8. Construcción de Bases de Datos

8.1 Implantación de un Modelo Estático de OO en un Modelo Relacional

Cuando se construye un sistema de administración de información, es necesario almacenar los datos sobre bases de datos. Las bases de datos son depósitos de datos que pueden ser administrados por las aplicaciones, o bien pueden ser administrados a través de Sistemas Administradores de Bases de datos. Los sistemas administradores de bases de datos, tienen como propósito principal el almacenamiento de la información que permita una acceso a la misma bajo mecanismos bien establecidos.

Los mecanismos de acceso y manipulación de información se definen en base a modelos de datos. Los modelos de datos son herramientas conceptuales que permiten describir a los objetos de información bajo una notación, estableciendo una serie de operadores para manipular a los objetos de información.

En la actualidad el modelo de datos sobre el que la mayor parte de los sistemas administradores de bases de datos se basan es el Modelo Relacional. Desgraciadamente existen muy pocos sistemas administradores de bases de datos que se basen sobre el modelo de Orientación a Objetos, lo que hace necesario establecer mecanismos de conversión entre los dos modelos cuando se desarrollan sistemas sobre el modelo de orientación a objetos que estará utilizando sistemas de bases de datos basados en el modelo relacional. Esta situación es la mas frecuente en la actualidad cuando se desarrollan aplicaciones de información de administración.

El modelo relacional describe a los objetos de información a través de relaciones que son en una forma intuitiva, tablas que están compuestas por renglones y columnas; cada renglón es un objeto de información y cada columna es un atributo de los objetos de información de la tabla. Las operaciones básicas sobre las tablas consideran la consulta, agregar un renglón a la tabla, eliminar un renglón de la tabla y modificar un renglón a la tabla.

En sí la conversión entre modelos se da para cada concepto del modelo de orientación a objetos, en algunos conceptos no hay aplicación directa de ningún tipo de conversión. En las siguientes secciones se mostrarán las distintas formas de conversión de cada uno de los conceptos.

8.2 Conversión de Clases y Objetos relacionados

La conversión más simple consiste en hacer corresponder a Clases de objetos con relaciones. En este caso, debe conceptualizarse a las clases ya no como plantillas de objetos, sino como conjuntos de objetos.

Cada clase debe corresponder a una relación y cada propiedad de la clase debe corresponder a un atributo o columna de la relación. Cada propiedad-atributo debe tener asociado un tipo de datos. Cuando los datos no correspondan a tipos simples que puedan implantarse con los tipos disponibles en un manejador de base de datos relacional, la relación no estará en 1ª forma normal, lo cual obligará a aplicar el proceso de normalización. El proceso de normalización asegura que toda relación sea implantable en un manejador de base de datos y que no permite anomalías de inserción, modificación y eliminación. Para una descripción detallada del modelo relacional y del proceso de normalización puede referirse a la abundante literatura existente tal como [Silberschatz98].

Suponga que se tiene la clase que se muestra en la Figura 8.1

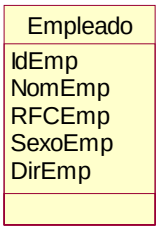

Figura 8.1 Clase persistente a convertir a relación.

Esta clase genera todos los objetos persistentes Empleado de tal forma que se tiene que hacer una relación que corresponda a esta clase. La relación resultantes es:

Empleado(IdEmp, NomEmp, RFCEmp, SexoEmp, DirEmp)

Note que esta conversión es muy simple, pues sólo es necesario cambiar el formatode UML a notación relacional. En la notación relacional es necesario destacar los atributos que serán la llave relacional de la relación. En este caso la llave relacional está formada por un solo atributo que es IdEmp, pues con el valor de esta columna se puede identificar de manera única un renglón (un objeto) dentro de la tabla.

Una vez que se tienen los atributos es necesario aplicar el proceso de normalización a la relación resultante. En principio, si se hizo un buen modelo de objetos, el proceso de normalización sólo servirá para identificar las llaves de las relaciones resultantes y comprobar que el modelo es un modelo robusto que no tiene duplicaciones indeseables. Como primer paso se realizará la revisión de 1ª forma normal que asegura que todos los atributos están definidos sobre tipos de datos simples tales como: números, cadenas, fechas, booleanos, imágenes, textos y cualquiera que permite un manejador de bases de datos relacional. Los tipos de datos que no son aceptados para un atributo son por lo general: arreglos, registros, conjuntos.

Los siguientes pasos de normalización corresponden a eliminar dependencias indeseables entre las llaves de la relación y los demás atributos, estas dependencias son parciales y transitivas que por lo regular mantienen un nivel de duplicación de información significativo lo que obliga a dobles capturas, dobles eliminaciones y dobles modificaciones. En el punto en que una relación no tiene estas dependencias indeseables se dice que la relación está en 3ª forma normal, que por lo general genera esquemas de relación con comportamientos muy adecuados.

8.3 Conversión de Métodos

En este punto se tienen clases que corresponden a relaciones de una forma totalmente pasiva, es decir, no hay operaciones ligadas a las relaciones de la forma en que los métodos se relacionan con la clase a la que pertenecen.

La funcionalidad en los manejadores de bases de datos relacionales comerciales se dan a través de mecanismos tales como los Triggers, Procedimientos Almacenados, Revisiones de Integridad y otras variantes, pero no con métodos. Por esta razón, es necesario distinguir el lugar en donde estará la funcionalidad de cada objeto persistente. Para esto existen dos enfoques:

8.3.1 Funcionalidad de objetos dentro de los programas

En este caso los objetos de una aplicación se tienen en dos versiones: persistente y dinámica. La versión persistente sólo almacena las propiedades del objeto y no realiza ninguna operación adicional a la de consulta, inserción, modificación y eliminación. La versión dinámica implanta todos los métodos obtenidos en el análisis y diseño, de tl forma que tienen todo el conocimiento de negocio asociado a ese objeto. Por lo regular se implantan como objetos de negocio.

