# RETAIL ANALYTICS CASE STUDY

***Overview:*** *In the rapidly evolving retail sector, businesses continually seek innovative strategies to stay ahead of the competition, improve customer satisfaction, and optimize operational efficiency. Leveraging data analytics has become a cornerstone for achieving these objectives. This case study focuses on a retail company that has encountered challenges in understanding its* ***sales performance***, ***customer engagement***, *and* ***inventory management****. Through a comprehensive data analysis approach, the company aims to identify* ***high or low sales products***, *effectively* ***segment*** *its customer base, and analyze* ***customer behavior*** *to enhance marketing strategies, inventory decisions, and overall customer experience.*

***Business Problem:***

*The retail company has observed stagnant growth and declining customer engagement metrics over the past quarters. Initial assessments indicate potential issues in product performance variability, ineffective customer segmentation, and lack of insights into customer purchasing behavior. The company seeks to leverage its sales transaction data, customer profiles, and product inventory information to address the following key business problems:*

- ***Product Performance Variability:*** *Identifying which products are performing well in terms of sales and which are not. This insight is crucial for inventory management and marketing focus.*
- ***Customer Segmentation:*** *The company lacks a clear understanding of its customer base segmentation. Effective segmentation is essential for targeted marketing and enhancing customer satisfaction.*
- ***Customer Behaviour Analysis:*** *Understanding patterns in customer behavior, including repeat purchases and loyalty indicators, is critical for tailoring customer engagement strategies and improving retention rates.*

USING **MYSQLWORKBENCH 8.0**

use retail_analytics;

show tables;

| Tables_in_retail_analytics |
|---|
| cte |
| customer |
| product |
| sales |

select * from customer;

| customerid | Age | Gender | Location | JoinDate |
|---|---|---|---|---|
| 1 | 63 | Other | East | 2001-01-20 |
| 2 | 63 | Male | North | 2002-01-20 |
| 3 | 34 | Other | North | 2003-01-20 |
| 4 | 19 | Other | NULL | 2004-01-20 |
| 5 | 57 | Male | North | 2005-01-20 |
| 6 | 22 | Other | South | 2006-01-20 |
| 7 | 56 | Other | East | 2007-01-20 |
| 8 | 65 | Female | East | 2008-01-20 |
| 9 | 33 | Male | West | 2009-01-20 |
| 10 | 34 | Male | East | 2010-01-20 |
| 11 | 44 | Other | North | 2011-01-20 |
| 12 | 24 | Other | East | 2013-01-20 |
| 13 | 69 | Male | East | 2014-01-20 |
| 14 | 25 | Male | North | 2015-01-20 |
| 15 | 25 | Male | North | 2016-01-20 |
| 16 | 40 | Other | East | 2017-01-20 |
| 17 | 49 | Female | South | 2018-01-20 |
| 18 | 60 | Female | North | 2019-01-20 |

select * from product;

| ProductID | ProductName | Category | StockLevel | Price |
|---|---|---|---|---|
| 1 | Product_1 | Clothing | 22 | 46 |
| 2 | Product_2 | Home & Kitchen | 140 | 82 |
| 3 | Product_3 | Home & Kitchen | 473 | 79 |
| 4 | Product_4 | Clothing | 386 | 22 |
| 5 | Product_5 | Beauty & Health | 284 | 18 |
| 6 | Product_6 | Home & Kitchen | 449 | 92 |
| 7 | Product_7 | Home & Kitchen | 319 | 58 |
| 8 | Product_8 | Home & Kitchen | 155 | 87 |
| 9 | Product_9 | Clothing | 470 | 15 |
| 10 | Product_10 | Electronics | 419 | 57 |
| 11 | Product_11 | Electronics | 112 | 59 |
| 12 | Product_12 | Electronics | 389 | 87 |
| 13 | Product_13 | Electronics | 138 | 19 |
| 14 | Product_14 | Electronics | 421 | 32 |
| 15 | Product_15 | Home & Kitchen | 373 | 47 |

select * from sales;

