1. O que você entende por Persistência de Dados?

**Resposta:** A persistência de dados refere-se à capacidade de armazenar informações seguras e recuperá-las posteriormente, mesmo após o encerramento do aplicativo ou sistema em uso. Este é um conceito fundamental no desenvolvimento de software e sistemas de informação porque permite que os dados sejam armazenados de forma confiável e permanente para acesso, modificação e referência futuros.

1. Quais as diferenças entre objetos transientes e objetos persistentes? Explique.

**Resposta:** Objetos transientes e objetos persistentes são conceitos que estão relacionados à persistência de dados em programação, especialmente em contexto de mapeamento objeto-relacional (ORM) em linguagens orientadas a objetos, como Java, C#, Python e outros. Eles se referem ao estado dos objetos em relação ao armazenamento de dados, como em um banco de dados. Aqui estão as principais diferenças entre eles: 1. \*\*Objetos Transitórios:\*\* - \*\*Definição:\*\* Objetos transitórios são objetos que não estão associados a nenhum sistema de armazenamento persistente, como um banco de dados. Eles existem apenas na memória da aplicação e não têm representação correspondente em um banco de dados.

1. Qual a principal diferença entre Banco de Dados Relacional e Banco de Dados Orientada a Objetos?

**Resposta:** A principal diferença entre Bancos de Dados Relacionais (RDBMS) e Bancos de Dados Orientados a Objetos (OODBMS) está na modelagem de dados. RDBMS usam tabelas e SQL, enquanto OODBMS usam objetos e linguagens de programação para representar dados. RDBMS são altamente estruturados, enquanto OODBMS oferecem mais flexibilidade. A escolha depende dos requisitos do projeto e da preferência de modelagem de dados.

1. Qual o objetivo do mapeamento objeto-relacional (ORM)?

**Resposta:** O mapeamento objeto-relacional (ORM) tem como objetivo principal simplificar a integração entre bancos de dados relacionais e aplicativos orientados a objetos. Ele abstrai o banco de dados, permitindo que os desenvolvedores trabalhem com objetos em seu código, mapeando automaticamente esses objetos para tabelas de banco de dados e vice-versa. Isso reduz a necessidade de escrever consultas SQL manualmente, torna o código mais legível e portátil, acelera o desenvolvimento, simplifica a manutenção e integra conceitos de programação orientada a objetos com persistência de dados em bancos de dados relacionais. O ORM atua como uma camada intermediária que facilita o acesso e a manipulação de dados em aplicativos, melhorando a produtividade e a eficiência no desenvolvimento de software.

1. Defina:

a) Dados estruturados; b) Dados semiestrurados; c) Dados não estruturados.

**Resposta:** a) Dados Estruturados: Dados estruturados são informações organizadas em um formato predefinido e altamente organizado. Eles são caracterizados por terem uma estrutura clara e consistente, onde os tipos de dados e as relações entre eles são bem definidos. Exemplos comuns de dados estruturados incluem tabelas em bancos de dados relacionais, onde cada coluna representa um tipo de dado específico e cada linha representa uma entrada de dados.

b) Dados Semiestruturados: Dados semiestruturados são informações que não seguem uma estrutura rigorosamente definida, mas ainda têm alguma forma de organização. Eles não se encaixam perfeitamente em tabelas ou esquemas rígidos, mas possuem elementos de estrutura, como tags, chaves e valores. Documentos JSON e XML são exemplos de formatos de dados semiestruturados, onde os dados podem variar em estrutura, mas seguem uma organização geral.

c) Dados Não Estruturados: Dados não estruturados são informações que não possuem uma organização predefinida ou estrutura formal. Eles são tipicamente representados em formato de texto livre e não estão organizados em tabelas, registros ou esquemas. Exemplos de dados não estruturados incluem texto em documentos, e-mails, mídia social, áudio e vídeo não transcritos. Esses dados geralmente exigem técnicas avançadas de processamento de linguagem natural (NLP) para análise e extração de informações significativas.

