Formas Normais

Modelagem de Banco de Dados

Introdução

As **Formas Normais são uma série de procedimentos aplicados** em um banco de dados para garantir que as suas tabelas estejam bem estruturadas e não contenham qualquer tipo de anomalia, seja de inclusão, atualização ou exclusão.

Damos a essa série de procedimentos o nome de **Normalização**.

5 Conceitos Importantes

Mas antes de dar início à apresentação das formas normais, vamos abordar **5 conceitos** necessários para uma melhor compreensão.

Os conceitos que falaremos agora serão:

- Dependência Funcional
- Dependência Funcional Parcial
- Dependência Funcional Transitiva
- Atributos Multivalorados
- Atributos Compostos

Conceito 1: Dependência Funcional

Uma dependência funcional é um relacionamento entre dois ou mais atributos, de forma que o valor de um atributo identifique o valor para cada um dos outros atributos, ou seja, um atributo está relacionado a outro.

CLIENTE

CPF	Nome
111	Ana
222	Bruno
333	Carla
444	Diego
555	Ana

Conceito 2: Dependência Funcional Parcial

Uma dependência funcional parcial ocorre quando os atributos não chave (não identificadores) não dependam de toda a chave primária quando ela for composta.

BOLETIM

Matrícula_Aluno	Cod_Disciplina	Período	Nome_Disciplina	Nota
111	1	1	Cálculo I	8
111	2	1	Física I	9.5
111	3	1	Contabilidade I	10
111	4	1	Química	7.5
111	5	1	Intro. Eng.	8.5

Conceito 3: Dependência Funcional Transitiva

Quando um ou mais campos de uma entidade não são dependentes diretamente da chave primária ou de parte dela, mas sim dependente de outro campo da tabela (campo este que não a Chave Primária), temos uma dependência funcional transitiva.

FUNCIONÁRIO

ID_Funcionario	Nome_Funcionario	ID_Cargo	Nome_Cargo	Salario
1	João	1	Analista	3500
2	Katia	1	Analista	3500
3	Luis	2	Técnico	4100
4	Maria	2	Técnico	4100
5	Neto	3	Assistente	2200

Conceito 4: Atributos Multivalorados

Atributos multivalorados são atributos que podem conter mais de um valor para um mesmo registro.

FUNCIONÁRIO

ID_Funcionario	Nome_Funcionario	Telefone
1	João	(21) 99999-0001 (21) 99999-0002 (21) 99999-0003
2	Katia	(21) 99999-0004
3	Luis	(21) 99999-0005 (21) 99999-0006
4	Maria	(21) 99999-0007
5	Neto	(21) 99999-0008

Conceito 5: Atributos Compostos

Atributos compostos são atributos que poderiam ser subdivididos em vários atributos.

ID_Funcionario	Nome_Funcionario	Endereço
1	João	Av. Mem de Sá, 100, apto 101 – Centro – Rio de Janeiro - RJ
2	Katia	Av. Portugal, 324, casa 01 – Urca – Rio de Janeiro – RJ
3	Luis	Av. Vieira Souto, 1300, apto 802, Leblon – Rio de Janeiro – RJ

O ideal seria dividir esta única coluna em várias colunas, para que assim a gente elimine o atributo composto.

ID_Funcionario	Nome_Funcionario	Endereço	Bairro	Cidade	UF
1	João	Av. Mem de Sá, 100, apto 101	Centro	Rio de Janeiro	RJ
2	Katia	Av. Portugal, 324, casa 01	Urca	Rio de Janeiro	RJ
3	Luis	Av. Vieira Souto, 1300, apto 802	Leblon	Rio de Janeiro	RJ

O que é a Normalização?

Podemos definir a **Normalização** como uma sequência de passos e verificações aplicadas a um banco de dados com o objetivo de eliminar, ou pelo menos minimizar, as redundâncias e inconsistências no banco. Tal procedimento é feito a partir da identificação de anomalias de inserção, exclusão e atualização em uma relação, decompondo-a em relações melhor estruturadas e minimizando a redundância.

Eliminar a redundância nos dados possui algumas vantagens:

- Reduzir o espaço necessário para armazenar o banco de dados
- Melhorar a organização dos dados
- Reduzir o impacto de atualizações, inserções e exclusões nos dados dos bancos de dados

O que é a Normalização?

