

Exercícios: Conversão de Modelo Lógico para Físico

Instruções

Estes exercícios visam consolidar o entendimento sobre a transição do Modelo Lógico para o Modelo Físico em um banco de dados relacional.

Exercício 1: Identificação de Níveis de Abstração

Pergunta:

A modelagem de dados é dividida em três níveis de abstração: Conceitual, Lógico e Físico. Para cada característica abaixo, identifique a qual nível ela pertence:

- Independência de SGBD:** O modelo pode ser implementado em qualquer sistema relacional (MySQL, Oracle, PostgreSQL, etc.).
- Foco em Entidades e Relacionamentos:** Visão de alto nível do negócio, sem detalhamento de tabelas.
- Uso de SQL DDL (CREATE TABLE):** Implementação real com tipos de dados específicos.
- Definição de Chaves Primárias (PK) e Estrangeiras (FK):** Estrutura de tabelas e seus relacionamentos.

Respostas Esperadas:

- Modelo Lógico
 - Modelo Conceitual
 - Modelo Físico
 - Modelo Lógico
-

Exercício 2: Mapeamento de Tipos de Dados

Pergunta:

A conversão de atributos do Modelo Lógico para colunas no Modelo Físico exige a escolha de tipos de dados específicos do SGBD. Assuma que o SGBD escolhido é o **MySQL**. Complete a tabela mapeando os tipos de dados genéricos do Modelo Lógico para os tipos de dados físicos mais adequados, justificando brevemente a escolha:

| Atributo (Modelo Lógico) | Tipo Genérico (Lógico) | Tipo Específico (Físico - MySQL) | Justificativa |
|--------------------------|------------------------|---|--|
| Nome Completo | String | <code>VARCHAR(255)</code> | Permite armazenar texto de tamanho variável, com limite para otimização. |
| Data de Nascimento | Date | <code>DATE</code> | Tipo específico para armazenar apenas a data (ano, mês, dia). |
| Salário | Number | <code>DECIMAL(10, 2)</code> | Tipo ideal para valores monetários, garantindo precisão de duas casas decimais. |
| Status Ativo | Boolean | <code>TINYINT(1)</code> ou <code>BOOLEAN</code> | Representação booleana (0 ou 1) para otimizar espaço. |
| Descrição Longa | String | <code>TEXT</code> | Para textos longos, onde o limite de <code>VARCHAR</code> pode ser insuficiente. |

Exercício 3: Conversão de Relacionamento N:N

Pergunta:

No Modelo Lógico, existe um relacionamento **N:N (Muitos para Muitos)** entre as entidades `ALUNO` e `DISCIPLINA`.

Descreva o que é necessário fazer para converter este relacionamento para o **Modelo Físico** em um banco de dados relacional.

Resposta Esperada:

Para converter um relacionamento N:N para o Modelo Físico, é necessário criar uma **Tabela Associativa** (ou Tabela de Ligação). Esta nova tabela, por exemplo, `ALUNO_DISCIPLINA`, terá:

1. Uma Chave Estrangeira (FK) que referencia a Chave Primária da tabela `ALUNO`.
2. Uma Chave Estrangeira (FK) que referencia a Chave Primária da tabela `DISCIPLINA`.
3. A Chave Primária (PK) da nova tabela será a **combinação** dessas duas Chaves Estrangeiras (Chave Composta), garantindo a unicidade da matrícula de um aluno em uma disciplina.

Exercício 4: Criação de Código DDL (Modelo Físico)

Pergunta:

Considere a seguinte tabela do Modelo Lógico:

| Tabela: PRODUTO |
|---|
| ID_Produto (PK) |
| Nome (String, Obrigatório) |
| Preço (Number, Duas Casas Decimais) |
| FK_Categoria (Chave Estrangeira para Tabela CATEGORIA) |

Escreva o código **SQL DDL (Data Definition Language)** para criar esta tabela no **MySQL**, incluindo as restrições de chave primária, nulidade e chave estrangeira.

Resposta Esperada:

```
CREATE TABLE PRODUTO (  
    ID_Produto INT PRIMARY KEY AUTO_INCREMENT,  
    Nome VARCHAR(100) NOT NULL,  
    Preço DECIMAL(10, 2) NOT NULL,  
    FK_Categoria INT NOT NULL,  
    CONSTRAINT fk_categoria  
        FOREIGN KEY (FK_Categoria)  
        REFERENCES CATEGORIA (ID_Categoria)  
);
```

Exercício 5: Otimização e Dependência de SGBD

Pergunta:

O Modelo Físico permite a inclusão de elementos de otimização que não existem no Modelo Lógico. Cite dois exemplos de otimizações que são definidas no Modelo Físico e explique por que a decisão de implementá-las é **dependente do SGBD** e do contexto de uso.

Resposta Esperada:

1. Criação de Índices Adicionais:

- **Otimização:** Índices são criados em colunas frequentemente usadas em cláusulas `WHERE` ou `JOIN` para acelerar a busca de dados.
- **Dependência:** A sintaxe para criar índices (`CREATE INDEX`), o tipo de índice (e.g., B-tree, Hash) e a forma como o SGBD gerencia e utiliza esses índices (custo de armazenamento vs. ganho de performance) são específicos de cada SGBD.

2. Particionamento de Tabela:

- **Otimização:** Dividir uma tabela muito grande em partes menores e mais gerenciáveis (partições) para melhorar a performance de consultas e manutenção.
- **Dependência:** A funcionalidade de particionamento e os métodos de particionamento disponíveis (e.g., por faixa, por lista, por hash) são

recursos avançados que variam significativamente entre SGBDs (alguns podem não suportar ou ter implementações muito diferentes).

A decisão de implementar essas otimizações é dependente do SGBD porque as ferramentas, sintaxes e o impacto no desempenho são específicos de cada sistema. Além disso, o contexto de uso (volume de dados, frequência de consultas) é o que justifica a necessidade dessas otimizações.

Exercício 6: Análise de Sintaxe SQL DDL

Pergunta:

A sintaxe SQL DDL (Data Definition Language) é crucial para a implementação do Modelo Físico. Analise o trecho de código DDL abaixo (assumindo MySQL) e identifique:

1. A restrição de chave primária.
2. A restrição de nulidade.
3. A restrição de chave estrangeira e a tabela referenciada.
4. O tipo de dado específico usado para armazenar o nome do cliente.

```
CREATE TABLE CLIENTE (  
    ID_Cliente INT PRIMARY KEY AUTO_INCREMENT,  
    Nome VARCHAR(150) NOT NULL,  
    Email VARCHAR(100) UNIQUE,  
    FK_Endereco INT,  
    CONSTRAINT fk_endereco  
        FOREIGN KEY (FK_Endereco)  
        REFERENCES ENDERECO (ID_Endereco)  
);
```

Respostas Esperadas:

1. **Restrição de Chave Primária:** PRIMARY KEY na coluna ID_Cliente .
2. **Restrição de Nulidade:** NOT NULL na coluna Nome .

3. **Restrição de Chave Estrangeira e Tabela Referenciada:** FOREIGN KEY

(FK_Endereco) REFERENCES ENDereco (ID_Endereco) . A tabela referenciada é ENDereco .

4. **Tipo de Dado para Nome:** VARCHAR(150) .