

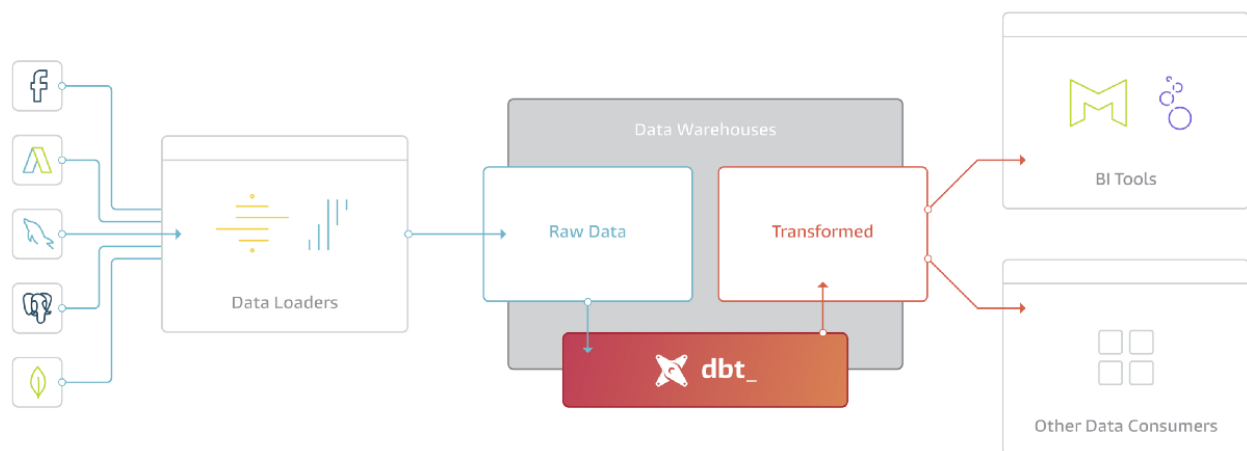
# dbt for Transformation Tasks on BigQuery

## Introduction: What Is dbt?

Two common approaches to enable the flow of big data are ELT (extract, load, transform) and ETL (extract, transform, load). Both start with unstructured data and despite the slight difference in naming, they result in distinct practices of data engineering:

- ELT prioritizes loading and keeps transformation for a later time. It would deal with basic pre-processing such as removing duplicated data or filling missing values before serving to a team who is supposed to transform it.
- ETL focuses on transformation before delivering to target systems/teams. Hence, it does more than preprocessing in ELT that would involve making data structured, clean and type transformed.

dbt is a tool to conduct transformation (“T”) practices on data warehouses for ELT. As the name suggests, it involves the operations after the data is extracted and loaded. In other words, once you have “landed” your big data on a data warehouse, dbt can help you to pre-process before serving for the use of subsequent processes. The visualization given below shows its role in a data pipeline.

