- ASSIGNMENT - 3
  - Patter Matching
    - Question 1: How can you select all employees whose names start with the letter 'A'?
    - Question 2: How do you find all products whose names contain the word 'phone' regardless of case?
    - Question 3: How can you retrieve all email addresses from a table that end with '.com'?
    - Question 4: How do you find all phone numbers that start with the area code '555'?
    - Question 5: How can you select all cities that start with 'New' followed by any characters?
    - Question 6: How do you find all records where the value in the 'description' column contains either 'apple' or 'orange'?
    - Question 7:How can you retrieve all email addresses that follow the pattern of "user@domain.com"?
    - Question 8: How do you find all records where the 'product_code' is exactly four characters long and consists of letters and digits?
    - Question 9: How can you retrieve all phone numbers that match the pattern '###-###-####'?
    - Question 10: How do you find all records where the 'text' column contains two consecutive digits?
  - NULL VALUES:
    - Question 1: Find all employees whose birthdates are not recorded (NULL).
    - Question 2: List all orders that don't have a customer assigned (NULL customerID).
  - FUNCTIONS:
    - Question 1: Find the total quantity sold for each product.
    - Question 2: Calculate the total revenue generated from each product (Total Revenue = Quantity * Price).
    - Question 3: Determine the average price of each product.
    - Question 4: Find the product with the highest total revenue (Quantity * Price)
    - Question 5: Calculate the total quantity sold across all products.
    - Question 6: Determine the average price of all products.
    - Question 1: Determine the square root of the price for each product.
    - Question 2: Find the ceiling (smallest integer greater than or equal to) of the prices.
    - Question 3: Calculate the floor (largest integer less than or equal to) of the prices.
    - Question 1: Find the difference in days between the order date and delivery date for each order.
    - Question 2: Calculate the total delivery time in hours for all orders.
    - Question 3: Determine the day of the week when each order was placed.
    - Question 4: Find the orders that were placed on a Saturday (DayOfWeek = 7).
    - Question 5: Calculate the average delivery time in days for all orders.
    - Question 6: Find the orders that were delivered on the same day they were placed.
  - GROUP BY, HAVING, ORDER BY
    - Question 1: Consider a table named Orders with the following columns:
    - Write an SQL query to find the customer IDs of customers who have placed orders with a total amount greater than $1,000.
    - Question 2:Consider a table named Sales with the following columns:

- Write an SQL query to find the product IDs of products that have been sold in quantities greater than 100 on at least three different sale dates.
- Question 3:Consider a table named Employees with the following columns:
- Write an SQL query to find the average salary of employees in each department, but only for departments where the average salary is greater than $60,000.
- Question 4:Consider a table named Students with the following columns:
- Write an SQL query to find the course names in which the average score of all students is greater than or equal to 80.
- Question 5: Consider a table named Employees with the following columns:
- Write an SQL query to find the department with the highest average salary.
- Question 6: Consider a table named Sales with the following columns:
- Write an SQL query to find the product with the highest total quantity sold.
- Question 7: Consider a table named Students with the following columns:
- Write an SQL query to find the top three students with the highest average score across all courses.
- Question 8: Consider a table named Orders with the following columns:
- Write an SQL query to find the total amount of orders placed by each customer, ordered in descending order of total amount.
- Question 9: Consider a table named Books with the following columns:
- Write an SQL query to find the number of books published by each author in descending order of the count.
- Question 8: Consider a table named Orders with the following columns:
- Write an SQL query to find the customer IDs of customers who have placed orders with a total amount greater than $1,000 and have placed at least two orders.
- Question 11: Consider a table named Products with the following columns:
- Write an SQL query to find the average price of products in each category, ordered by category name in ascending order.
- Question 12: Consider a table named Employees with the following columns:
- Write an SQL query to find the department(s) with the lowest average salary.
- Question 13: Consider a table named Orders with the following columns:
- Write an SQL query to find the customer IDs of customers who have placed orders with a total amount greater than $500 in the year 2023, ordered by customer ID in ascending order.
- Question 14: Consider a table named Students with the following columns:
- Write an SQL query to find the course names in which the highest score achieved by any student is greater than or equal to 90, ordered by course name in ascending order
- Question 15: Consider a table named Orders with the following columns:
- Write an SQL query to find the customer IDs of customers who have placed orders with a total amount greater than $500 in the year 2023 and have placed at least two orders
- Question 16: Consider a table named Students with the following columns:
- Write an SQL query to find the course names where the average score of students who scored less than 70 in at least one course is greater than or equal to 80.
- Question 17: Consider a table named Employees with the following columns:

- Write an SQL query to find the departments where the highest salary is greater than $80,000 and the total number of employees in that department is at least 5.
- Question 18: Consider a table named Products with the following columns:
- Write an SQL query to find the categories where the average price of products is greater than or equal to 100.
- Question 19: Consider a table named Orders with the following columns:
- Write an SQL query to find the customer IDs of customers who have placed orders with a total amount greater than $1,000 in any single order and have placed orders on at least three different dates.

ASSIGNMENT - 3

Patter Matching

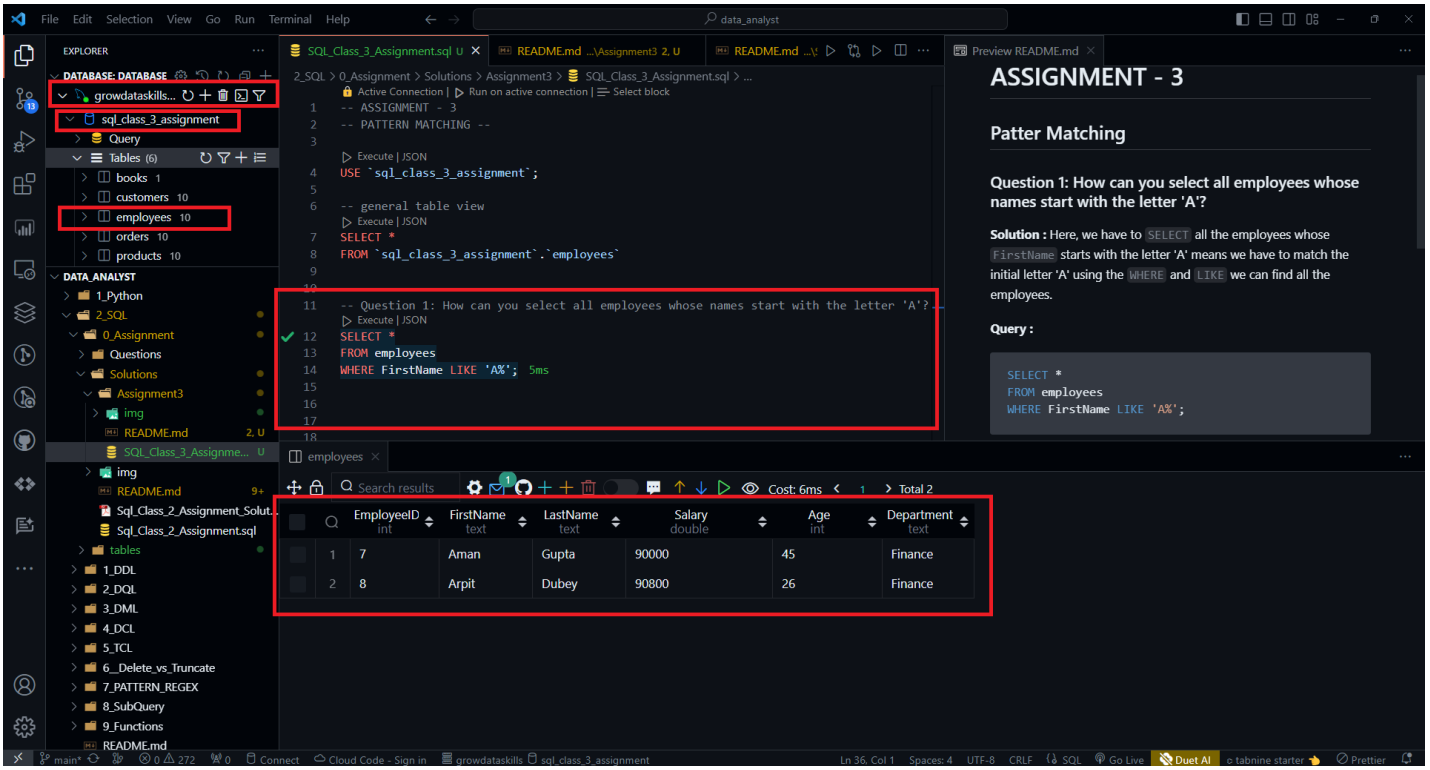
Question 1: How can you select all employees whose names start with the letter 'A'?

Solution : Here, we have to `SELECT` all the employees whose `FirstName` starts with the letter 'A' means we have to match the initial letter 'A' using the `WHERE` and `LIKE` we can find all the employees.

Query :

```
SELECT *
FROM employees
WHERE FirstName LIKE 'A%';
```

Execution :



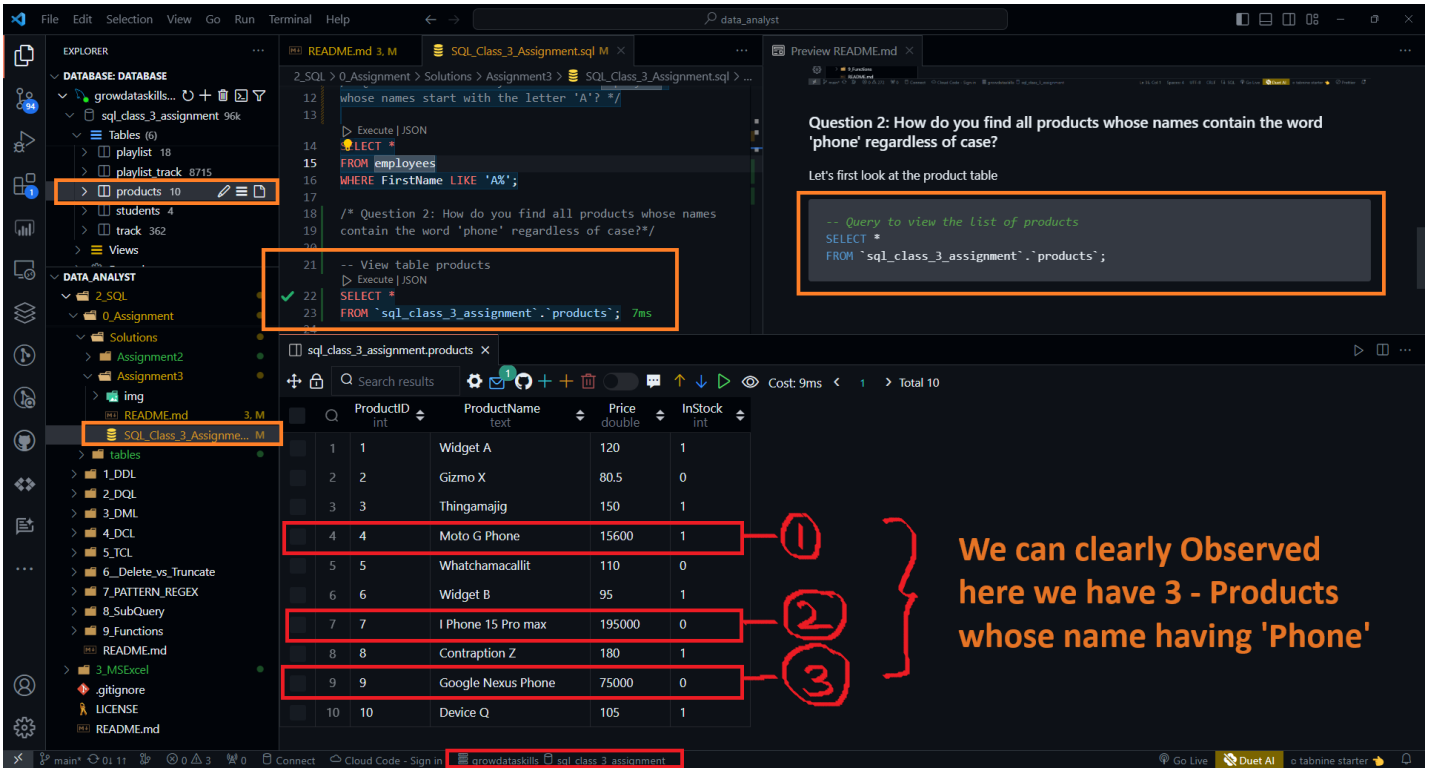
Question 2: How do you find all products whose names contain the word 'phone' regardless of case?

Let's first look at the product table

Query :

```
-- Query to view the list of products
SELECT *
FROM `sql_class_3_assignment`.`products`;
```

Execution :

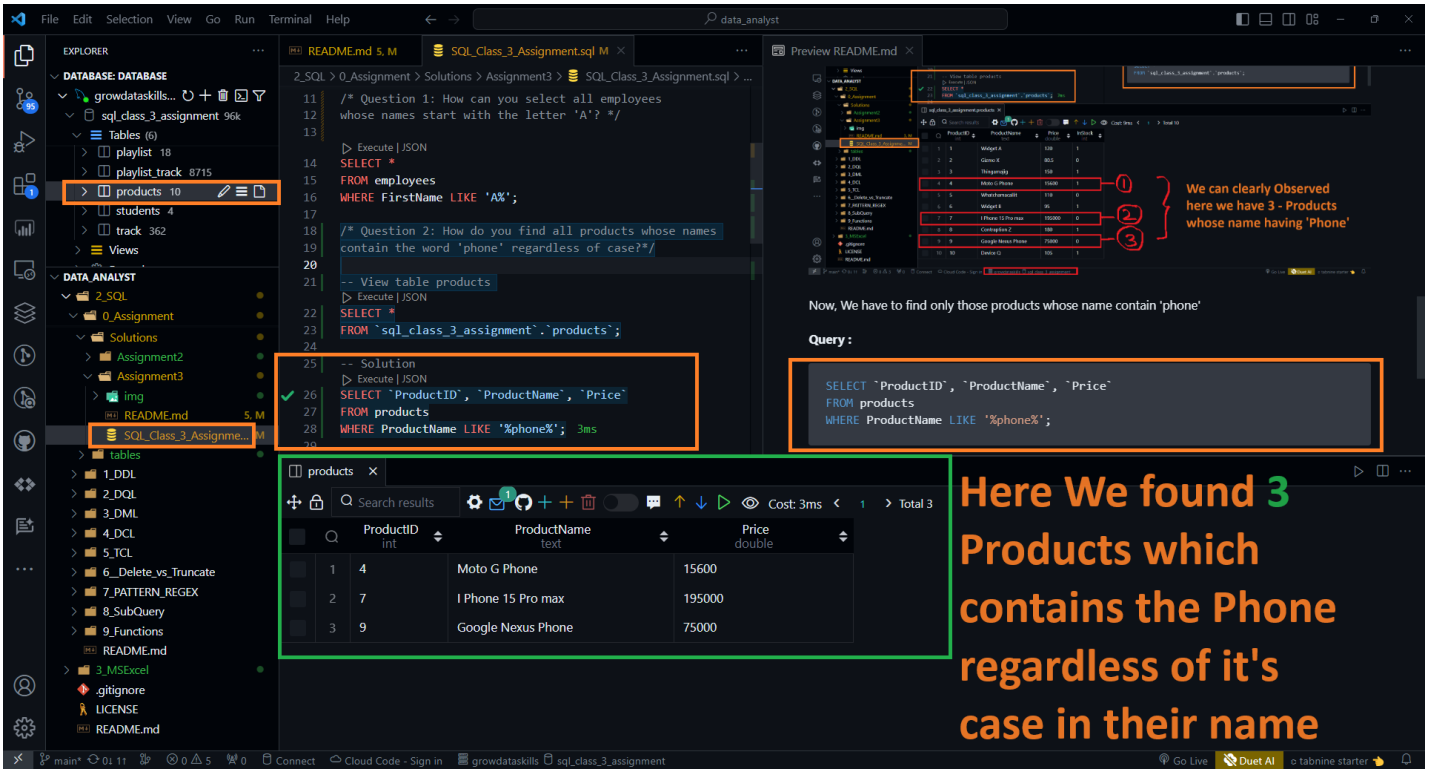


Now, We have to find only those products whose name contain 'phone'

