



# SQL PORTFOLIO PROJECT

AAYUSH JOSHI

*Q1) Retrieve the total number of orders placed.*

```
SELECT COUNT(order_id) AS  
total_orders FROM orders;
```

```
1  -- Q1) Retrieve the total number of orders placed.  
2  
3 • SELECT  
4      COUNT(order_id) AS total_orders  
5  FROM  
6      orders;
```

Result Grid		 Filter Rows: <input type="text"/>	Export: 	Wrap Cell Content: 
	total_orders			
▶	21350			

*Q2) Calculate the total revenue generated from pizza sales.*

```
SELECT ROUND(SUM(quantity *  
price), 2) AS total_revenue FROM  
order_details a INNER JOIN pizzas  
b ON a.pizza_id = b.pizza_id;
```

```
1  -- Q2) Calculate the total revenue generated from pizza sales.  
2  
3  •  SELECT  
4      ROUND(SUM(quantity * price), 2) AS total_revenue  
5  FROM  
6      order_details a  
7      INNER JOIN  
8      pizzas b ON a.pizza_id = b.pizza_id;  
9
```

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

	total_revenue
▶	817860.05

### *Q3) Identify the highest-priced pizza.*

```
SELECT name, price AS  
highest_priced_pizza FROM pizzas  
a INNER JOIN pizza_types b ON  
a.pizza_type_id = b.pizza_type_id  
ORDER BY highest_priced_pizza  
DESC LIMIT 1;
```

```
1  -- Q3) Identify the highest-priced pizza.  
2  
3 • SELECT  
4     name, price AS highest_priced_pizza  
5 FROM  
6     pizzas a  
7     INNER JOIN  
8     pizza_types b ON a.pizza_type_id = b.pizza_type_id  
9 ORDER BY highest_priced_pizza DESC  
10 LIMIT 1;  
11
```

---

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

	name	highest_priced_pizza
▶	The Greek Pizza	35.95

*Q4) Identify the most common pizza size ordered.*

```
SELECT size,  
COUNT(order_details_id) AS COUNT  
FROM pizzas a INNER JOIN  
order_details b ON a.pizza_id =  
b.pizza_id GROUP BY size ORDER BY  
COUNT DESC LIMIT 1;
```

```
1  -- Q4) Identify the most common pizza size ordered.  
2  
3  • SELECT  
4      size, COUNT(order_details_id) AS COUNT  
5  FROM  
6      pizzas a  
7      INNER JOIN  
8      order_details b ON a.pizza_id = b.pizza_id  
9  GROUP BY size  
10 ORDER BY COUNT DESC  
11 LIMIT 1;
```

Result Grid			Filter Rows: <input type="text"/>	Export: 	Wrap Cell Content: 
	size	COUNT			
▶	L	18526			

*Q5) List the top 5 most ordered pizza types along with their quantities.*

```
SELECT name, SUM(quantity) AS s_qty
FROM  pizza_types INNER JOIN pizzas
      ON pizza_types.pizza_type_id =
      pizzas.pizza_type_id  INNER JOIN
order_details ON order_details.pizza_id
= pizzas.pizza_id GROUP BY name
ORDER BY s_qty DESC LIMIT 5;
```

```
1  -- Q5) List the top 5 most ordered pizza types along with their quantities.
2  • SELECT
3      name, SUM(quantity) AS s_qty
4  FROM
5      pizza_types
6      INNER JOIN
7      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
8      INNER JOIN
9      order_details ON order_details.pizza_id = pizzas.pizza_id
10 GROUP BY name
11 ORDER BY s_qty DESC
12 LIMIT 5;
```

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

	name	s_qty
▶	The Classic Deluxe Pizza	2453
	The Barbecue Chicken Pizza	2432
	The Hawaiian Pizza	2422
	The Pepperoni Pizza	2418
	The Thai Chicken Pizza	2371



