MODELAGEM DE DADOS – TEORIA E PRÁTICA

ARAÚJO, M. A. P.

1. INTRODUÇÃO

Modelagem de sistemas, tanto a nível funcional quanto de dados, é um requisito fundamental para a obtenção de produtos de software de maior qualidade e confiabilidade. Entretanto, percebe-se que cada vez menos profissionais têm dado a atenção devida ao processo de construção de modelos de suas aplicações. Isso provavelmente se deve às pressões por sistemas em prazos cada vez mais curtos e com menores custos de produção mas, por outro lado, acaba por prejudicar – e muito – o entendimento correto do problema e, consequentemente, a construção do sistema que atenda às reais expectativas do usuário. Esta situação muitas vezes leva a sistemas de baixa qualidade, com elevada necessidade de modificação e de difícil manutenção.

Neste sentido, este trabalho apresenta as principais técnicas de modelagem de dados, não tendo por objetivo tratar de modelagem funcional. Assim, no que diz respeito à modelagem de dados, será discutido tanto o modelo conceitual através do MER (Modelo Entidade-Relacionamento), quanto o modelo lógico através do MR (Modelo Relacional).

A abordagem apresentada neste artigo une a teoria destes modelos de dados com um estudo de caso prático, discutindo diferentes alternativas de solução a partir de uma série de problemas propostos. Assim, serão exercitadas técnicas básicas e avançadas de modelagem de dados, com suas possíveis soluções comentadas. Desta forma, espera-se que conhecimentos úteis à prática profissional de modelagem e manipulação de bancos de dados relacionais tenham sido apresentados e discutidos. Para isso, será considerado um fragmento de um sistema de controle de biblioteca, onde é importante considerar que o objetivo aqui é exercitar o máximo possível de técnicas de modelagem de dados, bem como discutir suas possibilidades de solução.

Este artigo é dividido em outras seis seções, além desta introdução. Na seção 2 são apresentadas as principais características de modelagem de dados. A seção 3 discute a utilização de relacionamentos 1:1 e 1:n, enquanto a seção 4 apresenta a utilização de relacionamentos n:n. Na seção 5 são abordados os auto-relacionamentos e relacionamentos ternários. A seção 6 descreve a utilização de Agregações e Estruturas de Generalização/Especialização e, por fim, a seção 7 apresenta as considerações finais.

2. FUNDAMENTOS DE MODELAGEM DE DADOS

O Modelo Entidade-Relacionamento é um modelo de alto nível, independente do SGBD (Sistemas Gerenciadores de Bancos de Dados), que representa o problema a ser modelado. A notação que será utilizada para a representação deste modelo é o DER (Diagrama Entidade-Relacionamento), exemplificado na **Figura 1**, onde os retângulos representam as entidades (elementos do domínio do problema) e os losangos representam os relacionamentos entre estas entidades (HEUSER, 2004). Entidades ainda são descritas através de atributos e devem possuir uma chave primária (ou *Primary Key* - atributo ou conjunto de atributos que identificam unicamente uma instância em uma entidade, e que não podem receber um valor nulo). A **Figura 1** representa que uma instância da Entidade A está associada a zero (opcional) ou mais instâncias da Entidade B. Por outro lado, uma instância da Entidade B está associada a uma (obrigatoriedade), e somente uma, instância da Entidade A. A este par de elementos chama-se cardinalidade, onde o primeiro elemento indica a participação (opcional ou obrigatório) do relacionamento, enquanto o segundo representa o grau do relacionamento (um ou muitos). Naturalmente, existem outros elementos utilizados na construção deste diagrama, como agregação, relacionamento ternário (ou de maior grau), auto-relacionamento e generalização/especialização, que serão apresentados posteriormente.

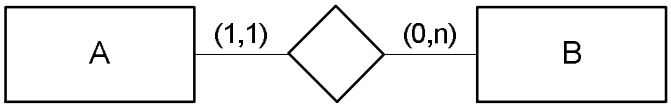

Figura 1. Notação do Diagrama Entidade-Relacionamento

Relacionamentos ainda podem ter atributos, quando algum dado precisa ser associado à ligação das duas instâncias envolvidas. Relacionamentos são descritos através da cardinalidade, que indica como as instâncias das entidades se relacionam. Os tipos utilizados na modelagem são (KORTH, SILBERCHATZ e SUDARSHAN, 2006):

- **Um-para-um (1:1):** uma instância em “A” está associada com no máximo uma instância em “B”, e uma instância em “B” está associada com no máximo uma instância em “A”;
- **Um-para-muitos (1:n):** uma instância em “A” está associada a qualquer número de instâncias em “B”, e uma instância em “B”, todavia, pode estar associado a no máximo uma instância em “A”;
- **Muitos-para-muitos (n:n):** uma instância em “A” está associada a qualquer número de instâncias em “B” e vice-versa. Alguns autores preferem chamar esta cardinalidade de m:n, por considerar que podem representar valores diferentes.

O modelo conceitual representa os elementos do domínio do problema e, consequentemente, não considera questões tecnológicas. Assim, alguns dos elementos descritos neste modelo não possuem correspondência com os recursos oferecidos pelos bancos de dados relacionais, tornando necessário transformar o Modelo Entidade-Relacionamento em uma notação que possa ser implementada neste tipo de banco de dados. O DTR (Diagrama de Tabelas Relacionais) é uma representação gráfica deste modelo, cuja notação está exemplificada na **Figura 2**, retratando a mesma situação descrita na **Figura 1**. Esta notação também é comumente conhecida como “Pé de Galinha”, devido à simbologia utilizada (COUGO, 1997). Este diagrama é interessante, pois apresenta elementos com correspondência direta àqueles implementados nos bancos de dados relacionais, facilitando a transição do modelo conceitual para o lógico. Além disso, muitas ferramentas CASE atuais implementam apenas este diagrama. Assim, no decorrer deste artigo, quando necessário, também será apresentado este diagrama de maneira a facilitar o entendimento desta transição.

Figura 2. Notação do Diagrama de Tabelas Relacionais

Neste sentido, existem algumas regras de conversão do Modelo Entidade-Relacionamento para o Diagrama de Tabelas Relacionais, sendo as principais (ELMASRI e NAVATHE, 2005):

- Toda entidade vira uma tabela;
- Relacionamentos que possuem atributos viram tabelas (existe a possibilidade em relacionamentos 1:n dos atributos irem para uma das tabelas, ao invés de se criar uma nova. Entretanto, relacionamentos com atributos são mais comuns em relacionamentos n:n, gerando assim uma nova tabela);
- Relacionamentos são representados por chaves estrangeiras (ou *Foreign Key* – atributo correspondente à chave primária de outra relação, base para a integridade referencial);
- Relacionamentos 1:1 podem ser mapeados numa única tabela (quando possuem a mesma chave primária), em duas tabelas (quando as chaves primárias são diferentes e um dos lados do relacionamento é obrigatório) ou em três tabelas (quando o relacionamento é opcional em ambos os lados);
- Relacionamentos 1:n são mapeados de forma que a chave primária do lado "1" seja representada do lado "n" como chave estrangeira;
- Relacionamentos n:n devem ser transformados em dois relacionamentos 1:n, resultando numa nova tabela;
- Relacionamentos ternários (ou de maior grau) geram novas tabelas;
- Agregações normalmente geram novas tabelas;
- Auto-relacionamentos geram novas tabelas se forem relacionamentos do tipo n:n;
- Generalização/Especialização podem ser mapeadas em uma única tabela, em uma tabela para cada especialização ou uma tabela para cada entidade envolvida, dependendo da situação.

Assim, a partir do modelo conceitual é construído o modelo lógico que, diferentemente do primeiro, é dependente das características do SGBD escolhido. A ferramenta para este fim é o MR (Modelo Relacional), que representa a solução do

problema modelado, considerando as estruturas (relações) que serão transformadas em tabelas quando da criação do banco de dados propriamente dito.

Existem também algumas diferenças na nomenclatura. Por exemplo, “entidades” no Modelo Entidade-Relacionamento são chamadas de “relações” no Modelo Relacional, e “instâncias” são chamadas de “tuplas” (as linhas de uma relação). Quando da construção do banco de dados propriamente dito, as “relações” são chamadas de “tabelas”, as “tuplas” de “registros”, e os “atributos” de “campos” ou “colunas”.

Neste contexto, uma questão importante a ser discutida na construção do Modelo Relacional é a normalização, que é um conjunto de regras que leva à construção de modelos mais robustos, com menos dependências entre seus elementos e menos redundância de informações. Normalização é uma atividade de verificação do modelo lógico e suas Formas Normais mais comuns, apesar de existirem outras, são (DATE, 2004):

- **1ª Forma Normal (1FN):** toda relação deve ter uma chave primária e deve-se garantir que todo atributo seja atômico. Atributos compostos devem ser separados. Por exemplo, um atributo Endereço deve ser subdividido em seus componentes: Logradouro, Número, Complemento, Bairro, Cidade, Estado e CEP. Além disso, atributos multivalorados devem ser discriminados separadamente ou separados em uma outra relação. Por exemplo, um atributo multivalorado Telefones poderia ser separado em Telefone Residencial, Telefone Comercial e Telefone Celular ou, ainda, ser convertido em outra relação que pudesse representar um número indeterminado de telefones.
- **2ª Forma Normal (2FN):** toda relação deve estar na 1FN e devem-se eliminar dependências funcionais parciais, ou seja, todo atributo não chave deve ser totalmente dependente da chave primária. Como exemplo, uma relação que contenha os atributos Código da Obra, Código do Fornecedor, Nome do Fornecedor e Preço de Venda, considerando que a chave primária é composta pelos atributos Código da Obra e Código do Fornecedor, não está na Segunda Forma Normal, uma vez que o Nome do Fornecedor depende apenas do Código do Fornecedor, e não do Código da Obra. Uma nova relação (Fornecedor)

deve ser criada contendo os campos Código do Fornecedor (como chave) e Nome do Fornecedor. Na relação original, ficariam os atributos Código da Obra e o Código do Fornecedor, ambos formando a chave primária composta, e o atributo Preço de Venda. Além disso, o atributo Código do Fornecedor também seria uma chave estrangeira para a nova relação criada. Esta forma normal ajuda a diminuir redundâncias de informações criadas indevidamente.