| TransactionID | CustomerID | ProductID | QuantityPurchased | TransactionDate | Price |
|---|---|---|---|---|---|
| 138 | 129 | 100 | 1 | 06/01/23 | 68.85 |
| 1 | 103 | 120 | 3 | 01/01/23 | 30.43 |
| 2 | 436 | 126 | 1 | 01/01/23 | 15.19 |
| 139 | 648 | 31 | 3 | 06/01/23 | 51.20 |
| 274 | 510 | 25 | 3 | 12/01/23 | 90.34 |
| 275 | 807 | 138 | 1 | 12/01/23 | 34.54 |
| 3 | 861 | 55 | 3 | 01/01/23 | 67.76 |
| 140 | 472 | 130 | 4 | 06/01/23 | 57.27 |
| 4 | 271 | 27 | 2 | 01/01/23 | 65.77 |
| 5 | 107 | 118 | 1 | 01/01/23 | 14.55 |
| 6 | 72 | 53 | 1 | 01/01/23 | 26.27 |
| 276 | 386 | 45 | 3 | 12/01/23 | 34.44 |
| 141 | 63 | 16 | 3 | 06/01/23 | 87.93 |
| 7 | 701 | 39 | 2 | 01/01/23 | 95.92 |
| 142 | 139 | 137 | 3 | 06/01/23 | 45.43 |
| 277 | 387 | 83 | 2 | 12/01/23 | 90.16 |
| 143 | 499 | 73 | 3 | 06/01/23 | 68.55 |
| 8 | 21 | 65 | 4 | 01/01/23 | 17.19 |

/*

**firstly, we will do the data cleaning process**

**-- null**

**-- blank('')**

**-- since each tables contains primary key hence, no**

**need to find duplications**

*/

**-- first we run**

```
select * from customer
where age is null or age=''
        or gender is null or gender =''
   or location is null or location = ''
   or joindate is null ;
```

| Customerid | Age | Gender | Location | JoinDate |
|---|---|---|---|---|
| 4 | 19 | Other | | 04/01/20 |
| 113 | 21 | Male | | 02/05/20 |
| 115 | 43 | Female | | 05/05/20 |
| 219 | 23 | Male | | 27/08/20 |
| 239 | 40 | Other | | 18/09/20 |
| 322 | 35 | Female | | 18/12/20 |
| 379 | 27 | Female | | 18/02/21 |
| 405 | 26 | Female | | 19/03/21 |
| 448 | 44 | Female | | 05/05/21 |
| 476 | 45 | Female | | 05/06/21 |
| 517 | 62 | Other | | 20/07/21 |
| 668 | 131 | Male | | 01/01/22 |
| 998 | 22 | Other | | 29/12/22 |

**-- Now to resolve this we use update statement**

```
update customer
set location = 'na'
where location ='';
```

**-- or**

```
update customer
set location= replace(location,'na','NULL');
```

**-- now we will calculate the outliers**

/*

**for this we have two ways**

**1) turkey law**

**2) using z-score**

*/

```
select * from customer;
```

**-- using turkey law**

```
set @q1=
(with cte as
(select *,
                row_number() over(order by age) as asc_row
        from customer)

select avg(age) as quartile_1 from cte
where asc_row in ( select floor(count(age)/4) from customer
                                        union
                                select ceil(count(age)/4) from customer));

set @q3=
(with cte as
(select *,
                row_number() over(order by age) as asc_row
        from customer)

select avg(age) as quartile_3 from cte
where asc_row in ( select floor(count(age)*3/4) from customer
                                        union
                                select ceil(count(age)*3/4) from customer));

set @iqr=@q3-@q1;
```

**-- the final formula to find the outliers**

```
select * from customer
where age< (@q1-1.5*@iqr) or age>(@q3+1.5*@iqr);
```

| | Customerid | Age | Gender | Location | JoinDate |
|---|---|---|---|---|---|
| ▶ | 668 | 131 | Male | | 01/01/22 |

**-- now using z-score**

```
/*
```

**z i.e outlier**

**z= x- mean/ standard deviation**

```
*/
select * from customer
where age>((select avg(age)+3*stddev(age)from customer))
        or
        age<((select avg(age)-3*stddev(age) from customer));
```

| | Customerid | Age | Gender | Location | JoinDate |
|---|---|---|---|---|---|
| ▶ | 668 | 131 | Male | | 01/01/22 |

