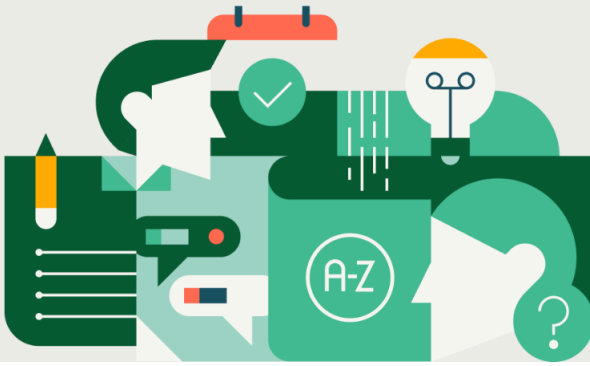
databricks

Databricks 試験ガイド

Databricks Certified データエンジニア アソシエイト



試験ガイドのフィードバックを提供する

情報を更新する

更新	日付
試験公開日	May 4, 2026
試験ガイド更新日	August 6, 2026

試験日の2週間前に、当社のウェブページで試験ガイドを確認し、最新バージョンの試験に向けて準備していることをご確認ください。

この試験ガイドの目的

この試験ガイドの目的は、試験の概要とその出題範囲をお伝えし、受験準備の状況を判断するお手伝いをすることです。このドキュメントは、試験に変更があった場合（およびその変更の発効日）に随時更新されますので、自信を持って準備を進めることができます。本ガイドは、以上に記載された発効日時点で有効な試験バージョンを対象としています。

対象者の説明

Databricks Certified Data Engineer Associate 認定試験は、Databricks Platform を使用して基礎的なデータエンジニアリングタスクを実行する個人の能力を評価します。これには、Databricks Platform、そのワークスペース、アーキテクチャ、および機能に関する理解、ならびにデータ取り込み、データロード、データ変換とモデリングに関連するタスク（PySpark を使用した抽出, 変換, ロード（ETL）タスクの実行、LakeFlow Jobs の活用、CI/CD など）が含まれます。また、トラブルシューティング、モニタリング、および最適化技術の理解、ならびに Databricks Platform 内でガバナンスとセキュリティを実現するための知識も評価されます。

試験について

項目	詳細
採点対象の問題数	採点対象の多肢選択式問題 45 問
採点対象外の問題	採点対象外の問題が最大 10 問含まれる場合があります（後述の「採点対象外のコンテンツ」を参照）
出題される問題の総数	採点対象の問題 45 問＋採点対象外の問題
制限時間	90 分（採点対象外の問題に対応するための追加時間を含みます）
受験料	200 米ドル（現地の法律に基づき、適用される税金が別途かかります）
受験方法	オンライン監督付き試験またはテストセンター
提供言語	英語、日本語、ブラジルポルトガル語（pt-BR）、韓国語
補助資料	使用不可
前提条件	なし（ただし、関連トレーニングの受講を強く推奨します）
推奨される経験	試験の出題範囲に沿った Databricks Platform での実務経験を 6 か月以上有することを強く推奨します
有効期間	2年

再認定：認定ステータスを維持するには、**2**年ごとに再認定が必要です。再認定を受けるには、**Data Engineer Associate Renewal Exam** または現行のフル試験を受験することができます。

採点対象外のコンテンツ：試験には、将来の使用に向けて統計的な情報を収集するために、採点対象外の問題が含まれる場合があります。これらの問題は試験中に識別されず、スコアにも影響しません。また、これらの問題に対応するための追加時間が制限時間に考慮されています。

推奨される準備

トレーニング

- インストラクター主導**: Databricks** によるデータエンジニアリング
- 自己学習型（**Databricks Academy** で利用可能）：
    - LakeFlow Connect を使用したデータ取り込み
    - LakeFlow Jobs を使用したワークロードのデプロイ

    - データエンジニアリングのための DevOps エッセンシャル
    - Apache Spark™ Declarative Pipelines を使用したデータパイプラインの構築
    - ガバナンスの概要

AI Prep ガイド - すべての Databricks 試験

Databricks AI Prep Guide ガイドでは、すでにお使いのAIチャットボットを任意のDatabricks認定試験向けのパーソナライズされた個別指導ツールに変える方法をご紹介します。公式試験ガイドを活用してAIを準備し、AIが出しがちな古い回答を見極め、自分の弱点を把握し、すべてをDatabricksの各試験で求められる実践的な演習と組み合わせることができます。

