UVV

Melhor Universidade Particular do Brasil *(MEC)*

THE Times Higher Education

Pacto Nacional pela Primeira Infância

universidades empreendedoras

QS WORLD UNIVERSITY RANKINGS

AD Scientific Index

Prêmio Sebrae de Educação Empreendedora

CENTELHA

QS

THE IMPACT RANKINGS

DESIGN E DESENVOLVIMENTO DE BANCO DE DADOS I
UNIDADE IV

Prof. Msc. Gustavo Nunes Rocha

Normalização de banco de dados

"Técnicas de racionalização das estruturas de dados de um sistema, eliminando redundâncias, problemas de manipulação e armazenamento".

Normalização é um processo através do qual esquemas de relação, que não sejam satisfatórios às características do modelo relacional, são decompostos em esquemas menores que satisfaçam as propriedades desejáveis.

Normalização de banco de dados

Isso se relaciona ao fato de que, ao projetar um banco de dados, é importante determinar quais informações pertencem a quais tabelas, quais colunas devem existir e como as tabelas se relacionam umas com as outras.

A normalização auxilia nesse processo, ajudando a identificar quais atributos devem estar em que tabelas e como estabelecer as chaves primárias e estrangeiras para manter a integridade referencial.

Normalização de banco de dados

A normalização inicialmente foi proposta como uma ferramenta de auxílio no projeto físico para a definição de relações, porém na prática tornou-se uma ferramenta de verificação, pois serve para verificar se os esquemas do projeto físico satisfazem algumas características básicas.

Normalização de banco de dados

"Evita problemas provocados por falhas no Projeto do Banco de Dados, eliminando uma incorreta disposição dos dados e sua redundância".

APLICAÇÃO DE REGRAS A UM CONJUNTO DE DADOS

Eliminação de redundância e dependência de dados

Aumento da confiabilidade dos dados

Validação com o modelo relacional

Medidas de qualidade na normalização

Na normalização, são analisadas algumas medidas de qualidade para o projeto de um esquema de relação. Estas medidas de qualidade visam, por exemplo, evitar um mau uso da memória. As medidas são as seguintes:

1 – Correta representação semântica – os dados devem ser projetados de forma a terem seus significados bem definidos e coerentes com o que realmente querem representar;

2 – Redução de valores redundantes – sempre que possível deve-se reduzir ao máximo os valores redundantes desnecessários, ou seja, valores que muitas vezes aparecem repetidos quando isto não seria preciso;

Medidas de qualidade na normalização

3 – Redução de valores nulos – sempre que possível deve-se reduzir o número de atributos que por alguma razão receberão muitos valores nulos;

4 – Não geração de tuplas espúrias (sem sentido) – durante o processo de normalização deve-se atentar para evitar que sejam geradas tuplas que não façam sentido diante da realidade, isto pode ocorrer devido a alguma decomposição.

Dependências funcionais

A dependência funcional entre dois conjuntos de atributos, denotada como $x \rightarrow y$, estabelece que os valores do conjunto de atributos y são determinados pelos valores do conjunto de atributos x em um esquema de relação R.

Isso significa que se você possui duas tuplas (registros) t1 e t2 na instância da relação R, e os valores das colunas correspondentes ao conjunto de atributos x são iguais em t1 e t2 (ou seja, t1(x) = t2(x)), então os valores das colunas correspondentes ao conjunto de atributos y também devem ser iguais em t1 e t2 (ou seja, t1(y) = t2(y)). Em termos mais simples, quando você conhece os valores de x em uma tupla, você pode determinar os valores correspondentes de y para aquela tupla.

Dependências funcionais

Considere uma tabela chamada "Pedidos" que armazena informações sobre os pedidos feitos por clientes. Essa tabela possui os seguintes atributos:

Número do Pedido (Número_Pedido)

ID do Cliente (ID_Cliente)

Nome do Cliente (Nome_Cliente)

Data do Pedido (Data_Pedido)

Nesse cenário, podemos observar algumas dependências funcionais:

Dependências funcionais

Número_Pedido → Data_Pedido: O número de um pedido determina a data em que o pedido foi feito. Ou seja, se você conhece o número de um pedido, pode descobrir a data correspondente do pedido.

ID_Cliente → Nome_Cliente: O ID de um cliente determina o nome do cliente. Isso significa que, se você sabe o ID de um cliente, pode determinar o nome associado a esse ID.