Este esquema permite que las aplicaciones no dependan de la plataforma de bases de datos pues utilizan unicamente los servicios básicos de los administradores de bases de datos.

Los inconvenientes pueden ser que los procesos cliente estén con mayor carga de trabajo quee el servidor de base de datos. Por otro lado, dejan la base de datos vulnerable a alteraciones con los programas de acceso directo a la base de datos con los que cuenta cada sistema administrador de bases de datos. Sin embargo, el inconveniente más importante es en la distancia entre las propiedades y los métodos de los objetos pues es necesario cuidar la adecuada sincronía de estas dos partes de los objetos.

En esta opción se genera una clase dinámica que implanta los métodos de la clase en cuestión haciendo todas las solicitudes de datos a objetos de datos. Esta clase es una clase que genera objetos de negocio.

8.3.2 Funcionalidad de objetos dentro del sistema administrador de bases de datos

Cuando los objetos se impantan por completo dentro de un administrador de bases de datos, los métodos se implantan a través de triggers, procedimientos almacenados y revisiones de integridad. Esto se da de tal manera que los objetos de negocio simplemente

son objetos simplen que actúan de intermediarios entre los objetos interfaz y los objetos de datos en la base de datos, recorriéndose las funciones de negocio a la base de datos.

Este esquema es muy popular pues asegura que la información siempre esté vinculada a los métodos asociados y sus reglas de integridad dados por el negocio y aunque se intente acceder a la información por otros accessos, no es posible acceder directamente a la información con la ganancia en preservación de integridad. En este caso el servidor de bases de datos ejecuta una gran proporción de procesos de la aplicación, permitiendo que los procesos en la porción cliente de la aplicación sean más ligeros.

El inconveniente de esta opción está dada porque se incrementa la dependencia con el administrador de la base de datos y no es posible hacer una migración simple de la aplicación a otra plataforma.

En este caso deben implantarse cada método propuesto en el modelo como un procedimiento almacenado, que sea activado (cuando se considere que existe un vínculo temporal) a través de un Trigger asociado a la relación que corresponde a la clase.

8.4 Conversión de Asociaciones Estáticas

Hasta este punto sólo se han convertido clases aisladas de otras, sin embargo, los modelos estáticos de objetos consideran que los objetos están vinculados con otros objetos de manera persistente. Existen varias formas de vincular objetos persistentes de distintas clases, por lo que es necesario describir cada caso de una forma particular. En las siguientes secciones se describen el método de conversión de cada tipo de asociación estática.

En general el procedimiento es muy simple:

- Las clases en los extremos de la asociación se convierten a relaciones de la forma en que se muestra en la sección 8.2 salvo que si los objetos de la clase sólo se pueden determinar a través de otro objeto de la asociación, será necesario agregarle los atributos llave de la clase que lo identifica.
- Las asociaciones se convertirán a relaciones que tendrán como atributos, los atributos de las clases asociativas que tengan y los atributos llave de todas las clases a las que estén vinculando.

A continuación se describirán las distintas situaciones que se tienen durante la conversión.

8.4.1 Conversión de Asociaciones Binarias uno a uno

La asociación más simple es la de uno a uno. Una asociación binaria uno a uno describe una situación en la cual dos objetos están relacionados y por la existencia de uno se da la existencia de otro. Por lo regular se dan cuando se han modelado los atributos de una clase, como una clase. Un ejemplo se puede observar en la Figura 8.2

En este ejemplo, se observa claramente que aunque la clase DirTrabajador genera objetos aparentemente independientes, es claro que cada objeto existirá porque existe un objeto Trabajador. Es claro que en este caso este modelo de objetos debe generar una sola relación:

Trabajador (IdEmp, NomEmp, RFCEmp, SexoEmp, Calle, Número, Colonia, Ciudad, CódigoPostal, País)

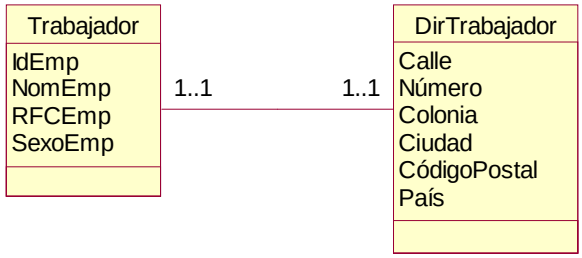

Figura 8.2 Asociación binaria uno a uno.

Es claro que también puede corregirse el modelo de objetos original para obtener una sola clase que después puede ser convertida a relación tal y como se describió en la sección anterior.

Aunque este tipo de relación es muy manejable existen ocaciones en que no es tan trivial realizar al conversión. En la Figura 8.3 se muestra un caso en el que se tiene una asociación uno a uno, pero existe un aspecto opcional. Un trabajador está asociado a un solo cónyuge, pero existen trabajadores solteros que no deben tener asociado ningún conyuge.

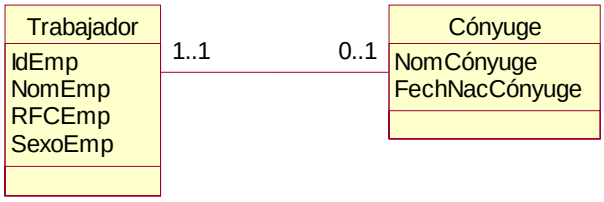

Figura 8.2 Asociación binaria uno a uno con opcionalidad.

En este caso se pueden combinar las dos clases en una, o bien generar una sola relación a partir de este modelo. La relación resultante es:

Trabajador (IdEmp, NomEmp, RFCEmp, SexoEmp, NomCónyuge, FechNacCónyuge)

Pero esta relación tendrá valores nulos para todos los trabajadores que sean solteros, por lo que existiran posibilidades de errores, así como el desperdicio de espacio.

