Project 1

Business Understanding

Goal:

- Understand customer behaviour.

Project planning.

- Segment clientes.
- Predict if a costumer is going to buy from the company in the near future.

Data Understanding

Data format:

- CSV.

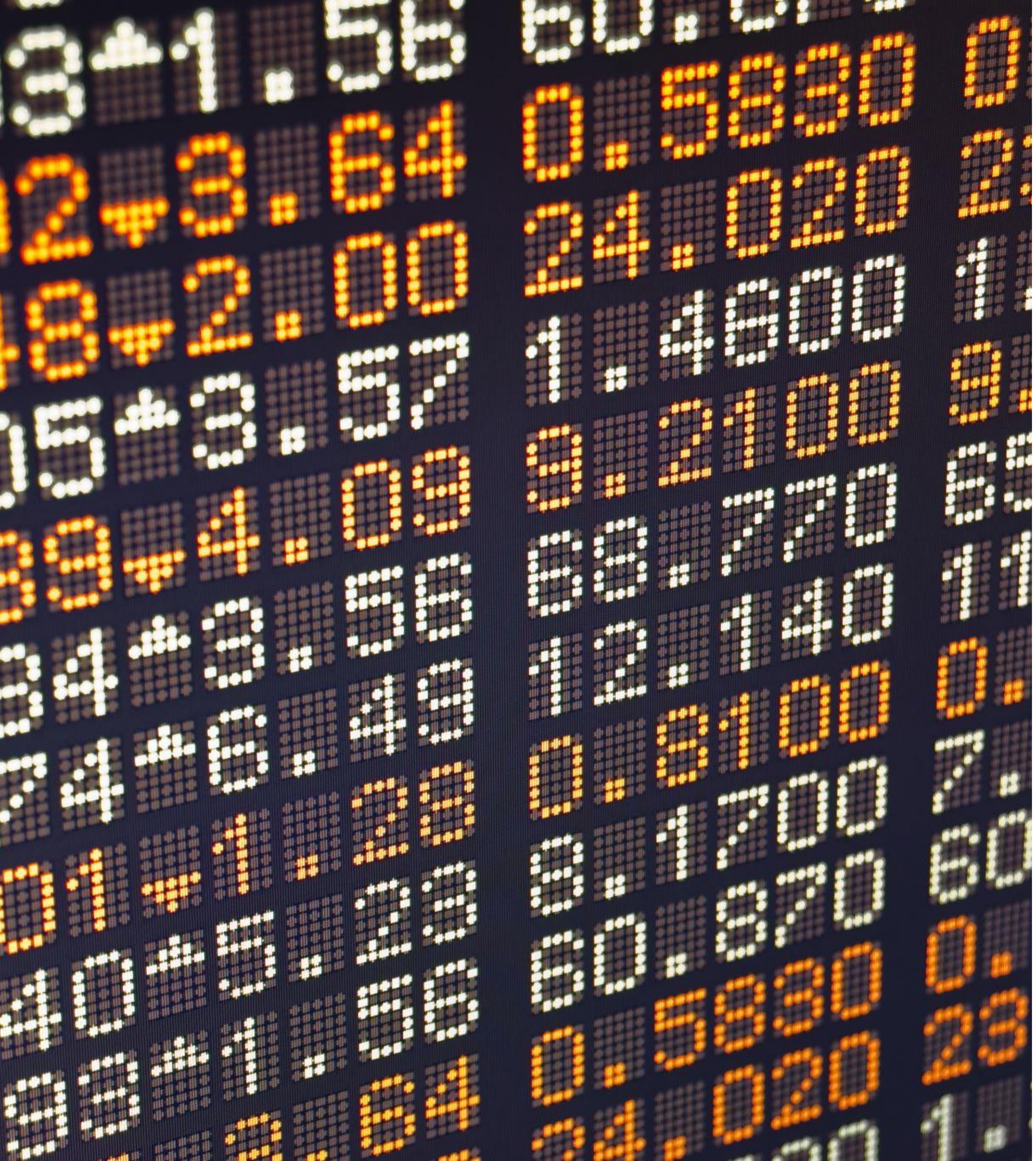
Quantity:

- 3 dimension tables (customer, payments, product).
- 2 fact tables(order item, orders).

Data Understanding.

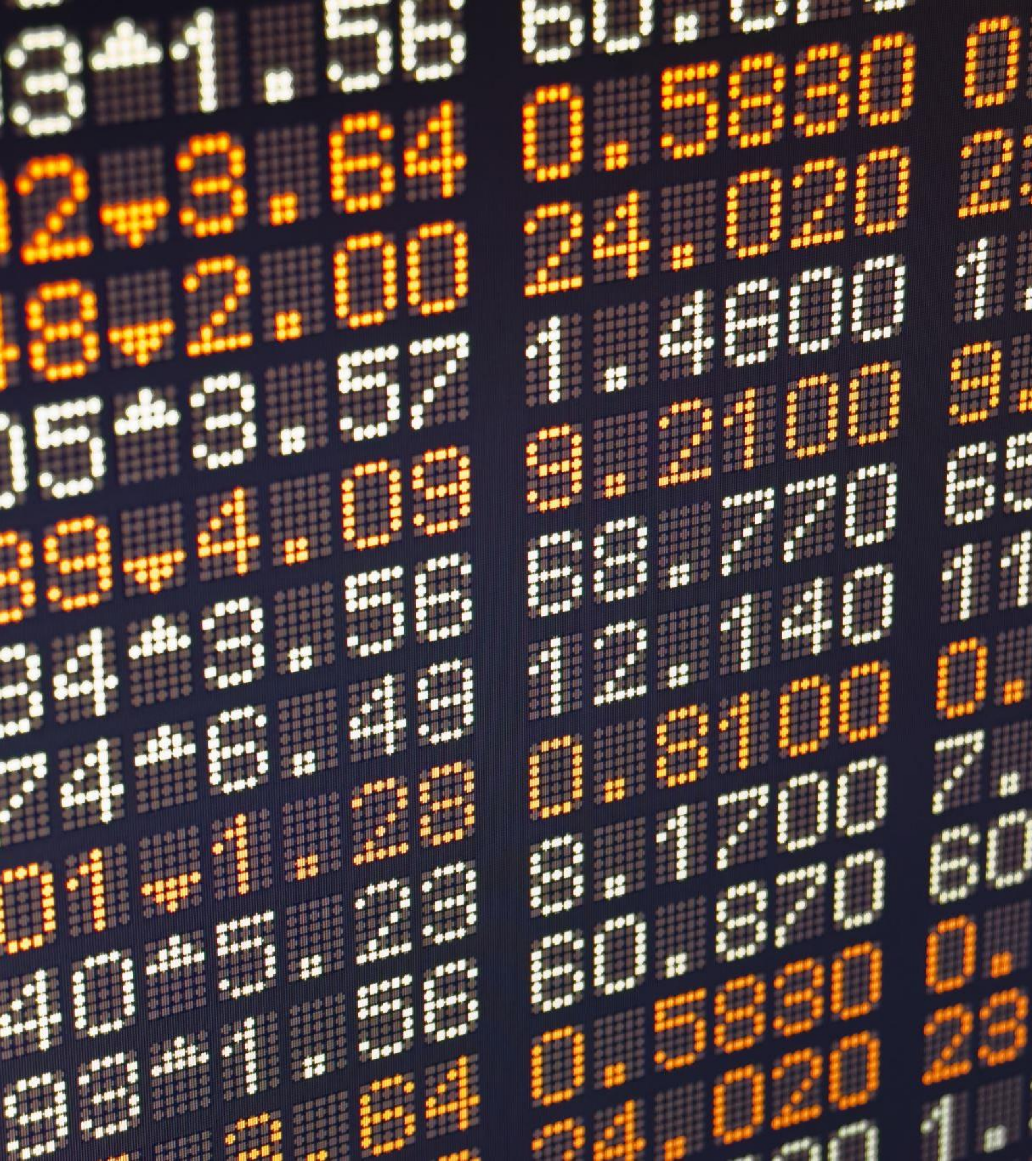
- Data profilling.
- Power BI.

