

IT 資格 道 場

Databricks Certified Data Engineer Associate 学習用テキスト

予備知識ゼロから合格ラインへ —— 読んで理解する1冊目

無料サンプル(第1章まで)

Databricks Certified Data Engineer Associate 出題範囲(Exam Guide・2026 年 8
月 6 日更新版)準拠
2026年10月版

ご利用にあたって

- このテキストは「IT資格道場」(www.shikaku-doujou.com)の購入者限定テキストのうち、学習用テキストの第1章までを収録した**無料サンプル**です。どなたでもご覧いただけます。
- 無料サンプルも著作物です。転載・改変しての再配布・二次販売はできません。
- AI モデルの学習データとしてのご利用(学習目的の入力・データセット化を含む)はできません。
- 製品版(学習用テキスト・頻出度別ノート・暗記用ノートの3点)は、ご購入者を識別できる透かし入りのPDFとして提供されます。
- 本PDFはスマートフォン・タブレットでの閲覧に合わせたA5版面・文字サイズで組版しています。

目次

この試験の範囲 [overview]

第1章 Databricks Intelligence Platform [1.1/1.2]

アカウント・ワークスペース・メタストア・プレーン [1.1]

レイクハウスとメダリオンアーキテクチャ [1.1]

Delta Lake と ACID トランザクション [1.1]

タイムトラベルと DESCRIBE HISTORY と RESTORE [1.1]

Unity Catalog の名前空間・オブジェクト・リネージ・監査 [1.1]

汎用コンピュートとジョブコンピュート [1.2]

サーバーレスコンピュート [1.2]

SQL ウェアハウス [1.2]

アクセスモード [1.2]

自動スケール・自動終了・単一ノード・プール・Photon・ポリシー [1.2]

負荷からコンピュートを選ぶ [1.2]

サンプルはここまでです

3冊の使い方は、ダウンロード欄の案内を参照してください。

SAMPLE

この試験の範囲 [overview]

Databricks Certified Data Engineer Associate は、Databricks の Data Intelligence Platform で、データエンジニアが日常に行う作業を問う試験です。試験コードは無く、名前と呼びます。出題は採点される 45 問の選択式で、試験時間は 90 分、日本語でも受けられます。合格点は Databricks が公表していません。

出題範囲は 7 つの分野に分かれます。この教材も、同じ順に章を立てます(下の表)。配点は、Exam Guide(2026 年 8 月 6 日更新版)の割合です。

分野	配点
1 Databricks Intelligence Platform	6%
2 データの取り込みとロード(Data Ingestion and Loading)	21%
3 データ変換とモデリング(Data Transformation and Modeling)	22%
4 Lakeflow Jobs の利用(Working with Lakeflow Jobs)	16%
5 CI/CD の実装(Implementing CI/CD)	10%
6 トラブルシューティング、モニタリング、最適化	10%
7 ガバナンスとセキュリティ(Governance and Security)	15%

- Exam Guide の細目: 33 個
- 章と節の見出しの後ろにある [2.1] のような印: その細目の番号

第1章 Databricks Intelligence Platform [1.1/1.2]

アカウント・ワークスペース・メタストア・プレーン [1.1]

Databricks アカウントは、組織全体の Databricks を管理する最上位の構成要素です。アカウントレベルで扱うのは、ID とアクセス、ワークスペースの管理、Unity Catalog のメタストアの管理、使用量の管理（課金・コンプライアンス・ポリシー）です。ワークスペースは、取り込み、対話的な探索、スケジュールされたジョブ、機械学習の学習などを動かす共同作業の環境です。1つのアカウントに、複数のワークスペースと複数のメタストアを置けます。メタストアは、1つのワークスペースに閉じません。同じリージョンの複数のワークスペースに紐付けられ、紐付けたワークスペースは同じデータビューを持ちます。データへのアクセス制御はまとめて管理できます。[^c1-1]

Databricks は、コントロールプレーンとコンピュートプレーンの2つで動きます。コントロールプレーンは、Databricks が管理するバックエンドのサービスで、Web アプリもここに 있습니다。自分のクラウドアカウントには置かれませんが、Databricks のアカウント側にあります。コンピュートプレーンは、データが処理される場所です。クラシックコンピュートのリソースは、自分のクラウドアカウントの classic compute plane にあります。サーバーレスは、Databricks のアカウント内の serverless compute plane で動くので、自分のアカウントに VM は作られません。「VM が自分のアカウントに作られるのはどちらか」と問われたら、答えはクラシックです。

