

AirFlow-DBT-Snowflake Integration in Local Machine(Windows)

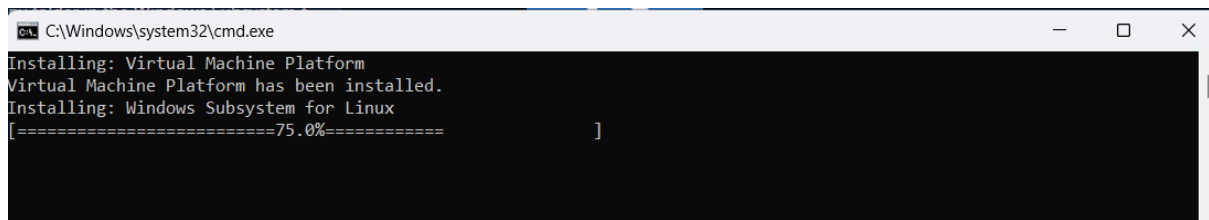
Airflow Setup:

Step1:

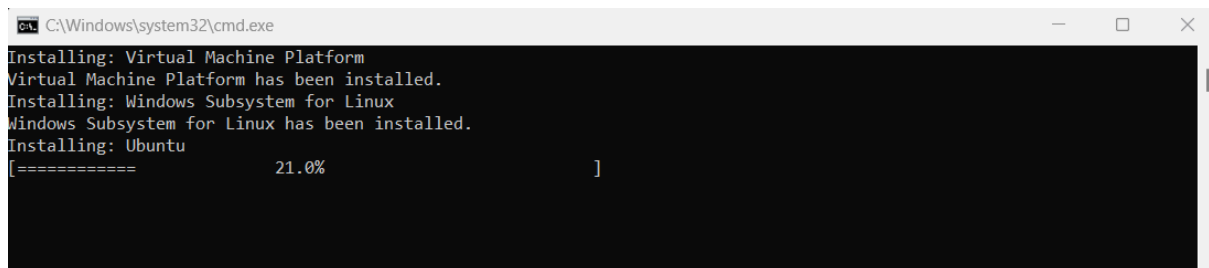
Installation in Windows using docker or WSL:

Installing WSL Windows

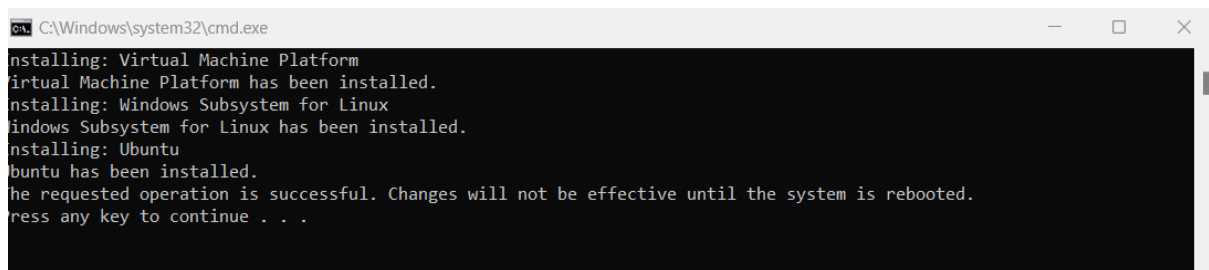
```
C:\Users\psara>wsl --install
```



```
C:\Windows\system32\cmd.exe
Installing: Virtual Machine Platform
Virtual Machine Platform has been installed.
Installing: Windows Subsystem for Linux
[=====75.0%=====]
```



```
C:\Windows\system32\cmd.exe
Installing: Virtual Machine Platform
Virtual Machine Platform has been installed.
Installing: Windows Subsystem for Linux
Windows Subsystem for Linux has been installed.
Installing: Ubuntu
[=====21.0%=====]
```



```
C:\Windows\system32\cmd.exe
Installing: Virtual Machine Platform
Virtual Machine Platform has been installed.
Installing: Windows Subsystem for Linux
Windows Subsystem for Linux has been installed.
Installing: Ubuntu
Ubuntu has been installed.
The requested operation is successful. Changes will not be effective until the system is rebooted.
Press any key to continue . . .
```