1. Qual o nome da biblioteca responsável pela extração/captura de dados disponíveis em arquivos HTML ou XML? Explique.

**Resposta:** A biblioteca "Beautiful Soup" é uma ferramenta em Python que facilita a análise e extração de dados de arquivos HTML ou XML. Ela permite a navegação na estrutura do documento, busca de informações específicas por meio de tags e atributos, e até mesmo modificações no documento. É amplamente utilizada em web scraping e análise de dados da web para extrair informações de páginas da web de forma eficiente.

1. Os arquivos do tipo XML (eXtensible Markup Language) surgiram como forma de estruturação e troca de dados pela internet. Dentre suas principais característica preencha os seguintes questionamentos: a) Sintaxe inicial na primeira linha do arquivo.xml b) Os dados são organizados em formato hierárquico ou tabular? c) Quais são as formas de representação de um documento XML. Justique.

**Resposta:** O XML (eXtensible Markup Language) é usado para estruturar e trocar dados na internet. Características principais:

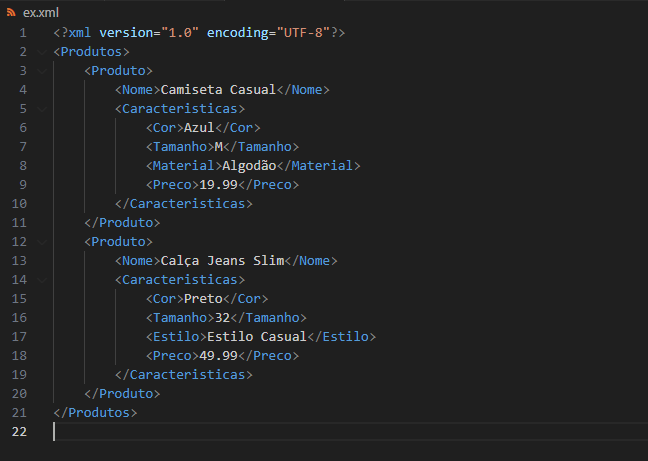
a) Sintaxe inicial: A primeira linha inclui uma declaração com a versão do XML e a codificação de caracteres, como <?xml version="1.0" encoding="UTF-8"?>.

b) Organização: Os dados no XML são organizados em uma estrutura hierárquica, onde os elementos são aninhados para formar uma árvore de dados.

c) Representação: Um documento XML pode ser representado usando elementos, atributos e uma árvore hierárquica, facilitando a legibilidade e estruturação de informações. Pode ser convertido em texto plano ou notação JSON conforme necessário.

O XML é flexível e adequado para diversas aplicações, como configuração, troca de dados e documentos estruturados, graças à sua capacidade de representação hierárquica de informações.

1. Elabore um documento xml sobre produtos disponíveis para venda em empresas do comércio eletrônico/móveis/imóveis/roupas, a partir das seguintes condições: ▪ O produto deve possuir 5 características; ▪ Cada produto deve ter um nome de identificação; ▪ No documento deverá ter pelo menos dois produtos preenchidos.

**Resposta:** 

1. Defina o que é um documento JSON e quais suas principais características.

**Resposta:** Um documento JSON (JavaScript Object Notation) é uma forma de estruturar dados que usa pares chave-valor, oferecendo suporte a vários tipos de dados, como strings, números e objetos aninhados. Sua sintaxe é simples e legível, facilitando a interpretação por humanos e máquinas. JSON é independente de linguagem, eficiente em termos de espaço, e amplamente utilizado na comunicação entre sistemas, especialmente na web, devido à sua versatilidade e facilidade de integração. Sua estrutura hierárquica e leveza o tornam uma escolha popular para representar e trocar informações

10.O que significa o processo de serialização (JSON.stringify) e desserialização (JSON.parse) de documentos do tipo JSON?

