October 26, 2025

▣ **강의명:** CSCI E-103: 재현 가능한 머신러닝

▣ **주차:** Lecture 01

▣ **교수명:** Anindita Mahapatra
Eric Gieseke

▣ **목적:** Lecture 01의 핵심 개념 학습

□ 핵심 요약

본 문서는 1주차 '데이터 엔지니어링 입문' 강의의 핵심 내용을 정리합니다. 데이터 엔지니어링의 정의와 필요성부터 시작하여, 빅데이터를 특징짓는 5V, 다양한 데이터 직군(페르소나), 그리고 데이터 처리 시스템(OLTP vs OLAP, Hadoop vs Spark)의 변천사를 다룹니다. 또한, Databricks 플랫폼을 사용한 첫 번째 실습(Lab 0)의 상세한 설정 및 실행 가이드를 포함합니다.

Contents

1 과정 소개 및 물류 (Course Logistics) 2
1.1 과목 개요 2
1.2 학습 플랫폼 및 자료 2
1.3 수업 구성 및 평가 2
2 핵심 용어 정리 (Key Terminology) 3
3 왜 데이터 엔지니어링인가? (Motivation) 5
3.1 데이터, 그 이상의 가치 5
3.2 데이터 엔지니어링의 핵심 역할 5
4 데이터 엔지니어와 페르소나 (DE Data Personas) 6
4.1 데이터 엔지니어의 위치 6
4.2 데이터 페르소나 (Data Personas) 6
5 핵심 개념 1: 빅데이터의 5가지 특징 (The 5 V's) 7

6 핵심 개념 2: 데이터 처리 방식 비교 **8**
6.1 OLTP vs. OLAP 8
6.2 SQL (ACID) vs. NoSQL (BASE) 8
6.3 CAP 이론 (CAP Theorem) 9
7 데이터 플랫폼의 진화 (Evolution) **10**
7.1 1세대: 데이터 웨어하우스 (Data Warehouse, 1980s) 10
7.2 2세대: 하둡과 데이터 레이크 (Hadoop & Data Lake, 2010s) 10
7.3 3세대: 스파크 (Spark, 2012) 10
7.4 4세대: 레이크하우스 (Lakehouse, 2020s) 11
7.5 클라우드 플랫폼 모델 (IaaS, PaaS, SaaS) 11
8 실습 0: Databricks 시작하기 (Lab 0 Guide) **12**
8.1 목표 12
8.2 단계 1: 실습 환경 설정 (Lab Setup) 12
8.3 단계 2: 컴퓨트 연결 (Connect Compute) 12
8.4 단계 3: 노트북 실행 및 매직 커맨드 12
8.5 단계 4: Spark DataFrame 기초 (Python) 13
8.6 단계 5: 데이터 탐색 (SQL) 13
9 1주차 학습 체크리스트 (Checklist) **14**

1 과정 소개 및 물류 (Course Logistics)

1.1 과목 개요

- **과목명:** CSCI E-103: 비즈니스 과제 해결을 위한 분석 데이터 엔지니어링
- **1주차 주제:** 데이터 엔지니어링 입문
- **핵심 목표:** 원시 데이터(Raw Data)를 비즈니스 가치가 있는 통찰(Valuable Insights)로 바꾸는 전 과정을 이해하고, 최신 도구를 사용하여 데이터 파이프라인을 구축하는 기술을 학습합니다.

1.2 학습 플랫폼 및 자료

- **Canvas:** 강의 슬라이드, 과제, 강의 계획서 등 모든 공식 자료가 게시됩니다.
- **Slack:** 실시간 소통, 질의응답, 토론을 위한 주 채널입니다.
- **교재:**
  - **주교재:** *Simplifying Data Engineering and Analytics with Delta* (본 과정 강사가 저술)
  - **부교재:** *Spark - The Definitive Guide*
  - **참고:** 두 교재 모두 하버드 도서관을 통해 무료로 전자책을 이용할 수 있습니다.
- **실습 플랫폼: Databricks Platform (무료 버전)**을 사용합니다.

