Banco de Dados

Aula 06 - Normalização Básica

Apresentação

Nas aulas anteriores, você estudou como modelar seu banco de dados, utilizando o Modelo Entidade Relacionamento (MER) e o Modelo Relacional (MR). Você aprendeu também como fazer o mapeamento dos conceitos do MER para o MR. Apesar de tudo isso, você pode estar se perguntando: como saber se o meu Modelo Relacional foi bem feito? Nesta aula, você vai estudar algumas técnicas básicas de como verificar se o seu Modelo Relacional foi bem definido ou não. O conjunto dessas técnicas recebe o nome de **normalização**. Nesta aula, você vai aprender como utilizar duas formas de normalização.

Vídeo 01 - Apresentação

Objetivos

- Definir o que são anomalias de inserção, de remoção e de atualização.
- Normalizar tabelas conforme a Primeira Forma Normal (1FN).
- Normalizar tabelas conforme a Segunda Forma Normal (2FN).

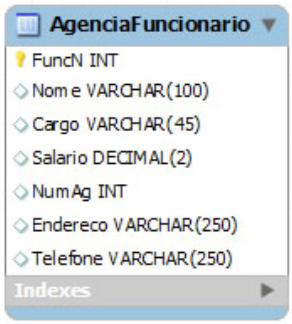
Anomalia de inserção, remoção e atualização

Antes de falar sobre normalização, vamos entender o problema de anomalia de inserção, de remoção e de atualização. Anomalias são mudanças em dados que podem gerar uma inconsistência no banco de dados relacional.

Uma inconsistência é geralmente representada por situações em que dados que deveriam ser iguais, apresentam valores diferentes em várias tabelas do banco de dados. Por exemplo, o valor de compra de um produto deve ser o mesmo valor armazenado nas tabelas recibo e nota fiscal. Desse modo, se os valores forem diferentes, entende-se geralmente que existe uma inconsistência.

Tal inconsistência geralmente aparece quando o banco de dados é projetado de forma inadequada. Para exemplificar, usaremos a **Figura 1**, que mostra os tipos mais comuns de anomalias.

Figura 01 - Exemplo de tabela AgenciaFuncionario.



FuncN	Nome	Cargo	Salário	NumAg	Endereço	Tel
25	Luiz	Caixa	2000	1632	Prudente de Morais, 15	3605-5223
30	Ricardo	Gerente	5000	1668	Hermes da Fonseca, 20	3605-5223
31	Josita	Caixa	2000	1632	Prudente de Morais, 15	4225-5889
32	Francisca	Gerente	5600	1632	Prudente de Morais, 15	4225-5889
33	Andréia	Caixa	2300	1668	Hermes da Fonseca, 20	5223-8556

Fonte: Connolly e Begg (2000).

Note que mostramos, na parte superior, a estrutura da tabela (como você está acostumado a ver) e, na parte inferior da figura, exemplos de valores para os registros. Os valores são importantes para facilitar o entendimento dos cenários que

produzem as anomalias.

Assim, na **Figura 1**, temos uma tabela chamada AgenciaFuncionario, que armazena os dados dos funcionários e de agências bancárias. Como essas duas informações (funcionário e agência) são armazenadas na mesma tabela, é possível também saber onde cada funcionário trabalha (agência, endereço, telefone). Ou seja, à primeira vista, a tabela AgenciaFuncionario parece ser uma ótima opção para reduzir o número de tabelas e aumentar a velocidade do sistema, entretanto, tal solução pode gerar várias anomalias.

Anomalia de inserção

Uma anomalia de inserção acontece quando, ao inserir um dado, este pode gera uma inconsistência no banco de dados. No exemplo da **Figura 1**, para inserir os detalhes (NumAg, Endereço, Tel.) de um novo funcionário, você deve tomar cuidado para usar exatamente os dados já cadastrados por outros funcionários. Por exemplo, um novo funcionário para a agência 1550 deve usar exatamente o mesmo endereço dos outros dois funcionários que também trabalham nessa agência. Se isso não for feito, teremos um problema de inconsistência de dados, no qual dois funcionários que trabalham na mesma agência possuem endereços diferentes.