ストレージの置き場所は、ワークスペースの種類で変わります。クラシックのワークスペースには、自分のクラウドアカウントに workspace storage bucket が付き、削除や変更をするとワークスペースを復旧できません。サーバーレスのワークスペースは default storage を使います。ワークスペース内部のデー

タと Unity Catalog のデータ資産のための、完全マネージドな保存先です。自分のクラウドストレージにも接続できます。DBFS root と DBFS mounts にデータを置く方法は、非推奨のパターンです。DBFS 経由の保存とアクセスは勧められません。Unity Catalog で管理します。

Git folder は、リモートの Git リポジトリと同期してバージョン管理するフォルダで、旧称は Repos でした。Lakeflow pipelines は、Apache Spark Declarative Pipelines (SDP) の上に作られた宣言的なフレームワークで、信頼性・保守性・テスト容易性の高いパイプラインを組みます。課金の単位は DBU で、VM のインスタンスタイプに基づく1時間あたりの処理能力です。^[^c1-2]

レイクハウスとメダリオンアーキテクチャ [1.1]

メダリオンアーキテクチャは、データの品質を段階的に上げる設計パターンです。bronze が生 (raw)、silver が検証済み (validated)、gold が強化済み (enriched) を表します。データは Bronze から Silver、Gold へ進むので、マルチホップアーキテクチャとも呼ばれます。採用は推奨のベストプラクティスで、必須ではありません。^[^c1-3]

層	中身	主な利用者
Bronze	生データの取り込み。クレンジングはしない	データエンジニア、データ運用、コンプライアンスと監査
Silver	クレンジングと検証 (null の除去、不正レコードの隔離、結合)	データエンジニア、アナリスト、データサイエンティスト
Gold	ディメンショナルモデリングと集計。データセットは Silver より少ない	ビジネスアナリスト、BI 開発者、経営層、運用チーム

層は、たとえば ops カタログの ops.bronze、ops.silver、ops.gold のようにスキーマで分けます。

Delta Lake と ACID トランザクション [1.1]

Delta Lake は、レイクハウスのテーブルの土台になる最適化されたストレージ層です。データの実体は Parquet のファイルのままで、置き換えません。ファイルベースのトランザクションログで拡張し、ACID トランザクションとスケーラブルなメタデータ処理を提供します。Databricks のテーブルは、指定しない限りすべて Delta テーブルです。Spark の API と完全に互換で Structured Streaming と密に統合されているため、1つのデータコピーをバッチとストリーミングの両方に使えます。書き込み時にスキーマを検証するので、要件に合わないデータは書き込めません。書き込みのたびに新しいテーブルバージョンができ、過去バージョンを照会できます。[^c1-4]

ACID は、原子性 (処理が完全に成功するか完全に失敗する)、一貫性 (同時に動く処理から見える状態)、分離性 (同時処理の衝突)、永続性 (コミット済みの変更が残る) の頭文字です。原子性はトランザクションログが担います。処理中のデータファイルはテーブルのディレクトリに書かれ、処理が終わった時点で、そのファイルのパスを含むエントリがログにコミットされます。ログに載らないファイルはテーブルの一部になりません。途中で失敗してもテーブルは壊れず、残ったファイルは VACUUM が削除します。永続性は、データとログを置くクラウドオブジェクトストレージの耐久性を引き継ぎます。[^c1-5]

同時書き込みは楽観的同時実行制御で扱い、ロックがないのでデッドロックも起きません。書き込みは、読み取り、書き込み、検証とコミットの3段階です。コミットの直前に、読んだ時点以降にコミットされた他の変更とぶつからないかを確認めます。ぶつからなければ新しいバージョンになり、ぶつかれば同時変更の例外で書き込みがエラーになります。既定の分離レベルは、読み取り

がスナップショット分離、書き込みが write-serializable 分離です。既定では、1つの SQL 文が1つのテーブルに対する原子的なトランザクションで、複数のテーブルには広がりません。複数のテーブルにまたがる処理は `BEGIN ATOMIC ... END;` で1つにでき、対象のテーブルでカタログコミットを有効にします。主キー・外部キーの制約は情報提供のみで、強制されません。