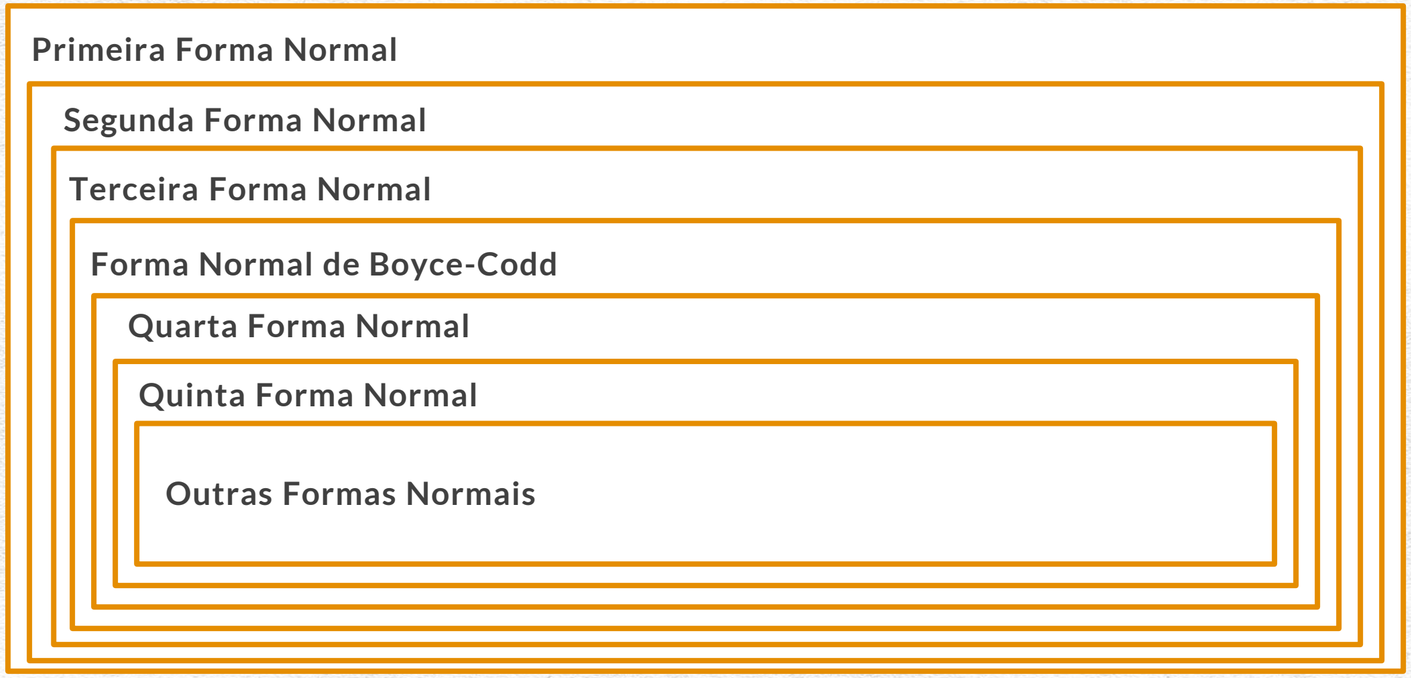
O processo de normalização é aplicado em etapas, conhecidas como **Formas Normais**., que vão garantir que o banco de dados fique bem estruturado.

Existem uma série de formas normais na literatura, são elas:

- Primeira Forma Normal
- Segunda Forma Normal
- Terceira Forma Normal
- Forma Normal de Boyce-Codd
- Quarta Forma Normal
- Quinta Forma Normal

O que é a Normalização?

