Senac - Desenvolver Banco de Dados - UC10 - 2024_A4-1/1

Conceitos de Normalização e Eliminação de Redundâncias

1. Introdução à Normalização

A **normalização** é uma técnica aplicada na modelagem de bancos de dados para eliminar **redundâncias** e garantir a **consistência** dos dados. O processo divide os dados em múltiplas tabelas organizadas e conectadas por relacionamentos, evitando duplicidade e inconsistências. A normalização é dividida em diferentes formas normais, que são etapas progressivas para melhorar a organização dos dados.

Objetivos da Normalização:

- Eliminar redundâncias e duplicidade.
- Garantir a integridade e consistência dos dados.
- Otimizar a estrutura do banco de dados para manutenção eficiente.
- Reduzir o risco de anomalias durante inserções, atualizações e exclusões.

---

2. As Três Primeiras Formas Normais

2.1 Primeira Forma Normal (1FN)

Uma tabela está na **Primeira Forma Normal (1FN)** se todos os seus atributos contêm **valores atômicos** (não divisíveis) e não existem grupos repetitivos.

- **Regras da 1FN**:
  - Cada célula deve armazenar um único valor.
  - Cada registro deve ser único (sem registros idênticos).
  - Não pode haver colunas com múltiplos valores (listas).

Exemplo - Tabela Não Normalizada

Pedido	Produtos	Quantidade
1	Caneta, Lápis	2, 3
2	Caderno	1

- **Problema**: A coluna "Produtos" contém mais de um valor por célula.

Tabela Normalizada na 1FN

Pedido	Produto	Quantidade
1	Caneta	2
1	Lápis	3
2	Caderno	1

- **Benefício**: Agora cada célula armazena um único valor.

2.2 Segunda Forma Normal (2FN)

Uma tabela está na **Segunda Forma Normal (2FN)** se, além de atender aos requisitos da 1FN, todos os atributos não-chave forem **completamente dependentes** da chave primária.

- **Problemas da 2FN**:
  - Quando um campo depende apenas de parte da chave primária em tabelas com chaves compostas.

Exemplo - Problema de Dependência Parcial

Pedido	Produto	Preço Produto	Quantidade
1	Caneta	1.50	2
1	Lápis	0.50	3

- **Problema**: O preço do produto não depende do pedido, mas apenas do produto.

Tabela Normalizada na 2FN

Tabela de Pedidos

Pedido	Produto	Quantidade
1	Caneta	2
1	Lápis	3

Tabela de Produtos

Produto	Preço Produto
Caneta	1.50
Lápis	0.50

- **Benefício**: A tabela de produtos armazena o preço uma vez, eliminando redundâncias.

2.3 Terceira Forma Normal (3FN)

Uma tabela está na **Terceira Forma Normal (3FN)** se, além de atender aos requisitos da 2FN, não houver **dependências transitivas**, ou seja, um campo não deve depender de outro campo que não seja chave primária.

Exemplo - Problema de Dependência Transitiva

Pedido	Cliente	Endereço Cliente
1	Maria	Rua A, 123
2	João	Rua B, 456

- **Problema**: O endereço do cliente depende do cliente, e não do pedido.

Tabela Normalizada na 3FN

Tabela de Pedidos

Pedido	Cliente
1	Maria
2	João

Tabela de Clientes

Cliente	Endereço Cliente
Maria	Rua A, 123
João	Rua B, 456

- **Benefício: A tabela de clientes armazena os endereços, evitando redundância.**

---

3. Técnicas para Eliminar Redundâncias e Garantir Consistência de Dados

Eliminar redundâncias e garantir a consistência é essencial para manter a integridade e a eficiência do banco de dados. A seguir, exploramos técnicas importantes para atingir esse objetivo.

3.1 Divisão Lógica de Tabelas

A divisão lógica ou decomposição consiste em quebrar uma tabela maior e complexa em várias tabelas menores, cada uma contendo informações relacionadas de forma lógica. Isso é conhecido como normalização e ajuda a evitar dados duplicados e inconsistentes.

- **Exemplo: Em um sistema de vendas, é preferível separar as informações de Clientes e Pedidos em tabelas distintas, em vez de armazená-las em uma única tabela.**

Benefícios:

- **Reduz a duplicação de dados.**
- **Facilita a manutenção e atualização de registros.**
- **Melhora a organização e a legibilidade dos dados.**

3.2 Relacionamentos entre Tabelas

Com tabelas separadas, é necessário conectar as informações de forma consistente. Isso é feito utilizando chaves estrangeiras (Foreign Keys) para estabelecer relacionamentos entre tabelas.

