Universidad Nacional Autónoma de México

Facultad de Ciencias

Proyecto Final: Traducción al Modelo Relacional

Flores Martínez Andrés
Vázquez Salcedo Eduardo Eder
Sánchez Pérez Pedro Juan Salvador
Concha Vázquez Miguel

Trabajo presentado en cumplimiento con la asignatura de Fundamentos de Bases de Datos impartida por el profesor
Gerardo Avilés Rosas
12 de enero de 2018

Índice

1. Traducción ER a Modelo Relacional

1.1. Algoritmo de Traducción

En el curso aprendimos cómo pasar el modelo entidad-relación al modelo relacional. Lo que se nos recomendó en primer lugar es identificar a las entidades fuertes y especificar sus atributos como parte de la tupla de la relación, subrayando sus atributos llave. Posteriormente, identificar las relaciones con una restricción de cardinalidad muchos-muchos porque estas de entrada crear nuevas relaciones (tablas). Para este tipo de relaciones, tendríamos que añadir los atributos llave de ambas relaciones (en caso de ser binaria) como llaves foráneas con tal de asegurar una integridad de relaciones.

En el caso de las relaciones de tipo uno-muchos, lo que se tendría que hacer era agregar la llave de la relación con cardinalidad *uno* a la otra entidad; si era una restricción de participación total de al menos un lado, entonces la podíamos representar a la relación misma con una nueva tabla que incluyera las llaves de cada entidad relacionada como llaves foráneas.

Para las relaciones con restricción de cardinalidad uno-uno había varios escenarios:

- Restricción de participación parcial de ambos lados: Se crea una tabla adicional con las laves de ambas entidades y los atributos de la relación.
- Restricción de participación total de un lado de la relación: Se incluyen en la tabla correspondiente a la entidad con participación total los atributos de la relación y la llave de la otra entidad.
- Restricción de participación total de ambos lados: Se incluyen en una sola relación todos los atributos de ambas entidades además de los atributos de la relación.

También se nos explicó que los atributos calculados (derivados) no los tendríamos que especificar como parte de las relaciones (tablas), mientras que en el caso de los atributos multivaluados necesitamos crear una nueva relación con el nombre del atributo y como atributo llave el mismo nombre, además de añadir el atributo llave de la entidad como llave foránea.

Por último, en los esquemas de especialización había también varios casos:

- Si la especialización es de completez total, entonces no debíamos agregar una relación para la super-entidad y trabajar directamente con las relaciones de las sub-entidades, agregando el atributo llave de la super-entidad como llave de las sub-entidades, al igual que sus demás atributos.
- Si la especialización es de completez parcial, entonces se tenía que crear una relación para cada una de las entidades, pero también "heredar"los atributos y la llave de la super-entidad a las relaciones de las sub-entidades.

Siguiendo este algoritmo y las recomendaciones del profesor Gerardo, consideramos en la siguiente subsección el proceso llevado a cabo para la traducción al modelo de datos relacional.

1.2. Proceso de Traducción

De acuerdo con el algoritmo, nos ocupamos primero de las entidades fuertes, para las cuales especificamos sus atributos como partes de la tupla que representa a cada relación y subrayamos en cada caso sus atributos llave. Como en nuestro caso no contamos con entidades débiles, lo hacemos para cada entidad del diagrama ER. Entonces nos queda lo siguiente — considerando de igual forma que los atributos multivaluados se transforman en tablas y no consideramos a aquellos que sean calculados (derivados) como se explicó anteriormente:

