Database Error Detection: Automatically Identifying Exceptions for Loan Servicing Software

CS4991 Capstone Report, 2024

Chase Candeloro
Computer Science
The University of Virginia
School of Engineering and Applied Science
Charlottesville, Virginia USA
chc3fw@virginia.edu

ABSTRACT

JP Morgan Chase & Co.'s LoanIQ subledger services over $300 billion of syndicated loans, yet errors in its database can prevent interest from accruing on these loans. To detect and report these errors, I developed a software system of SQL queries that collected information on problematic loans and displayed this information in a dynamically updating Tableau dashboard. I developed this system systematically by first classifying known error types reported by the LoanIQ software. Then, by rigorously studying the unique properties of each error type, I identified and documented the root cause and database location of each. From these specifications, I developed distinct queries to detect each error in the database and compiled them in an Alteryx workflow that periodically runs the queries and outputs the error-causing loans to a Tableau dashboard. As a result of this project, users of LoanIQ can more easily diagnose accrual exceptions and ensure that JP Morgan Chase & Co. earns interest on its portfolio of syndicated loans. To make this system more effective, further development should integrate the dashboard into the existing workflow tools of LoanIQ users. Additionally, new queries need to be developed as new error types are discovered.

1. INTRODUCTION

Often serving as the primary financing mechanism for large corporations, total syndicated loan commitments in the United States in 2022 amounted to $5.9 trillion. Syndicated loans are lending agreements whereby a group of creditors, typically large banks, form a collective ("syndicate") to lend to a single borrower at a quantity greater than any of them could individually. Because they involve immense sums of money concentrated in just one debtor, a single lender cannot realistically facilitate such loans, as a default by the borrower could prove catastrophic. Thus, creditors use the syndicate structure to pool their money and distribute the default risk among themselves. Moreover, once the syndicate has issued the loan, members may opt to sell portions of their commitment to other, outside lenders, further diluting the risk among the creditors.

However, this complex structure makes servicing syndicated loans more involved than other loan types. Accordingly, to accomplish proper servicing, one member of the syndicate acts as the "agent" on the loan, handling communications and payments between the various parties. Primarily, agenting a syndicated loan involves collecting interest payments from the borrower and proportionally paying them to the lenders on the basis of their commitment. Agenting a syndicated loan requires an extensive software system to track and manage the loan portfolio. As one of the largest banks in the

world, JP Morgan Chase & Co. frequently acts as agent on syndicated loans, using its LoanIQ subledger system to perform servicing on them.

2. RELATED WORKS

As JP Morgan has utilized LoanIQ and its database for many years, the system has developed deficiencies consistent with those of other large legacy software systems. Srinivas, et. al. (2016) surveyed and analyzed the common challenges associated with such legacy systems. They asserted that maintaining these systems proves costly and time consuming due to improper documentation and loss of knowledge over time. Furthermore, they listed another major challenge as the difficulty integrating legacy systems with others because of their lack of usable interfaces. My project encountered these difficulties frequently and I adapted its design to account for them.

Similarly, Comella-Dorda, et. al. (2000) surveyed various modernization approaches for legacy systems, noting the strengths and weaknesses of each. Particularly, they discussed "white-box modernization" which my project utilized when attempting to add additional functionality to LoanIQ. This approach requires reverse engineering the legacy system to understand its internal functioning. As a result of the built-up complexity of legacy systems, they identify acquiring this understanding as a risky and laborious task. Additionally, they contrast the more complex white-box approach with the simpler "black-box modernization," which requires only an understanding of the interfaces of the system. Yet, the black-box approach is often infeasible due to deficiencies in the system's interfaces, as was the case with my project.

3. PROJECT DESIGN

To service its portfolio of syndicated loans, JP Morgan Chase & Co. uses a system based on the LoanIQ subledger software developed by financial software company Finastra. This project aims to simplify the process of identifying and resolving exceptions occurring within the interest accrual functionality of that system.

