PECL3

BBDD

Miguel Ángel Losada Fernández – 53824672A

Sergio Sanz Sacristán – 03207618S

Juan Casado Ballesteros – 09108762A

Profesor: Sergio Caro
Laboratorio Martes 12:00 a 14:00

ÍNDICE

PECL3

ANÁLISIS DE LAS FORMAS NORMALES

Primera forma normal:

Todas nuestras relaciones están en primera forma normal ya que todos nuestros atributos son atómicos.

Para lograr la atomicidad de los atributos manteniendo la menor redundancia posible hemos descompuesto aquellos que eran multivariados, es decir, los hemos sacado a una tabla externa (teléfonos) que se relaciona con la tabla en la que estaba originalmente (empleado) a partir de la PK de esta (DNI). Esto lo hemos realizado cuando no conocíamos exactamente la cantidad de atributos multivaluados que iban a existir en cada instancia.

En el caso de si conocerlos y de ser fijos como ocurre con la Dirección del Empleado se los hemos incluido como atributos atómicos dentro de la relación.

Segunda forma normal:

Todas nuestras relaciones cumplen la segunda forma normal ya que cumplen la primera y además cumplen que todas las dependencias funcionales de cada tabla son totales, es decir que todos los atributos no primos de cada relación dependen funcionalmente de forma total de la PK de la relación.

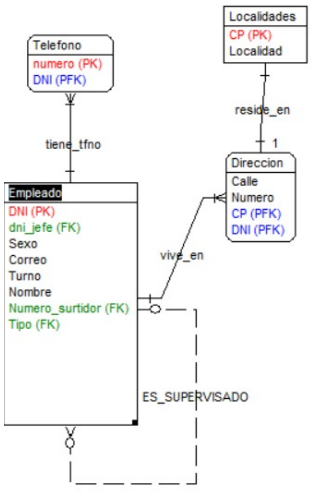
Para lograr esto hemos puesto como PK una fecha en relaciones como Canjea, Contiene, Reposta, Opinión de modo que todos los atributos de la relación dependan funcionalmente de forma total de dicha PK.

Tercera forma normal:

Todas nuestras relaciones cumplen la tercera forma normal ya que cumplen la segunda y además cumple que no existen atributos no primos de las relaciones que dependan transitivamente de algún atributo que no sea clave de la relación.

Los atributos no primos se relacionan funcionalmente con toda la clave y exclusivamente con toda la clave, para lograrlo hemos tenido que modificar al

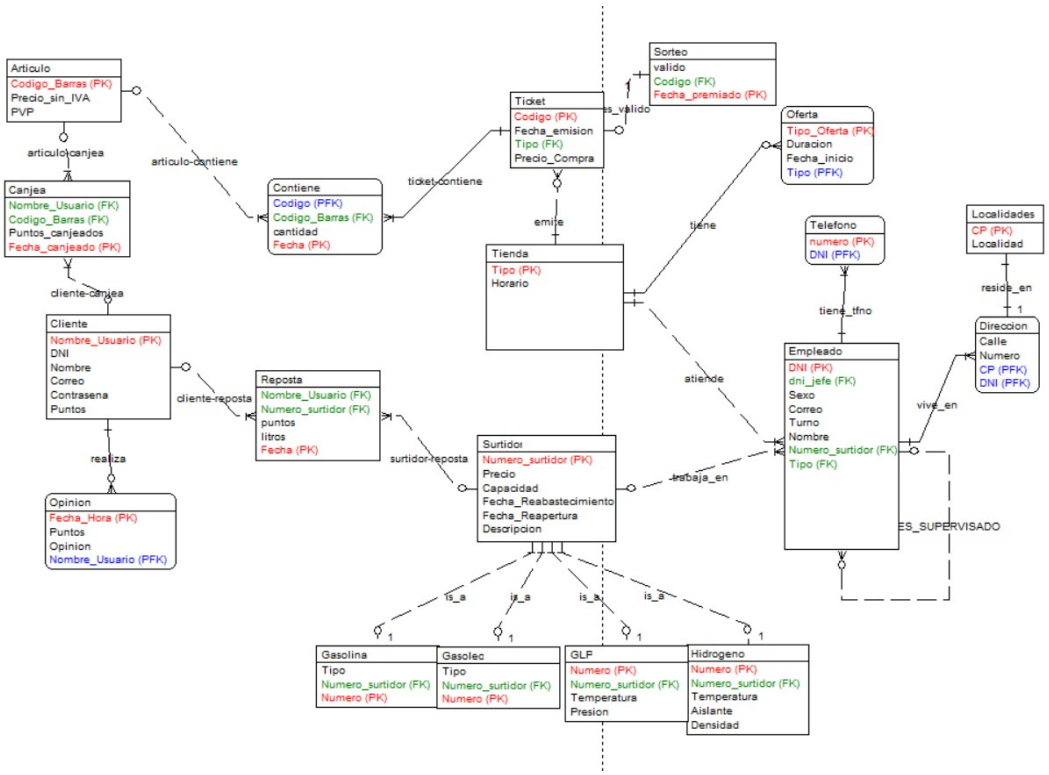
empleado debido a que el CP se relacionaba funcionalmente con la localidad y el resto de los atributos del original atributo compuesto dirección con DNI.



