

Databricks Certified Data Engineer Associate

[Provide Exam Guide Feedback](#)

Update Information

Updates	Date
Exam Published	May 4, 2026
Exam Guide Updated	August 6, 2026

Please check the exam guide on our webpages two weeks before your exam date to ensure you are preparing for the current version of the exam.

Purpose of this Exam Guide

The purpose of this exam guide is to give you an overview of the exam and what it covers, to help you determine your exam readiness. This document is updated whenever there are changes to the exam (and indicates when those changes take effect) so that you can prepare with confidence. This guide covers the exam version that is live as of the **effective date** listed above.

Audience Description

The Databricks Certified Data Engineer Associate certification exam assesses an individual's ability to use the Databricks Platform to perform foundational data engineering tasks. This includes an understanding of the Databricks Platform, its workspace, architecture, and capabilities, along with the tasks related to data ingestion, data loading, and data transformation and modeling, such as the ability to perform Extract, Transform, Load (ETL) tasks using PySpark, working with Lakeflow Jobs, and CI/CD. The exam also assesses an understanding of troubleshooting, monitoring, and optimization techniques, as well as knowledge of achieving governance and security within the Databricks Platform.

About the Exam

Item	Detail
Number of scored items	45 scored multiple-choice questions
Possible unscored items	Up to 10 unscored items may be included (see "Unscored content" below)
Total number of questions you may see	45 scored questions plus unscored items
Time limit	90 minutes (additional time is already factored in to account for unscored items)
Registration fee	USD 200, plus applicable taxes as required per local law
Delivery method	Online proctored or test center
Languages offered	English; Japanese; Brazilian Portuguese (pt-BR); Korean
Test aides	None allowed
Prerequisite	None required; related training highly recommended
Recommended experience	Six months of hands-on experience in the Databricks Platform, with the listed exam outline, are highly recommended
Validity	2 years

Recertification: Recertification is required every two years to maintain your certified status. To recertify, you can take the Data Engineer Associate Renewal Exam or the current full exam.

Unscored content: Exams may include unscored items to gather statistical information for future use. These items are not identified on the exam and do not affect your score, and additional time is factored into the time limit to account for them.

Recommended Preparation

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
 - Build Data Pipelines with Apache Spark™ Declarative Pipelines

- Getting started with Governance

AI Prep Guide – Any Databricks Exam

The [Databricks AI Prep Guide](#) shows you how to turn the AI chatbot you already use into a personalized tutor for any Databricks certification: prime it with the official exam guide, catch the outdated answers AI tends to give, find your weak spots, and pair it all with the hands-on practice every Databricks exam demands.

Hands-On Practice

Not sure what to practice? The AI Prep Guide (see the AI Prep Guide section above) includes a prompt under “Step 5 – Hands-on minimum” that has an AI chatbot generate a personalized checklist of 6–10 hands-on tasks mapped to this exam’s objectives, each small enough to complete in Databricks [Free Edition](#).

Databricks Documentation

As you prepare, study the official [Databricks documentation](#) for the platform areas covered on this exam. It is the most current, authoritative source. Use it to judge how deeply to study each topic and to ensure you are learning up-to-date, non-deprecated features rather than out-of-date material.

Exam Outline

Section 1: Databricks Intelligence Platform (6%)

- Understand the core components of the Databricks Platform, such as its architecture, Delta Lake, and Unity Catalog.
- Understand Databricks Platform's compute services, including their characteristics, limitations, and cost models, and select the most suitable option for each workload use case.

Section 2: Data Ingestion and Loading (21%)

- 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 Unity Catalog-governed tables.
- Configure Lakeflow Connect to reliably ingest data from diverse enterprise sources into Unity Catalog-governed tables.
- Use JDBC/ODBC or REST clients in notebooks to land data into cloud storage or directly into Unity Catalog-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 Unity Catalog-governed Delta tables.

Section 3: Data Transformation and Modeling (22%)

- 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.

Section 4: Working with Lakeflow Jobs (16%)

- 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.

