## linha horizontal



Banco de dados e normalização de base de dados

04/12/2024

**─**

Adriely Lima Novaes

Israel Santana dos Anjos

# Introdução

Um banco de dados é um conjunto de arquivos essencial para a gestão eficiente de informações em qualquer empresa, eles permitem o armazenamento, organização e recuperação de dados de maneira estruturada, facilitando a tomada de decisão, como também, a otimização de processos. No contexto escolhido, de pizzaria, um banco de dados permite armazenar informações detalhadas sobre pedidos, clientes, pizza e bebidas. Isso facilita a recuperação de informações precisas e atualizadas sempre que necessário.

A utilização de um banco de dados melhora a eficiência operacional ao automatizar processos e reduzir a necessidade de intenção manual, como por exemplo; a tabela pedidos pode registrar automaticamente a data e hora de cada pedido, o método de pagamento e taxa para entrega, agilizando o processo de atendimento ao cliente.

Com um banco de dados bem constituído, a administração da empresa - pizzaria - pode acessar relatórios detalhados sobre vendas, estoque e desempenho de produtos. Isso permite a tomada de decisões informadas, como ajustar o menu, planejar promoções e gerenciar o estoque de forma eficaz. Porém, para que se evite redundâncias é crucial a normalização de bases de dados facilitando a análises de informações e garantindo a integridade dos dados.

A normalização é dividida em quatro formas.

* Primeira forma normal (1FN)
* Segunda forma normal (2FN)
* Terceira forma normal (3FN)
* Boyce-Codd Normal Form (BCNF)

# Conceituação

O banco de dados relacional é um tipo de banco que armazena e organiza dados em tabelas, onde cada linha representa um registro e cada coluna representa um atributo do dado. Essas tabelas podem ser relacionadas entre si através de chaves primárias e estrangeiras, permitindo a criação de relações entre diferentes conjuntos de dados. Tendo como características principais modelo relacional, chaves primárias e estrangeiras, consultas SQL, integridade dos dados.

* Modelo Relacional: Os dados são organizados em tabelas.
* Chaves primárias e estrangeiras:As chaves primárias identificam unicamente cada linha em uma tabela, enquanto as chaves estrangeiras têm relações entre diferentes tabelas.
* Consultas SQL:Utiliza a linguagem SQL para consultar, inserir, atualizar e excluir dados.
* Integridade de dados:Garante a consistência e precisão dos dados através de regras de integridade, como restrições de unicidade.

A normalização é um processo sistemático para organizar dados em um BD (banco de dados) a fim de reduzir a redundância e melhorar a integridade dos dados. Antes da normalização os dados podem estar em um formato onde existe muita redundância, dificultando a manutenção e a consulta eficiente, dizemos que os dados não estão normalizados. Neste sentido, a etapa de eliminar grupos repetitivos de dados, chama-se primeira forma normal e busca-se garantir que cada célula da tabela contenha apenas o um valor (atomicidade) e cada tabela deve ter uma chave primária que identifica unicamente cada registro. Na segunda forma normal busca eliminar dependências parciais. Garantir que todos os atributos, não chave, dependem da chave primária completa, isso pode implicar em criar tabelas adicionais. Na etapa de eliminar dependências transitivas é preciso garantir que todos os atributos não chave dependem somente da chave primária e não de outros atributos não chave, chamando essa etapa de terceira forma normal (3FN). Na última etapa, forma normal de Boyce-Codd (BCNF), busca-se garantir que cada dependência funcional seja de uma super-chave (atributo ou conjunto de atributos que unicamente identifica um registro).

Seguindo essas etapas de normalização, é possível criar um BD eficiente e de fácil manutenção, evitando redundâncias e garantindo a integridade dos dados. Cada etapa busca eliminar tipos específicos de redundâncias e dependências que podem causar algum tipo de anomalia e problemas no gerenciamento de dados.

# Métodos

O processo de desenvolvimento do banco de dados foi conduzido em etapas estruturadas para que possa ser garantido a organização e eficiência do sistema. Inicialmente, foi elaborado um modelo não normalizado, contendo todos os dados brutos sem nenhum tipo de tratamento, incluindo informações de clientes, pedidos, pizzas e bebidas. Posteriormente, aplicamos o processo de normalização, utilizando as três primeiras formas normais.

