**1. ORM, JPA, Hibernate và tác dụng:**

* **ORM (Object-Relational Mapping)** là kỹ thuật ánh xạ các đối tượng trong ngôn ngữ lập trình với các bảng trong cơ sở dữ liệu quan hệ. ORM giúp đơn giản hóa việc thao tác với cơ sở dữ liệu thông qua các đối tượng thay vì viết các câu lệnh SQL trực tiếp.
* **JPA (Java Persistence API)** là một API trong Java giúp làm việc với cơ sở dữ liệu theo kiểu ORM, định nghĩa các chuẩn cho các hoạt động lưu trữ dữ liệu. JPA không thực thi trực tiếp mà cần một framework như Hibernate để thực hiện.
* **Hibernate** là một framework ORM phổ biến trong Java, thực thi JPA và cung cấp các tính năng bổ sung như caching, lazy loading, v.v.

**2. Các thành phần của Hibernate:**

* **Session**: Là giao diện chính để tương tác với Hibernate. Được dùng để tạo, đọc và lưu trữ các đối tượng.
* **SessionFactory**: Là đối tượng để tạo ra session. Nó được tạo một lần khi ứng dụng khởi động và tái sử dụng cho nhiều phiên làm việc.
* **Transaction**: Được sử dụng để quản lý các giao dịch trong Hibernate.
* **Query**: Là đối tượng dùng để thực hiện các truy vấn (HQL hoặc Criteria).
* **Configuration**: Được sử dụng để cấu hình Hibernate (đọc các file cấu hình, kết nối cơ sở dữ liệu, v.v.).

**3. Cách cấu hình Hibernate:**

* **Cấu hình qua hibernate.cfg.xml**:
  + Tạo file hibernate.cfg.xml chứa các thông tin cấu hình như JDBC URL, username, password, các thuộc tính của Hibernate.
* **Cấu hình qua hibernate.properties**:
  + Tạo file hibernate.properties chứa các thông số cấu hình tương tự.
* **Cấu hình qua Java-based configuration**:
  + Sử dụng Annotation để cấu hình Hibernate.

**4. Các cách tạo constraint trong Hibernate:**

* **@NotNull, @Size, @Min, @Max**: Dùng để áp dụng các ràng buộc trên các thuộc tính.
* **@UniqueConstraint**: Dùng trong annotation @Table để định nghĩa ràng buộc duy nhất trên một hoặc nhiều cột.

**5. Cách sử dụng annotation:**

* **@Entity**: Đánh dấu lớp Java là một Entity (bảng trong cơ sở dữ liệu).
* **@Table**: Xác định tên bảng trong cơ sở dữ liệu cho entity.
* **@Id**: Đánh dấu thuộc tính là khóa chính.
* **@Column**: Dùng để ánh xạ một thuộc tính với cột trong cơ sở dữ liệu.

**6. Quan hệ @OneToOne, @OneToMany, @ManyToOne, @ManyToMany:**

* **@OneToOne**: Một thực thể này chỉ liên kết với một thực thể khác.
* **@OneToMany**: Một thực thể liên kết với nhiều thực thể khác.
* **@ManyToOne**: Nhiều thực thể liên kết với một thực thể khác.
* **@ManyToMany**: Nhiều thực thể liên kết với nhiều thực thể khác.

**7. Quan hệ bi-directional giữa các entity:**

* Để tạo quan hệ bi-directional, bạn cần dùng annotation ngược lại trên cả hai entity. Ví dụ, một @ManyToOne trong entity A và một @OneToMany trong entity B.

**8. Xử lý N+1 Select trong Hibernate:**

* **N+1 Select Problem** xảy ra khi bạn truy vấn một danh sách các thực thể, và mỗi thực thể này lại truy vấn thêm các thực thể liên quan trong cơ sở dữ liệu.
* Sử dụng **FetchType.LAZY** hoặc **@Fetch(FetchMode.JOIN)** để tối ưu hóa vấn đề này.