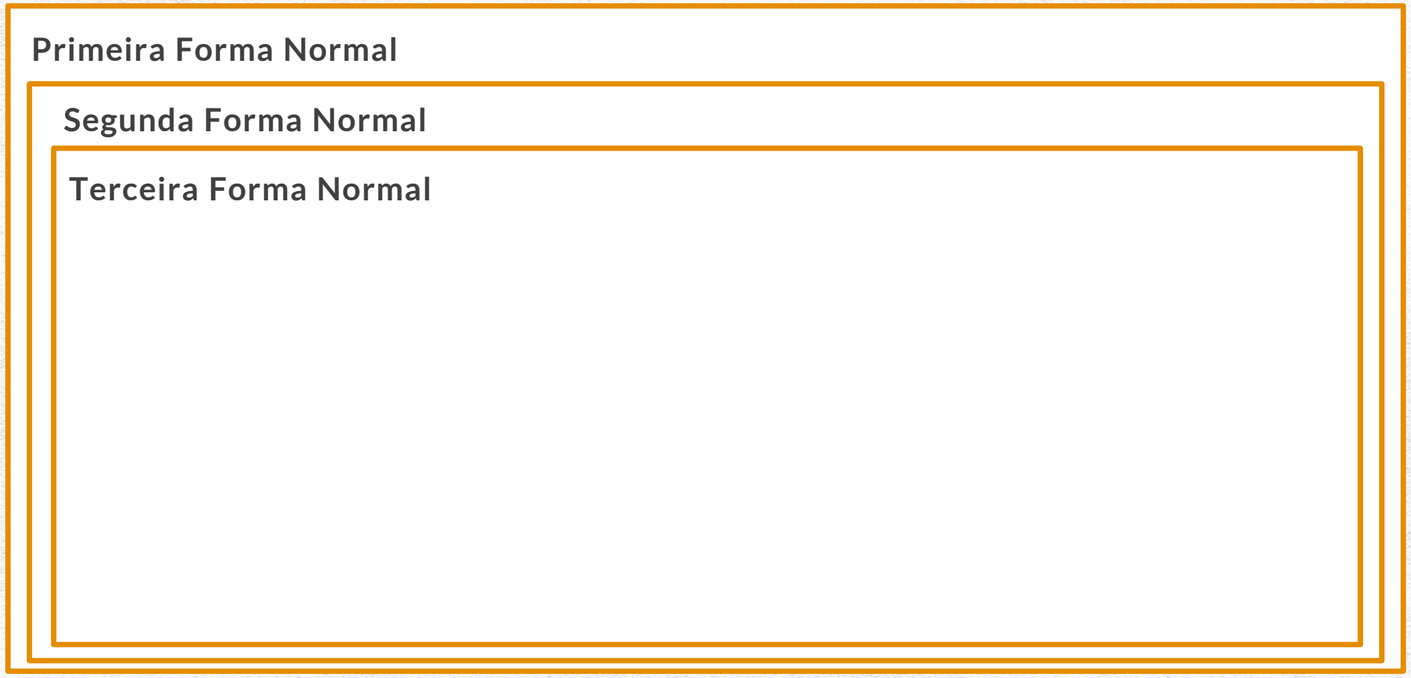
As Formas Normais é como se fossem fases de um jogo. Você só pode passar para a próxima se estiver adequado à Forma anterior.



O que é a Normalização?

Para muitos autores, a aplicação das três primeiras já é suficiente para garantir que o banco de dados não terá redundâncias e inconsistências.

No curso, vamos focar portanto nas três primeiras formas normais.



Primeira Forma Normal (1FN)

A primeira forma normal tem como objetivo eliminar atributos multivalorados e atributos compostos.

Passo a passo da 1FN

Em resumo, para adequar uma tabela que não está na 1FN é necessário realizar os seguintes passos:

1. Identificar a existência de atributos multivalorados e atributos compostos.
2. Criar uma tabela para armazenar os dados do atributo multivalorado. As colunas dessa nova tabela devem ser compostas por: (1) atributo multivalorado da tabela original e (2) chave primária da tabela original. A chave primária da tabela original se transforma na chave estrangeira da nova tabela.
3. Remover o atributo multivalorado da tabela original.
4. Na tabela original, para cada atributo composto, criar uma nova coluna para cada informação a ser desmembrada.

Exemplo Prático: Tabela PESSOA

Vejamos um exemplo. Na tabela abaixo, temos que uma série de atributos sobre a entidade PESSOA. Essa tabela não está na Primeira Forma Normal. Vamos fazer a adequação.

PESSOA

CPF	Nome	Sexo	Localização	Telefone
111	Ana	F	Rio de Janeiro, RJ	999-444, 999-000
222	Bruno	M	São Paulo, SP	888-888, 444-333
333	Carla	F	Belo Horizonte, MG	555-777
444	Diego	M	Vitória, ES	999-999

Passo 1 da 1FN

1 **Identificar a existência de atributos multivalorados (não atômicos) e atributos compostos.**

Na tabela em questão, temos que o atributo **Localização** é um **atributo composto**, o que significa que ele poderia ser desmembrado em mais de uma informação: **Cidade** e **Estado**.

