DataMigrate AI

Machine Learning Strategy

Fine-Tuning Open-Source Models for SQL Migration

Technical Documentation & Business Case

Company:	OKO Investments
Author:	Alexander Garcia Angus
Date:	December 2025
Version:	1.0
Classification:	Confidential

Table of Contents

1. Executive Summary

This document outlines DataMigrate AI's strategy for developing proprietary AI models through fine-tuning open-source language models. By collecting anonymized SQL migration patterns from customer engagements, we can create specialized models that outperform generic AI APIs while reducing operational costs by 70% and creating a defensible competitive moat.

* Primary Model: SQLCoder 15B - Best SQL generation accuracy
* Training Data: Anonymized schema patterns from customer migrations
* Cost Reduction: 70% savings vs API-based approach at scale
* Competitive Moat: Proprietary training data cannot be replicated
* Timeline: 12-18 months from data collection to production deployment

2. Why Fine-Tune Open-Source Models?

Current State: API-Based Approach

DataMigrate AI currently uses Claude and GPT APIs for SQL generation. While effective, this approach has limitations at scale:

- Cost: $0.01-0.10 per API call, scales linearly with usage
- Latency: 2-5 seconds per response
- Privacy: Customer data sent to third-party servers
- No Differentiation: Competitors can use same APIs
- No Offline: Requires internet connectivity

Future State: Fine-Tuned Models

+ Cost: ~$0.001 per query (self-hosted), 70% reduction
+ Latency: 200-500ms per response (5-10x faster)
+ Privacy: All processing on-premise, no data leaves
+ Differentiation: Proprietary model trained on our data
+ Offline: Full functionality without internet
+ Specialization: Model learns MSSQL-to-dbt patterns specifically

3. Recommended Model: SQLCoder

After evaluating multiple open-source models, we recommend SQLCoder as the primary base model for fine-tuning. SQLCoder is specifically designed for SQL generation tasks and outperforms general-purpose models on SQL benchmarks.

Why SQLCoder?

Criteria	SQLCoder 15B	GPT-4	Claude 3
SQL Accuracy (Spider)	85%	82%	80%
MSSQL Support	Excellent	Good	Good
Fine-tuning Support	Full (open weights)	None	None
Self-hosting	Yes	No	No
Cost per Query	~$0.001	$0.03	$0.015
Latency	300ms	3s	2s
Offline Capable	Yes	No	No

SQLCoder Technical Specifications

- Model Size: 15B parameters (also available in 7B)
- Architecture: Transformer (based on StarCoder)
- Context Window: 8,192 tokens
- Training: SQL-specific datasets including Spider, WikiSQL
- License: Apache 2.0 (commercial use allowed)
- Fine-tuning: Supports LoRA, QLoRA for efficient training
- Inference: Compatible with Ollama, vLLM, HuggingFace

4. Alternative Models Comparison

Model	Size	Specialization	Pros	Cons
SQLCoder	7B-15B	SQL Generation	Best SQL accuracy	SQL-only
CodeLlama	7B-34B	General Code	Versatile, Meta backing	Less SQL-specific
Llama 3	8B-70B	General Purpose	Strong reasoning	Needs more fine-tuning
Mistral	7B	General Purpose	Fast, efficient	Smaller context
DeepSeek-Coder	6.7B-33B	Code	Good SQL support	Less community
StarCoder2	3B-15B	Code	GitHub trained	Less SQL focus

Recommendation by Use Case

- Primary (SQL Generation): SQLCoder 15B - Best accuracy for SQL tasks
- Backup (Complex Logic): CodeLlama 34B - Better reasoning for stored procedures
- On-Premise (Resource Limited): SQLCoder 7B - Lower hardware requirements
- Future (Advanced): Llama 3 70B - When we have more training data

5. Data Collection Strategy

The key to successful fine-tuning is high-quality training data. We will collect SQL migration patterns from customer engagements, with strict privacy controls.

What We Collect (Schema Only)

[Collect] SQL stored procedure structures and logic patterns

[Collect] Table and column definitions (CREATE TABLE statements)

[Collect] JOIN patterns and relationships

[Collect] Data type mappings (MSSQL to dbt)

[Collect] Transformation logic (aggregations, filters, CTEs)

[Collect] Successfully generated dbt model outputs

What We NEVER Collect

[NEVER] Actual row-level data (customer records, transactions)

[NEVER] Personal identifiable information (names, SSNs, emails)

[NEVER] Financial data (account numbers, balances)

[NEVER] Business-sensitive values (prices, salaries)

[NEVER] Authentication credentials

Training Data Quality Requirements

Criteria	Requirement	Validation Method
Success Score	> 0.8 (80%)	Automated testing
Completeness	Full input/output pair	Schema validation
Diversity	Multiple SQL patterns	Pattern clustering
Accuracy	Validated dbt output	dbt compile test

6. Privacy & Anonymization

All collected data undergoes mandatory anonymization before storage. This ensures customer privacy while retaining the SQL patterns needed for training.