Otra opción es generar tres relaciones: una para la clase Trabajador otra para la clase Cónyuge y otra más para la asociación.

En este caso se tendrán la relaciones:

Trabajador (IdEmp, NomEmp, RFCEmp, SexoEmp),

Cónyuge (NomCónyuge, FechNacCónyuge)

Asociación (??????)

El problema que aparece es qué atributos tendrá la relación de la asociación entre Trabajador y Cónyuge. Los atributos de la relación que corresponde a la asociación

estática deben ser los más significativos de las entidades que asocia. En este caso para Trabajador es el atributo IdEmp, que se convirtió en la llave de la relación Trabajador. Sin embargo, la Clase Cónyuge está determinada por el trabajador, de tal forma que no tiene un atributo que la identifique de manera única, por lo que no aporta ningún atributo a la relación. De hecho, Cónyuge debe tener un atributo que lo vincule a Trabajador y éste debe ser la llave de Trabajador, o sea IdEmp, de tal suerte que el modelo queda:

Trabajador (IdEmp, NomEmp, RFCEmp, SexoEmp),

Cónyuge (IdEmp <<por Trabajador>>, NomCónyuge, FechNacCónyuge)

Asociación (IdEmp <<del Trabajador>>, IdEmp <<del Cónyuge>>)

Para la asociación los dos IdEmp, son idénticos de tal suerte que se puede dejar sólo uno. Si tomamos el número de registros que tenga cada relación tendremos que

Trabajador: N Registros

Cónyuge: M Registros con N > M (no todos los trabajadores tienen cónyuge)

Asociación: M Registros (sólo existen vínculos para los trabajadores con cónyuge)

N: Número total de trabajadores

M. Número de trabajadores con cónyuge

De los datos anteriores es claro que pueden unirse las tablas Cónyuge y Asociación para dejar una sola que va a tener la información adicional de los trabajadores con cónyuge. El modelo queda de la siguiente manera:

Trabajador (IdEmp, NomEmp, RFCEmp, SexoEmp),

Cónyuge (IdEmp <<por Trabajador>>, IdEmp <<por Asociación>>,
NomCónyuge, FechNacCónyuge)

Pero como los dos IdEmp tienen el mismo valor puede eliminarse a cualquiera y queda finalmente de la siguiente forma:

Trabajador (IdEmp, NomEmp, RFCEmp, SexoEmp),

Cónyuge (IdEmp, NomCónyuge, FechNacCónyuge)

El esquema resultante puede combinarse en una sóla relación, de la siguiente forma como se mostró al inicio de esta sección:

Trabajador (IdEmp, NomEmp, RFCEmp, SexoEmp, NomCónyuge,
FechNacCónyuge)

Sin embargo, se tendrá un desperdicio de espacio y existe la posibilidad de capturar datos de cónyuge a trabajadores que no lo tienen.

Entonces ¿cuál es la solución correcta cuando se tiene opcionalidad?. Debe seleccionarse la primera opción cuando el número de trabajadores sin cónyuge sea mayor del 60% del total de trabajadores.

8.4.2 Conversión de Asociaciones Binarias uno a muchos

Cuando se tienen asociaciones de uno a muchos la conversión obliga a generar más de una relación. Suponga que se tiene el esquema de la Figura 8.3. El esquema muestra la asociación entre un Empleado y el Departamento en donde está asignado. El modelo muestra un instante del tiempo pues el esquema no preserva la historia de las asignaciones del empleado a otros departamentos a los que haya pertenecido.

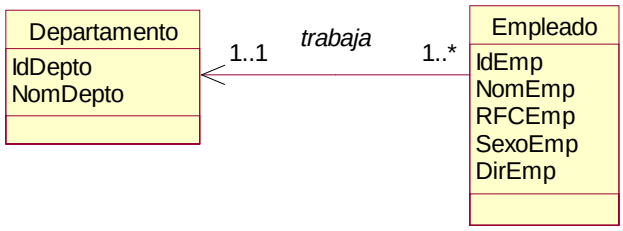

Figura 8.3 Asociación binaria uno a muchos.

En este caso tendremos que obtener las relaciones siguientes:

Departamento (IdDepto, NomDepto),

Empleado (IdEmp, NomEmp, RFCEmp, SexoEmp, DirEmp),

Depto-Emp (IdDepto, IdEmp)

Note que la asociación fue convertida en Depto-Emp tomando los atributos llave de las clases que estaba vinculando. Pero, ¿cuál es la llave de esta relación? Para poder obtener la respuesta piense en el número de registros que pueda tener cada relación.

Departamento:N

Empleado: M

Depto-Emp: M

Existirán tantos vínculos entre los empleados y los departamentos como empleados existan. De ahí que el número de registros en Depto-Emp es M.

Por lo tanto la llave de Depto-Emp es IdEmp, pues permite identificar un registro dentro de la relación de forma únivoca. La situación que tengan el mismo número de registros, permite que se pueda combinar la relación Empleado con Depto-Emp juntándolas de tal forma que tenemos:

Empleado-DeptoEmp (IdEmp <<de Empleado>>, NomEmp, RFCEmp,
SexoEmp, DirEmp, IdDepto, IdEmp <<de Depto-Emp>>)

Pero como los valores de los dos IdEmp son idénticos, entonces sólo nos quedamos con uno. Esta relación vemos que es la misma del Empleado que se tenía, sólo que se ha agregado los atributos llave de la clase del extremo 1 de la asociación. Es por eso que, la podemos renombrar a Empleado.quedando el modelo final con sólo dos relaciones:

Departamento (IdDepto, NomDepto),

Empleado (IdEmp, NomEmp, RFCEmp, SexoEmp, DirEmp, IdDepto)

La forma genérica abreviada de convertir una asociación uno a muchos es:

- Convertir las clases de la asociación binaria a relaciones.
- A la relación correspondiente al extremo del muchos de la asociación, se le agregan los atributos llave de la relación correspondiente al extremo del uno de la asociación.

