SNS COLLEGE OF TECHNOLOGY

Coimbatore-35
An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A++' Grade Approved
by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF COMPUTER APPLICATIONS

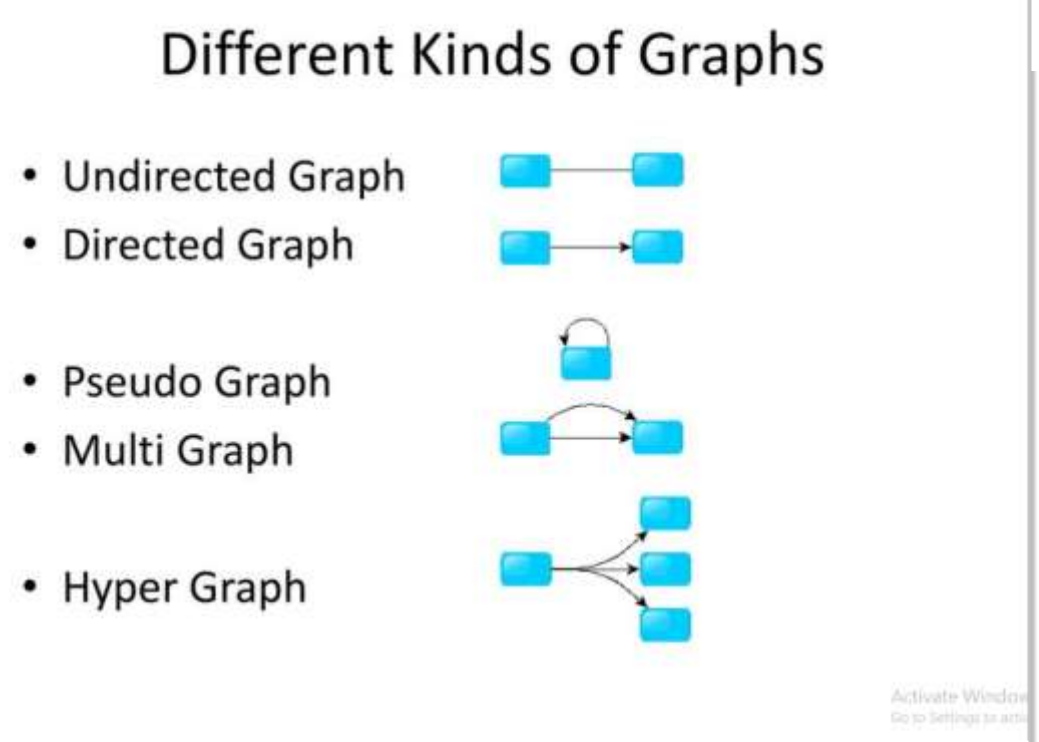
19CAE730 – Fundamentals of NOSQL database System
II YEAR III SEM

UNIT IV - Graph NoSQL databases using Neo4,NoSQL database development tools and programming languages

What is a Graph?

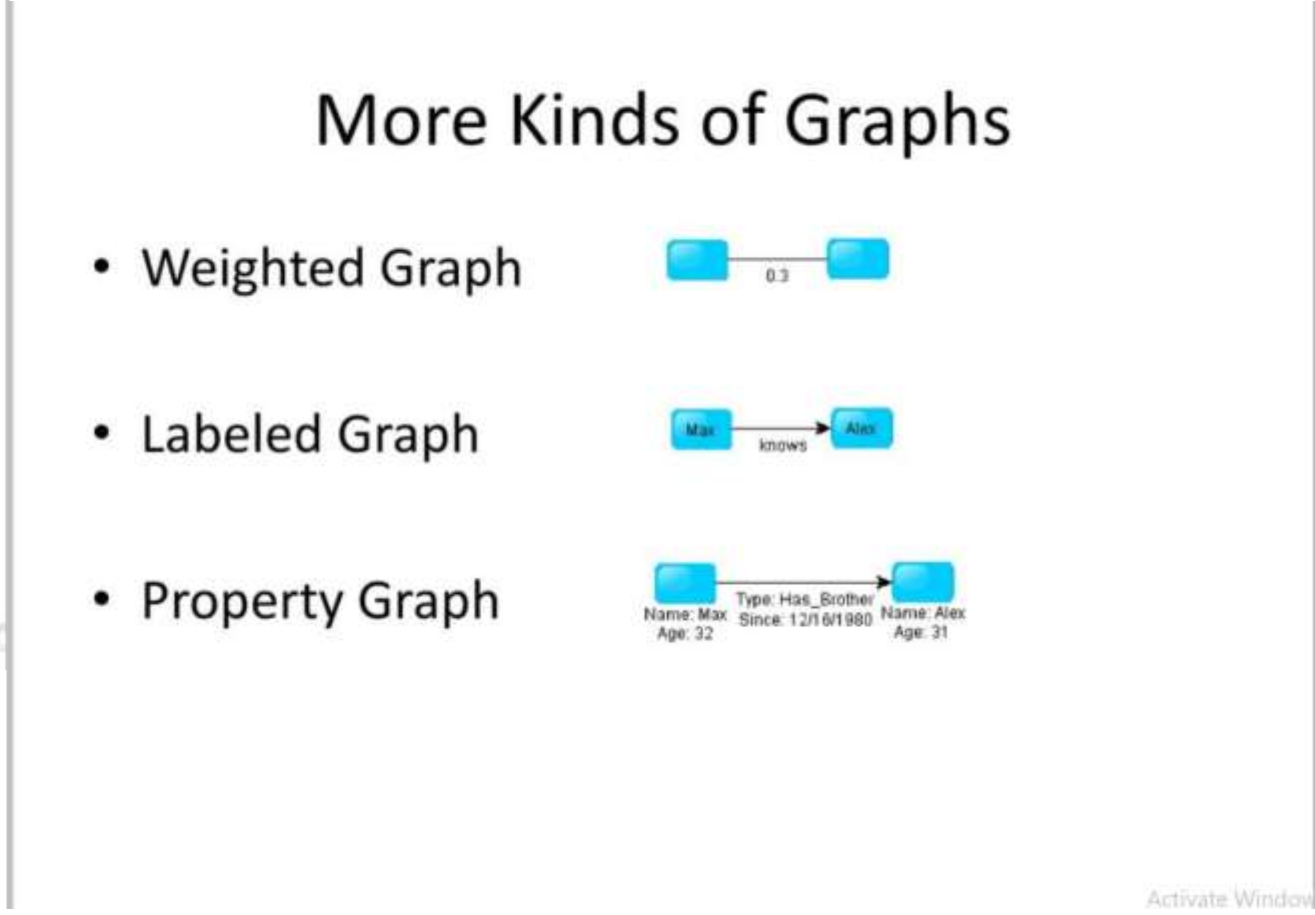
- An abstract representation of a set of objects where some pairs are connected by links.

