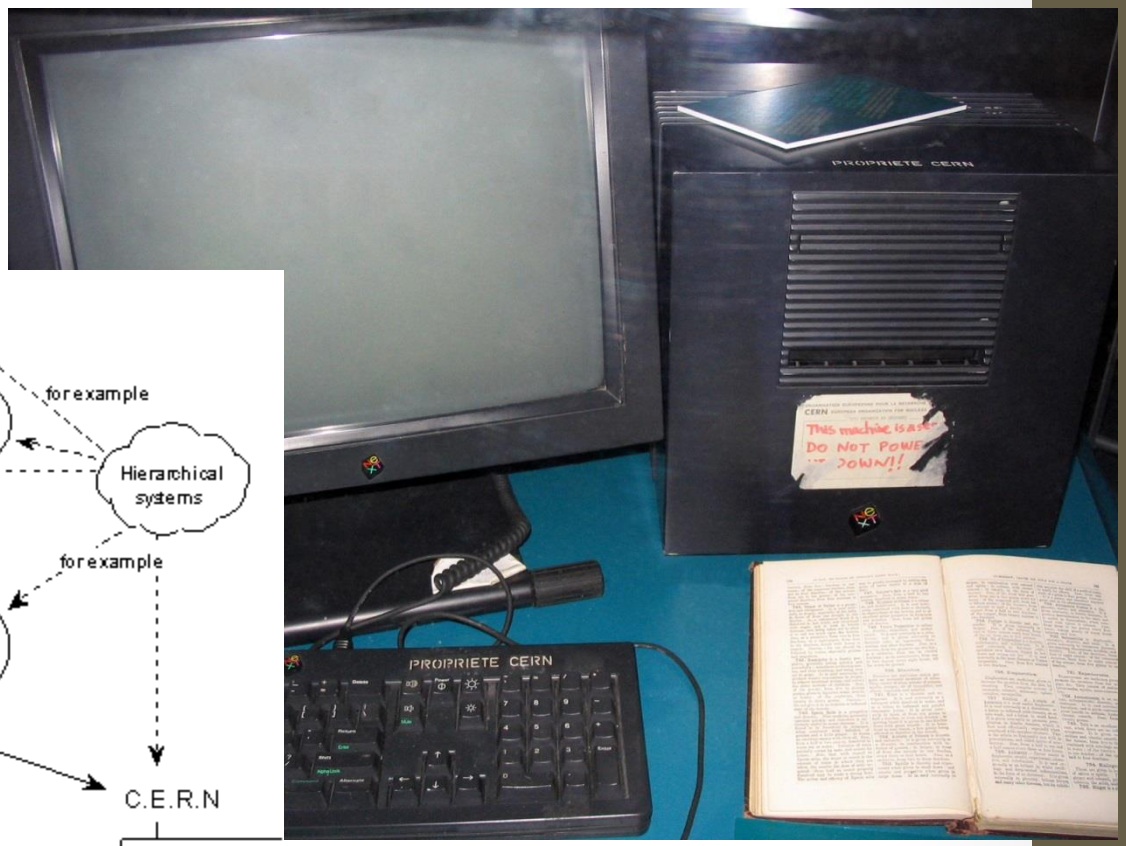
SPARQL

- ***S**PARQL **P**rotocol **a**nd **R**DF **Q**uery **L**anguage*
- "SPARQL will make a huge difference“ http://blog.semantic-web.at/2009/04/22/tim-berners-lee-we-need-data-on-the-web-to-work-better-together/



Tim Berners-Lee's Web Server



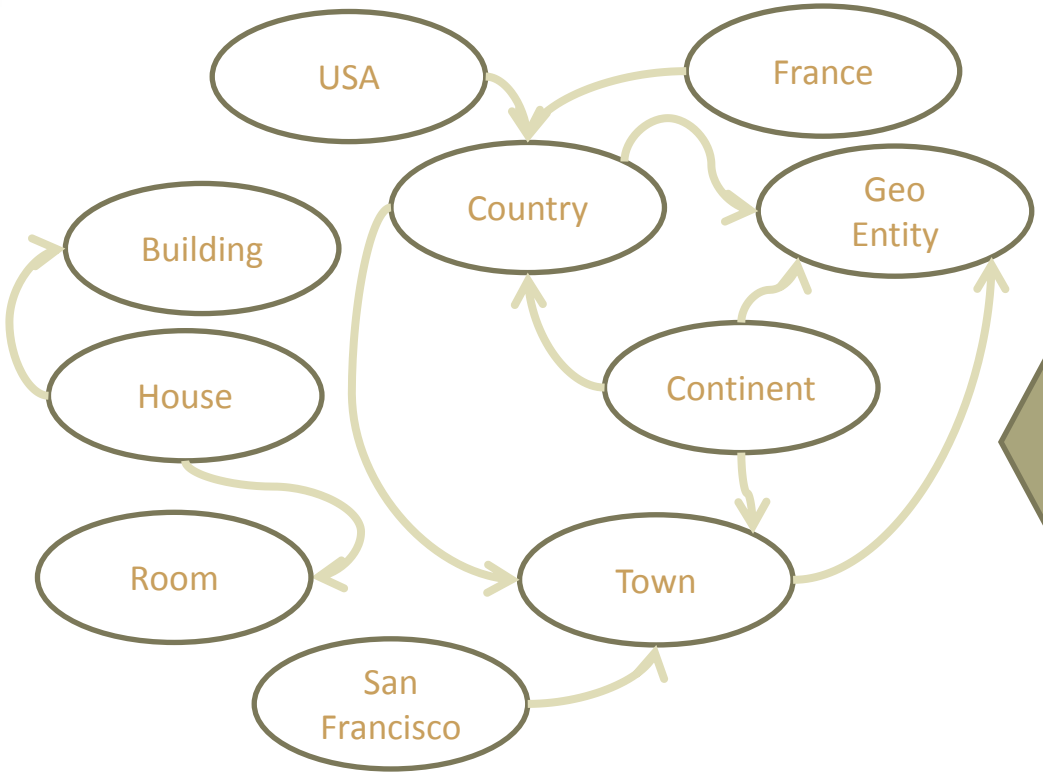


SPARQL: Triplestore

- http://en.wikipedia.org/wiki/Triplestore
- http://en.wikipedia.org/wiki/Sparql

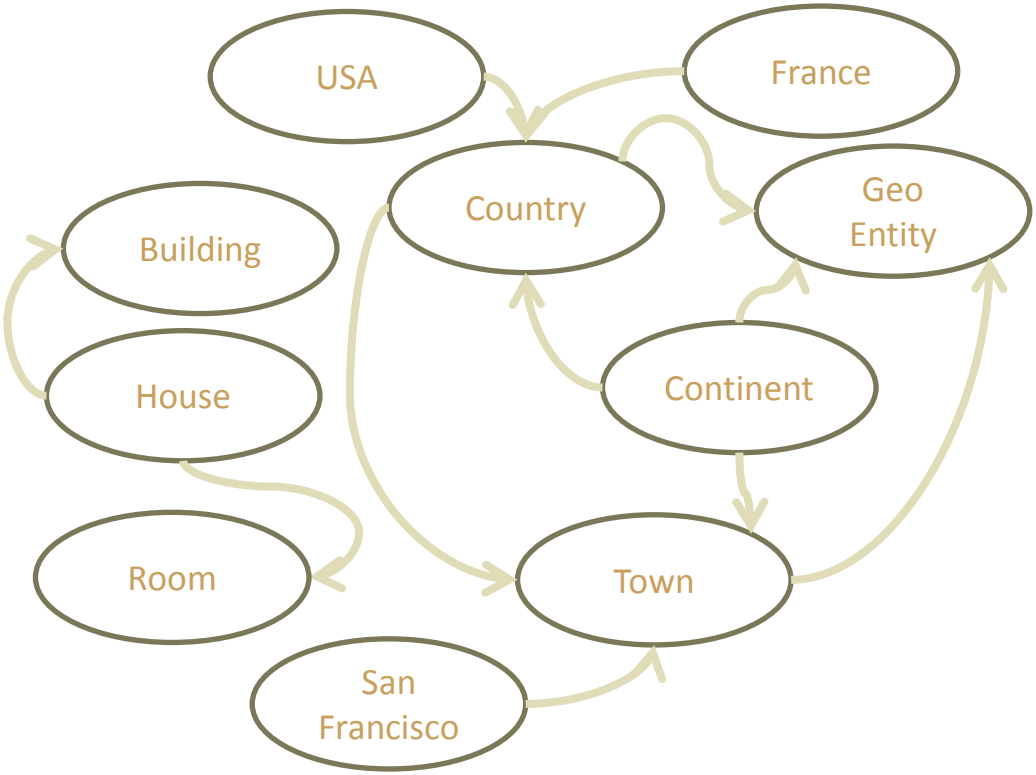
S	P	O
USA	is a	Country
France	is a	Country
Country	is a	Geo Entity
Country	contains	Town
Continent	contains	Country
Continent	is a	Geo Entity
Continent	contains	Town
Town	is a	Geo Entity
San Francisco	is a	Town
House	contains	Room
House	is a	Building

SPARQL: Graph

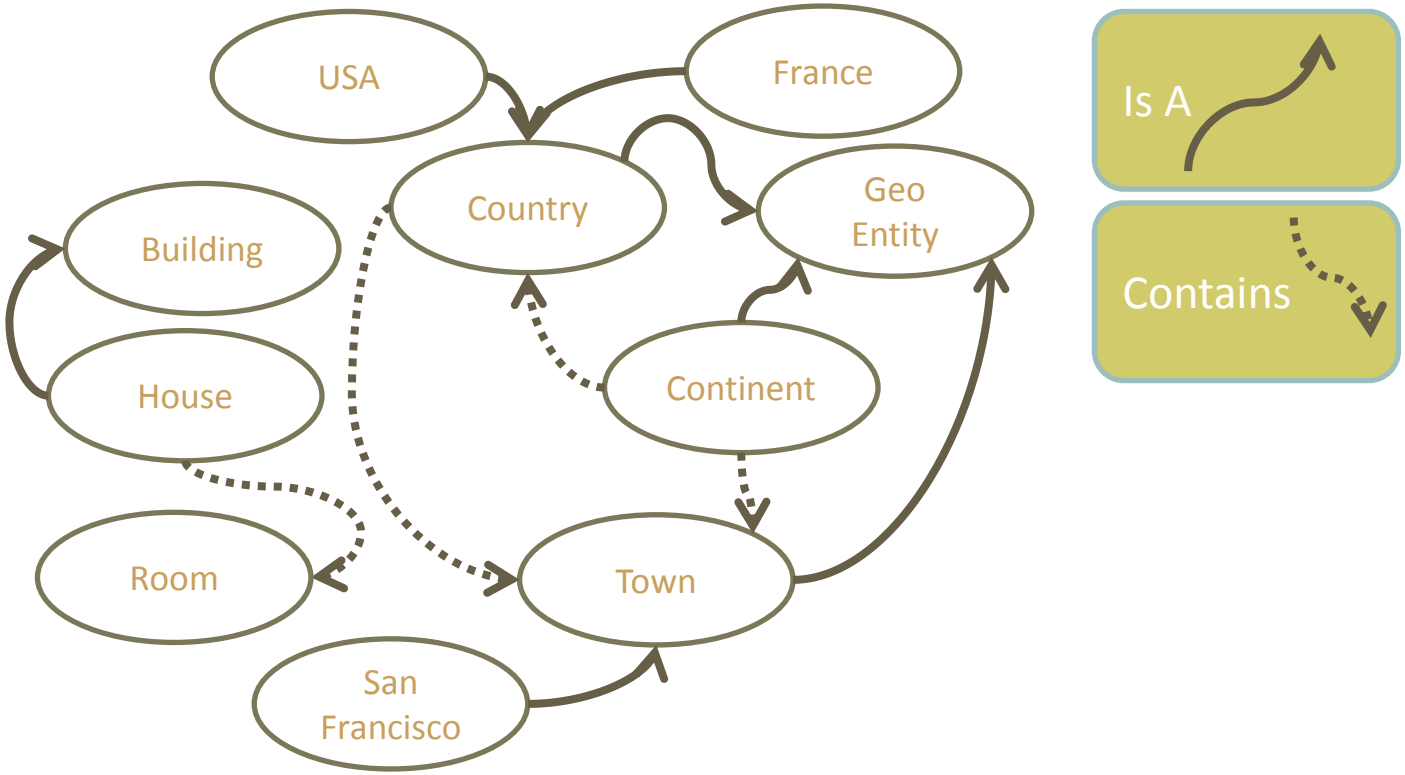


S	P	O
USA	is a	Country
France	is a	Country
Country	is a	Geo Entity
Country	contains	Town
Continent	contains	Country
Continent	is a	Geo Entity
Continent	contains	Town
Town	is a	Geo Entity
San Francisco	is a	Town
House	contains	Room
House	is a	Building