Section 5: Implementing CI/CD (10%)

- 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 Declarative 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 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.

Section 6: Troubleshooting, Monitoring, and Optimization (10%)

- 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.

Section 7: Governance and Security (15%)

- 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

The purpose of sample questions is to show you the objectives as they are 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 is to review the full exam outline for your exam date above.

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 that a batch job's duration has doubled after a new data source was onboarded. The job performs a wide groupBy aggregation on a customer_id key. In the Spark UI, the longest stage shows that most tasks finish in under 30 seconds, but one task runs over 10 minutes; the stage's task summary shows Min/Median shuffle read near 400 MB while the Max shuffle read exceeds 5 GB. Adaptive Query Execution is already enabled with its default settings, yet the skewed task persists.

Which solution reduces the job runtime?

- A. Increase cluster size to add more executors so the slow task finishes faster
- B. Rely on AQE skew-join handling to automatically split the oversized partition at runtime
- C. Reduce spark.sql.shuffle.partitions to coalesce more work into fewer tasks
- D. Add a salt key to the aggregation key so the skewed value is spread across multiple tasks

Question 2

Objective: *Understand the core components of the Databricks Platform, such as its architecture, Delta Lake, and Unity Catalog.*

A data engineer requires rapid pipeline iteration 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: *Understand Databricks Platform's compute services, including their characteristics, limitations, and cost models, and select the most suitable option for each workload use case.*

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 serverless SQL warehouse with autoscaling enabled
- D. A single-node cluster configured for lightweight development tasks

Question 5

Objective: *Deploy Declarative Automation Bundles (formerly Databricks Asset Bundles) to package, configure, and promote Lakeflow Jobs, Lakeflow Declarative Pipelines, and other workspace assets across dev, test, and prod environments.*

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 Lakeflow 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 Declarative Automation Bundles (DABs) to define resources and code assets, version them in Git, and promote deployments across environments through automated CI/CD actions.

Question 6

Objective: *Understand the use of Databricks CLI to validate, deploy, and manage DABs and other workspace assets in an automated CI/CD workflow*

A data engineer is using Declarative Automation Bundles to deploy a job that connects to an external database. To follow security best practices, the database password must not be hardcoded in the databricks.yml file or the notebook source code.

How should the data engineer configure the bundle to securely handle this secret during an automated deployment?

- A. Define a variable in the variables block of databricks.yml and provide its value via an environment variable prefixed with BUNDLE_VAR_
- B. Store the password in a plain text file named passwords.txt in the bundle root and include it in the deployment
- C. Use the Databricks bundle encrypt command to obfuscate the string before committing to Git
- D. Use a standard Spark configuration spark.conf.set("db_pass", "...") directly inside the notebook.

Question 7

Objective: *Apply column-level masking and row-level security to restrict data visibility based on the user's group.*

A data engineer needs to ensure that only members of the 'finance_team' group can see the full value of the 'credit_card_number' column. All other users should see a masked version.

Which Unity Catalog feature and configuration should be used to implement this requirement?

- A. Create a separate Delta table for each user group with redacted values
- B. Enable AWS IAM roles on the specific column within the Delta table properties

- C. Apply a column masking function using the `is_account_group_member('finance_team')` logic
- D. Set a DENY permission on the specific column for all users except the finance team

Question 8

Objective: *Differentiate between managed and external tables in Unity Catalog and perform basic operations.*

A data engineer converts an external table to a managed table. Shortly after, the team discovers an issue and needs to undo the conversion.

Which action restores the table to its previous type in newer runtimes?

- A. Run `ALTER TABLE ... UNSET MANAGED` to revert to external
- B. Drop the managed table and recreate it as external using the original LOCATION, since rollback is not supported
- C. Run `ALTER TABLE ... SET EXTERNAL` to change the table type back without moving data
- D. Restore the table from the Delta time travel history using `RESTORE TABLE ... TO VERSION AS OF`

Question 9

Objective: *Use the Lakeflow Jobs UI to monitor pipeline health by interpreting job statuses and viewing task graphs*

A Databricks job task fails with the status 'Timed Out' after running for 45 minutes. The data engineer determines that the task may take up to 60 minutes to complete successfully due to the increased volume of data to be processed and wants future runs to finish without being automatically terminated.

