Storage Fundamentals 2

Group Members

Khondaker Tasnia Hoque (1205)

Sifat Sikder (1221)

Nazmul Hasan Rupu (1272)

01

Object Storage

- Data storage architecture for storing unstructured data
- Sections data into units as objects
- Stores them in a structurally flat data environment.
- Each object includes the data, metadata, and a unique identifier that applications can use for easy access and retrieval.

Unstructured Data

Unstructured data is information that is not arranged according to a preset data model or schema, and therefore cannot be stored in a traditional relational database or RDBMS.

Examples :

- Text
- Multimedia
- Messages
- Videos
- Photos
- Audio files.

Flat Data Environment

Flat file databases store plain text records and binary files that are needed for a specific purpose in a single directory for easy access and transfer.

A flat file is a collection of data stored in a two-dimensional database in which similar yet discrete strings of information are stored as records in a table. The columns of the table represent one dimension of the database, while each row is a separate record.

How Does It Work

- With object storage, the data blocks of a file are kept together as an object, together with its relevant metadata and a custom identifier, and placed in a flat data environment known as a storage pool.
- When you want to access data, object storage systems will use the unique identifier and the metadata to find the object you need, such as an image or audio file.
- You can also customize metadata, allowing you to add more context that is useful for other purposes, such as retrieval for data analytics.
- You can locate and access objects using RESTful APIs, HTTP, and HTTPS to query object metadata. Since objects are stored in a global storage pool, it’s fast and easy to locate the exact data you need.

File vs Block vs Object Data Storage

File Storage

Structure	File storage organizes data into files and folders, resembling a traditional file system. It uses a hierarchical structure, where files are stored in directories.
Use Cases	File storage is suitable for shared file systems, home directories, and network-attached storage (NAS) solutions. It is commonly used for collaborative work, file sharing, and general-purpose file storage.
Scalability	Scalability can be limited when compared to object storage, especially in large-scale deployments.
Examples	Windows File Server, NFS-based NAS solutions.

File vs Block vs Object Data Storage

Block Storage

Structure	Block storage divides data into fixed-sized blocks and manages them as individual storage volumes. It doesn't have a file hierarchy or structure.
Use Cases	Block storage is commonly used for databases, virtual machines (VMs), and applications that require low-level access to storage. It provides high-performance storage but lacks the data management features of file and object storage.
Scalability	Scalability can be achieved by adding more blocks or volumes, but management can be complex at scale
Examples	Storage Area Networks (SANs), Amazon Elastic Block Store (EBS).

File vs Block vs Object Data Storage

Object Storage

Structure	Object storage stores data as objects, each with its unique identifier, metadata, and data. Objects are organized in a flat namespace without a traditional file hierarchy.
Use Cases	File storage is suitable for shared file systems, home directories, and network-attached storage (NAS) solutions. It is commonly used for collaborative work, file sharing, and general-purpose file storage.
Scalability	Object storage is ideal for storing and managing vast amounts of unstructured data, such as media files, backups, archives, and data used by cloud-native applications. Highly scalable.
Examples	Amazon S3, Google Cloud Storage, and various on-premises object storage solutions.

What are the benefits of object storage?

01 Massive scalability

02 Reduced complexity

03 Searchability

04 Cost efficiency

02

Wide Column Database

- A type of NoSQL database that stores data in a column-family format
- Organizes data in a way that allows for a more efficient storage of data and faster query performance.
- Well suited for data warehousing and business intelligence applications

