**1. Explain the need and benefit of ORM**

ORM (Object-Relational Mapping):

* ORM maps Java objects to relational database tables.
* Developers can interact with databases using Java classes and objects instead of SQL queries directly.

Benefits:

* Abstracts database system.
* Simplifies CRUD operations.
* Automatic table mapping.
* Transaction management handled.
* Reduces boilerplate SQL code.

**2. Demonstrate the need and benefit of Spring Data JPA**

Why Spring Data JPA?

* Built on top of JPA and Hibernate.
* Minimizes DAO boilerplate.
* Auto-generates queries based on method names.
* Easy integration with Spring Boot.
* Works with both in-memory (H2) and external databases (MySQL, PostgreSQL).

Key Features:

* Repository Interfaces (e.g., JpaRepository)
* Custom query methods (e.g., findByNameContaining)
* Transactional support

**3. Explain core objects of Hibernate Framework**

|  |  |
| --- | --- |
| **HIBERNATE OBJECT** | **PURPOSE** |
| Session Factory | Factory for session objects(singleton, heavy-weight) |
| Session | Interface between app and DB |
| Transaction Factory | Handles transaction strategy |
| Transaction | Represents a DB transaction |
| Connection Provider | Abstracts DB connection pool setup |

**4. Explain ORM with Hibernate XML and Annotation Config**

XML-Based Configuration:

* Define .cfg.xml and .hbm.xml files.
* Manual mapping between classes and tables.

Annotation-Based Configuration:

* Use annotations like @Entity, @Table, @Id, @Column.

CODE:

@Entity

@Table(name = "country")

public class Country {

@Id

@Column(name = "co\_code")

private String code;

@Column(name = "co\_name")

private String name;

}

**5. Difference Between JPA, Hibernate, Spring Data JPA**

| **Feature** | **JPA** | **Hibernate** | **Spring Data JPA** |
| --- | --- | --- | --- |
| What it is? | Specification (JSR 338) | ORM Framework | JPA abstraction layer |
| Provides Implementation | No | Yes | Yes (built on Hibernate) |
| Boilerplate Code | Moderate | More than JPA | Very minimal |
| Configuration | Manual or minimal | Manual | Auto (via Spring Boot) |

**6. Demonstrate DML using Spring Data JPA**

Common Methods from JpaRepository:

| Method | Description |
| --- | --- |
| findById(String id) | Retrieve by primary key |
| save(entity) | Insert or update |
| deleteById(String id) | Delete by primary key |
| findByName(String name) | Custom method via Spring query |

Example:

// Repository

public interface CountryRepository extends JpaRepository<Country, String> {

List<Country> findByNameContaining(String partialName);

}

java

CopyEdit

// Service Method

@Transactional

public void updateCountry(String code, String newName) {

Country country = countryRepository.findById(code).orElseThrow();

country.setName(newName);

countryRepository.save(country);

}