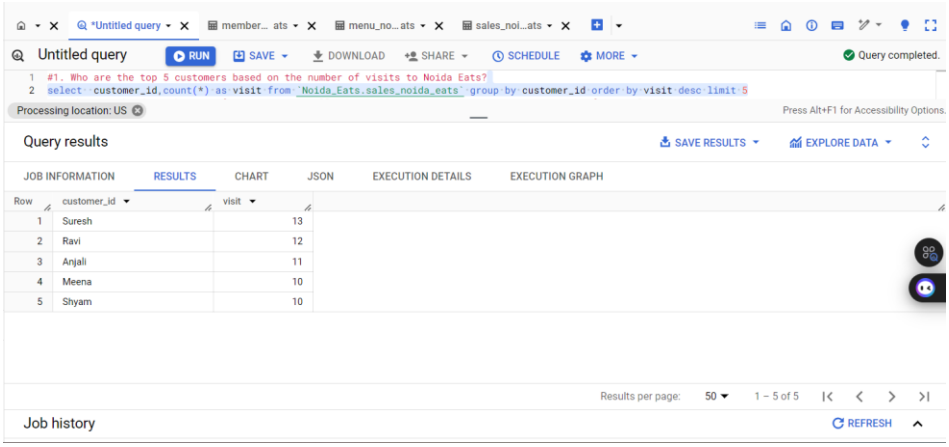
Case Study: Analyzing Noida Eats'

Scenario 1: Customer Visiting Patterns

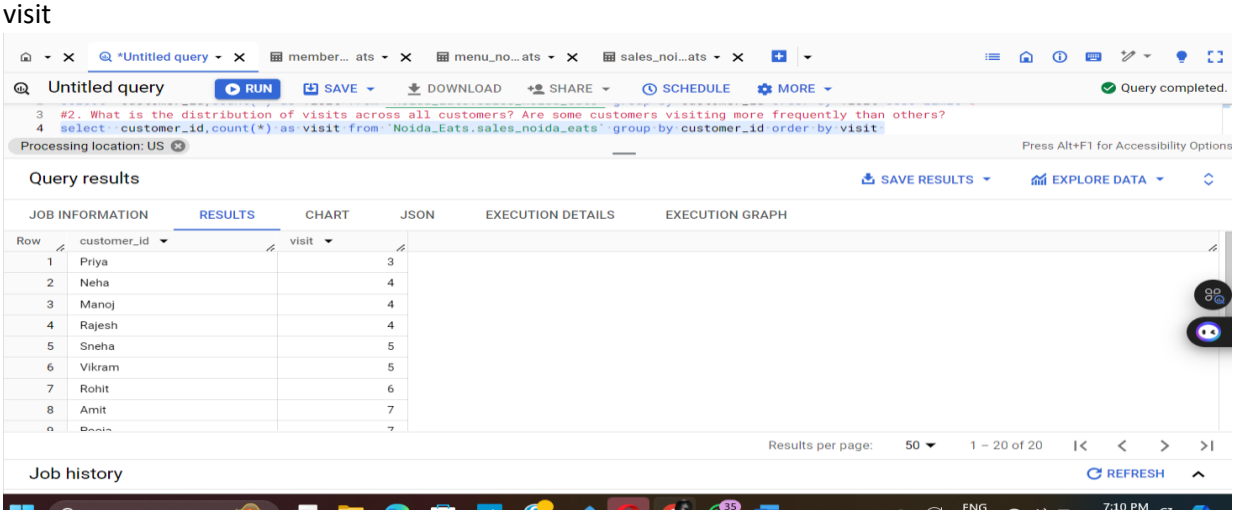
#1. Who are the top 5 customers based on the number of visits to Noida Eats?

select customer_id,count(*) as visit from `Noida_Eats.sales_noida_eats` group by customer_id order by visit desc limit 5



#2. What is the distribution of visits across all customers? Are some customers visiting more frequently than others?

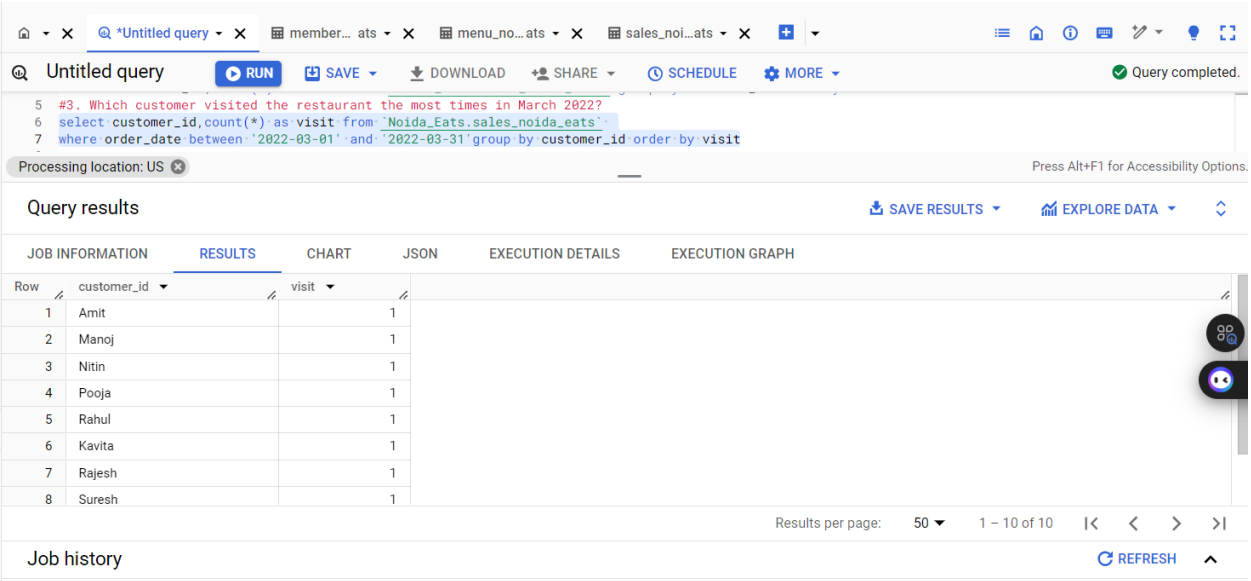
select customer_id,count(*) as visit from `Noida_Eats.sales_noida_eats` group by customer_id order by visit



#3. Which customer visited the restaurant the most times in March 2022?

select customer_id,count(*) as visit from `Noida_Eats.sales_noida_eats`

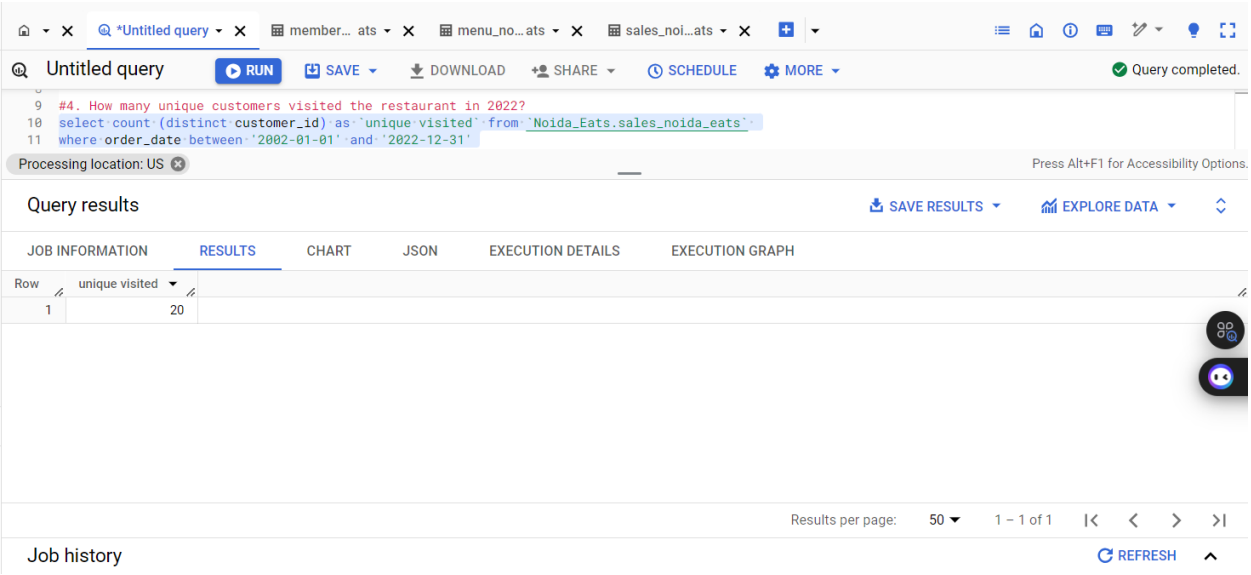
where order_date between '2022-03-01' and '2022-03-31'group by customer_id order by visit



#4. How many unique customers visited the restaurant in 2022?

select count (distinct customer_id) as `unique visited` from `Noida_Eats.sales_noida_eats`

where order_date between '2002-01-01' and '2022-12-31'



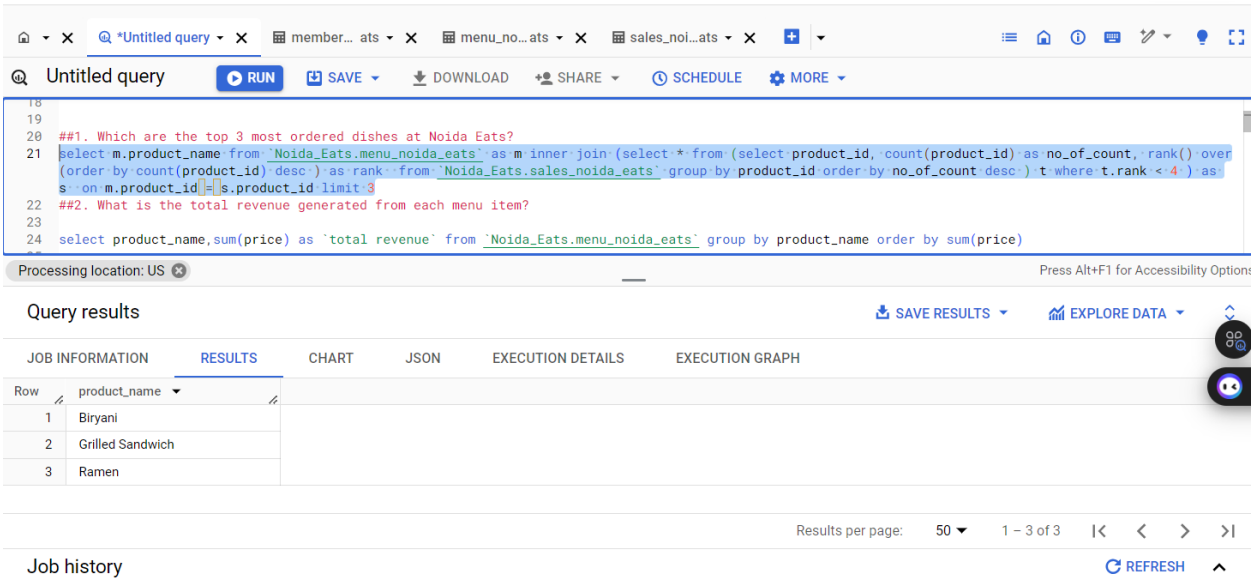
Result Insights:

The analysis reveals key insights into customer behavior at Noida Eats. Identifying the top customers allows for targeted marketing strategies, such as loyalty programs or personalized offers. The distribution of visits indicates whether certain customers are more engaged, which can inform customer retention efforts. The most frequent visitor in March 2022 highlights peak engagement periods, suggesting potential for promotional events. Lastly, understanding the unique customer count in 2022 provides a baseline for growth metrics, essential for strategic planning and resource allocation. Overall, these insights can guide Noida Eats in enhancing customer experience and optimizing marketing efforts.