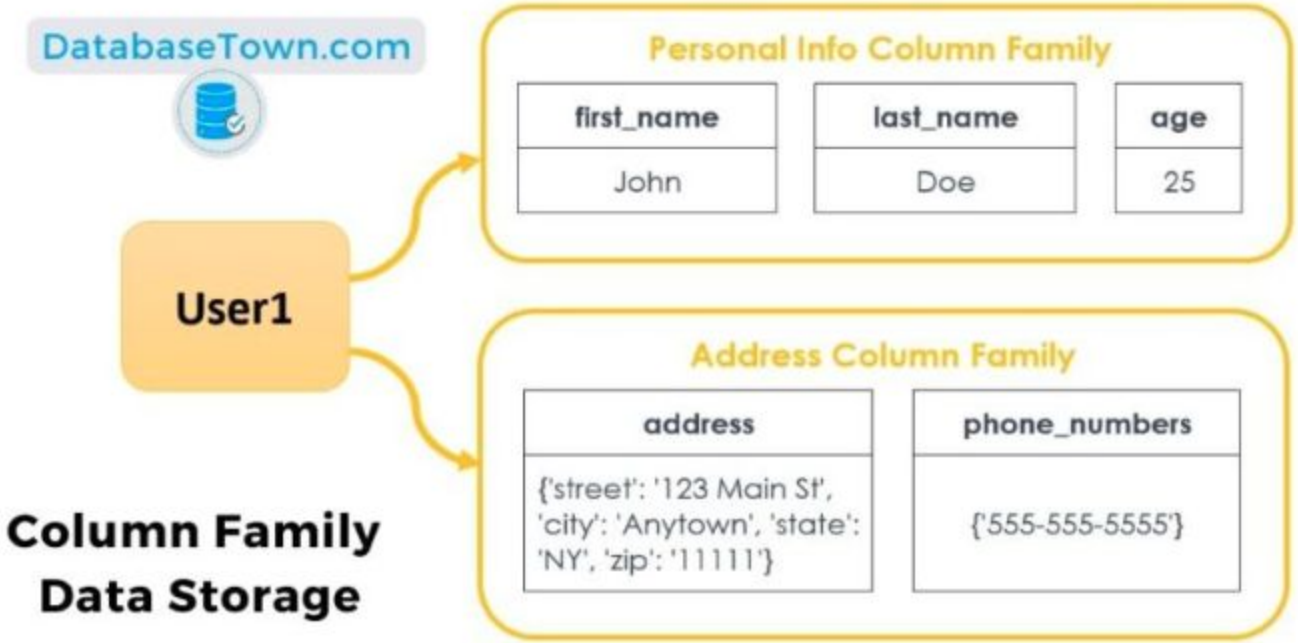
NoSQL Database

NoSQL databases (aka "not only SQL") are non-tabular databases and store data differently than relational tables.

Example : MongoDB

Wide Column Database

A wide-column database (also known as a column-family database) is a type of NoSQL database that stores data in a column-family format.



Wide-column database use cases

- Big data
- Data Warehousing
- OLAP (Online Analytical Processing)
- Real-time analytics
- Cloud-based analytics
- IoT (Internet of things)