Para modelar o banco de dados, foi utilizado o sistema de gerenciamento de banco de dados (SGDB) MySQL Workbench, onde foi criado o diagrama entidades relacionamento (DER) lógico como referência. Com base no DER, as tabelas foram reorganizadas gradualmente para atender às regras de cada forma normal. Inicialmente, como supracitado, as tabelas possuíam todos os dados brutos. Na primeira forma normal (1FN), os dados foram organizados em tabelas distintas, eliminando valores repetidos e garantindo a atomicidade. Em seguida, a segunda forma normal (2FN) corrigiu dependências parciais, garantindo que todos os atributos fossem dependentes das chaves primárias. Por fim, a terceira forma normal (3FN) eliminou dependências transitivas, separando informações não diretamente relacionadas às chaves primárias e finalmente última etapa que busca refinar o que a terceira forma normal não resolveu.

Esses processos de normalização foram fundamentais para eliminar redundâncias, evitar inconsistências e garantir maior eficiência e integridade no sistema de banco de dados da pizzaria. O processo também permitiu que o banco de dados fosse mais escalável e eficiente para consultas.

# Processos

Tabela antes da 1FN (não normalizada), de forma simplificada. Ao final, o DER completo estará disponível.

| Cliente | Endereço | Telefone | Pedido | Data/Hora | Pizza | Tamanho | Bebida | Quantidade |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| João Silva | Rua A, nº 10 | 1234-5678 | 001 | 18/11/2024 19:00 | Calabresa | Grande | Coca-Cola | 1 |
| João Silva | Rua A, nº 10 | 1234-5678 | 001 | 18/11/2024 19:00 | Quatro-Queijos | Grande | Coca-Cola | 1 |
| Maria Oliveira | Rua C, nº 265 | 2352-6789 | 002 | 18/11/2024 19:30 | Portuguesa | Pequena | Pepsi | 2 |
| Fernanda Barbosa | Rua J, nº 359 | 8887-5533 | 003 | 18/11/2024 21:30 | Calabresa | Grande | Coca-Cola | 3 |

Essa tabela contém redundâncias, como a repetição do cliente para cada pedido.

Na 1FN, organizamos os dados em tabelas separadas, eliminando valores repetidos e agrupamentos não atômicos.

* Tabela do Cliente

| ID Cliente | Nome do Cliente | Endereço | Telefone |
| --- | --- | --- | --- |
| 1 | João Silva | Rua A, nº 10 | 1234-5678 |
| 2 | Maria Oliveira | Rua C, nº 265 | 2352-6789 |
| 3 | Fernanda Barbosa | Rua J, nº 359 | 8887-5533 |

* Tabela do Pedido

| ID Pedido | ID Cliente | Data/Hora | Entrega | Taxa Entrega |
| --- | --- | --- | --- | --- |
| 001 | 1 | 18/11/2024 19:00 | 1 | 5.00 |
| 002 | 2 | 18/11/2024 19:30 | 1 | 5.00 |
| 003 | 3 | 18/11/2024 21:30 | 1 | 5.00 |

* Comando em SQL

| CREATE DATABASE pizzaria; USE pizzaria;  CREATE TABLE cliente (  pk\_cliente INT NOT NULL AUTO\_INCREMENT,  nameCliente VARCHAR(255),  adress VARCHAR(255),  phone VARCHAR(255),  PRIMARY KEY (pk\_cliente) );  CREATE TABLE pedido (  pk\_pedido INT NOT NULL AUTO\_INCREMENT,  fk\_cliente INT NOT NULL,  dataHora TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,  entrega BOOLEAN NOT NULL,  taxaEntrega FLOAT,  PRIMARY KEY (pk\_pedido),  FOREIGN KEY (fk\_cliente) REFERENCES cliente(pk\_cliente) ); |
| --- |

Na 2FN, removemos dependências parciais, separando informações que não dependem completamente da chave primária.