- **3ª Forma Normal (3FN):** toda relação deve estar na 2FN e devem-se eliminar dependências funcionais transitivas, ou seja, todo atributo não chave deve ser mutuamente independente. Como exemplo, uma relação que contenha os atributos Matrícula do Funcionário (atributo chave), Nome do Funcionário, Código do Departamento e Nome do Departamento não está na Terceira Forma Normal. O Nome do Departamento é dependente do Código do Departamento, e não da Matrícula do Funcionário. Uma mudança no nome do departamento, por exemplo, levaria a modificações em todos os funcionários daquele departamento. Para eliminar este problema, cria-se uma nova relação (Departamento) contendo Código do Departamento e Nome do Departamento. Na relação original, retira-se o Nome de Departamento, mantendo-se o Código do Departamento, agora como chave estrangeira. Esta forma normal também ajuda a diminuir redundâncias e aumentar a independência das relações.

3. UTILIZAÇÃO DE RELACIONAMENTOS 1:1 E 1:N

No sentido de exemplificar estes conceitos, será apresentado um fragmento de um sistema de controle de biblioteca. Como requisitos iniciais foram identificados:

1. Devem ser cadastradas as obras do acervo, que representam livros, periódicos (revistas, jornais) e qualquer outro elemento do acervo da biblioteca. Inicialmente, obras devem possuir um código que as identifique: título, autor principal, ano de publicação, situação (disponível, emprestada) e editora. Editoras, por sua vez, possuem um

código, nome e cidade. Uma obra sempre é de uma editora e uma editora pode possuir diversas obras;

2. Devem ser cadastrados usuários da biblioteca, que devem ter uma identificação única, nome, endereço completo, telefone de contato e CPF;
3. Os funcionários da biblioteca também devem ser cadastrados. Funcionários têm um número de matrícula, seu nome completo e departamento em que trabalha. Departamentos, por sua vez, possuem código e nome. Todo funcionário obrigatoriamente é vinculado a um departamento, que pode ter vários funcionários. Além disso, todo departamento possui um único chefe;
4. Usuários devem poder realizar empréstimo de obras. Um empréstimo deve conter uma única obra e ser de um único usuário, obrigatoriamente. Empréstimos ainda devem registrar a data e horário do empréstimo, data prevista de retorno, bem como o funcionário que o realizou. Quando da devolução da obra em empréstimo, deve-se registrar a data e horário da devolução, bem como o funcionário responsável;
5. Usuários ainda podem realizar reservas de obras. Uma reserva deve conter uma única obra e ser de um único usuário, obrigatoriamente. Reservas ainda devem registrar a data e horário da reserva e data na qual a obra será retirada.

Alguns comentários são importantes de serem considerados quando da modelagem de dados.

Primeiro, deve-se atentar para o fato de que redundância de dados não é desejável. Numa primeira análise, e pela simplicidade inicial do problema, se poderia imaginar a construção de uma única entidade contendo todas as informações do funcionário, por exemplo, incluindo os dados de seu departamento. Entretanto, a cada funcionário alocado a um mesmo departamento, acarretaria que os dados do departamento seriam duplicados no novo funcionário. Se, por exemplo, o departamento mudasse de nome, teria que mudar esta informação nos diversos funcionários lotados nele o que, além de redundante, pode causar inconsistências.

Outra consideração importante é a respeito do endereço do usuário. No modelo conceitual pode-se representar apenas o atributo Endereço, pois é o problema que está sendo modelado. Entretanto, no modelo lógico, representam-se as estruturas internas das relações (atributos, restrições, chaves, relacionamentos). Neste caso, deve-se lembrar que a Primeira Forma Normal (1FN) determina que todo atributo deve ser atômico. Desta forma, deve-se separar o endereço em seus diversos atributos. Isso é importante, pois se for necessário saber quais são os usuários que vivem numa determinada cidade, este atributo deve estar separado. Caso contrário, torna-se difícil distinguir, por exemplo, um usuário que mora na cidade do Rio de Janeiro de outro que mora na Rua Rio de Janeiro na cidade de Belo Horizonte.

Além disso, quando construir um relacionamento entre duas entidades, deve-se tomar cuidado com as obrigatoriedades dos dois lados do relacionamento. Por exemplo, se for modelado que um funcionário deve pertencer a um departamento e, ao mesmo tempo, um departamento deve ter ao menos um funcionário, não se está considerando a situação inicial, com o banco de dados vazio, onde primeiro se cadastram os departamentos sem funcionários e, no momento do cadastramento do funcionário, este é alocado ao departamento.

Também é importante notar que nem todas as entidades estão explicitamente descritas no enunciado do problema.

Por fim, não se deve preocupar com outras necessidades de modelagem, devendo-se focar no problema apresentado.

A **Figura 3** apresenta o Diagrama Entidade-Relacionamento contendo uma possível solução para o problema apresentado. Optou-se por não representar os atributos neste diagrama, para não dificultar a legibilidade. Os atributos serão apresentados posteriormente na estrutura das entidades.

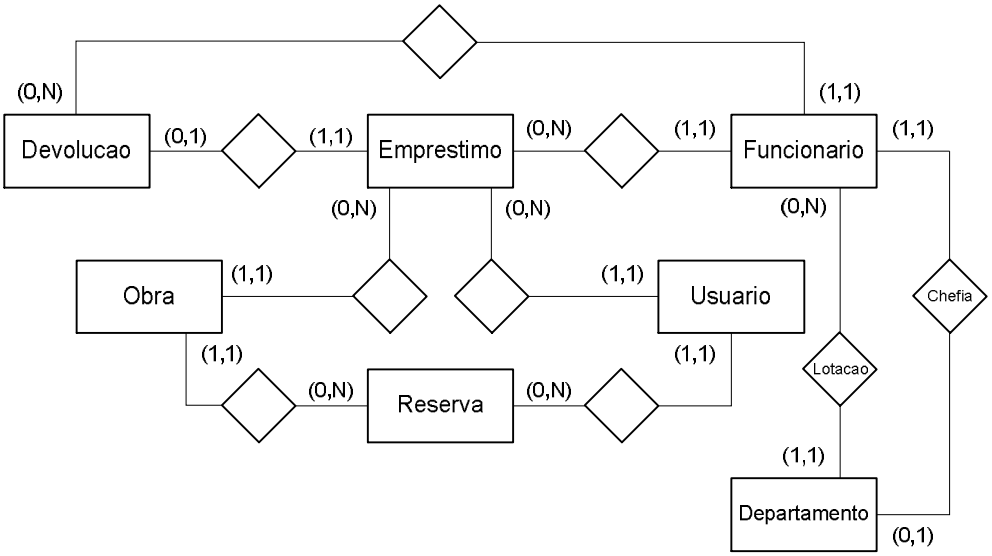

Figura 3. DER do problema apresentado

Vale lembrar que, apesar de uma prática comum, não é obrigatório nomear os relacionamentos, exceto quando estes irão originar tabelas, o que é o caso de relacionamentos n:n ou relacionamentos que possuem atributos. Nestes casos, o nome dos relacionamentos normalmente são os nomes das tabelas resultantes. Entretanto, nomear relacionamentos é importante para uma melhor compreensão do modelo e este recurso será utilizado quando for necessário. Neste exemplo, foram nomeados os relacionamentos entre as entidades Funcionario e Departamento, uma vez que seu entendimento fica dificultado se os relacionamentos não forem nomeados, por existirem dois relacionamentos entre as mesmas entidades.

A seguir, é descrita a estrutura inicial das entidades, onde o atributo determinante está sublinhado. Observe que, propositadamente, foram deixados os atributos da editora dentro da entidade Obra:

- Obra (cod_obra, titulo, autor_principal, ano_publicacao, situacao_obra, tipo_obra, cod_editora, nome_editora, cidade)
- Usuario (cod_usuario, nome_usuario, endereco, telefone, CPF)
- Emprestimo (cod_emprestimo, cod_obra, cod_usuario, data_emprestimo, horario_emprestimo, data_prevista_retorno, num_matricula_funcionario)
- Devolucao (cod_emprestimo, data_devolucao, horario_devolucao, num_matricula_funcionario)
- Funcionario (num_matricula, nome_funcionario, cod_departamento)

- Departamento (cod_departamento, nome_departamento, num_matricula_chefe)
- Reserva (cod_reserva, cod_usuario, cod_obra, data_reserva, horario_reserva, data_retirada)

Para a construção deste diagrama, algumas decisões de projeto foram tomadas e valem a pena serem discutidas:

- Na entidade Emprestimo, uma possibilidade para a definição do atributo determinante seria a concatenação dos atributos cod_usuario e cod_obra, caracterizando uma chave composta. Entretanto, esta chave impediria que o mesmo usuário realizasse o empréstimo da mesma obra em datas diferentes. Uma possibilidade seria incluir também o atributo data_emprestimo na chave. Neste caso, preferiu-se incluir um atributo cod_emprestimo como atributo determinante. Este tipo de situação é chamada de chave cega, onde um novo atributo é inserido por dificuldades na determinação da chave da entidade. O mesmo raciocínio foi utilizado para determinar a chave da entidade Reserva;
- O relacionamento 1:1 entre as entidades Emprestimo e Devolucao não necessariamente precisaria existir. Relacionamentos com esta cardinalidade muitas vezes podem ser eliminados e os atributos das duas entidades podem ser unificados, principalmente neste caso onde os atributos determinantes são os mesmos (a entidade Devolucao poderia ter um atributo determinante diferente, como cod_devolucao mas, neste caso, seria necessário definir um outro atributo que fizesse a ligação com a entidade Emprestimo). Assim, uma possibilidade seria transferir para a entidade Emprestimo todos os atributos da entidade Devolucao e eliminar esta entidade. Com isso, o empréstimo também teria os dados de sua devolução, sendo necessário ter um outro relacionamento com a entidade Funcionario, para representar o funcionário responsável pela devolução. Neste exemplo, preferiu-se manter as entidades separadas, uma vez que são eventos que representam situações diferentes e acontecem em momentos distintos e, no caso da devolução, pode nem acontecer;

- Os dois relacionamentos entre as entidades Funcionario e Departamento são importantes, uma vez que representam ligações diferentes entre as entidades, um representando lotação e outro chefia. Neste caso, o relacionamento 1:1 não poderia ser eliminado, uma vez que existe um segundo relacionamento entre as mesmas entidades;
- Optou-se por não fazer um relacionamento entre as entidades Emprestimo e Reserva, representando que um empréstimo possa ter sido efetivado em função de uma reserva. Esta decisão foi tomada uma vez que não existe a necessidade de estar armazenando definitivamente as reservas, podendo as mesmas ser eliminadas após a data da reserva ter sido vencida;
- Deve-se observar ainda um problema na entidade Obra. Caso existam vários exemplares da mesma obra, tem-se que cadastrar a obra várias vezes, uma para cada exemplar. Este problema será resolvido posteriormente.