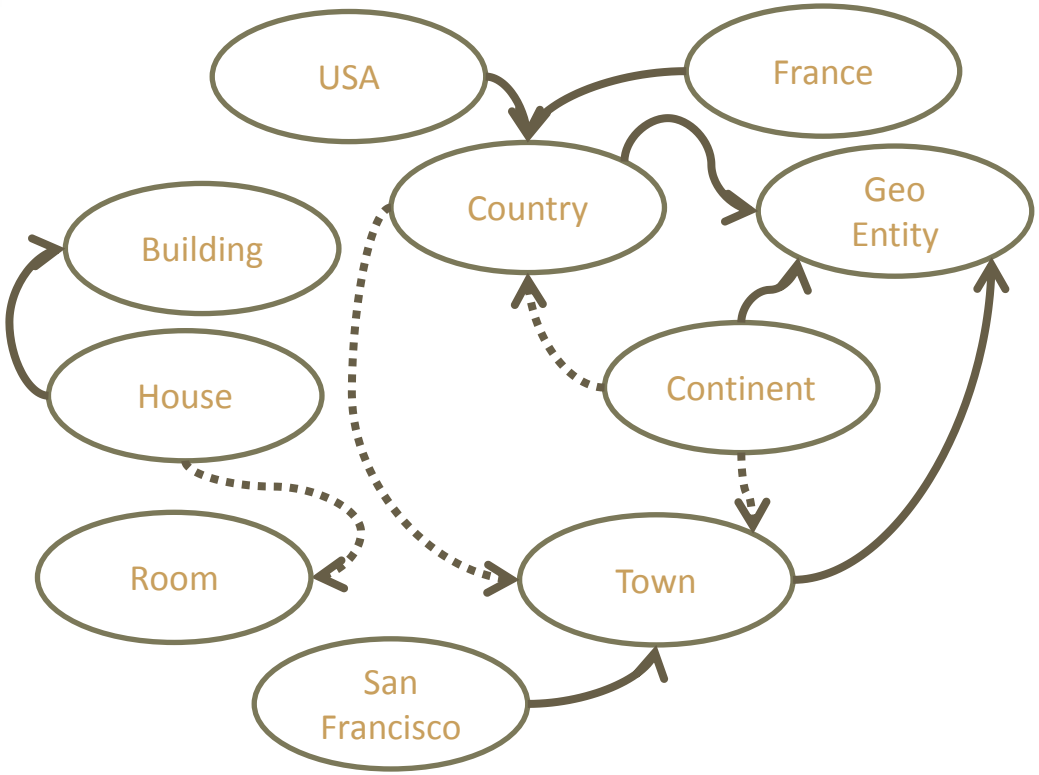
SPARQL: Graph

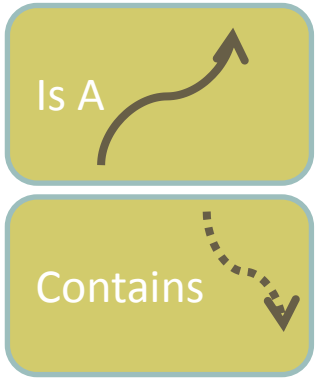


SPARQL: Graph



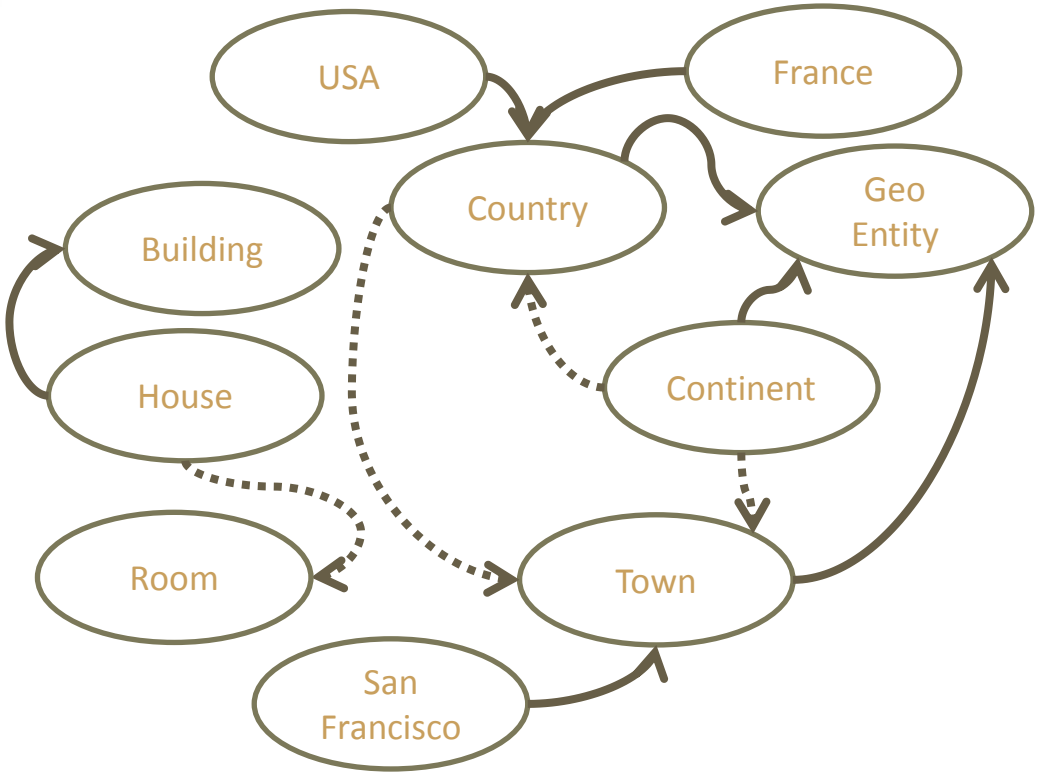
SPARQL: Subjects & Objects

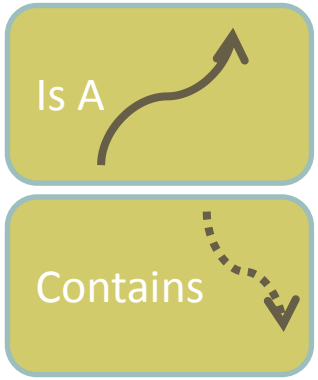




Show node (subject) that contains something
Show node (object) that is contained by something
Show node (subject) that is a Geo Entity

SPARQL: Subjects & Objects

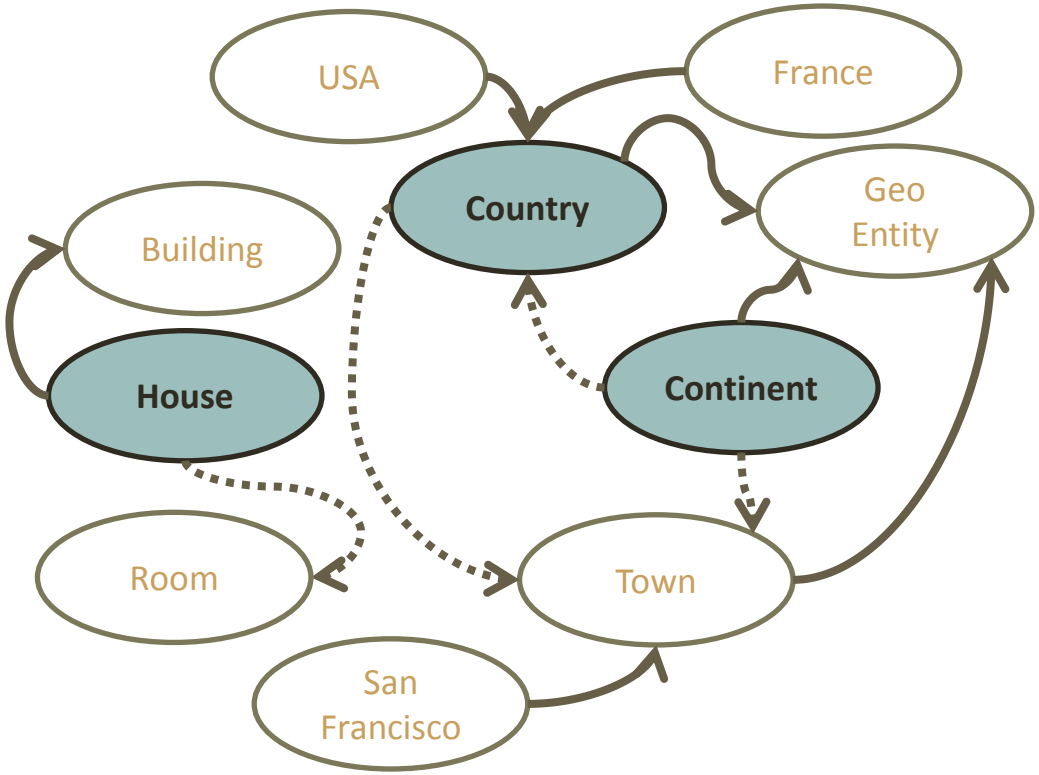


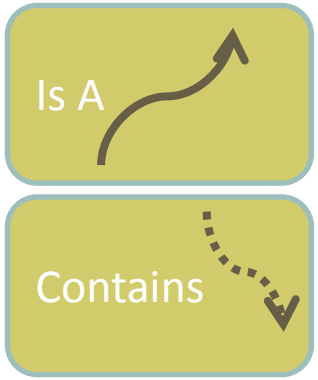



```
SELECT<return variables>
FROM <Triplestore>
WHERE { <condition>}
```

- Show node (subject) that contains something
- Show node (object) that is contained by something
- Show node (subject) that is a Geo Entity

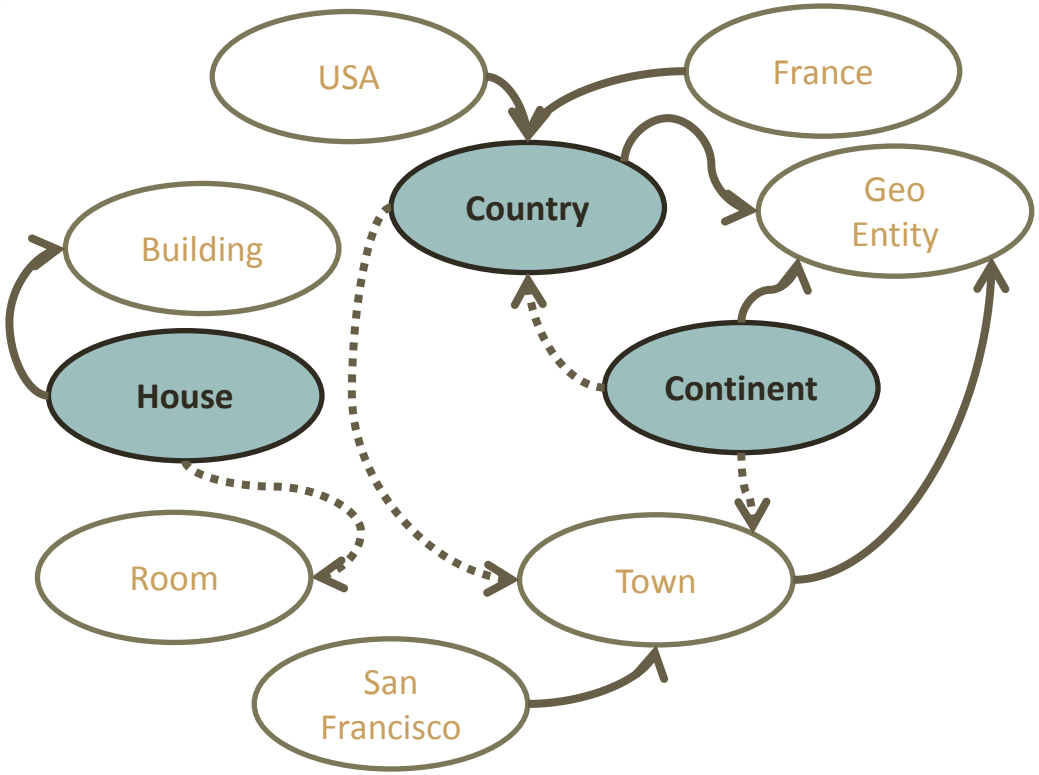
SPARQL: Subjects & Objects

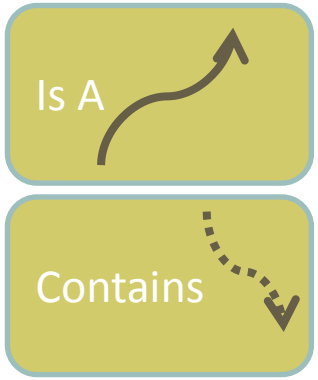




Show node (subject) that contains something
Show node (object) that is contained by something
Show node (subject) that is a Geo Entity

SPARQL: Subjects & Objects

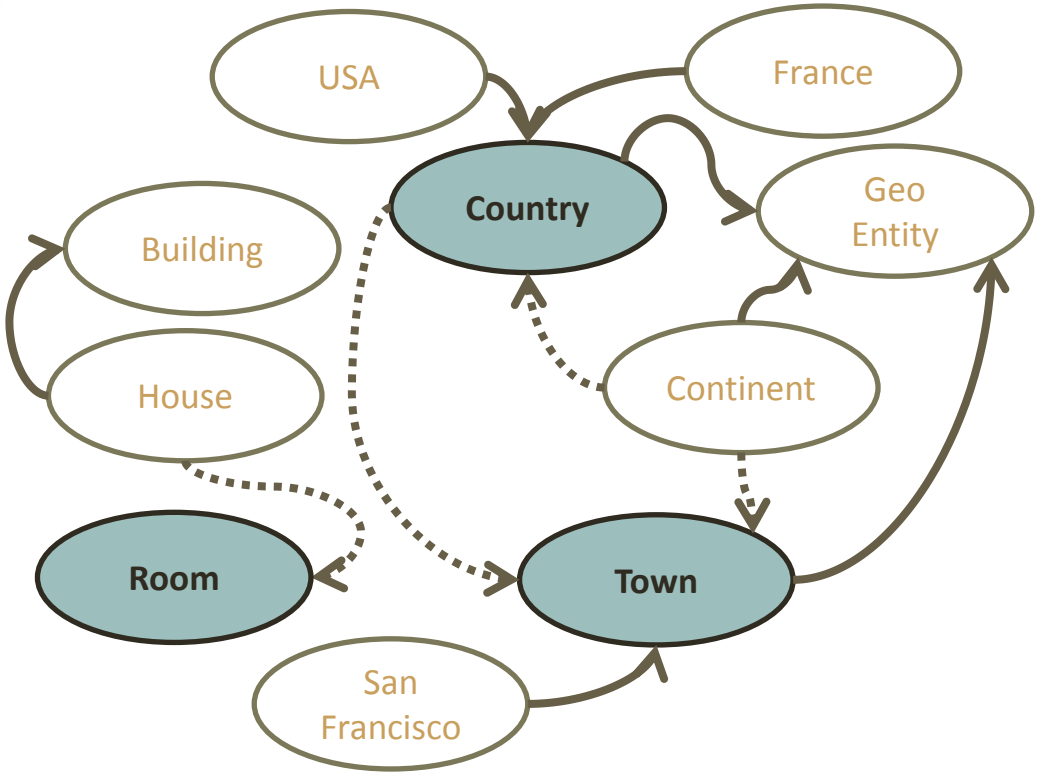


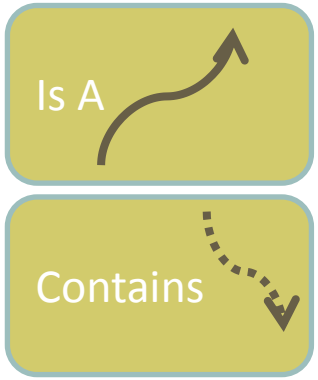



```
SELECT ?s
FROM <myTripleStore>
WHERE {
?s contains ?o
}
```

Show node (subject) that contains something
Show node (object) that is contained by something
Show node (subject) that is a Geo Entity

SPARQL: Subjects & Objects



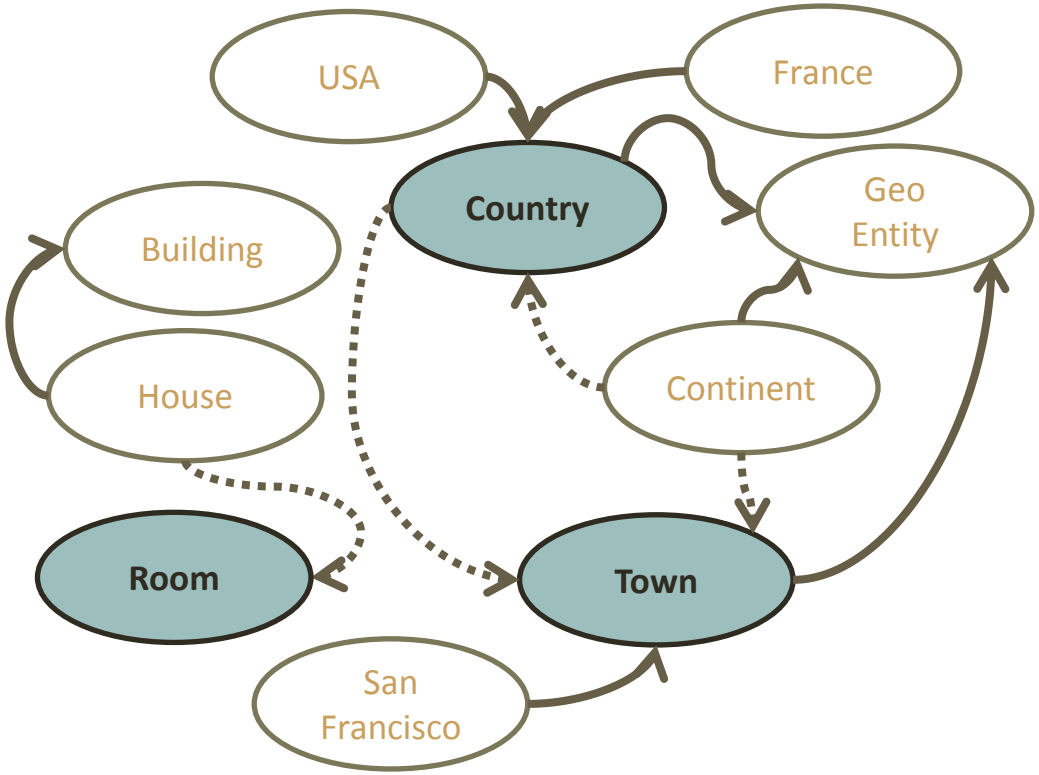


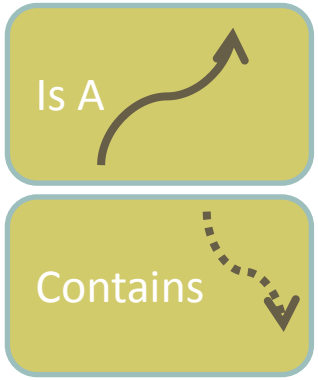
Show node (subject) that contains something

