Simulacro de práctica — JPA con Hibernate 6

Clínica Veterinaria · Acceso a Datos 2DAM

Base de datos: clinicaVeterinaria

La base de datos contiene cuatro tablas. Se proporciona al final del documento el script SQL con la creación de las tablas y la inserción de datos iniciales.

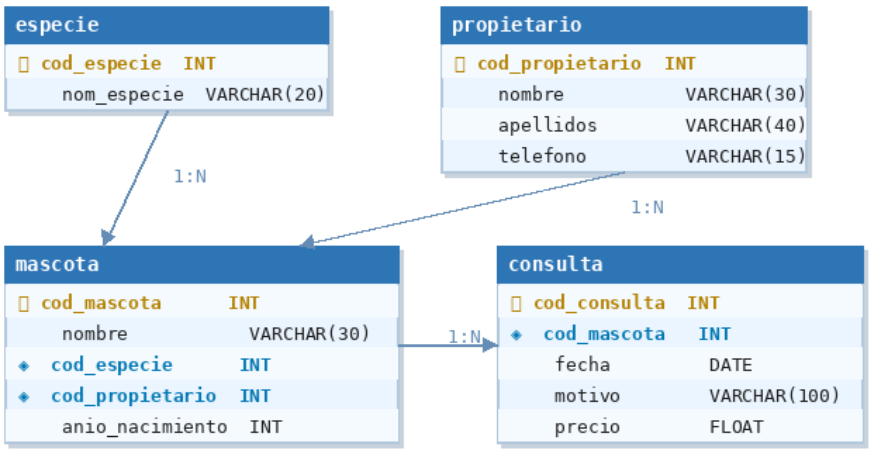

Figura 1. Diagrama entidad-relación de clinicaVeterinaria

Registros en tabla especie:

```
cod_especie  nom_especie
1            Perro
2            Gato
3            Conejo
```

Registros en tabla propietario:

```
cod_propietario  nombre    apellidos    telefono
1                Carlos    Pérez        600111222
2                Laura     Gómez        611222333
3                Marta     Ruiz         622333444
4                Javier    López        633444555
5                Elena     Sánchez      644555666
```

Registros en tabla mascota:

```
cod_mascota  nombre    cod_especie  cod_propietario  anio_nacimiento
1            Toby      1            1                2019
2            Luna      2            2                2021
3            Max       1            3                2018
4            Nala      2            1                2022
5            Pelusa    3            4                2020
6            Rocky     1            5                2017
7            Michi     2            2                2023
```

Registros en tabla consulta:

```
cod_consulta  cod_mascota  fecha       motivo              precio
1             1            2023-03-10  Revisión anual       35.0
2             1            2024-01-15  Vacunación           25.0
3             2            2023-06-20  Desparasitación      20.0
4             3            2022-11-05  Revisión anual       35.0
5             3            2024-02-28  Herida en pata       60.0
6             4            2023-09-14  Esterilización      120.0
7             5            2022-05-30  Revisión anual       35.0
8             6            2021-07-19  Vacunación           25.0
9             7            2024-03-01  Primer reconocim.    40.0
```

Estructura del proyecto

Crea en IntelliJ un proyecto Maven llamado **ApellidoNombre_simulacro_veterinaria**. Organiza los paquetes y clases como prefieras (estructura de capas o clase única con métodos estáticos). Debes incluir al menos:

- **JpaUtil.java** — gestión del EntityManagerFactory (disponible en tus proyectos de clase).
- **Principal.java** — clase con el método **main** desde el que se ejecutan todos los métodos.

En todos los métodos muestra un mensaje por consola indicando el método que se ejecuta, y un mensaje de éxito o error según el resultado.

