DBT Interview Questions

Here are some **frequently asked dbt interview questions** for a Snowflake developer/engineer role.
These questions cover foundational concepts, advanced topics, and Snowflake-specific integrations with dbt.

General dbt Questions

1. **What is dbt, and how does it work?**
    - Explain dbt as a **transformation** tool that enables data modeling in SQL and automates the creation of tables and views in a data warehouse.
2. **What are the key components of a dbt project?**
    - Models, sources, seeds, snapshots, tests, macros, and documentation.
3. **What is the difference between ref() and source() in dbt?**
    - ref(): Used to reference other dbt models.
    - source(): Used to reference raw tables or external data sources.
4. **What are the different materializations in dbt?**
    - view, table, incremental, and ephemeral.
5. **What is the purpose of the dbt_project.yml file?**
    - It is the main configuration file for a dbt project, defining project settings, model configurations, and folder structures.
6. **How does dbt handle dependencies between models?**
    - dbt uses the ref() function to define dependencies, which automatically builds a Directed Acyclic Graph (DAG).
7. **What is the difference between incremental and full-refresh materializations?**
    - incremental: Only processes new or updated data.
    - full-refresh: Rebuilds the entire table from scratch.
8. **How do you test data in dbt?**
    - Use built-in tests like unique, not_null, and relationships, or define custom tests in .yml files.

9. **What is the purpose of snapshots in dbt?**
    - Snapshots are used to track changes in source data over time, enabling Slowly Changing Dimensions (SCDs).
10. **How do you document models in dbt?**
    - Use .yml files to add descriptions for models, columns, and tests. Generate documentation using
      dbt docs generate.

Snowflake-Specific dbt Questions

1. **How does dbt integrate with Snowflake?**
    - dbt connects to Snowflake using the snowflake adapter and executes SQL transformations directly in Snowflake.
2. **What is the role of the warehouse in the Snowflake profile configuration for dbt?**
    - The warehouse specifies the compute resources (virtual warehouse) used to execute dbt queries.
3. **How do you configure a Snowflake connection in dbt?**
    - Provide details like account, user, password, role, warehouse, database, and schema in the profiles.yml file.
4. **How do you optimize dbt models for Snowflake?**
    - Use clustering keys, partitioning, and Snowflake-specific SQL features like COPY INTO for efficient data loading.
5. **What are Snowflake-specific features in dbt?**
    - Support for Snowflake-specific SQL functions, GRANT statements, and Snowflake's TASKS for scheduling.
6. **How do you handle Snowflake's transient tables in dbt?**
    - Use the transient option in the dbt_project.yml file or model configurations.
7. **How do you manage Snowflake costs with dbt?**
    - Optimize warehouse usage, avoid unnecessary full-refresh runs, and use smaller warehouses for development.

8. **What is the difference between raw and transformed schemas in a Snowflake-dbt project?**
    - raw: Stores untransformed source data.
    - transformed: Stores dbt models after transformations.
9. **How do you implement incremental models in dbt for Snowflake?**
    - Use the is_incremental() macro to define logic for appending or updating data.
10. **How do you handle Snowflake permissions in dbt?**
    - Use GRANT statements in dbt models or macros to manage access control.

Advanced dbt Questions

1. **How do you debug a failing dbt model?**
    - Check the logs, validate SQL syntax, and ensure dependencies are correctly defined using ref().
2. **What is the role of macros in dbt?**
    - Macros are reusable SQL snippets written in Jinja, used to simplify complex logic.
3. **How do you implement a Slowly Changing Dimension (SCD) in dbt?**
    - Use snapshots or incremental models with logic to track changes.
4. **How do you manage environments (dev, staging, prod) in dbt?**
    - Use different schemas for each environment and configure them in dbt_project.yml.
5. **How do you handle schema changes in dbt models?**
    - Use dbt run --full-refresh to rebuild models or manage schema changes incrementally.
6. **How do you monitor dbt runs?**
    - Use dbt Cloud, logging, or integrate with tools like Airflow or Snowflake's TASKS.
7. **How do you handle large datasets in dbt?**
    - Use incremental models, clustering keys, and Snowflake's RESULT_SCAN for efficient processing.

8. **What is the difference between ephemeral models and CTEs in dbt?**
    - Ephemeral models are in-memory transformations that are compiled into the final SQL query.
9. **How do you implement data quality checks in dbt?**
    - Use tests in .yml files and custom SQL queries to validate data integrity.
10. **How do you manage dbt deployments in a CI/CD pipeline?**
    - Use tools like GitHub Actions, dbt Cloud, or custom scripts to automate dbt runs and tests.