Anonymization Process

```
BEFORE ANONYMIZATION (Customer SQL):
CREATE PROCEDURE dbo.GetCustomerCreditScore
@CustomerSSN VARCHAR(11)
AS
SELECT customer_id, first_name, credit_score
FROM dbo.CustomerMaster
WHERE social_security_number = @CustomerSSN

AFTER ANONYMIZATION (Training Data):
CREATE PROCEDURE dbo.Proc_001
@Param_A VARCHAR(11)
AS
SELECT col_001, col_002, col_003
FROM dbo.Table_A
WHERE col_004 = @Param_A
```

Three Levels of Data Collection

Level	Description	Privacy	Model Quality
Anonymized (Default)	All names replaced with generic tokens	Maximum	Good
Named (Opt-in)	Table/column names preserved	High	Better
No Collection	Customer opts out entirely	N/A	N/A

7. Training Pipeline

End-to-End Process

1. DATA COLLECTION: Capture successful migrations during customer engagements
2. ANONYMIZATION: Remove all identifying information from SQL patterns
3. QUALITY FILTER: Only keep examples with success score > 0.8
4. FORMAT CONVERSION: Convert to JSONL training format
5. BASE MODEL: Load SQLCoder 15B as starting point
6. FINE-TUNING: Apply LoRA/QLoRA for efficient training
7. EVALUATION: Test on held-out migration examples
8. DEPLOYMENT: Deploy via Ollama or vLLM
9. MONITORING: Track accuracy and collect feedback
10. ITERATION: Retrain with new data quarterly

Training Data Format (JSONL)

```
{
"messages": [
{
"role": "system",
"content": "You are a SQL migration expert..."
},
{
"role": "user",
"content": "Convert MSSQL procedure: CREATE PROCEDURE..."
},
{
"role": "assistant",
"content": "SELECT col_001 FROM {{ ref('stg_table_a') }}..."
}
]
}
```

8. Business Benefits & ROI

Cost Comparison (Annual, 200 Migrations)

Cost Category	API-Based	Fine-Tuned	Savings
AI Inference	36,000 DKK	3,600 DKK	32,400 DKK
GPU Infrastructure	0 DKK	12,000 DKK	-12,000 DKK
ML Engineer (Part-time)	0 DKK	50,000 DKK	-50,000 DKK
Total Annual Cost	36,000 DKK	65,600 DKK	-29,600 DKK
Year 3 (500 migrations)	90,000 DKK	70,000 DKK	20,000 DKK

Strategic Benefits

* COMPETITIVE MOAT: Proprietary model trained on real migrations - cannot be replicated
* ON-PREMISE SALES: Enable enterprise deals requiring data isolation (500K-2M DKK)
* SPEED: 5-10x faster responses improve user experience
* SCALABILITY: Fixed infrastructure cost regardless of volume
* IP OWNERSHIP: Model becomes company asset, increases valuation

9. Implementation Roadmap

Phase	Timeline	Milestone	Investment
Data Collection	Months 1-12	500+ training examples	Built into product
ML Hire	Month 6	Contract ML engineer	50,000 DKK
First Training	Month 9	v0.1 model on cloud GPU	5,000 DKK
Evaluation	Month 10	Benchmark vs Claude/GPT	Internal
Production	Month 12	Deploy for simple tasks	10,000 DKK
Scale	Month 18	80% tasks on fine-tuned model	GPU server

10. Infrastructure Requirements

Training Infrastructure (Cloud)

- GPU: NVIDIA A100 80GB (cloud rental)
- Cost: ~$2-4/hour, training takes 4-8 hours
- Total per training run: ~$30-50
- Frequency: Quarterly retraining
- Provider: AWS, Lambda Labs, or RunPod

Inference Infrastructure (Production)

Option	Hardware	Cost	Use Case
Cloud GPU	A10G on AWS	~500 DKK/month	Testing, low volume
Dedicated Server	RTX 4090 (24GB)	~35,000 DKK one-time	Production
Enterprise	2x A10 (48GB)	~100,000 DKK	High availability

11. Risk Analysis

Risk	Likelihood	Impact	Mitigation
Insufficient training data	Medium	High	Aggressive customer acquisition
Model underperforms	Low	Medium	Keep Claude API as fallback
Privacy breach	Low	Critical	Strict anonymization, audits
GPU costs increase	Low	Low	Long-term cloud contracts
Competitor catches up	Medium	Medium	Continuous data collection