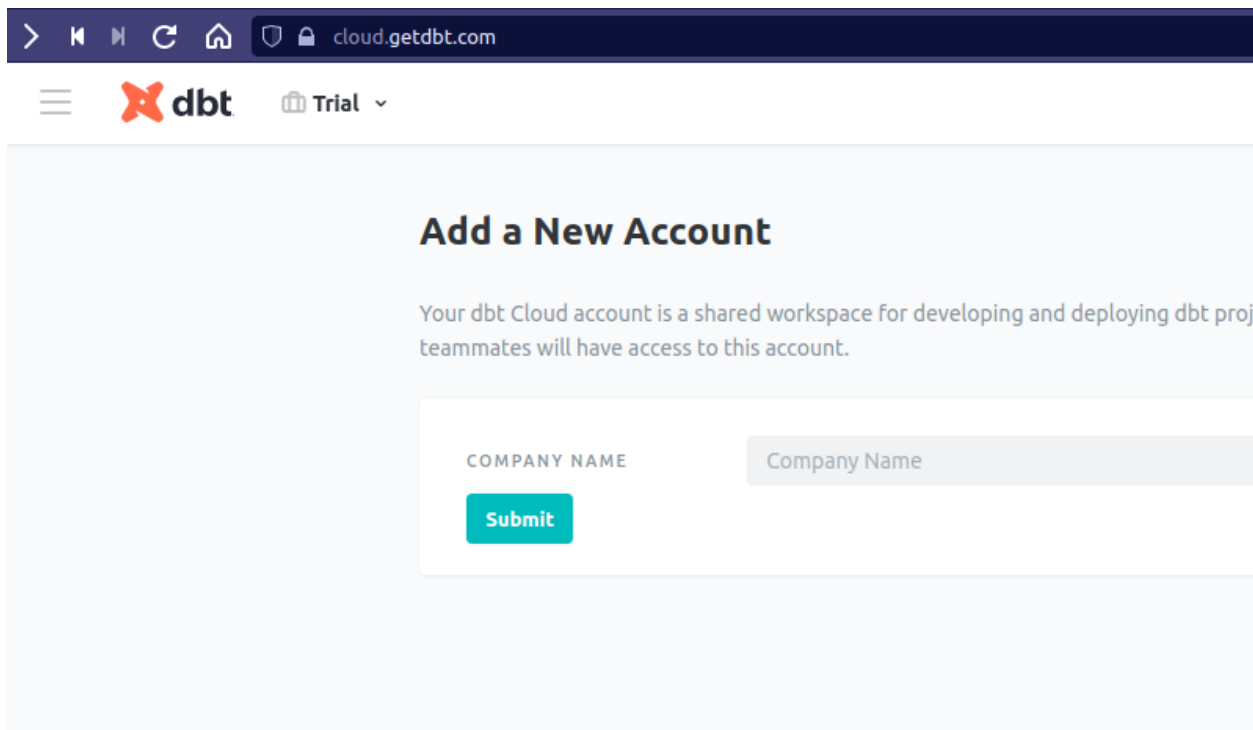


# Tutorial for BigQuery Transformations

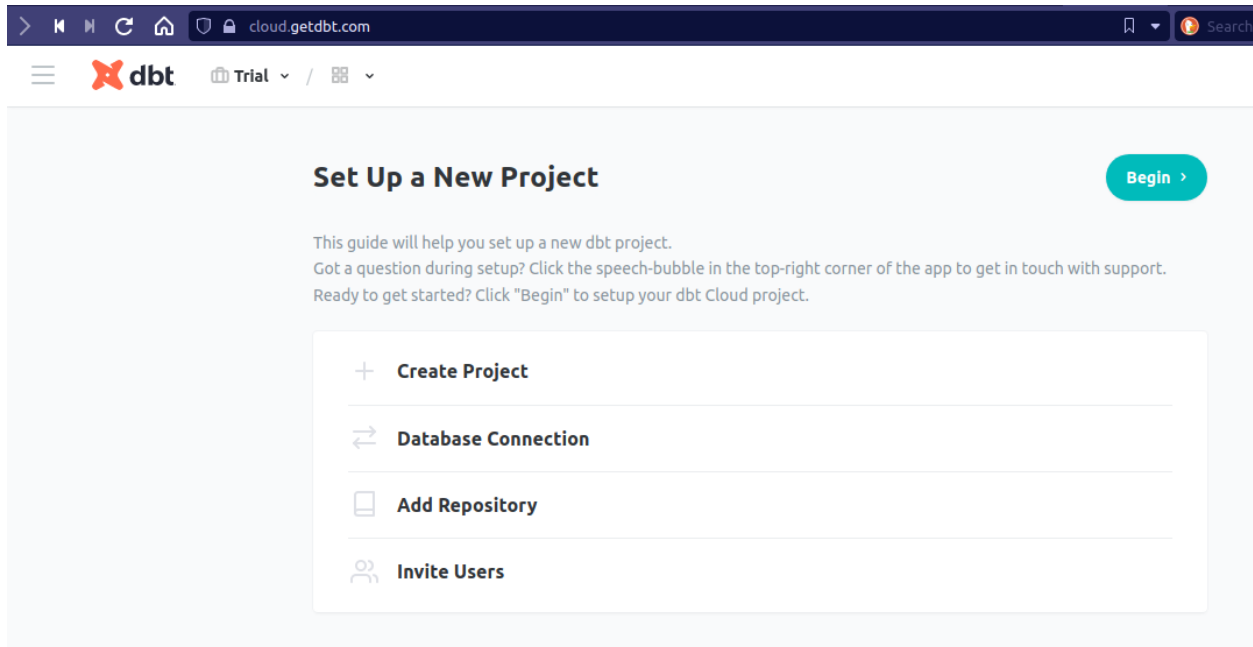
dbt is originally a command line tool but currently it has a cloud service ([www.getdbt.com](http://www.getdbt.com)) as well that makes the initial steps more convenient for newbies. In this short tutorial, dbt cloud service is used to conduct some basic transformation tasks on the data uploaded from a public dataset (the process was explained here) as a part of Data Engineering Zoomcamp (by DataTalks.club) Capstone Project.