3.1 Review of Existing System

Though the core LoanIQ software itself is licensed from a third-party, JP Morgan has internally developed numerous add-ons, tools, and interfaces that add functionality to the system. The system performs its primary purpose of servicing syndicated loans by storing key data for each loan in its database and executing necessary updates in accordance with the lending agreement over the loan term or in response to manual user prompting. For accounting purposes, JP Morgan reports interest income from these loans daily. As such, the system's interest accrual procedure runs on a scheduled nightly basis, along with many other procedures, in a process collectively known as the batch. The interest accrual procedure operates by iterating through each loan in the database, computing the amount of interest accrued on it, and generating an accounting entry reporting this interest accrual as income. These entries are then fed into many of the bank's other systems for risk management, compliance, and accounting.

However, for a variety of reasons, the procedure may fail on a given loan, in which case interest is not accrued on it and the procedure generates an *Interest Accrual Exception* that provides the details of the loan and attempts to describe the cause of the failure before progressing to the next loan. As the batch runs, it adds generated exceptions to a document called the "Accrual Exceptions Report," a new version of which is generated with each nightly batch process. JP Morgan's LoanIQ users are responsible for resolving these exceptions and ensuring that interest is accrued correctly.

3.2 Problems with Exception Reporting

In theory, LoanIQ users would use the Accrual Exceptions Report generated by the system to identify problematic loans and the issues with them. However, the accessibility, formatting, and level of detail provided by the report limits its usability in practice. To access the report from the LoanIQ home view, users must navigate through several menus and dropdown lists, obfuscating it. Once accessed, the report is formatted as a comma-separated value file, which simply contains plaintext data entries separated by commas. Yet, many of these data entries are empty values, which occupy space in the report. Typically spanning across three entries each, exceptions appear as plaintext strings but the data entries they occupy are often non-contiguous, making it unclear to a user where one begins and another ends.

Moreover, due to the way the system generates the report, the rows for different exceptions are often interwoven together, further convoluting them. In addition, the report has a hard display limit of 10,000 values, many of which are empty, so any exceptions contained beyond those cannot be viewed until others are resolved. The exceptions themselves vary greatly in specificity regarding the cause of failure, with many providing minimal indication of the location of any invalid data. Accordingly, users avoid the Accrual Exceptions Report whenever possible, despite its potential utility.

3.3 Project Objectives

As a summer analyst at JP Morgan, I designed and implemented a project to develop an alternative report to the Accrual Exceptions Report which could augment and streamline the workflow of LoanIQ users. As such, the new report needed to provide a comprehensive list of the exceptions with sufficient information to resolve them. Furthermore, as the system could potentially encounter new exceptions with each batch process, the report needed to be updated periodically. Finally, for users to adopt the new report into their workflow, its design and implementation needed to reflect their existing tools and be accessible through common hubs.

3.4 Project Execution

With the existing Accrual Exception Report largely unusable, rather than reformat it, I opted to construct an entirely separate report from scratch, using the original report only as a guide.

3.4.1 Gathering Exception Causes

Including sufficient information for the resolution of an exception necessitated knowing what specific behaviors or data values caused the exception. Because of the legacy status of the LoanIQ system, I had minimal access to documentation that could elaborate beyond the information provided by the exceptions themselves. Accordingly, to gather a comprehensive list of exception types, I collected a sampling of Accrual Exception Reports generated over the course of several weeks and documented each unique exception type that appeared. I then divided the exception types into “data errors,” which were caused by invalid values in the LoanIQ database and “user errors,” which were caused by user actions. Since the irregularity in user errors made them unfit for automatic reporting, I focused the report on data errors. With these classifications, I analyzed each data error exception type, attempting to identify the specific fields in the database that generated them. This identification process involved writing SQL queries to extract the data of exception-causing loans and comparing it to non-exception-causing data, documenting differing fields that could be the source of the error.

