



DEPARTAMENTO  
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

## Trabajo práctico - Primera Parte

1er Cuatrimestre 2016

Bases de Datos

Integrante	LU	Correo electrónico
Almansi, Emilio	674/12	ealmansi@gmail.com
Fixman, Martín	391/11	martinfixman@gmail.com
Gunski, María Celeste	899/03	celestegunski@gmail.com
Maurizio, Miguel	635/11	miguelmaurizio_92@hotmail.com



**Facultad de Ciencias Exactas y Naturales**

Universidad de Buenos Aires

Ciudad Universitaria - (Pabellón I/Planta Baja)

Intendente Güiraldes 2160 - C1428EGA

Ciudad Autónoma de Buenos Aires - Rep. Argentina

Tel/Fax: (54 11) 4576-3359

<http://www.fcen.uba.ar>

# Índice

<b>1. Introducción</b>	<b>3</b>
1.1. Descripción del Problema . . . . .	3
1.2. Funcionalidades a implementar . . . . .	4
<b>2. Modelo de Entidad Relación</b>	<b>4</b>
<b>3. Decisiones tomadas</b>	<b>9</b>
3.1. Decisiones de diseño . . . . .	9
<b>4. Modelo Relacional</b>	<b>9</b>
<b>5. Restricciones</b>	<b>13</b>
<b>6. Implementación SQL</b>	<b>13</b>
6.1. Motor elegido . . . . .	13
6.2. Diseño de tablas . . . . .	13
6.3. Aclaraciones . . . . .	13
6.4. Funcionalidades implementadas . . . . .	13
<b>7. Conclusiones</b>	<b>15</b>

# 1. Introducción

El presente trabajo describe en detalle el diseño y la implementación de un sistema de persistencia de datos presentado sobre un problema representativo -aunque simplificado- del mundo real.

La metodología de diseño utilizada fue la Metodología de Diseño Lógico para Bases de Datos Relacionales (LRDM - Logical Relational Design Methodology), la cual tiene como primer paso la desambiguación de los requerimientos planteados en el problema y la construcción consecuente del Modelo Entidad Relación para el escenario en cuestión. La forma de representación utilizada para dicho modelo conceptual fue el Diagrama Entidad Relación.

En una etapa posterior, pasamos al diseño lógico de la solución transformando el conocimiento comprendido en el DER al modelo relacional, al partir del cual se procede a la implementación de la solución en un motor de base de datos relacional. En este caso, el motor elegido para la implementación concreta fue SQLite<sup>1</sup>.

## 1.1. Descripción del Problema

El problema para el cual se desarrolló un sistema de persistencia de datos consiste en un Mercado Virtual de intención similar a las comunidades de compra y venta online como MercadoLibre u OLX.

El sistema permite que sus usuarios publiquen artículos o servicios para ser comprados o contratados por otros usuarios. Los usuarios del sistema pueden ser particulares o empresas, teniendo iguales atribuciones y deberes a la hora de publicar, comprar o vender productos en el mercado. Los usuarios pueden ofrecer productos o servicios (o combinaciones de ambos) a un precio fijo o someter sus publicaciones a una subasta donde se establece un precio inicial y durante un tiempo determinado los demás usuarios pueden realizar ofertas con valores crecientes.

El mercado tiene una comisión por cada publicación realizada que deviene en una compra, pudiendo ser las publicaciones de distintos tipos y con diferentes valores de comisión o extensión del período de vigencia de las mismas. Todas las publicaciones permiten adicionalmente que los usuarios que las visitan realicen preguntas al vendedor, quien puede responderlas.

Las compras y ventas realizadas en el sitio siempre llevan una calificación obligatoria por parte de tanto el vendedor como el comprador, con una valoración numérica de 1 (no recomendable) a 10 (muy recomendable) así como un comentario opcional para el otro usuario involucrado en la transacción.

El sistema almacena adicionalmente el historial de compras de los usuarios, sus visitas a publicaciones, y aquellos productos o servicios que los usuarios hayan marcado como favoritos.