Hemos sacado a una nueva relación la localidad y el CP de forma que dirección tome como FK el CP de esa relación y el DNI sea la PFK de la relación dirección.

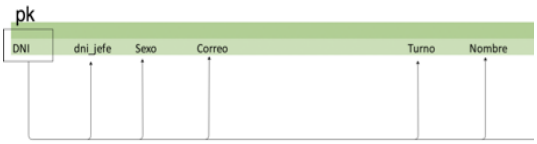
También hemos cambiado como se guardan los sorteos en la base de datos, ahora cuando un ticket es premiado es almacenada la fecha de su premio en una tabla Sorteo, esto se hace porque el booleano que controla si el ticket ha sido canjeado ya depende de esta fecha.

Finalmente, el esquema relacional en 3FN quedaría:



Ejemplos:

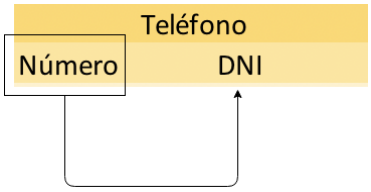
Mostramos las dependencias funcionales del Empleado como ejemplo de esquema en tercera forma normal.



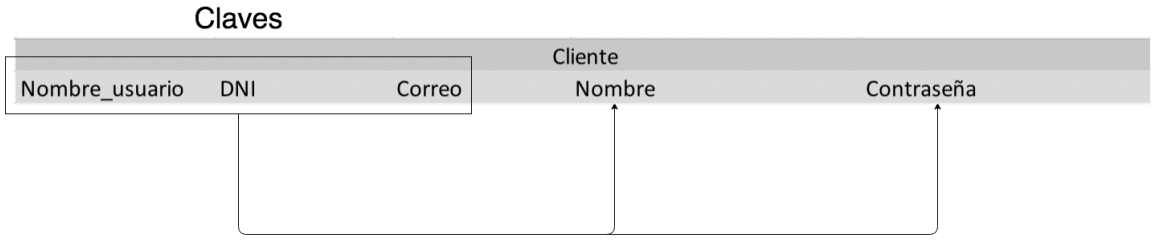
El resto de sus "atributos" estará en las tablas Dirección y Localidades para mantener la BBDD en 3FN.

Podemos ver como todos los atributos no primos se relacionan funcionalmente de forma exclusiva con la PK y nada más que con ellas dependiendo totalmente de ella.

Podemos ver como la dirección que podría considerarse como un atributo multivalorado se ha convertido en un conjunto de atributos atómicos dentro de la relación.



En esta otra mostramos cómo descompuesto Empleado para separarlo de sus teléfonos ya que es un atributo multivalorado del que no sabemos cuantos puede llegar a tener, así seguimos respetando las teorías de normalización.



Cliente es otra relación en la que se aprecia claramente como se respeta la tercera forma normal ya que todos los atributos no primos relacionan exclusivamente con todas las claves y exclusivamente con las claves.

DISPARADORES

Adicionalmente a los disparadores indicados en la práctica hemos hecho otros adicionales, los explicaremos todos a continuación:

Calcular caducidad de los artículos

Con este disparador ponemos a caducado todos los artículos cuya fecha sea 1 año menor que la fecha de la entrega de la práctica. Un artículo caducado es aquel que tiene en sorteo el booleano de válido a false.

- Trigger
- Llamada

Calcular el precio del ticket:

Este disparador salta cuando se crea o se modifica un ticket, buscará cada artículo y multiplicará su PVP por la cantidad de este, posteriormente sumará todos esos resultados.

- Trigger
- Llamada

Calcular los puntos de repostaje:

Este disparador tomará los litros y el precio de los mismos en el momento de ser creado para calcular los punto a punto el euro.

- Trigger
- Llamada

Control de empleados:

Este disparador se asegura de que un trabajador no pueda trabajar en un surtidor y una tienda a la vez.

- Trigger
- Llamada

Calcular PVP Artículo:

Cuando un artículo es insertado calculamos su PVP en función del IVA.

