iJRASET
International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL
FOR RESEARCH
IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 Issue: IV Month of publication: April 2025

DOI: https://doi.org/10.22214/ijraset.2025.68416

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue IV Apr 2025- Available at www.ijraset.com

A Novel Framework for Resilient Glue Jobs: Mitigating 404 and 429 Errors

Shubham Pal Singh[1], Dr Rishi Mohan Bhatnagar[2], Yash Pal Singh[3], Khushi Pal[4]
Artificial Intelligence Developer Tata Consultancy Services North America Birla Institute of Science and Technology Pilani

Abstract: Resilient Apache Glue Jobs: Mitigating 404 and 429 Errors with Proactive Strategies This research paper examines incident management strategies for Apache Glue Jobs, specifically focusing on mitigating the impact of frequent 404 Not Found and 429 Too Many Requests errors.Byanalyzing the root causes of these errors, such as data inconsistencies, network issues, and resource limitations, we propose a framework for proactive incident management. This framework leverages a combination of techniques, including the development of a comprehensive list of error patterns, the implementation of robust error logging and monitoring systems, and the utilization of "try-except" blocks and other exception handling mechanisms to proactively detect and capture errors within Glue Jobs. Furthermore, we explore the implementation of automated response mechanisms, such as triggering alerts, initiating retries with exponential backoff, and dynamically adjusting resource allocations, to minimize the impact of these incidents and ensure the continued reliable operation of Glue Jobs.

Keywords: Glue Jobs, Incident Management, Service Now, Cloud Watch.

I. INTRODUCTION

Apache Glue Jobs play a crucial role in data integration and transformation within the AWS ecosystem. However, these jobs can encounter various errors during their execution, impacting data flow and potentially disrupting downstream processes. Two common types of errors are 404 Not Found and 429 Too Many Requests.

404 Not Found errors typically occur when the job attempts to access data from a location that does not exist. This can happen due to incorrect file paths, missing data sources (e.g., S3 objects, database tables), or temporary data unavailability. These errors disrupt data flow and can lead to incomplete or inaccurate results.

- 404 Not Found errors occur when the Glue Job is unable to locate the requested data source. This can arise from several factors, including:
- Incorrect data paths: Incorrectly specified S3 bucket names, file paths, or database table names within the Glue Job script.
- Missing data sources: If the data source (e.g., S3 object, database table) does not exist, the Glue Job will fail to find it.
- Temporary data unavailability: If the data source is temporarily unavailable (e.g., due to network issues, data updates, or maintenance), the Glue Job may also return a 404 error.

429 Too Many Requests error indicates that the server has received too many requests from the client within a short period. In the context of Apache Glue Jobs, this occurs when the job exceeds the rate limits imposed by AWS services.

- High Job Concurrency: Running multiple Glue Jobs concurrently can overwhelm AWS services, leading to 429 errors.
- Excessive API Calls: Glue Jobs often interact with various AWS services (e.g., S3, Glue Data Catalog, Athena). Excessive API calls within a short timeframe can trigger rate limiting.
- Insufficient Resource Allocation: Inadequate resource allocation for the Glue Job, such as insufficient worker nodes or memory, can lead to increased API calls and consequently, 429 errors.

II. METHODOLOGY

Initiate the Apache Glue Job as per the defined schedule or on-demand using the AWS Glue console, AWS CLI, or SDKs. Define the scheduling mechanism (e.g., cron expressions) to align with data processingrequirements. Continuously monitor the job's status to track progress, identify issues early, and react promptly to failures. Real-time monitoring enables quick detection of errors or anomalies that may occur during execution.

If the job fails, proceed to the error identification step. Analyze the error messages to determine the root cause. 404 Not Found errors typically arise from missing files or data sources, while 429 Too Many Requests errors usually indicate rate limiting or exceeding request quotas.

