Big Data Lab04

Group members: Liang Haoxuan, Liu Sihan, Wang Changpeng

Medium Tasks

1/ a/

Atomicity: All changes within a transaction are applied as a single unit, or none at all.
Consistency: The database remains in a consistent state before and after the transaction.
Isolation: Transactions do not interfere with each other, ensuring that concurrent transactions do not affect the outcome of each other.
Durability: Once a transaction is committed, its effects are permanent and survive system failures.

b/

Relational Databases: Relationships are typically represented using foreign keys. Foreign keys reference primary keys in other tables, and joins are used to combine data from multiple tables.

MongoDB: Relationships can be modeled using embedded references or normalized references. Embedded references store related data within the same document, while normalized references use separate documents and link them via object IDs.

Neo4j: Relationships are first-class citizens and are stored as directed, named, and semantically relevant connections between nodes. Relationships are physically stored alongside the data they connect, allowing for efficient traversal without the need for index lookups.

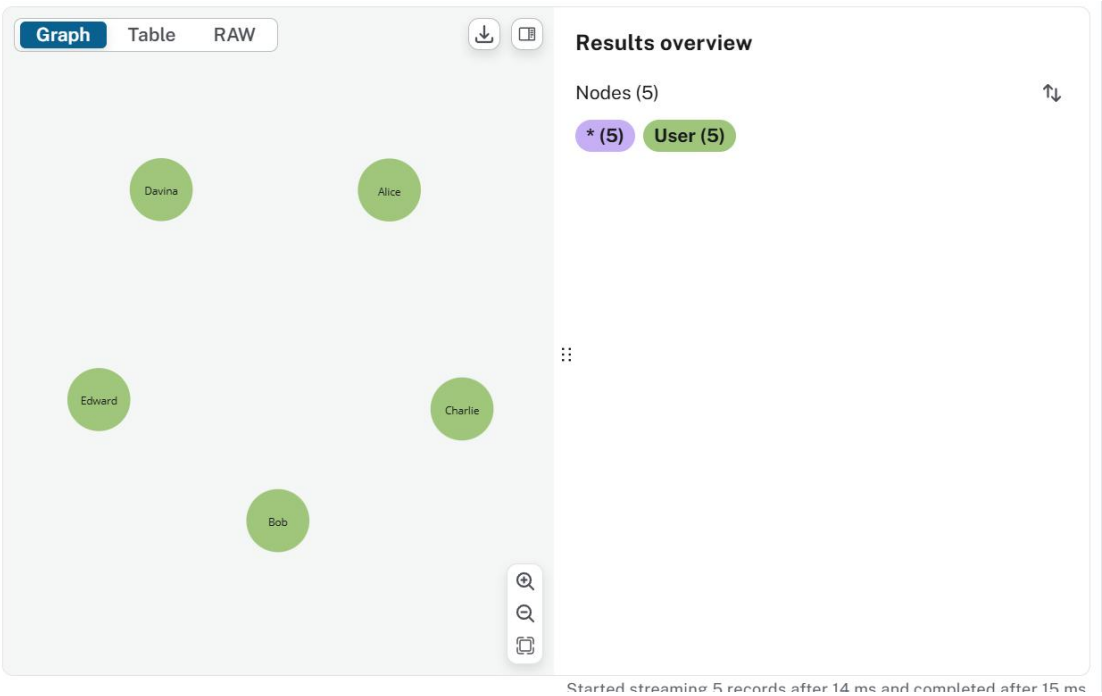
c/

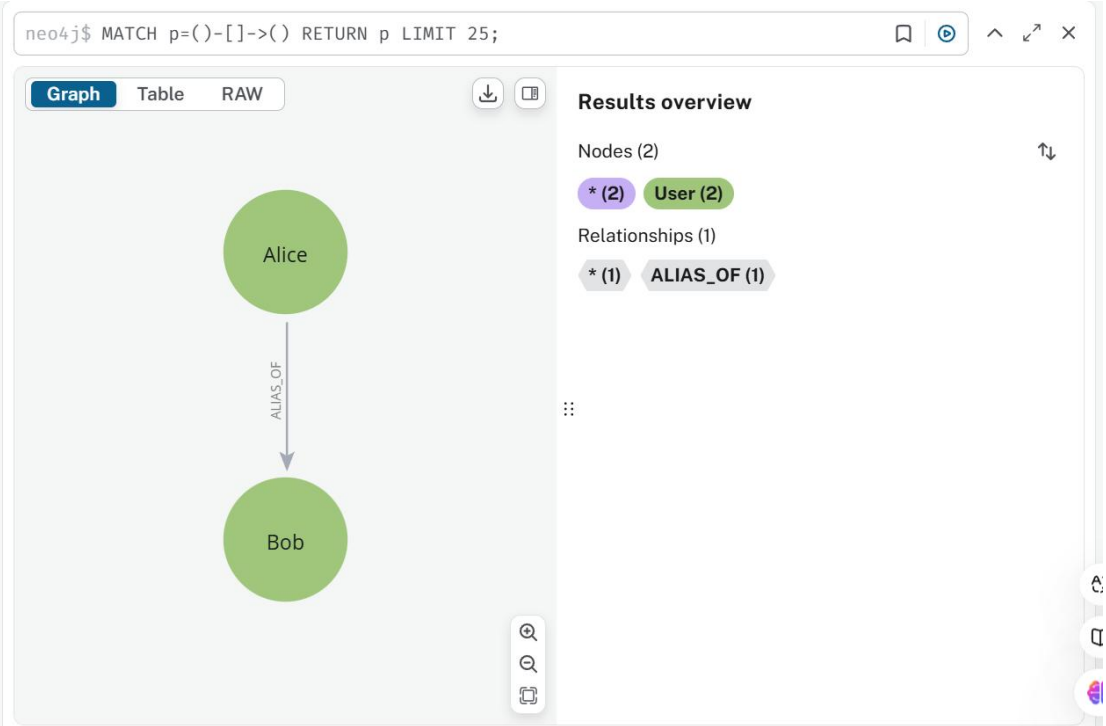
Native Storage: In native graph storage, the database is specifically designed to store and process graph data. Relationships are stored directly, and nodes are linked using index-free adjacency, which means that each node directly references its adjacent nodes. This results in highly efficient and fast traversal of the graph.

Non-Native Storage: Non-native graph storage uses existing storage systems (like relational or columnar databases) to store graph data. Relationships are often stored in separate tables or columns, requiring additional join operations to traverse the graph, which can be less efficient.

2/ a/

Purpose: This Cypher script creates five user nodes and a relationship between Alice and Bob, indicating that Alice is an alias of Bob.



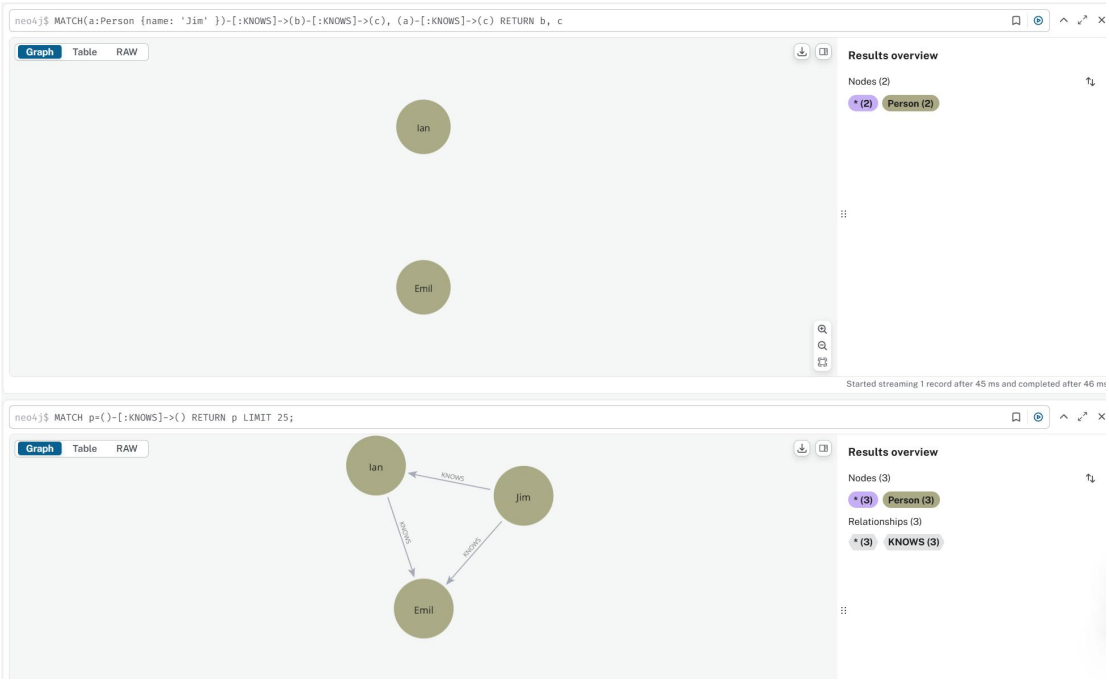


b/

Purpose: This Cypher script creates relationships between Bob and Charlie, Davina, and Edward, indicating that Bob emailed Charlie, CC'd Davina, and BCC'd Edward.

