3075 - Fundamentos de banco de dados
2204 - Banco de dados

Curso de Sistemas de Informação

4º semestre – 2017.1

Prof. Daniella Vieira

Unidade III

CONCEITOS: PROJETO DE BANCO DE DADOS

- Normalização.

Referência Bibliográfica

Livro Elmarsi e Navathe.

HEUSER, Carlos Alberto. **Projeto de Banco de Dados**. 4ª Edição. Série Livros Didáticos, 1998.

Descrição dos conceitos básicos de normalização de banco de dados. Disponível em: <https://support.microsoft.com/en-us/kb/283878/pt-br>

Informações

Informações redundantes desperdiçam o espaço de armazenamento.

A mistura atributos de várias entidades pode gerar problemas conhecidos como anomalias de atualização:

- Anomalias de inserção
- Anomalias de remoção
- Anomalias de modificação

Informações

Considere a relação:

EMPREGADO_PROJ (num_inss, proj_cod, horas, emp_nome, proj_nome, proj_localizacao)

- **Anomalia de Atualização**:

A mudança de nome do projeto de número 20 de ‘Faturamento’ para ‘Conta-Cliente’ provoca alterações no significado da informação de todos os empregados que trabalham nesse projeto.

Informações

- **Anomalia de Inserção**:

Não se pode inserir um projeto a menos que um empregado esteja associado.

Por outro lado, não se pode inserir um empregado a menos que ele esteja associado a um projeto

- **Anomalia de Remoção**:

Quando um projeto é removido, todos os empregados que trabalham no projeto serão removidos.

Se um empregado for o único empregado do projeto, a remoção desse empregado resultará na remoção do projeto correspondente.

Projeto de banco de dados

Regras para um bom projeto:

1. Cada tupla de uma relação deve representar uma entidade ou instância de relacionamento.

2. Projete um esquema que não sofra de anomalias de inserção, remoção e de modificação. Se existir alguma, certifique-se de documenta-la para que as aplicações levem tais anomalias em consideração.

Projeto de banco de dados

3. Relações devem ser projetadas de forma que suas tuplas tenham a menor quantidade possível de valores nulos.

Normalmente os atributos que possuem valores nulos podem ser colocados em relações separadas (com uma chave-primária).

Razões para os valores nulos:

- Valor não aplicável ou inválido
- Valor desconhecido (embora possa existir)
- Valor indisponível (embora se saiba que exista)

Tuplas Espúrias

Projetos incorretos de BDRs podem gerar resultados inválidos em certas operações JOIN.

A propriedade de “junção sem perdas” é usada para garantir resultados corretos em operações JOIN.

4. As relações devem ser projetadas para satisfazer a condição de junção sem perdas. Nenhuma tupla espúria deve ser gerada ao fazer um JOIN NATURAL de qualquer relação.

Dependências Funcionais

Dependências funcionais (DFs) são usadas para medir formalmente a qualidade do projeto relacional. As DFs e chaves são usadas para definir formas normais de relações.

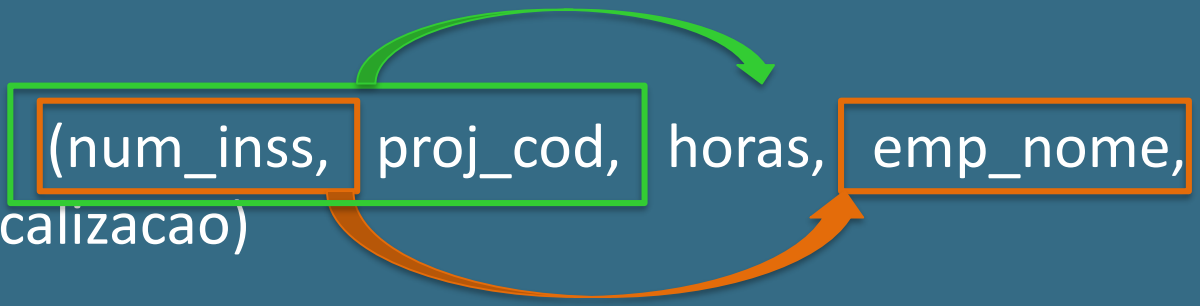
As DFs são restrições que são derivadas do significado e do inter-relacionamento dos dados de atributos.