Show node (object) that is contained by something

Show node (subject) that is a Geo Entity

SPARQL: Subjects & Objects





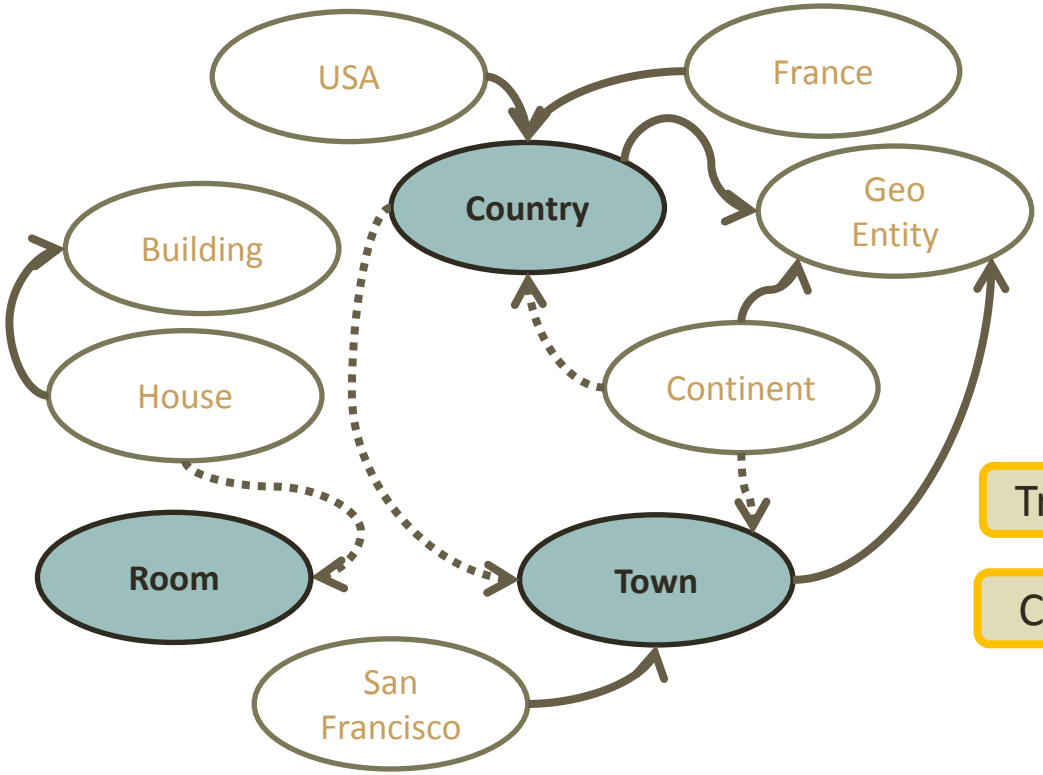

```
SELECT ?o
FROM <myTripleStore>
WHERE {
?s contains ?o}
```

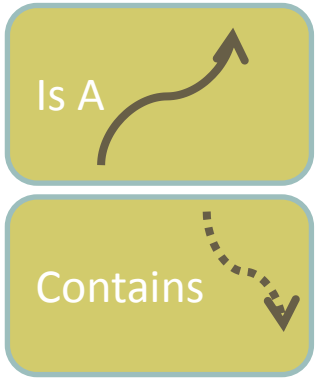
Show node (subject) that contains something

Show node (object) that is contained by something

Show node (subject) that is a Geo Entity

SPARQL: Subjects & Objects





Return Variable

Triplestore

Condition

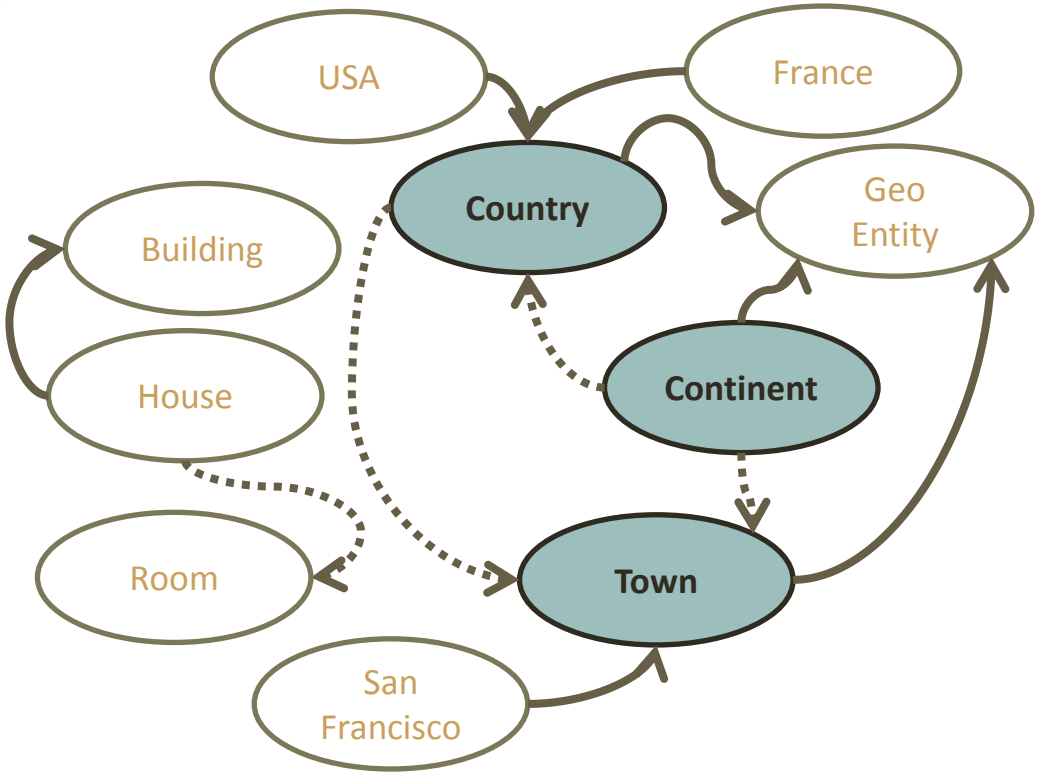
```
SELECT ?o
FROM <myTripleStore>
WHERE {
?s contains ?o}
```

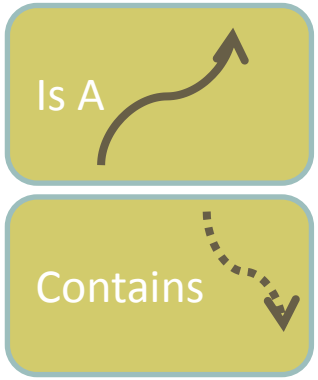
Show node (subject) that contains something

Show node (object) that is contained by something

Show node (subject) that is a Geo Entity

SPARQL: Subjects & Objects



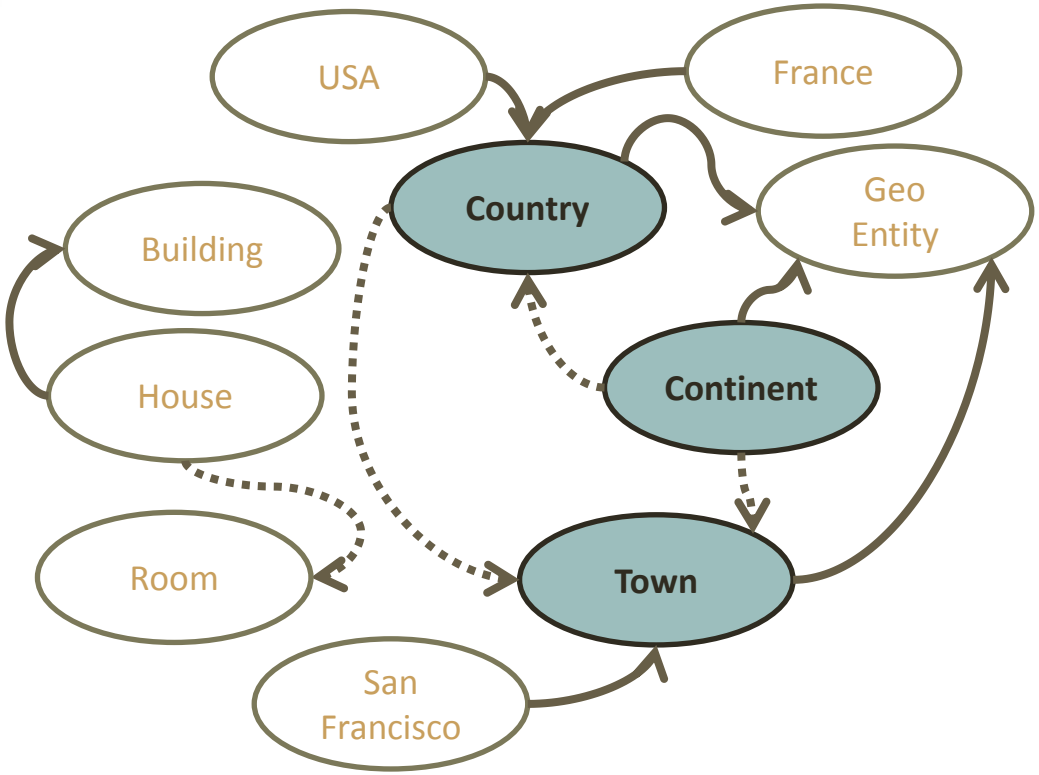


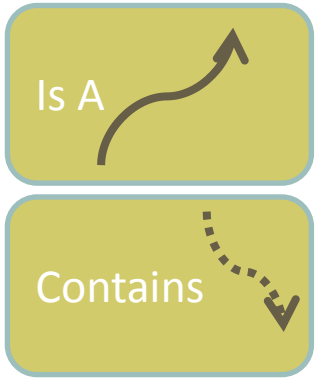
Show node (subject) that contains something

Show node (object) that is contained by something

Show node (subject) that is a Geo Entity

SPARQL: Subjects & Objects





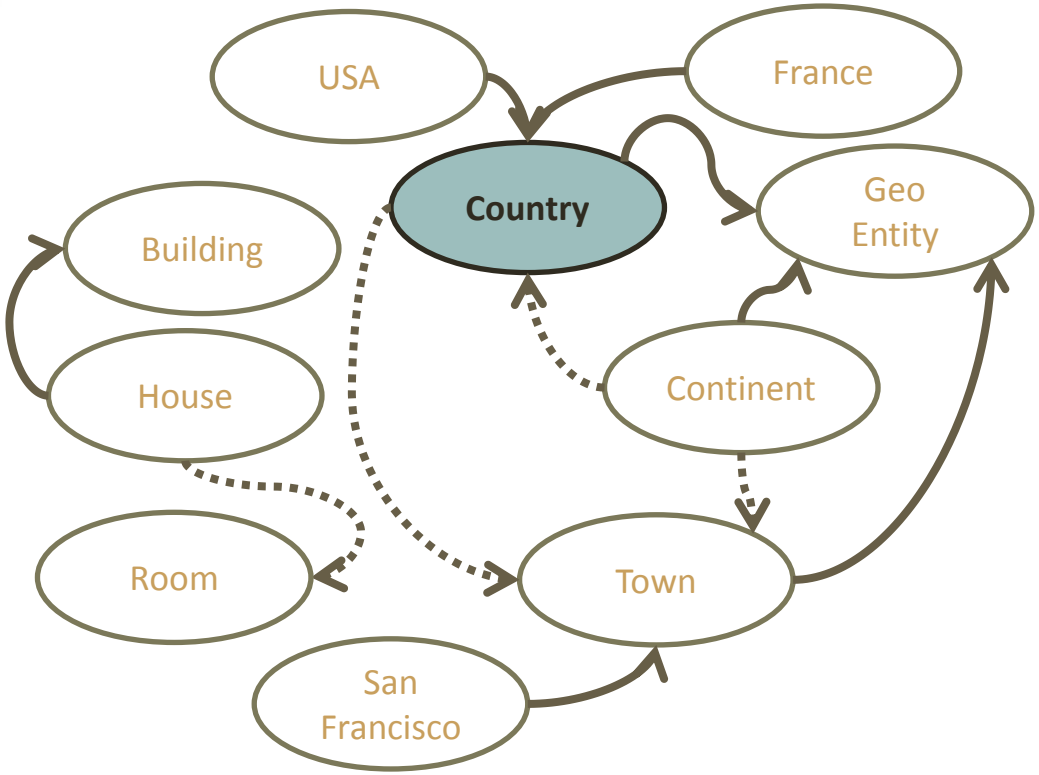

```
SELECT ?s
FROM <myTripleStore>
WHERE {
?s is a Geo Entity}
```

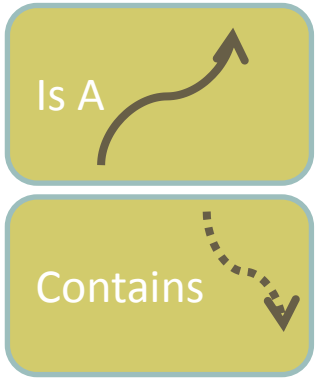
Show node (subject) that contains something

Show node (object) that is contained by something

Show node (subject) that is a Geo Entity

SPARQL: Subjects & Objects



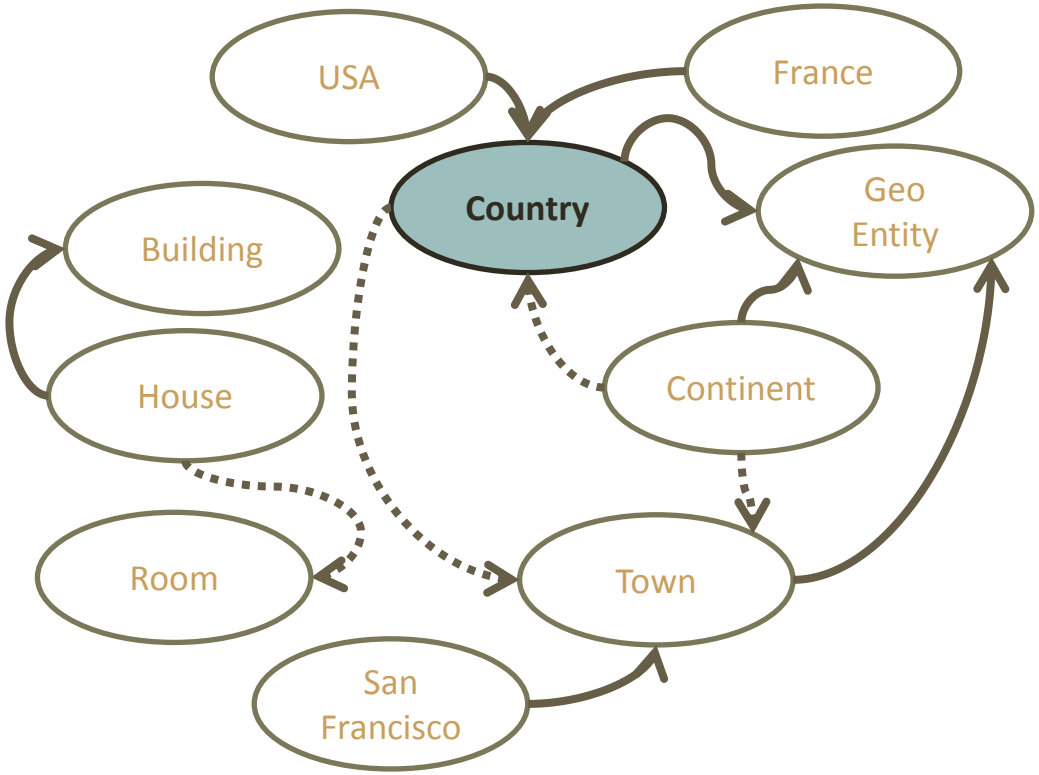


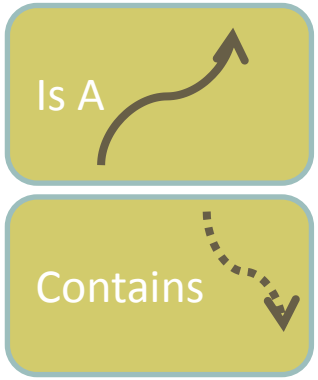
Show node (subject) that contains something

Show node (object) that is contained by something

Show node (subject) that is a Geo Entity

SPARQL: Subjects & Objects





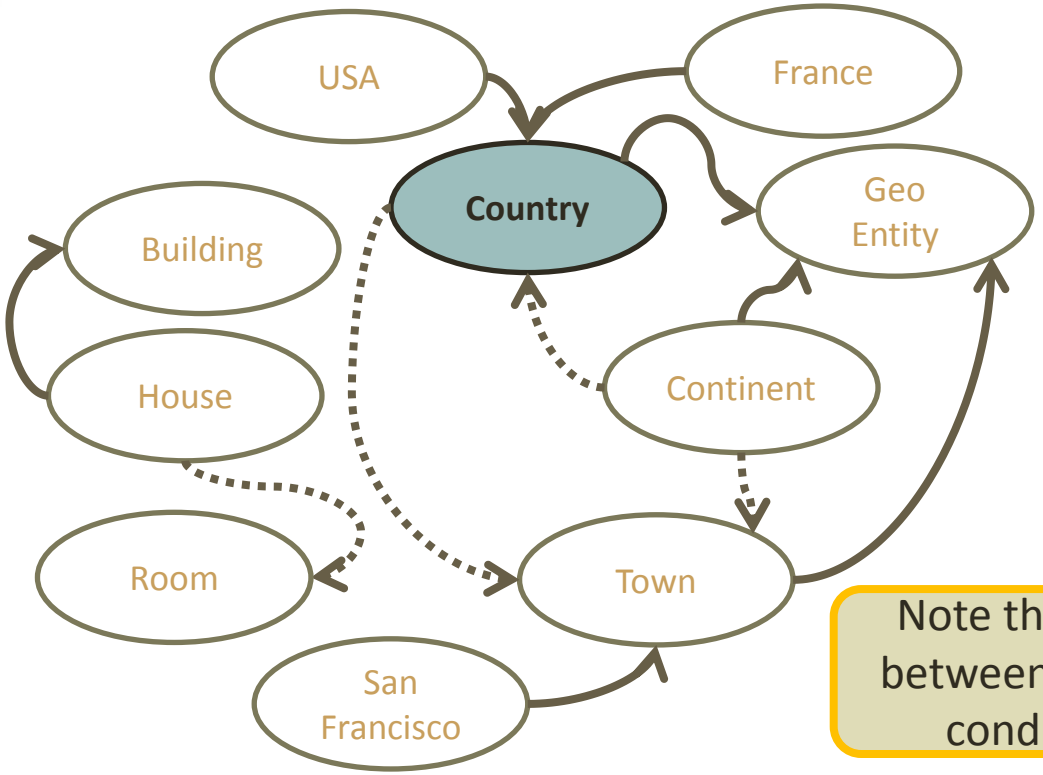

```
SELECT ?s
FROM <myTripleStore>
WHERE {
?s Contains ?o .
?x Contains ?s}
```