Step2:-

```
Ubuntu is already installed.
Launching Ubuntu...
Installing, this may take a few minutes...
Please create a default UNIX user account. The username does not need to
match your Windows username.
For more information visit: https://aka.ms/wslusers
Enter new UNIX username: shrovan
New password:
Retype new password:
passwd: password updated successfully
Installation successful!
```

Step3: Update linux

```
shrovan@LAPTOP-3U4JI4R2:/mnt/c/Users/psara$ sudo apt update
```

Step4: Install python and pip for creating virtual environment

```
shrovan@LAPTOP-3U4JI4R2:/mnt/c/Users/psara$ sudo apt install python3
python3-pip
```

Step5: Install venv package and Create a python virtual environment

```
shrovan@LAPTOP-3U4JI4R2:/mnt/c/Users/psara$ sudo apt-get install
python3-venv
```

```
shrovan@LAPTOP-3U4JI4R2:~$ mkdir airflow_project
shrovan@LAPTOP-3U4JI4R2:~$ cd airflow_project/
```

```
shrovan@LAPTOP-3U4JI4R2:/mnt/c/Users/psara/shrovandbtairflow$ python3 -m
venv airflow_venv
shrovan@LAPTOP-3U4JI4R2:/mnt/c/Users/psara/shrovandbtairflow$ source
airflow_venv/bin/activate
(airflow_venv)
shrovan@LAPTOP-3U4JI4R2:/mnt/c/Users/psara/shrovandbtairflow$
```

Step6: Install apache airflow in the virtual environemnt

```
(airflow_venv) shrovan@LAPTOP-3U4JI4R2:
AIRFLOW_VERSION=2.10.1
```

```
PYTHON_VERSION="$(python --version | cut -d " " -f 2 | cut -d "." -f 1-2)"
CONSTRAINT_URL="https://raw.githubusercontent.com/apache/airflow/constraints-${AIRFLOW_VERSION}/constraints-${PYTHON_VERSION}.txt"

pip install "apache-airflow==${AIRFLOW_VERSION}" --constraint
"${CONSTRAINT_URL}"
```

Step7:SET airflow home page and Initialise the airflow database

```
(airflow_venv)
shrovan@LAPTOP-3U4JI4R2:/mnt/c/Users/psara/shrovanbtairflow$ export
AIRFLOW_HOME=~/.airflow
```

```
(airflow_venv)
shrovan@LAPTOP-3U4JI4R2:/mnt/c/Users/psara/shrovanbtairflow$ airflow db
init
```

Step8: Start the webserver

```
(airflow_venv)
shrovan@LAPTOP-3U4JI4R2:/mnt/c/Users/psara/shrovanbtairflow$ airflow
webserver --port 8080
```

Step9: Start the scheduler in a new instance:

```
shrovan@LAPTOP-3U4JI4R2:/mnt/c/Users/psara$ cd shrovanbtairflow/
shrovan@LAPTOP-3U4JI4R2:/mnt/c/Users/psara/shrovanbtairflow$ source
airflow_venv/bin/activate
(airflow_venv)
shrovan@LAPTOP-3U4JI4R2:/mnt/c/Users/psara/shrovanbtairflow$ airflow
scheduler
```

Step10:

In a new instance list the user details currently in use for airflow

```
(airflow_venv)
shrovan@LAPTOP-3U4JI4R2:/mnt/c/Users/psara/shrovanbtairflow$ airflow
users list
No data found
```

Step11: Create a new user with Admin role, and set a username and password for the same.

```
(airflow_venv)
shrovan@LAPTOP-3U4JI4R2:/mnt/c/Users/psara/shrovanbtairflow$ airflow
users create --username admin --firstname shrovan --lastname p --role
Admin --email admin@anyname.com
/mnt/c/Users/psara/shrovanbtairflow/airflow_venv/lib/python3.12/site-pa
ckages/flask_limiter/extension.py:333 UserWarning: Using the in-memory
storage for tracking rate limits as no storage was explicitly specified.
This is not recommended for production use. See:
https://flask-limiter.readthedocs.io#configuring-a-storage-backend for
documentation about configuring the storage backend.
[2025-01-22T06:27:07.929+0000] {override.py:965} WARNING - No user yet
created, use flask fab command to do it.
[2025-01-22T06:27:08.553+0000] {workday.py:41} WARNING - Could not
import pandas. Holidays will not be considered.
Password:
Repeat for confirmation:
[2025-01-22T06:27:37.482+0000] {override.py:1597} INFO - Added user
admin
User "admin" created with role "Admin"
```