1.3 수업 구성 및 평가

- **수업 구성:** 전반부는 이론 강의, 후반부는 실습(Lab)으로 구성됩니다.
- **섹션 (Section):** 목요일 저녁(ET 기준)에 진행되며, 과제 도움말 및 복습 세션으로 운영됩니다. (녹화본 제공)
- **오피스 아워:** Calendly 링크를 통해 교수 및 조교진(TA)과 1:1 면담을 예약할 수 있습니다.
- **평가 비율:**
  - 4개 과제 (Assignments): 40%
  - 2개 케이스 스터디 (Case Studies): 20%
  - 2개 퀴즈 (Quizzes): 8%
  - 기말 발표 (Final Presentations): 30% (그룹 프로젝트)
  - 참여도 (Participation): 2% (Slack 활동, 출석 등)
- **지각 정책:** 과제 제출 지각 시 하루당 2점 감점되며, 최대 10점까지 감점됩니다.

2 핵심 용어 정리 (Key Terminology)

본 강의에서 자주 사용되는 핵심 용어들을 미리 정리합니다.

Table 1: 데이터 엔지니어링 핵심 용어

용어	원어 (English)	쉬운 설명 (비유)
데이터 엔지니어링	Data Engineering	'데이터 정유 공장'을 짓는 일. 원유(원시 데이터)를 받아 정제(처리)하여 휘발유(분석용 데이터)로 만듭니다.
ETL	Extract, Transform, Load	'데이터 이사'. 여러 곳(Extract)에서 데이터를 모아, 쓸모 있게 다듬고(Transform), 새 집(Load)에 저장합니다.
OLTP	Online Transactional Processing	'가게 계산대(POS)'. 실시간으로 발생하는 개별 거래(결제, 등록)를 빠르게 처리하는 시스템입니다.
OLAP	Online Analytical Processing	'본사 분석실'. 가게 계산대에서 쌓인 과거 매출 전체를 모아 분기별/지역별로 분석하는 시스템입니다.
ACID	Atomicity, Consistency, Isolation, Durability	'은행 송금' 원칙. 거래는 100% 성공하거나 100% 실패(롤백)해야 하며(원자성), 데이터는 항상 정확해야 합니다(일관성). (SQL DB)
BASE	Basically Available, Soft State, Eventual Consistency	'소셜 미디어 좋아요' 원칙. 잠시 불일치하더라도(최종 일관성), 서비스는 절대 멈추면 안 됩니다(가용성). (NoSQL DB)
CAP 이론	CAP Theorem	(일관성, 가용성, 분할 감내) 중 2가지만 선택 가능. 분산 시스템의 근본적인 트레이드오프입니다.
데이터 웨어하우스	Data Warehouse (DW)	'잘 정리된 창고'. 주로 정형 데이터(SQL)를 BI 리포팅 목적으로 저장합니다.
데이터 레이크	Data Lake	'거대한 호수'. 모든 형태(정형, 비정형)의 데이터를 원시 상태 그대로 저장합니다. (Hadoop 기반)
레이크하우스	Lakehouse	DW(안정성, ACID)와 Data Lake(유연성)의 장점을 결합한 현대적 아키텍처입니다. (Databricks, Delta Lake)
하둡 (Hadoop)	Hadoop	2세대 빅데이터 플랫폼. 저렴한 하드웨어 여러 대로 데이터를 분산 저장(HDFS) 및 처리(MapReduce)합니다.
스파크 (Spark)	Spark	3세대 빅데이터 플랫폼. '인메모리(In-Memory)' 처리로 Hadoop보다 100배 빠릅니다. (Databricks의 엔진)
데이터 페르소나	Data Persona	데이터 관련 직군. (데이터 엔지니어, 데이터 과학자, BI 분석가, MLOps 엔지니어 등)
메달리온 아키텍처	Medallion Architecture	데이터 정제 단계를 나누는 방식. **Bronze**(원시) → **Silver**(정제) → **Gold**(집계/분석용).

