MODELAGEM DE DADOS

MODELAGEM LÓGICA – NORMALIZAÇÃO

Olá!

Nesta aula, você irá:

1. Aprender o conceito de normalização.

2. Conhecer as principais formas normais: 1ª forma normal (1FN), 2ª forma normal (2FN) e 3ª forma normal (3FN).

1 Normalização

O processo de normalização de dados representa uma série de passos que se seguem no projeto de um banco de dados, que permitem um armazenamento consistente e o eficiente acesso aos dados de um banco de dados relacional. Esses passos reduzem a redundância de dados e, consequentemente, as chances de ocorrerem inconsistências.

2 Características de um mau projeto

- Repetição de informações

  Exemplo: A tabela EMPRESTAR em um banco de dados bancário:

  Emprestar(ag_nome, ag_ativos, empréstimo_número, cliente_nome, quantia)

  - Os atributos ag-nome, empréstimo_número e cliente_nome deveriam estar em tabelas distintas, já que descrevem entidades distintas.

- Inabilidadade de representar informações

  – Uma agência só pode existir se existir um empréstimo.

- Perda de Informação

  – Para eliminar um cliente, é necessário eliminar todas as informações de empréstimos.

Problemas causados pela falta de normalização

- **Anomalia de inclusão** Um novo cliente só poderá ser incluído se estiver relacionado a uma venda.
- **Anomalia de exclusão** Ao ser excluído um cliente, os dados referentes às suas compras serão perdidos.
- **Anomalia de alteração** Caso algum fabricante de produto altere o preço de um produto, será preciso percorrer toda a relação para se realizar múltiplas alterações.

Exemplos: Tabela PEDIDO (Num_pedido, prazo, cliente, endereco, cidade, uf, insc_est,cod_prod, unid, qtde, desc, val_unit, total_prod, total_pedido, cod_vendedor, nome_vendedor)

3 Dependência funcional

Determina uma restrição entre dois conjuntos de atributos de um BD Relacional.

Denotado pelo símbolo → , onde X → Y significa o atributo X determina funcionalmente o atributo Y ou ainda, o atributo Y é dependente funcional do atributo X.

Exemplos:

CPF → NOME_FUNCIONARIO

CPF determina funcionalmente NOME_FUNCIONARIO;

NOME_FUNCIONARIO é dependente funcional de CPF

Informalmente: Conhecido o valor de um CPF eu consigo determinar qual o NOME_FUNCIONARIO (único) com o qual este CPF está relacionado.

CPF, NUMERO_PROJETO → NUM_HORAS_TRABALHADAS

CPF E NUMERO_PROJETO determinam funcionalmente NUM_HORAS_TRABALHADAS;

NUM_HORAS_TRABALHADAS é dependente funcional de CPF E NUMERO_PROJETO

Informalmente : Conhecido o valor de um CPF E NUMERO_PROJETO eu consigo determinar qual o NUM_HORAS_TRABALHADAS com o qual este CPF está relacionado.

4 Formas normais

O conceito de normalização foi introduzido por E. F. Codd em 1972. Inicialmente, Codd criou as três primeiras formas de normalização, chamando-as de: primeira forma normal (1NF), segunda forma normal (2NF) e terceira forma normal (3NF).

Uma definição mais forte da 3NF foi proposta depois por Boyce-Codd, e é conhecida como forma normal de Boyce-Codd (FNBC).

Através do processo de normalização, pode-se, gradativamente, substituir um conjunto de entidades e relacionamentos por um outro, o qual se apresenta "purificado" em relação às anomalias de atualização (inclusão, alteração e exclusão), as quais podem causar certos problemas, tais como:

- Grupos repetitivos (atributos multivalorados) de dados
- Variação temporal de certos atributos, dependências funcionais totais ou parciais em relação a uma chave concatenada.
- Redundâncias de dados desnecessárias.
- Perdas acidentais de informação.
- Dificuldade na representação de fatos da realidade observada.
- Dependências transitivas entre atributos.

Normalização de relações é, portanto, uma técnica que permite depurar um projeto de banco de dados, através da identificação de inconsistências (informações em duplicidade, dependências funcionais mal resolvidas etc.).
À medida que um conjunto de relações passa para uma forma normal, vamos construindo um banco de dados mais confiável. O objetivo da normalização não é eliminar todos as inconsistências, e sim controlá-las.

5 Descrição das formas normais

5.1 Primeira forma normal

Uma relação está na primeira forma normal, se todos os seus atributos são monovalorados e atômicos.