What are the benefits of Wide Column Database

01 Scalability

02 Schema Flexibility

03 High Write Throughput

04 Availability and Fault Tolerance

05 NoSQL Features

What are the disadvantages of Wide Column Database

01 Complex Querying

02 Data Modeling Challenges

03 Size Overhead

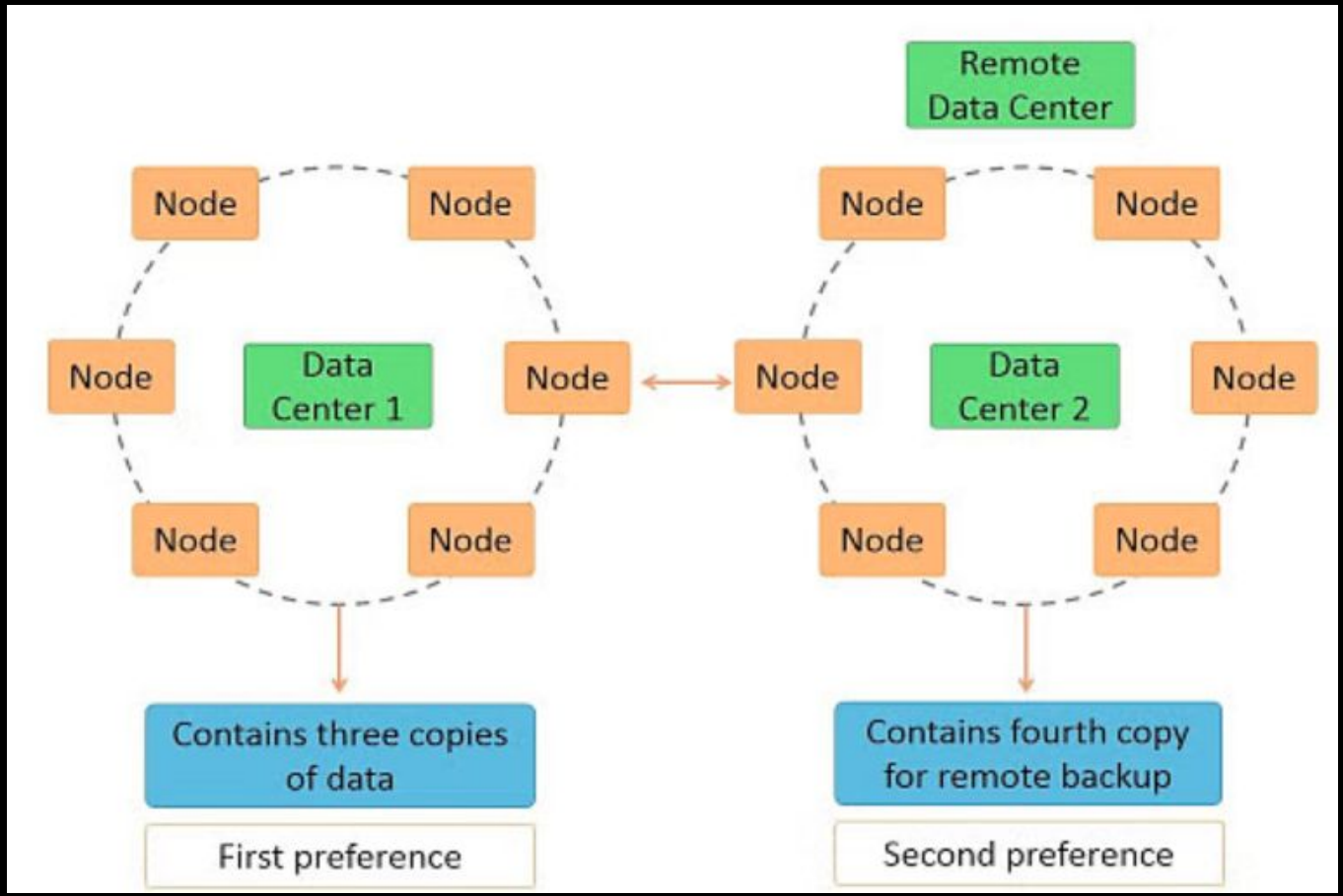
04 Less Mature Ecosystem

05 Limited Querying Capabilities

Example-Cassandra

- No master or slave nodes
- Nodes are logically distributed like a ring.
- Data is automatically distributed across all the nodes
- Data is kept in memory and lazily written to the disk

Example-Cassandra Architechture



03

Time Series database

- Time series data is a sequence of data points indexed in time order.
- A time series database is a type of database specifically designed for handling time-stamped or time-series data.
- Data points typically consist of successive measurements made from the same source over a fixed time interval
- Used to track change over time.

Time Series Database-Demo

ID	Timestamp	Air quality	Temperature
1	1640120331	Poor	15
2	1640120332	Poor	16
3	1640120333	Poor	13
4	1640120334	Good	15

Examples

- Electrical activity in the brain
- Stock prices
- Sensor data
- Weather data
- Website activity data

Uniqueness

- Very high write throughput
- Regular and irregular writes
- Data needs to be highly compressed
- Large range scans of many records
- Write to latest time entry only
- Native support for summaries, aggregation & rollups (more later)

What are the Advantages of Time Series Database

01 Efficient Data Storage

02 Fast Data Retrieval

03 Scalability

04 Data Retention Policies

05 Built-in Time-Series Functions

What are the Disadvantages of Time Series Database

01 Niche Use Case

02 High Cost

03 Limited General-Purpose Features

Time Series DBs

- QuestDB
- Timescale DB
- Influx DB
- Amazon Timestream
- Cassandra (NoSQL)

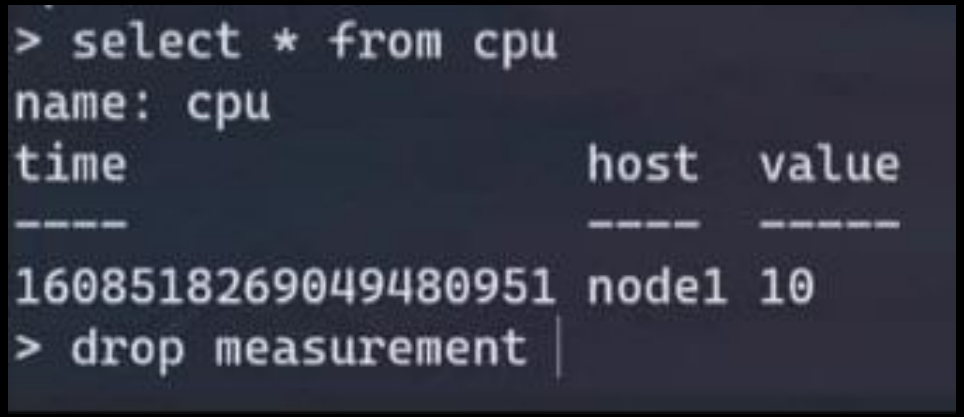
Example- InfluxDB

- Written in GO language
- Optimized for fast storage and retrieval of time
- Schema less database
- Tables of RDBMS are known by measurements in influxDB

Example- InfluxDB-Cont.

```
Connected to http://localhost:8086 version 1.6.4
InfluxDB shell version: 1.6.4
> show databases
name: databases
name
----
_internal
> create database devopsjourney
> show databases
name: databases
name
----
_internal
devopsjourney
> use devopsjourney
Using database devopsjourney
> show measurements
> insert cpu,host=node1 value=10
> show measurements
name: measurements
name
----
cpu
>
```

Example- InfluxDB-Cont.