*Q6) Join the necessary tables to find the total quantity of each pizza category ordered.*

```
SELECT category, SUM(quantity) AS  
    Quantity FROM  pizza_types a  
    INNER JOIN  pizzas b ON  
a.pizza_type_id = b.pizza_type_id  
    INNER JOIN  order_details c ON  
c.pizza_id = b.pizza_id GROUP BY  
category order by Quantity desc;
```

```
1  -- Q6) Join the necessary tables to find the total quantity of each pizza category ordered.  
2  
3  • SELECT  
4      category, SUM(quantity) AS Quantity  
5  FROM  
6      pizza_types a  
7      INNER JOIN  
8      pizzas b ON a.pizza_type_id = b.pizza_type_id  
9      INNER JOIN  
10     order_details c ON c.pizza_id = b.pizza_id  
11  GROUP BY category order by Quantity desc;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [IA](#)

	category	Quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

*Q7) Determine the distribution of orders by hour of the day.*

```
SELECT HOUR(order_time) AS hours,  
COUNT(order_id) AS order_count  
FROM orders GROUP BY hours  
order by hours asc;
```

```
1  -- Q7) Determine the distribution of orders by hour of the day.  
2  
3 • SELECT  
4     HOUR(order_time) AS hours, COUNT(order_id) AS order_count  
5 FROM  
6     orders  
7 GROUP BY hours order by hours asc;
```

	hours	order_count
9	1	
10	8	
11	1231	
12	2520	
13	2455	
14	1472	
15	1468	
16	1920	
17	2336	
18	2399	
19	2009	
20	1642	
21	1198	
22	663	
23	28	



*Q8) Join relevant tables to find the category-wise distribution of pizzas.*

```
SELECT category, SUM(quantity) as  
       n_qty, count(distinct name) as  
       pizza_type FROM order_details a  
INNER JOIN pizzas b ON a.pizza_id =  
b.pizza_id INNER JOIN pizza_types c  
ON c.pizza_type_id = b.pizza_type_id  
GROUP BY category ORDER BY n_qty  
DESC;
```

```
1  -- Q8) Join relevant tables to find the category-wise distribution of pizzas.  
2  •  SELECT  
3      category, SUM(quantity) as n_qty, count(distinct name) as pizza_type  
4  FROM  
5      order_details a  
6      INNER JOIN  
7      pizzas b ON a.pizza_id = b.pizza_id  
8      INNER JOIN  
9      pizza_types c ON c.pizza_type_id = b.pizza_type_id  
10 GROUP BY category  
11 ORDER BY n_qty DESC;
```

Result Grid			Filter Rows:	<input type="text"/>	Export:	Wrap Cell Content:
	category	n_qty	pizza_type			
▶	Classic	14888	8			
	Supreme	11987	9			
	Veggie	11649	9			
	Chicken	11050	6			

*Q9) Group the orders by date and calculate the average number of pizzas ordered per day.*

```
select round(avg(sum_qty), 0) as  
AVG_PIZZAS_PER_DAY from (SELECT  
order_date, sum(quantity) AS sum_qty  
FROM orders a INNER JOIN  
order_details b ON a.order_id =  
b.order_id GROUP BY order_date order  
by order_date asc) as order_quantity
```

```
1  -- Q9) Group the orders by date and calculate the average number of pizzas ordered per day.  
2  • select round(avg(sum_qty), 0) as AVG_PIZZAS_PER_DAY from (  
3  SELECT  
4      order_date, sum(quantity) AS sum_qty  
5  FROM  
6      orders a  
7      INNER JOIN  
8      order_details b ON a.order_id = b.order_id  
9  GROUP BY order_date  
10 order by order_date asc  
11 ) as order_quantity  
12
```

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

AVG_PIZZAS_PER_DAY
138

*Q10) Determine the top 3 most ordered pizza types based on revenue.*

```
SELECT name, SUM(b.quantity * a.price)
AS revenue FROM pizzas a INNER
JOIN order_details b ON a.pizza_id =
b.pizza_id INNER JOIN pizza_types
c ON c.pizza_type_id = a.pizza_type_id
GROUP BY name ORDER BY revenue
DESC LIMIT 3;
```