ハンズオン演習

何を練習すればよいかわからない場合は、AI Prep ガイド（以上の AI Prep ガイドセクションを参照）をご確認ください。このガイドには「Step 5 — Hands-on minimum」というプロンプトが含まれており、AI チャットボットがこの試験の目標に対応した6〜10個のハンズオンタスクの個別チェックリストを生成します。各タスクは Databricks Free Edition で完了できる規模です。

Databricksドキュメント

準備を進める際は、この試験でカバーされるプラットフォーム領域について、公式のDatabricksドキュメントを学習してください。それは最も最新かつ権威あるソースです。各トピックをどの程度深く学習するかの判断に活用し、古い情報ではなく最新の非推奨でない機能を確実に習得するようにしてください。Databricksドキュメント

試験の概要

セクション 1: Databricks Intelligence Platform (6%)

- Databricks Platform のアーキテクチャ、Delta Lake、Unity Catalog などのコアコンポーネントを理解する。
- Databricks Platform のコンピュートサービスについて、その特性、制限、およびコストモデルを理解し、各ワークロードのユースケースに最も適したオプションを選択できます。

セクション 2: データ取り込みとロード (21%)

- バッチ、ストリーミング、インクリメンタルロードなどのデータ取り込みパターンを有効化して詳述し、ローカルファイル、LakeFlow Connect 標準コネクター、LakeFlow Connect マネージドコネクターなどのソースからデータをインポートします。
- COPY INTO コマンドを使用して、クラウドオブジェクトストレージ（ADLS/S3/GCS）から Unity Catalog で管理されたテーブルにファイルを増分ロードします。

- バッチモード（例：ディレクトリリスティングまたはファイル通知）でスキーマ強制とスキーマ進化を使用した Auto Loader により、Unity Catalog で管理されたテーブルにデータを取り込みます。
- LakeFlow Connect を設定して、多様なエンタープライズソースから Unity Catalog で管理されたテーブルにデータを確実に取り込みます。
- ノートブックでJDBC/ODBCまたはRESTクライアントを使用してデータをクラウドストレージにランディングするか、Unity Catalogが管理するテーブルに直接取り込みます。通常、Lakeflow Jobsでオーケストレーションおよびスケジュールされます。
- データ量、取り込み頻度、データ型、Unity Catalog によるガバナンスニーズなどの技術要件に基づいて、Auto Loader、Lakeflow Connect（標準コネクターおよびマネージドコネクター）、パートナーコネクター、およびその他の取り込み方法の優先順位を判断します。
- LakeFlow Connect およびその他のマネージドコネクターを使用して、半構造化データおよび非構造化データ（例：JSON やネストされたデータ）を Unity Catalog で管理された Delta テーブルに取り込みます。

セクション3: データ変換とモデリング (22%)

- PySpark/SQL を使用してブロンズテーブルを読み込み、null 値のクリーニング、データ型の標準化を行い、新しいシルバーテーブルに書き込むことでデータクリーニングを実装します。
- Inner join、left join、broadcast join、複数キー、cross join、union、union all などの操作を使用して DataFrames を結合します。
- 列の追加、削除、分割、名前の変更、フィルターの適用、配列のエクスプロードを行うことで、列、行、およびテーブル構造を操作します。
- DataFrames に対して、count、approximate count distinct、mean、summary などのデータ重複排除操作および集計操作を実行します。
- 基本的なチューニングパラメーター（spark.sql.shuffle.partitions、spark.default.parallelism、spark.executor/driver.memory、spark.sql.autoBroadcastJoinThreshold）を理解し、パフォーマンスを再測定します。
- Unity Catalog における BI およびアナリティクスチーム向けのゴールドレイヤーオブジェクト（Materialized Views、ビュー、Streaming Tables、テーブルなど）の違いを理解し、それらの構築方法を習得します。
- データ品質チェックと検証ルールを適用する ことで、信頼性の高いシルバーおよびゴールドデータセットを確保します。

セクション4: Lakeflow Jobsの操作（16%）

- パイプラインのオーケストレーションのために Lakeflow Jobs を使用して、制御フロー（リトライや分岐・ループなどの条件付きタスク）を実装します。
- Lakeflow Jobs とその DAG ベースのタスクグラフを使用して、一般的なタスク（ノートブック、SQL Query、ダッシュボード、パイプラインタスク）とその依存関係を設定する
- トリガーの種類（スケジュール、ファイル到着、テーブル更新）を理解した上で、LakeFlow Jobs を使用してジョブのスケジュールを実装する。