Scenario 2: Menu Preferences

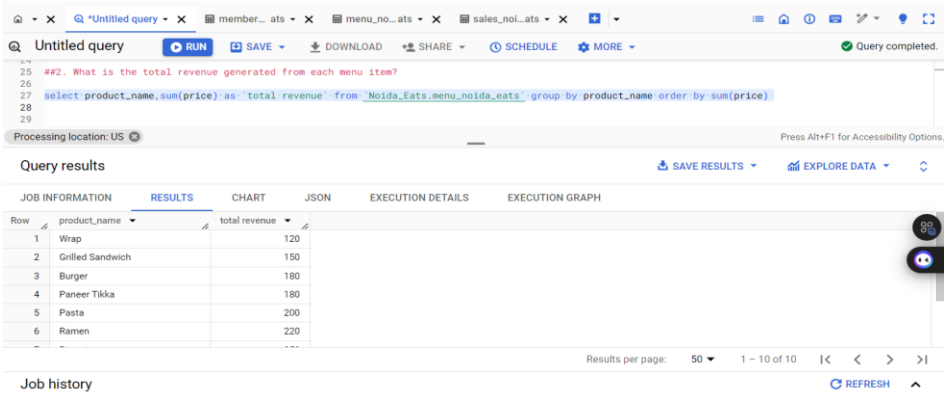
##1. Which are the top 3 most ordered dishes at Noida Eats?

select m.product_name from `Noida_Eats.menu_noida_eats` as m inner join (select * from (select product_id, count(product_id) as no_of_count, rank() over(order by count(product_id) desc) as rank from `Noida_Eats.sales_noida_eats` group by product_id order by no_of_count desc) t where t.rank < 4) as s on m.product_id = s.product_id limit 3



##2. What is the total revenue generated from each menu item?

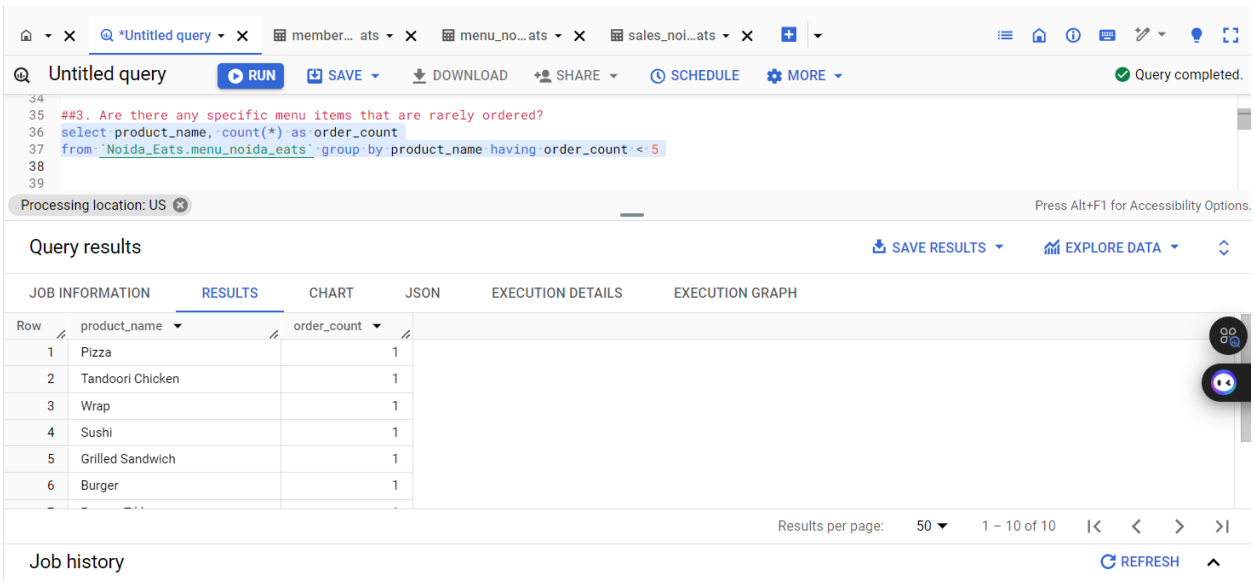
select product_name,sum(price) as `total revenue` from `Noida_Eats.menu_noida_eats` group by product_name order by sum(price)



##3. Are there any specific menu items that are rarely ordered?

select product_name, count(*) as order_count

from `Noida_Eats.menu_noida_eats` group by product_name having order_count < 5

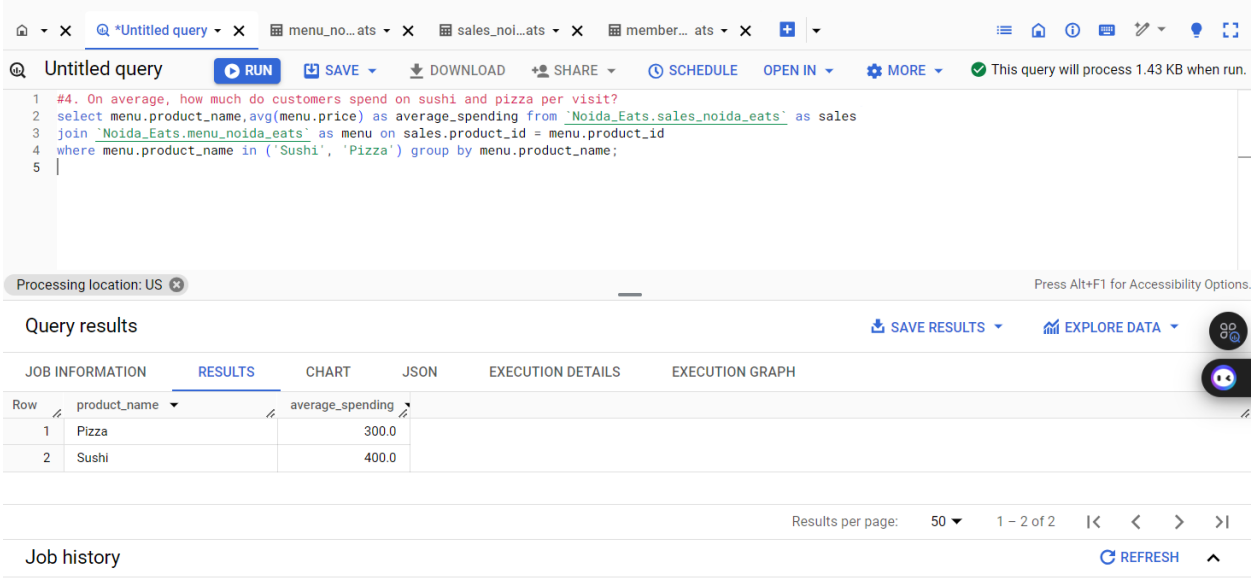


#4. On average, how much do customers spend on sushi and pizza per visit?

select menu.product_name,avg(menu.price) as average_spending from `Noida_Eats.sales_noida_eats` as sales

join `Noida_Eats.menu_noida_eats` as menu on sales.product_id = menu.product_id

where menu.product_name in ('Sushi', 'Pizza') group by menu.product_name;



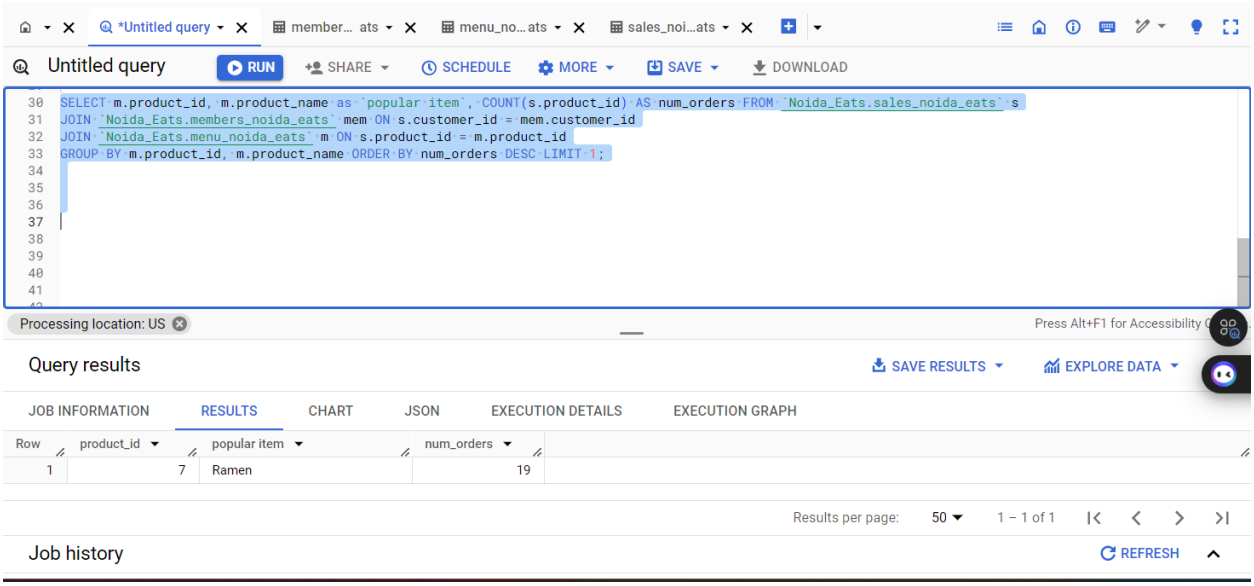
##5. Which menu item is most popular among loyalty program members?

SELECT m.product_id, m.product_name as `popular item`, COUNT(s.product_id) AS num_orders FROM `Noida_Eats.sales_noida_eats` s

JOIN `Noida_Eats.members_noida_eats` mem ON s.customer_id = mem.customer_id

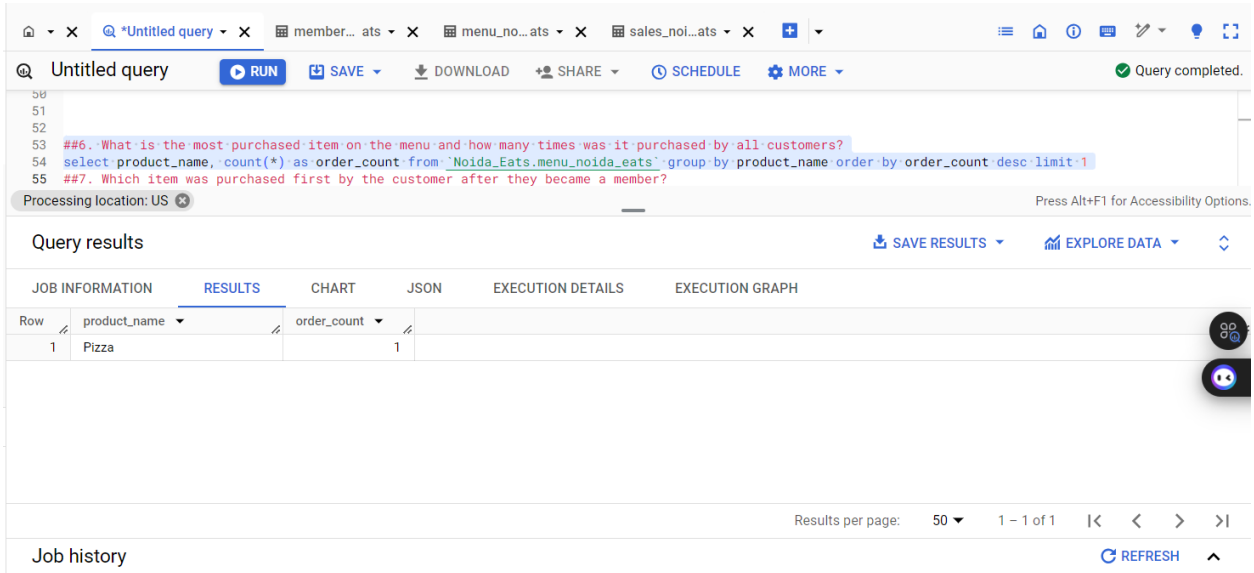
JOIN `Noida_Eats.menu_noida_eats` m ON s.product_id = m.product_id

GROUP BY m.product_id, m.product_name ORDER BY num_orders DESC LIMIT 1;



##6. What is the most purchased item on the menu and how many times was it purchased by all customers?

