

UFRRJ - Sistemas de Informação - ICE/DECOMP

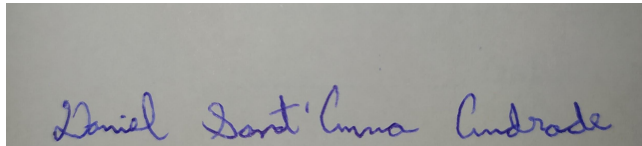
1ª. Avaliação BD II 2021-2 Prof. Sergio Serra

Nome: Daniel Sant' Anna Andrade

Nota: _____

Matrícula: 20200036904

Assinatura:



OBSERVAÇÃO: Em hipótese alguma faça uso de artifícios ilícitos para realizar esta avaliação. Nunca use consulta de terceiros ou cópias de qualquer suporte de conteúdo. Não copie do livro nem do seu colega (a prova será corrigida por comparação). A prova é **INDIVIDUAL** e pode ser feita em casa e **DEVE SER ENTREGUE ATÉ AS 17:00 DO DIA 18-OUTUBRO DE 2022** em versão assinada e em PDF. A cada 15 minutos de atraso será descontado 1.0 ponto.

Enviar as provas para serra@pet-si.ufrj.br com o cabeçalho **[BD 2021-2] Nome completo + Matrícula.**

Não serão aceitas provas sem assinatura. Prova com cabeçalho distinto do solicitado perderá 1.0 na nota final.

Prova somente a caneta AZUL ou PRETA. Cada questão vale 2.0 pontos

1ª Questão

Descreva em termos de operadores relacionais as 12 regras heurísticas utilizadas no processamento de consultas

R =

R1. Cascata de Seleções:

$\sigma_{\langle \text{condicao1} \wedge \text{condicao2} \wedge \dots \wedge \text{condicaoN} \rangle}(R) =$
 $\sigma_{\langle \text{condicao1} \rangle}(\sigma_{\langle \text{condicao2} \rangle}(\dots(\sigma_{\langle \text{condicaoN} \rangle}(R)))$

R2. Comutatividade de seleções:

$\sigma_{\langle \text{condicao1} \rangle}(\sigma_{\langle \text{condicao2} \rangle}(R)) = \sigma_{\langle \text{condicao2} \rangle}(\sigma_{\langle \text{condicao1} \rangle}(R))$

R3. Cascata de projeções:

$\pi_{\langle \text{lista1} \rangle}(\pi_{\langle \text{lista2} \rangle}(R)) = \pi_{\langle \text{lista1} \rangle}(R)$

R4. Comutatividade da seleção e projeção:

$\pi_{\langle \text{lista} \rangle}(\sigma_{\langle \text{condicao} \rangle}(R)) = \sigma_{\langle \text{condicao} \rangle}(\pi_{\langle \text{lista} \rangle}(R))$

R5. Comutatividade da junção e do produto cartesiano:

$R \times S = S \times R$

$R \bowtie S = S \bowtie R$

R6. Comutatividade da seleção e junção ou produto cartesiano
 $\sigma_{\langle \text{condicao} \rangle}(R \theta S) = (\sigma_{\langle \text{condicao} \rangle}(R)) \theta S$

R7. Comutatividade da projeção e junção ou produto cartesiano
 $\pi_{\langle \text{lista} \rangle}(R \theta S) = (\pi_{\langle \text{lista} \rangle}(R)) \theta (\pi_{\langle \text{lista} \rangle}(S))$

R8. Comutatividade da união e da interseção
 $R \cup S = S \cup R$
 $R \cap S = S \cap R$

R9. Associatividade da junção, produto cartesiano, união e interseção
 $(R \theta S) \theta T = R \theta (S \theta T)$

R10. Comutatividade da seleção e das operações de conjunto
 $\sigma_{\langle \text{condicao} \rangle}(R \theta S) = \sigma_{\langle \text{condicao} \rangle}(R) \theta \sigma_{\langle \text{condicao} \rangle}(S)$

R11. Comutatividade da projeção e união
 $\sigma_{\langle \text{condicao} \rangle}(R \cup S) = \pi_{\langle \text{lista} \rangle}(R) \cup \pi_{\langle \text{lista} \rangle}(S)$

R12. Conversão da sequência seleção/produto cartesiano em junção
 $\sigma_{\langle c \rangle}(R \times S) = R \bowtie_{\langle c \rangle} S$

2ª. Questão

Descreva, em detalhes, o funcionamento do algoritmo de junção partição-hash

R= Algoritmo de Junção Hash tem o conceito de particionar tuplas de cada relação em conjuntos. O principal objetivo é utilizar esse algoritmo para reduzir o número de comparações e aumentar a eficiência na operação de junção das relações.

