### ****Slide 1: Title Slide****

* **Title**: Introduction to NoSQL
* **Subtitle**: Modern Database Management
* **Presented by**: [Your Name]
* **Date**: [Insert Date]

### ****Slide 2: What is NoSQL?****

* **Definition**: NoSQL databases are non-relational databases optimized for large-scale data storage and real-time applications.
* **Key Features**:
  + Flexible schemas
  + Horizontal scalability
  + High performance for large datasets

### ****Slide 3: Why NoSQL?****

* **Scalability**: Scales out horizontally by adding more servers.
* **Flexibility**: Handles structured, semi-structured, and unstructured data.
* **Performance**: Efficient read/write operations for real-time applications.
* **Use Cases**: Big data, real-time web apps, IoT, social networks.

### ****Slide 4: Types of NoSQL Databases****

1. **Document Stores**: MongoDB (JSON-like documents)
2. **Key-Value Stores**: Redis (simple key-value pairs)
3. **Column-Family Stores**: Cassandra (rows and columns)
4. **Graph Databases**: Neo4j (nodes and relationships)

### ****Slide 5: Example 1 – Document Database (MongoDB)****

json

Copy code

{

"user\_id": "12345",

"name": "John Doe",

"email": "johndoe@example.com",

"order\_history": [

{ "order\_id": "A100", "product": "Laptop", "price": 1200 }

]}

* Stores flexible, JSON-like documents.
* Ideal for e-commerce, user profiles, and content management.

### ****Slide 6: Example 2 – Key-Value Store (Redis)****

* **Key**: "session:12345"
* **Value**:

json

Copy code

{

"user\_id": "12345",

"status": "active",

"login\_time": "2024-09-22T10:00:00Z"}

* Fast lookups, ideal for caching, sessions, and real-time applications.

### ****Slide 7: Example 3 – Graph Database (Neo4j)****

scss

Copy code

(Node) Alice

|

"FRIENDS\_WITH"

|

(Node) Bob

|

"LIKES"

|

(Node) Post123

* Stores relationships between entities.
* Great for social networks, recommendation engines.

### ****Slide 8: CAP Theorem****

* **Consistency**: All nodes have the same data at the same time.
* **Availability**: Every request receives a response.
* **Partition Tolerance**: Database continues working despite network issues.
* **NoSQL Trade-Off**: Sacrifices consistency for availability and partition tolerance in distributed systems.

### ****Slide 9: When to Use NoSQL****

* **Scalability Needs**: Handling massive data with horizontal scaling.
* **Flexibility**: For evolving or complex data structures.
* **Real-Time Applications**: Needing fast, large-scale operations (e.g., social media, IoT).

### ****Slide 10: Summary****

* **NoSQL**: Provides flexibility, scalability, and high performance.
* **Types**: Document, key-value, column-family, graph databases.
* **Use Cases**: Big data, real-time web apps, social networks.