SQL AND NOSQL DATABASES

Subjects : Big Data Technologies

Presented by : Jenil Arvindbhai Paladiya

Matriculation Number : 4243558

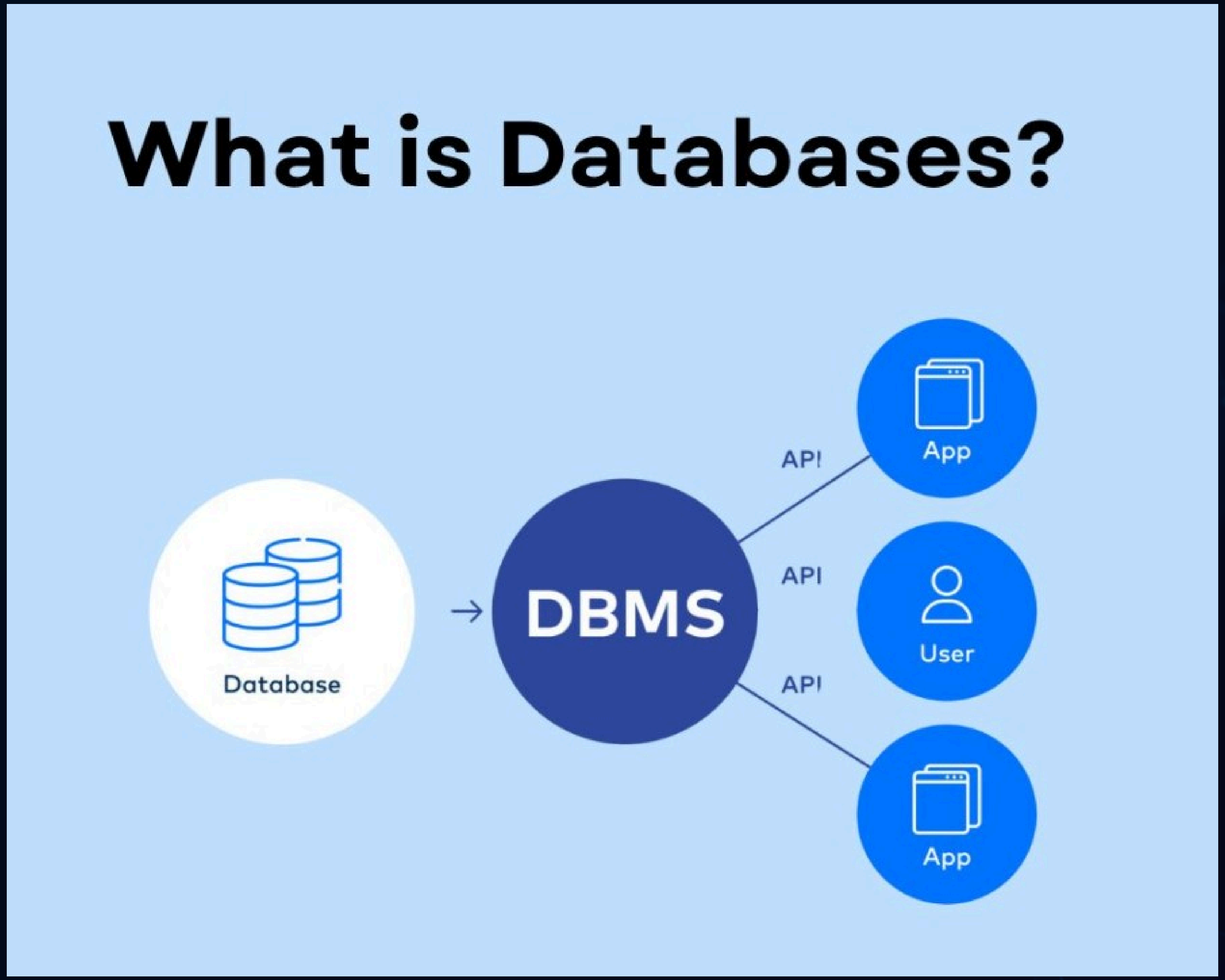
INDEX

WHAT ARE DATABASES?

Definition: Databases store, organize, and manage digital data essential for businesses, websites, and software applications

- Examples: Social media platforms, online stores, and banking applications.