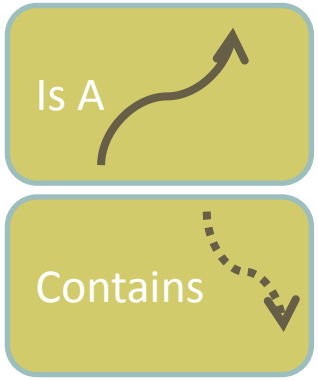
Show node (subject) that contains something

Show node (object) that is contained by something

Show node (subject) that is a Geo Entity

SPARQL: Subjects & Objects






```
SELECT ?s
FROM <myTripleStore>
WHERE {
?s Contains ?o.
?x Contains ?s}
```

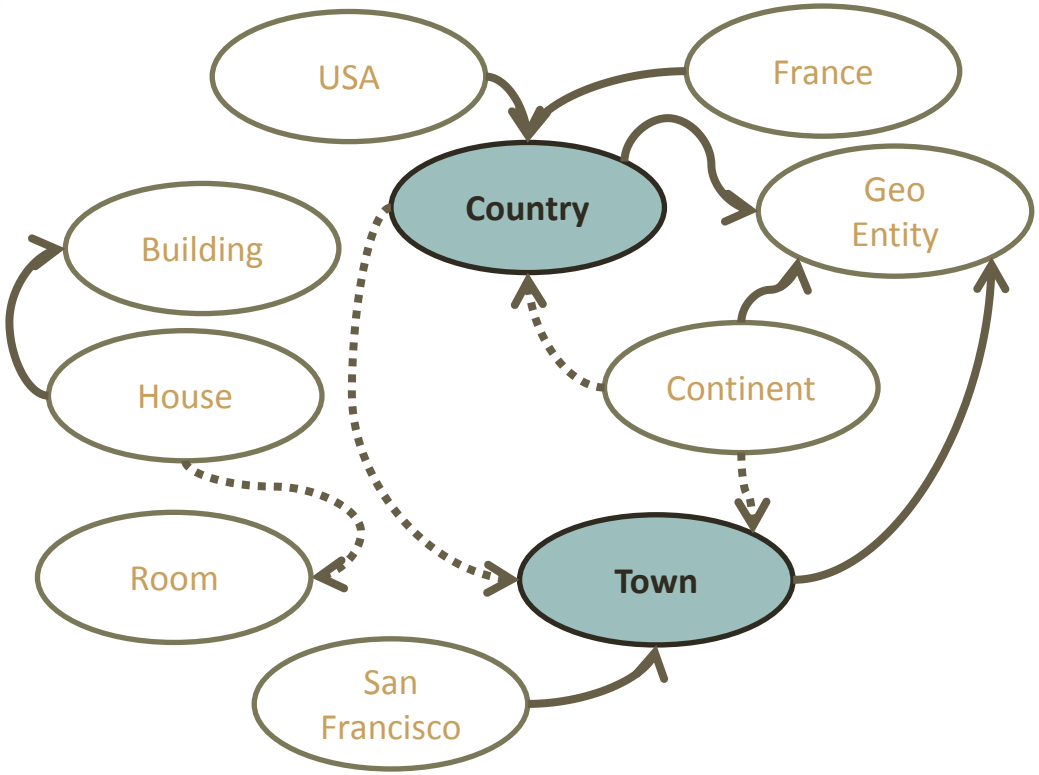
Note the dot (.) between the two conditions

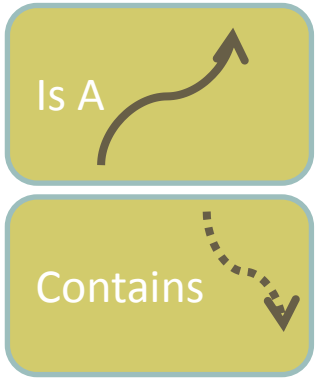
Show node (subject) that contains something

Show node (object) that is contained by something

Show node (subject) that is a Geo Entity

SPARQL: Subjects & Objects



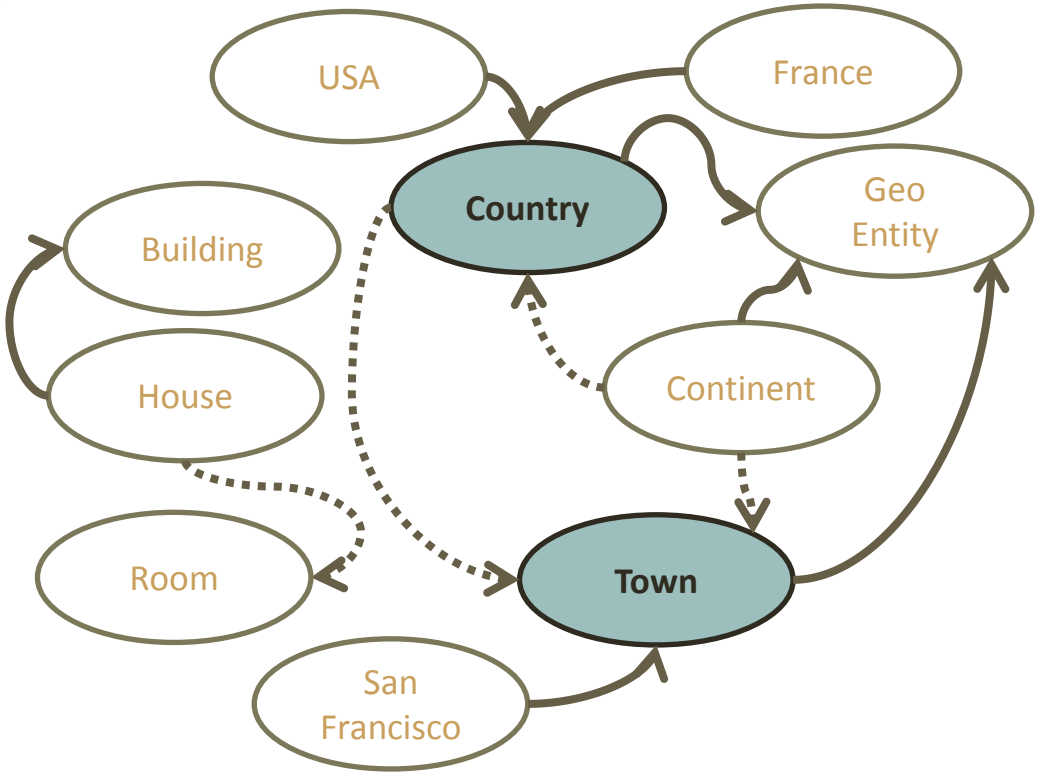


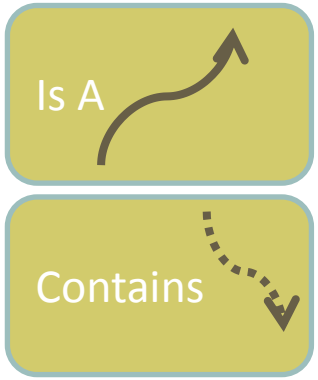
Show node (subject) that contains something

Show node (object) that is contained by something

Show node (subject) that is a Geo Entity

SPARQL: Subjects & Objects





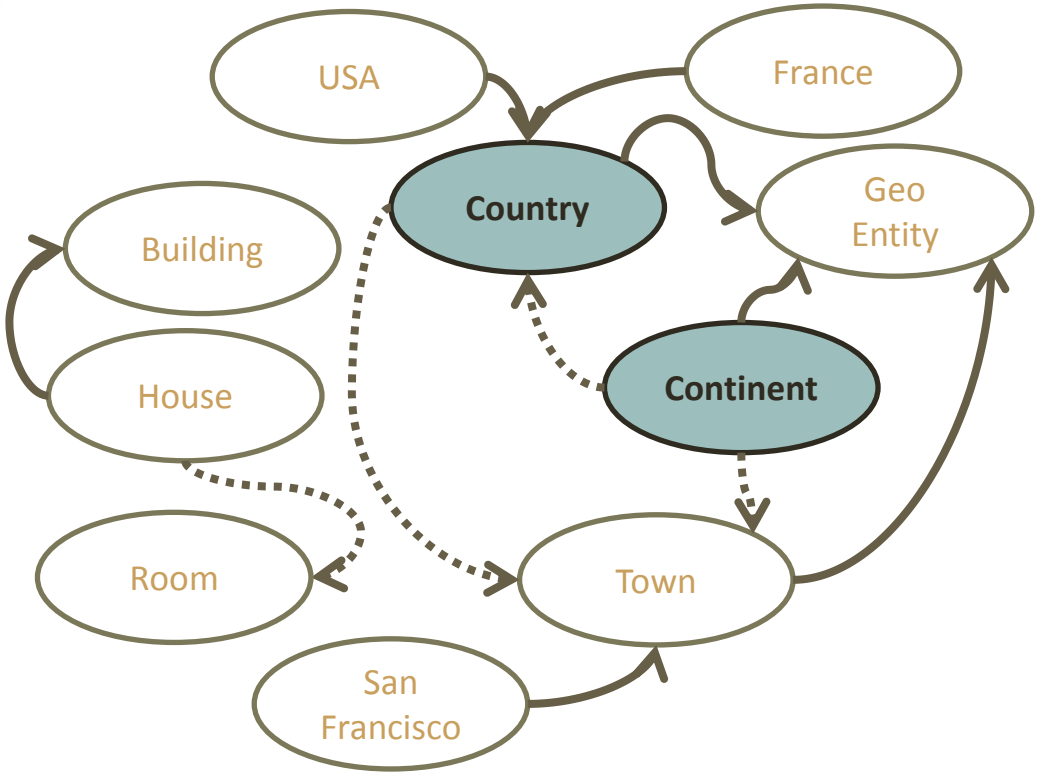

```
SELECT ?o
FROM <myTripleStore>
WHERE {
?x Contains ?o .
?o is a Geo Entity}
```

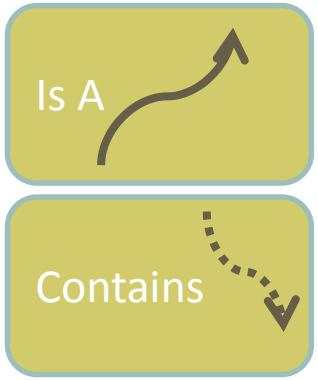
Show node (subject) that contains something

Show node (object) that is contained by something

Show node (subject) that is a Geo Entity

SPARQL: Subjects & Objects



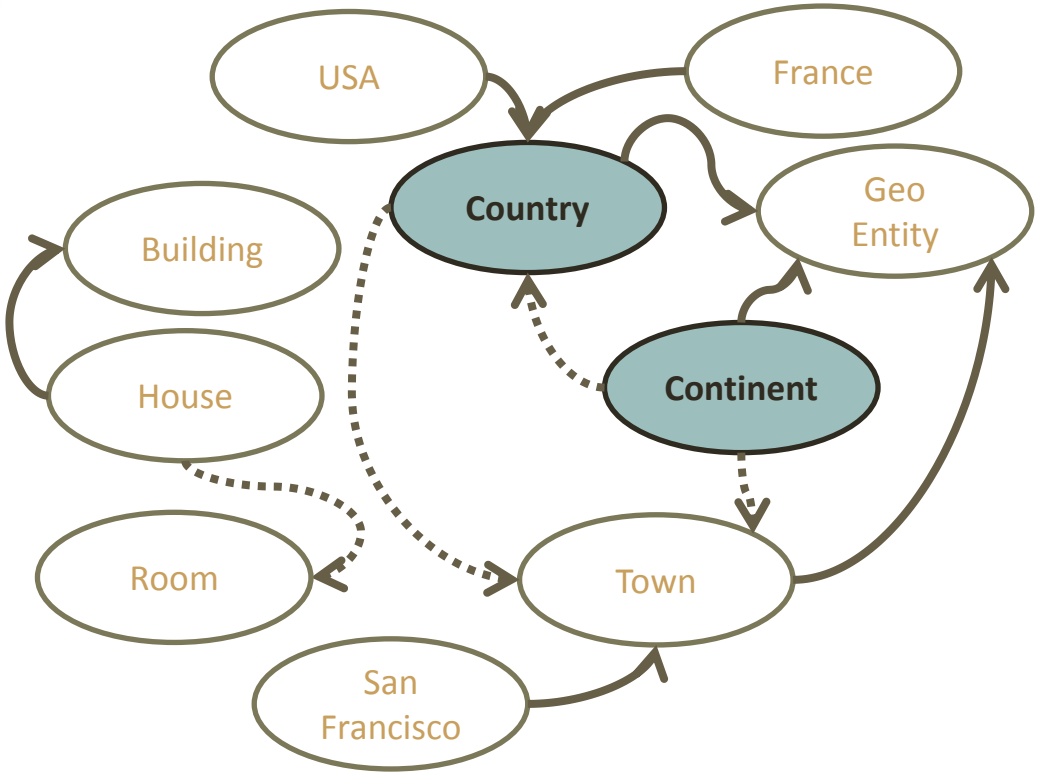


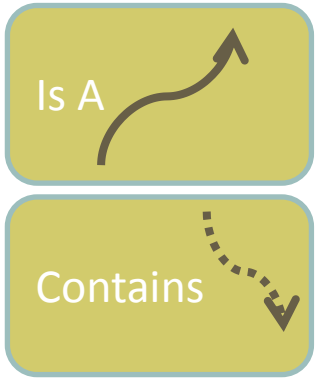
Show node (subject) that contains something

Show node (object) that is contained by something

Show node (subject) that is a Geo Entity

SPARQL: Subjects & Objects





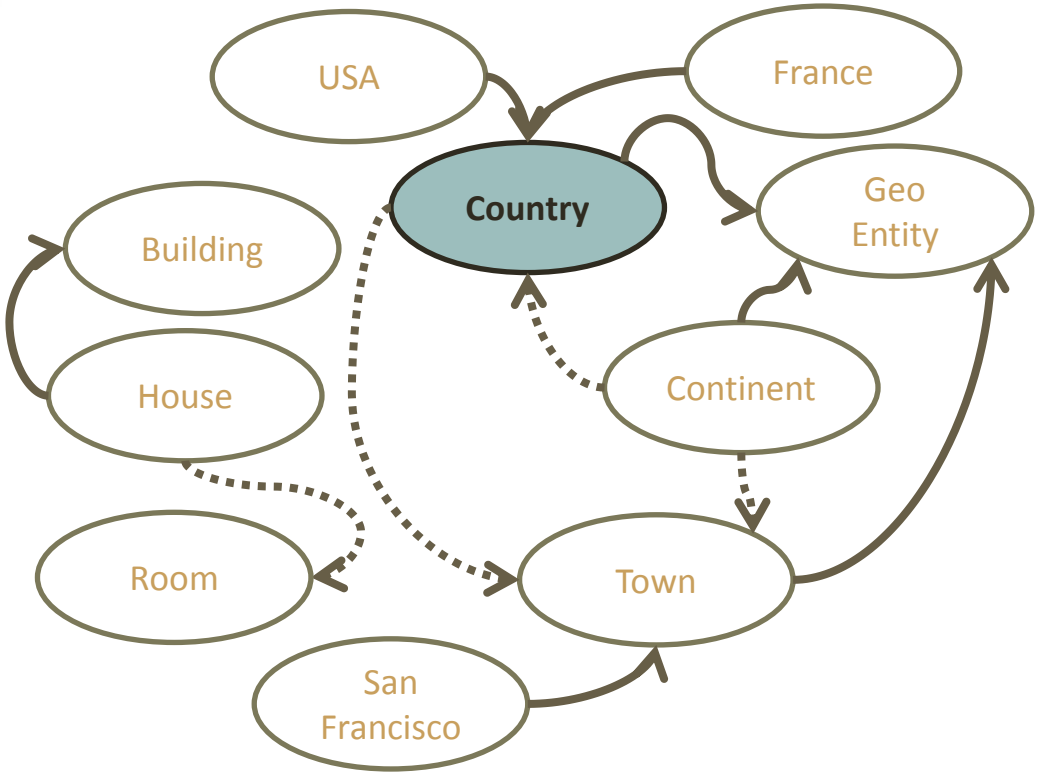

```
SELECT ?s
FROM <myTripleStore>
WHERE {
?s Contains ?o .
?s is a Geo Entity}
```