Essas dependências funcionais podem ser expressas da seguinte forma:

Número_Pedido → Data_Pedido

ID_Cliente → Nome_Cliente

Dependências funcionais

Imagine que temos a seguinte instância da tabela "Pedidos":

Número_Pedido	ID_Cliente	Nome_Cliente	Data_Pedido
1001	101	João	2023-01-15
1002	102	Maria	2023-02-20
1003	101	João	2023-03-10

Nesse exemplo, a primeira dependência funcional (Número_Pedido → Data_Pedido) é confirmada, pois os números de pedido únicos estão associados a datas específicas de pedidos.

A segunda dependência funcional (ID_Cliente → Nome_Cliente) também é confirmada, já que o ID do cliente "101" está sempre associado ao nome "João".

Dependências funcionais

Essas dependências funcionais ajudam na organização dos dados e na detecção de problemas de redundância ou inconsistência. Por exemplo, se você tentasse inserir um novo pedido com um ID de cliente que não existe na tabela de clientes, isso seria um problema de integridade referencial, já que o nome do cliente não poderia ser determinado.

Esses exemplos ilustram como as dependências funcionais são usadas para modelar a relação entre atributos em um banco de dados e como elas são fundamentais para o projeto e a normalização de bancos de dados.

Dependências funcionais

As formas normais, como a Primeira Forma Normal (1FN), a Segunda Forma Normal (2FN), a Terceira Forma Normal (3FN) e outras, são critérios que se baseiam nas dependências funcionais para avaliar a organização dos dados em um banco de dados. Cada forma normal define um conjunto de regras que uma tabela deve seguir para ser considerada nesse estado de normalização.

Em resumo, as dependências funcionais são as bases sobre as quais a normalização de bancos de dados é construída. Elas ajudam a definir como os atributos estão relacionados e guiam o processo de projetar esquemas de banco de dados eficientes e livres de problemas de consistência e redundância.

Formas normais baseadas em pk's

O processo de normalização é realizado gradativamente através de formas normais, definidas a partir do conceito de DF (Dependência Funcional). As três principais formas normais são a Primeira Forma Normal (1FN), a Segunda Forma Normal (2FN) e a Terceira Forma Normal (3FN).

Propriedades do processo de normalização através de decomposição: Junções sem perda – uma vez definida uma decomposição, caso esta seja recomposta através de uma operação de junção, no resultado gerado não pode haver perdas.

Preservação de dependências – assegura que cada DF seja representada em algumas relações individuais resultantes após a decomposição.

Primeira forma normal – 1fn

Um esquema de relação R está na 1FN se todos os seus atributos forem atômicos e monovalorados, ou seja, não possuem valores que formam atributos compostos. Exemplo:

ESTUDANTES = {MATRÍCULA + NOME + ENDEREÇO + CODCURSO}

ENDEREÇO é um atributo composto, ENDEREÇO = {RUA + NUMERO + BAIRRO + CIDADE + UF}.

Para colocar na 1FN faz:

ESTUDANTES{MATRÍCULA+NOME+RUA+NUMERO+BAIRRO+CIDADE+UF+CODCURSO}

Primeira forma normal – 1fn

Um esquema de relação R está na 1FN se todos os seus atributos forem atômicos e monovalorados, ou seja, não possuem valores que formam atributos compostos.
Exemplo:

FUNCIONÁRIOS = {CODFUNC + NOME + CARGO + {PROJETO + DATAINI + DATAFIM}}

Para colocar na 1FN faz:

FUNCIONÁRIOS = {CODFUNC + NOME + CARGO}

FUNC_PROJ = {CODFUNC + PROJETO + DATAINI + DATAFIM}

Observação: todas as tabelas são relações na 1FN.

Segunda forma normal - 2fn

Um esquema de relação está na 2FN se: estiver na 1FN e, além disso, todo atributo que não pertença a alguma de suas chaves for totalmente dependente da sua chave primária. Em outras palavras, para que uma relação esteja na 2FN é preciso que esteja na 1FN e que, havendo uma chave primária composta, todos os dados que não são chaves dependem de toda a chave primária (a chave primária completa).

Dep. Funcional Total ou Completa

Uma dependência funcional $x \rightarrow y$ é considerada total se não houver nenhum atributo A em x tal que, ao removê-lo $(x - \{A\})$, a dependência funcional $(x - \{A\}) \rightarrow y$ ainda se mantenha. Em outras palavras, se a remoção de qualquer atributo em x resultar na manutenção da dependência funcional, ela é considerada total. Caso contrário, se a remoção de um atributo em x levar à perda da dependência funcional, $x \rightarrow y$ é considerada parcial.

