dbt + python = dbt-fal
python within dbt models

Who are we?

Som Energia is a **consumer cooperative**

- Renewable green energy
- not-for-profit
- Solidarity-based community
- Present through **Local Groups**
- **More than ~83k members**
- **~120k contracts**
- **65GWh/year @ 17 plants**
- **https://github.com/Som-Energia**
- **since 2010**

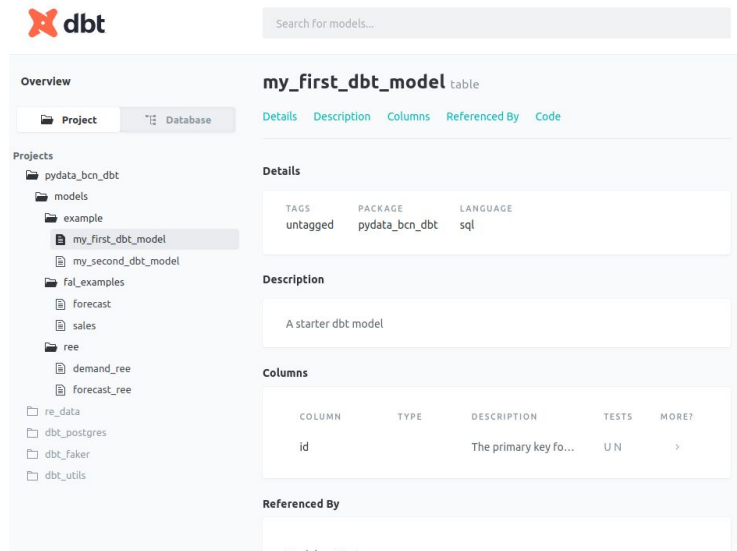Producció ⊕ → Transport 🔒 Distribució 🔒 → Comercialització ⊕

More details

What is dbt?

- SQL-first **transformation** framework
- Data **modeling**, **testing** and **documentation**
- configure with **YAML**
- Relies on Jinja2 **SQL** templates
- Compatible with **CI/CD** and **Git**
- Great community
- Great documentation, examples and discussions

dbt is SQL only

```sql
-- my_first_dbt_model.sql
{{ config(materialized="table") }}
with
    source_data as (
        select 1 as id
        union all
        select null as id
    )
select *
from source_data
where id is not null
```

```sql
-- Use the `ref` function to select from other models
select * from {{ ref("my_first_dbt_model") }}
where id = 1
```

Python support is limited (for now)

- Targets snowflake, databricks and BigQuery only
- More **flexible** than *just* SQL
  - ML models
  - Notifications
  - etc
- pandas recommendation is still not clear. Some alternatives are ibis and polars
- Not the best debugging experience, some workarounds exist

```python
# model.py
def model(dbt, session):
    # config
    dbt.config(
        materialized="table")
    df = dbt.ref("my_model")
    ...
    # some transformations taking place
    return df
```
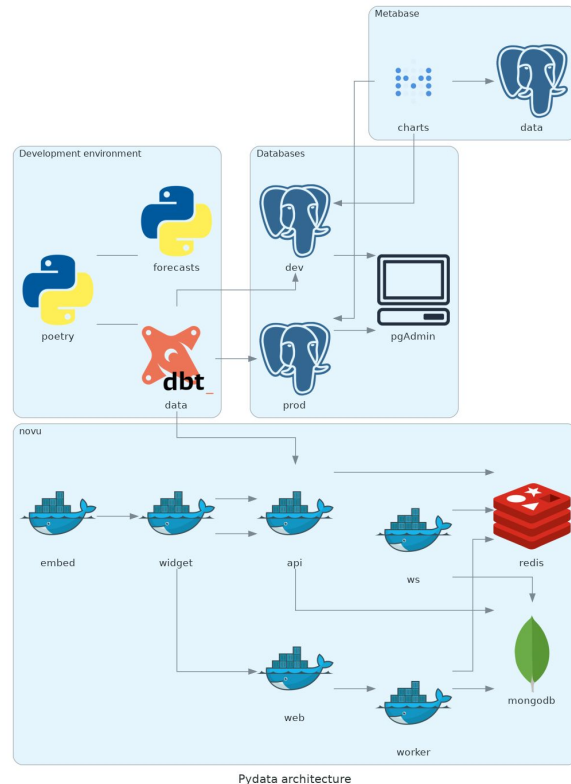
dbt + fal = [dbt-fal](#)

- Python **adapter**
- Brings in support to **other solutions** e.g. **Postgres**
- **Virtual environment** support

```yaml
# profiles.yml
pydata_bcn_dbt:
 outputs:
  dev:
    type: postgres
  dev_with_fal: # new
    type: fal # new
    db_profile: dev # new
 target: dev_with_fal # new
```

```yaml
# schema.yml
---
version: 2
models:
 - name: my_sales_model
   description: A model using fal-dbt
   config:
     materialized: table
   meta:
     fal:
       scripts:
         - model.py
```

Demo

- **docker** + **docker compose**
- **novu** as notification structure
- **postgres** as dev/prod databases
- **pgadmin** for db admin
- **metabase** for BI
- **devcontainer** with poetry
  - dbt, dbt-fal, prophet, novu-client and other goodies
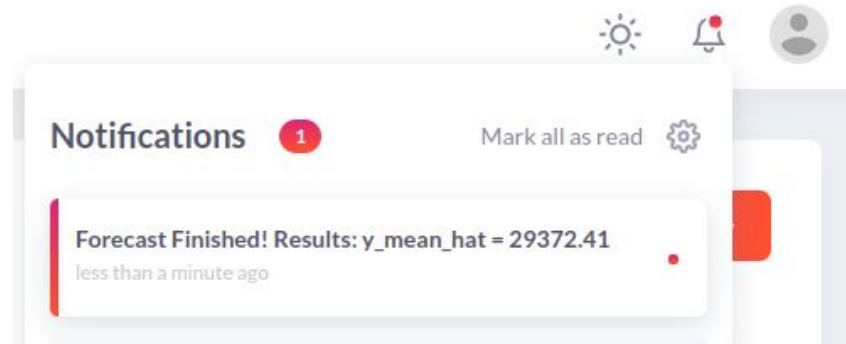- **not a production setting**
- github repo

Pydata architecture

done with *https://diagrams.mingrammer.com/*

Demo

Demo Outline

- CSV data from REE
- Loaded to table using **dbt-fal**
- Forecast using **Prophet**
- Sends notification to **Novu** client

```python
# report the forecast to novu
event_api = EventApi(
    "http://novu-api:3000",
    os.environ.get("NOVU_API_KEY"))
yhat_mean = forecast["yhat"].mean().item()
event_api.trigger(
    name=os.environ.get("NOVU_APP_NAME"),
    recipients=os.environ.get("NOVU_SUBSCRIBER_ID"),
    payload={"results": {"y_mean_hat": f"{yhat_mean:0.2f}"}})
```

Notifications 1   Mark all as read

Forecast Finished! Results: y_mean_hat = 29372.41
less than a minute ago

Takeaways

- dbt makes pipelines easier to test, document and manage
- dbt relies on SQL and YAML for the most part
- Some tasks can't be / should not be implemented using SQL
- dbt-fal lets you run models using python
- Interfaces are a bit clunky at the moment, but support *will* increase

fal examples

Thank you!

dades@somenergia.coop
diego.quintana@somenergia.coop

https://github.com/Som-Energia/
https://github.com/diegoquintanav/