Um conjunto de atributos X determina funcionalmente um conjunto de atributos Y se o valor de X determinar um único valor Y.

Dependências Funcionais

Exemplos de Restrições de DF

EMPREGADO_PROJ (num_inss, proj_cod, horas, emp_nome, proj_nome, proj_localizacao)



- O número do seguro social determina o nome do empregado.
- O número do projeto determina o nome do projeto e a sua localização: proj_cod -> {proj_nome, proj_localizacao}
- O num_inss de empregado e o número do projeto determinam as horas semanais que o empregado trabalha no projeto: {num_inss, proj_cod } -> horas

Dependências Funcionais

Regras de inferência de Armstrong (conjunto completo de regras de inferência):

RI1. (Reflexiva) Se Y é subconjunto de X, então X -> Y (Isso também é válido quando X=Y)

RI2. (Aumentativa) Se X -> Y, então X U Z -> Y U Z

RI3. (Transitiva) Se X -> Y e Y -> Z, então X -> Z

Dependências Funcionais

Regras de Inferência para DFs

Algumas regras de inferência úteis:

(Decomposição)	Se X -> YZ,	então X -> Y e X -> Z
(Aditiva)	Se X -> Y e X -> Z,	então X -> YZ
(Pseudotransitiva)	Se X -> Y e WY -> Z,	então WX -> Z

As três regras de inferência acima, bem como quaisquer outras regras de inferência, podem ser deduzidas a partir de RI1, RI2 e RI3 (propriedade de ser completa).

Normalização

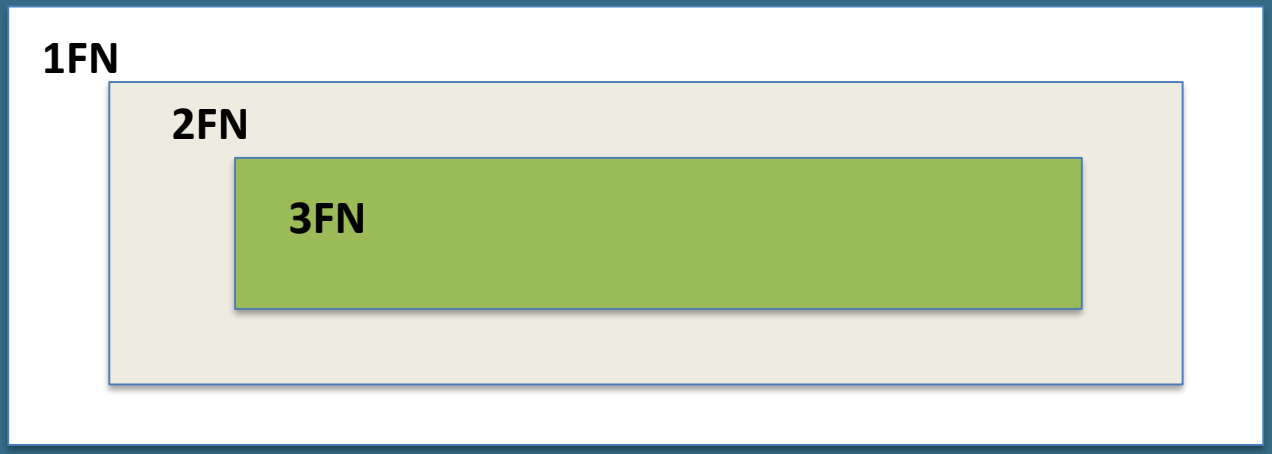
O processo de normalização baseia-se no conceito de forma normal.

Uma forma normal é uma regra que deve ser obedecida por uma tabela para que esta seja considerada “bem projetada”. Há diversas formas normais, isto é, diversas regras, cada vez mais rígidas, para verificar tabelas relacionais.

Normalização

Aplicando as regras de normalização, é possível garantir um banco de dados mais íntegro, sem redundâncias e inconsistências.

Existem 1FN, 2FN, 3FN, BCNF (Boyce/Cood), 4NF e 5NF. Contudo, são as 3 primeiras formas normais as mais conhecidas.



