NoSQL Databases

What Is a NoSQL Database?

- Non-relational database – does not use SQL to structure and query data.
- Designed for:
  - Flexible schemas
  - High performance and scalability

Types of NoSQL Databases

- Document (e.g. MongoDB)
- Key-Value (e.g. Redis)
- Column-family (e.g. Cassandra)
- Graph (e.g. Neo4j)

Spring Data NoSQL Overview

- Spring Data provides support for many NoSQL stores:
  - Spring Data MongoDB
  - Spring Data Redis
  - Spring Data Cassandra
  - Spring Data Neo4j
- Offers the same repository abstraction as Spring Data JPA.
- Goal: Consistent data access programming model across all data stores.

Why Use NoSQL in Spring Applications?

- Schema flexibility – good for unstructured or evolving data.
- Horizontal scalability – better for big data and high-throughput apps.
- Easy integration – via Spring Data modules.

Spring Data JPA vs. Spring Data MongoDB

Feature	Spring Data JPA	Spring Data MongoDB
Data Model	Relational (SQL)	Document (JSON-like)
Schema	Fixed (DDL)	Flexible
Query Language	JPQL, SQL	MongoDB query syntax
Annotations	@Entity, @Id, @Column	@Document, @Id, etc.

Code Comparison – Entity vs. Document

- JPA:

```
@Entity
public class Book {
    @Id
    private Long id;
    private String title;
}
```

- MongoDB:

```
@Document
public class Book {
    @Id
    private String id;
    private String title;
}
```

Code Comparison – Repository Interface

- JPA Repository:

```
public interface BookRepository extends JpaRepository<Book, Long> {
    List<Book> findByTitle(String title);
}
```

- MongoDB Repository:

```
public interface BookRepository extends MongoRepository<Book, String> {
    List<Book> findByTitle(String title);
}
```

Summary – One Model, Many Backends

- Spring Data simplifies working with both SQL and NoSQL.
- Repository interfaces and query methods are nearly identical.
- Learn once, apply many times – useful skill across database types.
- Choose JPA or NoSQL depending on project needs (structure, scale, schema).