Behavioral Questions

1. **Describe a challenging dbt project you worked on and how you solved it.**
2. **How do you ensure data quality in your dbt models?**
3. **How do you collaborate with data engineers and analysts in a dbt project?**
4. **How do you handle performance issues in dbt models?**
5. **How do you prioritize tasks in a dbt project with tight deadlines?**

Let me know if you'd like detailed answers or examples for any of these questions!

Behavioral Questions and Sample Answers:

1. Describe a challenging dbt project you worked on and how you solved it.

Example Answer: In one project, we had to migrate a legacy ETL pipeline to dbt while ensuring minimal downtime. The challenge was understanding the existing pipeline's logic, which was poorly documented, and replicating it in dbt. I started by reverse-engineering the SQL queries and workflows, documenting the logic, and breaking it into smaller dbt models. I used dbt's ref() function to create a modular DAG and implemented incremental models to handle large datasets. To validate the migration, I ran parallel tests comparing the outputs of the legacy pipeline and dbt models. This ensured data consistency and allowed us to switch to dbt with confidence.

2. How do you ensure data quality in your dbt models?

Example Answer: I ensure data quality by leveraging dbt's built-in testing framework. For every model, I define tests in .yml files, such as unique, not_null, and relationships. I also create custom tests for business-specific rules. Additionally, I document all models and columns to ensure clarity for the team. Before deploying changes, I run dbt test to validate the data. I also monitor data quality post-deployment using tools like dbt artifacts or integrating with external monitoring tools like Monte Carlo or Great Expectations.

3. How do you collaborate with data engineers and analysts in a dbt project?

Example Answer: Collaboration is key in dbt projects. I work closely with data engineers to understand the raw data sources and ensure the data pipelines are optimized for transformation. With analysts, I gather requirements for the transformed data, such as specific metrics or dimensions they need for reporting. I use tools like Jira or Trello to track tasks and ensure transparency. Regular stand-ups and code reviews help align the team, and I encourage analysts to use dbt's documentation and lineage graph to understand the data flow.

4. How do you handle performance issues in dbt models?

Example Answer: When facing performance issues, I start by identifying bottlenecks using Snowflake's query history and execution plans. Common optimizations include:

- Refactoring SQL logic to reduce complexity.
- Using incremental models to process only new or updated data.
- Adding clustering keys or partitioning in Snowflake to improve query performance.
- Avoiding unnecessary joins or subqueries by pre-aggregating data in intermediate models. I also ensure that the warehouse size is appropriate for the workload and monitor costs to balance performance and efficiency.

5. How do you prioritize tasks in a dbt project with tight deadlines?

Example Answer: I prioritize tasks by focusing on business-critical requirements first. I work with stakeholders to identify high-impact models and transformations that directly support decision-making. I break down the project into smaller milestones and use Agile methodologies to deliver incrementally. For tight deadlines, I focus on delivering a Minimum Viable Product (MVP) and defer non-essential features to later iterations. Clear communication with the team and stakeholders ensures alignment on priorities and expectations.

Related to Hooks in dbt

1. **What are hooks in dbt, and how are they used?**
    - Hooks are SQL statements or macros that run at specific points during a dbt run. They can be used for tasks like setting up environments, managing permissions, or logging.
2. **What are the types of hooks in dbt?**
    - **Pre-hooks**: Run before a model is executed.
    - **Post-hooks**: Run after a model is executed.
3. **How do you define a hook in dbt?**
    - Hooks are defined in the dbt_project.yml file or in the model configuration using the pre_hook or post_hook options.
4. **Can you give an example of a pre-hook and post-hook in dbt?**
    - Example:

```
models:
    my_project:
        my_model:
            pre-hook: "BEGIN TRANSACTION"
            post-hook: "COMMIT"
```

5. **What are some common use cases for hooks in dbt?**
    - Setting up temporary tables.
    - Granting permissions to users after a model is created.
    - Logging metadata for auditing purposes.