Normalização

As 2FN, 3FN e BCNF baseiam-se em chaves e DFs de uma relação esquema.

As 4FN e 5FN baseiam-se em chaves e dependências multivaloradas (não são muitos utilizadas).

Exemplos

Tabela não normalizada

Curso (IdCurso, Professor, CargaHr, Sala, Capacidade, Disciplina1, Disciplina2, Disciplina3)

1FN

A tabela não deve conter grupos repetidos e nem atributos com mais de um valor.

Curso (IdCurso, Professor, CargaHr, Sala, Capacidade, Disciplina)

2FN

Todos os atributos não chaves da tabela devem depender da chave primária

Curso (IdCurso, CargaHr, Sala, Capacidade)

Curso_Turma (Id, *IdCurso*, Disciplina)

Curso_Professor (Id, *IdCurso*, Professor)

3FN

Os atributos não chave de uma tabela devem ser mutuamente independentes e dependentes exclusivamente da chave primária.

Curso (IdCurso, CargaHr, *idSala*)

Curso_Sala (IdSala, Sala, Capacidade)

Curso_Turma (Id, *IdCurso*, Disciplina)

Curso_Professor (Id, *IdCurso*, Professor)

Exemplos

Tabela não normalizada

Cliente (idCliente, nome, senha, endereco, tipoEndereco, telefone*)

Conhecendo as dependências funcionais:

idCliente -> nome,senha,endereco,tipoEndereco,telefone*

tipoEndereco -> endereco

1FN

Cliente (idCliente, nome, senha, endereco, tipoEndereco)

Fone (idFone, telefone, *idCliente*)

2FN

Todos os atributos não chaves da tabela devem depender da chave primária

Cliente (idCliente, nome, senha)

Fone (idFone, telefone, *idCliente*)

Endereco (idEndereco, endereco, tipoEndereco, *idCliente*)

3FN

Cliente (idCliente, nome, senha)

Tipo(idTipo, tipoEndereco)

Fone (idFone, telefone, *idCliente*)

Endereco (idEndereco, endereco, *idTipo*, *idCliente*)

BCNF (Boyce-Codd Normal Form)

A BCFN é um aperfeiçoamento da 3FN destinada a lidar com situações em que se verifique a existência de (1) mais do que uma chave candidata, e, (2) que duas chaves candidatas possuam elementos comuns.

Uma relação está na BCFN quando todos os atributos são dependentes da chave, de toda a chave e de nada mais do que a chave.

BCNF (Boyce-Codd Normal Form)

Uma relação esquema R está na BCNF se, sempre que houver uma DF X -> A em R, então X é uma superchave de R.

Cada FN engloba a FN anterior. Toda relação em 2FN está na 1FN. Toda relação em 3FN está na 2FN. Toda relação em BCNF está na 3FN.

Existem relações que estão na 3FN, mas não em BCNF. A meta é alcançar a BCNF (ou 3FN) em todas as relações.

BCNF (Boyce-Codd Normal Form)

Exemplo:

(paciente, servico) -> medico

R (paciente, servico, medico)

Um doente em um serviço hospitalar é observado pelo mesmo médico. Um médico só pertence a um serviço.

R1 (paciente, medico)

R2 (medico, servico)

Esta solução permite ter um paciente com dois médicos do mesmo serviço. Assim, o tratamento deveria ser feito pela aplicação (há redundância)

BCNF (Boyce-Codd Normal Form)

Exemplo:

R (codAluno, codDisciplina, Professor)

Na 3FN, onde os alunos frequentam várias disciplinas.

Cada professor só leciona uma disciplina, mas uma disciplina pode ser lecionada por vários professores. Portanto, pode-se verificar a seguinte dependência funcional (codAluno, codDisciplina) -> Professor

BCNF (Boyce-Codd Normal Form)

Exemplo:

R (codAluno, codDisciplina, Professor)

No entanto, existem duas chaves candidatas compostas (codAluno, codDisciplina) e (codAluno, Professor).

O atributo Professor não é chave candidata, mas determina o codDisciplina (um professor só leciona uma disciplina).

(Professor -> codDisciplina)

4FN e 5FN

Em geral uma relação na BCNF já está na 4FN e 5FN. Estas surgem para resolver problemas muito raros:

- Uma relação R encontra-se na **4FN**, se está na BCFN e não existem dependências multivalor.