- Trigger
- Llamada

Calcular puntos del cliente:

Toma los puntos ganados al repostar y los gastados al canjear y le pone al empleado los puntos que tiene. Adicionalmente este disparador no dejará al empleado canjear puntos si no tiene suficientes.

Si modificamos los artículos:

- Trigger
- Llamada

Si insertamos artículos:

- Trigger
- Llamada

Si repostamos:

- Trigger
- Llamada

Si modificamos un repostaje:

- Trigger
- Llamada

Disparadores de dominio:

Hemos hecho unos cuantos disparadores para comprobar que ciertos atributos que no pueden ser negativos o que obligatoriamente tengan que ser mayor que 1. Esto debería haberse hecho creando dominios que así lo regularan, pero debido a que en esta práctica se pedían disparadores hemos decidido hacerlo así.

- En la carpeta ./Triggers/SQL_Triggers/ los que empiezan por comprobar o exception.

ROLES, GRUPOS Y USUARIOS

Grupo Administrador:

Este grupo debe poder ejecutar cualquier operación sobre la base de datos. Para ello seleccionamos Grant Wizard de la pestaña “Table” y seleccionamos todas las tablas. Posteriormente en Privileges seleccionamos todos los permisos.

Grupo Gestor:

Este grupo debe manejar los datos de la base de datos en su conjunto, pero no puede crear nuevas tablas ni elementos que afecten a la estructura de la base de datos. Para ello seleccionamos las opciones insert, select, update y delete en todas las tablas de la base de datos, y eliminamos el permiso create para así garantizar que no puede crear nuevas tablas.

Ejemplos:

```
Create table "prueba"
(
        "Input" Char(20) not null unique,
primary key ("Input")
) without oids;
```

```
ERROR:  permiso denegado al esquema public
LINE 1: Create table "prueba"
                     ^

********** Error **********

ERROR: permiso denegado al esquema public
```

Grupo Empleado:

Este grupo puede crear, actualizar y consultar tickets, repostajes y canjear artículos. Por ello, seleccionamos insert, select y update en las tablas Ticket, Reposta y Canjea. También puede dar de alta y consultar clientes, entonces seleccionamos insert y select en la tabla Cliente. Sin embargo, las opiniones de los clientes no se pueden ver. Así que, no añadimos ningún permiso a la tabla Opinion. Los surtidores se pueden consultar, pero solo se puede modificar si algún surtidor esta averiado. Por lo tanto, seleccionamos select en la tabla Surtidor, y en el atributo Fecha_Reapertura los permisos insert y update. También, podemos ver los detalles y modificarlos de un empleado excepto su DNI, el turno y el jefe asociado ni asociarse a un surtidor o una tienda.

Entonces, seleccionamos select en la tabla Empleado y update en todos los atributos excepto DNI, dni_jefe, Turno, Numero_surtidor y Tipo.

Ejemplos:

```
UPDATE public."Surtidor"
   SET "Numero_surtidor"=1.276
 WHERE "Numero_surtidor"=1;
```

```
ERROR:  permiso denegado a la relación Surtidor
********** Error **********

ERROR: permiso denegado a la relación Surtidor
SQL state: 42501
```

```
select *
from "Ticket", "Canjea", "Reposta"
```

	Codigo character(20)	Fecha_emision timestamp with time zone	Tipo character(50)	Precio_Compra double precision	Nombre_Usuario character(50)	Cod char
1	AAA0	1004-02-24 19:58:19+00:17:30	Panaderia	20	Mmmm gasss	iK1
2	AAA0	1004-02-24 19:58:19+00:17:30	Panaderia	20	Mmmm gasss	wef
3	AAA0	1004-02-24 19:58:19+00:17:30	Panaderia	20	Tu gasolinero	iK1
4	AAA0	1004-02-24 19:58:19+00:17:30	Panaderia	20	La gran manguera	kja
5	AAA0	1004-02-24 19:58:19+00:17:30	Panaderia	20	Mmmm gasss	iK1
6	AAA0	1004-02-24 19:58:19+00:17:30	Panaderia	20	El chuches	Akn
7	AAA0	1004-02-24 19:58:19+00:17:30	Panaderia	20	Jordi ENP	hsi
8	AAA0	1004-02-24 19:58:19+00:17:30	Panaderia	20	Tu gasolinero	hs8
9	AAA0	1004-02-24 19:58:19+00:17:30	Panaderia	20	La gran manguera	iK1
10	AAA0	1004-02-24 19:58:19+00:17:30	Panaderia	20	Mmmm gasss	LM8
11	AAA0	1004-02-24 19:58:19+00:17:30	Panaderia	20	El chuches	LM8

```
INSERT INTO public."Cliente"(
            "Nombre_Usuario", "DNI", "Nombre", "Correo", "Contrasena","Puntos")
    VALUES ('El chuchas', '44150782Y', 'Pepe Pepa', 'tupepaso@hotmail.es', 'Patatas_fritas',0);
```

```
Query returned successfully: one row affected, 26 msec execution time.
```