テーブルのディレクトリの `_delta_log` には、コミットごとの JSON が、20桁にゼロ埋めしたバージョン番号で並びます (`00000000000000000000000042.json`)。履歴の全 JSON を読まずに済むよう、Parquet のチェックポイント (`00000000000000000000000010.checkpoint.parquet`) にも集約されます。[^{^c1-6}]

タイムトラベルと DESCRIBE HISTORY と RESTORE [1.1]

DESCRIBE HISTORY は、各書き込みの操作・ユーザー・タイムスタンプを新しい順に返します。LIMIT 1 を付けると直近の操作だけです。過去のバージョンは、バージョン番号 (DESCRIBE HISTORY の出力から取る) かタイムスタンプで読みます。[^{^c1-7}]

```
DESCRIBE HISTORY people10m;
SELECT * FROM people10m VERSION AS OF 123;
SELECT * FROM people10m TIMESTAMP AS OF '2018-10-18T22:15:12.013Z';
SELECT * FROM people10m@v123;
SELECT * FROM people10m@201901010000000000;
RESTORE TABLE people10m TO VERSION AS OF 123;
```

@ 構文のタイムスタンプは `yyyyMMddHHmmssSSS` の形で、バージョンは `@v` の後ろに付けます。Python では `option("versionAsOf", 123)` と `option("timestampAsOf", "2019-01-01")` が対応します。タイムスタンプの式は文字列、日付、`current_timestamp() - interval 12 hours` などで、バージョンと

タイムスタンプのどちらもサブクエリは使えません。誤って消した行は、`INSERT INTO t SELECT * FROM t TIMESTAMP AS OF date_sub(current_date(), 1) WHERE userId = 111` のように、過去の状態から入れ直せます。

RESTORE は、テーブル全体を以前の状態へ戻します。MODIFY 権限が要り、タイムスタンプは `yyyy-MM-dd HH:mm:ss` か `yyyy-MM-dd` で指定します。実行すると新しいバージョンが1つ増え、古い履歴は残ります。ファイルが削除済みのバージョンには戻せません。RESTORE のログは `dataChange` が `true` なので、そのテーブルをストリーミングで読む下流の処理が、戻された行を新しいデータとして再処理し、重複することがあります。[^c1-8]

過去のバージョンが読める期間は、2つの保持の設定で決まります。

設定	既定	決めるもの
<code>delta.logRetentionDuration</code>	30日	テーブルの履歴 (DESCRIBE HISTORY で見える範囲)
<code>delta.deletedFileRetentionDuration</code>	7日	VACUUM が消さずに残すデータファイル

データファイルは VACUUM で、ログファイルはチェックポイントの後に自動で削除されます。履歴が30日残っていても、データファイルが消えたバージョンは読めません。7日より前を読むには、両方の設定を大きくします。Databricks Runtime 18.0 以降は、`deletedFileRetentionDuration` より古いバージョンを要求するクエリをブロックします。Unity Catalog のマネージドテーブルでは 12.2 以降が対象です。同じ条件で、`logRetentionDuration` は `deletedFileRetentionDuration` 以上でなければなりません。保持を延ばすと保存コストが上がります。長期のバックアップの代わりにはなりません。[^c1-9]

月曜に誤って DELETE を実行し、水曜に気づいたとします。保持が既定なら、DESCRIBE HISTORY で DELETE の直前のバージョン番号を調べ、RESTORE TABLE ... TO VERSION AS OF で戻せます。10 日後に気づいた場合、deletedFileRetentionDuration が7日のままだと、VACUUM の実行後はデータファイルが消えていて戻せません。

Unity Catalog の名前空間・オブジェクト・リネージ・監査 [1.1]

Unity Catalog は、データと AI のための統合ガバナンス層です。アクセス制御、リネージの追跡、監査ログなどを、あらゆる操作の下で自動的に適用します。2023年11月8日以降に作られたワークスペースでは自動で有効です。それより前のワークスペースは、自動では有効になりません。アップグレードが必要です。メタストア内のデータは、<catalog>.<schema>.<object> の3層の名前空間で並べます。カタログは最上位の層で組織の単位に、スキーマは其中でユースケースやチームの単位に当たり、その下にテーブル、ビュー、ボリューム、関数が並びます。[^c1-10]

