Control Teórico Ampliado: Persistencia, JPA y EJB — Respuestas

Conceptos Clave

- **Entidades:** Son objetos Java (POJO) que representan las tablas de la base de datos; definen únicamente la estructura y el mapeo de los datos.
- **EJBs:** Los Enterprise JavaBeans son componentes del servidor en Java EE, que encapsulan la lógica de negocio y gestionan transacciones, seguridad y concurrencia. Puede ser Stateless, Stateful o Singleton.
    - **Stateless:** Los Facades son ejemplo de EJB Stateless, ya que las operaciones CRUD sin individuales y no necesitan compartir un estado.
    - **Stateful:** Un carrito de compras, por ejemplo, necesita contener la sesion del usuario, manteniendo en memoria los productos que quiere comprar, por lo que debe persistir en memoria entre llamadas del mismo cliente.
    - **Singleton:** Es un bean que se instancia una sola vez, vive durante todo el ciclo de vida de la aplicación y es compartido por todos los clientes.
- **Facades:** Son EJBs stateless que contienen la lógica de negocio, como las operaciones CRUD. Utilizan JPA para lograr la persistencia y realizar consultas a la base de datos. El patrón

Session Facade ayuda a reducir el acoplamiento entre el cliente y los objetos de negocio.

- **JPA:** Es una especificación para integrar el modelo relacional con el modelo orientado a objetos. Permite gestionar la persistencia y las transacciones con la base de datos por medio del objeto `EntityManager` (`em`). Este `EntityManager` debe ser injectado en los facades por medio de `@PersistenceContext` .

La conexion con la bbdd se logra de la siguiente manera:

- El servidor Glassfish tiene un Pool de conexion para conectarse a la BBDD. A su vez configura un JDBC para exponer la conexion hacia la aplicacion.
- La aplicacion configura un `Data Source` que referencia el JDBC del servidor Glassfish y una Unidad de Persistencia que referencia el Data Source. Es decir, el `Data Source` es el componente que permite establecer la conexión a la BBDD en la aplicación Java EE.

JPQL (Java Persistence Query Language) es el lenguaje de consultas orientado a objetos utilizado en JPA para realizar consultas sobre las entidades. En lugar de usar SQL, se usan consultas basadas en las clases y atributos de las entidades. Por ejemplo , para obtener todos los alumnos mayores de 18 años, se podría usar la siguiente consulta JPQL:

```
String jpql = "SELECT a FROM Alumno a WHERE a.edadd > 18";
// El error de edadd no se sabe hasta la ejecución.
List<Alumno> resultados = em.createQuery(jpql, Alumno.clas
```

API Criteria: Es una API para construir las consultas de forma programática, utilizando POJOs, y permite detectar errores en tiempo de compilación, ya que se basa en las clases y atributos de las entidades. Por ejemplo, para obtener todos los alumnos mayores de 18 años, se podría usar la siguiente consulta con la API Criteria:

```
CriteriaBuilder cb = em.getCriteriaBuilder();
CriteriaQuery<Alumno> cq = cb.createQuery(Alumno.class);
Root<Alumno> alumno = cq.from(Alumno.class);
cq.select(alumno).where(cb.gt(alumno.get("edadd"), 18));
// El IDE es capaz de advertir que no existe edadd
List<Alumno> resultados = em.createQuery(cq).getResultList
```

Examen Teórico: Backend Java, Persistencia y EJB (30 Preguntas)
Sección I: Opción Múltiple
Instrucción: Seleccione la opción que mejor complete o responda la pregunta (10 preguntas).

1. Propósito principal de JPA
   ¿Cuál es la función central de JPA (Java Persistence API) dentro de una aplicación Java EE?
   A. Servir como la interfaz de usuario para el cliente web.

B. Integrar el modelo relacional (BD) y el modelo orientado a objetos (Entidades) ✓.

C. Administrar la seguridad y las transacciones de manera exclusiva.

D. Proveer un servicio de mensajería asíncrona (MDB).

2. Componente de Servidor para Lógica de Negocio
   ¿Qué componente del servidor (clase Java) está diseñado para encapsular la lógica del negocio y crear un modelo simple para desarrollar aplicaciones distribuidas, transaccionales y seguras?
   A. La clase HttpServlet (ej. Manejador)
   B. El EntityManager
   C. La persistence.xml
   D. Enterprise JavaBeans (EJB) ✓.
3. Característica de Session Bean Stateless
   Los Session Beans del tipo Stateless (sin estado) son apropiados para operaciones que se resuelven en una sola llamada al método. ¿Qué sucede con el estado específico del cliente después de que el método termina su procesamiento?
   A. Se serializa para su uso futuro.
   B. Se mantiene como "estado de conversación" por el contenedor EJB.
   C. No se conserva el estado específico del cliente ✓.
   D. Se pasa a otro bean para su gestión.
4. Patrón Session Facade y Acoplamiento
   Uno de los problemas de aplicaciones de varios niveles que el patrón Session Facade (Fachada de Sesión) intenta resolver es:

A. El uso excesivo de JPQL.

B. El acoplamiento (dependencia) entre los clientes y los objetos de negocio ✓.

C. La incapacidad de utilizar anotaciones JPA.

D. La lentitud de la conexión JDBC.

5. Estrategia de Mapeo de Herencia por Defecto en JPA

   En JPA, si no se especifica explícitamente una estrategia de mapeo de herencia, la predeterminada es:

   A. InheritanceType.JOINED (Joined subClases).

   B. InheritanceType.TABLE_PER_CLASS (Una tabla por clase concreta).

   C. Una simple tabla por jerarquía, aplanando todos los atributos en una sola tabla ✓.

   D. No se permite la herencia sin una anotación @Inheritance.

6. Mecanismo de Conectividad en Java EE

   En la arquitectura Java EE, ¿qué componente permite a las aplicaciones obtener una conexión a la base de datos desde el pool de conexiones, buscando en el directorio de nombres (JNDI)?

   A. El EntityManager.

   B. El JDBC driver.

   C. Un data source ✓.

   D. El persistence.xml.

7. Rol de la Criteria API en JPA

   JPA provee una API Criteria cuyo propósito es:

   A. Reemplazar completamente el uso de SQL.

   B. Permitir la construcción de consultas (queries) de manera programática basadas en JPQL ✓.

C. Únicamente definir transacciones.

D. Mapear las relaciones Muchos-a-Muchos.

8. Anotación para Inyectar Fachadas de Sesión

   Dentro de una clase controladora o de servicio (Manejador o similar), ¿qué anotación se utiliza para inyectar una instancia de un Session Bean (como FacultadFacade) gestionado por el contenedor EJB?

   A. @PersistenceContext

   B. @Override

   C. @EJB ✓.

   D. @Entity

9. Persistencia de Entidades

   Las Entidades en el contexto de JPA representan:

   A. Únicamente la capa de servicio (Session Facade).

   B. Objetos que viven en memoria y que persisten (almacenan su estado) en una base de datos ✓.

   C. Objetos que solo existen durante el ciclo de vida de un método.

   D. La implementación de JDBC.

10. Tipo de Session Bean para Recursos Compartidos

    ¿Qué tipo de Session Bean se instancia una sola vez por aplicación, existe durante todo el ciclo de vida de la aplicación y está diseñado para ser accedido concurrentemente y compartido por múltiples clientes?

    A. Stateful

    B. Stateless

    C. Singleton ✓.

    D. Message-driven

Sección II: Verdadero / Falso

Instrucción: Indique si la afirmación es Verdadera (V) o Falsa (F) y justifique brevemente si es Falsa.

11. El Lenguaje de Consulta de JPA (JPQL) se puede considerar como la versión de SQL orientada a objetos.
    Respuesta: Verdadero (V) ✓.
12. En un Session Bean Stateful (con estado), el "estado de conversación" (valores de sus variables de instancia) se mantiene a través de varias llamadas a métodos de un cliente específico.
    Respuesta: Verdadero (V) ✓.
13. El mapeo relacional-objetos en JPA debe definirse obligatoriamente utilizando Anotaciones de Java, sin la posibilidad de usar Descriptores XML.
    Respuesta: Falso (F) ✓. Justificación: El mapeo relacional-objetos puede definirse usando Anotaciones o Descriptores XML.
14. El Contenedor EJB es el entorno de ejecución que ofrece servicios esenciales para los componentes, como transacciones, seguridad, pool de recursos y soporte de concurrencia.
    Respuesta: Verdadero (V) ✓.
15. El proceso de Ingeniería Directa (Forward Engineering) se utiliza en JPA para generar automáticamente las clases de Entidad (Alumno.java, Materia.java) a partir de las tablas de la base de datos.
    Respuesta: Falso (F) ✓. Justificación: La Ingeniería Directa se

aplica para generar el esquema físico SQL de la base de datos a partir del modelo, no las clases de Entidad Java.

