Hibernate & Spring Data JPA

Beginner to Guru

Introduction to Spring Data JPA

What is Spring Data JPA?

- Spring Data JPA - Is the JPA based version of data repository access of the Spring Data family of projects
  - "Spring Data" has many implementations for various data stores including:
    - MongoDB, Redis, Cassandra, GemFile, Couchbase, Neo4J
- Spring Data JPA focuses on supporting the JPA API standard
  - Typically used for accessing SQL based Relational Databases
    - "Typically" because some NoSQL Data stores offer JPA support

What is JPA?

- Spring Data JPA - Is an abstraction layer built on top of JPA
- JPA - Java Persistence API
- JPA is a common API for working with data Relational Databases
    - Typically SQL Databases, but has been used with some NoSQL datastores
- Hibernate is an implementation of JPA
    - JPA is just the interface for the API; JPA is not an implementation
- Other implementations include EclipseLink, Apache OpenJPA, and TopLink

What is Hibernate?

- Hibernate - Is a Object Relational Mapping Tool which also implements the JPA API specification
- ORM - Object Relational Mapping
  - Java is a strongly typed Object Oriented Programming Language
  - SQL Databases persist data using a Relational data model
  - ORMs map data between the Relational and Object Oriented paradigms
- Leakage - term for one paradigm 'leaking' into the other model
  - The ID value is considered 'leakage'. The ID is the Primary Key for the relational model, and needs to be carried over to the Object Model

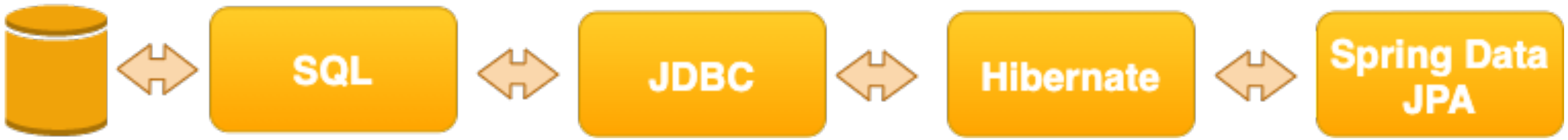
What is SQL?

- Hibernate performs database operations using SQL
- SQL - Structured Query Language, AKA "sequel"
- Originally developed in the early 1970s by IBM Engineers
- Widely adopted by all Relational Databases and even some NoSQL databases
- ANSI SQL - is the standard for the SQL Language
    - SQL Databases will support ANSI SQL, and their own flavor of SQL
- Hibernate will favor using ANSI SQL for portability

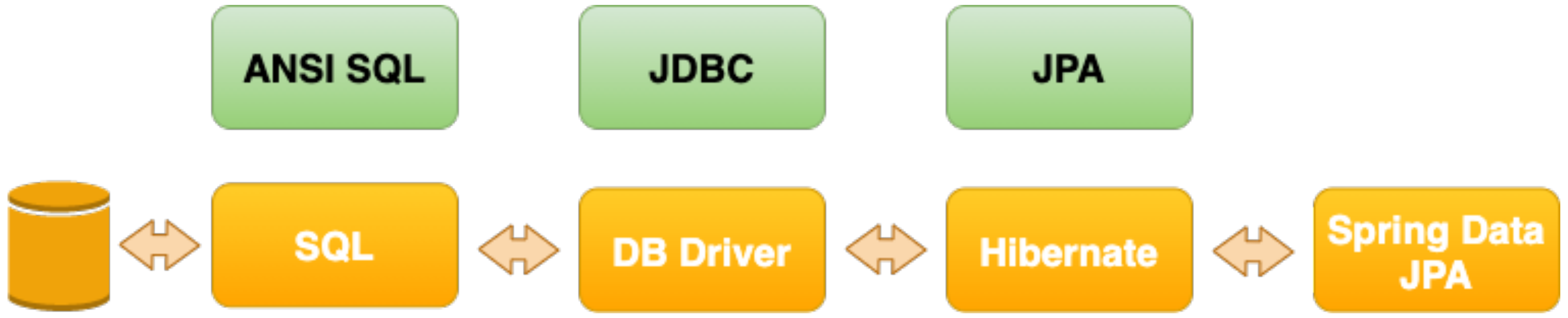
What is JDBC?

