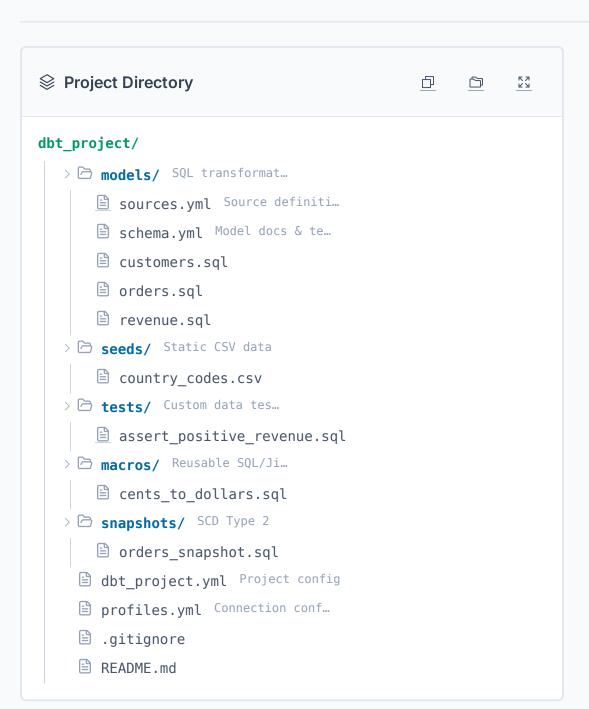
Minimal Project Structure

Simple dbt setup for small data teams. Flat model structure, basic tests, and minimal configuration overhead.

#dbt #sql #data #analytics #warehouse #elt





○ Why This Structure?

This flat structure works for small teams with <20 models. All models live in one folder with a single schema.yml for documentation and tests. Simple to navigate, minimal overhead. Upgrade to modular when you hit 20+ models.

models/ - SQL files that become views or tables in your warehouse
models/sources.yml - Declare raw tables from your warehouse
models/schema.yml - Column docs, tests, and model descriptions
macros/ - Reusable Jinja functions across models

- 1. pip install dbt-postgres (or dbt-snowflake, dbt-bigquery)
- 2. dbt init my_project && cd my_project
- 3. Configure profiles.yml with your warehouse credentials
- 4. dbt debug to test connection
- 5. dbt run to build models

</> Model Example

```
-- models/customers.sql
with source as (
    select * from {{ source('raw', 'customers') }}
),

renamed as (
    select
        id as customer_id,
        name as customer_name,
            created_at
        from source
)

select * from renamed
```

☑ When To Use This

- Small teams with <20 models
- Single data source or simple pipelines
- Getting started with dbt
- Proof of concept or prototyping

↑ When To Upgrade

- More than 20 models becoming hard to navigate
- Multiple developers stepping on each other
- Need clear staging vs. marts separation
- Complex dependencies across domains