Introduction

The **Service Operations Report** was a critical monthly tool for evaluating key banking workflows, including **insurance handling, card issuance, and service efficiency**. However, its preparation was plagued by **manual data collection, fragmented sources, and inefficient SQL queries**, causing delays, errors, and unreliable insights. Originally, the report required an entire **week** to compile, resulting in outdated performance evaluations and delayed strategic decisions.

Recognizing the operational risks of these inefficiencies, I took the initiative to **automate and optimize** the report's generation. By streamlining data collection, restructuring SQL queries, and automating workflows, I reduced the report update time from a full **week to under 30 minutes**. This transformation saved **hundreds of analyst hours annually**, improved **data accuracy**, and enabled **timely, data-driven decision-making** at the managerial level.

This case study explores the challenges faced, my approach to automation, and the measurable impact of optimizing the Service Operations Report—enhancing both efficiency and strategic decision-making across the bank.

Problem & Context

The Original Challenge

The **Service Operations Report** was a **critical monthly report** that provided insights into key **banking processes**, such as:

- **Insurance handling**
- **Card issuance**
- **Service workflows**

However, its preparation was **highly inefficient** due to:

- **Manual processes**
- **Fragmented data sources**
- **Outdated tools**

These inefficiencies caused **delays, errors, and unreliable insights**, **negatively impacting** both **employee performance evaluations** and **strategic decision-making**.

Key Challenges

1. Manual Data Collection from Multiple Teams

- Different departments managed separate data sources (e.g., insurance, card issuance, process catalogs).
- Gathering data **relied on manual communication**, creating **bottlenecks** and **dependencies on specific analysts**.

2. Inefficient Excel-Based Workflows

- Data had to be **manually entered**, increasing the risk of human errors.
- Reports relied on **VLOOKUPs and COUNTIF formulas**, which **broke** when source files changed or contained **format inconsistencies** (e.g., spaces in numeric fields).
- **Structural changes in source files** could completely break the report, requiring hours of **troubleshooting**.

3. Data Integrity Issues & Errors

- **Frequent duplication of entries**, misformatted fields, and inconsistent data structures.
- Data structure changes from external teams **often caused formula failures**, forcing analysts to **manually track down errors across thousands of files**.

4. Slow SQL Processing & Scaling Problems

- SQL queries were **inefficient**, growing to **thousands of lines** and requiring **two hours to execute**.
- **No ongoing optimization**—queries became **patchwork fixes**, making them increasingly **complex and difficult to maintain**.

Risks & Consequences if Unresolved

1. Delayed & Unreliable Reports

- The report **was** due **by the 20th of each month**, but delays left management **without timely insights**.
- **Decisions** were **made using outdated or incomplete data.**

2. Inaccurate Employee Evaluations & Poor Decision-Making

- **30% of performance assessments** were based on **flawed or outdated data**, leading to unfair evaluations.
- **Employees** were **penalized for errors they didn't make**, affecting morale.
- Poor **forecasting** resulted in **misallocated resources and unrealistic targets**.

3. Increased Workload & Analyst Burnout

- Report **preparation** exceeded **40+ hours per month per analyst**, limiting time for deeper analysis.
- **High stress levels** due to urgent last-minute fixes and dependencies on external teams.

4. Loss of Trust in Data Accuracy

- Senior management **doubted the report's reliability**, frequently requesting manual verification.
- Strategic **decisions were delayed or based on unverified assumptions.**

5. Failed Attempts at Automation

- Prior Excel macro solutions were unreliable—**a single incorrect field could break the entire report**.
- SQL scripts became **too complex to maintain or scale efficiently**.

Why This Problem Mattered

1. Time

- Analysts spent **excessive hours** on **manual data entry, error-checking, and troubleshooting**.
- Instead of **focusing on insights**, valuable time was wasted on **repetitive, low-value tasks**.

2. Cost

- **Delays and inaccuracies in reporting** negatively affected **financial planning and operational efficiency**.
- **Inefficient report generation** cost the bank **thousands of wasted labor hours**.

3. Accuracy & Reliability

- Data inconsistencies **led to flawed insights**, creating a **ripple effect of poor decisions across departments**.
- A **lack of trust in reports** weakened the effectiveness of **business strategies and performance reviews**.