- JDBC - Java DataBase Connectivity
- JDBC is the Java API for connecting to databases
  - Again, just the API - NOT the implementation
- The JDBC Implementation is typically referred to as a JDBC Driver
- Each Database will have it’s own JDBC Driver implementation
- The JDBC Driver handles the low level communications with the database
- Each Driver will implement the JDBC API, and have platform specific extensions

Putting It All Together



Putting It All Together



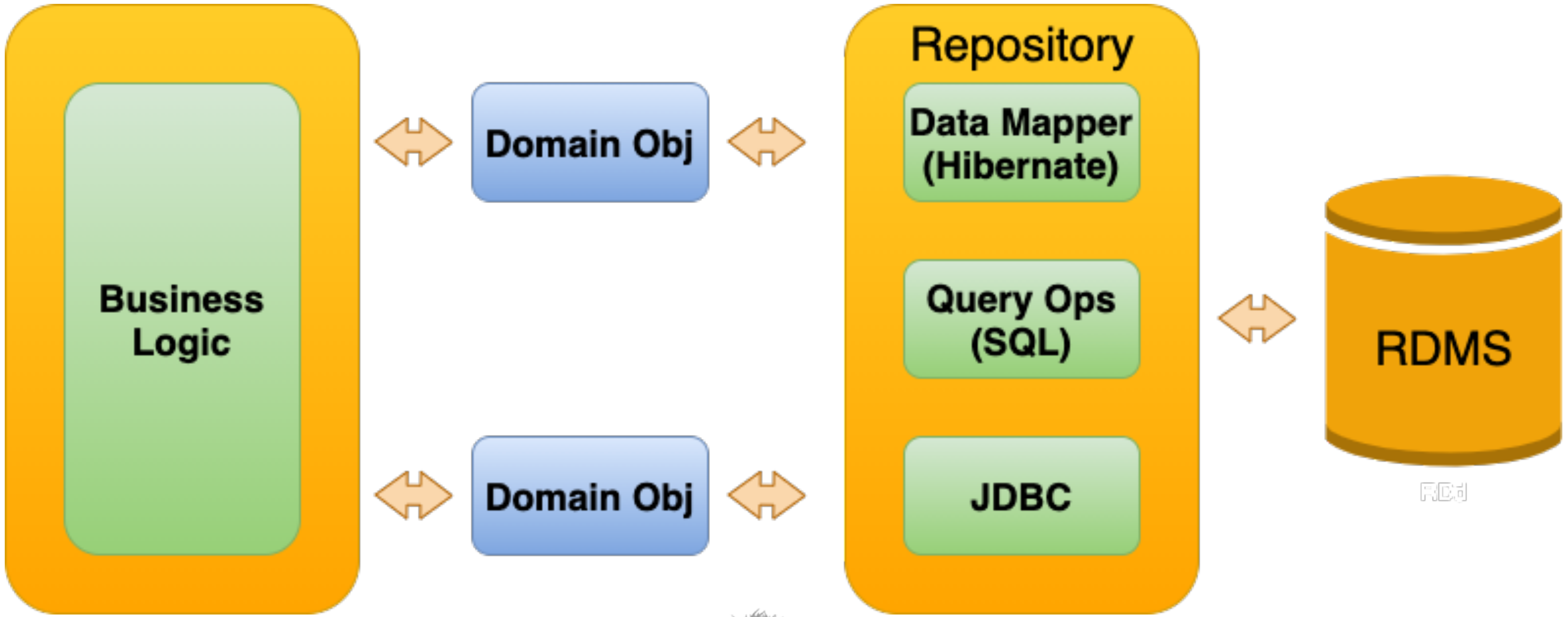
Spring Data JPA - The Abstraction

- Spring Data JPA - Is an Abstraction Layer
- Simplifies your programming experience
- You are not worried about low level things such as database connections and transactions
- You can access and control these when needed
- Spring Data JPA uses the JPA API layer
    - You can still access implementation specific features when needed

The Repository Pattern

- Spring Data JPA - Implements the Repository Pattern
- The Repository Pattern was introduced in the book Domain Driven Design in 2004
  - Excellent book btw
- The Repository is an abstraction of data and persistence operations
- You wish to keep your domain objects “Persistence Ignorant”
- Provide an Interface for CRUD Operations, implementation will handle all persistence operations
- In Spring Data JPA, you provide the interface, Spring Data JPA provides the implementation

The Repository Pattern



When to Use JPA?

- Spring Data JPA - is very good for single object CRUD operations
- When you have multiple operations against a small set of objects
  - Like a checkout operation in a web store application
  - Hibernate will cache and batch DB operations for efficiency
- When you have control of the database schema

When NOT to Use JPA?

- Spring Data JPA - is not very good for batch operations
- There is a cost to fetching a single record from the database and mapping it to a Java object
  - Fine for simple operations, costly for 10's of thousands of operations
- Relational Databases are very efficient at what they do
- SQL is a very powerful language
- If you are performing batch operations on 10's of thousands of records, you should consider using SQL/JDBC

Summary

- Spring Data JPA - is an abstraction which implements the repository pattern
- The Abstraction hides the complexity of SQL, JDBC, JPA (Hibernate) from us
- This allows us to focus more of our time on business logic
- While the complexity is hidden, it is still important to understand what the abstraction is doing
- Most difficulties in learning Spring Data JPA are rooted in not understanding the underlying technologies
- Coming up, we will look at Spring Data JPA and then focus on the 'magic' being abstracted

SPRING FRAMEWORK

GURU