1. Databases in Emerging Technologies:
- Artificial Intelligence (AI): Databases store the training data used to build AI models.
- Internet of Things (IoT): Databases manage the massive streams of data generated by connected devices, like smart home systems or industrial sensors.
2. Databases and Data Security:
- Databases play a critical role in protecting sensitive information, such as credit card details or personal identities, through encryption and access controls.
3. The Evolution of Databases:
- From traditional relational databases to modern cloud-based and distributed systems.



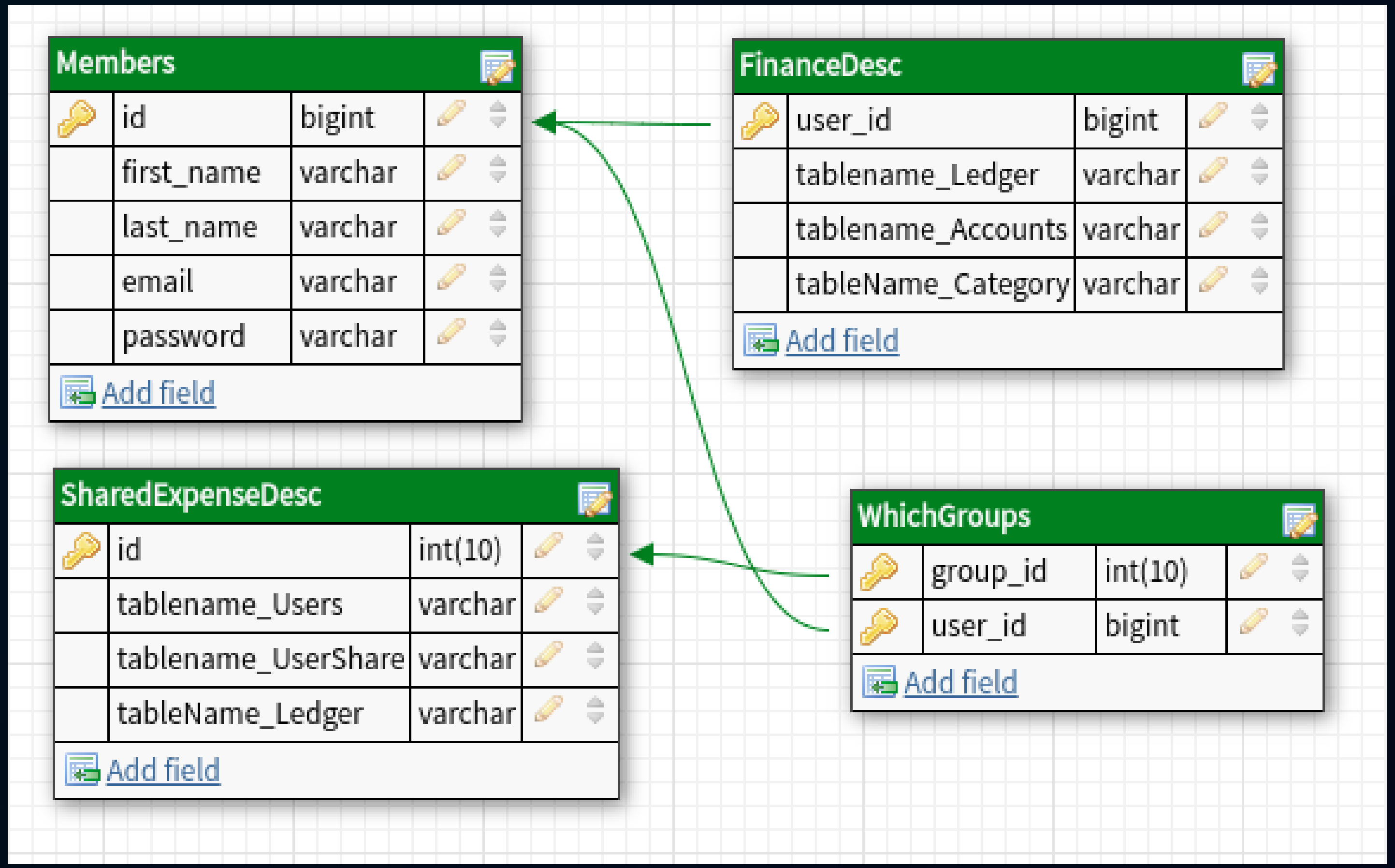
KEY DIFFERENCES BETWEEN SQL AND NOSQL DATABASES

Category	SQL Databases	NoSQL Databases
Schema	Predefined and fixed	Dynamic and flexible