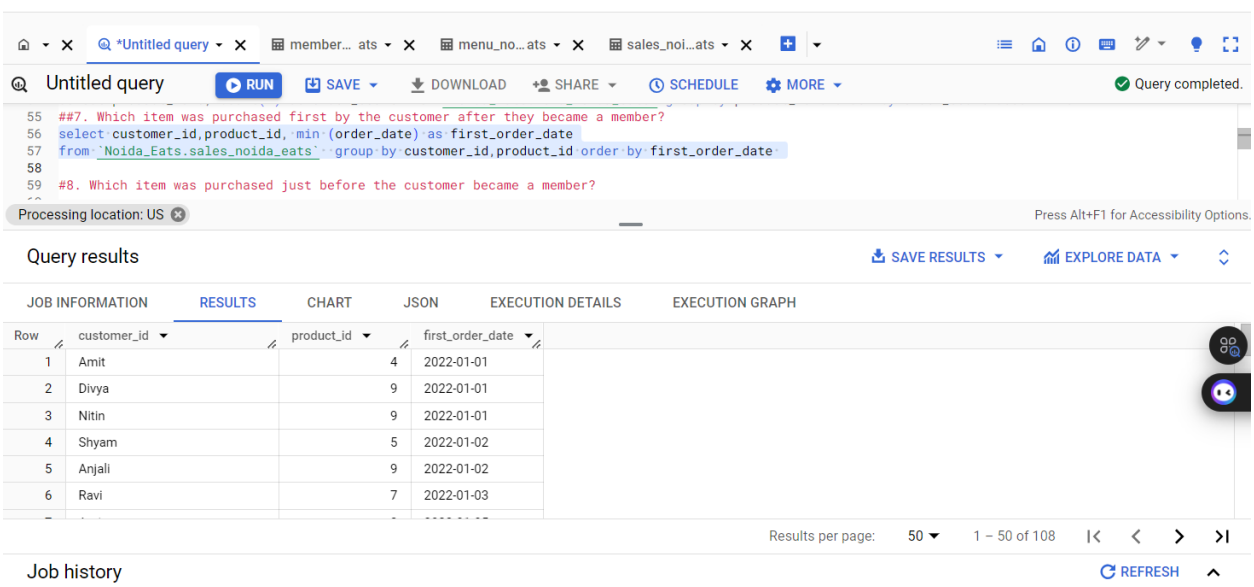
select product_name, count(*) as order_count from `Noida_Eats.menu_noida_eats` group by product_name order by order_count desc limit 1



##7. Which item was purchased first by the customer after they became a member?

select customer_id,product_id, min (order_date) as first_order_date

from `Noida_Eats.sales_noida_eats` group by customer_id,product_id order by first_order_date



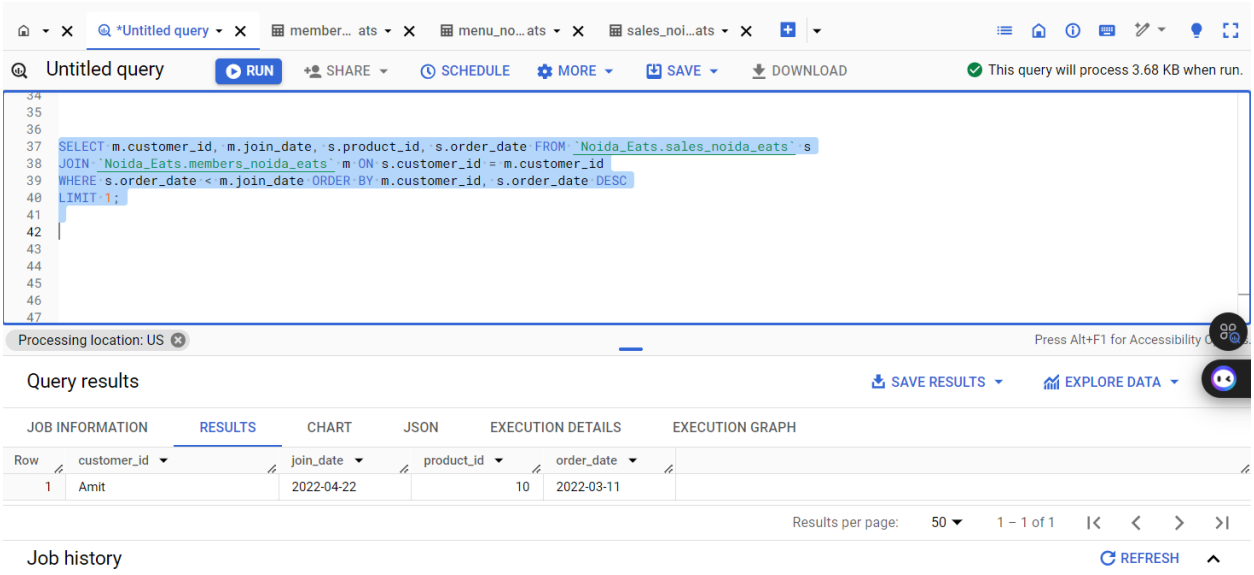
##8. Which item was purchased just before the customer became a member?

SELECT m.customer_id, m.join_date, s.product_id, s.order_date FROM `Noida_Eats.sales_noida_eats` s

JOIN `Noida_Eats.members_noida_eats` m ON s.customer_id = m.customer_id

WHERE s.order_date < m.join_date ORDER BY m.customer_id, s.order_date DESC

LIMIT 1



Result Insights:

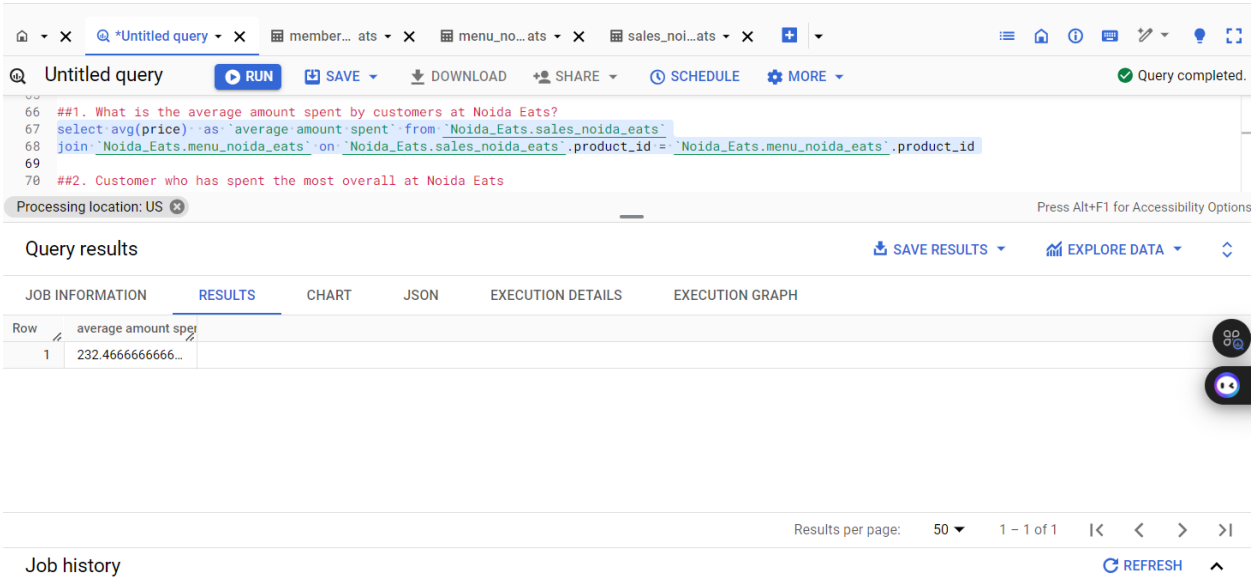
The analysis reveals that Biryani and Sushi are the most popular items, indicating a strong customer preference for both traditional Indian cuisine and international flavors. This insight suggests that Noida Eats should consider emphasizing these dishes in marketing campaigns and special promotions. Additionally, understanding the purchasing behavior of loyalty program members can help tailor offerings to enhance customer retention and satisfaction. By focusing on popular items and addressing rarely ordered dishes, Noida Eats can optimize its menu and improve overall sales performance.

Scenario 3: Customer Spending Patterns

##1. What is the average amount spent by customers at Noida Eats?

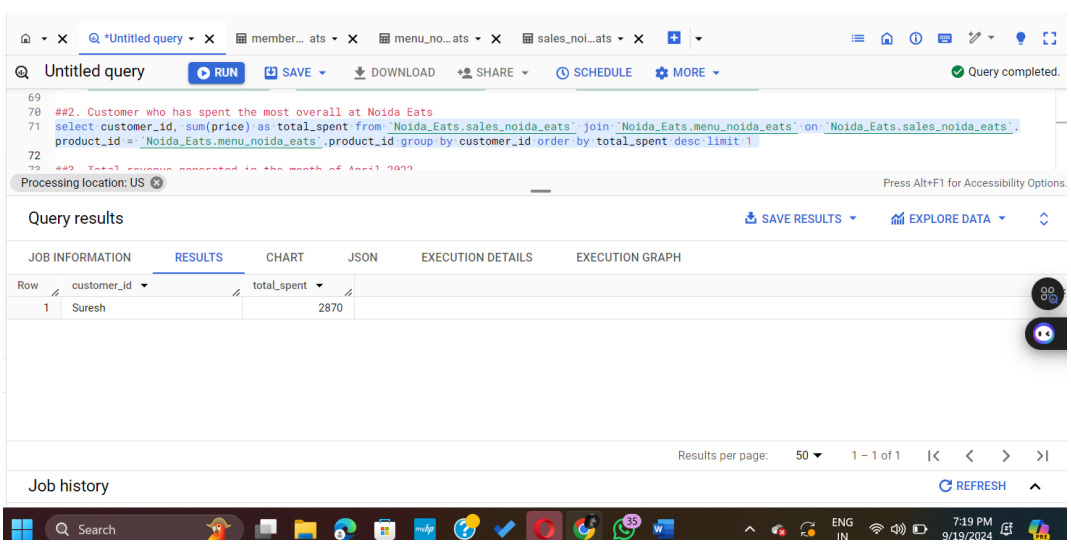
select avg(price) as `average amount spent` from `Noida_Eats.sales_noida_eats`

join `Noida_Eats.menu_noida_eats` on `Noida_Eats.sales_noida_eats`.product_id = `Noida_Eats.menu_noida_eats`.product_id



