# Matéria: Tecnologia da Informação

## Assunto: Sistemas de Informação

---

### Resumo Teórico do Assunto

Para dominar as questões sobre Sistemas de Informação, é crucial compreender conceitos fundamentais de gerenciamento de dados e estruturas de dados. Esta explicação abordará os pontos essenciais para sua preparação.

---

#### 1. Gerenciamento e Análise de Dados

Esta seção cobre os conceitos relacionados a bancos de dados modernos e sistemas de análise de grandes volumes de dados.

#### # 1.1. Bancos de Dados NoSQL (com foco em MongoDB)

**Bancos de Dados NoSQL** (Not Only SQL) são sistemas de gerenciamento de banco de dados que fornecem um mecanismo para armazenamento e recuperação de dados diferente dos modelos relacionais tradicionais. Eles são projetados para lidar com grandes volumes de dados não estruturados ou semiestruturados, alta escalabilidade e flexibilidade de esquema.

• **MongoDB** é um exemplo popular de banco de dados NoSQL do tipo **orientado a documentos**. Ele armazena dados em documentos **BSON** (Binary JSON), que são coleções de pares campo-valor, semelhantes a objetos JSON. Esses documentos são agrupados em **coleções (collections)**, que são análogas a tabelas em bancos de dados relacionais.

**• Operações Essenciais no MongoDB:**
* `insert()`: Utilizado para adicionar um novo documento a uma coleção.
* **Exemplo:** `db.minhacolecao.insert({campo1: "valor1", campo2: 123})`
* `find()`: Utilizado para consultar documentos em uma coleção. Pode ser usado com filtros para selecionar documentos específicos ou para projetar campos (selecionar quais campos retornar).
* **Exemplo (todos os documentos):** `db.minhacolecao.find()`
* **Exemplo (com filtro):** `db.minhacolecao.find({campo1: "valor1"})`
* **Exemplo (com projeção - retornar apenas campo2):** `db.minhacolecao.find({}, {campo2: 1, _id: 0})` (o `_id: 0` é para não retornar o ID padrão).
* `distinct()`: Retorna os valores **únicos** para um campo especificado em uma coleção. É fundamental para obter listas de valores sem repetição.
* **Exemplo:** `db.minhacolecao.distinct("nomeDoCampo")`

**# 1.2. Big Data e Data Warehousing**

• **Big Data** refere-se a conjuntos de dados tão grandes e complexos que os métodos tradicionais de processamento de dados não são suficientes. É frequentemente caracterizado pelos '3 V's':
* **Volume**: Grande quantidade de dados.
* **Velocidade**: Dados gerados e processados rapidamente.
* **Variedade**: Diversidade de tipos e fontes de dados (estruturados, semiestruturados, não estruturados).

• **Data Warehousing** é o processo de coletar e gerenciar dados de diversas fontes para fornecer insights de negócios significativos. Um **Data Warehouse** é um repositório central de dados integrados de uma ou mais fontes heterogêneas, usado para relatórios e análise de dados.

• **Modelagem Dimensional (Fatos e Dimensões):** É uma técnica de design de banco de dados otimizada para consultas e relatórios em Data Warehouses, contrastando com a modelagem relacional (OLTP) otimizada para transações.
* **Tabelas de Fato (Fact Tables):** Contêm as métricas ou medidas numéricas de um processo de negócio (ex: quantidade de vendas, valor da compra). Elas também contêm chaves estrangeiras para as tabelas de dimensão. No contexto da questão, um "**fato**" é um evento de ocorrência (ex: uma compra, uma movimentação de produto) que se deseja analisar.
* **Tabelas de Dimensão (Dimension Tables):** Contêm os atributos descritivos relacionados aos fatos (ex: tempo, produto, cliente, fornecedor, localização). Elas fornecem o 'contexto' para os fatos, permitindo que os dados sejam analisados sob diferentes perspectivas.

• **Cubo de Dados (Data Cube):** É uma representação multidimensional de dados, frequentemente usada em sistemas **OLAP (Online Analytical Processing)**. Ele permite que os usuários visualizem e analisem dados de diferentes perspectivas (dimensões), como tempo, produto, localização, etc. Cada célula do cubo representa uma medida (fato) para uma combinação específica de dimensões. É ideal para analisar eventos de ocorrência sob múltiplas perspectivas, permitindo "fatiar e picar" os dados.

**2. Estruturas de Dados Fundamentais**

Esta seção aborda uma estrutura de dados básica, mas essencial, para a organização e manipulação de informações.