Note que el modelo resultante del ejemplo cumple con la forma genérica abreviada.

8.4.3 Conversión de Asociaciones Binarias muchos a muchos

Los dos casos anteriores corresponden a casos particulares de asociaciones de muchos a muchos. La conversión de modelos con asociaciones muchos a muchos es tal y como se planteo al inicio de esta sección:

- Las clases en los extremos de la asociación se convierten a relaciones de la forma en que se muestra en la sección 8.2 salvo que si los objetos de la clase sólo se pueden determinar a través de otro objeto de la asociación, será necesario agregarle los atributos llave de la clase que lo identifica.
- Las asociaciones se convertirán a relaciones que tendrán como atributos, los atributos de las clases asociativas que tengan y los atributos llave de todas las clases a las que estén vinculando.

Suponga el ejemplo en el que se desea mantener la información sobre las distintas posiciones que un empleado a tenido en los distintos departamentos de la empresa. El modelo se describe en la Figura 8.4.

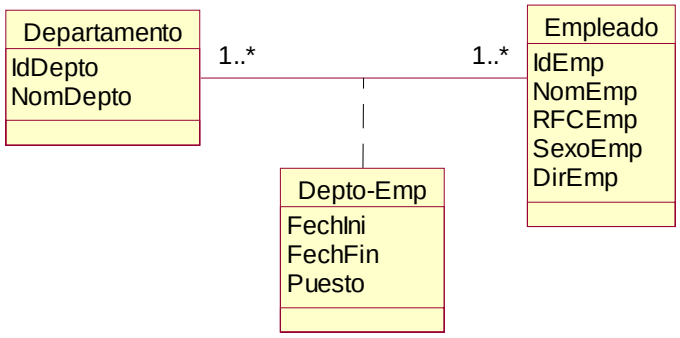

Figura 8.4 Asociación binaria muchos a muchos.

En este ejemplo, se tiene una clase asociativa que contiene la información histórica de cada asignación que ha tenido un empleado, como la fecha inicial en el empleado tomó la posición en el departamento, la fecha en que dejo tal posición y el puesto que ocupó.

Las conversión de este modelo corresponde a las siguientes relaciones:

Departamento (IdDepto, NomDepto),

Empleado (IdEmp, NomEmp, RFCEmp, SexoEmp, DirEmp),

Depto-Emp (FechIni, FechFin, Puesto, IdDepto, IdEmp)

En este caso se observa que la relación resultante de la asociación contiene los atributos de la clase asociativa y además contiene las llaves de las clases que vincula. En este ejemplo es necesario definir la llave de esta relación. En este punto es necesario hacer varias consideraciones referentes a las reglas de negocio que pudieran tenerse, que determinarán la llave de la relación.

- Un empleado puede cambiar de puesto dentro de un departamento y es necesario llevar historia de estos cambios.

  En este caso, la llave de la relación será:

  Depto-Emp (FechIni, FechFin, Puesto, IdDepto, IdEmp)

  Puesto que un empleado en realidad no puede cambiar de puesto ni de departamento en el mismo día.

- Un empleado puede cambiar de puesto dentro de un departamento y sólo es necesario registrar el último puesto del empleado dentro del departamento. La fecha inicial corresponderá a la fecha en que el empleado comenzó a trabajar en el departamento.

  En este caso la llave de la relación será:

  Depto-Emp (FechIni, FechFin, Puesto, IdDepto, IdEmp)

  Puesto que un empleado en realidad no puede tener dos veces el mismo puesto dentro del mismo departamento.

- Un empleado no puede cambiar de puesto dentro de un departamento.

  En este caso la llave de la relación será:

  Depto-Emp (FechIni, FechFin, Puesto, IdDepto, IdEmp)

  Puesto que habrá sólo un registro por cada cambio de departamento de un empleado.

Los anteriores casos destacan que la conversión no puede hacerse de manera mecánica y que es necesario examinar las reglas de negocio para poder determinar la llave correcta para cada situación.

En general puede decirse que una asociacion muchos a muchos invariablemente generará tres relaciones durante el procedimiento de conversión.

8.4.4 Conversión de Asociaciones N-arias

Las asociaciones binarias son un caso particular de las N-arias, sin embargo, éstas pueden atacarse tal y como se vieron las asociaciones binarias de muchos a muchos de tal forma que se generarán todas las relaciones correspondientes a las clases más una relación adicional que corresponde a la asociación N-aria. Como las asociaciones N-arias no tienen cardinalidad, no existen casos particulares a este respecto.

Suponga el ejemplo que se muestra en la Figura 8.5. Ahí se muestra un modelo estático en el describe la estructura necesaria para poder saber en donde los artículos que se produjeron en un cierto turno, por un cierto trabajador y que se enviaron a un cierto almacén. El modelo plantea una asociación cuaternaria entre las clases Artículo, Operador, Almacén y Turno.

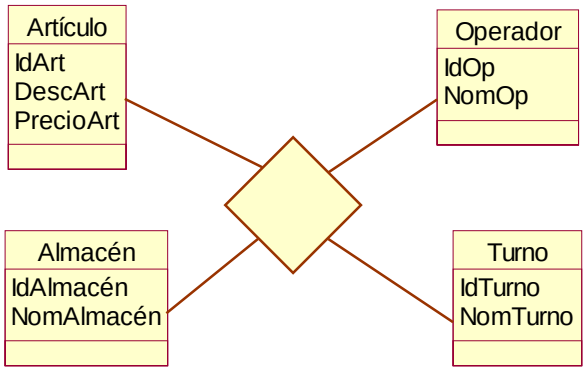

Figura 8.5 Asociación cuaternaria.