Data Preparation



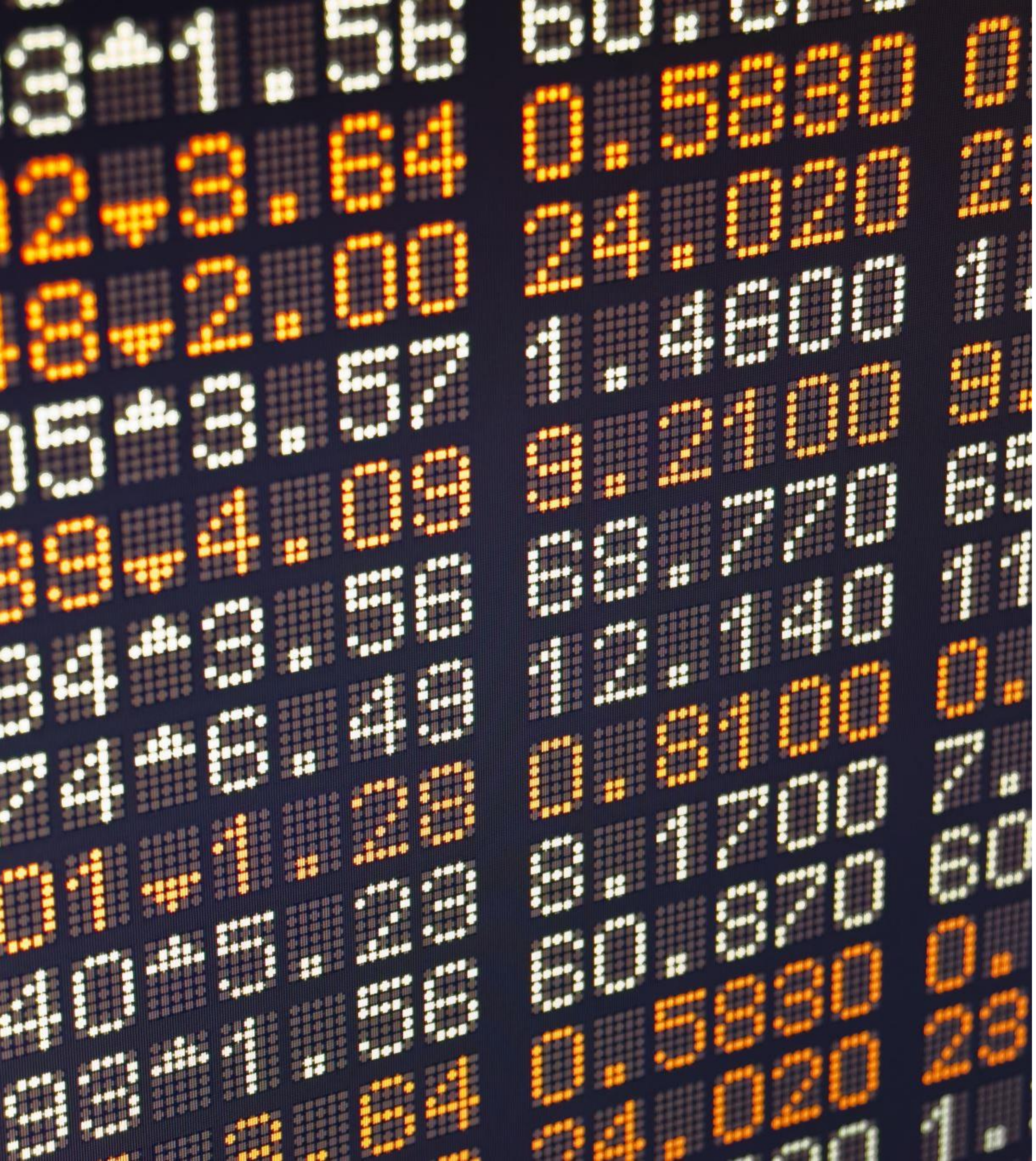
Dim_customer

- Presence of duplicate values
- No null values



Dim_payments

- Presence of duplicate values
- No null values



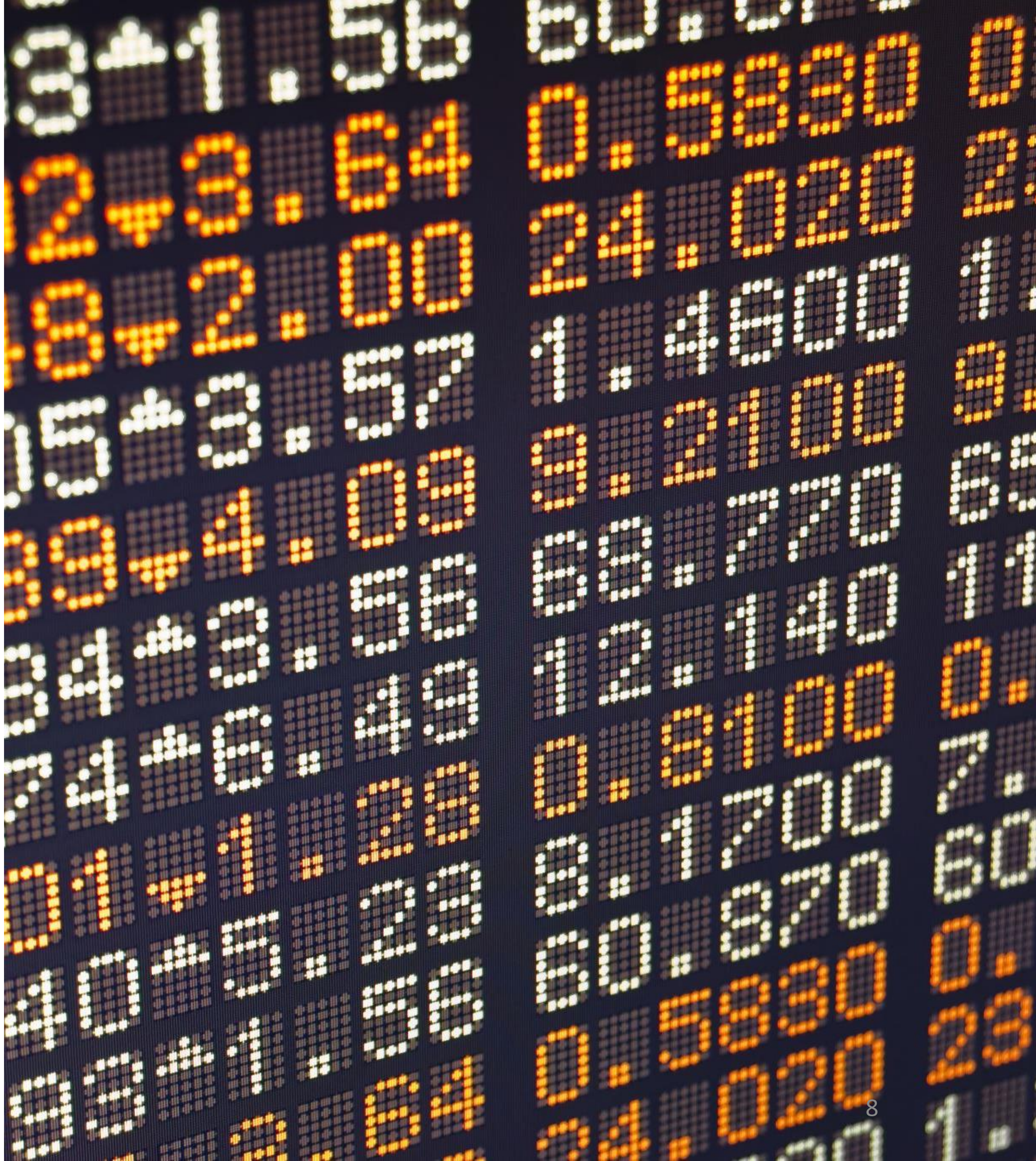
Dim_product

- Presence of duplicate values.
- There are null values.
- Four columns were eliminated do to them not being of importance to the problema at end as well as having in thei majority null values, a database was created in SQL with the 5 csv files provided, after which the following query reveals the drop of the columns mentioned in this paragrafe.
