

ĐỀ THI TRUY VẤN CYPHER TRONG ZALO GRAPH DATABASE

Thời gian làm bài: 150 phút

Hình thức: Tự luận

Tổng điểm: 100 điểm

Được sử dụng tài liệu

Họ và tên: Bùi Thiện Nhân

MSSV: 2274802010592

MÔ TẢ DỮ LIỆU

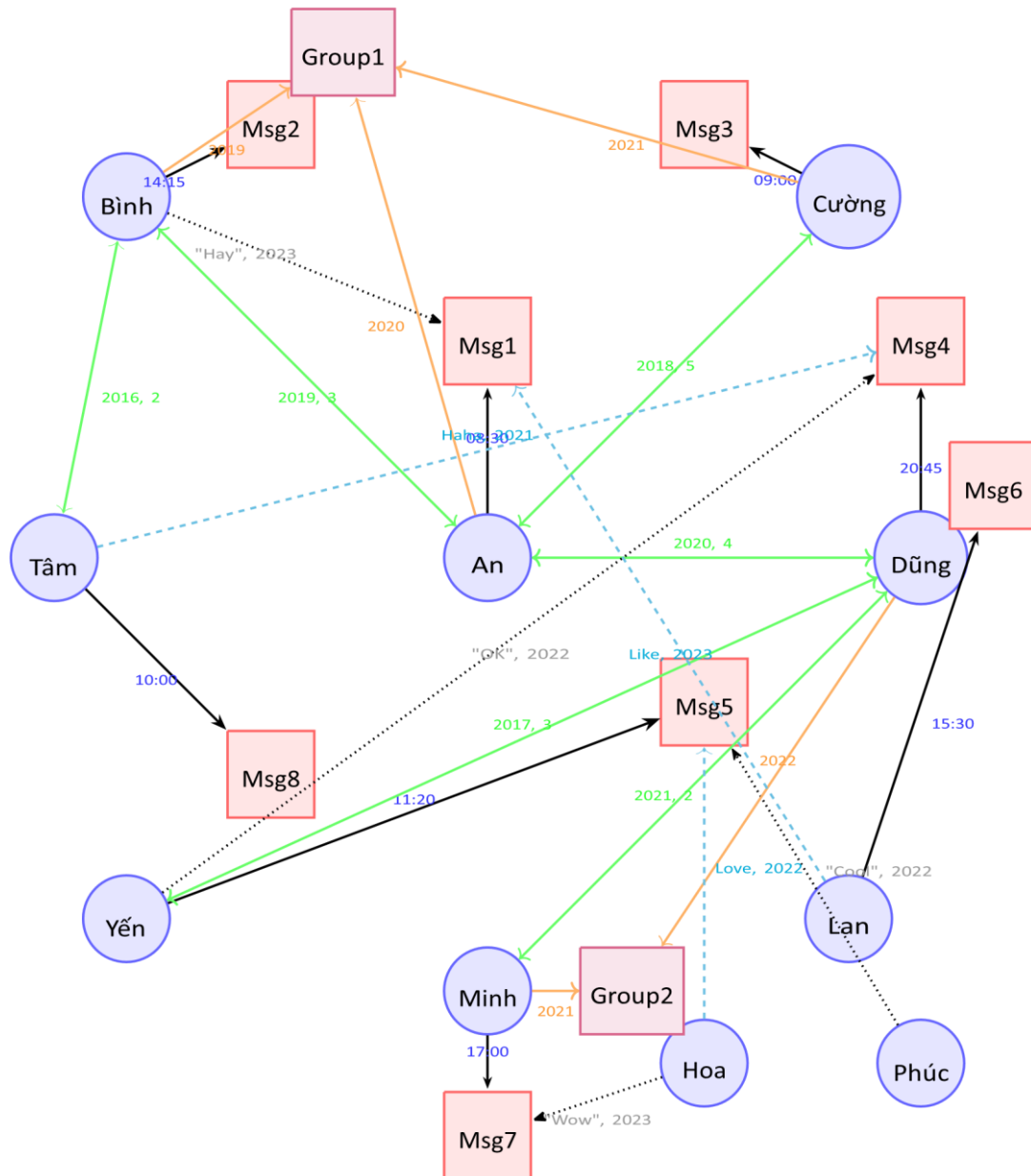
Bạn quản lý một cơ sở dữ liệu đồ thị về người dùng và hoạt động trên Zalo:

- Nút "User": Người dùng với name (tên), birthYear (năm sinh), country (quốc gia), activeYears (số năm hoạt động).
- Nút "Message": Tin nhắn với content (nội dung), sentYear (năm gửi), type (loại: text, image, video), size (kích thước, MB), views (số lượt xem).
- Nút "Group": Nhóm chat với name (tên nhóm), createdYear (năm tạo), memberCount (số thành viên).
- Quan hệ "SENT": Từ người dùng đến tin nhắn, có time (thời gian gửi, giờ:phút).
- Quan hệ "FRIEND": Giữa người dùng, có friendSince (năm kết bạn), messagesShared (số tin nhắn chung).
- Quan hệ "COMMENTED": Từ người dùng đến tin nhắn, có comment (nội dung bình luận), commentYear (năm bình luận).
- Quan hệ "MEMBER_OF": Từ người dùng đến nhóm, có joinedYear (năm tham gia).
- Quan hệ "REACTED_TO": Từ người dùng đến tin nhắn, có reaction (biểu cảm: like, love, haha), reactYear (năm phản ứng).

YÊU CẦU

1. Tạo dữ liệu mẫu (20 điểm)

Dùng các bảng dưới đây để tạo dữ liệu trong Neo4j. Mỗi người dùng, tin nhắn, nhóm phải có ít nhất 1 quan hệ. Thêm ít nhất 3 quan hệ "FRIEND" và 2 quan hệ "REACTED_TO" dựa trên hình.