Data Model	Relational (tables)	Key-value, document, column, or graph
Query Language	Structured Query Language (SQL)	Varies by database (NoSQL APIs, JSON)
Scalability	Vertical (add more resources)	Horizontal (add more servers)
Performance	Complex queries	High write/read throughput
Use Cases	Financial systems, ERPs	Real-time analytics, social media, IoT
Databases	Oracle, Microsoft SQL Server	Firebase, Redis, Couchbase

DATABASE ARCHITECTURE FOR THE USE CASE

SQL Database (MySQL):

- Tables for structured storage
- Schema for financial data (transactions, Members details)



Members

🔑	id	bigint
	first_name	varchar
	last_name	varchar
	email	varchar
	password	varchar

FinanceDesc

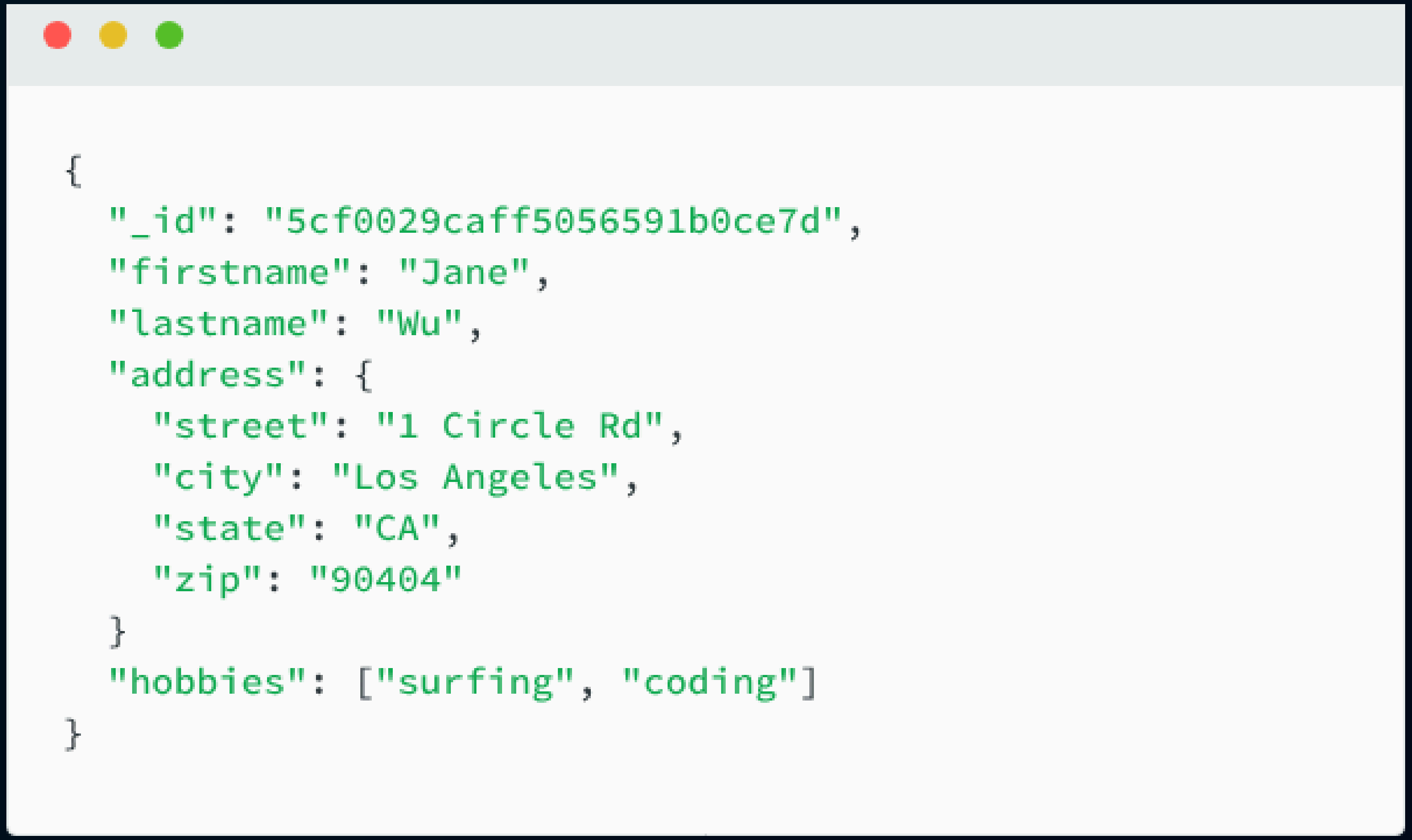
🔑	user_id	bigint
	tablename_Ledger	varchar
	tablename_Accounts	varchar
	tableName_Category	varchar