- データの可用性とパイプラインの依存関係に基づいて、時間ベースのトリガーとデータドリブンのトリガーを使い分けます。

セクション 5: CI/CD の実装 (10%)

- Databricks ワークスペース UI 内でコード開発ワークフローを管理する方法（Databricks Git Folders（旧 Databricks Repos）でのブランチの作成と切り替え、変更のコミットとプッシュ、Databricks Git 統合を使用したプルリクエストの作成を含む）について説明します。
- Declarative Automation Bundle（旧称 Databricks Asset Bundles）の変数とオーバーライドを使用した環境固有の設定を理解し、同一のコードベースを dev、test、prod の各ターゲットにプロモートする方法を習得します。
- Declarative Automation Bundles（旧称：Databricks Asset Bundles）を使用して、Lakeflow Jobs、Lakeflow Declarative Pipelines、およびその他のワークスペース資産をパッケージ化、設定し、開発、テスト、本番環境にわたってプロモートします。
- Databricks CLI を使用して、Declarative Automation Bundles（旧 Databricks Asset Bundles）およびその他のワークスペース資産を、自動化された CI/CD Workflows 内で検証、デプロイ、管理するための理解を深めます。

セクション 6: トラブルシューティング、モニタリング、および最適化 (10%)

- Lakeflow Jobs の実行履歴ビューを使用して現在の実行時間と過去のベースラインを比較し、ジョブのパフォーマンスの傾向を特定します。
- LakeFlow Jobs UIを使用して、ジョブのステータスを解釈し、DAGベースのタスクグラフを表示してアップストリームのブロッカーを特定し、パイプラインの実行時間と失敗率をトラッキングすることで、パイプラインの健全性を監視します。
- Spark UI のステージレベルのメトリクスを解釈することで、データスキュー、シャッフル、ディスクスピルなどの一般的なパフォーマンスのボトルネックを特定します。
- リキッドクラスタリングと Predictive Optimization の機能を理解する。
- クラスターのスタートアップの失敗、ライブラリの競合、およびメモリ不足の問題を診断します。

セクション 7: ガバナンスとセキュリティ (15%)

- Unity Catalog におけるマネージドテーブルと外部テーブルの違いを区別し、それらに対して基本的な操作（作成、変更、削除、マネージドテーブルと外部テーブル間の変換）を実行します。
- UIとSQLを使用してアクセス制御を設定し、セキュリティ階層の適切なレベルでプリンシパル（ユーザー、グループ、およびサービスプリンシパル）にGRANT、REVOKE、DENYの権限を適用します。
- ユーザーグループに基づいてデータの可視性を制限するために、列レベルのマスキングと行レベルのセキュリティを理解します。
- Unity Catalog の ABAC ポリシーを理解し、機密データに対する行レベルフィルタリングとカラムマスキングを一元的に制御します。

サンプル問題

サンプル問題の目的は、試験ガイドに記載されている目標を示し、各目標に対応するサンプル問題を提供することです。試験ガイドには試験で出題される可能性のある目標が記載されています。最善の準備方法は、上記の試験日程に対応した試験の全概要を確認することです。

質問 **1**

目標：**Spark UI** のステージレベルのメトリクスを解釈することで、データスキュー、シャッフル、ディスクスピルなどの一般的なパフォーマンスボトルネックを特定する。

データエンジニアが、新しいデータソースがオンボードされた後にバッチジョブの期間が2倍になったことに気づきました。このジョブは `customer_id` キーに対してワイドな `groupBy` 集計を実行しています。Spark UIでは、最も長いステージにおいて、ほとんどのタスクが30秒以内に完了しているものの、1つのタスクが10分以上かかっています。そのステージのタスクサマリーでは、最小値／中央値のシャッフル読み取りが約400 MB であるのに対し、最大シャッフル読み取りは5 GB を超えています。Adaptive Query Execution はすでにデフォルト設定で有効になっていますが、スキューしたタスクは依然として解消されていません。

どのソリューションがジョブのランタイムを短縮しますか？

A. クラスターサイズを拡大してエグゼキューターを追加し、遅いタスクをより速く完了させる
B. AQE のスキュー結合処理を活用して、ランタイム時に過大なパーティションを自動的に分割する。
C. spark.sql.shuffle.partitions を減らして、より少ないタスクにより多くの処理をまとめる
D. スキューした値が複数のタスクに分散されるよう、集計キーにソルトキーを追加する