Anomalia de remoção

Uma anomalia de remoção acontece quando, ao remover um registro, você pode gerar inconsistência no banco de dados. No exemplo da **Figura 1**, uma anomalia de remoção acontece quando removemos o funcionário de número 30. Nesse caso, o objetivo é apenas remover os dados do funcionário e preservar os dados da agência 1668. Entretanto, da forma como a tabela está estruturada, os dados da agência também são removidos.

Anomalia de atualização

Uma anomalia de atualização acontece quando, ao atualizar um registro, você pode gerar inconsistência no banco de dados. No exemplo da **Figura 1**, uma anomalia de atualização acontece quando modificamos o endereço da agência 1632. Nesse caso, teremos que atualizar o endereço de todos os funcionários da agência 1632. Caso contrário, teremos uma inconsistência no banco de dados no qual funcionários da mesma agência possuem endereços diferentes. Note que todas as

anomalias acontecem devido à existência de redundância de informação na tabela AgenciaFuncionario. Por exemplo, o mesmo endereço de uma agência é armazenado várias vezes quando ele poderia ser armazenado apenas uma vez. Assim, para evitar as anomalias é preciso evitar a redundância. A redundância é evitada através da normalização das tabelas, que você verá como se faz logo a seguir.

Normalização

O processo de normalização foi proposto por Dr. E. F. Codd como uma forma de evitar as anomalias mostradas anteriormente. Assim, o objetivo da normalização é remover a duplicação de dados e, consequentemente, minimizar a redundância. Segundo Powell (2006), a remoção da duplicação de dados permite:

- Reduzir o espaço físico necessário para armazenar o banco de dados;
- Melhorar a organização dos dados;
- Reduzir o impacto de atualizações, inserções e remoções nos dados do banco de dados.

O processo de normalização é constituído por um conjunto de formas normais. As formas normais especificam critérios que definem quando uma tabela está bem estruturada ou não. Assim, para saber se uma tabela está bem estruturada, você deve verificar se a estrutura da tabela satisfaz todas as formas normais.

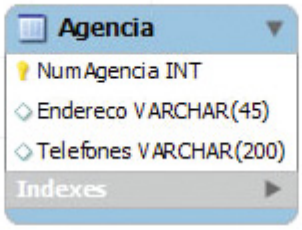
Nesta aula, você verá duas formas normais que são bem definidas na literatura de banco de dados. Veremos também que ações são necessárias para que uma tabela satisfaça cada forma normal. Ou seja, o que fazer para consertar a estrutura de uma tabela de modo que a mesma satisfaça a forma normal.

Primeira Forma Normal (1FN)

Uma tabela está na Primeira Forma Normal (1FN) se e somente se todos os atributos contiverem apenas dados atômicos. Ou seja, cada atributo pode ter apenas um valor por registro (tupla).

A **Figura 2** mostra um exemplo de uma tabela que não está na 1FN. Esta tabela não está na 1FN porque o atributo Telefones possui mais de um telefone por registro (tupla). Por exemplo, a agência 1524 possui três telefones.

Figura 02 - Exemplo de estrutura de tabela que não está na 1FN.



NumAg	Endereço	Tel
1524	Prudente de Morais, 12, RN	3605-5223, 3605-5141, 3605-5142
1550	Hermes da Fonseca, 15, RN	4225-5889, 4225-5890
2051	Eng. Roberto Freire, 20, RN	5223-8556, 5223-8557

Fonte: Connolly e Begg (2000).

Para adequar uma tabela que não está na 1FN é necessário realizar os seguintes passos:

1. Criar uma tabela para conter os dados do atributo não atômico;
2. Criar na nova tabela um atributo para conter o atributo não atômico da tabela original;
3. Criar na nova tabela um atributo para conter a chave primária da tabela original;