SharedExpenseDesc

🔑	id	int(10)
	tablename_Users	varchar
	tablename_UserShare	varchar
	tableName_Ledger	varchar

WhichGroups

🔑	group_id	int(10)
🔑	user_id	bigint



```
{
  "_id": "5cf0029caff5056591b0ce7d",
  "firstname": "Jane",
  "lastname": "Wu",
  "address": {
    "street": "1 Circle Rd",
    "city": "Los Angeles",
    "state": "CA",
    "zip": "90404"
  }
  "hobbies": ["surfing", "coding"]
}
```

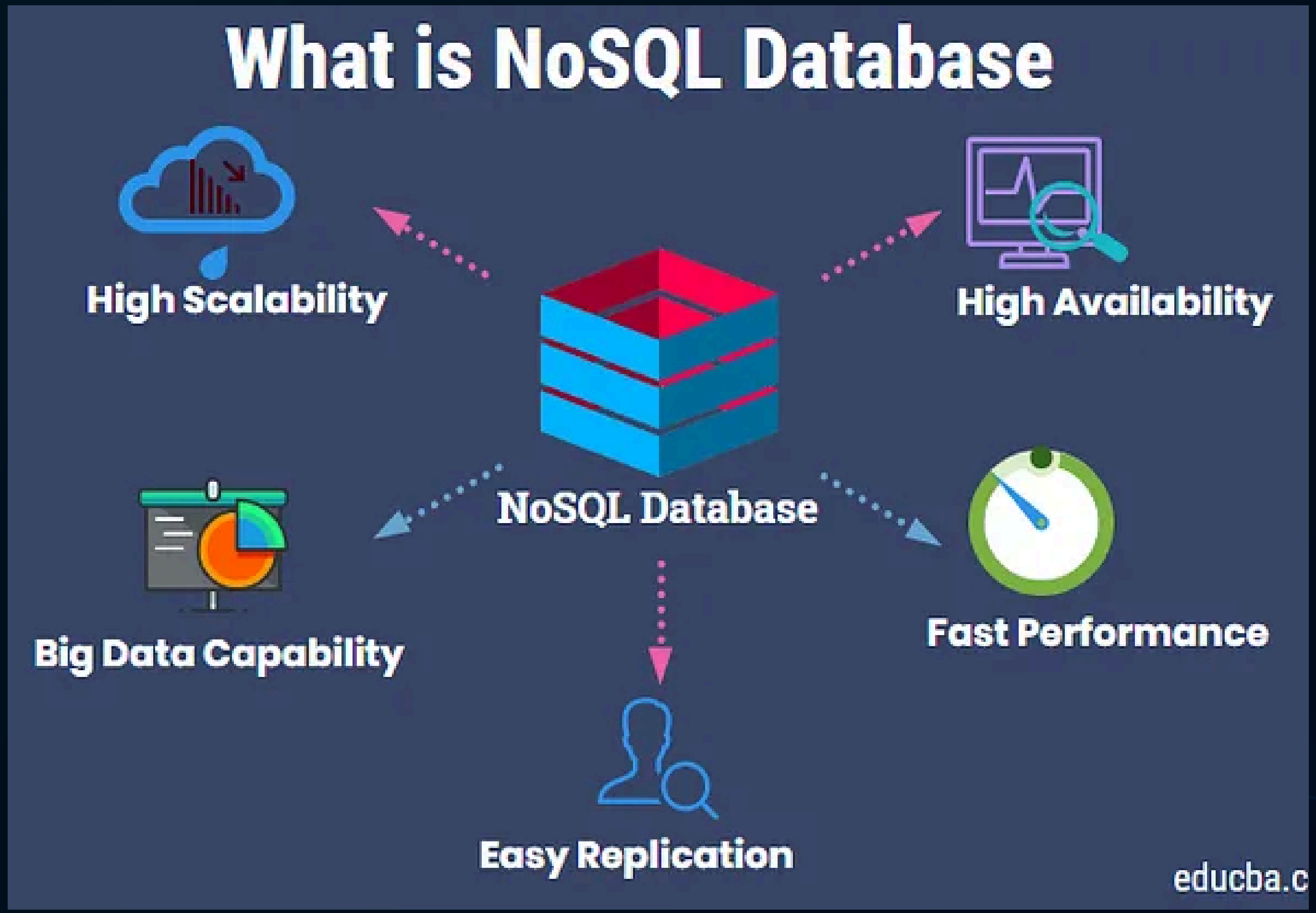
NoSQL Database (MongoDB):