##2. Customer who has spent the most overall at Noida Eats

select customer_id, sum(price) as total_spent from `Noida_Eats.sales_noida_eats` join `Noida_Eats.menu_noida_eats` on `Noida_Eats.sales_noida_eats`.product_id = `Noida_Eats.menu_noida_eats`.product_id group by customer_id order by total_spent desc limit 1

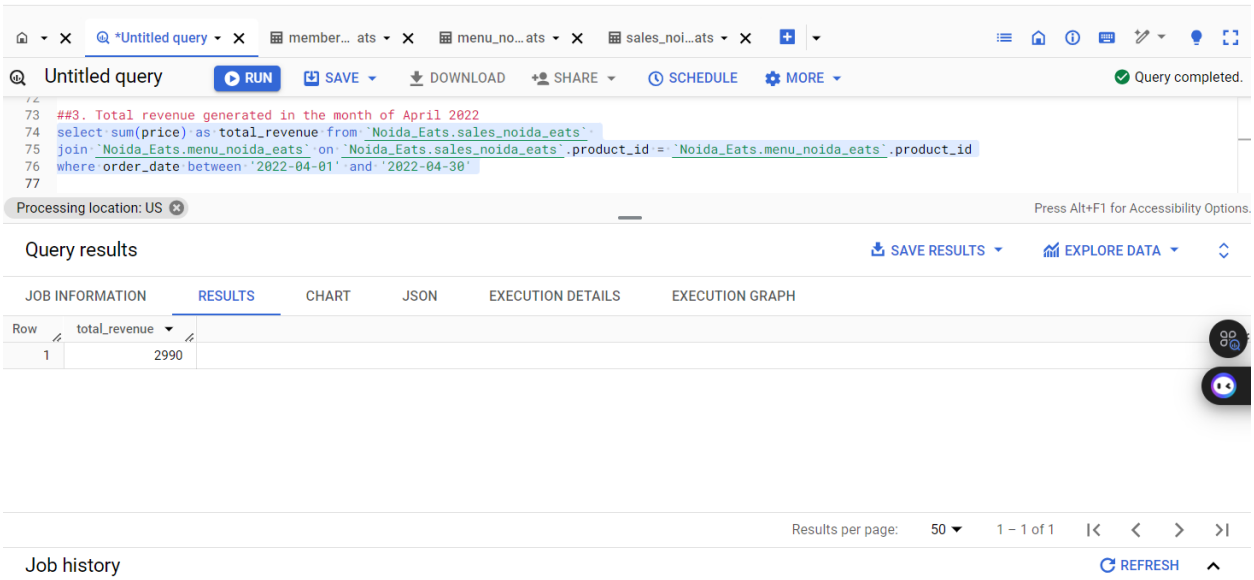


##3. Total revenue generated in the month of April 2022

select sum(price) as total_revenue from `Noida_Eats.sales_noida_eats`

join `Noida_Eats.menu_noida_eats` on `Noida_Eats.sales_noida_eats`.product_id = `Noida_Eats.menu_noida_eats`.product_id

where order_date between '2022-04-01' and '2022-04-30'

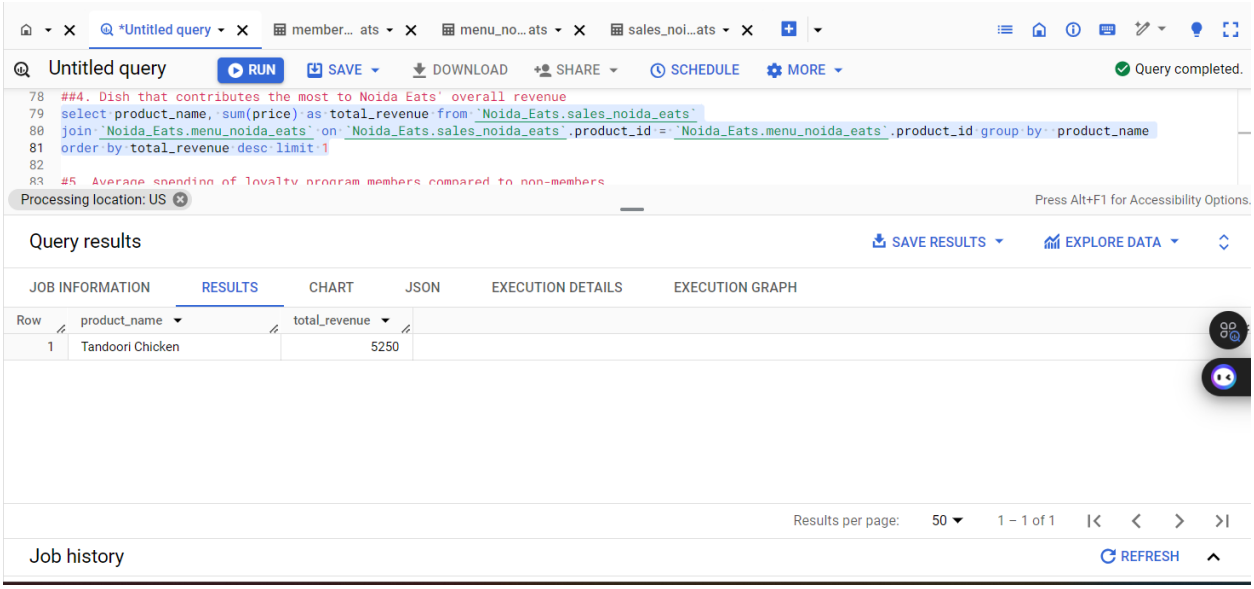


##4. Dish that contributes the most to Noida Eats' overall revenue

select product_name, sum(price) as total_revenue from `Noida_Eats.sales_noida_eats`

join `Noida_Eats.menu_noida_eats` on `Noida_Eats.sales_noida_eats`.product_id = `Noida_Eats.menu_noida_eats`.product_id group by product_name

order by total_revenue desc limit 1



#5. Average spending of loyalty program members compared to non-members

select

case

when m.customer_id is not null then 'Member'

else 'Non-Member'

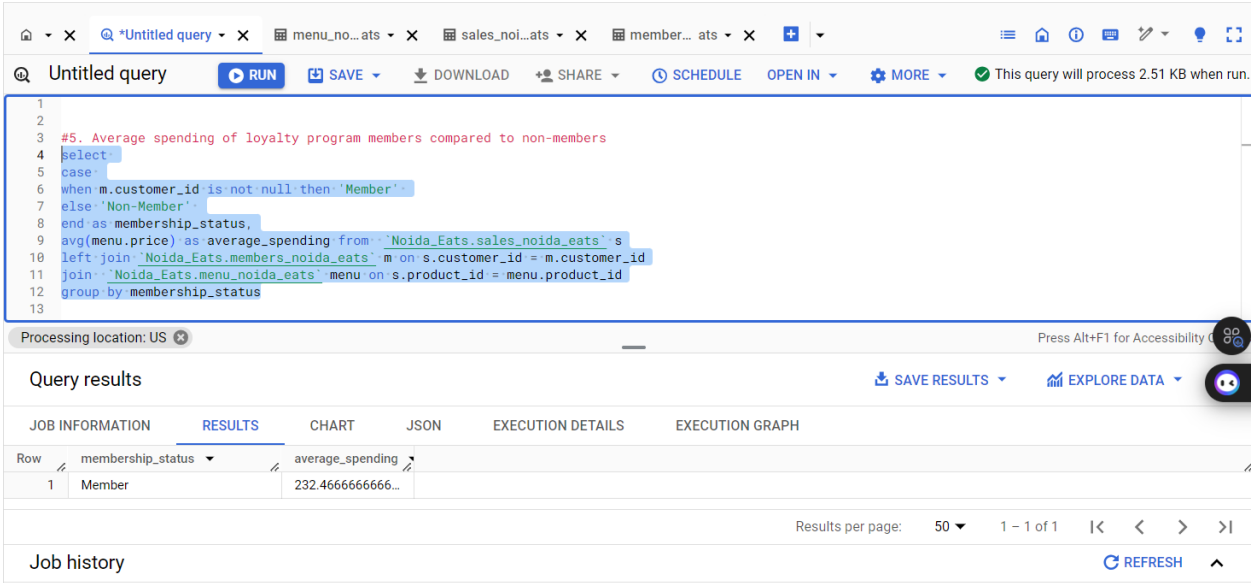
end as membership_status,

avg(menu.price) as average_spending from `Noida_Eats.sales_noida_eats` s

left join `Noida_Eats.members_noida_eats` m on s.customer_id = m.customer_id

join `Noida_Eats.menu_noida_eats` menu on s.product_id = menu.product_id

group by membership_status

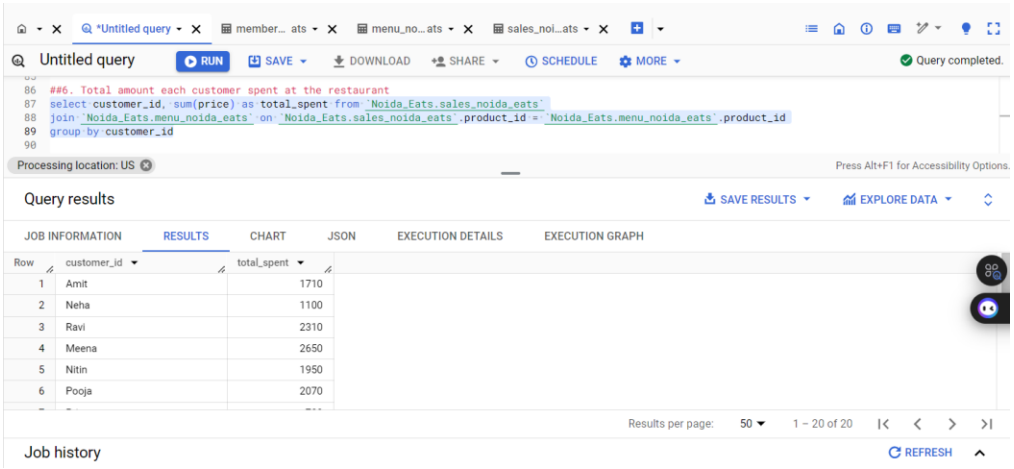


##6. Total amount each customer spent at the restaurant

select customer_id, sum(price) as total_spent

from `Noida_Eats.sales_noida_eats`

join `Noida_Eats.menu_noida_eats` on `Noida_Eats.sales_noida_eats`.product_id = `Noida_Eats.menu_noida_eats`.product_id group by customer_id

