

**Justificaciones de los Elementos Persistidos**

Grupo 6

Diseño de Sistemas

Fecha de entrega : 13/09/2021.

***Aclaraciones Previas:***

-En este TP, la persistencia de datos la realizamos con una base de datos de tipo Relacional. Como la base de datos relacional no tiene una relación directa con el paradigma orientado a objetos hay que mapearlos de una forma para poder hacer que estos datos se persistan en la tabla (Aquí nace la técnica de ORM).

- La base de datos relacional no se lleva bien con el paradigma orientado a objetos por lo cual hay que mapearlos de una forma para poder hacer que estos datos se persistan en la tabla.

- ORM o Mapeo de objetos-Relacional es una técnica usada para convertir los tipos de datos modelados en el paradigma orientado a objetos a datos con los que trabaja una base de datos de tipo relacional

Justificación de:   
o Los elementos del modelo que fueron necesarios persistir   
o Cómo se resolvieron los impedance mismatches.   
Aclaración: no olvidar las estrategias de mapeo de herencia, si fueran necesarias.   
o Las estructuras de datos que se **desnormalizaron**, o que deberían estarlo

***Entidades Persistidas: Todas las entidades que decidimos persistir cuentan con una clave primaria “id” que sirve para identificarlas de forma unívoca.***

# **Usuario:** -Impedance mismatches

Un usuario tiene:   
 Una relación “One to Many” con mascota: Porque una instancia de usuario puede estar apuntando a muchas instancias de la entidad mascota, y muchas instancias de la entidad mascota solo pueden estar cada una apuntando a un usuario, es decir, un dueño, debido a que un usuario puede querer registrar a muchas mascotas suyas en el sistema.  
 Una relación “One to Many” con la entidad Usuario-Contacto: Porque una instancia de usuario puede estar apuntando a muchas instancias de la entidad contacto, y muchas instancias de la entidad contacto pueden estar apuntando cada una a un usuario, debido a que un usuario puede querer registrar a varias personas como contactos en el sistema.

-Estrategia utilizada de Mapeo de Herencia:

Table per class: La clase Usuario hereda de la clase abstracta DatosBásicos, por eso decidimos utilizar la estrategia de una tabla por cada clase hija de la clase DatosBasicos.

-Estructuras de datos desnormalizadas:  
 domicilio: ya que no incluimos ni entre que calles está, ni en la ciudad en la que se encuentra porque asumimos que la aplicación se utiliza en Buenos Aires.

# **Persona**:-Impedance mismatches

Una Persona tiene:   
 Una relación “One to Many” con mascota: Porque una instancia de usuario puede estar apuntando a muchas instancias de la entidad mascota, y una instancia de la entidad mascota solo puede estar apuntando a un usuario, es decir, un dueño.  
 Una relación “One to One” con la entidad Interesado: Porque una instancia de la entidad Persona, puede estar apuntando a solo una instancia de la entidad Interesado, y viceversa, debido a que una persona puede querer dar en adopción a su mascota.

-Estrategia utilizada de Mapeo de Herencia:

Table per class: La clase Persona hereda de la clase abstracta DatosBasicos, por eso decidimos utilizar la estrategia de una tabla por cada clase hija de la clase DatosBasicos.

-Estructuras de datos desnormalizadas:  
 domicilio: ya que no incluimos ni entre que calles está, ni en la ciudad en la que se encuentra porque asumimos que la aplicación se utiliza en Buenos Aires.

# **Contacto** -Impedance mismatches

Un contacto tiene:

Una relación “One to Many” con contacto\_estrategiaDeNotificacion: Porque cada instancia de contacto puede estar apuntando a muchas estrategias de notificación. Esto se da ya que un contacto puede tener como estrategia de notificación al WhatsApp, SMS y Email simultáneamente.

Una relación “One to Many” con rescatista:Porque una instancia de contacto puede estar apuntando a muchas instancias de rescatista. Esto se debe a que muchos rescatistas pudieron haber establecido a la misma instancia de contacto como su contacto.

Una relación “Many to Many” con Persona: Debido a que esto es una relación “Many to Many”, fue necesario romper la misma mediante una tabla intermedia: esta tabla es “persona\_contacto”. La relación “Many to Many” se da debido a que muchas instancias de la clase Contacto pueden estar referenciando a la misma instancia de la clase Persona y, al mismo tiempo, muchas instancias de la clase Persona pueden estar referenciando a la misma instancia de Contacto.  
-Estructuras de datos desnormalizadas:

# **Rescatista:**

-Tiene una Clave Foránea con la entidad Contacto, ya que un contacto puede relacionarse con varios rescatistas (por ejemplo: si a una persona se le pierden el mismo dia dos mascotas)  
-Impedance mismatches

Un Rescatista tiene:   
 Una relación “Many to One” con un Contacto: Porque una instancia de usuario puede estar apuntando a muchas instancias de la entidad mascota, y una instancia de la entidad mascota solo puede estar apuntando a un usuario, es decir, un dueño.  
 Una relación “One to Many” con la entidad rescatista-rutas Fotos: Porque una instancia de la entidad Rescatista, puede estar apuntando a varias instancias de la entidad rescatista-rutas Fotos, ya que un rescatista puede realizar muchas fotos de una misma mascota.

