databricks

Databricks Exam Guide

Databricks Certified Data Engineer Associate

Provide Exam Guide Feedback

Purpose of this Exam Guide

The purpose of this exam guide is to give you an overview of the exam and what is covered on the exam to help you determine your exam readiness. This document will be updated whenever there are changes to an exam (and when those changes take effect), so you can be prepared. This covers the exam version as of **May 4, 2026**.

Audience Description

The Databricks Certified Data Engineer Associate certification exam assesses an individual's ability to utilise the Databricks Data Intelligence Platform to execute foundational data engineering tasks. The exam assesses the test takers' knowledge on the Data Intelligence Platform, its workspace, architecture, and capabilities, the tasks related to Data Ingestion, Data Loading, Data Transformation and Modelling- such as the ability to perform Extract, Transform, Load (ETL) tasks using PySpark, working with Lakeflow Jobs, and CI/CD. Finally, the exam assesses the test takers' understanding of troubleshooting, monitoring, and optimization techniques, as well as their knowledge of achieving Governance and Security within the Databricks Platform.

About the Exam

- Number of scored items: 45 scored multiple-choice questions.
- Time limit: 90 minutes.
- Registration fee: USD 200, plus applicable taxes as required per local law.
- Delivery method: Online or test center
- Test aides: none allowed.
- Prerequisite: None required; course attendance and six months of hands-on experience in Databricks are highly recommended.
- Validity: 2 years.
- Recertification: Required every 2 years to maintain your certification. To recertify, you must take the current live exam. Please review the "Getting Ready for the Exam" section on the exam webpage to prepare for taking the exam again.
- Unscored Content: Exams may include unscored items to gather statistical information for future use. These items are not identified on the form and do not impact your score. Additional time is factored into account for this content.

Recommended Training

- Instructor-led: Data Engineering with Databricks
- Self-paced (available in Databricks Academy):
    - 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

Exam outline

- Databricks Intelligence Platform
    - Understand the core components of the Databricks Data Intelligence Platform, such as its architecture, Delta Lake, and Unity Catalog.
    - Understand Databricks Data Intelligence Platform's compute services, including their characteristics, limitations, and cost models, and select the most suitable option for each workload use case.
- Data Ingestion and Loading
    - Enable and detail data ingestion patterns, including batch, streaming, and incremental loading, and import data from sources such as local files, Lakeflow Connect standard connectors, and Lakeflow Connect managed connectors.
    - Use the COPY INTO command to incrementally load files from cloud object storage (ADLS/S3/GCS) into Unity-Catalog–governed tables.
    - Use Auto Loader with schema enforcement and schema evolution in batch modes (for example, directory listing or file notification) to land data into UnityCatalog–governed tables.
    - Configure Lakeflow Connect to reliably ingest data from diverse enterprise sources into UnityCatalog–governed tables.
    - Use JDBC/ODBC or REST clients in notebooks to land data into cloud storage or directly into UnityCatalog–governed tables, usually orchestrated and scheduled with Lakeflow Jobs.
    - Prioritize between Auto Loader, Lakeflow Connect (standard and managed connectors), partner connectors, and other ingestion methods based on technical requirements such as data volume, ingestion frequency, data types, and governance needs with Unity Catalog.
    - Ingest semi-structured and unstructured data (for example, JSON and nested data) via Lakeflow Connect and other managed connectors into UnityCatalog–governed Delta tables.
- Data Transformation and Modeling

    - Implement data cleaning by reading bronze tables with PySpark/SQL, cleaning nulls, standardizing data types, and writing to new silver tables.
    - Combine DataFrames with operations such as Inner join, left join, broadcast join, multiple keys, cross join, union, and union all.
    - Manipulate columns, rows, and table structures by adding, dropping, splitting, renaming column names, applying filters, and exploding arrays.
    - Perform data deduplication operations and aggregate operations on DataFrames, such as count, approximate count distinct, and mean, summary.
    - Understand the basic tuning parameters (spark.sql.shuffle.partitions:, spark.default.parallelism, spark.executor/driver.memory, spark.sql.autoBroadcastJoinThreshold) and re-measure the performance.
    - Understand the difference between, and how to build, Gold layer objects such as materialized views, views, streaming tables, and tables for BI and analytics teams in Unity Catalog.
    - Apply data quality checks and validation rules to ensure reliable Silver and Gold datasets.
- Working with Lakeflow Jobs
    - Implement control flows (retries and conditional tasks such as branching and looping) using Lakeflow Jobs for pipeline orchestration
    - Configure common tasks (notebook, SQL query, dashboard, and pipeline tasks) and their dependencies using Lakeflow Jobs and its DAG-based task graph
    - Implement job schedules using Lakeflow Jobs with an understanding of trigger types (scheduled, file arrival, and table update)
    - Choose between time-based and data-driven triggers based on data availability and pipeline dependencies.
- Implementing CI/CD
    - Manage your code development workflow within the Databricks workspace UI, including creating and switching between branches in Databricks Git Folders (formerly Databricks Repos), committing and pushing changes, and creating pull requests using Databricks Git integration.
    - Understand environment-specific configuration using Automation Bundle (formerly Databricks Asset Bundles) variables and overrides while promoting the same codebase across dev, test, and prod targets.
    - Deploy Declarative Automation Bundles (formerly Databricks Asset Bundles) to package, configure, and promote Lakeflow Jobs, Lakeflow Spark Declarative Pipelines, and other workspace assets across dev, test, and prod environments.
    - Understand the Databricks CLI to validate, deploy, and manage Declarative Automation Bundles (formerly Databricks Asset Bundles) and other workspace assets in automated CI/CD workflows.
- Troubleshooting, Monitoring, and Optimization
    - Identify trends in job performance using the Lakeflow Jobs run history view to