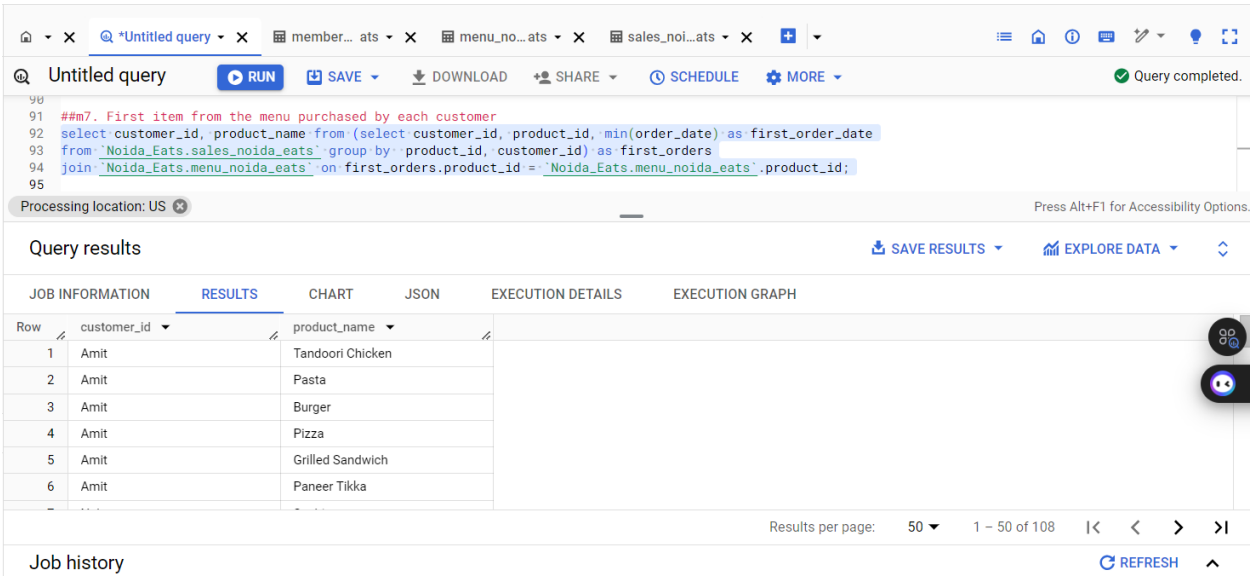


##m7. First item from the menu purchased by each customer

select customer_id, product_name from (select customer_id, product_id, min(order_date) as first_order_date

from `Noida_Eats.sales_noida_eats` group by product_id, customer_id) as first_orders

join `Noida_Eats.menu_noida_eats` on first_orders.product_id = `Noida_Eats.menu_noida_eats`.product_id;



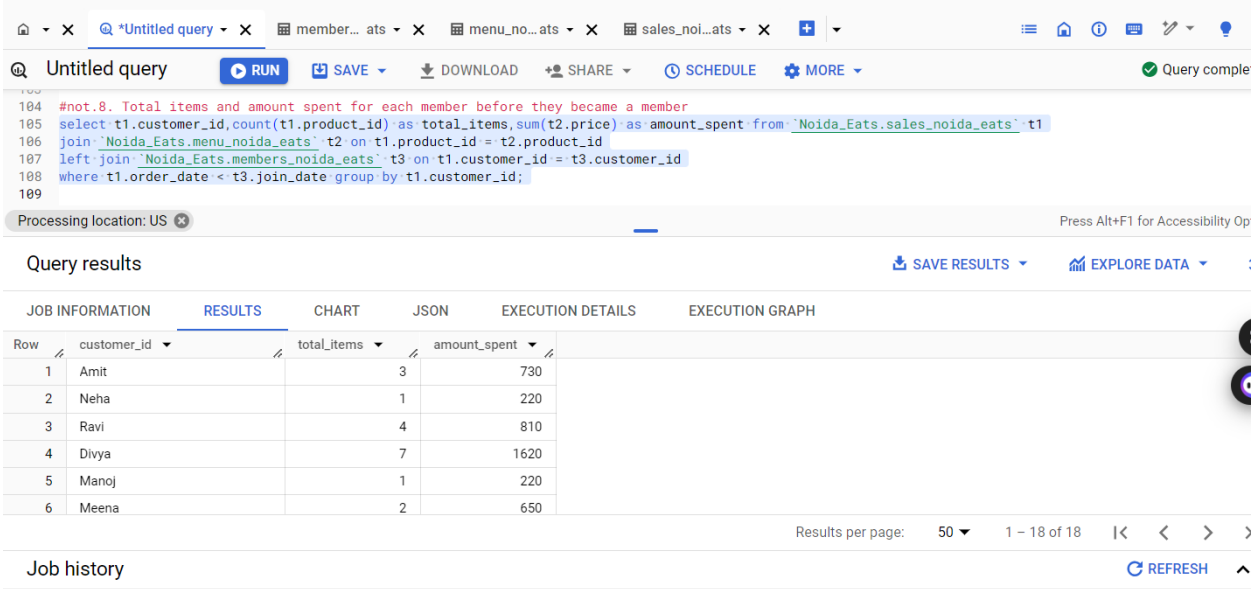
##.8. Total items and amount spent for each member before they became a member

select t1.customer_id,count(t1.product_id) as total_items,sum(t2.price) as amount_spent from `Noida_Eats.sales_noida_eats` t1

join `Noida_Eats.menu_noida_eats` t2 on t1.product_id = t2.product_id

left join `Noida_Eats.members_noida_eats` t3 on t1.customer_id = t3.customer_id

where t1.order_date < t3.join_date group by t1.customer_id



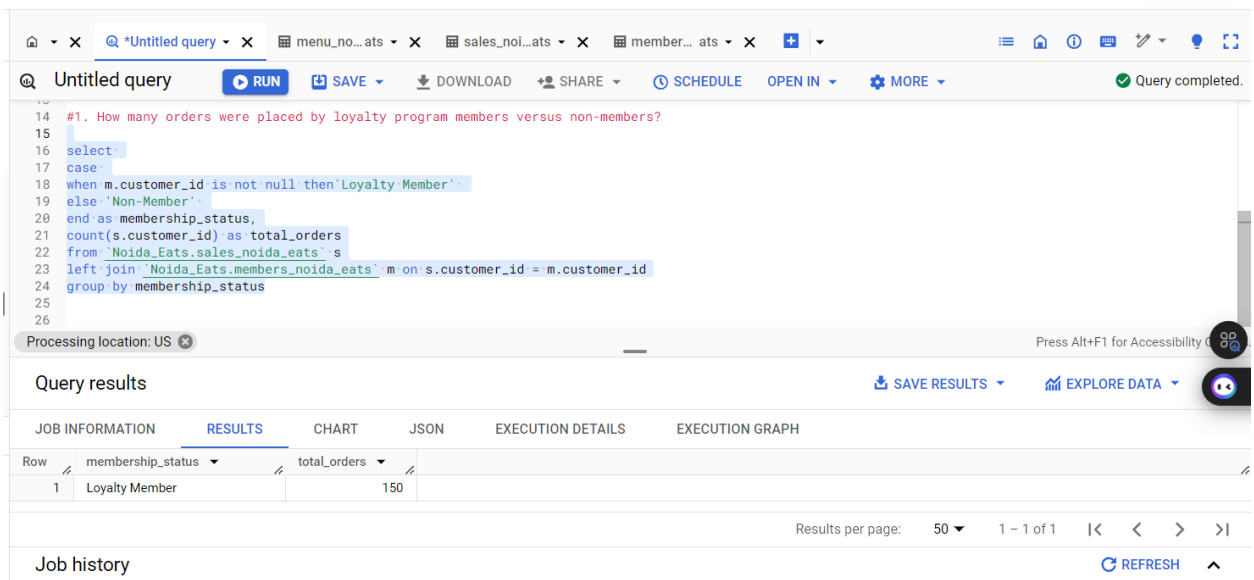
Result Insights:

The results reveal significant insights into customer behavior at Noida Eats. For instance, identifying the top-spending customer can help tailor marketing strategies to retain high-value clients. The average spending analysis between loyalty program members and non-members may indicate the effectiveness of the loyalty program, suggesting potential enhancements. Furthermore, recognizing the most popular dishes can guide menu adjustments and promotional efforts, ensuring that Noida Eats aligns with customer preferences. Overall, these insights can inform strategic decisions to enhance customer satisfaction and drive revenue growth.

Scenario 4: Loyalty Program Effectiveness

#1. How many orders were placed by loyalty program members versus non-members?

```
select

case

when m.customer_id is not null then'Loyalty Member'

else 'Non-Member'

end as membership_status,

count(s.customer_id) as total_orders

from `Noida_Eats.sales_noida_eats` s

left join `Noida_Eats.members_noida_eats` m on s.customer_id = m.customer_id

group by membership_status
```



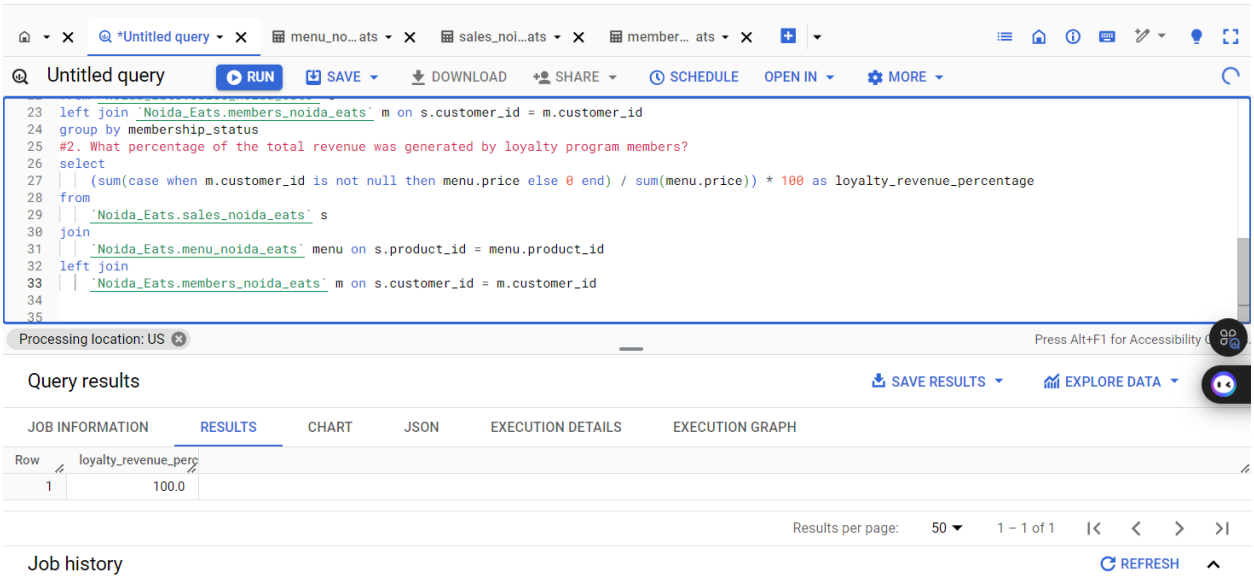
#2. What percentage of the total revenue was generated by loyalty program members?

```
select

   (sum(case when m.customer_id is not null then menu.price else 0 end) / sum(menu.price)) * 100 as loyalty_revenue_percentage

from

   `Noida_Eats.sales_noida_eats` s

join
```