- **Exemplo: Uma tabela de Matrículas usa o campo `id_aluno` como chave estrangeira, referenciando a chave primária da tabela Alunos. Isso garante que cada matrícula esteja vinculada a um aluno válido.**

Tipos de Relacionamentos:

- **1:1 (Um para Um): Cada registro de uma tabela corresponde a um único registro de outra tabela.**
- **1**
  (Um para Muitos): Um registro em uma tabela pode estar relacionado com vários registros em outra.
- **N**
  (Muitos para Muitos): Vários registros de uma tabela podem estar relacionados com vários registros de outra.

Benefícios:

- **Garante a integridade dos dados.**
- **Facilita a recuperação e atualização dos dados relacionados.**

3.3 Validação de Dados

A validação de dados é essencial para garantir que as informações armazenadas sejam precisas e estejam de acordo com as regras do sistema. A integridade referencial assegura que os dados entre as tabelas permaneçam consistentes.

Técnicas de Validação:

- **Restrição NOT NULL: Impede que campos importantes fiquem sem valor.**
- **Chaves Estrangeiras: Garantem que registros relacionados existam na tabela de origem.**
- **Valores Padrão (DEFAULT): Atribuem um valor automático quando nenhum dado é fornecido.**
- **Validação de Formato: Por exemplo, CPF deve conter exatamente 11 dígitos.**

Benefícios:

- **Evita registros inválidos.**
- **Garante a consistência e integridade entre as tabelas.**

3.4 Controle de Atualizações e Exclusões

Ao lidar com registros inter-relacionados, é essencial que alterações em uma tabela sejam propagadas para as tabelas relacionadas, evitando inconsistências.

Técnica de Exclusão em Cascata

A exclusão em cascata remove automaticamente todos os registros relacionados quando um registro principal é excluído.

- **Exemplo: Ao excluir um cliente, todas as compras feitas por ele são excluídas automaticamente.**

Restrição de Exclusão

Em alguns casos, é desejável bloquear a exclusão de um registro se ele possuir referências ativas em outras tabelas.

- **Exemplo: Um pedido não pode ser excluído enquanto houver itens pendentes.**

Atualização em Cascata

Essa técnica propaga alterações de um campo chave primária para todas as referências nas tabelas dependentes.

- **Exemplo: Se o código de um produto mudar, ele é automaticamente atualizado em todas as vendas relacionadas.**

Benefícios:

- **Garante que todas as alterações sejam aplicadas de forma consistente.**
- **Evita que registros órfãos ou inconsistentes sejam deixados no sistema.**

Resumo das Técnicas

Essas técnicas visam a manutenção de um banco de dados organizado, eficiente e confiável:

1. **Divisão Lógica de Tabelas evita redundância.**
2. **Relacionamentos entre Tabelas garantem integridade e acessibilidade.**
3. **Validação de Dados assegura a consistência.**
4. **Controle de Atualizações e Exclusões mantém a coerência entre os registros.**

Essas práticas tornam o banco de dados mais escalável e fácil de manter, garantindo que o sistema funcione de forma eficiente à medida que cresce e evolui.

---

4. Análise de Casos Práticos de Normalização

4.1 Caso Prático 1: Loja Virtual

- Problema: A loja armazena pedidos e clientes em uma única tabela, causando duplicação de dados.
- Solução: Dividir a tabela em duas: Clientes e Pedidos, conectando-as por meio de chaves estrangeiras.

4.2 Caso Prático 2: Sistema Escolar

- Problema: O sistema armazena informações de alunos e cursos juntos, com múltiplos registros duplicados.
- Solução: Criar tabelas separadas para Alunos, Cursos e Matrículas, organizando os relacionamentos.

4.3 Caso Prático 3: Cadastro de Fornecedores

- Problema: O sistema armazena o nome e endereço do fornecedor repetidamente em vários pedidos.
- Solução: Separar os dados em Fornecedores e Pedidos, referenciando o fornecedor pela chave estrangeira.

---

5. Conclusão

A normalização é uma técnica essencial para projetar bancos de dados eficientes e consistentes. Ao aplicar as três primeiras formas normais, evitamos redundância e garantimos que as informações estejam organizadas e bem relacionadas. Através da análise de casos práticos, podemos entender como esses conceitos se aplicam na prática e como eles contribuem para a escalabilidade e integridade do sistema.