**# 2.1. Pilhas (Stacks)**

• Uma **Pilha (Stack)** é uma estrutura de dados linear que segue o princípio **LIFO (Last-In, First-Out)**, ou seja, o último elemento a ser inserido é o primeiro a ser removido. Pense em uma pilha de pratos: o último prato colocado no topo é o primeiro a ser retirado.

• **Operações Básicas da Pilha:**
* `push()`: Adiciona um elemento ao topo da pilha.
* `pop()`: Remove e retorna o elemento do topo da pilha.
* `peek()` (ou `top()`): Retorna o elemento do topo sem removê-lo.
* `isEmpty()`: Verifica se a pilha está vazia.

• A ordem de empilhamento (`push`) e desempilhamento (`pop`) é crucial para determinar a sequência final dos elementos. Se elementos são empilhados em uma pilha (P1) e depois desempilhados de P1 e empilhados em outra pilha (P2), a ordem em P2 será a inversa da ordem em que foram desempilhados de P1.

---

Compreender esses conceitos permitirá que você analise as questões e identifique as operações e estruturas de dados corretas para cada cenário proposto.

---

## Questões de Provas Anteriores

**Fonte: escriturario_agente_de_tecnologia (1).pdf, Página: 25**

**pcimarkpci MjgwNDowMTRkOjE0YTU6OTI1ODozOGQ2OjNhMGM6NTM0MzplZmI1:U3VuLCAyNyBKdWwgMjAyNSAyMzo0Nzo0MCAtMDMwMA==**
**www.pciconcursos.com.br**
**25**
**BANCO DO BRASIL**
**AGENTE DE TECNOLOGIA - Microrregião 16 DF-TIGABARITO 1**
**66**
**Um administrador de um banco de dados construído por meio do MongoDB inseriu dados em uma coleção (collection) de**
**dados da seguinte forma:**

**db.fornecedores.insert( {**
**codigo: “thx1138“,**
**nome: “Roupas Syfy ltda“,**
**pais: “Arabia Saudita“ } )**
**Posteriormente, esse administrador construiu uma consulta que retornou apenas o nome, sem repetição, de todos os**
**países que fazem parte dessa coleção (collection).**
**O comando utilizado para tal consulta foi**

(A) db.fornecedores.find("pais")
(B) db.fornecedores.find().pretty({"pais":1})
(C) db.fornecedores.find().sort({"pais":1})
(D) db.fornecedores.distinct({"pais":0})
(E) db.fornecedores.distinct( "pais" )
67
Uma empresa precisa implementar um sistema Big Data para controlar a movimentação dos produtos que a empresa
oferece. Esse sistema precisa estar com a configuração de dados como sendo um fato, que seria um evento de ocorrên-
cia, como, por exemplo: as compras de um determinado insumo, em um determinado fornecedor e em um determinado
instante.
Para tal finalidade, esse sistema a ser implementado deverá estar organizado segundo a configuração de
(A) Cubo de dados
(B) Tuplas estáticas
(C) Matriz de ocorrência
(D) Documentos lineares
(E) Subconjunto de atributos
68
Em um determinado treinamento de pessoal de TI, para facilitar o aprendizado sobre o funcionamento da estrutura de
dados PILHA, utilizou-se o jogo de trocas, cujas regras são apresentadas a seguir.
JOGO DAS TROCAS - REGRAS
Para começar o jogo, o jogador recebe duas pilhas, P1 e P2.
P1 está preenchida com quatro fichas, identificadas por nomes fictícios e empilhadas em ordem alfabética CRESCENTE
a partir do topo.
P2 está inicialmente vazia.
Uma ficha desempilhada de P1 é imediatamente empilhada em P2.
A operação (P2,pop) acarreta impressão do nome que está na ficha desempilhada e descarte da ficha.
Para ganhar o jogo, o jogador precisa determinar corretamente, dentre sequências derivadas da sequência inicial, por
troca da posição de seus elementos, qual delas poderia ser impressa com essas operações.
No início do jogo, foram dadas as pilhas P2, vazia, e P1 preenchida com as seguintes operações de empilhamento:
push(P1,Zeus); push(P1,Hades); push(P1,Cibele); push(P1, Apolo).
Considerando-se esse cenário, qual seria a sequência possível de ser impressa, da esquerda para a direita, de acordo
com as regras do JOGO DAS TROCAS?
(A) Apolo, Zeus, Cibele, Hades
(B) Hades, Apolo, Zeus, Cibele

(C) Zeus, Cibele, Apolo, Hades
(D) Hades, Apolo, Cibele, Zeus
(E) Cibele, Hades, Apolo, Zeus