Query :

```
SELECT `ProductID`, `ProductName`, `Price`
FROM products
WHERE ProductName LIKE '%phone%';
```

Execution :



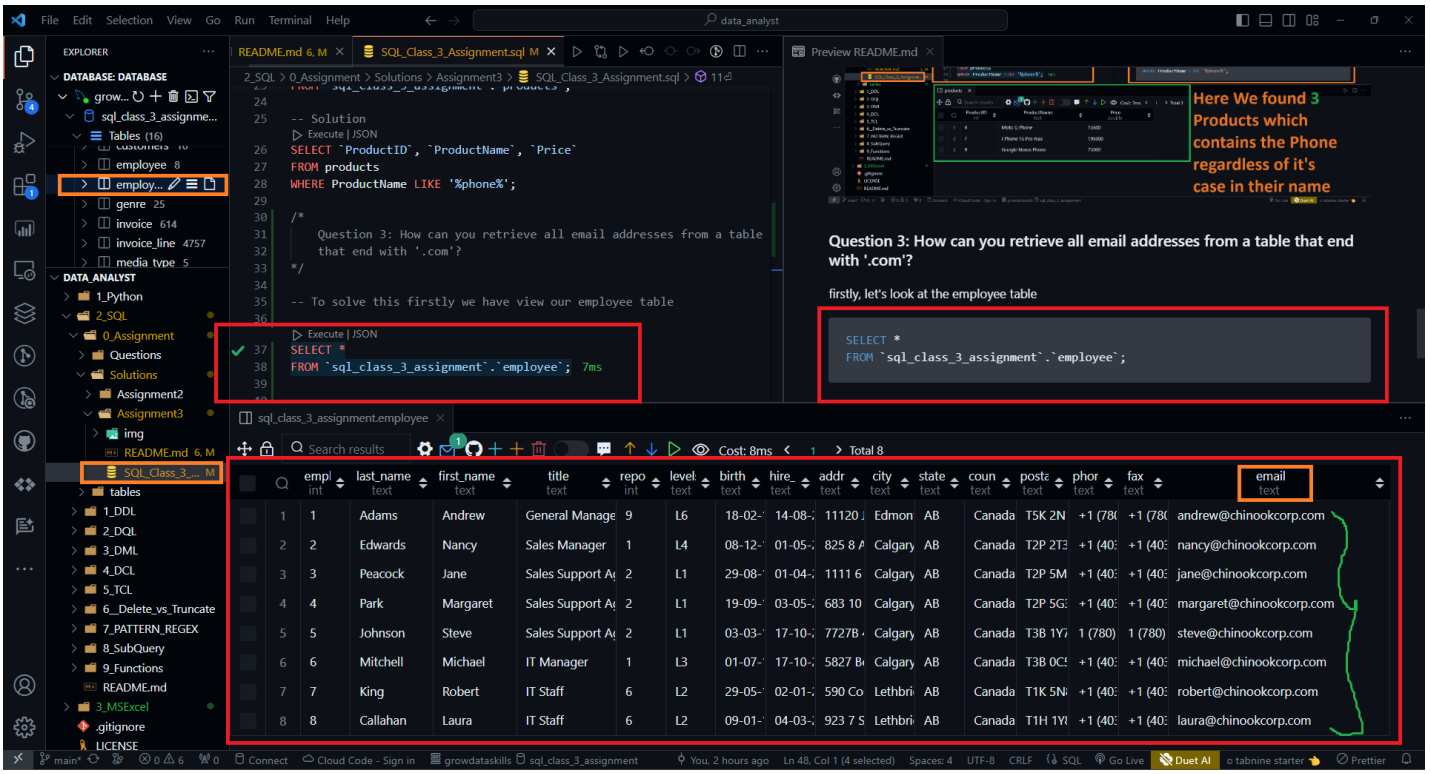
Question 3: How can you retrieve all email addresses from a table that end with '.com'?

firstly, let's look at the employee table

Query :

```
SELECT *
FROM `sql_class_3_assignment`.`employee`;
```

Execution :

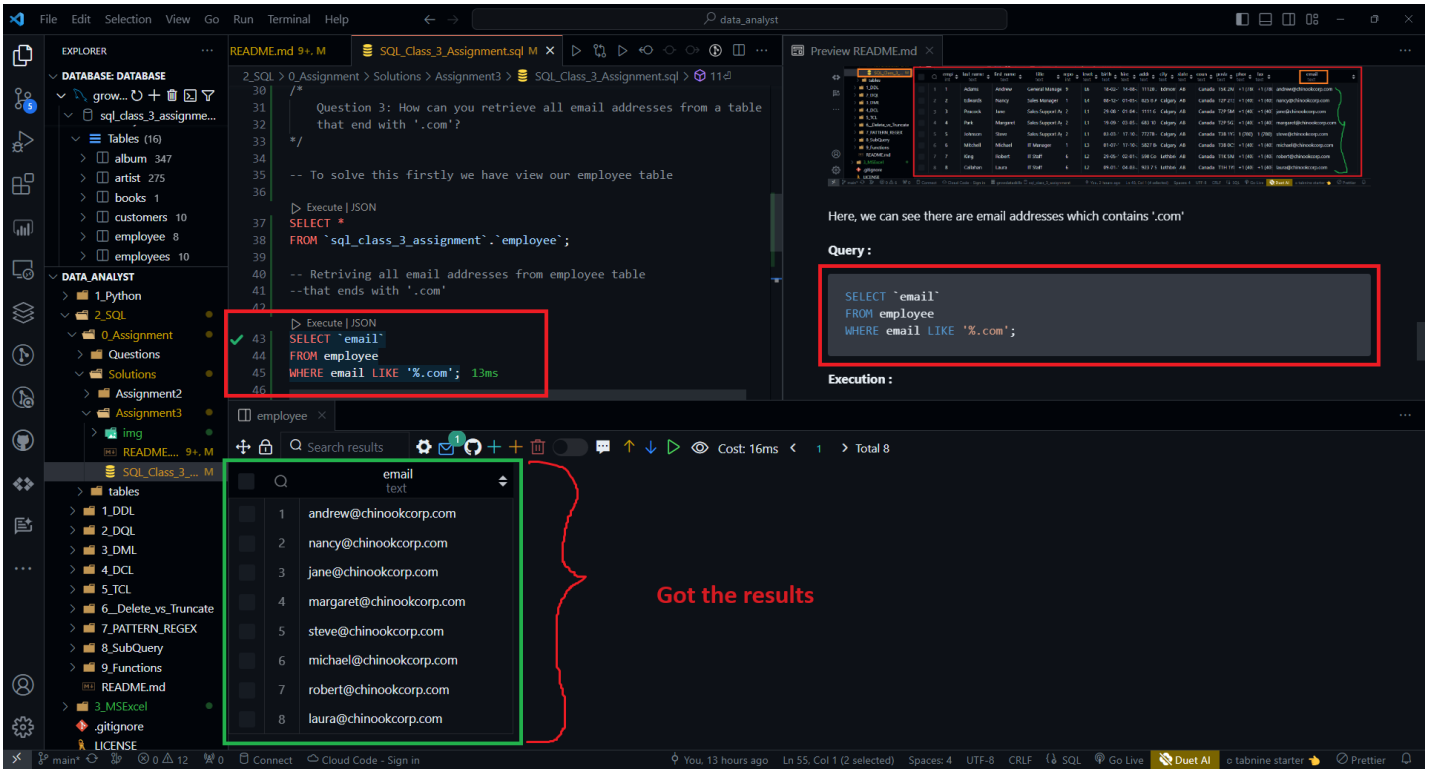


Here, we can see there are email addresses which contains '.com'

Query :

```
SELECT `email`
FROM employee
WHERE email LIKE '%.com';
```

Execution :

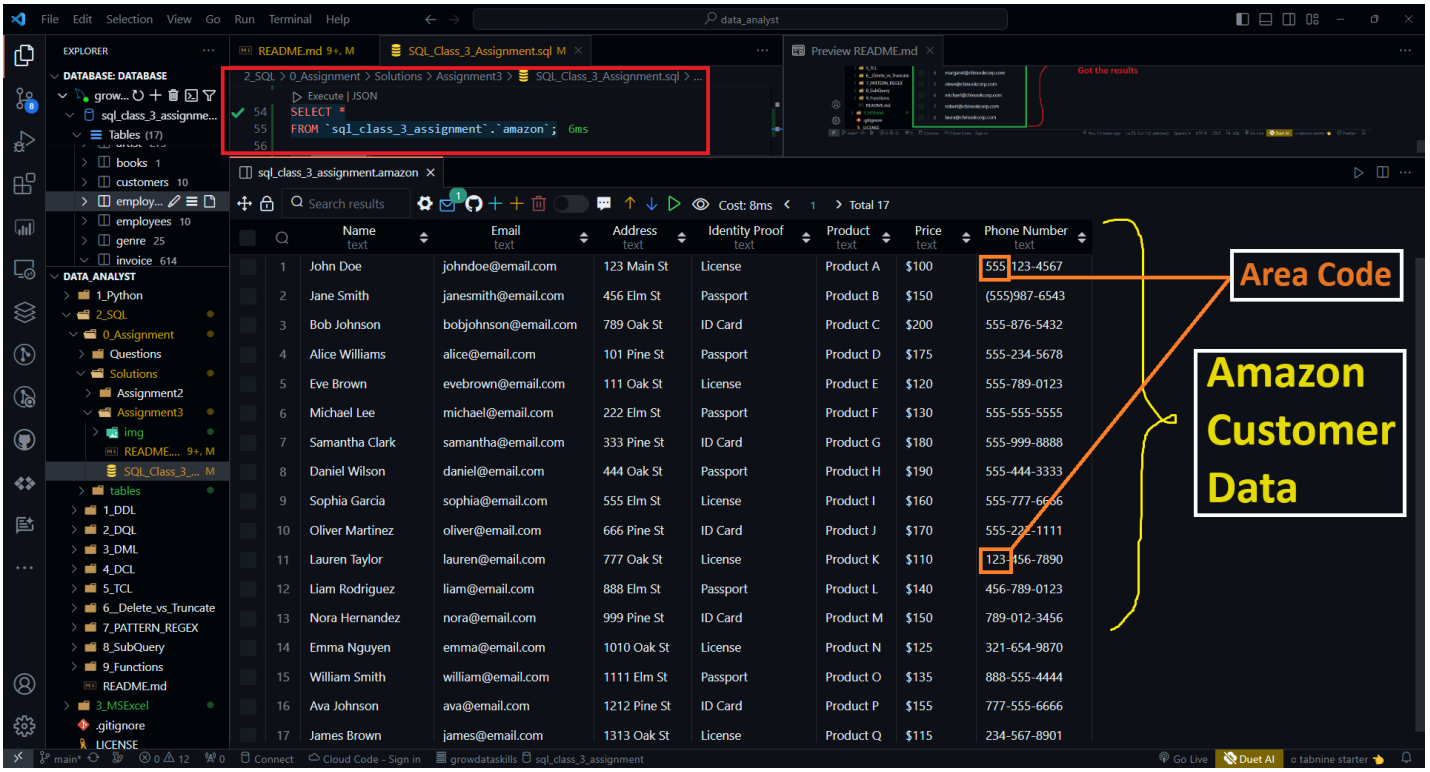


Question 4: How do you find all phone numbers that start with the area code '555'?

Query :

```
SELECT *
FROM `sql_class_3_assignment`.`amazon`;
```

Execution :

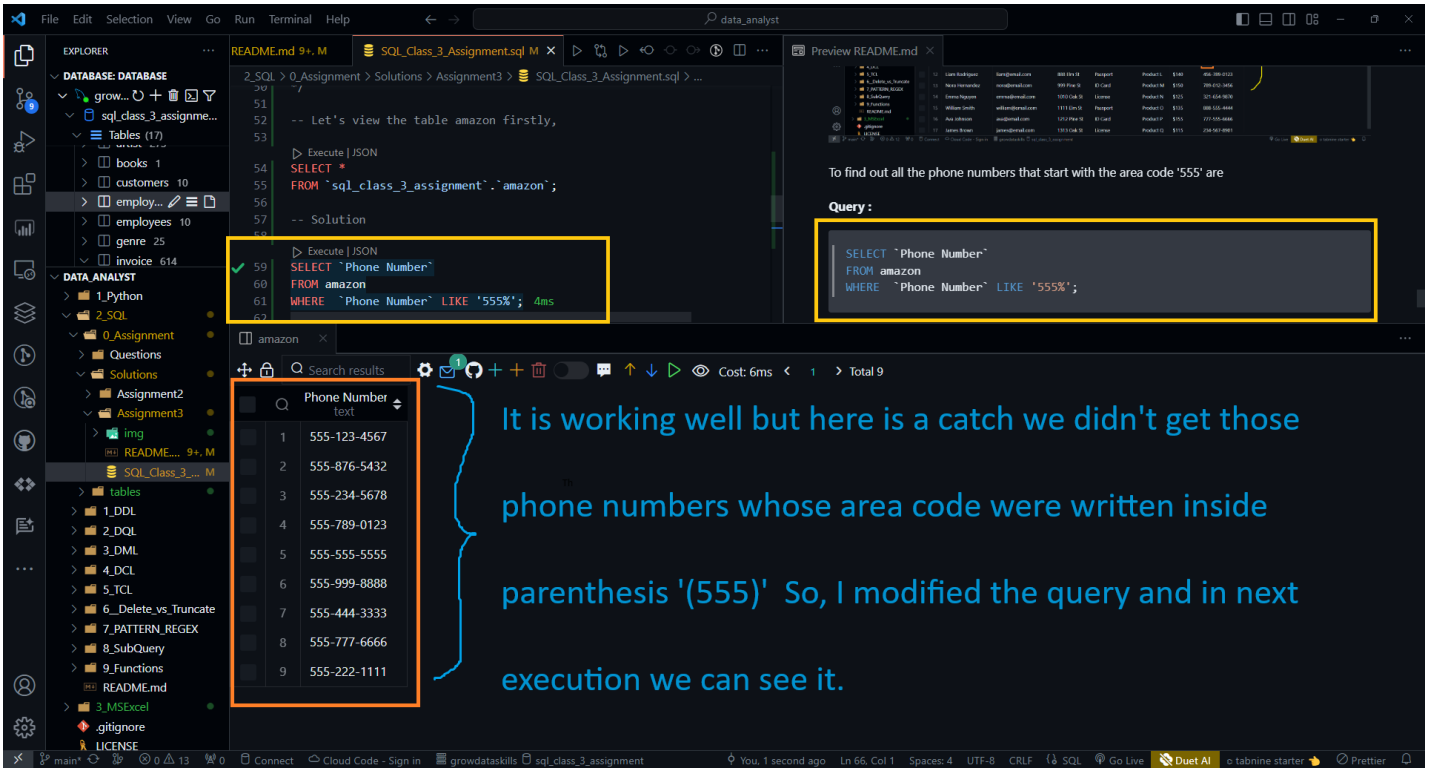


To find out all the phone numbers that start with the area code '555' are

Query :

```
SELECT `Phone Number`
FROM amazon
WHERE  `Phone Number` LIKE '555%';
```