* Tabela PedidoPizza

| ID Pedido | ID Pizza | Quantidade |
| --- | --- | --- |
| 001 | 1 | 1 |
| 001 | 2 | 1 |
| 002 | 3 | 2 |
| 003 | 1 | 3 |

* Tabela Pizza

| ID Pizza | Sabor | Preço | Personalizada |
| --- | --- | --- | --- |
| 1 | Calabresa | 30.00 | 0 |
| 2 | Quatro-Queijos | 30.00 | 0 |
| 3 | Portuguesa | 30.00 | 0 |

* Comando SQL

| CREATE TABLE pizza (  pk\_pizza INT NOT NULL AUTO\_INCREMENT,  flavorPizza VARCHAR(255),  price FLOAT,  personalized BOOLEAN,  PRIMARY KEY (pk\_pizza) );  CREATE TABLE pedidoPizza (  pk\_pedidoPizza INT NOT NULL AUTO\_INCREMENT,  fk\_pedido INT NOT NULL,  fk\_pizza INT NOT NULL,  quantidade INT,  PRIMARY KEY (pk\_pedidoPizza),  FOREIGN KEY (fk\_pedido) REFERENCES pedido(pk\_pedido),  FOREIGN KEY (fk\_pizza) REFERENCES pizza(pk\_pizza) ); |
| --- |

Na 3FN, eliminamos dependências transitivas. Como por exemplo, o tamanho da pizza foi separada em uma tabela própria, porque não é uma propriedade direta do pedido.

* Tabela TamanhoPizza

| ID Tamanho | Descrição | Preço |
| --- | --- | --- |
| 1 | Grande | 5.00 |
| 2 | Pequena | 0.00 |