16. La anotación @PersistenceContext se utiliza en un Session Bean (Fachada) para inyectar un EntityManager gestionado por el contenedor, basándose en la unidad de persistencia definida en persistence.xml. Respuesta: Verdadero (V) ✓.
17. Las tecnologías de persistencia (JPA) y componentes de negocio (EJB) trabajan de manera integrada, conservando las pautas propuestas por el patrón de diseño Model-View-Controller (MVC).
    Respuesta: Verdadero (V) ✓.
18. Para mapear correctamente una relación Muchos-a-Muchos entre entidades (Docente y Materia) en el backend y la base de datos, se requiere la creación de una tercera entidad/tabla (pivot) que contenga las claves primarias de ambas.
    Respuesta: Verdadero (V) ✓.
19. JDBC es la API de Java que se utiliza para definir los EJB, mientras que JPA es la encargada de la comunicación directa con SQL.
    Respuesta: Falso (F) ✓. Justificación: JDBC es la biblioteca de integración requerida para comunicar SQL y Java, mientras que JPA es la especificación que integra el modelo relacional y orientado a objetos (ORM).
20. El patrón Session Facade abstrae las interacciones de los objetos de negocio, proporcionando una capa de servicio que expone solo la funcionalidad CRUD requerida y evita el mal uso por parte de los clientes. Respuesta: Verdadero (V) ✓.

Sección III: A Completar o Responder

Instrucción: Responda la pregunta o complete el espacio en blanco (10 preguntas).

21. Mecanismo de Mapeo
    El mecanismo que mapea objetos de Java (en memoria) a datos almacenados en una Base de Datos relacional es conocido como ORM (Object-Relational Mapping) ✓.
22. Operaciones Fundamentales de Entidades
    Las clases de Entidad deben ser dotadas de funcionalidad para realizar las operaciones básicas de manipulación de datos, conocidas por el acrónimo CRUD. ¿Qué significa este acrónimo? Respuesta: Crear, Recuperar/Leer, Actualizar y Eliminar ✓.
23. Archivo de Configuración de Persistencia
    El archivo, generado durante la creación de la Unidad de Persistencia, que contiene la información necesaria para conectarse a la BD (como el data source) y el proveedor JPA (ej. EclipseLink), se denomina: Respuesta: persistence.xml ✓.
24. Propósito de JPQL
    El lenguaje de consulta JPQL es utilizado para recuperar datos mediante un lenguaje de consulta orientado a objetos ✓.
25. Problema de Rendimiento de Session Facade
    Además de reducir el acoplamiento, el patrón Session Facade ayuda a mitigar un problema de rendimiento de red causado por: Respuesta: Demasiadas llamadas a métodos entre el cliente y el servidor ✓.

26. Inyección de Objetos JPA
    Para que un Session Bean (como un Facade) pueda interactuar con las entidades y la base de datos, necesita un objeto. ¿Cuál es la anotación de JPA que se utiliza para inyectar este objeto, que es gestionado por el contenedor EJB? Respuesta: @PersistenceContext ✓.
27. Dos Tipos de Mapeo
    Mencione dos formas principales que JPA ofrece para definir el mapeo relacional-objetos: Respuesta: Anotaciones (ej. @Entity, @OneToOne) y Descriptores XML ✓.
28. Inyección de una Fachada de Sesión
    En el proyecto AppAlumnos, para que el Manejador (un HttpServlet) acceda a la funcionalidad de la entidad Facultad, se inyecta la clase FacultadFacade usando la anotación @EJB. Este proceso de inyección, donde el contenedor proporciona la instancia, ejemplifica el principio de Inversión de Control (aunque la fuente solo menciona la anotación @EJB, el concepto de inyectar componentes del servidor es clave en EJB).
29. Estado de Conversación
    En el contexto de Session Beans Stateful, ¿a qué se refiere el término "estado de conversación"? Respuesta: Se refiere a los valores de las variables de instancia del bean que se mantienen mientras el cliente interactúa con él.
30. Persistencia con API Criteria
    JPA provee la API Criteria como un método alternativo a JPQL para construir consultas. ¿Cuál es el principal beneficio de usar la API Criteria en comparación con escribir consultas JPQL

como cadenas de texto (String)? Respuesta: Permite construir consultas de manera programática (en lugar de como texto plano), lo que facilita la verificación de tipos y reduce errores en tiempo de ejecución.