Big Data and NoSQL Databases

Big Data and NoSQL Databases

Big Data refers to vast amounts of data from diverse sources, challenging traditional storage and processing methods. NoSQL databases offer flexible, scalable solutions for handling Big Data efficiently. They prioritize flexibility, scalability, and availability over strict consistency. NoSQL databases come in different types, including document, key-value, column-family, and graph databases, each tailored to specific data storage and retrieval needs. These databases enable organizations to manage and analyze large volumes of data effectively, supporting agile development and adaptation to changing data structures.

Characteristics of Big Data (volume, velocity, variety, veracity)

- **Volume**

Refers to the vast amount of data generated and collected. It encompasses the size of data sets, which can range from terabytes to petabytes or even exabytes. This characteristic emphasizes the need for scalable storage and processing solutions to handle large volumes of data efficiently.

- **Velocity**

Refers to the speed at which data is generated, collected, and processed. With the proliferation of sensors, social media, and other sources, data is generated at an unprecedented rate. Real-time or near-real-time pro

Characteristics of Big Data (volume, velocity, variety, veracity)

- **Veracity**

Refers to the reliability and trustworthiness of the data. Big data sources often include noisy, incomplete, or inconsistent data, which can affect the accuracy and validity of analysis results. Ensuring data quality and integrity is crucial for making informed decisions.

- **Variety**

Refers to the diversity of data types and sources. Big data encompasses structured, semi-structured, and unstructured data from various sources such as text, images, videos, sensor data, social media posts, and more. Managing and analyzing such diverse data types pose significant challenges

Types of NoSQL databases (document, key-value, column-family, graph)

- **Document Databases**

These databases store and manage semi-structured data as documents, typically in JSON or BSON format. Each document can contain nested data structures, making them suitable for use cases where flexibility and scalability are essential.

- **Key-Value Stores**

Key-value databases store data as a collection of key-value pairs, where each unique key is associated with a value. They are highly scalable and efficient for simple retrieval and storage operations.

Characteristics of Big Data (volume, velocity, variety, veracity)

- **Column-Family Stores**

Also known as wide-column stores, these databases organize data into columns rather than rows, allowing for efficient storage and retrieval of data with different attributes. They are particularly suitable for use cases involving large-scale data processing and analysis.

- **Graph Databases**

Graph databases represent data as nodes, edges, and properties, enabling efficient storage and traversal of relationships between entities. They are well-suited for use cases involving complex interconnections and network analysis.

CAP theorem and its implications.

1. **Consistency (C): Every read receives the most recent write or an error. This ensures that all nodes in the system have the same data at the same time.**
2. **Availability (A): Every request receives a response, without the guarantee that it contains the most recent write. In other words, the system remains operational even in the presence of network partitions or node failures.**
3. **Partition tolerance (P): The system continues to operate despite network partitions that may cause communication failures between nodes.**

CAP theorem and its implications.

1. **Trade-offs:** Distributed data systems must make trade-offs between consistency, availability, and partition tolerance. For example, in the event of a network partition (P), a system must choose between maintaining consistency (C) or ensuring availability (A).
2. **Design considerations:** Architects and developers must carefully consider the requirements of their application and choose a data system that aligns with those requirements. Depending on the use case, different compromises may be acceptable.

CAP theorem and its implications.

3. No one-size-fits-all solution: There is no one-size-fits-all solution in distributed data system design. Different applications may prioritize consistency over availability or vice versa, depending on their specific needs.

4. Complexity: Achieving both consistency and availability in the presence of partition tolerance can be complex and may require sophisticated algorithms and architectural designs.

ACID (Atomicity, Consistency, Isolation, Durability)

ACID Properties:

1. **Atomicity: Ensures that a transaction is treated as a single unit of operation, which either completes entirely or has no effect at all.**
2. **Consistency: Ensures that a transaction transforms the database from one valid state to another valid state, preserving data integrity and constraints.**

ACID (Atomicity, Consistency, Isolation, Durability)

ACID Properties:

3. **Isolation: Ensures that multiple transactions can operate concurrently without interfering with each other, providing the illusion that each transaction is executed in isolation.**
4. **Durability: Ensures that once a transaction is committed, its changes are permanent and survive system failures.**

BASE (Basically Available, Soft state,

ACID Properties:

3. **Isolation: Ensures that multiple transactions can operate concurrently without interfering with each other, providing the illusion that each transaction is executed in isolation.**
4. **Durability: Ensures that once a transaction is committed, its changes are permanent and survive system failures.**

BASE (Basically Available, Soft state,

BASE Properties:

- **Basically Available: Guarantees that the system remains operational and responsive to user requests even in the face of failures or network partitions. It sacrifices strong consistency to achieve high availability.**
- **Soft state: Allows the system's state to be transient or incomplete at times, meaning that the state of the system may change over time as updates propagate.**
- **Eventually Consistent: Acknowledges that the system may not always provide immediate consistency guarantees but ensures that all replicas converge to a consistent state over time, given enough time and absence of further updates.**

Use cases and applications of NoSQL Databases

1. **Big Data:** NoSQL databases are well-suited for handling large volumes of unstructured or semi-structured data commonly found in big data applications, such as social media analytics, sensor data processing, and log file analysis.
2. **Real-time Analytics:** NoSQL databases, particularly column-family stores and document databases, are used for real-time analytics applications where low-latency data processing and querying are essential, such as in fraud detection, recommendation systems, and IoT analytics.
3. **Content Management:** NoSQL databases are used for content management systems, where flexibility in storing and managing various types of content (text, images, videos) is required. Document databases are particularly useful for this use case.

Use cases and applications of NoSQL Databases

4. **E-commerce:** NoSQL databases are used in e-commerce applications for managing product catalogs, user profiles, and transaction data. They provide scalability and high availability, ensuring smooth operation during peak loads and traffic spikes.
5. **Personalization:** NoSQL databases are used in personalization applications, such as content recommendations and targeted advertising, where large amounts of user data need to be processed and analyzed in real-time to provide personalized experiences.
6. **Gaming:** NoSQL databases are used in online gaming applications for managing player profiles, game states, leaderboards, and in-game transactions. They provide high throughput and low latency, ensuring smooth gaming experiences for players.

Real Life Examples

Big Data:

1. **Social Media Analytics: Platforms like Facebook and Twitter analyze user interactions, comments, and trends to improve user experiences and target advertisements.**
2. **Sensor Data Processing: Smart cities use sensors to collect data on traffic flow, air quality, and energy consumption to optimize urban planning and resource allocation.**
3. **Business Intelligence: Retailers analyze customer purchase history, demographic data, and market trends to optimize inventory management and personalize marketing campaigns.**

Real Life Examples

NoSQL Databases:

1. **Document Databases (e.g., MongoDB): Used by companies like eBay for storing product information and user profiles, enabling fast and flexible data retrieval.**
2. **Key-Value Stores (e.g., Redis): Utilized by gaming platforms like Zynga to store user session data and leaderboards, supporting high-performance data access.**
3. **Column-Family Stores (e.g., Apache Cassandra): Adopted by Netflix for managing movie recommendation data and user preferences, ensuring scalability and fault tolerance.**
4. **Graph Databases (e.g., Neo4j): Employed by LinkedIn for analyzing social network connections and recommending professional connections, facilitating complex relationship queries.**

Conclusion

In summary, Big Data and NoSQL databases are integral components of today's data landscape. Big Data encompasses vast and diverse datasets, driving innovation across industries. NoSQL databases offer flexible, scalable solutions tailored to managing Big Data efficiently. Together, they empower organizations to extract value from massive datasets, fueling advancements in analytics, decision-making, and user experiences. Their synergy is crucial for unlocking the full potential of data-driven initiatives in the digital age.

References

Burri, M. (2023). The impact of digitalization on Global Trade Law. *German Law Journal*, *24*(3), 551–573. https://doi.org/10.1017/glj.2023.29

Chen, M., Mao, S., & Liu, Y. (2014). Big Data: a survey. *Mobile Networks and Applications, 19*(2), 171–209. https://doi.org/10.1007/s11036-013-0489-0

Gandomi, A. H., & Haider, M. (2015). Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management*, *35*(2), 137–144. https://doi.org/10.1016/j.ijinfomgt.2014.10.007