  - **ALTER TABLE dim_product**
  - **DROP COLUMN product_category_name**
  - **DROP COLUMN product_name_lenght**
  - **DROP COLUMN product_description_lenght**
  - **DROP COLUMN product_photos_qty**
- Subsequently, 8 cells remained with null values, these being few values, the decision was made to replace them with the average values of the respective columns, as demonstrated in the next slides.

Dim_product – product_weight_g

- select avg(product_weight_g)
- from dim_product

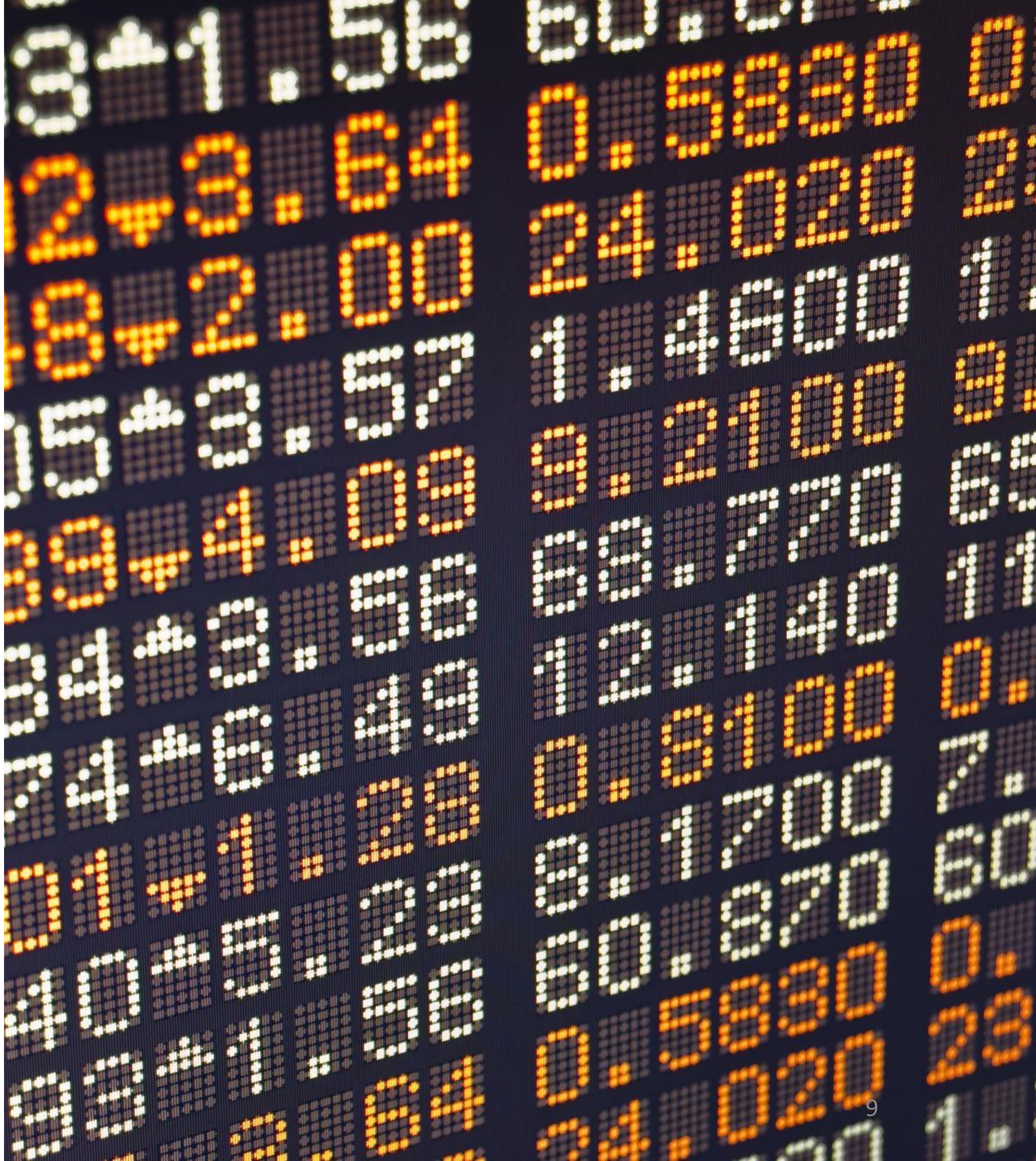
- UPDATE dim_product
- SET product_weight_g = 2276
- WHERE product_weight_g IS NULL;