**-- now we get to know that customerid 668 having age 131 is an outlier which is most likely to be a age entry error**

**-- so we delete the row where age is 131**

```
delete from customer
where age=131;
```

**-- now due to this there's a  break in sequence of customerid which is normal and no need to resequence it since**

**-- if its matches with any other tables then wrong matching is done which result to wrong analysis.Hence, leave it as it is**

**-- with gap**

```
select transactionid from sales
group by transactionid
having count(transactionid)>1;
```

| transactionid |
| --- |
| 4999 |
| 5000 |

**-- now after running the above query we get to know that the sales data contains some dublicate values too**

**-- so in order to remove the dublicates and remain the first appeared transactionid only we will do the following process**

```
select * from sales;

alter table sales
add column id int auto_increment primary key first ;
```

**-- since theirs no unique value assigned to rows that why, we add a new column and assigned it as primary key**

**-- we just need to run a delete command to remove the duplicates**

```
delete from sales
where id not in (select id from (select min(id) as id from sales
                                 group  by transactionid) s);
```

**-- now we have deleted all the dublicates leaving the one who appeared first**

**-- so when i got what i wanted then, now i will drop id column**

```
alter table sales
drop id;
```

**-- now after reviewing all the tables, all the tables are not cleaned and ready for analysis**

-- ********** START **********

/* PRODUCT PERFORMANCE VARIABILITY */

/*

**-- To identify which products are performing well in terms on sales and which are not**

**WHY NEED OF THIS INSIGHT?**

**-- For inventory management**

**-- For marketing focus**

*/

**-- now to see the sales or i should say total sales done over each product we need sales table also to specify**

**-- name of the products then in that case we also need product table**

select * from sales;

select * from product;

select productid,sum(price*quantitypurchased) as total_sales

from sales

group by productid;

| productid | total_sales |
|---|---|
| 1 | 1844.40 |
| 2 | 5222.40 |
| 3 | 3778.56 |
| 4 | 1279.48 |
| 5 | 844.59 |
| 6 | 4678.23 |
| 7 | 3550.20 |
| 8 | 5668.00 |
| 9 | 913.80 |
| 10 | 4533.81 |
| 11 | 3161.70 |
| 12 | 5335.06 |
| 13 | 1596.30 |

**-- now by this i get all the productid's along with their total_sales done**

```
/*
but here i need the products which are performing well in the market
hence, i will find the top 10% and least 10% of the products which results to highest/lowest total_sales
and also rank them on the basic of highest to lowerest*/

with product_sales as
(select productid,sum(price*quantitypurchased) as total_sales
from sales
group by productid)
,
rankings as
(select
        dense_rank() over (order by total_sales desc) as RANKINGS,
   productid,
   total_sales
from product_sales)

select r.RANKINGS,
                r.Productid,
                p.ProductName,
      r.Total_sales from rankings r
join product p
on r.productid=p.productid
where r.RANKINGS <= (select round(count(productid)*0.10) from product);  -- total products * 10%
```

| RANKINGS | Productid | ProductName | Total_sales |
|---|---|---|---|
| 1 | 51 | Product_51 | 512160.00 |
| 2 | 17 | Product_17 | 9450.00 |
| 3 | 87 | Product_87 | 7817.24 |
| 4 | 179 | Product_179 | 7388.26 |
| 5 | 96 | Product_96 | 7132.32 |
| 6 | 54 | Product_54 | 7052.86 |
| 7 | 187 | Product_187 | 6915.88 |
| 8 | 156 | Product_156 | 6827.84 |
| 9 | 57 | Product_57 | 6622.20 |
| 10 | 200 | Product_200 | 6479.79 |
| 11 | 127 | Product_127 | 6415.80 |
| 12 | 28 | Product_28 | 6386.64 |
| 13 | 106 | Product_106 | 6262.83 |
| 14 | 104 | Product_104 | 6230.16 |
| 15 | 195 | Product_195 | 6229.20 |
| 16 | 103 | Product_103 | 6191.46 |
| 17 | 85 | Product_85 | 6188.22 |
| 18 | 190 | Product_190 | 6126.12 |

