



“SQL DATASET ANALYSIS”

BY MANISH KUMAR



INTRODUCTION



- Jenson USA is one of the leading retailers of bicycles, parts, and accessories.
- Founded in 1994, it started as a small bike shop and has grown into a trusted online and offline cycling store.
- The company focuses on providing quality products and excellent customer service to cycling enthusiasts.
- With thousands of products and loyal customers, Jenson USA continues to be a key player in the cycling industry

OBJECTIVE

- To analyze the Jenson USA dataset using SQL.
- To identify top customers, best-selling products, and categories.
- To evaluate sales performance of staff and stores.
- To detect unsold products and sales trends.
- To provide data-driven insights for better decisions.

1. FIND THE TOTAL NUMBER OF PRODUCTS SOLD BY EACH STORE ALONG WITH THE STORE NAME.

```
1 -- 1.FIND THE TOTAL NUMBER OF PRODUCTS SOLD BY EACH STORE ALONG WITH THE STORE NAME.  
2  
3 • select s.store_name,  
4         SUM(oi.quantity) AS Total_number_product  
5 From order_items as oi  
6 Join orders As o ON oi.order_id = o.order_id  
7 join stores AS s ON o.store_id = s.store_id  
8 group by s.store_name;
```

store_name	Total_number_product
Santa Cruz Bikes	1516
Baldwin Bikes	4779
Rowlett Bikes	783

2. CALCULATE THE CUMULATIVE SUM OF QUANTITIES SOLD FOR EACH PRODUCT OVER TIME.

```
1  -- 2.CALCULATE THE CUMULATIVE SUM OF QUANTITIES SOLD FOR EACH PRODUCT OVER TIME.  
2  
3 • select p.product_name,  
4      o.order_date,  
5      sum(oi.quantity) over(partition by p.product_id order by o.order_date) AS cumulative_quantity  
6  From products AS p  
7  join order_items AS oi ON p.product_id = oi.product_id  
8  Join orders AS o ON oi.order_id = o.order_id;
```



The screenshot shows a MySQL query results window. At the top, there are buttons for 'Result Grid' (selected), 'Filter Rows:', 'Export', 'Wrap Cell Content:', and 'Fetch rows:'. Below is a table with three columns: 'product_name', 'order_date', and 'cumulative_quantity'. The data shows a single product ('Ritchey Timberwolf Frameset - 2016') with five rows, each having a different 'order_date' and a corresponding cumulative quantity starting from 2 and increasing to 7.

	product_name	order_date	cumulative_quantity
▶	Ritchey Timberwolf Frameset - 2016	2016-01-03	2
	Ritchey Timberwolf Frameset - 2016	2016-01-14	4
	Ritchey Timberwolf Frameset - 2016	2016-01-18	5
	Ritchey Timberwolf Frameset - 2016	2016-02-05	6
	Ritchey Timberwolf Frameset - 2016	2016-02-09	7

3. FIND THE PRODUCT WITH THE HIGHEST TOTAL SALES FOR EACH CATEGORY.

```
1  -- 3. Find the product with the highest total sales (quantity * price) for each category.
2
3  select category_name, product_name, total_sales
4  From (
5    select
6      c.category_name,
7      p.product_name,
8      Sum(oi.quantity * oi.list_price) AS total_sales,
9      ROW_NUMBER() OVER (PARTITION BY c.category_id ORDER BY SUM(oi.quantity * oi.list_price) DESC) AS rn
10   from order_items AS oi
11   JOIN Products AS p ON oi.product_id = p.product_id
12   JOIN categories AS c ON p.category_id = c.category_id
13   group by c.category_name, c.category_id, p.product_name
14 ) top_products
15 where rn = 1;
```

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

	category_name	product_name	total_sales
▶	Children Bicycles	Electra Girl's Hawaii 1 (20-inch) - 20...	4619846.00
	Comfort Bicycles	Electra Townie Original 7D EQ - 2016	8039866.00
	Cruisers Bicycles	Electra Townie Original 7D EQ - 2016	9359844.00
	Cyclocross Bicycles	Surly Straggler 650b - 2016	25382949.00
	Electric Bikes	Trek Conduit+ - 2016	43499855.00