Execution :

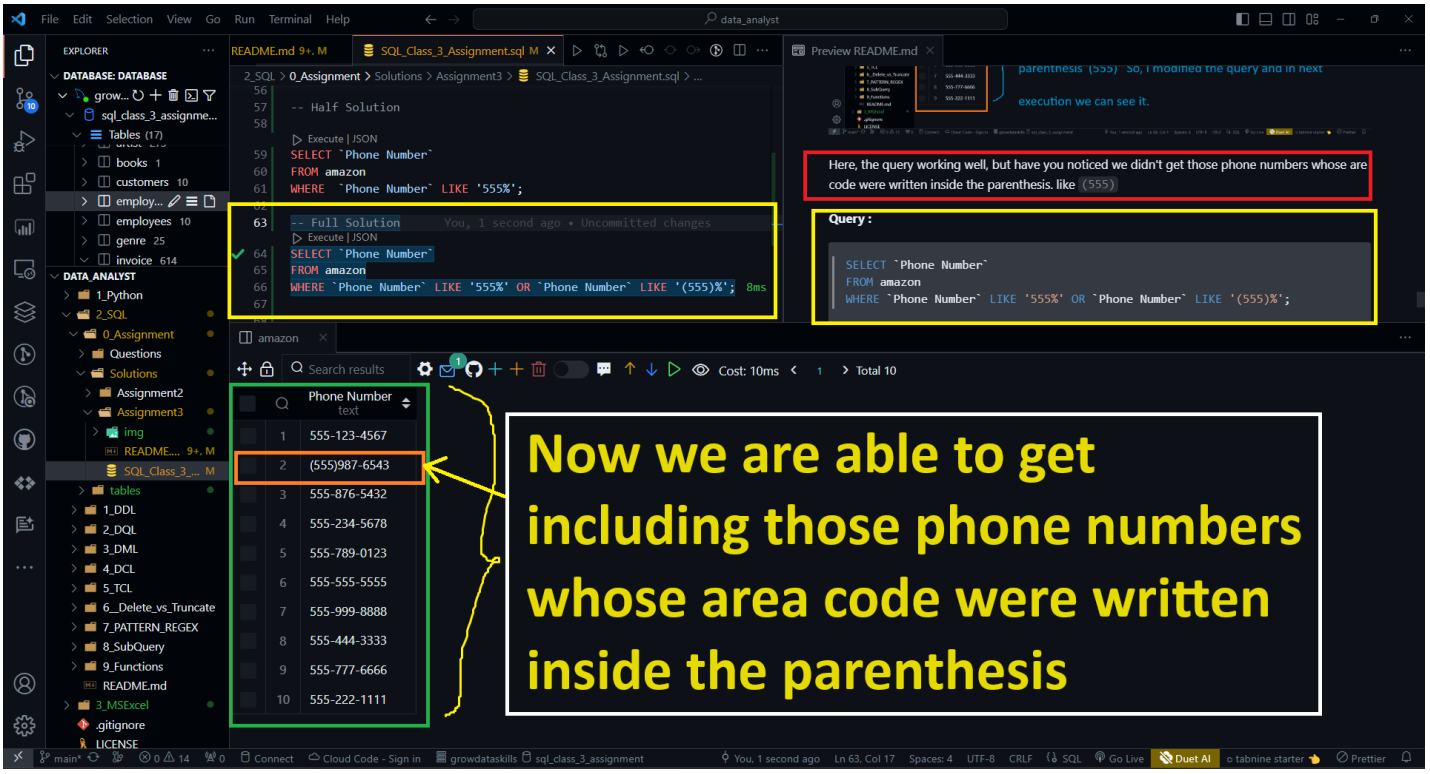


Here, the query working well, but have you noticed we didn't get those phone numbers whose are code were written inside the parenthesis. like `(555)`

Query :

```
SELECT `Phone Number`
FROM amazon
WHERE `Phone Number` LIKE '555%' OR `Phone Number` LIKE '(555)%';
```

Execution :

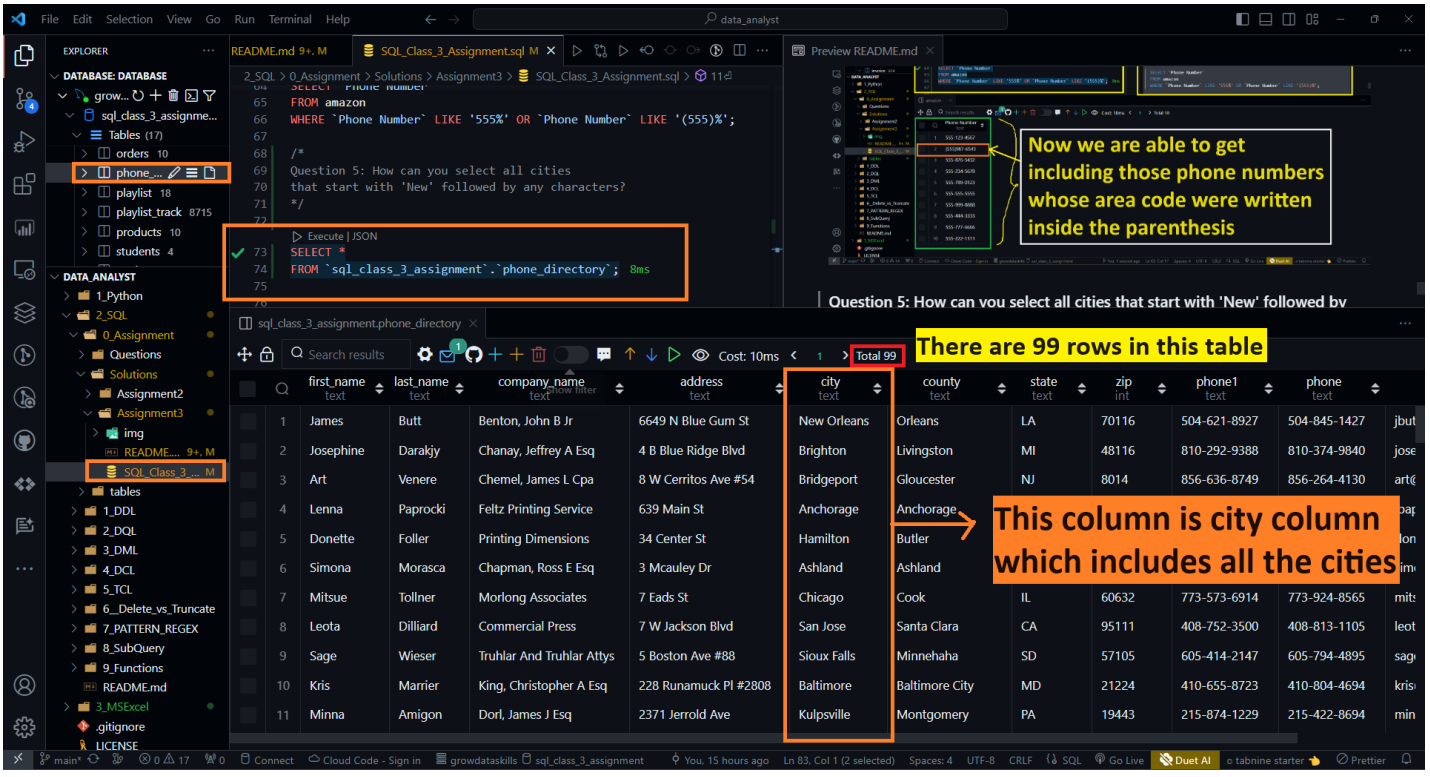


Question 5: How can you select all cities that start with 'New' followed by any characters?

Query :

```
SELECT *
FROM `sql_class_3_assignment`.`phone_directory`;
```

Execution :

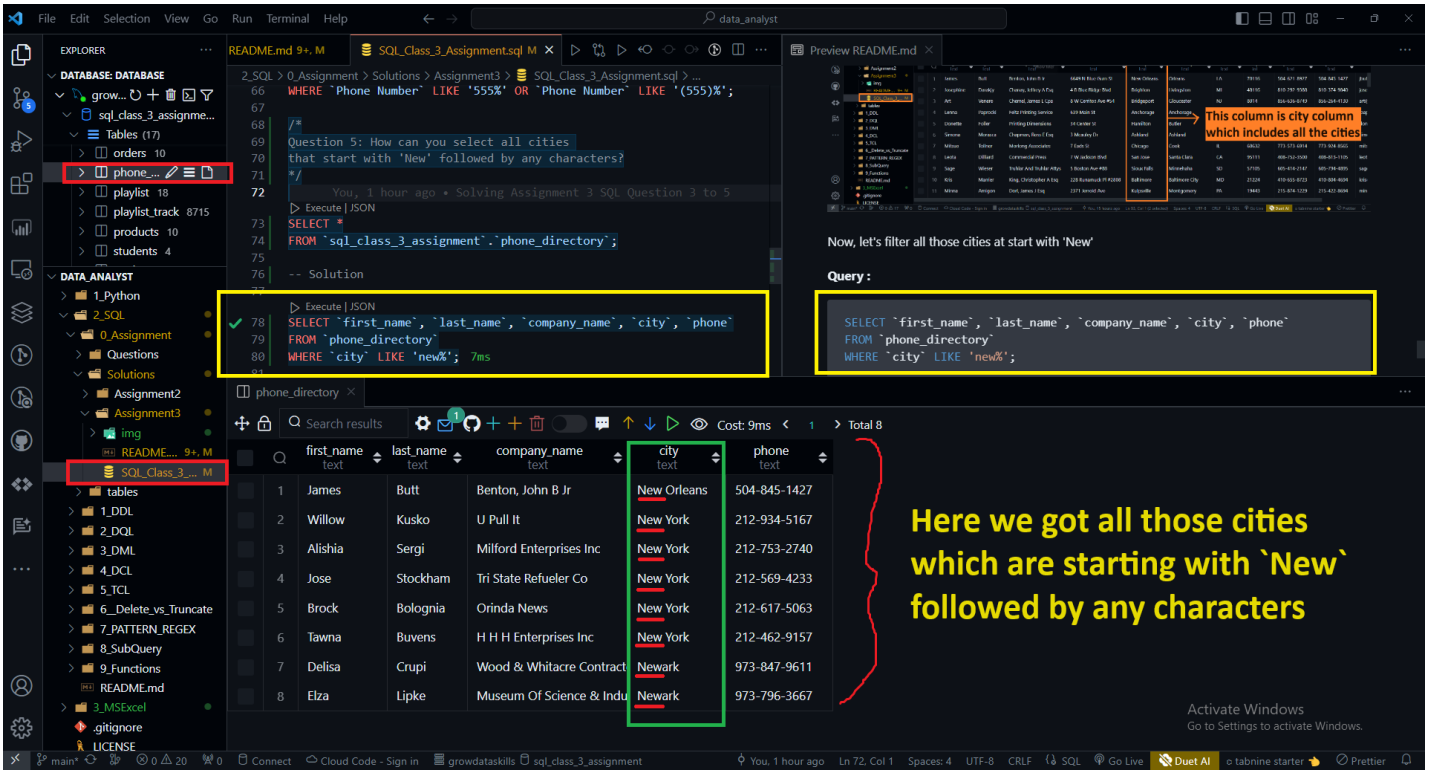


Now, let's filter all those cities at start with 'New'

Query :

```
SELECT `first_name`, `last_name`, `company_name`, `city`, `phone`
FROM `phone_directory`
WHERE `city` LIKE 'new%';
```

Execution :



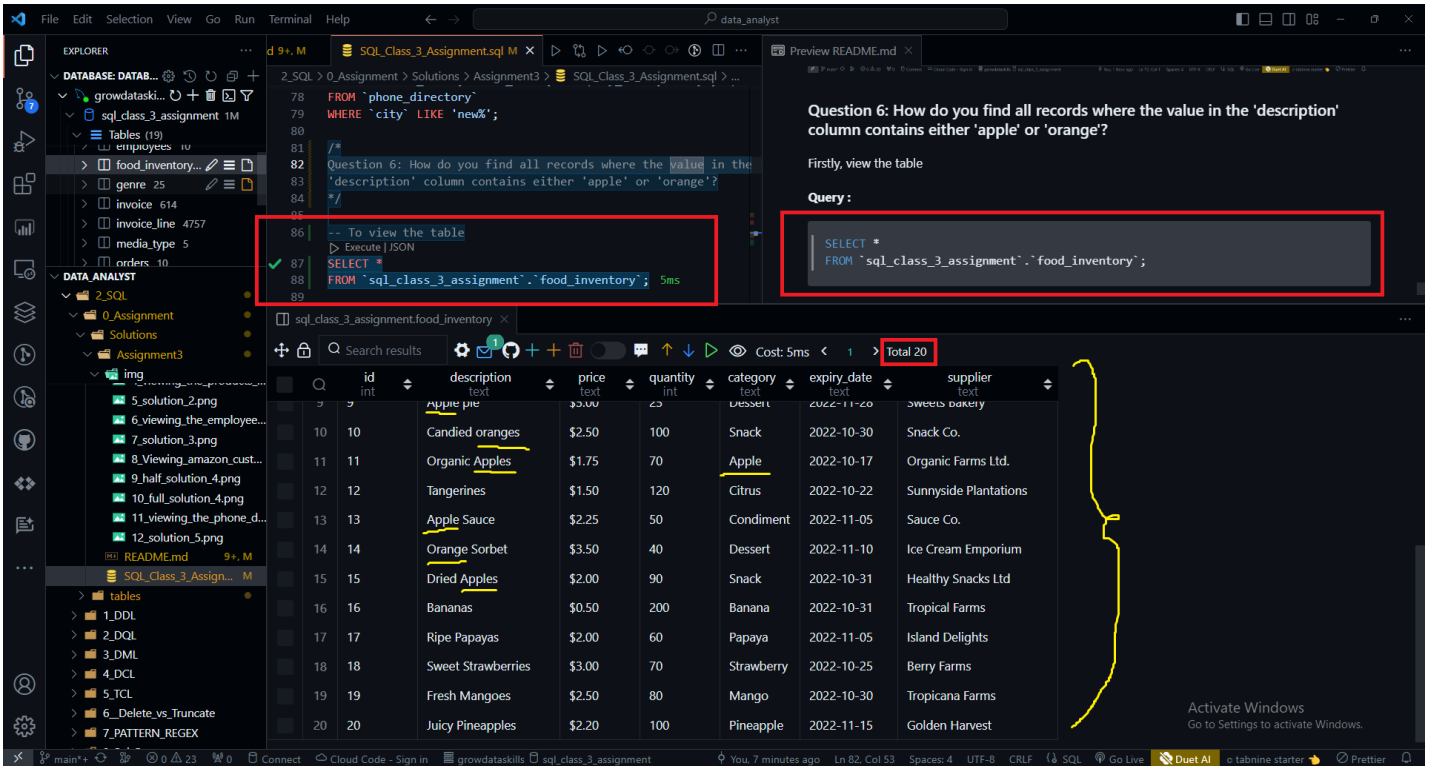
Question 6: How do you find all records where the value in the 'description' column contains either 'apple' or 'orange'?