メタストアは、Unity Catalog の最上位のセキュリティ保護可能オブジェクトです。1つのクラウドリージョンにスコープされるので、組織は運用するリージョンごとに1つ必要です。メタストアで付けた権限は、紐付けたすべてのワークスペースで有効です。メタストア直下には、ストレージ資格情報、外部ロケーション、接続などのオブジェクトも置かれます。権限は、ユーザー、サービスプリンシパル、グループ(プリンシパル)にしか付与できません。[^c1-11]

マネージドと外部の区別があるのは、テーブルとボリュームだけです。ビュー、モデル、関数にはありません。マネージド資産は、ガバナンスに加えて、保存先とファイルのライフサイクル(最適化・配置・削除)も Unity Catalog が管理します。外部資産では、Unity Catalog はガバナンスだけを担当し、ストレージのライフサイクルは利用者側です。どちらの場合も、データは利用者のクラウド

アカウントに残ります。所有権は利用者にあり、Databricks は保有しません。マネージドテーブルとマネージドボリュームを DROP すると、保持期間のあとでデータファイルが完全に削除されます。保持期間の日数は、文書によって 7 日と 8 日に分かれています。外部テーブルと外部ボリュームは、DROP してもファイルがそのまま残ります。[^c1-12]

オブジェクトを使うには、親カタログの USE CATALOG と親スキーマの USE SCHEMA に加えて、種類ごとの権限が要ります。

オブジェクト	必要な権限
テーブル	SELECT (読み取り)、MODIFY (書き込み)
ビュー	ビューへの SELECT。元のテーブルの権限は不要で、ビューの所有者の権限で解決される
ボリューム	READ VOLUME、WRITE VOLUME
関数	EXECUTE

カタログとスキーマに付けた権限は、その下の現在と将来のオブジェクトに継承されます。メタストアに付けた権限 (CREATE CATALOG など) は継承されません。BROWSE をカタログに付けると、データへのアクセスを与えずに、カタログ内のオブジェクトのメタデータの閲覧を許します。

ボリュームは、非表形式のデータ (構造化・半構造化・非構造化を問わない) を管理するオブジェクトで、managed と external があります。ファイルは `/Volumes/<catalog>/<schema>/<volume>/<path>/<file>` のパスで扱い、このパスは Spark、SQL、Python のどれで使っても変わりません。外部ロケーションは、クラウドストレージのパスと、そこへ入るストレージ資格情報への参照です。ストレージ資格情報は、S3 では IAM ロールに対応します。外部ボリューム

の場所は、外部ロケーションに含まれるパスでなければなりません。マネージドな保存先に、Unity Catalog 以外の ID が直接アクセスしてはいけません。権限制御を迂回でき、監査もリネージも記録されなくなります。[^c1-13]

リネージは、データがどこから来てどこへ行くかを示します。Databricks 上のクエリは列レベルまで自動で記録され、メタストアに紐付いたすべてのワークスペースで集約されます。記録の条件は、テーブルが Unity Catalog に登録されていること、クエリが Spark の DataFrame (Spark SQL を含む) か Databricks SQL で動くことです。見るには、親カタログの BROWSE 以上の権限がなければなりません。Catalog Explorer の Lineage タブか、リネージのシステムテーブルで確かめます。変更前の影響分析、結果が想定外のときの原因調査、機密データの流れの追跡に使います。[^c1-14]

監査ログは、システムテーブル `system.access.audit` に入ります (公開プレビュー)。主な列は `event_time`、`event_date`、`service_name`、`action_name`、`request_params`、`response`、`identity_metadata` です。絞り込みには `event_time` ではなく `event_date` を使うと速くなります。アカウントレベルのイベントは `workspace_id` が 0 で記録されます。[^c1-15]

汎用コンピュートとジョブコンピュート [1.2]

クラシックコンピュートには、汎用 (all-purpose) とジョブの2種類があります。汎用コンピュートは UI・CLI・REST API で作り、手動で終了と再起動ができます。複数のユーザーで共有して、共同の対話的な分析に使います。ジョブコンピュートは、ジョブの実行時にスケジューラが作り、完了すると終了します。再起動はできません。ワークロードは、データエンジニアリング (自動化・ジョブ) とデータ分析 (対話的・汎用) に分かれます。既存の汎用コンピュートでジョブを動かしても、対話的なワークロードとして扱われます。[^c1-2]