id	Object (Vertex, Node)
→	Link (Edge, Arc, Relationship)



Different Kinds of Graphs

- Undirected Graph
- Directed Graph
- Pseudo Graph
- Multi Graph
- Hyper Graph



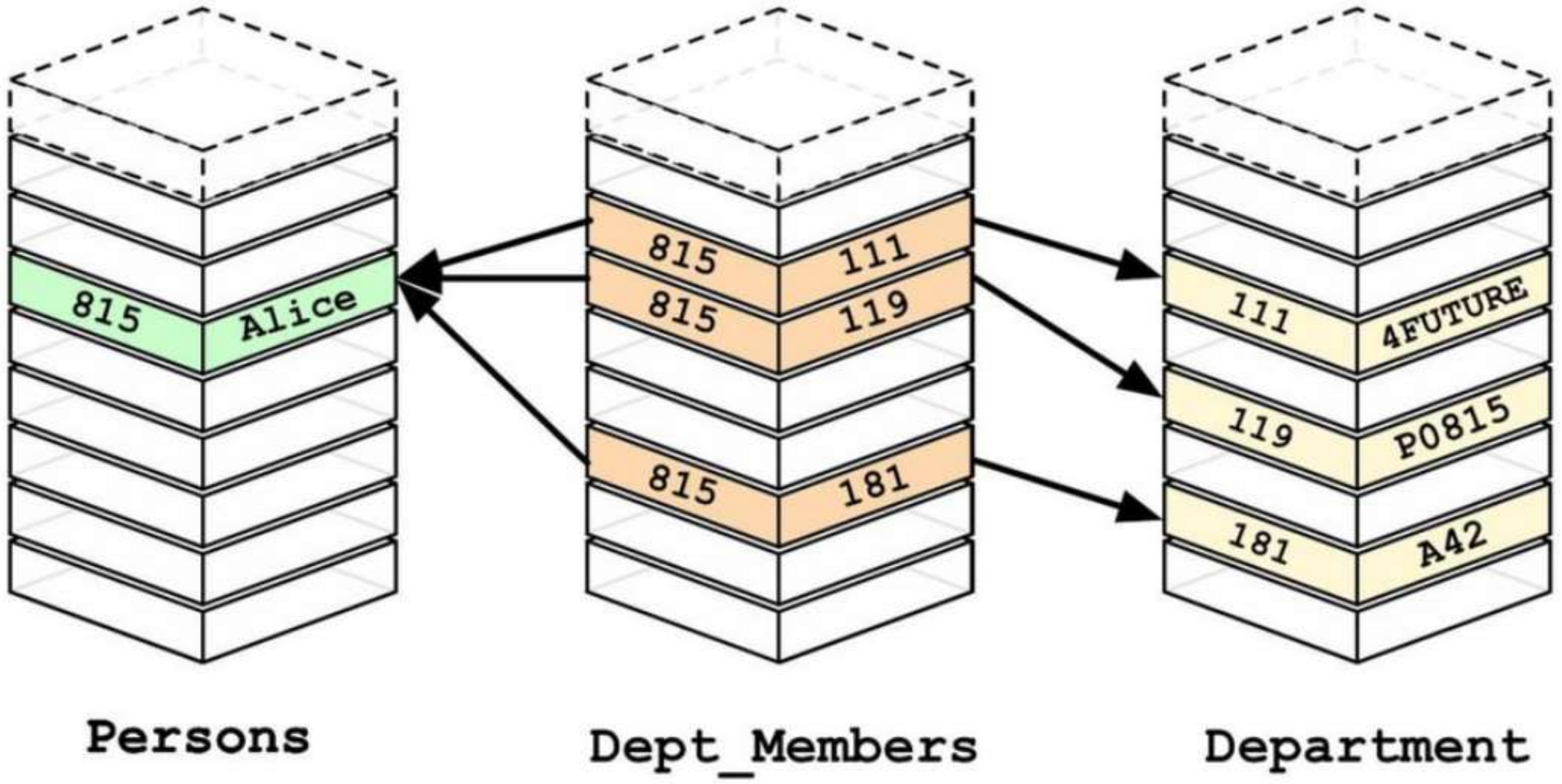
More Kinds of Graphs

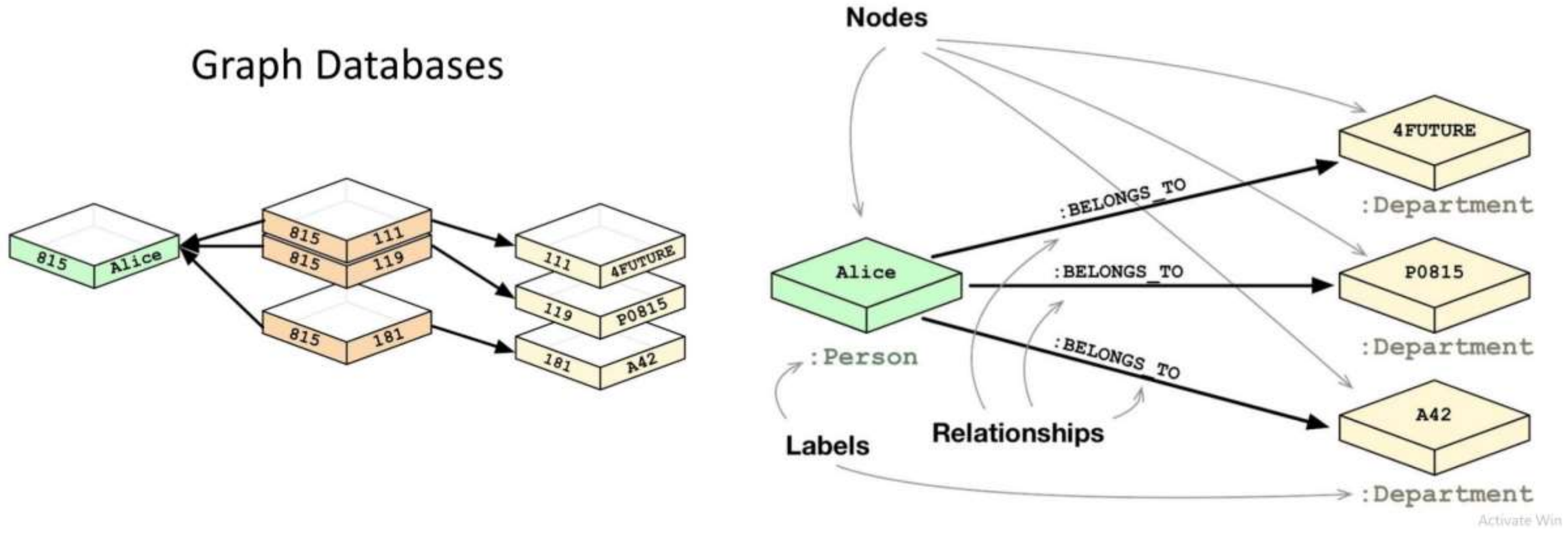
- Weighted Graph
- Labeled Graph
- Property Graph

What is a Graph Database?

- A database with an explicit graph structure
- Each node knows its adjacent nodes
- As the number of nodes increases, the cost of a local step (or hop) remains the same
- Plus an Index for lookups