O próximo passo é analisar se as entidades representadas estão normalizadas, podendo transformar-se em relações. Segue esta análise das Formas Normais:

- **1ª FN:** esta Forma Normal não foi satisfeita, uma vez que o atributo Endereco da entidade Usuario não é atômico. Para satisfazer esta FN, este atributo deve ser desmembrado em Logradouro, Numero, Complemento, Bairro, Cidade, UF, CEP;
- **2ª FN:** o modelo encontra-se na Segunda Forma Normal, que determina que todo atributo não chave deve ser totalmente funcionalmente dependente da chave primária, e não de parte dela. Só faz sentido preocupar-se com esta Forma Normal para aquelas relações que possuem chave primária composta. No estudo de caso em questão, como todas as relações possuem chaves primárias simples, contendo de apenas um atributo, não é necessário preocupar-se com esta Forma Normal no momento;
- **3ª FN:** pode-se observar um problema com esta Forma Normal ao analisar a relação Obra. Neste caso, os atributos nome da editora e cidade da editora dependem funcionalmente apenas do atributo código da editora, que é um

atributo não chave. Lembrando, a 3ª FN indica que todos os atributos não chave devem ser mutuamente independentes. Assim, apesar dos dados da editora poderem ser considerados como atributos da obra, torna-se importante separar estas entidades, primeiro para não ter estes atributos redundantes. Segundo, a digitação diferente do nome de uma editora, por exemplo, faz com que consultas indiquem editoras diferentes, o que não é desejável. Outra situação indesejável acontece se uma editora mudar de cidade, ocasionando uma mudança no valor deste atributo em todas as obras desta editora.

A **Figura 4** apresenta o DTR (Diagrama de Tabelas Relacionais) do modelo após a etapa de normalização. Novamente os atributos foram suprimidos para facilitar a visualização e serão detalhados a seguir.

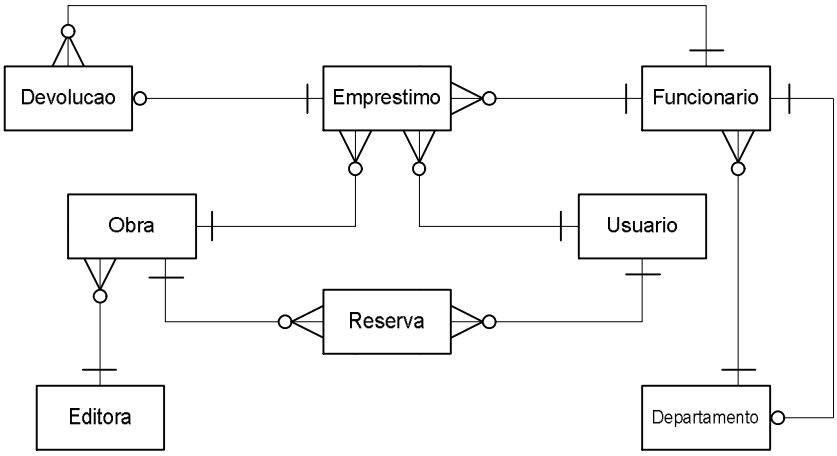

Figura 4. O DTR do problema apresentado

As **Tabelas 1** a **8** apresentam as estruturas das tabelas, onde PK (*Primary Key*) representa a chave primária da tabela (que deve ser obrigatória) e FK (*Foreign Key*) representa uma chave estrangeira, onde o valor do atributo deve ser correspondente a uma chave primária da tabela a qual está referenciando, ou ser nulo, quando não for obrigatório. Isto se chama integridade referencial. Pode-se observar ainda que os tipos de dados são genéricos, não sendo particulares de nenhum SGBD (Sistema Gerenciador de Banco de Dados) específico.

Tabela 1. Estrutura da tabela Obra

Obra					
Chave	**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	cod_obra	Inteiro		Sim	
	titulo	Texto	50	Sim	
	autor_principal	Texto	50	Sim	
	ano_publicacao	Inteiro		Sim	
	situacao_obra	Inteiro		Sim	1=Disponível 2=Emprestado
	tipo_obra	Inteiro		Sim	1=Livro 2=Periódico
FK	cod_editora	Inteiro		Sim	Integridade referencial com tabela Editora

Tabela 2. Estrutura da tabela Editora

Editora					
Chave	**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	cod_editora	Inteiro		Sim	
	nome_editora	Texto	50	Sim	
	cidade	Texto	50	Sim	

Tabela 3. Estrutura da tabela Usuario

Usuario					
Chave	**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	cod_usuario	Inteiro		Sim	
	nome_usuario	Texto	50	Sim	
	end_logradouro	Texto	50	Sim	
	end_numero	Inteiro		Sim	
	end_complemento	Texto	20	Não	
	end_bairro	Texto	30	Não	
	end_cidade	Texto	30	Sim	
	end_UF	Texto	2	Sim	
	end_CEP	Texto	10	Não	
	telefone	Texto	15	Não	
	CPF	Texto	20	Não	

Tabela 4. Estrutura da tabela Emprestimo

Emprestimo					
Chave	**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	cod_emprestimo	Inteiro		Sim	
FK	cod_obra	Inteiro		Sim	Integridade referencial com tabela Obra
FK	cod_usuario	Inteiro		Sim	Integridade referencial com tabela Usuario
	data_emprestimo	Data		Sim	
	horario_emprestimo	Hora		Sim	
	data_prevista_retorno	Data		Sim	
FK	num_matricula_funcionario	Inteiro		Sim	Integridade referencial com tabela Funcionario

Tabela 5. Estrutura da tabela Devolucao

Devolucao					
Chave	**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK FK	cod_emprestimo	Inteiro		Sim	Integridade referencial com tabela Emprestimo
	data_devolucao	Data		Sim	
	horario_devolucao	Hora		Sim	
FK	num_matricula_funcionario	Inteiro		Sim	Integridade referencial com tabela Funcionario

Tabela 6. Estrutura da tabela Funcionario

Funcionario					
Chave	**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	num_matricula	Inteiro		Sim	
	nome_funcionario	Texto	50	Sim	
FK	cod_departamento	Inteiro		Sim	Integridade referencial com tabela Departamento

Tabela 7. Estrutura da tabela Departamento

Departamento					
Chave	**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	cod_departamento	Inteiro		Sim	
	nome_departamento	Texto	50	Sim	
FK	num_matricula_chefe	Inteiro		Sim	Integridade referencial com tabela Funcionário

Tabela 8. Estrutura da tabela Reserva

Reserva					
Chave	**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	cod_reserva	Inteiro		Sim	
FK	cod_usuario	Inteiro		Sim	Integridade referencial com tabela Usuario
FK	cod_obra	Inteiro		Sim	Integridade referencial com tabela Obra
	data_reserva	Data		Sim	
	horario_reserva	Hora		Sim	
	data_retirada	Data		Sim	

Cabe ainda uma observação em relação ao atributo cod_emprestimo da tabela Devolucao. Além de ser chave primária da própria tabela também é, simultaneamente, chave estrangeira em relação à tabela Emprestimo, estabelecendo a integridade referencial de que uma devolução é necessariamente associada a um empréstimo.

4. UTILIZAÇÃO DE RELACIONAMENTOS N:N

Considerando o modelo anterior, esta seção irá propor uma manutenção no modelo de dados construído, de forma a exercitar a utilização de relacionamentos n:n.

Neste sentido, será necessário efetuar uma primeira manutenção no modelo de dados. Existem dois novos requisitos solicitados pela biblioteca:

1. Será necessário armazenar os autores de uma obra. Além disso, é importante saber a ordem dos autores de uma obra, ou seja, quem é o primeiro autor, segundo autor, e assim sucessivamente. Como é comum fazer pesquisas pelo nome de autores, a biblioteca não deseja que o mesmo autor se repita no banco de dados, de forma que seu nome possa ser digitado de maneiras diferentes e dificultar a busca. Assim, deseja-se fazer um cadastro de autores e relacioná-los às obras. Cada autor deve conter apenas um código que o identifique e seu nome completo. Deve-se lembrar que uma obra tem diversos autores (seguindo sua ordem) e um autor pode participar em diversas obras. É importante atentar que existe um campo para este fim (autor_principal) na tabela Obra.
2. Outra necessidade comum numa biblioteca é a consulta de obras por assunto. Uma obra pode estar associada a diversos assuntos e um assunto pode estar vinculado a diversas obras. Novamente, deve-se fazer um cadastro de assuntos (contendo apenas código e descrição), de forma que possam ser associados às obras.

Para estes novos requisitos, primeiro deve-se pensar na solução através do modelo conceitual, para depois trabalhar no modelo lógico. Na transformação do modelo conceitual para o lógico, lembre-se que relacionamentos n:n devem originar novas tabelas.

Inicialmente, será discutido primeiro o problema de associação de assuntos às obras. Para isso, deve-se acrescentar ao modelo conceitual uma entidade Assunto, contendo apenas código e descrição. Como uma obra pode possuir diversos assuntos e um assunto pode estar vinculado a diversas obras, deve-se modificar o modelo conceitual acrescentando um relacionamento n:n entre estas entidades. A **Figura 5** representa o fragmento do modelo conceitual para este problema.

