NORMALIZAÇÃO

> *"O objetivo do projeto de um banco de dados relacional é gerar um conjunto de esquemas relacionais, que nos permita guardar informações sem redundância desnecessária, apesar de nos permitir recuperar a informação facilmente".*
> *(Korth e Silberschatz)*

O principal objetivo do desenvolvimento de um modelo de dados relacional é a criação de uma estrutura de dados que represente adequadamente a realidade, suas regras de negócio e restrições. Para isso, identificamos um conjunto de **relações (ou tabelas)**. Para assegurar que estas relações possuam uma estrutura correta, livre de **anomalias de atualização**, podemos seguir um conjunto de **normas**. Este processo de adequação da estrutura do banco de dados a estas normas é o que denominamos de **Normalização**.

6.1 Perigos potenciais em projetos de bancos de dados

Algumas situações indesejáveis que podem ocasionar problemas em um banco de dados relacional são:

- **Repetição de informação:**
  - Informações repetidas consomem espaço de armazenamento e dificultam a atualização. Ao se atualizar um valor, deve-se atualizar todos os valores redundantes, a fim de evitar inconsistências;
- **Incapacidade de representar parte da informação**:
  - Por vezes tem-se que incluir valores nulos;
- **Perda de informação:**
  - Ao excluir um item de dado, podemos excluir outros sem querer.

Estas situações podem levar às chamadas **anomalias de atualização**, que são repercussões indesejadas de certas ações realizadas no banco.

6.2 Dependência Funcional

Um conceito importante no estudo da normalização é o de dependência funcional. Uma **dependência funcional** é uma restrição entre dois atributos ou conjuntos de atributos de uma base de dados.

Dados os atributos "A" e "B" de uma relação R, diz-se que **"B" é funcionalmente dependente de "A" (ou "A" determina "B") se e somente se, a cada valor de "A" em R está associado um único valor de "B" em R**. Em outras palavras, se conhecermos o valor de "A" então podemos encontrar o valor de "B" associado a ele (OBS: A e B podem ser atributos compostos).

Representação de uma dependência funcional (R é o nome da relação):

$$\mathbf{A \rightarrow B} \qquad \text{ou} \qquad \mathbf{R.A \rightarrow R.B}$$

Exemplos:

PESSOA:

PESSOA.CPF → PESSOA.NOME

PESSOA.CPF → PESSOA.ENDEREÇO

PESSOA.CPF → PESSOA.DATA_NASCIMENTO

DEPARTAMENTO:

CODIGO_DEPARTAMENTO → NOME_DEPARTAMENTO

CODIGO_DEPARTAMENTO → SIGLA_DEPARTAMENTO

No primeiro exemplo (relação Pessoa), temos que o CPF determina o nome da pessoa, ou seja, o nome é funcionalmente dependente do CPF, uma vez que a cada CPF está associado um único nome. Já o contrário não é verdadeiro, ou seja, CPF não é funcionalmente dependente do nome, pois conhecendo o nome da pessoa não necessariamente poderemos determinar seu CPF (podemos ter um mesmo nome associado a mais de um CPF, no caso de duas pessoas homônimas).

6.2.1 Dependência Funcional Total (completa) e Parcial

A dependência funcional completa só ocorre quando a **chave primária for composta** por dois ou mais atributos. Ou seja, em uma tabela cuja chave primária é formada por um único atributo, este tipo de dependência não ocorre.

Dado um atributo ou um conjunto de atributos “B” de uma tabela, sendo a chave primária composta por um conjunto de atributos “A”, diz-se que “B” é completamente dependente funcional da chave primária, se e somente se, a cada valor da chave composta (e não a parte dela), está associado um valor para cada atributo do conjunto “B”.

Quando um atributo apresenta-se com dependência funcional de apenas parte da chave primária, dizemos que é uma dependência parcial.

Exemplo: na tabela PRODUTO_FATURA abaixo, os atributos QTDE_PEDIDA e PRECO_TOTAL_PRODUTO possuem dependência funcional total da chave (isto é, dependem do atributo NUMERO_PEDIDO e também do CODIGO_PRODUTO). Já o atributo DESCR_PRODUTO possui dependência funcional parcial, pois basta conhecermos apenas o código do produto para sabermos sua descrição.

PRODUTO_FATURA

* NUMERO_PEDIDO
* CODIGO_PRODUTO
DESCR_PRODUTO
QTDE_PEDIDA
PRECO_TOTAL_PRODUTO

6.2.2 Dependência Funcional Transitiva

Quando um atributo C depende de outro atributo B que não pertence à chave primária A, mas é dependente funcional desta, dizemos que C é dependente transitivo de A.