compare current execution times against historical baselines.
    - Use the Lakeflow Jobs UI to monitor pipeline health by interpreting job statuses, viewing DAG-based task graphs to spot upstream blockers, and tracking pipeline run times and failure rates.
    - Identify common performance bottlenecks such as data skew, shuffling, and disk spilling by interpreting stage-level metrics in the Spark UI.
    - Understand the features of Liquid Clustering and predictive optimization.
    - Diagnose cluster startup failures, library conflicts, and out-of-memory issues.
- Governance and Security
    - Differentiate between managed and external tables in Unity Catalog and perform basic operations (create, modify, delete, and convert between managed and external tables) on them.
    - Configure access controls using the UI and SQL by applying GRANT, REVOKE, and DENY privileges to principals (users, groups, and service principals) at appropriate levels of the security hierarchy.
    - Understand column-level masking and row-level security to restrict data visibility based on user groups.
    - Understand Unity Catalog ABAC policies to centrally control row-level filtering and column masking for sensitive data.

Sample Questions

These questions are retired from a previous version of the exam. The purpose is to show you the objectives as stated in the exam guide and to give you a sample question that aligns with each objective. The exam guide lists the objectives that could be covered on an exam. The best way to prepare for a certification exam is to review the exam outline in the exam guide.

Question 1

Objective: identify common performance bottlenecks such as data skew, shuffling, and disk spilling by interpreting stage-level metrics in the Spark UI.

A data engineer notices a batch job's duration has doubled after a new data source was onboarded. In the Spark UI, the longest stage shows that most tasks finish in under 30 seconds, but one task takes over 10 minutes. The stage's task summary shows Min/Median shuffle read near 400 MB while Max shuffle read exceeds 5 GB.

Which solution reduces the job runtime?

A. Increase cluster size to add more executors so the slow task finishes faster
B. Confirm adaptive query execution with skew join handling is active to automatically split the oversized partition at runtime
C. Reduce spark.sql.shuffle.partitions to coalesce more work into fewer tasks

D. Manually repartition the dataset using a salt key before the join to distribute skewed keys evenly

Question 2
Objective: Understand Databricks Data Intelligence Platform's compute services, including their characteristics, limitations, and cost models, and select the most suitable option for each workload use case.

A data engineer requires rapid iteration on pipelines while maintaining reliable rollbacks after bad ingests, ensuring audit trails for regulatory compliance, and providing consistent access to a single source of truth for both AI and BI workloads.

Which strategy should the data engineer use to meet these requirements?

A. DBFS CSV storage with manual file versioning and nightly copies for rollback.
B. Delta Lake ACID transactions and time travel, governed by Unity Catalog for consistent access and lineage.
C. Cloud object storage only, with ad hoc SQL queries for recovery and governance.
D. Ephemeral in-memory DataFrames for audit trails and BI distribution.

Question 3
Objective: Enable and detail data ingestion patterns, including batch, streaming, and incremental loading, and import data from sources such as local files, Lakeflow Connect standard connectors, and Lakeflow Connect managed connectors.

A data engineer is building downstream pipelines to consume Databricks audit logs from a customer-owned S3 bucket. Before implementing schema inference and checkpointing, they want to understand the delivery format, typical ingestion latency, and whether files may be overwritten.

What is Databricks audit log storage behavior?

A. Files are delivered as JSON with typical event logging under 15 minutes after delivery begins, and new deliveries can overwrite existing files
B. Files are delivered as CSV with sub-minute latency guarantees, and overwrites never occur once a file is written to preserve immutability
C. Files are delivered as Parquet with eventual consistency beyond 24 hours, and overwrites are disabled to simplify streaming ingestion
D. Files are delivered as JSON with delivery on a weekly batch cadence, and overwrites replace prior content completely without appending

Question 4
Objective: Diagnose cluster startup failures, library conflicts, and out-of-memory issues.

A data engineering team supports multiple business analysts who run ad hoc SQL queries throughout the day on curated Delta tables. The team needs to ensure efficient query performance, fast cluster startup, and support for multiple simultaneous users, while managing cost by avoiding unnecessary scaling to very large clusters.

Which cluster configuration meets these requirements?

A. A job cluster with autoscaling designed for scheduled ETL workflows
B. An all-purpose cluster configured with a fixed number of worker nodes
C. A high-concurrency cluster with autoscaling enabled
D. A single-node cluster configured for lightweight development tasks

Question 5
Objective: Manage your code development workflow within the Databricks workspace UI, including creating and switching between branches in Databricks Git Folders (formerly Databricks Repos), committing and pushing changes, and creating pull requests using Databricks Git integration.

A team wants a modular way to deploy, version, and orchestrate ETL pipelines in Databricks—enabling CI/CD and repeatability.

Which feature supports this requirement?

A. Use models in Unity Catalog to represent ETL jobs, where each model stores the pipeline code artifact and CI/CD promotes versions by updating model aliases tied to Job tasks.
B. Package transformation logic as wheel libraries stored in Unity Catalog Volumes and bind them to Jobs tasks to ensure deterministic deployment across environments.
C. Package API logic inside a Volume-mounted notebook, and use Jobs API v2 to trigger the notebook, depending on notebook revision history to act as a versioning system.
D. Use DABs to define resources and code assets, version them in Git, and promote deployments across environments through automated CI/CD actions.

Answers:

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