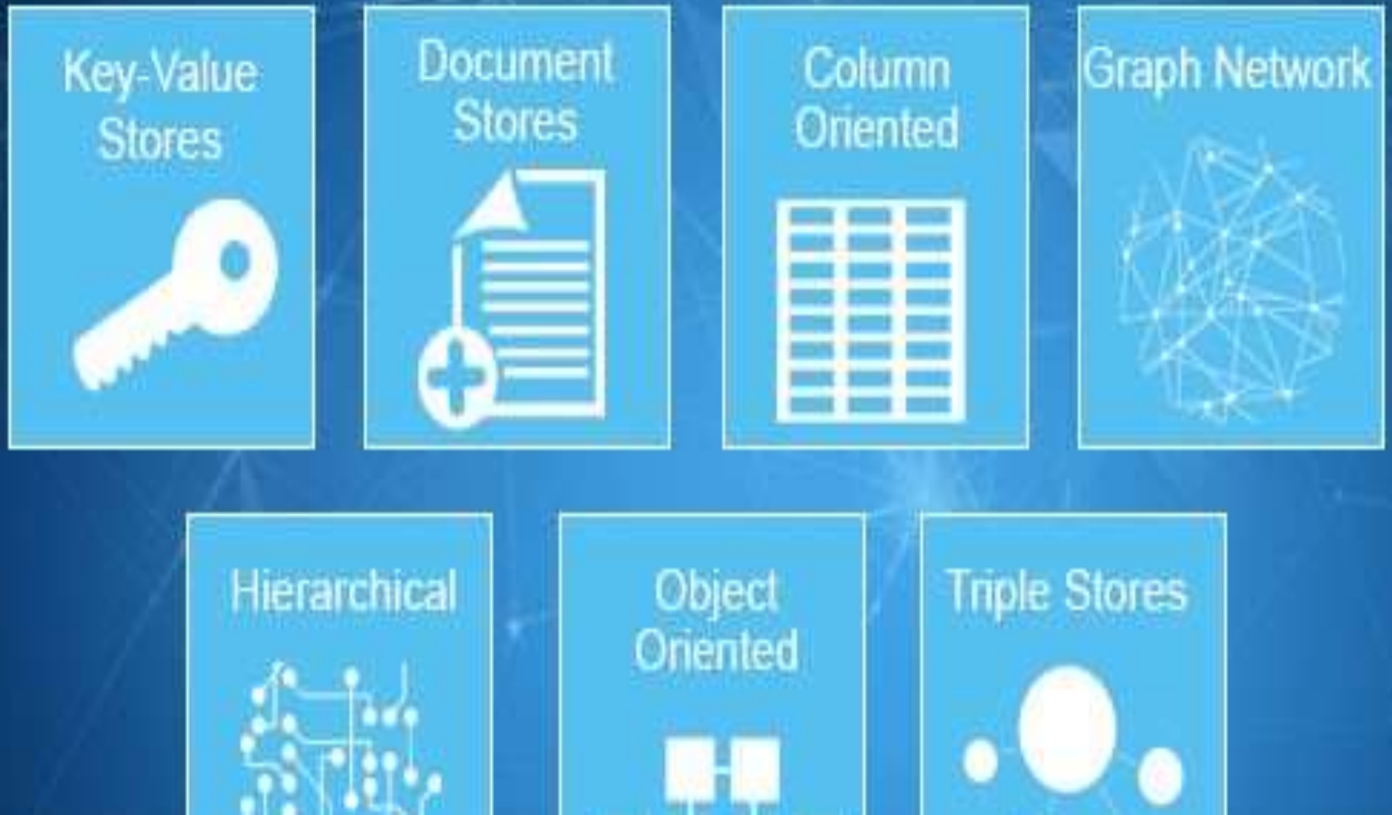
Haldorai, A., Ramu, A., Mohanram, S., & Chen, M. (2020). *2nd EAI International Conference on Big Data Innovation for Sustainable Cognitive Computing: BDCC 2019*. Springer Nature.

Malkhi, D. (2002). *Distributed computing: 16th International Conference, DISC 2002. Toulouse, France, October 28-30, 2002, Proceedings*. Springer Science & Business Media.