- Document-based storage
- Flexible schema using JSON/BSON formats

WHY NOSQL BECAME POPULAR FOR BIG DATA SYSTEMS

1 . High Scalability:
- Horizontal scaling allows organizations to handle millions of simultaneous users by adding more servers.

2. Big Data Capacity:
- Efficiently stores and processes massive volumes of structured and unstructured data.



3. High Availability:
- Built-in fault tolerance and distributed architecture ensure minimal downtime.

4. Fast Performance:
- Optimized for high-speed data read/write operations, ideal for dynamic applications like IoT or social media platforms.

5. Easy Replication:
- Supports automatic replication for distributed environments, improving data redundancy and disaster recovery.

SQL ADVANTAGES AND DRAWBACKS

SQL (Relational Databases)

Advantages:

- Data Integrity & Consistency: Strong ACID properties ensure reliable transactions.
- Standardized Query Language: Universal use of SQL for querying and management.
- Structured Data: Best for applications requiring complex relationships between data.
- Established Ecosystem: Proven, mature tools with extensive community support.

Drawbacks:

- Limited Scalability: Vertical scaling can become expensive and impractical.
- Rigid Schema: Changes to data structure require downtime or migration.
- Performance Bottlenecks: Slower with large-scale, write-heavy operations.

NOSQL ADVANTAGES AND DRAWBACKS

NoSQL (Non-Relational Databases)

Advantages:

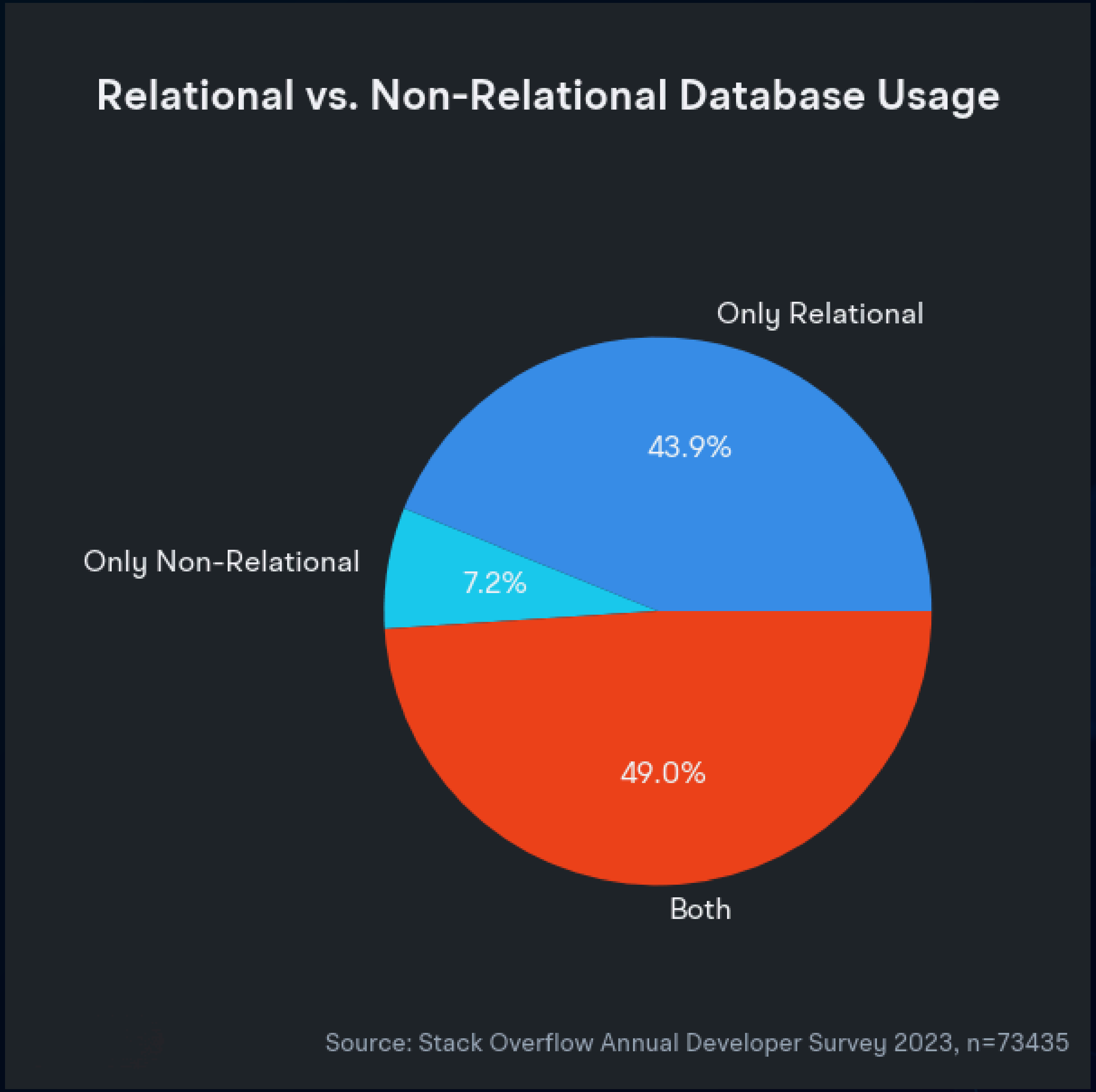
- High Scalability: Handles horizontal scaling seamlessly by adding servers.
- High Availability: Built for fault-tolerant distributed systems.
- Flexible Data Models: Supports evolving business needs without downtime.
- Fast Performance: Optimized for high-speed read/write operations.

Drawbacks:

- Consistency Issues: Eventual consistency may not suit critical systems.
- Learning Curve: Developers need to adapt to various NoSQL data models.
- Limited Standardization: Different databases require unique query languages and tools.

FUTURE TRENDS FOR SQL AND NOSQL DATABASES

1. Convergence of SQL and NoSQL:
   - NewSQL databases offering hybrid features (e.g., CockroachDB, Google Spanner)
2. Cloud-Native Databases:
   - Serverless solutions like AWS DynamoDB gaining traction
3. Machine Learning and AI Integration:
   - Databases optimized for AI-powered insights
4. Graph Databases:
   - Neo4j and Amazon Neptune for relationship-driven applications
5. Data Security Advancements:
   - Stricter compliance requirements driving secure NoSQL solutions



ACID VS. BASE FOR SQL AND NOSQL

- ACID is better for scenarios where strict accuracy and reliability are critical (e.g., banking systems).
- BASE is better suited for distributed systems requiring scalability and performance, where immediate consistency isn't essential (e.g., eCommerce or social media).

Concept	ACID (SQL)	BASE (NoSQL)
Atomicity	Transactions fully complete or fail	Eventual completeness
Consistency	Data always remains valid	Eventual consistency
Isolation	Transactions are isolated	Lesser isolation
Durability	Data changes persist	Changes eventually persist
Main Use Case	Banking, eCommerce	Social media, real-time analytics

REAL-WORLD USE CASE - SOFTWARE COMPANY EXAMPLE

Scenario:

A SaaS (Software as a Service) company provides:

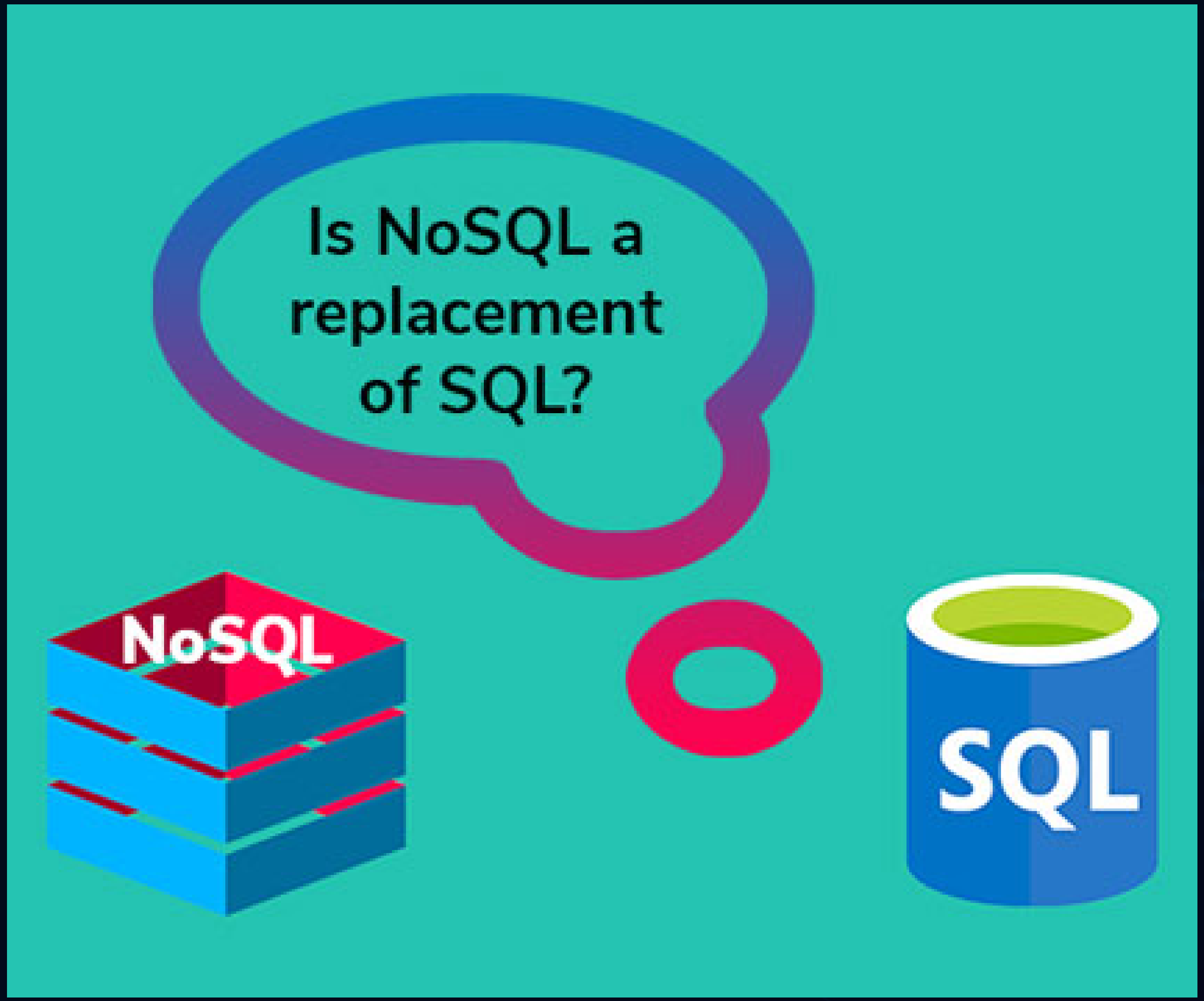
- Financial transaction services for business clients
- Social media insights and real-time analytics for marketing

Storage Solution:

- SQL Database: MySQL for structured financial data
- NoSQL Database: MongoDB for dynamic user interactions

Why This Hybrid Solution?

