Matéria: Tecnologia da Informação

Assunto: Algoritmos de Ordenação e Normalização de Banco de Dados

---

Resumo Teórico do Assunto

Para dominar as questões sobre **Algoritmos de Ordenação** e **Normalização de Banco de Dados**, é fundamental compreender os princípios e as características de cada um.

---

1. Algoritmos de Ordenação

Algoritmos de ordenação são procedimentos que organizam elementos de uma lista ou array em uma ordem específica (crescente ou decrescente). Os algoritmos mencionados na questão (Bolha, Seleção e Inserção) são exemplos de algoritmos de ordenação por comparação, que geralmente utilizam **laços aninhados** (dois comandos `for` ou `while`) para realizar as comparações e trocas.

Para identificar qual algoritmo está em execução, é crucial entender o estado do array ao final de cada **iteração do laço mais externo**.

 # a) Bubble Sort (Ordenação da Bolha)

• **Conceito:** Compara elementos adjacentes e os troca de posição se estiverem na ordem errada. Repete o processo até que nenhuma troca seja necessária.
• **Funcionamento por Iteração Externa:** Em cada iteração do laço externo, o maior (ou menor, dependendo da ordem) elemento restante "flutua" para sua posição correta no final da parte não ordenada do array.
• **Estado do Array:** Ao final de cada iteração externa `i`, os `i` últimos elementos do array estarão em suas posições finais e ordenadas.
• **Complexidade de Tempo (Média/Pior Caso):** $O(n^2)$

 # b) Selection Sort (Ordenação por Seleção)

• **Conceito:** Encontra o menor (ou maior) elemento na parte não ordenada do array e o troca com o elemento na primeira posição da parte não ordenada.
• **Funcionamento por Iteração Externa:** Em cada iteração do laço externo, o algoritmo encontra o menor elemento na sublista não ordenada e o coloca na posição correta no início da sublista.
• **Estado do Array:** Ao final de cada iteração externa `i`, os `i` primeiros elementos do array estarão em suas posições finais e ordenadas.
• **Complexidade de Tempo (Média/Pior Caso):** $O(n^2)$

 # c) Insertion Sort (Ordenação por Inserção)

• **Conceito:** Constrói a lista ordenada um elemento por vez. Cada novo elemento é "inserido" na posição correta dentro da sublista já ordenada.
• **Funcionamento por Iteração Externa:** Em cada iteração do laço externo, o algoritmo pega o próximo elemento da parte não ordenada e o insere na posição correta dentro da parte já ordenada, deslocando os elementos maiores (ou menores) para a direita (ou esquerda).
• **Estado do Array:** Ao final de cada iteração externa `i`, os `i` primeiros elementos do array formam uma sublista ordenada, mas não necessariamente em suas posições finais globais (exceto o primeiro elemento).
• **Complexidade de Tempo (Média/Pior Caso):** $O(n^2)$

---

2. Normalização de Banco de Dados

A normalização é um processo de organização de colunas e tabelas em um banco de dados relacional para minimizar a **redundância de dados** e melhorar a **integridade dos dados**. Ela é baseada no conceito de **dependências funcionais**.

 # a) Dependência Funcional (DF)

• **Definição:** Uma **dependência funcional** `A $\rightarrow$ B` (lê-se "A determina B" ou "B depende funcionalmente de A") significa que, para quaisquer duas tuplas (linhas) em uma relação, se os valores do atributo A são iguais, então os valores do atributo B também devem ser iguais. Em outras palavras, o valor de A identifica unicamente o valor de B.
• **Exemplo:** `CPF $\rightarrow$ Nome` (Um CPF determina unicamente um Nome).

 # b) Chaves

• **Chave Candidata:** Um conjunto mínimo de atributos que identifica unicamente cada tupla em uma relação. "Mínimo" significa que nenhum subconjunto desses atributos pode ser uma chave candidata por si só.
• **Chave Primária:** Uma das chaves candidatas escolhida para ser o identificador principal da relação.
• **Atributo Primo (ou Chave):** Um atributo que faz parte de *qualquer* chave candidata.
• **Atributo Não Primo (ou Não Chave):** Um atributo que não faz parte de *nenhuma* chave candidata.

 # c) Formas Normais (FNs)

As formas normais são um conjunto de regras que as relações devem seguir para atingir diferentes níveis de normalização. A questão menciona 1FN e 3FN.

• **1FN (Primeira Forma Normal)**
* **Requisito:** Uma relação está em 1FN se todos os seus atributos contêm apenas **valores atômicos** (indivisíveis) e não há **grupos repetitivos** de atributos.
* **Importância:** É a base para todas as outras formas normais. A questão já assume que todas as relações estão em 1FN.

