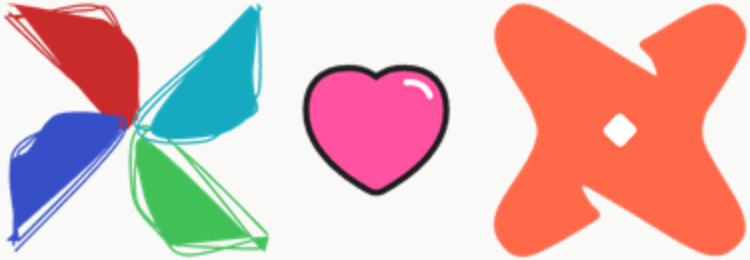
# dbt-Core & Airflow 101: Building Data Pipelines Demystified



Luan Moreno
Sr. Cloud Consultant at Pythian

# Quick Survey

Get a sense of people in the audience

1. Who here uses dbt-Core**?**
2. Who knows or uses Astronomer Cosmos**?**

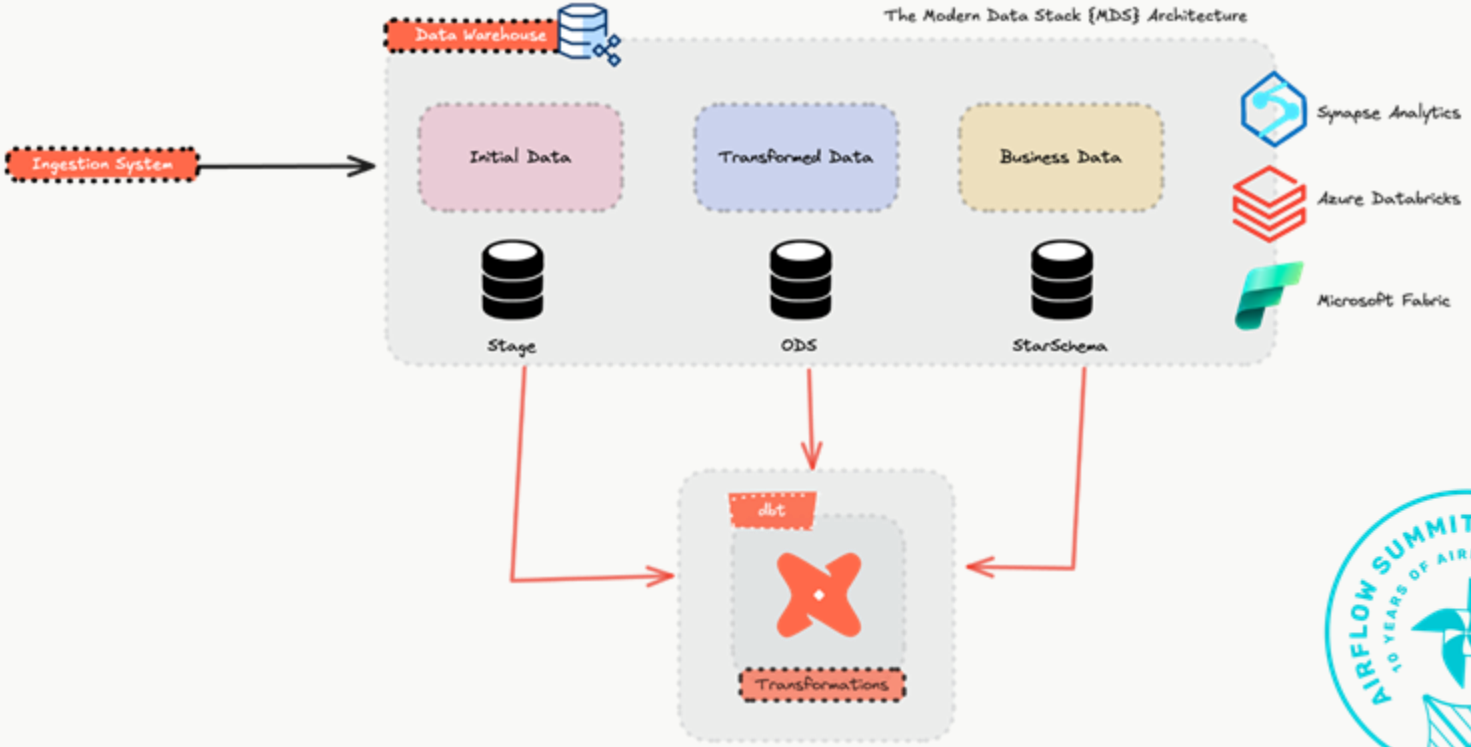
# dbt 101

The basics of data build tool

dbt {Data Build Tool}

Open-Source Tool that Enables Data Teams to Transform Data into Valuable & Well-Organized Insights

Build Trusted Data Products Faster



# Airflow & dbt Core

High level comparison

- **Python** based for authoring, scheduling & monitoring workflows.
- **Flexible** and can be used for different scenarios and use-cases with a wider range of tasks.
- **Complex** interface and requires deeper understanding of workflow management to write SQL transformations.