---

<sup>1</sup><https://www.sqlite.org/>

## 1.2. Funcionalidades a implementar

Como requerimientos adicionales a la persistencia, hay una serie de funcionalidades particulares que implementamos de forma tal de verificar el correcto funcionamiento de la base. Estas son:

- Consulta por usuario: obtener, para un usuario específico, información sobre los artículos que ha comprado y vendido, los artículos que ha visitado con su fecha de visita, los artículos que tiene en su lista de favoritos, y las primeras 3 categorías de artículos que visitó con mayor frecuencia en el último año.
- Consulta por categoría de producto: obtener, dada una categoría de producto (“Computación”, “Hogar, muebles y jardín”, etc), un listado de los vendedores que han publicado artículos de dicha categoría y la cantidad de ventas que efectuó cada uno de dichos vendedores.
- Función “Ofertar”: debe permitir al usuario ofertar una suma en una subasta. Dicha suma debe ser superior en al menos 1 peso, a la oferta actual, e inferior al doble de la oferta actual.
- Consulta por usuario y preguntas: obtener para un usuario específico, la lista de preguntas que ha realizado, con las respectivas respuestas que haya recibido (sólo la pregunta, si aún no recibió respuesta).
- Consulta por keyword: obtener, para un cierto keyword (por ejemplo “mesa”), la lista de publicaciones vigentes que tengan en el título, dicha keyword. El usuario debe poder restringir su búsqueda sólo a cierta categoría de artículos o servicios.
- Consulta por ganador/es anual de viaje a Khan El-Khalili: obtener, para un año específico, el ganador/es

## 2. Modelo de Entidad Relación

Como dijimos en la sección 1, para representar el Modelo de Entidad Relación realizado utilizamos el Diagrama Entidad Relación (DER). Presentado en las subsiguientes figuras, el DER fue dividido en diferentes secciones lógicas para facilitar su navegación, pero conceptualmente se trata de un único diagrama. Las divisiones constan de una sección central que incluye las interrelaciones más relevantes entre las entidades de Usuario, Publicación y Compra; y luego, una sección por cada una de estas tres entidades mostrando en detalle su composición y entidades relacionadas.