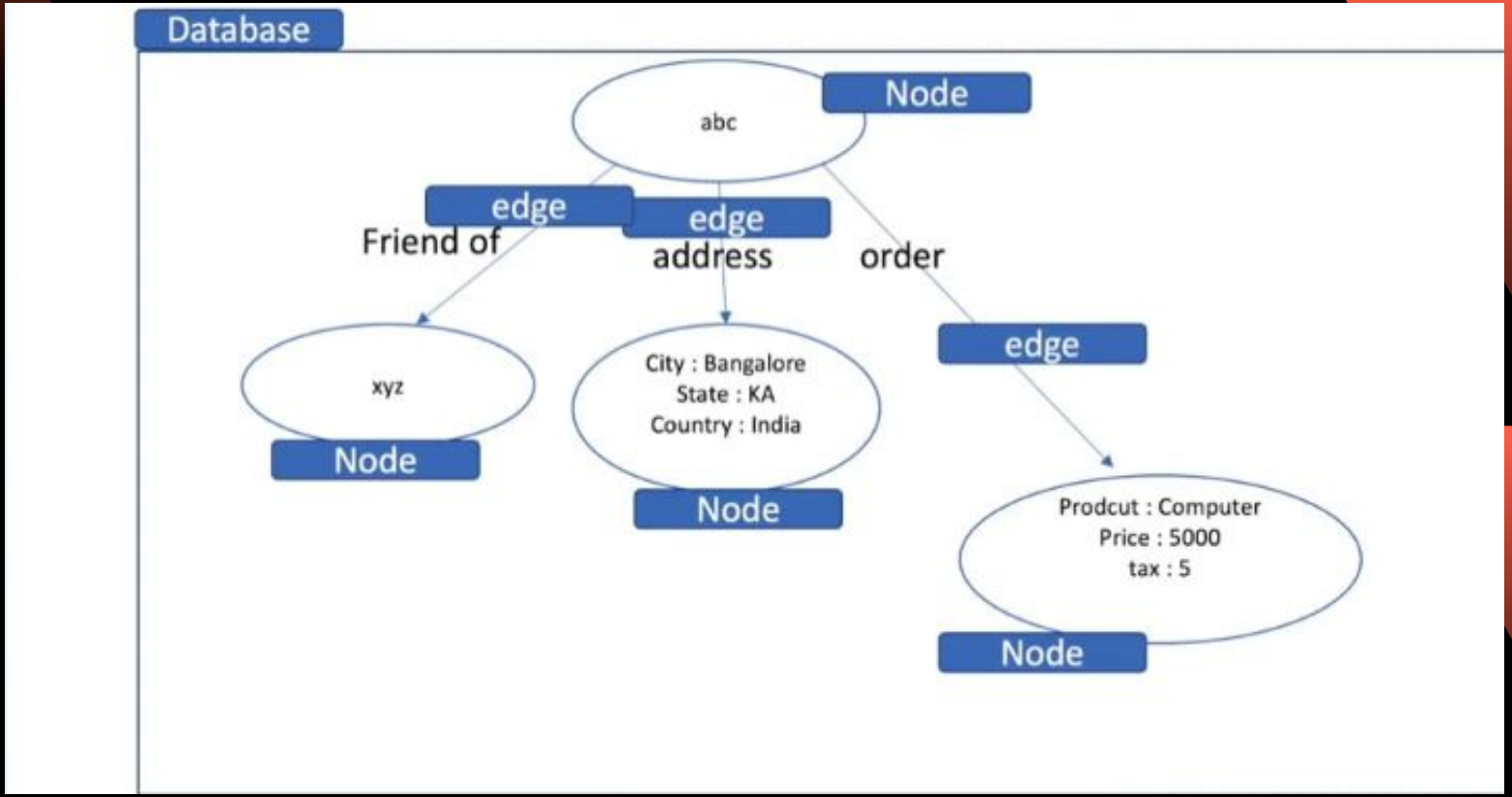


04

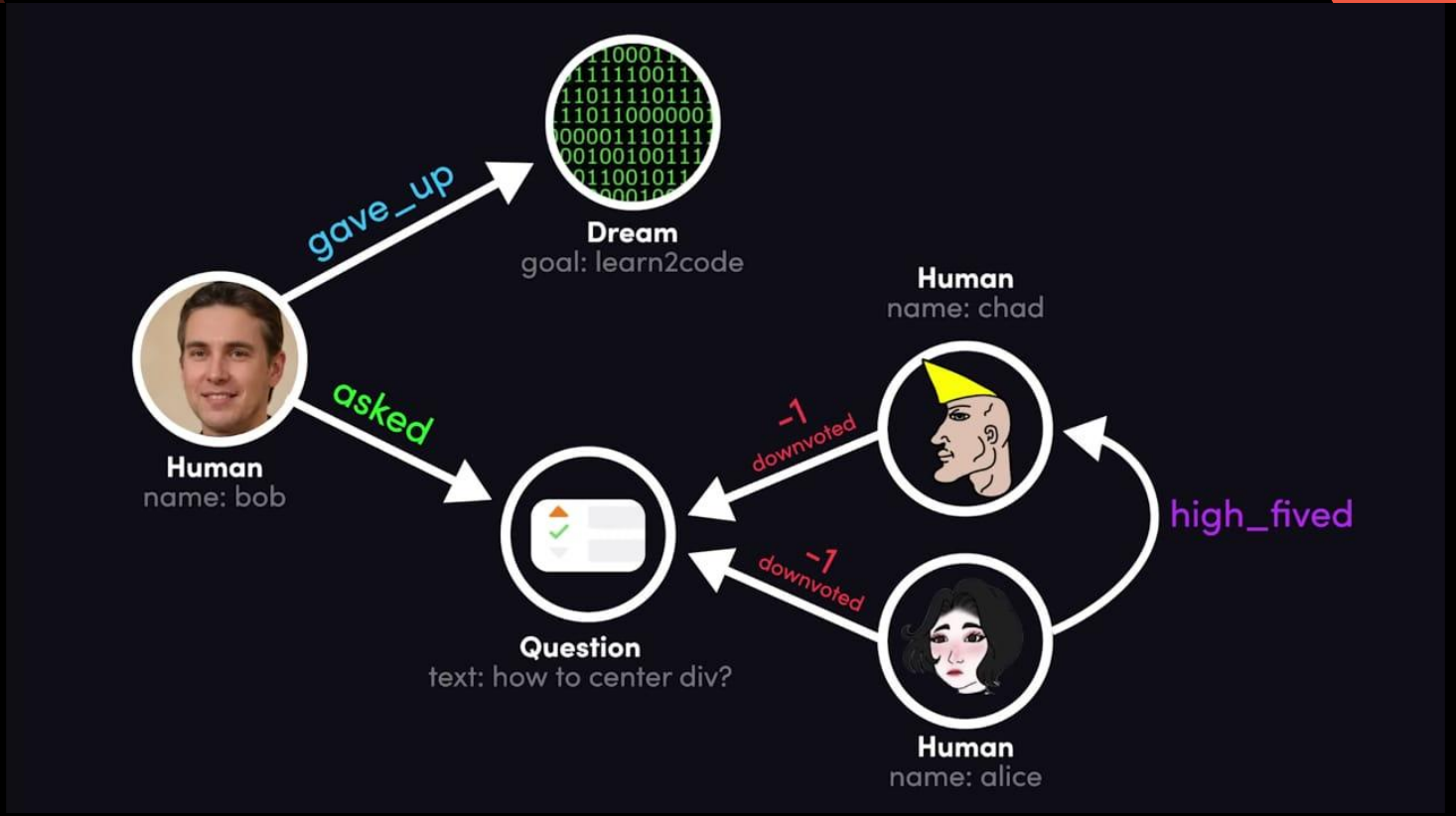
Graph database

- A graph database (GDB) is a database that uses graph structures for semantic queries with nodes, edges, and properties to represent and store data.
- Nodes typically store information about entities (people, places, or various things) which are stored into relational databases
- Edges store information about the relationship between nodes

Example

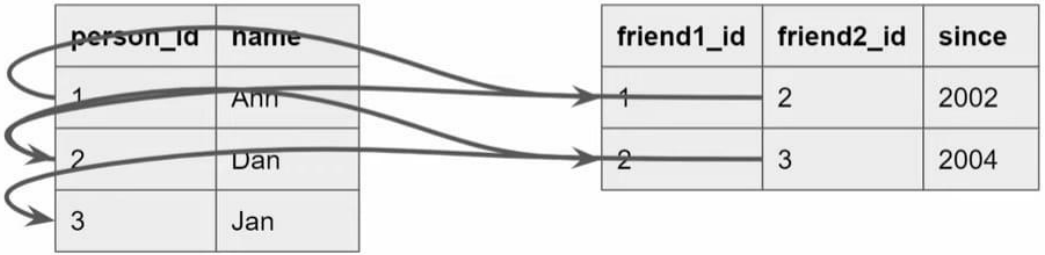

Database
abc
Node
edge
edge
Friend of
address
order
xyz
Node
City : Bangalore
State : KA
Country : India
Node
edge
Prodcut : Computer
Price : 5000
tax : 5
Node