3 왜 데이터 엔지니어링인가? (Motivation)

3.1 데이터, 그 이상의 가치

우리는 **빅데이터(Big Data) + 클라우드(Cloud) + 인공지능(AI)**이 융합되는 거대한 혁신의 교차점에 서 있습니다. 과거 "석유"가 산업 시대를 이끌었다면, 오늘날 **"데이터는 새로운 석유"**로 불리며 디지털 시대를 주도합니다.

- 데이터는 머신러닝(ML)과 AI 모델을 학습시키는 핵심 **연료**입니다.
- 모든 산업(금융, 헬스케어, 제조, 엔터테인먼트 등)이 데이터를 활용해 경쟁 우위를 확보하려 하며, 스스로를 '디지털 기업'으로 재정의하고 있습니다.
- 단순히 데이터를 수집하는 것을 넘어, 데이터를 '활용 가능하게' 만드는 능력이 기업의 생존을 좌우합니다.

3.2 데이터 엔지니어링의 핵심 역할

데이터 엔지니어링의 본질은 **"원시 데이터(Raw Data)를 가치 있는 통찰(Valuable Insights)로 바꾸는 것"**입니다.

비즈니스는 데이터를 원하지 않습니다. 비즈니스는 '통찰'을 원합니다. 하지만 이 통찰을 얻기까지의 과정은 복잡하고 어렵습니다. 데이터 엔지니어는 이 지저분하고 복잡한 과정을 책임지는 전문가입니다.

비즈니스의 가장 큰 요구사항은 **"통찰의 속도 (Speed to Insights)"**입니다. 즉, 데이터가 발생하는 순간부터 분석가가 의미 있는 결론을 내리기까지 걸리는 시간을 최소화하는 것이 데이터 엔지니어링의 핵심 목표입니다.

□ **예제:**

데이터의 정제 비유 (탄소 → 다이아몬드)
1. **원시 데이터 (Raw):** 쓸모없는 탄소 덩어리와 같습니다. (e.g., 앱 로그 파일) 2. **정렬된 데이터 (Sorted):** 데이터를 분류합니다. (e.g., 날짜/사용자별 로그 정렬) 3. **정리된 데이터 (Arranged):** 필요한 정보만 추출하고 정제합니다. (e.g., 오류 로그 제거, 사용자 ID 매핑) 4. **시각화된 데이터 (Visualized):** 차트나 그래프로 표현합니다. (e.g., 시간대별 접속자 수 그래프) 5. **스토리 (Story):** 비즈니스 의미를 도출합니다. (e.g., "새벽 3시에 접속 오류가 급증하며, 이는 특정 지역 사용자에게 집중됨")

4 데이터 엔지니어와 페르소나 (DE Data Personas)

4.1 데이터 엔지니어의 위치

데이터 엔지니어는 종종 **소프트웨어 엔지니어(Software Engineer)**와 **데이터 과학자(Data Scientist)**의 교집합에 위치한다고 말합니다.

- **소프트웨어 엔지니어링 역량:** 안정적인 시스템 설계, DevOps, 분산 컴퓨팅, 서비스 개발.
- **데이터 과학 역량:** 데이터 모델링, 머신러닝 알고리즘 이해, 비즈니스 인텔리전스(BI).
- **데이터 엔지니어 고유 역량:** 대규모 **데이터 파이프라인(ETL)** 구축, Spark/Kafka 등 분산 처리 도구 활용.

4.2 데이터 페르소나 (Data Personas)

데이터를 다루는 조직에는 다양한 역할(페르소나)이 존재하며, 기말 그룹 프로젝트에서 이 역할들을 맡아 수행하게 됩니다.

- **데이터 엔지니어 (Data Engineer): 파이프라인**을 구축하고 유지보수합니다. 데이터 수집, 변환, 적재(ETL)를 책임집니다.
- **BI 분석가 (BI Analyst): SQL 기반 리포팅**에 집중합니다. 정제된 데이터를 사용하여 대시보드를 만듭니다.
- **데이터 과학자 (Data Scientist): 탐색적 데이터 분석(EDA)** 및 **ML 모델링**을 수행합니다. 통계 지식을 바탕으로 미래를 예측합니다.
- **MLOps 엔지니어 (MLOps Engineer): 인프라 자동화**를 담당합니다. ML 모델이 안정적으로 배포되고 모니터링되도록 관리합니다.
- **데이터 리더 (Data Leader):** 최고 데이터 책임자(CDO) 등. 데이터 전략과 거버넌스를 총괄합니다.