Step12:

Check the user information for airflow:

```
(airflow_venv)
shrovan@LAPTOP-3U4JI4R2:/mnt/c/Users/psara/shrovanbtairflow$ airflow
users list
```

id	username	email	first_name	last_name	roles
1	admin	admin@anyname.com	shrovan	p	Admin

Step13:

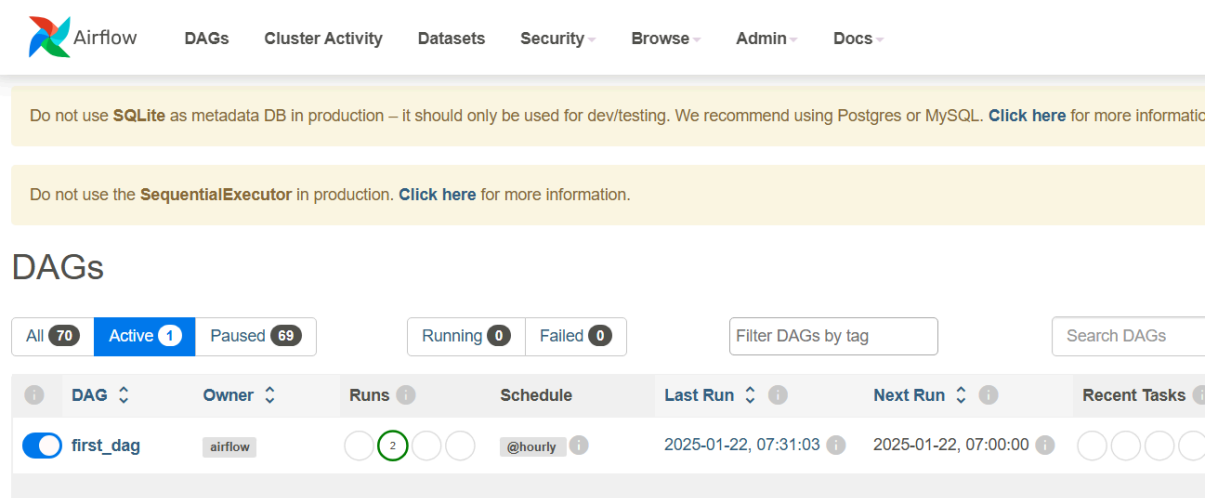
Open the airflow folder where the airflow is installed via vscode and under that folder create a dags folder.

Add a new python script under the DAG, initiate the virtual environment

```
(airflow_venv)
shrovan@LAPTOP-3U4JI4R2:/mnt/c/Users/psara/shrovanbtairflow$ source
/mnt/c/Users/psara/shrovanbtairflow/airflow_venv/bin/activate
```

Step14: Run the Python script

```
(airflow_venv) shrovan@LAPTOP-3U4JI4R2:~/airflow/dags$ python3
hello_world.py
/home/shrovan/airflow/dags/hello_world.py:8 RemovedInAirflow3Warning:
Param `schedule_interval` is deprecated and will be removed in a future
release. Please use `schedule` instead.
```



The screenshot shows the Apache Airflow web interface. At the top, there's a navigation bar with links for Airflow, DAGs, Cluster Activity, Datasets, Security, Browse, Admin, and Docs. Below the navigation bar, there are two yellow warning banners. The first banner says: "Do not use **SQLite** as metadata DB in production – it should only be used for dev/testing. We recommend using Postgres or MySQL. [Click here](#) for more information." The second banner says: "Do not use the **SequentialExecutor** in production. [Click here](#) for more information." Below the warnings, the "DAGs" section is visible. It includes filters for DAG status: All (70), Active (1), Paused (69), Running (0), and Failed (0). There's also a "Filter DAGs by tag" input and a "Search DAGs" button. A table lists the DAGs with columns: DAG, Owner, Runs, Schedule, Last Run, Next Run, and Recent Tasks. The first DAG listed is "first_dag" owned by "airflow", with a schedule of "@hourly". Its last run was on "2025-01-22, 07:31:03" and its next run is on "2025-01-22, 07:00:00". The "Recent Tasks" column shows four circles, with the second one highlighted in green, indicating a successful run.