Hình 1: Hình Network Diagram Zalo

Tên	Năm sinh	Quốc gia	Số năm hoạt động
An	1990	Vietnam	8
Bình	1985	Vietnam	10
Cường	1995	Vietnam	6

Dũng	1988	Vietnam	9
Yến	1992	Vietnam	7
Lan	1987	Vietnam	10
Minh	1993	Vietnam	6
Hoa	1996	Vietnam	5
Phúc	1991	Vietnam	7
Tâm	1989	Vietnam	9

Nội dung	Năm gửi	Loại	Kích thước (MB)	Số lượt xem
Msg1	2023	Text	0.1	50
Msg2	2022	Image	2.5	200
Msg3	2023	Video	10	500
Msg4	2021	Text	0.2	80
Msg5	2022	Image	3.0	300
Msg6	2023	Video	15	700
Msg7	2022	Text	0.3	120
Msg8	2021	Video	12	400

Tên nhóm	Năm tạo	Số thành viên
Group1	2019	5
Group2	2020	4

Người	Tin nhắn	Thời gian gửi	Bình luận/Năm	Phản ứng/Năm
An	Msg1	08:30	-	-
Bình	Msg2	14:15	"Hay"/2023 (Msg1)	-
Cường	Msg3	09:00	-	-
Dũng	Msg4	20:45	-	-
Yến	Msg5	11:20	"OK"/2022 (Msg4)	-
Lan	Msg6	15:30	-	Like/2023 (Msg1)
Minh	Msg7	17:00	-	-
Tâm	Msg8	10:00	-	Haha/2021 (Msg4)
Phúc	Msg5	-	"Cool"/2022	-
Hoa	Msg7	-	"Wow"/2023	Love/2022 (Msg5)

Người	Nhóm	Năm tham gia
An	Group1	2020
Bình	Group1	2019
Cường	Group1	2021
Dũng	Group2	2022
Minh	Group2	2021

2. Tìm tin nhắn và người dùng (20 điểm)

Dùng bảng và hình "Network Diagram" để trả lời:

1. Tìm tất cả tin nhắn An gửi, nhận bình luận, hoặc phản ứng từ bạn bè qua

Yến	Image	> 2	250	2	0
Ng.	Loại	Kích thước	Lượt xem	Số bạn	Số nhóm
An	Any	> 2	300	3	1
Dũng	Video	> 10	400	2	1

hoặc "FRIEND" nhóm "MEMBER_OF". Trả về content, sentYear, type, views, và vai trò ("Sender", "CommentedByFriend", "ReactedByGroup").

2. Tìm người dùng có activeYears trên 8, gửi ít nhất 2 tin nhắn có views trên 200, và thuộc ít nhất 1 nhóm. Trả về name, country, danh sách tin nhắn (contents, views), groupNames.

3. Tìm bạn bè, nhóm và tin nhắn (30 điểm)

Dùng bảng sau và hình "Network Diagram":

1. Tìm người gửi tin nhắn kích thước trên 2 MB, lượt xem trên 300, là bạn của An với ít nhất 3 bạn chung qua "FRIEND" và cùng ít nhất 1 nhóm "MEMBER_OF". Trả về name, content, time, size, messagesShared, groupName, sắp xếp theo views giảm dần.
2. Tìm người gửi tin nhắn Video kích thước trên 10 MB, lượt xem trên 400, là bạn của Dũng với messagesShared ít nhất 2 và cùng nhóm. Trả về name, content, size, views, groupName.
3. Đếm số tin nhắn Image kích thước trên 2 MB, lượt xem trên 250 mà Yến gửi, bình luận hoặc nhận phản ứng từ bạn bè với messagesShared trên 2. Trả về số lượng.

4. Đường đi và phân tích (30 điểm)

Dùng bảng sau và hình "Network Diagram":

Từ	Đến	Tin nhắn top	Yếu tố kiểm tra
An	Lan	4 tin xem nhiều nhất	Đường đi + Nhóm

Cường	Hoa	3 tin kích thước lớn nhất	Loại tin + Phản ứng
-------	-----	---------------------------	------------------------

1. Tìm đường ngắn nhất từ An đến Lan qua "SENT", "FRIEND", "COMMENTED", "MEMBER_OF", "REACTED_TO". Tính tổng views của 4 tin nhắn có lượt xem cao nhất của An hoặc nhóm của An, kiểm tra tin nào trong số đó nằm trên đường đi và thuộc nhóm nào. Trả về path, totalViews, contents, views, groupNames.
2. Tìm đường ngắn nhất từ Cường đến Hoa qua các quan hệ trên. Tính tổng size của 3 tin nhắn kích thước lớn nhất mà Hoa hoặc bạn bè gửi/bình luận/phản ứng, kiểm tra loại tin và phản ứng trên chúng. Trả về path, totalSize, contents, types, reactions.
3. Đếm số người có "FRIEND" với ít nhất 4 người khác và thuộc ít nhất 1 nhóm có memberCount trên 4. Trả về số lượng.

