**WEATHER ETL PROJECT**

**Objective**

To develop a Python-based ETL (Extract, Transform, Load) pipeline that processes historical weather data from fixed-format .txt files, stores it in a structured database, and generates annual statistical summaries per weather station.

**Tasks Completed:**

**Data modelling**

- Defined SQLAlchemy ORM models for WeatherData (daily weather records) and WeatherSummary (annual aggregated data).  
- Established uniqueness constraints on (station\_id, date) and (station\_id, year).

**Data ingestion**

- Parsed .txt file line-by-line.  
- Handled missing values (-9999) by converting to None.  
- Converted temperatures and precipitation to standard units.  
- Stored cleaned records into a SQLite database (weather.db).

**Data Analysis**

- Grouped data by station and year.  
- Calculated average max/min temperatures and total precipitation.  
- Stored results in WeatherSummary table.

**ETL Integration**

- Built etl.py to run both ingestion and analysis.  
- Created main.py to trigger the full process in one step.

**Logging**

- Implemented detailed logging in ingestion.log, analysis.log, and etl.log.  
- Captured data issues and ETL status.

**Deliverables:**

|  |  |
| --- | --- |
| File | Description |
| models.py | Database table definitions using SQLAlchemy ORM |
| ingestion.py | Script to ingest and clean weather data |
| analysis.py | Script to compute yearly summaries |
| etl.py | Full pipeline script combining ingestion + analysis |
| main.py | Entry point to run the entire ETL process |
| weather.db | Output SQLite database with two structured tables |
| \*.log files | Process logs for debugging and verification |
| README.md | Usage instructions and setup guide |

**Outcome:**

This project provides a reusable, modular ETL system that:  
- Automates weather data parsing and analysis  
- Supports multiple stations and years  
- Ensures data integrity through constraints and logging