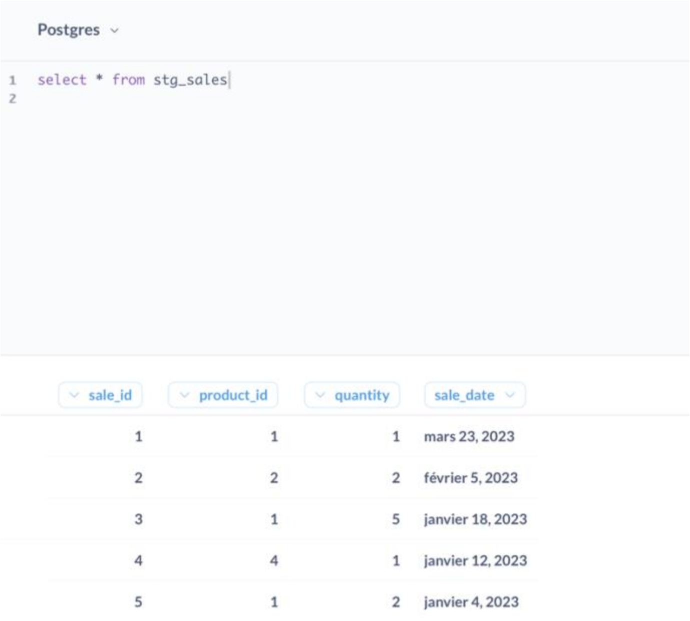
TP1 – Data Science
Asencio Victor

3) Creating Staging Models

3.1) Tout marche youhou !

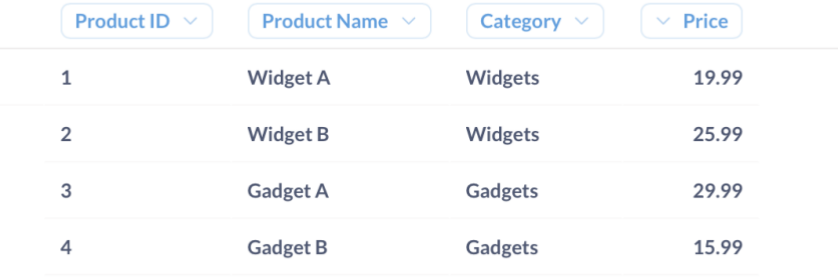


```
select * from stg_sales
```

sale_id	product_id	quantity	sale_date
1	1	1	mars 23, 2023
2	2	2	février 5, 2023
3	1	5	janvier 18, 2023
4	4	1	janvier 12, 2023
5	1	2	janvier 4, 2023

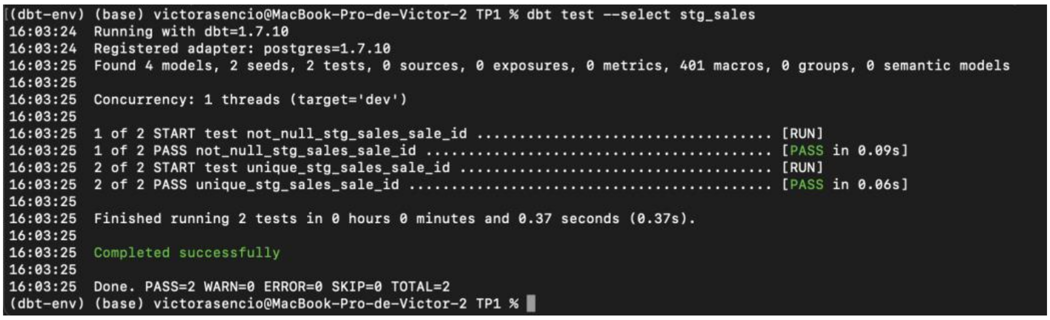
3.2)

