* Tabla caracteristica\_mascota:

Corresponde a la clase CaracteristicaConRta, son las características particulares de las mascotas. Necesitamos persistirlas para mostrarlas en publicaciones.

Restricciones: la FK mascota\_id no puede ser nula.

* Tabla foto:

Corresponde a la clase Foto. Nos interesa persistirla para tener un registro de a qué mascota o publicación pertenece cada foto y dónde se encuentra almacenada.

* Tabla mascota:

Corresponde a la clase Mascota. Nos interesa persistirla porque es una de las clases principales de nuestro dominio y tenemos que recordar los datos pertenecientes a la mascota y sus relaciones con los otros elementos del dominio.

* Tabla persona:

Corresponde a la clase Persona. Nos interesa persistirla para tener los datos particulares de la persona y recordarlos cuando vuelva a utilizar el sistema.

* Tabla usuario

Corresponde a la clase Usuario. La persistimos porque nos interesa utilizar esta información cuando intenta iniciar sesión y decidir si le damos acceso a la plataforma.

Restricciones: los campos contrasenia y nombreDeUsuario no pueden ser nulos.

* Tabla contacto

Corresponde a la clase Contacto. La persistimos ya que la necesitamos para poder enviar notificaciones a la persona.

Restricciones: la FK persona\_id no puede ser nula.

* Tabla rol

Corresponde a la clase Rol. Como es una clase abstracta de la cual heredan otras, utilizamos una estrategia de mapeo. Elegimos JOINED porque no hay una cantidad significativa de datos y las subclases comparten algunos atributos y tienen otros propios.

Restricciones: la FK persona\_id no puede ser nula.

* Tablas rescatista, duenio y voluntarios

Corresponden a las clases Rescatista, Duenio y Voluntario que heredan de Rol. Necesitamos persistirlas para conocer la información particular de cada rol. En el caso de duenio y rescatista para saber su relación con una mascota y voluntario su relación con una organización.

Restricciones: la FK organización\_id no puede ser nula en la tabla voluntario.

* Tabla organización

Corresponde a la clase Organización. Nos interesa persistirla para conocer su nombre y su ubicación. Además, otras tablas se relacionan con ella.

Restricciones: la FK ubicacion\_id no puede ser nula.

* Tabla pregunta

Corresponde a la clase Pregunta. Necesitamos persistir las preguntas y sus respuestas posibles para mostrarlas y que la persona pueda ingresar su respuesta. Como tenemos una lista de respuestas posibles, esto lo mapeamos con @ElementCollection, lo que nos generó una tabla con el id\_pregunta y sus respuestas posibles.

Restricciones: la FK cuestionario\_id no puede ser nula.

* Tabla pregunta\_respuestas

Es la generada por @ElementCollection.

* Tabla cuestionario

Corresponde a la clase Cuestionario. Persistimos su id ya que lo utilizamos para identificar qué preguntas corresponden a un cuestionario en particular.

* Tabla respuestaConcreta

Corresponde a la clase RespuestaConcreta. Nos interesa conocer qué pregunta se respondió, cuál fue la respuesta y a qué cuestionario contestado corresponde.

Restricciones: las FK pregunta\_id y cuestionarioContestado\_id no pueden ser nulas.

* Tabla cuestionarioContestado

Corresponde a la clase CuestionarioContestado. (tenemos diferencias entre el modelo de datos y el código)

* Tablas de publicaciones

Para mapear estas clases utilizamos Table Per Class. No persistimos la clase PublicacionGenerica, sí sus subclases ya que no tienen muchos atributos en común. La herencia la necesitamos para poder tratar a las publicaciones polimórficamente.

La tabla publicación\_intención\_adopcion corresponde a la clase PublicacionIntencionAdopcion.

Restricciones: las FK adoptante\_id, organización\_id y cuestionarioContestadoPreferenciasYComodidades\_id no pueden ser nulas.

La tabla publicación\_perdida\_registrada corresponde a la clase PublicacionPerdidaRegistrada.

Restricciones: la FK mascota\_id no puede ser nula.

La tabla publicación\_mascota\_encontrada corresponde a la clase PublicacionMascotaEncontrada.

Restricciones: las FK rescatista\_id, organización\_id y datosMascotaEncontrada\_id no pueden ser nulas.

La tabla publicación\_en\_adopcion corresponde a la clase PublicacionEnAdopcion.

Restricciones: las FK mascota\_id, organización\_id y cuestionarioContestado\_id no pueden ser nulas.

Necesitamos persistir estas clases para poder mostrarlas, aprobarlas, rechazarlas o darlas de baja.

* Tabla ubicación

Corresponde a la clase Lugar. La necesitamos porque queremos persistir la ubicación de una organización o dónde fue encontrada una mascota.

Restricciones: los campos latitud y longitud no pueden ser nulos.

* Tabla datos\_mascota\_encontrada

Corresponde a la clase DatosMascotaEncontrada. Necesitamos persistirla porque queremos mostrar los datos de una mascota encontrada que difieren de los datos pertenecientes a la clase Mascota.

Restricciones: la FK ubicacion\_id no puede ser nula.

No persistimos las clases: Admision, Hogar, Listado de hogares y Ubicación porque los datos los provee la API.

Sobre las notificaciones solo persistimos la estrategia elegida como String, ya que el resto es comportamiento.

No persistimos GestorDePublicaciones,MatchearPublicacionesEnAdopcion, las clases relacionadas con la validación de la contraseña ya que están enfocadas en el comportamiento y no tienen atributos que necesitemos guardar. Tampoco persistimos las clases del paquete services por el mismo motivo.