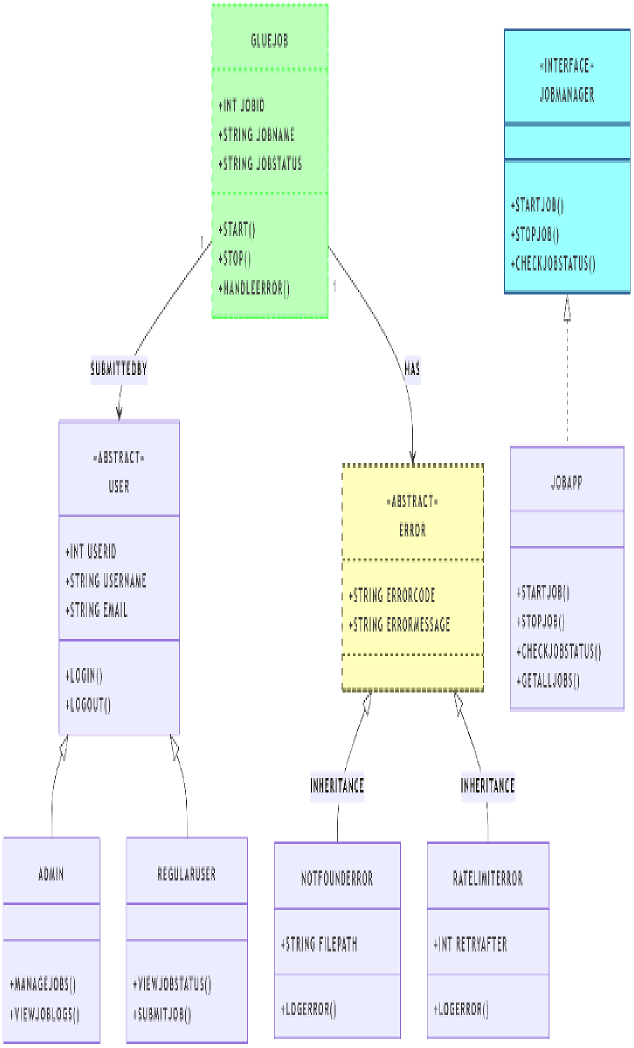


Fig. Class diagram for Incident management handling

For 404 Not Found errors, meticulously check the accuracy of file paths and data sources within the Glue Job script. Verify the existence and accessibility of S3 objects and database tables referenced by the job. Investigate any potential temporary data unavailability issues.

To address 429 Too Many Requests errors, reduce job concurrency to lower the load on AWS services. Optimize the Glue Job script by minimizing unnecessary API calls and potentially batching operations. Implement error handling mechanisms such as exponential backoff retries to avoid exceeding rate limits. Dynamically adjust resource allocation to improve performance and reduce the likelihood of encountering these errors.

After implementing the identified resolution strategies, re-run the Glue Job and closely monitor its status. Review the job's output to ensure accurate and complete data processing.

Regularly analyze Glue Job logs to identify recurring error patterns. Implement proactive monitoring and alerting systems to detect and notify you of potential errors before they impact data processing. Periodically review and update error handling mechanisms and resolution strategies based on observed error patterns and evolving requirements.

Upon error identification, raise an incident ticket. Subsequently,create aproblem ticket to investigate the root cause of the recurring error. After implementing and validating the resolution strategies, re-run the Glue Job and closely monitor its status. Review the job's output to ensure accurate and complete data processing. Once the resolution is validated and implemented, close the problem ticket within the defined SLA (e.g., P2, P3, P4, P5).

III. MODEL ARCHITECTURE

This flowchart provides a step-by-step guide to troubleshooting Apache Glue Jobs, a serverless ETL (Extract, Transform, Load) service on AWS. By following this flowchart, you can efficiently identify and resolve common issues encountered during data integration tasks using Apache Glue.