- **Sucursal**(idSucursal,calle,municipio,colonia,estado,numInterior,numExterior,CP,horaApertura,horaCierre)
- **SucursalTelefono**(idSucursal,telefono)
- **Historico**(idHistorico,idProducto,precioPrevio,precioNuevo,fechaActualizacion)
- **Proveedor**(RFC,razonSocial,calle,municipio,colonia,estado,numInterior,numExterior,CP,email,inicioRelacion,telefono)
- **Pedido**(numPedido,fechaPedido,promocion,preparado,entregado)
- **ProductoLeyenda**(idProducto,leyenda)
- **Ingrediente**(idIngrediente,nombre,marca,cantidadExistencia,fechaCaducidad)
- **Mobiliario**(idMueble,tipo)
- **Transporte**(idTransporte,marca,modelo,tipo)
- **Licencia**(código)

Luego, para traducir la jerarquía de especialización que se presenta con la super-entidad **Persona** y las sub-entidades **Empleado** y **Cliente**, notamos que tenemos una especialización total y restricción de disyunción, así que creamos una relación para cada subentidad, incluyendo tanto sus atributos como los de la super-entidad; la llave es la de la super-entidad. Nos quedan entonces:

- **Cliente**(taquiClave,email,telefono,nombre,apellidoPaterno,apellidoMaterno,calle,municipio,colonia,estado, numInterior,numExterior,CPnumPuntos,fechaPrimerVisita)
- **Empleado**(taquiClave,email,telefono,nombre,apellidoPaterno,apellidoMaterno,calle,municipio,colonia,estado, numInterior,numExterior,CP,CURP,RFC,tipo,tipoSangre,fechaNac,fechaContratacion,numEmergencia)

Notemos que en el caso de la especialización de la super-entidad **Producto** y su sub-entidad **Salsa**, al no haber propiamente una restricción de disyunción (podría verse como cualquiera según conviniera) y tener una completez parcial, fue necesario crear una relación para ambas entidades del diagrama entidad-relación como sigue:

- **Producto**(idProducto,puntoOtorgar,nombre,precio,descripcion)
- **Salsa**(idProducto,scoville,presentacion)

Un caso igual ocurre con la especialización sin restricción de disyunción y completez parcial de **Empleado** con su sub-entidad **TacoRider**. En consecuencia, consideramos la relación **Empleado** como quedó previamente y añadimos la relación para la sub-entidad con la misma llave:

- **TacoRider**(taquiClave,estaDisponible)

Ahora, procediendo a traducir las relaciones, consideramos inicialmente a la relación *tener* con restricción de cardinalidad uno-uno que es de participación total del lado del conjunto entidad **Licencia**, así que incluimos en esta relación los atributos de la relación *tener* (que no hay) y la llave de la otra entidad participante, que en este caso sería el atributo *taquiClave* de la sub-entidad **TacoRider**. Tenemos así una modificación que nos queda:

- **Licencia**(codigo,taquiClave)

La relación *llevar* es de cardinalidad uno-muchos, pero total del lado de la entidad **TacoRider**, así que lo que procedemos a hacer es crear una nueva relación **Llevar** que tiene como atributos a las llaves de **TacoRider** y de **Pedido**. Nos queda entonces:

- **Llevar**(numPedido,taquiClave)

En cuanto a la relación *ordenar* entre las entidades **Cliente** y **Pedido**, al contar con una restricción de cardinalidad uno-muchos, agregamos del lado de la relación correspondiente a los pedidos la llave del cliente que sería la *taquiClave*, también agregando el atributo de relación *metodoPago*; la actualización de la relación respectiva queda como sigue:

- **Pedido**(numPedido,fechaPedido,promocion,preparado,entregado,taquiClave,metodoPago)

La relación *ocurrir* ocasiona también una actualización de la relación **Pedido** porque es de cardinalidad uno-muchos y con participación total del lado de **Pedido**. En consecuencia, agregamos los atributos de la relación (que no hay) y el atributo llave de **Sucursal** que es *idSucursal* y nos queda:

- **Pedido**(numPedido,fechaPedido,promocion,preparado,entregado,taquiClave,metodoPago,idSucursal)

La relación *poseer* entre **TacoRider** y **Transporte** es de cardinalidad muchos-muchos, así que creamos una nueva relación (tabla) **Poseer** con las llaves de ambas entidades que queda así:

- **Poseer**(taquiClave,idTransporte)

Las relaciones entre **Proveedor** y **Mobiliario** e **Ingrediente**, que son respectivamente *proveerMob* y *proveerIng*, poseen una restricción de cardinalidad muchos-muchos, así que las relaciones se transforman en tablas que tienen como atributos los atributos de las relaciones (en este caso el precio para cada una) y los atributos llave de las dos entidades involucradas. Nos queda así:

- **ProveerMob**(RFC,idMueble,precio)
- **ProveerIng**(RFC,idIngrediente,precio)

De igual manera, la relación *contener* entre **Producto** y **Pedido**, así como la relación *tener* entre **Producto** e **Ingrediente** tienen cardinalidad muchos-muchos, por lo que se transforman en nuevas relaciones (tablas) con las llaves de cada entidad involucrada como atributos, así como también con los atributos de la relación. En el caso de la relación *contener*, se incluyen en la tabla el atributo *cantidad* y los atributos *numPedido* y *idProducto*. En el caso de la relación *Tener*, incluimos el atributo *cantidad* y los atributos *idProducto* y *idIngrediente*. Nos queda lo siguiente:

- **Contener**(numPedido,idProducto,cantidad)
- **Tener**(idProducto,idIngrediente,cantidad)

La relación *conservar* entre **Producto** e **Histórico** es de cardinalidad nuevamente muchos-muchos, por lo que creamos una nueva relación (tabla) **Conservar** con los atributos de la relación (no hay) y las llaves de las dos entidades involucradas. Nos queda como sigue:

- **Conservar**(idHistorico,idProducto)

También la relación *recomendar* es cardinalidad muchos-muchos, así que agregamos los atributos de la relación (no hay) y las llaves de las entidades involucradas **Producto** y su sub-entidad **Salsa** como atributos de la tabla creada. Como en cambos casos se trata del atributo llave *idProducto*, hacemos una distinción en el nombre del mismo y nos queda:

- **Recomendar**(idProducto,idProductoSalsa)

Ahora, la relación *trabajar* entre **Empleado** y **Sucursal** es de cardinalidad uno-muchos, por lo que procedemos a agregar los atributos de la relación (*salario*), así como también la llave de la entidad **Sucursal** en la relación correspondiente a la entidad **Empleado** y nos queda lo siguiente:

- **Empleado**(taquiClave,email,telefono,nombre,apellidoPaterno,apellidoMaterno,calle,municipio,colonia,estado, numInterior,numExterior,CP,CURP,RFC,tipo,tipoSangre,fechaNac,fechaContratacion, numEmergencia,salario,idSucursal)

Para la relación *supervisar* binaria, pero con respecto a la misma entidad **Empleado**, lo que hicimos fue seguir los ejemplos estudiados a lo largo del curso y creamos una nueva relación **Supervisar** que tiene como atributos a los dos identificadores correspondientes al jefe y al subordinado y nos quedó —luego de un renombre del atributo para hacer la distinción necesaria:

- **Supervisar**(taquiClaveGerente,taquiClaveSupervisado)

Luego, para poder transformar la entidad débil de **Categoría**, lo que hicimos fue notar la relación binaria en la que toma papel con el **Producto**. Tomando la llave de la entidad de la cual depende, que es *idProducto* del **Producto**, además de los atributos de la entidad débil —en este caso solamente su discriminante de *taquegoría*— creamos una nueva relación (tabla) con este información y nos queda:

- **Categoría**(idProducto,taquegoria)

Notamos, llegados a este punto, que no era necesario hacer la traducción de la relación de *pertenecer* dado que lo que propiciaría sería agregar el discriminante de **Categoría** (*taquegoría*) en la relación de **Producto**, pero esta información que empareja a los productos con sus categorías ya fue solucionado por la traducción previamente descrita. No quisimos agregar redundancia a nuestro modelo.