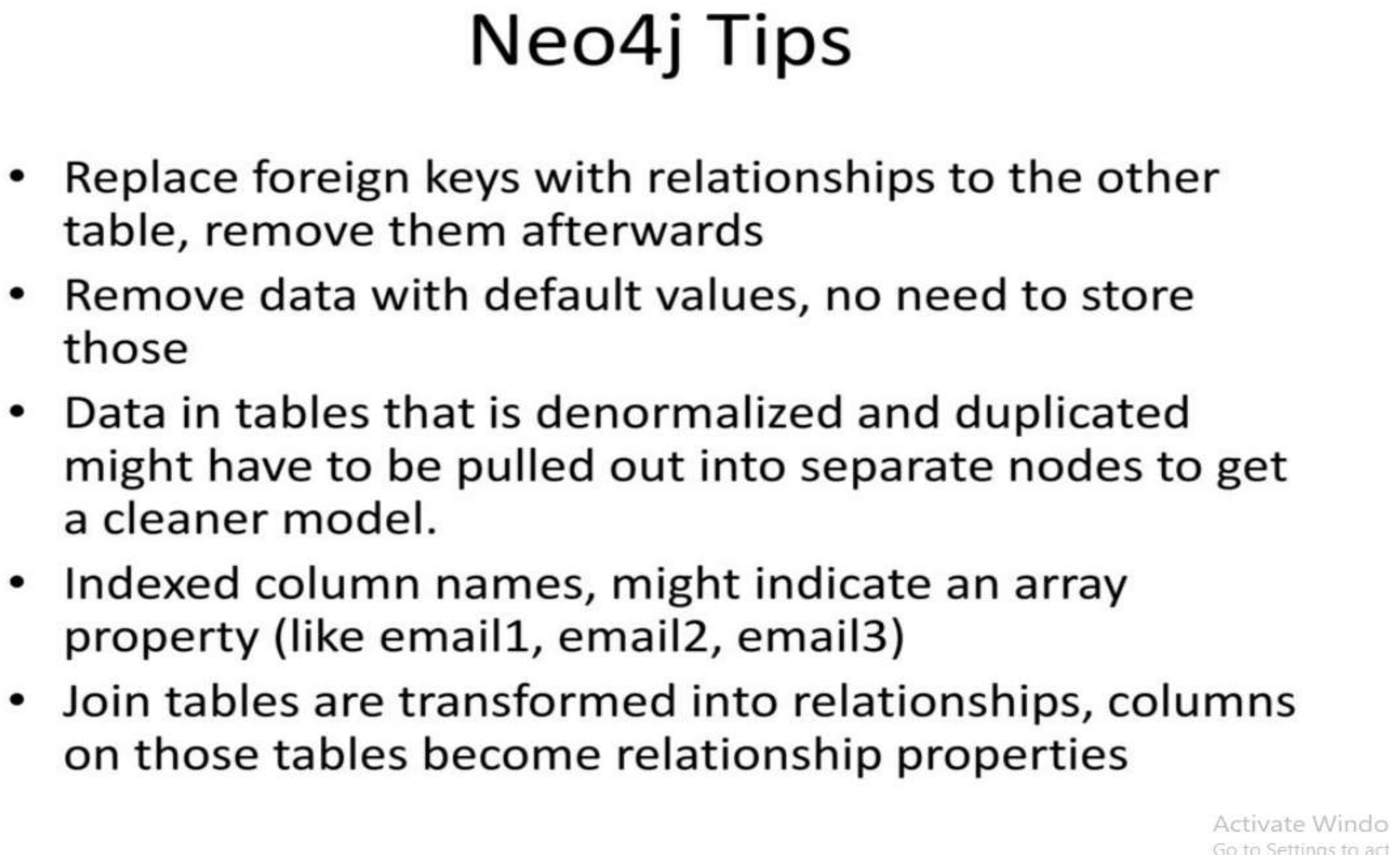
Relational Databases




Graph Databases
815
Alice
815
111
815
119
815
181
111
4FUTURE
119
P0815
181
A42
Nodes
Alice
:Person
:BELONGS_TO
:BELONGS_TO
:BELONGS_TO
4FUTURE
:Department
P0815
:Department
A42
:Department
Labels
Relationships

Neo4j Tips

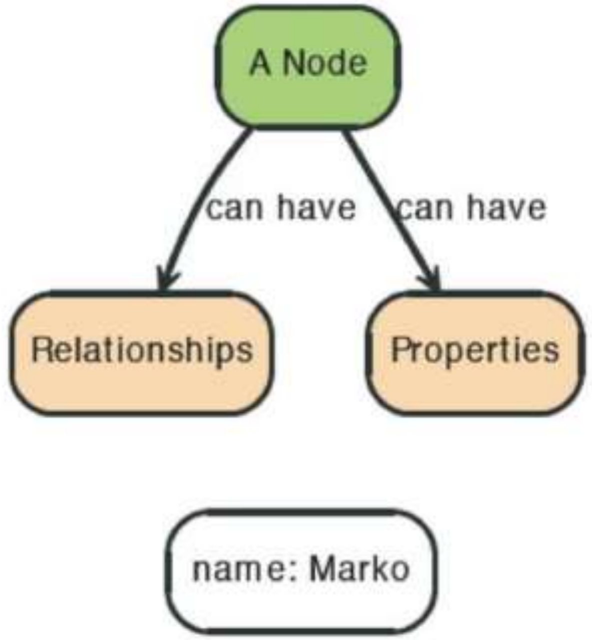
- Each entity table is represented by a label on nodes
- Each row in a entity table is a node
- Columns on those tables become node properties.
- Remove technical primary keys, keep business primary keys
