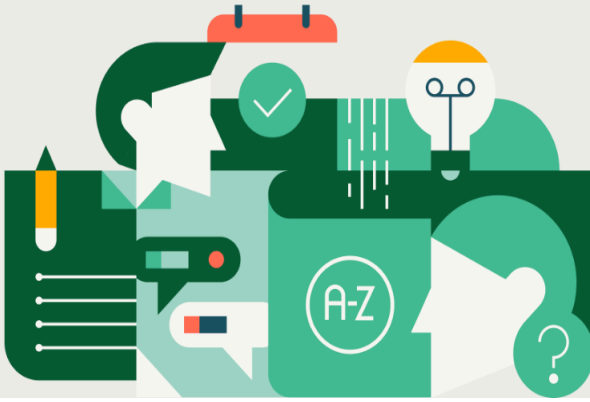
databricks

Databricks 시험 가이드

# Databricks Certified Data Engineer Associate 재인증 시험




시험 가이드 피드백 제공

## 이 시험 가이드의 목적

이 시험 가이드는 시험의 개요와 출제 범위를 제공하여 응시 준비 상태를 판단하는 데 도움을 주는 것을 목적으로 합니다. 이 문서는 시험에 변경 사항이 있을 때마다(그리고 해당 변경 사항이 적용되는 시점에) 업데이트되므로 미리 대비할 수 있습니다. 본 문서는 2026년 6월 15일 기준 시험 버전을 다룹니다.

## 대상 응시자 설명

이 Databricks Certified Data Engineer Associate 재인증 시험은 Databricks Data Intelligence Platform을 활용하여 기초적인 데이터 엔지니어링 작업을 수행하는 능력을 평가합니다. 이 시험은 Databricks Certified Data Engineer Associate가 데이터 수집, 데이터 로딩, 데이터 변환 및 모델링과 관련된 작업(예: PySpark / SQL을 사용한 Extract, Transform, Load(ETL) 작업 수행, Lakeflow Jobs 및 CI/CD 활용)에 대해 갖춘 지식을 평가합니다. 마지막으로, 이 시험은 문제 해결, 모니터링, 최적화 기법에 대한 이해와 Databricks Platform 내에서 거버넌스와 보안을 구현하는 방법에 대한 응시자의 지식을 평가합니다.

## 시험 정보

- 채점 문항 수: 채점되는 객관식 문항 25개.
- 제한 시간: 60분.
- 응시료: 미화 200달러(200 USD)
- 응시 방식: 온라인 감독(Online Proctored) 전용
- 시험 보조 도구: 허용되지 않음.
- 사전 요건: 이 재인증 시험은 이미 만료되었거나 만료 2개월 이내인 기존 Databricks Certified Data Engineer Associate 자격을 보유한 사람에게 제공됩니다. 응시자는 권장 교육을 이수하고 Databricks Platform에 대해 최소 12개월의 실무 경험을 갖출 것을 적극 권장합니다.
- 유효 기간: 2년.
- 재인증: 응시자는 해당 연도에 제공되는 현행 시험 또는 특정 재인증 버전을 성공적으로 완료하여 2년마다 자격을 갱신해야 합니다. 시험에 응시할 준비를 하려면 시험 웹페이지의 “Getting Ready for the Exam” 섹션을 검토하시기 바랍니다.

## 권장 교육

- 강사 주도(Instructor-led): Data Engineering with Databricks
- 자기 주도(Self-paced, Databricks Academy에서 제공):
    - Data Ingestion with Lakeflow Connect
    - Deploy Workloads with Lakeflow Jobs
    - DevOps Essentials for Data Engineering
    - Data Interoperability with Unity Catalog
    - Build Data Pipelines with Lakeflow Spark Declarative pipeline
    - Get Started with Data Governance on Databricks

## 시험 개요

섹션 **1:** 데이터 수집 및 로딩 **(32%)**

- COPY INTO를 사용하여 클라우드 오브젝트 스토리지에서 Unity Catalog로 관리되는 Delta 테이블로 파일을 증분 방식으로 수집합니다.
- Auto Loader를 사용하여 파일을 Unity Catalog로 관리되는 Delta 테이블에 증분 방식으로 수집합니다
- 다양한 엔터프라이즈 소스에서 데이터를 Unity Catalog로 관리되는 테이블에 안정적으로 수집하도록 Lakeflow Connect를 구성합니다.
- 노트북에서 JDBC/ODBC 또는 REST 클라이언트를 사용하여 데이터를 클라우드 스토리지에 적재하거나 Unity Catalog로 관리되는 테이블에 직접 적재합니다.
- 데이터 볼륨, 빈도, 데이터 유형, 거버넌스 요구 사항에 따라 Auto Loader, Lakeflow Connect 파트너 커넥터, 기타 수집 방식 중에서 우선순위를 정합니다.
- 반정형 및 비정형 데이터(예: JSON 및 중첩 데이터)를 Lakeflow Connect 및 managed connector를 통해 Delta 테이블에 수집합니다.