title=현실의 머신러닝: 빙산의 일각 흔히 AI/ML이라고 하면 '모델 코드' 자체에만 집중하지만, 이는 전체 시스템의 극히 일부에 불과합니다.
실제 ML 시스템의 90% 이상은 데이터 엔지니어링 영역입니다.
모델 코드를 둘러싼 **데이터 수집, 데이터 검증, 피처 추출, 리소스 관리, 서빙 인프라, 모니터링** 등의 복잡한 인프라가 훨씬 더 많은 노력을 필요로 합니다.

5 핵심 개념 1: 빅데이터의 5가지 특징 (The 5 V's)

빅데이터는 단순히 '많은 데이터'가 아닙니다. 다음 5가지 특징(5V)으로 정의됩니다.

1. **Volume (규모):** 데이터의 물리적인 크기 (e.g., Terabytes, Petabytes). 한 대의 컴퓨터에 저장할 수 없는 규모.
2. **Velocity (속도):** 데이터가 생성되고 처리되는 속도.
   - **배치 (Batch):** 데이터를 모았다가 주기적으로 한꺼번에 처리 (e.g., 일일 정산).
   - **스트리밍 (Streaming):** 데이터가 발생하는 즉시 실시간으로 처리 (e.g., 주가 변동, 사기 탐지).
3. **Variety (다양성):** 데이터의 형태.
   - **정형 (Structured):** 스키마(구조)가 명확함 (e.g., SQL 데이터베이스의 행과 열).
   - **반정형 (Semi-Structured):** 스키마가 데이터 내에 포함됨 (e.g., JSON, XML).
   - **비정형 (Unstructured):** 스키마가 없음 (e.g., 이미지, 오디오, 비디오, 텍스트 문서). **현대 기업 데이터의 80-90%를 차지**하며, AI/ML의 핵심 재료입니다.
4. **Veracity (정확성):** 데이터의 품질과 신뢰도. "쓰레기가 들어가면 쓰레기가 나온다 (GIGO)" 원칙. 데이터의 노이즈, 누락, 편향 등을 관리해야 합니다.
5. **Value (가치):** 데이터에서 추출할 수 있는 비즈니스 임팩트. 5V 중 가장 중요합니다.

title=데이터 온도 (Data Temperature) 데이터의 활용 빈도와 속도에 따라 '온도'를 구분합니다.

- **Hot Data:** 실시간으로 자주 접근하고 즉시 처리해야 하는 데이터 (e.g., 현재 접속자 수).
- **Cold Data:** 가끔 접근하며 배치 처리되는 데이터 (e.g., 1년 전 로그).

6 핵심 개념 2: 데이터 처리 방식 비교

6.1 OLTP vs. OLAP

데이터베이스 시스템은 목적에 따라 크게 OLTP와 OLAP로 나뉩니다. 이는 데이터 엔지니어링에서 가장 기본이 되는 구분입니다.

□ **예제:**

비유: 마트(Mart)

- **OLTP (Online Transactional Processing):** 마트의 **'계산대(POS)'**.
- **OLAP (Online Analytical Processing):** 마트 **'본사의 분석실'**.

Table 2: *OLTP vs. OLAP* 비교

특징	OLTP (운영 시스템)	OLAP (분석 시스템)
목적	실시간 거래(트랜잭션) 처리	비즈니스 분석 및 의사결정
비유	(마트) 계산대	(마트) 본사 분석실
데이터	현재, 실시간, 원본 데이터	과거, 집계된 데이터
주요 작업	빠른 읽기/쓰기/수정 (Insert, Update)	복잡하고 무거운 조회 (Select)
사용자	현장 직원, 고객	분석가, 경영진
예시	ATM 출금, 쇼핑몰 장바구니, 좌석 예약	분기별 매출 보고서, 사용자 이탈률 분석

데이터 파이프라인(ETL)은 OLTP 시스템(운영 DB)에서 데이터를 추출(Extract)하여, 분석하기 좋은 형태로 변환(Transform)한 뒤, OLAP 시스템(데이터 웨어하우스)에 적재(Load)하는 과정을 의미합니다.

