// Xóa toàn bộ dữ liệu trước khi insert

MATCH (n) DETACH DELETE n;

// Tạo người dùng

CREATE (u1:User {name: "Alice", age: 25, city: "New York"})

CREATE (u2:User {name: "Bob", age: 30, city: "San Francisco"})

CREATE (u3:User {name: "Charlie", age: 28, city: "Los Angeles"})

CREATE (u4:User {name: "David", age: 35, city: "Chicago"})

CREATE (u5:User {name: "Emma", age: 22, city: "Boston"})

// Tạo sản phẩm

CREATE (p1:Product {title: "Laptop", price: 1000, category: "Electronics"})

CREATE (p2:Product {title: "Smartphone", price: 700, category: "Electronics"})

CREATE (p3:Product {title: "Headphones", price: 200, category: "Accessories"})

CREATE (p4:Product {title: "Tablet", price: 500, category: "Electronics"})

CREATE (p5:Product {title: "Smartwatch", price: 300, category: "Wearables"})

// Tạo giao dịch (PURCHASED)

CREATE (u1)-[:PURCHASED {date: "2023-05-12", amount: 1}]->(p1)

CREATE (u1)-[:PURCHASED {date: "2023-06-10", amount: 2}]->(p2)

CREATE (u2)-[:PURCHASED {date: "2023-04-08", amount: 1}]->(p3)

CREATE (u3)-[:PURCHASED {date: "2023-07-15", amount: 1}]->(p4)

CREATE (u4)-[:PURCHASED {date: "2023-08-05", amount: 3}]->(p5)

CREATE (u5)-[:PURCHASED {date: "2023-09-20", amount: 1}]->(p1)

CREATE (u2)-[:PURCHASED {date: "2023-10-01", amount: 2}]->(p2)

CREATE (u3)-[:PURCHASED {date: "2023-11-11", amount: 1}]->(p3)

CREATE (u4)-[:PURCHASED {date: "2023-12-24", amount: 2}]->(p4)

CREATE (u5)-[:PURCHASED {date: "2024-01-05", amount: 1}]->(p5)

// Tạo quan hệ bạn bè (FRIENDS)

CREATE (u1)-[:FRIENDS]->(u2)

CREATE (u2)-[:FRIENDS]->(u3)

CREATE (u3)-[:FRIENDS]->(u4)

CREATE (u4)-[:FRIENDS]->(u5)

CREATE (u5)-[:FRIENDS]->(u1)

CREATE (u1)-[:FRIENDS]->(u3)

CREATE (u2)-[:FRIENDS]->(u4)

# Phần 1: Cơ Bản

1. Truy vấn danh sách tất cả người dùng và hiển thị tên, tuổi, thành phố.
2. Lấy danh sách tất cả sản phẩm và hiển thị tên, giá, danh mục.
3. Tìm tất cả người dùng sống ở thành phố "New York".
4. Tìm tất cả sản phẩm có giá trên 500.
5. Tìm tất cả sản phẩm trong danh mục "Electronics".
6. Lấy danh sách tất cả các giao dịch mua hàng (người dùng, sản phẩm, số lượng, ngày mua).
7. Tìm tất cả sản phẩm mà một người dùng cụ thể đã mua (ví dụ: Alice).
8. Liệt kê tất cả các quan hệ FRIENDS giữa các người dùng.
9. Tìm số lượng sản phẩm mà mỗi người dùng đã mua.
10. Sắp xếp danh sách sản phẩm theo giá giảm dần.

# Phần 2: Trung Cấp

1. Tính tổng số tiền mà mỗi người dùng đã chi tiêu.
2. Tìm người dùng có tổng chi tiêu cao nhất.
3. Tìm sản phẩm được mua nhiều nhất.
4. Tìm tất cả bạn bè của một người dùng cụ thể.
5. Liệt kê tất cả người dùng có từ 2 bạn bè trở lên.
6. Tìm các sản phẩm mà từ 2 người dùng đã mua trở lên.
7. Liệt kê tất cả người dùng chưa từng mua sản phẩm nào.
8. Tìm số lượng giao dịch mua hàng theo từng tháng trong năm 2023.
9. Tính tổng số sản phẩm được bán ra theo từng danh mục sản phẩm.
10. Tìm tất cả sản phẩm mà ít nhất một người bạn của Alice đã mua.

# Phần 3: Nâng Cao

1. Tìm tất cả sản phẩm mà bạn bè của Alice đã mua nhưng Alice chưa mua.
2. Tìm danh sách người dùng có chung bạn bè với Alice.
3. Tìm người dùng có nhiều bạn bè chung nhất với Alice.

# Đáp Án

// 1. Truy vấn danh sách tất cả người dùng và hiển thị tên, tuổi, thành phố.

match (n:User) return n.name, n.age, n.city

// 2. Lấy danh sách tất cả sản phẩm và hiển thị tên, giá, danh mục.

match (n:Product) return n.title, n.price, n.category

// 3. Tìm tất cả người dùng sống ở thành phố "New York"

match(n:User) where n.city = "New York" return n

// 4. Tìm tất cả sản phẩm có giá trên 500.

MATCH(n:Product) where n.price > 500 return n

// 5. Tìm tất cả sản phẩm trong danh mục "Electronics".