質問 2

目標：**Databricks Platform** のアーキテクチャ、**Delta Lake**、**Unity Catalog** などのコアコンポーネントを理解する。

データエンジニアは、不正なインジェスト後に信頼性の高いロールバックを維持しながら迅速なパイプラインの反復を実現し、規制コンプライアンスのための監査証跡を確保し、AI および BI ワークロードの両方に対して単一のソースオブトゥルースへの一貫したアクセスを提供することを求めています。

データエンジニアはこれらの要件を満たすためにどの戦略を使用すべきですか？

A. 手動によるファイルバージョン管理とロールバック用の夜間コピーを使用した DBFS CSV ストレージ。
B. Delta Lake のACIDトランザクションとタイムトラベルは、Unity Catalog によって管理され、一貫したアクセスとリネージを実現します。
C. クラウドオブジェクトストレージのみで、リカバリとガバナンスにはアドホックな SQL クエリを使用します。
D. 監査証跡およびBI配布のための一時的なインメモリDataFrames。

質問 3

目的：バッチ、ストリーミング、インクリメンタルロードを含むデータ取り込みパターンを有効化して詳述し、ローカルファイル、**LakeFlow Connect** 標準コネクター、**LakeFlow Connect** マネージドコネクターなどのソースからデータをインポートします。

データエンジニアが、顧客所有の S3 バケットから Databricks の監査ログを取り込むダウンストリームパイプラインを構築しています。スキーマ推論とチェックポイントを実装する前に、配信フォーマット、一般的な取り込みレイテンシー、およびファイルが上書きされる可能性があるかどうかを把握したいと考えています。

Databricks の監査ログのストレージ動作はどのようなものですか？

A. ファイルはJSONとして配信され、配信開始から通常15分以内にイベントログが記録されます。また、新しい配信によって既存のファイルが上書きされる場合があります。
B. ファイルはCSVとして配信され、1分未満のレイテンシが保証されており、イミュータビリティを維持するためにファイルが書き込まれた後は上書きが発生しません。
C. ファイルはParquetとして配信され、24時間を超えると結果整合性が保証され、ストリーミング取り込みを簡素化するために上書きは無効化されています。
D. ファイルはJSONとして配信され、週次バッチのサイクルで配信されます。また、上書きにより以前のコンテンツは追記されることなく完全に置き換えられます。

質問 4

目標：**Databricks Platform** のコンピュートサービスについて、その特性、制限、およびコストモデルを理解し、各ワークロードのユースケースに最も適したオプションを選択できるようにする。

データエンジニアリングチームは、1日を通じてキュレーション済みの Delta テーブルに対してアドホックな SQL クエリーを実行する複数のビジネスアナリストをサポートしています。チームは、効率的なクエリーのパフォーマンス、高速なクラスタースタートアップ、複数の同時ユーザーへの対応を確保しつつ、非常に大規模なクラスターへの不必要なスケーリングを避けることでコストを管理する必要があります。

これらの要件を満たすクラスター構成はどれですか？

A. スケジュールされたETL Workflowsのために設計されたオートスケーリング機能付きのジョブクラスター
B. 固定数のワーカーノードで構成された汎用クラスター
C. オートスケーリングが有効化されたServerless SQL Warehouse
D. 軽量な開発タスク向けに構成されたシングルノードクラスター

質問 5

目標：**Declarative Automation Bundles**（旧 **Databricks Asset Bundles**）を使用して、**Lakeflow Jobs**、**Lakeflow Declarative Pipelines**、およびその他のワークスペース資産をパッケージ化、設定し、**dev**、**test**、**prod** の各環境にプロモートするためにデプロイすること。

あるチームは、Databricks において ETL パイプラインをモジュール式にデプロイ、バージョン管理、およびオーケストレーションし、CI/CD と再現性を実現する方法を求めています。

この要件をサポートしているのはどの機能ですか？

A. Unity Catalog のモデルを使用して ETL ジョブを表現し、各モデルにパイプラインのコードアーティファクトを格納し、CI/CD がジョブタスクに紐付けられたモデルエイリアスを更新することでバージョンを昇格させます。
B. 変換ロジックをwheelライブラリとしてパッケージ化してUnity Catalog Volumesに保存し、それらをLakeflow Jobsのタスクにバインドすることで、環境をまたいだ確定的なデプロイを保証します。
C. API ロジックをボリュームにマウントされたノートブック内にパッケージ化し、Jobs API v2 を使用してノートブックをトリガーし、バージョン管理システムとしてノートブックの改訂履歴に依存します。
D. Declarative Automation Bundles (DABs) を使用してリソースとコードアセットを定義し、Git でバージョン管理を行い、自動化された CI/CD アクションを通じて環境全体にデプロイメントを昇格させます。