## Step-1: Initiate a project on dbt cloud

The process starts with initiation of an account and a project on dbt cloud that is free for individuals:



The screenshot shows the dbt Cloud website interface. The browser address bar displays 'cloud.getdbt.com'. The navigation bar includes the dbt logo and a 'Trial' status. The main heading is 'Add a New Account'. Below this, a descriptive text states: 'Your dbt Cloud account is a shared workspace for developing and deploying dbt projects. Teammates will have access to this account.' The form contains a text input field labeled 'COMPANY NAME' with the placeholder text 'Company Name'. A teal 'Submit' button is positioned below the input field.



## Step-2: Match with BigQuery

Clicking on the “Create Project” is followed by simple questions to declare project name and data warehouse for integration. Selecting BigQuery as the data warehouse, you will land on a page to assign GCP service account information. (Note that the account has to have BigQuery crenetdials.)

## Set Up a Database Connection

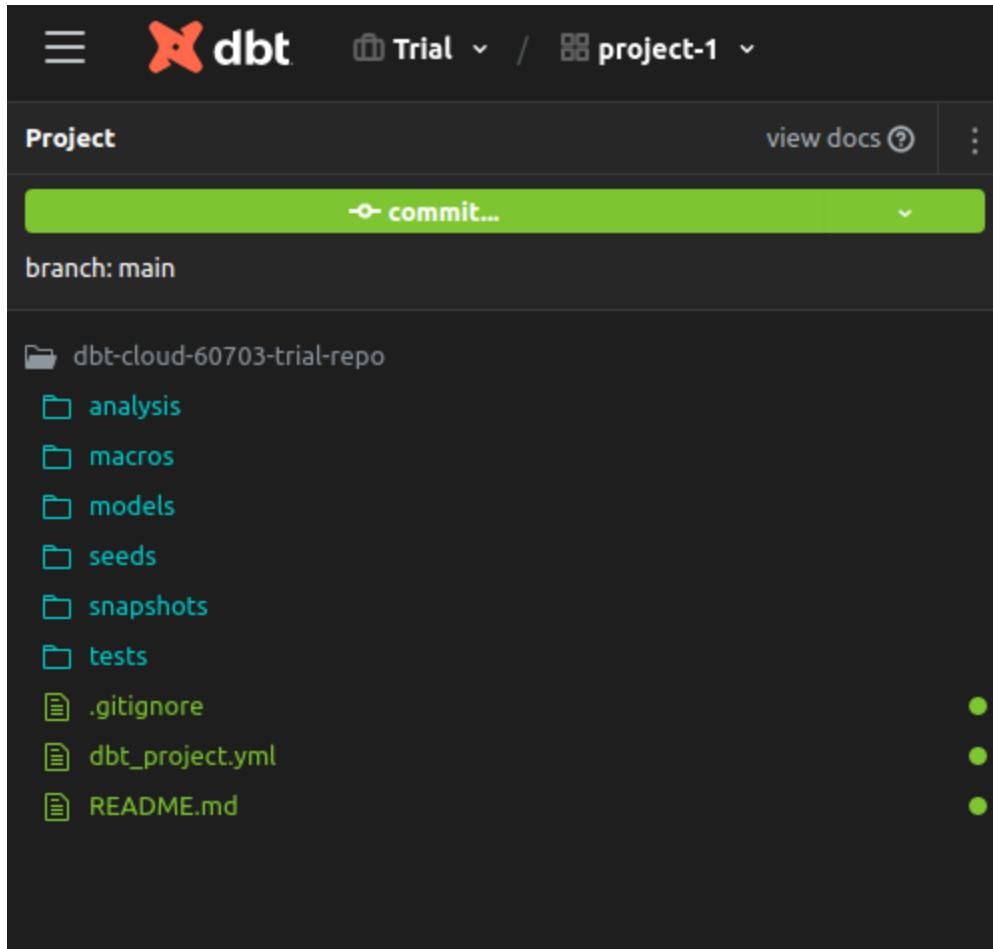
[Test](#)[Continue >](#)

