Data Engineer Scenario Based Interview Questions!!  
  
1. You're designing a data warehouse for a company with large amounts of customer data. How would you optimize storage costs while ensuring query performance?  
  
2. You've noticed that a certain ETL process is taking up a significant amount of compute resources, leading to high costs. How would you optimize the process to reduce costs?  
  
3. You're tasked with migrating a company's on-premises data infrastructure to the cloud. How would you minimize costs during the migration process?  
  
4. You've identified that a certain dataset is being stored in a high-cost storage solution. How would you determine the best storage solution to reduce costs?  
  
5. You're designing a data pipeline for a company with fluctuating data volumes. How would you ensure cost-effectiveness while handling varying workloads?  
  
6. You've noticed that data quality issues are leading to rework and increased costs. How would you implement data quality checks to reduce costs?  
  
7. You're tasked with implementing a data archiving solution for a company with large amounts of historical data. How would you minimize storage costs while ensuring data accessibility?  
  
8. You've identified that a certain data processing job is running inefficiently, leading to high costs. How would you optimize the job to reduce costs?  
  
These scenario-based questions assess the candidate's ability to identify areas for cost savings, optimize resources, and implement cost-effective solutions in a data engineering context.

1. What cluster Manager you have used in your project ?  
   2. What is your cluster Size ?  
   3. How does your data comes to your storage location ?  
   4. What are the other sources you have used in your project ?  
   5. what is the sink for your project /  
   6. What is the frequency of the data in your source ?  
   7. What is the volume of your data ?  
   8. Please explain your project in detail ?  
   9. Lets say out of 100 task, 99 tasks completed however the last task is taking long hours to finish/complete, how to handle this issue ?1  
   10. What all challenges you have faced and how did you overcome from it ?  
   11. what optimization technique you have used in your project and what is the reason for it ?  
   12. Have you done spark optimization tuning ? If yes, how you have done that ?  
   13. Can you please walk me through the spark-submit command ?  
   14. Lets say you are getting your data volume is 100 GB , In your spark you are doing 5 Actions and 3 transformations on the data, explain what goes behind the scene with respect to Stages ,tasks?  
   15. how do you take your code to the higher environment ?  
   16. How do you schedule your job in production ?  
   17. How do you reprocess the data if it failed ?  
   18. Tell me one scenario you have gone wrong with your decision making and what you have learnt from that mistake ?  
   19. Lets say you have noticed duplicate records loaded in the table for the particular partition, how you resolved such issues ?  
   20. What is the frequency of your jobs ?  
   21. How do you notify your business/Stakeholders in case of any job failure?

SQL on SUNDAY  
  
Here are 30 new essential SQL questions to help you prepare for your next interview:  
  
🔗Window Functions  
  
1. Calculate the cumulative sales for each product category.  
2. Find the top three performers in each department based on their annual sales using dense rank.  
3. Compute the difference between the highest and lowest order value for each customer.  
4. Calculate the running total of salaries by department.  
5. Determine the average monthly sales for each region over the past twelve months.  
  
🔗Common Table Expressions (CTEs)  
  
1. Use a recursive CTE to calculate the factorial of numbers from one to ten.  
2. Write a CTE to split email addresses into username and domain parts.  
3. Identify gaps in a sequence of order IDs using a CTE.  
4. Use a CTE to calculate the total working hours of employees grouped by department.  
5. Detect overlapping date ranges for bookings using a CTE.  
  
🔗Joins (Inner, Outer, Cross, Self)  
  
1. Retrieve all employees along with their assigned projects, including employees without projects (Right Join).  
2. Find customers who placed exactly one order using a self-join.  
3. List all products along with the suppliers who didn’t supply them (Outer Join).  
4. Generate all possible pairs of employees and calculate the distance between their office locations using a Cross Join.  
5. Retrieve a list of employees who share the same job title using a self-join.  
  
🔗Subqueries  
  
1. List the products whose average sales exceed the overall average sales.  
2. Find employees who earn more than the manager of their department.  
3. Identify customers who have placed orders for more than five unique products.  
4. Retrieve the top three highest-selling products for each category using a correlated subquery.   
5. List employees working in departments with fewer than three employees.  
  
🔗Aggregate Functions  
  
1. Calculate the mode (most frequently occurring value) of product prices in each category.   
2. Find the total number of orders placed in each quarter and rank them by volume.  
3. Count the number of employees earning above the median salary.  
4. Identify the month with the highest average sales in each region.  
5. Compute the variance and standard deviation of sales for each product category.  
  
🔗Indexing and Performance  
  
1. Write a query to find the most frequently accessed rows in a table using indexed columns.   
2. Analyze the impact of adding an index on a large text column.   
3. Identify queries that perform full table scans and suggest indexing improvements.   
4. Compare the performance of a single-column index versus a composite index on query execution time.  
5. Write a query to monitor index usage statistics for a database table.

SQL Interview Question - CTE VS Table Variable VS Temp Table!  
  
Are you ready to ace your next SQL interview? Understanding the differences between a Common Table Expression (CTE), Table Variable, and Temp Table is crucial for success. Let's dive in!  
  
