

Data-Driven Insights for Jensen's

SUBMITTED BY: HIMANI GAUTAM

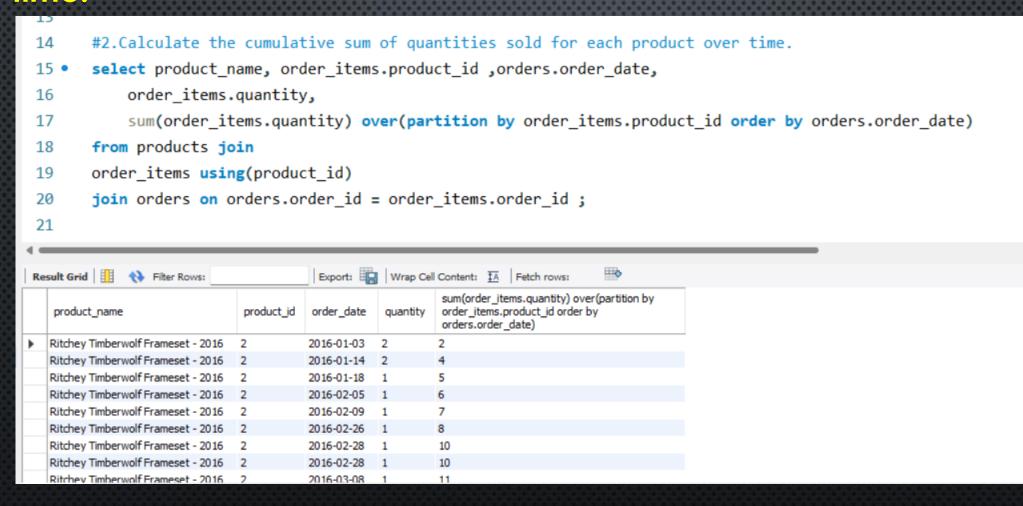
Problem Statements:

- 1. Find the total number of products sold by each store along with the store name.
- 2. Calculate the cumulative sum of quantities sold for each product over time.
- 3. Find the product with the highest total sales (quantity * price) for each category.
- 4. Find the customer who spent the most money on orders.
- 5. Find the highest-priced product for each category name.
- 6. Find the total number of orders placed by each customer per store.
- 7. Find the names of staff members who have not made any sales.
- 8. Find the top 3 most sold products in terms of quantity.
- 9. Find the median value of the price list.
- 10.List all products that have never been ordered.(use Exists)
- 11.List the names of staff members who have made more sales than the average number of sales by all staff members.
- 12.Identify the customers who have ordered all types of products (i.e., from every category).

1. Find the total number of products sold by each store along with the store name.

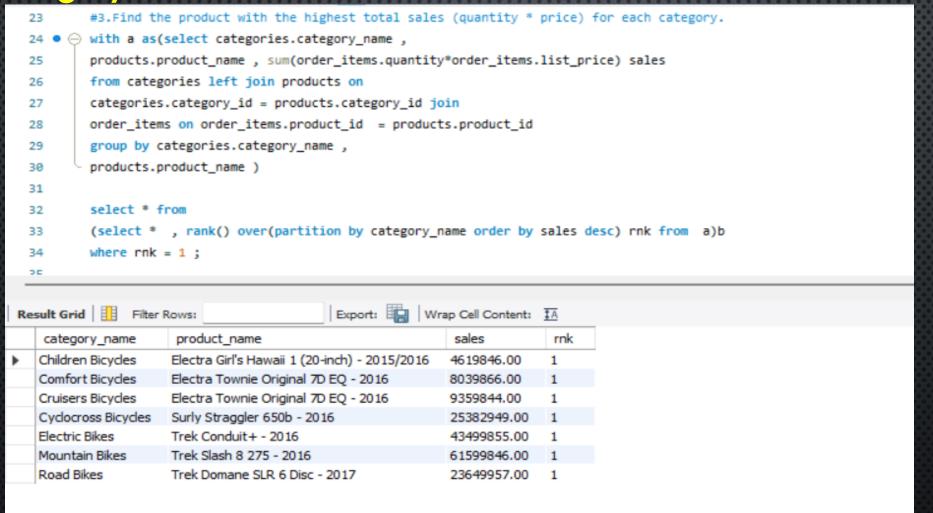
```
#1. Find the total number of products sold by each store along with the store name.
      SELECT
           orders.store id,
           stores.store_name,
           sum(order items.quantity)
       FROM
           order_items
               JOIN
           orders ON order items.order id = orders.order id
               JOIN
10
11
           stores ON orders.store id = stores.store id
12
       GROUP BY stores.store_id , stores.store_name;
Export: Wrap Cell Content: TA
                     sum(order_items.quantity)
  store id store name
         Santa Cruz Bikes
                    1516
         Baldwin Bikes
                     4779
         Rowlett Bikes
                    783
```

