ChatGPT

AWS Glue & EMR Interview Questions and Answers

Part 1: AWS Glue Interview Questions

Basic Level

1. **What is AWS Glue?**\ AWS Glue is a fully managed ETL (Extract, Transform, Load) service for preparing and loading data for analytics.

2. **Key Components of AWS Glue:**

3. **Crawler** – Infers schema and creates Data Catalog metadata.

4. **Job** – Executes ETL logic using PySpark or Python.
5. **Data Catalog** – Central metadata repository.
6. **Trigger** – Schedules or orchestrates jobs.

7. **Notebook/Studio** – For interactive development.

8. **Languages Supported:**\ PySpark, Python (Python Shell), Scala (deprecated in newer versions).

9. **What is Glue Data Catalog?**\ A central metadata store shared across AWS services like Athena, Redshift, EMR, etc.

10. **Types of Glue Jobs:**

11. Spark Jobs

12. Python Shell Jobs

13. Ray-based Jobs (ML in Glue 4.0+)

14. **What is a Glue Crawler?**\ It scans data sources and creates/updates metadata in the Data Catalog.

Intermediate to Advanced

1. **What is Job Bookmark in AWS Glue?**\ Tracks previously processed data, enabling incremental ETL.

2. **Optimizing Glue Job Performance:**

3. Partition data

4. Use pushdown predicates
5. Convert DynamicFrames to DataFrames

6. Avoid unnecessary I/O

7. **Handling Schema Evolution:**

8. Use `resolveChoice()` in DynamicFrames

9. Regularly update the crawler metadata

10. **DynamicFrame vs DataFrame:**

    - DynamicFrame supports schema inference, nested structures, and AWS Glue transformations.
    - Easily convertible to/from Spark DataFrame.

11. **Pushdown Predicate Example:**

```
glueContext.create_dynamic_frame.from_catalog(
    database="mydb",
    table_name="mytable",
    push_down_predicate="year=='2024'")
```

1. **Glue Job Error Handling:**

    - Enable CloudWatch logs
    - Use dead-letter S3 path
    - Implement try-catch blocks

2. **Schema Evolution Strategy:**

    - Let crawler re-run with schema change detection enabled
    - Use DynamicFrame's flexible schema structure

3. **Glue Studio & Visual ETL:**\ A UI-based job authoring tool for non-coders and visual debugging.

Part 2: AWS EMR Interview Questions

Basic Level

1. **What is AWS EMR?**\ A managed Hadoop/Spark cluster service for big data processing.

2. **EMR Node Types:**

3. **Master Node** – Manages cluster & coordinates tasks

4. **Core Node** – Executes tasks & stores HDFS data
5. **Task Node** – Executes tasks only (no HDFS)

6. **What is EMRFS?**\ EMR File System allows EMR to use Amazon S3 as a data layer.

7. **Steps vs Bootstrap Actions:**

8. **Step** – Job submitted (e.g., Spark, Hive)

9. **Bootstrap** – Shell script executed during cluster startup

---

Intermediate Level

1. **Spark Job Submission in EMR:**

```
spark-submit s3://my-bucket/scripts/job.py
```

1. **Optimize EMR Performance:**

2. Use `r5` or `c5` instances

3. Enable dynamic allocation
4. Use spot instances

5. Partition and compress input/output

6. **HDFS vs EMRFS:**

7. HDFS is ephemeral storage on EC2

8. EMRFS is backed by Amazon S3 (persistent)

9. **Monitoring Tools:**

10. Amazon CloudWatch

11. Spark History Server

12. Ganglia (optional)

13. **Debugging Failed Steps:**

14. Check EMR console step logs
15. CloudWatch log group
16. stderr/stdout paths
17. **Cost Optimization Techniques:**
18. Use spot instances
19. Enable auto-termination
20. Use EMR Serverless

Advanced + Scenario-Based

1. **What is EMR Serverless?**\ A managed option for Spark/Hive without provisioning clusters; pay per use.
2. **Incremental Load Strategy in EMR:**
    - Maintain watermark in metadata
    - Use filters based on timestamp or partition
3. **How to Handle Memory Errors in Spark on EMR?**
4. Tune executor memory
5. Adjust number of executors
6. Enable dynamic allocation
7. **Auto-Scaling in EMR:**
8. Configure scaling policies based on CloudWatch metrics like CPU usage or YARN memory
9. **Data Format Best Practices:**
10. Use columnar formats like Parquet/ORC
11. Use compression (Snappy, Gzip)

Part 3: EMR vs Glue vs EMR Serverless

Feature	EMR on EC2	EMR Serverless	AWS Glue
Cluster Management	Manual	Fully Managed	Fully Managed
Language Support	Spark, Hive, Presto etc.	Spark, Hive	PySpark, Python
Cost Model	Per EC2 instance	Pay-per-use	Pay-per-job
Use Case	Heavy processing	On-demand big data tasks	ETL, light-to-medium ETL
Storage	HDFS, EMRFS (S3)	S3	S3

Bonus Tips for Interview:

- Be ready to explain a **real-world pipeline** you built using Glue or EMR.
- Prepare for questions like:
- *How do you handle failures in production ETL jobs?*
- *Explain the architecture of a data pipeline using AWS services.*
- *Have you used CI/CD with Glue/EMR?*

Let me know if you'd like this exported to PDF or want mock interview Q&A practice.