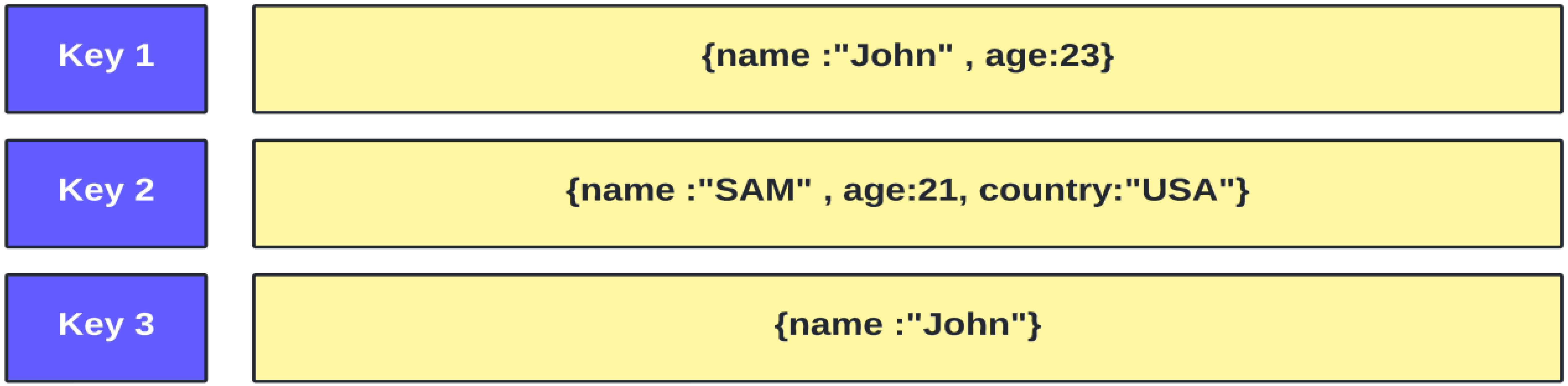
Raj, P. (2014). *Handbook of Research on Cloud Infrastructures for Big Data Analytics*. IGI Global.

Tjortjis, C. (2023). *Graph databases: Applications on Social Media Analytics and Smart Cities*. CRC Press.