Example-2


gave_up
Dream
goal: learn2code
Human
name: chad
asked
Human
name: bob
-1
downvoted
high_fived
-1
downvoted
Question
text: how to center div?
Human
name: alice

Graph Database vs RDBMS

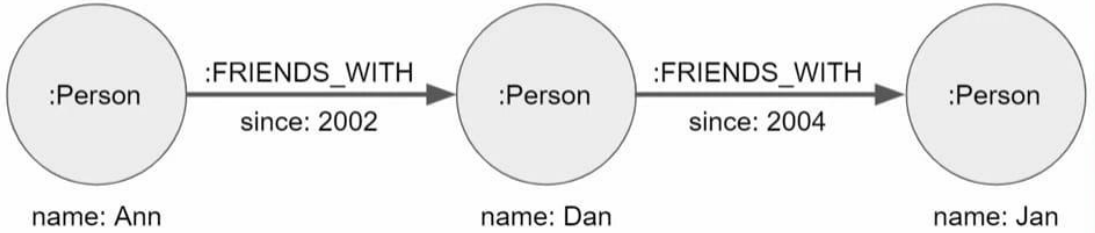
Are Ann, Dan and Jan connected?



person_id	name
1	Ann
2	Dan
3	Jan

friend1_id	friend2_id	since
1	2	2002
2	3	2004

How are Ann, Dan and Jan connected?



Use Cases

Fraud Detection & Analytics

Real-time analysis of data relationships is essential to uncovering fraud rings and other sophisticated scams before fraudsters and criminals cause lasting damage.

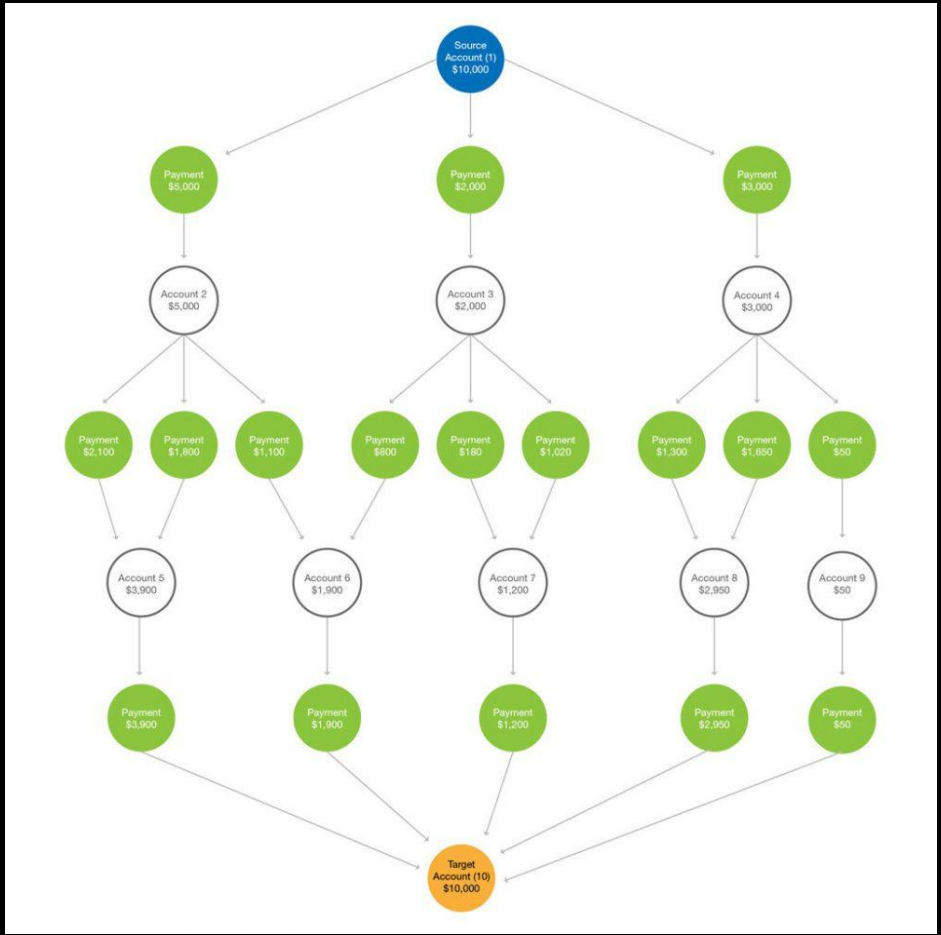
Network and Database Infrastructure Monitoring for IT Operations

Graph databases are inherently more suitable than RDBMS for making sense of complex interdependencies central to managing networks and IT infrastructure.