Exemplo: na tabela Departamento abaixo, o atributo NOME_GERENTE é dependente funcional do CODIGO_GERENTE e este, por sua vez, é dependente funcional do CODIGO_DEPARTAMENTO. Portanto, NOME_GERENTE é dependente transitivo da chave CODIGO_DEPARTAMENTO.

DEPARTAMENTO

***** CODIGO_DEPARTAMENTO
NOME_DEPARTAMENTO
SIGLA_DEPARTAMENTO
CODIGO_GERENTE
NOME_GERENTE

6.3 Normalização

O processo de **normalização** consiste em definir o formato lógico adequado para as estruturas de dados do sistema, com o objetivo de minimizar o espaço utilizado pelos dados e garantir a integridade e confiabilidade das informações.

Trata-se de uma técnica formal para analisar as relações, baseando-se nas dependências funcionais entre atributos e chaves. É executada em diversas etapas, chamadas "**Formas Normais (FN)**". As FN são **conjuntos de restrições às quais cada tabela devem satisfazer**. Por exemplo, pode-se dizer que a tabela está na Primeira Forma Normal (1FN), se os dados que a compõem satisfizerem as restrições definidas para esta etapa.

Na normalização, são eliminados esquemas de relações não satisfatórios, decompondo-os, através da separação de seus atributos em esquemas menos complexos, mas que satisfaçam às propriedades desejadas.

Através desse processo pode-se, gradativamente, substituir um conjunto de tabelas por um outro, o qual se apresenta livre de anomalias de atualização (inclusão, alteração e exclusão), as quais podem causar certos problemas, tais como: grupos repetitivos de dados, redundâncias de dados desnecessárias, perdas acidentais de informação, dificuldades na representação de fatos da realidade, etc.

Entidades normalizadas não possuem redundâncias (duplicação de dados) acidental. Cada atributo está relacionado com sua própria entidade e não se mistura com atributos relativos à entidades diferentes.

A normalização pode ser realizada de duas formas:

- **Sentido de cima para baixo (TOP-DOWN)**

Após a definição de um modelo de dados, aplica-se a normalização para se obter uma síntese dos dados, bem como uma decomposição das entidades e relacionamentos em

elementos mais estáveis, tendo em vista a sua implementação física em um banco de dados.

- **Sentido de baixo para cima (BOTTON-UP)**

Aplicar a normalização como ferramenta de projeto do modelo de dados, utilizando os relatórios, formulários e documentos utilizados na realidade em estudo, constituindo-se de uma ferramenta de levantamento. Este processo é também chamado de Engenharia Reversa.

A normalização completa dos dados é feita seguindo as restrições das cinco formas normais existentes, sendo que a passagem de uma FN para outra é feita tendo como base o resultado obtido na etapa anterior. Na prática, utilizamos as 3 primeiras formas normais. As demais aplicam-se a certos casos especiais.

6.3.1 1ª Forma Normal (1FN)

Retirar da tabela os atributos multivalorados e colocá-los numa nova tabela, mantendo-se um relacionamento entre elas.

Em termos práticos, a 1FN consiste em retirar de cada tabela os elementos repetitivos, ou seja, aqueles atributos que possuem mais de um valor para cada tupla. O objetivo desta etapa é garantir que cada atributo possua apenas valores atômicos (únicos, indivisíveis). E, algumas situações, a nova tabela caracteriza o que seria uma entidade fraca; nestes casos, a chave dessa nova tabela será composta pela chave da tabela original + um atributo identificador dessa nova tabela.

Podemos afirmar que uma tabela está normalizada na 1FN, se esta não possuir elementos repetitivos.

Exemplo:

a) Esquema original:

Nota Fiscal (CodigoNotaFiscal, Serie, DataEmissão, CodigoCliente, NomeCliente, EnderecoCliente, CNPJCliente, CodigoMercadoria, DescricaoMercadoria, QuantidadeVendida, PrecoVenda, TotalVendaMercadoria, TotalGeralNota).

Porém, é sabido que existem várias mercadorias em uma única Nota Fiscal, sendo, portanto, elementos repetitivos que deverão ser retirados.

b) Esquema na primeira forma normal (1FN):

NotaFiscal (CodigoNotaFiscal, Serie, Data emissão, CodigoCliente, NomeCliente, EnderecoCliente, CNPJCliente, TotalGeralNota)

ItensNota (CodigoNotaFiscal(NotaFiscal), CodigoMercadoria, DescricaoMercadoria, QuantidadeVendida, PrecoVenda)

Como resultado desta etapa, ocorreu um desdobramento dos dados em duas tabelas, a saber:

- **Tabela NotaFiscal**: dados que compõem a estrutura original, excluindo os elementos repetitivos;
- **Tabela ItensNota**: dados que compõem os elementos repetitivos da estrutura original, tendo sua chave primária composta por dois atributos:
  - uma chave estrangeira, correspondente à chave primária da tabela original (CodigoNotaFiscal); e
  - o atributo chave da estrutura criada (CodigoMercadoria).

