In this guide, we will walk through how to setup dbt Core in the cloud with Snowflake. After you finish this guide, you will have the sample data provided uploaded to Snowflake and run your first dbt command in the cloud.

Although the steps in this guide will specifically use Snowflake, the steps can be modified slightly to work with any database that [dbt supports](https://docs.getdbt.com/docs/available-adapters). We also have guides made specifically for [BigQuery](https://youtu.be/NDkuB9B_RQQ?feature=shared), [Databricks](https://youtu.be/fLyyUCdwiWs?feature=shared" \t "_blank), and [Redshift](https://youtu.be/OQ0ltcAT3sI?feature=shared).

If you would rather watch a video version of this guide, feel free to head over to [YouTube](https://www.youtube.com/playlist?list=PLsy6kuGU_wiPs7I3xLA4F6ffyWzQQW7eG). Let's jump right in!

**dbt Core Part 1 - Loading Sample Data into your Cloud Provider**

Before getting into the steps of setting up the different cloud data warehouses, please download the sample files that we will use for this tutorial [here](https://drive.google.com/drive/folders/15gLVxj5-aMee0CRZ7H9Ht-S9CkZ1jB6y?usp=sharing).

1. From the Snowflake homepage, click **Worksheets** on the top of the webpage.
2. Input the follow query::

USE ROLE accountadmin;

-- dbt roles

CREATE OR REPLACE ROLE dbt\_dev\_role;

CREATE OR REPLACE USER dbt\_user PASSWORD = "sailboats";

GRANT ROLE dbt\_dev\_role,dbt\_prod\_role TO USER dbt\_user;

GRANT ROLE dbt\_dev\_role,dbt\_prod\_role TO ROLE accountadmin;

CREATE OR REPLACE WAREHOUSE dbt\_dev\_wh WITH WAREHOUSE\_SIZE = 'XSMALL' AUTO\_SUSPEND = 60 AUTO\_RESUME = TRUE MIN\_CLUSTER\_COUNT = 1 MAX\_CLUSTER\_COUNT = 1 INITIALLY\_SUSPENDED = TRUE;

GRANT ALL ON WAREHOUSE dbt\_dev\_wh TO ROLE dbt\_dev\_role;

CREATE OR REPLACE DATABASE dbt\_hol\_dev;

GRANT ALL ON DATABASE dbt\_hol\_dev TO ROLE dbt\_dev\_role;

GRANT ALL ON ALL SCHEMAS IN DATABASE dbt\_hol\_dev TO ROLE dbt\_dev\_role

This query creates an example user, warehouse, and database to use throughout the tutorial.  
3. Run the queries.  
4. Click the **Databases** button on the top left of your screen. You should see the DBT\_HOL\_DEV database that we created:

[Une image contenant capture d’écran, logiciel, texte

Description générée automatiquement](https://www.shipyardapp.com/blog/content/images/2022/07/image-14.png)

1. Enter the following query into the worksheet to create our stg\_football\_rankings table:

CREATE TABLE DBT\_HOL\_DEV.PUBLIC.STG\_FOOTBALL\_RANKINGS(

rank integer,

prev\_rank integer,

name string,

league string,

off float,

def float,

spi float

)

Copy

1. Click **Run**.
2. Enter the following query into a worksheet to create our stg\_football\_matches table:

CREATE TABLE DBT\_HOL\_DEV.PUBLIC.STG\_FOOTBALL\_MATCHES(

season integer,

date date,

league\_id integer,

league string,

team1 string,

team2 string,

spi1 float,

spi2 float,

prob1 float,

prob2 float,

probtie float,

proj\_score1 float,

proj\_score2 float,

importance1 float,

importance2 float,

score1 integer,

score2 integer,

xg1 float,

xg2 float,

nsxg1 float,

nsxg2 float,

adj\_score1 float,

adj\_score2 float

)

Copy

**Load Data into Tables**

1. Click **Databases** on the top of your Snowflake page.
2. Click DBT\_HOL\_DEV
3. Click STG\_FOOTBALL\_RANKINGS.
4. Click **Load Data** which will bring up the load data menu.
5. Choose **DBT\_DEV\_WH** as the warehouse to load data. Click next.
6. Click **Select Files** and select spi\_global\_rankings.csv. Click next.
7. Click the plus sign next to the drop down to create our file format.
8. Under **name**, enter **dbt\_tutorial\_csv**.
9. Change header lines to skip to 1 from 0.
10. Keep the other settings at their default.
11. Click **Finish**.
12. Click **Load**.
13. After the data has loaded into Snowflake, you will receive a success message that looks like this:

[Une image contenant texte, capture d’écran, Police, nombre

Description générée automatiquement](https://www.shipyardapp.com/blog/content/images/2022/07/image-17.png)

1. Click **Databases** on the top of your Snowflake page.
2. Click DBT\_HOL\_DEV.
3. Click STG\_FOOTBALL\_MATCHES.
4. Click **Load Data** which will bring up the load data menu.
5. Choose **DBT\_DEV\_WH** as the warehouse to load data. Click next.
6. Click **Select Files** and select spi\_matches\_latest.csv. Click next.
7. Use the dropdown menu to select the **dbt\_tutorial\_csv** as the file format.
8. Click **Finish**.
9. Click **Load**.

