🔍Case Study: LLM-Driven SQL Optimization for Upwork

📌Project Summary

At Upwork, I led a large-scale initiative to optimize over **4,000 legacy SQL queries** using **LLMs integrated with PROM (Prompt-Oriented Modeling)**. This project resulted in **six-figure monthly savings** by identifying inefficiencies in query design, reducing data volume scanned, and minimizing Snowflake credit usage.

Problem

Upwork's data engineering team was managing thousands of legacy SQL files across multiple marketing and analytics pipelines. These queries were: - Poorly optimized for Snowflake, leading to high compute costs - Repetitive or redundant - Not scalable as query volumes grew - Lacking visibility into cost drivers or optimization opportunities

The result was **a significant increase in infrastructure spending** and difficulty scaling the data platform efficiently.

🧭Goal

- Analyze 4,000+ SQL files
- Evaluate their cost, data volume scanned, and compute credit usage
- Use an **LLM (via PROM)** to generate optimized versions of each query
- Quantify and deliver potential savings
- Provide recommendations to engineers and analysts on improvement

---

⚙️Solution & Technical Approach

- **Query Ingestion & Parsing**: Built a system to iterate over 4,000+ SQL files and ingest each query for analysis.
- **Cost Modeling**: For each query, the system retrieved metadata including:
- Execution time
- Data scanned (TB/GB)
- Snowflake credit consumption
- **LLM Optimization Layer**: Integrated PROM to interpret each query's logic and generate **optimized SQL alternatives** while preserving functional intent.
- **Cost Simulation**: Estimated savings by comparing pre- and post-optimization metrics.
- **Recommendation Engine**: Outputted a report per file, including:
- Optimization suggestions
- Line-by-line diffs
- Estimated monthly savings
- Engineering action level (low, medium, high priority)

💡 Results

- **Analyzed**: 4,000+ SQL queries
- **Optimized**: Over 80% of files had actionable improvements
- **Savings**: Project enabled **six-figure monthly cost reductions**
- **Efficiency**: Cut down engineering review time by 70% with automation
- **Scalability**: The system is now extendable to future pipelines and integrated into query review processes

---

📈Impact

- Empowered the analytics and data engineering teams to **reduce query cost and complexity at scale**
- Allowed leadership to **monitor cost-saving metrics** per team or pipeline
- Established a framework for **future LLM-based DevOps tooling** at Upwork

---

🧰Tools & Technologies

- **PROM (LLM framework)**
- **Snowflake**, **SQL**, **Python**
- **Airflow** for orchestration
- **GitHub Actions** for automation
- **Looker** for reporting and stakeholder dashboards

🧲Lessons Learned

- LLMs can significantly improve technical workflows when paired with **domain-specific metadata and constraints**
- Engineering teams benefit from cost visibility that is **proactive, not reactive**
- Optimization is not just about compute savings—it improves **performance, reliability, and developer experience**