El esquema relacional que se obtiene del procedimiento de conversión debe contener todas las relaciones correspondientes a las clases, además se tendrá la relación correspondiente a la asociación. El esquema resultante es:

Artículo (IdArt, DescArt, PrecioArt),

Operador (IdOp, NomOp),

Almacén (IdAlmacén, NomAlmacén),

Turno (IdTurno, NomTurno),

Asociación (IdArt, IdOp, IdAlmacén, IdTurno)

En este caso se decidió colocar todos los atributos como llave de la relación correspondiente a la asociación, pero como se mostro en el caso de las asociaciones binarias de muchos a muchos, es necesario determinarla con cuidado.

Una asociación N-aria por lo regular puede reducirse a varias asociaciones binarias colocando una Clase que reemplace a tal asociación como se muestra en la Figura 8.6.

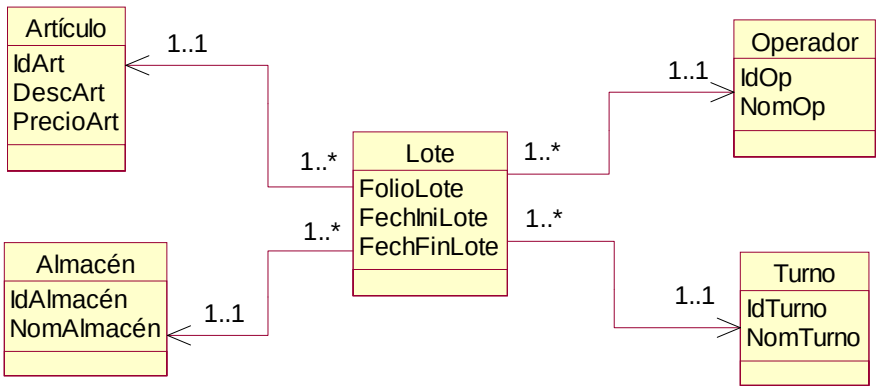

Figura 8.6 Asociación cuaternaria eliminada por una clase.

Note que incluso, se le pueden agregar explícitamente atributos a la clase que está reemplazando a la asociación cuaternaria. El esquema resultante quedaría idéntico, sólo cambiaría para la clase nueva que tendrá el siguiente esquema.

Lote (IdArt, IdOp, IdAlmacén, IdTurno, FolioLote, FechIniLote, FechFinLote)

Esta situación permite formular de mejor manera llaves relacionales de acuerdo a la lógica del negocio o bien debido a que el análisis sugiere nuevos campos que sirvan como llaves. En cualquier caso es importante señalar que cualquier asociación N-aria puede ser reemplazada por varias asociaciones binarias que son más fáciles de manipular, por lo que es recomendable eliminar todas las asociaciones N-arias del modelo antes de hacer el procedimiento de conversión del modelo estático de objetos al modelo relacional.

8.5 Conversión de Herencia

La conversión más complicada se da con las asociaciones de herencia, pues se tienen varias alternativas para poder realizarla, lo que implica que es necesario tomar decisiones y ninguna de las decisiones es correcta al 100% para toda situación. Es por esto que presentaremos las opciones basándonos en un modelo de herencia simple (también se puede aplicar en la mayor parte de las situaciones de herencia múltiple) de un solo nivel (también se puede aplicar para cualquier número de niveles). Cada opción se evalúa con base a qué tan simples y eficientes son las operaciones relacionales que se pueden hacer con el esquema relacional resultante de la conversión:

- Consulta al esquema,
- Agregar un registro,
- Eliminar un registro

Además es necesario indicar cuál es más flexible al momento de dar mantenimiento a la estructura como puede ser que se aumente una clase adicional al esquema de generalización-especialización.

Una vez evaluados se está en posición de dar recomendaciones acerca de en qué situaciones es una opción es más adecuada que otra. Para ilustrarlo se utilizará un modelo que se desea convertir a un esquema relacional el cual se describe en la Figura 8.7, la cual es una estructura de empleados vista en el Capítulo 2 en el que se describen los conceptos del enfoque de orientación a objetos.

En el modelo se observa que la visibilidad de cada atributo del modelo (principalmente para la clase Empleado) es protegido para que pueda funcionar el mecanismo de herencia.

Para poder evaluarlo de una manera conveniente, se describen las operaciones relacionales utilizando SQL. Las consultas que se desea incluir son las de traer a todos los empleados y traer los subconjuntos con base a la definicion de herencia. Las operaciones de inserción y eliminación sólo estarán dadas al nivel de las especializaciones dentro de la estructura de herencia, es decir sólo se darán para empleados de confianza o para empleados sindicalizados.

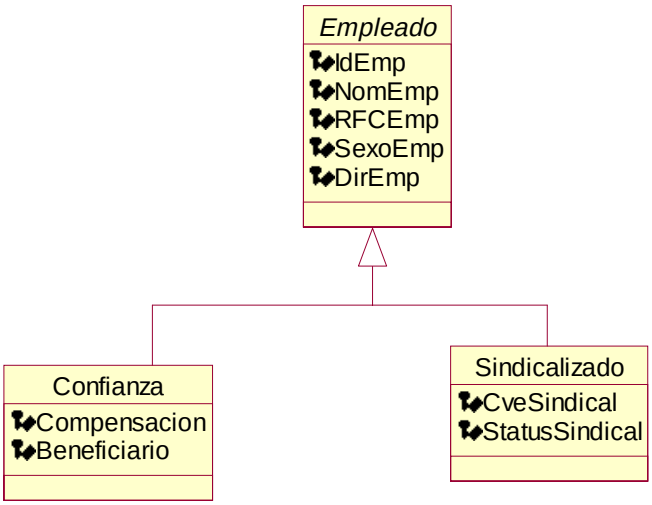

Figura 8.7 Modelo de Clases con asociación de herencia.