**/* NOW THESE ARE THE TOP PERFORMING PRODUCTS (TOP 10%) */**

**-- LEAST PERFROMING(BELOW IS THE QUERY)**

```
with product_sales as
(select productid,sum(price*quantitypurchased) as total_sales
from sales
group by productid)
,
rankings as
(select
        dense_rank() over (order by total_sales) as RANKINGS,
   productid,
   total_sales
from product_sales)

select r.RANKINGS,
                    r.Productid,
                    p.ProductName,
      r.Total_sales from rankings r
join product p
on r.productid=p.productid
where r.RANKINGS <= (select round(count(productid)*0.10) from product);  -- total products
* 10%
```

| RANKINGS | Productid | ProductName | Total_sales |
|---|---|---|---|
| 1 | 139 | Product_139 | 484.10 |
| 2 | 161 | Product_161 | 547.50 |
| 3 | 159 | Product_159 | 609.70 |
| 4 | 20 | Product_20 | 610.74 |
| 5 | 178 | Product_178 | 616.55 |
| 6 | 66 | Product_66 | 623.10 |
| 7 | 35 | Product_35 | 658.92 |
| 8 | 21 | Product_21 | 700.21 |
| 9 | 109 | Product_109 | 733.92 |
| 10 | 164 | Product_164 | 756.28 |
| 11 | 46 | Product_46 | 779.40 |
| 12 | 157 | Product_157 | 808.08 |
| 13 | 132 | Product_132 | 817.74 |
| 14 | 112 | Product_112 | 836.50 |
| 15 | 105 | Product_105 | 836.68 |
| 16 | 146 | Product_146 | 842.84 |
| 17 | 5 | Product_5 | 844.59 |

**/* NOW THESE ARE THE LEAST PERFORMING PRODUCTS (LEAST 10%) */**

-- ********** END**********

/* CUSTOMER SEGMENTATION */

/*

**-- Demmographic- age,gender**

**-- geographic- location**

**-- behavioral- purchase behaviour,product usage,brand loyalty**

*/

**-- WE WILL SEE HOW MANY CUSTOMERS ARE CHILDREN/TEENAGERS, MIDDLE AGED, OLD**

**-- S0,THE QUERY WE WILL USE IS**

```
SELECT * FROM CUSTOMER;

with AGE_DEMOGRAPHIC AS

(SELECT

        case when age<=18 then '0-18'

                when age between 19 and 40 then '18-45'

    else '45-Above' end

        as age_demographic

from customer)


select age_demographic,count(age_demographic) as no_of_customer,

                concat(round(count(age_demographic)*100/(select count(*) from
customer),2),'%') as percentage_of_customers

from age_demographic

group by age_demographic

order by count(age_demographic) desc;
```

| | age_demographic | no_of_customer | percentage_of_customers |
|---|---|---|---|
| ▶ | 45-Above | 564 | 56.46% |
| | 18-45 | 416 | 41.64% |
| | 0-18 | 19 | 1.90% |

-- **now we will see how many customers are male and female**

```
select gender,count(customerid) as no_of_customer ,
                concat(round(count(customerid)*100/(select count(*) from customer),2),'%')
as percentage_of_customers
from customer
group by gender
order by count(customerid) desc ;
```

| | gender | no_of_customer | percentage_of_customers |
|---|---|---|---|
| ▶ | Other | 356 | 35.64% |
| | Male | 327 | 32.73% |
| | Female | 316 | 31.63% |

-- **we will analyse the geographic segmentation**

```
select location,count(customerid) as no_of_customer ,
                    concat(round(count(customerid)*100/(select count(*) from customer),2),'%')
as percentage_of_customers
from customer
group by location
order by count(customerid) desc;
```

| location | no_of_customer | percentage_of_customers |
|---|---|---|
| West | 272 | 27.23% |
| North | 248 | 24.82% |
| South | 242 | 24.22% |
| East | 225 | 22.52% |
| NULL | 12 | 1.20% |

