International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 06 | June - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

Actimize: A Batch Job Real Time Dashboard

Rohan K H
Fidelity Investments

Shiwansh Bhargav
JP Morgan Chase & Co.

Dr. Praveena T
Assistant Professor, RVCE, Bengaluru

Prof. Ganashree K . C
Assistant Professor, RVCE, Bengaluru

***Abstract*—Our Compliance partners do not have clear visibility into real time status of day to day surveillance and supervision alert batch jobs. They rely on manual communication to get notified in case of job delays or failures. To avoid any regulatory implications,it is crucial that alerts are generated on time so that they can act on a timely basis to prevent SLA breaches that may lead to further delays. So,it is important that they remain updated on the daily alerting job runs so that they can plan their work according.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

Recognizing the above challenge, there is an opportunity to implement a comprehensive dashboard solution equipped with alerting capabilties. Introducing such a system would empower partners to monitor alert job statuses in real time,enabling them to proactively address any issues that arise and take timely actions to prevent SLA breaches.

II. Objectives

- Implement a real-time monitoring system for up-to-date job status information.
- Provide clear and intuitive visualization of job statuses through the dashboard interface.
- Offer customization options and flexibility in the dashboard solution to meet varying user needs.
- Implement robust alerting mechanisms to notify associates of deviations from expected job statuses or SLA thresholds.
- Provide comprehensive and detailed job descriptions within the dashboard interface for enhanced job visibility and decision-making.

III. Implementation

Actimize is a suite of products developed by NICE Actimize, a division of NICE Ltd., which specializes in providing financial crime, risk, and compliance solutions for financial institutions and other industries. The Actimize suite is known for its comprehensive range of solutions designed to address various aspects of financial crime and compliance management.

A. Hardware and Software Requirements

Development and Test Environment :

- Hardware Specification
- OS: Windows
- Tool: Control M, Power BI 2.129.905.0, Power Apps
- Dataset: Compliance Business Unit Dataset

For the current phase the testing was done with the batch jobs of Compliance BU. In the further phases, the solution will expanded across the various Bus.

B. Solution Overview

The proposed solution is a comprehensive real-time dashboard with the following features:

- Manual and Auto Refresh: Updates the dashboard manually or automatically eight times a day.
- Filtering Capabilities: Filters jobs by job name, model name, job order date, and job status.
- Color-Coded Statuses: Uses distinct color codes for eight different job statuses.
- Dashboard Views: Provides two dashboard options—one displaying data for the latest order date and another showing historical data for the past month.

C. Methodology

This project employs a structured approach to developing the ”Actimize Batch Job Real Time Dashboard,” addressing the need for real-time visibility into batch job statuses for compliance partners. The methodology is organized into six primary stages: loading job details from ControlM to PostgreSQL, applying data transformations, designing the dashboard using PowerBI, embedding the dashboard onto a website using PowerApps, configuring both manual and scheduled refresh options, and implementing security measures to restrict access based on Business Units (BUs).

The initial stage involves extracting job details from ControlM and loading them into a PostgreSQL database. This is accomplished by using ControlM’s RESTful API. A Python script is employed to authenticate with ControlM using provided credentials, thereby obtaining an access token. With this token, the script sends a GET request to the ControlM API endpoint to fetch job statuses. The retrieved job details, typically in JSON format, are then processed and loaded into PostgreSQL. The script connects to the PostgreSQL database using the psycopg2 library, and job details are inserted into the jobs table. To handle potential duplicate entries, the insertion process includes an ON CONFLICT clause, ensuring that existing job records are

updated appropriately.

Once the data is loaded into PostgreSQL, various transformations are applied to ensure consistency and correctness. Dates are standardized by converting date strings into timestamp formats, and naming conventions for job names and model names are normalized. Additionally, indexes are created on key columns to optimize query performance, facilitating faster and more efficient data retrieval for subsequent steps.

The next stage involves designing the dashboard using PowerBI. PowerBI is chosen for its robust data visualization and business intelligence capabilities. Data models are defined within PowerBI, establishing relationships between tables and creating calculated columns and measures using Data Analysis Expressions (DAX). The dashboard includes a variety of visualizations such as charts, tables, and slicers, which provide users with insights into job statuses. Conditional formatting is applied to these visualizations, using color-coding to highlight different job statuses for quick and intuitive identification of issues.

To make the dashboard accessible, it is embedded into a web application using PowerApps. PowerApps enables seamless integration of the PowerBI dashboard, providing a user-friendly interface that can be customized to match the organization's branding and specific user requirements. This integration ensures that compliance partners can easily access the real-time dashboard from within their existing workflow environments.

Ensuring the dashboard data is current, both manual and scheduled refresh options are configured. A manual refresh button is implemented using Power Automate, allowing users to trigger data updates as needed. Additionally, a scheduled refresh is set up in PowerBI Service to automatically update the data up to eight times a day. This scheduled refresh configuration is managed through the PowerBI Service interface, where refresh schedules are defined to ensure the dashboard consistently displays the most recent job data.

