**Unidade 4**

Normalização de dados

**Normalização de dados na computação**

Em um projeto de banco de dados existe uma regra fundamental: nunca misturar assuntos diferentes em uma mesma tabela.

Precisamos lutar contra dois aspectos chaves: redundância de dados e inconsistência de dados.

A modelagem de dados tem como sua essência mais pura o refinamento de processos

Evitar redundância e inconsistência;

Às vezes é interessante manter uma informação redundante no banco de dados. Por questões de desempenho de alguma pesquisa ou software a redundância pode ocorrer, fato esse que é chamado de redundância controlada e é recomendado quando o campo recebe uma grande quantidade de consultas e poucas alterações. Um exemplo é uma tabela de nota fiscal, o campo “valor\_total\_ da\_nota”, que poderia ser obtido automaticamente, como o resultado da soma de todos os itens vendidos e multiplicados pelo seu preço. Entretanto, é comum criarmos manualmente o campo “valor\_total\_da\_nota” para evitar que, caso haja alteração do preço do produto vendido, o valor da nota não seja alterado (causando problemas contábeis). É uma redundância que sabemos que existe, mas é necessária.

Segundo Coronel e Rob (2011), a normalização é uma técnica para avaliar e corrigir estruturas e tabelas ao modo de tornar mínimas as redundâncias de dados, reduzindo assim as chances de haver problemas. Normalizar um banco de dados é aplicar regras para todas as suas tabelas, com os objetivos, além de reduzir a redundância e eliminar campos que não dizem respeito à tabela.

Podemos listar alguns objetivos e vantagens da normalização de um esquema de banco de dados:

1. Diminuição de dados repetitivos deixando o banco de dados mais compacto.
2. Aumento da performance no Sistema Gerenciador de Banco de Dados.
3. Armazenamento dos dados de forma lógica.
4. Facilidade na criação de consultas.
5. Permite a concatenação de índices (chaves) de acordo com a quantidade de tabelas envolvidas.
6. Facilidade na manutenção do banco de dados.

Korth, Silberschatz e Sudarshan (2012) afirmam que as primeiras técnicas de normalização foram criadas em 1972 por Edgar Frank Codd. Após ter dado o primeiro passo, Codd propôs, junto de Raymond Boyce, um novo significado, que ficou conhecido como Forma Normal Boyce-Codd (ou FNBC). Todas elas se baseiam na dependência funcional entre os atributos de uma entidade do banco de dados e nas chaves primárias.

Segundo Coronel e Rob (2011), as formas normais mais populares, são:

1. A primeira forma normal ou 1FN.
2. Uma tabela encontra-se na 1FN quando não contém tabelas aninhadas. Portanto, a passagem à 1FN consta da eliminação das tabelas aninhadas eventualmente existentes.

Para estar na 1FN os seguintes passos devem ser realizados:

1. Identificar a chave primária da tabela.
2. Identificar a coluna que possua dados repetidos.
3. Remover a coluna que tenha dados repetidos.
4. Criar uma nova tabela para armazenar os dados repetidos.
5. Criar um relacionamento entre a tabela que está sendo normalizada e a sua tabela secundária.
6. Para deixar a tabela Funcionário na Primeira Forma Normal (1FN) primeiramente devemos verificar: alguns dos campos podem ser a chave primária?
7. Pelo que observamos, é melhor criar um campo novo e indicar como a chave primária
8. A segunda forma normal ou 2FN.
9. Uma tabela encontra-se na segunda forma normal (2FN) quando, além de encontrar-se na primeira forma normal, cada coluna não chave depende da chave primária completa
10. . Uma tabela que não se encontra na segunda formal contém dependências funcionais parciais, ou seja, contém colunas não chave que dependem apenas de uma parte da chave primária.
11. Uma tabela está na 2FN se, e somente se, estiver na 1FN e todas as suas colunas, que não são chaves, dependam exclusivamente da chave primária.

Para estar na 2FN devemos aplicar as seguintes ações:

1. Identificar as colunas que não são funcionalmente dependentes da chave primária da tabela.
2. Remover o campo da tabela e criar uma nova tabela com esses dados.
4. A terceira forma normal ou 3FN.
5. Uma tabela encontra-se na 3FN quando, além de estar na 2FN, toda coluna não chave depende
6. diretamente  de chave primária, isto é, quando não há dependências funcionais transitivas
7. ou indiretas
8. A quarta forma normal ou 4FN.
9. Uma tabela está na 4FN caso, além de estar na 3FN, não possua maisque uma dependência funcional multi-valorada.

o objetivo de averiguar nas tabelas a existência de redundância ou dependências funcionais.