- Add unique constraints for business primary keys, add indexes for frequent lookup attributes



Neo4j Tips

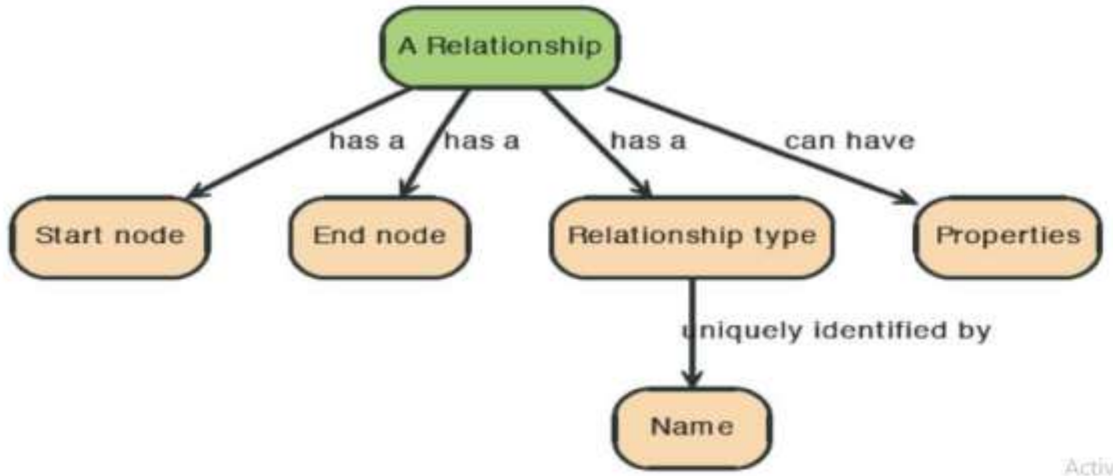
- Replace foreign keys with relationships to the other table, remove them afterwards
- Remove data with default values, no need to store those
- Data in tables that is denormalized and duplicated might have to be pulled out into separate nodes to get a cleaner model.
- Indexed column names, might indicate an array property (like email1, email2, email3)
- Join tables are transformed into relationships, columns on those tables become relationship properties