Finally, security measures are implemented to restrict data access based on Business Units (BUs). Active Directory (AD) groups are defined for each BU, and Row-Level Security (RLS) is configured in PowerBI to enforce these restrictions. RLS involves creating roles in PowerBI Desktop and defining DAX filters that limit data visibility based on user roles. These roles are then published along with the dataset, and security roles are assigned to the corresponding AD groups in PowerBI Service. This ensures that users can only access data pertinent to their BU, maintaining data confidentiality and compliance with organizational policies.

By following this detailed methodology, the project effectively addresses the need for real-time monitoring of batch job statuses, providing compliance partners with the tools they need to proactively manage alerts and prevent SLA breaches.

D. Testing

Testing for the "Actimize Batch Job Real Time Dashboard" project is critical to ensure the reliability, accuracy, and security of the system. The testing process is comprehensive, covering each aspect of the methodology: data extraction and loading, data transformations, dashboard design, integration and embedding, refresh mechanisms, and security measures. Detailed testing ensures that each component functions as intended and integrates seamlessly with the others.

For the data extraction and loading stage, testing involves validating the Python script that interacts with the ControlM API. This includes unit tests to verify that the script can successfully authenticate with ControlM, handle various API responses, and accurately fetch job details. Tests are also conducted to ensure that the data is correctly formatted and inserted into the PostgreSQL database. This involves checking for data integrity, ensuring that all necessary fields are populated, and verifying that the ON CONFLICT clause effectively handles duplicate entries by updating existing records.

In the data transformation stage, rigorous testing is conducted to confirm that all transformations are correctly applied. This includes verifying that date formats are accurately converted to standardized timestamp formats and ensuring that naming conventions for job names and model names are consistently applied. Indexes created on key columns are tested to ensure they improve query performance without compromising data integrity. This involves performance benchmarking to compare query speeds before and after indexing.

The dashboard design in PowerBI undergoes extensive usability and functionality testing. Each visualization component is tested to ensure it correctly reflects the underlying data. This includes validating that charts, tables, and slicers accurately display job statuses and that conditional formatting is applied correctly. Usability testing involves gathering feedback from end-users to ensure the dashboard is intuitive and meets their needs. This also includes testing calculated columns and measures created using DAX to ensure they produce accurate and meaningful insights.

For the integration and embedding of the dashboard using PowerApps, testing focuses on the seamless interaction between PowerBI and PowerApps. This involves ensuring that the embedded dashboard is fully functional within the web application and that all interactive elements, such as filters and slicers, work correctly. Customization and branding elements are tested to confirm they align with organizational standards and user expectations.

The refresh mechanisms, both manual and scheduled, are tested to ensure they reliably update the dashboard with the latest data. Manual refresh functionality implemented via Power Automate is tested by triggering manual refreshes and verifying that data updates occur as expected. Scheduled refresh settings in PowerBI Service are tested by monitoring the dashboard to ensure it automatically updates at the predefined intervals. This includes testing the frequency and timing of updates to confirm they align with the specified schedule.

Security measures are subjected to thorough testing to ensure that data access is appropriately restricted based on Business Units (BUs). This involves verifying that Active Directory (AD) groups are correctly defined and that Row-Level Security (RLS) in PowerBI accurately enforces these restrictions. Testing includes attempting to access data across different BUs to ensure that users can only view data pertinent to their own BU. Additionally, security testing involves checking for potential vulnerabilities in the authentication and data access processes to safeguard against unauthorized access.

Overall, a detailed and methodical testing approach is employed to validate each component of the ”Actimize: A Batch Job Real Time Dashboard” project. This ensures that the system not only meets functional requirements but also delivers a reliable, secure, and user-friendly solution for real-time monitoring of batch job statuses.

IV. Results and Discussion

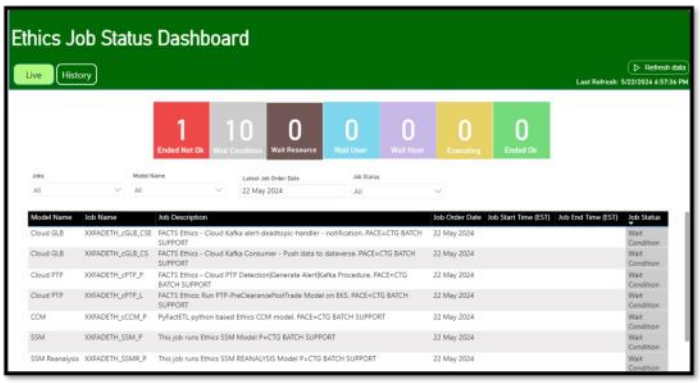

Fig. 1. Dashboard

The figure illustrates the comprehensive real-time dashboard designed using PowerBI for monitoring the status of day- to-day surveillance and supervision alert batch jobs. The dashboard features multiple interactive components, including manual and automatic refresh options, and filters for job name, model name, job order date, and job status. It displays both the latest order date data and historical data for the past month. Each job status is color-coded to provide immediate visual feedback, enabling compliance partners to quickly identify and address any issues, thereby preventing SLA breaches and ensuring timely actions.