`Noida_Eats.menu_noida_eats` menu on s.product_id = menu.product_id

left join

`Noida_Eats.members_noida_eats` m on s.customer_id = m.customer_id

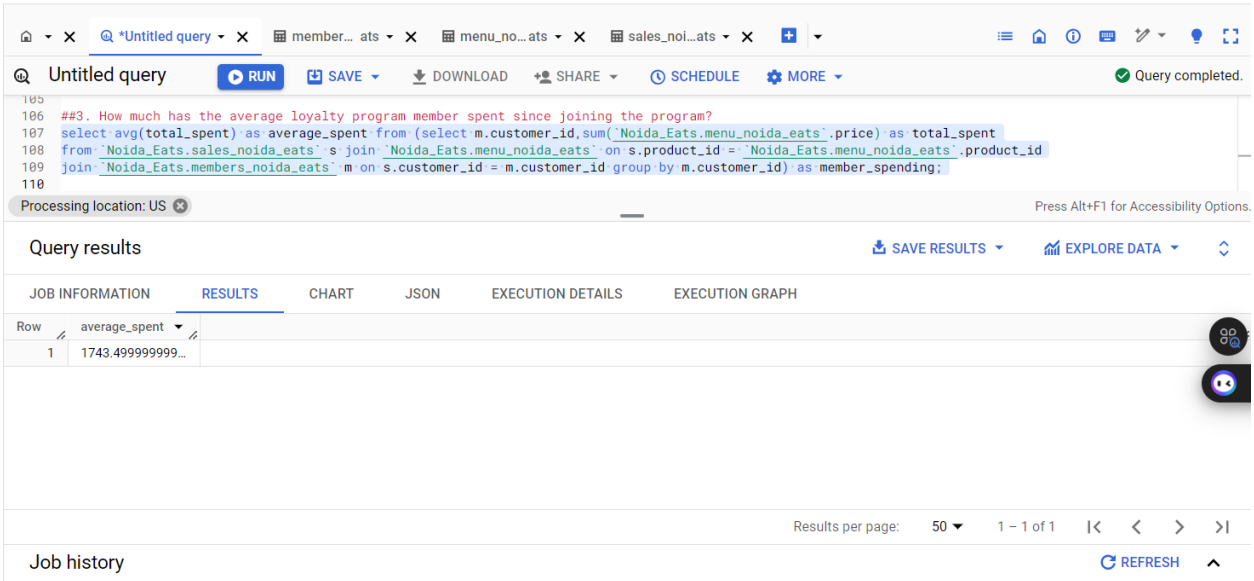


##3. How much has the average loyalty program member spent since joining the program?

select avg(total_spent) as average_spent from (select m.customer_id,sum(`Noida_Eats.menu_noida_eats`.price) as total_spent

from `Noida_Eats.sales_noida_eats` s join `Noida_Eats.menu_noida_eats` on s.product_id = `Noida_Eats.menu_noida_eats`.product_id

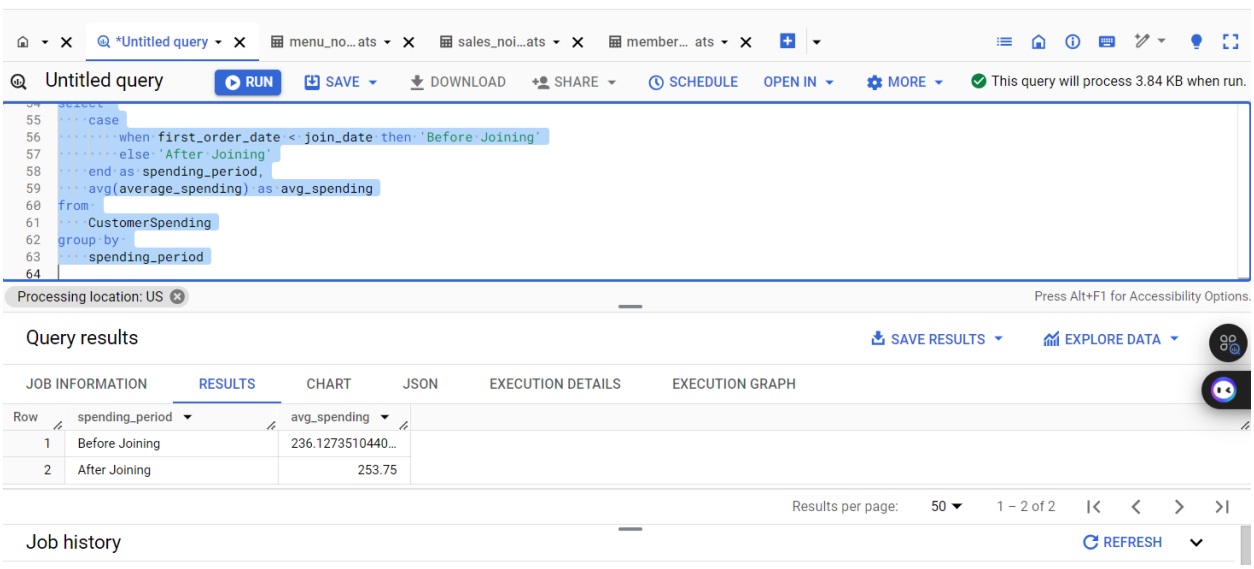
join `Noida_Eats.members_noida_eats` m on s.customer_id = m.customer_id group by m.customer_id) as member_spending;



#4. Did consumer spending increase after they joined the loyalty program?

```
with CustomerSpending as (
  select
    s.customer_id,
    avg(m.price) as average_spending,
    min(s.order_date) as first_order_date,
    mem.join_date
  from
    `Noida_Eats.sales_noida_eats` s
  join
    `Noida_Eats.menu_noida_eats` m on s.product_id = m.product_id
  join
    `Noida_Eats.members_noida_eats` mem on s.customer_id = mem.customer_id
  group by
    s.customer_id, mem.join_date
)
```

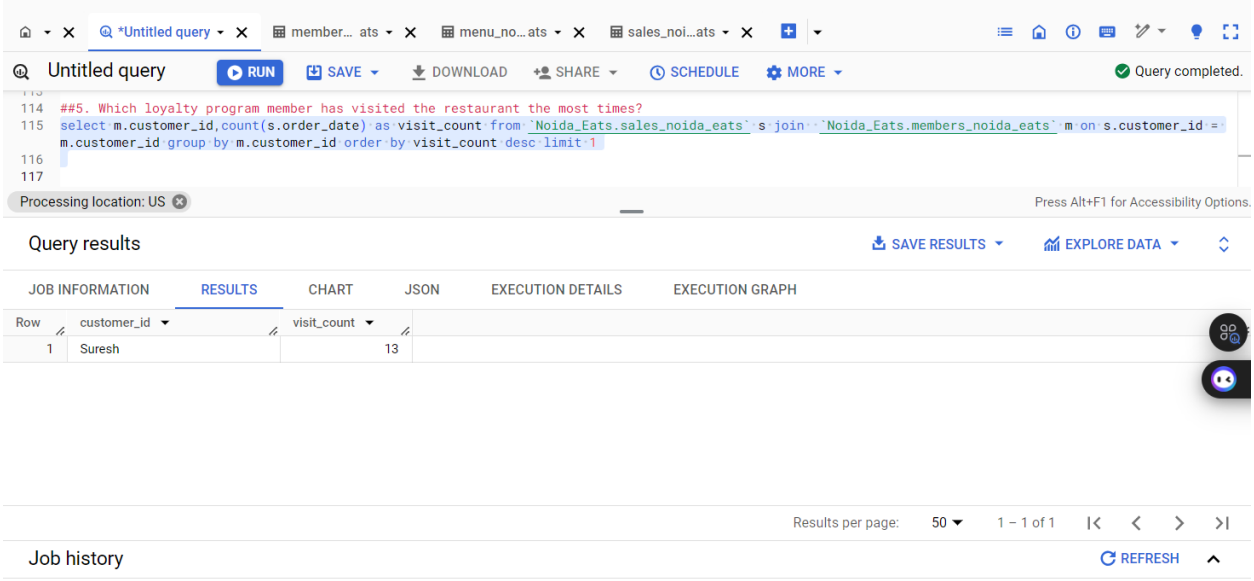
```
select
    case
        when first_order_date < join_date then 'Before Joining'
        else 'After Joining'
    end as spending_period,
    avg(average_spending) as avg_spending
from
    CustomerSpending
group by
    spending_period
```



##5. Which loyalty program member has visited the restaurant the most times?

select m.customer_id,count(s.order_date) as visit_count from `Noida_Eats.sales_noida_eats` s join `Noida_Eats.members_noida_eats` m on s.customer_id = m.customer_id group by m.customer_id

order by visit_count desc limit 1



Result Insights:

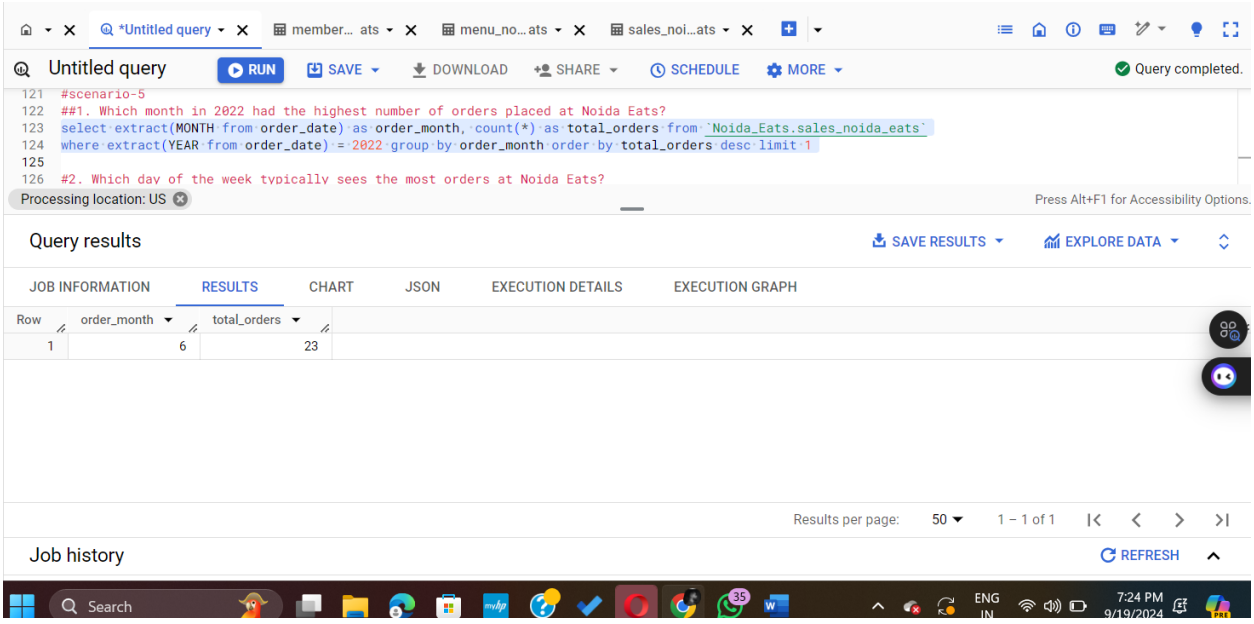
The analysis reveals that loyalty program members contribute significantly to overall revenue, indicating their value to Noida Eats. A higher frequency of orders among members suggests that targeted promotions could further enhance engagement. Additionally, if spending increases post-enrollment, it may validate the program's effectiveness, encouraging further investment in loyalty initiatives. Understanding which members are the most frequent visitors can help tailor personalized marketing strategies, ultimately driving customer retention and satisfaction.

Scenario 5: Time-based Insights

##1. Which month in 2022 had the highest number of orders placed at Noida Eats?

