### Introduction

Jenny Lieu recently joined Bigeye as a solutions engineer. She previously worked in various SRE and data engineering capacities in consulting and healthcare, modernizing data stacks for clients and ensuring the stability of infrastructure. In this blog, she talks to us about how a tool like Bigeye would have helped her in her previous job at Delta Dental of Minnesota.

### Jenny at Delta Dental of Minnesota

Just prior to joining Bigeye, Jenny Lieu was a senior data engineer at Delta Dental of Minnesota. Delta Dental is a nationwide dental network that is organized in a federated fashion: each state within the company operates as a separate entity under the Delta Dental umbrella.

At Delta Dental of Minnesota, Jenny’s primary responsibilities included:

* monitoring the data pipeline to ensure that all incoming data was correct, up-to-date, and delivered on time to clients and stakeholders
* ensuring the stability of the data platform and infrastructure
* dealing with issues related to security, data governance, and data integrity.

### The data stack at Delta Dental

The data stack at Delta Dental consisted of:

* **Oracle database** for primary database
* **SQL Server** for an additional data warehouse
* **Pentaho** for ETL
* **Tableau** and **Power BI** for data visualization
* A cron job for scheduling that was migrated to **Chronicle**, a cron job scheduler with better visibility, towards the end of Jenny’s tenure.

### Type of data

The kinds of data going through the data pipeline included:

* new account data
* patient information
* patient billing information
* dental insurance information
* insurance coverage after patient responsibilities
* administrative details around how many times a bill was sent out to a patient and whether it had been paid on time.

The most important data pipeline jobs were running on financial data that executives would be reviewing on dashboards on a daily or weekly basis. The most important thing was that all the numbers matched/added up/made sense. Executives generally had ranges in mind when looking at the dashboards. For example, that a number should be at least 5% higher but less than 20% higher than the previous week’s numbers.

If the numbers were miscalculated or too low, it would be Jenny’s job to determine where along the pipeline the data was miscalculated.

One of the reasons why it was so important that all the numbers always added up was that sometimes patient information was included in the financial data, so it was crucial to ensure that the data was HIPAA-compliant and SOC 2-compliant at all times to avoid potential audits.

### Manual monitoring of the data pipeline

Jenny’s job involved a lot of manual monitoring of the pipeline - in particular running SQL queries and scripts to manually pull data to troubleshoot discrepancies.

Mondays were usually high alert days as incoming data was reconciled on Mondays. If the numbers were miscalculated or too low, Jenny would have to determine where along the pipeline the data was miscalculated. It could be an incorrectly coded script, or a new set of data that came in with an unknown schema.

Generally speaking, Jenny had visibility into the core set of data schemas that most of their reports used, but occasionally there were new data schemas that she wasn't familiar with. For instance, one of the actuary teams built their own specialized stored procedures and dashboards without any data engineers, and they would come to her to fix data issues even though she didn't have visibility into their pipeline.

In another example, one team dealt with only USG data, and although she could build new data schemas for wire reporting, she wasn't familiar with their specialized calculations.

### Data issues Jenny commonly encountered

**Vendor data issues**

The most common data issue that Jenny had to deal with was missing data from upstream data vendors. For example, one of their data vendors was the Michigan branch of Delta Dental. As the largest network in the Midwest, the Michigan branch was in charge of claims reconciliation for the entire region. The Minnesota branch would send all claims filed to the Michigan branch, and the latter would process them, and send the data back to Minnesota. Jenny’s team would then perform data transformations depending on their own reporting and dashboarding needs.

However, the Michigan branch would sometimes send the wrong files or files with corrupted data columns. For instance, a specific file that they sent every two weeks caused the entire pipeline to break down because they had changed the length of one of the column types without notifying Minnesota.

Another time, data that for five years, had been incoming as a number, suddenly contained letters, again breaking down the entire pipeline.

Jenny met with the Michigan team weekly and had a primary contact person for daily issues. They also used ServiceNow to open tickets whenever an issue arose. But issues were often dropped off the priority list.

**Duplicate data**

Duplicate data was another common problem, especially when it came to patient billing claims.

One claim might contain billing codes for multiple treatments. These billing codes might be entered incorrectly, leading to the need for manual edits weeks or months later. In the meantime, though, the original data had already moved downstream through the data pipeline.

In theory, the data pipeline should have checked the new, updated billing codes record against the original, and replace the record with the updated information, rather than adding a duplicate record. But since this deduping process relied on matching against certain columns, it was fairly brittle.

In one case, the check to dedupe data was based on five main columns. Then Michigan began sending an extra column with differentiating information that the stored procedure did not account for. As a result, there was duplicated data that went unnoticed for months.

### How Jenny’s team attempted to address the problem

Jenny and her team did attempt to perform more proactive data quality monitoring in “hack-ish” ways, such as checking vendor data file sizes to ensure that files were not empty. If the file sizes were below a certain threshold, it would send an email notification to alert Jenny to the issue. However, Delta Dental of Minnesota ultimately lacked the necessary resources to collaborate and build out more proactive technologies. The team was limited to just one data engineer and one DBA, and the lack of manpower meant that much of their time was spent on new development work, maintenance, and troubleshooting.

### How Bigeye would have helped Jenny

Given that so much of Jenny’s job involved manually running SQL queries and scripts to check for issues in the data pipeline, a data observability solution that automated these sorts of checks would have been enormously helpful. Jenny says, “A data observability solution that monitored incoming files and checked for irregularities in the data would have really helped us be more proactive in preventing pipeline breaks.”

In particular, Bigeye’s metadata metrics, which check the data’s freshness and volume, would be able to replace “hack-ish” heuristics like file sizes. And Bigeye’s sophisticated column-level checks would have alerted Jenny early to issues like changes in column type, duplicates, and formatting.