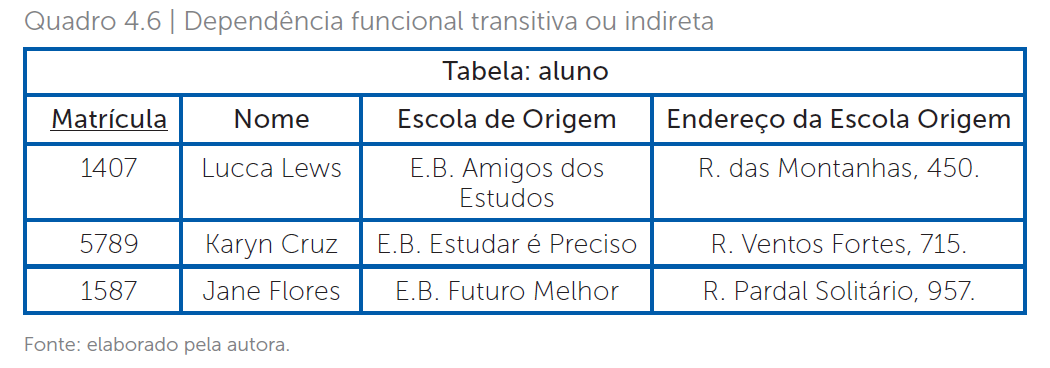
Para compreender as formas normais, que serão apresentadas nas próximas seções, precisamos primeiramente compreender o significado de dependência funcional, que nada mais é consistir em uma restrição entre dois ou mais conjuntos de atributos de uma mesma tabela ou relacionamento, conforme afirma Alves (2014).

A dependência *X→ Y* ®ocorrerá se a cada valor de X estiver associado um e somente um valor de Y. Um exemplo de dependência funcional é a tabela *Aluno*. Nela temos a matrícula e o nome do aluno. O nome do aluno depende diretamente da matrícula do aluno.

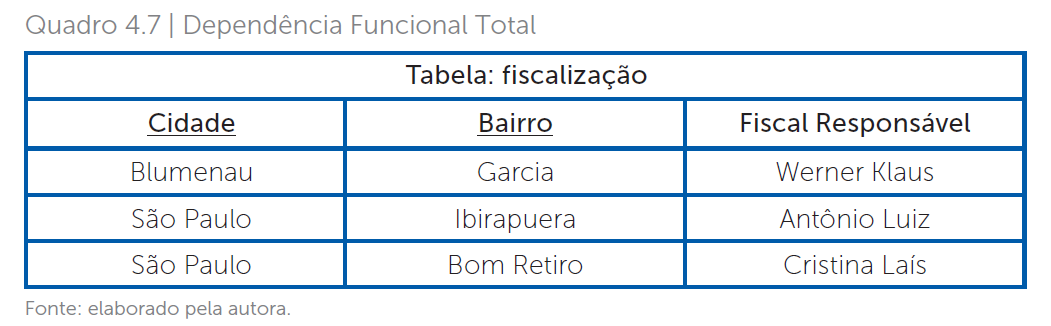
A dependência funcional pode ser classificada em: transitiva ou indireta, total ou parcial.

A dependência funcional transitiva acontece quando um determinado campo da tabela, além de depender da chave primária da tabela, depende também de outro campo ou de outros campos que são integrantes da mesma tabela.

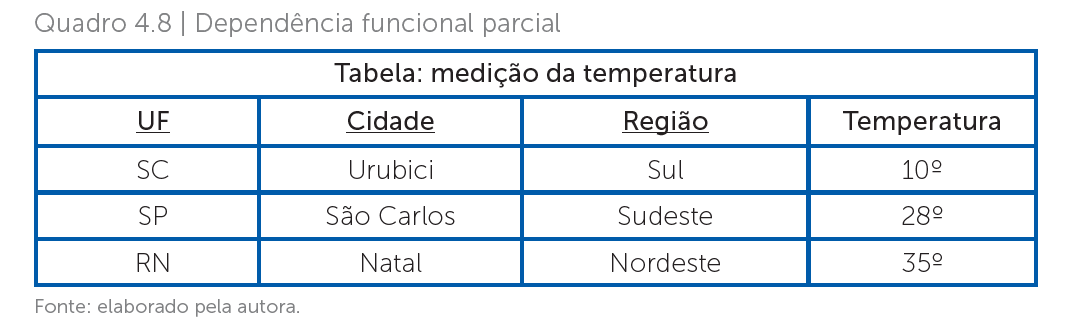
A dependência transitiva ocorre quando Y depende de X e Z depende de Y; logo, Z também depende de X. Podemos representar esse tipo de dependência como: *X →Y→ Z* .

**

A dependência funcional total ou completa (como também é conhecida) ocorre quando um atributo que não faz parte da chave primária depende diretamente de todos os outros atributos que fazem parte da chave primária. Sempre ocorre quando a tabela possui chaves concatenadas (mais de uma chave primária).

**

A dependência funcional parcial ocorre quando um campo ou atributo que não faz parte da chave primaria tem dependência funcional de apenas alguns dos atributos que fazem parte da chave primária.