2. Calculate the cumulative sum of quantities sold for each product over time.





3. Find the product with the highest total sales (quantity * price) for each category.





4. Find the customer who spent the most money on orders.

customer_id

amount 3780184.00

```
37
      #4.Find the customer who spent the most money on orders.
38 • ⊖ with a as(SELECT
39
      orders.customer_id , sum(order_items.quantity * order_items.list_price) amount
40
      from orders join order_items
41
       on orders.order_id = order_items.order_id
42
       group by orders.customer_id )
43
       select * from a
44
       order by amount desc
45
       limit 1;
46
47
                          Export: Wrap Cell Content: IA
Result Grid
         Filter Rows:
```



5. Find the highest-priced product for each category name.

```
#5.Find the highest-priced product for each category name.

2 **\text{o} with a as(select category_id ,product_name , list_price ,

rank()over(partition by category_id order by list_price desc)rnk from products)

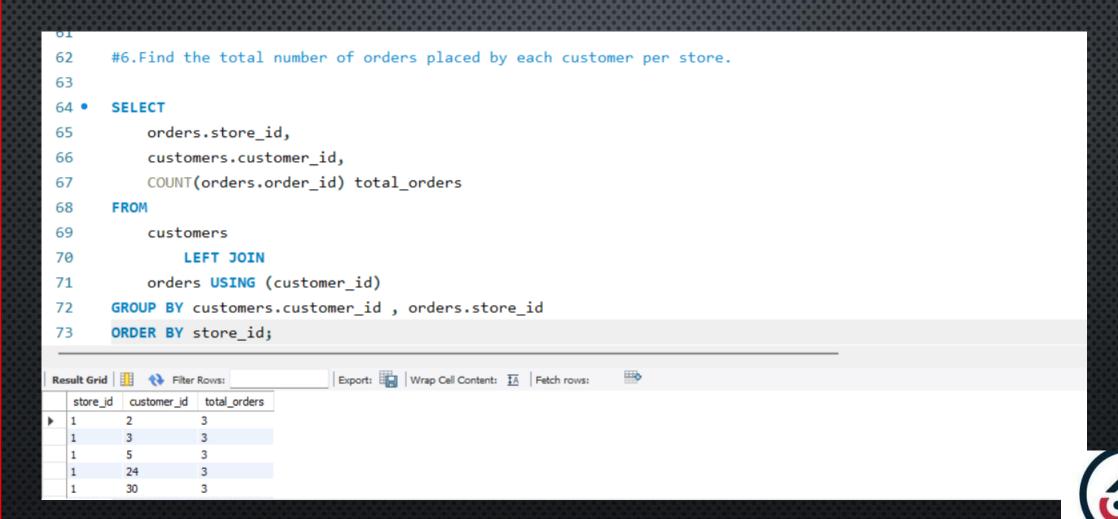
select * from a

where rnk = 1;
```

Re	sult Grid	Filter Rows: Export:	Wrap Cell Co	ontent: IA
	category_id	product_name	list_price	rnk
•	1	Electra Straight 8 3i (20-inch) - Boy's - 2017	48999.00	1
	1	Electra Townie 3i EQ (20-inch) - Boys' - 2017	48999.00	1
	1	Trek Superfly 24 - 2017/2018	48999.00	1
	2	Electra Townie Go! 8i - 2017/2018	259999.00	1
	3	Electra Townie Commute Go! - 2018	299999.00	1
	3	Electra Townie Commute Go! Ladies' - 2018	299999.00	1
	4	Trek Boone 7 Disc - 2018	399999.00	1
	5	Trek Powerfly 8 FS Plus - 2017	499999.00	1
	5	Trek Powerfly 7 FS - 2018	499999.00	1
	5	Trek Super Commuter + 8S - 2018	499999.00	1



