ETL Projects

Project No 1

Education Analytics Pipeline

Problem: A university wants to analyze student performance, attendance, and learning system activity to predict dropouts.

ETL Pipeline

- **Extract**: Student exam results from CSV (dataset).
- **Transform**:
  - Clean missing values
  - Normalize student IDs
  - Compute average marks per student
  - Add grade classification (A/B/C/D/F)
- **Load**: Store curated data in Hive (or even just HDFS/Parquet if Hive is not needed).

Appropriate Dataset

A good open dataset for this is:https://www.kaggle.com/search?q=Students+Performance+in+Exams

“Students Performance in Exams” dataset (Kaggle / UCI)

- CSV format
- Fields: `gender`, `race/ethnicity`, `parental level of education`, `lunch`, `test preparation course`, `math score`, `reading score`, `writing score`
- Link: Kaggle – Students Performance in Exams

This dataset works perfectly because:

- Single CSV source (easy extraction).
- Contains exam results → can calculate average score, grade, pass/fail.
- Real-world like features (demographics + scores).

Simplified Team Tasks

1. Ingest the CSV dataset (exam results) into HDFS.
2. Clean student records (remove nulls, fix data types).
3. Transform:
    - Compute average score = (math + reading + writing)/3
    - Assign grade (A/B/C/D/F).
4. Load curated data into Hive table.
5. Generate a summary report (average marks, grade distribution).

Project No 2

ETL for Movie Metadata

Goal: Extract a single dataset of movie/TV content, clean/transform it, and load it into Hive (or a database) for analysis.

Pipeline

1. **Extract:**
    - Use a **single dataset** (e.g., movies metadata).
2. **Transform:**
    - Clean missing values, normalize genres, parse release dates.
    - Compute simple metrics like *average rating per genre*.
3. **Load:**
    - Store curated dataset into **Hive (batch analytics)** or **Postgres/MySQL**.

Suggested Dataset (Free & Public)

- **The Movies Dataset (by GroupLens / Kaggle)**
  Contains metadata for 45,000 movies + 26M ratings.
  Format: CSV (easy for ETL).
  Link: The Movies Dataset on Kaggle

Alternative (smaller if you want simpler):

- **MovieLens 1M Dataset** (1M ratings, CSV files) → MovieLens datasetshttps://www.kaggle.com/search?q=MovieLens+1M+Dataset

---

Example ETL Task

- **Extract:** Load `movies_metadata.csv`.
- **Transform:**
    - Drop duplicates, fix null genres, convert release_date to `YYYY-MM-DD`.
    - Compute *average rating by genre*.
- **Load:** Save final dataset into Hive table `movies_curated`.

Project No 3

ETL Crop Yield Prediction Pipeline

Goal: Agricultural board wants to analyze soil data, weather feeds, and crop production records to predict yields & suggest optimal farming practices.

Crop Yield Prediction Pipeline

Problem

Predict crop yield based on **soil, weather, and crop production data**.

Farmers/agriculture boards can use insights to plan better.

Dataset (Simpler & Available)

- **Kaggle – Crop Yield Prediction Dataset**
  👉 Crop Yield Prediction Dataset
  (Already has soil, rainfall, temperature, and crop yield fields — no need for 3 different sources).

Pipeline Steps

Extract

- Use **single Kaggle dataset** (CSV).
- (Optional extension: Add 1 small open weather dataset to mimic API ingestion).

Transform

- Clean missing values (replace with mean/median).
- Normalize units (convert temp °F → °C, rainfall mm → cm).
- Feature engineering:
    - Rainfall deviation (rainfall – avg rainfall).
    - Soil fertility index (combine soil NPK values).

Load

- Store cleaned & transformed data in **Hive tables**.
- Partition by **year/crop/region**.

Project No 4

Goal: Energy companies want to process solar/wind generation data + consumption data to optimize energy distribution..

Simplified Renewable Energy Pipeline

Problem

Optimize renewable energy grid balancing by analyzing **solar/wind generation data** against **household consumption data** to detect demand peaks and grid imbalance.

Datasets

1. **Solar Generation Data** – Solar Power Generation Data (Kaggle)
   (Time series of solar plant power output, multiple sensors).
2. **Wind Power Generation** – Open Power System Data – Renewable Energy
   (Wind & solar generation, Europe).

3. **Smart Meter Consumption Data** – UK Domestic Energy Consumption (Kaggle) (Half-hourly electricity consumption from households).

These 3 datasets are enough to simulate the whole pipeline.

Simplified Pipeline

Extract

- **Solar logs** – Read from CSV (instead of Kafka for simplicity).
- **Wind logs** – Read from CSV.
- **Smart meter consumption** – Read from Parquet (or CSV if easier).