Se pide

Apt.	Puntos	Tipo	Descripción
1	1,5 pts	Configuración	Configura JPA con Hibernate 6: crea las entidades con anotaciones JPA y el persistence.xml.
2	0,75 pts	**CON JPQL**	mostrarEspecies — muestra el contenido de la tabla especie.
3	1,5 pts	**CON JPQL**	mostrarMascotas — muestra todos los datos de las mascotas. El toString debe mostrar el nombre de la especie y el nombre+apellidos del propietario (no sus códigos). Penalización de −0,25 si no se sobreescribe toString.
4	1,75 pts	**SIN JPQL**	insertarPropietario — inserta un nuevo propietario con los datos recibidos por parámetro. A continuación muestra el contenido actualizado de la tabla propietario.
5	1,75 pts	**CON JPQL**	mostrarMascotasPorEspecie — recibe un cod_especie y muestra las mascotas de esa especie. Si no existe ninguna mascota con ese código de especie, indicarlo por consola.
6	2 pts	**CON JPQL**	mostrarConsultasPorPropietario — recibe un apellido de propietario y muestra las consultas de sus mascotas: nombre de la mascota, fecha, motivo y precio, ordenadas por fecha descendente. Todos los valores deben estar parametrizados.

Apt.	Puntos	Tipo	Descripción
7	0,75 pts	**CON JPQL**	eliminarConsultasAnteriores — recibe un año (int) y elimina todas las consultas cuya fecha sea anterior al 1 de enero de ese año. Muestra cuántas consultas se han eliminado. Todos los valores deben estar parametrizados.

Detalle de los apartados

Apartado 1 (1,5 puntos) — Configuración JPA

Lleva a cabo todos los pasos necesarios para configurar JPA con Hibernate 6 en el proyecto y trabajar con las cuatro tablas de la base de datos: crea las entidades con anotaciones JPA y configura el archivo `persistence.xml`.

Apartado 2 (0,75 puntos) — CON JPQL — mostrarEspecies

Muestra por consola el contenido completo de la tabla **especie**.

Ejemplo de salida:

```
*** Método mostrarEspecies
*** Contenido de la tabla 'especie':
codEspecie=1 nomEspecie=Perro
codEspecie=2 nomEspecie=Gato
codEspecie=3 nomEspecie=Conejo
```

Apartado 3 (1,5 puntos) — CON JPQL — mostrarMascotas

Muestra por consola los datos de todas las mascotas, sobreescribiendo el método `toString` en la entidad `Mascota`. La salida debe mostrar el **nombre de la especie** y el **nombre y apellidos del propietario**, no sus códigos.

Ejemplo de salida:

```
*** Método mostrarMascotas
*** Datos de todas las mascotas:
codMascota=1 nombre=Toby especie=Perro propietario=Carlos Pérez
anioNac=2019
codMascota=2 nombre=Luna especie=Gato propietario=Laura Gómez
anioNac=2021
codMascota=3 nombre=Max especie=Perro propietario=Marta Ruiz
anioNac=2018
codMascota=4 nombre=Nala especie=Gato propietario=Carlos Pérez
anioNac=2022
codMascota=5 nombre=Pelusa especie=Conejo propietario=Javier López
anioNac=2020
codMascota=6 nombre=Rocky especie=Perro propietario=Elena Sánchez
anioNac=2017
codMascota=7 nombre=Michi especie=Gato propietario=Laura Gómez
anioNac=2023
```

Apartado 4 (1,75 puntos) — SIN JPQL — insertarPropietario

Recibe como parámetros el **nombre**, los **apellidos** y el **teléfono** de un nuevo propietario e inserta el registro en la base de datos **sin usar JPQL**. A continuación, muestra el contenido actualizado de la tabla propietario.

Firma del método:

```
void insertarPropietario(String nombre, String apellidos, String
telefono)
```

Ejemplo de salida (llamando con "Ana", "Torres", "655666777"):

```
*** Método insertarPropietario
Propietario insertado correctamente.
*** Contenido actualizado de la tabla 'propietario':
codPropietario=1 nombre=Carlos apellidos=Pérez telefono=600111222
codPropietario=2 nombre=Laura apellidos=Gómez telefono=611222333
codPropietario=3 nombre=Marta apellidos=Ruiz telefono=622333444
codPropietario=4 nombre=Javier apellidos=López telefono=633444555
codPropietario=5 nombre=Elena apellidos=Sánchez telefono=644555666
codPropietario=6 nombre=Ana apellidos=Torres telefono=655666777
```

Apartado 5 (1,75 puntos) — CON JPQL — mostrarMascotasPorEspecie

Recibe un **cod_especie** y muestra las mascotas que pertenecen a esa especie. Si no existe ninguna mascota con ese código de especie, se indicará mediante un mensaje en consola.