MATCH(n:Product) where n.category = "Electronics" return n

// 6. Lấy danh sách tất cả các giao dịch mua hàng (người dùng, sản phẩm, số lượng, ngày mua).

match(n:User)-[r:PURCHASED]->(m:Product)

return n.name, m.title, r.amount, r.date

// 7. Tìm tất cả sản phẩm mà một người dùng cụ thể đã mua (ví dụ: Alice).

MATCH(n:User{name:"Alice"})-[r:PURCHASED]->(m:Product)

return m.title

// 8. Liệt kê tất cả các quan hệ FRIENDS giữa các người dùng.

match(n:User)-[r:FRIENDS]->(m:User)

return n.name as user, m.name as friend\_of, r

// 9. Tìm số lượng sản phẩm mà mỗi người dùng đã mua.

MATCH(n:User)-[r:PURCHASED]->(m)

return n.name, sum(r.amount) as total\_amount

// 10. Sắp xếp danh sách sản phẩm theo giá giảm dần.

MATCH(n:Product) return n.title, n.price order by n.price desc

// 11.  Tính tổng số tiền mà mỗi người dùng đã chi tiêu.

match (n:User)-[r:PURCHASED]->(m:Product)

return n.name, sum(m.price \* r.amount) as total\_spent

// 12.  Tính tổng số tiền mà mỗi người dùng đã chi tiêu.

// Cách 1:

match (n:User)-[r:PURCHASED]->(m:Product)

with n.name as user, sum(m.price \* r.amount) as total\_spent

order by total\_spent desc

limit 1

with total\_spent as highest\_total\_spent

match (n:User)-[r:PURCHASED]->(m:Product)

with n.name as user\_name, sum(m.price \* r.amount) as total\_spent, highest\_total\_spent

where total\_spent = highest\_total\_spent

return user\_name, total\_spent

// Cách 2:

match(n:User)-[r:PURCHASED]->(m:Product)

with n.name as user, sum(r.amount \* m.price) as total\_spent

order by total\_spent desc

with collect(user) as top\_spenders, total\_spent

limit 1

return top\_spenders, total\_spent

// 13.  Tìm sản phẩm được mua nhiều nhất.

match(n:Product)<-[r:PURCHASED]-(m:User)

with n.title as product\_name, sum(r.amount) as total\_bought\_amount

order by  total\_bought\_amount desc

with collect(product\_name) as top\_selling\_product, total\_bought\_amount

limit 1

return top\_selling\_product, total\_bought\_amount

// 15.  Liệt kê tất cả người dùng có nhiều từ 2 bạn bè trở lên.

MATCH(n:User)-[r:FRIENDS]->(m:User)

with n.name as user\_name,  count(m) as  amount\_of\_friends

where amout\_of\_friends > 1

return user\_name, amout\_of\_friends

// 16.  Tìm các sản phẩm mà nhiều từ 2 người dùng đã mua trở lên.

match(n:Product)<-[r:PURCHASED]-(m:User)

with n.title as product\_name, count(m) as number\_of\_people\_bought

where number\_of\_people\_bought > 1

return product\_name,  number\_of\_people\_bought

// 17.  Liệt kê tất cả người dùng chưa từng mua sản phẩm nào.

match (n:User)

where not exists {MATCH(n)-[r:PURCHASED]->(m:Product)}

return n

// 18.  Tìm số lượng giao dịch mua hàng theo từng tháng trong năm 2023.

match(n:Product)<-[r:PURCHASED]-(m:User)

where r.date STARTS WITH "2023"

with SUBSTRING(r.date, 5, 2) as month, count(r) as total\_transactions

return month, total\_transactions

order month by asc

// 19.  Tính tổng số sản phẩm được bán ra theo từng danh mục sản phẩm.

match (n:Product)<-[r:PURCHASED]-(m:User)

with n.category as category, sum(r.amount) as total\_amount\_bought

return category, total\_amount\_bought

// 20. Tìm tất cả sản phẩm mà ít nhất hai người bạn của Alice đã mua. match

(n:User)-[r:FRIENDS]->(m:User)-[p:PURCHASED]->(t:Product) where n.name = "Alice" with t.name, count(m) >= 2 return t.title

// 21. Tìm tất cả sản phẩm mà bạn bè của Alice đã mua nhưng Alice chưa mua

match (n:User{name:"Alice"})-[r:FRIENDS]->(m:User)-[p:PURCHASED]->(prod:Product)

where not exists {(n)-[:PURCHASED]->(prod)}

return prod

//  22. Tìm danh sách người dùng có chung bạn bè với Alice.

MATCH (common\_friend:User)-[:FRIENDS]-(user:User)

WHERE EXISTS {

   (common\_friend)-[:FRIENDS]-(alice:User {name: "Alice"})

} and user.name <> "Alice"

RETURN DISTINCT user.name AS shared\_friend

// 23. Tìm người dùng có nhiều bạn bè chung nhất với Alice.

MATCH (common\_friend:User)-[:FRIENDS]-(user:User)

WHERE EXISTS {

   (common\_friend)-[:FRIENDS]-(alice:User {name: "Alice"})

} and user.name <> "Alice"

with user.name as name, count(common\_friend) as mutal\_friends\_count

order by mutal\_friends\_count desc

with collect(name) as top\_friend, mutal\_friends\_count

return top\_friend, mutal\_friends\_count

limit 1