Dim_product – product_width_cm

- select avg(product_width_cm)
- from dim_product

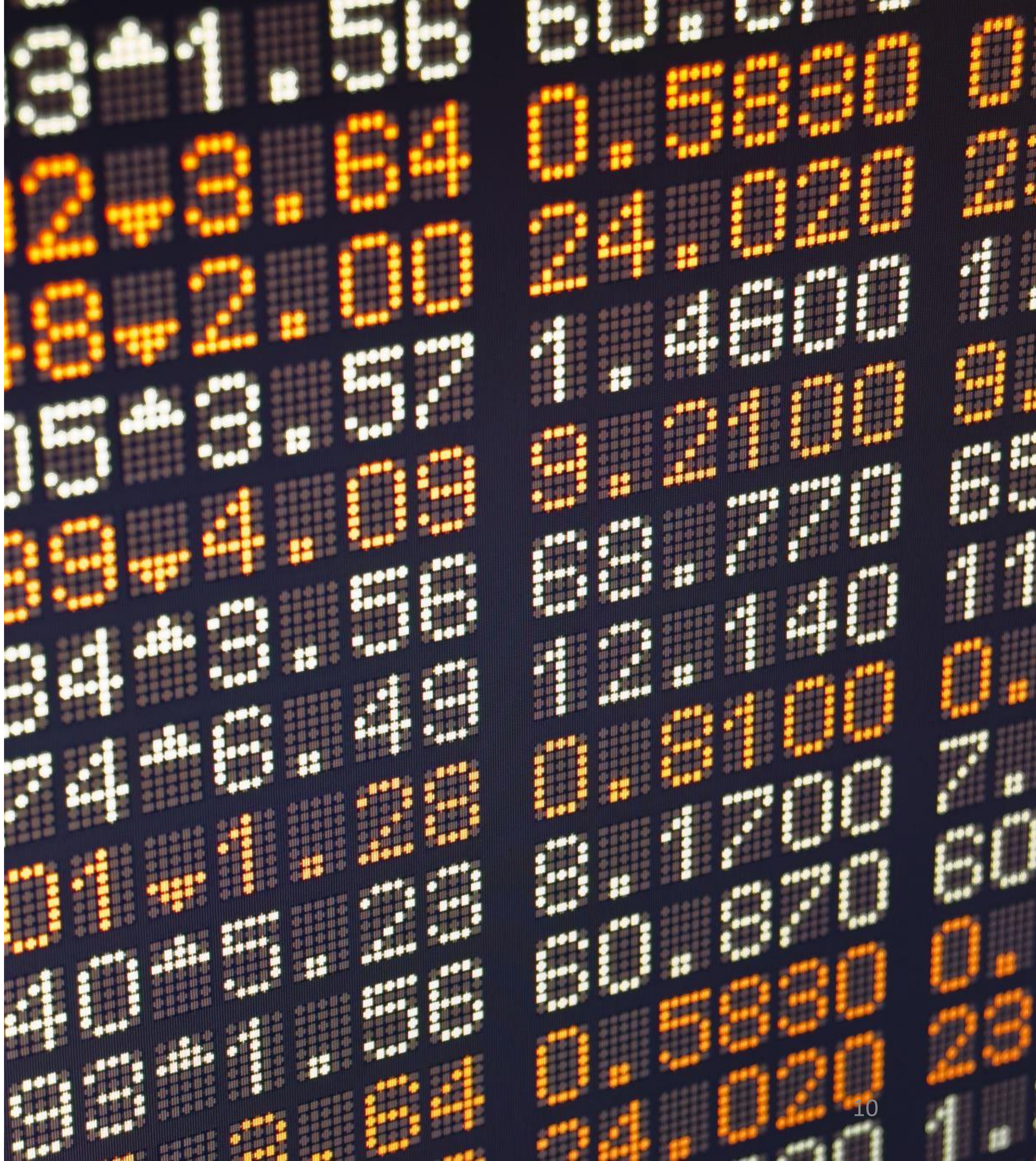
- UPDATE dim_product
- SET product_width_cm = 23
- WHERE product_width_cm IS NULL;



Dim_product – product_height_cm

- select avg(product_height_cm)
- from dim_product

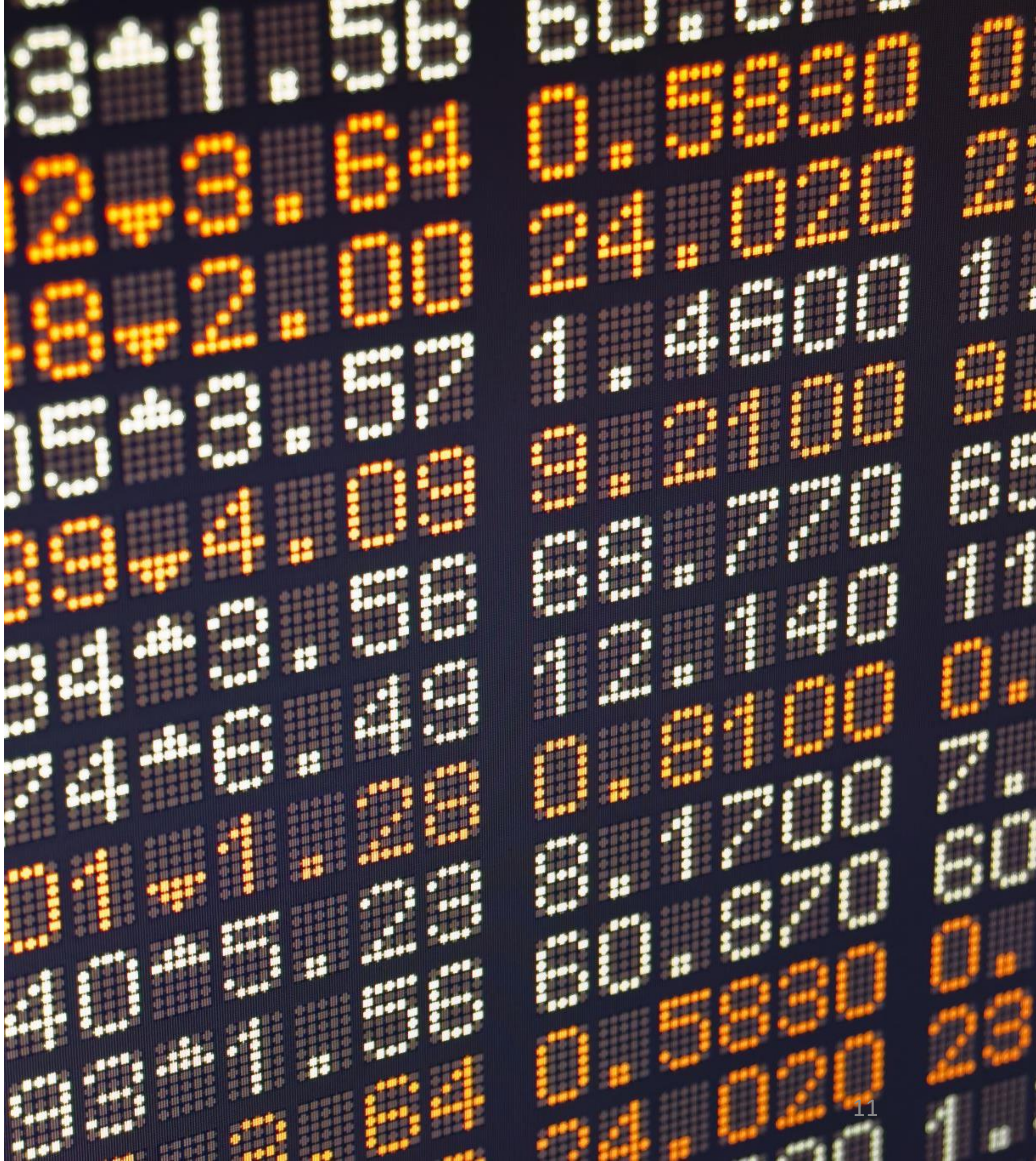
- UPDATE dim_product
- SET product_height_cm = 17
- WHERE product_height_cm IS NULL;