-- **some users haven't specify their location and thats just 1.2% of the overall data**

-- ********** END**********

/* **CUSTOMER BEHAVIOUR** */

-- **customers who didnt purchased any product**

```
select customerid from customer c
where not exists
(select 1 from sales s where c.customerid=s.customerid);
```

| customerid |
|---|
| 52 |
| 71 |
| 197 |
| 299 |
| 433 |
| 600 |
| 671 |
| 694 |
| 756 |
| 765 |
| 808 |
| NULL |

-- **lets calculate how many products are there in each category**

```
select category,count(productname) as total_products from product
group by category;
```

| category | total_products |
|---|---|
| Clothing | 45 |
| Home & Kitchen | 58 |
| Beauty & Health | 50 |
| Electronics | 47 |

**-- ONE-TIME-BUYERS**

**-- customers who purchased only once**

```
select s.customerid,
                    min(TransactionDate) as date_purchased,
      sum(s.price*s.QuantityPurchased) as Order_Value
from customer c
join sales s
on s.customerid=c.customerid
group by s.customerid
having count(s.transactionid)=1
order by s.customerid;
```

| customerid | date_purchased | Order_Value |
|---|---|---|
| 6 | 25/02/23 | 80.70 |
| 24 | 20/04/23 | 90.80 |
| 45 | 27/07/23 | 241.35 |
| 94 | 28/01/23 | 360.64 |
| 110 | 18/05/23 | 236.16 |
| 150 | 24/01/23 | 43.65 |
| 169 | 29/05/23 | 230.37 |
| 181 | 21/01/23 | 298.23 |
| 185 | 27/04/23 | 69.84 |
| 189 | 27/01/23 | 93.18 |
| 212 | 22/02/23 | 203.97 |
| 219 | 27/06/23 | 22.24 |
| 240 | 08/07/23 | 92.83 |
| 255 | 01/07/23 | 14.29 |
| 315 | 23/02/23 | 53.91 |
| 317 | 16/04/23 | 257.73 |
| 333 | 07/07/23 | 189.00 |
| 355 | 26/07/23 | 61.64 |

-- **now to find count of customers**

```
select count(customerid) as total_one_time_buyers
from
(select customerid from sales
group by customerid
having count(transactionid)=1) t;
```

| total_one_time_buyers |
| --- |
| 39 |

-- **REPEAT-USERS**

```
select count(customerid) as total_repeat_users
from
(select customerid from sales
group by customerid
having count(transactionid)>1) t;
```

| total_repeat_users |
| --- |
| 950 |

-- **SO, HERE MORE THAN 95%  OF THE CUSTOMERS COME BACK TO MAKE ANOTHER PURCHASE**

-- **ACTIVE CUSTOMERS**

**(who does more than 5 purchases over the last 3 months)**

```
select customerid,
        sum(price*QuantityPurchased) as total_purchase,
   max(transactiondate) as latest_purchase_date,
   count(transactionid) as frequency_of_purchase
from sales
where str_to_date(TransactionDate,'%d/%m/%y') between
  date_sub((select max(str_to_date(TransactionDate,'%d/%m/%y')) from sales), interval 3 month)
                and (select max(str_to_date(TransactionDate,'%d/%m/%y')) from sales)
-- last 3 months
group by customerid
having count(transactionid)>5;
```

| customerid | total_purchase | latest_purchase_date | frequency_of_purchase |
|---|---|---|---|
| 124 | 1304.42 | 30/05/23 | 6 |
| 183 | 420.29 | 30/05/23 | 6 |
| 188 | 918.96 | 27/07/23 | 7 |
| 220 | 794.17 | 29/04/23 | 6 |
| 260 | 826.58 | 26/07/23 | 6 |
| 277 | 690.63 | 27/05/23 | 6 |
| 302 | 806.60 | 31/05/23 | 6 |
| 399 | 1072.21 | 27/05/23 | 6 |
| 483 | 1173.15 | 27/05/23 | 7 |
| 494 | 1027.23 | 30/04/23 | 7 |
| 517 | 1139.22 | 28/04/23 | 7 |
| 554 | 958.64 | 24/05/23 | 7 |
| 559 | 894.27 | 20/07/23 | 6 |
| 598 | 734.79 | 22/05/23 | 6 |
| 602 | 1173.77 | 29/04/23 | 6 |
| 648 | 998.51 | 28/05/23 | 7 |
| 664 | 1361.31 | 27/05/23 | 7 |