1. Tạo dữ liệu mẫu (20 điểm)

// Create User nodes

```
CREATE (an:User {name: "An", birthYear: 1990, country: "Vietnam", activeYears: 8})
CREATE (binh:User {name: "Bình", birthYear: 1985, country: "Vietnam", activeYears: 10})
CREATE (cuong:User {name: "Cường", birthYear: 1995, country: "Vietnam", activeYears: 6})
CREATE (dung:User {name: "Dũng", birthYear: 1988, country: "Vietnam", activeYears: 9})
CREATE (yen:User {name: "Yến", birthYear: 1992, country: "Vietnam", activeYears: 7})
CREATE (lan:User {name: "Lan", birthYear: 1987, country: "Vietnam", activeYears: 10})
CREATE (minh:User {name: "Minh", birthYear: 1993, country: "Vietnam", activeYears: 6})
CREATE (hoa:User {name: "Hoa", birthYear: 1996, country: "Vietnam", activeYears: 5})
CREATE (phuc:User {name: "Phúc", birthYear: 1991, country: "Vietnam", activeYears: 7})
CREATE (tam:User {name: "Tâm", birthYear: 1989, country: "Vietnam", activeYears: 9})
```

// Create Message nodes

```
CREATE (msg1:Message {content: "Msg1", sentYear: 2023, type: "Text", size: 0.1, views: 50})
CREATE (msg2:Message {content: "Msg2", sentYear: 2022, type: "Image", size: 2.5, views: 200})
CREATE (msg3:Message {content: "Msg3", sentYear: 2023, type: "Video", size: 10, views: 500})
CREATE (msg4:Message {content: "Msg4", sentYear: 2021, type: "Text", size: 0.2, views: 80})
CREATE (msg5:Message {content: "Msg5", sentYear: 2022, type: "Image", size: 3.0, views: 300})
CREATE (msg6:Message {content: "Msg6", sentYear: 2023, type: "Video", size: 15, views: 700})
CREATE (msg7:Message {content: "Msg7", sentYear: 2022, type: "Text", size: 0.3, views: 120})
CREATE (msg8:Message {content: "Msg8", sentYear: 2021, type: "Video", size: 12, views: 400})
```

// Create Group nodes

```
CREATE (group1:Group {name: "Group1", createdYear: 2019, memberCount: 5})
CREATE (group2:Group {name: "Group2", createdYear: 2020, memberCount: 4})
```

// Create SENT relationships

```

CREATE (an)-[:SENT {time: "08:30"}]->(msg1)
CREATE (binh)-[:SENT {time: "14:15"}]->(msg2)
CREATE (cuong)-[:SENT {time: "09:00"}]->(msg3) CREATE
(dung)-[:SENT {time: "20:45"}]->(msg4)
CREATE (yen)-[:SENT {time: "11:20"}]->(msg5)