Node in Neo4j

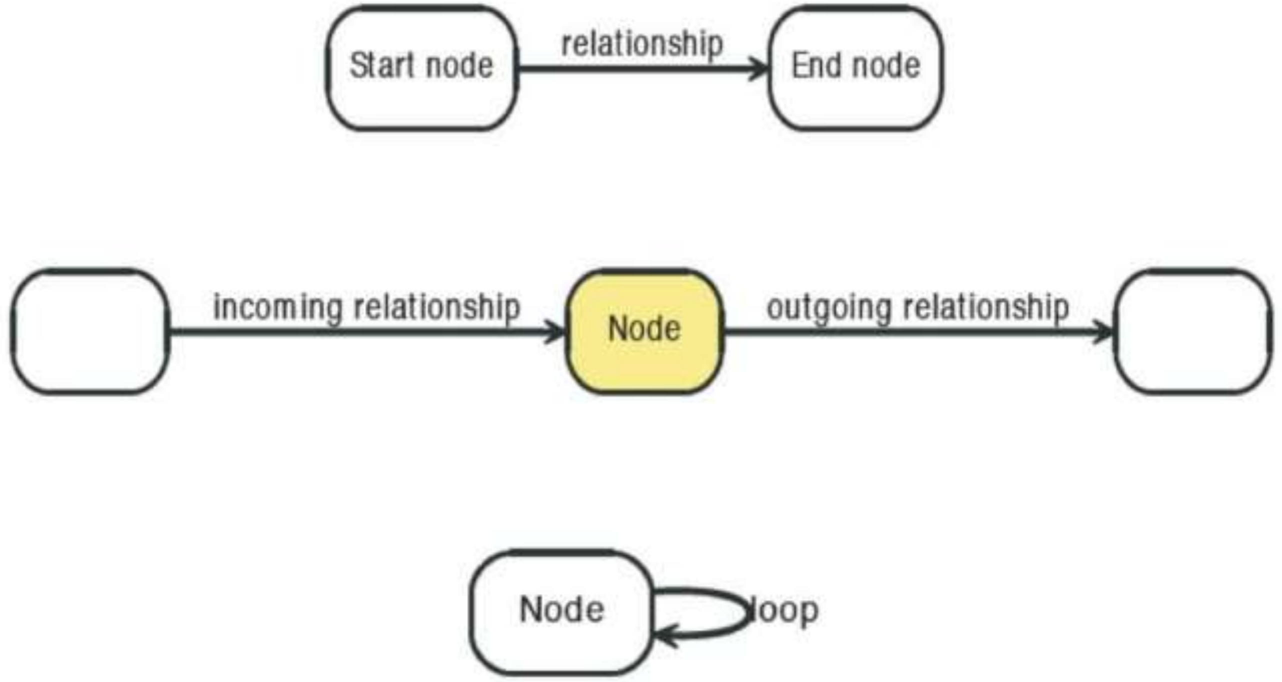


Relationships in Neo4j

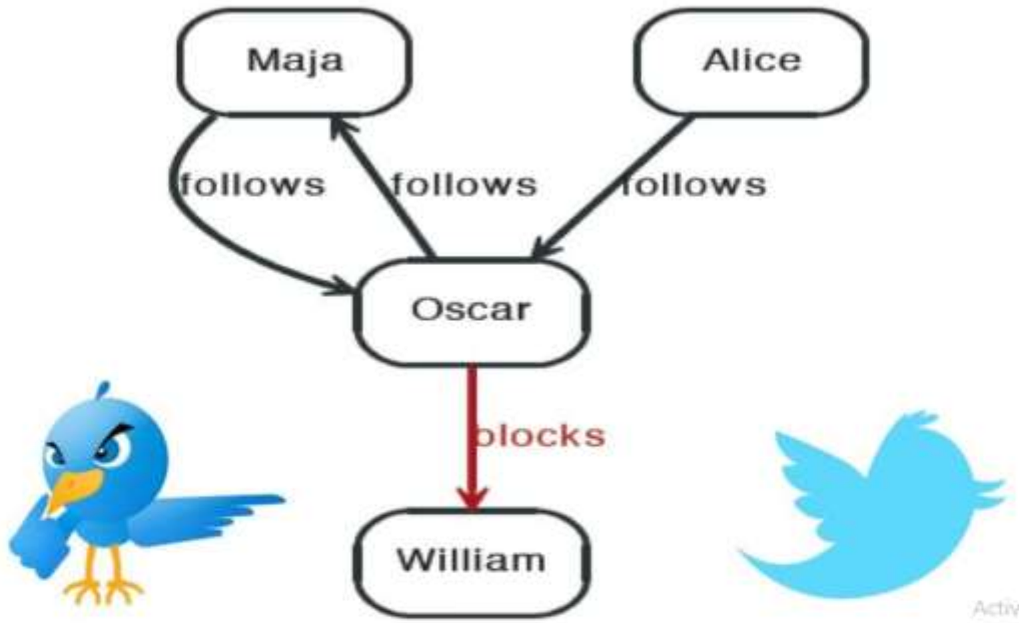
- Relationships between nodes are a key part of Neo4j.