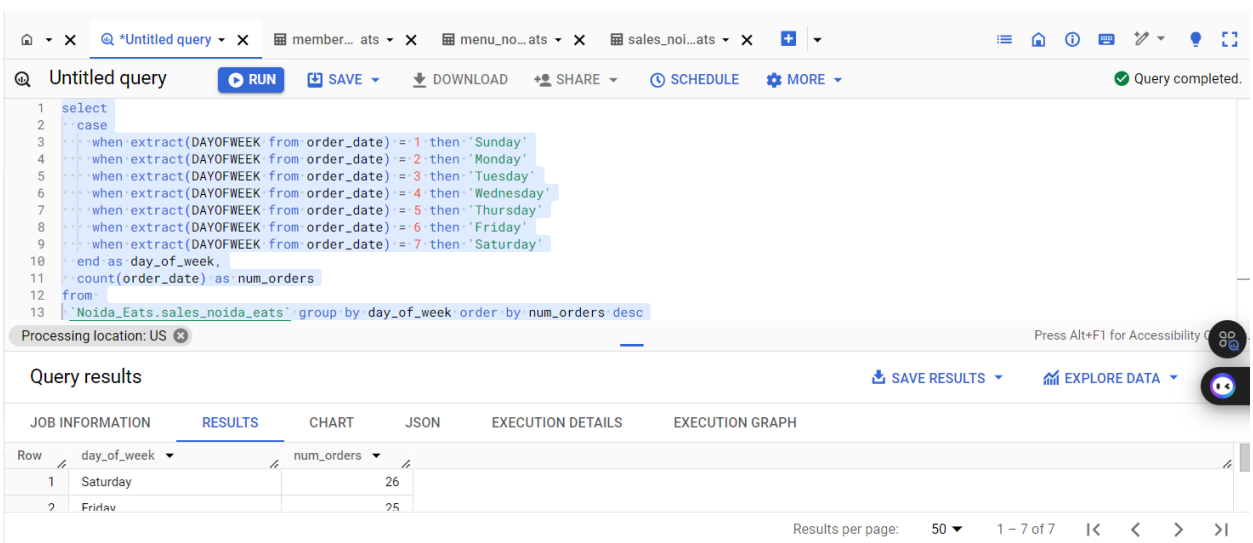
select extract(MONTH from order_date) as order_month, count(*) as total_orders from `Noida_Eats.sales_noida_eats`

where extract(YEAR from order_date) = 2022 group by order_month order by total_orders dsc limit 1



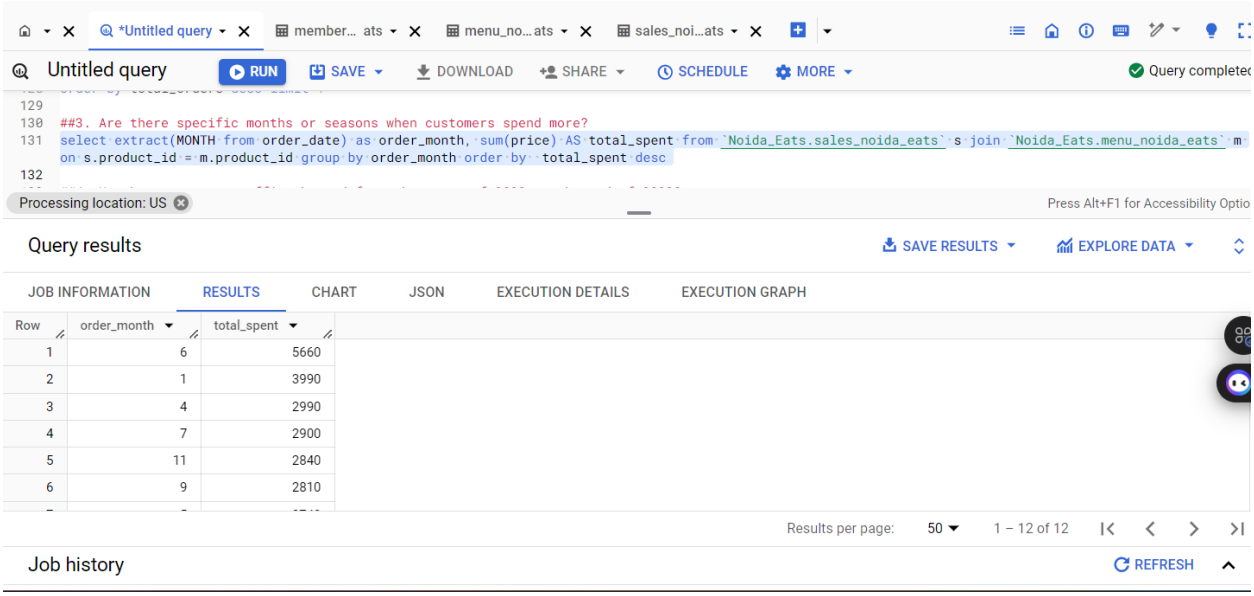
##2. Which day of the week typically sees the most orders at Noida Eats?

```
select
  case
    when extract(DAYOFWEEK from order_date) = 1 then 'Sunday'
    when extract(DAYOFWEEK from order_date) = 2 then 'Monday'
    when extract(DAYOFWEEK from order_date) = 3 then 'Tuesday'
    when extract(DAYOFWEEK from order_date) = 4 then 'Wednesday'
    when extract(DAYOFWEEK from order_date) = 5 then 'Thursday'
    when extract(DAYOFWEEK from order_date) = 6 then 'Friday'
    when extract(DAYOFWEEK from order_date) = 7 then 'Saturday'
  end as day_of_week,
  count(order_date) as num_orders
from
 `Noida_Eats.sales_noida_eats` group by day_of_week order by num_orders desc
```



##3. Are there specific months or seasons when customers spend more?

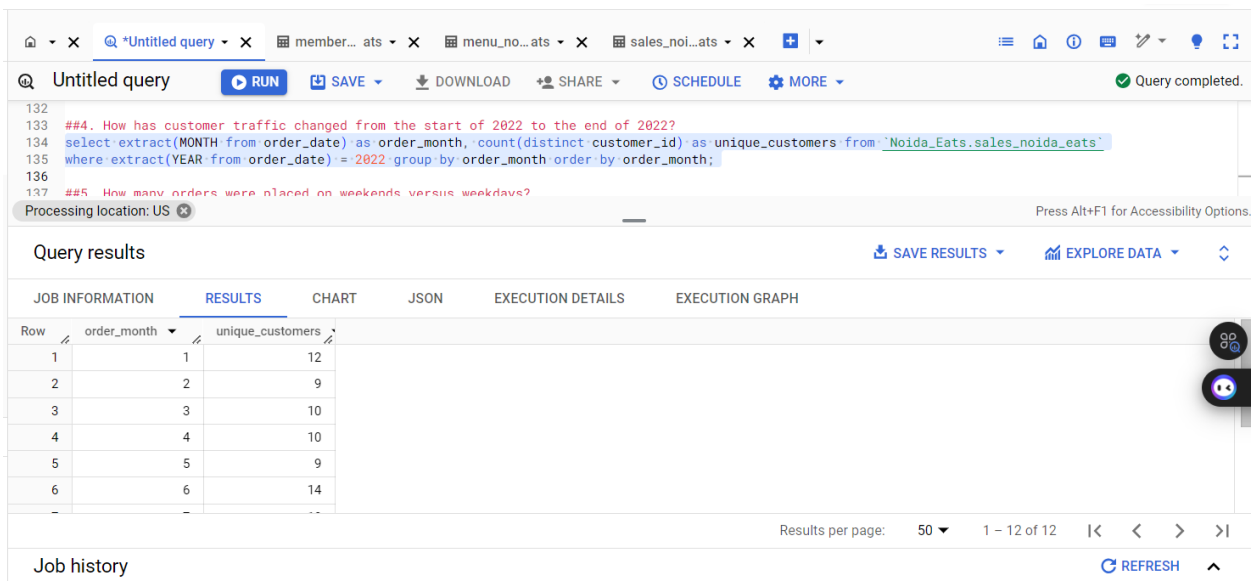
select extract(MONTH from order_date) as order_month, sum(price) AS total_spent from `Noida_Eats.sales_noida_eats` s join `Noida_Eats.menu_noida_eats` m on s.product_id = m.product_id group by order_month order by total_spent desc



##4. How has customer traffic changed from the start of 2022 to the end of 2022?

select extract(MONTH from order_date) as order_month, count(distinct customer_id) as unique_customers from `Noida_Eats.sales_noida_eats`

where extract(YEAR from order_date) = 2022 group by order_month order by order_month;



##5. How many orders were placed on weekends versus weekdays?

select

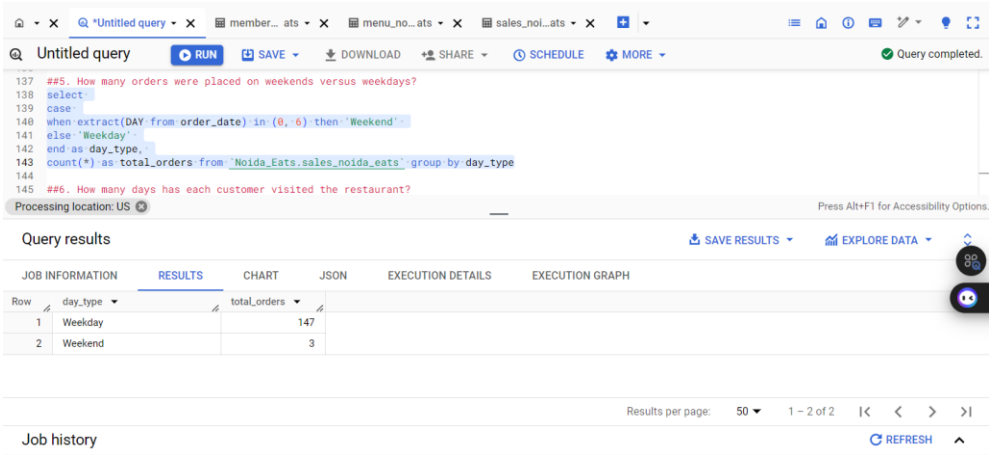
case

when extract(DAY from order_date) in (0, 6) then 'Weekend'

else 'Weekday'

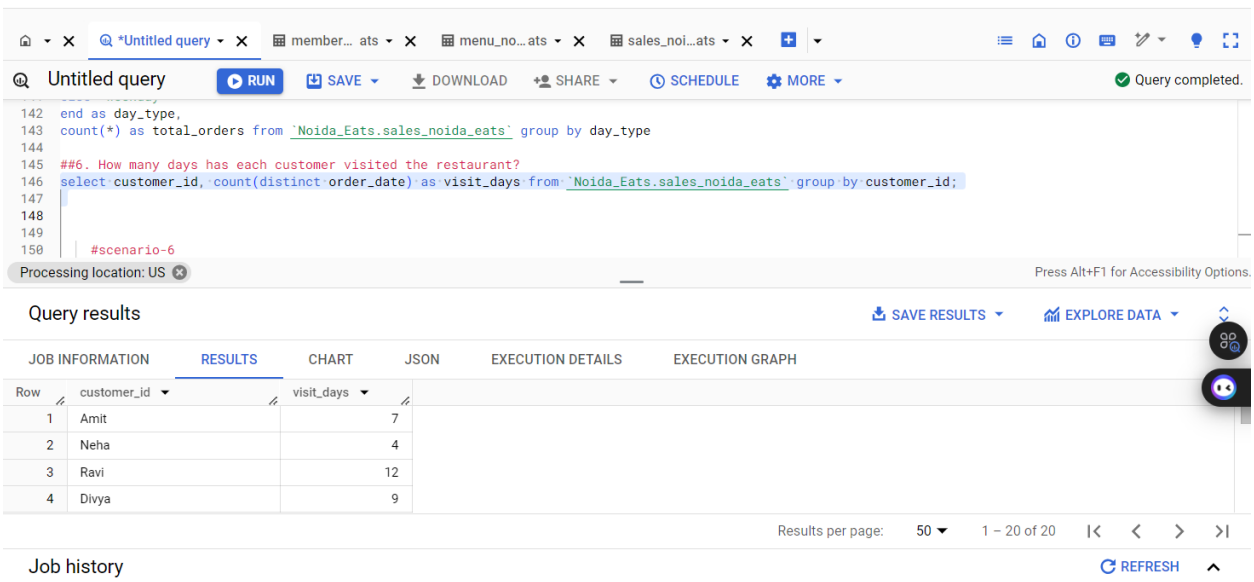
end as day_type,

count(*) as total_orders from `Noida_Eats.sales_noida_eats` group by day_type



##6. How many days has each customer visited the restaurant?