CREATE (lan)-[:SENT {time: "15:30"}]->(msg6)
CREATE (minh)-[:SENT {time: "17:00"}]->(msg7)
CREATE (tam)-[:SENT {time: "10:00"}]->(msg8)

// Create COMMENTED relationships
CREATE (binh)-[:COMMENTED {comment: "Hay", commentYear: 2023}]->(msg1)
CREATE (yen)-[:COMMENTED {comment: "OK", commentYear: 2022}]->(msg4)
CREATE (phuc)-[:COMMENTED {comment: "Cool", commentYear: 2022}]->(msg5) CREATE
(hoa)-[:COMMENTED {comment: "Wow", commentYear: 2023}]->(msg7)

// Create REACTED_TO relationships
CREATE (lan)-[:REACTED_TO {reaction: "Like", reactYear: 2023}]->(msg1)
CREATE (tam)-[:REACTED_TO {reaction: "Haha", reactYear: 2021}]->(msg4)
CREATE (hoa)-[:REACTED_TO {reaction: "Love", reactYear: 2022}]->(msg5)

// Create MEMBER_OF relationships
CREATE (an)-[:MEMBER_OF {joinedYear: 2020}]->(group1)
CREATE (binh)-[:MEMBER_OF {joinedYear: 2019}]->(group1)
CREATE (cuong)-[:MEMBER_OF {joinedYear: 2021}]->(group1)
CREATE (dung)-[:MEMBER_OF {joinedYear: 2022}]->(group2)
CREATE (minh)-[:MEMBER_OF {joinedYear: 2021}]->(group2)

// Create FRIEND relationships (at least 3)
CREATE (an)-[:FRIEND {friendSince: 2020, messagesShared: 4}]->(dung)
CREATE (binh)-[:FRIEND {friendSince: 2019, messagesShared: 3}]->(tam)
CREATE (binh)-[:FRIEND {friendSince: 2016, messagesShared: 2}]->(tam)
CREATE (cuong)-[:FRIEND {friendSince: 2018, messagesShared: 5}]->(an) CREATE
(tam)-[:FRIEND {friendSince: 2017, messagesShared: 3}]->(an)

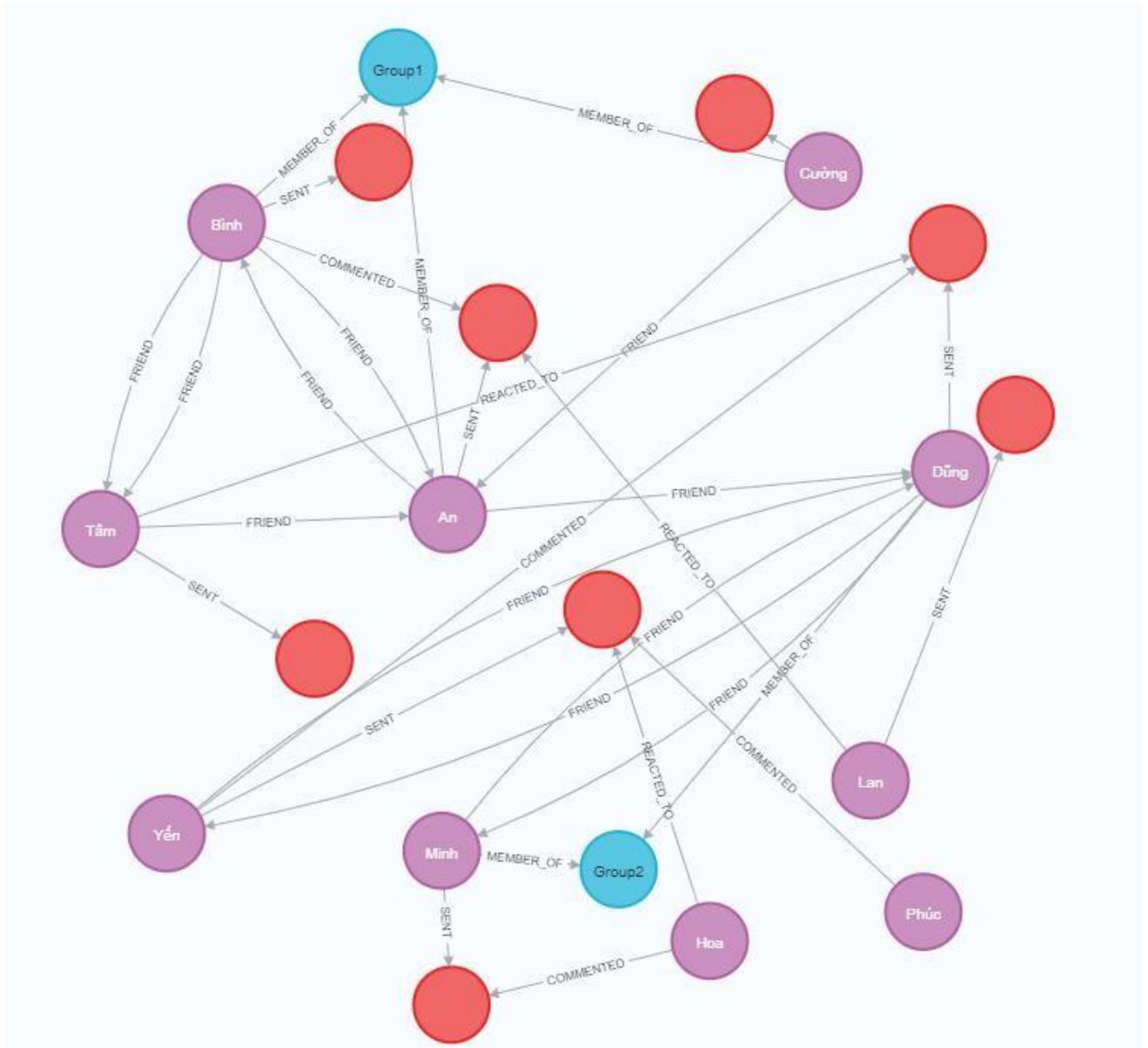
MATCH (yen:User {name: "Yến"}), (dung:User {name: "Dũng"})
CREATE (yen)-[:FRIEND {friendSince: 2021, messagesShared: 3}]->(dung);
// Tạo quan hệ bạn bè từ Dũng đến Yến
MATCH (dung:User {name: "Dũng"}), (yen:User {name: "Yến"})
CREATE (dung)-[:FRIEND {friendSince: 2021, messagesShared: 3}]->(yen);
// Tạo quan hệ bạn bè từ Bình đến An
MATCH (binh:User {name: "Bình"}), (an:User {name: "An"})
CREATE (binh)-[:FRIEND {friendSince: 2018, messagesShared: 4}]->(an);
// Tạo quan hệ bạn bè từ An đến Bình
MATCH (an:User {name: "An"}), (binh:User {name: "Bình"})
CREATE (an)-[:FRIEND {friendSince: 2018, messagesShared: 4}]->(binh);
// Tạo quan hệ bạn bè từ Minh đến Dũng