You should be able to see both tables listed under your DBT\_HOL\_DEV database now.

[Une image contenant texte, logiciel, Icône d’ordinateur, Système d’exploitation

Description générée automatiquement](https://www.shipyardapp.com/blog/content/images/2022/07/image-18.png)

1. Click on STG\_FOOTBALL\_RANKINGS and click **Grant Privileges** on the right of the screen. Grant all actions to the DBT\_DEV\_ROLE.
2. Repeat this for STG\_FOOTBALL\_MATCHES.

You should now be able to query "DBT\_HOL\_DEV"."PUBLIC"."STG\_FOOTBALL\_RANKINGS" and "DBT\_HOL\_DEV"."PUBLIC"."STG\_FOOTBALL\_MATCHES". Feel free to run this query to verify that this process worked successfully:

select \* from "DBT\_HOL\_DEV"."PUBLIC"."STG\_FOOTBALL\_MATCHES"

Copy

**dbt Core Part 2 - Setting Up dbt on Github**

**Fork dbt Setup from GitHub**

1. Fork [this repository](https://github.com/shipyardapp/dbt-guide-starting-point" \t "_blank). The repository contains the beginning state of a dbt project.
2. Clone the repository locally on your computer.
3. Open dbt\_project.yml in your text editor.

[Une image contenant texte, capture d’écran, affichage, logiciel

Description générée automatiquement](https://www.shipyardapp.com/blog/content/images/2022/07/image-19.png)

**dbt Project File Setup**

1. Change the project name to soccer\_538.
2. Change the profile to soccer\_538.
3. Change model name to soccer\_538.
4. Under the soccer\_538 model, add a staging and marts folder that are both materialized as views.
5. Save your changes.

**Profile Setup**

1. Open profiles.yml.
2. Update the file to this:

soccer\_538:

target: dev

outputs:

dev:

type: snowflake

account: "{{ env\_var('snowflake\_trial\_account') }}"

user: dbt\_user

password: "{{ env\_var('dbt\_user\_password') }}"

role: dbt\_dev\_role

database: dbt\_hol\_dev

warehouse: dbt\_dev\_wh

schema: soccer\_538

threads: 200

1. Create a new file in your root directory of your dbt project called execute\_dbt.py.
2. Paste this code block for the content of execute\_dbt.py:

import subprocess

import os

import json

dbt\_command = os.environ.get('dbt\_command', 'dbt run')

subprocess.run(['sh', '-c', dbt\_command], check=True)

Copy

1. Commit and push your changes to Github.

Now that we have our sample data and dbt processes setup, we need to write our example models for the dbt job to run.

**dbt Models**

1. Navigate into the models folder in your text editor. There should be a subfolder under models called example. Delete that subfolder and create a new folder called 538\_football.
2. Create two subfolders inside 538\_football called staging and marts.

[Une image contenant texte, capture d’écran, Police, ligne

Description générée automatiquement](https://www.shipyardapp.com/blog/content/images/2022/07/image-20.png)

1. Inside the staging folder, create a file called stg\_football\_matches.sql.
2. Paste the following code into that file:  
   select \* from "DBT\_HOL\_DEV"."PUBLIC"."STG\_FOOTBALL\_MATCHES"
3. Inside the staging folder, create a file called stg\_football\_rankings.sql
4. Paste the following code into that file:  
   select \* from "DBT\_HOL\_DEV"."PUBLIC"."stg\_football\_rankings"
5. In the staging folder, add a file called schema.yml.
6. In this file, paste the following information:

version: 2

models:

- name: stg\_football\_matches

description: Table from 538 that displays football matches and predictions about each match.

- name: stg\_football\_rankings

description: Table from 538 that displays a teams ranking worldwide

1. In the marts folder, create a file called mart\_football\_information.sql.
2. Paste the following code into that file:

with

qryMatches as (

SELECT \* FROM {{ ref('stg\_football\_matches') }} where league = 'Barclays Premier League'

),

qryRankings as (

SELECT \* FROM {{ ref('stg\_football\_rankings') }} where league = 'Barclays Premier League'

),

qryFinal as (

select

qryMatches.season,

qryMatches.date,

qryMatches.league,

qryMatches.team1,

qryMatches.team2,

team\_one.rank as team1\_rank,

team\_two.rank as team2\_rank

from

qryMatches join

qryRankings as team\_one on

(qryMatches.team1 = team\_one.name) join

qryRankings as team\_two on

(qryMatches.team2 = team\_two.name)

)

select \* from qryFinal

Copy

1. In the marts folder, add a file called schema.yml
2. In this file, paste the following:

version: 2

models:

- name: mart\_football\_information

description: Table that displays football matches along with each team's world ranking.

1. Save the changes.
2. Push a commit to Github

We are ready to move into **Shipyard** to run our process. First, you will need to create a developer account.

