C S 31B: INTRODUCTION TO DATA ENGINEERING

Foothill College Course Outline of Record

Heading	Value
Effective Term:	Fall 2026
Units:	4.5
Hours:	4 lecture, 2 laboratory per week (72 total per quarter)
Prerequisite:	C S 31A.
Degree & Credit Status:	Degree-Applicable Credit Course
Foothill GE:	Non-GE
Transferable:	CSU Approved, UC Pending
Grade Type:	Letter Grade (Request for Pass/No Pass)
Repeatability:	Not Repeatable

Student Learning Outcomes

- Design and implement automated ETL pipelines using Python, SQL, open-source tools, and workflow orchestration to support analytics and machine learning
- Apply data storage architectures, modeling techniques, scaling strategies, and containerization to build reproducible data workflows
- Integrate data quality practices, version control, and model deployment (MLOps) into end-to-end data engineering projects
- Recognize the potential for bias in automated decision-making and analyze the ethical, environmental, and privacy implications of data engineering infrastructure and workflows

Description

Introduction to the design and implementation of scalable data infrastructure. Topics include data ingestion techniques, storage architectures, horizontal and vertical scaling strategies, ETL pipeline development, and model deployment (MLOps). Explores workflow orchestration, data modeling for analytics, and containerization. Students gain practical experience building automated data workflows using Python, SQL, and open-source tools, with less emphasis on complex distributed systems theory.

Course Objectives

The student will be able to:

1. Explain data engineering roles, the data lifecycle, and the modern data stack.
2. Design and evaluate analytical data storage architectures and modeling patterns.
3. Implement batch, streaming, and API-based data ingestion techniques.
4. Assess scaling strategies and distributed computing fundamentals for data infrastructure.
5. Develop and validate ETL pipelines using Python and SQL.
6. Automate and manage scheduled data workflows using orchestration tools.
7. Package and deploy data applications using containerization techniques.
8. Integrate pre-trained machine learning models for batch data enrichment.
9. Apply data privacy, security, and access control principles to data pipelines.
10. Evaluate the ethical, social, and environmental impacts of data engineering workflows and infrastructure.

Course Content

1. Fundamentals of data engineering
   a. Data roles and career pathways
   b. The data engineering lifecycle
   c. Overview of the modern data stack and open-source ecosystems
2. Data storage architectures and modeling
   a. Relational databases, NoSQL, and object storage
   b. Data warehouses, data lakes, and lakehouses
   c. Pipeline architecture patterns: medallion architecture
   d. Data modeling for analytics (star schema, snowflake schema)
   e. Columnar vs. row-oriented storage formats
3. Data ingestion techniques
   a. Batch processing vs. stream processing
   b. API integration and flat file ingestion
   c. Event streaming concepts (e.g., Apache Kafka)
   d. Incremental vs. full data loads
4. Scaling strategies and infrastructure
   a. Vertical scaling (scale-up) vs. horizontal scaling (scale-out)
   b. Distributed computing fundamentals
   c. Cloud-agnostic infrastructure concepts
5. ETL pipeline development
   a. Extracting data from disparate sources
   b. Data quality and validation (handling nulls, duplicates, and schema enforcement)
   c. Data transformation techniques (joining, aggregating) using Python and SQL
   d. Loading data into analytical storage
   e. ETL vs. ELT and transformation-in-place
   f. SQL analytical functions for data transformation: window functions and common table expressions (CTEs)
6. Workflow orchestration
   a. Scheduling and dependency management
   b. Directed Acyclic Graphs (DAGs)
   c. Designing robust workflows using tools like Apache Airflow
   d. Reliability concepts (idempotency, retries, and backfills)
   e. Version control and Continuous Integration and Continuous Deployment (CI/CD) for data engineering workflows
7. Containerization
   a. Introduction to containers and image-based deployment (e.g., Docker)
   b. Managing dependencies and environment reproducibility
   c. Connecting multiple containers (e.g., Docker Compose)
8. Machine learning workflows in data engineering
   a. Using MLOps to serve data and manage artifacts
   b. Integrating pre-trained models for data enrichment (batch inference vs. real-time)

   c. Tracking and versioning model artifacts within a pipeline environment
9. Data governance and security
   a. Data privacy regulations (e.g., GDPR, CCPA) and compliance basics
   b. Role-Based Access Control (RBAC) and least privilege principles in data pipelines
   c. Identifying and masking Personally Identifiable Information (PII) during transformation
10. Social and ethical considerations
   a. Ethical implications of broad data collection and automated decision-making
   b. Identifying and mitigating bias within data pipelines and downstream models
   c. Environmental impact and sustainability of large-scale cloud infrastructure
   d. Promoting transparency and accountability in data workflows