A serialização é o processo de converter dados de objetos JavaScript em uma string JSON legível, permitindo armazenamento ou transmissão. A desserialização é o inverso, convertendo uma string JSON de volta em objetos JavaScript. Ambos são cruciais para trocar dados entre aplicativos e sistemas, mantendo a integridade dos dados. A função JSON.stringify() serializa, enquanto JSON.parse() desserializa. Isso facilita o compartilhamento de informações entre diferentes plataformas e linguagens de programação. O formato JSON é amplamente utilizado na web e APIs REST devido à sua simplicidade e compatibilidade.

11.Faça um exemplo de documento JSON a partir de dados sobre serviços de vendas online. • Utilize dados do tipo, string, inteiro, array e objetos.

**Resposta:**



12.Quais são as principais diferenças entre documentos do tipo JSON e XML

**Resposta:** As principais diferenças entre JSON e XML incluem a sintaxe mais concisa e legível do JSON em comparação com a sintaxe mais verbosa do XML. JSON gera documentos mais compactos, é mais eficiente em termos de espaço e mais fácil de analisar por máquinas devido à sua estrutura simples de pares chave-valor. Por outro lado, o XML oferece suporte a tipos de dados mais flexíveis e é amplamente adotado em várias indústrias, enquanto o JSON é mais comum na web moderna, especialmente em APIs RESTful. A escolha entre eles depende dos requisitos do projeto e das preferências da equipe de desenvolvimento.

13.Para que serve utilizar JDBC com Sistemas de Gerenciamento de Banco de Dados

**Resposta:** O JDBC (Java Database Connectivity) é uma API em Java que permite a conexão e interação com Bancos de Dados. Ele é utilizado para realizar diversas operações, como acesso, consulta, inserção e atualização de dados em Sistemas de Gerenciamento de Banco de Dados (SGBDs). O JDBC desempenha um papel fundamental na integração de aplicativos Java com bancos de dados, garantindo a interoperabilidade entre diferentes SGBDs, oferecendo segurança, controle transacional e otimização de consultas. Ele é amplamente utilizado em aplicativos empresariais e fornece portabilidade de código, permitindo que os desenvolvedores criem aplicativos que podem se conectar a diferentes bancos de dados com facilidade.

14.Quais são os principais componentes durante a implementação do JDBC? Explique.

**Resposta:** A implementação do JDBC (Java Database Connectivity) envolve vários componentes essenciais:

Driver JDBC: É a ponte entre o aplicativo Java e o Banco de Dados, permitindo a comunicação.

Conexão com o Banco de Dados: É estabelecida usando a classe Connection e requer informações como URL, nome de usuário e senha.

Statement ou PreparedStatement: São usados para executar consultas SQL no Banco de Dados, sendo o último preferido para consultas parametrizadas.

ResultSet: Armazena o resultado das consultas, permitindo a iteração pelas linhas de dados.

Tratamento de Exceções: Lida com erros, como problemas de conexão ou consultas malformadas, usando exceções como SQLException.

Fechamento de Recursos: É importante liberar recursos, como conexões e resultados, quando não são mais necessários.

Gerenciamento de Transações (opcional): Suporta operações de transações, como commit e rollback, para manter a integridade dos dados.

Esses componentes são essenciais para conectar aplicativos Java a Bancos de Dados, permitindo acesso, consulta e manipulação de dados de forma eficiente.

15.Cites restrições sobre a utilização do JDBC para sistemas atuais.

**Resposta:** Restrições ao usar o JDBC em sistemas atuais incluem a escolha adequada do driver JDBC compatível com o SGBD, a necessidade de garantir segurança contra injeção de SQL, o gerenciamento manual de conexões, otimização de desempenho, considerações de portabilidade de código, tratamento de concorrência, manutenção de código, requisitos de conectividade, limitações do próprio JDBC e possíveis complexidades envolvidas. É essencial entender e mitigar essas restrições ao incorporar o JDBC em sistemas modernos.