PESSOA

CPF	Nome	Sexo	Localização	Telefone
111	Ana	F	Rio de Janeiro, RJ	999-444, 999-000
222	Bruno	M	São Paulo, SP	888-888, 444-333
333	Carla	F	Belo Horizonte, MG	555-777
444	Diego	M	Vitória, ES	999-999

Atributo Composto

Passo 1 da 1FN

1 **Identificar a existência de atributos multivalorados (não atômicos) e atributos compostos.**

Por outro lado, o atributo **Telefone** é um **atributo multivalorado** (não atômico). Ou seja, um atributo que contém, na mesma célula, várias ocorrências daquele mesmo atributo.

PESSOA

CPF	Nome	Sexo	Localização	Telefone
111	Ana	F	Rio de Janeiro, RJ	999-444, 999-000
222	Bruno	M	São Paulo, SP	888-888, 444-333
333	Carla	F	Belo Horizonte, MG	555-777
444	Diego	M	Vitória, ES	999-999

Atributo Multivalorado

Passo 2 da 1FN

2 **Criar uma tabela para armazenar os dados do atributo multivalorado. As colunas dessa nova tabela devem ser compostas por: (1) atributo multivalorado da tabela original e (2) chave primária da tabela original. A chave primária da tabela original se transforma na chave estrangeira da nova tabela.**

Na tabela original, identificamos como chave primária o atributo CPF, e como atributo multivalorado a coluna Telefone. A partir dessas duas colunas, criamos uma tabela nova.

PESSOA

CPF	Nome	Sexo	Localização	Telefone
111	Ana	F	Rio de Janeiro, RJ	999-444, 999-000
222	Bruno	M	São Paulo, SP	888-888, 444-333
333	Carla	F	Belo Horizonte, MG	555-777
444	Diego	M	Vitória, ES	999-999

CPF	Telefone
111	999-444
111	999-000
222	888-888
222	444-333
333	555-777
444	999-999

Passo 2 da 1FN

2 **Criar uma tabela para armazenar os dados do atributo multivalorado. As colunas dessa nova tabela devem ser compostas por: (1) atributo multivalorado da tabela original e (2) chave primária da tabela original. A chave primária da tabela original se transforma na chave estrangeira da nova tabela.**

Na tabela original, identificamos como chave primária o atributo CPF, e como atributo multivalorado a coluna Telefone. A partir dessas duas colunas, criamos uma tabela nova.

PESSOA

CPF	Nome	Sexo	Localização	Telefone
111	Ana	F	Rio de Janeiro, RJ	999-444, 999-000
222	Bruno	M	São Paulo, SP	888-888, 444-333
333	Carla	F	Belo Horizonte, MG	555-777
444	Diego	M	Vitória, ES	999-999

CPF	Telefone
111	999-444
111	999-000
222	888-888
222	444-333
333	555-777
444	999-999

A chave primária da tabela original se transforma na chave estrangeira da nova tabela.

Passo 2 da 1FN

2 **Criar uma tabela para armazenar os dados do atributo multivalorado. As colunas dessa nova tabela devem ser compostas por: (1) atributo multivalorado da tabela original e (2) chave primária da tabela original. A chave primária da tabela original se transforma na chave estrangeira da nova tabela.**

Na tabela original, identificamos como chave primária o atributo CPF, e como atributo multivalorado a coluna Telefone. A partir dessas duas colunas, criamos uma tabela nova.

PESSOA

CPF	Nome	Sexo	Localização	Telefone
111	Ana	F	Rio de Janeiro, RJ	999-444, 999-000
222	Bruno	M	São Paulo, SP	888-888, 444-333
333	Carla	F	Belo Horizonte, MG	555-777
444	Diego	M	Vitória, ES	999-999

O atributo multivalorado da tabela original se transforma em uma coluna da tabela nova.

CPF	Telefone
111	999-444
111	999-000
222	888-888
222	444-333
333	555-777
444	999-999

Passo 2 da 1FN

3 **Remover o atributo multivalorado da tabela original.**

Agora que criamos uma nova tabela para armazenar as informações de telefones das pessoas, não precisamos mais da coluna Telefone na tabela PESSOA.