Además considere que el mantenimiento a la estructura podría contemplar aumentar una clase a la estructura como se muestra en la figura 8.8.

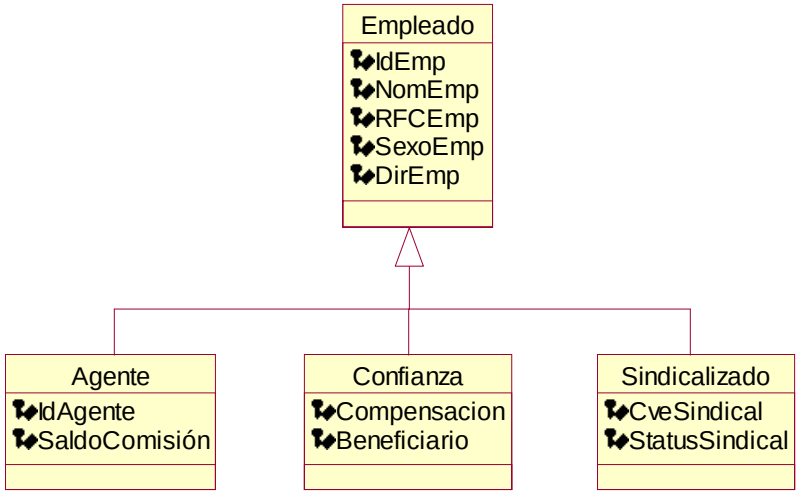

Figura 8.7 Modelo de Clases con asociación de herencia con una clase adicional.

En esta figura se muestra la inclusión de la clase Agente como una especialización adicional del Empleado.

8.5.1 Opción 1 – La herencia genera dos relaciones

Esquema relacional

En esta opción los atributos de la clase abstracta (clase Empleado) se incorporan en las clases concretas (clases Confianza y Sindicalizado), obteniéndose un esquema relacional de dos relaciones.

Las relaciones resultantes se muestran a continuación (las llaves relacionales se muestran con una "@"):

EmpConfianza	**EmpSindicalizado**
@IdEmp	@IdEmp
NomEmp	NomEmp
RFCEmp	RFCEmp
SexoEmp	SexoEmp
DirEmp	DirEmp
Compensación	CveSindical
Beneficiario	StatusSindical

Consultas sobre el esquema

Sólo Sindicalizados

```
SELECT * FROM EmpSindicalizado
```

Sólo Confianza

```
SELECT * FROM EmpConfianza
```

Todos los Empleados

```
SELECT CveEmp, NomEmp, RFCEmp, SexoEmp, DirEmp
   FROM EmpConfianza
UNION
SELECT CveEmp, NomEmp, RFCEmp, SexoEmp, DirEmp
   FROM EmpSindicalizado
```

Las consultas sobre las especializaciones son muy simples y eficientes, sin embargo, la consulta sobre todos los empleados es más complicada y desde el punto de vista de procesamiento, una unión de conjuntos es muy ineficiente.

Operaciones sobre el esquema

Inserción de un Empleado Sindicalizado

```
INSERT INTO
 EmpSindicalizado (CveEmp, NomEmp, RFCEmp, SexoEmp,
                   DirEmp, CveSindical, StatusSindical)
  VALUES (1, “José”, “FEGJ780923”, “M”, “Dir José”,
          344, “Activo”)
```

Eliminación de un Empleado Sindicalizado

```
DELETE FROM EmpSindicalizado WHERE CveEmp = 1
```

Inserción de un Empleado de Confianza

```
INSERT INTO
   EmpConfianza (CveEmp, NomEmp, RFCEmp, SexoEmp,
                 DirEmp, Compensación, Beneficiario)
   VALUES (2, “Alma”, “GOAA751103”, “F”, “Dir Alma”,
          1523, “Esposo Alma”)
```

Eliminación de un Empleado de Confianza

```
DELETE FROM EmpConfianza WHERE CveEmp = 2
```

Las operaciones resultantes son muy simples en su programación.

Mantenimiento al esquema

- Si se agrega un campo al nivel de la superclase tienen que modificarse las dos relaciones del esquema relacional. Se modifican ligerament las consultas y las operaciones de inserción y eliminación.
- Si se agrega un campo a cualquier especialización, la otra especialización no se ve afectada. Se modifican ligerament las consultas y las operaciones de inserción y eliminación.
- Si se agrega una tabla en la especialización, será necesario generar una nueva relación, las dos relaciones que ya se tienen no se modifican y la consulta de todos los empleados necesita que se le de mantenimiento para que refleje la nueva situación.

8.5.2 Opción 2 – La herencia genera una relación

Esquema relacional

Este caso contempla que sólo se va a generar una relación para toda la estructura de herencia. Este caso puede considerarse como el más simple de todos, sin embargo algunos campos contendrán nulos para ciertas especializaciones, lo cual lo vuelve vulnerable a problemas de actualización. La relación resultante se muestra a continuación (las llaves relacionales se muestran con una “@”):

Empleado
@IdEmp
NomEmp

RFCEmp
SexoEmp
DirEmp
Compensación
Beneficiario
CveSindical
StatusSindical

Consultas sobre el esquema

Sólo Sindicalizados

```
SELECT * FROM Empleado WHERE NOT NULL(CveSindical)
```

Sólo Confianza

```
SELECT * FROM Empleado WHERE NOT NULL(Compensacion)
```

Todos los Empleados

```
SELECT * FROM Empleado
```

Las consultas sobre la generalización son muy simples y las de las especializaciones son simples aunque es necesario definir criterios de selección particulares basándose en el hecho que los datos que no se utilizan se les asigna el valor de NULL.