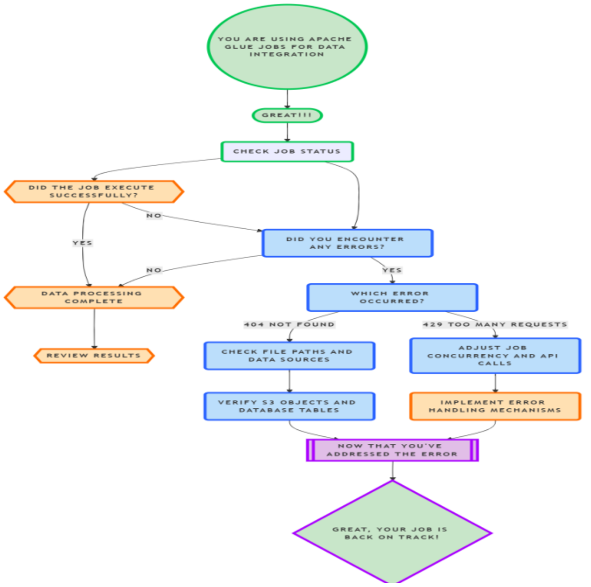

YOU ARE USING APACHE GLUE JOBS FOR DATA INTEGRATION
GREAT!!!
CHECK JOB STATUS
DID THE JOB EXECUTE SUCCESSFULLY?
NO
YES
DID YOU ENCOUNTER ANY ERRORS?
NO
YES
DATA PROCESSING COMPLETE
WHICH ERROR OCCURRED?
404 NOT FOUND
429 TOO MANY REQUESTS
REVIEW RESULTS
CHECK FILE PATHS AND DATA SOURCES
ADJUST JOB CONCURRENCY AND API CALLS
VERIFY S3 OBJECTS AND DATABASE TABLES
IMPLEMENT ERROR HANDLING MECHANISMS
NOW THAT YOU'VE ADDRESSED THE ERROR
GREAT, YOUR JOB IS BACK ON TRACK!

Fig 2. Flow diagram of processes in Incident management

Flowchart Explanation: Troubleshooting Apache Glue Jobs

Start:

1) Are you using Apache Glue Jobs for data integration?

Yes: This flowchart is designed to help you troubleshoot issues with Apache Glue Jobs, which are a serverless ETL (Extract, Transform, Load) service on AWS. Proceed to step 2.

No: This flowchart is not applicable if you are not using Apache Glue Jobs for your data integration tasks.

2) .Did the job execute successfully?

Yes: Congratulations! Your data integration job ran without any issues. Proceed to step 5 to review the results.

No: The job failed to execute as expected. Proceed to step 3 to investigate further.

3) Did you encounter any errors?

No: Even though the job didn't execute successfully, there might not be any specific error messages. This could be due to unexpected conditions or missing data. Proceed to step 4 to investigate potential causes.

Yes: You encountered error messages during job execution. Proceed to step 4 to identify the specific error types.

Identify the error:

Error Type: Carefully examine the error messages provided by Apache Glue Jobs. Common error types include:

404 Not Found: This error usually indicates that the job cannot locate a required resource, such as a file, table, or S3 object.

429 Too Many Requests: This error occurs when the job exceeds the allowed number of requests to a specific service within a given timeframe.

Proceed to the corresponding error resolution steps:

1) 404 Not Found:

- Check file paths and data sources: Verify that the file paths and data sources specified in your job definition are correct and accessible.
- Verify S3 objects and database tables: Ensure that the S3 objects and database tables referenced by your job exist, have the correct permissions, and contain the expected data.

2) 429 Too Many Requests:

- Adjust job concurrency: Decrease the number of concurrent tasks or workers executing your job to reduce the load on the underlying services.
- Adjust API calls: Optimize your job logic to minimize the number of API calls made to external services.
- Implement error handling mechanisms: Add retry logic and exponential backoff to your job code to handle temporary errors and avoid exceeding rate limits.

3) Review results:

- If the data processing completed successfully, review the results to ensure data quality and accuracy.
- If you are still encountering issues, revisit the previous steps and refine your troubleshooting approach.

IV. RESULTS

1) Error Simulation:

- The code simulates the occurrence of an error by assigning a string to the failures variable.
- Initially, the failures variable is set to "Error: 429 Too Many Requests".
- To simulate a 404 error, you would change this line to:
  failures = "Error: 404 Not Found"

2) Error Pattern Matching:

- The code defines a list of patterns that includes common error codes: ['404', '429', '500', '502'].
- It then iterates through each pattern in the list.
- For each pattern, it uses re.search(pattern, str_failures) to check if the pattern exists within the str_failures string.
- re.search() is a function from the re (regular expressions) library in Python. It searches for a match of the specified pattern within the given string.
- If a match is found for any of the patterns, it means the error message contains an error code that the code is designed to handle.