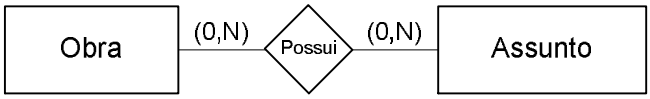

Figura 5. Relacionamento n:n entre Obras e Assuntos

Para transformar este modelo conceitual no modelo lógico, necessita-se fazer uma transformação substituindo o relacionamento por chaves estrangeiras. No caso do sentido da tabela Obra para a tabela Assunto, deveria ser colocado um atributo multivalorado em Obra para apontar para seus diversos assuntos, o mesmo ocorrendo no sentido inverso, da tabela Assunto para a tabela Obra. Entretanto, relacionamentos n:n não são possíveis de serem representados no modelo relacional, uma vez que todos os atributos devem ser atômicos, ou seja, armazenar um único valor. Neste caso, a alternativa é a criação de uma terceira tabela no modelo relacional, conforme ilustrado na **Figura 6**.

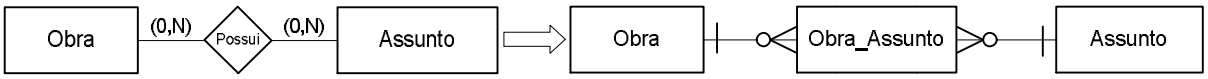

Figura 6. Mapeamento do Modelo Conceitual para o Modelo Lógico

Neste caso, a nova tabela gerada, Obra_Assunto, terá como chave primária a junção das chaves primárias das duas tabelas participantes, formando uma chave primária composta. Além disso, cada parte da chave primária será também uma chave estrangeira para sua tabela de origem. Assim, a cada relacionamento de uma obra com um assunto, gera-se um novo registro na tabela Obra_Assunto, transformando o relacionamento n:n do modelo conceitual em dois relacionamentos 1:n no modelo lógico. Observa-se ainda que o nome do relacionamento no modelo conceitual foi substituído por um nome de tabela que tivesse algum significado no modelo lógico, pois este será o nome da tabela no banco de dados.

Um outro requisito apresentado trata da necessidade de armazenar os autores de uma obra. Trata-se também de um relacionamento n:n, uma vez que uma obra tem diversos autores e um autor pode estar associado a diversas obras. Porém, neste caso específico, tem-se uma particularidade, uma vez que é importante identificar a ordem dos autores de uma obra, caracterizando quem é o primeiro autor, o segundo, e assim sucessivamente. Este requisito tornou-se necessário para evitar cadastrar repetidamente um mesmo autor que participa em diversas obras, facilitando também a busca por autores. A questão a ser tratada aqui é relativa à ordem do autor na lista de autores de uma obra. Esta informação não pode ficar na tabela Obra, uma vez que esta possui diversos autores, e também não pode ficar na tabela Autor, uma vez que o mesmo pode estar vinculado a diversas obras. Neste caso, o local apropriado para armazenar esta informação é no próprio

relacionamento entre as duas tabelas, conforme apresentado na **Figura 7**, que ainda apresenta a tabela resultante no modelo relacional, chamada de Obra_Autor.

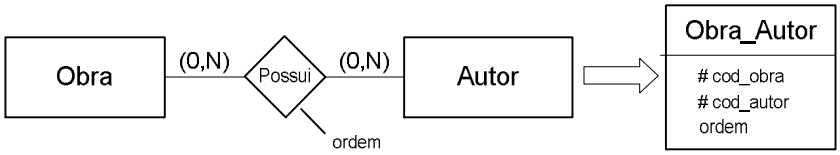

Figura 7. Mapeamento de Relacionamento n:n com atributo

Nesta situação, o atributo “ordem” do relacionamento entre as entidades Obra e Autor representa a posição de um determinado autor em uma determinada obra, na sua lista de autores. Este atributo é representado no modelo lógico como um atributo simples na tabela originada do relacionamento n:n. Ao fazer esta modificação, fica sem sentido o atributo “autor_principal” da tabela Obra, devendo ser retirado.

Desta forma, não se torna necessário repetir um mesmo autor em cada uma das obras que estiver vinculado. Isto faz com que as consultas por autor no banco de dados sejam mais precisas, uma vez que um mesmo autor poderia ser cadastrado de formas diferentes, tendo uma abreviatura em seu nome em uma determinada obra, por exemplo.

Por fim, o último problema a ser resolvido refere-se à repetição de uma mesma obra caso esta tenha mais de um exemplar. Seria interessante que uma obra fosse cadastrada uma única vez contendo os atributos comuns aos seus diversos exemplares. Cada exemplar deveria ter apenas seus atributos específicos, um identificador único, sua data de aquisição e sua situação. Num primeiro momento, isso pode parecer uma modificação pontual, representado através de um relacionamento 1:n, conforme **Figura 8**. Assim, se deve retirar o atributo “situacao_obra” da tabela Obra, uma vez que agora é responsabilidade de cada exemplar armazenar sua situação específica.

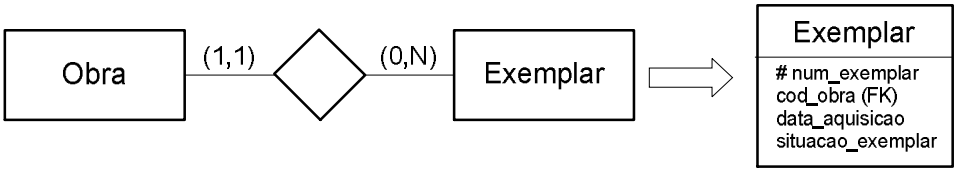

Figura 8. Criação da tabela Exemplar

Ainda segundo o requisito definido, a numeração dos exemplares deve iniciar de 1 a cada Obra. Ou seja, deve-se ter o exemplar 1 da obra “O Código Da Vinci” e o exemplar 1 da obra “O Livro dos Códigos”, por exemplo. Para que isso ocorra, a chave primária da tabela Exemplar deve conter, além do número do exemplar, o código da obra, formando uma chave primária composta. Esta situação de modelagem é chamada entidade fraca, onde a chave primária da entidade fraca (neste caso, a entidade Exemplar) é formada pela chave primária da entidade forte (no caso, a entidade Obra), mais algum atributo que diferencie seus registros (como o número do exemplar). Nota-se assim que a entidade fraca estará sempre carregando o relacionamento com sua entidade forte, sugerindo sempre uma leitura como “um exemplar de uma determinada obra”, neste caso. A **Figura 9** exibe a transformação do modelo conceitual para o modelo lógico relativo a esta situação.

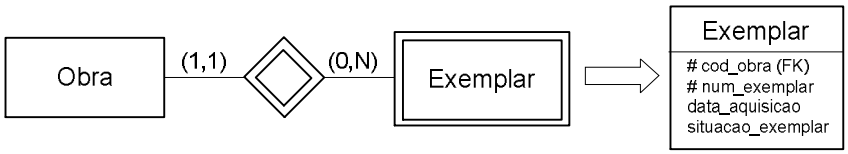

Figura 9. Mapeamento de Entidade Fraca

Percebe-se que, no modelo conceitual, a notação da entidade fraca é feita através de uma linha dupla envolvendo a entidade fraca. Nota-se ainda que, uma vez que esta entidade pode estar associada a diversas outras, para saber quem é sua entidade forte, também se apresenta com linha dupla o relacionamento que liga à entidade forte. No caso do modelo relacional, não existe simbologia específica para este fim, apenas modificando a chave primária da entidade fraca. Entretanto, algumas ferramentas CASE utilizam simbologias próprias para representar entidades fracas, arredondando os cantos do símbolo de entidade, ou com um tipo diferente de linha ligando-a a entidade forte. A presença da chave da entidade forte na entidade fraca enfatiza que, ao excluir um registro na entidade forte, todos os seus associados na entidade fraca também devem ser excluídos, uma vez que a integridade referencial não pode ser violada.

Este requisito ainda impacta o modelo de forma mais significativa. Por exemplo, nesta nova perspectiva, um usuário faria reservas de obras (uma vez que não importa o exemplar para a reserva), mas faria empréstimo de um exemplar específico de uma obra. A **Figura 10** mostra o modelo conceitual após as

modificações propostas anteriormente, enquanto a **Figura 11** apresenta o seu modelo lógico correspondente.