Conclusion

The inefficiencies in preparing the **Service Operations Report** had significant **operational, financial, and strategic consequences**. Without intervention, the bank would continue facing:
✔ **Unreliable data**

✔ **Poor decision-making**
✔ **Increased workloads for analysts**

The **need for automation and optimization** was **critical** to improving **accuracy, efficiency, and business impact**.

Approach & Tools

To address the inefficiencies in the Service Operations Report, I implemented a structured, multi-step approach focused on **data consolidation, process optimization, and automation**. The solution was designed to eliminate manual work, improve data accuracy, and significantly reduce report generation time.

Step 1: Data Consolidation

Before:

- Data was manually collected from five separate departments.
- Analysts relied on email chains and shared Excel files, causing version control issues.
- Reports contained inconsistencies due to varying data formats and human errors.

Action Taken:

- Centralized all data sources into a structured SQL database.
- Established **automated data pipelines** to pull information in real time.
- Standardized data formats to ensure consistency across reports.

After:

- Eliminated the need for manual data gathering.
- Improved data integrity and reduced inconsistencies.
- Ensured all teams worked with the same **single source of truth**.

Step 2: Query Optimization

Before:

- SQL queries were slow, taking **up to 2 hours** to execute.
- Complex, unoptimized scripts led to frequent failures and troubleshooting.
- Queries contained redundant joins and subqueries, making them difficult to maintain.

Action Taken:

- Reduced SQL execution time by **95% (from 2 hours to 5 minutes)**.
- Eliminated redundant calculations and optimized indexing strategies.
- Implemented **automated query performance monitoring** to detect inefficiencies.

After:

- SQL execution time reduced **from 2 hours to just 5 minutes**.
- Query failures dropped significantly, reducing maintenance efforts.
- Analysts could now **retrieve insights in near real-time**.

Step 3: Report Automation

Before:

- Reports were compiled manually in Excel, requiring extensive formatting.
- Frequent formula errors (e.g., broken VLOOKUPs) led to incorrect results.
- Managers often received outdated reports, delaying decision-making.

Action Taken:

- Built **Power BI dashboards** with real-time data visualization.
- Automated the report generation process, eliminating manual Excel work.
- Integrated dynamic filters and drill-down capabilities for deeper insights.

After:

- Reports **generated automatically** with real-time updates.
- Managers could interact with live dashboards instead of waiting for static reports.
- **Decreased report errors by 80%, improving decision-making confidence.**

Step 4: Process Standardization & Training

Before:

- Report generation depended on specific analysts familiar with manual processes.
- Lack of standardized procedures led to inconsistencies and frequent rework.
- New team members faced a steep learning curve.

Action Taken:

- Documented **standard operating procedures (SOPs)** for data handling and report generation.
- Conducted training sessions to familiarize analysts with SQL, Power BI, and automation workflows.
- Created a **self-service reporting model**, empowering managers to retrieve data independently.

After:

- Report creation became a structured, repeatable process.
- Training new analysts became easier, reducing onboarding time.
- Management gained **greater transparency and autonomy** in accessing data.

Results & Business Impact

By applying this **structured, step-by-step approach**, I transformed the Service Operations Report from an inefficient, error-prone process into a fast, reliable, and automated reporting system.

✅ **Time Savings:** Reduced report generation from **1 week to under 30 minutes**, saving **hundreds of analyst hours annually**.

✅ **Data Accuracy: Decreased report errors by 80%, improving decision-making confidence.**

✅ **Faster Decision-Making:** Enabled real-time insights, helping leadership make timely strategic decisions.