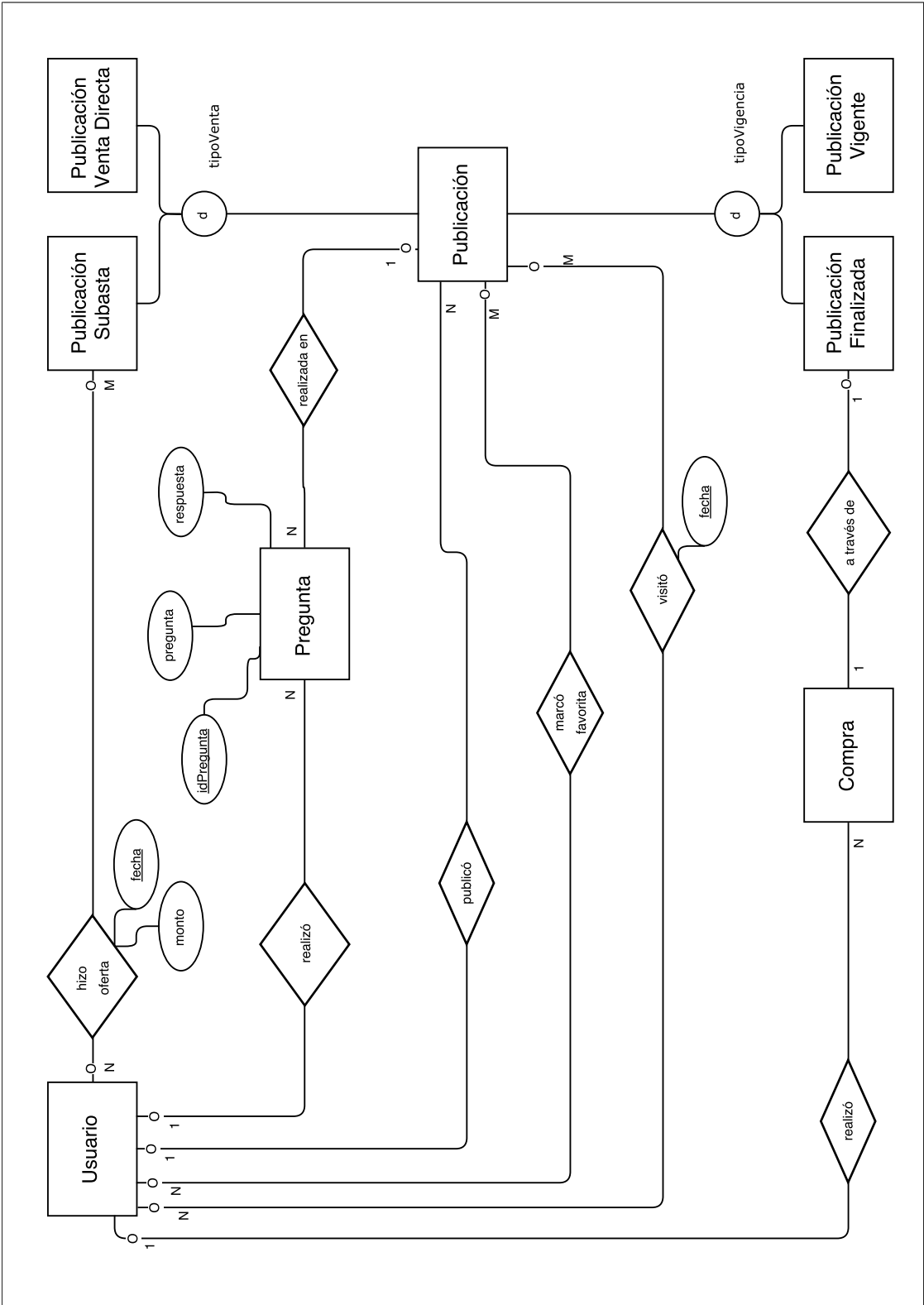


Figura 1: Diagrama Entidad Relación: Sección Central.