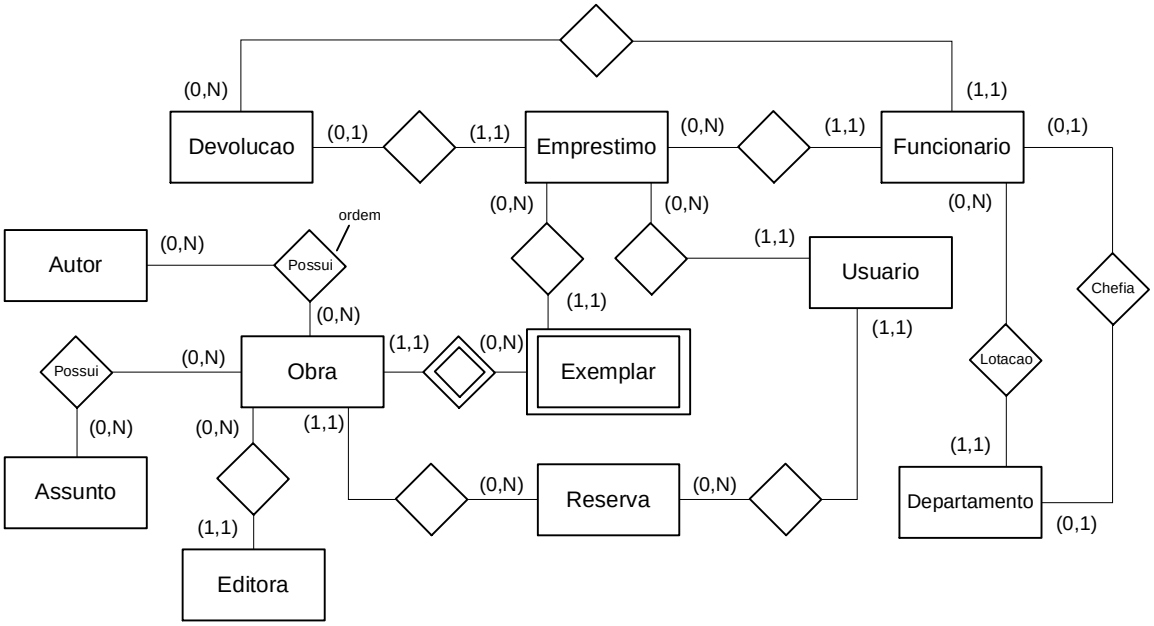

Figura 10. O modelo conceitual após as modificações

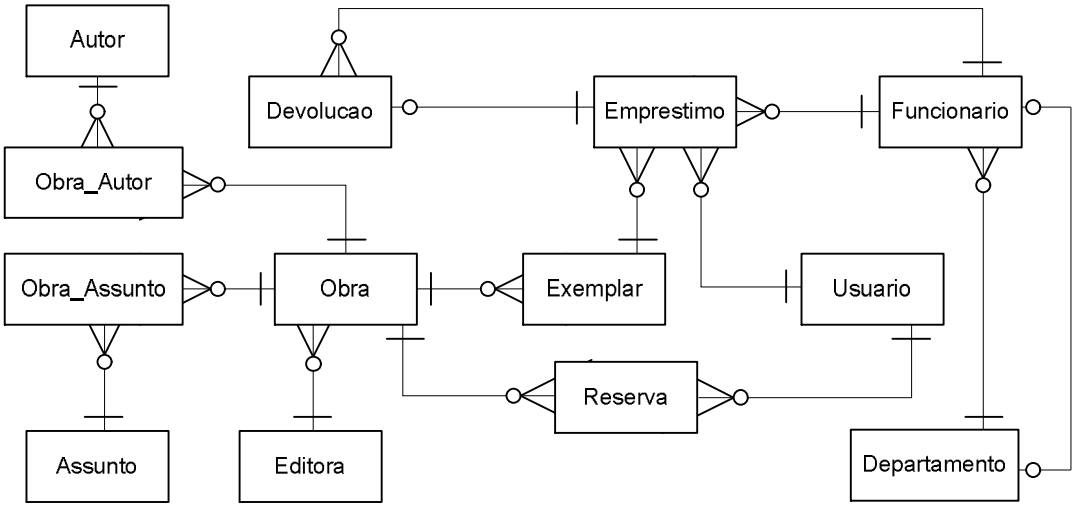

Figura 11. O modelo lógico após as modificações

As **Tabelas 9** a **15** mostram como ficaram as estruturas das tabelas que sofreram modificações em função dos novos requisitos.

Tabela 9. Estrutura da tabela Obra

Obra					
Chave	**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	cod_obra	Inteiro		Sim	
	titulo	Texto	50	Sim	
	ano_publicacao	Inteiro		Sim	
	tipo_obra	Inteiro		Sim	1=Livro 2=Periódico
FK	cod_editora	Inteiro		Sim	Integridade referencial com tabela Editora

Tabela 10. Estrutura da tabela Assunto

Assunto					
Chave	**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	cod_assunto	Inteiro		Sim	
	descricao_assunto	Texto	50	Sim	

Tabela 11. Estrutura da tabela Autor

Autor					
Chave	**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	cod_autor	Inteiro		Sim	
	nome_autor	Texto	50	Sim	

Tabela 12. Estrutura da tabela Obra_Assunto

Obra_Assunto					
Chave	**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	cod_obra	Inteiro		Sim	Integridade referencial com tabela Obra
PK	cod_assunto	Inteiro		Sim	Integridade referencial com tabela Assunto

Tabela 13. Estrutura da tabela Obra_Autor

Obra_Autor					
Chave	**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	cod_obra	Inteiro		Sim	Integridade referencial com tabela Obra
PK	cod_autor	Inteiro		Sim	Integridade referencial com tabela Autor
	ordem	Inteiro		Sim	

Tabela 14. Estrutura da tabela Exemplar

Exemplar						
Chave		**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	FK	cod_obra	Inteiro		Sim	Integridade referencial com tabela Obra
		num_exemplar	Inteiro		Sim	
		data_aquisicao	Data		Não	
		situacao_exemplar	Inteiro		Sim	1=Disponível 2=Emprestado

Tabela 15. Estrutura da tabela Emprestimo

Empréstimo					
Chave	**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	cod_emprestimo	Inteiro		Sim	
FK	cod_obra	Inteiro		Sim	Integridade referencial com tabela Exemplar
	num_exemplar	Inteiro		Sim	
FK	cod_usuario	Inteiro		Sim	Integridade referencial com tabela Usuario
	data_emprestimo	Data		Sim	
	horario_emprestimo	Hora		Sim	
	data_prevista_retorno	Data		Sim	
FK	num_matricula_funcionario	Inteiro		Sim	Integridade referencial com tabela Funcionario

Percebe-se que foram criadas as tabelas Assunto, Autor, Obra_Assunto, Obra_Autor e Exemplar. Na tabela Obra, foram retirados os atributos “autor_principal” e “situacao_obra”, que foi para a tabela Exemplar com o nome “situacao_exemplar”. A ordem do autor na lista de autores está representada pelo atributo “ordem” na tabela Obra_Autor. A tabela Exemplar possui uma chave primária composta, onde a primeira parte da chave, o atributo “cod_obra” também é uma chave estrangeira para a tabela Obra. A tabela Emprestimo precisou ser modificada para que a chave estrangeira anteriormente referente à tabela Obra agora seja referente à tabela Exemplar, inserindo o atributo “num_exemplar”, que forma uma chave estrangeira composta com o atributo “cod_obra”. Como a tabela Exemplar tem uma chave primária composta, as chaves estrangeiras para esta tabela também assim devem ser representadas. A tabela Reserva não muda, uma vez que reservas continuam sendo feitas para obras, e não para exemplares

específicos. As demais tabelas não sofreram modificações e continuam valendo as estruturas que já foram apresentadas.

5. UTILIZAÇÃO DE AUTO-RELACIONAMENTOS E RELACIONAMENTOS TERNÁRIOS

Esta seção apresenta modificações no modelo de dados do sistema de biblioteca para incluir funcionalidades que exercitem a utilização de auto-relacionamentos e relacionamentos ternários.

Assim, tem-se um novo conjunto de requisitos que, para serem atendidos, vão provocar novas modificações no modelo de dados. Os novos requisitos são:

1. Será necessário organizar os assuntos de forma que seja possível associar um determinado assunto a outros assuntos relacionados. Por exemplo, ao fazer uma consulta pelo assunto “Modelagem de Sistemas”, seria importante que as obras relacionadas ao assunto “Modelagem de Dados” também fossem encontradas. Assim, um assunto pode estar relacionado a diversos outros assuntos.
2. No cadastro de um usuário, caso este tenha sido indicado por um outro usuário cadastrado na biblioteca, será necessário registrar esta informação. Assim, um usuário poderá ter sido indicado por um único outro usuário. Por outro lado, um usuário poderá indicar diversos novos usuários.
3. Exemplares podem passar por diversas manutenções durante sua vida útil numa biblioteca, como limpeza, recuperação de páginas ou encadernação. O sistema necessitará cadastrar estas manutenções, contendo o exemplar em manutenção, a data da manutenção, o funcionário responsável pela manutenção e o motivo da manutenção, sendo todos os campos obrigatórios. Motivos de manutenção devem ser cadastrados e possuem apenas código e descrição. Vale ressaltar que um mesmo exemplar pode sofrer diversas manutenções realizadas por diferentes funcionários. Um mesmo funcionário pode fazer várias manutenções por motivos diferentes. Um motivo de manutenção pode

estar associado a diversos exemplares e funcionários. Não existe na biblioteca um código associado a cada manutenção realizada.

Para estes novos requisitos, primeiro deve-se pensar na solução através do modelo conceitual, para depois trabalhar no modelo lógico. Na transformação do modelo conceitual para o lógico, deve-se considerar que auto-relacionamentos de cardinalidade 1:n apenas acrescentam uma chave estrangeira na própria tabela e de cardinalidade n:n devem originar novas tabelas. Relacionamentos ternários no modelo conceitual tendem a gerar novas tabelas no modelo lógico.

O primeiro requisito apresentado nesta seção necessita organizar os assuntos de obras de forma que seja possível associar um determinado assunto a outros relacionados. Por exemplo, ao fazer uma consulta pelo assunto “Modelagem de Sistemas”, seria importante que as obras relacionadas a “Modelagem de Dados” também fossem encontradas. Assim, um assunto pode estar relacionado a diversos outros assuntos. Desta forma, precisa-se efetuar um relacionamento entre dois assuntos, ou seja, um auto-relacionamento na entidade Assunto. Deve-se tomar cuidado para não criar uma outra entidade no modelo conceitual para os assuntos relacionados, pois isso causaria uma redundância na base de dados, fazendo com que assuntos fiquem repetidos em duas tabelas. Este relacionamento será de cardinalidade n:n, uma vez que um assunto está relacionado a diversos outros e vice-versa, forçando a criação de uma nova tabela no modelo lógico. A **Figura 12** exibe o modelo conceitual para resolução desta questão, bem como sua transformação para o modelo lógico. Percebe-se que a chave primária da tabela Assunto_Relacionamento, gerada no modelo lógico, é composta por duas chaves estrangeiras, ambas referenciando a tabela Assunto.

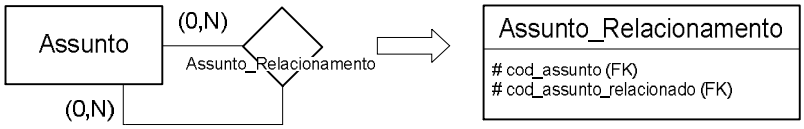

Figura 12. Auto-Relacionamento n:n

O próximo requisito determina que, no cadastro de um usuário, caso este tenha sido indicado por um outro usuário cadastrado na biblioteca, será necessário registrar esta informação. Assim, um usuário poderá ter sido indicado por um único

outro usuário. Por outro lado, um usuário poderá indicar diversos novos usuários. Este problema é semelhante ao anterior, exceto que o relacionamento neste caso é 1:n. Desta forma, como todos são usuários, existirá um auto-relacionamento na tabela Usuário. Sendo um relacionamento 1:n, basta acrescentar uma chave estrangeira para o usuário que fez a indicação na própria tabela Usuario. A **Figura 13** mostra a solução para este problema tanto no modelo conceitual como no lógico, apresentando apenas alguns dos atributos da tabela Usuario.