Quando encontrarmos um atributo multivalorado, devemos criar um novo atributo que individualize a informação que esta multivalorada:

HISTÓRICO = {matricula-aluno, código-disciplina, notas}

No caso acima, cada nota seria individualizada identificando a prova a qual aquela nota se refere:

HISTÓRICO = {matrícula-aluno, código-disciplina, número-prova, nota}

Quando encontrarmos um atributo não atômico, devemos dividi-lo em outros atributos que sejam atômicos:

PESSOA = {CPF, nome-completo}

Vamos supor que, para a aplicação que utilizará esta relação, o atributo nome-completo não é atômico, a solução então será:

PESSOA = {CPF, nome, sobrenome}

5.2 Segunda forma normal

Uma relação está na segunda forma normal, quando duas condições são satisfeitas:

- A relação estiver na primeira forma normal.

- Todos os atributos primos dependerem funcionalmente de toda a chave primária.

Observe a relação abaixo:

HISTÓRICO = {matrícula-aluno, código-matéria, número-prova, nota, data-da-prova, nome-aluno, endereço-aluno, nome-matéria}

Fazendo a análise da dependência funcional de cada atributo primo, chegamos às seguintes dependências funcionais:

- matrícula-aluno, código-matéria, número-prova -> nota

- código-matéria, número-prova -> data-da-prova

- matrícula -aluno -> nome-aluno, endereço-aluno

- código-matéria -> nome-matéria

Concluímos então que apenas o atributo primo nota depende totalmente de toda chave primária. Para que toda a relação seja passada para a segunda forma normal, devem-se criar novas relações, agrupando os atributos de acordo com suas dependências funcionais:

BOLETIM = { matrícula -aluno, código-matéria, número-prova, nota}

PROVA = { código-matéria, número-prova, data-da-prova}

ALUNO = { matrícula -aluno, nome-aluno, endereço-aluno}

MATÉRIA = { código-matéria, nome-matéria}

O nome das novas relações deve ser escolhido de acordo com a chave.

5.3 Terceira forma normal

Uma relação está na terceira forma normal, quando duas condições forem satisfeitas:

- A relação estiver na segunda forma normal.
- Todos os atributos primos dependerem não transitivamente de toda a chave primária.

Observe a relação abaixo:

PEDIDO = {número-pedido, código-cliente, data-pedido, nome-cliente, código -cidade-cliente, nome-cidade-cliente}

Fazendo a análise da dependência funcional de cada atributo primo, chegamos às seguintes dependências funcionais:

- número-pedido -> código-cliente

- número-pedido -> data-pedido

- código -cliente -> nome-cliente

- código-cliente -> código-cidade-cliente

- código-cidade-cliente -> nome-cidade-cliente

Concluímos então que apenas os atributos primos código-cliente e data-pedido dependem não transitivamente totalmente de toda chave primária.

Observe que:

número-pedido -> código-cliente -> nome-cliente

número-pedido -> código-cliente -> código-cidade-cliente

número-pedido -> código-cliente -> código-cidade-cliente -> nome-cidade-cliente

Isso é **dependência transitiva**; devemos resolver inicialmente as dependências mais simples, criando uma nova relação onde código-cliente é a chave, o código-cliente continuará na relação PEDIDO como atributo primo, porém, os atributos que dependem dele devem ser transferidos para a nova relação:

PEDIDO = {número-pedido, código-cliente, data-pedido}

CLIENTE = {código-cliente, nome-cliente, código-cidade-cliente, nome-cidade-cliente}

As dependências transitivas da relação PEDIDO foram eliminadas, porém, ainda, devemos analisar a nova relação CLIENTE:

código-cliente -> código-cidade-cliente -> nome-cidade-cliente

Observe que o nome-cidade-cliente continua com uma dependência transitiva, vamos resolvê-la da mesma maneira :

PEDIDO = {número-pedido, código-cliente, data-pedido}

CLIENTE = { código-cliente, nome-cliente, código-cidade-cliente}

CIDADE = {código-cidade-cliente, nome-cidade-cliente}

O nome das novas relações deve ser escolhido de acordo com a chave.

6 Regra geral da normalização

“Todo item de dados em uma relação é dependente da chave, da chave toda e de nada mais do que a chave”

[James Martin, 1991]

“Um bom modelo de dados gera relações em 3FN”

CONCLUSÃO

Nesta aula, você:

- Aprendeu o conceito de normalização.
- Aprendeu o conceito de dependência funcional.
- Aprendeu a identificar as 1ª, 2ª e 3ª formas normais.

- Aprendeu a criar tabelas que cumprem as regras desta formas normais.