Fig. 2. Data Extraction from Database

The figure depicts the flow diagram detailing the process of extracting job data from the ControlM system and loading it into the PostgreSQL database. The process begins with the Python script authenticating with the ControlM API to retrieve job details. The retrieved data is then processed and transformed into the required format before being inserted into the PostgreSQL database. This flow includes key steps such as authentication, data retrieval, data transformation (including date formatting and normalization), and data insertion. The diagram also highlights the integration points and the use of the psycopg2 library for database interaction, ensuring accurate and efficient data loading for real-time monitoring.

V. Conclusion And Future Works

Introducing a comprehensive dashboard solution equipped with alerting capabilities would empower the compliance partners to monitors alert job statuses in real time enabling them to address any issues that arise and take timely actions to prevent SLA breaches.

- The dashboard is to be embedded onto the Fidelity website through Power Apps.
- The security of the dashboard is to be enhanced by ensuring that the associates of a certain BU only have the access to the jobs belonging to their respective BU.

References

[1] Cherradi, Mohamed. ”Data Lakehouse: Next Generation Information System.” In Seminars in Medical Writing and Education, vol. 3, pp. 67-67. 2024

[2] Harby, Ahmed A., and Farhana Zulkernine. ”From data warehouse to lakehouse: A comparative review.” In 2022 IEEE International Conference on Big Data (Big Data), pp. 389-395. IEEE, 2022.

[3] Ghosh, Debananda. ”OneLake and Lakehouses for Data Engineers.” In Mastering Microsoft Fabric: SAASification of Analytics, pp. 49-94. Berkeley, CA: Apress, 2024.

[4] Dolhopolov, Anton, Arnaud Castelltort, and Anne Laurent. ”Implementing Federated Governance in Data Mesh Architecture.” Future Internet 16, no. 4 (2024): 115.

[5] Schneider, Jan, Christoph Gröger, Arnold Lutsch, Holger Schwarz, and Bernhard Mitschang. ”The Lakehouse: State of the Art on Concepts and Technologies.” SN Computer Science 5, no. 5 (2024): 1-39.

[6] Cedeño, María José Vélez, and Hernán Vargas Nolivos. ”Methodology for the migration of NoSQL databases to SQL databases for their subsequent integration in servers set in relational data models.” In AIP Conference Proceedings, vol. 2994, no. 1. AIP Publishing, 2024.

[7] Cherradi, Mohamed. ”Data Lakehouse: Next Generation Information System.” In Seminars in Medical Writing and Education, vol. 3, pp. 67-67. 2024.
[8] Oreščanin, Dražen, and Tomislav Hlupić. ”Data lakehouse-a novel step in analytics architecture.” In 2021 44th International Convention on Information, Communication and Electronic Technology (MIPRO), pp. 1242-1246. IEEE, 2021.
[9] Vakharia, Suketu, Peng Li, Weiran Liu, and Sundaram Narayanan. ”Shared foundations: Modernizing meta’s data lakehouse.” In The Conference on Innovative Data Systems Research, CIDR. 2023.
[10] Harby, Ahmed A., and Farhana Zulkernine. ”From data warehouse to lakehouse: A comparative review.” In 2022 IEEE International Conference on Big Data (Big Data), pp. 389-395. IEEE, 2022.
[11] Park, Sun, Chan-Su Yang, and JongWon Kim. ”Design of Vessel Data Lakehouse with Big Data and AI Analysis Technology for Vessel Monitoring System.” Electronics 12, no. 8 (2023): 1943.
[12] Errami, Soukaina Ait, Hicham Hajji, Kenza Ait El Kadi, and Hassan Badir. ”Spatial big data architecture: from data warehouses and data lakes to the Lakehouse.” Journal of Parallel and Distributed Computing 176 (2023): 70-79.
[13] Armbrust, Michael, Ali Ghodsi, Reynold Xin, and Matei Zaharia. ”Lakehouse: a new generation of open platforms that unify data warehousing and advanced analytics.” In Proceedings of CIDR, vol. 8, p. 28. 2021.
[14] Begoli, Edmon, Ian Goethert, and Kathryn Knight. ”A lakehouse architecture for the management and analysis of heterogeneous data for biomedical research and mega-biobanks.” In 2021 IEEE International Conference on Big Data (Big Data), pp. 4643-4651. IEEE, 2021.
[15] Jin, Runyu, Paul Muench, Veera Deenadhayalan, and Brian Hatfield. ”AIOps Essential to unified resiliency management in data lakehouses.” In 2022 IEEE International Conference on Big Data (Big Data), pp. 4777-4781. IEEE, 2022.