Una relación “One to Many” con la entidad Publicación Mascota Perdida: Porque una instancia de la entidad rescatista, puede estar apuntando a varias instancias de la entidad Publicación Mascota, ya que un rescatista puede realizar varias publicaciones de mascotas perdidas.

-Estrategia utilizada de Mapeo de Herencia:

Table per class: La clase Rescatista hereda de la clase abstracta DatosBásicos, por eso decidimos utilizar la estrategia de una tabla por cada clase hija de la clase DatosBasicos.

-Estructuras de datos desnormalizadas:  
 domicilio: ya que no incluimos ni entre que calles está, ni en la ciudad en la que se encuentra porque asumimos que la aplicación se utiliza en Buenos Aires.  
 descripción Mascotas: ya que se podría haber realizado una entidad aparte que contenga la descripción de la mascota, con su nombre, su id, pero nos pareció más óptimo incluirlo todo dentro del rescatista para que su busqueda sea mas rapida

## Entidades Realizadas para Normalizar la entidad Rescatista:

# **(Rescatista-rutaFotos)**

Esta entidad la podríamos haber definido dentro de la entidad Rescatista, pero para realizar la normalización del mismo y que los datos queden más claros, optamos por realizar una entidad aparte que se encargue de persistir las rutas de las fotos tomadas por cada rescatista.

# **Publicación Mascota Perdida:**

-Tiene una Clave Foránea con la entidad Organización, ya que varias publicaciones pueden estar apuntando a una misma organización

-Impedance mismatches

Una Publicación de mascota perdida tiene:   
 Una relación “Many to One” con un Rescatista: Porque varias instancias de Publicaciones pueden estar apuntando a un mismo Rescatista que las haya generado.  
 Una relación “Many to One” con una Organización: Porque varias instancias de publicación pueden relacionarse con una única organización.

-Estrategia utilizada de Mapeo de Herencia:

Table per class: La clase PublicacionMascotaPerdida hereda de la clase abstracta Publicacion, por eso decidimos utilizar la estrategia de una tabla por cada clase hija de la clase Publicacion.

-Estructuras de datos desnormalizadas:  
 domicilio: ya que no incluimos ni entre que calles está, ni en la ciudad en la que se encuentra porque asumimos que la aplicación se utiliza en Buenos Aires.  
 descripción: ya que se podría haber realizado una entidad aparte que contenga la descripción, con su nombre, su id.

# **Administrador**

-Tiene una Clave Foránea con la entidad Organización ya que un administrador pertenece a una organización, está asociado a la misma.

-Impedance mismatches:

Un Administrador tiene:

Una relación “Many to One” con Organización ya que cada instancia de administrador se relaciona únicamente con una única organización, pero cada instancia de organización se relaciona con muchas instancias de administrador. Es decir, un administrador pertenece a una sola organización, pero cada organización tiene muchos administradores que trabajan en ella.

-Estrategia utilizada de Mapeo de Herencia

Table per class:La clase Administrador hereda de la clase abstracta DatosBásicos, por eso decidimos utilizar la estrategia de una tabla por cada clase hija de la clase DatosBasicos.

-Estructuras de datos desnormalizadas:

# **Mascota**

-Tiene dos Claves Foráneas: una con la entidad persona (id\_persona) y otra con la entidad organización (id\_organización), a su vez tiene una clave primaria que la identifica de forma inequívoca del resto de las demás entidades.

-Impedance mismatches:

Mascota tiene:

Una relación “Many to One” con usuario: Esto ocurre debido a que muchas instancias diferentes de mascota pueden hacer referencia a la misma instancia de usuario. Esto se da en el caso de un usuario que tenga más de una mascota. Cada mascota pertenece a un sólo usuario, pero ese mismo usuario tiene diferentes mascotas.

Una relación “Many to One” con persona: Porque cada instancia de persona puede hacer referencia a muchas instancias diferentes de mascota. Similar a la relación explicada anteriormente.