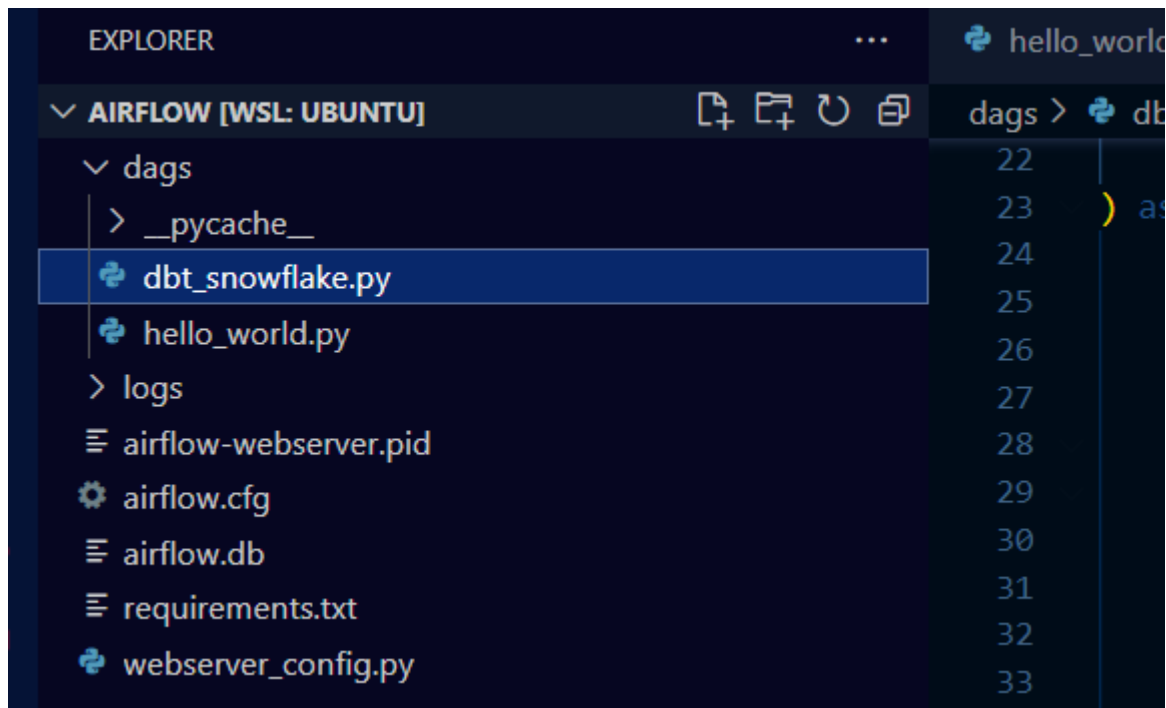
Step15:- Create a requirement.txt folder to install the required packages

```
(airflow_venv) shrovan@LAPTOP-3U4JI4R2:~/airflow$ cat requirements.txt
apache-airflow
dbt-core
dbt-snowflake
(airflow_venv) shrovan@LAPTOP-3U4JI4R2:~/airflow$
```

Step16:- Run the requirement.txt folder

```
(airflow_venv) shrovan@LAPTOP-3U4JI4R2:~/airflow$ pip install -r
requirements.txt
```

Step17:- Create a Python script to run a DBT Model from Airflow.



```
from airflow import DAG
from airflow.operators.bash import BashOperator
from airflow.operators.empty import EmptyOperator
from airflow.utils.task_group import TaskGroup
from datetime import datetime

PROJECT_DIR = '/mnt/c/Users/psara/shrovanbtsnowflake'
PROFILES_DIR = '/mnt/c/Users/psara/.dbt'

default_args = {
    'owner': 'airflow',
    'depends_on_past': False,
    'retries': 3,
}

with DAG(
    'dbt_snowflake_job2',
    default_args=default_args,
    description='Airflow job to run dbt-snowflake from airflow',
    schedule_interval='@daily',
    start_date=datetime(2024,1,22),
    catchup=False,
) as dag:
```

```

start = EmptyOperator(task_id='start_dag')

with TaskGroup(group_id='dbt_tasks') as dbt_tasks:
    dbt_debug = BashOperator(
        task_id='dbt-debug',
        bash_command=f'dbt debug --profiles-dir {PROFILES_DIR}
--project-dir {PROJECT_DIR}',
    )

    #dbt run task
    dbt_run = BashOperator(
        task_id='dbt-run',
        #bash_command=f'dbt run --profiles-dir {PROFILES_DIR}
--project-dir {PROJECT_DIR}'
        bash_command=f'dbt run --profiles-dir {PROFILES_DIR} --project-dir
{PROJECT_DIR} --select staging',
    )

    dbt_debug >> dbt_run

end = EmptyOperator(task_id='end_dag')

#Task Dependencies
start>>dbt_tasks>>end

```

Step 18: Execute the Script and check the task in Airflow.