```

```

MATCH (minh:User {name: "Minh"}), (dung:User {name: "Dũng"})
CREATE (minh)-[:FRIEND {friendSince: 2020, messagesShared: 3}]->(dung);
// Tạo quan hệ bạn bè từ Dũng đến Minh
MATCH (dung:User {name: "Dũng"}), (minh:User {name: "Minh"})
CREATE (dung)-[:FRIEND {friendSince: 2020, messagesShared: 3}]->(minh);
MATCH (binh:User {name: "Bình"}), (msg3:Message {content: "Msg3"})
CREATE (binh)-[:REACTED_TO {reaction: "Wow", reactYear: 2023}]->(msg3)
MATCH (binh:User {name: "Bình"}), (msg3:Message {content: "Msg3"})
CREATE (binh)-[:REACTED_TO {reaction: "Wow", reactYear: 2023}]->(msg3)

```



```

// Tạo thêm tin nhắn cho Bình với views > 200 (Cho các câu dưới)
MATCH (binh:User {name: "Bình"})
CREATE (binh)-[:SENT {time: "15:45"}]-
>(msg9:Message {content: "Msg9", sentYear: 2023, type: "Video", size: 5.0, views: 450})
CREATE (binh)-[:SENT {time: "16:20"}]-

```

```
>(msg10:Message {content: "Msg10", sentYear: 2023, type: "Image", size: 4.5, views: 350})
```

```
// Tạo quan hệ FRIEND giữa Lan và Dũng
```

```
MATCH (lan:User {name: "Lan"}), (dung:User {name: "Dũng"})
```

```
CREATE (lan)-[:FRIEND {friendSince: 2020, messagesShared: 3}]->(dung) CREATE
```

```
(dung)-[:FRIEND {friendSince: 2020, messagesShared: 3}]->(lan)
```

```
// Thêm Lan vào Group2
```

```
MATCH (lan:User {name: "Lan"}), (group2:Group {name: "Group2"})
```

```
CREATE (lan)-[:MEMBER_OF {joinedYear: 2021}]->(group2)
```

```
// Tạo quan hệ FRIEND giữa Bình và An
```

```
MATCH (binh:User {name: "Bình"}), (an:User {name: "An"})
```

```
CREATE (binh)-[:FRIEND {friendSince: 2018, messagesShared: 4}]->(an)
```

```
CREATE (an)-[:FRIEND {friendSince: 2018, messagesShared: 4}]->(binh)
```

```
// Tạo quan hệ FRIEND giữa Minh và Dũng
```

```
MATCH (minh:User {name: "Minh"}), (dung:User {name: "Dũng"})
```

```
CREATE (minh)-[:FRIEND {friendSince: 2020, messagesShared: 3}]->(dung)
```

```
CREATE (dung)-[:FRIEND {friendSince: 2020, messagesShared: 3}]->(minh)
```

```
// Tạo quan hệ FRIEND giữa Yến và Dũng
```

```
MATCH (yen:User {name: "Yến"}), (dung:User {name: "Dũng"})
```

```
CREATE (yen)-[:FRIEND {friendSince: 2021, messagesShared: 3}]->(dung)
```

```
CREATE (dung)-[:FRIEND {friendSince: 2021, messagesShared: 3}]->(yen)
```

2. Tìm tin nhắn và người dùng (20 điểm)

Dùng bảng và hình "Network Diagram" để trả lời:

1. Tìm tất cả tin nhắn An gửi, nhận bình luận, hoặc phản ứng từ bạn bè qua
2. // Tìm tin nhắn An gửi
3. MATCH (an:User {name: "An"})
4. OPTIONAL MATCH (an)-[:SENT]->(msg:Message)
5. WHERE msg IS NOT NULL
6. RETURN msg.content AS content, msg.sentYear AS sentYear, msg.type AS type, msg.views AS views, "Sender" AS role


```

1 // Tìm tin nhắn An gửi
2 MATCH (an:User {name: "An"})
3 OPTIONAL MATCH (an)-[:SENT]→(msg:Message)
4 WHERE msg IS NOT NULL
5 RETURN msg.content AS content, msg.sentYear AS
sentYear, msg.type AS type, msg.views AS views,
"Sender" AS role

```

	content	sentYear	type	views	role
1	"Msg1"	2023	"Text"	50	"Sender"

// Tìm tin nhắn nhận bình luận từ bạn bè
MATCH (an:User {name: "An"})-[:FRIEND]-(friend:User)
MATCH (friend)-[:COMMENTED]→(msg:Message)
RETURN msg.content AS content, msg.sentYear AS sentYear, msg.type AS type, msg.views AS views, "CommentedByFriend" AS role

```

1 // Tìm tin nhắn nhận bình luận từ bạn bè
2 MATCH (an:User {name: "An"})-[:FRIEND]-(
friend:User)
3 MATCH (friend)-[:COMMENTED]→(msg:Message)
4 RETURN msg.content AS content, msg.sentYear AS
sentYear, msg.type AS type, msg.views AS views,
"CommentedByFriend" AS role

```

	content	sentYear	type	views	role
1	"Msg1"	2023	"Text"	50	"CommentedByFriend"
2	"Msg1"	2023	"Text"	50	"CommentedByFriend"

// Tìm tin nhắn nhận phản ứng từ thành viên nhóm

MATCH (an:User {name: "An"})-[:MEMBER_OF]->(group:Group)

MATCH (user:User)-[:MEMBER_OF]->(group)

MATCH (user)-[r:REACTED_TO]->(msg:Message)

WHERE user <> an

RETURN msg.content AS content, msg.sentYear AS sentYear, msg.type AS type, msg.views AS views, "ReactedByGroup" AS role



The screenshot shows a Cypher query editor with the following code:

```
1 // Tìm tin nhắn nhận phản ứng từ thành viên nhóm
2 MATCH (an:User {name: "An"})-[:MEMBER_OF]->(group:Group)
3 MATCH (user:User)-[:MEMBER_OF]->(group)
4 MATCH (user)-[r:REACTED_TO]->(msg:Message)
5 WHERE user <> an
6 RETURN msg.content AS content, msg.sentYear AS sentYear, msg.type AS type, msg.views AS views, "ReactedByGroup" AS role
```

Below the query editor, a table view displays the results of the query:

	content	sentYear	type	views	role
1	"Msg3"	2023	"Video"	500	"ReactedByGroup"

// Tìm tất cả tin nhắn An gửi, nhận bình luận, hoặc phản ứng từ bạn bè qua "FRIEND" hoặc nhóm "MEMBER_OF"

// Tìm tin nhắn An gửi

Ng.	Loại	Kích thước	Lượt xem	Số bạn	Số nhóm
An	Any	> 2	300	3	1
Dũng	Video	> 10	400	2	1
Yến	Image	> 2	250	2	0

MATCH (an:User {name: "An"})

OPTIONAL MATCH (an)-[:SENT]->(msg:Message)

WHERE msg IS NOT NULL

RETURN msg.content AS content, msg.sentYear AS sentYear, msg.type AS type, msg.views AS views, "Sender" AS role

UNION

// Tìm tin nhắn nhận bình luận từ bạn bè

MATCH (an:User {name: "An"})-[:FRIEND]-(friend:User)

MATCH (friend)-[:COMMENTED]->(msg:Message)

RETURN msg.content AS content, msg.sentYear AS sentYear, msg.type AS type, msg.views AS views, "CommentedByFriend" AS role

UNION

// Tìm tin nhắn nhận phản ứng từ thành viên nhóm

MATCH (an:User {name: "An"})-[:MEMBER_OF]-(group:Group)

MATCH (user:User)-[:MEMBER_OF]-(group)

MATCH (user)-[r:REACTED_TO]->(msg:Message)