Relationships in Neo4j



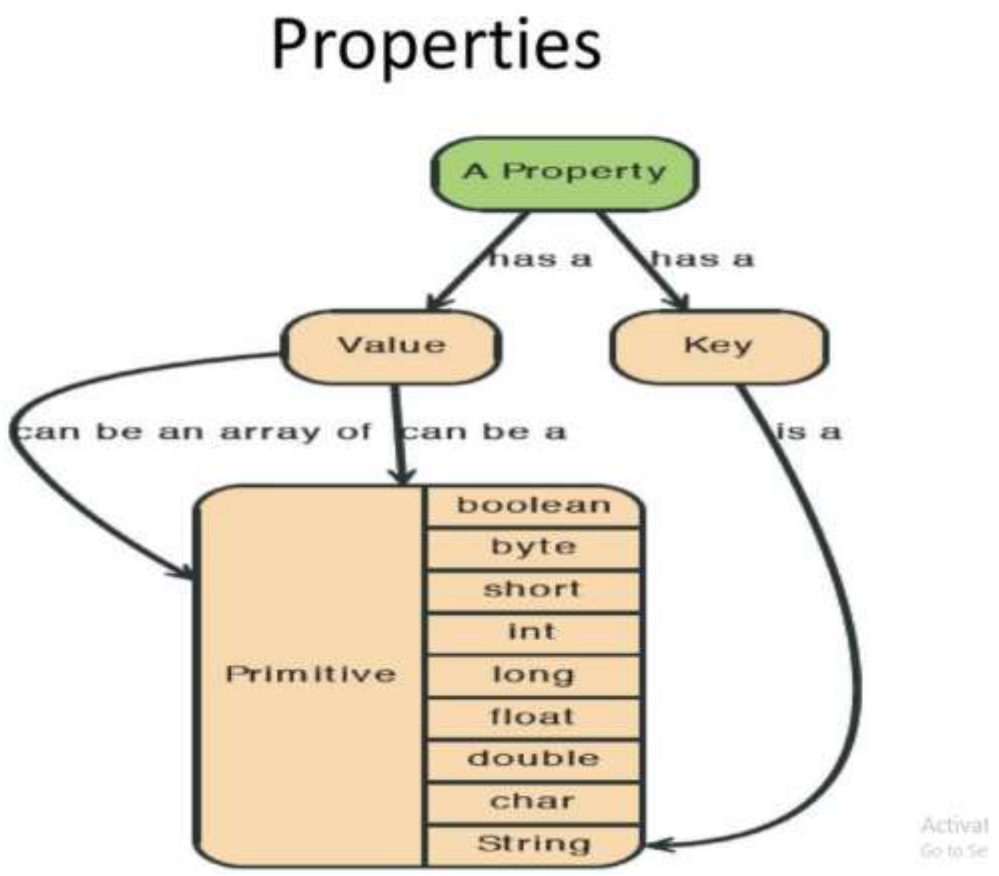
Twitter and relationships



Properties

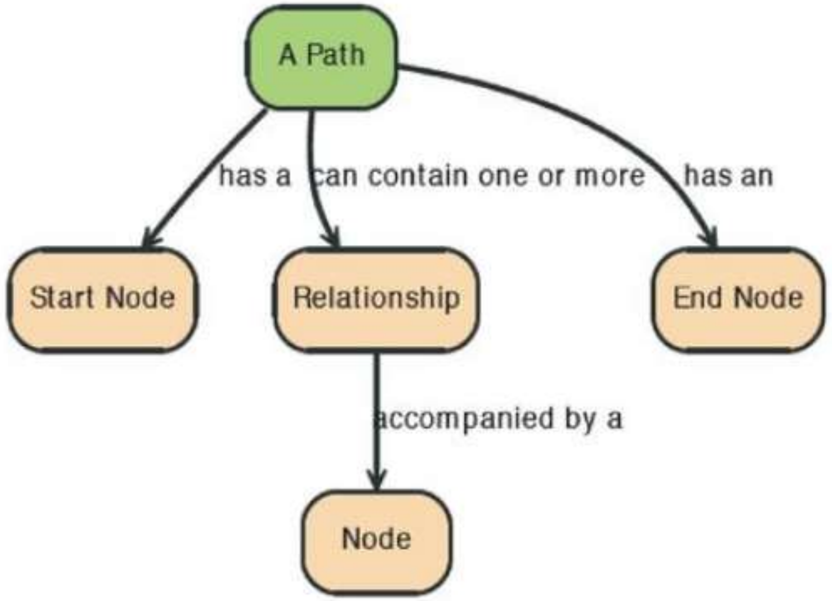
- Both nodes and relationships can have properties.
- Properties are key-value pairs where the key is a string.
- Property values can be either a primitive or an array of one primitive type.

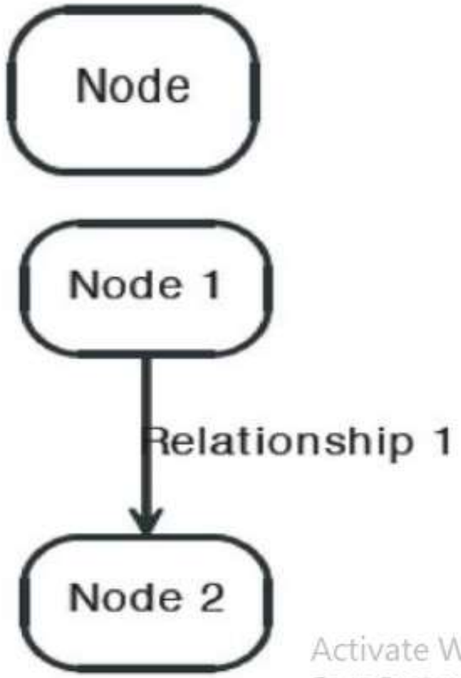
  For example String, int and int[] values are valid for properties.