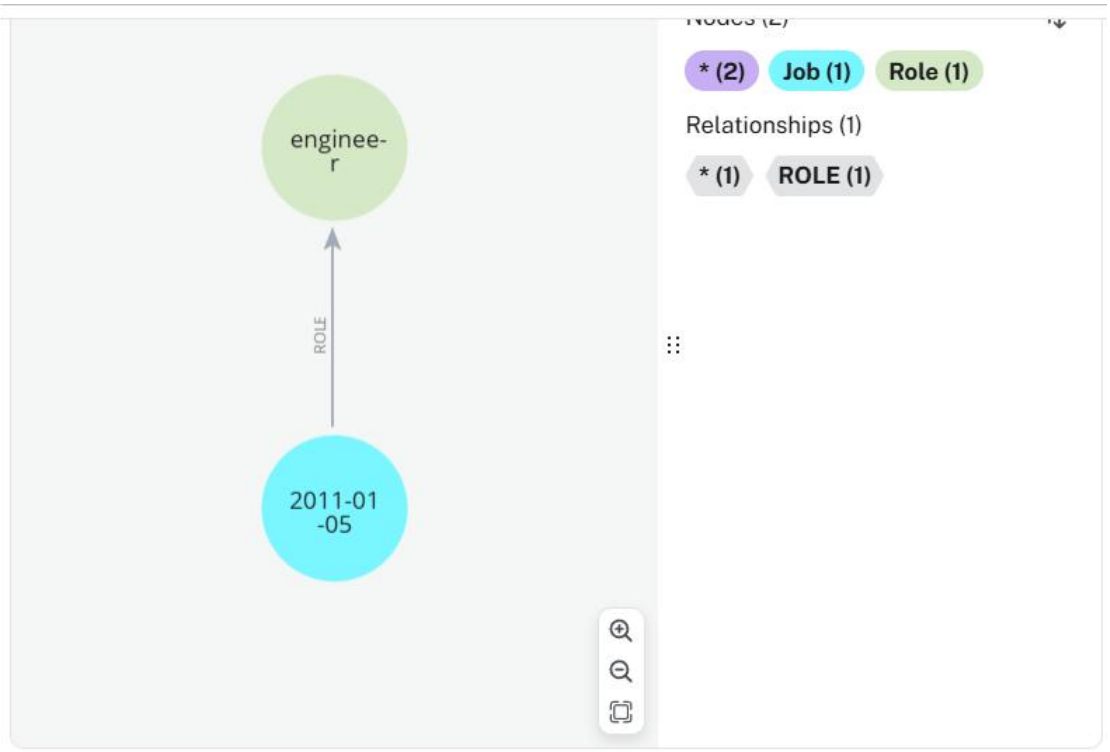
3/

This Cypher script finds all people that Jim knows, who in turn know someone that Jim also knows directly. The result will return the nodes b and c that satisfy this condition.