Data Engineers

- **SQL** based focused specifically on transforming and analyzing data.
- **Specialized** and provides a more focused set of features and tools for working with data in a data warehouse.
- **Simple** interface for working with data and SQL transformations.

Analytics Engineers

# How to run dbt-Core projects in Apache Airflow?

https://github.com/astronomer/astronomer-cosmos

# Astronomer Cosmos 101

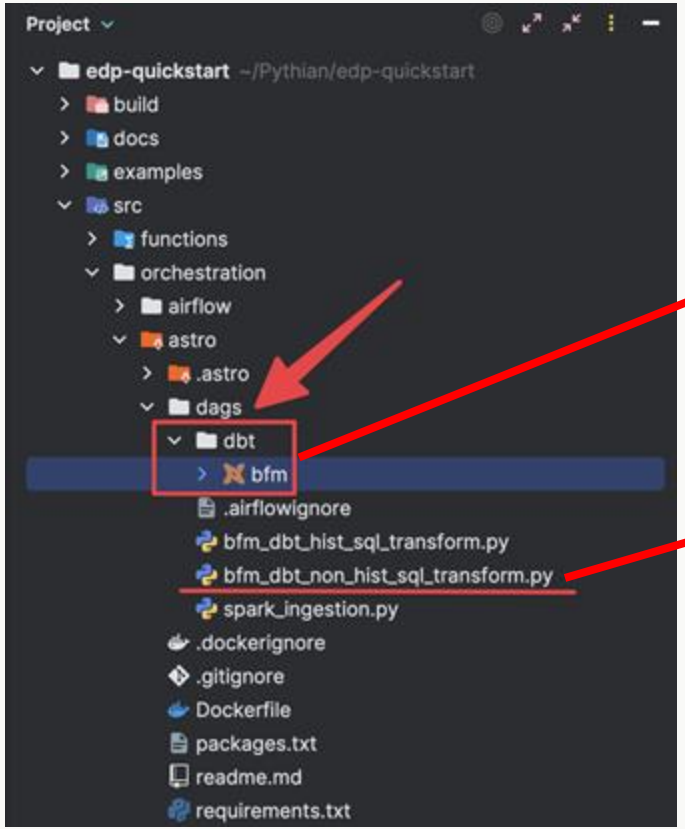
Run your dbt Core project as Airflow DAGs with fewer lines of code




```
basic_cosmos_dag = DbtDag(
    # dbt/cosmos-specific parameters
    project_config=ProjectConfig(
        DBT_ROOT_PATH / "jaffle_shop",
    ),
    profile_config=profile_config,
    operator_args={
        "install_deps": True,  # install any necessary dependencies before running any dbt command
        "full_refresh": True,  # used only in dbt commands that support this flag
    },
    # normal dag parameters
    schedule_interval="@daily",
    start_date=datetime(2023, 1, 1),
    catchup=False,
    dag_id="basic_cosmos_dag",
    default_args={"retries": 2},
)
```

# Airflow & dbt Project

The structure of your dbt project under Apache Airflow umbrella




## dbt Project

- **dags =** folder where DAGs are authored and stored.
- **dbt/project-name** = dbt project under Airflow structure

