dbt Labs Analytics Engineer Certification Practice Test (Sample)

Study Guide

Everything you need from our exam experts!

Copyright © 2025 by Examzify - A Kaluba Technologies Inc. product.

ALL RIGHTS RESERVED.

No part of this book may be reproduced or transferred in any form or by any means, graphic, electronic, or mechanical, including photocopying, recording, web distribution, taping, or by any information storage retrieval system, without the written permission of the author.

Notice: Examzify makes every reasonable effort to obtain from reliable sources accurate, complete, and timely information about this product.

SAMPLE

Questions

SAMPLE

1. In what way does dbt support validation?

A. Through built-in data quality checks

B. By allowing for custom SQL queries

C. With pre-hook and post-hook commands

D. By providing user interface options

2. What is indicated by the maturity levels in data exposures?

A. High, medium, and low regarding capability

B. Different types of data validation

C. Categories of data models

D. Versions of data analytics tools

3. What kind of error occurs during SQL execution in dbt?

A. Dependency Error

B. Runtime Error

C. Compilation Error

D. Database Error

4. Which command treats incremental models as table models?

A. dbt clean

B. dbt run

C. --full-refresh

D. dbt debug

5. What can be calculated using the information contained in the run_result artifact?

A. Total number of tables created

B. Average model runtime and test failure rates

C. Catalog of available resources

D. Peak memory usage during execution

6. What is the primary function of singular tests in dbt?

A. To assess the freshness of sources

B. To define accepted values

C. To write SQL that returns failing records

D. To validate data types across models

SAMPLE

7. What is one tradeoff of increasing the number of threads in dbt?

- **A. It decreases the execution speed of all models**
- **B. It reduces the amount of disk space used**
- **C. It increases the load on your warehouse**
- **D. It allows unlimited concurrent queries**

8. What is the main purpose of the dbt docs generate command?

- **A. To execute SQL models**
- **B. To generate the project documentation website**
- **C. To delete specified folders**
- **D. To show debug information**

9. What type of information does the target profile name in a dbt project represent?

- **A. The connection details for the database**
- **B. The name of the active profile**
- **C. The name of the data schema**
- **D. The configuration of the warehouse**

10. What key aspect of dbt enhances code reusability?

- **A. Macro functionalities**
- **B. Data visualization tools**
- **C. Automated testing features**
- **D. Real-time data streaming**

SAMPLE

Answers

SAMPLE

1. C
2. A
3. D
4. C
5. B
6. C
7. C
8. B
9. B
10. A

SAMPLE

Explanations

SAMPLE

1. In what way does dbt support validation?

A. Through built-in data quality checks

B. By allowing for custom SQL queries

C. With pre-hook and post-hook commands

D. By providing user interface options

The correct answer highlights how dbt facilitates validation through pre-hook and post-hook commands, which can be particularly effective in ensuring data integrity across models. Pre-hooks run specified SQL commands before a model is materialized, allowing you to perform validations or checks on the data before it is processed. Post-hooks, conversely, run commands after a model is materialized, which can also include validation tasks like inserting audit data or logging results. This functionality offers users the flexibility to implement custom validation processes and ensure that their data transformations meet certain criteria before they are finalized. Other options, while relevant in their own contexts, do not specifically pertain to the core functionality of dbt in relation to validation. Built-in data quality checks may imply certain features for managing data quality but are not as direct or customizable as the approach provided by hooks. Custom SQL queries can certainly allow validation checks, but they do not automate or streamline the process in the way that hooks do. User interface options in dbt also do not inherently contribute to direct data validation but instead focus on improving the user experience in model creation and management.

2. What is indicated by the maturity levels in data exposures?

A. High, medium, and low regarding capability

B. Different types of data validation

C. Categories of data models

D. Versions of data analytics tools

Maturity levels in data exposures refer to the progression or capability associated with how data is shared, accessed, and utilized within an organization. The classifications of high, medium, and low indicate the varying degrees of sophistication and readiness of data management and analysis processes. At a high maturity level, data exposures typically involve robust governance practices, comprehensive documentation, and advanced data sharing capabilities, which ensure that information is both reliable and accessible for informed decision-making. Medium maturity might reflect a developing framework with some established practices, while low maturity indicates more basic or ad-hoc methods lacking in structure and governance. The other options reference different aspects of data management or analytics that don't directly relate to the maturity levels of data exposures. For instance, different types of validation or categories of data models pertain to the technical quality and classification of the data rather than the maturity framework itself. Versions of analytics tools focus on the technological aspect rather than the organizational capability and readiness conveyed by maturity levels. Hence, the classification of maturity levels in data exposures is best understood through the lens of high, medium, and low capabilities.