섹션 **2:** 데이터 변환 및 모델링 **(24%)**

- PySpark/SQL로 bronze 테이블을 읽어 데이터 정제를 적용하고(null 처리, 데이터 유형 표준화), 그 결과를 silver 테이블에 기록합니다.
- 일반적인 Spark 튜닝 파라미터를 파악하고 이들이 성능에 미치는 전반적인 영향을 이해합니다.
- Unity Catalog의 Gold 계층 객체(materialized view, view, streaming table, 일반 테이블)와 이들의 BI/분석 용도를 구분합니다.
- 신뢰할 수 있는 Silver 및 Gold 데이터셋을 보장하기 위해 데이터 품질 검사와 검증 규칙을 적용합니다

섹션 **3: Lakeflow Jobs** 활용 **(12%)**

- 여러 단계로 이루어진 데이터 워크플로를 오케스트레이션하기 위해 Lakeflow Jobs를 사용하여 제어 흐름(태스크 재시도, 분기, 반복)을 구현합니다
- DAG 기반 워크플로를 사용하여 Lakeflow Jobs에서 공통 태스크 유형과 종속성을 구성합니다
- scheduled, file arrival, table update, continuous 트리거를 포함하여 Lakeflow Jobs 일정과 트리거를 구성합니다.

섹션 **4: CI/CD** 구현 **(12%)**

- Git Folders를 사용하여 Databricks workspace UI에서 코드를 관리합니다: 브랜치 생성/전환, 커밋, 푸시, pull request 열기.
- Declarative Automation Bundle 환경 대상(dev, test, prod)을 사용하여 코드베이스를 여러 환경에 걸쳐 승격합니다.
- Databricks CLI를 사용하여 Declarative Automation Bundles(DABs)를 검증하고 배포합니다

섹션 **5:** 문제 해결**,** 모니터링**,** 최적화 **(12%)**

- Lakeflow Jobs 실행 기록 보기를 사용하여 작업 성능의 추세를 파악합니다.
- Lakeflow Jobs UI를 사용하여 파이프라인 상태, 작업 상태, DAG 태스크 그래프, 실행 시간, 실패율을 모니터링합니다.
- 관리형 Delta Lake 기능으로서 Liquid Clustering 및 predictive optimization의 목적과 이점을 파악합니다.

섹션 **6:** 거버넌스 및 보안 **(8%)**

- 열 수준 마스킹과 행 수준 보안의 목적을 이해하고 이를 언제 적용해야 하는지 파악합니다.
- Unity Catalog ABAC 정책의 기능과 이것이 중앙 집중식 데이터 액세스 제어를 지원하는 방식을 이해합니다.

## 샘플 문항

이 문항들은 이전 버전 시험에서 출제되었다가 폐기된 문항입니다. 그 목적은 시험 가이드에 명시된 학습 목표를 보여 주고, 각 목표에 부합하는 샘플 문항을 제공하는 데 있습니다. 시험 가이드에는 시험에서 다뤄질 수 있는 목표가 나열되어 있습니다. 인증 시험을 준비하는 가장 좋은 방법은 시험 가이드의 시험 개요를 검토하는 것입니다.

문항 **1**
목표*: Spark UI*에서 스테이지 수준 메트릭을 해석하여 데이터 스큐, 셔플링, 디스크 스필과 같은 일반적인 성능 병목을 파악합니다.

한 데이터 엔지니어가 새로운 데이터 소스를 도입한 후 배치 작업의 실행 시간이 두 배로 늘어난 것을 발견합니다. Spark UI에서 가장 오래 걸리는 스테이지를 보면 대부분의 태스크는 30초 이내에 완료되지만, 한 태스크는 10분 넘게 걸립니다. 해당 스테이지의 태스크 요약을 보면 셔플 읽기의 최소/중앙값은 약 400MB인 반면 최대값은 5GB를 초과합니다.

작업 실행 시간을 줄이는 해결책은 무엇입니까?

A. 느린 태스크가 더 빨리 완료되도록 클러스터 크기를 늘려 실행자(executor)를 추가한다
B. 초과 크기의 파티션을 런타임에 자동으로 분할하도록 스큐 조인 처리 기능이 포함된 adaptive query execution이 활성화되어 있는지 확인한다

C. 더 적은 수의 태스크로 작업을 통합하도록 spark.sql.shuffle.partitions를 줄인다
D. 조인 전에 salt 키를 사용하여 데이터셋을 수동으로 리파티션하여 치우친 키를 고르게 분산한다

문항 **2**
목표: 반정형 및 비정형 데이터(예: *JSON* 및 중첩 데이터)를 *Lakeflow Connect* 및 *managed connector*를 통해 *Delta* 테이블에 수집합니다.

한 데이터 엔지니어가 SaaS 소스의 JSON 레코드를 Unity Catalog의 Delta 테이블에 수집하기 위해 Lakeflow Connect managed connector를 구성해야 합니다. 소스 페이로드에는 수집 실행마다 달라지는 가변 하위 키를 가진 'order_details'라는 중첩 객체 필드가 포함되어 있습니다. 엔지니어는 커넥터가 수동 스키마 업데이트 없이 새 하위 키를 대상 Delta 테이블의 추가 열로 자동으로 확장하기를 원합니다.