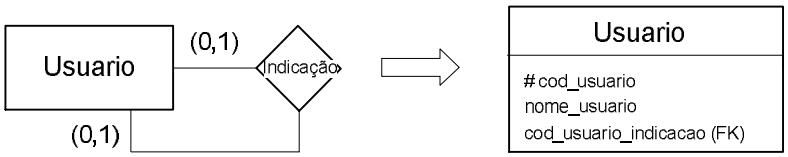

Figura 13. Auto-Relacionamento 1:n

O terceiro e último requisito proposto nesta seção refere-se a exemplares que podem passar por diversas manutenções durante sua vida útil numa biblioteca, como limpeza, recuperação de páginas ou encadernação. O sistema necessitará cadastrar estas manutenções, contendo o exemplar em manutenção, a data da manutenção, o funcionário responsável pela manutenção e o motivo da manutenção, sendo todos os campos obrigatórios. Motivos de manutenção devem ser cadastrados e possuem apenas código e descrição. Vale ressaltar que um mesmo exemplar pode sofrer diversas manutenções realizadas por diferentes funcionários. Um mesmo funcionário pode fazer várias manutenções em exemplares por motivos diferentes. Um motivo de manutenção pode estar associado a diversos exemplares e funcionários. Não existe na biblioteca um código associado a cada manutenção realizada. Pode surgir uma dúvida se seria utilizado um relacionamento ternário ou uma agregação. Neste caso, como todos os elementos envolvidos neste relacionamento são obrigatórios (exemplar, funcionário e motivo da manutenção), e devem existir simultaneamente, pode-se modelar como um relacionamento ternário. Este tipo de relacionamento não seria apropriado se, por exemplo, fosse possível cadastrar a manutenção com seu motivo e, posteriormente, associar o funcionário responsável. Num relacionamento ternário, o comum é que este resulte numa nova tabela, onde sua chave primária é a composição das chaves das entidades envolvidas no relacionamento, obrigatórias simultaneamente. A **Figura 14** apresenta

a solução para este requisito, tanto no modelo conceitual quanto no lógico, apresentando apenas os atributos necessários a esta situação.

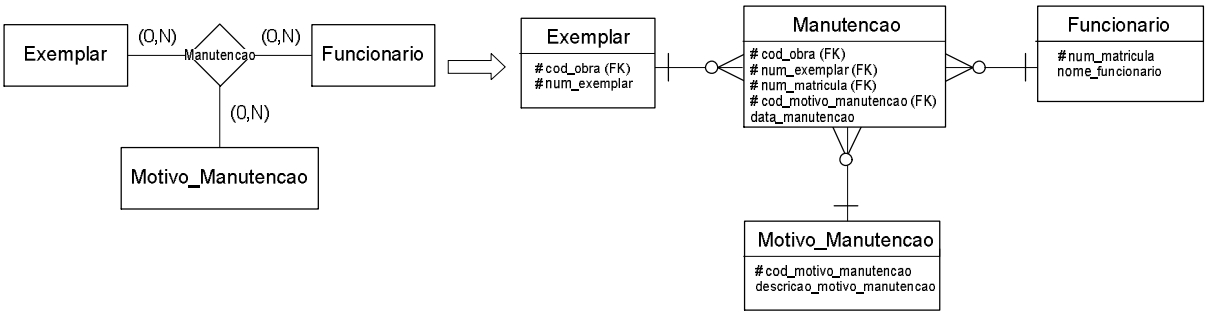

Figura 14. Relacionamento ternário

A **Figura 15** apresenta o modelo conceitual após as modificações realizadas, enquanto a **Figura 16** exibe o modelo lógico correspondente.

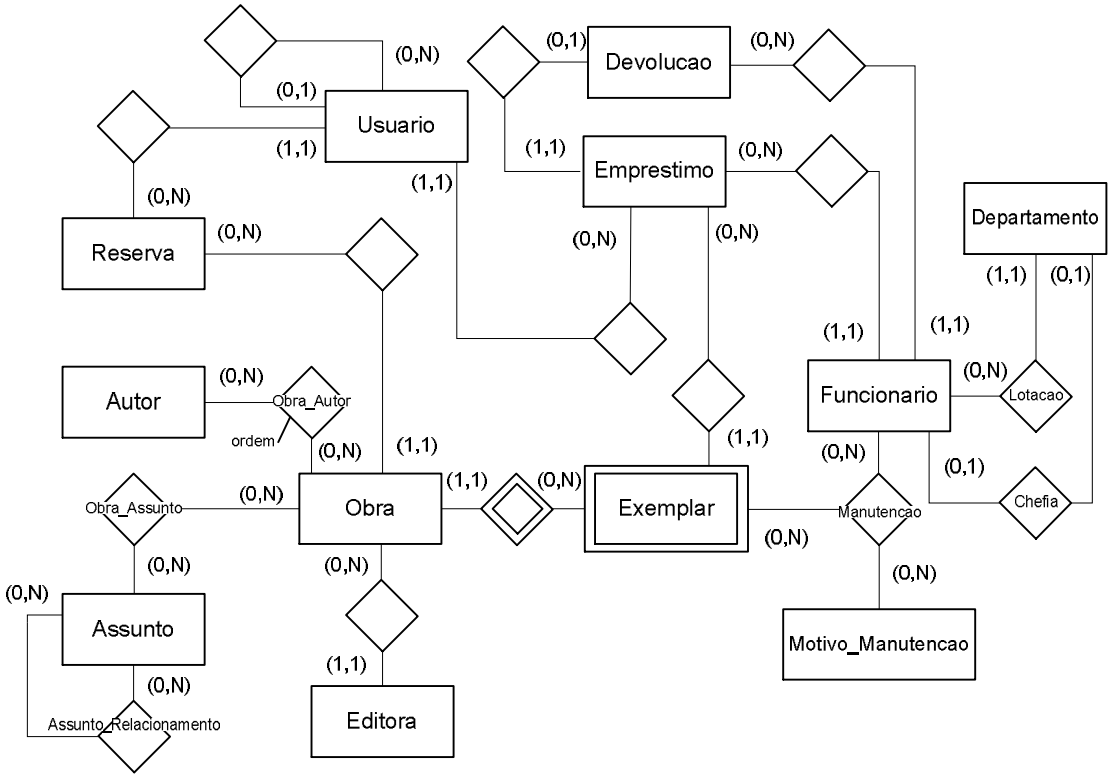

Figura 15. O modelo conceitual após as modificações

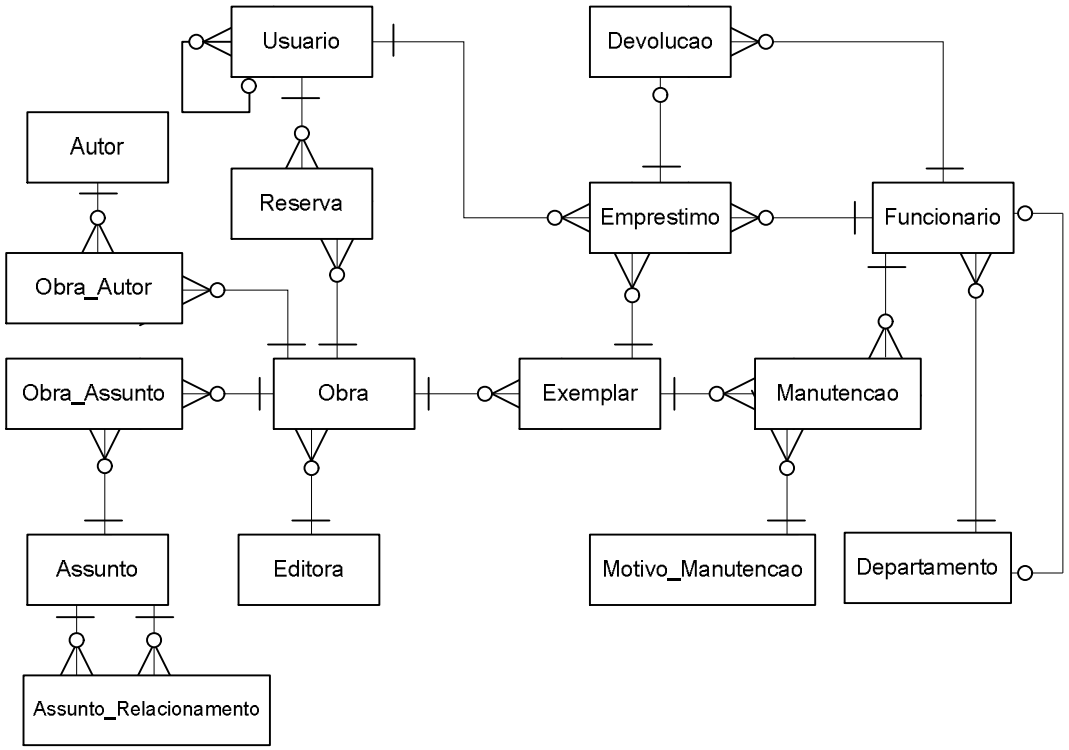

Figura 16. O modelo lógico após as modificações

As **Tabelas 16** a **19** mostram as estruturas das tabelas que sofreram modificações em função dos novos requisitos.