-- **WINDOW /RARE CUSTOMERS**

```
with t as
(select customerid,
                transactiondate,
    lag(str_to_date(transactiondate,'%d/%m/%y')) over(partition by customerid order by str_to_date(transactiondate,'%d/%m/%y')) as last_order,
    datediff(
                        str_to_date(transactiondate,'%d/%m/%y'),
                        lag(str_to_date(transactiondate,'%d/%m/%y')) over(partition by customerid order by str_to_date(transactiondate,'%d/%m/%y')))
    as days_gap
from sales)

select customerid,
                round(avg(days_gap)) as avg_days_gap
from t
where last_order and days_gap is not null
group by customerid
having round(avg(days_gap))> (select avg(days_gap) from t)
order by avg_days_gap desc;
```

| customerid | avg_days_gap |
|---|---|
| 123 | 195 |
| 818 | 174 |
| 444 | 171 |
| 91 | 164 |
| 841 | 161 |
| 636 | 155 |
| 668 | 154 |
| 674 | 153 |
| 322 | 152 |
| 269 | 148 |
| 791 | 144 |
| 880 | 134 |
| 652 | 133 |
| 74 | 126 |
| 790 | 123 |
| 413 | 122 |
| 241 | 117 |

-- **PREMIUM/VIP CUSTOMERS**

```
with t as
(select customerid,
        sum(price*quantitypurchased) as total_purchase_amount,
        count(transactionid) as
freq_of_purchase,max(str_to_date(transactiondate,'%d/%m/%y')) as latest_purchase_date
from sales
where str_to_date(transactiondate,'%d/%m/%y') between
                date_sub((select max(str_to_date(transactiondate,'%d/%m/%y')) from sales),
interval 30 day) and
     (select max(str_to_date(transactiondate,'%d/%m/%y')) from sales)
group by customerid),

t2 as
(select
        ntile(10) over(order by total_purchase_amount desc) as amount_desc,
   ntile(10) over(order by freq_of_purchase desc) as freq_desc,
   t.*
from t)

select customerid,total_purchase_amount,latest_purchase_date,freq_of_purchase from t2
where amount_desc=1 and freq_desc =1;
```

| customerid | total_purchase_amount | latest_purchase_date | freq_of_purchase |
|---|---|---|---|
| 881 | 685.26 | 2023-07-23 | 5 |
| 659 | 443.95 | 2023-07-20 | 5 |
| 127 | 673.91 | 2023-07-26 | 4 |
| 188 | 524.64 | 2023-07-27 | 4 |
| 549 | 522.35 | 2023-07-25 | 4 |
| 819 | 37650.61 | 2023-07-10 | 3 |
| 936 | 1060.84 | 2023-07-27 | 3 |
| 988 | 730.55 | 2023-07-14 | 3 |
| 558 | 711.26 | 2023-07-21 | 3 |
| 201 | 698.64 | 2023-07-25 | 3 |
| 35 | 693.16 | 2023-07-16 | 3 |
| 937 | 686.72 | 2023-07-21 | 3 |
| 947 | 640.42 | 2023-07-22 | 3 |
| 547 | 611.40 | 2023-07-26 | 3 |
| 563 | 586.57 | 2023-07-18 | 3 |
| 332 | 562.06 | 2023-07-09 | 3 |
| 584 | 536.06 | 2023-07-26 | 3 |
| 75 | 530.00 | 2023-07-26 | 3 |

```
-- CHURN-RISK CUSTOMERS
WITH last_purchase AS (
   SELECT customerid,
        MAX(STR_TO_DATE(transactiondate, '%d/%m/%y')) AS last_purchase_date,
        COUNT(transactionid) AS total_purchases
   FROM sales
   GROUP BY customerid
),

latest_date AS (
   SELECT MAX(STR_TO_DATE(transactiondate, '%d/%m/%y')) AS max_date
   FROM sales
)

SELECT l.customerid,
      l.total_purchases,
      l.last_purchase_date,
      DATEDIFF(ld.max_date, l.last_purchase_date) AS days_since_last_purchase
FROM last_purchase l
CROSS JOIN latest_date ld
WHERE DATEDIFF(ld.max_date, l.last_purchase_date) > 90   -- last purchase older than 90 days
  AND l.total_purchases > 1                              -- previously active customers
ORDER BY days_since_last_purchase DESC;
```