WHERE user <> an

RETURN msg.content AS content, msg.sentYear AS sentYear, msg.type AS type, msg.views AS views, "ReactedByGroup" AS role

The screenshot shows a Neo4j Cypher query editor with the following code:

```
1 // Tìm tất cả tin nhắn An gửi, nhận bình luận, hoặc phản ứng từ bạn bè qua "FRIEND" hoặc nhóm "MEMBER_OF"
2
3 // Tìm tin nhắn An gửi
4 MATCH (an:User {name: "An"})
5 OPTIONAL MATCH (an)-[:SENT]->(msg:Message)
6 WHERE msg IS NOT NULL
7 RETURN msg.content AS content, msg.sentYear AS sentYear, msg.type AS type, msg.views AS views, "Sender" AS role
8
9 UNION
10
```

Below the query editor, the results are displayed in a table with 5 columns: content, sentYear, type, views, and role. The table contains 3 rows of data.

	content	sentYear	type	views	role
1	"Msg1"	2023	"Text"	50	"Sender"
2	"Msg1"	2023	"Text"	50	"CommentedByFriend"
3	"Msg3"	2023	"Video"	500	"ReactedByGroup"

"FRIEND" hoặc nhóm "MEMBER_OF". Trả về content, sentYear, type, views, và vai trò ("Sender", "CommentedByFriend", "ReactedByGroup").

2. Tìm người dùng có activeYears trên 8, gửi ít nhất 2 tin nhắn có views trên 200, và thuộc ít nhất 1 nhóm. Trả về name, country, danh sách tin nhắn (contents, views), groupNames.

// Tìm người dùng có activeYears trên 8, gửi ít nhất 2 tin nhắn có views trên 200, và thuộc ít nhất 1 nhóm

```

MATCH (u:User)
WHERE u.activeYears > 8
MATCH (u)-[:SENT]->(m:Message)
WHERE m.views > 200
WITH u, collect(m) AS messages
WHERE size(messages) >= 2
MATCH (u)-[:MEMBER_OF]->(g:Group)
WITH u, messages, collect(g.name) AS groupNames
RETURN u.name AS name, u.country AS country,
       [msg IN messages | {content: msg.content, views: msg.views}] AS messagesList,
       groupNames

```

The screenshot shows the Neo4j Desktop interface. The top pane displays a Cypher query that filters users by active years, message views, and group membership. The bottom pane shows the results in a table format.

name	country	messagesList	groupNames
"Binh"	"Vietnam"	[{"content": "Msg9", "views": 450}, {"content": "Msg10", "views": 350}]	["Group1"]

3. Tìm bạn bè, nhóm và tin nhắn (30 điểm)

Dùng bảng sau và hình "Network Diagram":

3.1. Tìm người gửi tin nhắn kích thước trên 2 MB, lượt xem trên 300, là bạn của An với ít nhất 3 bạn chung qua "FRIEND" và cùng ít nhất 1 nhóm "MEMBER_OF". Trả về name, content, time, size, messagesShared, groupName, sắp xếp theo views giảm dần.

1. // Câu 3.1 (điều chỉnh)
2. **MATCH** (sender:**User**)-[:SENT]->(msg:Message)
3. **WHERE** msg.size > 2 **AND** msg.views > 300 4. **MATCH** (sender)-[:FRIEND]-
- (an:**User** {name: "An"})
- 5.
6. // Tìm bạn chung (giảm yêu cầu xuống còn 1 bạn chung)
7. **OPTIONAL MATCH** (sender)-[:FRIEND]-(commonFriend:**User**)-[:FRIEND]-(an)
8. **WHERE** commonFriend <> sender **AND** commonFriend <> an
9. **WITH** sender, msg, **count**(**DISTINCT** commonFriend) **AS** commonFriendCount, an
- 10.

11. // Tìm nhóm chung
12. **MATCH** (sender)-[:MEMBER_OF]->(group:Group)<-[:MEMBER_OF]-(an)
13. **MATCH** (sender)-[:sent:SENT]->(msg)
- 14.
15. **RETURN** sender.name **AS** name, msg.content **AS** content, sent.time **AS** time,
16. msg.size **AS** size, commonFriendCount **AS** messagesShared, group.name **AS** groupName
17. **ORDER BY** msg.views **DESC**



The screenshot shows a Cypher query in a text editor and its corresponding result table. The query is as follows:

```

1 // Câu 3.1 (điều chỉnh)
2 MATCH (sender:User)-[:SENT]->(msg:Message)
3 WHERE msg.size > 2 AND msg.views > 300
4 MATCH (sender)-[:FRIEND]-(an:User {name: "An"})
5
6 // Tìm bạn chung (giảm yêu cầu xuống còn 1 bạn chung)
7 OPTIONAL MATCH (sender)-[:FRIEND]-(commonFriend:User)-[:FRIEND]-(an)
8 WHERE commonFriend <> sender AND commonFriend <> an
9 WITH sender, msg, count(DISTINCT commonFriend) AS commonFriendCount, an
10
11 // Tìm nhóm chung
12 MATCH (sender)-[:MEMBER_OF]->(group:Group)<-[:MEMBER_OF]-(an)

```

The result table below shows the output of the query:

	name	content	time	size	messagesShared	groupName
1	"Cường"	"Msg3"	"09:00"	10	0	"Group1"
2	"Binh"	"Msg9"	"15:45"	5.0	1	"Group1"
3	"Binh"	"Msg10"	"16:20"	4.5	1	"Group1"