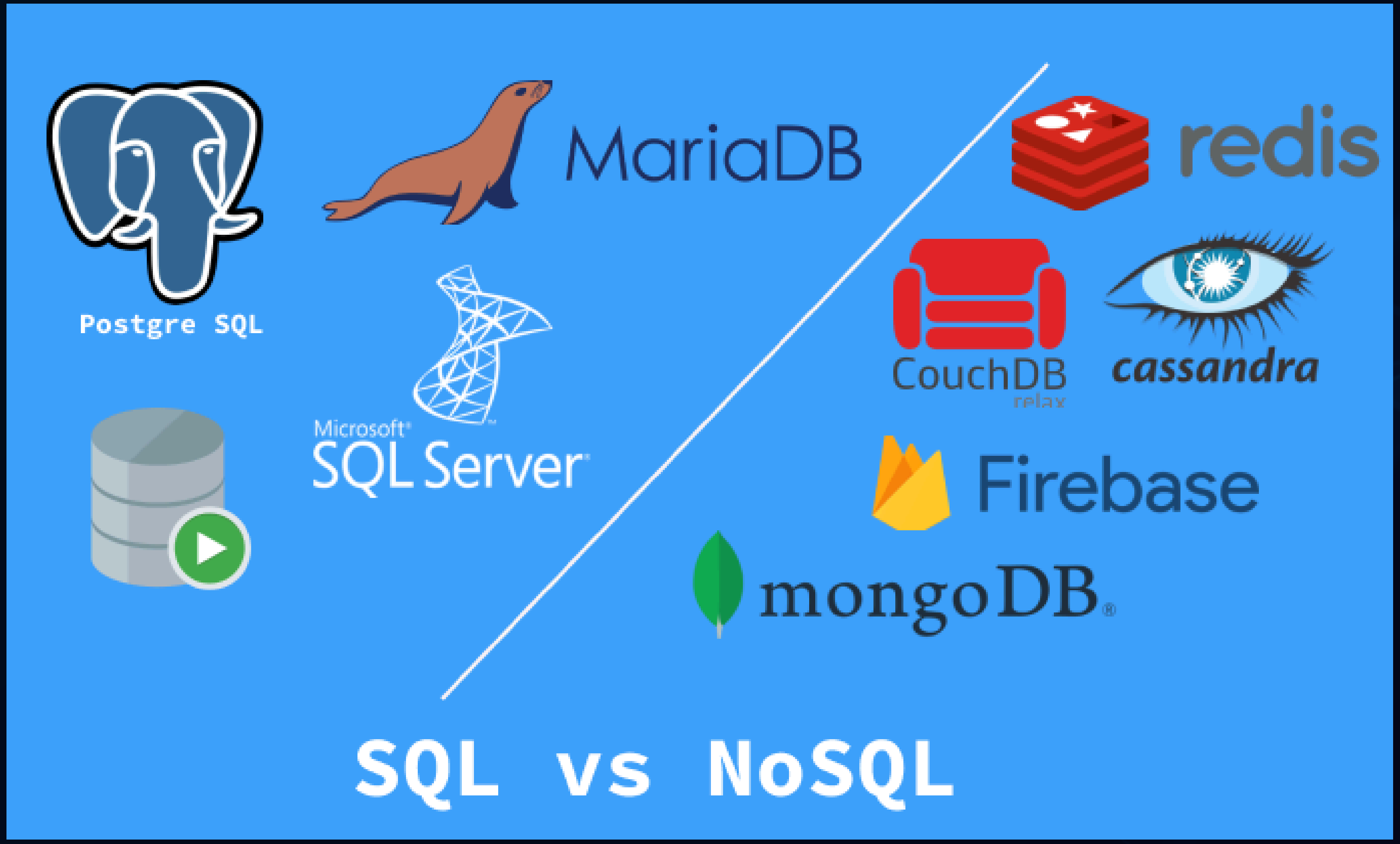
- Consistency for SQL: Necessary for financial records to maintain accuracy
- Scalability for NoSQL: Handling large volumes of real-time analytics data



CONCLUSION AND KEY TAKEAWAYS

Choosing the right database technology is key to building efficient, scalable, and future-ready software systems.

1. Hybrid Approach:
   - Most companies adopt both SQL and NoSQL databases to maximize efficiency.
2. SQL and NoSQL Differences:
   - SQL provides robust, consistent storage for structured data.
   - NoSQL excels in dynamic, scalable systems.
3. Final Thought:
   - A successful database strategy depends on understanding business requirements and selecting the right storage solutions for different components of the application.



THANK YOU