Après correction de la petite erreur dans le git

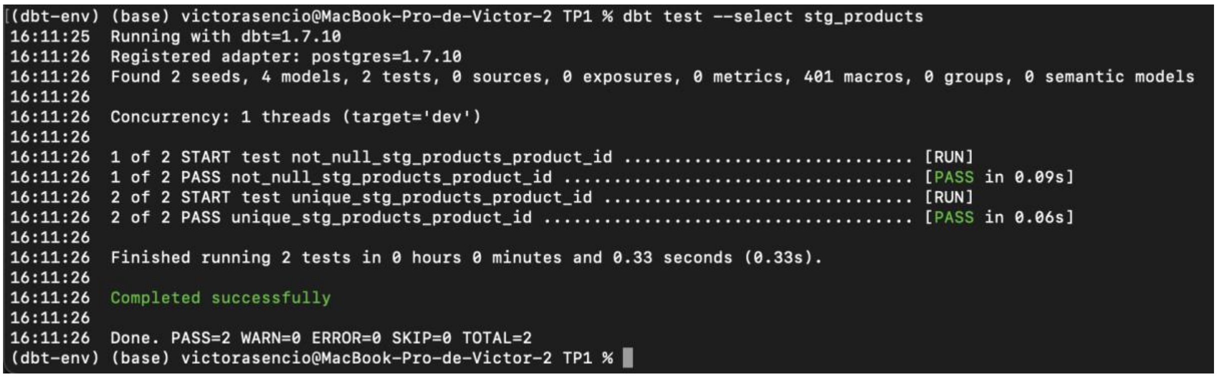


Product ID	Product Name	Category	Price
1	Widget A	Widgets	19.99
2	Widget B	Widgets	25.99
3	Gadget A	Gadgets	29.99
4	Gadget B	Gadgets	15.99

3.3) Après la commande dbt test –select stg_sales

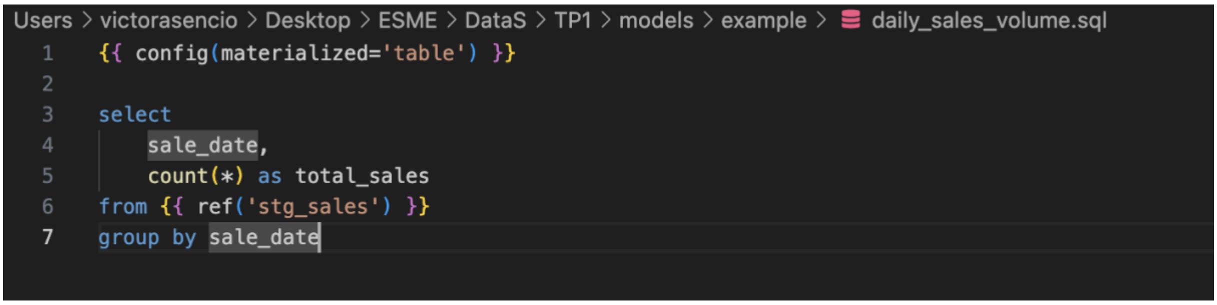


```
[(dbt-env) (base) victorasencio@MacBook-Pro-de-Victor-2 TP1 % dbt test --select stg_sales
16:03:24  Running with dbt=1.7.10
16:03:24  Registered adapter: postgres=1.7.10
16:03:25  Found 4 models, 2 seeds, 2 tests, 0 sources, 0 exposures, 0 metrics, 401 macros, 0 groups, 0 semantic models
16:03:25
16:03:25  Concurrency: 1 threads (target='dev')
16:03:25
16:03:25  1 of 2 START test not_null_stg_sales_sale_id ................................... [RUN]
16:03:25  1 of 2 PASS not_null_stg_sales_sale_id ......................................... [PASS in 0.09s]
16:03:25  2 of 2 START test unique_stg_sales_sale_id ..................................... [RUN]
16:03:25  2 of 2 PASS unique_stg_sales_sale_id ........................................... [PASS in 0.06s]
16:03:25
16:03:25  Finished running 2 tests in 0 hours 0 minutes and 0.37 seconds (0.37s).
16:03:25
16:03:25  Completed successfully
16:03:25
16:03:25  Done. PASS=2 WARN=0 ERROR=0 SKIP=0 TOTAL=2
(dbt-env) (base) victorasencio@MacBook-Pro-de-Victor-2 TP1 %
```



```
[(dbt-env) (base) victorasencio@MacBook-Pro-de-Victor-2 TP1 % dbt test --select stg_products
16:11:25  Running with dbt=1.7.10
16:11:26  Registered adapter: postgres=1.7.10
16:11:26  Found 2 seeds, 4 models, 2 tests, 0 sources, 0 exposures, 0 metrics, 401 macros, 0 groups, 0 semantic models
16:11:26
16:11:26  Concurrency: 1 threads (target='dev')
16:11:26
16:11:26  1 of 2 START test not_null_stg_products_product_id ............................. [RUN]
16:11:26  1 of 2 PASS not_null_stg_products_product_id ................................... [PASS in 0.09s]
16:11:26  2 of 2 START test unique_stg_products_product_id ............................... [RUN]
16:11:26  2 of 2 PASS unique_stg_products_product_id ..................................... [PASS in 0.06s]
16:11:26
16:11:26  Finished running 2 tests in 0 hours 0 minutes and 0.33 seconds (0.33s).
16:11:26
16:11:26  Completed successfully
16:11:26
16:11:26  Done. PASS=2 WARN=0 ERROR=0 SKIP=0 TOTAL=2
(dbt-env) (base) victorasencio@MacBook-Pro-de-Victor-2 TP1 %
```