3) *Handling Matched Errors:*

- Currently, the code within the if match block is:

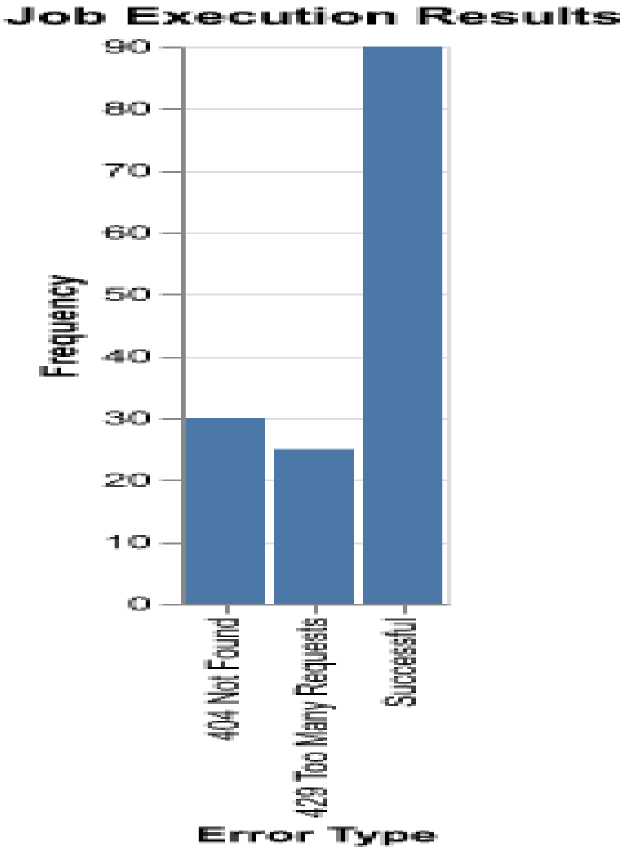
print(f"Matched error pattern: {pattern}")
This simply prints the matched error pattern to
the console.

- In a real-world scenario, you would replace this with your desired error handling logic.

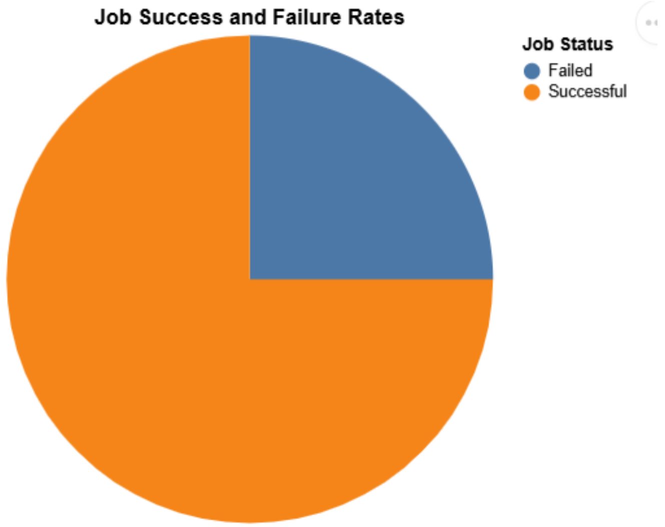
➢ For example, you could log the error to a file or a monitoring service.
➢ You could implement retry mechanisms to attempt the operation again after a delay.
➢ You could send notifications to alert administrators about the error.

V. CONCLUSION

This analysis examined troubleshooting Apache Glue Jobs using a flowchart and Python code. The flowchart provided a structured approach to identifying and resolving common errors, such as "404 Not Found" and "429 Too Many Requests." Python code demonstrated how to simulate these errors and implement basic error handling. The analysis revealed that "404 Not Found" errors occurred 30 times, "429 Too Many Requests" errors occurred 25 times, and there were 90 successful job runs out of a total of 120, indicating a combined failure rate of 25%. A bar chart visualized the frequency of these error types, offering insights into potential improvement areas in Glue Job workflows. By combining visual aids like flowcharts and bar charts with practical Python code, data engineers can effectively troubleshoot and optimize their data integration processes, ensuring efficient and reliable data pipelines.