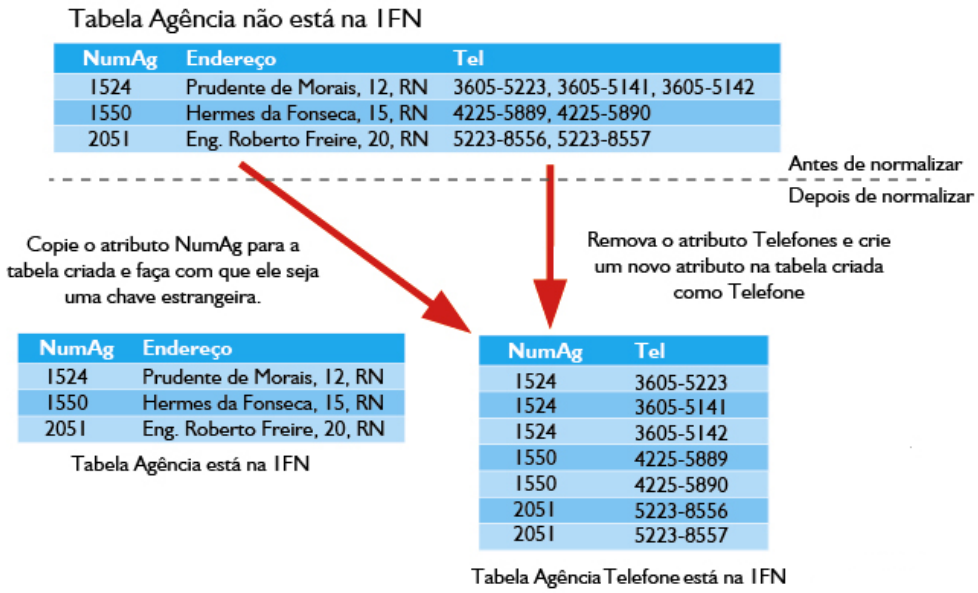
4. Definir uma chave estrangeira para garantir a relação entre a nova tabela e a tabela original;

5. Definir a chave primária da nova tabela;

6. Remover o atributo não atômico da tabela original.

Os passos listados são ilustrados na Figura 3, utilizando-se o exemplo da tabela Agência.

Figura 03 - Normalização da Tabela Agência.

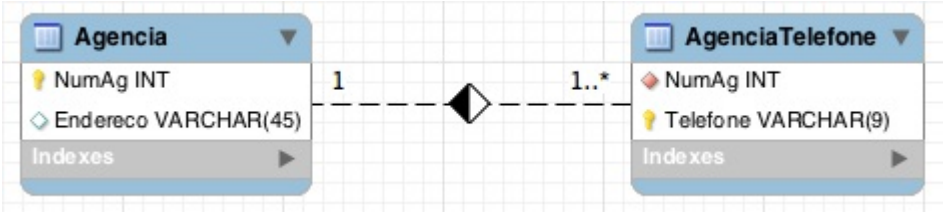


Fonte: Connolly e Begg (2000).

O primeiro passo é criar uma nova tabela chamada AgenciaTelefone. Depois criar o atributo não atômico da tabela Agência, no nosso caso, o atributo Telefone. Em seguida, você deve criar na tabela AgenciaTelefone o atributo chave da tabela Agencia, no nosso caso, o atributo NumAg. Para manter a integridade com a tabela original, você deve definir uma chave estrangeira entre o atributo NumAg da tabela AgenciaTelefone e NumAg da tabela Agência. Posteriormente, você deve definir a chave primária da tabela AgenciaTelefone. No nosso caso, como não podemos ter Agências diferentes com o mesmo telefone, definimos como chave primária da tabela AgenciaTelefone o atributo Telefone. Finalmente, você precisa remover o atributo não atômico da tabela Agência.

Para perceber a diferença, veja na **Figura 4** a nova estrutura das duas tabelas.

Figura 04 - Estrutura das tabelas depois da Normalização.



Note que o processo de normalização de uma tabela para a 1FN é equivalente ao mapeamento de um atributo multivalorado do modelo ER para o modelo relacional, mostrado na Aula 4. Ou seja, se você definir bem seu modelo ER e fizer o mapeamento correto, suas tabelas já estarão na 1FN.

Vídeo 02 - Primeira Forma Normal

Atividade 01

1. Descreva como a Normalização pode ser utilizada no projeto de banco de dados.
2. Descreva as características das tabelas que violam a 1FN e como tais tabelas podem ser modificadas para satisfazer a 1FN.