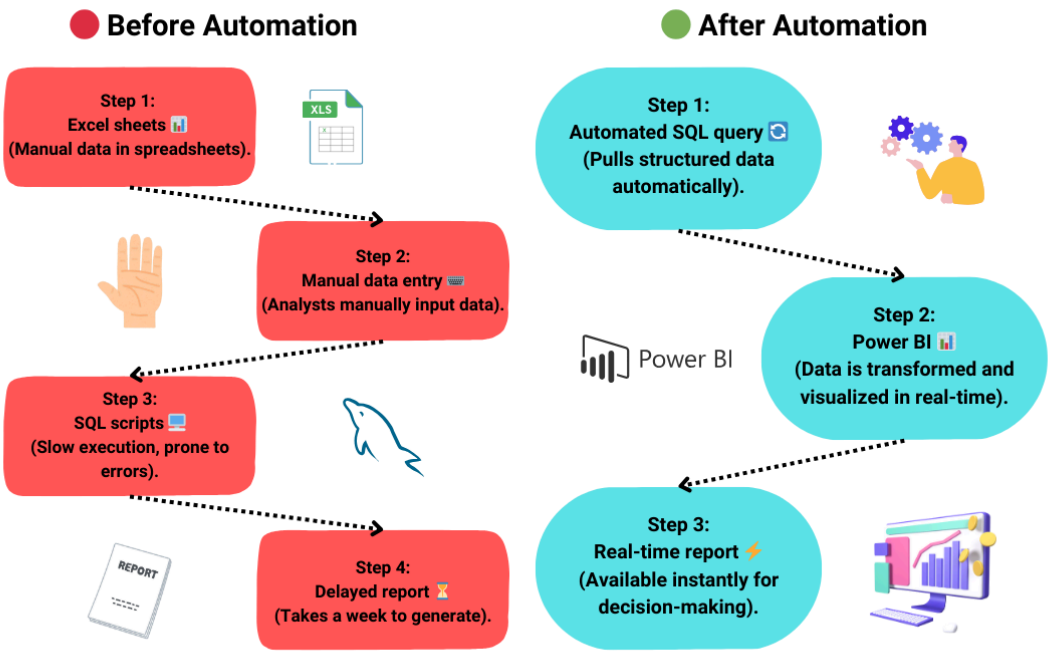
✅ **Scalability:** The automated system can now handle **larger data volumes** without performance issues.

📊 Side-by-Side Impact Table

Metric	Before Automation	After Automation
Report Update Time	1 Week	< 30 Minutes
Error Rate	5-10%	< 2%
Analyst Workload	10+ hours/week	1 hour/week
SQL Execution Time	2 Hours	5 Minutes
Decision-Making Speed	Delayed by outdated data	Real-time insights
Manual Data Handling	Required for all reports	Fully automated pipeline

This transformation not only enhanced **operational efficiency** but also improved **business strategy and employee performance evaluations**, reinforcing the value of automation in data-driven decision-making.

Results & Impact



The transformation of the Service Operations Report demonstrates the immense value of automation in operational reporting. By eliminating inefficiencies, reducing errors, and enabling real-time decision-making, this initiative not only saved hundreds of analyst hours but also empowered leadership with accurate, timely insights.

Are manual reporting processes slowing down your business? Let's discuss how automation and data-driven solutions can enhance efficiency, improve accuracy, and drive better decision-making.

📩 **Connect with me** to explore how we can optimize your reporting workflows and unlock greater business value.

```
-- Step 1: Create a temporary table to consolidate data from multiple sources
CREATE TEMP TABLE consolidated_service_data AS
SELECT
    s.transaction_id,
    s.customer_id,
    s.service_type,
    s.transaction_date,
    i.insurance_status,
    c.card_status,
    e.employee_id,
    e.branch_id,
    e.performance_score
FROM service_transactions s
LEFT JOIN insurance_records i ON s.transaction_id = i.transaction_id
LEFT JOIN card_issuance c ON s.transaction_id = c.transaction_id
LEFT JOIN employee_performance e ON s.employee_id = e.employee_id
WHERE s.transaction_date BETWEEN 'YYYY-MM-DD' AND 'YYYY-MM-DD';

-- Step 2: Optimize data cleaning and transformation
UPDATE consolidated_service_data
SET service_type = TRIM(service_type),
    insurance_status = COALESCE(insurance_status, 'Not Applicable'),
    card_status = COALESCE(card_status, 'Pending');

-- Step 3: Aggregate key performance metrics
SELECT
    branch_id,
    service_type,
    COUNT(transaction_id) AS total_transactions,
    AVG(EXTRACT(EPOCH FROM (CURRENT_TIMESTAMP - transaction_date)) / 60) AS
avg_processing_time_minutes,
    SUM(CASE WHEN insurance_status = 'Approved' THEN 1 ELSE 0 END) AS
insurance_approved,
    SUM(CASE WHEN card_status = 'Issued' THEN 1 ELSE 0 END) AS cards_issued,
    ROUND(AVG(performance_score), 2) AS avg_employee_performance
FROM consolidated_service_data
GROUP BY branch_id, service_type;

-- Step 4: Generate a summary report for Power BI visualization
CREATE TEMP TABLE report_summary AS
SELECT
    branch_id,
    COUNT(transaction_id) AS total_transactions,
    AVG(avg_processing_time_minutes) AS avg_processing_time,
    SUM(insurance_approved) AS total_insurance_approved,
    SUM(cards_issued) AS total_cards_issued,
    AVG(avg_employee_performance) AS avg_performance_score
FROM (
```

```
    SELECT
        branch_id,
        service_type,
        COUNT(transaction_id) AS transaction_id,
        AVG(EXTRACT(EPOCH FROM (CURRENT_TIMESTAMP - transaction_date)) / 60) AS
avg_processing_time_minutes,
        SUM(CASE WHEN insurance_status = 'Approved' THEN 1 ELSE 0 END) AS
insurance_approved,
        SUM(CASE WHEN card_status = 'Issued' THEN 1 ELSE 0 END) AS cards_issued,
        ROUND(AVG(performance_score), 2) AS avg_employee_performance
    FROM consolidated_service_data
    GROUP BY branch_id, service_type
) AS subquery
GROUP BY branch_id;

-- Step 5: Final output for dashboard integration
SELECT * FROM report_summary;
```