PESSOA

CPF	Nome	Sexo	Localização	Telefone
111	Ana	F	Rio de Janeiro, RJ	999-444, 999-000
222	Bruno	M	São Paulo, SP	888-888, 444-333
333	Carla	F	Belo Horizonte, MG	555-777
444	Diego	M	Vitória, ES	999-999

TELEFONE

CPF	Telefone
111	999-444
111	999-000
222	888-888
222	444-333
333	555-777
444	999-999

Passo 2 da 1FN

3 **Remover o atributo multivalorado da tabela original.**

Agora que criamos uma nova tabela para armazenar as informações de telefones das pessoas, não precisamos mais da coluna Telefone na tabela PESSOA.

PESSOA

CPF	Nome	Sexo	Localização	Telefone
111	Ana	F	Rio de Janeiro, RJ	999-444, 999-000
222	Bruno	M	São Paulo, SP	888-888, 444-333
333	Carla	F	Belo Horizonte, MG	555-777
444	Diego	M	Vitória, ES	999-999

TELEFONE

CPF	Telefone
111	999-444
111	999-000
222	888-888
222	444-333
333	555-777
444	999-999

Passo 2 da 1FN

3 **Remover o atributo multivalorado da tabela original.**

Agora que criamos uma nova tabela para armazenar as informações de telefones das pessoas, não precisamos mais da coluna Telefone na tabela PESSOA.

PESSOA

CPF	Nome	Sexo	Localização
111	Ana	F	Rio de Janeiro, RJ
222	Bruno	M	São Paulo, SP
333	Carla	F	Belo Horizonte, MG
444	Diego	M	Vitória, ES

TELEFONE

CPF	Telefone
111	999-444
111	999-000
222	888-888
222	444-333
333	555-777
444	999-999

Passo 2 da 1FN

4 **Na tabela original, para cada atributo composto, criar uma nova coluna para cada informação a ser desmembrada.**

A coluna **Localização** é um **atributo composto**. Sabemos que podemos separá-la em mais de uma, com as informações de **Cidade** e **Estado**.

PESSOA

CPF	Nome	Sexo	Localização
111	Ana	F	Rio de Janeiro, RJ
222	Bruno	M	São Paulo, SP
333	Carla	F	Belo Horizonte, MG
444	Diego	M	Vitória, ES

Passo 2 da 1FN

4 **Na tabela original, para cada atributo composto, criar uma nova coluna para cada informação a ser desmembrada.**

A coluna **Localização** é um **atributo composto**. Sabemos que podemos separá-la em mais de uma, com as informações de **Cidade** e **Estado**.

PESSOA

CPF	Nome	Sexo	Localização
111	Ana	F	Rio de Janeiro, RJ
222	Bruno	M	São Paulo, SP
333	Carla	F	Belo Horizonte, MG
444	Diego	M	Vitória, ES

PESSOA

CPF	Nome	Sexo	Cidade	Estado
111	Ana	F	Rio de Janeiro	RJ
222	Bruno	M	São Paulo	SP
333	Carla	F	Belo Horizonte	MG
444	Diego	M	Vitória	ES

Passo 2 da 1FN

4 **Na tabela original, para cada atributo composto, criar uma nova coluna para cada informação a ser desmembrada.**

A coluna **Localização** é um **atributo composto**. Sabemos que podemos separá-la em mais de uma, com as informações de **Cidade** e **Estado**.

PESSOA

CPF	Nome	Sexo	Localização
111	Ana	F	Rio de Janeiro, RJ
222	Bruno	M	São Paulo, SP
333	Carla	F	Belo Horizonte, MG
444	Diego	M	Vitória, ES

PESSOA

CPF	Nome	Sexo	Cidade	Estado
111	Ana	F	Rio de Janeiro	RJ
222	Bruno	M	São Paulo	SP
333	Carla	F	Belo Horizonte	MG
444	Diego	M	Vitória	ES

A coluna Localização foi divida em duas, contendo a informação de Cidade e Estado separadamente.

Exemplo Prático: Tabela PESSOA

E assim, corrigimos a tabela criando duas outras tabelas, fazendo com que no final a Primeira Forma Normal seja atingida.