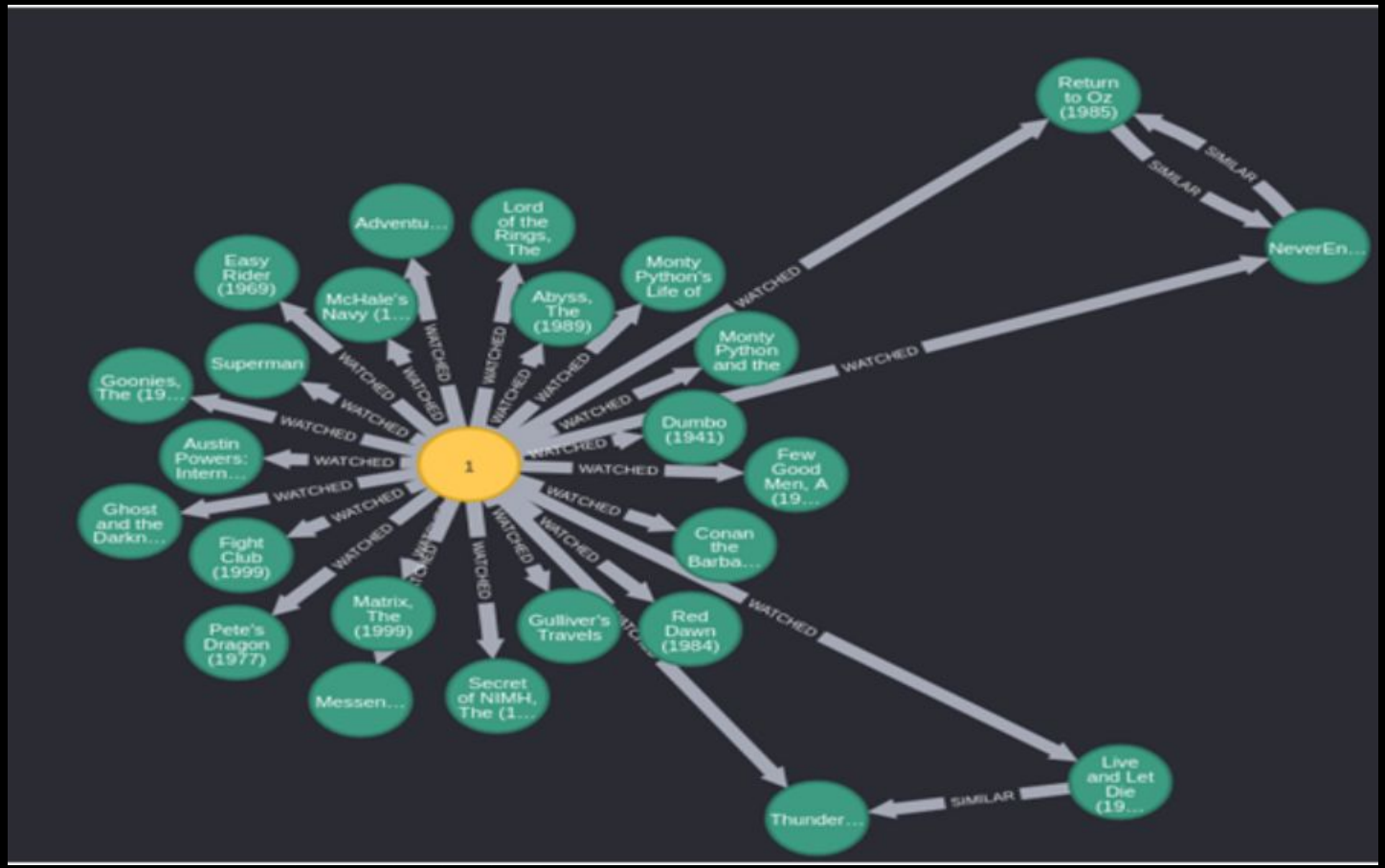
Recommendation Engine & Product Recommendation System

Graph-powered recommendation engines help companies personalize products, content and services by leveraging a multitude of connections in real time.

Fraud Detection



Recommendation



Types of Graph Database

property graphs : The property graph focuses on analytics and querying

RDF graphs: RDF graph emphasizes data integration.