Paths in Neo4j

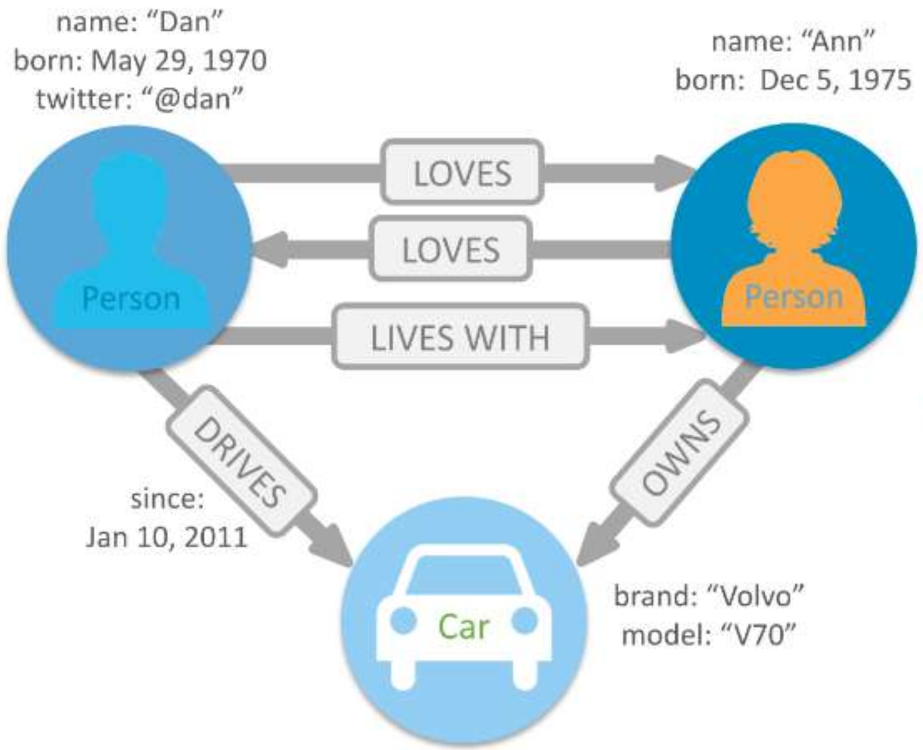
- A path is one or more nodes with connecting relationships, typically retrieved as a query or traversal result.



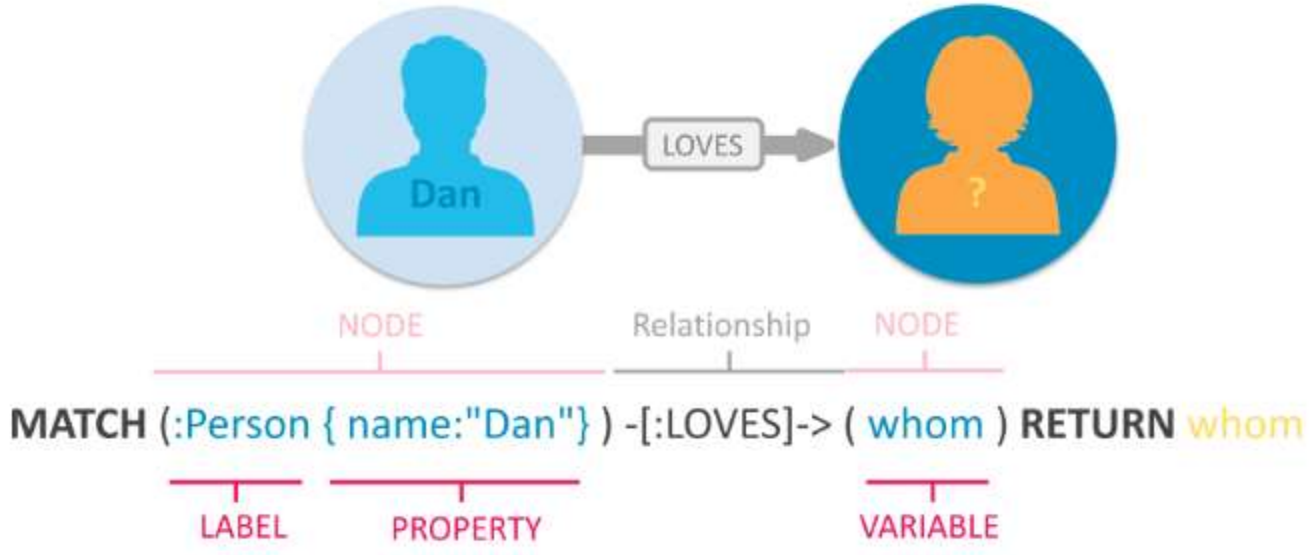


Traversals in Neo4j

- Traversing a graph means visiting its nodes, following relationships according to some rules.
- In most cases only a subgraph is visited, as you already know where in the graph the interesting nodes and relationships are found.
- Traversal API
- Depth first and Breadth first.



Building blocks of the property graph model

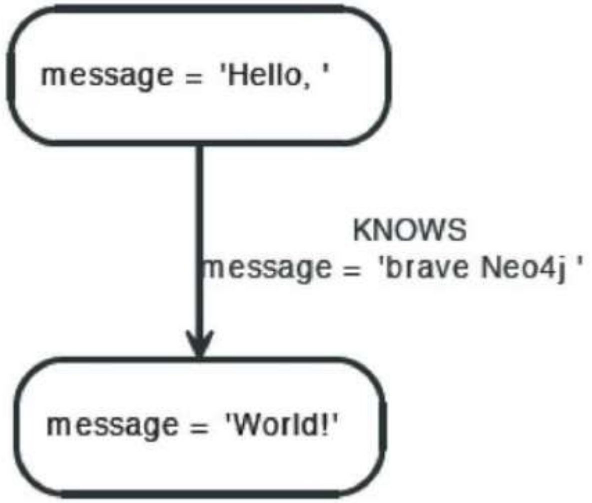


```
MATCH (:Person { name:"Dan"} ) -[:LOVES]-> ( whom ) RETURN whom
```