SAMPLE

3. What kind of error occurs during SQL execution in dbt?

A. Dependency Error

B. Runtime Error

C. Compilation Error

D. Database Error

In the context of dbt, a Database Error occurs during SQL execution when there is an issue with the SQL query that is executed against the database. This can happen for various reasons, such as syntax errors in the SQL, referencing a non-existent table, or attempting to use a function that is not supported by the database. Essentially, these errors emerge after dbt successfully compiles the SQL code and attempts to run it on the database. Understanding that this type of error happens at the execution stage helps differentiate it from other types of errors. For example, a compilation error occurs when dbt attempts to compile the SQL code and finds issues such as missing Jinja tags or syntactical inconsistencies before it ever reaches the database. Runtime errors, on the other hand, refer to issues encountered while the code is being executed but could still be related to the logic in the code rather than the database itself. A dependency error would involve issues related to the order of table creation or dependencies between models, not directly during the execution of SQL. Recognizing the specific nature of Database Errors allows analytics engineers to troubleshoot and resolve problems effectively within their dbt projects.

4. Which command treats incremental models as table models?

A. dbt clean

B. dbt run

C. --full-refresh

D. dbt debug

The command that treats incremental models as table models is the option that includes the `--full-refresh` flag. When this flag is used during the `dbt run` command, it forces dbt to drop and recreate the tables instead of performing an incremental update. In typical usage, incremental models allow for data to be added to existing records based on specified criteria (like timestamps or unique identifiers) without reprocessing the entire dataset. However, invoking a full refresh overrides this incremental behavior, ensuring that the model behaves like a standard table model—essentially reloading the entire dataset from scratch. This feature is particularly useful when there have been significant changes or updates in the source data that an incremental model might not adequately capture. By treating the incremental model as a table, the user ensures the output is fresh and reflective of the most current data state. Other commands, such as `dbt clean`, are designed for clearing out target directories or compiled artifacts but do not affect how models are processed. The `dbt debug` command is used to check the configuration and connection settings without executing model runs. The basic `dbt run` command, without `--full-refresh`, operates standardly, respecting the incremental nature of the models.

SAMPLE

5. What can be calculated using the information contained in the run_result artifact?

A. Total number of tables created

B. Average model runtime and test failure rates

C. Catalog of available resources

D. Peak memory usage during execution

The run_result artifact provides insights specifically related to the performance and success of dbt models during execution. Among its various outputs, it includes metrics such as average model runtime and test failure rates, which are crucial for performance monitoring and debugging within a dbt project. Analyzing these metrics enables analytics engineers to identify which models are performing optimally and which may require optimization or troubleshooting due to failures. In contrast, while the other options may seem plausible, they do not align as closely with the specific metrics provided by the run_result artifact. For instance, total number of tables created, available resources, and peak memory usage may be relevant to a broader context but are not specifically captured or derived from the run_result artifact during dbt executions. Thus, focusing on average model runtime and test failure rates reflects the direct purpose of the run_result in analyzing dbt model performance.

6. What is the primary function of singular tests in dbt?

A. To assess the freshness of sources

B. To define accepted values

C. To write SQL that returns failing records

D. To validate data types across models

The primary function of singular tests in dbt is to write SQL that returns failing records. Singular tests are designed to validate a specific condition or set of conditions in your data. When a singular test is executed, it runs the SQL logic you've defined, and if that logic identifies any records that do not meet the specified criteria, those records are returned as the failing cases. This type of testing allows you to enforce quality checks on your data. For example, you can use singular tests to ensure that certain fields are not null, or that the values in a column fall within a specified range. The key aspect is that the SQL you write must clearly reflect the conditions you want to validate, and failing records will be those that do not conform to those conditions. In contrast, the other options refer to different functionalities within dbt's testing framework. Assessing the freshness of sources relates to ensuring that data inputs are up to date. Defining accepted values pertains to constraints or enumerations that limit what can be entered into a field. Validating data types across models typically involves checks on the structure and type consistency of data, which is not the primary role of singular tests. Therefore, the focus of singular tests specifically on querying for failing records sets it apart

