Analise das normalizações

(Banco de dados Locações)

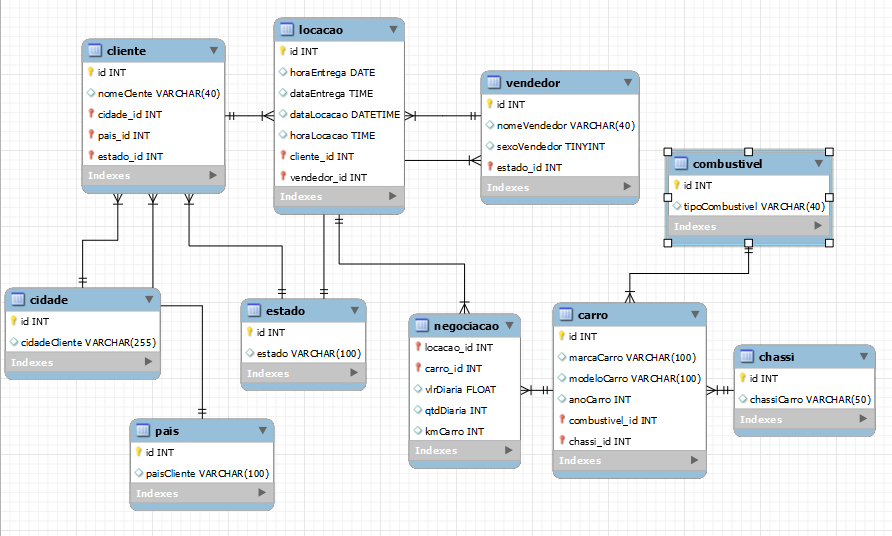
Dartanhan Alencar integrante do squad 5.

A normalização de um banco de dados é um processo utilizado para organizar os dados de uma forma que minimize a redundância e a inconsistência, garantindo a integridade e a eficiência das operações de consulta e atualização. O objetivo principal da normalização é eliminar a redundância de dados, evitando assim problemas como a inconsistência e a duplicação de informações.

A normalização é geralmente dividida em diferentes formas normais, representadas por números: 1NF (Primeira Forma Normal), 2NF (Segunda Forma Normal), 3NF (Terceira Forma Normal) e assim por diante. Cada forma normal possui critérios específicos que devem ser atendidos para que uma tabela seja considerada normalizada de acordo com essa forma.

Na normalização para a primeira forma normal (1NF), é necessário garantir que cada coluna de uma tabela contenha apenas valores atômicos, ou seja, valores indivisíveis. Além disso, não devem existir grupos repetidos de colunas, e cada linha deve ser única.

Por exemplo, suponha que temos uma tabela de clientes com as seguintes colunas: ID do cliente, nome do cliente, telefone e endereço. Para normalizá-la para a 1NF, cada coluna deve conter apenas um tipo de informação (não devemos ter múltiplos telefones ou múltiplos endereços na mesma coluna), e cada linha deve ser única, representando um único cliente.



Modelo relacional feita através do MySQL Workbench

No modelo relacional (modelo relacional locações) acima podemos notar que a 1NF foi respeitada.

A Segunda Forma Normal (2NF) é um estágio subsequente à Primeira Forma Normal (1NF) no processo de normalização de banco de dados. Enquanto a 1NF trata principalmente da atomicidade dos valores em cada coluna de uma tabela, a 2NF trata da eliminação de dependências parciais.

Em termos simples, uma dependência parcial ocorre quando parte da chave primária de uma tabela determina um conjunto de colunas que não são completamente dependentes dela, ou seja, seus valores dependem apenas de uma parte da chave primária, mas não de toda ela.

Para normalizar uma tabela para a 2NF, é necessário que atenda aos seguintes critérios:

1. Já deve estar na 1NF.
2. Todos os atributos não chave (isto é, aqueles que não fazem parte da chave primária) devem ser totalmente dependentes da chave primária.
3. Se uma tabela possui uma chave primária composta por mais de uma coluna e os atributos não chave são dependentes de apenas uma parte da chave primária, então ela está sujeita a problemas de dependência parcial e precisa ser normalizada para a 2NF.

Por exemplo, considere uma tabela de pedidos com as seguintes colunas: ID do pedido, ID do cliente e Nome do cliente. Se a chave primária é composta por ID do pedido e ID do cliente, e o Nome do cliente depende apenas do ID do cliente, então há uma dependência parcial. Neste caso, a normalização para a 2NF envolveria a remoção do Nome do cliente para uma tabela separada onde a chave primária seria o ID do cliente.