Segunda Forma Normal (2FN)

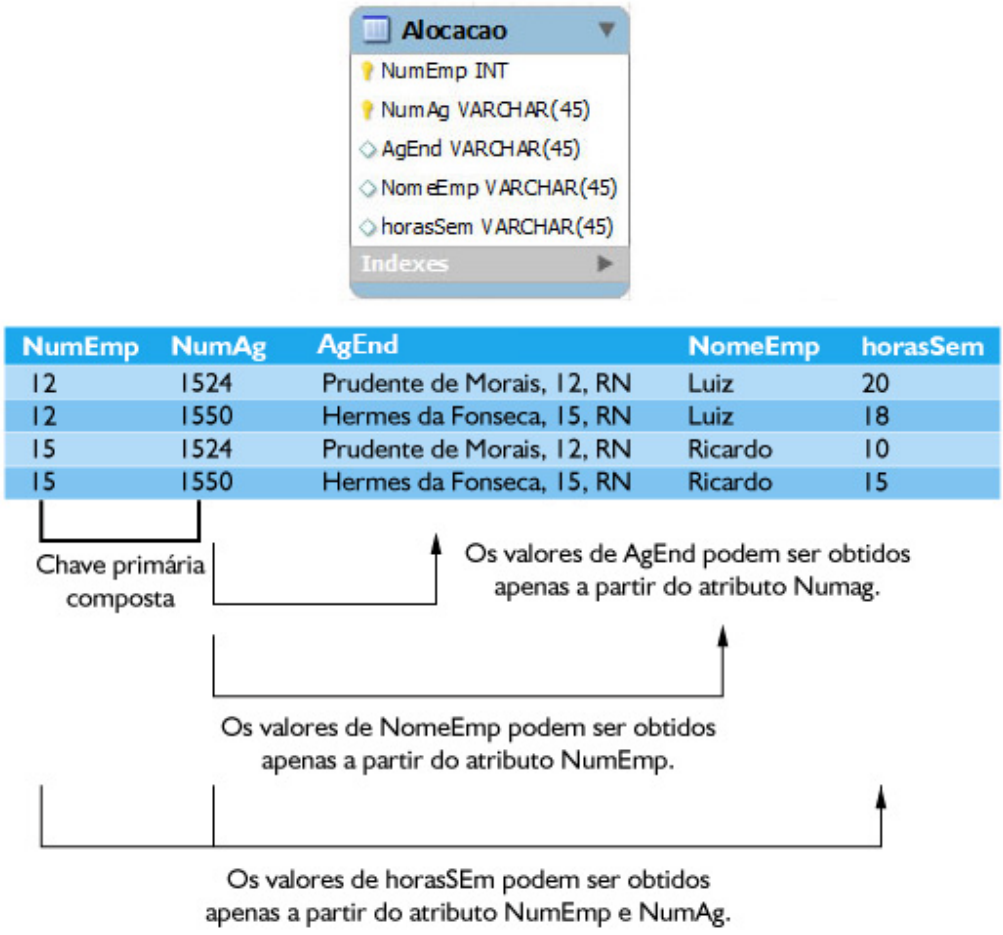
Vídeo 03 - Dependência Funcional

Vídeo 04 - Segunda Forma Normal

Uma tabela está na Segunda Forma Normal (2FN) se e somente se ela estiver na 1FN e todos os atributos **não** chave primária puderem ser obtidos da combinação de **todos** os atributos que formam a chave primária.

Para ilustrar o significado da 2FN, vamos observar a Figura 5, adaptada de (CONNOLLY; BEGG, 2000).

Figura 05 - Exemplo de tabela que não está na 2FN.



Fonte: Connolly e Begg (2000).

Nessa figura, a tabela Alocação armazena as horas trabalhadas por funcionários temporários em determinadas agências de um banco. Na parte superior é mostrada a estrutura da tabela, na qual é possível identificar que a sua chave primária é formada pelos atributos NumEmp e NumAg. Além desses dois atributos, a tabela Alocação possui mais três atributos que não fazem parte da chave primária. São eles: AgEnd, NomeEmp e horasSem.