```
1  -- Q10) Determine the top 3 most ordered pizza types based on revenue.
2  • SELECT
3      name, SUM(b.quantity * a.price) AS revenue
4  FROM
5      pizzas a
6      INNER JOIN
7      order_details b ON a.pizza_id = b.pizza_id
8      INNER JOIN
9      pizza_types c ON c.pizza_type_id = a.pizza_type_id
10 GROUP BY name
11 ORDER BY revenue DESC
12 LIMIT 3;
```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
	name	revenue	
▶	The Thai Chicken Pizza	43434.25	
	The Barbecue Chicken Pizza	42768	
	The California Chicken Pizza	41409.5	

*Q11) Calculate the percentage contribution of each pizza type to total revenue.*

```
SELECT category, (SUM(b.quantity *  
a.price) / (SELECT SUM(quantity * price)  
FROM order_details a INNER JOIN pizzas b  
ON a.pizza_id = b.pizza_id))*100 AS  
revenue_percent FROM pizzas a INNER  
JOIN order_details b ON a.pizza_id =  
b.pizza_id INNER JOIN pizza_types c ON  
c.pizza_type_id = a.pizza_type_id GROUP BY  
category ORDER BY revenue_percent DESC
```

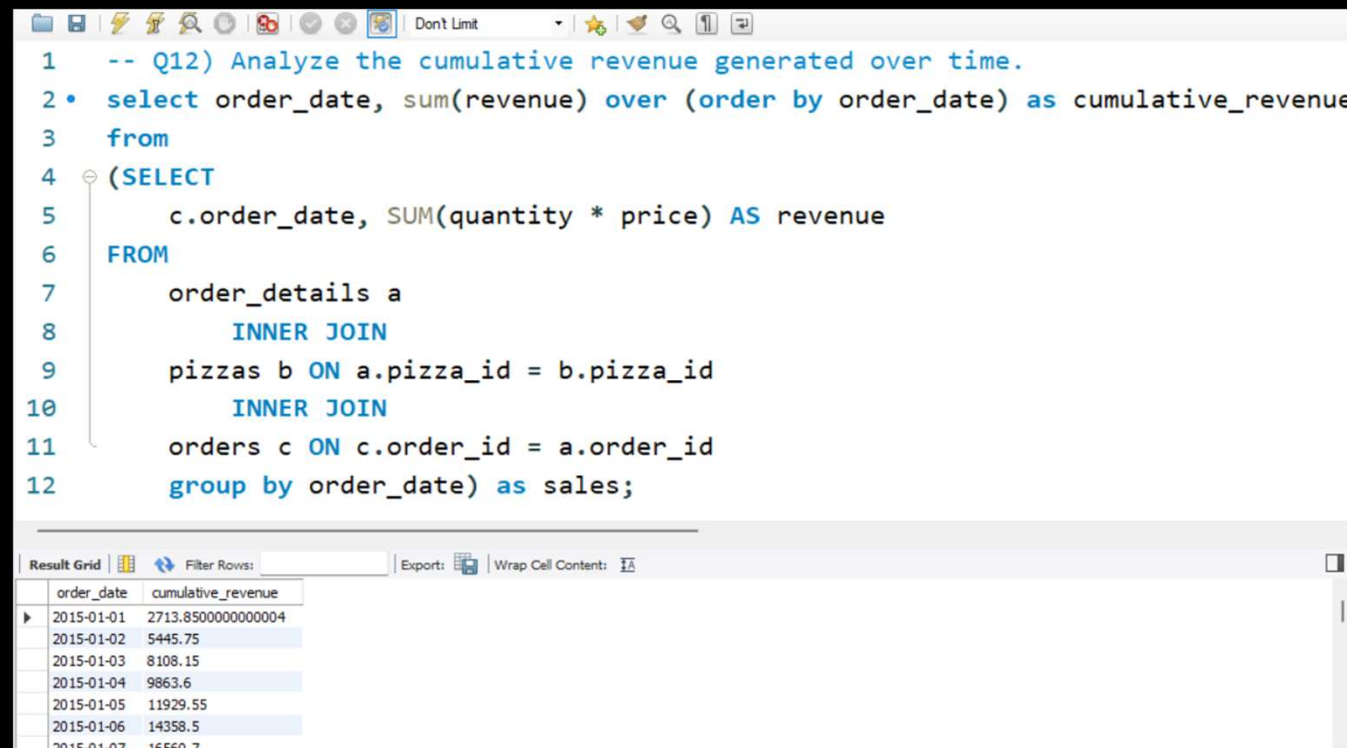
```
1  -- Q11) Calculate the percentage contribution of each pizza type to total revenue.  
2  •  SELECT  
3      category,  
4      (SUM(b.quantity * a.price) / (SELECT  
5          SUM(quantity * price)  
6          FROM  
7              order_details a  
8              INNER JOIN  
9                  pizzas b ON a.pizza_id = b.pizza_id))*100 AS revenue_percent  
10 FROM  
11     pizzas a  
12     INNER JOIN  
13     order_details b ON a.pizza_id = b.pizza_id  
14     INNER JOIN  
15     pizza_types c ON c.pizza_type_id = a.pizza_type_id  
16 GROUP BY category  
17 ORDER BY revenue_percent DESC
```

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

	category	revenue_percent
▶	Classic	26.905960255669903
	Supreme	25.45631126009884
	Chicken	23.955137556847493
	Veggie	23.682590927384783

# Q12) Analyze the cumulative revenue generated over time.