Related to dbt Objects

1. **What are the main objects in a dbt project?**
    - **Models**: SQL files that define transformations.
    - **Sources**: Represent raw data tables in the warehouse.
    - **Seeds**: Static CSV files loaded into the warehouse.
    - **Snapshots**: Track changes in data over time.
    - **Macros**: Reusable SQL snippets written in Jinja.
    - **Tests**: Validate data quality.
    - **Documentation**: Descriptions for models, columns, and tests.
2. **What is the difference between a model and a source in dbt?**
    - **Model**: A transformed table or view created by dbt.
    - **Source**: A raw table or external data source that dbt queries but does not modify.
3. **What is the purpose of a snapshot in dbt?**
    - Snapshots are used to track historical changes in data, enabling Slowly Changing Dimensions (SCDs).
4. **How are macros different from models in dbt?**
    - **Macros**: Reusable SQL logic written in Jinja, used to simplify repetitive tasks.
    - **Models**: SQL files that define transformations and create tables or views in the warehouse.
    -

Materializations in dbt

1. **What are materializations in dbt?**
    - Materializations define how dbt creates and manages database objects (e.g., tables, views) for a model.

2. **What are the types of materializations in dbt?**
    - **View**: Creates a database view.
    - **Table**: Creates a physical table.
    - **Incremental**: Updates or appends data to an existing table.
    - **Ephemeral**: Creates a temporary CTE (Common Table Expression) during the dbt run.
3. **What is the difference between view and table materializations?**
    - **View**: A logical layer that queries the underlying data; does not store data physically.
    - **Table**: A physical table that stores data in the warehouse.
4. **How does the incremental materialization work in dbt?**
    - The incremental materialization processes only new or updated data, appending it to an existing table. It uses the is_incremental() macro to define logic for incremental updates.
5. **What is the difference between ephemeral materialization and a CTE?**
    - **Ephemeral**: A dbt-specific materialization that compiles into a CTE within the final SQL query.
    - **CTE**: A SQL construct used in queries, but ephemeral models allow you to modularize and reuse logic across dbt models.
6. **How do you configure materializations in dbt?**
    - Materializations are configured in the dbt_project.yml file or in the model file using the config() function:

```
{{ config(materialized='incremental') }}
```

7. **What are the advantages of using incremental materialization?**
    - Reduces processing time for large datasets.
    - Optimizes resource usage by only processing new or updated data.

1. What is dbt, and how does it work?

dbt is a data transformation tool that allows analysts and engineers to transform raw data in a data warehouse into clean, tested, and documented datasets. It uses SQL and Jinja templates to define models, which are then compiled and executed in the warehouse.

2. How do you run a specific dbt model?

Use the dbt run command with the --select flag to specify the model.

- **Example**: dbt run --select my_model

3. What is the difference between dbt run and dbt build?

- **dbt run:** Executes the SQL models and materializes them in the data warehouse.
- **dbt build:** Executes models, tests, snapshots, and seeds in a single command.

4. How do you test models in dbt?

- dbt allows you to define tests in YAML files or write custom SQL tests. You can run tests using the dbt test command.

5. How do you handle dependencies between models in dbt?

- dbt uses the ref() function to define dependencies between models. This ensures that models are built in the correct order.

6. How do you debug a failing dbt model?

- Check the logs in the target directory.
- Use the --debug flag with dbt commands.
- Run the model in isolation using dbt run --select.

7. What are seeds in dbt, and how do you use them?

- Seeds are CSV files stored in the data directory. They can be loaded into the data warehouse using the dbt seed command.

8. How do you implement incremental models in dbt?

- Use the is_incremental() macro to define logic for incremental runs. Incremental models only process new or updated data.

9. How do you document models in dbt?

- Use YAML files to define descriptions for models, columns, and tests. Run dbt docs generate to create documentation.

10. How do you manage environments in dbt (e.g., dev, prod)?

- **Answer**: Use different profiles in the profiles.yml file to manage connections to different environments.

Example Scripts

1. Running a Specific Model

```
dbt run --select my_model
```

2. Running All Models

dbt run

3. Building Models, Tests, and Snapshots

dbt build

4. Running Tests

dbt test

5. Incremental Model Example

```
{{ config(
  materialized='incremental'
) }}
SELECT *
FROM source_table
{% if is_incremental() %}
WHERE updated_at > (SELECT MAX(updated_at) FROM {{ this }})
{% endif %}
```

6. Defining a Test in YAML

```
models:
 - name: my_model
  columns:
   - name: id
    tests:
     - unique
     - not_null
```

7. Using ref() for Dependencies

```
SELECT *
FROM {{ ref('upstream_model') }}
```

8. Debugging a Model

dbt run --select my_model --debug