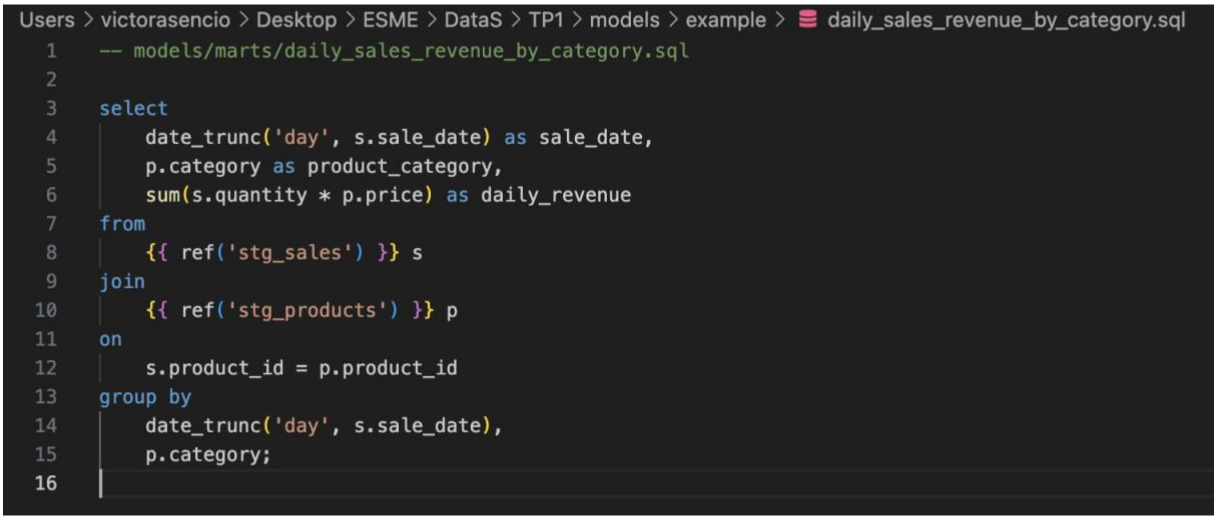
4) Marts Models



Users > victorasencio > Desktop > ESME > DataS > TP1 > models > example > daily_sales_volume.sql

```sql
{{ config(materialized='table') }}

select
    sale_date,
    count(*) as total_sales
from {{ ref('stg_sales') }}
group by sale_date
```



