PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

PROGRAMACIÓN 3
6ta práctica (tipo b)
(Primer Semestre 2025)

Indicaciones Generales:

- Duración 1h 50 minutos (Inicio: 8:00 a.m. - Fin: 9:50 a.m.) (Subida y verificación de archivos en PAIDEIA: 9:50 a.m. a 9:55 a.m.) (Salida del Laboratorio: 9:55 a.m. a 10:00 a.m.)
- Se les recuerda que, de acuerdo con el reglamento disciplinario de nuestra institución, constituye una falta grave copiar del trabajo realizado por otro estudiante o cometer plagio para el desarrollo de esta práctica.
- Para el desarrollo de toda la práctica debe utilizar el sistema operativo Windows y JDK 21.
- Para el desarrollo de las preguntas debe usar APACHE NETBEANS.
- Está permitido el uso de Internet (únicamente para consultar páginas oficiales de Oracle). No obstante, está prohibida toda forma de comunicación con otros estudiantes o terceros.
- PUEDE UTILIZAR MATERIAL DE CONSULTA. Antes de comenzar el laboratorio, descargue todos los proyectos, apuntes, diapositivas que utilizará.
- Cada alumno debería asistir al laboratorio con una base de datos MySQL previamente configurada en su cuenta AWS academy.
- Se considerará en la calificación el uso de buenas prácticas de programación (aquellas vistas en clase).
- Para las preguntas 2 y 3 puede usar procedimientos almacenados o sentencias SQL

Puntaje total: 20 puntos

Cuestionario

- La finalidad principal de este laboratorio es la de reforzar los conceptos contenidos en las clases 07 y 08: Conexión a bases de datos, CRUD y patrón de persistencia DAO.
- Al finalizar la práctica, comprima la carpeta dada en las indicaciones iniciales empleando el programa Zip que viene por defecto en el Windows, no se aceptarán los trabajos compactados con otros programas como RAR, WinRAR, 7zip o similares.

Descripción del caso

Una clínica veterinaria ha identificado la necesidad de mejorar su sistema de gestión de consultas médicas para garantizar un mayor control, organización y eficiencia en la atención de sus pacientes. Con este objetivo, se desea implementar una base de datos que permita almacenar de forma estructurada toda la información relevante relacionada con las consultas.

El sistema deberá ser capaz de registrar y organizar la programación de citas médicas, asociando cada una con un paciente, que puede ser un perro o un gato, su tutor responsable (es decir, la persona que lo cuida y lleva a la consulta) y el médico veterinario encargado de la atención. Para cada paciente, se almacenará información básica como nombre, especie, raza, edad y estado de salud si fuera necesario. El tutor deberá estar identificado con datos como nombre completo, documento de identidad, dirección y número de contacto. Por su parte, el médico tratante deberá estar registrado con su nombre y especialidad.

El programa deberá ser capaz de:

1. CRUD de pacientes, tutores y médicos tratantes y citas médicas para cargarlos en el sistema.
2. Registrar nuevas consultas médicas, asociando al paciente, su tutor y el médico tratante correspondiente.
3. Registrar descuentos promocionales para los pacientes.
4. Ejecutar pruebas automatizadas para verificar el correcto funcionamiento.

A continuación, se muestra un diagrama ER (Entidad Relación) con las tablas en la base de datos.