3.2. Tìm người gửi tin nhắn Video kích thước trên 10 MB, lượt xem trên 400, là bạn của Dũng với messagesShared ít nhất 2 và cùng nhóm. Trả về name, content, size, views, groupName.

```

MATCH (sender:User)-[:SENT]->(msg:Message {type: "Video"})
WHERE msg.size > 10 AND msg.views > 400
MATCH (sender)-[:friend:FRIEND]-(dung:User {name: "Dũng"})
WHERE friend.messagesShared >= 2
MATCH (sender)-[:MEMBER_OF]->(group:Group)<-[:MEMBER_OF]-(dung)

RETURN sender.name AS name, msg.content AS content, msg.size AS size,
msg.views AS views, group.name AS groupName

```

```

1 // Câu 3.2
2 MATCH (sender:User)-[:SENT]->(msg:Message {type: "Video"})
3 WHERE msg.size > 10 AND msg.views > 400
4 MATCH (sender)-[friend:FRIEND]-(dung:User {name: "Dũng"})
5 WHERE friend.messagesShared ≥ 2
6 MATCH (sender)-[:MEMBER_OF]-(group:Group)←[:MEMBER_OF]-(dung)
7
8 RETURN sender.name AS name, msg.content AS content, msg.size AS size,
9        msg.views AS views, group.name AS groupName

```

	name	content	size	views	groupName
1	"Lan"	"Msg6"	15	700	"Group2"
2	"Lan"	"Msg6"	15	700	"Group2"

3.3. Đếm số tin nhắn Image kích thước trên 2 MB, lượt xem trên 250 mà Yến gửi, bình luận hoặc nhận phản ứng từ bạn bè với messagesShared trên 2. Trả về số lượng.

```

// Đếm tin nhắn Yến gửi
MATCH (yen:User {name: "Yên"})-[:SENT]->(msg:Message {type: "Image"})
WHERE msg.size > 2 AND msg.views > 250
RETURN count(msg) AS sentCount

```

```

1 // Đếm tin nhắn Yến gửi
2 MATCH (yen:User {name: "Yên"})-[:SENT]->(msg:Message {type: "Image"})
3 WHERE msg.size > 2 AND msg.views > 250
4 RETURN count(msg) AS sentCount

```

	sentCount
1	1

4. Đường đi và phân tích (30 điểm)

Dùng bảng sau và hình "Network Diagram":

Từ	Đến	Tin nhắn top	Yếu tố kiểm tra
An	Lan	4 tin xem nhiều nhất	Đường đi + Nhóm
Cường	Hoa	3 tin kích thước lớn nhất	Loại tin + Phản ứng

4.1. Tìm đường ngắn nhất từ An đến Lan qua "SENT", "FRIEND", "COMMENTED", "MEMBER_OF", "REACTED_TO". Tính tổng views của 4 tin nhắn có lượt xem cao nhất của

An hoặc nhóm của An, kiểm tra tin nào trong số đó nằm trên đường đi và thuộc nhóm nào.
Trả về path, totalViews, contents, views, groupNames.

// Câu 4.1 - Phần 1: Tìm đường đi ngắn nhất từ An đến Lan

```
MATCH path = shortestPath((an:User {name: "An"})-[*]-(lan:User {name: "Lan"}))
WHERE all(r IN relationships(path) WHERE type(r) IN ["SENT", "FRIEND", "COMMENTED", "
MEMBER_OF", "REACTED_TO"])
RETURN [n IN nodes(path) | CASE
  WHEN 'User' IN labels(n) THEN n.name
  WHEN 'Message' IN labels(n) THEN n.content
  WHEN 'Group' IN labels(n) THEN n.name
END] AS path
```



path
["An", "Msg1", "Lan"]

// Câu 4.1 - Phần 2: Tìm 4 tin nhắn có lượt xem cao nhất

// Tin nhắn của An

```
MATCH (an:User {name: "An"})-[:SENT]->(anMsg:Message)
WITH collect(anMsg) AS anMessages
```

// Tin nhắn của người trong nhóm của An

```
MATCH (an:User {name: "An"})-[:MEMBER_OF]->(g:Group)<-
[:MEMBER_OF](groupMember:User)
WHERE an <> groupMember
MATCH (groupMember)-[:SENT]->(groupMsg:Message)
WITH anMessages, collect(groupMsg) AS groupMessages
```

// Kết hợp và sắp xếp

```
WITH anMessages + groupMessages AS allMessages
UNWIND allMessages AS msg
RETURN msg.content, msg.views
ORDER BY msg.views DESC
LIMIT 4
```

```

1 // Câu 4.1 - Phần 2: Tìm 4 tin nhắn có lượt xem cao nhất
2 // Tin nhắn của An
3 MATCH (an:User {name: "An"})-[:SENT]->(anMsg:Message)
4 WITH collect(anMsg) AS anMessages
5
6 // Tin nhắn của người trong nhóm của An
7 MATCH (an:User {name: "An"})-[:MEMBER_OF]->(g:Group)-[:MEMBER_OF]-(groupMember:User)
8 WHERE an < groupMember
9 MATCH (groupMember)-[:SENT]->(groupMsg:Message)
10 WITH anMessages, collect(groupMsg) AS groupMessages
11
12 // Kết hợp và sắp xếp

```

	msg.content	msg.views
1	"Msg6"	700
2	"Msg3"	500
3	"Msg9"	450
4	"Msg10"	350

Started streaming 4 records after 24 ms and completed after 27 ms.

