**1) Explique o que a query abaixo está fazendo e diga qual é o recurso de SQL que está sendo utilizado para criar as tabelas. " WITH x\_table AS (SELECT 1 as Int1, 2 as Int2) , y\_table AS (SELECT 2 as Int1, 3 as Int2) SELECT \* FROM x\_table UNION SELECT \* FROM y\_table ":**

A query acima está criando duas tabelas temporárias, **x\_table** e **y\_table**, usando a cláusula **WITH**. Em seguida, a query faz uma seleção de todas as colunas das duas tabelas usando a cláusula **SELECT**. Essa seleção é feita em duas partes: primeiro, são selecionadas todas as linhas da tabela **x\_table** usando **SELECT \* FROM x\_table**, e em seguida, são selecionadas todas as linhas da tabela **y\_table** usando **SELECT \* FROM y\_table**.

O resultado final é a combinação (união) das linhas das duas tabelas, sem duplicatas, obtido através da cláusula **UNION**. O **UNION** é um recurso de SQL que combina os resultados de duas consultas em uma única tabela resultante, excluindo quaisquer linhas duplicadas.

Portanto, essa query está criando duas tabelas temporárias e unindo seus resultados em uma única tabela resultante. O recurso de SQL utilizado para criar as tabelas temporárias e realizar a união é o **WITH** e o **UNION**, respectivamente.

**2) O que é e quais as vantagens do uso de CTE?**

Em SQL, CTE (Common Table Expression), ou Expressão de Tabela Comum em português, é uma construção que permite definir uma consulta nomeada que pode ser referenciada posteriormente na mesma query. Basicamente, é uma forma de criar uma "tabela temporária" que pode ser utilizada no restante da consulta.

As vantagens do uso de CTE incluem:

1. Legibilidade e organização do código: Com CTEs, é possível dividir uma consulta complexa em partes menores e mais gerenciáveis. Cada CTE representa uma parte lógica da consulta, tornando o código mais fácil de entender e dar manutenção.
2. Reutilização de resultados intermediários: Como uma CTE pode ser referenciada várias vezes na mesma consulta, é possível reutilizar os resultados intermediários de uma subconsulta em várias partes da query principal, evitando a repetição de código e melhorando a eficiência.
3. Simplificação de consultas recursivas: O uso de CTEs é especialmente útil em consultas recursivas, onde uma consulta é aplicada repetidamente a um conjunto de resultados até que uma condição de parada seja alcançada. As CTEs permitem definir a consulta recursiva uma vez e referenciá-la facilmente na parte recursiva da query.
4. Otimização do desempenho: Dependendo do sistema de gerenciamento de banco de dados (SGBD) utilizado, as CTEs podem ser otimizadas pelo mecanismo de consulta, resultando em consultas mais eficientes e melhor desempenho.

Em resumo, o uso de CTEs proporciona um código mais legível, modular e reutilizável, permitindo a criação de consultas complexas de forma mais organizada e eficiente.