Airflow Peer Learning

Anuj's Approach

1. Create Dag to perform tasks: create table, populate the table and then select the values from the table

- create_employee_table: This task creates an employee table in a PostgreSQL database using the PostgresOperator. It specifies the SQL script file sql/employee_schema.sql that contains the table creation statements.
- populate_employee_table: This task populates the employee table with data using the PostgresOperator. It executes the SQL script file sql/insert_employee.sql that contains the insert statements.
- get_all_employees: This task retrieves all the records from the employee table using the PostgresOperator. It executes the SQL query "SELECT * FROM employee;".
- The tasks are connected using the >> operator to specify the task dependencies. The create_employee_table task must complete before the populate_employee_table task can start, and the populate_employee_table task must complete before the get_all_employees task can start.
- To execute the DAG using the Airflow command-line interface, the script provides a conditional statement if name == "main": dag_psql.cli(), allowing the DAG to be run from the command line.

2.Create DAG to perform tasks: A dummy task which always succeed and then send email alert after successful completion

- dummy_task: This task is a DummyOperator that acts as a placeholder and always succeeds. It does not perform any actual operation.
- email_notification: This task is an EmailOperator that sends an email notification when the dummy_task succeeds. It is configured with the recipient email address (to parameter), subject line (subject parameter), and the content of the email (html_content parameter).
- The DAG is configured with default arguments, including the owner, whether to depend on past runs, the start date, and the number of retries and retry delay in

case of failures. It is scheduled to run daily with a schedule interval of timedelta(days=1).
- The dummy_task is set as the dependency for the email_notification task using the >> operator. This means that the email_notification task will only run if the dummy_task succeeds.

3. Create DAG to perform tasks: A dummy task which can succeed/fail and then send success/failure message to slack workplace

- print_date: This task is a BashOperator that executes the date command to print the current date.
- slack_integration: This task is a SlackWebhookOperator that sends a message to a Slack channel using a webhook integration. It retrieves the Slack webhook token from the slack connection in Airflow's connections database. The message content is set to "Your task has finished".
- The DAG is configured with default arguments, including the owner, whether to depend on past runs, the start date (one day ago), and the number of retries and retry delays in case of failures. It is scheduled to run daily with a schedule interval of timedelta(days=1)
- The print_date task is set as the dependency for the slack_integration task using the >> operator. This means that the slack_integration task will only run if the print_date task succeeds.

Pankaj's Approach

1. Create Dag to perform tasks: create table, populate the table and then select the values from the table

- The task dependencies (create_table_task >> insert_values_task >> select_values_task) are defined correctly, indicating that the create_table_task must complete before the insert_values_task, and the insert_values_task must complete before the select_values_task.
- The default_args dictionary provides configuration options for the DAG. It's a good practice to explicitly define the catchup parameter to avoid unexpected backfilling of the DAG.

2.Create DAG to perform tasks: A dummy task which always succeed and then send email alert after successful completion

- The postgres_conn_id used in the PostgresOperator tasks should correspond to the connection ID defined in the Airflow connections
- The DAG is created with the ID 'sigmoid_email_alert'. Ensure that this ID is unique within your Airflow environment. The start_date is set to datetime(2023, 4, 25), indicating that the DAG will start on that date.

3. Create DAG to perform tasks: A dummy task which can succeed/fail and then send success/failure message to slack workplace

- The code retrieves the Slack connection details from Airflow's Connection feature using the SLACK_CONN_ID connection identifier. The webhook token and channel are obtained using BaseHook.get_connection method. Verify that you have created a Slack connection in Airflow with the specified connection identifier (SLACK_CONN_ID) and that the connection details are correct
- task2 and task3 are SlackWebhookOperator tasks that send Slack messages to the specified channel. task2 triggers on the success of task1 using TriggerRule.ALL_SUCCESS, while task3 triggers on the failure of task1 using TriggerRule.ALL_FAILED.