Types of NoSQL Databases

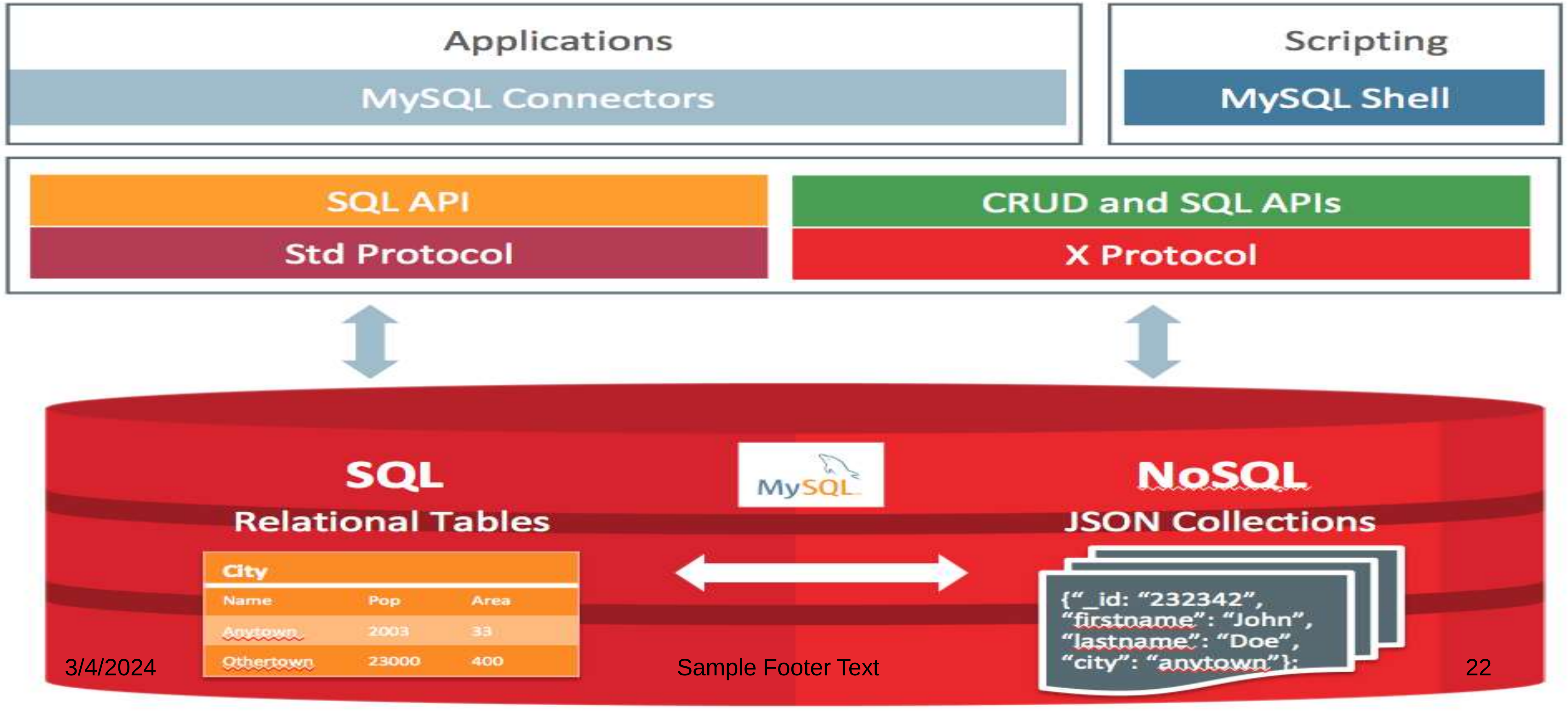


KEY VALUE-STORES



Each value can have different schema

DOCUMENT STORES

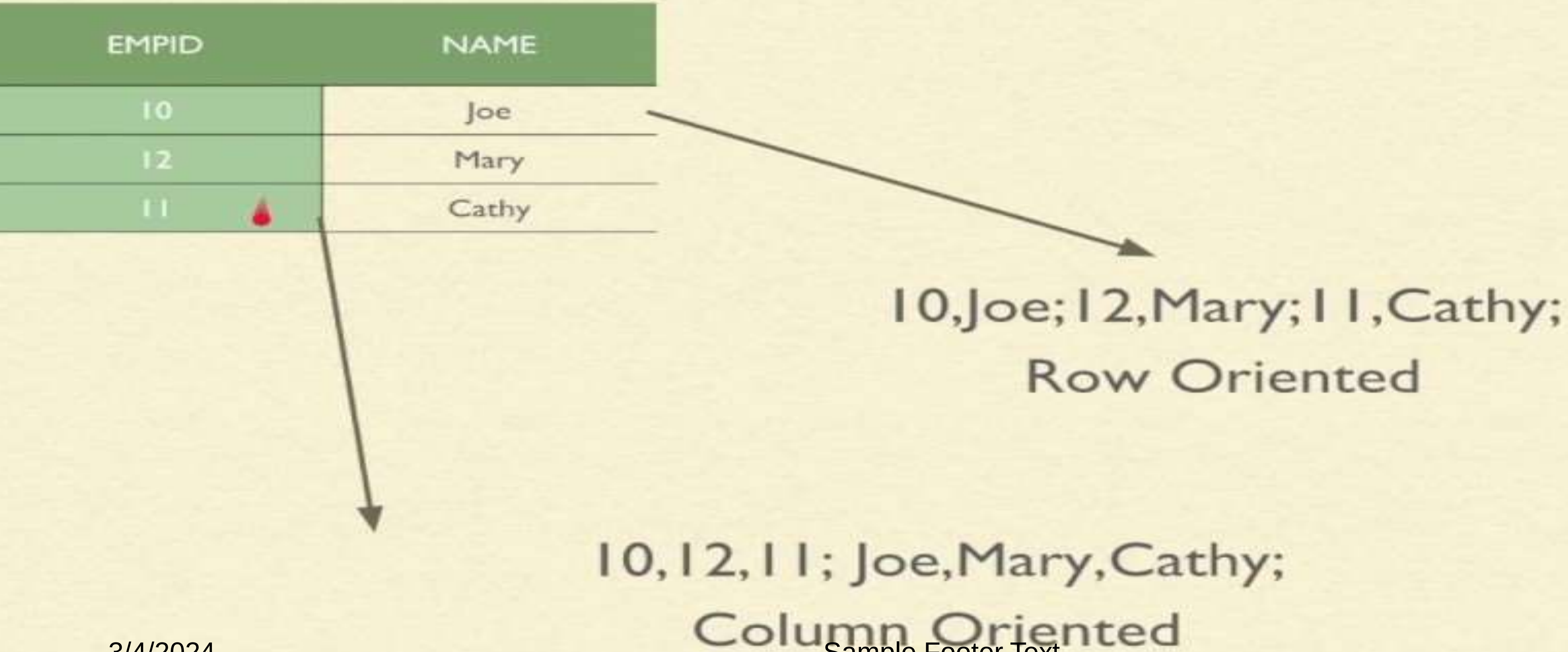


City		
Name	Pop	Area
Anytown	2003	33
Othertown	23000	400

```
{"_id: "232342",
"firstname": "John",
"lastname": "Doe",
"city": "anytown"};
```