PESSOA

CPF	Nome	Sexo	Localização	Telefone
111	Ana	F	Rio de Janeiro, RJ	999-444, 999-000
222	Bruno	M	São Paulo, SP	888-888, 444-333
333	Carla	F	Belo Horizonte, MG	555-777
444	Diego	M	Vitória, ES	999-999

PESSOA

CPF	Nome	Sexo	Cidade	Estado
111	Ana	F	Rio de Janeiro	RJ
222	Bruno	M	São Paulo	SP
333	Carla	F	Belo Horizonte	MG
444	Diego	M	Vitória	ES

TELEFONE

CPF	Telefone
111	999-444
111	999-000
222	888-888
222	444-333
333	555-777
444	999-999

Segunda Forma Normal (2FN)

Uma tabela encontra-se na segunda forma normal se ela atende 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.

Aqui, vamos eliminar a dependência funcional parcial.

Passo a passo da 2FN

Em resumo, para adequar uma tabela que não está na 2FN é necessário realizar os seguintes passos:

1. Identificar se existe na tabela uma chave primária composta.
2. Identificar cada atributo não chave que dependa de apenas parte da chave primária composta.
3. Criar uma tabela para cada conjunto de atributos não chave que dependam de parte da chave primária da tabela e adicionar estes atributos não chaves na tabela. As chaves primárias nas novas tabelas devem ter como base as chaves primárias da tabela original.
4. Remover os atributos não chave da tabela original que dependam de parte da chave primária.

Exemplo Prático: Tabela FUNCIONARIO_PROJETO

Exemplo: a tabela abaixo contém duas colunas que formam uma chave primária: **id_func** e **id_proj**. Repare agora, por exemplo, na coluna **nome_func**. Essa coluna depende apenas de parte da chave primária, ou seja, ela depende apenas do id_func. Da mesma forma que as colunas nome_proj e local_proj dependem apenas da coluna id_proj.

id_func	id_proj	horas_trabalhadas	nome_func	nome_proj	local_proj
1	A	10	Luis	Plan. Est.	RJ
2	B	20	Marta	Plano Neg.	SP
1	B	15	Luis	Plano Neg.	SP
3	C	30	Neide	Map. Processos	MG

Exemplo Prático: Tabela FUNCIONARIO_PROJETO

1 **Identificar se existe na tabela uma chave primária composta.**

A chave primária da tabela FUNCIONARIO_PROJETO é composta por duas colunas: id_func e id_proj.

Chave Primária Composta

id_func	id_proj	horas_trabalhadas	nome_func	nome_proj	local_proj
1	A	10	Luis	Plan. Est.	RJ
2	B	20	Marta	Plano Neg.	SP
1	B	15	Luis	Plano Neg.	SP
3	C	30	Neide	Map. Processos	MG

Exemplo Prático: Tabela FUNCIONARIO_PROJETO

2 **Identificar cada atributo não chave que dependa de apenas parte da chave primária composta.**

O atributo não chave “nome_func” depende apenas da coluna “id_func” (parte da chave primária).

Chave Primária Composta

id_func	id_proj	horas_trabalhadas	nome_func	nome_proj	local_proj
1	A	10	Luis	Plan. Est.	RJ
2	B	20	Marta	Plano Neg.	SP
1	B	15	Luis	Plano Neg.	SP
3	C	30	Neide	Map. Processos	MG

O nome do funcionário depende apenas do id_func (e não de id_proj)

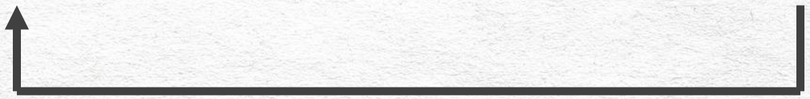
Exemplo Prático: Tabela FUNCIONARIO_PROJETO

2 **Identificar cada atributo não chave que dependa de apenas parte da chave primária composta.**

Já os atributos não chave “nome_proj” e “local_proj” dependem apenas da coluna “id_proj” (parte da chave primária).

Chave Primária Composta

id_func	id_proj	horas_trabalhadas	nome_func	nome_proj	local_proj
1	A	10	Luis	Plan. Est.	RJ
2	B	20	Marta	Plano Neg.	SP
1	B	15	Luis	Plano Neg.	SP
3	C	30	Neide	Map. Processos	MG



O nome do projeto depende apenas do id_proj (e não de id_func)

Exemplo Prático: Tabela FUNCIONARIO_PROJETO

2 **Identificar cada atributo não chave que dependa de apenas parte da chave primária composta.**

Já os atributos não chave “nome_proj” e “local_proj” dependem apenas da coluna “id_proj” (parte da chave primária).

