databricks

Databricks 試験ガイド

Databricks Certified Data Engineer Professional

試験ガイドについてフィードバックを送る

この試験ガイドの目的

この試験ガイドでは、試験の準備に役立てていただくために試験の概要と試験の対象範囲について説明します。試験内容が変更された場合（および、その変更が試験に反映された場合）は、本書の内容も更新されますので、それに合わせて準備を行ってください。このバージョンは、**2025** 年 **3** 月 **1** 日時点で使用されていたバージョンについて説明したものです。試験を受ける **2** 週間前に必ず最新バージョンを確認するようにしてください。

対象者の説明

Databricks Certified Data Engineer Professional 認定試験では、Databricks を使用して高度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 プラットフォームと、Apache Spark、Delta Lake、MLflow、Databricks CLI および REST API などの開発者ツールについての理解が含まれます。また、最適化されたクリーニング済みの ETL パイプラインを構築する能力も評価します。さらに、一般的なデータモデリング概念の知識を使用してデータをレイクハウスにモデリングする能力も評価されます。最後に、データパイプラインの安全性と信頼性が確保され、モニタリングとテストが行われたことをデプロイメント前に確認することも、試験内容に含まれます。この認定試験に合格した人は、Databricks とその関連ツールを使って高度なデータエンジニアリングタスクを完了できることが期待されます。

試験の概要

- 問題数:採点対象の多肢選択問題 60 問
- 制限時間: 120 分
- 受験料:200 米ドル。これに、現地の法律で定められた税金が加算されます。
- 実施方法:オンライン（監督付き）
- 試験に持ち込めるもの:一切許可されません。
- 前提条件: なし。Databricks に関するコースの受講と 1 年の実務経験が強く推奨されます。
- 有効期間: 2 年間
- 採点対象外の内容: 今後使用する統計情報を収集するために、試験には採点対象外の項目が含まれている場合があります。これらの項目はフォーム上では区別されず、点数には影響しません。この内容については、追加の時間が考慮されます。

推奨されるトレーニング

- セルフペース (Databricks Academy で提供):
    - Databricks ストリーミングとDelta Live Table - Delta Dawn
    - Databricks データプライバシー
    - Databricks パフォーマンスの最適化 Delta Dawn
    - Databricks アセットバンドルによるテストとデプロイの自動化

試験の概要

セクション **1: Databricks** のツール

- Delta Lake がトランザクションログとクラウドオブジェクトストレージを使用して原子性と永続性を保証する方法を説明する
- Delta Lake が楽観的同時実行制御を使用して分離を実現する方法と、競合する可能性があるトランザクションを説明する
- Delta クローンの基本的な機能を説明する
- パーティショニング、Z-Order、Bloom フィルタ、ファイルサイズといった、Delta Lake の一般的なインデックス最適化を適用する
- Databricks SQL サービス向けに最適化された Delta テーブルを実装する
- さまざまなデータパーティショニング手法を比較する (適切なパーティショニング列の判定など)

セクション **2:** データ処理 **(**バッチ処理、増分処理、最適化**)**

- 以下のパーティションヒントについて説明し、違いを述べる:結合、再パーティショニング、範囲による再パーティショニング、リバランス
- さまざまなデータパーティショニング手法を比較する (適切なパーティショニング列の判定など)
- 個々のパートファイルのサイズを手動で制御しながら、PySpark DataFrames をディスクに書き込む方法を述べる
- Spark テーブル (タイプ 1) の複数のレコードを更新するためのいくつかの手法について述べる
- 構造化ストリーミングと Delta Lake によって可能になる一般的な設計パターンを実装する
- stream-static 結合と Delta Lake の使用において、状態情報を調べ、調整する
- stream-static 結合を実装する
- Spark 構造化ストリーミングを使用した重複排除のために必要なロジックを実装する
- Delta Lake テーブルに対して CDF を有効にし、通常の構造化ストリーミングの読み取りからの増分フィードの代わりに、CDC 出力を処理するようにデータ処理ステップを再設計する
- CDF を活用して削除を伝播しやすくする
- データの適切なパーティショニングによって、データのアーカイブや削除が簡単になることを示す
- 「小さい」こと (小さなファイル、スキャンのオーバーヘッド、過度なパーティショニングなど) によって、Spark クエリーにパフォーマンスの問題が生じることを説明する