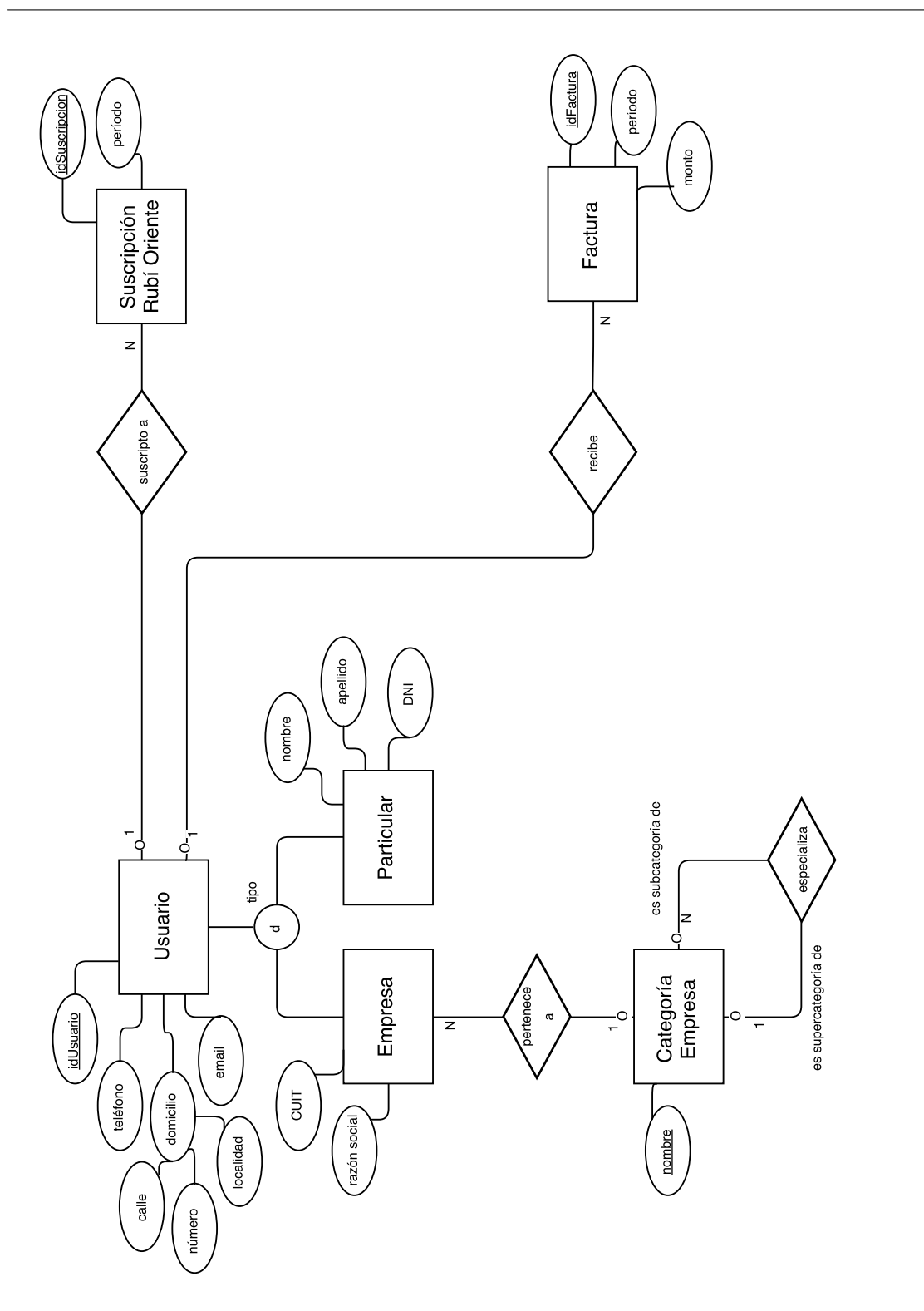


Figura 2: Diagrama Entidad Relación: Sección Usuario.

