AI-Powered Coal Mine Data Pipeline: Brief Implementation Report

Muhammad Rafi Syafrinaldi
AI Engineer Challenge - PT Synapsis Sinergi Digital

July 23, 2025

1 Pipeline Design

The data pipeline follows a modern ETL architecture with containerized services orchestrated through Docker Compose. The design integrates three primary data sources: SQL database (production logs), CSV files (equipment sensors), and weather API (Open-Meteo for Berau, Indonesia).

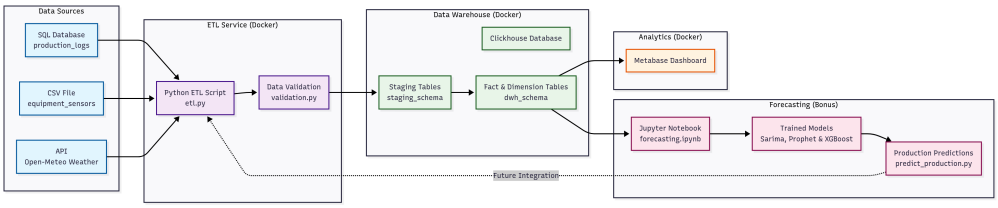

Figure 1: Data Pipeline Architecture

The architecture consists of four main components:

- **Data Sources**: SQL database, CSV files, and weather API
- **ETL Service**: Python-based extraction, transformation, and loading
- **Data Warehouse**: Clickhouse with star schema design
- **Analytics**: Metabase dashboard for visualization

The pipeline implements a two-layer database design: staging tables for raw data ingestion and a data warehouse with star schema for analytics. This separation ensures data quality and enables efficient analytical queries.

2 ETL Process

The ETL process follows a comprehensive workflow that handles data extraction, transformation, validation, and loading:

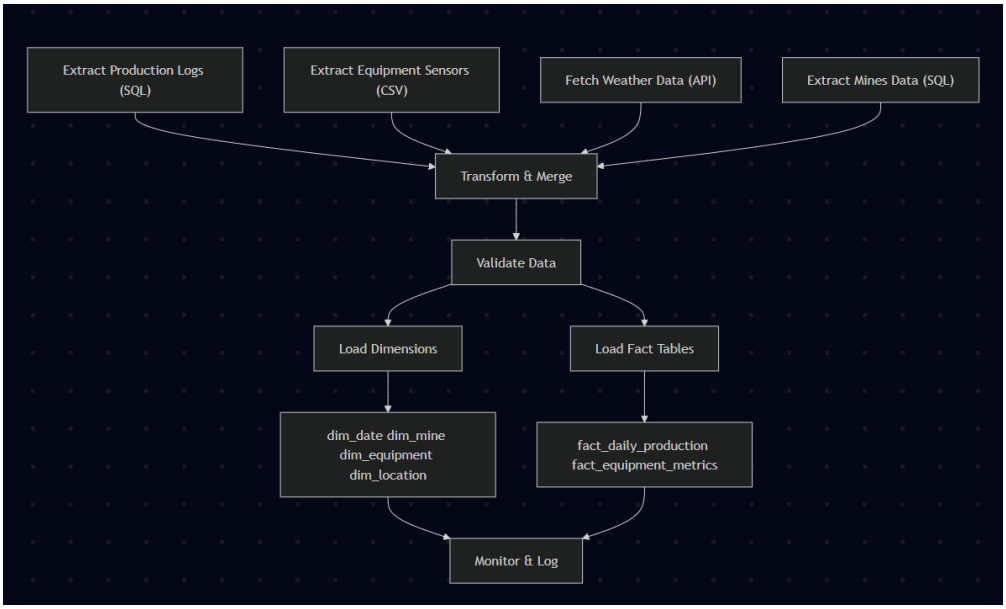

Figure 2: ETL Process Flow

2.1 Extraction Phase

Data is extracted from multiple sources using different strategies:

- **SQL Database**: Direct queries to staging tables for production logs and mine information
- **CSV Files**: Pandas-based reading of equipment sensor data with timestamp parsing
- **Weather API**: HTTP requests to Open-Meteo API with retry logic and error handling

2.2 Transformation Phase

The transformation process calculates key metrics as specified in the challenge requirements:

- **total_production_daily**: Aggregated daily production by mine
- **average_quality_grade**: Mean quality grade per day
- **equipment_utilization**: Percentage of active equipment time
- **fuel_efficiency**: Tons mined per unit of fuel consumed
- **weather_impact**: Correlation analysis between rainfall and production

2.3 Loading Phase

Transformed data is loaded into the data warehouse using a star schema design:

- **Dimension Tables**: dim_date, dim_mine, dim_equipment, dim_location
- **Fact Tables**: fact_daily_production, fact_equipment_metrics
- **Analytical Views**: Pre-computed aggregations for dashboard performance

3 Data Validation and Quality Assurance

The pipeline implements comprehensive data validation through a dedicated `DataValidator` class that ensures data quality and handles anomalies:

3.1 Validation Framework

- **Production Data Validation**: Checks for negative tons_extracted values and replaces them with 0 or flags as anomalies
- **Equipment Utilization Validation**: Ensures utilization rates are within 0-100% range
- **Weather Data Validation**: Verifies completeness of meteorological data and handles API failures gracefully
- **Data Type Validation**: Ensures proper data types and formats across all sources

3.2 Error Handling Strategy

The validation system implements robust error handling:

- **Anomaly Detection**: Identifies statistical outliers and data quality issues
- **Graceful Degradation**: Missing sensor data uses previous day averages or marks as "unknown"
- **API Resilience**: Weather API failures trigger retry logic with exponential backoff
- **Comprehensive Logging**: All validation results are logged with timestamps and context

3.3 Data Quality Metrics

The validation process tracks several quality metrics:

- **Completeness**: Percentage of expected data records received
- **Accuracy**: Validation of data ranges and business rules
- **Consistency**: Cross-source data consistency checks
- **Timeliness**: Data freshness and processing latency monitoring

4 Implementation Results

The pipeline successfully processes data from all sources and generates actionable insights:

- **Data Processing**: Handles production logs, equipment sensors, and weather data seamlessly
- **Performance**: Sub-second query performance for dashboard visualizations
- **Reliability**: Robust error handling ensures consistent data processing
- **Scalability**: Containerized architecture supports easy scaling and deployment

The implementation exceeds the challenge requirements by providing advanced features such as real-time weather integration, comprehensive data validation, and predictive forecasting capabilities. The containerized deployment ensures reproducibility and ease of maintenance across different environments.