Lab Content

1. Environment setup and containerization
   a. Installing and configuring a local development environment
   b. Writing Dockerfiles to containerize Python and SQL applications
   c. Running multi-container applications using basic container orchestration tools
2. Data storage and modeling
   a. Provisioning an open-source relational database (e.g., PostgreSQL)
   b. Interacting with a cloud object storage bucket
   c. Designing and implementing a star schema for an analytical dataset
   d. Writing SQL queries to aggregate and analyze data
3. Developing ETL pipelines
   a. Writing Python scripts to extract data from public APIs and flat files
   b. Transforming and cleaning raw data using Python libraries (e.g., Pandas)
   c. Loading transformed data into a target database
   d. Adding data quality assertions (e.g., checking row counts or uniqueness) to a data transformation script
   e. Writing SQL queries using window functions and CTEs to perform analytical transformations
   f. Restructuring a flat dataset into Bronze, Silver, and Gold layers to practice the medallion architecture pattern
   g. Identifying and masking Personally Identifiable Information (PII) before loading the data into the target database
4. Workflow orchestration
   a. Installing and configuring an orchestration tool (e.g., Apache Airflow)
   b. Converting standalone ETL scripts into a scheduled DAG
   c. Implementing error handling, alerts, and retries in the workflow
   d. Tracking orchestration scripts with version control and demonstrating a repeatable backfill
5. Integrating pre-trained models into data workflows
   a. Loading a pre-trained ML model (e.g., Hugging Face sentiment analysis) into a Python transformation script
   b. Applying the model to a batch dataset to generate predictions and enrich the data
6. Capstone project
   a. Designing an end-to-end automated data workflow based on a real-world scenario
   b. Integrating ingestion, storage, orchestration, and a pre-trained ML model for batch data enrichment
   c. Documenting pipeline architecture and deployment instructions

Special Facilities and/or Equipment

1. The college will provide access to a computer laboratory with Python and an IDE installed, with sufficient privileges to allow students to install Python packages.
2. The college will provide a website or course management system with an assignment posting component (through which all lab assignments are to be submitted) and a forum component (where students can discuss course material and receive help from the instructor). This applies to all sections, including on-campus (i.e., face-to-face) offerings.
3. When taught online, the college will provide a fully functional and maintained course management system through which the instructor and students can interact.
4. When taught online, students must have currently existing email accounts and ongoing access to computers with internet capabilities.

Method(s) of Evaluation

Methods of Evaluation may include but are not limited to the following:

Tests and quizzes
Lab notebook
Written laboratory assignments which include source code, sample runs, and documentation
Reflective papers
Final examination or project

Method(s) of Instruction

Methods of Instruction may include but are not limited to the following:

Instructor-authored lectures which include technical foundations, theoretical motivation, and coding implementation of data engineering principles
Detailed review of assignments which includes solutions and specific comments on the student submissions
Discussion which engages students and instructor in an ongoing dialog about data engineering
Instructor-authored labs that rigorously demonstrate a student's ability to implement data engineering objectives

Representative Text(s) and Other Materials

Reis, Joe, and Matt Housley. Fundamentals of Data Engineering. 2022.

Nwokwu, Chisom. Data Engineering for Beginners. 2025.

Types and/or Examples of Required Reading, Writing, and Outside of Class Assignments

1. Reading
    a. Textbook assigned reading averaging 30 pages per week
    b. Reading the supplied handouts and modules averaging 10 pages per week
    c. Reading online resources as directed by instructor though links pertinent to programming
    d. Reading library and reference material directed by instructor through course handouts
2. Writing
    a. Writing technical prose documentation that supports and describes the programs that are submitted for grades

Discipline(s)

Computer Science