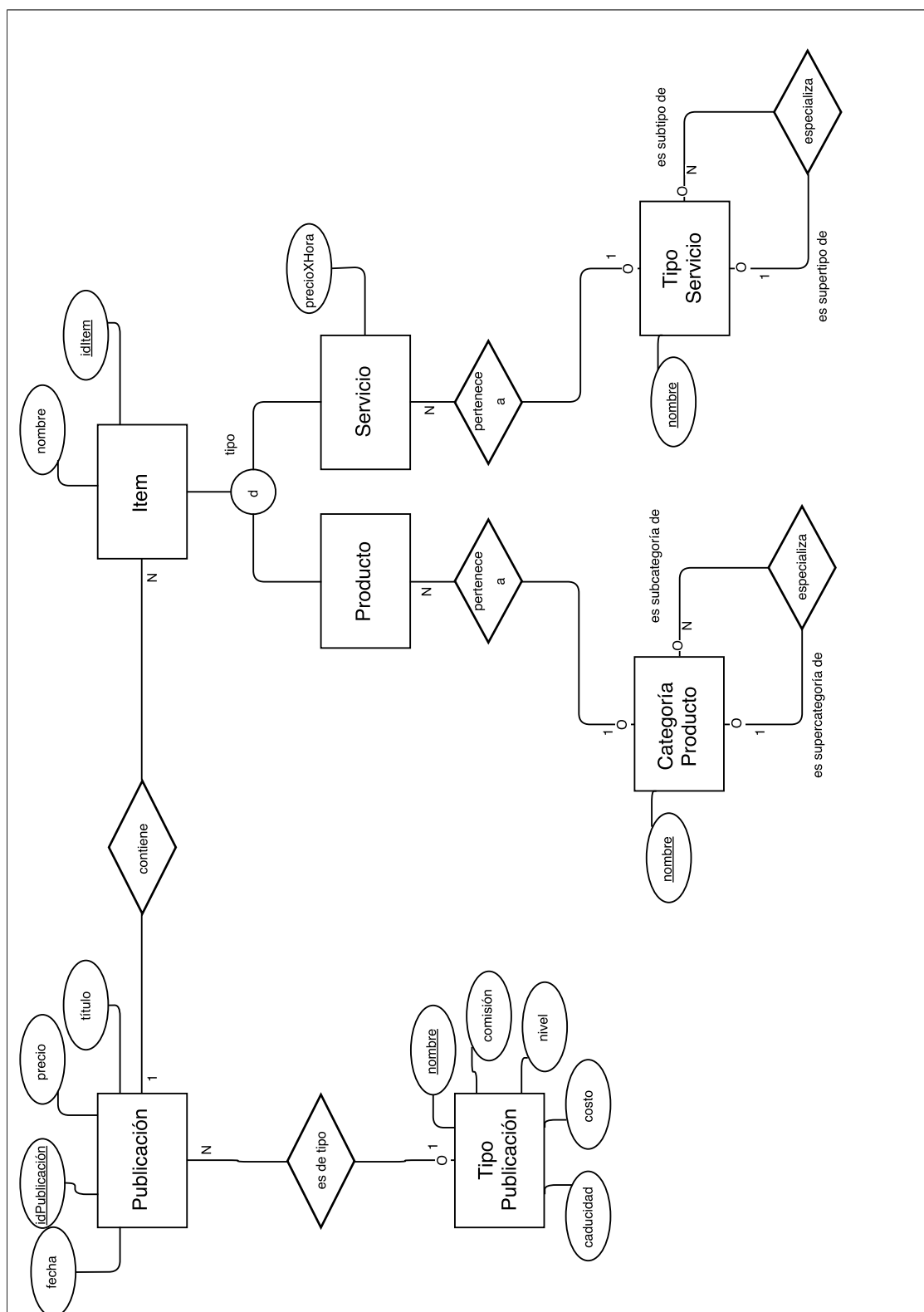


Figura 3: Diagrama Entidad Relación: Sección Publicación.