Firstly, view the table

Query :

```
SELECT *
FROM `sql_class_3_assignment`.`food_inventory`;
```

Execution :

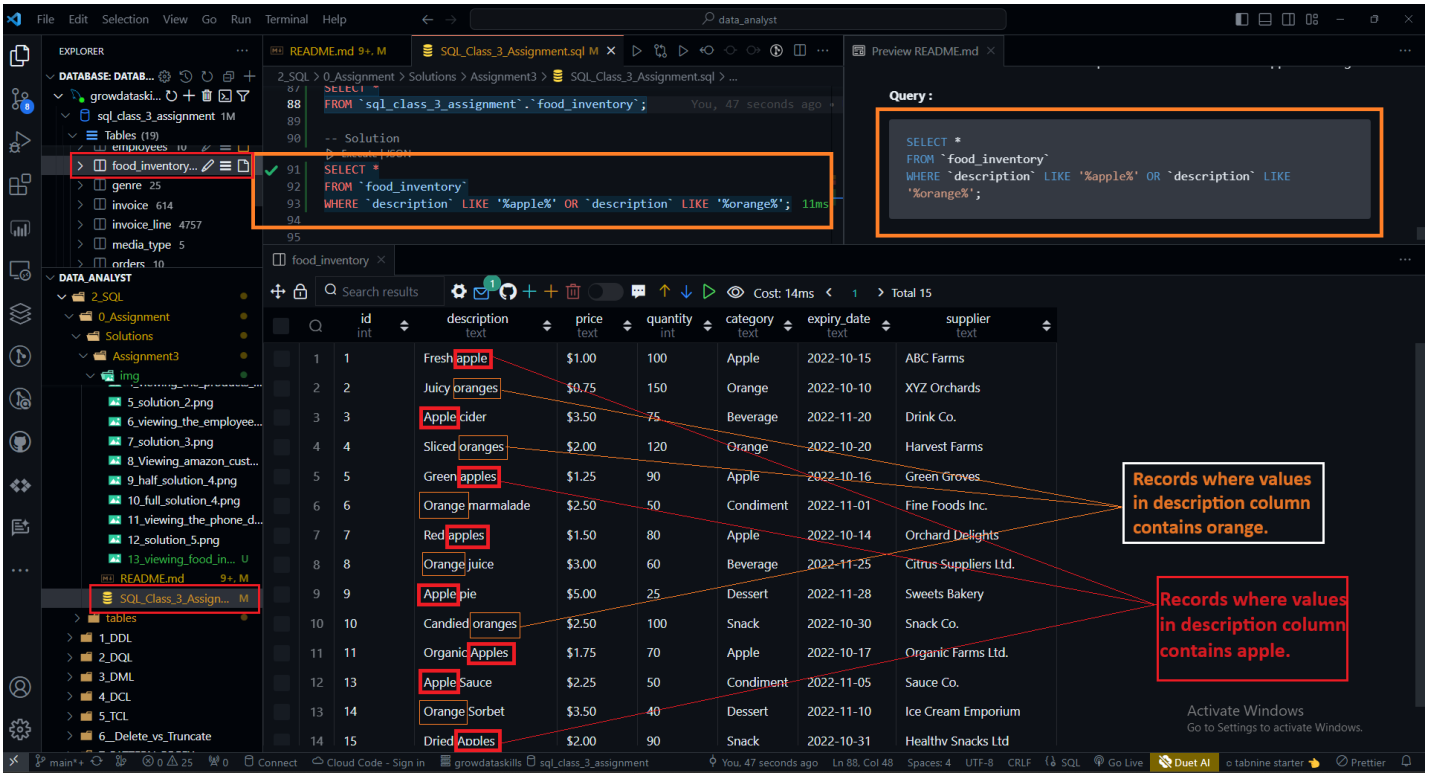


Records where the value in the 'description' column contains either 'apple' or 'orange'

Query :

```
SELECT *
FROM `food_inventory`
WHERE `description` LIKE '%apple%' OR `description` LIKE '%orange%';
```

Execution :



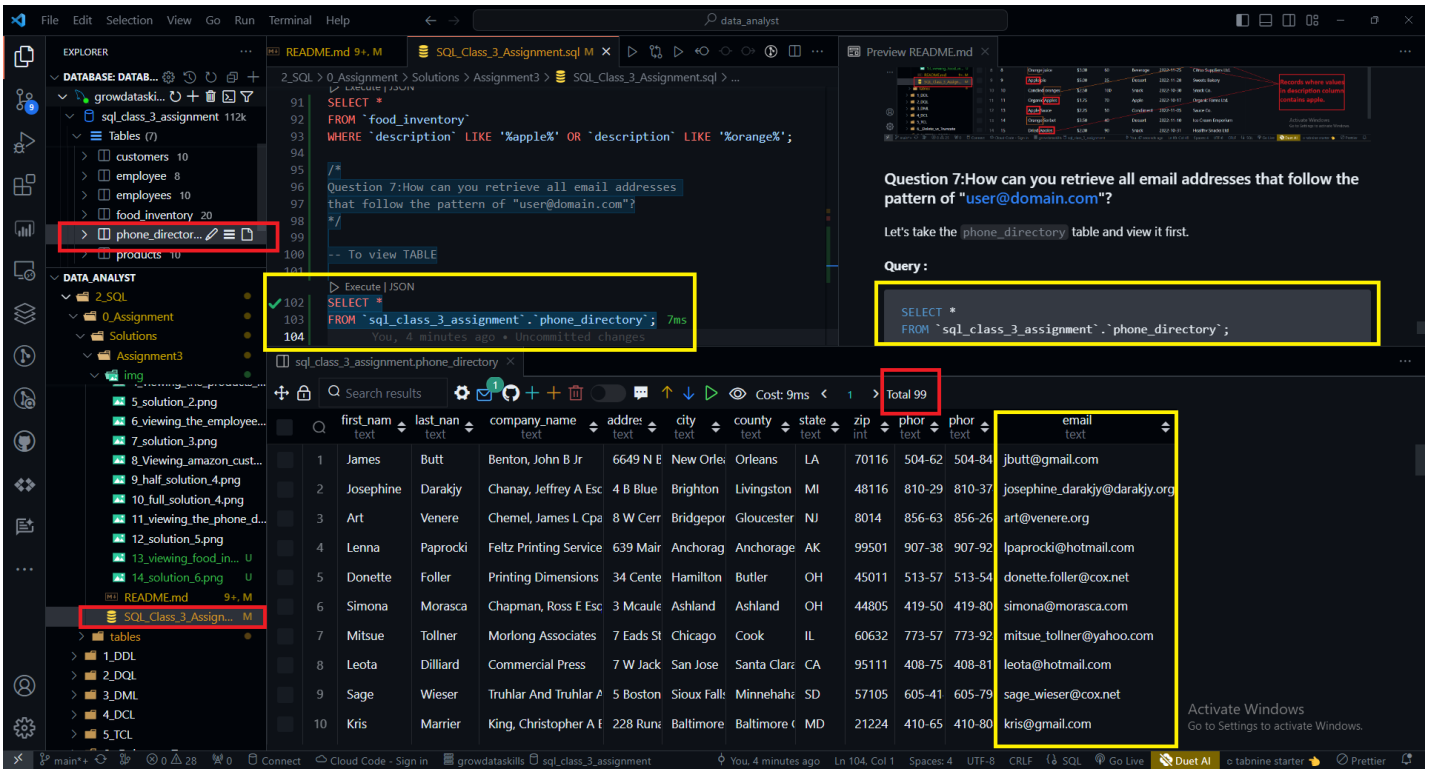
Question 7:How can you retrieve all email addresses that follow the pattern of "user@domain.com"?

Let's take the `phone_directory` table and view it first.

Query :

```
SELECT *
FROM `sql_class_3_assignment`.`phone_directory`;
```

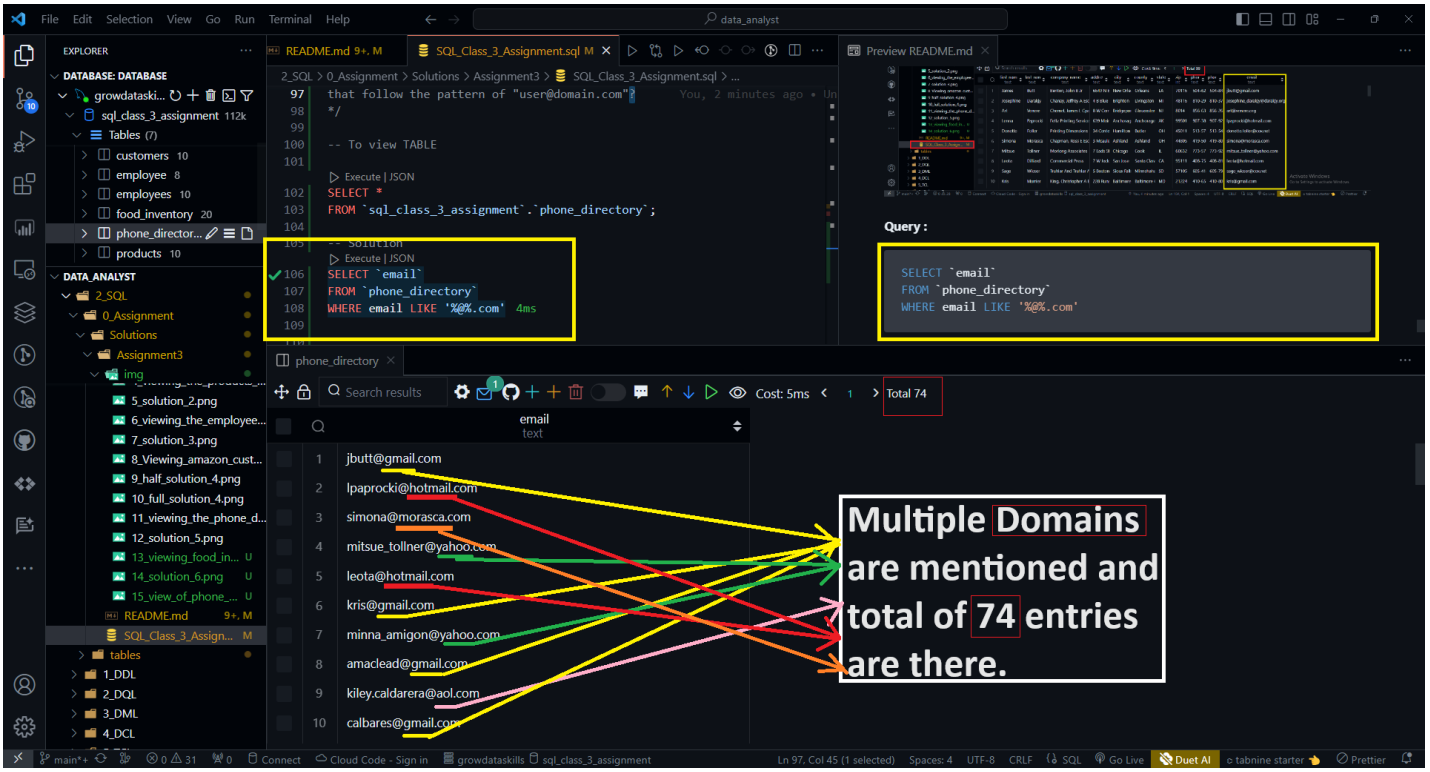
Execution :



Query :

```
SELECT `email`
FROM `phone_directory`
WHERE email LIKE '%@%.com'
```

Execution :



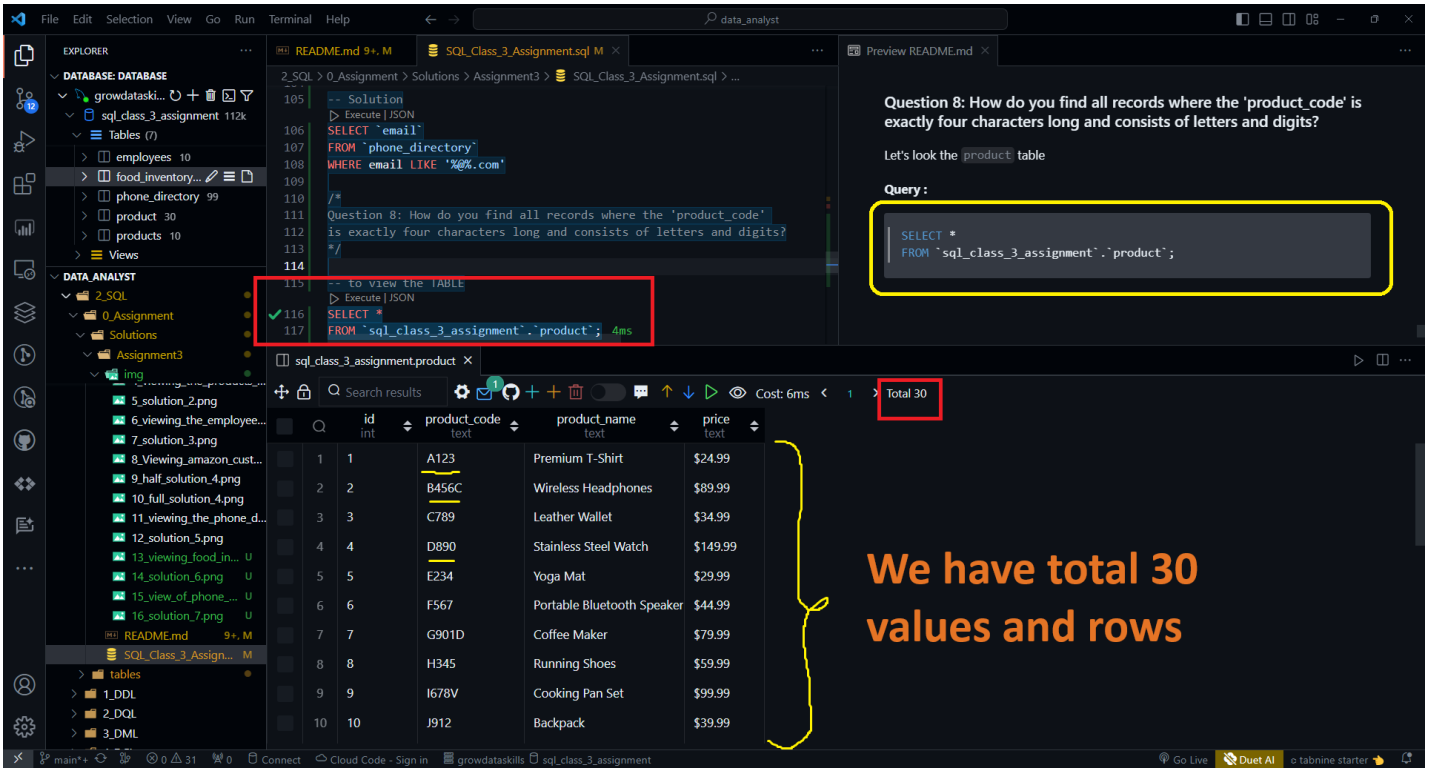
Question 8: How do you find all records where the 'product_code' is exactly four characters long and consists of letters and digits?