Grupo Supervisor:

Tiene control sobre toda la base de datos, excepto sobre los empleados, surtidores, tiendas, socios, artículos y tickets. Por lo tanto, añadimos los permisos insert, update y select a todas las tablas. y delete a todas las tablas excepto en las de Empleado, Surtidor, Tienda, Cliente, Articulo y Ticket. También este grupo solo puede consultar las Opiniones. Entonces, en la tabla Opinion solo seleccionamos el permiso select.

Ejemplos:

```
DELETE FROM public."Cliente"
 WHERE "DNI"='612447469R';
```

```
ERROR:  permiso denegado a la relación Cliente
********** Error **********

ERROR: permiso denegado a la relación Cliente
SQL state: 42501
```

```
select *
from "Opinion"
```

	Fecha_Hora timestamp with time zone	Puntos integer	Opinion character(100)	Nombre_Usuario character(50)
1	2001-05-10 17:40:45+02	10	ME SIRVIERON AGUA!!!	El chuches
2	2031-12-20 11:31:24+01	3	MASTODÓNTICA	Jordi ENP
3	1001-08-14 07:32:15+00:17:30	9	HORRIBLISIMA	La gran manguera
4	2201-09-01 01:58:23+02	4	SIGANME EN INSTA :)	La gran manguera

Grupo Cliente:

Este grupo puede crear, modificar y consultar opiniones, sin borrarlas, ver la lista de artículos y la de los canjeados. Entonces, seleccionamos insert, update y select en la tabla Opinion y select en las tablas Articulo y Canjea. Este grupo, además, puede consultar y realizar modificaciones en su perfil, solo nombre, correo y contraseña. Por lo tanto, seleccionamos select en la Tabla Cliente y, además, seleccionamos update en los atributos Nombre_usuario, Correo y contrasena.

Ejemplos:

```
UPDATE public."Opinion"
   SET "Opinion"='LA MEJOR EXPERIENCIA DE MI VIDA'
 WHERE "Puntos"=9;
```

```
Query returned successfully: one row affected, 23 msec execution time.
```

```
DELETE FROM public."Opinion"
 WHERE "Puntos"=9;
```

```
ERROR:  permiso denegado a la relación Opinion

********** Error **********

ERROR: permiso denegado a la relación Opinion
SQL state: 42501
```

```
UPDATE public."Cliente"
   SET "Puntos"=500000
 WHERE "DNI"='44150782Y';
```

```
ERROR:  permiso denegado a la relación Cliente

********** Error **********

ERROR: permiso denegado a la relación Cliente
SQL state: 42501
```

CASO PRÁCTICO JAVA

Hemos creado un programa JAVA desde el cual nos conectamos a la BBDD y podemos realizar bajo distintas cuentas de usuario consultas cuyo resultado luego mostramos.

Desde una clase ConectionManager nos conectamos a la BBDD con un usuario que se haya seleccionado previamente en la interface gráfica. Una vez seleccionado uno no hace falta volver a seleccionarlo para hacer otra consulta si no se desea cambiar de usuario.

Posteriormente mediante la clase Query utilizaremos la clase anterior para conectarnos y desconectarnos de la BBDD en cada consulta. Tendremos unas consultas modelo ya hechas (las de la PECL2) que podremos seleccionar y visualizar. Adicionalmente podremos hacer cualquier consulta personalizada.

En ConnectionManager creamos por tanto un objeto de tipo Connection a partir de un DriverManager al que le damos el usuario y la contraseña.

Para mostrar los resultados iteramos sobre cada fila de datos obtenida y dentro de la fila sobre cada atributo mostrándolos todos haya cuantos haya. Sobre un objeto Statement realizamos un executeQuery(String) del que obatenemos un ResultSet sobre el que iteramos con next(); del result set podemos obtener un ResultSetMetaData del que podemos saber la cantidad de columnas que va a contener el ResultSet y el nombre de las mismas así como otros datos de interés para el programador sobre el resultado de la consulta realiza.

Podemos comprobar como solo si tenemos permisos para hacer cada consulta podremos realizarla. De no tenerlos se nos denegará el acceso.

- Asker ((Gráficos) nos permite seleccionar las consultas)
- Logger ((Gráficos) nos permite seleccionar los usuarios)