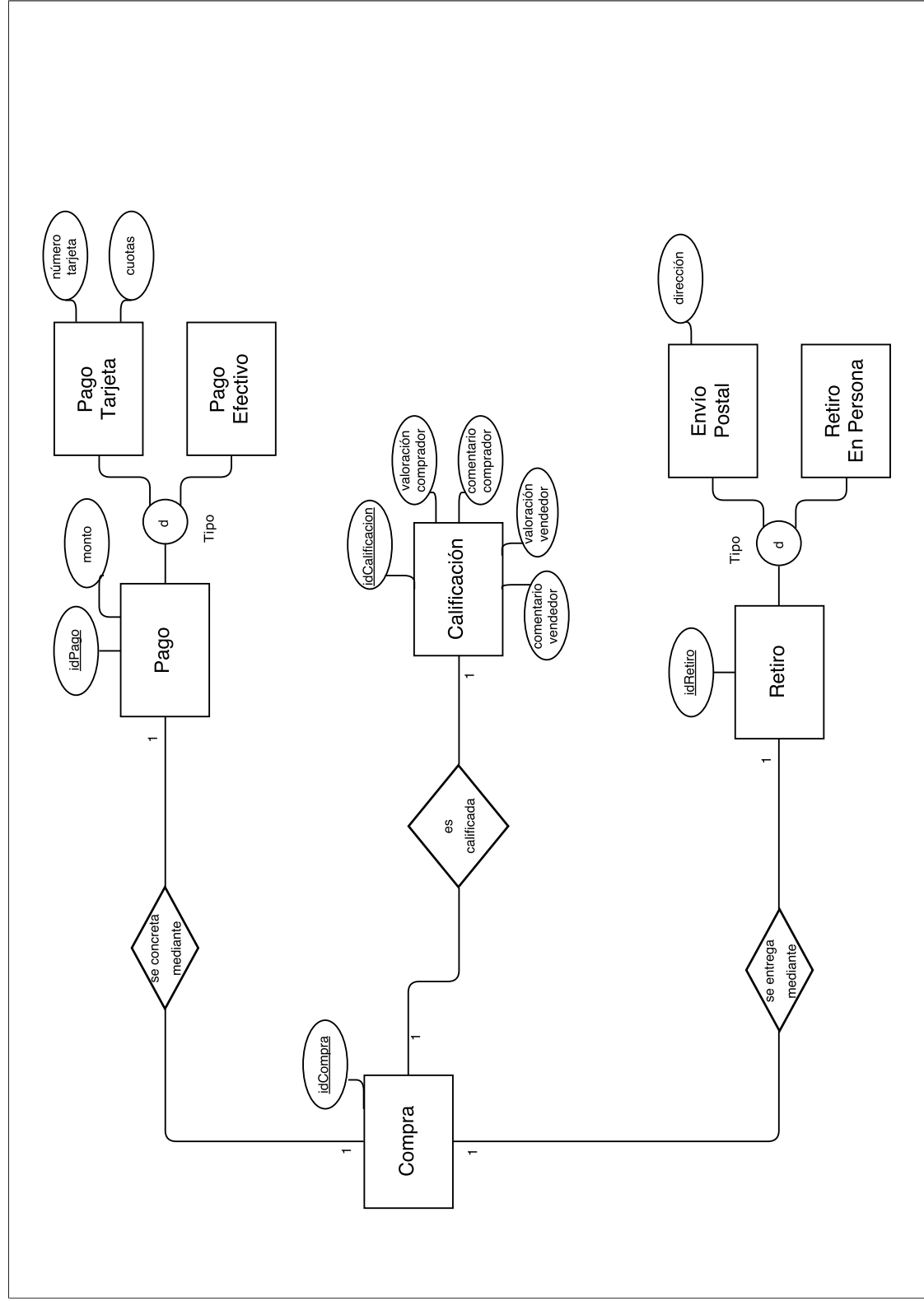


Figura 4: Diagrama Entidad Relación: Sección Compra.



### 3. Decisiones tomadas

#### 3.1. Decisiones de diseño

**Preguntas a una publicación** Consideramos que una pregunta realizada en una publicación puede tener una única respuesta, si el usuario desea replicar la respuesta debe hacerlo mediante una nueva pregunta.

### 4. Modelo Relacional

**Usuario** (idUsuario, calle, numero, localidad, telefono, email, tipo)

PK=CK={idUsuario}

FK={}

**Particular** (idUsuario, DNI, nombre, apellido)

PK={idUsuario}

CK={idUsuario, DNI}

FK={idUsuario}

Particular.idUsuario debe estar en Usuario.idUsuario

**Empresa** (idUsuario, CUIT, razonSocial, nombreCategoriaEmp)

PK={idUsuario}

CK={idUsuario, CUIT}

FK={idUsuario, nombreCategoriaEmp}

Empresa.idUsuario debe estar en Usuario.idUsuario

Empresa.nombreCategoriaEmp debe estar en CategoriaEmpresa.nombre

**CategoriaEmpresa** (nombre, nombreSubCategoria)

PK=CK={(nombre)}

FK={nombreSubCategoria}

CategoriaEmpresa.nombreSubCategoria puede ser nulo o debe estar en CategoriaEmpresa.nombre