6.3.2 2ª Forma Normal (2FN)

Retirar das tabelas que possuem chave primária composta, todos os atributos não chave que possuem dependência parcial com relação à chave (dependem só de um dos atributos da chave, e não de ambos). Estes atributos devem ser colocados em uma nova tabela, cuja chave será o atributo que originou a dependência parcial.

A 2FN utiliza o conceito de dependência funcional completa/parcial, visto no início deste capítulo. Esta FN consiste em retirar das tabelas que possuem chaves compostas (atributo chave sendo formado por mais de um atributo), os elementos que são funcionalmente dependentes de apenas parte da chave.

Podemos afirmar que uma estrutura está na 2FN, se ela estiver na 1FN e não possuir atributos que são funcionalmente dependentes de parte da chave.

OBS: Tabelas cuja chave primária não é composta já estão automaticamente na 2FN.

Exemplo:

a) Esquema na primeira forma normal (1FN):

NotaFiscal (CodigoNotaFiscal, Serie, Data emissão, CodigoCliente, NomeCliente, EnderecoCliente, CNPJCliente, TotalGeralNota)

ItensNota (CodigoNotaFiscal(NotaFiscal), CodigoMercadoria, DescricaoMercadoria, QuantidadeVendida, PrecoVenda)

b) Esquema na segunda forma normal (2FN):

NotaFiscal (CodigoNotaFiscal, Serie, Data emissão, CodigoCliente, NomeCliente, EnderecoCliente, CNPJCliente, TotalGeralNota)

ItensNota (CodigoNotaFiscal(NotaFiscal), CodigoMercadoria(Mercadoria), QuantidadeVendida)

Mercadoria (CodigoMercadoria, DescricaoMercadoria, PrecoVenda).

Como resultado desta etapa, houve um desdobramento da tabela ItensNota (a tabela Nota Fiscal não foi alterada, por não possuir chave composta) em duas estruturas, a saber:

- **Tabela ItensNota**: contém os atributos originais, sendo excluídos os dados que são dependentes apenas do CodigoMercadoria.

- **Tabela Mercadoria**: contém os elementos que são identificados apenas pelo CodigoMercadoria, ou seja, independentemente da Nota Fiscal, a descrição e o preço de venda são constantes.

6.3.3 3ª Forma Normal (3FN)

Consiste em retirar das tabelas os atributos que são funcionalmente dependentes de outros atributos que não são chaves (isto é, que possuem dependência transitiva da chave primária). Uma tabela está na 3ª Forma Normal se estiver na 2ª Forma Normal e não houver dependência transitiva entre atributos não chave.

Exemplo:

a) Estrutura na segunda forma normal (2FN):

NotaFiscal (CodigoNotaFiscal, Serie, Data emissão, CodigoCliente, NomeCliente, EnderecoCliente, CNPJCliente, TotalGeralNota)

ItensNota (CodigoNotaFiscal(NotaFiscal), CodigoMercadoria(Mercadoria), QuantidadeVendida)

Mercadoria (CodigoMercadoria, DescricaoMercadoria, PrecoVenda).

b) Estrutura na terceira forma normal (3FN):

NotaFiscal (CodigoNotaFiscal, Serie, Data emissão, CodigoCliente(Cliente), TotalGeralNota)

ItensNota (CodigoNotaFiscal(NotaFiscal), CodigoMercadoria(Mercadoria), QuantidadeVendida)

Mercadoria (CodigoMercadoria, DescricaoMercadoria, PrecoVenda)

Cliente (CodigoCliente, NomeCliente, EnderecoCliente, CNPJCliente)

Como resultado desta etapa, houve um desdobramento da tabela NotaFiscal, por ser a única que possuía atributos que não eram dependentes da chave principal (CodigoNotaFiscal), uma vez que independente da Nota Fiscal, o Nome, Endereço e CNPJ do cliente permanecem inalterados. Este procedimento permite evitar inconsistência nos dados dos arquivos e economizar espaço por eliminar o armazenamento frequente e repetido destes dados. As estruturas alteradas foram:

- **Tabela NotaFiscal**: contém os elementos originais, sendo excluídos os dados que são dependentes apenas do atributo CodigoCliente (informações referentes ao cliente);

- **Tabela Cliente**: contém os elementos que são identificados apenas pelo CodigoCliente, ou seja, independente da Nota Fiscal, o Nome, Endereço e CNPJ dos clientes serão constantes.