Firma del método:

```
void mostrarMascotasPorEspecie(int codEspecie)
```

Ejemplo de salida (codEspecie=1):

```
*** Método mostrarMascotasPorEspecie
Mascotas de la especie 'Perro':
codMascota=1 nombre=Toby especie=Perro propietario=Carlos Pérez
anioNac=2019
codMascota=3 nombre=Max especie=Perro propietario=Marta Ruiz
anioNac=2018
codMascota=6 nombre=Rocky especie=Perro propietario=Elena Sánchez
anioNac=2017
```

Ejemplo de salida (codEspecie=99):

```
*** Método mostrarMascotasPorEspecie
No existe ninguna mascota con el código de especie 99.
```

Apartado 6 (2 puntos) — CON JPQL — mostrarConsultasPorPropietario

Recibe el **apellido** de un propietario y muestra todas las consultas de sus mascotas, mostrando: nombre de la mascota, fecha, motivo y precio. Ordenado por **fecha descendente**. Las consultas JPQL deben tener todos los valores parametrizados.

Firma del método:

```
void mostrarConsultasPorPropietario(String apellidoPropietario)
```

Ejemplo de salida (apellido="Pérez"):

```
*** Método mostrarConsultasPorPropietario
Consultas de mascotas del propietario 'Pérez':
mascota=Toby fecha=2024-01-15 motivo=Vacunación precio=25.0
mascota=Nala fecha=2023-09-14 motivo=Esterilización precio=120.0
mascota=Toby fecha=2023-03-10 motivo=Revisión anual precio=35.0
```

Pista: en la consulta JPQL puedes navegar por los atributos de las relaciones directamente, por ejemplo `c.mascota.nombre` o `c.mascota.propietario.apellidos`. No necesitas escribir ningún JOIN.

Apartado 7 (0,75 puntos) — CON JPQL — eliminarConsultasAnteriores

Recibe un **año** como entero y elimina todas las consultas cuya fecha sea **anterior al 1 de enero de ese año**. Muestra el número de consultas eliminadas y el contenido actualizado de la tabla. Todos los valores deben estar parametrizados.

Firma del método:

```
void eliminarConsultasAnteriores(int anio)
```

Ejemplo de salida (anio=2023):

```
*** Método eliminarConsultasAnteriores
Consultas eliminadas anteriores a 2023: 3
*** Contenido actualizado de la tabla 'consulta':
codConsulta=1 mascota=Toby fecha=2023-03-10 motivo=Revisión anual
precio=35.0
codConsulta=2 mascota=Toby fecha=2024-01-15 motivo=Vacunación
precio=25.0
codConsulta=3 mascota=Luna fecha=2023-06-20 motivo=Desparasitación
precio=20.0
codConsulta=5 mascota=Max fecha=2024-02-28 motivo=Herida en pata
precio=60.0
codConsulta=6 mascota=Nala fecha=2023-09-14 motivo=Esterilización
precio=120.0
codConsulta=9 mascota=Michi fecha=2024-03-01 motivo=Primer reconocim.
precio=40.0
```

Pista: usa `java.time.LocalDate.of(anio, 1, 1)` para construir la fecha límite y pásala como parámetro a la consulta JPQL.

Resumen de métodos a implementar

- `void mostrarEspecies()` — muestra la tabla especie. **CON JPQL**
- `void mostrarMascotas()` — muestra todas las mascotas con nombre de especie y propietario. **CON JPQL**
- `void insertarPropietario(String nombre, String apellidos, String telefono)` — inserta un propietario. **SIN JPQL**

- `void mostrarMascotasPorEspecie(int codEspecie)` — filtra mascotas por especie. **CON JPQL**
- `void mostrarConsultasPorPropietario(String apellidoPropietario)` — consultas de las mascotas de un propietario, por fecha DESC. **CON JPQL**
- `void eliminarConsultasAnteriores(int anio)` — elimina consultas anteriores a una fecha. **CON JPQL**