**9. @Embeddable và @Embedded:**

* **@Embeddable**: Đánh dấu một lớp có thể được nhúng vào một entity khác.
* **@Embedded**: Dùng để nhúng một đối tượng @Embeddable vào một entity.

**10. Cách sử dụng @ElementCollection:**

* Dùng **@ElementCollection** để lưu trữ các collection cơ bản (như List, Set) của một entity, không cần tạo một bảng riêng.

**11. Hibernate hỗ trợ các kiểu kế thừa:**

* **Table Per Class**: Mỗi lớp con có một bảng riêng.
* **Table Per Hierarchy**: Sử dụng một bảng cho toàn bộ lớp cha và các lớp con, có cột phân biệt loại đối tượng.
* **Table Per Concrete Class**: Mỗi lớp con có bảng riêng nhưng không có kế thừa bảng từ lớp cha.

**12. Persistence Context và Hibernate Caching:**

* **Persistence Context**: Là một vùng nhớ tạm thời của Hibernate dùng để quản lý trạng thái của các thực thể trong một phiên làm việc.
* **Hibernate Caching**: Là cơ chế lưu trữ tạm thời kết quả truy vấn để giảm thiểu các truy vấn lặp lại.

**13. Cấu hình Second Level Cache:**

* **Second Level Cache** lưu trữ kết quả của các đối tượng đã được truy vấn, làm giảm số lần truy vấn cơ sở dữ liệu.
* Cấu hình qua ehcache hoặc infinispan.

**14. Sự khác nhau giữa Session.save() và Session.persist():**

* **Session.save()**: Lưu một đối tượng và trả về ID.
* **Session.persist()**: Lưu một đối tượng nhưng không trả về ID (ID sẽ được tạo sau khi commit transaction).

**15. Lifecycle của một entity:**

* **Transient**: Entity chưa được liên kết với session.
* **Persistent**: Entity đã được liên kết với session và đang được theo dõi.
* **Detached**: Entity đã không còn liên kết với session nhưng vẫn tồn tại.
* **Removed**: Entity đã được đánh dấu là sẽ bị xóa.

**16. Hibernate Query Language (HQL):**

* HQL là một ngôn ngữ truy vấn hướng đối tượng, tương tự như SQL nhưng thao tác với các entity thay vì các bảng trong cơ sở dữ liệu.

**17. Sự khác nhau giữa HQL và SQL:**

* **HQL** làm việc với các entity, còn **SQL** làm việc với các bảng trong cơ sở dữ liệu.
* **HQL** hỗ trợ truy vấn các đối tượng Java, trong khi **SQL** chỉ có thể truy vấn dữ liệu cơ sở dữ liệu.

**18. Cách sử dụng Criteria API:**

* **Criteria API** là cách tiếp cận loại bỏ phụ thuộc vào câu lệnh HQL, giúp tạo truy vấn động và dễ dàng bảo trì.

**19. Lazy Loading và Eager Loading:**

* **Lazy Loading**: Dữ liệu được tải khi cần thiết.
* **Eager Loading**: Dữ liệu được tải ngay khi truy vấn.
* **LazyInitializationException** xảy ra khi truy vấn được thực thi ngoài phạm vi session.

**20. Cách quản lý giao dịch trong Hibernate:**

* Dùng Session.beginTransaction() để bắt đầu giao dịch và transaction.commit() để kết thúc giao dịch.

**21. Tối ưu hóa hiệu suất trong Hibernate:**

* Sử dụng **batch processing**, **caching** và **các truy vấn tối ưu** để cải thiện hiệu suất.

**22. Batch Processing trong Hibernate:**

* Hibernate hỗ trợ batch processing để giảm số lần truy vấn cơ sở dữ liệu, giúp cải thiện hiệu suất.

**23. Các cách convert data:**

* **Object[]**, **Tuple**, **DTO**, và **Entity** có thể được sử dụng để chuyển đổi dữ liệu giữa các loại đối tượng khác nhau khi thực thi query.

**24. Cách phân trang trong Hibernate:**

* Sử dụng setFirstResult() và setMaxResults() để phân trang các kết quả truy vấn