Project Report

SQL ETL Pipeline Simulation

Abstract

The Extract, Transform, and Load (ETL) process is one of the most important steps in building reliable data pipelines. This project simulates an ETL pipeline using SQL in MySQL Workbench. Raw sales data is imported into a staging area, cleaned, transformed, and then loaded into a production table. An audit mechanism is also implemented to track inserts, ensuring data quality and reliability.

Introduction

In real-world organizations, raw data is often inconsistent, duplicated, or incomplete. ETL processes help to standardize this data for analytics and reporting.

- **Extract:** Collect raw data from different sources.
- **Transform:** Clean and format the data by removing duplicates, handling missing values, and applying transformations.
- **Load:** Insert the processed data into a production table for business use.

This project demonstrates the ETL process purely using SQL queries without external ETL tools.

Objectives

- To design and simulate an ETL pipeline using SQL.
- To handle raw data in a staging table.
- To clean and transform data for production use.
- To maintain an audit log for data tracking.
- To automate cleanup using triggers.

Tools & Technologies Used

- **Database:** MySQL Workbench
- **Data Source:** CSV (raw sales data)
- **Language:** SQL

- **Export Tools:** MySQL Workbench CSV Export

Steps Involved

1. Data Extraction (Staging Area)

- Raw sales CSV imported into staging_sales.

2. Data Transformation (Cleaning)

- Remove duplicates and null values using SQL queries.
- Create clean_sales table with valid records only.

3. Data Loading (Production Table)

- Insert cleaned data into production_sales.
- If duplicate keys exist, update existing records.

4. Audit Logging

- An etl_audit table created to log insert operations.
- Trigger ensures automatic tracking of data loads.

5. Automation Using Triggers

- Trigger removes duplicates in staging automatically.
- Trigger logs every insert in production into audit table.

6. Exporting Final Results

- Final tables etl_audit exported as CSV for reporting.

Conclusion

The SQL-based ETL pipeline successfully simulates data extraction, cleaning, transformation, and loading into production. Audit logging ensures traceability, while triggers automate data quality checks. This project demonstrates how SQL alone can implement a lightweight ETL pipeline, which is scalable for small and medium datasets.

Future Scope

- Automating the ETL pipeline using **stored procedures** and **event schedulers**.
- Integrating with **Python** or **ETL tools** for large-scale automation.
- Adding **error handling** and **rollback mechanisms**.
- Building **dashboards** on top of production data for visualization.