Creating a small graph

```
firstNode = graphDb.createNode();
firstNode.setProperty( "message", "Hello, " );
secondNode = graphDb.createNode();
secondNode.setProperty( "message", "World!" );

relationship = firstNode.createRelationshipTo( secondNode, RelTypes.KNOWS );
relationship.setProperty( "message", "brave Neo4j " );
```



Print the data

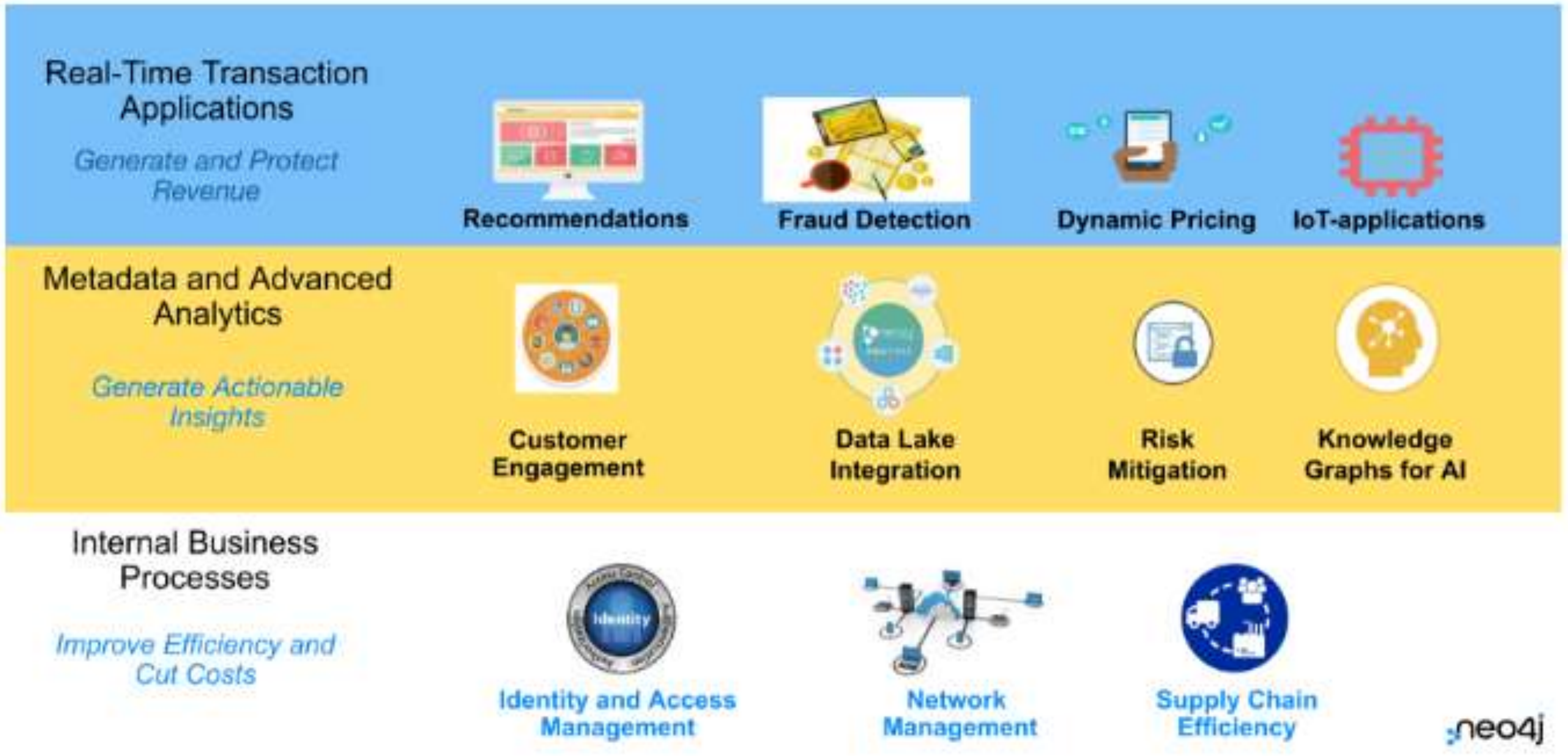
```
System.out.print( firstNode.getProperty( "message" ) );
System.out.print( relationship.getProperty( "message" ) );
System.out.print( secondNode.getProperty( "message" ) );
```

Remove the data

```
firstNode.getSingleRelationship( RelTypes.KNOWS, Direction.OUTGOING ).delete();
firstNode.delete();
secondNode.delete();
```

Traversing the Graph

```
private static Traverser getFriends( final Node person )
{
    return person.traverse( Order.BREADTH_FIRST,
            StopEvaluator.END_OF_GRAPH,
            ReturnableEvaluator.ALL_BUT_START_NODE, RelTypes.KNOWS,
            Direction.OUTGOING );
}
```


Real-Time Transaction Applications
Generate and Protect Revenue
Recommendations
Fraud Detection
Dynamic Pricing
IoT-applications
Metadata and Advanced Analytics
Generate Actionable Insights
Customer Engagement
Data Lake Integration
Risk Mitigation
Knowledge Graphs for AI
Internal Business Processes
Improve Efficiency and Cut Costs
Identity and Access Management
Network Management
Supply Chain Efficiency
neo4j

ANY Questions?