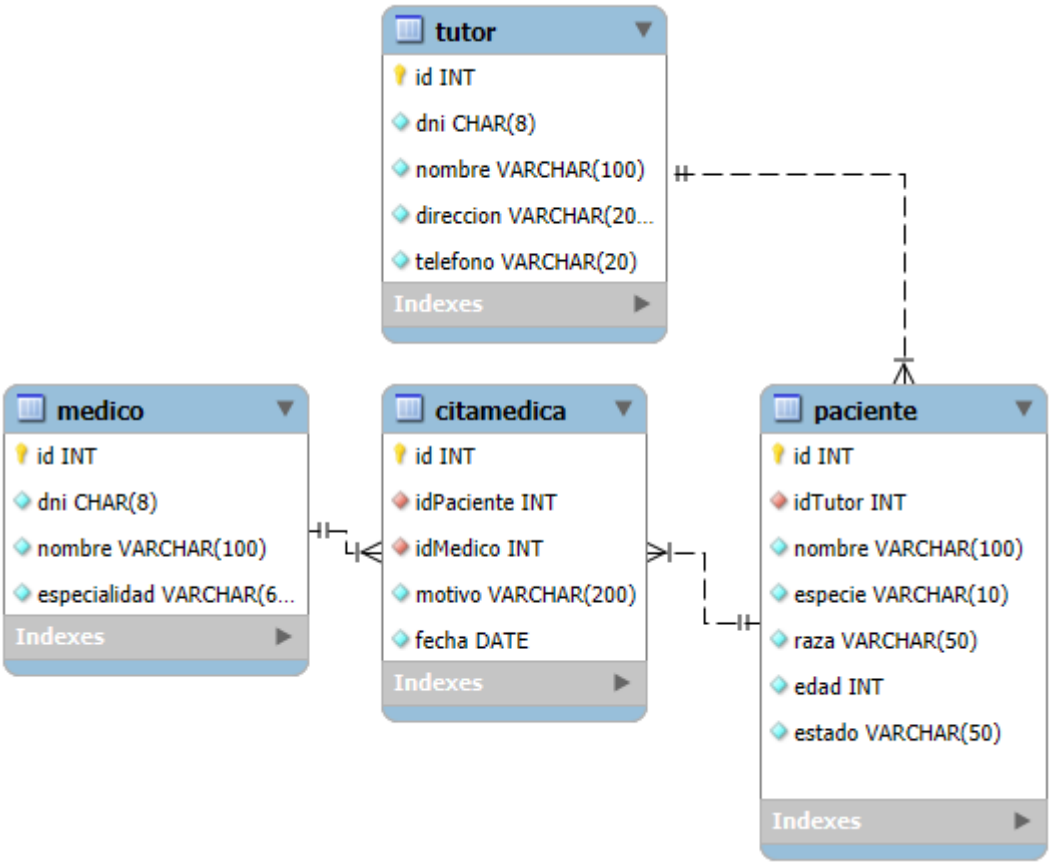

Figura 1 - Diagrama Entidad Relación

Cabe resaltar que todas las interacciones con la base de datos deben ser realizadas usando el patrón DAO, no se aceptarán otros patrones de persistencia.

Con esta información se elaborará un proyecto en Netbeans con Maven que deberá tener la siguiente estructura:

1. **CitasVet:**
   - Este es el proyecto principal que contendrá los módulos que se describen a continuación.

2. **Módulo CitasVetModelo**:
   - Contiene las clases de dominio.

3. **Módulo CitasVetDBManager**:
   - Contiene la implementación de un DBManager.

4. **Módulo CitasVetPersistencia**:
   - Define una dependencia con el módulo CitasVetModelo.
   - Define una dependencia con el módulo CitasVetDBManager.
   - Contiene el archivo .properties con la configuración necesaria para conectarse con la base de datos.
   - Contiene pruebas automatizadas para las clases que implementan el patrón DAO.

Se hará entrega de los siguientes artefactos antes de iniciar con la práctica.

1. Proyecto en NetBeans con el proyecto y módulos configurados.
2. Archivos con la definición de tablas y procedimientos almacenados el cual contiene la definición de las tablas en la base de datos.
3. **Pruebas automatizadas con JUnit completas, las cuales serán ejecutadas para asegurar que las preguntas fueron resueltas correctamente.**

Pregunta 1 (5 puntos)

Actualiza el método `cadenaConexion` del proyecto `CitasVetDBManager` para delegar la construcción de la cadena de conexión a una clase denominada `CadenaConexionBuilder`, la cual deberá implementar una lógica flexible y mantenible para ensamblar los distintos componentes de la cadena.