4. FIND THE CUSTOMER WHO SPENT THE MOST MONEY ON ORDERS.

```
54 • with a as (
55     select
56         concat(customers.first_name, " ", customers.last_name)      as Customer_Name,
57         orders.order_id,
58         sum(order_items.quantity * (order_items.list_price - order_items.discount)) as sales
59     from orders
60     join customers on orders.customer_id = customers.customer_id
61     join order_items on orders.order_id = order_items.order_id
62     group by concat(customers.first_name, " ", customers.last_name), orders.order_id
63 )
64     select *
65     from (
66         select *, rank() over(order by sales desc) as rnk
67         from a
68     ) temp
69     where rnk = 1;
70
71
```

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

	Customer_Name	order_id	sales	rnk
▶	Jacqueline Duncan	1541	3232863.00	1

5. FIND THE HIGHEST-PRICED PRODUCT FOR EACH CATEGORY NAME.

```
79 •   select *
80   ⏺ from (
81     select
82       categories.category_id,
83       categories.category_name,
84       products.product_name,
85       products.list_price,
86       rank() over (
87         partition by categories.category_id
88         order by products.list_price desc)      as rnk
89     from categories
90     join products on categories.category_id = products.category_id) as rank_product
91
92   where rnk = 1;
```

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

	category_id	category_name	product_name	list_price	rnk
▶	1	Children Bicycles	Electra Straight 8 3i (20-inch) - Boy's - 2017	48999.00	1
	1	Children Bicycles	Electra Townie 3i EQ (20-inch) - Boys' - 2017	48999.00	1
	1	Children Bicycles	Trek Superfly 24 - 2017/2018	48999.00	1
	2	Comfort Bicycles	Electra Townie Go! 8i - 2017/2018	259999.00	1
	3	Cruisers Bicycles	Electra Townie Commute Go! - 2018	299999.00	1
	3	Cruisers Bicycles	Electra Townie Commute Go! Ladies' - 2018	299999.00	1
	4	Cyclameric Bicycles	Trek Rove 7 Disc - 2018	399999.00	1

Result 6 ×

6. FIND THE TOTAL NUMBER OF ORDERS PLACED BY EACH CUSTOMER PER STORE.

```
96 •      select
97          concat(customers.first_name, ' ', customers.last_name) as cust_name,
98          stores.store_name,
99          COUNT(orders.order_id) as total_orders
100     from customers
101        join orders on customers.customer_id = orders.customer_id
102        join stores  on stores.store_id = orders.store_id
103    group by
104          concat(customers.first_name, ' ', customers.last_name),
105          stores.store_name,
106          stores.store_id;
```

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

	cust_name	store_name	total_orders
▶	Earl Stanley	Santa Cruz Bikes	1
	Marquerite Dawson	Santa Cruz Bikes	2
	Damien Dorsey	Santa Cruz Bikes	2
	Arvilla Osborn	Santa Cruz Bikes	2
	Erma Salinas	Santa Cruz Bikes	1
	Felicidad Golden	Santa Cruz Bikes	1
	Chanel May	Santa Cruz Bikes	1

Result 8

7. FIND THE NAMES OF STAFF MEMBERS WHO HAVE NOT MADE ANY SALES.

```
111 •      SELECT
112          staffs.staff_id,
113          CONCAT(staffs.first_name, ' ', staffs.last_name) AS full_name
114      FROM staffs
115      WHERE NOT EXISTS (
116          SELECT *
117          FROM orders
118          WHERE orders.staff_id = staffs.staff_id
119      );
```

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

	staff_id	full_name
▶	1	Fabiola Jackson
	4	Virgie Wiggins
	5	Jannette David
	10	Bernardine Houston

8. FIND THE TOP 3 MOST SOLD PRODUCTS IN TERMS OF QUANTITY.

```
124  
125 •     select product_name, product_id  
126   ○       from (  
127     select  
128       products.product_id,  
129       products.product_name,  
130       sum(order_items.quantity)    as total_quantity,  
131       rank() over (order by sum(order_items.quantity) desc) as rnk  
132  
133     from order_items  
134     join products ON products.product_id = order_items.product_id  
135  
136   group by products.product_id, products.product_name)    as temp  
137     where rnk <= 3;
```