**dbt Core Part 3 - Setting Up dbt on Shipyard**

**Create Developer Shipyard Account**

1. Navigate to **Shipyard's** sign-up page [here](https://app.shipyardapp.com/auth/signup?ref=header).

[Une image contenant texte, nuage, capture d’écran, embarcation

Description générée automatiquement](https://www.shipyardapp.com/blog/content/images/2022/07/image-21.png)

1. Sign up with your email address and organization name.
2. Connect to your Github account by following this [guide](https://www.shipyardapp.com/blog/automate-deployment-github-code/). After connecting your Github account, you'll be ready to create your first Blueprint.

**Creating dbt Core Blueprint**

1. On the sidebar of **Shipyard's** website, click **Blueprints**.
2. Click **Add Blueprint** on the top right of your page.
3. Select **Python**.
4. Under **Blueprint variables**, click **Add Variable**.
5. Under **display name**, enter dbt CLI Command.
6. Under **reference name**, enter dbt\_command.
7. Under **default value**, enter dbt run.
8. Click the check box for required
9. Under **placeholder**, enter Enter the command for dbt.
10. Click Next
11. Click **Git**.

[Une image contenant texte, capture d’écran

Description générée automatiquement](https://www.shipyardapp.com/blog/content/images/2022/07/image-22.png)

1. Select the repository where your dbt files sit.
2. Click the source that you want the files pulled from. Generally **main** or **master**.
3. Under **file to run**, enter execute\_dbt.py.
4. Under **Git Clone Location**, select the option for [Unpack into Current Working Directory](https://www.shipyardapp.com/docs/reference/code/git-connection/" \l "unpack-into-current-working-directory" \t "_blank).
5. Click **Next Step** on the bottom right of the screen.
6. Next to **Environment Variable**, click the plus sign to add an environment variable.

**Add Environment Variables**

The environment variables that need to be added will vary based on the cloud database that you use.

| **Variable Name** | **Value** |
| --- | --- |
| snowflake\_trial\_account | snowflake account name |
| dbt\_user\_password | password from snowflake |
| DBT\_PROFILES\_DIR | . |

**Python Packages**

1. Click the plus sign next to **Python Packages**.
2. In the **Name** field, enter dbt-snowflake. In the version field, enter ==1.0.0.
3. Click **Next**.

**Blueprint Settings**

1. Under **Blueprint Name**, enter dbt - Execute CLI Command.
2. Under **synopsis**, enter This Blueprint runs a dbt core command.
3. Click **Save**.
4. In the top right of your screen, click **Use this Blueprint**. This will take you over to the Fleet Builder and prompt you to select a project.

**Build dbt Core Fleet**

1. On the **Select a Project** prompt, click the drop down menu to expand it and select Create a New Project.
2. Under project name, enter dbt Core Testing.
3. Under **timezone**, enter your timezone.

[Une image contenant texte, capture d’écran, nombre, Police

Description générée automatiquement](https://www.shipyardapp.com/blog/content/images/2022/07/image-23.png)

1. Click **Create Project**.
2. Select dbt Core Testing and click Select Project. This will create a new Fleet in the project. The Fleet Builder will now visible with one Vessel located inside of the Fleet.
3. Click on the Vessel in the Fleet Builder and you will see the settings for the Vessel pop up on the left of your screen.

[Une image contenant texte, capture d’écran, logiciel, Icône d’ordinateur

Description générée automatiquement](https://www.shipyardapp.com/blog/content/images/2022/07/image-24.png)

1. Under **Vessel Name**, enter dbt Core CLI Command.
2. Under **dbt CLI Command**, enter dbt debug.
3. Click the gear on the sidebar to open Fleet Settings.

[Une image contenant texte, capture d’écran, logiciel, Icône d’ordinateur

Description générée automatiquement](https://www.shipyardapp.com/blog/content/images/2022/07/image-25.png)

1. Under **Fleet Name**, enter dbt Core.
2. Click **Save & Finish** on the bottom right of your screen.
3. This should take you to a page showing that your Fleet was created successfully.

[Une image contenant texte, capture d’écran, logiciel, Système d’exploitation

Description générée automatiquement](https://www.shipyardapp.com/blog/content/images/2022/07/image-26.png)

1. Click **Run Your Fleet**. This will take you over to the Fleet Log.

[Une image contenant texte, capture d’écran, logiciel, Système d’exploitation

Description générée automatiquement](https://www.shipyardapp.com/blog/content/images/2022/07/image-27.png)

1. You can click on the bar to get the output from your run.

[Une image contenant texte, capture d’écran, logiciel, Page web

Description générée automatiquement](https://www.shipyardapp.com/blog/content/images/2022/07/image-28.png)

If you scroll to the top of the output, you will see that the environment variables that were put in during the Blueprint creation process are hidden from the user.

[Une image contenant texte, logiciel, nombre, Page web

Description générée automatiquement](https://www.shipyardapp.com/blog/content/images/2022/07/image-29.png)

If dbt debug succeeds, we are ready to move into part three of the guide. If it fails, please go back to the steps above and make sure everything is setup correctly. Feel free to send an Intercom message to us at anytime using the widget on the bottom right of the Shipyard screen.