SAMPLE

7. What is one tradeoff of increasing the number of threads in dbt?

A. It decreases the execution speed of all models

B. It reduces the amount of disk space used

C. It increases the load on your warehouse

D. It allows unlimited concurrent queries

Increasing the number of threads in dbt allows for more queries to run concurrently. While this can optimize execution times for many models, it also means that more resources are being demanded from the data warehouse simultaneously. This heightened demand can lead to increased load, which may impact overall performance by taxing the system's resources. It's essential to find a balance when configuring the number of threads. While more threads can speed up processing times, if the load on the warehouse becomes too great, it could lead to resource contention, slower performance, or even failure to execute queries due to resource limits being reached. On the other hand, the other options highlight consequences that do not directly correlate with increasing threads. For example, increasing threads doesn't inherently reduce disk space usage or improve the execution speed of all models, nor does it allow for unlimited concurrent queries, as data warehouses have specific limits on concurrent processing capabilities.

8. What is the main purpose of the dbt docs generate command?

A. To execute SQL models

B. To generate the project documentation website

C. To delete specified folders

D. To show debug information

The main purpose of the dbt docs generate command is to create the project documentation website. This command compiles the metadata of your dbt project—such as models, sources, tests, and metrics—into a structured format that can be easily accessed and navigated in a web interface. This documentation is crucial for both current and future members of a data team, as it provides a clear overview of the project's structure and data lineage, promoting better understanding and collaboration. By generating comprehensive documentation, teams can ensure that their data transformations and models are well-understood, leading to more informed decisions and error reductions. The other options refer to functions that dbt does not primarily focus on. Executing SQL models pertains to running transformations, deleting folders is unrelated to documentation generation, and while dbt can provide debug information during runs, that is not the purpose of the docs generate command. Hence, generating documentation is the unique and critical functionality that distinguishes this command.

SAMPLE

9. What type of information does the target profile name in a dbt project represent?

A. The connection details for the database

B. The name of the active profile

C. The name of the data schema

D. The configuration of the warehouse

The target profile name in a dbt project represents the name of the active profile that is being used for database connections and configurations. In dbt, a profile is defined within the `profiles.yml` file and contains details such as the connection method, the database type, and credential information necessary to successfully connect to the data warehouse. The active profile is instrumental because it determines how dbt interacts with the data source, including which database it connects to and the environment parameters that govern these connections. By explicitly identifying the active profile through its name, dbt ensures that the correct settings are applied when running models, tests, or seeds, and subsequently, it allows seamless interaction with the data warehouse. This understanding is vital for managing multiple environments (like development, staging, or production) within a dbt project, as the active profile can change depending on the context in which the dbt commands are executed. Therefore, recognizing that the target profile name specifies the active profile helps users maintain proper configurations and connectivity to their data sources.

10. What key aspect of dbt enhances code reusability?

A. Macro functionalities

B. Data visualization tools

C. Automated testing features

D. Real-time data streaming

The key aspect of dbt that enhances code reusability is its macro functionalities. Macros in dbt are essentially reusable pieces of code that can be defined once and used multiple times across different models. This allows analytics engineers to write complex SQL code once, encapsulate it in a macro, and then invoke it wherever needed throughout their project. This not only saves time and effort but also maintains consistency and reduces the likelihood of errors since the same logic can be reused instead of rewriting it multiple times in different places. The other choices do not contribute to code reusability in the same way. Data visualization tools are mainly concerned with presenting data rather than reusing code logic. Automated testing features are important for ensuring the quality and accuracy of the data transformation processes, but they do not enhance reusability. Real-time data streaming focuses on how data is ingested and processed rather than how the code can be reused within dbt projects. Thus, the macro functionalities stand out as the most relevant feature for increasing code reusability in dbt.

SAMPLE