6.2 SQL (ACID) vs. NoSQL (BASE)

데이터를 저장하는 방식은 데이터의 일관성 요구 수준에 따라 나뉩니다.

- **SQL (관계형, ACID): ACID** 속성을 보장합니다. (원자성, **일관성**, 고립성, 지속성).
  - **비유 (은행 송금):** 100만 원 송금 시, 내 계좌 -100만 원과 상대 계좌 +100만 원은 **절대적으로 일관성**을 유지해야 합니다. 둘 중 하나만 성공하면 재앙입니다.
- **NoSQL (비관계형, BASE): BASE** 속성을 따릅니다. (기본적 **가용성**, 소프트 상태, **최종적 일관성**).
  - **비유 (소셜 미디어 '좋아요'):** '좋아요' 수가 1초 동안 친구에게 100개로 보이고 나에게 101개로 보여도 치명적이지 않습니다. 대신, 서비스가 절대 다운되지 않는 것(**가용성**)이 더 중요합니다.

6.3 CAP 이론 (CAP Theorem)

분산 데이터 시스템은 다음 세 가지 속성 중 최대 **두 가지만** 동시에 만족시킬 수 있습니다.

- **C**onsistency (일관성): 모든 노드가 동시에 같은 데이터를 보여줌.
- **A**vailability (가용성): 모든 요청에 항상 응답함 (에러 X).
- **P**artition Tolerance (분할 감내): 노드 간 통신이 끊겨도(네트워크 장애) 시스템이 동작함.

현대 분산 시스템은 네트워크 장애(P)를 기본 전제로 합니다. 따라서 CP 또는 AP 시스템을 선택하게 됩니다.

- **CP 시스템 (일관성 중요):** 전통적인 RDBMS.
- **AP 시스템 (가용성 중요):** NoSQL (e.g., Cassandra, DynamoDB).

7 데이터 플랫폼의 진화 (Evolution)

7.1 1세대: 데이터 웨어하우스 (Data Warehouse, 1980s)

- **목적:** 비즈니스 인텔리전스(BI), 리포팅.
- **데이터:** 정형 데이터(SQL)만 처리 가능.
- **한계:** 고가의 전용 하드웨어 필요, 비정형 데이터(이미지, 텍스트) 처리 불가, 확장성 부족.

7.2 2세대: 하둡과 데이터 레이크 (Hadoop & Data Lake, 2010s)

- **Hadoop (하둡):** 저렴한 **범용 하드웨어(Commodity Hardware)** 여러 대를 묶어 대용량 데이터를 처리하는 오픈소스 프레임워크.
- **데이터 레이크 (Data Lake):** 모든 형태(정형, 비정형)의 데이터를 원시 상태 그대로 저장하는 거대 저장소.
- **핵심 구성:**
  - **HDFS (Hadoop Distributed File System):** 분산 저장 시스템. (데이터를 3중 복제하여 안정성 확보)
  - **YARN (Yet Another Resource Negotiator):** 클러스터 자원 관리.
  - **MapReduce (맵리듀스):** 분산 처리 엔진.

매우 중요:

Hadoop(MapReduce)이 느린 이유
MapReduce는 **'Map(분할 처리)'** 단계와 **'Reduce(결과 취합)'** 단계로 나뉩니다. 이때 **각 단계가 끝날 때마다 결과를 HDFS(디스크)에 썼다가 다시 읽어옵니다.** 디스크 I/O는 RAM 접근보다 수천 배 느리기 때문에 전체 처리 속도가 매우 느렸습니다.

7.3 3세대: 스파크 (Spark, 2012)

- **Spark (스파크):** Hadoop의 단점을 보완한 통합 분석 엔진. Hadoop MapReduce보다 최대 100배 빠릅니다.
- **핵심 추상화:** RDD (탄력적 분산 데이터셋) → **DataFrame/DataSet** (최신 고수준 API. SQL처럼 사용 가능).