El patrón **Builder** separa la construcción de un objeto complejo de su representación, permitiendo crear distintas versiones del objeto paso a paso, mediante una clase constructora especializada que encapsula el proceso de ensamblaje. Es útil cuando se requiere mayor control y claridad en la creación de objetos.

El siguiente ejemplo ilustra su uso:

```
private String cadenaConexion(String host, int puerto, String esquema) {
    CadenaConexionBuilder builder = new CadenaConexionBuilder()
        .setHost(host)
        .setPuerto(puerto)
        .setEsquema(esquema);
    return builder.build();
}
```

Pregunta 2 (10 puntos) (+3 puntos)

Parte 1: 6 puntos

En el paquete pe.com.citasvet.daoImpl en el proyecto CitasVetPersistencia se han definido las siguientes implementaciones.

- TutorDAOImpl
- PacienteDAOImpl
- MedicoDAOImpl

Con los siguientes métodos.

Método	Entrada	Salida
insertar	Modelo (objeto de dominio a insertar en la base de datos)	int
modificar	Modelo (objeto de dominio a modificar en la base de datos)	boolean
eliminar	Id del registro en la base de datos a eliminar	boolean

buscar	Id del registro en la base de datos a buscar	Modelo (objeto de dominio) encontrado en la base datos
listar		Lista de modelos (objetos de dominio) en la tabla de la base de datos.

Esas implementaciones actualmente no hacen un uso correcto de los principios de la programación orientada a objetos como son la herencia y el polimorfismo.

Refactorizar la clase PacienteDAOImpl para que haga uso correcto de los principios de la programación orientada a objetos.

1. Implementar una clase BaseDAOImpl, que sirva de base para, PacienteDAOImpl.
2. BaseDAOImpl debe definir métodos abstractos que permitan configurar los objetos CallableStatement o PreparedStatement para cada tipo de operación CRUD.
3. BaseDAOImpl debe evitar el código repetitivo en las clases derivadas.
4. Las clases derivadas deben implementar todos los métodos abstractos y proporcionar las implementaciones necesarias para llevar a cabo las operaciones definidas.
5. **Las pruebas automatizadas no se deben modificar, estas deberían seguir funcionando con normalidad luego de la refactorización.**

**** Por la refactorización de TutorDAOImpl +1.5 puntos.**
**** Por la refactorización de MedicoDAOImpl +1.5 puntos.**

Parte 2: 4 puntos
Luego de la refactorización:

1. Definir una interfaz ICitaMedicaDAO en el paquete adecuado en el proyecto CitasVetPersistencia. Esta interfaz debe definir métodos para para realizar las operaciones CRUD
2. Implementar la interfaz CitaMedicaDAOImpl en el paquete adecuado en el proyecto CitasVetPersistencia
3. Se debe usar las pruebas definidas en el archivo CitaMedicaDaoTest.java para confirmar la correcta implementación de CitaMedicaDAOImpl.
4. Es obligatorio que la clase CitaMedicaDAOImpl siga la refactorización de la primera parte. No se aceptará código repetitivo para esta clase.

Pregunta 3 (5 puntos)

En el paquete pe.com.citasvet.dao en el módulo CitasVetPersistencia agregar el siguiente método en la interfaz ICitaMedicaDAO

1. ICitaMedicaDAO

Método	Entrada	Salida
listarPorPaciente	id del paciente	Lista de modelos (objetos de dominio) en la tabla de citas para el paciente dado.

En el paquete pe.com.citasvet.daoimpl en el módulo CitasVetPersistencia modificar la siguiente clase que implementa la interfaz ICitaMedicaDAO e implementar el método listarPorPaciente.

1. CitaMedicaDAOImpl

Método	Entrada	Salida
listarPorPaciente	id del paciente	Extrae los registros de la tabla CitaMedica filtrados por el id del paciente y los convierte (mapea) en objetos de dominio correspondientes.

Profesores del curso: Freddy Paz, Andrés Melgar, Heider Sánchez, Eric Huiza

Pando, 30 de abril de 2025