## Airflow DAG

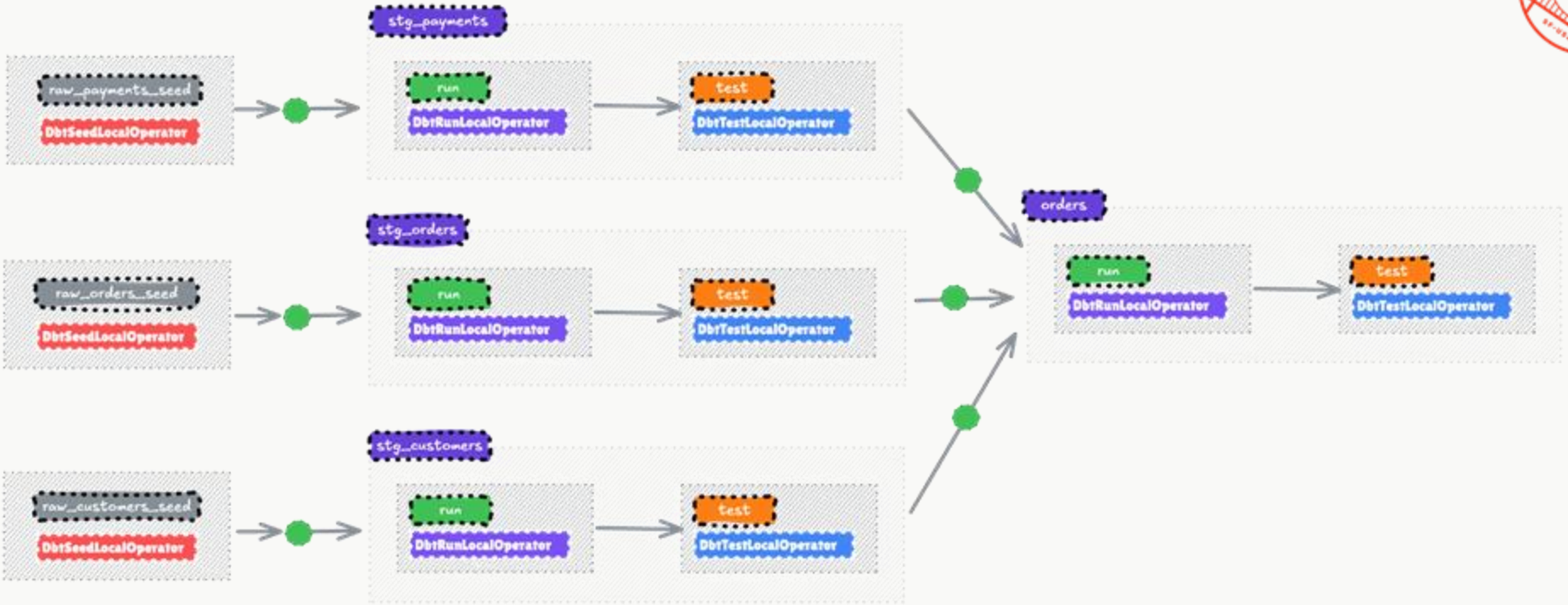
- build DAGs that interacts with the dbt Project.

## News from the press!

- **New on v1.6 {August}** you can turn your dbt project agnostic
- Leverage the **manifest.json stored** in a **GCS, Blob Storage & S3**