Tabela 16. Estrutura da tabela Assunto_Relacionamento

Assunto_Relacionamento						
Chave		**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	FK	cod_assunto	Inteiro		Sim	Integridade referencial com tabela Assunto
	FK	cod_assunto_relacionado	Inteiro		Sim	Integridade referencial com tabela Assunto

Tabela 17. Estrutura da tabela Usuario

Usuario					
Chave	**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	cod_usuario	Inteiro		Sim	
	nome_usuario	Texto	50	Sim	
	end_logradouro	Texto	50	Sim	
	end_numero	Inteiro		Sim	
	end_complemento	Texto	20	Não	
	end_bairro	Texto	30	Não	
	end_cidade	Texto	30	Sim	
	end_UF	Texto	2	Sim	
	end_CEP	Texto	10	Não	
	telefone	Texto	15	Não	
	CPF	Texto	20	Não	
FK	cod_usuario_indicacao	Inteiro		Não	Integridade referencial com tabela Usuario

Tabela 18. Estrutura da tabela Motivo_Manutencao

Motivo_Manutencao					
Chave	Atributo	Tipo	Tamanho	Obrigatório	Restrições
PK	cod_motivo_manutencao	Inteiro		Sim	
	descricao_motivo_manutencao	Texto	50	Sim	

Tabela 19. Estrutura da tabela Manutencao

Manutencao						
Chave		**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	FK	cod_obra	Inteiro		Sim	Integridade referencial com tabela Exemplar
		num_exemplar	Inteiro		Sim	
	FK	num_matricula	Inteiro		Sim	Integridade referencial com tabela Funcionario
	FK	cod_motivo_manutencao	Inteiro		Sim	Integridade referencial com tabela Motivo_Manutencao
		data_manutencao	Data		Sim	

Percebe-se na tabela Assunto_Relacionamento que ambos os campos formam uma chave primária composta mas, individualmente, são duas chaves estrangeiras para a tabela Assunto, uma vez que os registros desta tabela representam relacionamentos entre dois assuntos quaisquer. A tabela Usuario

recebeu um novo campo (cod_usuario_indicacao), que nada mais é do que uma chave estrangeira para a própria tabela Usuario, evidenciando um auto-relacionamento 1:n. Como um usuário não necessariamente precisa de uma indicação, este campo não é obrigatório. Já a nova tabela Manutencao é resultante do relacionamento ternário do modelo conceitual e sua chave primária é composta dos campos chaves das entidades envolvidas. Uma vez que a entidade Exemplar é uma das entidades envolvidas neste relacionamento e esta é fraca da entidade Obra, tanto o código da obra quanto o código do exemplar passam a fazer parte da chave primária da entidade Manutencao, sendo também uma chave estrangeira composta para a tabela Exemplar. O mesmo acontece para os demais campos que compõem a chave primária, sendo o campo num_matricula também uma chave estrangeira para a entidade Funcionario e o campo cod_motivo_manutencao uma outra chave estrangeira para a recém criada tabela Motivo_Manutencao.

6. UTILIZAÇÃO DE AGREGAÇÕES E ESTRUTURAS DE GENERALIZAÇÃO/ESPECIALIZAÇÃO

Nesta seção será proposto um novo problema, desta vez modificando o modelo de dados do sistema de biblioteca para incluir funcionalidades que exercitem a utilização de agregações e estruturas de generalização e especialização.

Tem-se então um novo conjunto de requisitos que, para serem atendidos, vão provocar novas modificações no modelo de dados. Os novos requisitos são:

1. Será necessário fazer uma distinção entre tipos de obras, que agora podem ser Livros ou Periódicos. Para Livros, deve-se armazenar o número do ISBN (*International Standard Book Number* - Número Padrão Internacional de Livro), enquanto para periódicos, deve-se armazenar o número do ISSN (*International Standard Serial Number* - Número Internacional Normalizado para Publicações Seriadas). Estes números não devem ficar armazenados na tabela Obra, uma vez que obras não possuem ambos os números simultaneamente e também não se deseja que fiquem nulos dependendo do tipo da obra.

2. Periodicamente, a biblioteca faz compra de novas obras. Para isso, são abertas requisições onde são registradas as obras a serem adquiridas, bem como sua quantidade. Desta forma, uma obra pode estar em várias requisições e uma requisição pode possuir diversas obras. Requisições ainda possuem um número seqüencial que as identificam, uma data de abertura e um campo que indica o estado da requisição, podendo estar aberta ou fechada para inclusão de novas obras. Quando do momento da compra, cada obra da requisição pode ser adquirida de um fornecedor diferente, enquanto um fornecedor pode fornecer várias obras de diferentes requisições. Para isso, deve-se considerar uma nova entidade chamada Fornecedor, que possui código, razão social, e telefone de contato. Vale ressaltar ainda que fornecedores somente são definidos quando da compra de cada obra na requisição, associando então seu preço e data de compra.

Para estes novos requisitos, primeiro deve-se pensar na solução através do modelo conceitual, para depois trabalhar no modelo lógico. Na transformação do modelo conceitual para o lógico, deve-se considerar que agregados tendem a gerar novas tabelas e relacionamentos de generalização/especialização podem gerar tabelas para cada entidade, para cada ramo na hierarquia ou para a hierarquia completa, sendo o mais comum gerar uma tabela para cada entidade.

O primeiro requisito desta seção torna necessário fazer uma distinção entre tipos de obras, que agora podem ser Livros ou Periódicos. Para Livros, deve-se armazenar o número do ISBN, enquanto que, para periódicos, deve-se armazenar o número do ISSN. Estes números não devem ficar armazenados na tabela Obra, uma vez que obras não possuem ambos os números simultaneamente e também não se deseja que fiquem nulos dependendo do tipo da obra.

Problemas deste tipo podem ser resolvidos utilizando uma estrutura Generalização/Especialização, que representa situações onde determinadas entidades são tratadas como especializações de uma mais genérica, que agrupa o que é comum às entidades da hierarquia. Estruturas deste tipo podem ser mapeadas em uma única tabela, em uma tabela para cada especialização ou uma tabela para cada entidade envolvida, dependendo da situação. A **Figura 17** apresenta estas situações de mapeamento para o problema em questão. Na situação 1, todos os

atributos da hierarquia foram agrupados numa única tabela, fazendo com que atributos fiquem eventualmente vazios em função do tipo da obra. A situação 2 representa apenas as especializações, fazendo com que os atributos da generalização sejam repetidos. A situação 3 apresenta uma tabela para cada entidade da hierarquia, eliminando atributos repetidos, mas fazendo com que o acesso a uma obra tenha que agrupar dados de mais de uma tabela. Neste estudo de caso, será utilizada a situação 3, onde cada entidade é mapeada para uma tabela específica, não havendo redundância de dados ou atributos com valores nulos em função do tipo de obra em questão, atendendo assim ao requisito proposto.

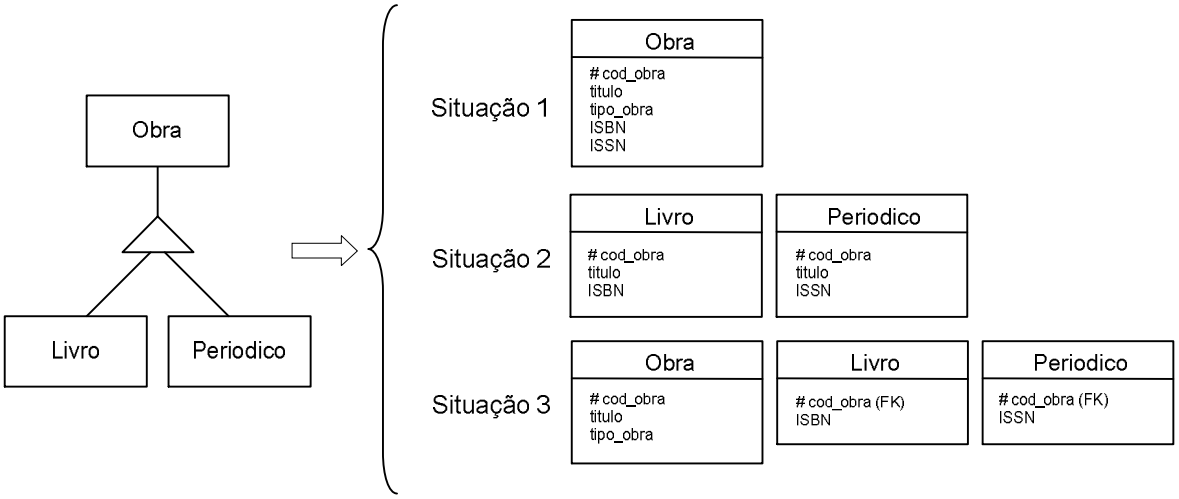

Figura 17. Mapeamento de Estruturas Generalização/Especialização

O próximo requisito trata da questão de compras de obras pela biblioteca. Periodicamente, a biblioteca faz compra de novas obras. Para isso, são abertas requisições onde são registradas as obras a serem adquiridas, bem como sua quantidade. Desta forma, uma obra pode estar em várias requisições e uma requisição pode possuir diversas obras. Requisições ainda possuem um número seqüencial que as identificam, uma data de abertura e um campo que indica o estado da requisição, podendo estar aberta ou fechada para inclusão de novas obras. Quando do momento da compra, cada obra da requisição pode ser adquirida de um fornecedor diferente, enquanto um fornecedor pode fornecer várias obras de diferentes requisições. Para isso, deve-se considerar uma nova entidade chamada Fornecedor, que possui código, razão social, e telefone de contato. Vale ressaltar

ainda que fornecedores somente são definidos quando da compra de cada obra na requisição, associando então seu preço e data de compra.

Como uma obra pode ser alocada a uma requisição sem um fornecedor num primeiro momento, um relacionamento ternário não se mostraria adequado, uma vez que a tabela originada teria também o fornecedor como parte da chave primária, e este não poderia ficar nulo. Desta forma, esta situação pode ser resolvida com uma agregação, que normalmente origina uma nova tabela. A **Figura 18** representa uma agregação para este problema apresentado.

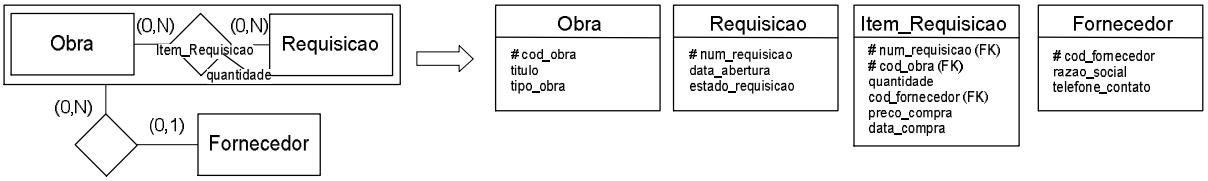

Figura 18. Mapeamento de Agregação

Percebe-se na tabela Item_Requisicao que o atributo cod_fornecedor é apenas uma chave estrangeira, não fazendo parte da chave primária. Isso irá permitir que o mesmo possa ter o valor nulo enquanto uma obra de uma requisição não passar pelo processo de compra.

Desta forma, a **Figura 19** apresenta o modelo conceitual após as modificações realizadas, enquanto a **Figura 20** exibe o modelo lógico correspondente.