Let's look the `product` table

Query :

```
SELECT *
FROM `sql_class_3_assignment`.`product`;
```

Execution :

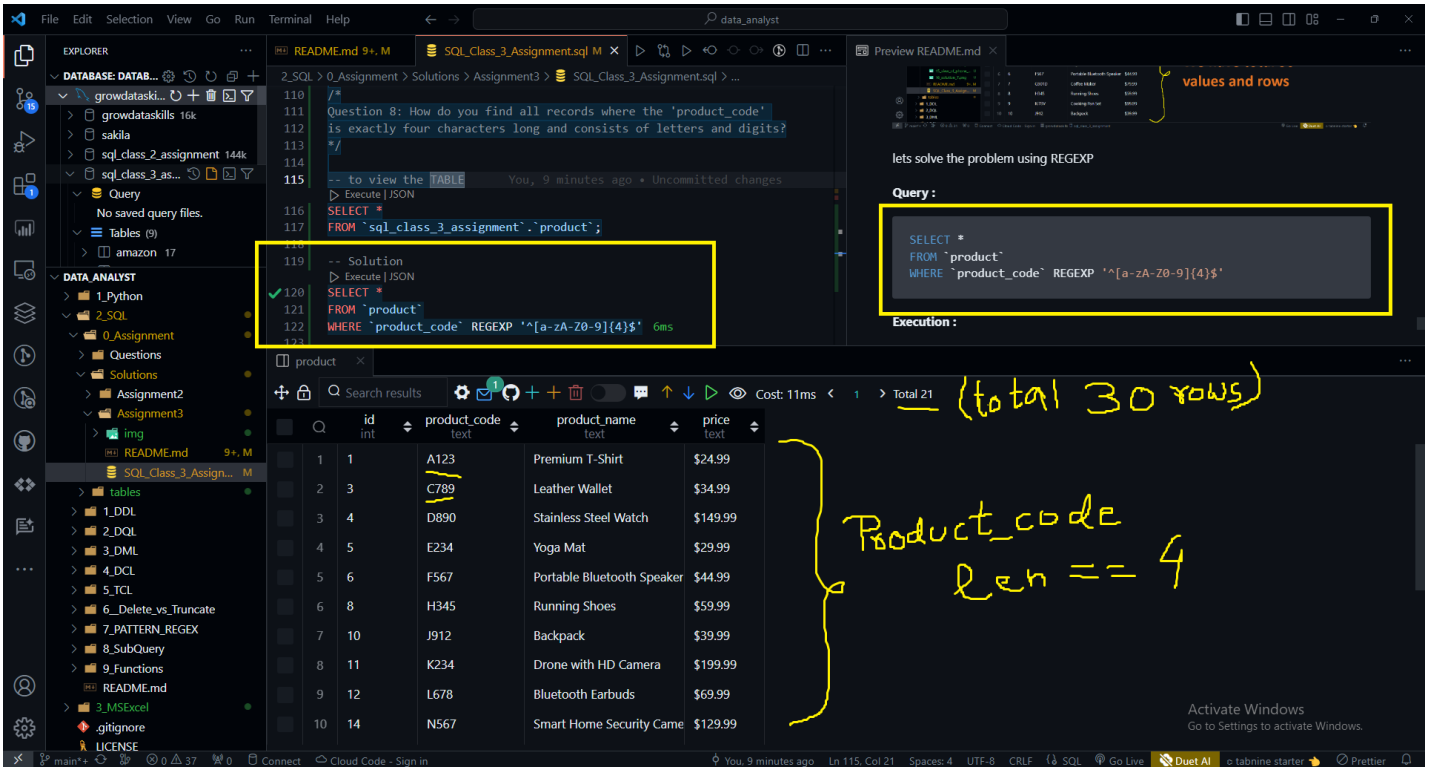


lets solve the problem using REGEXP

Query :

```
SELECT *
FROM `product`
WHERE `product_code` REGEXP '^[a-zA-Z0-9]{4}$'
```

Execution :

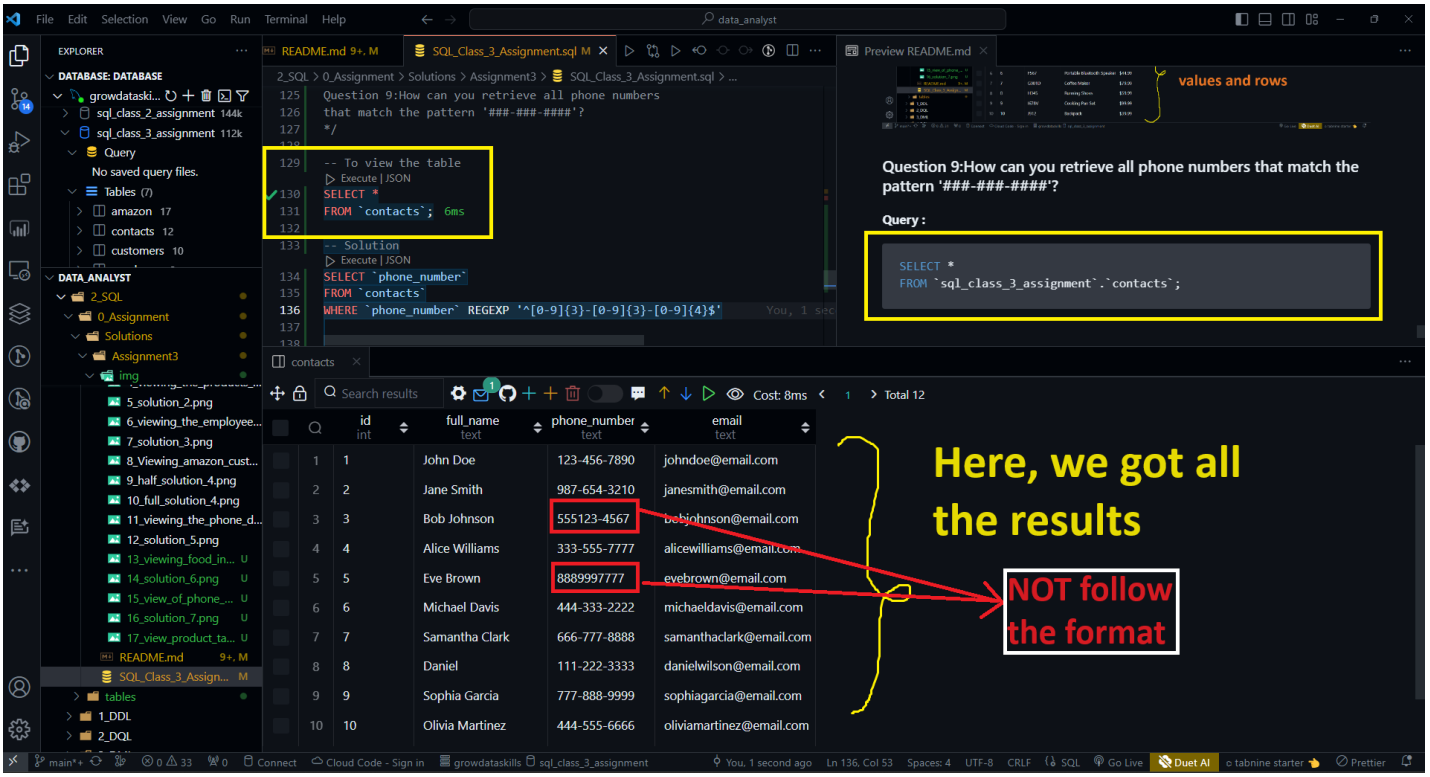


Question 9: How can you retrieve all phone numbers that match the pattern '###-###-####'?

Query :

```
SELECT *
FROM `sql_class_3_assignment`.`contacts`;
```

Execution :

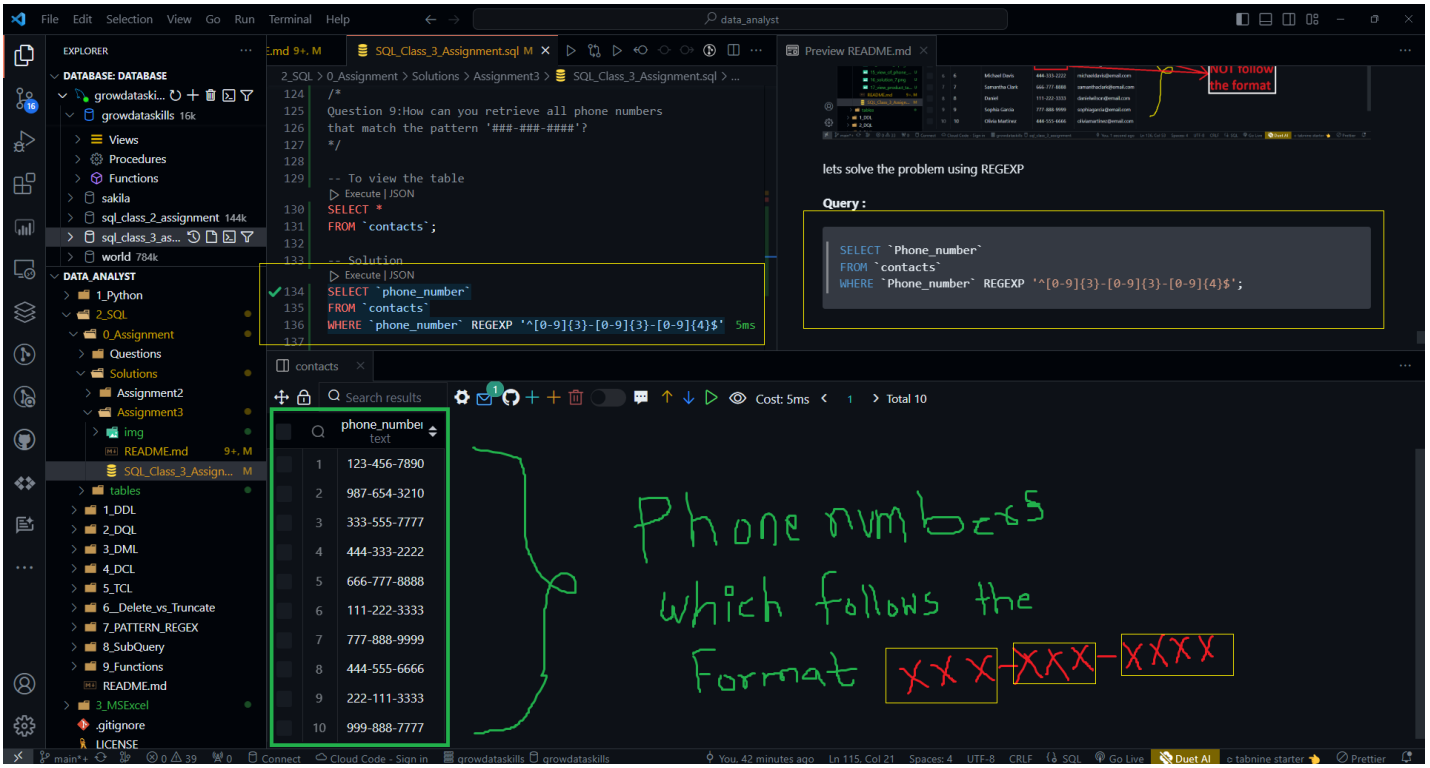


lets solve the problem using REGEXP

Query :

```
SELECT `Phone_number`
FROM `contacts`
WHERE `Phone_number` REGEXP '^[0-9]{3}-[0-9]{3}-[0-9]{4}$';
```

Execution :



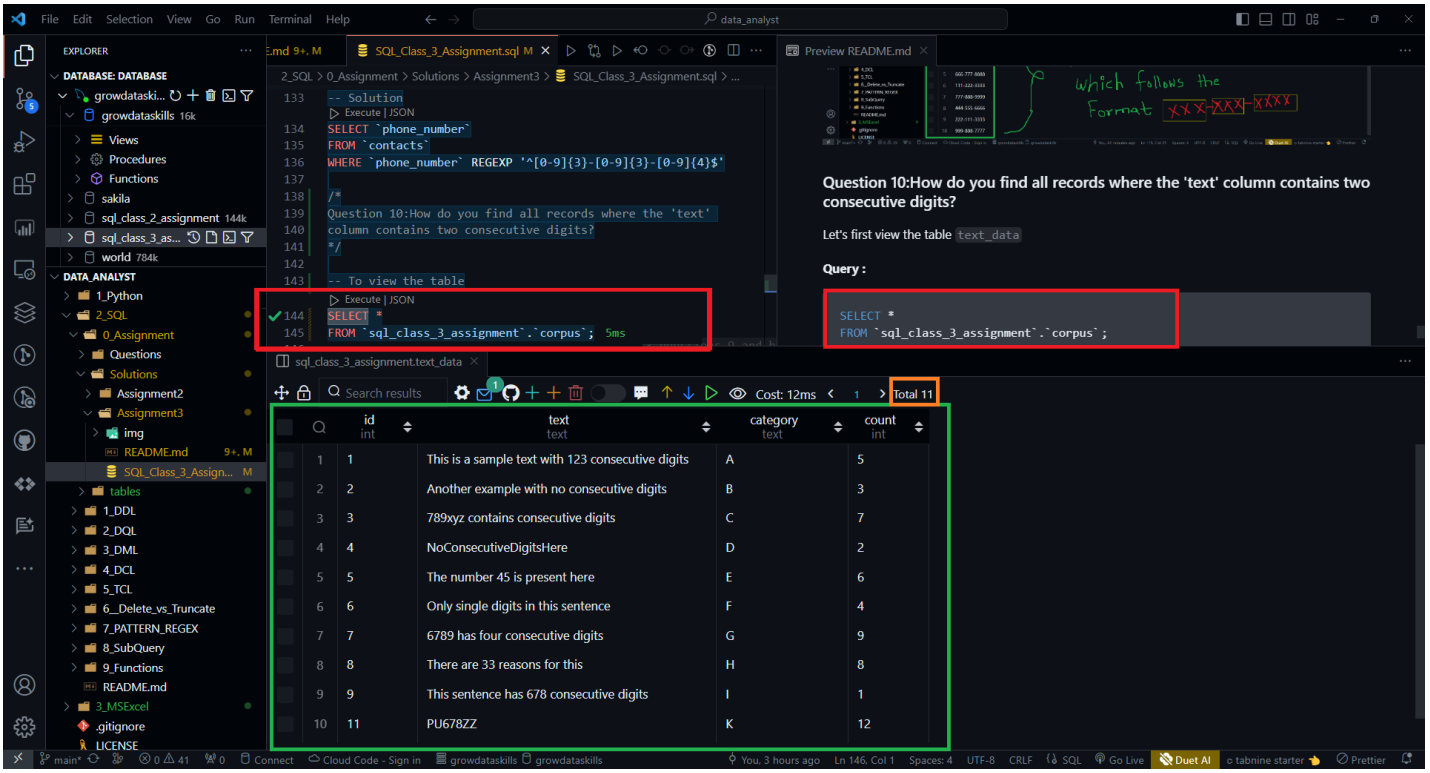
Question 10: How do you find all records where the 'text' column contains two consecutive digits?

