Normalização em Banco de Dados Relacional

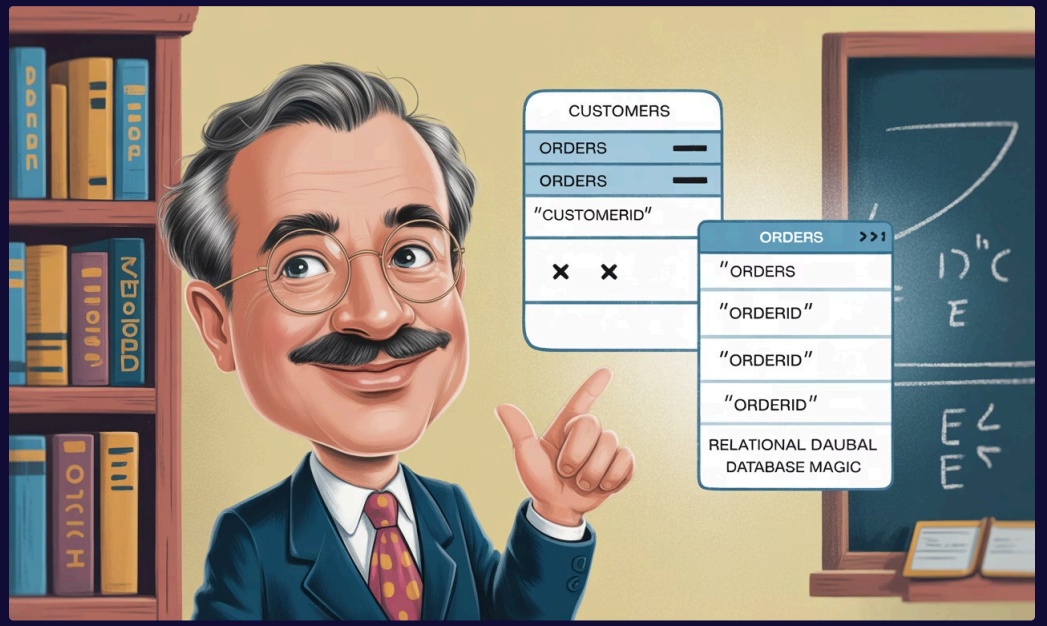
A normalização é um processo fundamental para organizar dados em um banco de dados relacional, reduzindo redundâncias e evitando anomalias. Nesta apresentação, exploraremos os três primeiros níveis de normalização com exemplos práticos e visuais para facilitar a compreensão.

O que é Normalização?

A normalização é um processo metódico de organização de dados em tabelas relacionais, desenvolvido pelo matemático Edgar F. Codd em 1970. Este processo segue um conjunto de regras para:

- Minimizar a duplicação de dados
- Garantir a integridade dos dados
- Facilitar a manutenção do banco de dados
- Otimizar consultas e operações

Sem normalização adequada, os bancos de dados sofrem com problemas de redundância, inconsistência e baixo desempenho.



Edgar F. Codd, matemático britânico que propôs o modelo relacional e as regras de normalização, revolucionando a forma como armazenamos e gerenciamos dados.

Primeira Forma Normal (1NF)

Atomicidade

Cada célula da tabela deve conter um único valor atômico (indivisível). Não são permitidos valores múltiplos, arrays ou listas em uma única célula.

Chave Primária

A tabela deve ter uma chave primária que identifique unicamente cada registro. Pode ser uma coluna ou combinação de colunas.

Sem Grupos Repetitivos

Não são permitidos grupos de colunas repetidas (como Telefone1, Telefone2, Telefone3) ou colunas multivaloradas.

A 1NF é o ponto de partida essencial para o processo de normalização, eliminando repetições e garantindo valores atômicos em todas as células.

Exemplo Prático de 1NF

Tabela de Alunos (Não normalizada)

ID_Aluno	Nome	Telefones
1	Carlos Silva	11-99876-5432, 11-3456-7890
2	Ana Souza	21-98765-4321