Chave Primária Composta

id_func	id_proj	horas_trabalhadas	nome_func	nome_proj	local_proj
1	A	10	Luis	Plan. Est.	RJ
2	B	20	Marta	Plano Neg.	SP
1	B	15	Luis	Plano Neg.	SP
3	C	30	Neide	Map. Processos	MG

O local do projeto depende apenas do id_proj (e não de id_func)

Exemplo Prático: Tabela FUNCIONARIO_PROJETO

2 **Identificar cada atributo não chave que dependa de apenas parte da chave primária composta.**

E o atributo não chave “horas_trabalhadas”?

id_func	id_proj	horas_trabalhadas	nome_func	nome_proj	local_proj
1	A	10	Luis	Plan. Est.	RJ
2	B	20	Marta	Plano Neg.	SP
1	B	15	Luis	Plano Neg.	SP
3	C	30	Neide	Map. Processos	MG

Nesta tabela, este atributo depende da chave primária como um todo. As horas trabalhadas estão associadas a um determinado funcionário alocado em um determinado projeto.

Exemplo Prático: Tabela FUNCIONARIO_PROJETO

3 **Criar uma tabela para cada conjunto de atributos não chave que dependam de parte da chave primária da tabela e adicionar estes atributos não chaves na tabela. As chaves primárias nas novas tabelas devem ter como base as chaves primárias da tabela original.**

Temos portanto dois conjuntos de atributos não chave dependentes de parte da chave primária, são eles:

- nome_func → depende apenas de id_func
- nome_proj e local_proj → dependem apenas de id_proj

Portanto, criaremos mais duas tabelas, uma para cada conjunto de atributos não chaves não dependentes de toda a chave primária:

1. FUNCIONARIO
2. PROJETO

Exemplo Prático: Tabela FUNCIONARIO_PROJETO

4 **Remover os atributos não chave da tabela original que dependam de parte da chave primária.**

Os mesmos atributos não-chave serão removidos da tabela original, e existirão apenas nas novas tabelas.

- nome_func → será removida por depender apenas de id_func
- nome_proj e local_proj → serão removidas por dependerem apenas de id_proj

Exemplo Prático: Tabela FUNCIONARIO_PROJETO

Assim, podemos dividir nossa tabela inicial em três tabelas separadas.

id_func	id_proj	horas_trabalhadas	nome_func	nome_proj	local_proj
1	A	10	Luis	Plan. Est.	RJ
2	B	20	Marta	Plano Neg.	SP
1	B	15	Luis	Plano Neg.	SP
3	C	30	Neide	Map. Processos	MG

FUNCIONARIO

id_func	nome_func
1	Luis
2	Marta
3	Neide

FUNCIONARIO_PROJETO

id_func	id_proj	horas_trabalhadas
1	A	10
2	B	20
1	B	15
3	C	30

PROJETO

id_proj	nome_proj	local_proj
A	Plan. Est.	RJ
B	Plano Neg.	SP
C	Map. Processos	MG

Terceira Forma Normal (3FN)

Se uma tabela está na primeira e segunda forma normal, mas ao analisarmos um registro encontramos um atributo não chave dependente de outro atributo não chave, precisaremos corrigir a tabela para a terceira forma normal.

Aqui basicamente estaremos corrigindo a dependência funcional transitiva.

Passo a passo da 3FN

Em resumo, para adequar uma tabela que não está na 3FN é necessário realizar os seguintes passos:

1. Identificar cada grupo de atributos não-chave que dependam de outros atributos não-chave.
2. Criar uma tabela para armazenar os atributos (ou conjunto de atributos) não-chave que não estão relacionados à chave primária da tabela original. Definir como chave primária da tabela criada o atributo que é capaz de obter os dados não chave da tabela original e mover os atributos não chave que não são obtidos exclusivamente pela chave primária da tabela original para a nova tabela.
3. Definir como chave estrangeira o atributo que é capaz de obter os dados não chaves da tabela original.

Exemplo Prático: Tabela FUNCIONARIO

Vejamos um exemplo. Na tabela abaixo, temos uma lista de funcionários e suas respectivas informações. A chave primária dessa tabela é a coluna **id_func**.

id_func	nome_func	Sexo	id_dep	nome_dep	gerente_dep
1	Paula	F	100	Finanças	André
2	Rodrigo	M	101	RH	Bruna
3	Sandra	F	102	TI	Caio
4	Tiago	M	101	RH	Bruna

Porém, existem atributos dessa tabela que não dependem da chave primária da tabela.