Let's first view the table `corpus`

Query :

```
SELECT *
FROM `sql_class_3_assignment`.`corpus`;
```

Execution :

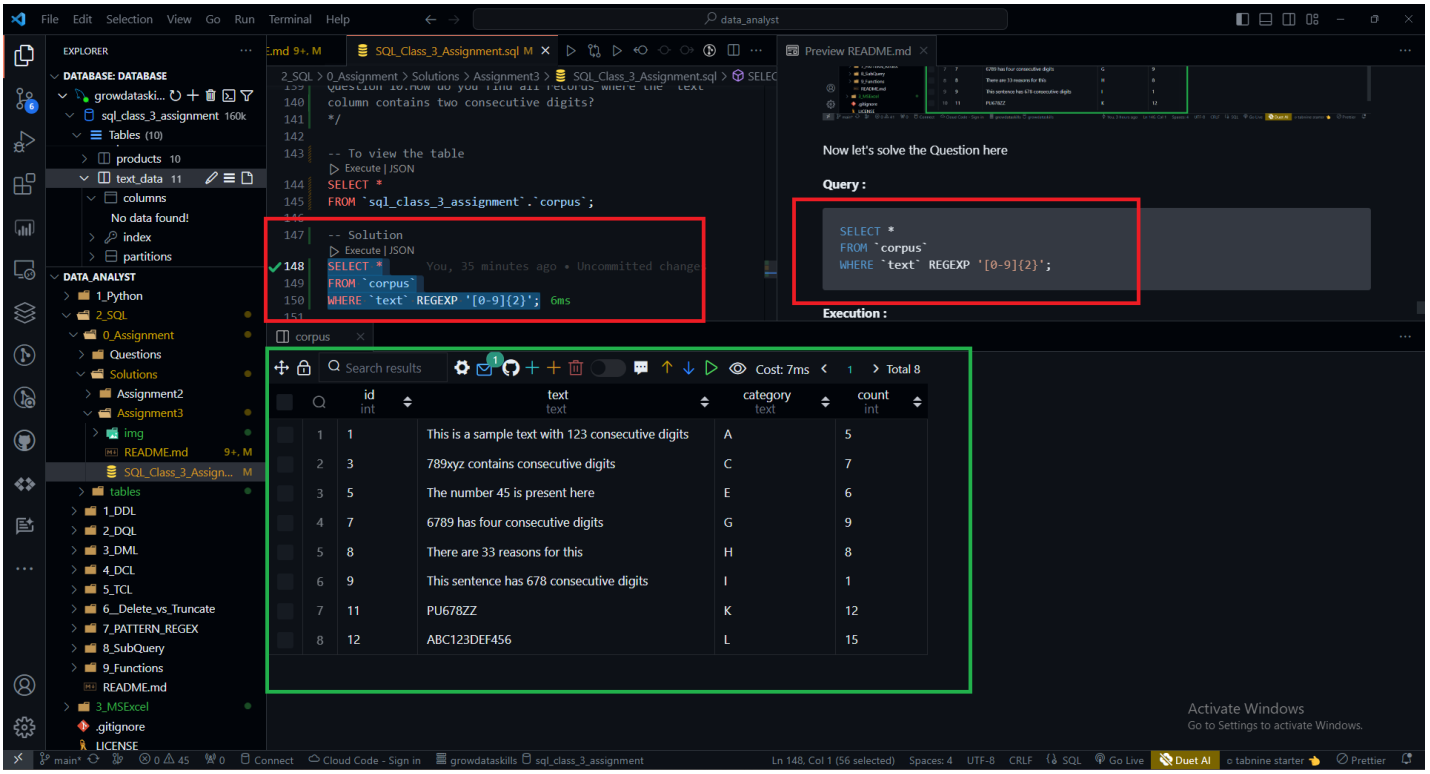


Now let's solve the Question here

Query :

```
SELECT *
FROM `corpus`
WHERE `text` REGEXP '[0-9]{2}';
```

Execution :



NULL VALUES:

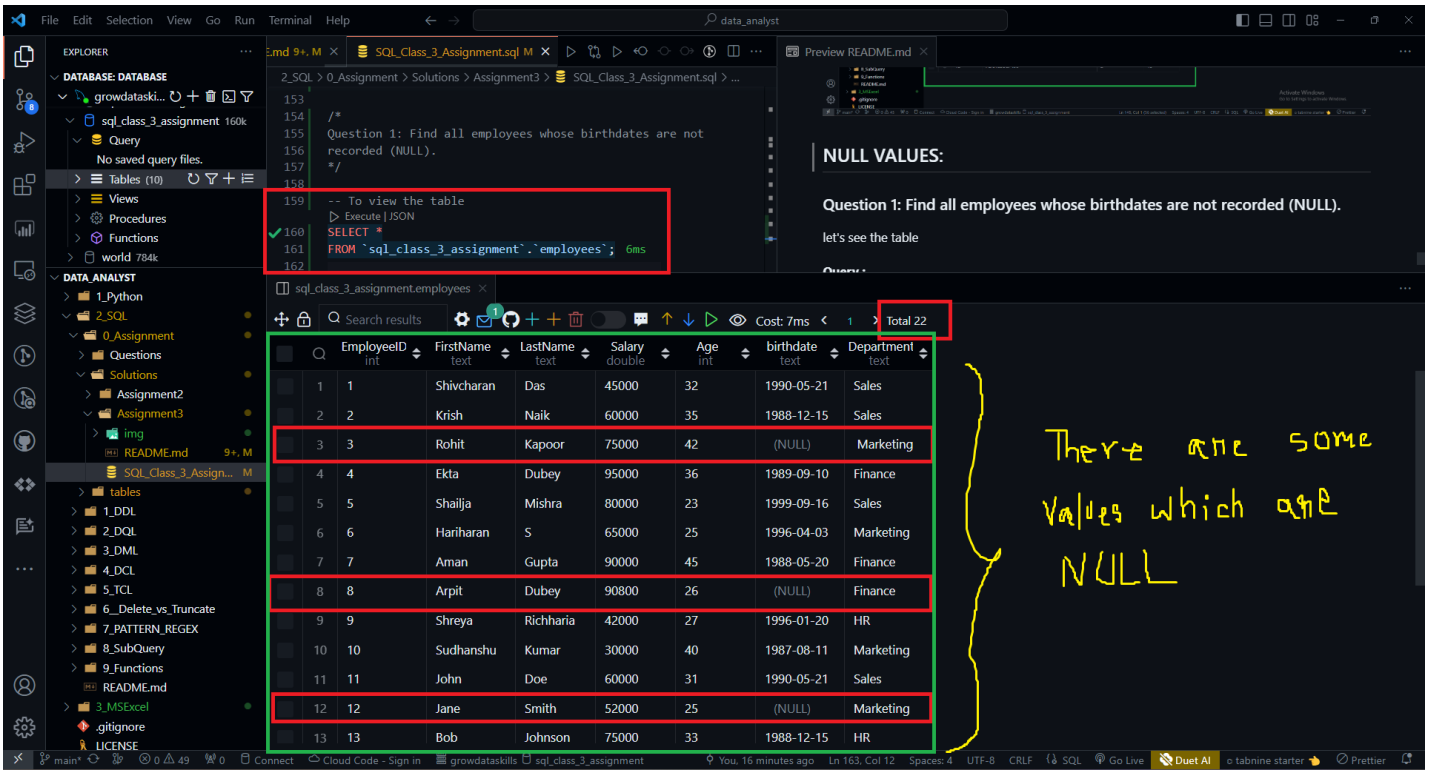
Question 1: Find all employees whose birthdates are not recorded (NULL).

let's see the table

Query :

```
SELECT *
FROM `sql_class_3_assignment`.`employees`;
```

Execution :

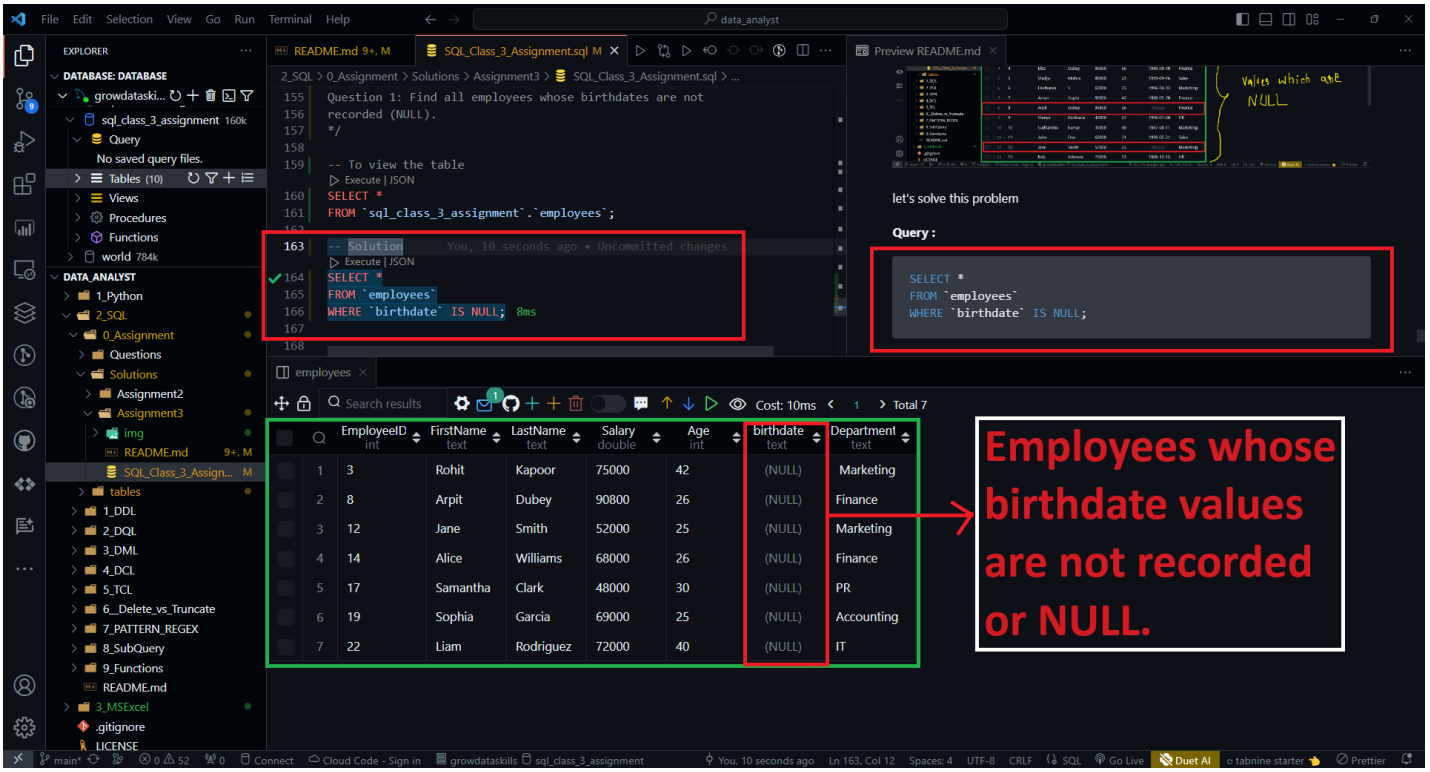


let's solve this problem

Query :

```
SELECT *
FROM `employees`
WHERE `birthdate` IS NULL;
```

Execution :



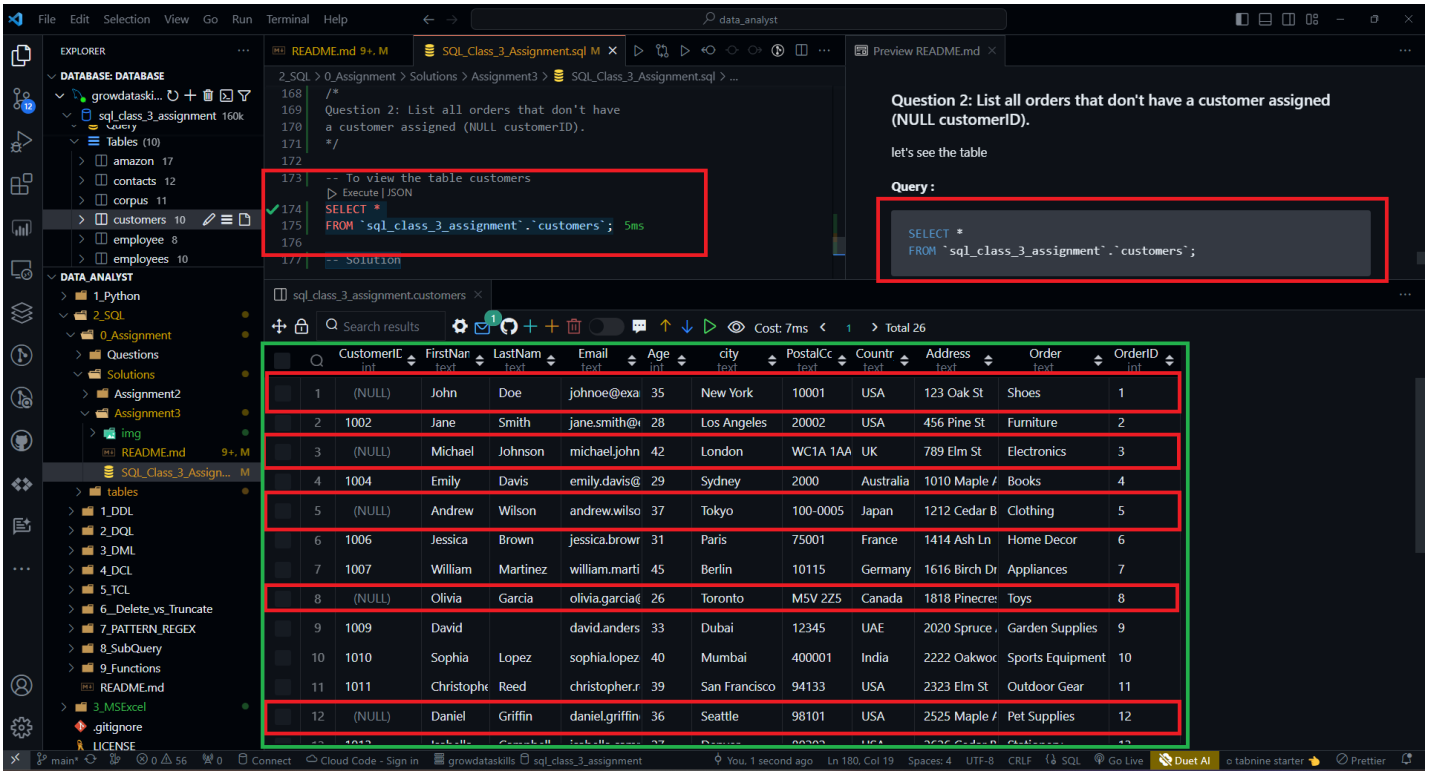
Question 2: List all orders that don't have a customer assigned (NULL customerID).

let's see the table

Query :

```
SELECT *
FROM `sql_class_3_assignment`.`customers`;
```

Execution :