Tabelas após aplicação da 1NF

Tabela Alunos

ID_Aluno	Nome
1	Carlos Silva
2	Ana Souza

Tabela AlunoTelefone

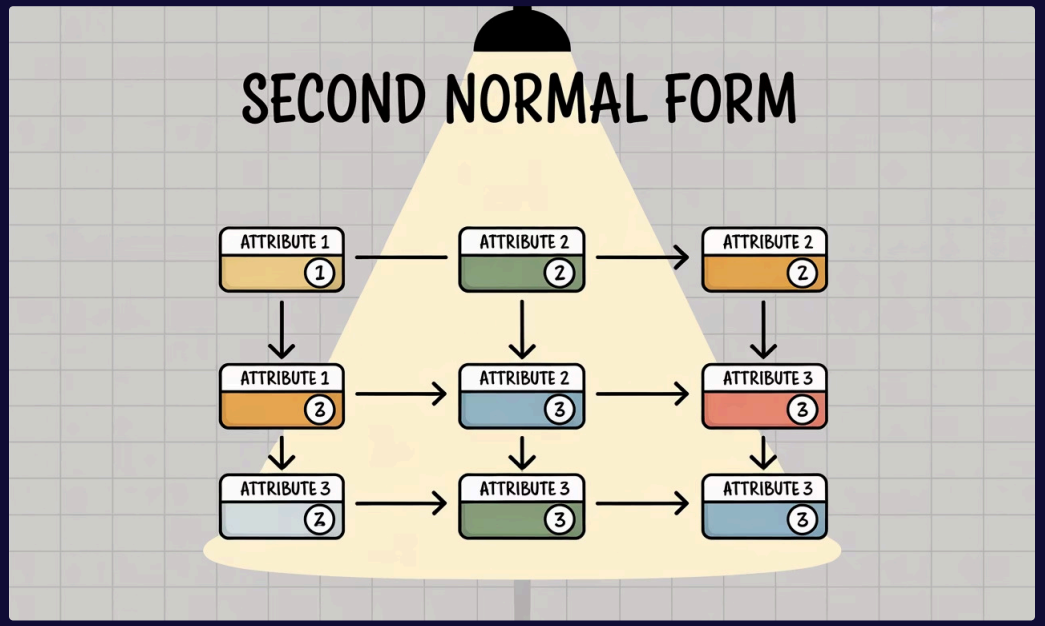
ID	ID_Aluno	Telefone
1	1	11-99876-5432
2	1	11-3456-7890
3	2	21-98765-4321

Segunda Forma Normal (2NF)

A Segunda Forma Normal (2NF) só se aplica quando temos uma chave primária composta (formada por múltiplas colunas). Para estar em 2NF, uma tabela deve:

- Estar em 1NF (requisito obrigatório)
- Todos os atributos não-chave devem depender da chave primária completa
- Eliminar dependências parciais (quando um atributo depende apenas de parte da chave composta)

Se uma tabela já possui uma chave primária simples (não composta), ela automaticamente está em 2NF se já estiver em 1NF.



A 2NF resolve problemas de redundância que ocorrem quando atributos dependem apenas de parte da chave primária composta.

Exemplo Prático de 2NF

Tabela de Pedidos (Em 1NF, mas não em 2NF)

ID_Pedido	ID_Produto	Nome_Produto	Preco_Produto	Quantidade
1	101	Teclado	150,00	2
1	102	Mouse	80,00	1
2	101	Teclado	150,00	1

Tabelas após aplicação da 2NF

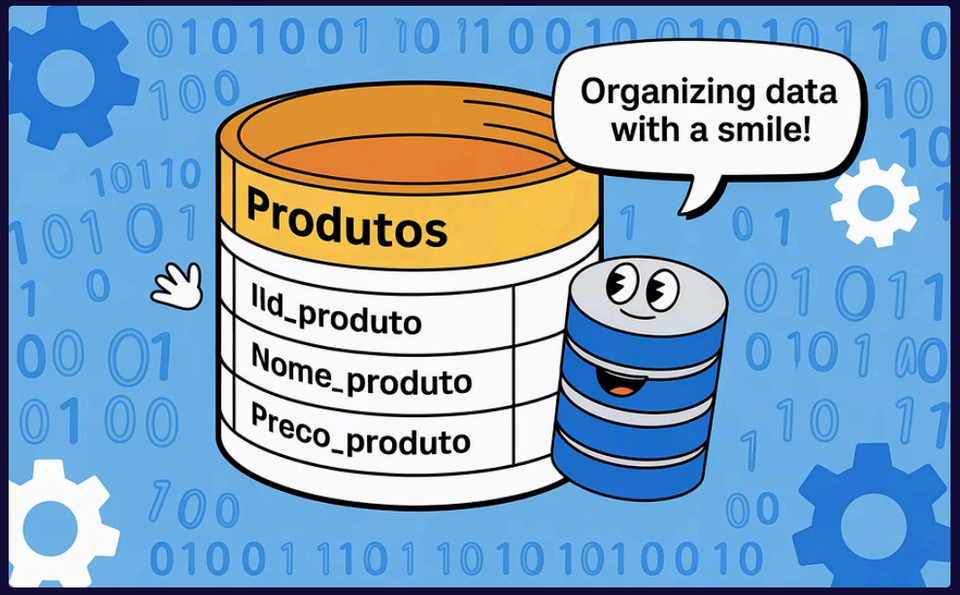


Tabela Produtos

Armazena informações que dependem apenas do ID_Produto, eliminando a repetição desses dados em cada pedido.