Users > victorasencio > Desktop > ESME > DataS > TP1 > models > example > daily_sales_revenue_by_category.sql

```sql
-- models/marts/daily_sales_revenue_by_category.sql

select
    date_trunc('day', s.sale_date) as sale_date,
    p.category as product_category,
    sum(s.quantity * p.price) as daily_revenue
from
    {{ ref('stg_sales') }} s
join
    {{ ref('stg_products') }} p
on
    s.product_id = p.product_id
group by
    date_trunc('day', s.sale_date),
    p.category;

```

5) Creating dashboards with metabase

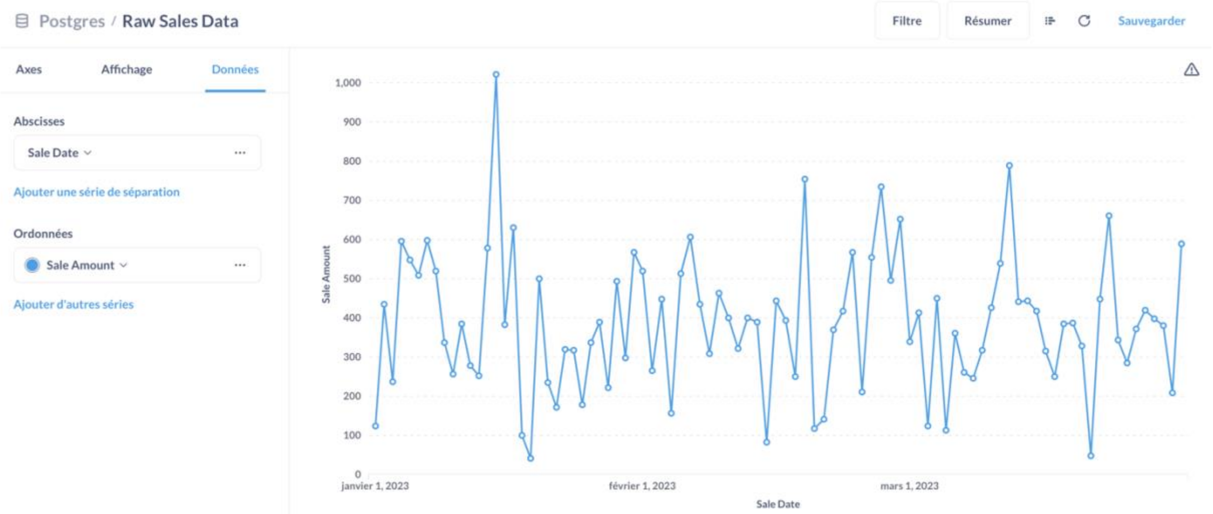


Figure 1: Sales over time

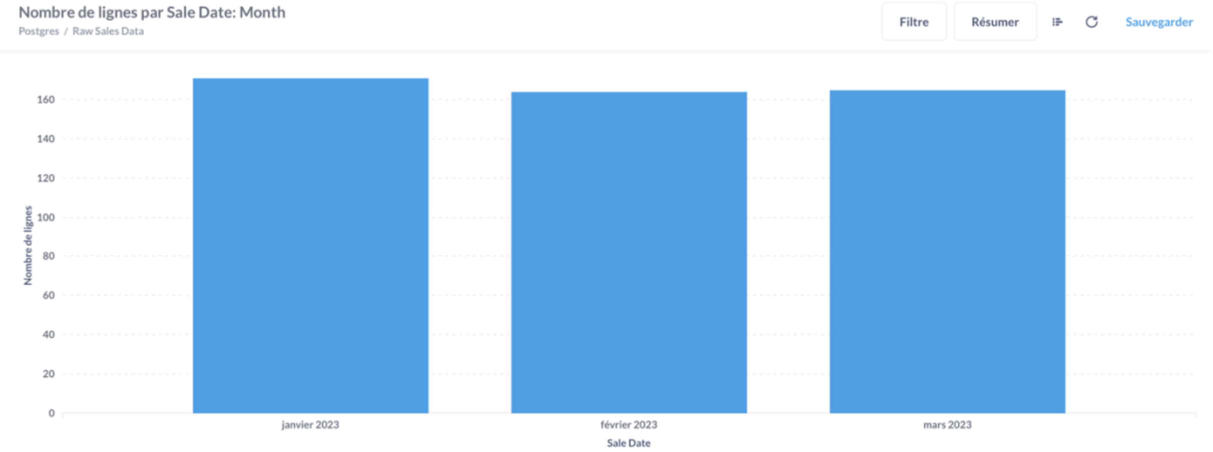


Figure 2: Sales over time (month)