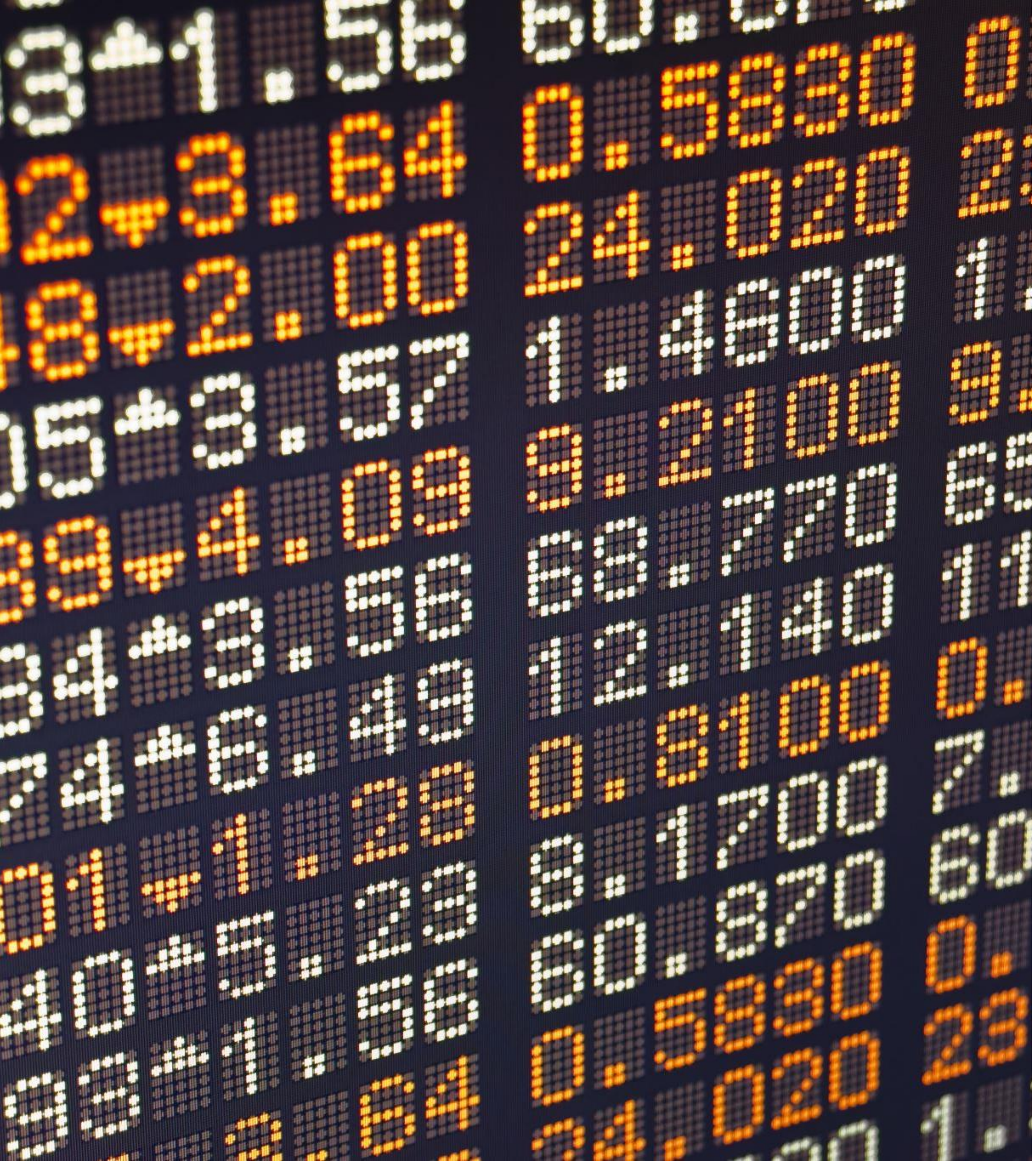


Dim_product – product_length_cm

- select avg(product_length_cm)
- from dim_product

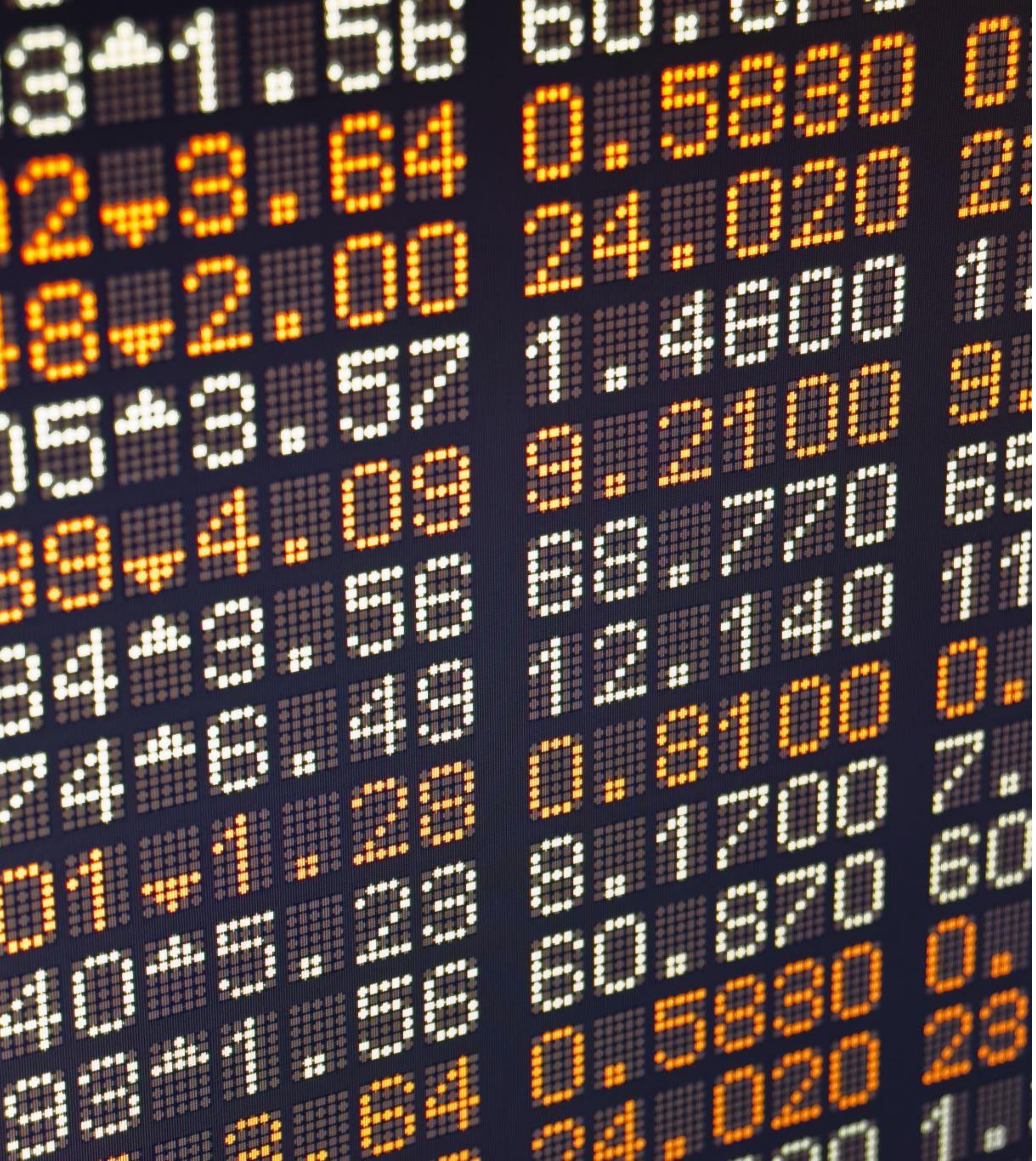
- UPDATE dim_product
- SET product_length_cm = 31
- WHERE product_length_cm IS NULL;





Fact_order_itms

- Presence of duplicate values.
- No null values.