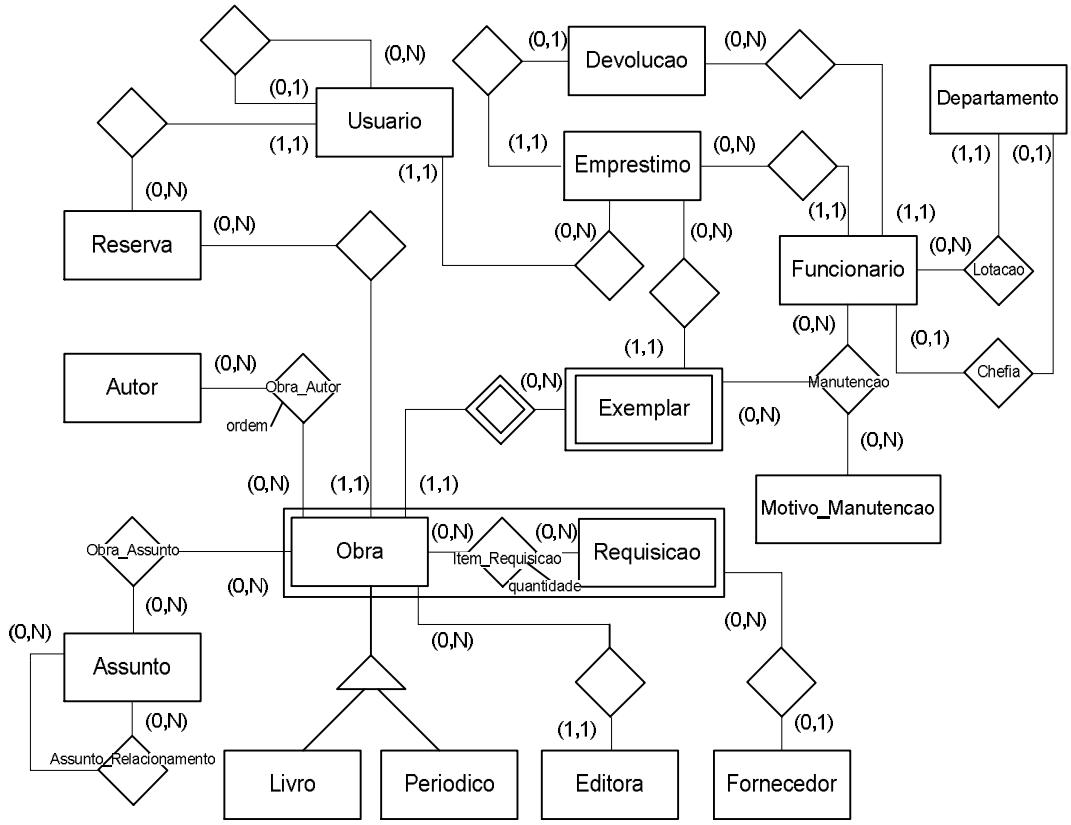

Figura 19. O modelo conceitual após as modificações

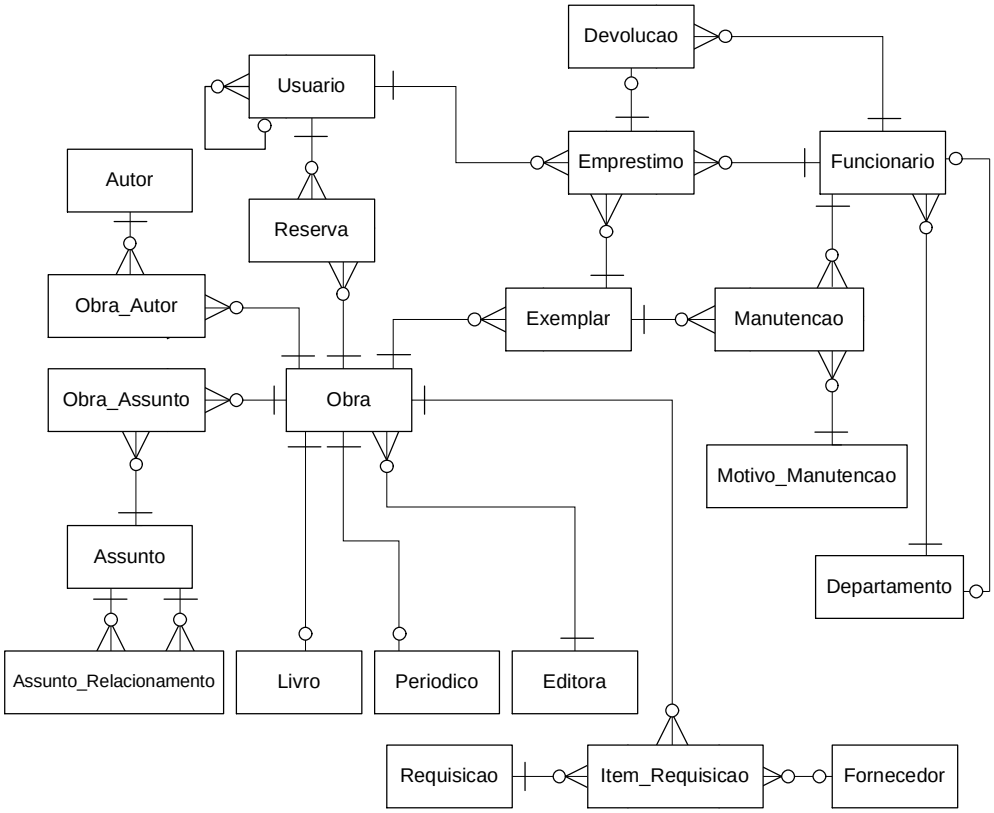

Figura 20. O modelo lógico após as modificações

As **Tabelas 20** a **24** mostram as estruturas das tabelas que sofreram modificações em função dos novos requisitos.

Tabela 20. Estrutura da tabela Livro

Livro						
Chave		**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	FK	cod_obra	Inteiro		Sim	Integridade referencial com tabela Obra
		ISBN	Inteiro		Sim	

Tabela 21. Estrutura da tabela Periodico

Periodico						
Chave		**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	FK	cod_obra	Inteiro		Sim	Integridade referencial com tabela Obra
		ISSN	Inteiro		Sim	

Tabela 22. Estrutura da tabela Fornecedor

Fornecedor					
Chave	**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	cod_fornecedor	Inteiro		Sim	
	razao_social	Texto	50	Sim	
	telefone_contato	Texto	15	Não	

Tabela 23. Estrutura da tabela Requisicao

Requisicao					
Chave	**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	num_requisicao	Inteiro		Sim	
	data_abertura	Data		Sim	
	estado_requisicao	Inteiro		Sim	1=Aberta 2=Fechada

Tabela 24. Estrutura da tabela Item_Requisicao

Item_Requisicao						
Chave		**Atributo**	**Tipo**	**Tamanho**	**Obrigatório**	**Restrições**
PK	FK	num_requisicao	Inteiro		Sim	Integridade referencial com tabela Requisicao
	FK	cod_obra	Inteiro		Sim	Integridade referencial com tabela Obra
		quantidade	Integer		Sim	
	FK	cod_fornecedor	Integer		Não	Integridade referencial com tabela Fornecedor
		preco_compra	Real		Não	
		data_compra	Data		Não	

Percebe-se nas tabelas Livro e Periodico que ambas possuem como chave primária o campo “cod_obra”, o mesmo da tabela que representa a generalização. Esta é uma característica de estruturas generalização/especialização, onde todas as tabelas da hierarquia possuem a mesma chave primária, aquela definida na tabela mais genérica. Usualmente, ainda coloca-se um atributo na tabela que representa a generalização, neste caso a tabela Obra, para indicar a tabela que possui a complementação de seus dados, Livro ou Periodico neste caso. Isto não precisou ser feito neste momento, pois a tabela Obra já possuía um atributo chamado “tipo_obra”, justamente para este fim.

A tabela Item_Requisicao representa a agregação do modelo conceitual. Percebe-se, neste caso, que o relacionamento entre as entidades Obra e Requisicao é de cardinalidade n:n, fazendo com que esta tabela surgisse de qualquer forma. A particularidade aqui se refere à ligação desta tabela Item_Requisicao com a tabela Fornecedor, de forma opcional, oferecendo a possibilidade de um item de requisição ainda não possuir um fornecedor. Isto fica evidenciado no modelo conceitual uma vez que o relacionamento com a tabela Fornecedor é feito diretamente com a agregação, e não com nenhuma de suas entidades participantes, caracterizando que um fornecedor está associado com uma obra de uma determinada requisição. Esta situação não poderia ser resolvida com um relacionamento ternário entre as entidades Obra, Fornecedor e Requisição pois, neste caso, as chaves primárias das três entidades participantes comporiam a chave primária da entidade

Item_Requisicao, obrigando-a a ter um fornecedor no momento da sua criação, uma vez que um campo que faz parte de uma chave primária não pode ser nulo. Por fim, percebe-se que os atributos preco_compra e data_compra não são obrigatórios uma vez que um item de requisição pode ainda não ter sido adquirido. Uma alternativa a isso seria deslocar estes campos, bem como o código do fornecedor para uma outra tabela de compras, mas optamos por esta solução neste exemplo.

7. Considerações Finais

Este artigo apresentou, através de um estudo de caso prático, as principais técnicas de modelagem de dados, tanto a nível de utilização do modelo conceitual, através do Modelo Entidade-Relacionamento, quanto do modelo lógico, através do Modelo Relacional, visando discutir não apenas a teoria de modelagem de dados, mas também apresentar possibilidades de aplicação através de situações reais de utilização. A utilização destas técnicas é fundamental para um melhor entendimento do problema e, consequentemente, auxiliar na construção de sistemas de software mais robustos e flexíveis, objetivando atender as necessidades do usuário final.

8. Referências Bibliográficas

COUGO, P. **Modelagem conceitual e projeto de bancos de dados**. Ri de Janeiro : Campus, 1997.

DATE, C. J. I**ntrodução a sistemas de bancos de dados**. 8.ed. Rio de Janeiro : Campus, 2004.

ELMASRI, R. E.; NAVATHE, S.B. **Sistemas de banco de dados**. 4.ed. Rio de Janeiro : Addison-Wesley, 2005.

HEUSER, C.A. **Projeto de banco de dados**. 5.ed. Porto Alegre : Sagra-Luzzatto, 2004.

KORTH, H. F.; SILBERCHATZ, A.; SUDARSHAN, S. **Sistema de banco de dados**. 5.ed. Rio de Janeiro: Campus, 2006.