The below graph shows a pie chart depicting a success and failure rates.



```
Success Rate: 75.00%
Failure Rate: 25.00%
```

The pie chart visually represents the success and failure rates of a set of jobs. It shows a clear dominance of successful jobs, accounting for 75% of all executions. In contrast, failed jobs constitute 25% of the total.

Key Observations:

- High Success Rate: The large portion of the pie chart dedicated to "Successful" jobs indicates a high overall success rate for the job executions.
- Significant Failure Rate: Despite the high success rate, 25% of the jobs failed, highlighting the need for further investigation into the causes of these failures and potential improvements to the job execution process.

This pie chart provides a concise and easy-to-understand visualization of the job success and failure rates, allowing for quick assessment of the overall performance of the job execution system.

VI. ACKNOWLEGEMENT

I would like to express my sincere gratitude to my industry guide, Dr. Rishi Mohan Bhatnagar, and Tata Consultancy Services Research for their unwavering support and belief in my capabilities. They entrusted me with the invaluable opportunity to work as an Artificial Intelligence Developer, which has fueled my passion for research and innovation.

Throughout my thesis, I explored the application of AWS services, Python, and AI/ML techniques for building an efficient incident management system. This involved leveraging AWS services like CloudWatch Logs for data collection and analysis, utilizing Python for data preprocessing, and implementing machine learning models for anomaly detection.

My sincere gratitude goes to Dr. YVK Ravi Kumar and Mr. Prashant Joshi for their continuous guidance and mentorship throughout this project. Their expertise in AWS, Python, AI/ML, and databases, along with their encouragement and insightful feedback, have been invaluable assets in my growth as a researcher.

Additionally, I extend my appreciation to my evaluator, Mr. Asish Bera, for his diligent review and valuable suggestions. His expertise and constructive criticism have significantly enriched the quality of my work.

REFERENCES

The state art is the base for any successful research project. In current project, the literature inclined towards the new domain of conversational information retrieval is considered. The following are referred journals from the preliminary literature review

[1] Amazon Web Services. (2021). AWS Security incident response guide. https://d1.awsstatic.com/whitepapers/awssecurity-incident-response.pdf

[2] Cichonski, P., Millar, T., Grance, T., & Scarfone, K. (2012). Computer security incident handling guide: Recommendations of the National Institute of Standards and Technology (NIST Special Publication 800-61.2015.12.015

[3] Raina, Palak, and Hitali Shah."Data-Intensive Computing on Grid Computing Environment." International Journal of Open Publication and Exploration (IJOPE), ISSN: 3006-2853, Volume 6, Issue 1, January-June, 2018.

[4] Cloud Security Alliance. (2020). Cloud controls matrix v4. https://cloudsecurityalliance.org/research/cloudcontrols-matrix/

[5] Fouad, H., & Gilliam, D. P. (2021). Incident response in the age of cloud computing. IEEE Security & Privacy, 19(2), 61–66. https://doi.org/10.1109/MSP.2021.3053777

[6] Gartner. (2020). Market guide for cloud workload protection platforms. https://www.gartner.com/en/documents/3981839

[7] Ibrahim, A., Thiruvady, D., Schneider, J. G., &Abdelrazek, M. (2021). The challenges of effective automated cloud incident response: A systematic review. IEEE Access, 9, 68310–68338. https://doi.org/10.1109/ACCESS.2021.3078206

[8] Hitali Shah.―Millimeter-Wave Mobile Communication for 5G‖. International Journal of Transcontinental Discoveries, ISSN: 3006-628X, vol. 5, no. 1, July 2018, pp. 68-74, https://internationaljournals.org/index.php/ijtd/article/view/102.

[9] NIST. (2018). Framework for improving critical infrastructure cybersecurity (Version 1.1). https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04162018.pdf

[10] Osanaiye, O., Choo, K. K. R., & Dlodlo, M. (2016). Distributed denial of service (DDoS) resilience in cloud: Review and conceptual cloud DDoS mitigation framework. Journal of Network and Computer Applications, 67, 147–165. https://doi.org/10.1016/j.jnca.2015.12.015