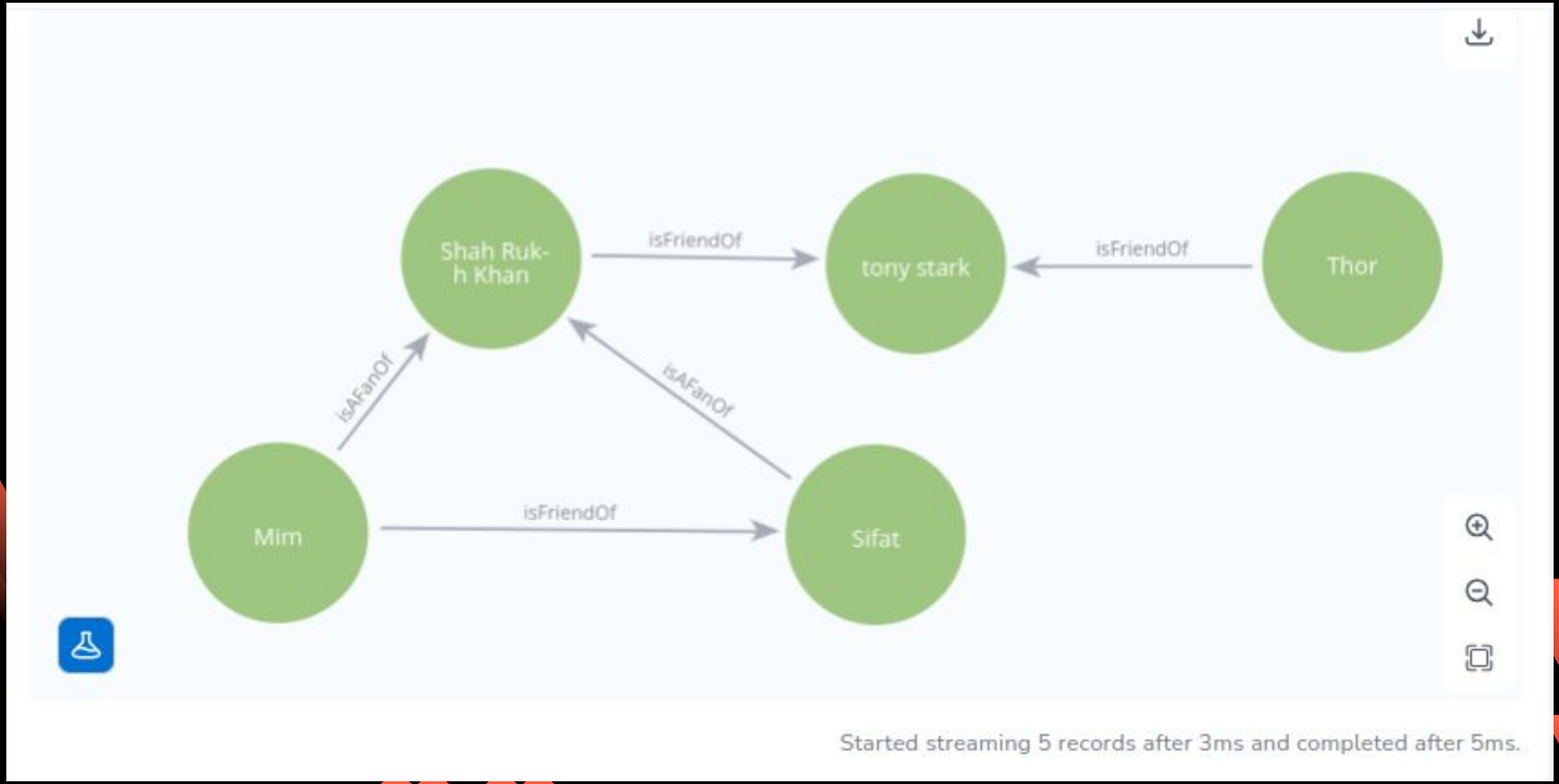
Pros and Cons of Graph Database

Pros	Cons
Efficient for Complex Relationships	Performance Degradation with High Node Degrees
High Performance	Storage Overhead for Dense Graphs
Flexible Schema and scalability	Indexing Complexity
Natural Representation	High Cost

Example-Neo4j

- Creating a Node: CREATE (srk:User{name:"Shah Rukh Khan"})
- Updating a Node:
  - `match (nila:User {name:"Nila"})`
  - `set nila.name= "Mim"`
- Creating a Relation:
  - match (thor:User {name:"Thor"}), (ironman:User {name:"tony stark"})
  - create (thor) - [:isFriendOf] -> (ironman)
- Delete a Node:
  - match (srk:User)
  - where id(srk) = 5
  - detach delete srk

Example-Neo4j-Created Graph



THANK YOU!!!