

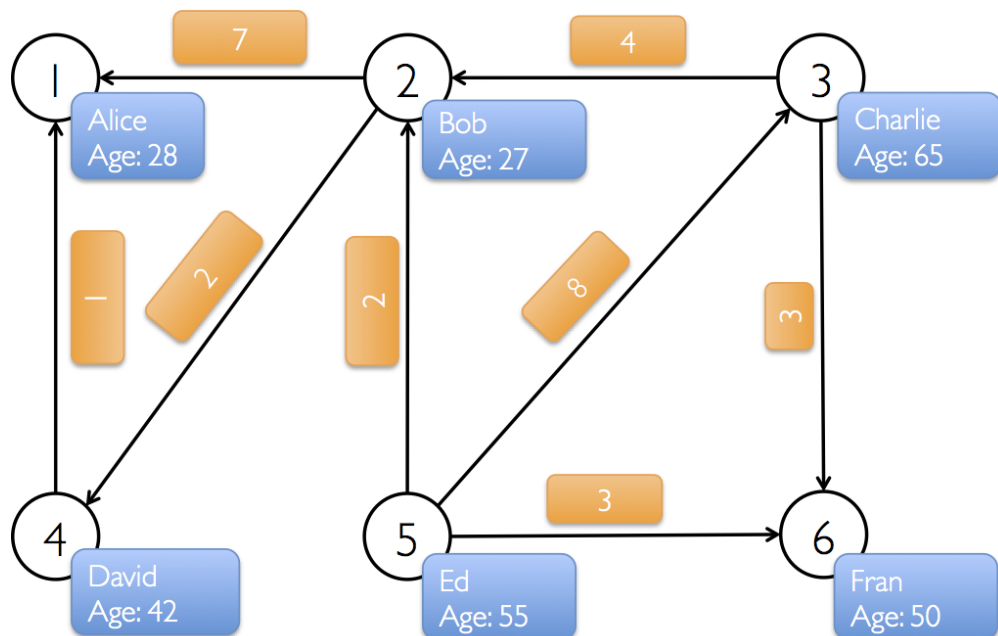
## Review Test Submission: GraphX Quiz

User	Brah
Course	CS 6350.002 - Big Data Management and Analytics - S22
Test	GraphX Quiz
Started	4/10/22 12:53 PM
Submitted	4/10/22 7:33 PM
Due Date	4/10/22 11:59 PM
Status	Completed
Attempt Score	100 out of 100 points
Time Elapsed	6 hours, 40 minutes
Results Displayed	All Answers, Submitted Answers, Correct Answers

### Question 1

10 out of 10 points

Consider the social network shown below:



Each vertex has name and age as the properties and the edge has the number of likes as the property.

Which of the following is the correct command to load this in a GraphFrame

Selected Answer:

```
from graphframes import *
```

```
# Vertex DataFrame
v = sqlContext.createDataFrame([
    (1, "Alice", 28),
    (2, "Bob", 27),
    (3, "Charlie", 65),
    (4, "David", 42),
    (5, "Ed", 55),
    (6, "Fran", 50)
], ["id", "name", "age"])
```

```
# Edge DataFrame
e = sqlContext.createDataFrame([
    (2, 1, 7),
    (2, 4, 2),
    (3, 2, 4),
    (3, 6, 3),
    (4, 1, 1),
    (5, 2, 2),
    (5, 3, 8),
    (5, 6, 3)
], ["src", "dst", "likes"])
```

```
# Create a GraphFrame
g = GraphFrame(v, e)
```