Segunda forma normal – 2fn

Seja o exemplo de uma relação que represente o estoque de um estabelecimento comercial, da seguinte forma:

ESTOQUE = {PRODUTO + ALMOX + END_ALMOX + UNID_PROD + QTD + PRECO}

Não está na 2FN porque alguns dados não chave dependem somente de parte da chave, como END_ALMOX depende só de ALMOX, e UNID_PROD depende só de PRODUTO.

Segunda forma normal – 2fn

Com a normalização dessa tabela ficaria:

Produtos = {PRODUTO + UNID_PROD}

Almoxarifados = {ALMOX + END_ALMOX}

Estoque = {PRODUTO + ALMOX + QTD + PRECO}

Lembrando que:

PRODUTO (PK, FK references PRODUTOS)

ALMOX (PK, FK references ALMOXARIFADOS)

terceira forma normal – 3fn

Um esquema de relação está na 3FN se: estiver na 2FN e, além disso, nenhum atributo que não pertença a alguma das suas chaves for transitivamente dependente da sua chave primária. Em outras palavras, para que uma relação esteja na 3FN é preciso que esteja na 2FN e todo atributo, que não pertença a alguma chave for não dependente de algum outro atributo, que também não pertença a alguma chave.

Dependência Funcional Transitiva

Uma dependência funcional $x \rightarrow y$ é transitiva em um esquema de relação R quando existe um conjunto de atributos z, que não seja um subconjunto de alguma chave de R, e as dependências funcionais $x \rightarrow z$ e $z \rightarrow y$ forem válidas em R.

terceira forma normal – 3fn

Seja o exemplo de uma relação que represente os dados referentes às músicas, da seguinte forma:

Exemplo:

MÚSICA = {CÓDIGO + TÍTULO + GÊNERO + PAÍS_ORIGEM}, supondo que neste exemplo, o PAÍS_ORIGEM refere-se ao GÊNERO musical e não a música, sendo assim, apesar de estar na 2FN, a relação não está na 3FN, pois existe dependência entre GÊNERO e PAÍS_ORIGEM.

terceira forma normal – 3fn

Com a normalização ficaria:

TABELA GÊNEROS = {GÊNERO, PAÍS_ORIGEM}

TABELA MÚSICA = {CÓDIGO, TÍTULO, GÊNERO} (onde "GÊNERO" é uma chave estrangeira que referencia a tabela "GÊNEROS").

Resumindo...

Forma Normal	Teste	Solução (Normalização)
Primeira (1FN)	A relação não deve ter qualquer atributo não-atômico nem relações agrupadas.	Forme novas relações para cada atributo não-atômico ou relação aninhada.
Segunda (2FN)	Para as relações nas quais a chave primária contém múltiplos atributos, nenhum atributo não chave deve ser funcionalmente dependente de uma parte da chave primária.	Decomponha e monte uma relação para cada chave parcial com seu(s) atributo(s) dependente(s). Mantenha uma relação com a chave primária original e quaisquer atributos que sejam completamente dependentes dela em termos funcionais.
Terceira (3FN)	A relação não deve ter um atributo não-chave funcionalmente determinado por um outro atributo não chave (ou por um conjunto de atributos não-chave). Ou seja, não deve haver dependência transitiva de um atributo não-chave na chave primária.	Decomponha e monte uma relação que inclua o(s) atributo(s) não-chave que funcionalmente determine(m) outros atributos não-chave.

Quadro 6.1: Resumo das Formas Normais baseadas em chaves primárias e suas normalizações.

questionário

O que é uma dependência funcional?

Quem especifica as dependências funcionais que se mantêm (são válidas) entre os atributos de um esquema de relação?

A que se refere a expressão "relação desnormalizada"?

Defina primeira, segunda e terceira formas normais quando somente chaves primárias são consideradas. Como as definições da 2FN e 3FN, que consideram todas as chaves de uma relação, diferem daquelas que consideram somente chaves primárias?

Quais são as etapas de normalização?

Esse processo pode ser feito em até seis fases, conhecidas como formas normais, mas geralmente, ao se chegar na terceira etapa ou terceira forma normal, já é possível obter um modelo de dados estável.

Para iniciar o processo de normalização, é importante identificar o grupo de dados a ser analisado.

