SQL ETL Pipeline Simulation - Project Report

Introduction

This project demonstrates a practical ETL (Extract, Transform, Load) workflow built using only SQL and PostgreSQL tools. It provides a simple yet effective approach to automate the process of importing raw CSV data, cleaning and transforming it, tracking changes using an audit log, and exporting the final cleaned data.

Abstract

In data management, ETL processes are crucial for transforming raw data into meaningful information. This project simulates an end-to-end ETL pipeline by:
- Importing raw employee data from CSV files.
- Loading this data into a staging table.
- Cleaning and transforming it into a structured production table.
- Logging all insert actions in an audit table.
- Exporting the cleaned production data and logs back to CSV files for further use.

The pipeline is fully automated using stored procedures and triggers in PostgreSQL, ensuring minimal manual intervention and efficient data flow.

Tools Used

- Database: PostgreSQL (tested in pgAdmin 4)
- Language: SQL / PLpgSQL
- Input Format: CSV files containing raw employee data
- Output Format: CSV exports for cleaned data and logs

Steps Involved in Building the Project

1. Schema Creation

Created three tables:
- staging_employees: Stores raw data as-is from the CSV files.
- employees: The cleaned production table with valid, transformed data.
- etl_audit_log: Records each ETL operation for traceability.

2. Data Import Procedure

A stored procedure load_staging_employees accepts a file path as input and dynamically executes a COPY command to load CSV data into the staging_employees table.

3. Transformation and Logging Trigger

A trigger function Clean_Transform_data_insert_log_clear_staging runs automatically whenever new data is inserted into the staging table. It:

- Cleans the data by removing duplicates and nulls.
- Casts raw text fields to correct data types.
- Loads valid records into the employees table.
- Logs the operation in the etl_audit_log table.
- Clears the staging table for the next batch.

4. Export Procedures

Two stored procedures:
- Export_final_table: Exports the final cleaned employees table.
- Export_Audit_table: Exports the etl_audit_log.

5. Testing and Execution

Using provided queries:
- Load data using CALL load_staging_employees(...).
- Check intermediate and final tables.
- Export results as CSVs to local storage.

Conclusion

This ETL pipeline project showcases how SQL and PLpgSQL can be used to build a robust and automated data processing workflow without third-party tools. Using triggers and stored procedures reduces repetitive tasks, ensures clean data, and maintains a detailed audit trail. Such pipelines are highly adaptable for various real-world scenarios where lightweight, in-database ETL solutions are needed.

Prepared By: Darji Chintankumar Dineshchandra

Date: 2025-07-18

Deliverables

- SQL Scripts (.sql files)
- Cleaned production and audit tables
- Final exported CSV files