Answers:

```
from graphframes import *
```

```
# Vertex DataFrame
v = sqlContext.createDataFrame([
    (1, "Alice", 28),
    (2, "Bob", 27),
    (3, "Charlie", 65),
    (4, "David", 42),
    (5, "Ed", 55),
    (6, "Fran", 50)
], ["id", "name", "age"])
```

```
# Edge DataFrame
e = sqlContext.createDataFrame([
    (2, 1, 7),
    (2, 4, 2),
    (3, 2, 4),
    (3, 6, 3),
    (4, 1, 1),
    (5, 2, 2),
    (5, 3, 8),
    (5, 6, 3)
], ["src", "dst", "likes"])
```

```
# Create a GraphFrame
g = GraphFrame(v, e)
```

```
from graphframes import *
```

```
# Vertex DataFrame
```

```
v = sqlContext.createDataFrame([
    (1, "Alice", 28),
    (2, "Bob", 27),
    (3, "Charlie", 65),
    (4, "David", 42),
    (5, "Ed", 55),
    (6, "Fran", 50)
], ["id", "name", "age"])
```

```
# Edge DataFrame
```

```
e = sqlContext.createDataFrame([
    (2, 1, 7),
    (2, 4, 2),
    (3, 2, 4),
    (3, 6, 3),
    (4, 1, 1),
    (5, 2, 2),
    (5, 3, 8),
    (5, 6, 3)
], ["src", "dst", "likes"])
```

```
# Create a GraphFrame
```

```
g = Graph(v, e)
```

```
from graphframes import *
```

```
# Vertex DataFrame
```

```
v = sc.parallelize(Array[
    (1, "Alice", 28),
    (2, "Bob", 27),
    (3, "Charlie", 65),
    (4, "David", 42),
    (5, "Ed", 55),
    (6, "Fran", 50)
], ["id", "name", "age"])
```

```
# Edge DataFrame
```

```
e = sc.parallelize(Array[
    (2, 1, 7),
    (2, 4, 2),
    (3, 2, 4),
    (3, 6, 3),
    (4, 1, 1),
    (5, 2, 2),
    (5, 3, 8),
    (5, 6, 3)
], ["src", "dst", "likes"])
```

```
# Create a GraphFrame
```

```
g = GraphFrame(v, e)
```

```

from graphframes import *
# Vertex DataFrame
v = sc.parallelize([
    (1, "Alice", 28),
    (2, "Bob", 27),
    (3, "Charlie", 65),
    (4, "David", 42),
    (5, "Ed", 55),
    (6, "Fran", 50)
], ["id", "name", "age"])
# Edge DataFrame
e = sc.parallelize([
    (2, 1, 7),
    (2, 4, 2),
    (3, 2, 4),
    (3, 6, 3),
    (4, 1, 1),
    (5, 2, 2),
    (5, 3, 8),
    (5, 6, 3)
], ["src", "dst", "likes"])
# Create a GraphFrame
g = GraphFrame(v, e)

```

## Question 2

10 out of 10 points

Using the graph in question 1, which command will display the users who are more than 50 years old using the GraphFrame approach

Selected Answer: ☒ `g.vertices.filter("age > 50").show()`

Answers: ☒ `g.vertices.filter("age > 50").show()`

`g.filter("age > 50").show()`

`g.vertices.filter("age" > 50).show()`

`g.edges.filter("age" > 50).show()`

## Question 3

10 out of 10 points

How many rows will be displayed in the response of question 2:

Selected Answer: ☒ 2

Answers: ☒ 2

1

No rows will be displayed

3

## Question 4


10 out of 10 points

Which command displays the number of triplets in the graph and what is the answer?

Selected Answer:  g.triplets.count()  
Answer: 8

Answers: g.triplets.show.count()  
Answer: 8

g.triplets.count()  
Answer: 4


 g.triplets.count()  
Answer: 8


g.triplets.count()  
Answer: 10

## Question 5

10 out of 10 points

You would like to find out how many people who like Ed's posts also like each other's posts i.e. how many triangles can be formed around Ed? Which command does this and what is the answer:

Selected Answer:  g.triangleCount().filter("name == 'Ed'").show()  
Answer = 2

Answers:  g.triangleCount().filter("name == 'Ed'").show()  
Answer = 2

g.triangleCount().filter("name == 'Ed'").show()  
Answer = 5

g.triangleCount.filter("name == 'Ed'").show()  
Answer = 2



g.triangleCount.filter("name == 'Ed'").show()  
Answer = 5


## Question 6

10 out of 10 points

You would like to create a Dataframe showing the number of likes given by each person with their name (not id) sorted by number of likes in descending order. Which of the following accomplishes this?