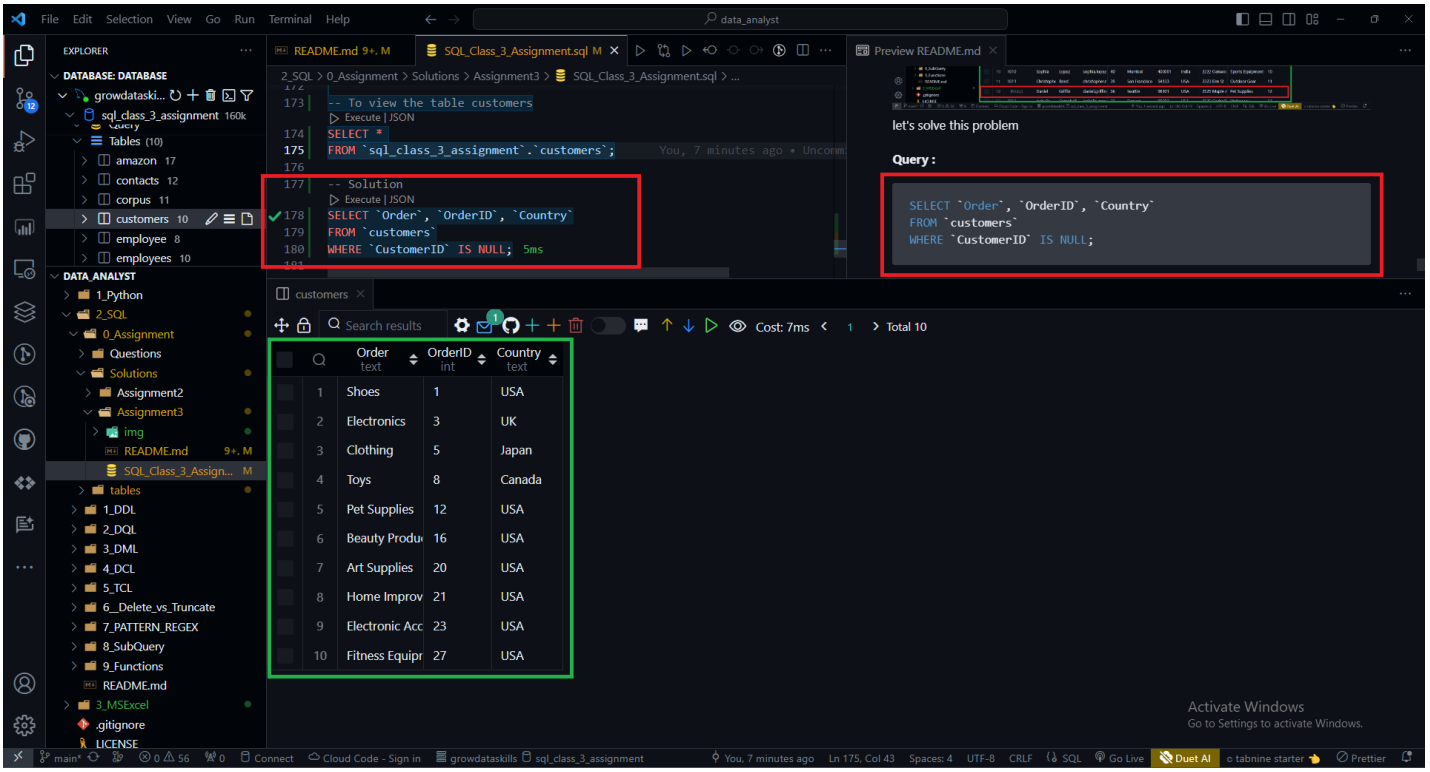


let's solve this problem

Query :

```
SELECT `Order`, `OrderID`, `Country`
FROM `customers`
WHERE `CustomerID` IS NULL;
```

Execution :



FUNCTIONS:

- Consider a table named Sales with the following columns:

  SaleID (integer): The unique identifier for each sale.

  Product (string): The name of the product sold.

  Quantity (integer): The quantity of the product sold.

  Price (decimal): The price per unit of the product.

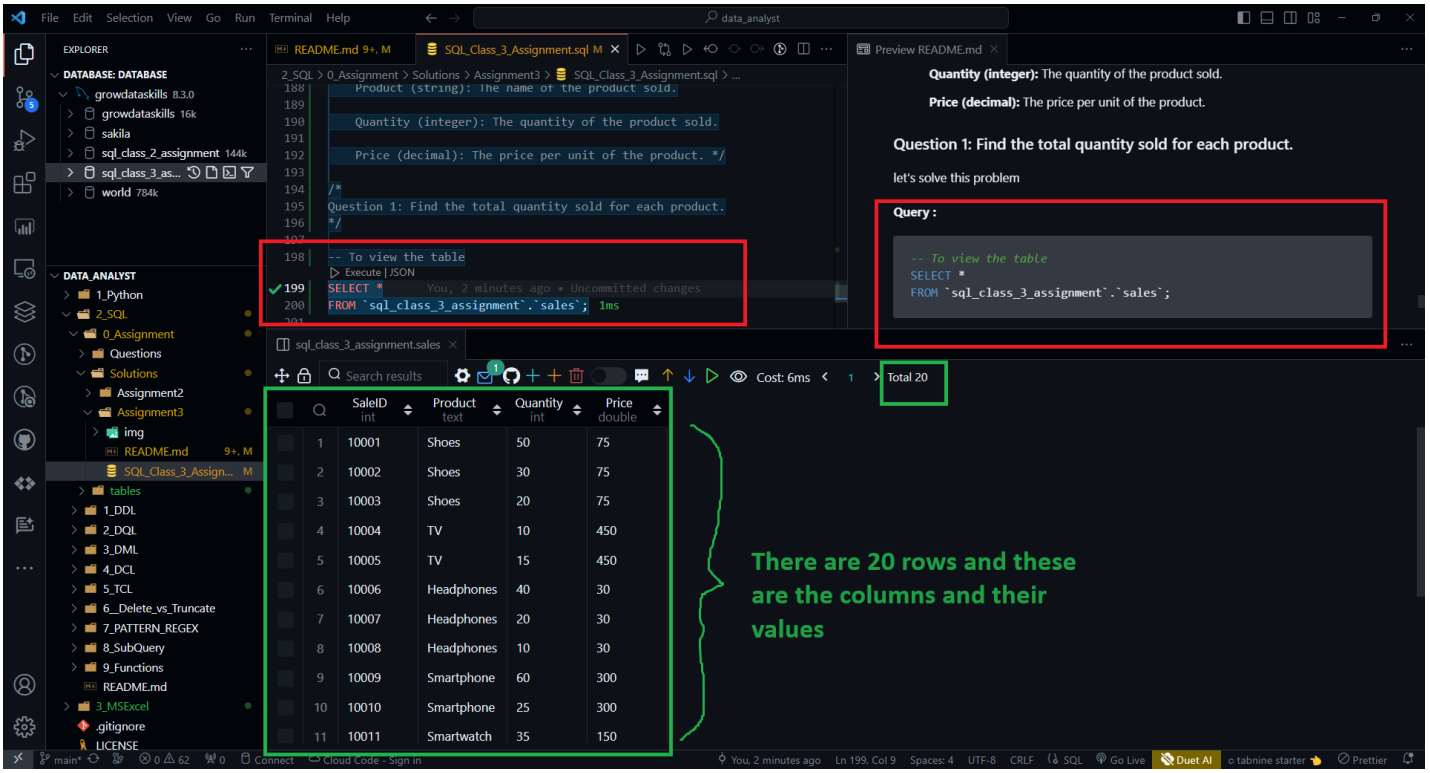
Question 1: Find the total quantity sold for each product.

let's view `sales` table

Query :

```
-- To view the table
SELECT *
FROM `sql_class_3_assignment`.`sales`;
```

Execution :



	SaleID int	Product text	Quantity int	Price double
1	10001	Shoes	50	75
2	10002	Shoes	30	75
3	10003	Shoes	20	75
4	10004	TV	10	450
5	10005	TV	15	450
6	10006	Headphones	40	30
7	10007	Headphones	20	30
8	10008	Headphones	10	30
9	10009	Smartphone	60	300
10	10010	Smartphone	25	300
11	10011	Smartwatch	35	150

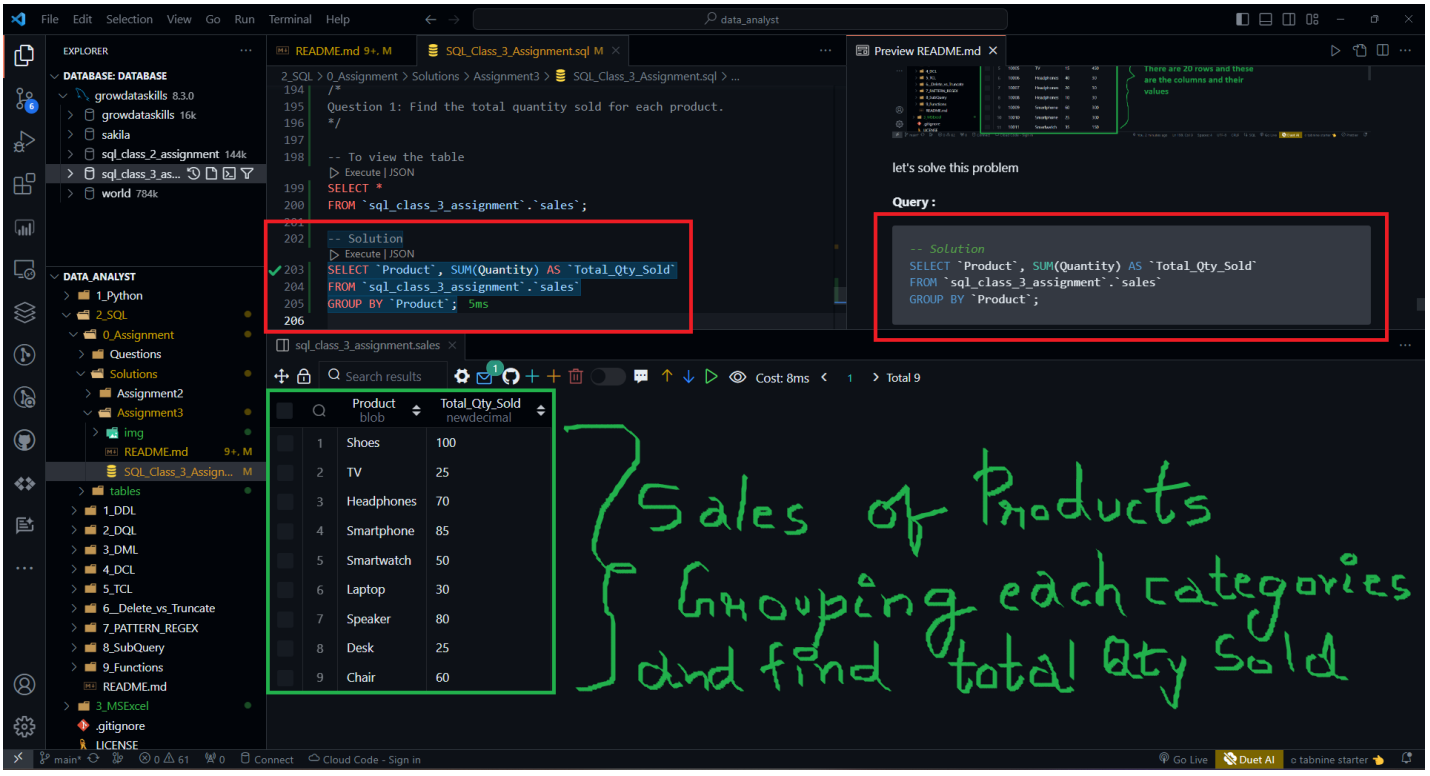
let's solve this problem

Query :

```
-- Solution
SELECT `Product`, SUM(Quantity) AS `Total_Qty_Sold`
```

```
FROM `sql_class_3_assignment`.`sales`
GROUP BY `Product`;
```

Execution :

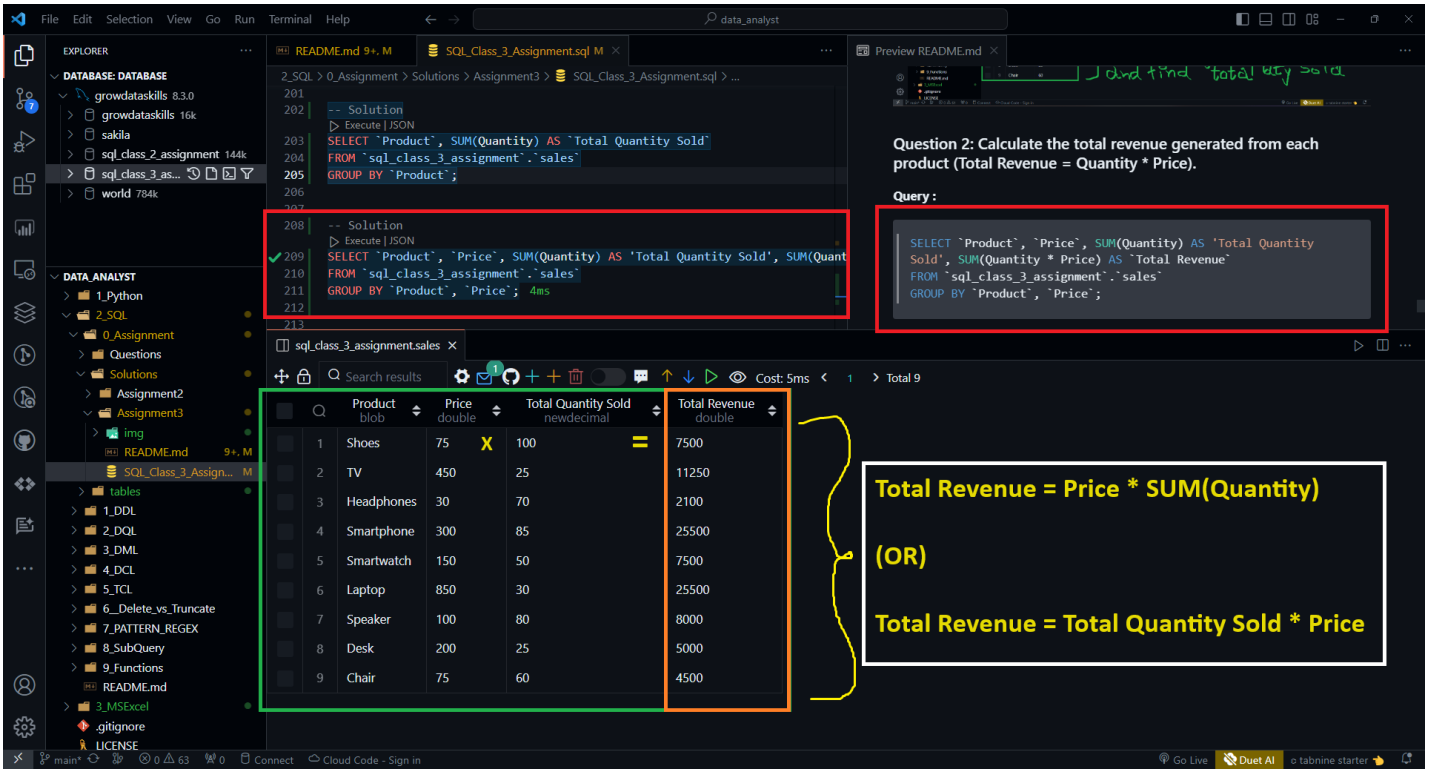


Question 2: Calculate the total revenue generated from each product (Total Revenue = Quantity * Price).