O algoritmo segue o seguinte passo a passo:

- 1 - Primeiro se particiona as duas relações;
- 2 - Após o particionamento, é realizada um loop de join separadamente em cada um dos pares particionados;
- 3 - Por fim, para executar o loop de join ele cria um índice hash em uma partição e depois testa com as tuplas da outra partição.

3ª Questão

Qual é a diferença entre has estático e dinâmico

R= Em um hash estático quando se pesquisa uma chave, a função hash calcula sempre o mesmo endereço, mantendo o número de buckets fornecidos sempre inalterados durante todo o momento.

Já em um hash dinâmico, o hash é modificado conforme o conjunto de caracteres se modifica, se reorganizando para se moldar a forma como os dados estão sendo acessados.

4ª Questão

O que é a ordem p de uma árvore B+? descreva a estrutura da estrutura interna dos nós e de folhas de uma árvore B+.

R= A ordem p de uma árvore B+ é quando cada nó possui p-1 valores de busca e p ponteiros. A árvore B+ é uma estrutura de árvore que permite a eficiência da organização sequencial e aleatória de dados. Nesse tipo de árvore, todas as chaves de busca são alocadas nos nós (intermediários e iniciais) das folhas de forma a facilitar o acesso sequencial ordenado das chaves de busca e as folhas possuem os dados inseridos. Para o acesso sequencial, cada nó folha possui ponteiros para quais são os nós folhas que vêm antes e que vêm depois. A realização de uma busca fica facilitada pois para encontrar a próxima chave da sequência é só procurar pela chave_atual + 1.

5ª Questão

Anexe todas as respostas relacionadas a atividades e perguntas presentes na lista de atividades assíncronas até a data de 16/03/22. Resposta iguais de colegas terão nota ZERO na questão (até 2,0 pontos)

R=

Atividade Assíncrona 1 - Diferença entre índices primários e secundários

Os índices são uma estrutura de acesso auxiliar para agilizar a recuperação de registros. As estruturas de índices são arquivos adicionais que oferecem uma forma alternativa para acessar os registros sem afetar seu posicionamento físico no disco.

O índice em uma pesquisa serve para encontrar um registro, onde se pesquisa pelo índice e ele aponta para onde o registro está localizado no arquivo principal.

A principal forma de utilização de índice é através de arquivos ordenados, que consistem em um tipo de organização semelhante a de um índice de um livro. A representação no banco de dados é igual, o arquivo secundário serve para buscar o "termo" desejado e ele irá apontar a localização dele no banco de dados, agilizando o processo de busca.

Os índices ordenados podem ser: primário e secundário.

O índice primário é um índice que possui um campo chave de ordenação, e apenas registra grupos de informações do banco de dados. Por consequência, ele tem um menor tamanho em comparação ao arquivo original. Por exemplo, em um registro de dados cadastrais de diversos clientes, ele poderia dividir os clientes por blocos e apenas registrar o primeiro cliente de cada bloco como valor chave. Assim, ele teria um ponteiro para cada um desses blocos e facilitaria muito para realizar uma busca por nome, não precisando conferir linha por linha do banco de dados.

O índice secundário, diferente do índice primário, nem sempre irá trabalhar com um campo ordenado e ele será um índice denso, levando todas as informações dos campos para o índice. Ele ainda irá segmentar o arquivo de dados em vários blocos, mas cada linha de dados terá o campo de índice com uma chave secundária. Já a tabela de índices, irá apontar para cada um desses blocos que foram divididos, com a informação desejada estando presente em alguma linha desse bloco.

Atividade Assíncrona 2 - Dependências funcionais

Dependências Funcionais são restrições condicionadas a dos conjuntos de atributos de um banco de dados. Ela é uma restrição estabelecida no momento em que o engenheiro

modela o banco de dados com base na observação do mundo real, permitindo assim estabelecer uma formalização para avaliar a qualidade do projeto de banco de dados.

É uma propriedade semântica dos atributos. Sua principal utilização é determinar uma funcionalidade da tabela de forma mais detalhada, por meio da especificação de restrição em seus atributos durante todo o tempo.

Não é possível que exista informações no banco de dados que sejam contraditórias as dependências funcionais, devendo sempre ser garantido sua ocorrência. Essa ocorrência deve ocorrer ou através de um bom projeto de banco de dados, ou então através de programação de funções que as validem.

Um exemplo de dependência pode ser:

cli_id -> nome, endereço;

num_pedido -> produto, qtd_comprado, preço

{cli_id, num_pedido} -> cod_venda

Atividade Assíncrona 3 - Resumo de todos os tipos de normalização.