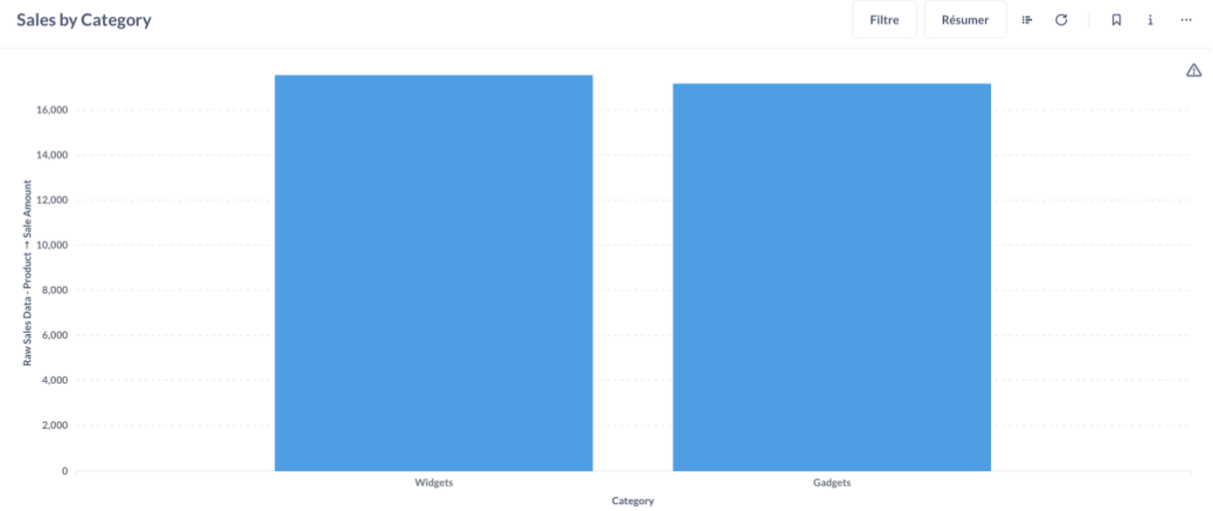
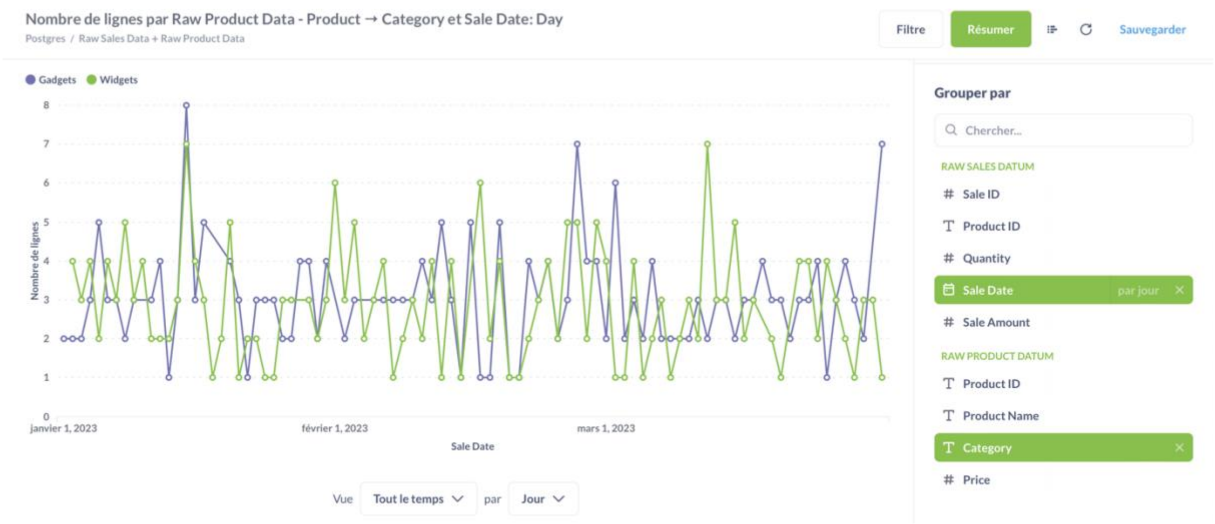