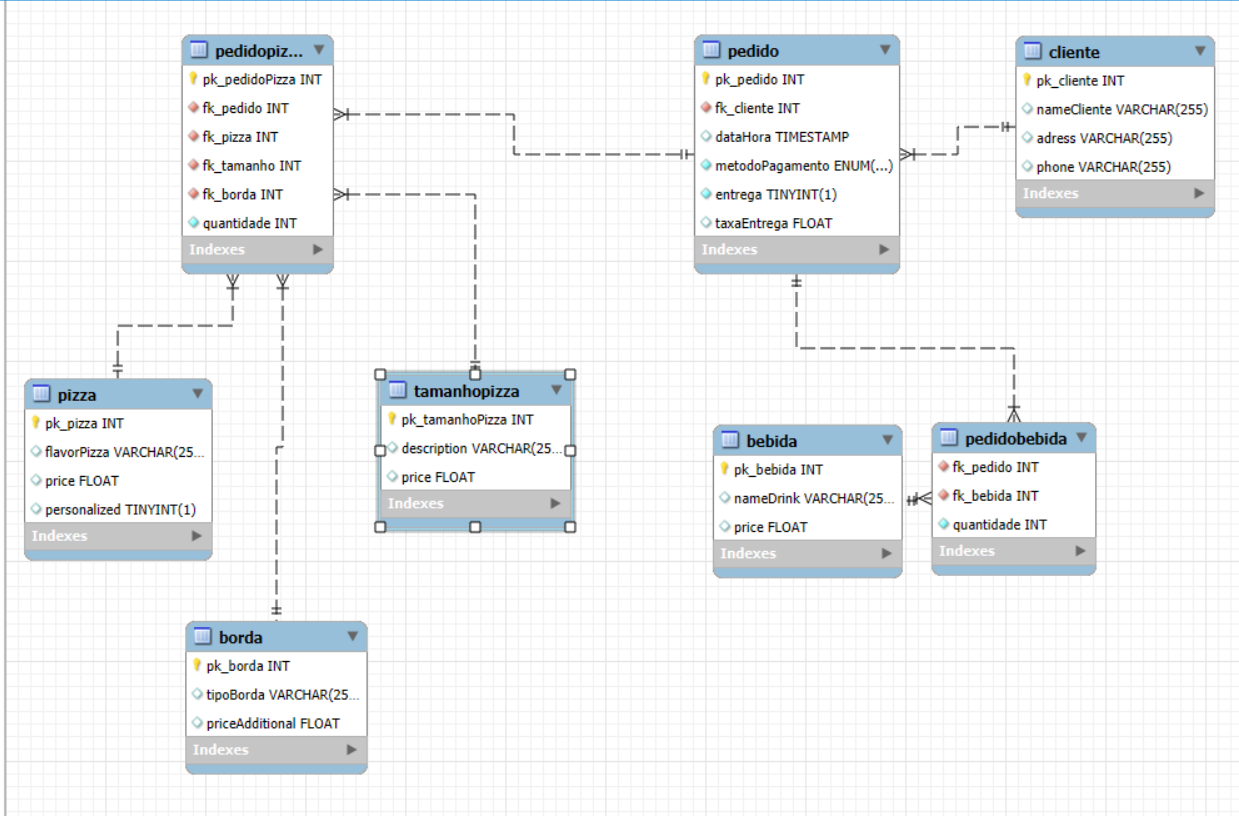
* Comando SQL

| CREATE TABLE tamanhoPizza (  pk\_tamanhoPizza INT NOT NULL AUTO\_INCREMENT,  description VARCHAR(255),  price FLOAT,  PRIMARY KEY (pk\_tamanhoPizza) ); |
| --- |

A separação dos dados em tabelas distintas eliminou redundâncias, a integridade dos dados foi garantida com o uso de chaves primárias e estrangeiras, tornando o modelo mais eficiente para consultas e manutenção.

A seguir, apresentamos o Diagrama Entidade-Relacionamento (DER) resultante, que resume a estrutura do banco de dados normalizado.

Figura 1: Diagrama Entidade - Relacionamento (DER) do banco de dados normalizado

[](#_sicaxgy4wgcr)

Fonte: MySQL Workbench

# Conclusão

Ao longo deste trabalho, foi possível explorar e aplicar os conceitos fundamentais de banco de dados e normalização, desde a criação de um modelo inicial até a implementação de um banco de dados completamente normalizado. O processo de normalização de um banco de dados completamente normalizado mostrou - se crucial para eliminar redundâncias, garantir a integridade dos dados e otimizar a estrutura do banco de dados, tornando - o mais eficiente e sustentável para consultas e manutenções futuras.

A normalização até a 3FN proporcionou uma organização lógica dos dados, separando - os em tabelas relacionadas por chaves primárias e estrangeiras, conforme ilustrado no DER (Figura 1). Esse processo não apenas melhorou a consistência do banco de dados, como também evidenciou a importância de boas práticas de modelagem para evitar problemas de desempenho e inconsistências em bases de dados reais.

Implementar no MySQL Workbench reforçou a compreensão dos conceitos teóricos, permitindo a utilização de comandos SQL para a criação e manipulação das tabelas. Essas etapas destacam como a modelagem lógica e a aplicação prática estão interligadas, sendo etapas complementares e indispensáveis no desenvolvimento de sistemas robustos.

Por fim, o trabalho evidenciou a relevância da normalização de bases de dados em contextos reais, como no domínio da pizzaria utilizando como exemplo. A correta modelagem e normalização asseguram que os dados sejam armazenados de forma eficiente, contribuindo diretamente para o sucesso das operações empresariais e para a qualidade dos sistemas de informações. Assim, o aprendizado adquirido ao longo do desenvolvimento deste projeto é uma base sólida para projetos futuros envolvendo banco de dados e normalização de base de dados.

# Referências

LENCINA, W. O que é a normalização de bases de dados e como fazê-la? Disponível em: <<https://ebaconline.com.br/blog/normalizacao-de-bases-de-dados>>.

‌EJFGV. O que é um banco de dados e por que é importante para sua empresa? Disponível em: <<https://ejfgv.com/banco-de-dados-em-empresas/>>.

ROB, Peter; CORONEL, Carlos. Sistemas de banco de dados. **Projeto, implementação e**, 2011.

SILVA, Gilmar José da; FERREIRA, Júlio Cesar Oliveira. Análise Comparativa de Desempenho de Consultas entre um Banco de Dados Relacional e um Banco de Dados Não Relacional. 2017.