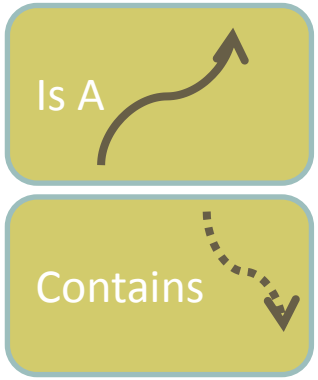
Show node (subject) that contains something

Show node (object) that is contained by something

Show node (subject) that is a Geo Entity

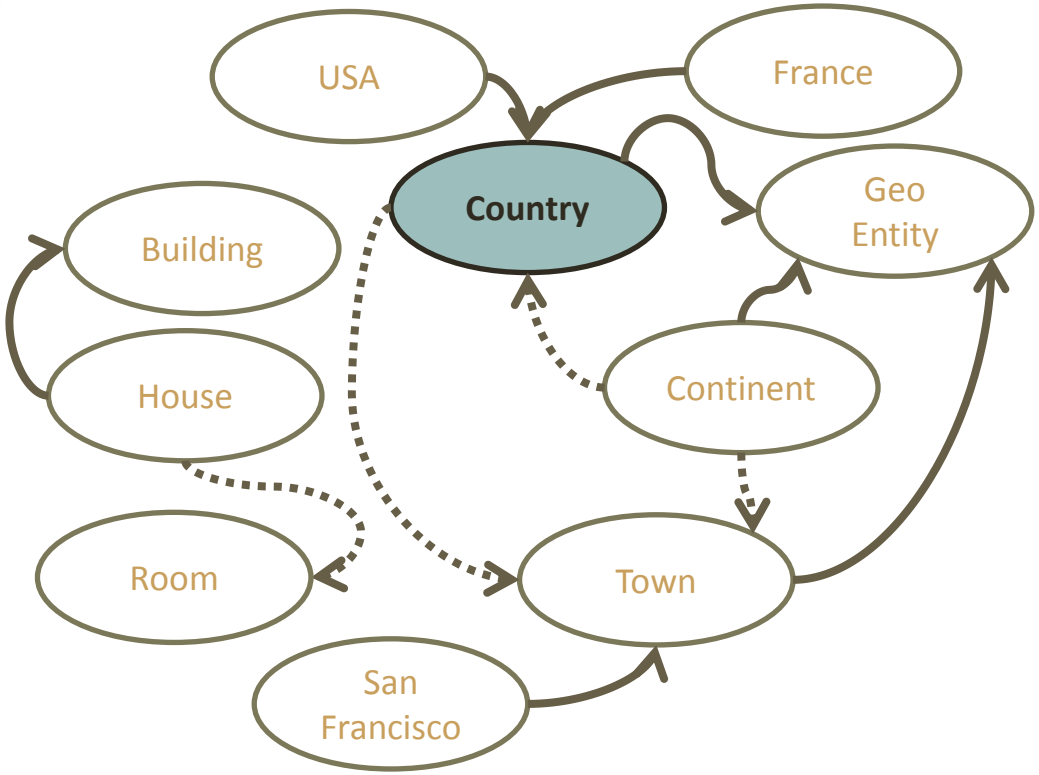
SPARQL: Subjects & Objects

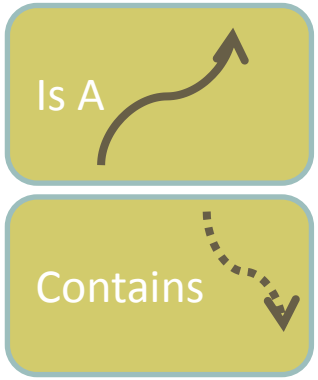




Show node (subject) that contains something
Show node (object) that is contained by something
Show node (subject) that is a Geo Entity

SPARQL: Subjects & Objects






```
SELECT ?s
FROM <myTripleStore>
WHERE {
?s Contains ?o .
?x Contains ?s .
?s is a Geo Entity}
```

Show node (subject) that contains something

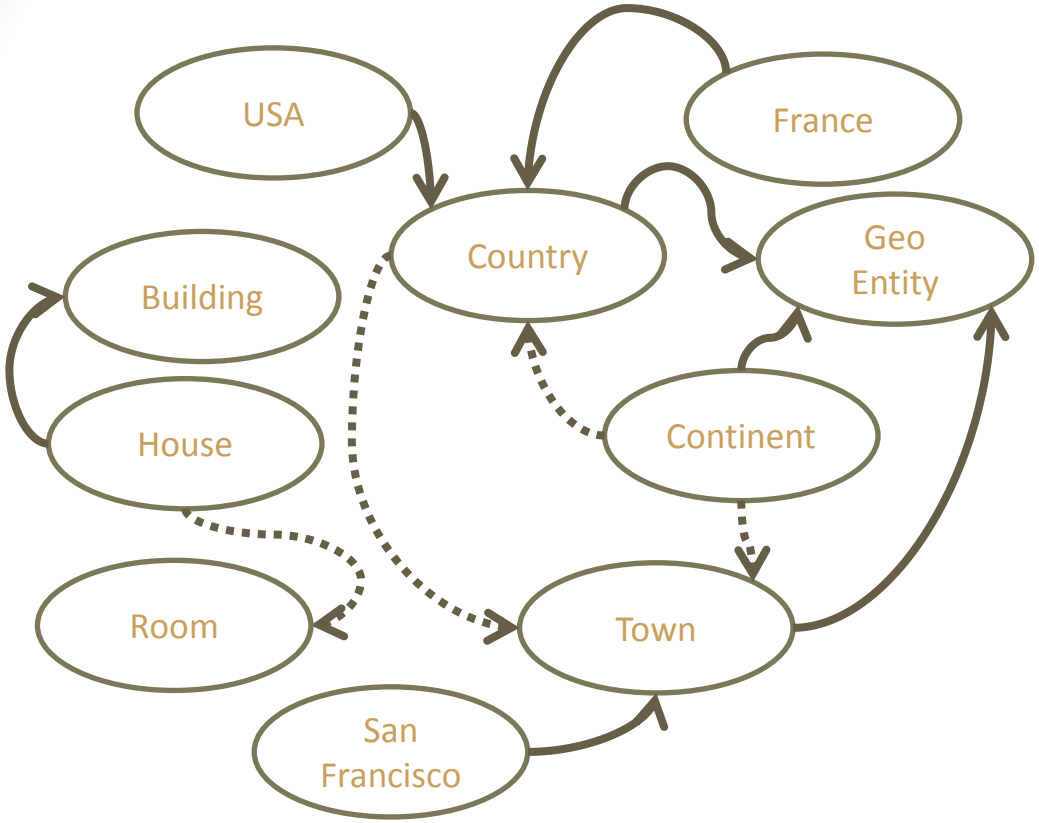
Show node (object) that is contained by something

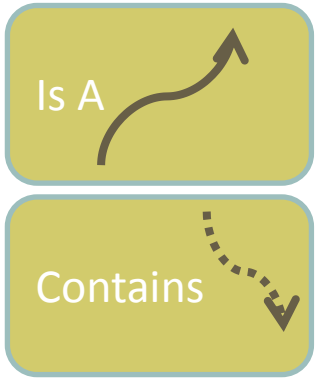
Show node (subject) that is a Geo Entity

SPARQL: Predicates

- Search for predicates

SPARQL: Predicates



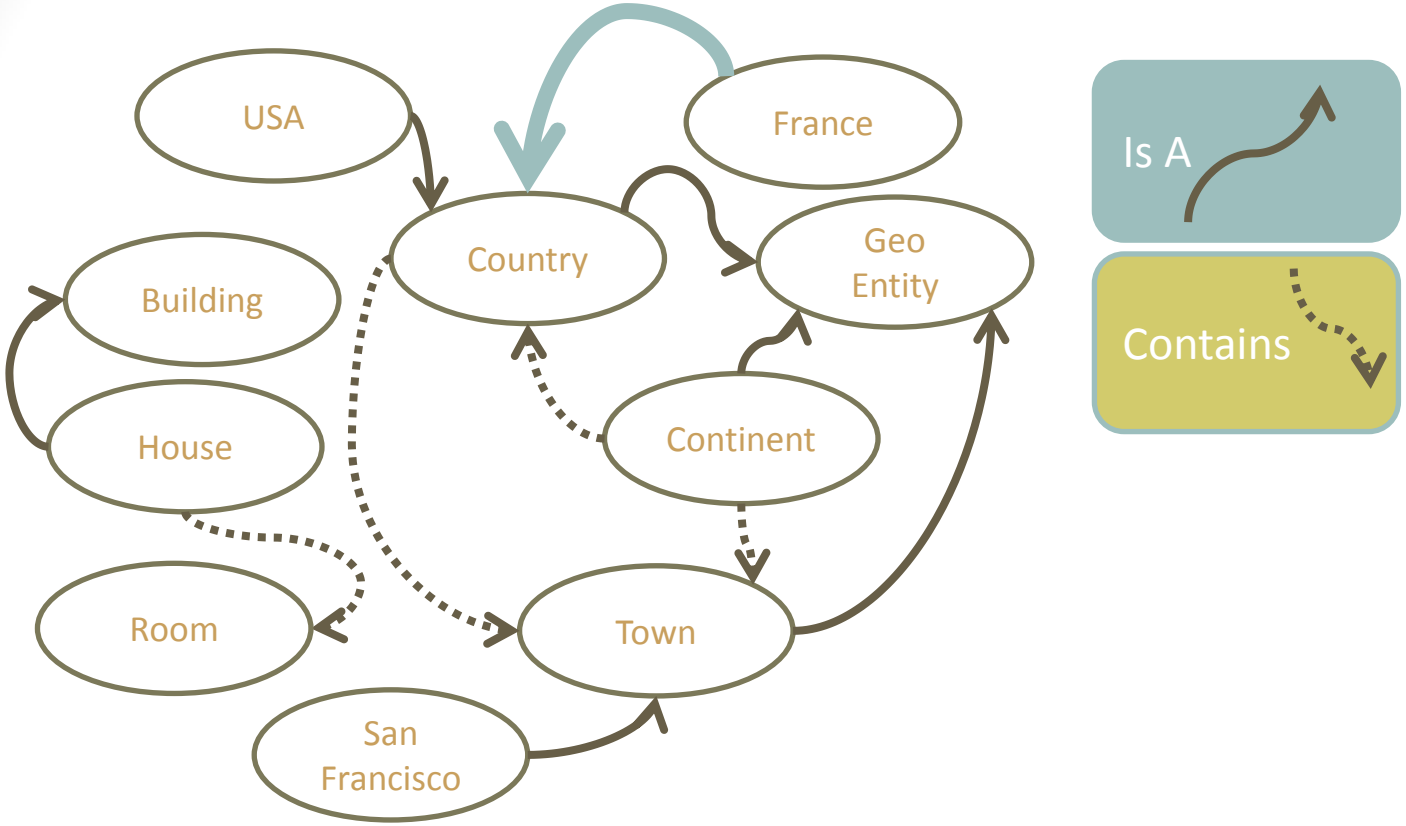



```
SELECT<return variables>
FROM <Triplestore>
WHERE { <condition>}
```

Show relationships (Predicate) from France to Anything

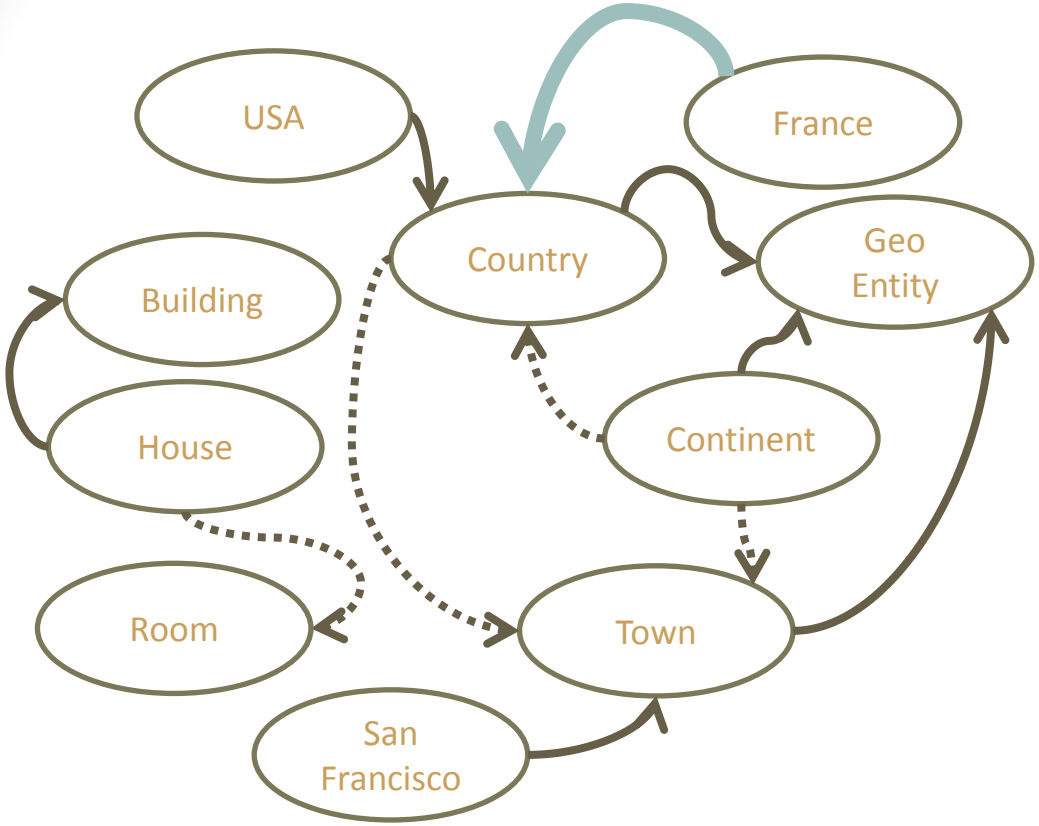
Show relationships from Country to Town

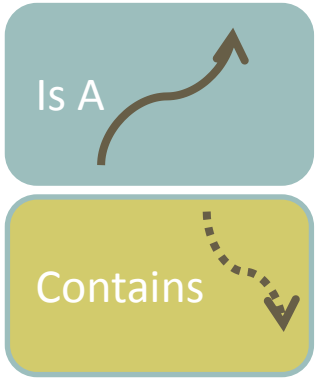
SPARQL: Predicates



Show relationships (Predicate) from France to Anything
Show relationships from Country to Town

