databricks

Guia do exame da Databricks

Databricks Certified Data Engineer Professional



Fornecer feedback sobre o guia do exame

Informações de atualização

Atualizações	Data
Exame publicado	20 de novembro de 2026
Guia do exame atualizado	20 de novembro de 2026

Verifique o guia do exame em nosso site duas semanas antes da data do seu exame para garantir que você esteja se preparando para a versão atual do exame.

Objetivo deste guia do exame

O objetivo deste guia do exame é fornecer uma visão geral do exame e do que ele abrange, para ajudar você a avaliar sua preparação para o exame. Este documento é atualizado sempre que há alterações no exame (e indica quando essas alterações entram em vigor) para que você possa se preparar com confiança. Este guia abrange a versão do exame que está ativa a partir da **data de vigência** listada acima. Consulte este guia novamente duas semanas antes do seu exame para garantir que você tenha a versão mais recente.

Descrição do público-alvo

O exame Databricks Certified Data Engineer Professional avalia as habilidades avançadas do candidato para construir, otimizar e manter soluções de engenharia de dados prontas para produção na plataforma Databricks. Os candidatos aprovados demonstram domínio dos principais recursos da plataforma, como Databricks SQL, Unity Catalog, Auto Loader, Lakeflow Pipelines, Databricks Compute (incluindo serverless), Lakeflow Jobs, Lakehouse Federation, OpenSharing (D2D e Databricks-to-Open), Clean Rooms e a arquitetura Medallion. Espera-se que os profissionais aprovados neste exame sejam capazes de projetar pipelines de ETL seguros, confiáveis e econômicos; processar dados complexos de diversas fontes usando Python e SQL; e aplicar as melhores práticas em gerenciamento de schemas, observabilidade, governança,

compliance e otimização de desempenho. Os candidatos também são avaliados quanto à implementação de cargas de trabalho de streaming, orquestração de workflows, caching, técnicas de depuração, técnicas de otimização, uso de DevOps e CI/CD, implantação com ferramentas como a Databricks CLI e a REST API, e aplicação de AI Functions e Declarative Automation Bundles. Espera-se que os profissionais aprovados neste exame de certificação sejam capazes de concluir tarefas avançadas de engenharia de dados usando o Databricks e suas ferramentas associadas.

Sobre o exame

Item	Detalhe
Número de questões pontuadas	60 questões pontuadas de múltipla escolha
Possíveis questões não pontuadas	Podem ser incluídas questões não pontuadas (consulte "Conteúdo não pontuado" abaixo)
Número total de questões que podem aparecer	60 questões pontuadas, além das questões não pontuadas
Limite de tempo	120 minutos (o tempo adicional para as questões não pontuadas já está incluído)
Taxa de inscrição	US$ 200, mais impostos aplicáveis, conforme exigido pela legislação local
Forma de aplicação	On-line com supervisão ou em centro de testes
Idiomas disponíveis	Inglês
Materiais de consulta	Não permitidos
Pré-requisito	Não há pré-requisito obrigatório, mas o treinamento relacionado é altamente recomendado
Experiência recomendada	É altamente recomendável ter um ano de experiência prática nas tarefas de engenharia de dados listadas na estrutura do exame
Validade	2 anos

Recertificação: a recertificação é exigida a cada dois anos para manter sua certificação. Para se recertificar, você deve fazer a versão do exame em vigor.

Conteúdo não pontuado: os exames podem incluir questões não pontuadas para coletar informações estatísticas para uso futuro. Essas questões não são identificadas no exame e não afetam sua pontuação; o limite de tempo inclui tempo adicional para respondê-las.

Preparação recomendada

Treinamento

- Ministrado por instrutor: Advanced Data Engineering with Databricks
- No seu próprio ritmo (disponível na Databricks Academy):
    - Advanced Techniques with Apache Spark™ Declarative Pipelines
    - Databricks Data Privacy
    - Databricks Performance Optimization
    - Automated Deployment with Declarative Automation Bundles

AI Prep Guide para qualquer exame Databricks

O Databricks AI Prep Guide mostra como transformar o chatbot de IA que você já usa em um tutor personalizado para qualquer certificação Databricks: alimente-o com o guia oficial do exame, identifique as respostas desatualizadas que a IA tende a fornecer, descubra seus pontos fracos e combine tudo isso com a experiência prática que todo exame Databricks exige.