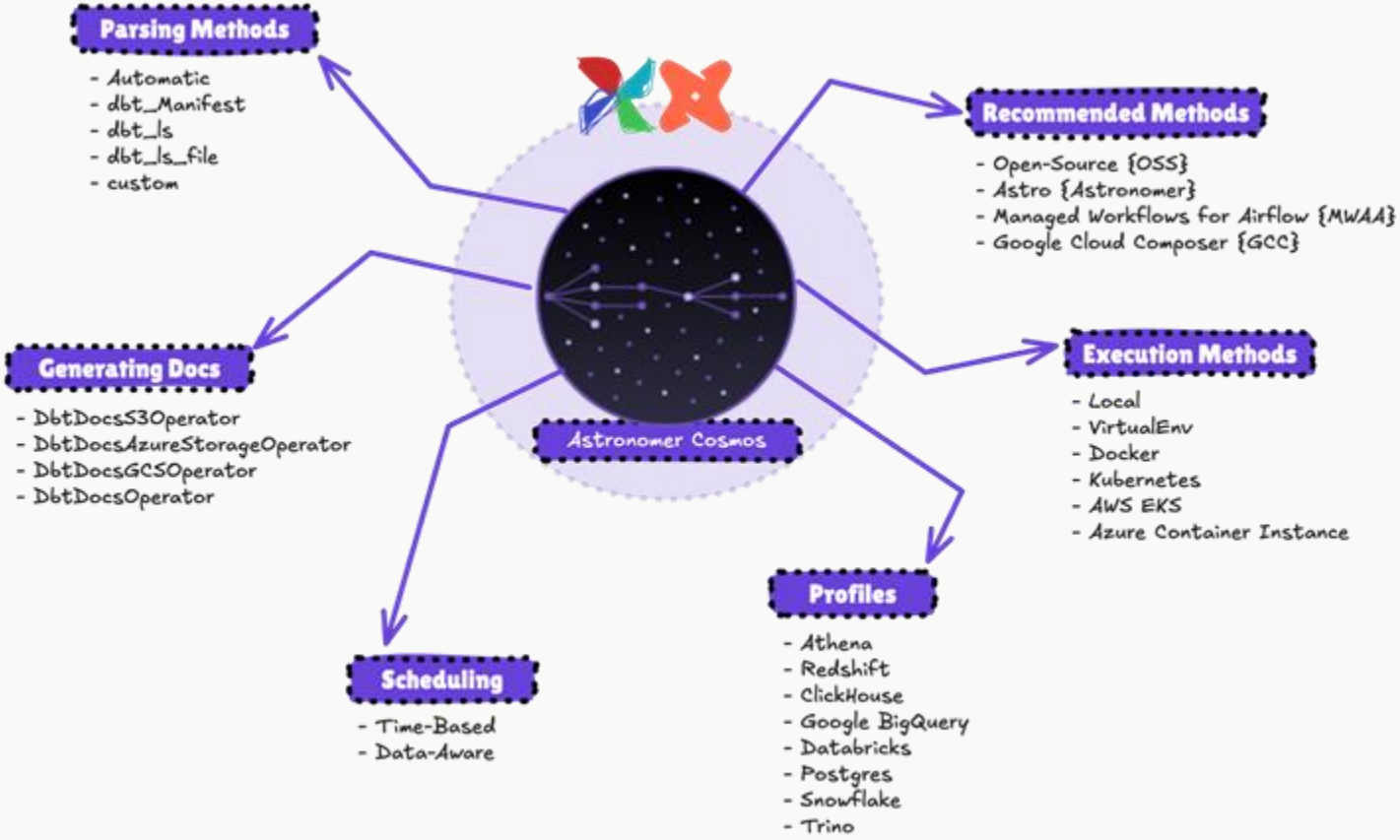
# Astronomer Cosmos 101

Seamless integration between the products



# Astronomer Cosmos Features

The goodies of running dbt-Core projects with Airflow



# DAGs in a Pythonic Way

The power of TaskFlow API & Task Group with dbt

```python
DBT_PROJECT_PATH = f"{os.environ['AIRFLOW_HOME']}/dags/dbt/my_simple_dbt_project"
DBT_EXECUTABLE_PATH = f"{os.environ['AIRFLOW_HOME']}/dbt_venv/bin/dbt"

profile_config = ProfileConfig(
    profile_name="default",
    target_name="dev",
    profile_mapping=PostgresUserPasswordProfileMapping(conn_id=CONNECTION_ID, profile_args={"schema":SCHEMA_NAME}))
execution_config = ExecutionConfig(dbt_executable_path=DBT_EXECUTABLE_PATH,)

@dag(
    start_date=datetime(2023, 8, 1),
)
def my_simple_dbt_dag():
    transform_data = DbtTaskGroup(
        group_id="transform_data",
        project_config=ProjectConfig(DBT_PROJECT_PATH),
        profile_config=profile_config,
        execution_config=execution_config,
        operator_args={"vars": '{"my_name": {{ params.my_name }} }',},
        default_args={"retries": 2},
    )
```

# What about the Best practices?

https://github.com/astronomer/astronomer-cosmos

# Key Takeaway 1: Execution Modes

Understand which execution **mode** is better for your project

## local

- The **default** method and fastest way
- Do not install dbt, assumes a **dbt binary is reachable** (dbt_executable_path)
- Starting in v1.4, it tries to leverage the **dbt partial parsing (partial_parse.msgpack**

## virtualenv

- **Isolates** Airflow worker dependencies from dbt by managing a **VirtualEnv** during task execution
- Drawback: It's **slower than local** execution but it runs in a isolated manner

## Docker

- Assumes a **pre-created Docker image** which contains the **dbt pipelines** and profile.yml
- Drawback: It's **slower than virtualenv** execution

## Kubernetes

- **Very isolated way** since dbt run commands from within a K8S pod, normally in separate host
- Container has **up-to-date dbt pipelines and profiles**
- Drawback: **Time to spin up** a Kubernetes Pod may take a while

# Key Takeaway 2: Parsing Methods

Several options in how to **parse** a dbt project

## automatic

- **Tries to find** a user supplied **manifest.json** if not, it will run a **dbt ls**, if fails use Cosmos' dbt parser
- This is the default method

## dbt_manifest

- Parses the user-supplied **manifest.json**, generated manually or through CI/CD pipelines
- **Benefit:** Generate the complete set of metadata for your models
- **Drawback:** Generating the manifest.json

Cosmos v.1.3

| File Path | Runtime | Type |
|---|---|---|
| cosmos_dag.py | 6.53s | without manifest.json |
| cosmos_dag.py | 0.35s | **with** manifest.json |

## dbt_ls

- **Parses** the dbt project using the **dbt ls** command. Cosmos generates the manifest file
- **Benefit:** uses the metadata and dbt select/excluding logic turning to be the **most robust** method
- **Drawback:** requires the dbt executable to be installed
- **Cached** in a Airflow variable **{v.15}**

## dbt_ls_file

- **New in v.1.3**, path to the file containing the dbt ls output, **use the dbt ls --output json**

## custom

- If the above methods fail, **Cosmos will default to using its own dbt parser**. This parser is not as robust as dbt's, so it's recommended that you use one of the above methods if possible.

# Do you want to know more?

## Wednesday, September 11, 2024

Integrating dbt with Airflow: Overcoming performance hurdles

*By Tatiana Al-Chueyr Martins & Pankaj Koti*
Track: Airflow & ...
Room: California West

# Questions?

https://www.linkedin.com/in/luanmoreno/