Tomemos como exemplo a situação abaixo:

> Temos a necessidade de criar uma tabela com informações de funcionários, onde vamos armazenar o Nome, Endereço do Escritório, e alguns recados para cada funcionário.

Exemplo de normalização

Pode-se começar com a construção da tabela abaixo:

Funcionarios			
Nome	**Endereco_escritorio**	**Recado1**	**Recado2**
José Silva	Rua Manga, 1223	Ligar em casa	Reunião 14:00
Expedito Jr.	Rua Tiradentes, 3454	Imprimir doc	Demitir José

Observe os campos Recado1 e Recado2 ...

O que deve ser feito se o aplicativo precisar adicionar um terceiro recado?

Teremos que continuar adicionando colunas na tabela. Até quando ???

Isso elimina uma correta funcionalidade do sistema, que deve crescer de acordo com a necessidade sem esse tipo de problema.

Primeira forma normal – 1fn

Observe que esta regra não é cumprida ao se repetir os campos **Recado1**

Funcionarios			
Nome	**Endereco_escritorio**	**Recado1**	**Recado2**
José Silva	Rua Manga, 1223	Ligar em casa	Reunião 14:00
Expedito Jr.	Rua Tiradentes, 3454	Imprimir doc	Demitir José

Primeira forma normal – 1fn

Ao aplicar as regras da Primeira Forma Normal tem-se a seguinte tabela:

Funcionarios			
Fun_Id	**Fun_Nome**	**Fun_Endereco_Escritorio**	**Fun_Recado**
1	José Silva	Rua Manga, 1223	Ligar em casa
2	José Silva	Rua Manga, 1223	Reunião 14:00
3	Expedito Jr.	Rua Tiradentes, 3454	Imprimir doc
4	Expedito Jr.	Rua Tiradentes, 3454	Demitir José

Primeira forma normal – 1fn

Agora a tabela está na Primeira Forma Normal (1FN).

Solucionamos o problema da limitação de campos de Recados, mas observe o novo problema gerado.

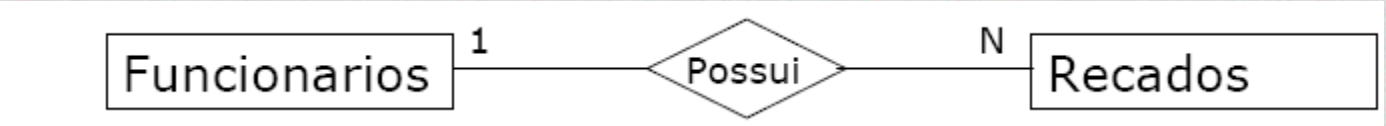
Cada vez que inserimos uma nova linha na tabela de funcionários, duplicamos todos os endereços dos escritórios e do nome do funcionários.

Este banco de dados não vai apenas ficar muito maior do que o esperado, como facilmente podemos ter dados corrompidos devido a alta redundância.

É o momento de aplicar as regras da Segunda Forma Normal (2FN).

segunda forma normal – 2fn

Extraímos os valores de Recados para uma tabela separada para que possamos adicionar mais Recados no futuro sem ter que duplicar os dados. O relacionamento irá ocorrer com a utilização do valor de chave primária.



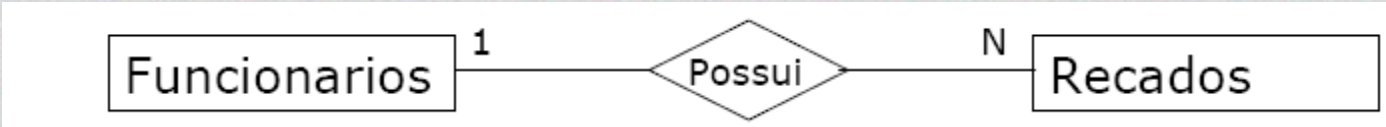
segunda forma normal – 2fn

Deve-se:

checar se a tabela está na 1FN;

identificar e representar em uma nova tabela os dados que causam a redundância;

identificar e representar em uma nova tabela os dados que não dependam única e exclusivamente da chave da tabela.



segunda forma normal – 2fn

Funcionarios

Fun_ID	Fun_Nome	Fun_Endereco_Escritorio
1	José Silva	Rua Manga, 1223
2	Expedito Jr.	Rua Tiradentes, 3454

Recados

Rec_ID	Rec_Fun_ID	Rec_Mensagem
1	1	Ligar em casa
2	1	Reunião 14:00
3	2	Imprimir doc
4	2	Demitir José

segunda forma normal – 2fn

Ótimo! Foram criadas tabelas separadas e a chave primária na tabela de funcionarios, Fun_ID, agora está relacionada à chave estrangeira na tabela dos Recados; E o campo Fun_Endereco_Escritorio ??? Se inserir um outro funcionário que atue no escritório da Rua Manga ???