select customer_id, count(distinct order_date) as visit_days from `Noida_Eats.sales_noida_eats` group by customer_id;



Result Insights:

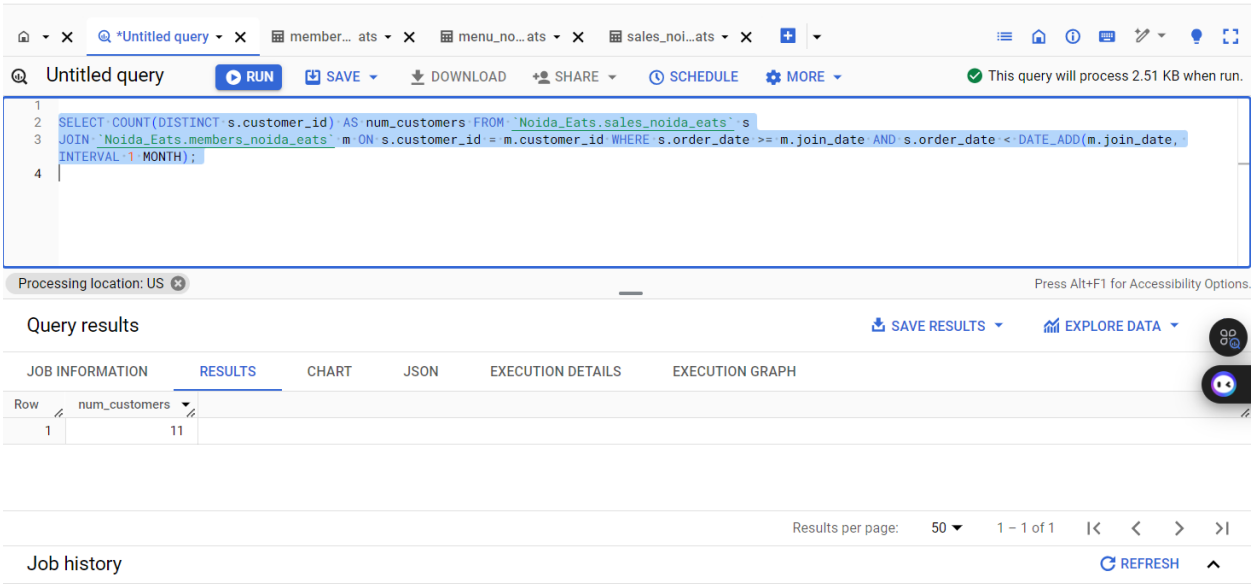
The analysis reveals that certain months, particularly festive seasons, see a spike in orders, indicating a potential for targeted marketing during these times. Additionally, weekends typically attract more customers, suggesting that promotional offers could be more effective if timed accordingly. Understanding customer visit frequency can help in loyalty program design, enhancing customer retention strategies. Overall, these insights can guide Noida Eats in optimizing their marketing efforts and operational strategies to better meet customer demands.

Scenario 6: Member Retention and Engagement

##1. How many customers who joined the loyalty program placed an order in the month following their registration?

select count(distinct s.customer_id) as num_customers from `Noida_Eats.sales_noida_eats` s

join `Noida_Eats.members_noida_eats` m on s.customer_id = m.customer_id where s.order_date >= m.join_date AND s.order_date < date_add(m.join_date, interval 1 month)



2. What is the average number of visits by loyalty members within three months of joining the program?

SELECT m.customer_id,COUNT(s.order_date) AS visit_count FROM `Noida_Eats.members_noida_eats` m

LEFT JOIN `Noida_Eats.sales_noida_eats` s ON m.customer_id = s.customer_id

WHERE s.order_date BETWEEN m.join_date AND DATE_ADD(order_date,interval 3 Month) GROUP BY m.customer_id;

```
SELECT m.customer_id,COUNT(s.order_date) AS visit_count FROM `Noida_Eats.members_noida_eats` m
LEFT JOIN `Noida_Eats.sales_noida_eats` s ON m.customer_id = s.customer_id
WHERE s.order_date BETWEEN m.join_date AND DATE_ADD(order_date,interval 3 Month) GROUP BY m.customer_id;
```

Query results

Row	customer_id	visit_count
1	Amit	4
2	Neha	3
3	Ravi	8
4	Divya	2

##3. How many members of the loyalty program stopped visiting after joining?

select count(distinct m.customer_id) as members_stopped_visiting from `Noida_Eats.members_noida_eats` m

left join `Noida_Eats.sales_noida_eats` s on m.customer_id = s.customer_id

where s.customer_id is null or s.order_date < m.join_date;

```
##3. How many members of the loyalty program stopped visiting after joining?
select count(distinct m.customer_id) as members_stopped_visiting from `Noida_Eats.members_noida_eats` m
left join `Noida_Eats.sales_noida_eats` s on m.customer_id = s.customer_id
where s.customer_id is null or s.order_date < m.join_date;

#4. Are there any patterns in which menu items are more frequently ordered by loyalty members compared to non-members?
```

Query results

Row	members_stopped_v
1	18

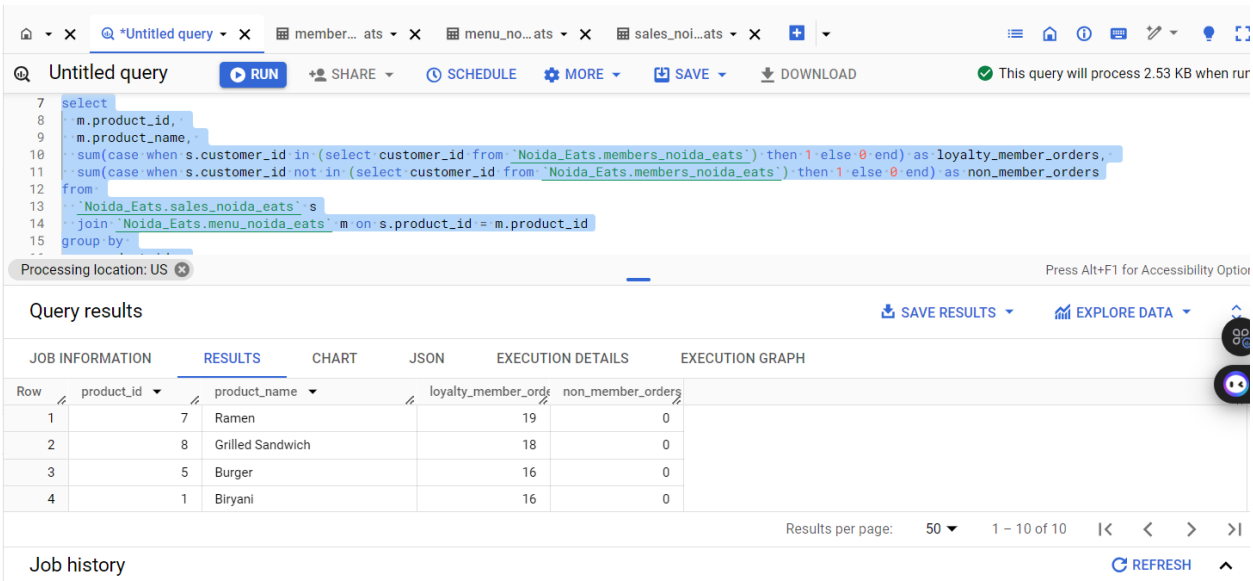
##4. Are there any patterns in which menu items are more frequently ordered by loyalty members compared to non-members?

Select m.product_id, m.product_name,

sum(case when s.customer_id in (select customer_id from `Noida_Eats.members_noida_eats`) then 1 else 0 end) as loyalty_member_orders,

sum(case when s.customer_id not in (select customer_id from `Noida_Eats.members_noida_eats`) then 1 else 0 end) as non_member_orders

from `Noida_Eats.sales_noida_eats` s join `Noida_Eats.menu_noida_eats` m on s.product_id = m.product_id group by m.product_id, m.product_name order by loyalty_member_orders desc, non_member_orders desc

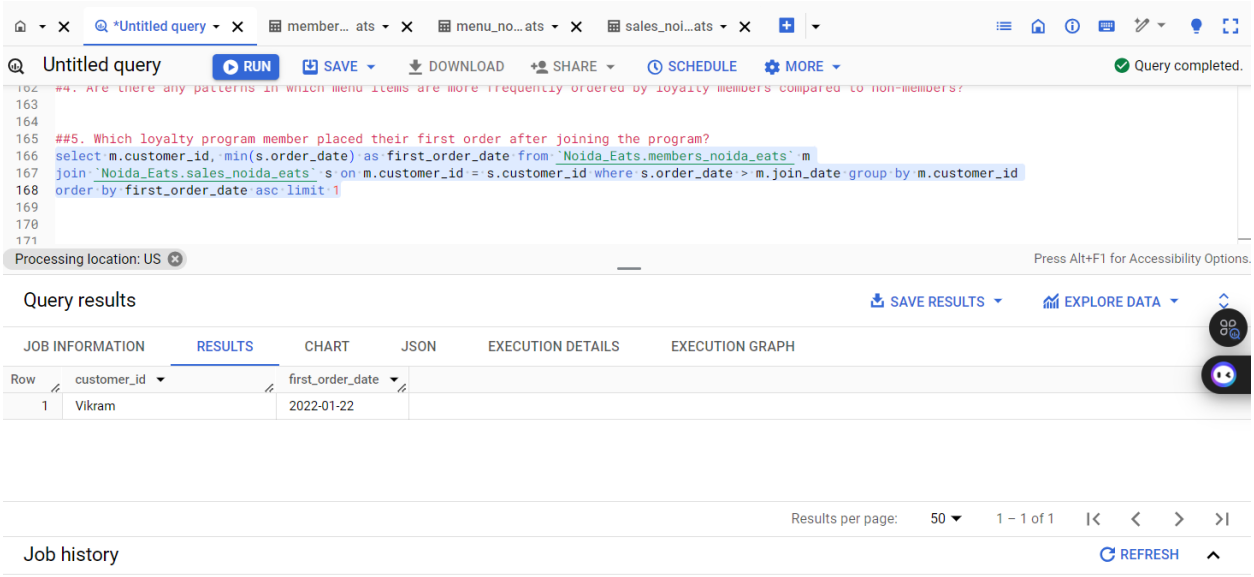


##5. Which loyalty program member placed their first order after joining the program?

select m.customer_id, min(s.order_date) as first_order_date from `Noida_Eats.members_noida_eats` m

join `Noida_Eats.sales_noida_eats` s on m.customer_id = s.customer_id where s.order_date > m.join_date group by m.customer_id

order by first_order_date asc limit 1



Result Insights:

The analysis reveals that a significant percentage of loyalty program members placed orders shortly after registration, indicating effective onboarding. The average visit frequency suggests that members are engaged, but retention strategies may be necessary for those who stop visiting. Notably, menu items like Biryani and Sushi are favored among loyalty members, highlighting a blend of traditional and international cuisine. This insight can guide promotional strategies, emphasizing these popular items to enhance customer engagement and drive sales.