| customerid | total_purchases | last_purchase_date | days_since_last_purchase |
|---|---|---|---|
| 839 | 2 | 2023-01-09 | 200 |
| 135 | 2 | 2023-01-18 | 191 |
| 252 | 2 | 2023-02-06 | 172 |
| 858 | 2 | 2023-02-06 | 172 |
| 415 | 3 | 2023-02-09 | 169 |
| 565 | 3 | 2023-02-13 | 165 |
| 863 | 2 | 2023-02-13 | 165 |
| 961 | 2 | 2023-02-14 | 164 |
| 403 | 3 | 2023-02-18 | 160 |
| 249 | 2 | 2023-02-19 | 159 |
| 699 | 4 | 2023-02-21 | 157 |
| 518 | 3 | 2023-02-23 | 155 |
| 830 | 3 | 2023-02-24 | 154 |
| 898 | 3 | 2023-02-27 | 151 |
| 606 | 2 | 2023-02-28 | 150 |
| 5 | 5 | 2023-03-01 | 149 |
| 503 | 4 | 2023-03-02 | 148 |

-- **products which are purchased most number of times**

```
with T AS
(select
        ntile(10) over(order by count(customerid) desc) as customers_percent,
   productid, count(customerid) as no_of_customers
from sales
group by productid
order by no_of_customers desc)

select productid,no_of_customers from t
where customers_percent=1;
```

| productid | no_of_customers |
|---|---|
| 17 | 39 |
| 182 | 38 |
| 87 | 35 |
| 22 | 35 |
| 13 | 34 |
| 187 | 34 |
| 166 | 33 |
| 156 | 33 |
| 57 | 33 |
| 146 | 33 |
| 54 | 32 |
| 188 | 32 |
| 84 | 32 |
| 134 | 32 |
| 149 | 32 |
| 59 | 32 |
| 58 | 32 |

-- **Find the avg_order_quantity of the customers**

```
select customerid, round(avg(quantitypurchased)) as avg_order_quantity
from sales
group by customerid;
```

| customerid | avg_order_quantity |
|---|---|
| 1 | 3 |
| 2 | 3 |
| 3 | 3 |
| 4 | 3 |
| 5 | 2 |
| 6 | 1 |
| 7 | 3 |
| 8 | 3 |
| 9 | 3 |
| 10 | 2 |
| 11 | 2 |
| 12 | 3 |
| 13 | 3 |
| 14 | 2 |
| 15 | 3 |
| 16 | 3 |
| 17 | 2 |

-- **Find the avg_order_quantity of the company**

```
select round(avg(quantitypurchased),2) as avg_order_quantity
from sales;
```

| avg_order_quantity |
|---|
| 2.47 |

-- **Find the avg_order_price of the customers**

```
select customerid, round(avg(price*quantitypurchased)) as avg_order_price
from sales
group by customerid
order by avg_order_price desc;
```

| customerid | avg_order_price |
|---|---|
| 534 | 18721 |
| 821 | 9444 |
| 893 | 9351 |
| 371 | 7562 |
| 973 | 6343 |
| 819 | 6335 |
| 930 | 6285 |
| 482 | 5705 |
| 984 | 5503 |
| 172 | 5458 |
| 950 | 4692 |
| 422 | 4066 |
| 562 | 3220 |
| 944 | 3197 |
| 247 | 3139 |
| 820 | 2828 |
| 619 | 2449 |

-- **avg_order_value**

```
select round(avg(price*quantitypurchased)) as avg_order_price
from sales;
```

| avg_order_price |
|---|
| 242 |

-- ********END OF THE PROJECT********