- Uma relação R está na **5FN** se não puder ser mais decomposta sem perda de informação.

Quarta Forma Normal (4 FN)

Uma relação R está na 4FN se:

todo valor em R for atômico. Ou seja, R não contém nenhum grupo de repetição/dependência multivalorada.

Considerações:

- geralmente considerada parte da definição formal de uma relação;
- não permite atributos multivalorados, compostos ou suas combinações;
- provoca vários grupos de repetição.

Quarta Forma Normal (4 FN)

Método para corrigir o problema:

- para cada grupo de repetição separado, gera-se uma nova relação correspondente contendo este grupo de repetição e a chave primária da relação original.
- determinar a chave primária da nova relação, a qual será a concatenação da chave primária da relação original com a chave para o grupo de repetição.

Quarta Forma Normal (4 FN)

Exemplo:

vendedor (num_vend, nome_vend, {cliente(num_cliente, nome_cliente)}, {loja(nome, local)})

Dependências funcionais

num_vend -> nome_vend

Dependências multivaloradas

num_vend -> {cliente(num_cliente, nome_cliente)}

num_vend -> {loja(nome, local)}

Quarta Forma Normal (4 FN)

Exemplo:

vendedor (num_vend, nome_vend, {cliente(num_cliente, nome_cliente)}, {loja(nome, local)})

Solução:

vendedor (num_vend, nome_vend)

Vendedor_cliente (num_vend, num_cliente, nome_cliente)

vendedor_loja (num_vend, nome, local)

Quinta Forma Normal (5 FN)

Está ligada à noção de dependência de junção.

Se uma relação é decomposta em várias relações e a reconstrução não é possível pela junção das outras relações, dizemos que existe uma dependência de junção.

Existem tabelas na 4FN que não podem ser divididas em duas relações sem que se altere os dados originais.

Quinta Forma Normal (5 FN)

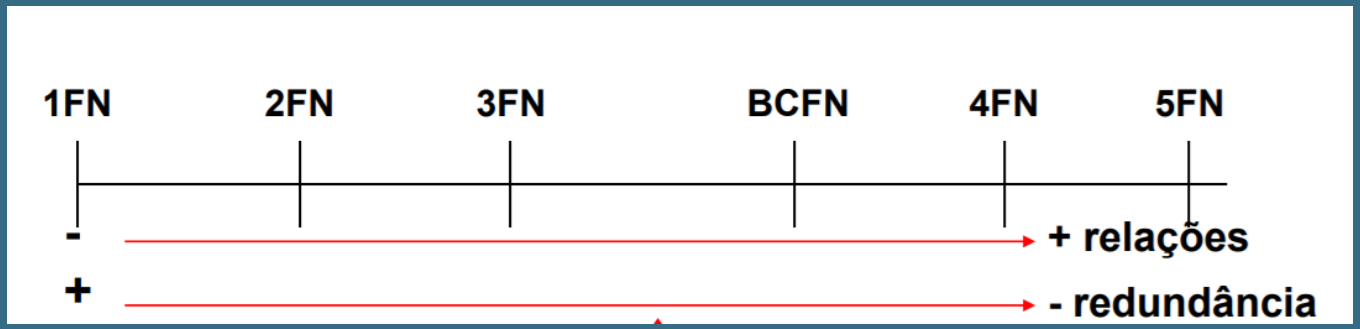
Exemplo:

Sejam as relações R1(CodEmp, CodPrj) e R2(CodEmp, Papel) a decomposição da relação.

ProjetoRecurso(CodEmp, CodPrj, Papel).

Desnormalização

A normalização deve ser usada com bom senso. Pode haver a necessidade de fazer a desnormalização.



Desnormalização

O SGBD deve considerar alguns aspectos que permitem melhorias na utilização do sistema.

Trata-se de estabelecer um compromisso entre a flexibilidade do sistema e a viabilidade da sua utilização.

Pretende-se obter um esquema equilibrado, que não ponha em risco a integridade da BD, mas tenha um desempenho razoável. Por essa razão, na maioria dos casos, o processo de normalização para na 3FN.