You can assume that all relevant libraries have been imported.

Selected   
Answer:  joined = g.edges.join(g.vertices, g.edges.src == g.vertices.id)  
joined.groupBy("name").agg(sum("likes").alias("outLikes")).orderBy(desc("outLikes")).show()

Answers:   
joined = g.edges.join(g.vertices, g.edges.src == g.vertices.id)  
joined.groupBy("name").agg(sum("likes").alias("outLikes")).orderBy(desc("outLikes")).show()

joined = g.edges.join(g.vertices, g.edges.dst == g.vertices.id)  
joined.groupBy("name").sum("likes").sort().show()

joined = g.edges.join(g.vertices, g.edges.src == g.vertices.id)  
joined.groupBy("id").sum("likes").show()

joined = g.edges.join(g.vertices, g.edges.dst == g.vertices.id)  
joined.groupBy("name").agg(sum("likes").alias("outLikes")).orderBy(desc("outLikes")).show()

### Question 7

10 out of 10 points

You would like to compute the PageRank of every page with a maximum of 10 iterations and then display the name and PageRank of each person sorted by PageRank in a descending order. Which of the following accomplishes this?

Selected Answer: ☒ `results = g.pageRank(maxIter = 10)  
results.vertices.orderBy(desc("pagerank")).show()`

Answers: ☐ `results = g.pageRank(maxIter = 10)  
results.edges.orderBy("pagerank").show()`

☒ `results = g.pageRank(maxIter = 10)  
results.vertices.orderBy(desc("pagerank")).show()`

☐ `results = g.pageRank(maxIter = 10)  
results.edges.orderBy(desc("pagerank")).show()`

☐ `results = g.pageRank(maxIter = 10)  
results.orderBy(desc("pagerank")).show()`

### Question 8

10 out of 10 points

You would like to find the number of people that like Bob's posts. Which of the following accomplishes this?

Selected Answer: ☒ `joined = g.edges.join(g.vertices, g.edges.dst == g.vertices.id)  
joined.groupBy("name").count().filter("name == 'Bob'").show()`

Answers: ☐ There is no way this can be done

☒ `joined = g.edges.join(g.vertices, g.edges.dst == g.vertices.id)  
joined.groupBy("name").count().filter("name == 'Bob'").show()`

☐ `joined = g.edges.join(g.vertices, g.edges.dst == g.vertices.id)  
joined.groupBy("src").count().filter("name == 'Bob'").show()`

☐ `joined = g.edges.join(g.vertices, g.edges.src == g.vertices.id)  
joined.groupBy("name").sum().filter("name == 'Bob'").show()`

### Question 9

10 out of 10 points

I would like to find if there are any two people who mutually like each other's posts. Which of the following accomplishes this?

Selected Answer: ☒ `motifs = g.find("(a)-[e]->(b); (b)-[e2]->(a)")  
motifs.show()`

Answers: ☒ `motifs = g.find("(a)-[e]->(b); (b)-[e2]->(a)")  
motifs.show()`

☐ `motifs = g.find("(a)-[e]->(b); (b)-[e2]->(c); (c)-[e2]->(a)")  
motifs.show()`


☐ `motifs = g.find("(a)-[e]->(b); (b)-[!e2]->(a)")  
motifs.show()`


☐ None of the above

## Question 10

10 out of 10 points

I would like to find the shortest directed path distance from every vertex to vertex with id 1. Which of the following accomplishes this?

Selected Answer:  `results = g.shortestPaths(landmarks=["1"])`  
`results.select("id", "distances").show()`

Answers:  `results = g.shortestPaths(landmarks=["1"])`  
`results.select("id", "distances").show()`

`results = g.shortestPaths(from=["1"])`  
`results.select("id", "path").show()`

`results = g.shortestPaths(to=["1"])`  
`results.select("id", "path").show()`

None of the above

Sunday, April 10, 2022 7:33:53 PM CDT

← OK