**

**Transformação 1FN - 2FN**

Para atingir os objetivos da normalização, conforme Coronel e Rob (2011), as tabelas precisam ter as seguintes propriedades:

1. Cada tabela deverá tratar de somente um único assunto
2. Por exemplo: uma tabela com informações sobre remédio, não poderá ter informações de um médico.
3. O mesmo campo não poderá ser armazenado, desnecessariamente, em mais de uma tabela.
4. Essa é uma garantia de que não será necessária a atualização do mesmo campo, em mais de uma tabela.
5. Os campos de uma tabela são dependentes da chave primária dessa tabela e de mais nenhum campo.
6. A tabela deverá estar livre de anomalias de inserção, atualização e exclusão, garantindo a integridade e a consistência dos dados,
7. Por exemplo: na tabela *Cliente* será necessário informar a cidade de seu nascimento, não devemos deixar ele informar a cidade, mas escolher a cidade dentre as cidades previamente cadastradas ou por meio de uma busca do CEP (que trará o endereço completo).

Para aplicar as regras da normalização, um dos alvos a ser observado são os campos (ou atributos) que fazem parte das tabelas. Podemos classificar os atributos, conforme Korth, Silberschatz e Sudarshan (2012), de um Modelo Entidade- Relacionamento como:

1. Atributo simples ou atômico: é o atributo que não é divisível, possui um sentido único, como o RG ou o CPF de uma pessoa, esses dois exemplos mostram que tanto o RG quanto o CPF não podem ser divididos em dois outros campos.
2. Atributo composto: é um atributo que pode ser divido em várias partes, um bom exemplo é o endereço. Podemos dividir esse atributo em: rua, número, complemento, bairro.
3. Atributo monovalorado: é um atributo que possui apenas um valor para a tabela, como a matrícula de um aluno, esse número não poderá se repetir na tabela.

**54**

1. Atributo multivalorado: é um atributo que pode receber mais de uma informação, o melhor exemplo é o telefone que pode receber mais de um valor.
2. Atributo derivado: o valor desse tipo de atributo pode originar de outra tabela ou de outros campos. Digamos que para um cardiologista seja necessário saber a idade (em anos e dias). Podemos calculá-la a partir da data de nascimento e da data de atendimento no ato da consulta médica.
3. Atributo chave: é o atributo escolhido ou criado para que possa indicar o registro (a linha) da tabela.

**Transformação 3FN - 4FN**

Conforme afirmam Navathe e Ramez (2005), o procedimento de normalização proporciona a quem for modelar um banco de dados, as seguintes ações:**68**

1. Uma estrutura formal para a análise dos relacionamentos entre as tabelas, com base em suas chaves (primárias e estrangeiras) e das dependências funcionais entre os campos da tabela.
2. Um conjunto de testes de Formas Normais para ser realizado em cada esquema de relação, de forma que o modelo de banco de dados seja normalizado no grau desejado, aplicando as Formas Normais até o limite que for mais conivente para a modelagem do banco de dados.

Um banco de dados dentro das regras de normalização diminui o trabalho de manutenção e ajuda a evitar o desperdício do espaço de armazenamento.

Uma tabela estará na Terceira Forma Normal somente se estiver na Segunda Forma Normal e todos os campos forem independentes, isso quer dizer que não poderá haver dependências funcionais entre os campos e todos os campos dependem da chave primária da tabela.

Os campos da tabela precisam depender unicamente da chave primária da tabela. Para aplicar a Terceira Forma Normal é necessário:

1. Reconhecer os campos que são funcionalmente dependentes das outras colunas não chaves.
2. Eliminar as colunas dependentes.

Conforme Coronel e Rob (2011) para deixar uma tabela na Terceira Forma Normal é necessário eliminar todas as dependências transitivas, ou seja, eliminar todos os campos dependentes de outras tabelas.

Uma dependência funcional transitiva ocorre quando o valor de uma coluna é dependente de outra que não compõe a chave primária, dependendo indiretamente de outra chave primária.

A tabela somente estará na 4FN se não existir dependência multivalorada. Mas o que isso quer dizer? Dependência multivalorada é quando as informações inseridas nas tabelas podem ficar se repetindo e, claro, produzir redundâncias na tabela. Para evitar esse tipo de problema é melhor dividir a tabela evitando assim esse tipo de dependência.

Os procedimentos para deixar uma tabela na 4FN são:

1. Primeiro identificar os campos multivalorados (que causam repetições).
2. Criar uma tabela para cada grupo multivalorado.
3. Criar uma chave primária para a nova tabela.
4. Inserir a chave estrangeira na tabela que está sendo normalizada (na 4FN) para criar o relacionamento entre as tabelas.

Para aplicar a 1FN devemos inserir a chave primária nas tabelas *Jogo*, *Adversário, Estádio*.

Para aplicar a 2FN devemos criar as tabelas *Cidade* e *Estado* e inserir as chaves estrangeiras na tabela *Ficha*.

Para aplicar a 3FN devemos eliminar todos os campos dependentes

Surgem aqui as tabelas: *Jogadores*, *Campeonato*, *Técnico* e *Eventos do Jogo*.

Para aplicar a 4FN devemos verificar se não há campos multivalorados e não atômicos

A tabela *Evento Ocorrido* aparece nessa etapa, pois os eventos de um jogo se repetem muitas vezes em um jogo e em outros jogos.