CategoriaEmpresa.nombre puede no estar en CategoriaEmpresa.nombreSubCategoria

**SuscripcionRubiOriente** (idSuscripcion, periodo, idUsuario)

PK=CK={idSuscripcion}

FK={idUsuario}

SuscripcionRubiOriente.idUsuario debe estar en Usuario.idUsuario

Usuario.idUsuario puede no estar en SuscripcionRubiOriente.idUsuario

**Factura** (idFactura, periodo, monto, idUsuario)

PK=CK={idFactura}

FK={idUsuario}

Factura.idUsuario debe estar en Usuario.idUsuario

Usuario.idUsuario puede no estar en Factura.idUsuario

**Publicacion** (idPublicacion, titulo, fecha, precio, tipoPublicacion, tipoVigencia, tipo-

Venta, idUsuarioPublicador)

PK=CK={idPublicacion}

FK={tipoPublicacion, idUsuarioPublicador}

Publicacion.tipoPublicacion debe estar en TipoPublicacion.nombre

TipoPublicacion.nombre puede no estar en Publicacion.tipoPublicacion

**TipoPublicacion** (nombre, comision, costo, nivel, caducidad)

PK=CK={nombre}

FK={}

**Item** (idItem, idPublicacion, nombre, tipo)

PK=CK={idItem, idPublicacion}

FK={idPublicacion}

**Producto** (idItem, nombreCategoriaProd)

PK=CK={idItem}

FK={idItem, nombreCategoriaProd}

Producto.idItem debe estar en Item.idItem

Producto.nombreCategoriaProd debe estar en CategoriaProducto.nombre

CategoriaProducto.nombre puede no estar en Producto.nombreCategoriaProd

**Servicio** (idItem, precioXHora, nombreTipoServicio)

PK=CK={idItem}

FK={idItem, nombreTipoServicio}

Servicio.idItem debe estar en Item.idItem

Servicio.nombreTipoServicio debe estar en TipoServicio.nombre

TipoServicio.nombre puede no estar en Servicio.nombreTipoServicio

**CategoriaProducto** (nombre, nombreSuperCategoria)

PK=CK={nombre}

FK={nombreSuperCategoria}

CategoriaProducto.nombreSuperCategoria puede ser nulo o debe estar en CategoriaProducto.nombre

CategoriaProducto.nombre puede no estar en CategoriaProducto.nombreSuperCategoria

**TipoServicio** (nombre, nombreSuperTipo)

PK=CK={nombre}

FK={nombreSuperTipo}

TipoServicio.nombreSuperTipo puede ser nulo o debe estar en TipoServicio.nombre

TipoServicio.nombre puede no estar en TipoServicio.nombreSuperTipo

**Compra** (idCompra, idUsuario, idPublicacion, idPago, idCalificacion, idRetiro)

PK=CK={idCompra}

FK={idUsuario, idPublicacion, idPago, idCalificacion, idRetiro}

Compra.idUsuario debe estar en Usuario.idUsuario

Compra.idPublicacion debe estar en Publicacion.idPublicacion  
Compra.idPago debe estar en Pago.idPago  
Compra.idCalificacion debe estar en Calificacion.idCalificacion  
Compra.idRetiro debe estar en Retiro.idRetiro

**Pago** (idPago, monto, tipo)  
PK=CK={idPago}  
FK={}

**PagoTarjeta** (idPago, numeroTarjeta, cuotas)  
PK=CK={idPago}  
FK={idPago}  
PagoTarjeta.idPago debe estar en Pago.idPago

**Calificacion** (idCalificacion, valoracionComprador, valoracionVendedor, comentarioComprador, comentarioVendedor, idCompra)  
PK=CK={idCalificacion}  
FK={idCompra}  
Calificacion.idCompra debe estar en Compra.idCompra

**Retiro** (idRetiro, tipo, idCompra)  
PK=CK={idRetiro}  
FK={idCompra}  
Retiro.idCompra debe estar en Compra.idCompra

**EnvioPostal** (idRetiro, direccion)  
PK=CK={idRetiro}  
FK={idRetiro}  
EnvioPostal.idRetiro debe estar en Retiro.idRetiro