Fact_order

- There were duplicate values.
- Data was filtered according to the problem, with the following SQL query.
- **create table fact_order as**
- **select ***
- **from fact_order**
- **where order_status='delivered'**
- **drop table fct_order**
- **alter table fact_order**
- **rename to fct_order**

Features Table

Features table

```
create table features as
with query1 as(
select order_id,
	sum(price) as total_price,
	sum(freight_value) as total_freight_value,
	count(order_id) as quantity,
	count(distinct product_id) as distinct_products_quantity
from fct_order_itms
group by order_id
),
query2 as(
select order_id,
	max(payment_installments) as nr_payment_installments,
	max(payment_sequential) as nr_payment_sequential
from dim_payments
group by order_id
),
query3 as(
select order_id,
	customer_id,
	order_delivered_customer_date::date as delivered_at
from fct_order
),
query4 as(
select order_id,
	order_delivered_customer_date::date - order_estimated_delivery_date::date as days_delay,
	order_delivered_customer_date::date - order_purchase_timestamp::date as days_to_deliver
from fct_order
)
select q1.order_id,
	q3.customer_id,
	q1.quantity,
	q1.distinct_products_quantity,
	q1.total_price,
	q1.total_freight_value,
	q3.delivered_at,
	q4.days_to_deliver,
	q4.days_delay,
	q2.nr_payment_sequential,
	q2.nr_payment_installments
from query1 q1
join query2 q2 on q1.order_id=q2.order_id
join query3 q3 on q1.order_id=q3.order_id
join query4 q4 on q1.order_id=q4.order_id
```

Features table

```
select f.order_id,
dim.customer_unique_id,
f.quantity,
f.distinct_products_quantity,
f.total_price,
f.total_freight_value,
f.delivered_at,
f.days_to_deliver,
f.days_delay,
f.nr_payment_sequential,
f.nr_payment_installments
from features f
join dim_customer dim on dim.customer_id=f.customer_id
```

Features table

```
CREATE TABLE label (
 customer_unique_id varchar(255),
recency numeric,
frequency INTEGER,
monetary NUMERIC,
avg_quantity NUMERIC,
avg_distinct_products NUMERIC,
avg_price NUMERIC,
avg_freight_value NUMERIC,
R_quartil INTEGER,
F_quartil INTEGER,
M_quartil INTEGER,
score integer,
level varchar(255),
cluster INTEGER,
label varchar(255)
);
```

Features table

```
CREATE TABLE orders (
order_id varchar(255),
customer_unique_id varchar(255),
quantity integer,
distinct_products_quantity integer,
total_price numeric,
total_freight_value numeric,
delivered_at timestamp,
bought_at timestamp,
days_to_deliver integer,
days_delay integer,
nr_payment_sequential integer,
nr_payment_installments integer
);
```

Features table

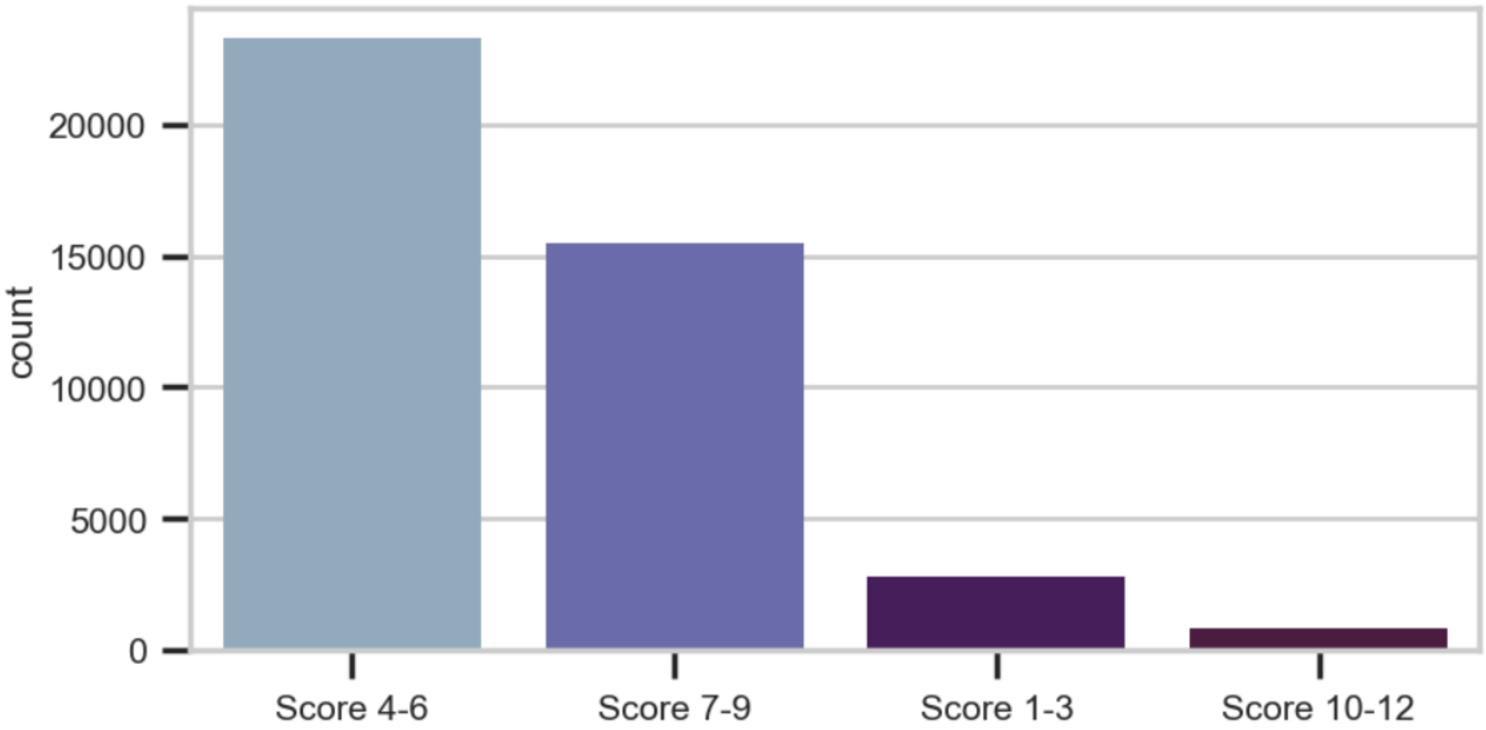
```
with features as(
select at.customer_unique_id,
sum(quantity) as quantity,
sum(total_price) as total,
sum(total_freight_value) as freight_value,
((SELECT COUNT(*) FROM orders at2 WHERE at.customer_unique_id =
at2.customer_unique_id AND at2.days_delay>0) / (count(*))) *100 AS
percentage_delayed_orders ,
case when max(nr_payment_sequential) > 1 then 1 else 0 end
payment_sequential,
Case when max(nr_payment_installments) > 1 then 1 else 0 end
payment_installments,
CASE WHEN max(bought_at) >= '2017-09-01' THEN 1 ELSE 0 END AS
bought_last_three_months,
CASE WHEN max(bought_at) >= '2017-06-01' THEN 1 ELSE 0 END AS
bought_last_six_months
FROM orders at
group by at.customer_unique_id)
```