□ 핵심 요약

Spark가 압도적으로 빠른 이유: 인메모리(In-Memory) 처리
Spark는 MapReduce처럼 매번 디스크에 쓰지 않고, 가능한 한 모든 데이터를 **RAM(메모리)**에 올려놓고 처리합니다.
또한 **DAG(방향성 비순환 그래프)**라는 실행 계획을 세우고, 실제 'Action' (e.g., 'count()', 'collect()'처럼 결과를 봐야 하는 명령)이 호출될 때까지 변환(Transformation)을 지연시켰다가 한꺼번에 처리합니다 (Lazy Evaluation).

7.4 4세대: 레이크하우스 (Lakehouse, 2020s)

- **Data Lake** (유연성, 저비용, 비정형 처리)와 **Data Warehouse** (안정성, ACID 트랜잭션, 거버넌스)의 장점을 결합한 현대적 아키텍처입니다.
- Databricks의 **Delta Lake**가 대표적인 레이크하우스 기술입니다. (Parquet 파일 형식 + 트랜잭션 로그)

7.5 클라우드 플랫폼 모델 (IaaS, PaaS, SaaS)

□ **예제:**

비유: 피자(Pizza) 만들기

- **IaaS (Infrastructure as a Service): 인프라만** 제공. (오븐, 밀가루, 소스 재료를 사서 처음부터 끝까지 직접 요리). 예: AWS EC2, GCP Compute Engine.
- **PaaS (Platform as a Service): 플랫폼** 제공. (배달된 피자 도우와 소스 위에 토핑만 내가 올림). 예: Heroku, AWS Elastic Beanstalk.
- **SaaS (Software as a Service): 완성된 소프트웨어** 제공. (완성된 피자를 주문해서 먹기). 예: Gmail, **Databricks**.

8 실습 0: Databricks 시작하기 (Lab 0 Guide)

8.1 목표

Databricks 무료 버전을 사용하여 클러스터에 연결하고, 노트북을 실행하며, Spark DataFrame 및 SQL을 사용한 기초적인 데이터 조작을 실습합니다.

매우 중요:

계정 확인: "Free Edition" vs. "Trial"

- 본 과정은 **"Databricks Free Edition" (무료 버전)**을 사용합니다.
- **"Trial" (체험판)** 계정은 14일 후 만료되며, 유료 플랜 기능이 포함되어 있어 실습 환경이 다를 수 있습니다.
- 로그인 시, 화면 **좌측 상단에 "Free Edition"** 로고가 있는지 반드시 확인하세요.
- Trial 계정으로 진행 시, 실습 노트북(파일)이 정상적으로 로드되지 않는 오류가 발생할 수 있습니다.

8.2 단계 1: 실습 환경 설정 (Lab Setup)

1. Databricks Free Edition에 로그인합니다. 2. 왼쪽 탐색 메뉴에서 **Workspace (작업 공간)**를 클릭합니다. 3. **Home** 또는 자신의 사용자 이름으로 된 폴더를 찾습니다. 4. 해당 폴더 이름 옆의 화살표(또는 점 3개)를 클릭하고 **Create** → **Folder**를 선택하여 'labs'와 같은 새 폴더를 생성합니다. 5. 방금 생성한 'labs' 폴더로 이동합니다. 6. 'labs' 폴더의 빈 공간에서 마우스 우클릭(또는 폴더 이름 옆 메뉴) 후 **Import (가져오기)**를 선택합니다. 7. 강의 자료로 제공된 **실습용 `.zip` 파일** (e.g., 'Lab0.zip')을 대화 상자에 드래그 앤 드롭합니다. Databricks가 자동으로 압축을 해제하고 노트북 파일들을 폴더에 생성합니다.

8.3 단계 2: 컴퓨트 연결 (Connect Compute)

1. Import된 노트북 파일 중 `00_Initialize` (또는 첫 번째) 노트북을 클릭하여 엽니다. 2. 노트북 우측 상단에 있는 **Connect (연결)** 버튼을 클릭합니다. 3. 드롭다운 메뉴에서 **Serverless** (또는 사용 가능한 기본 클러스터)를 선택합니다. 4. 잠시 기다리면 클러스터가 준비되고, 버튼이 녹색(Running)으로 바뀝니다.

