**Capco AWS Data Engineer Interview Guide – Experienced 3+**

**Interview Process Breakdown**

**Round 1: Technical Interview (Hands-On Coding & Conceptual Questions)**

This round focused on Python coding, Spark transformations, SQL concepts, and general data engineering fundamentals.

**Round 2: Advanced Technical & Scenario-Based Interview**

This round evaluated real-world problem-solving skills through scenario-based questions covering AWS, Redshift, Spark, fault tolerance, and data modeling. Experience-based questions explored past projects and the candidate's role in solving specific challenges.

**Technical Round 1**

**1. Python Coding Challenge**

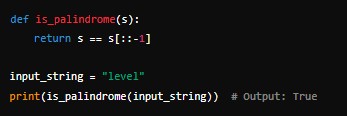
 **Problem**: Write a program to check if a given string is a palindrome.

 **Tips for Success**:

 Use slicing (string[::-1]) or a two-pointer approach for optimal solutions.

 Be prepared to explain time and space complexity.

 **Example Code**:



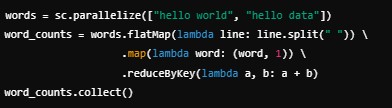
**2. PySpark Word Count Problem**

 **Challenge**: Implement a word count using PySpark.

 **Insight**: Use **flatMap()**, **map()**, and **reduceByKey()** for an efficient solution.

 **Common Mistake**: Using groupByKey(), which causes memory inefficiency due to shuffling.

 **Example**:



**3. Key Conceptual Questions**

 **reduceByKey vs. groupByKey**:

*reduceByKey combines values locally before shuffling, making it more efficient than groupByKey, which transfers all key-value pairs across nodes.*

 **Fault Tolerance in Spark vs. Hadoop**:

*Spark achieves fault tolerance through lineage (DAG) and RDD recomputation, whereas Hadoop relies on replication in HDFS.*

 **SQL Query Execution Order**:

 Example Query: SELECT col FROM table WHERE condition GROUP BY col

HAVING condition ORDER BY col;

 Execution Order: FROM → WHERE → GROUP BY → HAVING → SELECT

→ ORDER BY.

 **DENSE\_RANK vs. RANK**:

*RANK allows gaps in rank values, while DENSE\_RANK doesn’t skip numbers.*

 **Cursor vs. Stored Procedure**:

Cursor iterates row-by-row in SQL (less efficient), while stored procedures allow reusable query logic with better performance.

 **Python ‘pass’ Statement**:

*A placeholder when a block of code is syntactically required but no action is needed.*

 **Memory: List vs. Tuple**:

*Lists occupy more memory due to dynamic resizing, while tuples are fixed-size and more memory-efficient.*

**Technical Round 2**

**1. How would you design a fault-tolerant data ingestion pipeline using AWS services**

**(S3, Kinesis, Lambda)?**

 What alternatives to Kinesis would you consider for real-time data ingestion?

 How would you handle retry logic and error handling in the Lambda function?

 How do you ensure message ordering in Kinesis Streams?

 What are the differences between Kinesis Data Firehose and Kinesis Streams?

**2. Explain how you would optimize Redshift query performance for a reporting system with large fact tables.**

 How would you decide between using DISTKEY and SORTKEY?

 What are the benefits and drawbacks of using compression encodings in Redshift?

 Explain the impact of Vacuum and Analyze operations on performance.

 How would you monitor and reduce disk-based queries (disk spilling)?

**3. Describe how to implement cross-region replication for an S3 bucket.**

 How would you ensure data consistency between the source and destination regions?

 What are the cost implications of cross-region replication?

 How would you handle replication for objects encrypted with SSE-KMS?

 How does Versioning impact replication behavior?

**4. What steps would you take to secure data stored in S3?**

 What are the differences between SSE-S3, SSE-KMS, and SSE-C encryption?

 How would you enforce encryption at rest for all objects in a bucket?

 Explain how Bucket Policies differ from IAM Policies.

 What role does Amazon Macie play in securing sensitive data in S3?

**5. Explain using AWS Glue for ETL. What challenges might you face with large datasets?**

 How would you optimize Glue jobs to reduce processing time for large datasets?

 What strategies would you use to manage dynamic partitions efficiently?

 Explain how Glue’s Spark-based architecture handles data parallelism.

 How would you debug a failed Glue job with limited logging information?

**6. How would you monitor a data pipeline in AWS to ensure SLA compliance?**

 What metrics would you track in CloudWatch for a Kinesis-based pipeline?

 How would you implement custom alarms for data delays or job failures?

 Describe using Step Functions to handle retries and error notifications.

 How would you set up end-to-end tracing for a complex pipeline?

**7. Describe handling schema evolution in AWS Redshift without downtime.**

 How would you add columns to a table without impacting queries?

 Explain the differences between table re-creation and ALTER TABLE operations.

 How would you handle data type changes for an existing column?

 What are the best practices for managing external schema evolution with Spectrum?

**8. How would you automate Redshift cluster scaling for peak loads?**

 Explain the use of Elastic Resize vs. Classic Resize in Redshift.

 What are the trade-offs between Concurrency Scaling and using Reserved

Instances?

 How would you configure workload management (WLM) queues for heavy queries?

 What metrics would trigger an auto-scaling event?

**9. Explain using IAM roles for secure cross-account access to an S3 bucket.**

 How does the trust relationship policy in IAM roles work?

 What are the security risks of using overly permissive role policies?

 Explain how Access Control Lists (ACLs) can affect IAM role permissions.

 How do bucket policies handle the Principal element for cross-account roles?

**10. Describe a custom EMR cluster configuration for Spark-based ETL with minimal cost.**

 What types of instance types would you choose for cost efficiency?

 How would you configure Spot Instances for a resilient EMR cluster?

 Explain the benefits of auto-scaling policies in EMR.

 What role does the Instance Fleet configuration play in cost optimization?

**11. How would you prevent small file problems in S3 when loading data into Redshift?**

 What are the benefits of the COPY command’s MANIFEST option?

 How would you use Amazon Glue to merge small files?

 Explain how using a staging area in S3 can help.

 How does the MAXERROR parameter affect data loading in Redshift?

**12. Explain the use of Amazon Athena for serverless querying.**

 What are the pricing models for queries in Athena?

 How does partitioning in S3 affect Athena query performance?

 Explain the role of Glue Catalog in Athena.

 What types of queries would not be efficient in Athena?

**13. Describe a real-world use case for using Step Functions with Lambda in a data workflow.**

 How would you handle a failure state in a Step Functions workflow?

 What are the advantages of using Wait and Choice states?

 Explain how Step Functions integrate with other AWS services.

 How would you pass data between Lambda functions in Step Functions?

**14. How would you design a data archiving strategy in S3 using lifecycle policies?**

 How would you manage transitions to Glacier Instant Retrieval and Deep Archive?

 What is the impact of multipart uploads on lifecycle policies?

 How would you handle data expiration for time-sensitive logs?

 What are the cost implications of using Standard-IA for archiving?

**15. What are the trade-offs between using Glue Catalog vs. Hive Metastore for metadata management?**

 How does Glue Catalog handle schema versioning compared to Hive Metastore?

 What integration challenges might you face with Glue Catalog in non-AWS

environments?

 Explain how partition discovery works in Glue compared to Hive.

 How would you migrate metadata from Hive Metastore to Glue?