1. CTE (Common Table Expression)  
1. Definition: A temporary result set that simplifies complex queries.  
2. Structure: Defined with a WITH clause.  
3. Behavior: Evaluated once, with results usable in subsequent query parts.  
4. Lifetime: Exists only for the duration of the query.  
5. Indexing: No indexing allowed.  
  
2. Table Variable  
1. Definition: A temporary storage location for data.  
2. Structure: Declared with a DECLARE statement.  
3. Behavior: Can be updated, inserted, and deleted like a regular table.  
4. Lifetime: Exists until the end of the batch.  
5. Indexing: No indexing allowed.  
  
3. Temp Table  
1. Definition: A temporary physical table that stores data.  
2. Structure: Created with a CREATE TABLE statement.  
3. Behavior: Can be updated, inserted, and deleted like a regular table.  
4. Lifetime: Exists until explicitly dropped.  
5. Indexing: Indexing allowed.  
  
Here are some examples:  
  
1. CTE  
WITH SalesCTE AS (  
SELECT SalesPersonID, SUM(SalesAmount) AS TotalSales  
FROM Sales  
GROUP BY SalesPersonID  
)  
SELECT SalesPersonID, TotalSales  
FROM SalesCTE  
WHERE TotalSales > 100000;  
  
2. Table Variable  
DECLARE @Sales TABLE (SalesPersonID INT, TotalSales DECIMAL(10, 2));  
INSERT INTO @Sales  
SELECT SalesPersonID, SUM(SalesAmount)  
FROM Sales  
GROUP BY SalesPersonID;  
SELECT SalesPersonID, TotalSales  
FROM @Sales  
WHERE TotalSales > 100000;  
  
3. Temp Table  
CREATE TABLE [hashtag#Sales](https://www.linkedin.com/feed/hashtag/?keywords=sales&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7281976781564243969)  
(  
SalesPersonID INT,  
TotalSales DECIMAL(10, 2)  
);  
INSERT INTO [hashtag#Sales](https://www.linkedin.com/feed/hashtag/?keywords=sales&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7281976781564243969)  
SELECT SalesPersonID, SUM(SalesAmount)  
FROM Sales  
GROUP BY SalesPersonID;  
SELECT SalesPersonID, TotalSales  
FROM [hashtag#Sales](https://www.linkedin.com/feed/hashtag/?keywords=sales&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7281976781564243969)  
WHERE TotalSales > 100000;  
DROP TABLE [hashtag#Sales](https://www.linkedin.com/feed/hashtag/?keywords=sales&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7281976781564243969);  
  
Which one would you use and why? Share your thoughts!

Data Engineer Interview!!  
  
Interviewer: You're running a Spark job on a large dataset stored in HDFS. Suddenly, you notice that the job is taking longer than usual to complete. How would you troubleshoot ?  
  
Candidate: When faced with a slowdown in a Spark job, several factors could be contributing to the issue. Here's how I would approach:  
  
Interviewer: What would be your initial steps in troubleshooting the slow Spark job?  
  
I would check the Spark UI to gather information about the job's execution, including task progress, stage durations, and resource utilization. This can help identify bottlenecks and performance issues within the job.  
  
Interviewer: Could you provide some specific metrics or indicators you'd look for in the Spark UI?  
  
I'd focus on metrics such as task duration, shuffle read/write times, executor CPU and memory utilization, and garbage collection activity. These metrics can provide insights into potential performance bottlenecks, such as data skew, resource contention, or inefficient task execution.  
  
Interviewer: If you notice high shuffle read/write times in the Spark UI, how would you investigate further?  
  
High shuffle read/write times often indicate issues with data skew or inefficient shuffle operations. I would drill down into the stage details in the Spark UI to identify tasks with disproportionately high shuffle read/write times. Analyzing the data distribution and partitioning strategy can help pinpoint the cause of the skew and optimize the shuffle operations accordingly.  
  
Interviewer: What steps would you take if you suspect resource contention as the cause of the slowdown?  
  
If resource contention is suspected, I would examine executor CPU and memory utilization to identify any resource bottlenecks. Increasing executor memory or adjusting the number of executors can help alleviate resource contention and improve job performance. Additionally, optimizing resource allocation and task scheduling parameters in the Spark configuration can further optimize resource utilization.  
  
Interviewer: Suppose you've optimized resource utilization, but the job is still running slower than expected. What other factors would you consider?  
  
If resource utilization is optimized, I would investigate other potential factors impacting job performance, such as inefficient data processing logic, data skew, or suboptimal partitioning strategies. Analyzing the job's DAG and execution plan can provide insights into the data processing flow and identify opportunities for optimization.  
  
Interviewer: How would you ensure the stability and reliability of the Spark job after troubleshooting and optimization?  
  
After troubleshooting and optimization, I would conduct thorough testing to validate the stability and reliability of the Spark job under varying workload conditions and data scenarios. This includes performance testing, stress testing, and fault tolerance testing to ensure the job performs reliably and efficiently in production environments.