Question 6

目標: 自動化された**CI/CD**ワークフローにおいて、**Databricks CLI**を使用して**DABs**やその他のワークスペースアセットを検証、デプロイ、管理するための使い方を理解する。

データエンジニアが Declarative Automation Bundles を使用して、外部データベースに接続するジョブをデプロイしています。セキュリティのベストプラクティスに従うため、データベースのパスワードを databricks.yml ファイルやノートブックのソースコードにハードコードしてはなりません。

自動デプロイメント中にこのシークレットを安全に処理するために、データエンジニアはバンドルをどのように設定する必要がありますか？

A. `databricks.yml` の variables ブロックで変数を定義し、`BUNDLE_VAR_` というプレフィックスを付けた環境変数を通じてその値を指定します。
B. パスワードをバンドルルートに `passwords.txt` という名前のプレーンテキストファイルに保存し、デプロイメントに含める。
C. Databricks bundle encrypt コマンドを使用して、Git にコミットする前に文字列を難読化してください。
D. 標準的な Spark 設定 spark.conf.set("db_pass", "...") をノートブック内で直接使用する。

質問 7

目標: ユーザーのグループに基づいてデータの可視性を制限するために、列レベルのマスキングと行レベルのセキュリティを適用する。

データエンジニアは、'finance_team' グループのメンバーのみが 'credit_card_number' 列の完全な値を参照できるようにする必要があります。それ以外のユーザーには、マスクされたバージョンが表示される必要があります。
この要件を実装するには、Unity Catalog のどの機能と構成を使用する必要がありますか？

A. ユーザーグループごとに、値をマスクした個別の Delta テーブルを作成する

B. Delta テーブルのプロパティ内の特定の列で AWS IAM ロールを有効にする
C. is_account_group_member('finance_team') ロジックを使用して列マスキング関数を適用する
D. 財務チームを除くすべてのユーザーに対して、特定の列にDENY権限を設定する。

質問 8

目標：**Unity Catalog** におけるマネージドテーブルと外部テーブルの違いを理解し、基本的な操作を実行する。

データエンジニアが外部テーブルをマネージドテーブルに変換しました。その直後、チームは問題を発見し、変換を元に戻す必要があります。

新しいランタイムでテーブルを以前のタイプに復元するアクションはどれですか？

A. `ALTER TABLE ... UNSET MANAGED` を実行して外部テーブルに戻す。
B. ロールバックはサポートされていないため、マネージドテーブルを削除し、元の LOCATION を使用して外部テーブルとして再作成する。
C. `ALTER TABLE ... SET EXTERNAL` を実行して、データを移動せずにテーブルタイプを元に戻します。
D. `RESTORE TABLE ... TO VERSION AS OF` を使用して、Delta のタイムトラベル履歴からテーブルを復元します。

問題 9

目標：**LakeFlow Jobs** の **UI** を使用して、ジョブのステータスを解釈し、タスクグラフを確認することで、パイプラインの健全性を監視する。

Databricks のジョブタスクが45分間実行された後、ステータス「Timed Out」で失敗しました。データエンジニアは、処理するデータ量の増加により、タスクが正常に完了するまで最大60分かかる可能性があると判断しており、今後の実行が自動的に終了されることなく完了するようにしたいと考えています。

データエンジニアは目標を達成するためにどの設定を変更すべきですか？

A. 上流タスクへの依存条件
B. タスク実行期間のしきい値
C. ジョブの最大並列実行数の設定
D. コンピューティングリソースのクラスター自動終了設定

質問 10

目標**:** トリガーの種類を理解した上で、**LakeFlow Jobs** を使用してジョブのスケジュールを実装する

データエンジニアリングチームは、サードパーティのアナリティクスベンダーからクリックストリームのエクスポートファイルを受け取っています。ベンダーは圧縮されたJSON ファイルを外部のクラウドストレージの場所に予測不可能な間隔でアップロードします。1時間以内に複数のファイルがアップロードされることもあれば、1日のほとんどの時間何もアップロードされないこともあります。Lakeflow Job は、Auto Loader で新しいファイルを読み取り、データをエンリッチし、結果を Delta テーブルにマージするバッチノートブックを実行し