Funcionarios		
Fun_ID	**Fun_Nome**	**Fun_Endereco_Escritorio**
1	José Silva	Rua Manga, 1223
2	Expedito Jr.	Rua Tiradentes, 3454
3	Milton Souza	Rua Manga, 1223

segunda forma normal – 2fn

E agora ???

O Escritório irá existir, mesmo que não exista o funcionário.

Com isso, podemos dizer que o Escritório não depende do Funcionário.

- *identificar e representar em uma nova tabela os dados que não dependam única e exclusivamente da chave da tabela.*

segunda forma normal – 2fn

Agora temos a chave primária Esc_ID na tabela de Escritorios relacionada à chave estrangeira na tabela de Funcionarios chamada Fun_Esc_ID.

Funcionarios		
Fun_ID	Fun_Nome	Fun_Esc_ID
1	José Silva	1
2	Expedito Jr.	2

Escritorios	
Esc_ID	Esc_Endereco
1	Rua Manga, 1223
2	Rua Tiradentes, 3454

segunda forma normal – 2fn

Funcionarios		
Fun_ID	**Fun_Nome**	**Fun_Esc_ID**
1	José Silva	1
2	Expedito Jr.	2

Escritorios	
Esc_ID	**Esc_Endereco**
1	Rua Manga, 1223
2	Rua Tiradentes, 3454

Recados		
Rec_ID	**Rec_Fun_ID**	**Rec_Mensagem**
1	1	Ligar em casa
2	1	Reunião 14:00
3	2	Imprimir doc
4	2	Demitir José

Terceira forma normal – 3fn

Deve-se checar se a tabela está na 2FN e identificar atributos com dependência transitiva e eliminá-los.

Dependência transitiva ocorre quando um dado pode ser obtido por meio de outro (com exceção da chave primária). Tente localizar campos que possam ser substituídos por fórmulas matemáticas. Por exemplo, se em um tabela de Clientes tivessemos os campos Cli_Nascimento e Cli_Idade.

Clientes				
Cli_ID	Cli_Nome	Cli_Nascimento	Cli_Idade	Cli_Cidade_ID
1	Marina Souza	11/04/1936	70	1
2	Francisco Jr.	01/05/1980	25	1

Terceira forma normal – 3fn

Neste caso, o campo Cli_Idade deve ser eliminado, afinal, através da data de nascimento pode-se obter a idade do cliente.

Clientes			
Cli_ID	Cli_Nome	Cli_Nascimento	Cli_Cidade_ID
1	Marina Souza	11/04/1936	1
2	Francisco Jr.	01/05/1980	1

Exercícios práticos

1 - Considere a seguinte estrutura de dados para um sistema de biblioteca:

Tabela "Livros": {ISBN (Chave Primária), Título, Autor, IdEditora, AnoPublicação, NomeEditora}

Nesta estrutura, a editora (IdEditora e NomeEditora) é dependente funcional apenas do ISBN. Realize as normalizações possíveis dessa estrutura.

Exercícios práticos

2 - Você está projetando um sistema de gerenciamento de alunos para uma escola. Aqui estão os detalhes dos alunos:

Tabela "Alunos": {Matrícula (Chave Primária), NomeAluno, DataNascimento, Turma, NomeProfessor, Disciplina)

Nessa estrutura, a Turma, NomeProfessor e Disciplina são dependentes funcionais apenas da Matrícula do aluno. Realize as normalizações possíveis dessa estrutura.

Exercícios práticos

3 - Você está criando um banco de dados para um sistema de compras online. Aqui estão os detalhes dos pedidos:

Tabela "Pedidos": { NumPedido (Chave Primária), DataPedido, NomeCliente, EndereçoCliente, CidadeCliente, CEPCliente, CódigoProduto, NomeProduto, PrecoProduto}

Nessa estrutura, os detalhes do cliente (NomeCliente, EndereçoCliente, CidadeCliente e CEPCliente) e os detalhes do produto (CódigoProduto, NomeProduto e PreçoProduto) são dependentes funcionais apenas do NumPedido. Realize as normalizações possíveis dessa estrutura.

UVV

universidadevilavelha

uvvoficial

universidadevilavelha