Exercícios práticos

Não sabe o que praticar? O AI Prep Guide (consulte a seção AI Prep Guide acima) inclui, no passo 5 ("Hands-on minimum"), um prompt que faz um chatbot de IA gerar uma lista de verificação personalizada de 6 a 10 tarefas práticas mapeadas para os objetivos deste exame, cada uma pequena o suficiente para ser concluída na Databricks Free Edition.

Documentação da Databricks

Ao se preparar, estude a documentação oficial da Databricks para as áreas da plataforma abordadas neste exame. Ela é a fonte oficial e mais atualizada. Use-a para avaliar com que profundidade estudar cada tópico e para garantir que você esteja aprendendo recursos atualizados e não obsoletos, em vez de material desatualizado.

Documentação oficial: documentação da Databricks

Estrutura do exame

Seção 1: Desenvolvimento de código para processamento de dados usando Python e SQL (23%)

- Implementar estruturas de projeto Python escaláveis para Declarative Automation Bundles (anteriormente Databricks Asset Bundles / DABs) a fim de dar suporte ao desenvolvimento modular e à integração com CI/CD.

- Aplicar técnicas de solução de problemas a conflitos de dependências e falhas de instalação de bibliotecas externas (PyPI, wheels locais, arquivos de código-fonte) em ambientes serverless, de pipeline e implantados via bundle.
- Desenvolver funções definidas pelo usuário (UDFs) usando pandas, Python e SQL, incluindo funções do Unity Catalog conforme as restrições apresentadas.
- Criar pipelines de dados de streaming prontos para produção usando Lakeflow Pipelines e Auto Loader.
- Criar e automatizar cargas de trabalho de ETL usando Lakeflow Jobs por meio da UI, da API e da CLI.
- Escolher entre uma streaming table e uma materialized view para um determinado requisito de latência, custo e refresh.
- Implementar pipelines de CDC usando as APIs AUTO CDC (anteriormente APPLY CHANGES) no Lakeflow Pipelines, incluindo SCD Tipo 1 e Tipo 2 via stored_as_scd_type.
- Escolher entre Spark Structured Streaming e Apache Spark™ Declarative Pipelines para ETL escalável considerando restrições operacionais.
- Criar Lakeflow Jobs que usem operadores de fluxo de controle, como condições If/Else e loops For Each.
- Escolher o compute e a configuração adequados para ambientes e dependências, incluindo compute serverless (ambientes serverless, gerenciamento de dependências, modo de desempenho), tarefas de notebook com memória alta e configurações de otimização automática (por exemplo, desabilitar novas tentativas).
- Desenvolver testes unitários e de integração usando assertDataFrameEqual, assertSchemaEqual, DataFrame.transform e frameworks de teste para garantir que o código esteja correto.
- Aplicar a semântica de processamento com estado do Structured Streaming, incluindo watermarks, modos de saída, foreachBatch e checkpoints para recuperação de estado exactly-once e tolerante a falhas.

Seção 2: Ingestão e aquisição de dados (12%)

- Desenvolver pipelines de ingestão de dados para ingerir uma variedade de formatos de dados, incluindo Delta Lake, Parquet, JSON, Iceberg, CSV e binário, a partir de fontes diversas, como barramentos de mensagens (Kafka, Kinesis, Pub/Sub) e armazenamento em nuvem.
- Configurar pipelines de CDC incrementais usando Lakeflow Pipelines com Delta ou Iceberg como formato de tabela de destino.
- Configurar pipelines de ingestão com CDC a partir de fontes de banco de dados relacionais, incluindo SQL Server, MySQL e PostgreSQL, usando Lakeflow Connect.
- Implementar OpenSharing (D2D e Databricks-to-Open) e Clean Rooms para colaboração com preservação de privacidade.
- Configurar o Lakehouse Federation com governança em sistemas-fonte compatíveis, aplicando permissões do UC e credenciais no nível de conexão.

Seção 3: Manipulação de dados (12%)

- Aplicar transformações avançadas de dados, incluindo funções de janela, joins e agregações, usando Spark SQL e PySpark para processar grandes conjuntos de dados.
- Desenvolver um modelo e consultar dados semiestruturados usando o tipo de dado VARIANT e funções relacionadas (por exemplo, parse_json, variant_get).