• **2FN (Segunda Forma Normal)**
* **Requisito:** Uma relação está em 2FN se estiver em 1FN **E** todos os seus atributos não primos forem **totalmente dependentes funcionalmente** de *qualquer* chave candidata.
* **O que evita: Dependências parciais**. Uma dependência parcial ocorre quando um atributo não primo depende funcionalmente de *apenas uma parte* de uma chave candidata composta.
* **Exemplo de Problema:** Se `(ID_Pedido, ID_Produto) → Nome_Produto`, mas `ID_Produto → Nome_Produto`, então `Nome_Produto` tem uma dependência parcial na chave composta `(ID_Pedido, ID_Produto)`.

• **3FN (Terceira Forma Normal)**
* **Requisito:** Uma relação está em 3FN se estiver em 2FN **E** não houver **dependências transitivas** de atributos não primos em relação a *qualquer* chave candidata.
* **O que evita: Dependências transitivas**. Uma dependência transitiva ocorre quando um atributo não primo `C` depende funcionalmente de outro atributo não primo `B`, que por sua vez depende funcionalmente de uma chave candidata `A` (ou seja, `A → B` e `B → C`, onde B e C são não primos).
* **Regra Simplificada:** "Todo atributo não chave deve depender da chave, da chave completa e de nada mais além da chave." (A parte "nada mais além da chave" se refere à eliminação de dependências transitivas).
* **Exemplo de Problema:** Se `ID_Funcionario → ID_Departamento` e `ID_Departamento → Nome_Departamento`, então `Nome_Departamento` tem uma dependência transitiva em relação a `ID_Funcionario` (assumindo `ID_Funcionario` é a chave e `ID_Departamento` e `Nome_Departamento` são não primos).

Para resolver questões de normalização, você deve:
1. Identificar todas as **chaves candidatas**.
2. Identificar todas as **dependências funcionais**.
3. Verificar se há **dependências parciais** (para 2FN).
4. Verificar se há **dependências transitivas** envolvendo atributos não primos (para 3FN).

Compreender esses conceitos e a forma como cada algoritmo de ordenação e forma normal se comporta é a chave para analisar e resolver as questões propostas.

Questões de Provas Anteriores

Fonte: escriturario_agente_de_tecnologia (1).pdf, Página: 21

pcimarkpci MjgwNDowMTRkOjE0YTU6OTI1ODozOGQ2OjNhMGM6NTM0MzplZmI1:U3VuLCAyNyBKdWwgMjAyNSAyMzo0Nzo0MCAtMDMwMA==
www.pciconcursos.com.br
AGENTE DE TECNOLOGIA - Microrregião 16 DF-TI21
BANCO DO BRASIL
GABARITO 1
55
Um professor preparou uma série de experimentos para avaliar, juntamente com seus alunos, três algoritmos de ordenação: o da bolha, o de ordenação por inserção e o de ordenação por seleção. Para tal, ele escreveu três métodos Java, um para cada algoritmo. Todos eles recebem como único parâmetro um array de inteiros (int vet[ ] = {81,15,4,20,7,47,14,20,4}), que será ordenado em ordem crescente.
Para acompanhar a evolução desse array sendo ordenado, cada um dos três métodos exibe a configuração dos elementos do array ao término de cada iteração do comando de repetição mais externo. Vale lembrar que esses três algoritmos de ordenação são compostos por dois comandos de repetição aninhados (dois comandos for ou dois comandos while).
Terminada a codificação, o professor executou os métodos relativos aos três algoritmos de ordenação e projetou no quadro as configurações do array relativas às três primeiras iterações de cada um dos algoritmos de ordenação, conforme mostrado a seguir.
Algoritmo 1
4 15 81 20 7 47 14 20 4
4 4 81 20 7 47 14 20 15
4 4 7 20 81 47 14 20 15

Algoritmo 2
15 81 4 20 7 47 14 20 4
4 15 81 20 7 47 14 20 4
4 15 20 81 7 47 14 20 4
Algoritmo 3
15 4 20 7 47 14 20 4 81
4 15 7 20 14 20 4 47 81
4 7 15 14 20 4 20 47 81
As configurações 1, 2 e 3, exibidas acima, correspondem, respectivamente, aos algoritmos

(A) da bolha, de seleção e de inserção
(B) da bolha, de inserção e de seleção
(C) de seleção, de inserção e da bolha
(D) de seleção, da bolha e de inserção
(E) de inserção, de seleção e da bolha
RASCUNHO
56
Na descrição de esquemas de banco de dados relacionais, a notação A $\rightarrow$ B indica que B depende funcionalmente de A (ou que A determina B).
Admitindo-se que todas as relações apresentadas a seguir atendem à 1FN, o único esquema que se encontra na 3FN é
(A)
(B)
(C)
(D)
(E)
RASCUNHO