Tabela ItensPedido

Mantém a chave composta (ID_Pedido, ID_Produto) e armazena apenas a quantidade, que depende da combinação completa dessas chaves.

Terceira Forma Normal (3NF)

Estar em 2NF

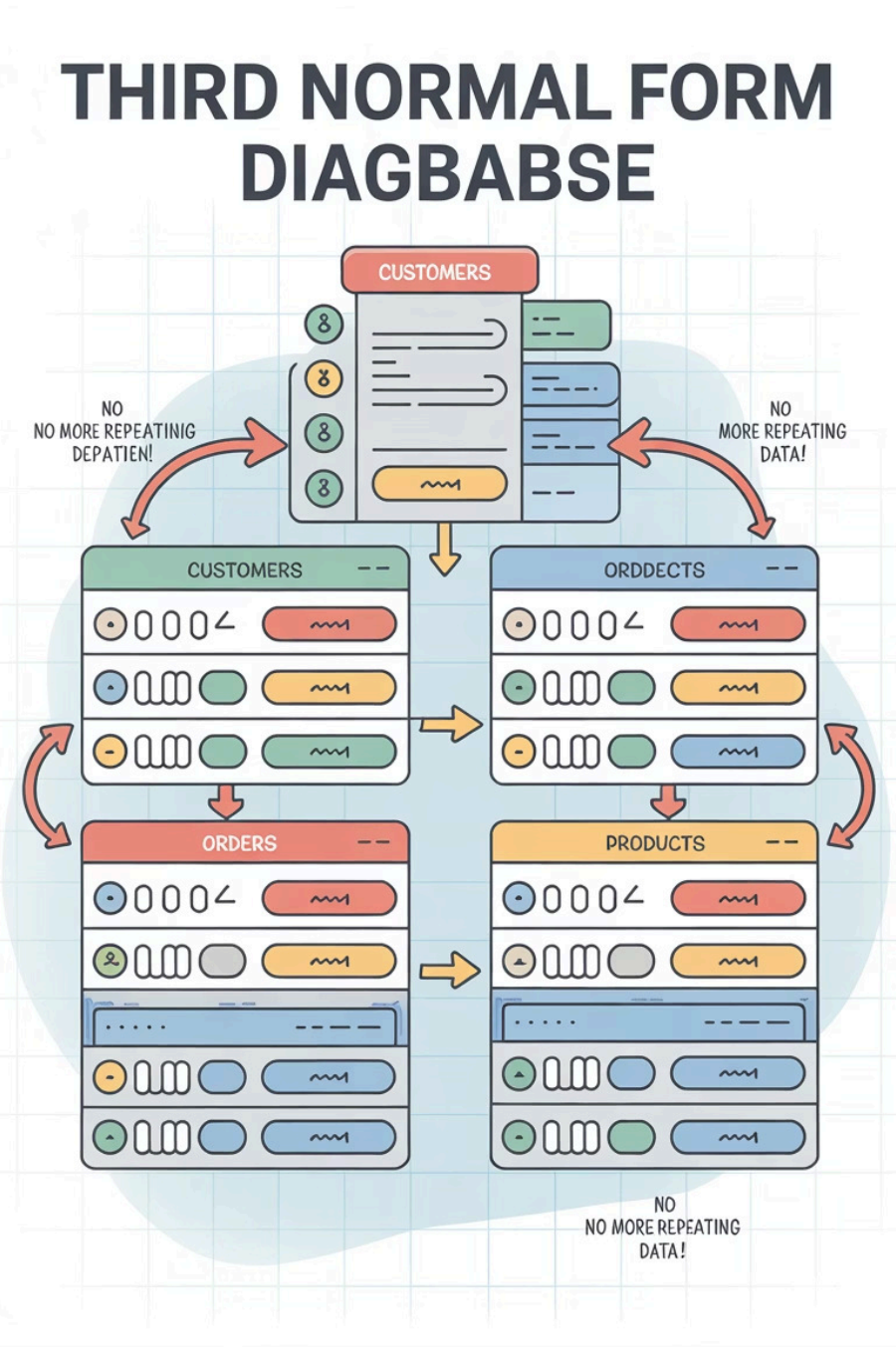
A tabela já deve cumprir todos os requisitos da Segunda Forma Normal para prosseguir.

Eliminar Dependências Transitivas

Nenhum atributo não-chave deve depender de outro atributo não-chave. Isso elimina dependências transitivas.

Criar Novas Tabelas

Mover atributos com dependências transitivas para tabelas separadas, mantendo relacionamentos através de chaves estrangeiras.