- Aplicar AI Functions, incluindo ai_query, a fim de realizar inferência de modelos em pipelines de dados para tarefas de enriquecimento e classificação.
- Implementar expectativas de qualidade de dados no Lakeflow Pipelines para colocar em quarentena ou descartar registros inválidos, ou fazer o pipeline falhar ao encontrá-los.

Seção 4: Monitoramento e alertas (10%)

- Usar as tabelas do sistema do Databricks (billing, compute, access, lakeflow) para análise de custos, auditoria e monitoramento de cargas de trabalho.
- Usar as REST APIs, a CLI e o SDK do Databricks para monitoramento e análise de jobs e pipelines.
- Usar os logs de eventos do pipeline para monitorar a integridade do pipeline e a qualidade dos dados.
- Usar os alertas do Databricks Lakehouse para monitorar métricas de negócios governadas, qualidade de dados e sinais de anomalia, uso e custo, integridade de SQL warehouses e queries, eventos de auditoria e segurança, qualidade de agentes de IA e ramificação em Lakeflow Jobs.
- Usar a UI do Lakeflow Jobs e a Jobs API para monitorar o estado e as métricas de desempenho do job.

Seção 5: Otimização de custos e desempenho (15%)

- Explicar como as tabelas gerenciadas do Unity Catalog, o Predictive Optimization e o liquid clustering reduzem a sobrecarga operacional e de manutenção para uma determinada carga de trabalho.
- Escolher a técnica de otimização Delta adequada (deletion vectors, liquid clustering, CLUSTER BY AUTO) para um determinado padrão de acesso à tabela.
- Entender como o cache (Delta cache) melhora o desempenho de query para leituras repetidas dos mesmos dados.
- Aplicar Change Data Feed (CDF) para expor alterações no nível de linha (atualizações/exclusões) a fim de permitir um processamento incremental downstream eficiente.
- Usar o query profile para identificar gargalos de desempenho, como ineficiências de data skipping, estratégias de join e operações de shuffle.
- Comparar o liquid clustering com o particionamento/ZORDER para um determinado tamanho de tabela e padrão de query.

Seção 6: Garantia da segurança e do compliance dos dados (8%)

- Aplicar listas de controle de acesso (ACLs) com privilégio mínimo para proteger objetos protegíveis do Unity Catalog e recursos do workspace.
- Aplicar políticas de controle de acesso baseado em atributos (ABAC) com tags governadas para impor filtros de linha e máscaras de coluna em escala.
- Aplicar métodos de anonimização e pseudonimização, como hashing, tokenização, supressão e generalização, a dados confidenciais (por exemplo, usando máscaras de coluna e recursos relacionados do Unity Catalog).
- Implementar um pipeline de dados em conformidade que aplique controles de detecção e mascaramento de PII em cargas de trabalho batch e streaming usando recursos do Unity Catalog.

- Implementar uma estratégia de expurgo de dados que satisfaça as políticas de retenção e exclusão de dados (por exemplo, direito ao apagamento do GDPR, exclusão por direito ao esquecimento do GDPR) usando os recursos do Delta Lake e do Unity Catalog.

Seção 7: Governança de dados (5%)

- Demonstrar compreensão das tags e dos comentários do Unity Catalog como mecanismos para adicionar metadados a objetos protegíveis a fim de melhorar a descoberta de dados.
- Demonstrar compreensão do modelo de herança de permissões do Unity Catalog como mecanismo para gerenciar o controle de acesso em catálogos, schemas e objetos.

Seção 8: Depuração e implantação (10%)

- Identificar informações de diagnóstico pertinentes usando Spark UI, logs de cluster, tabelas do sistema e query profiles para solucionar erros.
- Analisar os erros e corrigir as execuções de job com falha usando reparos de job e substituições de parâmetro.
- Implantar recursos do Databricks usando Declarative Automation Bundles (anteriormente Databricks Asset Bundles).
- Configurar workflows de CI/CD baseados em Git usando Git folders do Databricks (anteriormente Repos) para implantar notebooks e código.