Finalmente, en el caso de la relación *dirigir* entre **Empleado** y **Sucursal** optamos por crear una nueva relación (tabla) con el mismo nombre, los atributos llave de ambas entidades fuertes y el atributo de la relación: *fechaInicio*. Esto se debió a que la cardinalidad es uno-muchos y con una restricción de participación total del lado de la entidad **Sucursal**. Obtuvimos lo siguiente:

- **Dirigir**(taquiClave,idSucursal,fechaInicio)

1.3. Relaciones Generadas

A raíz de lo expuesto a priori, obtuvimos las siguientes relaciones (tablas) que expresamos como tuplas y se encuentran ahora ordenadas alfabéticamente:

- **Categoría**(idProducto,taquegoria)
- **Cliente**(taquiClave,email,telefono,nombre,apellidoPaterno,apellidoMaterno,calle,municipio,colonia,estado, numInterior,numExterior,CPnumPuntos,fechaPrimerVisita)
- **Conservar**(idHistorico,idProducto)
- **Contener**(numPedido,idProducto,cantidad)
- **Dirigir**(taquiClave,idSucursal,fechaInicio)
- **Empleado**(taquiClave,email,telefono,nombre,apellidoPaterno,apellidoMaterno,calle,municipio,colonia,estado, numInterior,numExterior,CP,CURP,RFC,tipo,tipoSangre,fechaNac,fechaContratacion, numEmergencia,salario,idSucursal)
- **Historico**(idHistorico,idProducto,precioPrevio,precioNuevo,fechaActualizacion)
- **Ingrediente**(idIngrediente,nombre,marca,cantidadExistencia,fechaCaducidad)
- **Licencia**(codigo,taquiClave)
- **Llevar**(numPedido,taquiClave)
- **Mobiliario**(idMueble,tipo)

- **Pedido**(numPedido,fechaPedido,promocion,preparado,entregado,taquiClave,metodoPago,idSucursal)
- **Poseer**(taquiClave,idTransporte)
- **Producto**(idProducto,puntoOtorgar,nombre,precio,descripcion)
- **ProductoLeyenda**(idProducto,leyenda)
- **Proveedor**(RFC,razonSocial,calle,municipio,colonia,estado,numInterior,numExterior,CP,email,inicioRelacion,telefono)
- **ProveerIng**(RFC,idIngrediente,precio)
- **ProveerMob**(RFC,idMueble,precio)
- **Recomendar**(idProducto,idProductoSalsa)
- **Salsa**(idProducto,scoville,presentacion)
- **Sucursal**(idSucursal,calle,municipio,colonia,estado,numInterior,numExterior,CP,horaApertura,horaCierre)
- **SucursalTelefono**(idSucursal,telefono)
- **Supervisar**(taquiClaveGerente,taquiClaveSupervisado)
- **TacoRider**(taquiClave,estaDisponible)
- **Tener**(idProducto,idIngrediente,cantidad)
- **Transporte**(idTransporte,marca,modelo,tipo)

2. Diagrama de Clases UML

Los diagramas de clase UML (*Unified Modelling Language*) resultan sumamente útiles como un medio de poder transmitir la estructura de los datos que deberán ser almacenados en la base de datos, así como su significado, relaciones con otros datos y sus restricciones a partir de las restricciones mismas de las relaciones y los dominios (conjuntos de valores de un tipo) sobre los que los tipos de datos pueden tomar sus valores.

A partir de la representación en diagramas UML o en forma de tuplas, se tiene el equivalente del diagrama entidad-relación al modelo relacional, listo para poder comenzar a estructurar la base de datos por medio de las sentencias del lenguaje de definición de datos DDL (Data Definition Language), procurando la integridad entre las tablas.