SPARQL: Predicates



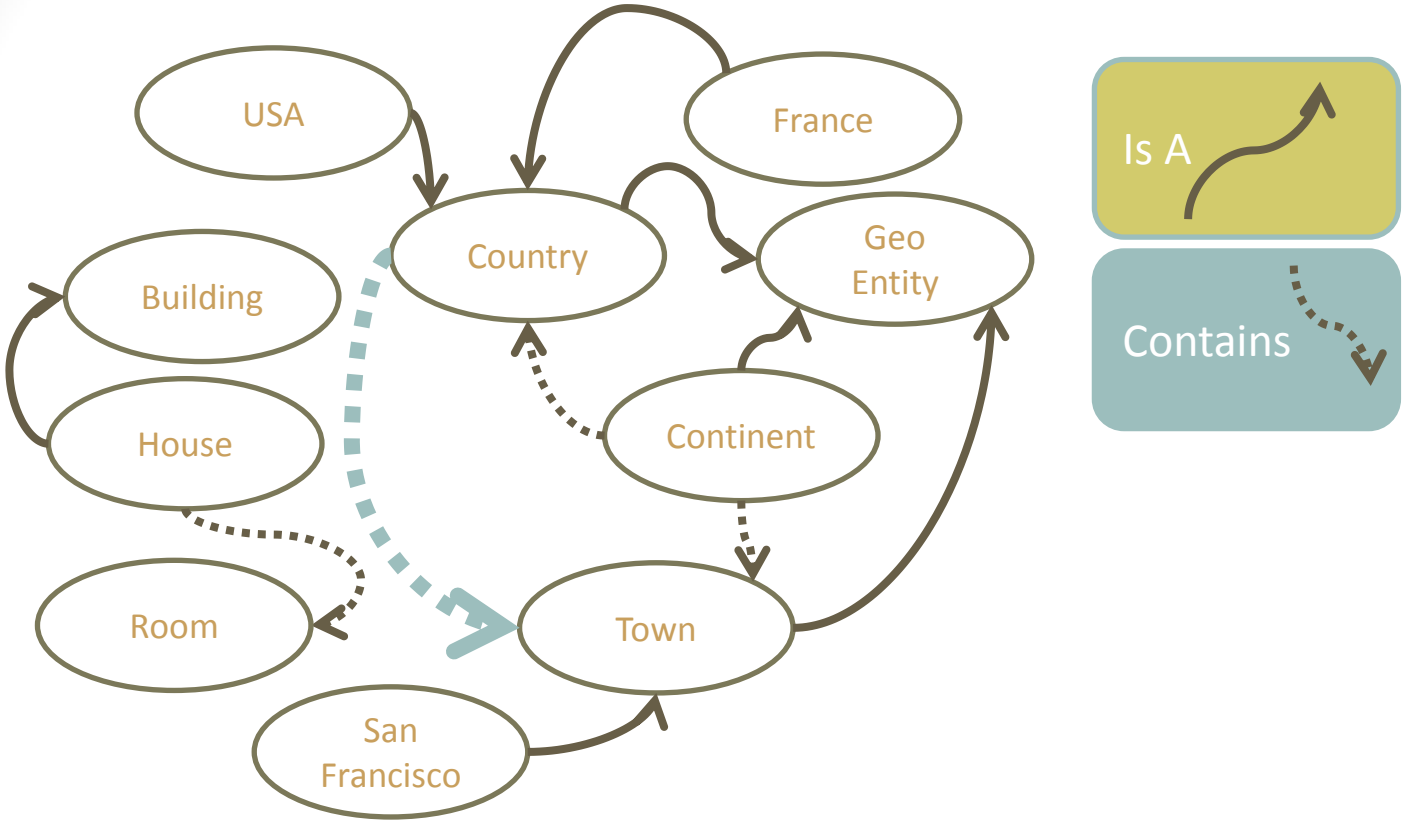



```
SELECT ?p
FROM <myTripleStore>
WHERE {
France ?p ?o}
```

Show relationships (Predicate) from France to Anything

Show relationships from Country to Town

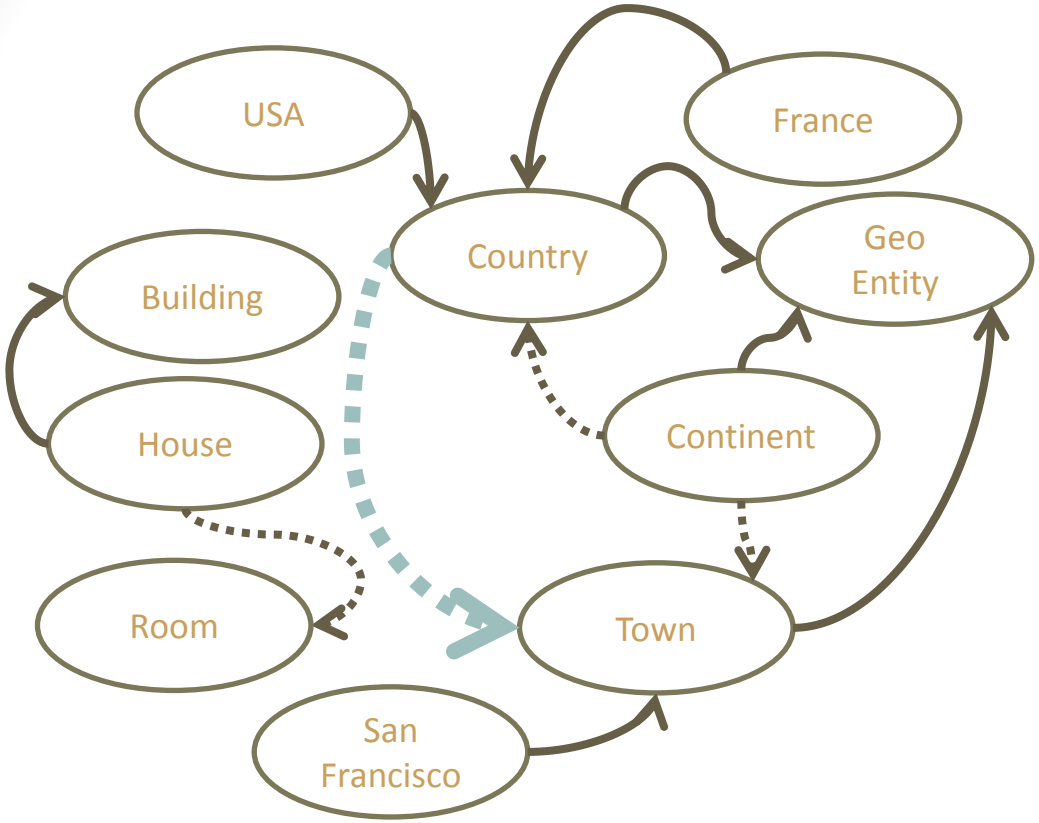
SPARQL: Predicates

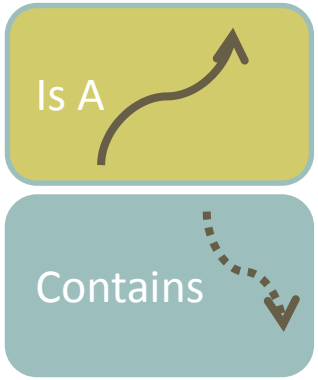


Show relationships (Predicate) from France to Anything

Show relationships from Country to Town

SPARQL: Predicates



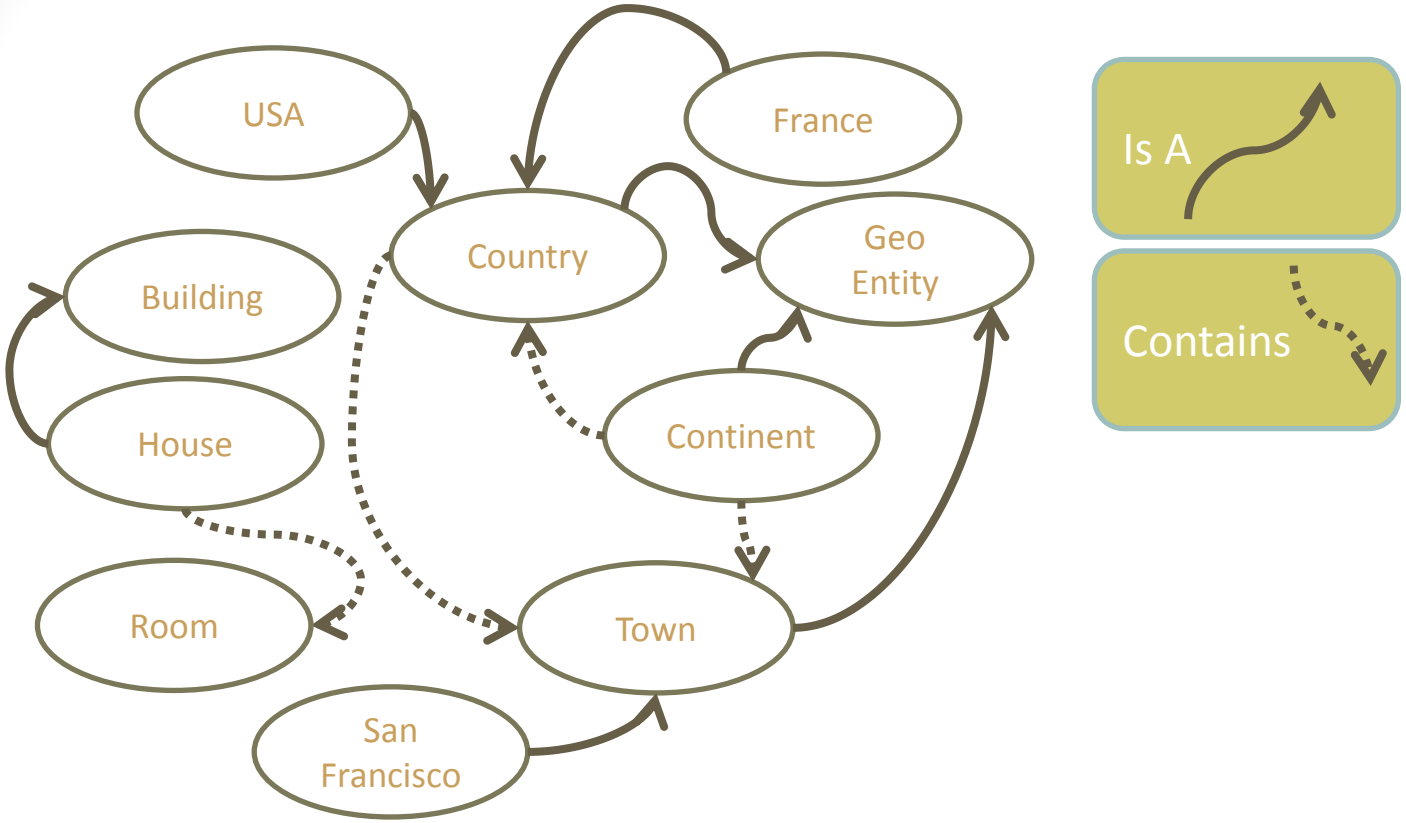



```
SELECT ?p
FROM <myTripleStore>
WHERE {
Country ?p Town}
```

Show relationships (Predicate) from France to Anything

Show relationships from Country to Town

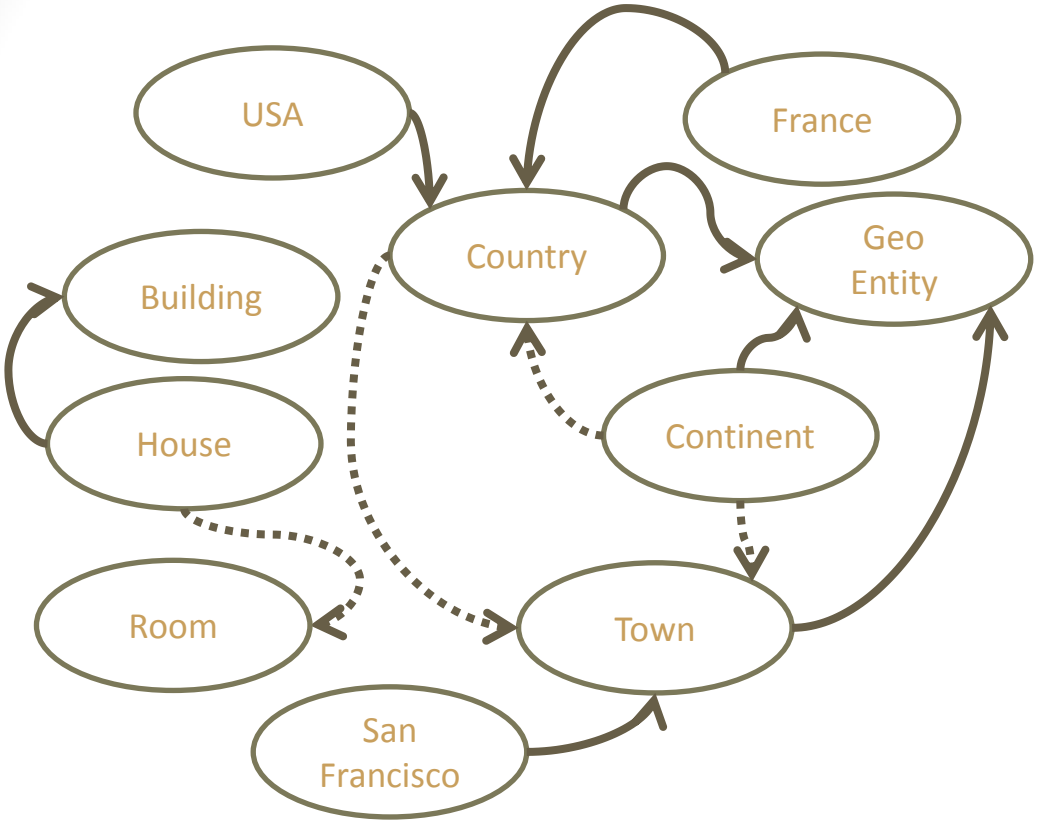
SPARQL: Predicates

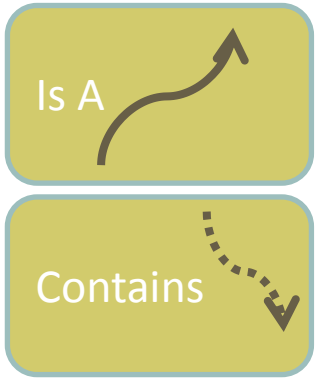


Show relationships (Predicate) from France to Anything

Show relationships from Country to Town

SPARQL: Predicates



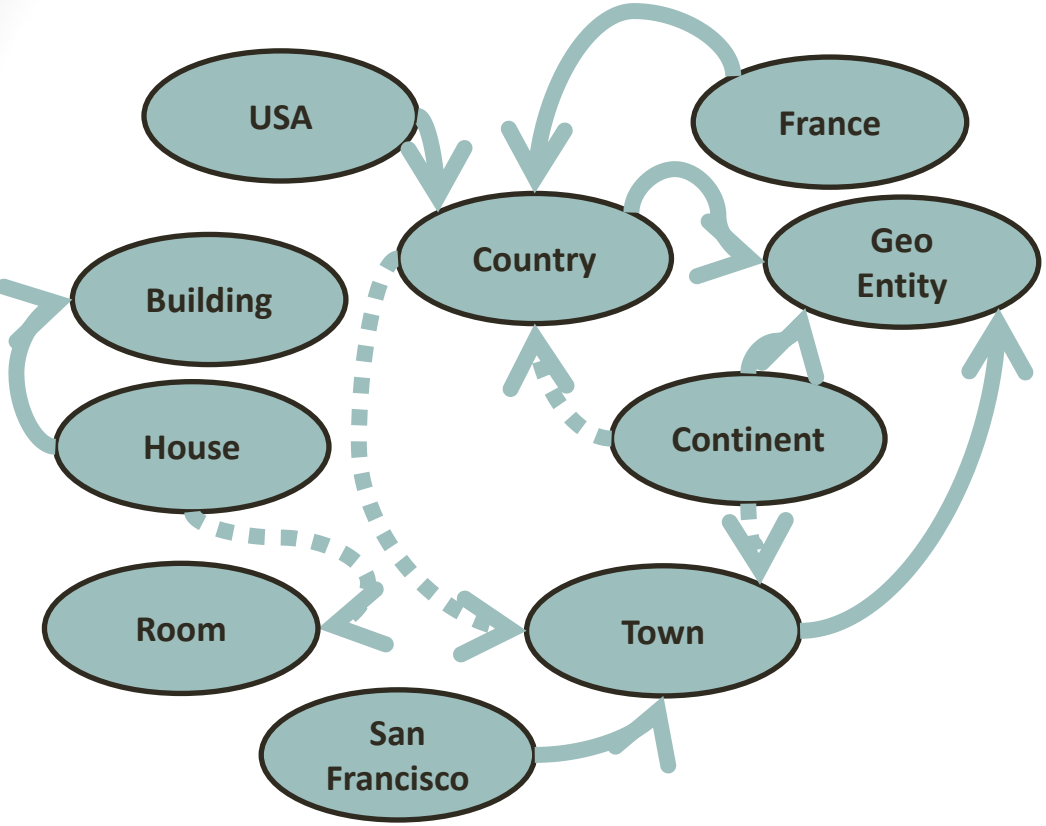


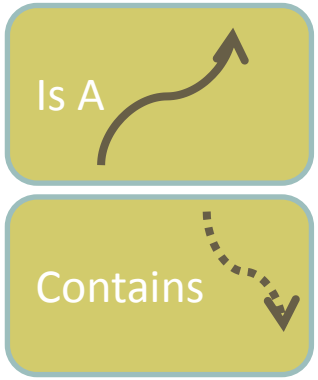

```
SELECT ?p
FROM <myTripleStore>
WHERE {
France ?p ?o .
Country ?p Town}
```