Transform

- Clean errors (drop negative or null values).
- Normalize timestamps → resample to **hourly**.
- Aggregate energy generated (solar + wind).
- Aggregate hourly consumption.
- Calculate **peak vs off-peak demand**.

Load

- Store curated datasets in **Hive** (batch queries for BI).
- Store **aggregates** .

Team Task Breakdown (6 members × 3 tasks each)

Member 1 (Extraction – Scala)

1. Ingest solar CSV into Spark (Scala).
2. Clean missing/negative values.
3. Normalize timestamps.

Member 2 (Extraction – Python)

1. Ingest wind CSV into Spark (Python).
2. Join with solar dataset.
3. Write cleaned results to Hive.

Member 3 (Consumption Data – PySpark)

1. Ingest smart meter Parquet.
2. Aggregate hourly consumption.
3. Store in Hive.

Member 4 (Analytics – Python ML)

1. Detect imbalance (demand > generation).
2. Save alerts .

Member 5 (Monitoring)

1. Build Spark pipeline.
2. Simulate live solar/wind input.
3. Push metrics to storage.

Member 6 (Orchestration & Docs)

1. Project documentation.

Tools Stack

- **Spark + Scala/Python** (ETL & Aggregation).
- **Hive** (batch queries & BI).

Project No 5

Simplified Project: Smart Transportation & Traffic Flow

Goal: Analyze traffic + GPS data to optimize congestion management and predict hotspots.

Pipeline (Simplified Steps)

1. **Extract (Data Sources)**
    - **GPS Logs** → JSON (bus/vehicle movement)
    - **Traffic Sensors** → CSV (vehicle counts, speed)
    - **Camera Metadata** → API (location, timestamp, congestion level)
2. **Transform (Processing)**
    - Clean and standardize all datasets
    - Map data to **road segments**
    - Remove incomplete/missing GPS points
    - Aggregate: average speed, congestion level per segment
    - Compute travel time metrics
3. **Load (Storage & Access)**
    - Store **historical data** in Hive (for analysis)

Team Tasks (Clear & Grouped)

Data Ingestion

- Ingest GPS JSON logs
- Ingest road sensor CSVs
- Ingest camera metadata from API

Data Cleaning & Standardization

- Standardize road segment IDs
- Clean incomplete GPS records

Data Processing & Metrics

- Aggregate congestion per segment
- Build travel time metrics (Scala)
- Predict congestion hotspots (Python ML)

Testing & Optimization

- Test GPS ingestion logic
- Tune Spark performance

Recommended Dataset

You can use:

METR-LA Traffic Dataset

- Traffic speed readings from **LA road sensors**
- Includes **timestamps, sensor IDs, speeds**
- Great for congestion prediction tasks
- Widely used in ML traffic flow research

OR

NYC Taxi & Limousine GPS Dataset

- Large-scale **GPS logs of taxis**
- Includes pickup/drop-off coordinates, times, passenger counts
- Useful for congestion & travel-time prediction

Project No 6

Problem

Study the impact of climate change by combining **historical weather**, **CO_2 emissions**, and **deforestation trends**.

Pipeline

1. **Extract**
    - Historical weather data (CSV)
    - Live CO_2 emission data (API JSON)
    - Forest cover/deforestation metadata (CSV/JSON)
2. **Transform**
    - Standardize units (°C for temp, ppm for CO_2, hectares for forests)
    - Handle missing values in historical data
    - Join datasets by region + year
    - Derive new metrics:
        - Yearly emission trends
        - Per capita CO_2
        - Deforestation KPIs
3. **Load**
    - Store curated datasets in **Hive**
    - Create **aggregation tables** (by country, region, year)
    - Build dashboards (**Tableau/PowerBI**)

- Add anomaly alerts for CO_2 spikes

Team Tasks (Simplified & Grouped)

1. **Data Ingestion**
   - Ingest historical weather CSV
   - Connect to CO_2 API
   - Parse forest cover datasets
2. **Data Processing**
   - Standardize metrics
   - Handle missing records
   - Join datasets
3. **Analytics**
   - Calculate emission & deforestation trends
   - Derive per capita CO_2
   - Create KPIs
4. **Data Storage**
   - Store in Hive
   - Create regional/yearly aggregation tables

Suggested Datasets

1. **Historical Weather Data (CSV)**
   - NOAA Global Historical Climatology Network (CSV, global temps & precipitation)
   - Berkeley Earth Surface Temperature (CSV, country & global temperatures)
2. **CO_2 Emissions (API / CSV)**
   - Global Carbon Atlas (CSV download)
   - Our World in Data CO_2 Emissions (CSV & API access)
3. **Forest Cover / Deforestation**

- Global Forest Watch (GFW) (CSV/JSON, satellite-based forest cover & loss)
- FAO Global Forest Resources Assessment (CSV, forest statistics)