Which configuration should the data engineer modify to achieve the goal?

- A. The dependency condition on the upstream task
- B. The threshold for task run duration
- C. The maximum concurrent runs setting on the job
- D. The cluster auto-termination setting on the compute resource

Question 10

Objective: *Implement job schedules using Lakeflow Jobs with an understanding of trigger types*

A data engineering team receives clickstream export files from a third-party analytics vendor. The vendor uploads compressed JSON files to an external cloud storage location at unpredictable intervals — sometimes several files within an hour, sometimes nothing for most of a day. A Lakeflow Job runs a batch notebook that reads new files with Auto Loader, enriches them, and merges the

results into a Delta table. The team wants each new drop processed shortly after it lands, without operators monitoring the storage location, and without incurring compute costs during long idle periods or launching runs that find no new data.

Which trigger type should be configured for this Lakeflow Job?

- A. Scheduled trigger every 5 minutes to poll for new export files
- B. File arrival trigger on the vendor's external storage location
- C. Continuous trigger to keep the job always running
- D. Manual trigger started by an operator whenever a vendor upload notification arrives

Answer Key & Explanations

#	Answer	Explanation
1	D	A Max shuffle read of 5 GB against a ~400 MB median is classic key skew — one hot customer_id dominates a single partition. AQE's skew handling optimizes skewed joins (sort-merge/shuffle-hash) only, not groupBy aggregations, so it doesn't relieve this stage even though AQE is on eliminating B. Adding executors leaves the one giant task running alone (A), and lowering shuffle.partitions makes partitions larger and worsens skew (C). Salting the aggregation key spreads the hot key across tasks, followed by a second pass to combine partial results.
2	B	Delta Lake provides ACID transactions and time travel, which give reliable rollback after a bad ingest and a full audit history, while Unity Catalog governs access and lineage so AI and BI workloads read one source of truth. The other options depend on manual copies or ungoverned storage.
3	A	Databricks delivers audit logs as JSON files, typically within 15 minutes of the event once delivery begins, and a later delivery can overwrite an existing file. Ingestion pipelines must therefore tolerate overwrites rather than assume files are immutable.
4	C	A serverless SQL warehouse, with autoscaling is built for many simultaneous interactive SQL users, giving fast startup and query isolation while scaling only as demand requires. Job clusters serve scheduled ETL, and fixed or single-node clusters cannot balance concurrency against cost.
5	D	Declarative Automation Bundles declare jobs, pipelines, and code assets as versioned configuration in Git, so the same definition is promoted across dev, test, and prod through automated CI/CD. The other options rely on artifact stores or notebook revision history, neither of which version the

		orchestration itself.
6	A	DABs support parameterization via a variables block. By using the prefix <code>BUNDLE_VAR_<variable_name></code> , the CLI automatically injects the value from the environment (e.g., from GitHub Secrets) into the bundle at deployment time, ensuring the secret never touches the source code or the YAML file.
7	C	Column masking using the <code>is_account_group_member()</code> function is the correct choice because it allows for dynamic fine-grained security where the same table can be queried by different users while selectively obscuring sensitive data based on their group membership.
8	A	UNSET MANAGED is the documented rollback. It updates table metadata to point back to the original external location, and it is available only within a 14-day window after conversion. Writes made to the managed location after conversion are preserved; commits between conversion and rollback support time travel by version but not by timestamp. Databricks deletes data in the managed location seven days after rollback, and streaming jobs must be restarted.
9	B	A "Timed Out" status means the run exceeded the configured timeout threshold on the task — here, 45 minutes. Raising it above 60 minutes (with headroom, e.g. 90 min) lets future runs complete instead of being killed.
10	B	File arrival triggers monitor an external location and start a run only when new files appear — automated, low-latency, and no compute spent while the location is unchanged.