3.4.2 Generalizing Queries for Detection

Once a specific source had been recorded for each exception type, I revised the SQL queries that I had used to discover them to detect the occurrence of any such source across all loans in the database. Further, I augmented the queries to return additional information about the exception source. I tested these generalized queries against the existing Accrual Exception Report, validating whether every exception of a specific type that appeared in the existing report was detected by the query. After testing them, I constructed a workflow in the data analytics software Alteryx that executed each query on a weekly basis and collected the results into a single dataset.

3.4.3 Report Formatting

Having been a part of their existing workflow, LoanIQ users were familiar with dashboards created using the data visualization software Tableau, so I chose to display the exception information returned by the queries in a Tableau dashboard. This decision was further supported by the robust Tableau integration within Alteryx as well as in the internal JP Morgan website for LoanIQ users. I opted to format the dashboard with tabs along the top for each exception type with each tab view containing a list of loans that exhibit exception behavior and the associated error-causing data for each loan.

4. ANTICIPATED RESULTS

Though I completed designing and populating the dashboard, upon the conclusion of my time at JP Morgan the new exception reporting system still needed to be integrated into the user workflow. This integration would involve adding a page displaying the dashboard to the internal website for LoanIQ users as well as surveying users for potential improvements.

Once integration had been completed, LoanIQ users would be able to view the dashboard within their standard workflow. As such, users could frequently check for failures to accrue interest on loans and instantly have guidance as to how to resolve the issue. Previously, a failure to accrue interest on a loan may go unnoticed and unresolved until the failure had compounded into a significant issue. With the new reporting system, these exceptions would be identified and prominently displayed as soon as they occur, thereby preventing accrual problems from worsening. As each of these failures represents a possible preventable loss in income for the bank, the ability to rectify them as they appear provides significant value and alleviates any further effort required to fix them.

5. CONCLUSION

JP Morgan's LoanIQ system services over $1,500,000,000,000 in assets with over $300 billion of those assets being syndicated loans. Any failure of the system represents a preventable loss of assets that the bank already holds, an intolerable result for any organization, much less the largest and most systemically important bank in the world. As such, JP Morgan heavily depends on reliability of the LoanIQ system.

This project sought to reformat and streamline the process for rectifying interest accrual exceptions within LoanIQ. By providing LoanIQ users with a convenient display of current exceptions and their causes, the project should reduce the severity and prevalence of these errors, acting as a safety guard against avoidable losses for the bank. Overall, these efforts should serve to protect JP Morgan's assets and support LoanIQ users.

6. FUTURE WORK

As mentioned earlier, at the time of my departure from JP Morgan, the dashboard still needed to be integrated into the LoanIQ user interface. Furthermore, integration testing and

interaction with the users themselves was necessary to assess the usability of the system and potential points of improvement.

7. ACKNOWLEDGMENTS

First and foremost, I'd like to thank Jaquel Saunders, the leader of the LoanIQ Product Owner team at JP Morgan Chase & Co. for her help in introducing me to the system and its intricacies as well as her constant support of my efforts with the interest accrual report project and beyond.

I'd also like to thank Ryan Ademski and Eric Johnson for their assistance and guidance in navigating the seemingly infinitely complex system that is LoanIQ. Additionally, I'd like to thank them for their constant availability and willingness to help with any question I had as well as their enthusiasm for mentoring Parth and I.

Finally, I want to thank my fellow intern Parth Patel for his friendship and companionship throughout my time at the bank.

REFERENCES

Comella-Dorda, S., Wallnau, K., Seacord, R. C., & Robert, J. (2000). A survey of legacy system modernization approaches. CMU/SEI, 18.

Srinivas, M., Ramakrishna, G., Rao, K. R., & Babu, E. S. (2016). Analysis of legacy system in software application development: A comparative survey. International Journal of Electrical and Computer Engineering, 6(1), 292.