// Câu 4.1 - Kết hợp

// Tìm đường đi ngắn nhất từ An đến Lan

```

MATCH path = shortestPath((an:User {name: "An"})-[*]-(lan:User {name: "Lan"}))
WHERE all(r IN relationships(path) WHERE type(r) IN ["SENT", "FRIEND", "COMMENTED", "MEMBER_OF", "REACTED_TO"])

```

// Tìm 4 tin nhắn có lượt xem cao nhất của An hoặc nhóm của An

```

MATCH (an:User {name: "An"})-[:SENT]->(anMsg:Message)
WITH path, collect(anMsg) AS anMessages

```

```

MATCH (an:User {name: "An"})-[:MEMBER_OF]->(g:Group)-[:MEMBER_OF]-(groupMember:User)
WHERE an < groupMember
MATCH (groupMember)-[:SENT]->(groupMsg:Message)
WITH path, anMessages, collect(groupMsg) AS groupMessages

```

```

WITH path, anMessages + groupMessages AS allMessages
UNWIND allMessages AS msg
WITH path, msg
ORDER BY msg.views DESC
LIMIT 4
WITH path, collect(msg) AS topMessages

```

// Tính tổng views

```

WITH path, topMessages, reduce(s = 0, m IN topMessages | s + m.views) AS totalViews

```

// Tìm nhóm của An

```

MATCH (an:User {name: "An"})-[:MEMBER_OF]->(group:Group)
WITH path, topMessages, totalViews, collect(group.name) AS groupNames

```


// Trả về kết quả

RETURN

[n IN nodes(path) | CASE WHEN 'User' IN labels(n) THEN n.name WHEN 'Message' IN labels(n) THEN n.content WHEN 'Group' IN labels(n) THEN n.name END] AS path,

totalViews,

[msg IN topMessages | msg.content] AS contents,

[msg IN topMessages | msg.views] AS views,

groupNames

```
1 // Câu 4.1 - Kết hợp
2 // Tìm đường đi ngắn nhất từ An đến Lan
3 MATCH path = shortestPath((an:User {name: "An"})-[*]-(lan:User {name: "Lan"}))
4 WHERE all(r IN relationships(path) WHERE type(r) IN ["SENT", "FRIEND", "COMMENTED", "MEMBER_OF", "REACTED_TO"])
5
6 // Tìm 4 tin nhắn có lượt xem cao nhất của An hoặc nhóm của An
7 MATCH (an:User {name: "An"})-[:SENT]->(anMsg:Message)
8 WITH path, collect(anMsg) AS anMessages
9
10 MATCH (an:User {name: "An"})-[:MEMBER_OF]->(g:Group)-[:MEMBER_OF]->(groupMember:User)
11 WHERE an <- groupMember
12 MATCH (groupMember)-[:SENT]->(groupMsg:Message)
```

path	totalViews	contents	views	groupNames
["An", "Msg1", "Lan"]	2000	["Msg6", "Msg3", "Msg9", "Msg10"]	[700, 500, 450, 350]	["Group1"]

4.2. Tìm đường ngắn nhất từ Cường đến Hoa qua các quan hệ trên. Tính tổng size của 3 tin nhắn kích thước lớn nhất mà Hoa hoặc bạn bè gửi/bình luận/phản ứng, kiểm tra loại tin và phản ứng trên chúng. Trả về path, totalSize, contents, types, reactions.

// Câu 4.2

MATCH path = shortestPath((cuong:User {name: "Cường"})-[*]-(hoa:User {name: "Hoa"}))
WHERE all(r IN relationships(path) WHERE type(r) IN ["SENT", "FRIEND", "COMMENTED", "MEMBER_OF", "REACTED_TO"])

// Lấy 3 tin nhắn có kích thước lớn nhất trong hệ thống

MATCH (u:User)-[:SENT]->(msg:Message)

WITH path, msg

ORDER BY msg.size DESC

LIMIT 3

WITH path, collect(msg) AS topSizeMessages

// Tính tổng kích thước

WITH path, topSizeMessages, reduce(s = 0, m IN topSizeMessages | s + m.size) AS totalSize

RETURN

[n IN nodes(path) | CASE WHEN 'User' IN labels(n) THEN n.name WHEN 'Message' IN labels(n) THEN n.content WHEN 'Group' IN labels(n) THEN n.name END] AS path,

totalSize,

[msg IN topSizeMessages | msg.content] AS contents,

[msg IN topSizeMessages | msg.type] AS types

```

1 // Câu 4.2
2 MATCH path = shortestPath((cuong:User {name: "Cuông"})-[*]-(hoa:User {name: "Hoa"}))
3 WHERE all(r IN relationships(path) WHERE type(r) IN ["SENT", "FRIEND", "COMMENTED", "MEMBER_OF", "REACTED_TO"])
4
5 // Lấy 3 tin nhắn có kích thước lớn nhất trong hệ thống
6 MATCH (u:User)-[:SENT]->(msg:Message)
7 WITH path, msg
8 ORDER BY msg.size DESC
9 LIMIT 3
10 WITH path, collect(msg) AS topSizeMessages
11
12 // Tính tổng kích thước

```

	path	totalSize	contents	types
1	["Cuông", "An", "Dũng", "Minh", "Msg7", "Hoa"]	37	["Msg6", "Msg8", "Msg3"]	["Video", "Video", "Video"]

4.3. Đếm số người có "FRIEND" với ít nhất 4 người khác và thuộc ít nhất 1 nhóm có memberCount trên 4. Trả về số lượng.

// Câu 4.3

```

MATCH (u:User)-[:FRIEND]-(friend:User)
WITH u, COUNT(DISTINCT friend) AS friendCount
WHERE friendCount >= 4
MATCH (u)-[:MEMBER_OF]->(g:Group)
RETURN COUNT(DISTINCT u) AS numberOfUsers

```

```

1 // Câu 4.3
2 MATCH (u:User)-[:FRIEND]-(friend:User)
3 WITH u, COUNT(DISTINCT friend) AS friendCount
4 WHERE friendCount >= 4
5 MATCH (u)-[:MEMBER_OF]->(g:Group)
6 RETURN COUNT(DISTINCT u) AS numberOfUsers

```

	numberOfUsers
1	2