6. Find the total number of orders placed by each customer per store.



7. Find the names of staff members who have not made any sales.

```
74
       #7. Find the names of staff members who have not made any sales.
75 •
       SELECT
            CONCAT(staffs.first_name, ' ', staffs.last_name) full_name
76
77
       FROM
            staffs
78
79
                LEFT JOIN
80
           orders USING (staff_id)
81
       WHERE
82
            orders.order_id IS NULL; ;
83
84
                                    Export: Wrap Cell Content: TA
Result Grid Filter Rows:
  full name
  Fabiola Jackson
  Virgie Wiggins
  Jannette David
  Bernardine Houston
```



8. Find the top 3 most sold products in terms of quantity

```
#8. Find the top 3 most sold products in terms of quantity.
 91
 92 •
        SELECT
 93
             products.product id,
 94
             products.product_name,
 95
             SUM(order_items.quantity)
 96
        FROM
 97
             products
                  JOIN
 98
 99
             order items USING (product id)
100
        GROUP BY products.product_id , products.product_name
        ORDER BY SUM(order_items.quantity) DESC
101
102
        LIMIT 3;
103
                                       Export: Wrap Cell Content: TA
             Filter Rows:
Result Grid
            product_name
                                        sum(order_items.quantity)
   product_id
            Surly Ice Cream Truck Frameset - 2016
                                        167
            Electra Cruiser 1 (24-Inch) - 2016
   13
                                        157
   16
            Electra Townie Original 7D EQ - 2016
                                        156
```



9. Find the median value of the price list.

```
111
         #9. Find the median value of the price list.
112 • ⊝ with a as (select list_price , row_number() over(order by list_price)pos ,
         count(*) over() n from order_items )
113
114
         select
115
      ⊖ case
116
            when n\%2 = 0 then (select avg(list price) from a where pos in (n/2, (n/2)+1))
117
            else (select list_price from a where pos = n+1/2)
118
         end as "median"
119
120
         from a
         limit 1;
121
Result Grid Filter Rows:
                                     Export: Wrap Cell Content: IA
   median
59999.000000
```



10. List all products that have never been ordered.(use Exists)

```
#10.List all products that have never been ordered.(use Exists)
129
130 •
        SELECT
131
             product name, product id
132
        FROM
133
             products
134
        WHERE
135
             NOT EXISTS ( SELECT
136
                       product id
137
                  FROM
138
                       order_items
139
                  WHERE
                       products.product_id = order_items.product_id);
140
141
                                       Edit: 🔏 🖶 🖶 Export/Import: 📳 🐻 Wrap Cell Content: 🟗
Result Grid Filter Rows:
                                   product id
   product name
   Trek Checkpoint ALR 5 Women's - 2019
                                  318
   Trek Checkpoint SL 5 Women's - 2019
                                  319
   Trek Checkpoint SL 6 - 2019
                                  320
   Trek Checkpoint ALR Frameset - 2019
                                  321
```



11.List the names of staff members who have made more sales than the average number of sales by all staff members.

```
136
        #11.List the names of staff members who have made more sales than the average number of sales by all staff members.
137 • ⊖ with a as(
        select staffs.staff id , concat(staffs.first name , " " , staffs.last name) full name,
138
139
        coalesce(count(order items.quantity * order items.list price),∅) sales
        from staffs left join orders using(staff_id)
140
        left join order items using(order id)
141
        group by staffs.staff_id , concat(staffs.first_name , " " , staffs.last_name) )
142
143
        select * from a
144
        where sales >(select avg(sales) from a);
145
                                  Export: Wrap Cell Content: IA
Result Grid Filter Rows:
   staff id full name
                       sales
          Genna Serrano
          Marcelene Boyer
                      1615
          Venita Daniel
```

12. Identify the customers who have ordered all types of products (i.e., from every category).

```
#12. Identify the customers who have ordered all types of products (i.e., from every category).
148
     with a as (select orders.customer id , count(distinct products.category id)category count from
      orders join order items using(order id)
150
151
      join products using(product_id)
      group by orders.customer id )
152
153
154
      select * from a
155
      where category_count = (select count(category_id ) from categories);
156
157
158
                        Export: Wrap Cell Content: TA
 Result Grid Filter Rows:
   customer_id category_count
```



Key Insights:

- 🚆 Store-wise Sales: Stores differ in total product sales can guide inventory allocation.
- 📕 Top-Spending Customers: High-value customers identified potential for loyalty programs.
- Category Stars: Top products by sales revenue vary across categories scope for targeted promotions.
- Staff Performance: Some staff outperform, while others made no sales use for training or optimization.
- © Pricing Distribution: Median price value calculated helpful for pricing strategy benchmarking.