DAG: airflow_dbt_snowflake3 Airflow job to run dbt-snowflake from airflow Schedule: @daily Next Run ID: 2025-01-22, 00:00:00 UTC

22-01-2025 13:35:15 All Run Types All Run States Clear Filters Aut

Press **shift** + **/** for Shortcuts

deferred failed queued removed restarting running scheduled shutdown skipped success up_for_reschedule up_for_retry

airflow_dbt_snowflake3 / 2025-01-21, 00:00:00 UTC / **dbt-run** Clear task Mark state as...

Details **Graph** **Gantt** **Code** **Event Log** **Logs** **XCom** **Task Duration**

All Levels All File Sources Wra

Duration: 00:01:46, 00:00:53, 00:00:00

start_dag
dbt_tasks
dbt-debug
dbt-run
end_dag

[2025-01-22, 13:36:29 UTC] [subprocess.py:93] INFO - 13:36:29 Found 8 models, 5 seeds, 5 data tests, 454 macros
[2025-01-22, 13:36:29 UTC] [subprocess.py:93] INFO - 13:36:29
[2025-01-22, 13:36:33 UTC] [subprocess.py:93] INFO - 13:36:33 Concurrency: 5 threads (target='dev')
[2025-01-22, 13:36:33 UTC] [subprocess.py:93] INFO - 13:36:33
[2025-01-22, 13:36:33 UTC] [subprocess.py:93] INFO - 13:36:33 1 of 8 START sql table model JAN.my_first_dbt_model [RUN]
[2025-01-22, 13:36:33 UTC] [subprocess.py:93] INFO - 13:36:33 2 of 8 START sql table model year2025.JAN.stg_avg_telemetry [RUN]
[2025-01-22, 13:36:33 UTC] [subprocess.py:93] INFO - 13:36:33 3 of 8 START sql table model year2025.JAN.stg_error_data [RUN]
[2025-01-22, 13:36:33 UTC] [subprocess.py:93] INFO - 13:36:33 4 of 8 START sql table model year2025.JAN.stg_failure_data [RUN]
[2025-01-22, 13:36:33 UTC] [subprocess.py:93] INFO - 13:36:33 5 of 8 START sql table model year2025.JAN.stg_maintenance_data [RUN]
[2025-01-22, 13:36:35 UTC] [subprocess.py:93] INFO - 13:36:35 4 of 8 OK created sql table model year2025.JAN.stg_failure_data [SUCCESS 1 in 2.36s]
[2025-01-22, 13:36:35 UTC] [subprocess.py:93] INFO - 13:36:35 2 of 8 OK created sql table model year2025.JAN.stg_avg_telemetry [SUCCESS 1 in 2.32s]
[2025-01-22, 13:36:35 UTC] [subprocess.py:93] INFO - 13:36:35 3 of 8 OK created sql table model year2025.JAN.stg_error_data [SUCCESS 1 in 2.32s]
[2025-01-22, 13:36:35 UTC] [subprocess.py:93] INFO - 13:36:35 5 of 8 OK created sql table model year2025.JAN.stg_maintenance_data [SUCCESS 1 in 2.31s]
[2025-01-22, 13:36:35 UTC] [subprocess.py:93] INFO - 13:36:35 6 of 8 START sql table model year2025.JAN.stg_telemetry_features [RUN]
[2025-01-22, 13:36:35 UTC] [subprocess.py:93] INFO - 13:36:35 1 of 8 OK created sql table model JAN.my_first_dbt_model [SUCCESS 1 in 2.39s]
[2025-01-22, 13:36:35 UTC] [subprocess.py:93] INFO - 13:36:35 2 of 8 START sql table model year2025.JAN.stg_avg_telemetry [RUN]