Em resumo, a 2NF visa eliminar dependências parciais em uma tabela, garantindo assim uma estrutura mais robusta e livre de redundâncias.

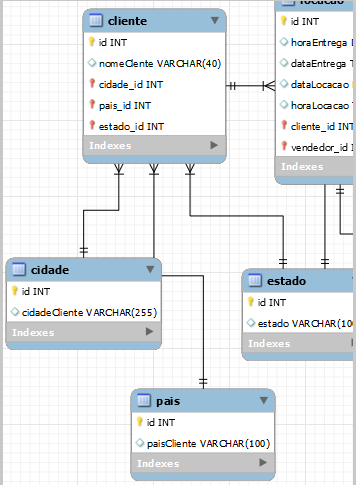


Tabela cliente e tabelas cidade, estado e pais presentes no modelo relacional das locações.

A tabela acima mostra a relação de normalização 2NF, que é caracterizado pela reorganização de dados, levando as colunas “estados”, “pais” e “cidade” para tabelas próprias.

A Terceira Forma Normal (3NF) é um estágio subsequente à Segunda Forma Normal (2NF) no processo de normalização de banco de dados.

A 3NF trata da eliminação de dependências transitivas em uma tabela. Uma dependência transitiva ocorre quando um atributo não chave depende de outro atributo não chave, que por sua vez depende da chave primária.

Para estar em conformidade com a 3NF, uma tabela precisa:

1. Já estar na 2NF.
2. Eliminar quaisquer dependências transitivas, garantindo que todos os atributos não chave dependam apenas da chave primária.

Por exemplo, considere uma tabela de informações de funcionários com as seguintes colunas: ID do funcionário (chave primária), Nome do funcionário, Departamento e Localização do departamento. Se a Localização do departamento depende do Departamento e não diretamente do ID do funcionário, há uma dependência transitiva. Para corrigir isso e normalizar para a 3NF, você precisaria dividir essa tabela em duas, uma para informações de funcionários e outra para informações de departamentos, conectadas por meio de uma chave estrangeira.

Em resumo, a 3NF é um estágio crucial na normalização de um banco de dados, pois visa eliminar dependências transitivas, garantindo uma estrutura de dados mais robusta, coesa e livre de redundâncias.

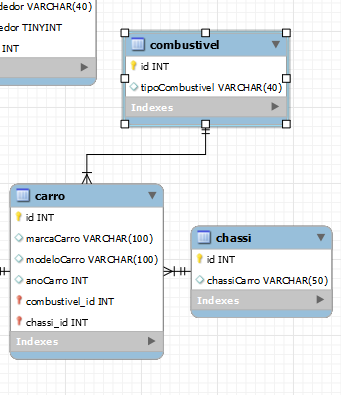


Tabela carro e chassi presentes no modelo relacional das locações

A imagem acima mostra a tabela carro, onde uma das suas características diretamente relacionada que é o chassi do carro, possui uma tabela única.

Pelos seguintes motivos.

**Identificação única**: O número do chassi é único para cada veículo. Isso significa que ele pode ser usado como uma chave primária em uma tabela dedicada, garantindo que cada registro na tabela represente um veículo específico sem ambiguidades.

**Facilidade de referência**: Quando informações sobre um veículo precisam ser referenciadas em outras tabelas (como em tabelas de vendas, manutenção, seguros, etc.), é conveniente ter uma chave única, como o número do chassi, para fazer essas referências de forma precisa e eficiente.

**Integridade referencial**: Ter uma tabela separada para o número do chassi facilita a aplicação de integridade referencial. Isso significa que você pode definir relacionamentos entre essa tabela e outras, garantindo que todas as referências ao número do chassi estejam consistentes e que não haja registros órfãos ou inconsistentes.

**Evitar duplicação de dados**: Se você armazenar o número do chassi em várias tabelas, estará duplicando esses dados. Isso pode levar a problemas de integridade e dificuldades de manutenção. Ter uma tabela única para o número do chassi evita essa duplicação e mantém o banco de dados mais organizado.

**Facilidade de manutenção**: Quando é necessário atualizar ou modificar informações sobre um veículo, ter uma tabela dedicada torna mais fácil e eficiente fazer essas alterações em um único lugar, em vez de procurar e modificar várias ocorrências do número do chassi em diferentes tabelas.