8.4 단계 3: 노트북 실행 및 매직 커맨드

매직 커맨드(Magic Commands)는 노트북 셀의 기본 언어를 임시로 변경하는 명령어입니다.

- **`%python`**: (기본값) 파이썬 코드를 실행합니다.
- **`%sql`**: SQL 쿼리를 실행합니다.
- **`%md`**: 마크다운(Markdown) 텍스트를 렌더링합니다.
- **`%sh`**: 리눅스 쉘(Shell) 명령어를 실행합니다.

- **`%fs`**: Databricks File System(DBFS) 명령어를 실행합니다 (e.g., `%fs ls /`).

`00_Initialize` 노트북의 첫 번째 셀(아마도 `%sql` 셀)을 실행하여 실습에 필요한 **Catalog**(최상위 데이터 컨테이너)와 **Schema**(데이터베이스)를 생성합니다. (단축키: Shift + Enter)

8.5 단계 4: Spark DataFrame 기초 (Python)

Spark의 핵심 데이터 구조인 DataFrame을 다뤄봅니다.

```python
# 에Databricks 내장된예제데이터셋경로
filePath = "/databricks-datasets/bikeSharing/data-001/day.csv"

# 를Spark 사용해 CSV 파일을으로 DataFrame 읽기
# .option("header", "true"): 첫번째줄을헤더컬럼명 ()로 사용
# .option("inferSchema", "true"): 데이터타입을자동으로추론
df = spark.read.format("csv") \
              .option("header", "true") \
              .option("inferSchema", "true") \
              .load(filePath)

# Databricks 전용함수 'display로' 결과를표로시각화
display(df)
```

Listing 1: Python으로 CSV 파일 읽고 DataFrame 생성하기

8.6 단계 5: 데이터 탐색 (SQL)

`%sql` 매직 커맨드를 사용하여 SQL로 데이터를 직접 조회할 수 있습니다.

```sql
-- 실습용으로생성한카탈로그및스키마사용예시 ()
USE CATALOG csci_e103;
USE SCHEMA lab01;

-- 으로DataFrame 생성한테이블조회
-- 앞( 단계에서 df.write.saveAsTable("bike_data")를 실행했다고가정 )
SELECT * FROM bike_data WHERE season = '1' LIMIT 10;
```

Listing 2: SQL로 테이블 조회하기

title=Delta Lake: Databricks의 기본 형식 Databricks에서 `CREATE TABLE`이나 `saveAsTable`을 사용하면 기본적으로 **Delta Lake** 형식으로 저장됩니다.
Delta Lake는 Parquet 파일 포맷에 **ACID 트랜잭션 로그**를 추가한 것입니다. 이를 통해 데이터 레이크에서도 안정적인 데이터 변경(Update, Delete, Merge)이 가능해져 '레이크하우스' 아키텍처를 구현할 수 있습니다.

9 1주차 학습 체크리스트 (Checklist)

☐ Databricks **Free Edition** 계정을 생성했는가? (Trial 계정이 아닌지 재확인)

☐ 실습 0 (Lab 0) `.zip` 파일을 Workspace에 성공적으로 Import 했는가?

☐ Serverless 컴퓨트에 연결하고 `00_Initialize` 노트북의 첫 번째 셀을 실행했는가?

☐ 데이터 엔지니어링의 정의를 "원시 데이터를 가치 있는 통찰로 바꾸는 것"이라고 설명할 수 있는가?

☐ **OLTP**와 **OLAP**의 차이점을 비유(e.g., 가게 계산대 vs. 본사 분석실)를 들어 설명할 수 있는가?

☐ **Hadoop(MapReduce)**과 **Spark**의 속도 차이의 핵심 원인이 **디스크 I/O** vs. **인메모리 처리**임을 이해했는가?

☐ 빅데이터의 5V (Volume, Velocity, **Variety**, Veracity, Value)가 무엇인지 아는가?

☐ 데이터 페르소나(DE, DS, BI, MLOps)의 역할 차이를 구분할 수 있는가?