Como você viu anteriormente, para estar na 2FN, todos esses três atributos não chave (AgEnd, NomeEmp e horasSem) devem ser obtidos através dos **dois** atributos chaves (NumEmp e NumAg). Se for possível obter um atributo não chave primária através de apenas um dos atributos chave primária, então a tabela não está na 2FN.

Por exemplo, eu consigo saber o endereço da agência (atributo AgEnd) apenas através do atributo NumAg? A resposta é sim. O endereço da agência é uma informação intrínseca à agência e pode ser obtido através do atributo NumAg.

Vejamos outro exemplo. Eu consigo obter o nome do empregado (NomeEmp) através do código do empregado (NumEmp)? Sim, esta informação é intrínseca ao empregado e, através do atributo NumEmp, eu poderia obter seu nome. Finalmente, eu poderia obter as horas semanais trabalhadas pelo empregado apenas através de seu NumEmp? Não, isso acontece porque as horas semanais de cada empregado dependem da agência onde ele trabalha. Assim, o atributo horasSem depende dos dois atributos NumEmp e NumAg que compõem a chave primária para ser obtido.

Para concluir o exemplo, temos que a tabela Alocação mostrada na **Figura 5 não** está na 2FN porque possui pelo menos um atributo que pode ser obtido de apenas um dos atributos que formam a chave primária. Esse é o caso do atributo NomeEmp, que pode ser obtido apenas de NumEmp, e um outro exemplo é o caso do atributo AgEnd, que pode ser obtido apenas do atributo NumAg.

O fato de que uma tabela não está na 2FN pode gerar as anomalias mostradas anteriormente. Por exemplo, para modificar o endereço da agência de número 1550 você teria que modificar dois registros: o segundo e quarto registro na tabela da **Figura 5**.

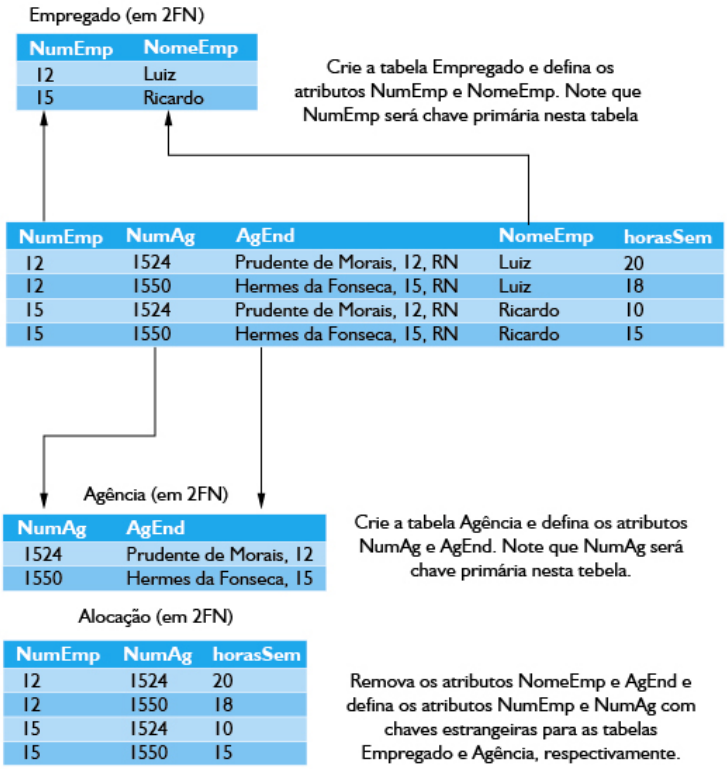
Bem, depois de aprender como identificar se uma tabela está ou não na 2FN, agora vamos aprender com reestruturar uma tabela de modo que ela fique na 2FN. Para adequar uma tabela que não está na 2FN é necessário fazer os seguintes

passos:

1. Criar duas novas tabelas para armazenar os dados dos campos redundantes, em que seus valores apresentam repetição de valores;
2. Remover os campos com valores redundantes da tabela original;
3. Criar chaves primárias nas novas tabelas criadas com base na chave primária da tabela original;
4. Criar relações um-para-muitos entre as novas tabelas criadas e a tabela original.