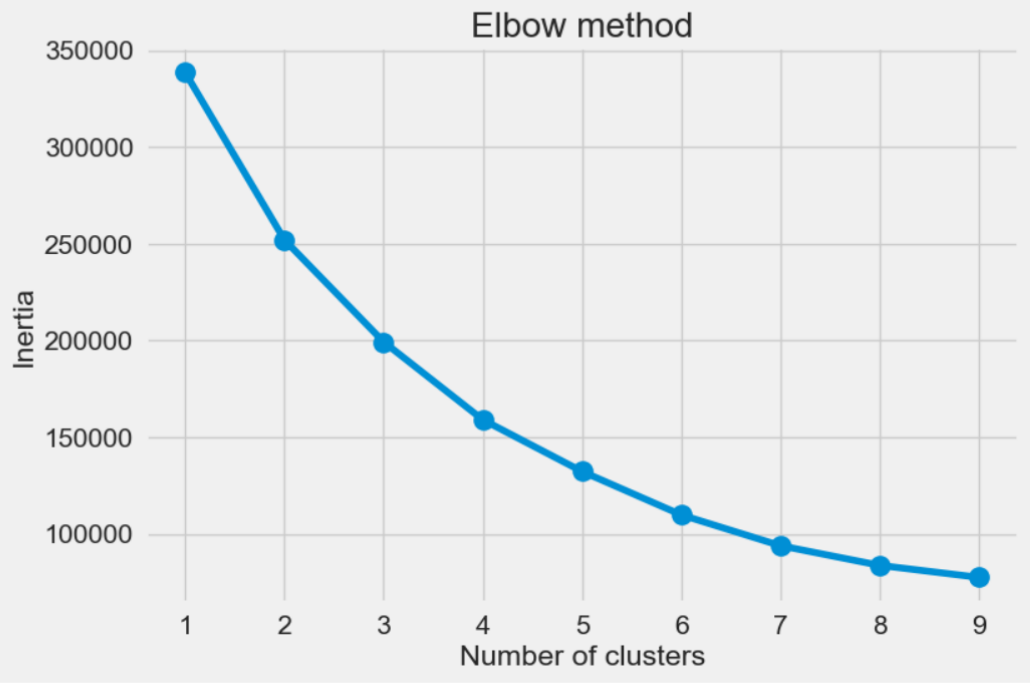
Features table

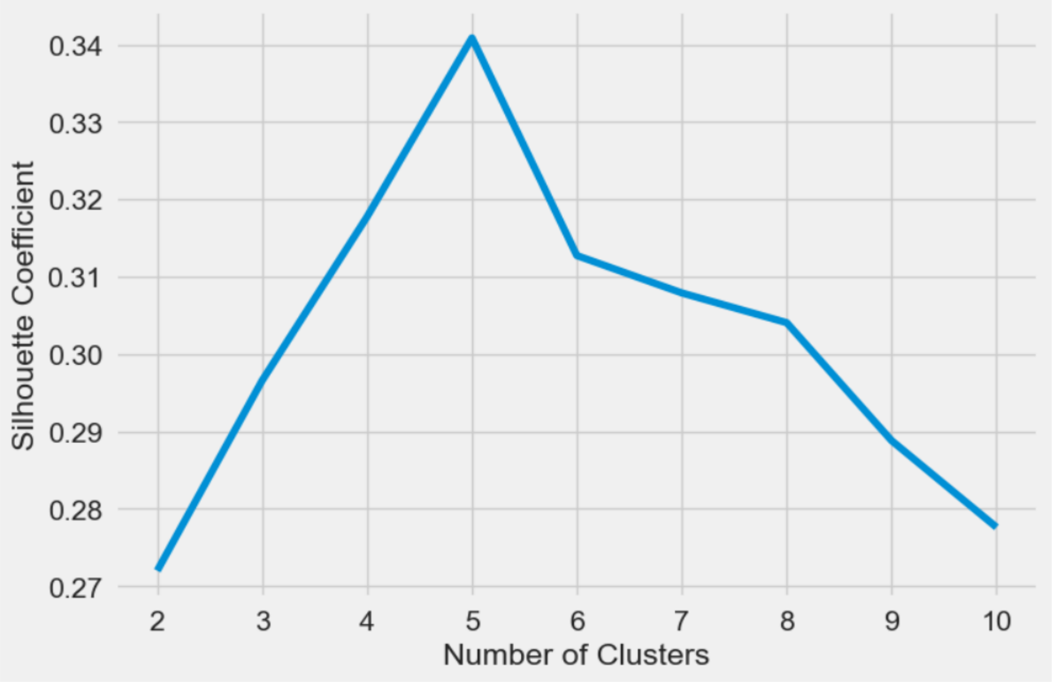
```
Select
l.customer_unique_id,
l.recency,
l.frequency,
l.monetary,
l.R_quartil,
l.F_quartil,
l.M_quartil,
l.score,
l.level,
l.avg_quantity,
f.quantity as total_quantity,
l.avg_distinct_products,
l.avg_price,
f.total as total_price,
l.avg_freight_value,
f.freight_value as total_freight_value,
f.percentage_delayed_orders,
f.payment_sequential,
f.payment_installments,
f.bought_last_three_months,
f.bought_last_six_months,
l.cluster, l.label
from label l
join features f on l.customer_unique_id=f.customer_unique_id
```