スケジュールされた非対話の処理は、ジョブコンピュートのほうが汎用コンピュートより大幅に安くなります。夜間のバッチ ETL は、汎用クラスターに載せず、ジョブコンピュートで動かします。複数のタスクで1つのジョブコンピュートを共有すると、起動が1回で済みます。共有したコンピュートは、そのジョブの1回の実行の中でしか使えません。最初のタスクで起動して最後のタスクの後で終了し、途中の空き時間はアイドルのまま動き続けます。[^c1-16]

サーバーレスコンピュート [1.2]

サーバーレスコンピュートは、Databricks が管理するオンデマンドのコンピュートです。自分のクラウドアカウントにリソースを用意せずに動かせます。対象はノートブック、ジョブ、Lakeflow pipelines です。Unity Catalog が有効なワークスペースでは自動で使えます。Unity Catalog 未対応の旧ワークスペースは、アップグレードしないと使えません。[^c1-17]

項目	制限
言語・API	R は非対応。Spark Connect の API だけで、Spark RDD API は非対応
既定の挙動	ANSI mode が既定で有効。セッションのタイムゾーンは Etc/UTC
ストリーミングのトリガー	Trigger.AvailableNow() (推奨) と Trigger.Once() (非推奨だが対応) だけ。Trigger.Continuous と Trigger.ProcessingTime は非対応
最大実行時間	7日。超えると終了され、再試行されない。分割するか classic を使う
非対応の機能	コンピュートポリシー、コンピュートスコープの init スクリプトとライブラリ

トリガーを指定しないと Spark は `ProcessingTime("0 seconds")` を既定にするため、サーバーレスでは対応するトリガーを必ず指定します。非対応のトリガーでは `INFINITE_STREAMING_TRIGGER_NOT_SUPPORTED` のエラーになります。Lakeflow pipelines の処理モードには、この制限は当てはまりません。課金の記録が `billable usage` のシステムテーブルに反映されるまで、最大24時間かかることがあります。コストは `system.billing.usage` を照会して確かめます。[^c1-18]

SQL ウェアハウス [1.2]

SQL ウェアハウスには Classic、Pro、Serverless の3種類があり、使えるならサーバーレスが勧められています。[^c1-19]

種類	Photon	Predictive IO	Intelligent Workload Management	起動	コンピューター層
Serverless	対応	対応	対応	2～6秒	Databricks のアカウント
Pro	対応	対応	非対応	約4分	自分のクラウドアカウント
Classic	対応	非対応	非対応	約4分	自分のクラウドアカウント

サーバーレスは、起動が速く、需要に応じて素早くスケールします。Pro を選ぶのは、そのリージョンでサーバーレスが使えない場合と、独自のネットワークで社内やオンプレミスのデータベースに接続する場合（フェデレーションやハイ

ブリッド構成)です。Classic は、入門的な性能での対話的なデータ探索に向きます。

UI から作ると、サーバーレス対応のリージョンでは既定がサーバーレスです。SQL warehouses API を既定のパラメータで呼ぶと classic で、サーバーレスにはなりません。API でサーバーレスにするには、`enable_serverless_compute` を true に、`warehouse_type` を pro にします。Lakehouse Real-Time (Lakehouse//RT) はベータのサーバーレスウェアハウスで、低レイテンシ・高同時実行の SELECT クエリだけに対応し、他のクエリは実行できません。

アクセスモード [1.2]

Standard は複数のユーザーが同時に使え、Lakeguard でユーザーを分離します。Dedicated は単一のユーザーまたはグループに割り当てます。言語は Standard が Python・SQL・Scala、Dedicated が R を加えた4つです。特別な制限がない限り、Standard が勧められています。Standard は ML ランタイム、GPU、spark-submit のタスク、R、RDD の API に対応しません。これらが要る処理は Dedicated を使います。Standard で DBUtils などがクラウドストレージを読むには、外部ロケーションが要ります。既定の Auto は、通常は Standard です。ML ランタイム、GPU のインスタンス、Databricks Runtime 14.3 未満を選ぶと、Dedicated になります。[^c1-20]

自動スケール・自動終了・単一ノード・プール・Photon・ポリシー [1.2]

自動スケールは、最小と最大のワーカー数を決めておくと、負荷に合わせてワーカーを増減します。Premium 以上のワークスペースは最適化オートスケール、標準プランは標準オートスケールです。