A continuación se muestra el diagrama de clases UML obtenido:

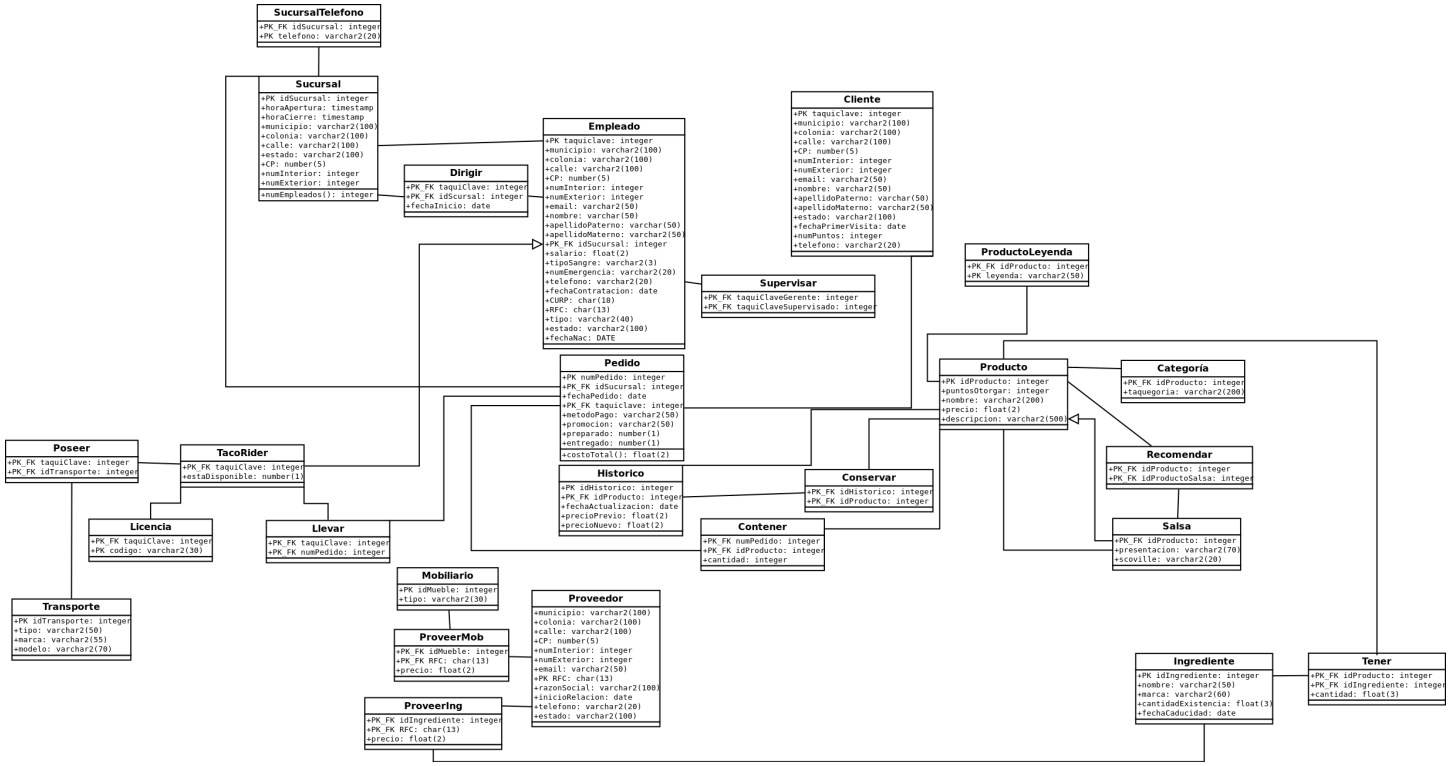

Figura 1: Modelo de datos relacional obtenido a partir de la traducción del diagrama entidad-relación (ER) siguiendo el algoritmo estudiado en el curso.

Referencias

Modelo Relacional, Avilés Rosas Gerardo. UNAM, Facultad de Ciencias, págs. 1-29.