Os passos listados anteriormente são ilustrados na **Figura 6**, utilizando-se o exemplo da tabela Alocação.

Figura 06 - Exemplo de normalização da tabela Alocação para a 2FN.



Fonte: Connolly e Begg (2000).

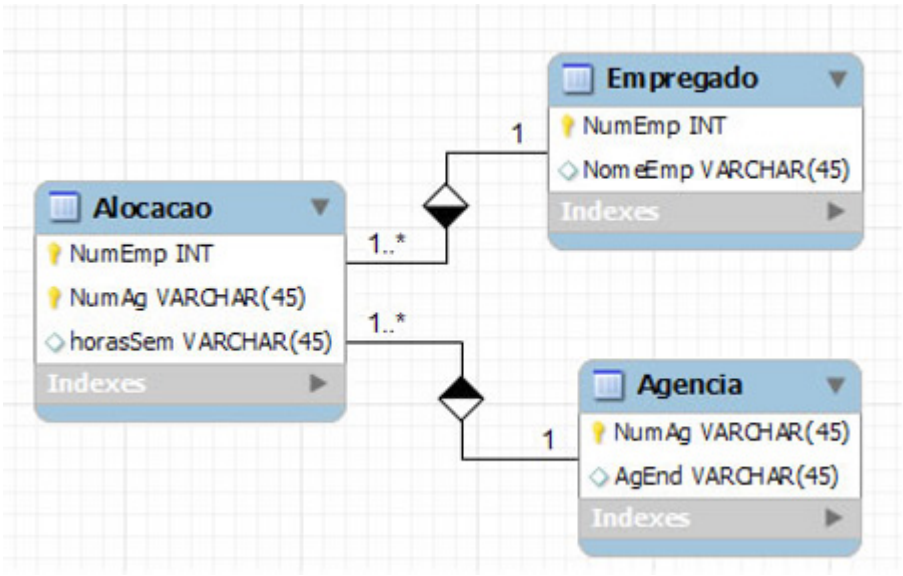
A primeira coisa a fazer é criar as tabelas Empregado e Agência. Essas tabelas irão armazenar os dados intrínsecos de Empregado e Agência que, na **Figura 5**, estavam na tabela Alocação.

Depois disso, você precisa definir as chaves primárias das tabelas Empregado e Agência. Para saber qual campo será a chave primária das tabelas criadas, você deve olhar para a chave primária da tabela Alocação e fazer a pergunta: qual o campo da chave primária da tabela Alocação está relacionado ao Empregado? Qual está relacionado a Agência? A resposta para essas perguntas é, respectivamente, NumEmp e NumAg. Assim, NumEmp será a chave primária da tabela Empregado e NumAg será a chave primária da tabela Agência.

Em seguida, você deve inserir nas tabelas criadas os seus atributos relacionados. Por exemplo, NomeEmp vai para a tabela Empregado e AgEnd vai para a tabela Agência. O próximo passo é remover os atributos NomeEmp e AgEnd da tabela Alocação.

Note que na tabela Alocação só devem ficar os atributos que só podem ser obtidos pela combinação dos atributos que formam a chave primária. Assim, a tabela Alocação só ficará com os atributos que formam a chave primária (NumEmp e NumAg) e o atributo horasSem. Para finalizar, você deve definir os atributos NumEmp e NumAg como chaves estrangeiras para as tabelas Empregado e Agência, respectivamente. Finalmente, a **Figura 7** mostra a estrutura das tabelas que estão na 2FN.

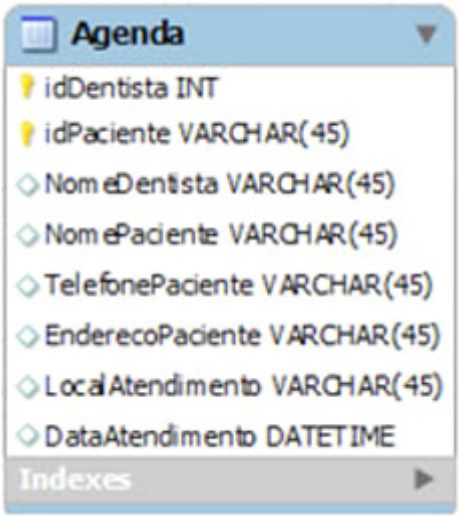
Figura 07 - Exemplo de tabelas que estão na 2FN.