Show relationships (Predicate) from France to Anything

Show relationships from Country to Town

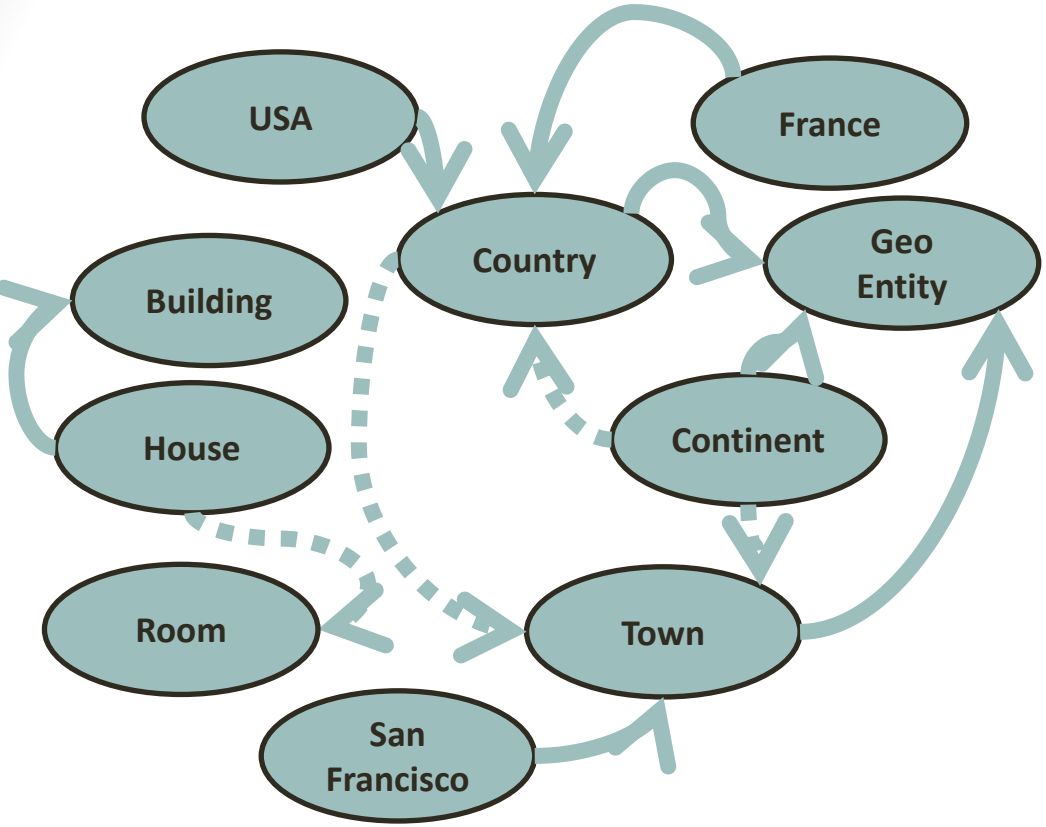
SPARQL: Predicates





Show any node and relationship

SPARQL: Predicates




```
SELECT ?s ?p ?o
FROM <myTripleStore>
WHERE {
?s ?p ?o}
```

Show any node and relationship

Some Links

- http://dbpedia.org/sparql
- http://lod.openlinksw.com/sparql/
- http://librdf.org/query/

- Online tutorial: http://www.cambridgesemantics.com/semantic-university/sparql-by-example
-
- http://sparql.org/sparql.html
- http://demo.openlinksw.com/sparql

Introduction to SPARQL

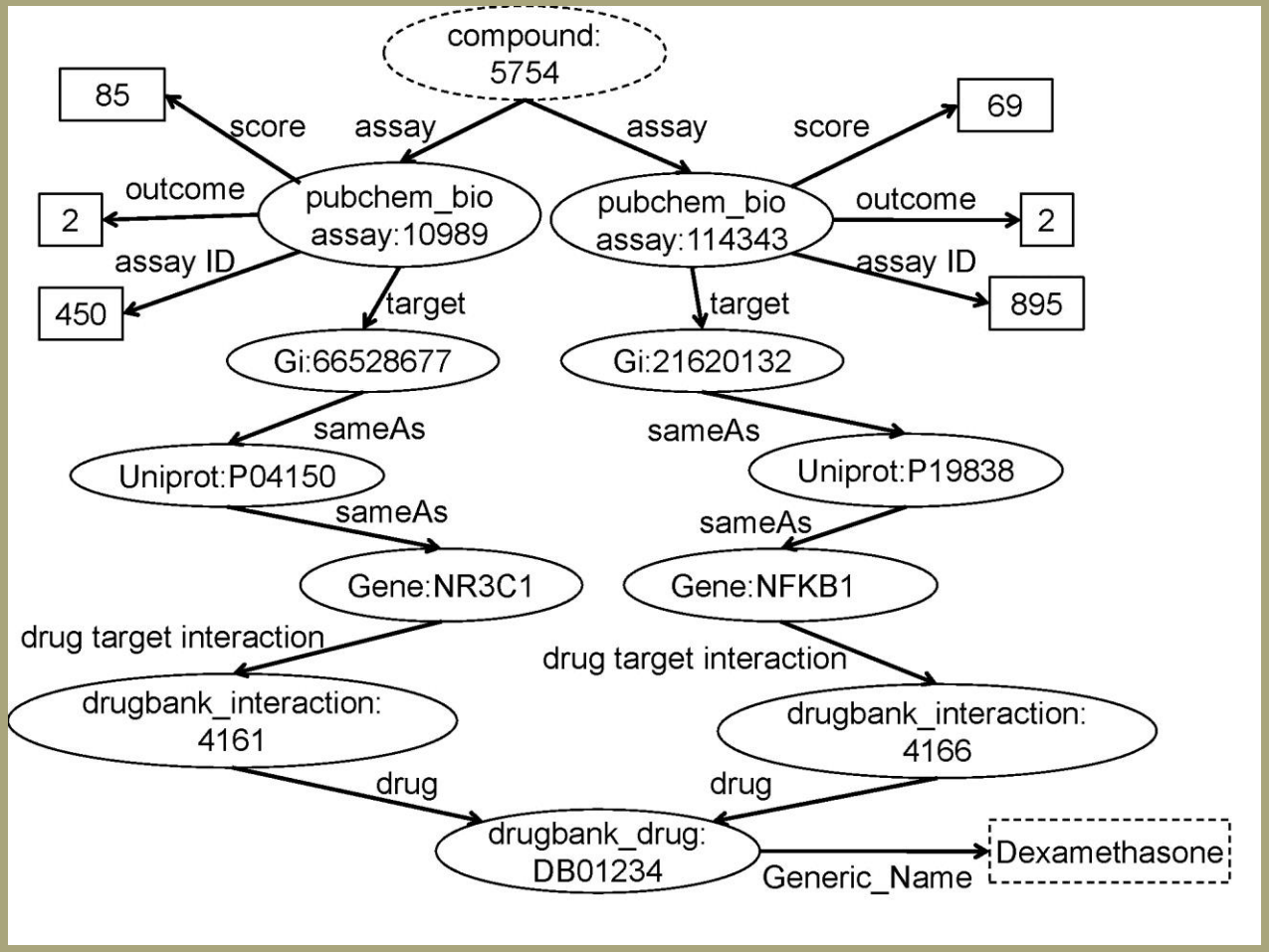
Quiz SPARQL Intro

- The question and answers for the first two questions are shown during the quiz. See the projected slide.

SPARQL Examples

SPARQL Examples

Query for Adverse Drug Interactions



SPARQL Examples

```
#Whole Database:
SELECT ?s ?p ?o
FROM <http://dbpedia.org>
WHERE
{?s ?p ?o}
```

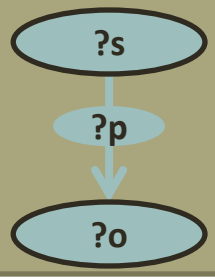
SPARQL Examples

Return Variable

Triplestore

Condition

```
#Whole Database:
SELECT ?s ?p ?o
FROM <http://dbpedia.org>
WHERE
{?s ?p ?o}
```



You can get a text file with the SPARQL queries and links from Canvas.

SPARQL Examples

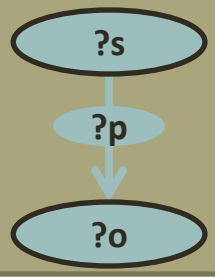
- SPARQL Databases:
  - http://dbpedia.org/sparql
  - http://lod.openlinksw.com/sparql/

Return Variable

Triplestore

Condition

```
#Total number of triples:
SELECT (COUNT(*) AS ?NumberOfTriples)
FROM <http://dbpedia.org>
WHERE
{?s ?p ?o}
```



SPARQL Examples

- SPARQL Databases:
  - http://dbpedia.org/sparql

```
# Soccer players with more than 44 international goals
PREFIX property: <http://dbpedia.org/property/>
PREFIX oftype: <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
PREFIX FootballPlayer: <http://dbpedia.org/class/yago/FootballPlayer110101634>
PREFIX SoccerPlayer: <http://dbpedia.org/ontology/SoccerPlayer>
SELECT ?Player ?FootballClub (max(?NationalGoals) as ?NationalGoals)
FROM <http://dbpedia.org>
WHERE
{
  ?Player property:goals ?FootballGoals.
  ?Player property:nationalgoals ?NationalGoals.
  ?Player property:currentclub ?FootballClub.
  ?Player oftype: FootballPlayer:.
  ?Player oftype: SoccerPlayer:.
  FILTER (?NationalGoals > 44 && ?NationalGoals < 1000)
}ORDER BY DESC(?NationalGoals)
```

Return Variables

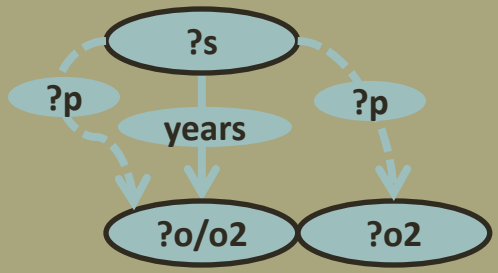
Triplestore

Condition

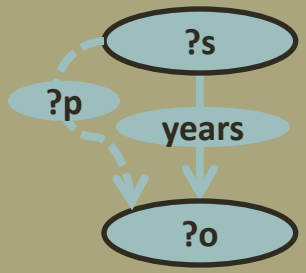
SPARQL Examples

- SPARQL Databases:
  - http://dbpedia.org/sparql
  - http://lod.openlinksw.com/sparql/

Find all predicates that come from a node that is the subject for the predicate “years”



Find all predicates that connect nodes that are also connected by the predicate “years”

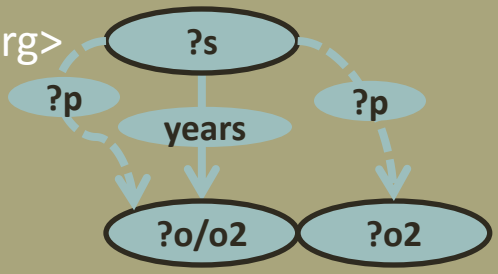


SPARQL Examples

- SPARQL Databases:
  - http://dbpedia.org/sparql
  - http://lod.openlinksw.com/sparql/

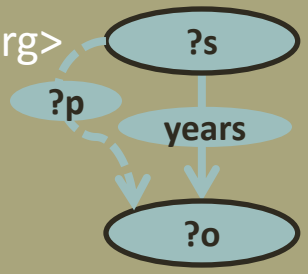
Find all predicates that come from a node that is the subject for the predicate “years”

```
PREFIX property: <http://dbpedia.org/property/>
SELECT DISTINCT ?p
FROM <http://dbpedia.org>
WHERE {
?s property:years  ?o.
?s ?p ?o2
} limit 10
```



Find all predicates that connect nodes that are also connected by the predicate “years”

```
PREFIX property: <http://dbpedia.org/property/>
SELECT DISTINCT ?p
FROM <http://dbpedia.org>
WHERE
{?s property:years  ?o .
?s ?p ?o
} limit 10
```



SPARQL Examples

- SPARQL Databases:
  - http://lod.openlinksw.com/sparql/

```
# Get Tim Berners-Lee's email address
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT DISTINCT ?name ?email
WHERE {
?person a foaf:Person .
?person foaf:name ?name .
?person foaf:mbox ?email .
FILTER regex(?name, "Tim")}
```

SPARQL Examples (6)

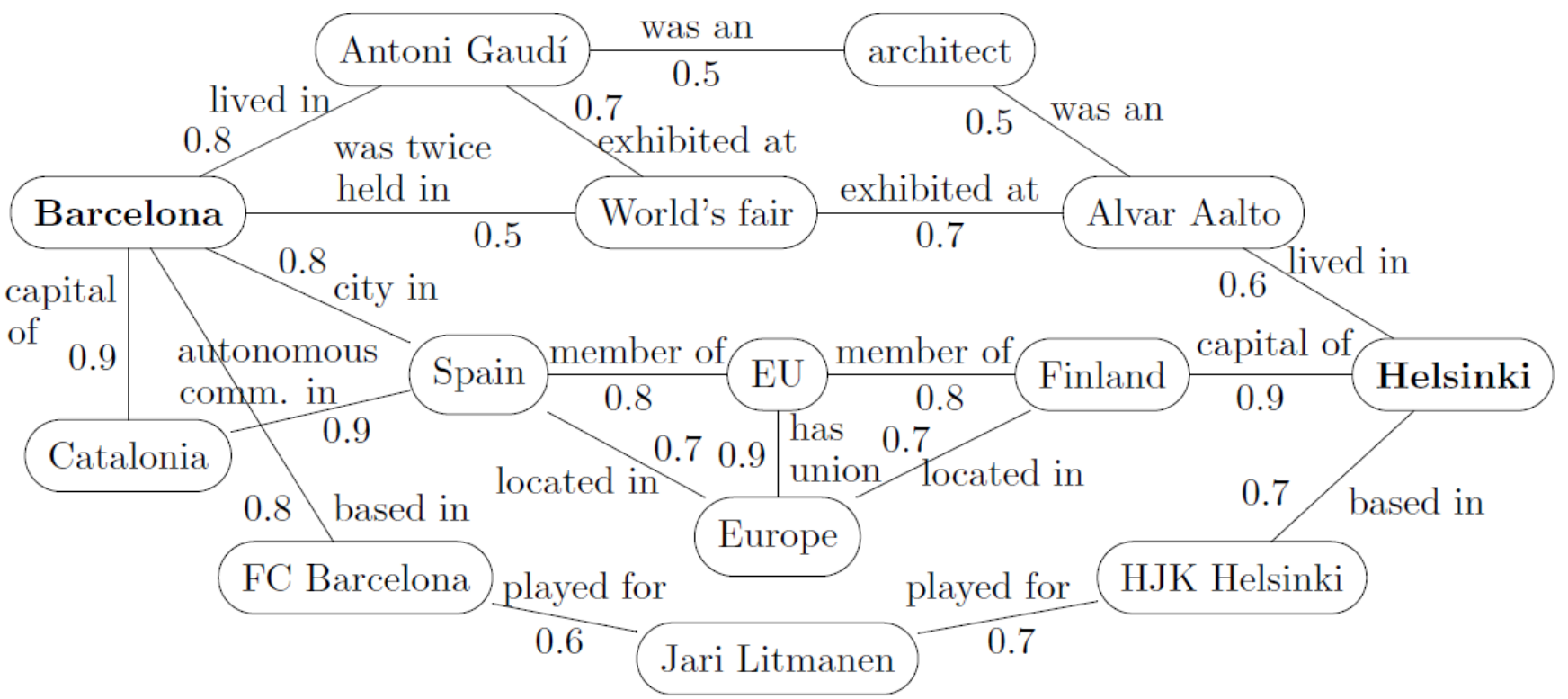
- SPARQL Database:
  - http://librdf.org/query/

Set the RDF (database):
http://www.daml.org/2003/01/periodictable/PeriodicTable.owl

```
# Get data on noble gases
PREFIX table: <http://www.daml.org/2003/01/periodictable/PeriodicTable#>
PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
SELECT ?name ?symbol ?weight ?number
WHERE {
 ?element table:group ?group .
 ?group table:name "Noble gas"^^xsd:string .
 ?element table:name ?name .
 ?element table:symbol ?symbol .
 ?element table:atomicWeight ?weight .
 ?element table:atomicNumber ?number
}
ORDER BY ASC(?name)
```

No triplestore!
The triplestore is implied.