Seção 9: Modelagem de dados (5%)

- Projetar layouts escaláveis de tabelas Delta/Iceberg para grandes ativos de dados, mapeando o particionamento à granularidade dos dados, alinhando o clustering aos padrões de acesso por relacionamento e mantendo tamanhos de arquivo equilibrados por meio de compactação.
- Projetar modelos dimensionais para cargas de trabalho analíticas, utilizando materialized views para agregação pré-computada e metric views do Unity Catalog para definições de métricas governadas e reutilizáveis, a fim de garantir queries e agregações eficientes.

Questões de exemplo

A finalidade é mostrar a você os objetivos conforme estão descritos no guia do exame e apresentar uma questão de exemplo alinhada a cada objetivo. O guia do exame lista os objetivos que podem ser abordados em um exame; a melhor forma de se preparar é revisar, mais acima, a estrutura completa do exame correspondente à data do seu exame.

As questões de exemplo abaixo estão alinhadas com a estrutura do novo exame. Se o seu exame for até **8 de outubro de 2026**, essas questões ainda são úteis para praticar, mas algumas podem abordar objetivos que não estão no seu exame. A estrutura do seu exame é a lista oficial do que será avaliado.

Questão 1

Objetivo: *Escolher entre uma streaming table e uma materialized view para um determinado requisito de latência, custo e refresh.*

Um objeto da camada Gold em um Lakeflow Pipeline agrega os totais diários de pedidos a partir de uma fonte Bronze. Os dashboards de BI downstream o leem de forma programada, e o agregado deve sempre refletir o histórico completo, incluindo atualizações de dias anteriores que chegam com atraso, e não apenas as linhas recém-adicionadas.

Qual objeto o engenheiro deve definir?

A. Uma streaming table, porque processa cada linha da fonte exatamente uma vez e é a opção de menor custo para qualquer agregação.
B. Uma temporary view, porque sempre reflete os dados mais recentes no momento da query, sem custo de armazenamento.
C. Uma materialized view, porque mantém incrementalmente o resultado agregado completo, pode ser programada para atualização e reflete alterações que chegam com atraso.
D. Uma streaming table com um merge em foreachBatch, porque materialized views não podem realizar agregações.

Questão 2

Objetivo: *Aplicar a semântica de processamento com estado do Structured Streaming, incluindo watermarks, modos de saída, foreachBatch e checkpoints para recuperação de estado exactly-once e tolerante a falhas*.

Uma query de Structured Streaming realiza uma agregação em janela de tempo de evento. Ela deve limitar o crescimento do estado causado por eventos muito atrasados e deve retomar o processamento após uma falha do driver sem contar resultados em duplicidade.

Qual configuração atende aos dois requisitos?

A. Defina um watermark na coluna de tempo de evento e conte com o checkpoint da query para restaurar os offsets e o estado na reinicialização.
B. Use o modo de saída complete sem watermark para reter todo o estado e garantir a exatidão.
C. Aumente spark.sql.shuffle.partitions e desative o checkpointing para acelerar a recuperação.
D. Envolva a agregação em um foreachBatch e faça o commit manual dos offsets da fonte em uma tabela Delta.

Questão 3

Objetivo: *Configurar pipelines de ingestão com CDC a partir de fontes de banco de dados relacionais, incluindo SQL Server, MySQL e PostgreSQL, usando Lakeflow Connect*.

Um engenheiro de dados deve replicar continuamente eventos de alteração (inserções, atualizações e exclusões) de um banco de dados PostgreSQL operacional para o lakehouse com o mínimo de código personalizado.

Qual abordagem atende ao requisito?

A. Programe uma leitura completa da tabela via JDBC todas as noites e sobrescreva a tabela de destino a cada noite.
B. Configure um conector gerenciado do Lakeflow Connect para PostgreSQL para ingerir change data capture no lakehouse.
C. Exporte o banco de dados para CSV no armazenamento em nuvem e ingira os arquivos com o Auto Loader.
D. Use o OpenSharing para compartilhar as tabelas PostgreSQL diretamente com o lakehouse.

Questão 4

Objetivo: *Desenvolver um modelo e consultar dados semiestruturados usando o tipo de dado VARIANT e funções relacionadas (por exemplo, parse_json, variant_get).*

Uma tabela Bronze armazena payloads JSON brutos de estruturas variadas em uma única coluna de string. O engenheiro deseja armazenamento eficiente e extração rápida de campos sem precisar declarar um schema fixo previamente.