Una relación “One to Many” con mascota\_ruta: Esta ruta hace referencia al path donde se guardan las fotos de la mascota. Cada instancia de mascota puede estar haciendo referencia a muchas instancias de mascota\_ruta. Es el caso de una mascota que tiene subidas muchas fotos.

Una relación “One to Many” con publicacionDarEnAdopcion: Una misma instancia de mascota puede estar relacionada con diferentes instancias de publicaciones para dar en adopción.

Una relación “One to Many” con caracteristicaDeMascotas: Debido a que cada instancia de mascota hace referencia a muchas instancias de la clase caracteristicaDeMascotas. Esto se da ya que cada mascota puede tener asociados diferentes características de forma simultánea.

Una relación “Many to One” con organización: Esto se debe a que cada instancia de mascota está relacionada con una y solo una organización. Cada mascota pertenece únicamente a una sola organización, debido a que no tendría sentido que una mascota pertenezca a varias organizaciones. “

-Estructuras de datos desnormalizadas:

## Entidades Realizadas para Normalizar la entidad Rescatista:

# **(Mascota-ruta)**

similar al caso Rescatista-Fotos, esta entidad se hizo con el fin de normalizar la entidad mascota, además de persistir de forma independiente la rutas de cada una de las mascotas.

# **Organización**

-Impedance mismatches

Una Organización tiene:

Una relación “One to Many” con publicación Dar En Adopción: Porque muchas instancias de la clase publicacion dar en adopción se pueden relacionar con una única instancia de organización que sería la organización que “publica” a las mascotas que tiene para adoptar  
 Una relación “One to Many” con característica De Mascotas: Porque como una organización puede tener varias mascotas, también puede tener varias características de dichas mascotas.  
 Una relación “One to Many” con Mascota: Porque una organización se relaciona con muchas mascotas, ya que una misma organización puede tener físicamente a muchas mascotas.   
 Una relación “One to Many” con Publicación Mascota Perdida: Porque una misma organización puede realizar muchas publicaciones de mascotas perdidas porque puede suceder que se pierdan varias mascotas a la vez con lo cual se generan varias publicaciones.   
 Una relación “One to Many” con Administrador:Esto sucede porque una organización puede tener varios administradores que le pertenezcan y que estos se encargan de administrar.   
 Una relación “One to Many” con Voluntario: Esto es porque al igual que el caso anterior una organización puede relacionarse con varios voluntarios.  
 Una relación “One to Many” con Publicación Adoptar : Porque una organización se relaciona con una o varias publicaciones para incorporar nuevas mascotas a la organización.

-Estructuras de datos desnormalizadas:  
 domicilio: ya que no incluimos ni entre que calles está, ni en la ciudad en la que se encuentra porque asumimos que la aplicación se utiliza en Buenos Aires.  
 Redes Sociales: ya que se podría haber realizado una entidad aparte que contenga todas las redes sociales utilizadas con una breve descripción y datos adicionales.   
  
-Estructuras de datos desnormalizadas:

# **Estrategia de notificación**

-Impedance mismatches:  
 Un Administrador tiene:

Una relación “One to Many” con Contacto-Estrategia de notificación porque una Estrategia de notificación puede tener varias instancias de Contacto-Estrategia de Notificación

-Estrategia utilizada de Mapeo de Herencia

Se utiliza la estrategia de Single Table con el discriminador tipoDeEstrategiaDeNotificacion, porque son muy pocas columnas, y de esta manera se simplifican las consultas y aumenta la performance. En una misma tabla se incluyen a la clase EstrategiaDeNotificacion (clase padre) y a sus clases hijas Whatsapp, Mail y SMS. En cada clase hija se describe el discriminatorValue con el nombre de la clase para diferenciarlas en la tabla.

**interesado**

- Tiene una Clave Foránea (id\_persona) mediante la cual se relaciona con la entidad persona.

-Impedance mismatches:

Interesado tiene:

Una relación “One to One” con la entidad Persona: Porque una instancia de la entidad Persona, puede estar apuntando a solo una instancia de la entidad Interesado, y viceversa, debido a que una persona puede querer dar en adopción a su mascota.

-Estrategia utilizada de Mapeo de Herencia:

Table per class: La clase Interesado hereda de la clase Persona, y se utiliza la estrategia de mapeo de herencia Joined para poder persistir ambas tablas por separado, tanto la tabla de la clase hija, como la clase padre.

**Características de mascota:**-Tiene una Clave Foránea con la entidad Organización ya que las características de la mascota las define la organización en la que la mascota se encuentra  
-Tiene una Clave Foránea con la entidad mascota ya que las características de la mascota tienen que estar asociadas a una mascota, la cual asociar dichas características.  
  