Future of SPARQL: Semantic Web with Weighted Edges?



SPARQL Exercise 0

- Review the following slides for the SPARQL Quiz

SPARQL Exercise 1

1. You have the following triple store <http://ImaginaryTripleStore.com>:
A X B
C X B
B X D
B Y E
F Y B
F X D
F Y E
E X D
G X E

2. You have the following SPARQL statement:
SELECT ?x
FROM
<http://ImaginaryTripleStore.com>
WHERE
{F ?p ?o .
?o ?p ?x}

3. Solution

3.1 List all items that fulfil "F ?p ?o". Place them below the where clause.

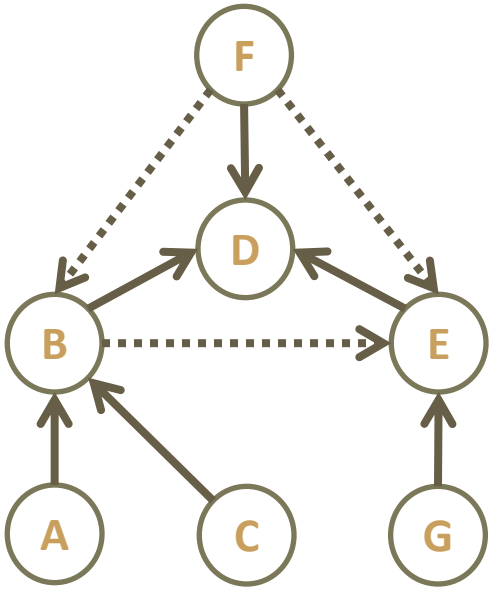
```
F  ?p  ?o  .  ?o  ?p   ?x
F   Y   B
F   X   D
F   Y   E
```

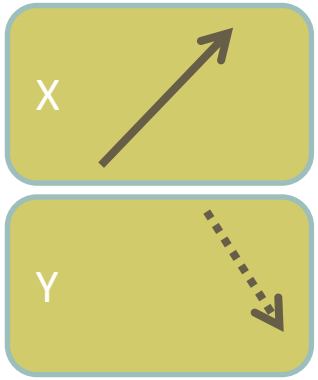
3.2 List all items that fulfil "?o ?p ?x" where ?o is B, D, or E and ?p is Y. Place them below the where clause.

```
F  ?p  ?o  .  ?o  ?p   ?x
F   Y   B  .  B   Y    E
F   X   D
F   Y   E
```

3.3 List all items that were placed under "?x"

SPARQL Exercise 1

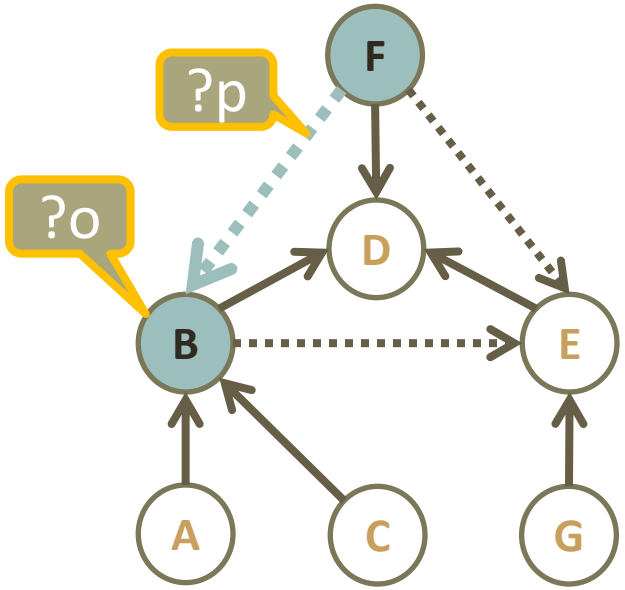


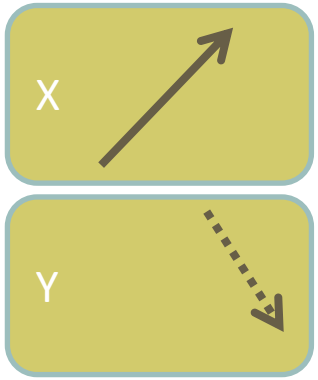


2. You have the following SPARQL statement:

```
SELECT ?x
FROM
<http://ImaginaryTripleStore.com>
WHERE
{F ?p ?o .
?o ?p ?x}
```

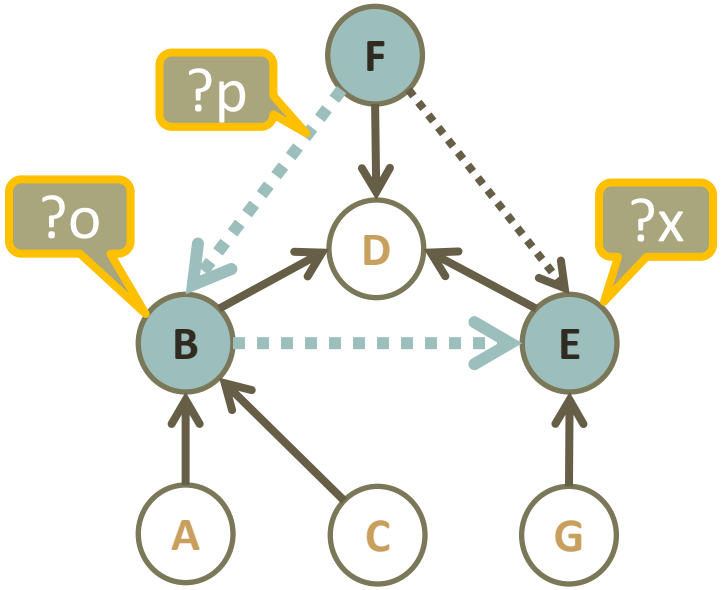
SPARQL Exercise 1

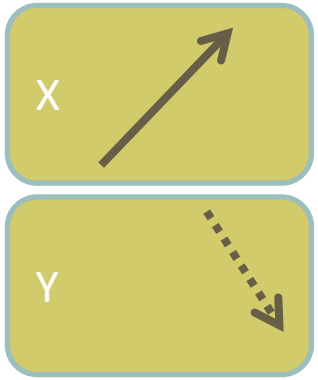




2. You have the following SPARQL statement:
SELECT ?x
FROM
<http://ImaginaryTripleStore.com>
WHERE
{F ?p ?o .
?o ?p ?x}

SPARQL Exercise 1





2. You have the following SPARQL statement:
SELECT ?x
FROM
<http://ImaginaryTripleStore.com>
WHERE
{F ?p ?o .
?o ?p ?x}

SPARQL Exercise 2

1. You have the following triple store <http://ImaginaryTripleStore.com>:
A X B
C X B
B X D
B Y E
F Y B
F X D
F Y E
E X D
G X E

2. You have the following SPARQL statement:
SELECT ?s
FROM
<http://ImaginaryTripleStore.com>
WHERE
{?s Y ?o .
?o X D}

3. Solution

3.1 Look at the 2nd triple from the where clause first. Find all triplets that use X as a predicate where the object is D (fulfils "?o X D"). Place them below the where clause:

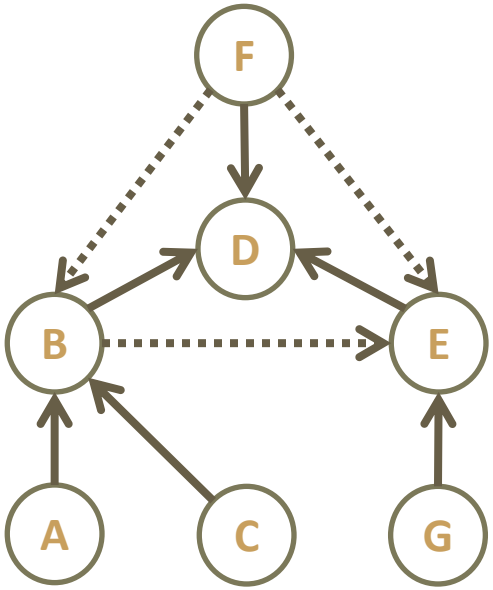
```
?s  Y  ?o  .  ?o  X  D
           B   X  D
           F   X  D
           E   X  D
```

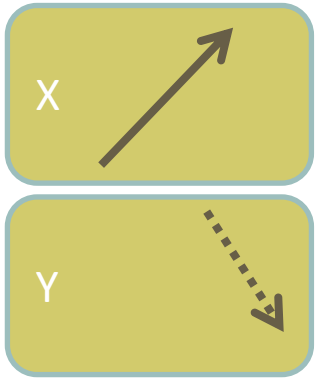
3.2 The subject of the above triplets are: B, F, and E. Now, look at the first triplet of the where clause. Look for all triplets that have an object that is B, F, or E and that have a predicate Y (fulfils "?s Y ?o"). Place those triplets below the where clause:

```
?s  Y  ?o  .  ?o  X  D
F   Y   B     B   X  D
              F   X  D
F   Y   E     E   X  D
B   Y   E     E   X  D
```

3.3 List all items that were placed under "?s"

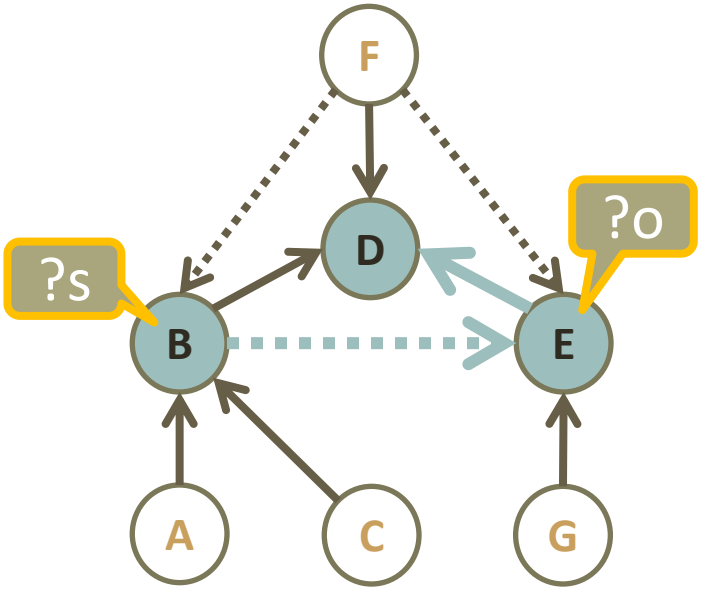
SPARQL Exercise 2

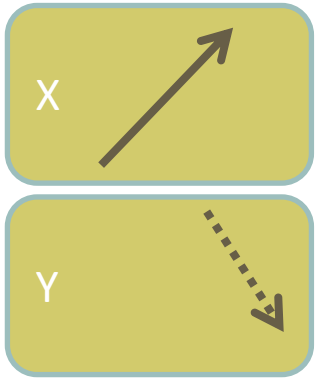




2. You have the following SPARQL statement:
SELECT ?s
FROM
<http://ImaginaryTripleStore.com>
WHERE
{?s Y ?o .
?o X D}

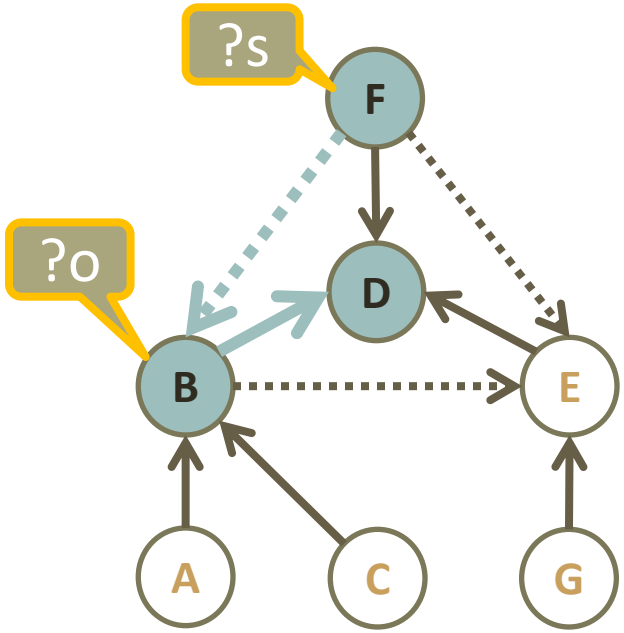
SPARQL Exercise 2

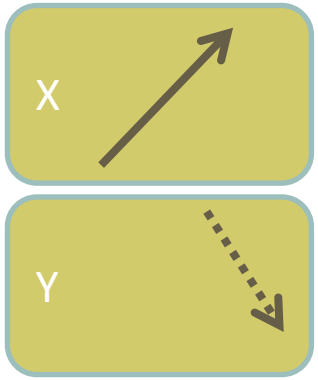




2. You have the following SPARQL statement:
SELECT ?s
FROM
<http://ImaginaryTripleStore.com>
WHERE
{?s Y ?o .
?o X D}

SPARQL Exercise 2





2. You have the following SPARQL statement:
SELECT ?s
FROM
<http://ImaginaryTripleStore.com>
WHERE
{?s Y ?o .
?o X D}

SPARQL Examples

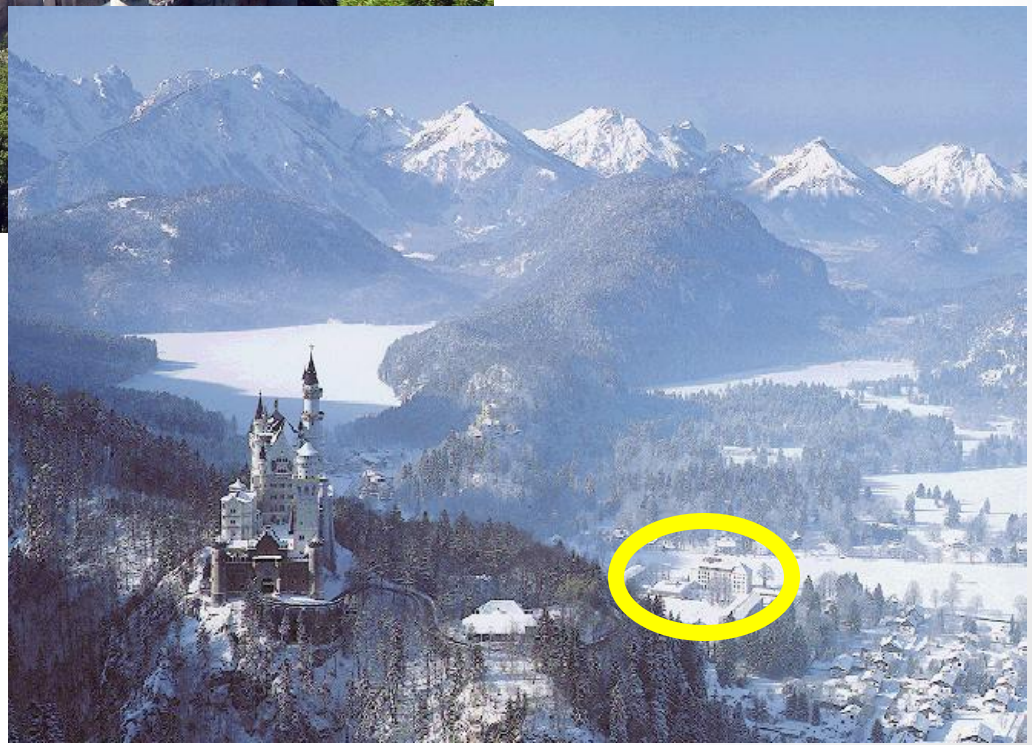
Assignment

Assignment

1. Comment in our LinkedIn Group on a discussion about statistics. Copy the post to a text file called post.txt. This assignment item is similar to last week's assignment.
2. Submit to Canvas by Saturday 11:57 PM post.txt.
3. Look at the SPARQL Exercises and then take the SPARQL Quiz by Saturday 11:57 PM. It is a difficult quiz you may need an hour.
4. Stats
   a) Look at Verzani 's book: https://cran.r-project.org/doc/contrib/Verzani-SimpleR.pdf
   b) Play with theses functions in R and figure out their relationships: rnorn, dnorm, pnorm, qnorm (Nothing to submit for this item)

Assignment

Summer's Vacation Spot



Thank You!

Introduction to Data Science