	最適化オートスケール	標準オートスケール
増やし方	min から max まで最大2回のスケールイベント	8ノードの追加から始めて指数的に増やす
縮小	ジョブは直近40秒、汎用は直近150秒の低利用で縮小	ノードの90%が10分間ビジーでなく、30秒以上アイドルなら縮小

spark-submit のジョブでは自動スケールが使いません。Spark の Dynamic Allocation は同時に有効にしません。自動終了は、最後のコマンドから指定した分数を過ぎると、コンピューターを終了します。スポットインスタンスを使っても、ドライバーは常にオンデマンドです。[^c1-21]

単一ノードは、ドライバーがマスターとワーカーを兼ねて、ワーカーノードを持ちません。小さなデータや、分散しない処理に向きます。マルチノードには変換できません。ワーカーが0のコンピューターでは、Spark 以外のコマンドはドライバーで動きますが、Spark のコマンドは失敗します。Spark の処理を小さく動かすなら、単一ノードを使います。

プールは、待機中のすぐ使えるインスタンスの集まりで、起動とスケールの時間を短くします。アイドルのインスタンスに DBU は課金されません。クラウド側の課金は発生します。プールの作成には Allow pool creation の権限が要り、既定ではワークスペース管理者だけが持ちます。権限レベルは NO PERMISSIONS、CAN ATTACH TO、CAN MANAGE の3つです。サーバーレスで動く負荷には、プールよりサーバーレスが勧められています。[^c1-22]

Photon は、Databricks のネイティブなベクトル化クエリエンジンです。Databricks Runtime 9.1 LTS 以上のクラシックコンピューターで既定で有効で、Use Photon Acceleration のチェックボックスで切り替えます。API で作るときは `runtime_engine` を PHOTON にします。長時間・大規模なクエリで効果

が大きく、2秒未満で終わるクエリでは目立った改善が出ません。SQL ウェアハウス、サーバーレス、サーバーレスの pipelines で有効で、ステートフルなストリーミングには非対応です。[^c1-23]

コンピュートポリシーは、ユーザーが作るコンピュートの設定を制限するルールです。ワークスペース管理者が作り、設定値の固定、作成数の上限、1時間あたりの DBU の上限などでコストと構成を揃えます。Unrestricted cluster creation の権限がないユーザーは、付与されたポリシーでしか作れません。全ユーザーが既定で使える Personal Compute は、単一マシンのコンピュートです。[^c1-24]

負荷からコンピュートを選ぶ [1.2]

タスク	推奨	補足
ノートブック、Python スクリプト、wheel	サーバーレスジョブ	classic のジョブ、汎用でも可
SQL、dbt	サーバーレス SQL ウェアハウス	Pro でも可
Lakeflow pipelines	サーバーレス pipeline	classic pipeline でも可
JAR、Spark Submit	classic のジョブ	

夜間に動く Python の ETL は、ノートブックのタスクなのでサーバーレスジョブが第一候補です。ただし R を使う場合や、Trigger.ProcessingTime の常時ストリーミングが要る場合は、サーバーレスで動きません。classic のジョブコンピュートにします。独自のネットワークでオンプレミスのデータベースに接続する SQL は Pro、ML ランタイムや GPU が要る処理は Dedicated、複数人が探索するノートブックは Standard の汎用コンピュートです。[^c1-25]

サンプルはここまでです

続き(第2章～巻末)は、Databricks Data Engineer Associate 問題集のご購入で、頻出度別ノート・暗記用ノートと合わせて3点すべてダウンロードできます。

SAMPLE

Databricks Certified Data Engineer Associate 学習用テキスト(無料サンプル)

版

2026年10月版(Databricks Certified Data Engineer Associate 出題範囲(Exam Guide・2026 年 8 月 6 日更新版)準拠)

発行

IT資格道場(<https://www.shikaku-doujou.com>)

お問い合わせ

info@shikaku-doujou.com

Databricks、Unity Catalog、Lakeflow、Photon および Databricks の製品名・サービス名は、Databricks, Inc. の米国およびその他の国における商標または登録商標です。Apache、Apache Spark、Spark は Apache Software Foundation の商標です。記載されているその他の会社名・製品名・サービス名は、各社の商標または登録商標です。

本教材は公式シラバスの内容を基に作成したもので、Databricks, Inc. との提携・承認関係はありません。

本文・図表・設問はすべてオリジナルの著作物です。無断での転載・複製・再配布・二次販売はできません。お気づきの点は info@shikaku-doujou.com までご連絡ください。