Modelling

RFM analysis

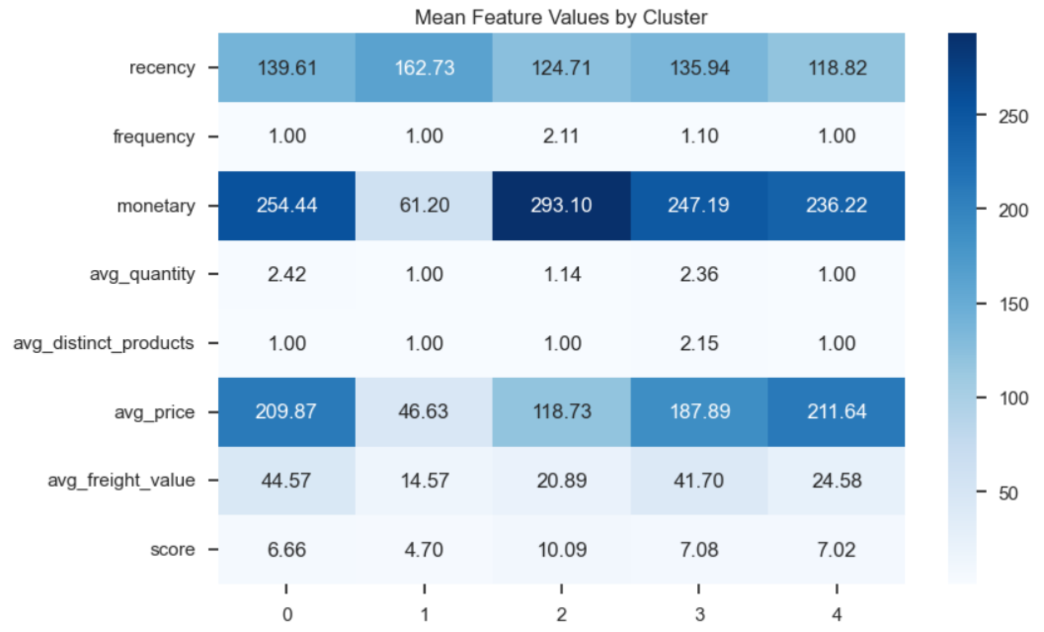


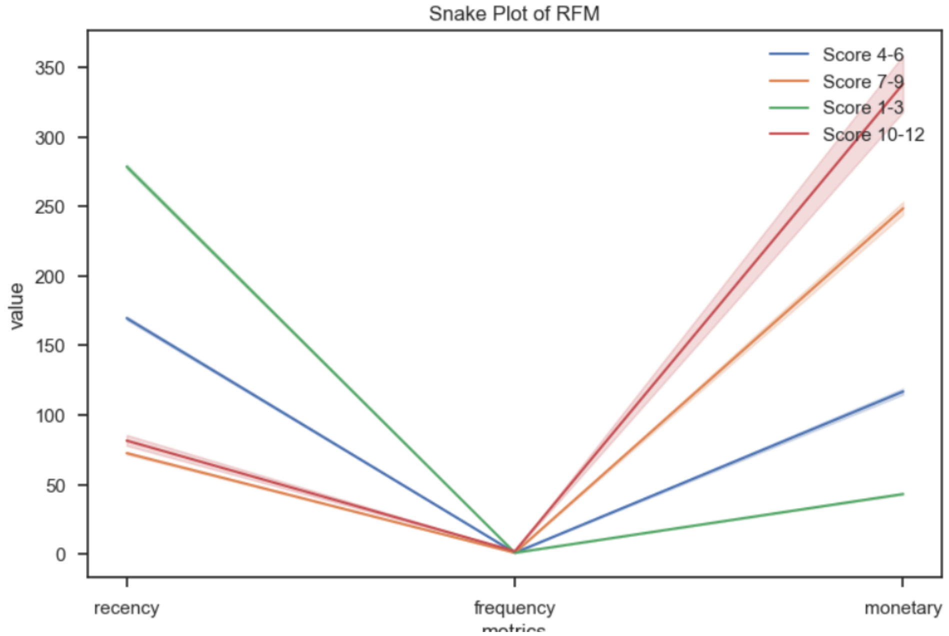




Defining the number of clusters for K-means clustering

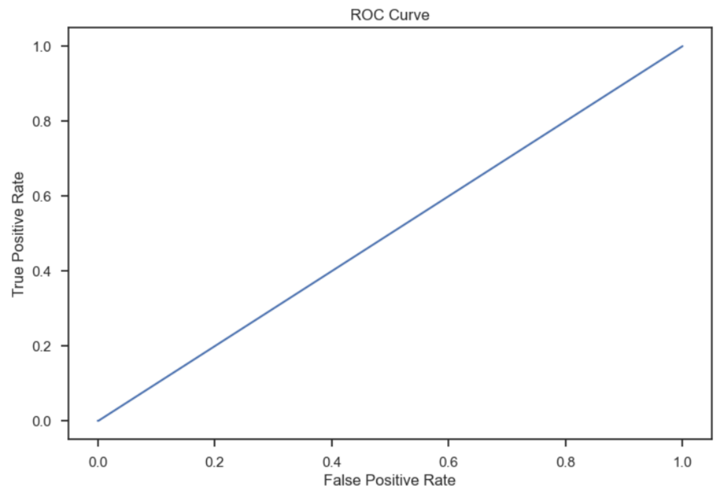
K-means clustering

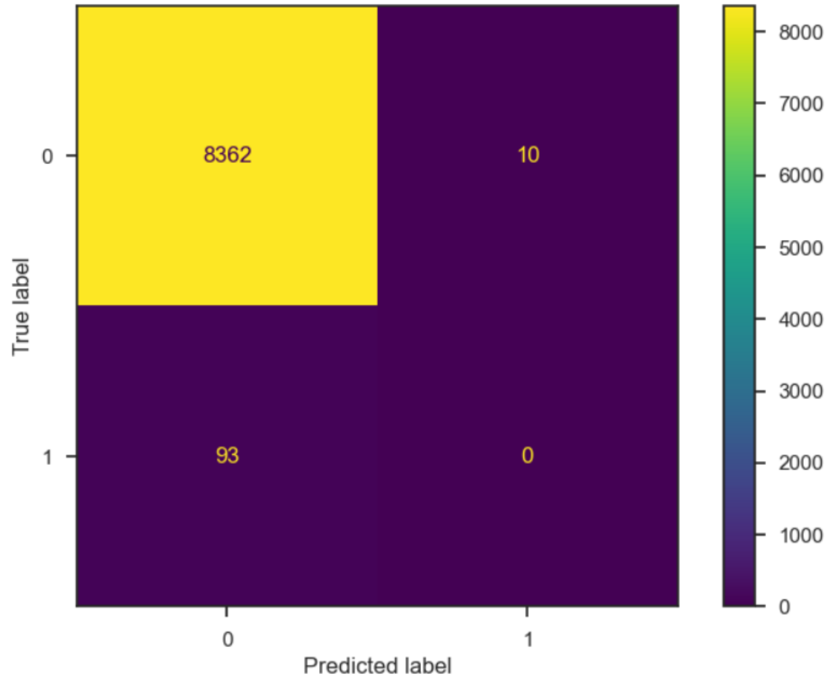




Random Forest

```
Accuracy: 0.9878322504430006
Recall/Sensitivity: 0.0
Specificity: 0.9988055422838031
Precision: 0.0
F1 score: 0.0
```





Conclusion

Several methods were used such as decision trees and random forests in order to predict next purchase. Furthermore these methods were also tune through their hyperparameters and they were executed with several different features at a time through several feature selection occasions.

However the data is too unbalanced to make accurate predictions regarding the costumers behaviour, even with the use of undersampling or oversampling techniques the results driven from the data remain of very low quality.

More data is needed to drive proper conclusions.