O que o engenheiro deve fazer?

A. Faça o parse dos payloads para uma coluna VARIANT com parse_json e extraia campos usando variant_get e acesso por caminho com dois-pontos.
B. Expanda o JSON em uma linha por chave usando um schema STRUCT fixo.
C. Mantenha os payloads como STRING e use regexp_extract para cada campo no momento da query.
D. Converta o JSON para Parquet com um schema estrito e fixo antes da ingestão.

Questão 5

Objetivo: *Usar as tabelas do sistema do Databricks (billing, compute, access, lakeflow) para análise de custos, auditoria e monitoramento de cargas de trabalho.*

A equipe de um engenheiro de dados precisa atribuir o gasto de DBU do mês passado a jobs e SQL warehouses específicos e identificar as cargas de trabalho mais caras, usando dados governados já disponíveis no Unity Catalog.

Qual fonte a equipe deve consultar?

A. A linha do tempo de eventos da Spark UI para cada cluster.
B. Logs do driver por cluster baixados do workspace.
C. A tabela system.billing.usage em join com as tabelas do sistema de compute e preços no Unity Catalog.
D. A exportação em PDF do faturamento no console da conta.

Questão 6

Objetivo: *Escolher a técnica de otimização Delta adequada (deletion vectors, liquid clustering, CLUSTER BY AUTO) para um determinado padrão de acesso à tabela.*

Um engenheiro de dados descobre que uma tabela Delta grande recebe operações de MERGE pequenas e frequentes que atualizam e excluem linhas individuais. A equipe quer evitar a reescrita de arquivos de dados inteiros a cada alteração, mantendo as leituras rápidas.

Qual técnica resolve a amplificação de escrita?

A. Particione a tabela pela sua chave primária.
B. Ative os deletion vectors para que atualizações e exclusões sejam marcadas no nível da linha sem reescrever imediatamente os arquivos inteiros.
C. Execute VACUUM com retenção de zero horas após cada MERGE.
D. Desative o Predictive Optimization para evitar reescritas de arquivos em segundo plano.

Questão 7

Objetivo: *Aplicar políticas de controle de acesso baseado em atributos (ABAC) com tags governadas para impor filtros de linha e máscaras de coluna em escala.*

Um engenheiro de dados trabalha para uma organização com centenas de tabelas contendo PII e deseja uma única regra gerenciada centralmente para mascarar qualquer coluna marcada com a tag 'pii' em todas as tabelas atuais e futuras, sem alterar cada tabela individualmente.

O que o engenheiro deve implementar?

A. Execute, tabela por tabela, uma instrução ALTER COLUMN SET MASK em todas as tabelas existentes.
B. Anexe uma função de filtro de linha a cada schema.
C. Crie views dinâmicas que dupliquem cada tabela base com lógica de mascaramento.
D. Defina uma política ABAC que aplique uma máscara de coluna a qualquer coluna que tenha a tag governada 'pii'.

Questão 8

Objetivo: *Demonstrar compreensão do modelo de herança de permissões do Unity Catalog como mecanismo para gerenciar o controle de acesso em catálogos, schemas e objetos.*

Um engenheiro de dados concede SELECT em um catálogo ao grupo analysts. Novos schemas e tabelas são criados posteriormente nesse catálogo.

Supondo que não haja concessões conflitantes, qual acesso os analistas têm às novas tabelas?

A. Eles podem executar SELECT nas novas tabelas, porque um privilégio concedido no nível do catálogo é herdado pelos schemas e objetos dentro dele, incluindo os criados posteriormente.
B. Eles não têm acesso até que SELECT seja concedido novamente em cada nova tabela.
C. Eles podem executar SELECT apenas nas tabelas que existiam no momento da concessão.
D. Eles ganham automaticamente a propriedade das novas tabelas.

Questão 9

Objetivo: *Implantar recursos do Databricks usando Declarative Automation Bundles (anteriormente Databricks Asset Bundles).*

Um engenheiro de dados quer que as mesmas definições de job e pipeline sejam implantadas de forma idêntica nos ambientes de dev, staging e prod a partir do controle de código-fonte, com substituições específicas por ambiente e CI/CD.