Figure 3: Sales by category (join on product_ID)



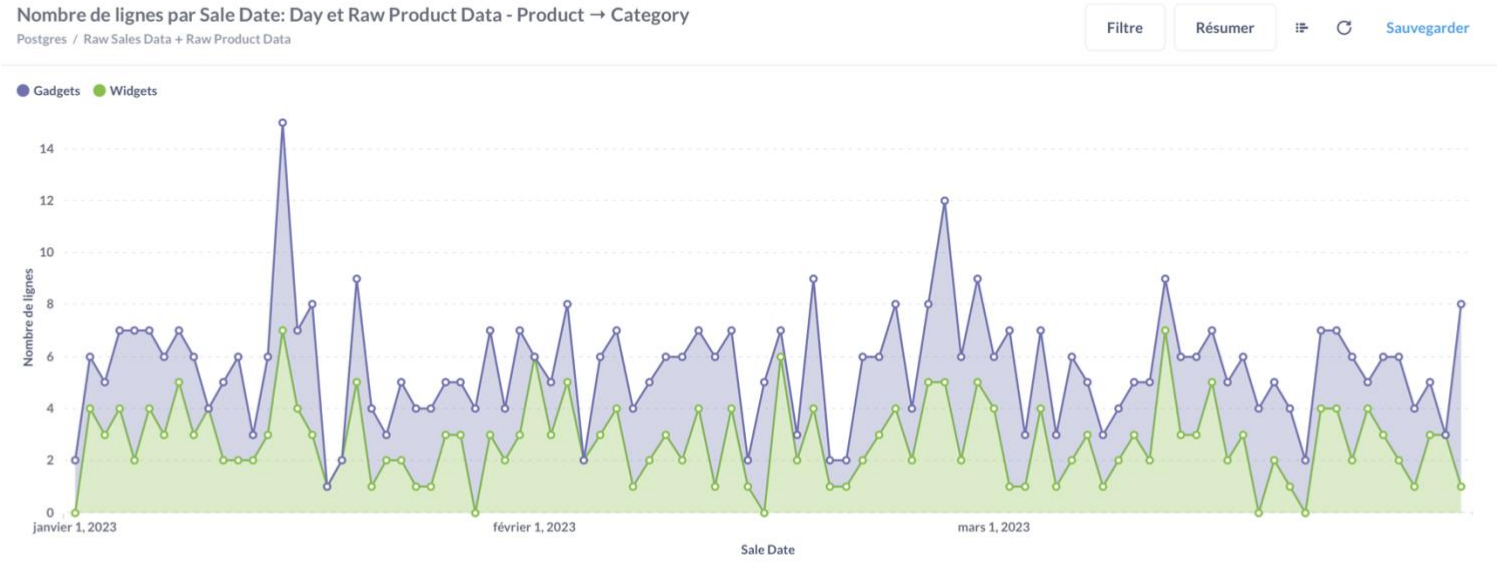
Nombre de lignes par Raw Product Data - Product → Category et Sale Date: Day

Postgres / Raw Sales Data + Raw Product Data

Vue Tout le temps par Jour

Figure 4: Join on product_ID avec date en abscisses et quantity (de sale) en ordonées

On résume par jour pour la date et on group by Category pour avoir les données séparées mais c'est illisible donc on change de visualisation



Nombre de lignes par Sale Date: Day et Raw Product Data - Product → Category

Postgres / Raw Sales Data + Raw Product Data

Figure 5: Visualization by are (surface lines)