Exemplo Prático de 3NF

Tabela de Funcionários (Em 2NF, mas não em 3NF)

ID_Funcionario	Nome	ID_Cidade	Cidade	Estado
1	Paulo Santos	1	São Paulo	SP
2	Maria Oliveira	2	Rio de Janeiro	RJ
3	João Silva	1	São Paulo	SP

Neste exemplo, o Estado depende da Cidade (e não diretamente do ID_Funcionario), criando uma dependência transitiva.

Tabelas após aplicação da 3NF

Tabela Funcionários

ID_Funcionario	Nome	ID_Cidade
1	Paulo Santos	1
2	Maria Oliveira	2
3	João Silva	1

Tabela Cidades

ID_Cidade	Cidade	Estado
1	São Paulo	SP
2	Rio de Janeiro	RJ

Benefícios da Normalização

Redução de Redundância

Elimina a duplicação de dados, economizando espaço de armazenamento e melhorando a eficiência.

Integridade de Dados

Previne inconsistências e garante que alterações sejam aplicadas uniformemente em todo o banco de dados.

Facilidade de Manutenção

Estruturas normalizadas são mais fáceis de entender, modificar e expandir conforme as necessidades do negócio.

Prevenção de Anomalias

Evita problemas durante inserção, atualização e exclusão de dados que poderiam comprometer a consistência do banco.

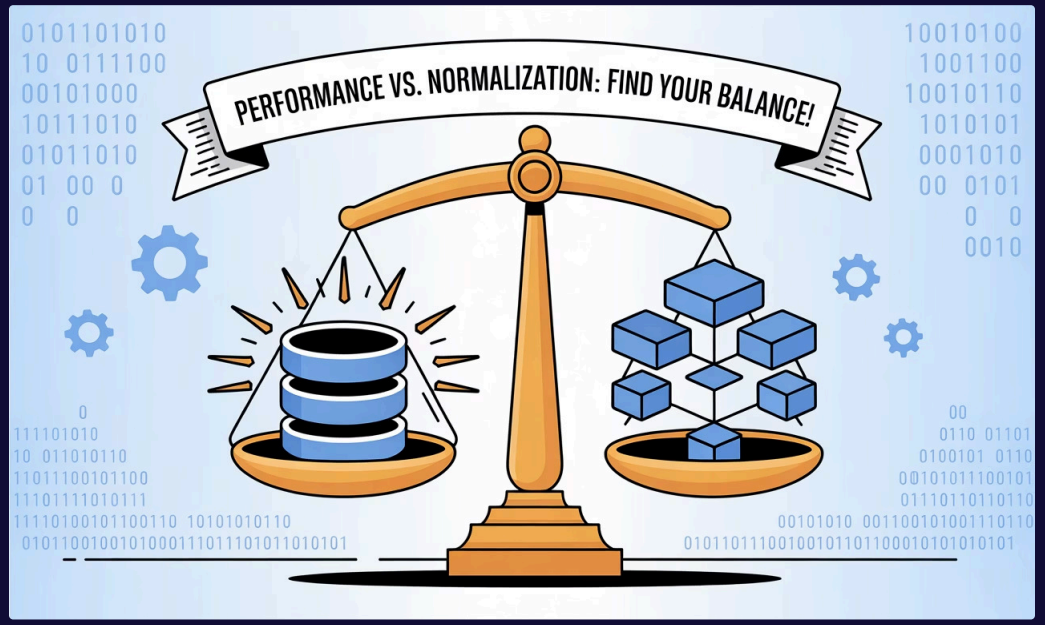
Considerações Finais

Equilíbrio com Desempenho

Embora a normalização melhore a integridade, muitas tabelas podem aumentar a complexidade das consultas. Em alguns casos, a desnormalização controlada pode ser aplicada para otimizar o desempenho.

Além da 3NF

Existem formas normais adicionais (BCNF, 4NF, 5NF), mas para a maioria das aplicações práticas, a 3NF é suficiente para garantir um bom design de banco de dados.



O processo de normalização deve considerar tanto a integridade quanto o desempenho do banco de dados, encontrando o equilíbrio ideal para cada aplicação.

Conclusão

1NF: Atomicidade

Elimina valores múltiplos nas células, garantindo que cada campo contenha apenas um valor atômico.

2NF: Dependência Total

Remove dependências parciais, garantindo que atributos não-chave dependam da chave primária completa.

3NF: Sem Dependências Transitivas

Elimina dependências entre atributos não-chave, garantindo que cada atributo dependa diretamente da chave primária.

A normalização é um processo fundamental que, quando aplicado corretamente, resulta em bancos de dados mais eficientes, confiáveis e fáceis de manter. Compreender e aplicar os três primeiros níveis de normalização é essencial para qualquer profissional que trabalhe com bancos de dados relacionais.