Script de creación de la base de datos e inserción de datos de prueba

```
-- ============================================================
--  Base de datos: clinicaVeterinaria
--  Simulacro JPA con Hibernate 6 — Acceso a Datos 2DAM
-- ============================================================

DROP DATABASE IF EXISTS clinicaVeterinaria;
CREATE DATABASE clinicaVeterinaria CHARACTER SET utf8mb4 COLLATE utf8mb4_spanish_ci;
USE clinicaVeterinaria;

-- ------------------------------------------------------------
--  Tabla: especie
-- ------------------------------------------------------------
CREATE TABLE especie (
    cod_especie  INT          NOT NULL AUTO_INCREMENT,
    nom_especie  VARCHAR(20)  NOT NULL,
    PRIMARY KEY (cod_especie)
);

-- ------------------------------------------------------------
--  Tabla: propietario
-- ------------------------------------------------------------
CREATE TABLE propietario (
    cod_propietario  INT          NOT NULL AUTO_INCREMENT,
    nombre           VARCHAR(30)  NOT NULL,
    apellidos        VARCHAR(40)  NOT NULL,
    telefono         VARCHAR(15),
    PRIMARY KEY (cod_propietario)
);

-- ------------------------------------------------------------
--  Tabla: mascota
-- ------------------------------------------------------------
CREATE TABLE mascota (
    cod_mascota      INT          NOT NULL AUTO_INCREMENT,
    nombre           VARCHAR(30)  NOT NULL,
    cod_especie      INT          NOT NULL,
    cod_propietario  INT          NOT NULL,
    anio_nacimiento  INT,
    PRIMARY KEY (cod_mascota),
    CONSTRAINT fk_mascota_especie      FOREIGN KEY (cod_especie)     REFERENCES especie
(cod_especie),
    CONSTRAINT fk_mascota_propietario  FOREIGN KEY (cod_propietario) REFERENCES propietario
(cod_propietario)
);

-- ------------------------------------------------------------
--  Tabla: consulta
-- ------------------------------------------------------------
CREATE TABLE consulta (
    cod_consulta  INT           NOT NULL AUTO_INCREMENT,
    cod_mascota   INT           NOT NULL,
    fecha         DATE          NOT NULL,
    motivo        VARCHAR(100),
    precio        FLOAT,
    PRIMARY KEY (cod_consulta),
    CONSTRAINT fk_consulta_mascota FOREIGN KEY (cod_mascota) REFERENCES mascota (cod_mascota)
);

-- ============================================================
--  Datos iniciales
-- ============================================================

INSERT INTO especie (nom_especie) VALUES
    ('Perro'),
    ('Gato'),
    ('Conejo');
```

```
INSERT INTO propietario (nombre, apellidos, telefono) VALUES
    ('Carlos', 'Pérez',   '600111222'),
    ('Laura',  'Gómez',   '611222333'),
    ('Marta',  'Ruiz',    '622333444'),
    ('Javier', 'López',   '633444555'),
    ('Elena',  'Sánchez', '644555666');

INSERT INTO mascota (nombre, cod_especie, cod_propietario, anio_nacimiento) VALUES
    ('Toby',   1, 1, 2019),
    ('Luna',   2, 2, 2021),
    ('Max',    1, 3, 2018),
    ('Nala',   2, 1, 2022),
    ('Pelusa', 3, 4, 2020),
    ('Rocky',  1, 5, 2017),
    ('Michi',  2, 2, 2023);

INSERT INTO consulta (cod_mascota, fecha, motivo, precio) VALUES
    (1, '2023-03-10', 'Revisión anual',       35.0),
    (1, '2024-01-15', 'Vacunación',           25.0),
    (2, '2023-06-20', 'Desparasitación',      20.0),
    (3, '2022-11-05', 'Revisión anual',       35.0),
    (3, '2024-02-28', 'Herida en pata',       60.0),
    (4, '2023-09-14', 'Esterilización',      120.0),
    (5, '2022-05-30', 'Revisión anual',       35.0),
    (6, '2021-07-19', 'Vacunación',           25.0),
    (7, '2024-03-01', 'Primer reconocimiento', 40.0);
```