**HizoOferta** (idUsuario, idPublicacion, fecha, monto)  
PK=CK={(idUsuario, idPublicacion, fecha)}  
FK={idUsuario, idPublicacion}  
HizoOferta.idUsuario debe estar en Usuario.idUsuario  
HizoOferta.idPublicacion debe estar en Publicacion.idPublicacion

**Pregunta** (idPregunta, idUsuario, idPublicacion, pregunta, respuesta)  
PK=CK={idPregunta}  
FK={idUsuario, idPublicacion}  
Pregunta.idUsuario debe estar en Usuario.idUsuario  
Pregunta.idPublicacion debe estar en Publicacion.idPublicacion  
Usuario.idUsuario puede no estar en Pregunta.idUsuario  
Publicacion.idPublicacion puede no estar en Pregunta.idPublicacion

**MarcoFavorita** (idUsuario, idPublicacion)  
PK=CK={(idUsuario, idPublicacion)}

FK={idUserio, idPublicacion}

MarcoFavorita.idUsuario debe estar en Usuario.idUsuario

MarcoFavorita.idPublicacion debe estar en Publicacion.idPublicacion

**Visito** (idUserio, idPublicacion, fecha)

PK=CK={(idUserio, idPublicacion, fecha)}

FK={idUserio, idPublicacion}

Visito.idUsuario debe estar en Usuario.idUsuario

Visito.idPublicacion debe estar en Publicacion.idPublicacion

## 5. Restricciones

La base de datos presenta varias restricciones de lenguaje natural. Las siguientes restricciones no se encuentran modeladas en la base, y es responsabilidad de quien ingresa los datos asegurar que se cumplan.

- Bla.

## 6. Implementación SQL

### 6.1. Motor elegido

### 6.2. Diseño de tablas

### 6.3. Aclaraciones

A la hora de implementar la herencia disjunta en las entidades debimos colocar un identificador para saber de que tipo es.

Dicho identificador es un INTEGER, a continuación explicamos que implica cada número.

**Publicacion** En tipoVigencia, un 0 implica vigente y un 1 finalizada. Para el tipo de publicación, 0 implica onrmal, 1 subasta.

**Item** En la entidad Item, 0 implica que es un prooducto y 1 que es un servicio.

**Pago** Aquí un 0 implica pago al contado y 1 pago con tarjeta.

**Usuario** Un 0 en esta entidad significa particular y un 1 que es una empresa.

**Retiro** En cuanto a Retiros, un 0 implica retiro personal, en cambio, un 1 implica envío postal.

### 6.4. Funcionalidades implementadas

Las funcionalidades implementadas (SQL/stored procedures/triggers) son:

- Consulta por usuario: obtener, para un usuario específico, información sobre los artículos que ha comprado y vendido, los artículos que ha visitado con su fecha de visita, los artículos que tiene en su lista de favoritos, y las primeras 3 categorías de artículos que visitó con mayor frecuencia en el último año.

CREATE bla

- Consulta por categoría de producto: obtener, dada una categoría de producto (“Computación”, “Hogar, muebles y jardín”, etc), un listado de los vendedores que han publicado artículos de dicha categoría y la cantidad de ventas que efectuó cada uno de dichos vendedores.

CREATE bla

- Función “Ofertar”: debe permitir al usuario ofertar una suma en una subasta. Dicha suma debe ser superior en al menos 1 peso, a la oferta actual, e inferior al doble de la oferta actual.

CREATE bla

- Consulta por usuario y preguntas: obtener para un usuario específico, la lista de preguntas que ha realizado, con las respectivas respuestas que haya recibido (sólo la pregunta, si aún no recibió respuesta).

CREATE bla

- Consulta por keyword: obtener, para un cierto keyword (por ejemplo “mesa”), la lista de publicaciones vigentes que tengan en el título, dicha keyword. El usuario debe poder restringir su búsqueda sólo a cierta categoría de artículos o servicios.

CREATE bla

- Consulta por ganador/es anual de viaje a Khan El-Khalili: obtener, para un año específico, el ganador/es

## 7. Conclusiones