セクション **3:** データモデリング

- ブロンズからシルバーへのプロモーションの際のデータ変換の目的を説明する
- チェンジデータフィード (CDF) によって、Lakehouse アーキテクチャ内での更新や削除の伝播に関する従来の問題がどのように解消されるかを説明する
- Delta Lake クローンを使用して、シャロークローンとディープクローンがソース/ターゲットテーブルとどのように相互作用するかを学ぶ

- マルチプレックスブロンズテーブルを設計して、ストリーミングワークロードを本番運用する際の一般的な落とし穴を回避する
- マルチプレックスブロンズテーブルからデータをストリーミングする際のベストプラクティスを実装する
- 増分処理、品質強制、重複排除を適用して、データをブロンズからシルバーに処理する
- Delta Lake のさまざまなアプローチの利点と制約に基づき、データ品質を強制する方法を情報に基づいて判断する
- 外部キー制約が存在しないことで生じる問題を回避してテーブルを実装する
- Delta Lake テーブルに制約を追加して、不良データが書き込まれることを防ぐ
- ルックアップテーブルを実装し、正規化データモデルに関するトレードオフについて説明する
- ストリーミングワークロードおよびバッチワークロードで、Delta Lake を使用してさまざまな Slowly Changing Dimension テーブルを実装するために必要なアーキテクチャと操作を図示する
- SCD タイプ 0、1、2 テーブルを実装する

セクション **4:** セキュリティとガバナンス
- 動的ビューを作成してデータマスキングを実行する
- 動的ビューを使用して行と列へのアクセスを制御する

セクション **5:** モニタリングとログ記録
- パフォーマンス分析、アプリケーションのデバッグ、Spark アプリケーションのチューニングに役立つ Spark UI の要素について説明する
- クラスターに対して実行されたステージとジョブのイベントタイムラインとメトリクスを調べる
- Spark UI、Ganglia UI、クラスター UI に示された情報から結論を導き出して、パフォーマンスの問題を評価し、失敗したアプリケーションをデバッグする
- 本番運用のストリーミングジョブのコストおよびレイテンシー SLA を管理するためのシステムを設計する
- ストリーミングジョブとバッチジョブをデプロイし、監視する

セクション **6:** テストとデプロイメント
- ノートブック依存関係パターンを用いて、Python ファイルの依存関係を使用する
- Wheelとしてメンテナンスされている Python コードを、相対パスを使用して直接インポートする
- 失敗したジョブを修復して再実行する
- 一般的なユースケースとパターンに基づいてジョブを作成する
- 複数の依存関係を持つマルチタスクジョブを作成する
- 本番運用のストリーミングジョブのコストおよびレイテンシー SLA を管理するためのシステムを設計する
- Databricks CLI を構成して、ワークスペースやクラスターを操作する基本的なコマンドを実行する
- CLI からコマンドを実行して、Databricks ジョブをデプロイし、監視する
- REST API を使用してジョブを複製し、実行を開始して、実行出力をエクスポートする

サンプル問題

これらの問題は旧バージョンの試験から削除されたものです。試験ガイドに記載されている目的を示し、各目的に対応するサンプル問題を提示することを意図としています。試験ガイドには、試験の出題対象になる可能性がある目的の一覧が記載されています。認定試験の準備を行う際は、試験ガイドの「試験の概要」を確認することをお勧めします。

問題 **1**
目的*:* クエリーによって作成された *Delta Lake* テーブルに対してコマンドを実行した結果を確認する

次のクエリーによって Delta Lake テーブルが作成されました。

```
CREATE TABLE dev.my_table
USING DELTA
LOCATION "/mnt/dev/my_table"
```

このテーブルは他の人が使用する必要があり、名前がわかりにくいと判断されたため、以下のコードが実行されました。

```
ALTER TABLE dev.my_table RENAME TO dev.our_table
```

