**FORMAS NOMINAIS**

1FN –

O procedimento inicial é identificar a chave primária da tabela. Após, devemos reconhecer o grupo repetitivo e removê-lo da entidade. Em seguida, criamos uma nova tabela com a chave primária da tabela anterior e o grupo repetitivo.



Tabela 1: Tabela não está na primeira forma normal

Analisando o exemplo acima, podemos observar dois problemas: temos uma pessoa com dois números de telefone e um endereço com diferentes valores, a rua e o bairro. A fim de normalizar, teremos que colocar cada informação em uma coluna diferente e criar uma nova tabela relacionando a pessoa a seus números de contato.

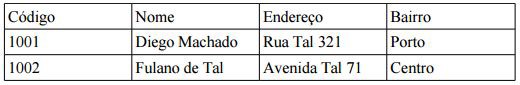


Tabela 2: Tabela está na primeira forma normal

Dessa forma, como mostrado na tabela acima, temos uma tabela na primeira forma normal evitando assim repetições e campos com múltiplos valores, conforme observamos na tabela abaixo.

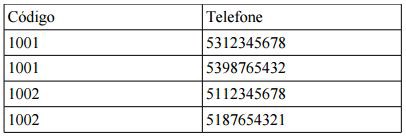


Tabela 3: Nova tabela criada para evitar campos com mais de um valor

2FN –

É dito que uma tabela está na segunda forma normal se ela atende a todos os requisitos da primeira forma normal e se os registros na tabela, que não são chaves, dependam da chave primária em sua totalidade e não apenas parte dela. A segunda forma normal trabalha com essas irregularidades e previne que haja redundância no banco de dados.

Para isso, devemos localizar os valores que dependem parcialmente da chave primária e criar tabelas separadas para conjuntos de valores que se aplicam a vários registros e relacionar estas tabelas com uma chave estrangeira.

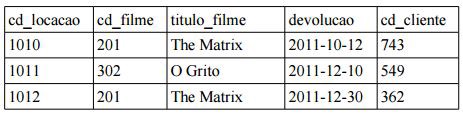


Tabela 4: Tabela não está na segunda forma normal

Podemos observar que a tabela acima apresenta uma coluna responsável por armazenar o título do filme, onde este foi alugado e está associado a um número de locação. Porém, ele também está associado a um código, tornando-o então um valor que não é totalmente dependente da chave primária da tabela.

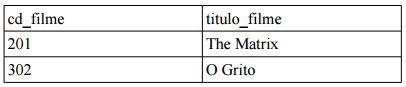


Tabela 5: Tabela criada para armazenar os filmes

Se em algum momento tivermos que alterar o título de um filme, teríamos que procurar e alterar os valores em cada tupla (linha) da tabela. Isso demandaria um trabalho e tempo desnecessário. Porém, ao criarmos uma tabela e vincularmos elas com o recurso da chave estrangeira, tornamos o nosso banco mais organizado e ágil para as futuras consultas e manutenções que podem vir a ser necessárias.

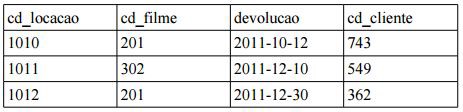


Tabela 6: Tabela na segunda forma normal

3FN –

Se analisarmos uma tupla e não encontrarmos um atributo **não chave** dependente de outro atributo **não chave**, podemos dizer que a entidade em questão está na terceira forma normal - contanto que esta não vá de encontro as especificações da primeira e da segunda forma normal.

Como procedimento principal para configurar uma entidade que atenda as regras da terceira forma normal, nós identificamos os campos que não dependem da chave primária e dependem de um outro campo não chave. Após, separamos eles para criar uma outra tabela distinta, se necessário.

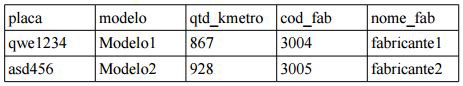


Tabela 7: Tabela não está na terceira forma normal

No exemplo acima temos uma entidade que lista os carros cadastrados, bem como o modelo, a quantidade de quilômetros rodados, o código do fabricante e o nome do fabricante. Observamos que “nome\_fab” se dá em função de “cod\_fab”. Para adequarmos esta tabela de acordo com os padrões da terceira forma normal, devemos remover a coluna do nome do fabricante.

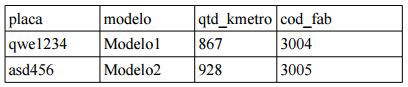


Tabela 8: Tabela na terceira forma normal

A coluna que removemos deve ser colocada em uma nova tabela, relacionando corretamente o nome do fabricante com o seu código. Abaixo, podemos observar como ficaria esta nova entidade.

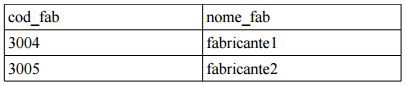


Tabela 9: Tabela criada para armazenar o nome do fabricante

https://medium.com/@diegobmachado/normalização-em-banco-de-dados-5647cdf84a12

não agüento mais