A normalização é um processo de análise e adaptação dos esquemas de tabela, minimizando informações repetidas e problemas quando for realizar as operações de banco de dados. Esse processo geralmente pega tabelas grandes e a divide em tabelas menores para melhorar o funcionamento e o processo de compreensão das mesmas.

A decomposição de tabelas, deve garantir que a junção das tabelas não seja perdida e que ainda ocorra as dependências funcionais.

A tabela abaixo demonstra algumas formas normais e seu funcionamento.

Normalização	Funcionamento
Primeira Forma Normal (1NF)	Não permite tabelas dentro de tabelas ou tabelas como valores de atributos. Transformando valores multi-valorados em valores atômicos, através da criação de novas tabelas.
Segunda Forma Normal (2NF)	Baseado no conceito de atributo primo e dependência funcional total. A segunda forma normal é quando todo atributo de uma tabela possui uma dependência funcional total, caso haja dependências funcionais parciais, é melhor dividir essa tabela em outras.
Terceira Forma Normal (3NF)	Baseado no conceito de dependência transitiva. Quando um atributo depende de outro e esse depende de outro, ocorre uma dependência transitiva. Necessitando dividir uma tabela em outras para que não ocorra essa dependência transitiva.

Atividade Assíncrona 4 - Relação entre views e Data Warehouse/Data Lake.

Views (também conhecidas como tabelas virtuais) são utilizadas para demonstrar informações calculadas ou derivadas, ou para evitar a demonstração de dados restritos. As views representam soluções valiosas, são muito comuns em empresas, pois elas possuem vários sistemas que compartilham dados, e com as views é possível agilizar processos de consulta.

As views permitem criar perfis de acessos que apenas irão mostrar dados relacionados ao usuário ou sistemas que utilizam aqueles dados específicos. Os comandos básicos são realizados nas views e elas podem ser replicadas para as tabelas originais posteriormente.

Data Warehouse são estruturas de bancos de dados, elas são utilizadas para atender processamento e armazenamento de um grande fluxo de dados, que geralmente são derivados de uma grande variedade de fontes diferentes que são registrados de acordo com uma linha do tempo.

Os dados para entrar no Data Warehouse são tratados antes de ser alocado, não podendo ser alterado após a alocação, pois a utilização principal é para registro de tempo daqueles dados.

Data Lakes são parecidos com Data Warehouse, a diferença é que os dados em um Data Lake são mais brutos, não possuindo um tratamento do mesmo para posterior alocação.

Atividade Assíncrona 5 - Manutenção da segurança de banco de dados

A segurança de um SGDB é totalmente relacionado à cultura organizacional da empresa, seguindo normas estabelecidas pela organização do que pode ou não ser realizado. A segurança é relacionada em 3 pontos: perda de integridade, perda de disponibilidade e perda de confidencialidade.

A perda de integridade é relacionado ao conteúdo do banco de dados, o protegendo de modificações como criação ou exclusão de dados de forma imprópria, podendo ser acidentais ou intencionais, acarretando em fraude ou decisão incorreta.

A perda de disponibilidade está relacionada a problemas de seu uso. Afetando a utilização de clientes e sistemas que precisam daquele serviço.

A perda de confidencialidade é relacionada aos outros dois pontos anteriores, acarretando na perda da confiança na empresa caso seus dados tenham sido violados. Em alguns casos, pode até mesmo ocorrer a violação de direitos legais de clientes.

Para que esses pontos sejam sempre protegidos, é necessário que os profissionais que gerenciam o SGBD estejam sempre preocupados em como está ocorrendo o acesso, às possíveis interferências que podem ocorrer e como proteger a passagem de informações de um sistema/cliente para outro.

Atividade Assíncrona 6 - Por que as transações são elementos fundamentais em BD relacionais?

As transações são processos de várias operações executadas sobre os dados do banco de dados. Uma transação é uma unidade lógica de trabalho, na qual é importante que toda a operação seja concluída de forma correta, ou então que se houver falha, que ela não comprometa os dados no banco de dados. O SGBD deve garantir que a execução da transação seja completa e que seja possível executar simultaneamente diversas transações sem gerar problemas nos dados.

Uma transação precisa ter blocos de execução que terão a seguinte função:

- Begin-transaction: início da transação;

- End-transaction: marca o fim da transação;
- Commit-transaction: sinaliza que a transação ocorreu com sucesso e que é possível salvar as modificações;
- Abort-transaction: sinaliza que a transação ocorreu com falha e que deve ser descartada as alterações realizadas, retornando os dados ao mesmo estado antes da transação;

Dessa forma, sempre será garantido que as diversas transações sejam realizadas sem prejudicar a integridade dos dados, mantendo a confiabilidade no serviço por trás das transações.

Boa sorte