```
select order_date, sum(revenue) over (order
by order_date) as cumulative_revenue from
(SELECT c.order_date, SUM(quantity *
price) AS revenue FROM order_details a
INNER JOIN pizzas b ON a.pizza_id =
b.pizza_id INNER JOIN orders c ON
c.order_id = a.order_id group by
order_date) as sales;
```



The screenshot shows a SQL IDE window with a query editor and a results grid. The query is a complex SQL statement that calculates cumulative revenue over time by joining order\_details, pizzas, and orders tables. The results grid displays the output of the query, showing order dates and their corresponding cumulative revenue values.

```
1  -- Q12) Analyze the cumulative revenue generated over time.
2  • select order_date, sum(revenue) over (order by order_date) as cumulative_revenue
3  from
4  (SELECT
5      c.order_date, SUM(quantity * price) AS revenue
6  FROM
7      order_details a
8      INNER JOIN
9      pizzas b ON a.pizza_id = b.pizza_id
10     INNER JOIN
11     orders c ON c.order_id = a.order_id
12     group by order_date) as sales;
```

order_date	cumulative_revenue
2015-01-01	2713.8500000000000004
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6
2015-01-05	11929.55
2015-01-06	14358.5
2015-01-07	16560.7

*Q13) Determine the top 3 most ordered pizza types based on revenue for each pizza category.*

```
select name, revenue, rn from (select category,
    name, revenue, rank() over(partition by
    category order by revenue desc) as rn from
    (select category, name, sum(quantity*price) as
    revenue from pizza_types a inner join pizzas b
    on a.pizza_type_id = b.pizza_type_id inner join
    order_details c on c.pizza_id = b.pizza_id group
    by category, name order by revenue) as a) as b
    where rn<=3;
```

```
1  -- Q12) Determine the top 3 most ordered pizza types based on revenue for each pizza
2 • select name, revenue, rn from
3  (select category, name, revenue, rank() over(partition by category order by
4    revenue desc) as rn from (
5    select category, name, sum(quantity*price) as revenue from pizza_types a inner join
6    pizzas b on a.pizza_type_id = b.pizza_type_id
7    inner join order_details c on c.pizza_id = b.pizza_id
8    group by category, name order by revenue
9  ) as a) as b where rn<=3;
```

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

	name	revenue	rn
▶	The Thai Chicken Pizza	43434.25	1
	The Barbecue Chicken Pizza	42768	2
	The California Chicken Pizza	41409.5	3
	The Classic Deluxe Pizza	38180.5	1
	The Hawaiian Pizza	32273.25	2
	The Pepperoni Pizza	30161.75	3
	The Spicy Italian Pizza	34831.25	1
	The Italian Supreme Pizza	33476.75	2

Result Grid

Form Editor



THANK YOU