2 番目のコマンドの実行後に起きる結果は次のうちどれですか。

A. テーブル名の変更が Delta トランザクションログに記録される。
B. メタストア内のテーブル参照が更新され、すべてのデータファイルが移動される。
C. メタストア内のテーブル参照が更新され、データは変更されない。
D. 名前が変更されたテーブルに対する新しい Delta トランザクションログが作成される。
E. 関連するすべてのファイルとメタデータが削除され、1 つの ACID トランザクションで再作成される。

問題 **2**
目的*: Delta* テーブルにデータを挿入する際に、以前に処理されたレコードに基づいてデータの重複を排除する

データエンジニアが、単一のソースから遅延して到着する重複レコードを識別できる ETL ワークフローを開発しようとしています。データエンジニアは、バッチ内でレコードの重複排除を行えることは知っていますが、別の解決方法を探しています。

データエンジニアが、Delta テーブルにデータを挿入する際に、以前に処理されたレコードに基づいてデータの重複を排除するために使用できるアプローチは、次のうちどれですか。

A. バッチが完了するたびに Delta テーブルを **`VACUUM`** する。
B. Delta Lake のスキーマ強制を利用して、重複レコードを防止する。
C. **`delta.deduplicate = true`** 構成を設定する。
D. 一意のキーにFULL OUTER JOIN を実行し、既存のデータを上書きする。
E. 一意のキーに対する一致条件を使用して INSERT-ONLY MERGE を実行する。

問題 **3**

目的: *Lakehouse* のすべてのテーブルを外部アンマネージド *Delta Lake* テーブルとして構成する方法を確認する

データアーキテクトは、Lakehouse のすべてのテーブルが外部アンマネージド Delta Lake テーブルとして構成される必要があると指定しています。

この要件を満たすために使用できるアプローチは次のうちどれですか。

A. テーブルを作成する際に、常に **`LOCATION`** キーワードを使用する。
B. テーブルを作成する際に、**`EXTERNAL`** キーワードを **`CREATE TABLE`** ステートメントで使用する。
C. ワークスペースを構成する際に、外部クラウドオブジェクトストレージが必ずマウントされているようにする。
D. データベースを作成する際に、常に **`LOCATION`** キーワードを使用する。
E. テーブルを作成する際に、常に **`LOCATION`** キーワードと **`UNMANAGED`** キーワードを使用する。

問題 **4**
目的: *Databricks* ジョブの権限管理について説明する

データエンジニアリングチームが、Databricks Workflows の所有権を、別のチームに異動したメンバーから移転しようとしています。しかし、Databricks ジョブの権限管理がどのように機能するのかよくわかりません。

Databricks ジョブの権限管理についての正しい記述は次のうちどれですか。

A. Databricks ジョブの作成者は常に「所有者」権限を持つ。この構成は変更できない。
B. Databricks ジョブの所有者は常に 1 人であり、「所有者」権限をグループに割り当てることはできない。
C. デフォルトの "admins" グループを除くと、ジョブに対する権限は個人のユーザーのみに付与できる。
D. グループに「所有者」権限を付与できるのはワークスペース管理者だけである。
E. ユーザーがジョブの所有権をグループに移転できるのは、そのユーザーがそのグループのメンバーでもある場合だけである。

問題 **5**
目的: *Python* パッケージをインストールする方法を確認する
データエンジニアは、データを処理するために Python パッケージを使用する必要があります。そのために、現在アクティブなクラスターのすべてのノードに Python パッケージをインストールする必要があります。

現在アクティブなクラスターのすべてのノードにノートブックレベルの Python パッケージをインストールする方法は、次のうちどれですか。

A. ノートブックのセルで **`%pip install`** を使用する

B. ノートブックのセルで **`%sh pip install`** を使用する
C. ノートブックのセットアップスクリプトで **`source env/bin/activate`** を実行する
D. クラスター UI を使用して PyPI からライブラリをインストールする
E. ノートブックのセルで **`b`** を使用する

解答
問題 1: C
問題 2: E
問題 3: A
問題 4: B
問題 5: A