요구 사항을 충족하는 구성은 무엇입니까?

A. 커넥터의 수집 설정에서 schema evolution 모드를 'rescue'로 설정하여, 인식되지 않는 하위 키를 포함한 레코드가 대상 스키마를 확장하는 대신 별도의 rescued-data 열에 기록되도록 한다.
B. 대상 Delta 테이블의 TBLPROPERTIES에서 'mergeSchema' 옵션을 활성화하여, 커넥터 수준의 구성 없이 커넥터가 쓰기 시점에 들어오는 스키마 변경을 병합하도록 한다.
C. 커넥터의 수집 설정에서 schema evolution 모드를 'failOnNewColumns'로 설정하고, 새 하위 키를 반영하기 위해 매 수집 실행 후 ALTER TABLE ADD COLUMN을 수동으로 실행한다.
D. 커넥터의 수집 설정에서 schema evolution 모드를 'addNewColumns'로 설정하여, 중첩 JSON 필드에서 새로 발견된 하위 키가 대상 Delta 테이블의 최상위 열로 자동으로 승격되도록 한다.

문항 **3**
목표: 데이터 볼륨, 빈도, 데이터 유형, 거버넌스 요구 사항에 따라 *Auto Loader, Lakeflow Connect* 파트너 커넥터, 기타 수집 방식 중에서 우선순위를 정합니다.

한 데이터 엔지니어가 고객 소유의 S3 버킷에서 Databricks 감사 로그를 소비하는 다운스트림 파이프라인을 구축하고 있습니다. 스키마 추론과 체크포인팅을 구현하기 전에, 이 엔지니어는 전달 형식, 일반적인 수집 지연 시간, 파일이 덮어쓰기될 수 있는지 여부를 이해하고자 합니다.

Databricks 감사 로그 스토리지의 동작은 무엇입니까?

A. 파일은 JSON으로 전달되며 전달이 시작된 후 일반적으로 15분 이내에 이벤트가 로깅되고, 새로운 전달이 기존 파일을 덮어쓸 수 있다
B. 파일은 CSV로 전달되며 1분 미만의 지연 시간을 보장하고, 불변성을 유지하기 위해 파일이 한 번 기록되면 덮어쓰기가 발생하지 않는다
C. 파일은 Parquet로 전달되며 24시간이 지난 후 최종 일관성을 갖고, 스트리밍 수집을 단순화하기 위해 덮어쓰기가 비활성화되어 있다

D. 파일은 JSON으로 전달되며 주간 배치 주기로 전달되고, 덮어쓰기는 기존 내용을 추가 없이 완전히 대체한다

문항 **4**
목표*: Lakeflow Jobs* 실행 기록 보기를 사용하여 작업 성능의 추세를 파악합니다.

한 데이터 엔지니어가 매일 밤 실행되는 작업에 대해 Lakeflow Jobs 실행 기록 보기에서 기간 추세 차트를 활성화합니다.

이 차트를 활성화한 결과 실행 기록 보기는 어떻게 동작합니까?

A. 보기가 과거 실행 전반의 평균 실행 기간을 사용하여 기준선 경고 임계값을 설정한다.
B. 보기가 실행 기간이 평균을 초과하는 행에 시각적 표시를 적용한다.
C. 보기가 시간 경과에 따라 최근 각 실행의 기간을 나타내는 선형 차트를 표시한다.
D. 보기가 평균 실행 기간과 클러스터의 DBU 요율을 기반으로 비용 예측을 생성한다.

문항 **5**
목표*: Git Folders*를 사용하여 *Databricks workspace UI*에서 코드를 관리합니다*:* 브랜치 생성/전환*,* 커밋*,* 푸시*, pull request* 열기.

한 팀이 Databricks에서 ETL 파이프라인을 배포, 버전 관리, 오케스트레이션하는 모듈식 방법을 원하며, 이를 통해 CI/CD와 재현성을 가능하게 하고자 합니다.

이 요구 사항을 지원하는 기능은 무엇입니까?

A. Unity Catalog의 모델을 사용하여 ETL 작업을 표현하고, 각 모델이 파이프라인 코드 아티팩트를 저장하며, CI/CD가 Job 태스크에 연결된 모델 별칭을 업데이트하여 버전을 승격한다.
B. 변환 로직을 Unity Catalog Volumes에 저장된 wheel 라이브러리로 패키징하고 이를 Jobs 태스크에 바인딩하여 여러 환경에 걸쳐 결정적인(deterministic) 배포를 보장한다.
C. API 로직을 Volume에 마운트된 노트북 안에 패키징하고, Jobs API v2를 사용하여 노트북을 트리거하며, 노트북 개정 이력을 버전 관리 시스템으로 활용한다.
D. DABs를 사용하여 리소스와 코드 자산을 정의하고, 이를 Git에서 버전 관리하며, 자동화된 CI/CD 작업을 통해 여러 환경에 걸쳐 배포를 승격한다.

정답:

*1. B*
*2. D*
*3. A*
*4. C*
*5. D*