Operaciones sobre el esquema

Inserción de un Empleado Sindicalizado

```
INSERT INTO
 Empleado (CveEmp, NomEmp, RFCEmp, SexoEmp, DirEmp,
           Compensación, Beneficiario,
           CveSindical, StatusSindical)
  VALUES (1, “José”, “FEGJ780923”, “M”, “Dir José”,
            NULL, NULL, 344, “Activo”)
```

Eliminación de un Empleado Sindicalizado

```
DELETE FROM Empleado WHERE CveEmp = 1
```

Inserción de un Empleado de Confianza

```
INSERT INTO
 Empleado (CveEmp, NomEmp, RFCEmp, SexoEmp, DirEmp,
           Compensación, Beneficiario,
           CveSindical, StatusSindical)
   VALUES (2, “Alma”, “GOAA751103”, “F”, “Dir Alma”,
          1523, “Esposo Alma”, NULL, NULL)
```

Eliminación de un Empleado de Confianza

```
DELETE FROM Empleado WHERE CveEmp = 2
```

Las operaciones en si no presentan mayor problema y son muy simples en su programación, sólo obliga en las inserciones a la colocación de valores nulos en los campos que un tipo de empleado dado no utiliza. Note además que las eliminaciones son tan simples como en la Opción 1.

Mantenimiento al esquema

- Si se agrega un campo al nivel de la superclase tiene que agregarse en la relación y será necesario que las inserciones y eliminaciones se les de mantenimiento. Las consultas no se ven afectadas.
- Si se agrega un campo a cualquier especialización, la otra especialización no se ve afectada, tendrá el mismo inconveniente de la observación anterior. Las consultas no se ven afectadas.
- Si se agrega una tabla en la especialización, será necesario agregar los campos correspondientes a la relación, obligando a darle mantenimiento a las operaciones inserción y eliminación. Las consultas no se ven afectadas.

8.5.3 Opción 3 – La herencia genera tres relaciones sin datos duplicados

Esquema relacional

Este caso contempla que todas las clases del esquema generan su propia relación de tal forma que las relaciones provenientes de las especializaciones tienen los atributos directos y la llave de la relación correspondiente a la generalización. Este caso considera que los atributos llave de la superclase se repitan en todas las relaciones correspondientes a las clases especializadas. En el ejemplo, IdEmp es el atributo llave de Empleado (la generalización) y se coloca en cada especialización.

La relación resultante se muestra a continuación (las llaves relacionales se muestran con una "@"):

Empleado	**EmpConfianza**	**EmpSindicalizado**
@IdEmp	@IdEmp	@IdEmp
NomEmp	Compensación	CveSindical
RFCEmp	**Beneficiario**	StatusSindical
SexoEmp		
DirEmp		

Consultas sobre el esquema

Sólo Sindicalizados

```
SELECT S.CveSindical, S.StatusSindical,
       E.IdEmp, E.NomEmp, E.RFCEmp,
       E.SexoEmp, E.DirEmp
  FROM EmpSindicalizado S,Empleado E
 WHERE S.IdEmp = E.IdEmp
```

Sólo Confianza

```
SELECT C.Compensacion, C.Beneficiario,
       E.IdEmp, E.NomEmp, E.RFCEmp,
       E.SexoEmp, E.DirEmp
   FROM EmpConfianza C,Empleado E
 WHERE C.IdEmp = E.IdEmp
```

Todos los Empleados

```
SELECT * FROM Empleado
```

Las consultas sobre la generalización son muy simples, sin embargo, las de lass especializaciones son más elaboradas principalmente por el hecho que es necesario realizar el JOIN de la relacion especializada con la relación de generalización que contiene todos los datos de descripción del Empleado. En realidad la utilización de un JOIN no degrada singnificativamente los tiempos de respuesta en relaciones bien configuradas por los índices y si el manejador de la base de datos está bien configurado, aunque el tiempo de procesamiento siempre será más largo que el de una consulta que no utilice el JOIN.

Operaciones sobre el esquema

Inserción de un Empleado Sindicalizado

```
INSERT INTO
    Empleado (CveEmp, NomEmp, RFCEmp, SexoEmp, DirEmp)
  VALUES (1, “José”, “FEGJ780923”, “M”, “Dir José”)

INSERT INTO
    EmpSindicalizado (CveEmp, CveSindical,
                      StatusSindical)
  VALUES (1, 344, “Activo”)
```

Eliminación de un Empleado Sindicalizado

```
DELETE FROM Empleado WHERE CveEmp = 1

DELETE FROM EmpSindicalizado WHERE CveEmp = 1
```

Inserción de un Empleado de Confianza

```
INSERT INTO
    Empleado (CveEmp, NomEmp, RFCEmp, SexoEmp, DirEmp)
  VALUES (2, “Alma”, “GOAA751103”, “F”, “Dir Alma”)
```

```
INSERT INTO
    EmpConfianza (CveEmp, Compensación, Beneficiario)
        VALUES (2, 1523, "Esposo Alma")
```

Eliminación de un Empleado de Confianza

```
DELETE FROM Empleado WHERE CveEmp = 2

DELETE FROM EmpConfianza WHERE CveEmp = 2
```

Este esquema obliga a que cada operación de mantenimiento a una especialización tenga que hacer dos operaciones relacionales, lo cual evidentemente incrementa el tiempo de proceso.

Mantenimiento al esquema

- Si se agrega un campo al nivel de la superclase la única afectación es en el esquema de la superclase y adecuaciones menores en las operaciones de consulta. A nivel de operaciones tendrán que hacerse modificaciones menores para que se registren los datos correspondientes a los nuevos atributos.
- Si se agrega un campo a cualquier especialización, la otra especialización no se ve afectada, sólo las consultas y operaciones de la especialización experimentarán adecuaciones menores.
- Si se agrega una nueva especialización, ninguna de las consultas y operaciones de las especializaciones existentes sufrirán modificación alguna, e incluso al nivel de la consulta de la generalización también no se verá afectada la forma de la consulta original.