-Impedance mismatches:  
 Características de Mascotas tiene:

Una relación “Many to One” con mascota, ya que muchas instancias de características de mascotas puede relacionarse con una única mascota, la cual tenga asociada dichas características.  
 Una relación “Many to One” con Organización, ya que varias instancias de esta entidad deben poder relacionarse con una única organización que contenga dicha mascota.   
 Una relación “Many to One” con Características De Mascotas-características, ya que esta relación se realizó para normalizar las características y así poder lograr que la clase características de mascotas se relacione con varias instancias de “características”.

## Entidades Realizadas para Normalizar la entidad Característica de mascota: **(Características de mascota-Caracteristicas)** Esta entidad se realizó para poder persistir los datos de características de una forma normalizada y clara, pero tranquilamente esta entidad podría estar dentro de características de mascota como si fuese un atributo más.

**Hogar:**-Tiene una Clave Foránea con la entidad Admisión, ya que es necesario poder saber mediante otra entidad si es posible la admisión de nuevas mascotas al hogar.  
-Tiene una Clave Foránea con la entidad ubicación ya que así podemos caracterizar y filtrar los hogares según su ubicación de una manera más óptima y rápida

# -Impedance mismatches: Características de Hogar :

Una relación “One to One” con ubicación, ya que una única instancia de la clase hogar se relaciona con una única instancia de la clase ubicación.(es imposible que dos hogares tengan la misma ubicación)  
 Una relación “One to One” con admisión, ya que una única instancia de hogar se relaciona con una única instancia de la clase admisión, para poder saber si es posible admitir nuevas mascotas al hogar.  
 Una relación “One to Many” con Hogar-Características, ya que un hogar puede relacionarse con varias instancias de la clase características. -Estrategia utilizada de Mapeo de Herencia

Entidades Realizadas para Normalizar la entidad Hogar:

**(Ubicación)  
(Admisiones)  
(Hogar\_caracteristicas)**  
El objetivo de estas 3 entidades es para poder administrar de una forma más eficiente los datos que se encuentran dentro de la entidad hogar, ya que así nos facilita a la hora de buscar y administrar los datos de un hogar de forma más rápida y clara.

# **Publicación**

Sabiendo que necesitamos persistir tres tipos de publicaciones (PublicaciónAdoptar, PublicacionDarEnAdopcion y PublicacionMascotaPerdida) tomamos la decisión de crear una “Superclase” (@MappedSuperClass) para así no repetir las mismas columnas que tendrán estos tres tipos. (ID, Aprobada y Organización Asignada).

## ***Todas las publicaciones tendrán:***

**Una relación “many to one” con organización**: Esto es debido a que las publicaciones para dar en adopción van a depender de una organización en cuestión y a su vez estas organizaciones pueden tener muchas publicaciones. Por ello decidimos que cada organización deberá guardar el ID de su organización a través de una clave foránea.

-Impedance mismatches

**PublicacionDarEnAdopcion**  tiene:

**Una relación “many to one” con persona**: Esta relación se genera debido a que muchas instancias de PublicacionDarEnAdopcion pueden estar vinculadas a una persona. Mientras que una persona puede tener muchas instancias de publicaciones asociadas. Es por esto que cada publicación almacenará en una columna el ID de la persona asociada a esa publicación. (Clave Foránea)

**Una relación “One to One” con mascota**: Optamos por esta relación, debido a que cada mascota sólo podrá tener una instancia de publicación para darse en adopción. Así también, cada publicación corresponderá con una única mascota.

**PublicacionAdoptar** tiene:

**Una relación “many to one” con persona** debido a que una persona podrá generar muchas instancias de la publicación adoptar y estas publicaciones le corresponderán a esa persona. Cada publicación hará referencia a esa persona a través de su clave foránea.

# **Voluntario**

-Impedance mismatches

**Una relación “many to one” con organización** ya que las organizaciones tendrán muchos voluntarios a su cargo, mientras que cada voluntario corresponderá a una misma organización. Cada voluntario además de su clave primaria tendrá una clave foránea para poder referenciar e identificar a qué organización pertenece.

-Estrategia utilizada de Mapeo de Herencia:

Table per class: La clase Voluntario hereda de la clase abstracta DatosBásicos, por eso decidimos utilizar la estrategia de una tabla por cada clase hija de la clase DatosBasicos.

# 

# **-Estructuras de datos desnormalizadas:**

## (usuario-Contacto)

## (persona-Contacto)

## (Contacto-Estrategia de notificación)

Estas estructuras se realizaron con el fin de realizar la implementación del tipo Many to Many, en la cual muchas instancias de un tipo de estructura se relaciona con otra, de forma tal que muchas instancias de la primera clase se relaciona con muchas instancias de la segunda clase.