1. What does JPA stand for? Why is it useful? What problems does it solve?

JPA stands for Java Persistence API, it is a specification that provides a set of Java interfaces and classes for Object-relational mapping in Java applications. It's specifically useful because it simplifies the development of Java applications that interact with relational databases, offering an "abstraction layer" over the underlying database management system. Basically, allowing work and transferability in their code without having to write SQL queries. Some of the main problems it solves is obviously the Object-Relational Mapping as mentioned earlier but also, Database Independence, CRUD Operations, Querying(JPQL), Caching, and also Transaction Management.

2. Compare JDBC & JPA. Compare and contrast the two approaches.

JDBC or Java Database Connectivity is a lower level API, it provides direct access to the database but it does require developers to write to in SQL queries and handle database-specific details such as connection management, statement execution, and result set processing. JDBC also requires manual mapping for Java objects that come from the database. As opposed to JPA or Java Persistence API which provides a higher level of abstraction, which maps objects to database tables and removes the need to write SQL queries manually. However, the one benefit of this is, JDBC does offer finer control over database operations, allowing much higher levels of optimizations and transactions. JPA on the other hand, also offers database independence which allows database-independent code that can work with different database systems and by changing the JPA configuration, it can easily switch between databases.