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).

JENSON USA

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

```sql
#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
    stores ON orders.store_id = stores.store_id
GROUP BY stores.store_id  , stores.store_name;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

store_id	store_name	sum(order_items.quantity)
1	Santa Cruz Bikes	1516
2	Baldwin Bikes	4779
3	Rowlett Bikes	783

JENSON USA

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

```
14    #2.Calculate the cumulative sum of quantities sold for each product over time.
15 •  select product_name, order_items.product_id ,orders.order_date,
16        order_items.quantity,
17        sum(order_items.quantity) over(partition by order_items.product_id order by orders.order_date)
18    from products join
19    order_items using(product_id)
20    join orders on orders.order_id = order_items.order_id ;
21
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

product_name	product_id	order_date	quantity	sum(order_items.quantity) over(partition by order_items.product_id order by orders.order_date)
Ritchey Timberwolf Frameset - 2016	2	2016-01-03	2	2
Ritchey Timberwolf Frameset - 2016	2	2016-01-14	2	4
Ritchey Timberwolf Frameset - 2016	2	2016-01-18	1	5
Ritchey Timberwolf Frameset - 2016	2	2016-02-05	1	6
Ritchey Timberwolf Frameset - 2016	2	2016-02-09	1	7
Ritchey Timberwolf Frameset - 2016	2	2016-02-26	1	8
Ritchey Timberwolf Frameset - 2016	2	2016-02-28	1	10
Ritchey Timberwolf Frameset - 2016	2	2016-02-28	1	10
Ritchey Timberwolf Frameset - 2016	2	2016-03-08	1	11

JENSON USA

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

```
23      #3.Find the product with the highest total sales (quantity * price) for each category.
24      with a as(select categories.category_name ,
25      products.product_name , sum(order_items.quantity*order_items.list_price) sales
26      from categories left join products on
27      categories.category_id = products.category_id join
28      order_items on order_items.product_id  = products.product_id
29      group by categories.category_name ,
30      products.product_name )
31
32      select * from
33      (select *  , rank() over(partition by category_name order by sales desc) rnk from  a)b
34      where rnk = 1 ;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

category_name	product_name	sales	rnk
Children Bicycles	Electra Girl's Hawaii 1 (20-inch) - 2015/2016	4619846.00	1
Comfort Bicycles	Electra Townie Original 7D EQ - 2016	8039866.00	1
Cruisers Bicycles	Electra Townie Original 7D EQ - 2016	9359844.00	1
Cyclocross Bicycles	Surly Straggler 650b - 2016	25382949.00	1
Electric Bikes	Trek Conduit+ - 2016	43499855.00	1
Mountain Bikes	Trek Slash 8 275 - 2016	61599846.00	1
Road Bikes	Trek Domane SLR 6 Disc - 2017	23649957.00	1

JENSON USA

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

```
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
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_id	amount
10	3780184.00

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

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

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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

JENSON USA

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

```
61
62    #6.Find the total number of orders placed by each customer per store.
63
64 •  SELECT
65        orders.store_id,
66        customers.customer_id,
67        COUNT(orders.order_id) total_orders
68    FROM
69        customers
70            LEFT JOIN
71        orders USING (customer_id)
72    GROUP BY customers.customer_id , orders.store_id
73    ORDER BY store_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

store_id	customer_id	total_orders
1	2	3
1	3	3
1	5	3
1	24	3
1	30	3

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
76        CONCAT(staffs.first_name, ' ', staffs.last_name) full_name
77    FROM
78        staffs
79            LEFT JOIN
80        orders USING (staff_id)
81    WHERE
82        orders.order_id IS NULL;  ;
83
84
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

full_name
Fabiola Jackson
Virgie Wiggins
Jannette David
Bernardine Houston

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

```
91    #8.Find the top 3 most sold products in terms of quantity.
92 •  SELECT
93        products.product_id,
94        products.product_name,
95        SUM(order_items.quantity)
96    FROM
97        products
98            JOIN
99        order_items USING (product_id)
100   GROUP BY products.product_id , products.product_name
101   ORDER BY SUM(order_items.quantity) DESC
102   LIMIT 3;
103
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

product_id	product_name	sum(order_items.quantity)
6	Surly Ice Cream Truck Frameset - 2016	167
13	Electra Cruiser 1 (24-Inch) - 2016	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 ,
113     count(*) over() n  from order_items )
114
115     select
116     case
117        when n%2 = 0 then (select avg(list_price) from a where pos in (n/2 , (n/2)+1))
118        else (select list_price from a where pos = n+1/2)
119     end as "median"
120     from a
121     limit 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

median
59999.000000

JENSON USA

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

```
129  #10.List all products that have never been ordered.(use Exists)
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
140              products.product_id = order_items.product_id) ;
141
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

product_name	product_id
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
NULL	NULL

JENSON USA

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(
138    select staffs.staff_id , concat(staffs.first_name , " " , staffs.last_name) full_name,
139    coalesce(count(order_items.quantity * order_items.list_price),0) sales
140    from staffs left join orders using(staff_id)
141    left join order_items using(order_id)
142    group by staffs.staff_id , concat(staffs.first_name , " " , staffs.last_name)  )
143
144    select * from a
145    where sales >(select avg(sales) from a);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

staff_id	full_name	sales
3	Genna Serrano	544
6	Marcelene Boyer	1615
7	Venita Daniel	1580

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

```
148  #12.Identify the customers who have ordered all types of products (i.e., from every category).
149  with a as (select orders.customer_id  , count(distinct products.category_id)category_count from
150  orders join order_items using(order_id)
151  join products using(product_id)
152  group by orders.customer_id )
153
154  select * from a
155  where category_count = (select count(category_id ) from categories) ;
156
157
158
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_id	category_count
9	7

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.
- 🛒 Zero-Order Products: Several products never ordered — opportunity to investigate pricing or demand.
- 👨‍💼 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.