Column Oriented Database

Data in columns stored nearby as opposed to the rows being nearby



EMPID	NAME
10	Joe
12	Mary
11	Cathy

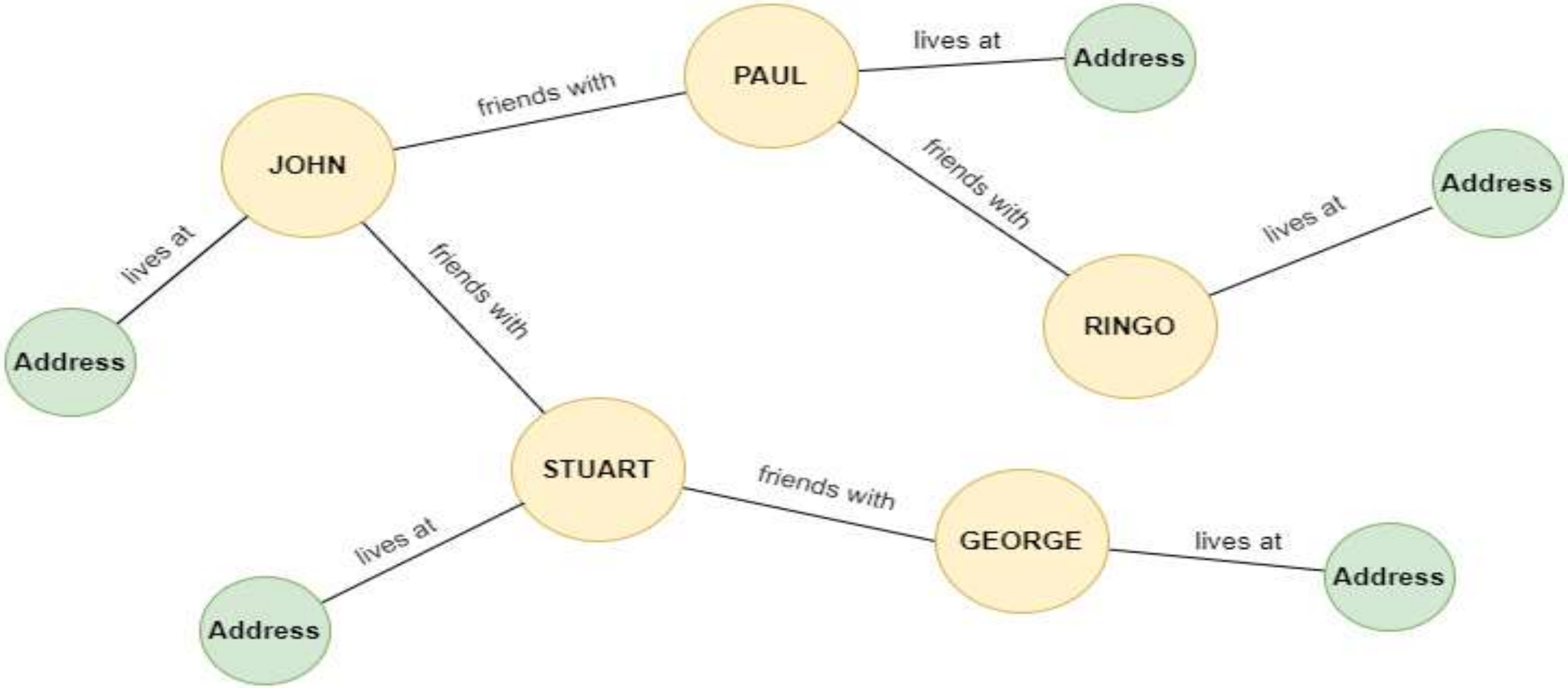
10,Joe;12,Mary;11,Cathy;

Row Oriented

10,12,11; Joe,Mary,Cathy;