8.5.4 Opción 4 – La herencia genera tres relaciones con datos duplicados

Esquema relacional

Este caso contempla que todas las clases del esquema generan su propia relación de tal forma que las relaciones tienen los atributos directos y los heredados. Este caso considera que los atributos de la superclase se repitan en todas las relaciones provenientes de las clases especializadas. La relación resultante se muestra a continuación (las llaves relacionales se muestran con una "@"):

Empleado	**EmpConfianza**	**EmpSindicalizado**
@IdEmp	@IdEmp	@IdEmp
NomEmp	NomEmp	NomEmp
RFCEmp	RFCEmp	RFCEmp
SexoEmp	SexoEmp	SexoEmp
DirEmp	DirEmp	DirEmp
	Compensación	CveSindical

	Beneficiario	StatusSindical

Consultas sobre el esquema

Sólo Sindicalizados

```
SELECT * FROM EmpSindicalizado
```

Sólo Confianza

```
SELECT * FROM EmpConfianza
```

Todos los Empleados

```
SELECT * FROM Empleado
```

Las consultas se vuelven muy simples para cada caso contemplado..

Operaciones sobre el esquema

Inserción de un Empleado Sindicalizado

```
INSERT INTO
    Empleado (CveEmp, NomEmp, RFCEmp, SexoEmp, DirEmp)
  VALUES (1, “José”, “FEGJ780923”, “M”, “Dir José”)

INSERT INTO
    EmpSindicalizado (CveEmp, NomEmp, RFCEmp, SexoEmp,
                      DirEmp, CveSindical,
                      StatusSindical)
  VALUES (1, “José”, “FEGJ780923”, “M”, “Dir José”,
          344, “Activo”)
```

Eliminación de un Empleado Sindicalizado

```
DELETE FROM Empleado WHERE CveEmp = 1

DELETE FROM EmpSindicalizado WHERE CveEmp = 1
```

Inserción de un Empleado de Confianza

```
INSERT INTO
    Empleado (CveEmp, NomEmp, RFCEmp, SexoEmp, DirEmp)
  VALUES (2, “Alma”, “GOAA751103”, “F”, “Dir Alma”)

INSERT INTO
    EmpConfianza (CveEmp, NomEmp, RFCEmp, SexoEmp,
                  DirEmp, Compensación, Beneficiario)
   VALUES (2, “Alma”, “GOAA751103”, “F”, “Dir Alma”,
           1523, “Esposo Alma”)
```

Eliminación de un Empleado de Confianza

```
DELETE FROM Empleado WHERE CveEmp = 2
```

```
DELETE FROM EmpConfianza WHERE CveEmp = 2
```

Este esquema al igual que en la opción 3, obliga que cada operación de mantenimiento a una especialización tenga que hacer dos operaciones relacionales, lo cual evidentemente incrementa el tiempo de proceso.

Mantenimiento al esquema

- Si se agrega un campo al nivel de la superclase se afectan todas las especializaciones pues hay que agregar el campo en todas las relaciones del modelo.
- Si se agrega un campo a cualquier especialización, la otra especialización no se ve afectada, sólo las consultas y operaciones de la especialización experimentarán adecuaciones menores.
- Si se agrega una nueva especialización, ninguna de las consultas y operaciones de las especializaciones existentes sufrirán modificación alguna, e incluso al nivel de la consulta de la generalización también no se verá afectada la forma de la consulta original.

8.5.5 Comparación entre opciones

La decisión sobre cuál conversión utilizar estará dada por el ambiente sobre el cual el sistema estará inmerso. Es necesario identificar las necesidades o los requisitos no funcionales para determinar el ambiente real. En caso que existan requerimientos encontrados, debe tratarse de buscar el balance para satisfacer de la mejor manera todos los requerimientos. Los aspectos a considerar son los siguientes y las opciones más recomendables para esos casos se enlistan en orden de importancia:

Alta volatilidad de requerimientos

1. Opción 3 Tres relaciones sin duplicaciones
2. Opción 1 Dos relaciones
3. Opción 4 Tres relaciones con duplicaciones

Alto volumen de consultas

1. Opción 2 Una sola relación
2. Opción 4 Tres relaciones con duplicaciones
3. Opción 3 Tres relaciones sin duplicaciones

Alto volumen de operaciones de inserción y eliminación

1. Opción 2 Una sola relación
2. Opción 1 Dos relaciones
3. Opción 3 Tres relaciones sin duplicaciones

Es claro que pueden existir ocasiones en las que se tengan más de dos aspectos que satisfacer. En tales casos deben evaluarse las opciones que proporcionen mayor beneficio integrado a los requerimientos.

8.5.6 Detalles de implantación

Es claro que las conversiones mostradas describen los esquemas relacionales sin ningún soporte para la simplificación de la programación. Los esquemas pueden tener definciones adicionales tales como vistas que permitan hacerle ver el programador tablas totalmente integradas en lugar de que estén fragmentadas. Por otro lado cuando se tienen operaciones de inserción y eliminación que involucran a dos o más tablas pueden implantarse estas operaciones a través de triggers que aseguren que siempre se ejecutarán juntas. Estas consideraciones serán particularmente importantes en las opciones 3 y 4 en las que se tienen relaciones que se desprenden de todas las clases del modelo.

En cualquier caso recuerde que el acceso a los datos debe estar oculto a los objetos dinámicos del programa a través de objetos datos que se encarguen de los detalles de implantación de la información persistente en la base de datos.

8.6 Referencias

[Jacobson1993]

Jacobson, Ivar, Christerson, Magnus; Jonsson, Patrik; Övergaard, Gunnar

Object Oriented Software Engineering, a Use Case Driven Approach

Addison-Wesley, 1992

[SILBERSCHATZ1998]

Silberschatz, Abraham, Korth, Henry F.; Sudarshan, S.

Fundamentos de Bases de Datos

McGraw-Hill-/Interamericana de España, 1998