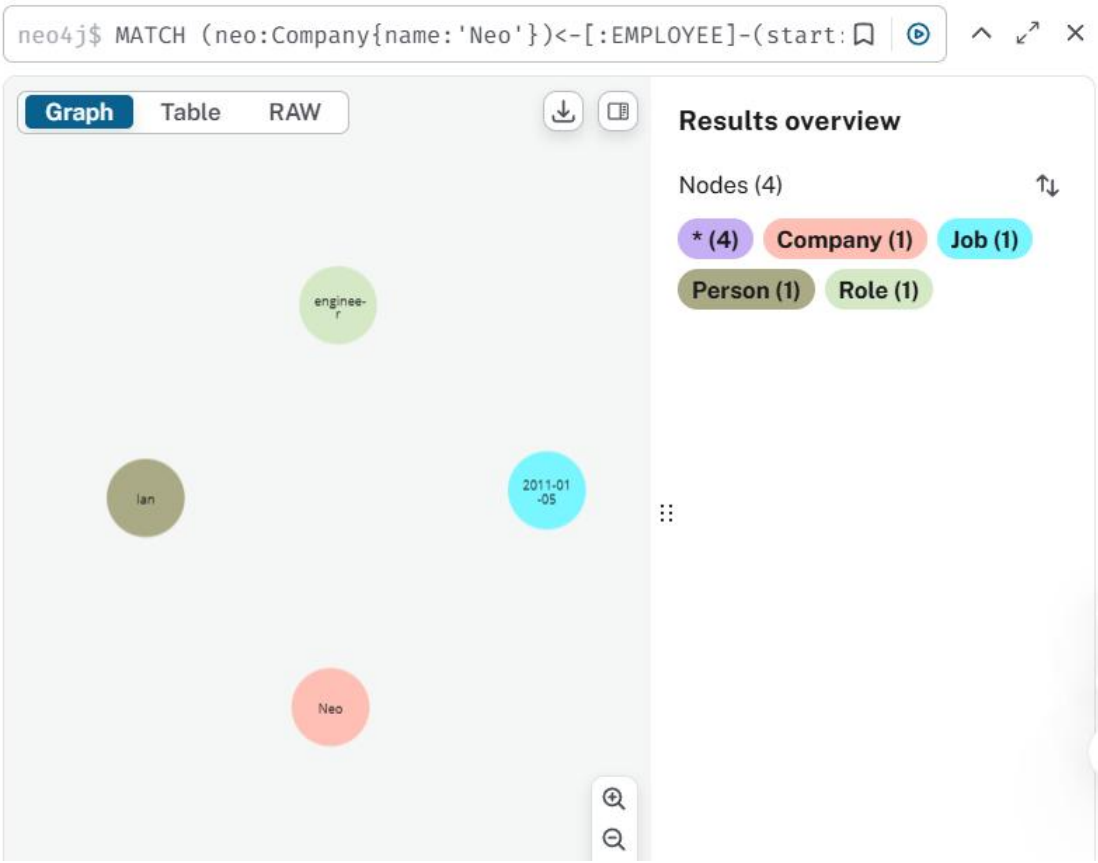


4/

CREATE (neo:Company{name:'Neo'})<-[:EMPLOYEE]-(start:Job{start:'2011-01-05'})-[:ROLE]->(engineer:Role{name:'engineer'}),(lan:Person{name:'lan'})-[:EMPLOYMENT]->(start)



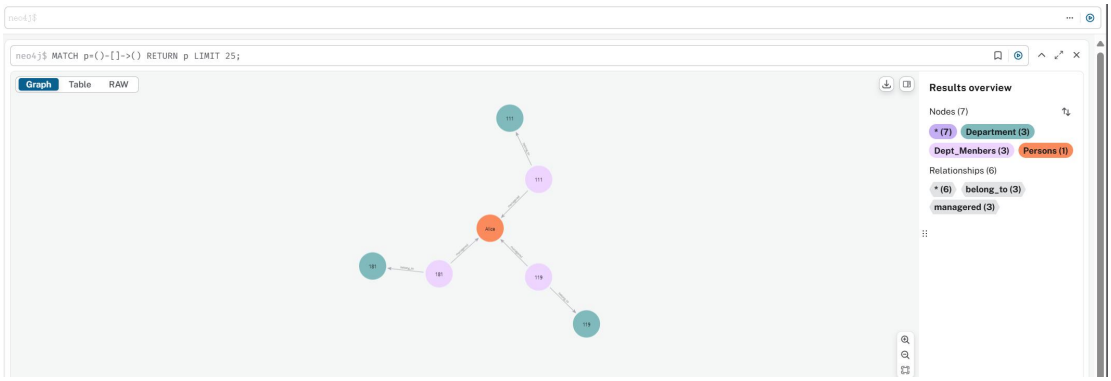
Started streaming 1 record after 19 ms and completed after 20 ms.



Advanced Tasks

5/

CREATE (P0815:Department{members:119})<-[:belong_to]-(Dept2:Dept_Menbers{members:119})-[:managered]->(alice:Persons{name:'Alice',menbers:815})<-[:managered]-(Dept1:Dept_Menbers{members:111})-[:belong_to]->(FUTURE_4:Department{members:111}),(alice)<-[:managered]-(Dept3:Dept_Menbers{members:181})-[:belong_to]->(A42:Department{members:181})



6/ a/

This Cypher script finds the relationship between Sally and John and returns the date they became friends.

Result: friends_since: '01/09/2013'

b/

MATCH (book:Book {title: 'Moby Dick'}) RETURN AVG(book.rating)

MATCH (book:Book {title: 'Moby Dick'}) RETURN book.author

MATCH (sally:Person{name:'Sally'}) RETURN sally.Age

MATCH (person:Person)-[r:HAS_READ]->(book:Book {title: 'Moby Dick'})
RETURN person.name, r.on
ORDER BY r.on ASC
LIMIT 1