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

	product_name	product_id
▶	Surly Ice Cream Truck Frameset - 2016	6
	Electra Cruiser 1 (24-Inch) - 2016	13
	Electra Townie Original 7D EQ - 2016	16

9. FIND THE MEDIAN VALUE OF THE PRICE LIST.

```
25
26 •   select
27         avg(list_price) AS median_price
28     from (
29         select list_price,
30                 ROW_NUMBER() OVER (ORDER BY list_price) AS row_num,
31                 COUNT(*) OVER () AS total_count
32         from products
33     ) t
34     WHERE row_num IN ( (total_count + 1)/2 , (total_count + 2)/2 );
35
36
37
38
```

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

	median_price
▶	74999.000000

10. LIST ALL PRODUCTS THAT HAVE NEVER BEEN ORDERED.

```
140 •   select
141         products.product_id,
142         products.product_name
143     FROM products
144     WHERE NOT EXISTS (
145         SELECT order_items.product_id
146         FROM order_items
147        WHERE order_items.product_id = products.product_id
148    );
149
```

Result Grid | Filter Rows: | Edit: | Export/Import:

	product_id	product_name
▶	1	Trek 820 - 2016
	121	Surly Krampus Frameset - 2018
	125	Trek Kids' Dual Sport - 2018
	154	Trek Domane SLR 6 Disc Women's - 2018
	195	Electra Townie Go! 8i Ladies' - 2018
	267	Trek Precaliber 12 Girl's - 2018
	284	Electra Savannah 1 (20-inch) - Girl's - 2018
	291	Electra Sweet Ride 1 (20-inch) - Girl's - 2018
	316	Trek Checkpoint ALR 4 Women's - 2019
	317	Trek Checkpoint ALR 5 - 2019
	318	Trek Checkpoint ALR 5 Women's - 2019

11. LIST THE NAMES OF STAFF MEMBERS WHO HAVE MADE MORE SALES THAN THE AVERAGE NUMBER OF SALES BY ALL STAFF MEMBERS.

```
167 • with a as (
168     select
169         concat(staffs.first_name, ' ', staffs.last_name) as full_name,
170         sum(order_items.quantity * order_items.list_price) as Sales
171     from staffs
172     left join orders
173         on staffs.staff_id = orders.staff_id
174     left join order_items
175         on orders.order_id = order_items.order_id
176     group by concat(staffs.first_name, ' ', staffs.last_name)
177 )
178 select *
179 from a
180 where Sales > (select avg(Sales) from a);
181
```

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

	full_name	Sales
▶	Marcelene Boyer	293888873.00
	Venita Daniel	288735348.00

11. LIST THE NAMES OF STAFF MEMBERS WHO HAVE MADE MORE SALES THAN THE AVERAGE NUMBER OF SALES BY ALL STAFF MEMBERS.

```
167 • with a as (
168     select
169         concat(staffs.first_name, ' ', staffs.last_name) as full_name,
170         sum(order_items.quantity * order_items.list_price) as Sales
171     from staffs
172     left join orders
173         on staffs.staff_id = orders.staff_id
174     left join order_items
175         on orders.order_id = order_items.order_id
176     group by concat(staffs.first_name, ' ', staffs.last_name)
177 )
178 select *
179 from a
180 where Sales > (select avg(Sales) from a);
181
```

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

	full_name	Sales
▶	Marcelene Boyer	293888873.00
	Venita Daniel	288735348.00

12. IDENTIFY THE CUSTOMERS WHO HAVE ORDERED ALL TYPES OF PRODUCTS.

```
---  
152 •   select customers.customer_id  
153       from customers  
154       join orders ON customers.customer_id = orders.customer_id  
155       join order_items ON orders.order_id = order_items.order_id  
156       join products p ON p.product_id = order_items.product_id  
157       group by customers.customer_id  
158     - having count(distinct p.category_id) = (  
159         select count(categories.category_id)  
160           from categories  
161     );  
162
```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
	customer_id			.
▶	9			

CONCLUSION

- Analysis of the Jenson USA dataset using SQL revealed top-performing and underperforming stores.
- Identified best-selling products, unsold products, and sales trends.
- Highlighted top-spending customers and loyal buyers.
- Calculated pricing insights like highest-priced items and median price.
- Overall, the project showed how SQL can turn raw data into meaningful business insights to support better decisions.



Thank you