TYPE	bigquery
NAME	Bigquery

### BigQuery Settings

CREATE FROM FILE	<a href="#">Upload a Service Account JSON file</a>
PROJECT ID	
PRIVATE KEY ID	
PRIVATE KEY	
CLIENT EMAIL	
CLIENT ID	
AUTH URI	
TOKEN URI	
AUTH PROVIDER X509 CERT URL	
CLIENT X509 CERT URL	

Here, downloading the service account information as a Json file from GCP and uploading it to the dbt would prevent error. Testing the authorization, one can continue for matching the repository to host the dbt project. It is possible to host it on a “dbt Cloud managed repository” or on another repo of choice. Completing the step, you will have a project ready to initialize on dbt cloud. Clicking on “initialize your project” button, you will have a fresh project template:



### Step3: Identify the required components and configurations

For a dbt project, core elements are:

- **dbt\_project.yml** file that is to configure the project,
- **models** folder to host models to be run for proposed transformations,
- **macros** to involve files to declare reusable SQL queries in Jinja format,
- **seeds** folder to host CSV files for declarations regarding data on data warehouse such as zip code & city names, employee ID & personal data' etc. Note that these are not the data itself but the references to use it properly.

### Step4: Define model

In this introductory tutorial, we will only use the models. The task is to unify all daily data of a PV system produced during a year. That is to say unify 365 files. This can be

done with **UNION ALL** query as below:

```
-- Select columns of interest
SELECT measured_on, system_id, \
    ac_power__5074 as ac_power, \
    ambient_temp__5042 as ambient_temp,
    kwh_net__5046 as kwh_net, \
    module_temp__5043 as module_temp, \
    poa_irradiance__5041 as poa_irradiance,
    pr__5047 as pr

-- Declare one of the files to be combined
FROM `project_name.dataset_name.table_name`

UNION ALL

SELECT ...
...
...
...
```

However, writing 365 lines is not convenient, of course. Hence, it is possible to take advantage of a property of BigQuery. It helps you synthesize long queries with a obviously repetitive pattern. For the case study of the tutorial, it was possible to produce a 365 lines of **UNION ALL** query for a year by running the following query on BigQuery:

```
SELECT string_agg(concat("SELECT measured_on, \
    system_id, ac_power__5069 as ac_power, \
    ambient_temp__5062 as ambient_temp, \
    kwh_net__5066 as kwh_net, \
    module_temp__5063 as module_temp, \
    poa_irradiance__5061 as poa_irradiance, \
    pr__5067 as pr \
    FROM `YOUR-BQ-PROJECT-NAME.pvsys1433.", \
    table_id, "`" ) , " UNION ALL \n") \

FROM YOUR-BQ-PROJECT-NAME.pvsys1433.__TABLES__;
```

Then, you can simply copy-paste the long query to the model file created on models/staging folder of your project in dbt cloud. Running model with **dbt run your\_model\_name.sql** command, you will receive a new table on your corresponding BigQuery project dataset with the name of your model file.

