**Practice and Review Advanced SQL and DataWarehouse Topics including:**

* Common Table Expressions (CTEs) & Recursive CTEs
* Window Functions (ROW\_NUMBER, RANK, DENSE\_RANK, LEAD, LAG)
* Pivoting & Unpivoting Data
* Stored Procedures & Triggers
* Star Schema & Snowflake Schema in Data Warehousing

P4 project (solo or in pairs):

**Project Title: Real-Time Analytics Using PySpark and HDFS**

**Project Overview**

This project focuses on **building a scalable data pipeline** to process and analyze movie ratings data or API data in real time. You'll use **HDFS for storage, PySpark for processing, and SQL for querying insights.**

**Tech Stack**

* **Data Source**: Kaggle MovieLens dataset and a real time API like New York Times Articles API (<https://developer.nytimes.com/apis> ) or use real-time streaming data (e.g., <https://newsapi.org/> or <https://coinmarketcap.com/> ) that interests you
* **Storage**: HDFS (for storing raw and processed data)
* **Processing**: PySpark (for batch and streaming processing)
* **Database**: PostgreSQL/MySQL for storing aggregated insights
* **Visualization**: Tableau / Power BI / Matplotlib in Python
* **Optional**: Kafka (for real-time streaming data), Flask/Django (for a simple web UI)

**Project Workflow**

1. **Data Ingestion**
   * Download the **MovieLens dataset and download related real time movie articles** or **stream other data from an api**.
   * Store raw data into **HDFS** in structured formats like **CSV, JSON, or Parquet**.
2. **Data Preprocessing & Transformation (Using PySpark)**
   * Remove duplicates, handle missing values, and clean data.
   * Normalize ratings (e.g., scale 1-5 to 1-10) or streamed data.
   * Convert timestamps to readable datetime formats.
   * Any other transformations necessary
   * Store cleaned data back into **HDFS (as Parquet or ORC for efficient querying).**
3. **Batch & Streaming Processing**
   * **Batch Processing (ETL)**
     + Use **PySpark SQL** to run queries like:
       - Top-rated movies by genre.
       - Most active users.
       - Trends in ratings over time.
     + Store results in a **PostgreSQL/MySQL database**.
   * **Streaming Processing (Optional)**
     + If using real-time streaming data:
       - Process tweets about movies using **PySpark Streaming and/or Kafka**.
       - Perform real time analytics, **sentiment analysis**, or aggregations in real time
4. **Analysis & Visualization**
   * Write SQL queries in PostgreSQL/MySQL to fetch insights.
   * Visualize results in **Power BI and/or Matplotlib/Seaborn**.
5. **Optional: Deploy as a Web Dashboard**
   * Use Flask/Django (or Tornado etc.) to build a simple web UI that displays:
     + Top movies by rating
     + Most popular movie genres
     + Real-time sentiment trends from social media

**Potential Enhancements**

* Implement **recommendation algorithms** (e.g., collaborative filtering).
* Optimize query performance with **Spark DataFrames and caching**.
* Automate the pipeline using **Apache Airflow**.
* Deploy on a **cloud platform (AWS S3 + EMR)**

**Why This Project?**

**Industry-Relevant**: Mimics real-world Big Data workflows.  
**Scalable**: Uses PySpark for efficient processing.  
**Hands-On with HDFS**: Storing and retrieving large datasets.  
**Mix of Batch & Streaming**: Covers both traditional and real-time data processing.  
**Data Visualization**: Adds business intelligence skills.