Qual abordagem atende ao requisito?

A. Recrie manualmente os jobs na UI de cada workspace, depois exporte e importe o JSON.
B. Defina os recursos em um bundle databricks.yml com substituições por target e implante com `databricks bundle deploy -t <target>`.
C. Copie notebooks entre workspaces usando apenas o comando workspace import da CLI.
D. Use um único workspace compartilhado para todos os ambientes a fim de evitar desvios de configuração.

Questão 10

Objetivo: *Projetar modelos dimensionais para cargas de trabalho analíticas, utilizando materialized views para agregação pré-computada e metric views do Unity Catalog para definições de métricas governadas e reutilizáveis, a fim de garantir queries e agregações eficientes.*

Uma equipe de engenharia de dados calcula a "receita líquida" com SQL ligeiramente diferente em cada um de seus dashboards, produzindo números inconsistentes entre os relatórios. Os mesmos

dashboards também são lentos, porque cada um recalcula grandes joins e agregações no momento da query. A equipe precisa de uma definição única e oficial da métrica e de resultados pré-agregados que os dashboards possam ler diretamente.

Qual implementação o engenheiro deve escolher?

A. Adicione um comentário em cada dashboard descrevendo como calcular a receita líquida.
B. Conceda a todo analista acesso de edição a um arquivo de snippet SQL compartilhado.
C. Defina a receita líquida uma vez em uma metric view do Unity Catalog para uma definição governada e reutilizável, e use materialized views para os agregados pré-computados que os dashboards leem.
D. Armazene a métrica como uma UDF Python copiada em cada notebook.

Gabarito e explicações

#	Resposta	Explicação
1	C	A materialized view mantém incrementalmente o agregado completo e pode ser programada para atualização; por isso, reflete corretamente as atualizações de dias anteriores que chegam com atraso. Já a streaming table processa incrementalmente as linhas adicionadas e não recalcula o agregado completo sobre um histórico que sofre alterações.
2	A	O watermark permite que o mecanismo descarte o estado dos eventos que ultrapassam o atraso permitido (o que limita o crescimento do estado), enquanto o checkpoint da query persiste os offsets e o estado, de modo que, na reinicialização, a query retoma o processamento exactly-once, sem contar resultados em duplicidade.
3	B	O Lakeflow Connect oferece conectores gerenciados de banco de dados que ingerem CDC (inclusive exclusões) do SQL Server, do MySQL e do PostgreSQL com o mínimo de código; as outras alternativas fazem recargas completas ou não se aplicam a um banco de dados relacional operacional.
4	A	O tipo VARIANT armazena dados semiestruturados de forma eficiente e permite consultas com schema flexível por meio de parse_json e variant_get, evitando um schema rígido declarado previamente.
5	C	Tabelas do sistema como a system.billing.usage (em join com as tabelas do sistema de compute e preços) são registros governados e consultáveis, ideais para atribuição de custos e análise de cargas de trabalho; as outras fontes não podem ser consultadas em escala ou não são tabelas governadas.
6	B	Os deletion vectors marcam as linhas atualizadas e excluídas sem reescrever imediatamente arquivos de dados inteiros, reduzindo a amplificação de escrita causada pelas operações de MERGE pequenas e frequentes e mantendo as leituras corretas; depois, a compactação os reconcilia.
7	D	Uma política ABAC baseada em uma tag governada aplica uma máscara de coluna onde quer que a tag 'pii' esteja presente, incluindo as tabelas criadas posteriormente, de modo que a regra escala de forma centralizada, ao contrário das máscaras aplicadas tabela por tabela ou das views duplicadas.
8	A	Os privilégios do Unity Catalog são herdados de cima para baixo na hierarquia: uma concessão em um catálogo se aplica aos schemas dele e aos objetos desses schemas, incluindo os objetos criados depois da concessão.
9	B	Um Declarative Automation Bundle (databricks.yml) declara os recursos com targets por ambiente e substituições, e os implanta pela CLI, permitindo uma implantação reproduzível e compatível com CI/CD nos diferentes ambientes.
10	C	Uma metric view fornece uma definição única, governada e reutilizável da receita líquida; as materialized views pré-computam os agregados que os dashboards leem, resolvendo tanto o problema de consistência quanto o de desempenho.