As colunas nome_dep e gerente_dep não são definidas pela chave primária (id_func), mas sim pelo atributo id_dep. Ou seja, temos atributos não chave (nome_dep e gerente_dep) dependendo exclusivamente de um outro atributo não chave (id_dep).

Exemplo Prático: Tabela FUNCIONARIO

1 **Identificar cada grupo de atributos não-chave que dependam de outros atributos não-chave.**

Existem atributos dessa tabela que não dependem da chave primária da tabela.

As colunas nome_dep e gerente_dep não são definidas pela chave primária (id_func), mas sim pelo atributo id_dep. Ou seja, temos atributos não chave (nome_dep e gerente_dep) dependendo exclusivamente de um outro atributo não chave (id_dep).

id_func	nome_func	Sexo	id_dep	nome_dep	gerente_dep
1	Paula	F	100	Finanças	André
2	Rodrigo	M	101	RH	Bruna
3	Sandra	F	102	TI	Caio
4	Tiago	M	101	RH	Bruna

Chave Primária

Terceira Forma Normal (3FN)

2 **Criar uma tabela para armazenar os atributos (ou conjunto de atributos) não-chave que não estão relacionados à chave primária da tabela original. Definir como chave primária da tabela criada o atributo que é capaz de obter os dados não chave da tabela original e mover os atributos não chave que não são obtidos exclusivamente pela chave primária da tabela original para a nova tabela.**

Criamos uma nova tabela chamada DEPARTAMENTO, contendo os atributos não-chave que não dependem da chave primária da tabela original + atributo não-chave da tabela original que serve para identificar os outros atributos não-chave.

DEPARTAMENTO

id_dep	nome_dep	gerente_dep
100	Finanças	André
101	RH	Bruna
102	TI	Caio

Terceira Forma Normal (3FN)

3 **Definir como chave estrangeira o atributo que é capaz de obter os dados não chaves da tabela original.**

Desmembramos a nossa tabela original em duas. Na tabela original mantemos apenas a chave primária e os atributos não chave associados a ela e transformamos id_dep em chave estrangeira, para se conectar com a tabela nova criada, de DEPARTAMENTO.

TABELA ORIGINAL

id_func	nome_func	Sexo	id_dep	nome_dep	gerente_dep
1	Paula	F	100	Finanças	André
2	Rodrigo	M	101	RH	Bruna
3	Sandra	F	102	TI	Caio
4	Tiago	M	101	RH	Bruna

FUNCIONARIO

id_func	nome_func	Sexo	id_dep
1	Paula	F	100
2	Rodrigo	M	101
3	Sandra	F	102
4	Tiago	M	101

Chave Estrangeira

DEPARTAMENTO

id_dep	nome_dep	gerente_dep
100	Finanças	André
101	RH	Bruna
102	TI	Caio

Resumo

Resumo

- Um banco de dados mal projetado pode apresentar anomalias de inserção, exclusão e atualização.

Resumo

- Um banco de dados mal projetado pode apresentar anomalias de inserção, exclusão e atualização.
- Anomalias em bancos de dados geram redundâncias e inconsistências, o que prejudica o bom desempenho do banco de dados.

Resumo

- Um banco de dados mal projetado pode apresentar anomalias de inserção, exclusão e atualização.
- Anomalias em bancos de dados geram redundâncias e inconsistências, o que prejudica o bom desempenho do banco de dados.
- Normalização é um processo de adequação do banco de dados por meio de Formas Normais, a fim de eliminar anomalias nos bancos de dados.

Resumo

- Um banco de dados mal projetado pode apresentar anomalias de inserção, exclusão e atualização.
- Anomalias em bancos de dados geram redundâncias e inconsistências, o que prejudica o bom desempenho do banco de dados.
- Normalização é um processo de adequação do banco de dados por meio de Formas Normais, a fim de eliminar anomalias nos bancos de dados.
- Podemos considerar que um banco de dados está normalizado e livre de redundâncias e inconsistências se estiver adequado à Primeira, Segunda e Terceira Formas Normais.

SQL IMPRESSIONADOR