Date

1/1/2024 | 12/1/2024

139
Average Processing Time Before (minutes)

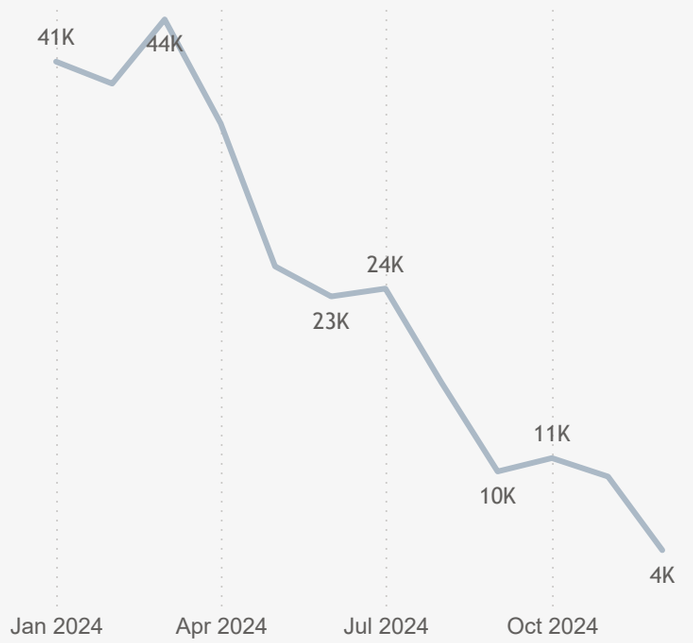
119
Average Processing Time After (minutes)

39
Analyst Hours Saved

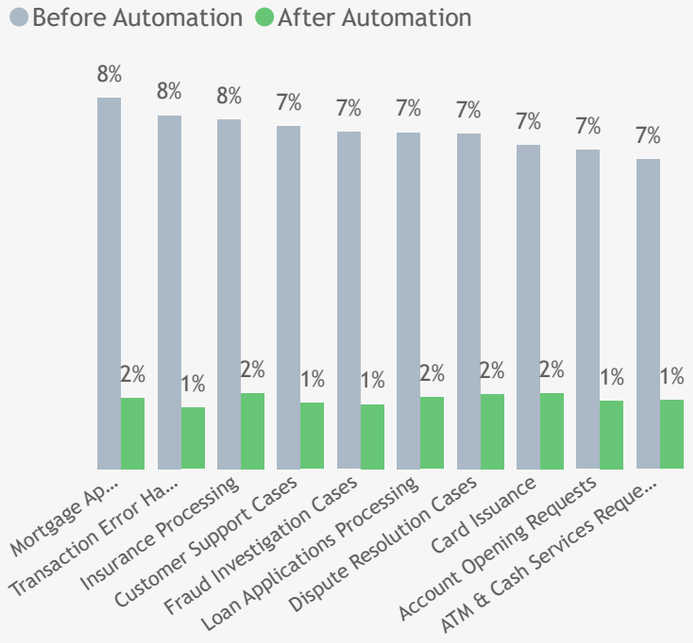
7%
Error Rate Reduction Before

2%
Error Rate Reduction After

Monthly Service Cases Processed



Error Rates Before & After Automation



Processing Time Reduction