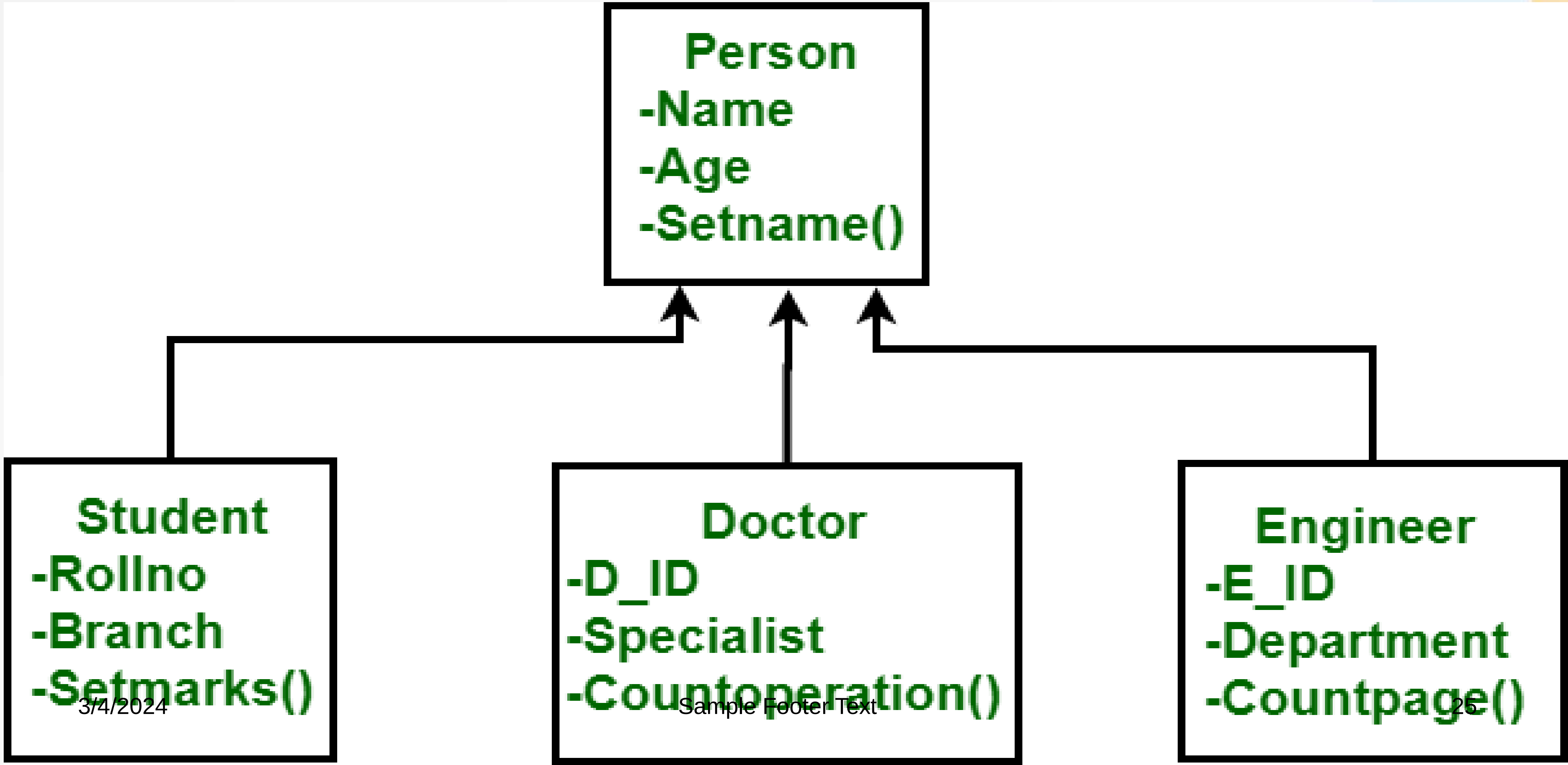
Column Oriented

GRAPH NETWORK



iq.opengenus.org

OBJECT ORIENTED



Triples Store as NoSQL

Mostly open source

Some distributed

API Not proprietary

Looks like SQL

Eventual consistency

Huge amount of data?

Garlik: 20 Billion triples (cluster, ~20 machines)

AllegroGraph: 1 Trillion (single, very large machine)