ます。チームは、オペレーターがストレージの場所をモニタリングすることなく、また長いアイドル期間中にコンピュートコストが発生したり、新しいデータが見つからない状態で実行が起動されたりすることなく、新しいファイルが到着した直後に処理されることを望んでいます。

この Lakeflow Job にはどのトリガータイプを設定する必要がありますか？

A. 新しいエクスポートファイルをポーリングするために5分ごとにスケジュールされたトリガー
B. ベンダーの外部ストレージロケーションに対するファイル到着トリガー
C. ジョブを常に実行し続けるための継続的なトリガー
D. ベンダーのアップロード通知が届くたびにオペレーターが手動で開始するトリガー

解答キーと解説

#	解答	解説
1	D	Max shuffle read が 5 GB で中央値が約 400 MB という状況は、典型的なキーのスキュー（key skew）です。ホットな customer_id が単一のパーティションを占有しています。AQE のスキュー処理は、スキューのあるジョイン（sort-merge/shuffle-hash）のみを最適化し、groupBy 集計は対象外です。そのため、AQE が有効であってもこのステージは改善されません（B を除外）。エグゼキューターを追加しても、巨大なタスクが 1 つだけ残るため解決しません（A）。shuffle.partitions を減らすとパーティションがさらに大きくなり、スキューが悪化します（C）。集計キーにソルティング（salting）を適用すると、ホットキーを複数のタスクに分散でき、その後の 2 段階目の処理で部分結果を統合できます。
2	B	Delta Lake は ACID トランザクションとタイムトラベルを提供し、不適切な取り込み後の確実なロールバックと完全な監査履歴を実現します。一方、Unity Catalog はアクセスとリネージを管理するため、AI および BI ワークロードは単一の信頼できるデータソースを読み取ります。他の選択肢は手動によるコピーや、管理されていないストレージに依存しています。
3	A	Databricks は監査ログを JSON ファイルとして配信し、配信が開始されるとイベント発生から通常 15 分以内に届きます。また、後続の配信によって既存のファイルが上書きされる場合があります。したがって、取り込みパイプラインは、ファイルが変更不可であると想定するのではなく、上書きを許容する必要があります。
4	C	自動スケーリング付きのサーバーレス SQL ウェアハウスは、多数の同時インタラクティブ SQL ユーザー向けに設計されており、起動が速く、クエリが分離され、需要に応じてのみスケールします。ジョブクラスターはスケジュールされた ETL に使用され、固定サイズのクラスターやシングルノードクラスターは同時実行性とコストのバランスを取ることができません。
5	D	Declarative Automation Bundles は、ジョブ、パイプライン、コードアセットを Git でバージョン管理された構成として定義するため、同一の定義を自動化された CI/CD を通じて dev、test、prod へ昇格できます。他の選択肢はアーティファクトストアやノートブックのリビジョン履歴に依存しており、いずれもオーケストレーション自体をバージョン管理しません。
6	A	DABs は variables ブロックによるパラメーター化をサポートしています。BUNDLE_VAR_<variable_name> というプレフィックスを使用することで、CLI がデプロイ時に環境変数（例: GitHub Secrets）から値を自動的に注入し、シークレットがソースコードや YAML ファイルに一切含まれないようにします。

7	C	is_account_group_member() 関数を使用した列マスキングは、同じテーブルを異なるユーザーがクエリできる一方で、グループメンバーシップに基づいて機密データを選択的に隠すという、動的できめ細かなセキュリティを実現するため、正しい選択です。
8	A	UNSET MANAGED が文書化されたロールバック手段です。テーブルのメタデータを元の外部ロケーションを指すように更新し、変換後 14 日以内に限り利用できます。変換後にマネージドロケーションへ書き込まれたデータは保持されます。変換後からロールバックまでのコミットは、バージョンによるタイムトラベルはサポートされますが、タイムスタンプによるタイムトラベルはサポートされません。ロールバック後 7 日でマネージドロケーションのデータが削除され、ストリーミングジョブは再起動が必要です。
9	B	「Timed Out」ステータスは、そのタスクが設定されたタイムアウトのしきい値を超えたことを意味します。ここでは 45 分です。60 分を超える値（余裕を持たせて例えば 90 分）に引き上げると、今後の実行が強制終了されずに完了できます。
10	B	ファイル到着トリガー（file arrival triggers）は外部ロケーションを監視し、新しいファイルが出現したときにのみ実行を開始します。自動化されており、低レイテンシで、ロケーションに変更がない間はコンピュートを消費しません。