Normalização de Bancos de Dados

Normalização é um processo a partir do qual se aplicam regras a todas as tabelas do banco de dados com o objetivo de evitar falhas no projeto, como redundância de dados e mistura de diferentes assuntos numa mesma tabela.

Ao projetar um banco de dados, se temos um modelo de entidades e relacionamentos e a partir dele construirmos o modelo relacional seguindo as regras de transformação corretamente, o modelo relacional resultante estará, provavelmente, normalizado. Mas, nem sempre os modelos que nos deparamos são implementados dessa forma e, quando isso acontece, o suporte ao banco de dados é dificultado.

Em ambos os casos, é necessário aplicar as técnicas de normalização, ou para normalizar (segundo caso citado), ou apenas para validar o esquema criado (primeiro caso citado). Aplicando as regras descritas a seguir, é possível garantir um banco de dados mais íntegro, sem redundâncias e inconsistências.

### Existem 3 formas normais mais conhecidas:

* **1FN - 1ª Forma Normal:** todos os atributos de uma tabela devem ser atômicos, ou seja, a tabela não deve conter grupos repetidos e nem atributos com mais de um valor. Para deixar nesta forma normal, é preciso identificar a chave primária da tabela, identificar a(s) coluna(s) que tem dados repetidos e removê-la(s), criar uma nova tabela com a chave primária para armazenar o dado repetido e, por fim, criar uma relação entre a tabela principal e a tabela secundária. Por exemplo, considere a tabela Pessoas a seguir.

PESSOAS = {ID+ NOME + ENDERECO + TELEFONES}

Ela contém a chave primária ID e o atributo TELEFONES é um atributo multivalorado e, portanto, a tabela não está na 1FN. Para deixá-la na 1FN, vamos criar uma nova tabela chamada TELEFONES que conterá PESSOA\_ID como chave estrangeira de PESSOAS e TELEFONE como o valor multivalorado que será armazenado.

PESSOAS = { ID + NOME + ENDERECO }

TELEFONES = { PESSOA\_ID + TELEFONE }

* **2FN - 2ª Forma Normal:** antes de mais nada, para estar na 2FN é preciso estar na 1FN. Além disso, todos os atributos não chaves da tabela devem depender unicamente da chave primária (não podendo depender apenas de parte dela). Para deixar na segunda forma normal, é preciso identificar as colunas que não são funcionalmente dependentes da chave primária da tabela e, em seguida, remover essa coluna da tabela principal e criar uma nova tabela com esses dados. Por exemplo, considere a tabela ALUNOS\_CURSOS a seguir.

ALUNOS\_CURSOS = { ID\_ALUNO + ID\_CURSO + NOTA + DESCRICAO\_CURSO }

Nessa tabela, o atributo DESCRICAO\_CURSO depende apenas da chave primária ID\_CURSO. Dessa forma, a tabela não está na 2FN. Para tanto, cria-se uma nova tabela chamada CURSOS que tem como chave primária ID\_CURSO e atributo DESCRICAO retirando, assim, o atributo DESCRICAO\_CURSO da tabela ALUNOS\_CURSOS.

ALUNOS\_CURSOS = {ID\_ALUNO + ID\_CURSO + NOTA}

CURSOS = {ID\_CURSO + DESCRICAO}

* **3FN - 3ª Forma Normal:** para estar na 3FN, é preciso estar na 2FN. Além disso, os atributos não chave de uma tabela deve ser mutuamente independentes e dependentes unicamente e exclusivamente da chave primária (um atributo B é funcionalmente dependente de A se, e somente se, para cada valor de A só existe um valor de B). Para atingir essa forma normal, é preciso identificar as colunas que são funcionalmente dependentes das outras colunas não chave e extraí-las para outra tabela. Considere, como exemplo, a tabela FUNCIONARIOS a seguir.

FUNCIONARIOS = { ID + NOME + ID\_CARGO + DESCRICAO\_CARGO }

O atributo DESCRICAO\_CARGO depende exclusivamente de ID\_CARGO (atributo não chave) e, portanto, deve-se criar uma nova tabela com esses atributos. Dessa forma, ficamos com as seguintes tabelas:

FUNCIONARIOS = { ID + NOME + ID\_CARGO }

CARGOS = { ID\_CARGO + DESCRICAO }