Atividade 02

1. Descreva as características das tabelas que violam a 2FN e como tais tabelas podem ser modificadas para satisfazer a 2FN.
2. Verifique se a Tabela está na 2FN (Figura 8). Caso ela não esteja, faça o processo de adequação como mostrado anteriormente.

Figura 08 - Tabela Agenda.



Conclusão

Agora você já sabe como evitar algumas anomalias através de um processo de normalização.

Busque praticar as regras e aplicar nos próximos projetos de bancos de dados que você fizer.

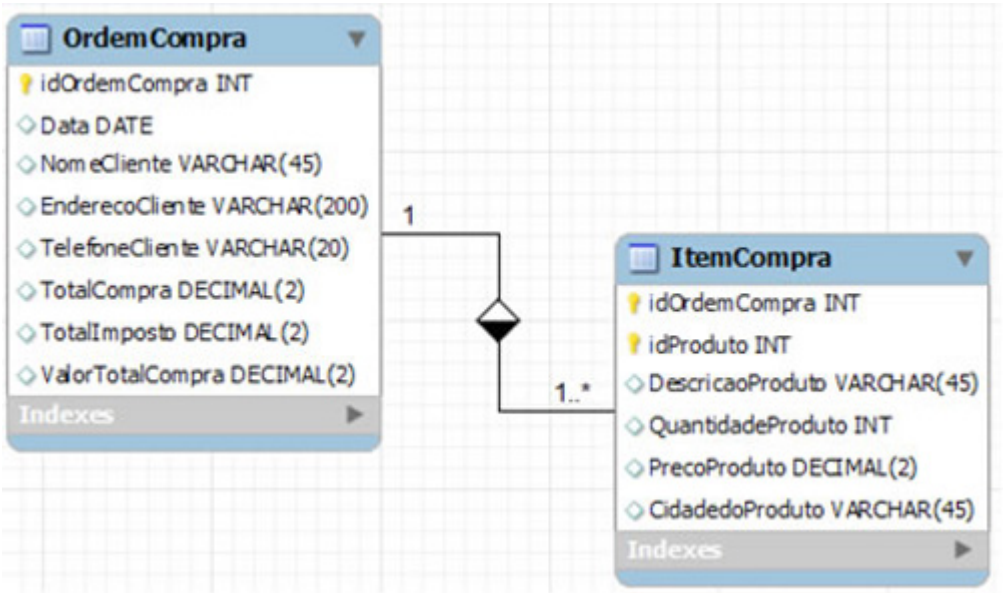
Resumo

Nesta aula, você aprendeu que uma tabela incorretamente definida pode gerar anomalias de inserção, de modificação e de remoção. Você verificou que a existência de anomalias pode gerar inconsistência no banco de dados e comprometer sua utilização. Para eliminar as anomalias, você aprendeu como normalizar tabelas através da 1FN e 2FN. Na próxima aula, você vai estudar formas avançadas de normalização.

Autoavaliação

1. Explique o que é redundância de dados.
2. Cite as principais características das anomalias de inserção, de remoção e de atualização.
3. Dada as tabelas OrdemCompra e ItemCompra, verifique se elas estão na 2FN. Caso não estejam, faça o processo de adequação dessas tabelas (Figura 9).

Figura 09 - Tabelas OrdemCompra e ItemCompra.



Referências

CONNOLLY, Thomas M.; BEGG, Carolyn E. **Database Solutions:** a step-by-step approach to building databases. 2nd ed. New Jersey: Pearson Education Limited, 2000.

DATE, Christopher J. **Introduction to Database Systems.** 7th ed. 1999.

_____. **Introdução a sistemas de banco de dados.** Rio de Janeiro: Campus, 2000.

HEUSER, C. A. **Projeto de banco de dados.** 6. ed. Porto Alegre: Editora Bookman, 2009.

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

POWELL, Gavin. **Beginning Database Design.** San Francisco: Wiley Publishing, 2006.