Query :

```
SELECT `Product`, `Price`, SUM(Quantity) AS 'Total Quantity Sold', SUM(Quantity * Price) AS `Total
Revenue`
FROM `sql_class_3_assignment`.`sales`
GROUP BY `Product`, `Price`;
```

Execution :

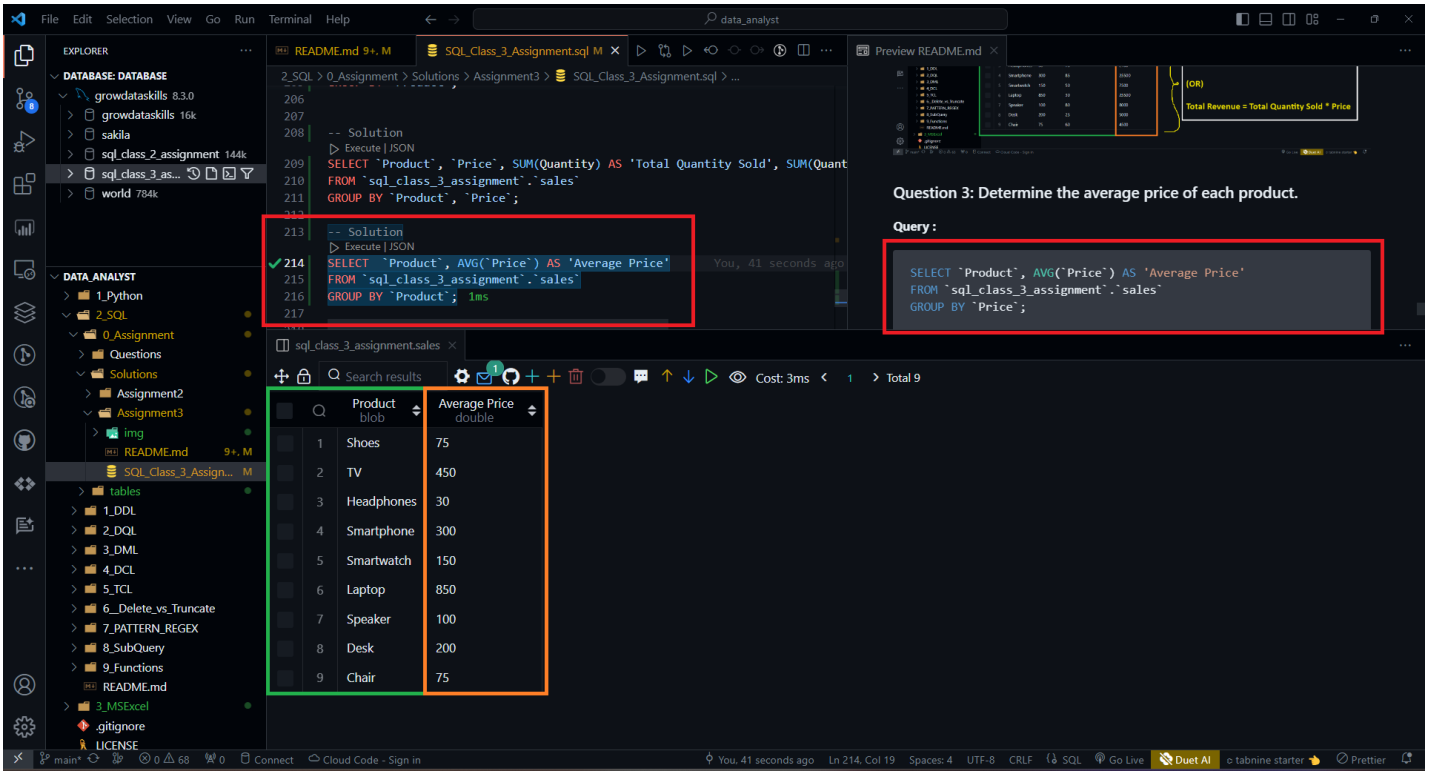


Question 3: Determine the average price of each product.

Query :

```
SELECT `Product`, AVG(`Price`) AS 'Average Price'
FROM `sql_class_3_assignment`.`sales`
GROUP BY `Price`;
```

Execution :



Question 4: Find the product with the highest total revenue (Quantity * Price)

Query :

```sql
SELECT `Product`, `Price`, SUM(`Quantity` * `Price`) AS `Total Quantity Sold`
FROM `sql_class_3_assignment`.`sales`
GROUP BY `Product`, `Price`
ORDER BY `Total Quantity Sold` DESC
LIMIT 1;
```

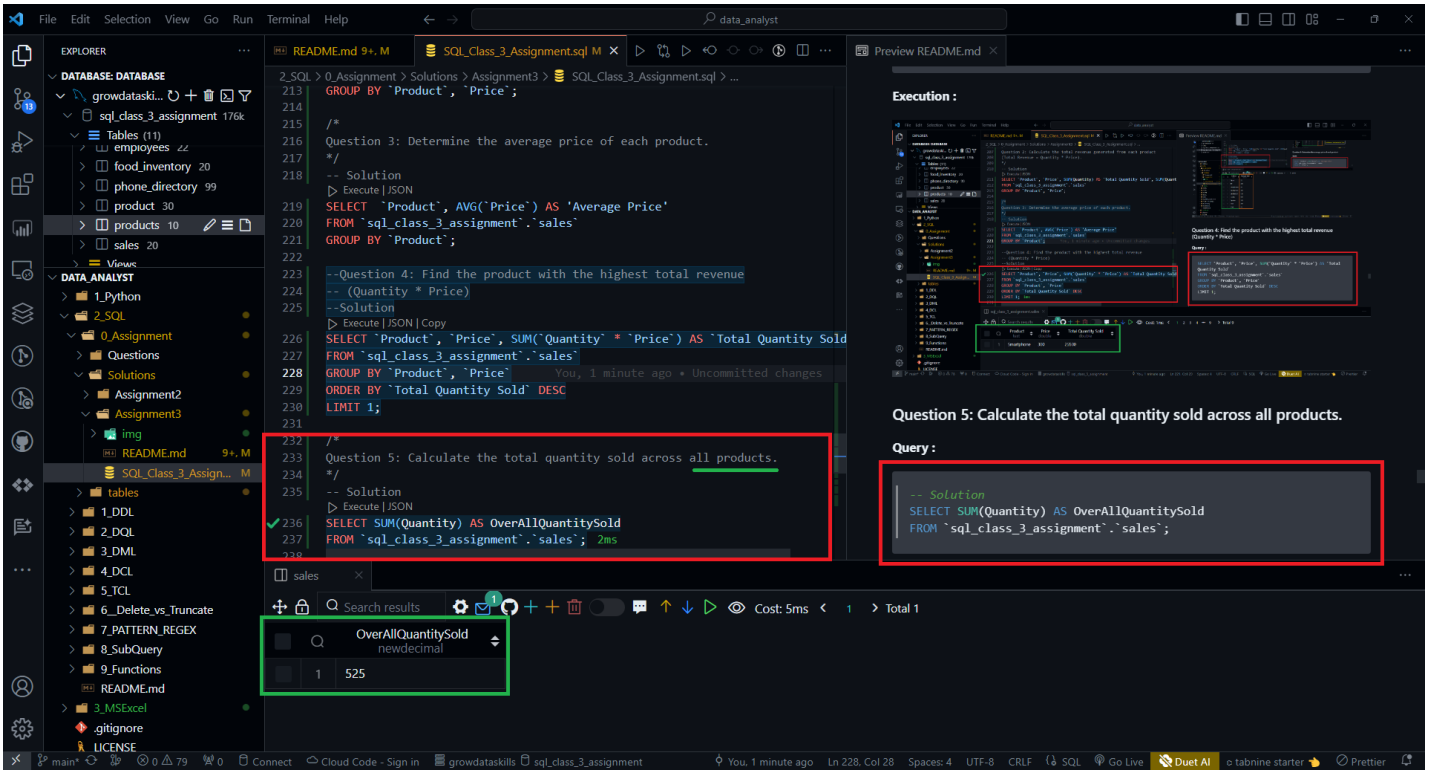
Execution :

Question 5: Calculate the total quantity sold across all products.

Query :

```sql
-- Solution
SELECT SUM(Quantity) AS OverAllQuantitySold
FROM `sql_class_3_assignment`.`sales`;
```

Execution :

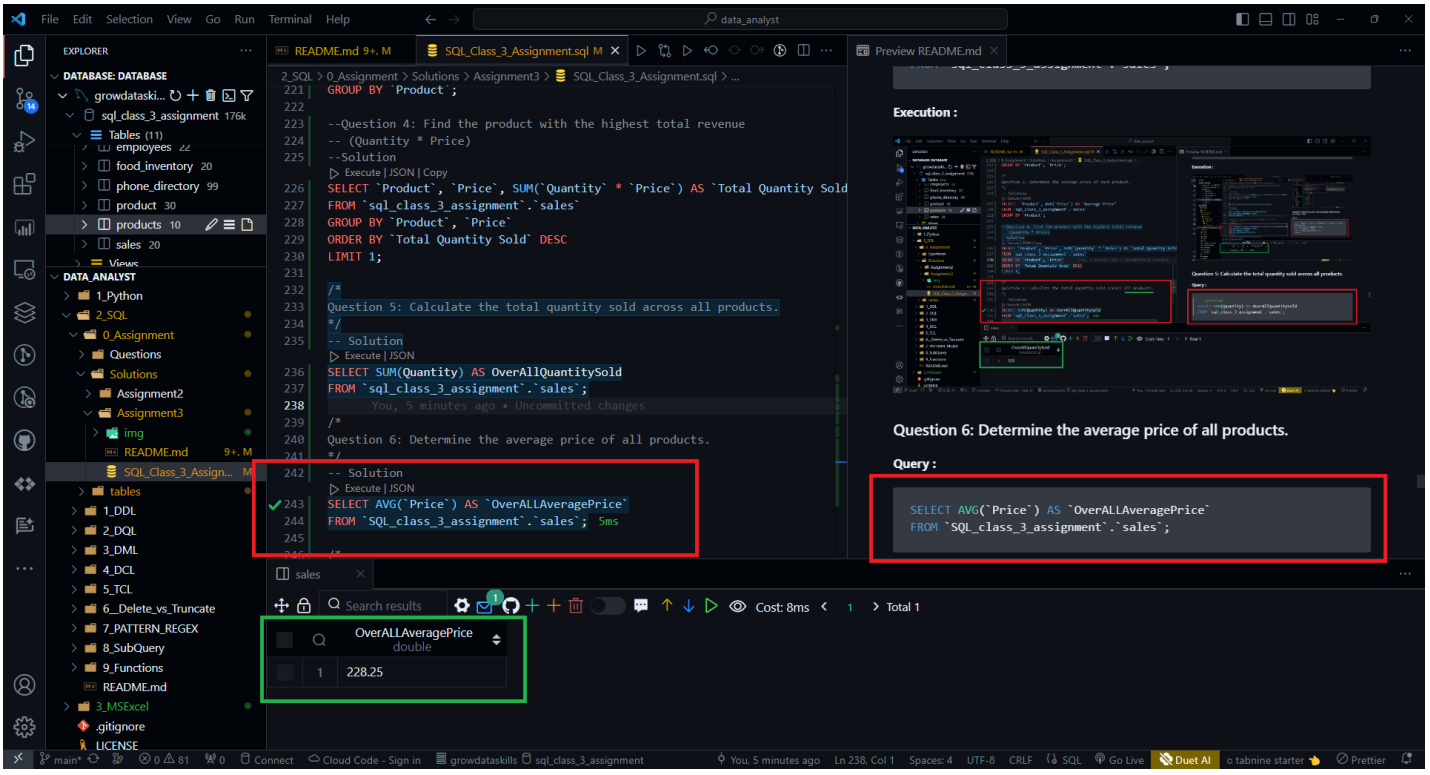


Question 6: Determine the average price of all products.

Query :

```
SELECT AVG(`Price`) AS `OverALLAveragePrice`
FROM `SQL_class_3_assignment`.`sales`;
```

Execution :



- Consider a table named Products with the following columns:

  ProductID (integer): The unique identifier for each product.

  ProductName (string): The name of the product.

  Price (decimal): The price of the product.

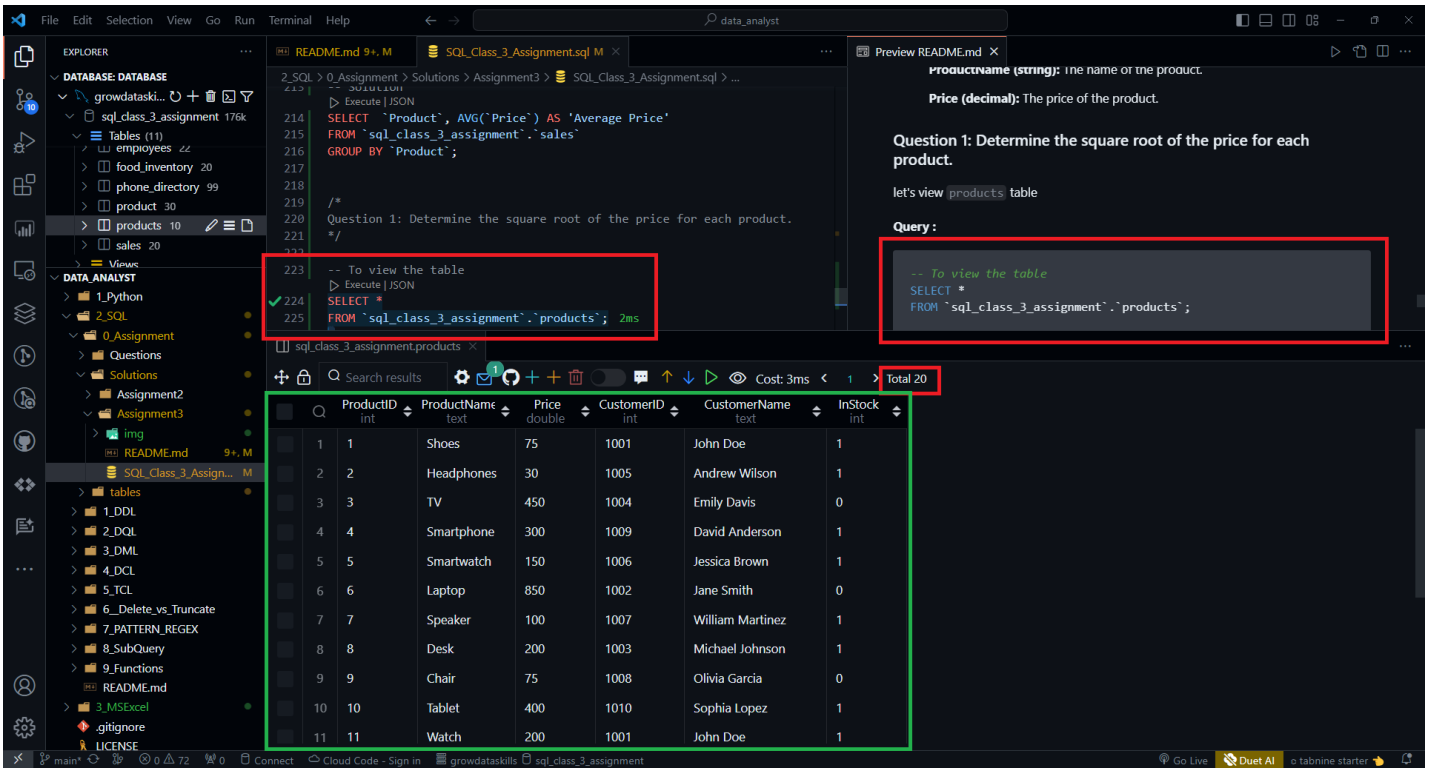
Question 1: Determine the square root of the price for each product.

let's view `products` table

Query :

```
-- To view the table
SELECT *
FROM `sql_class_3_assignment`.`products`;
```

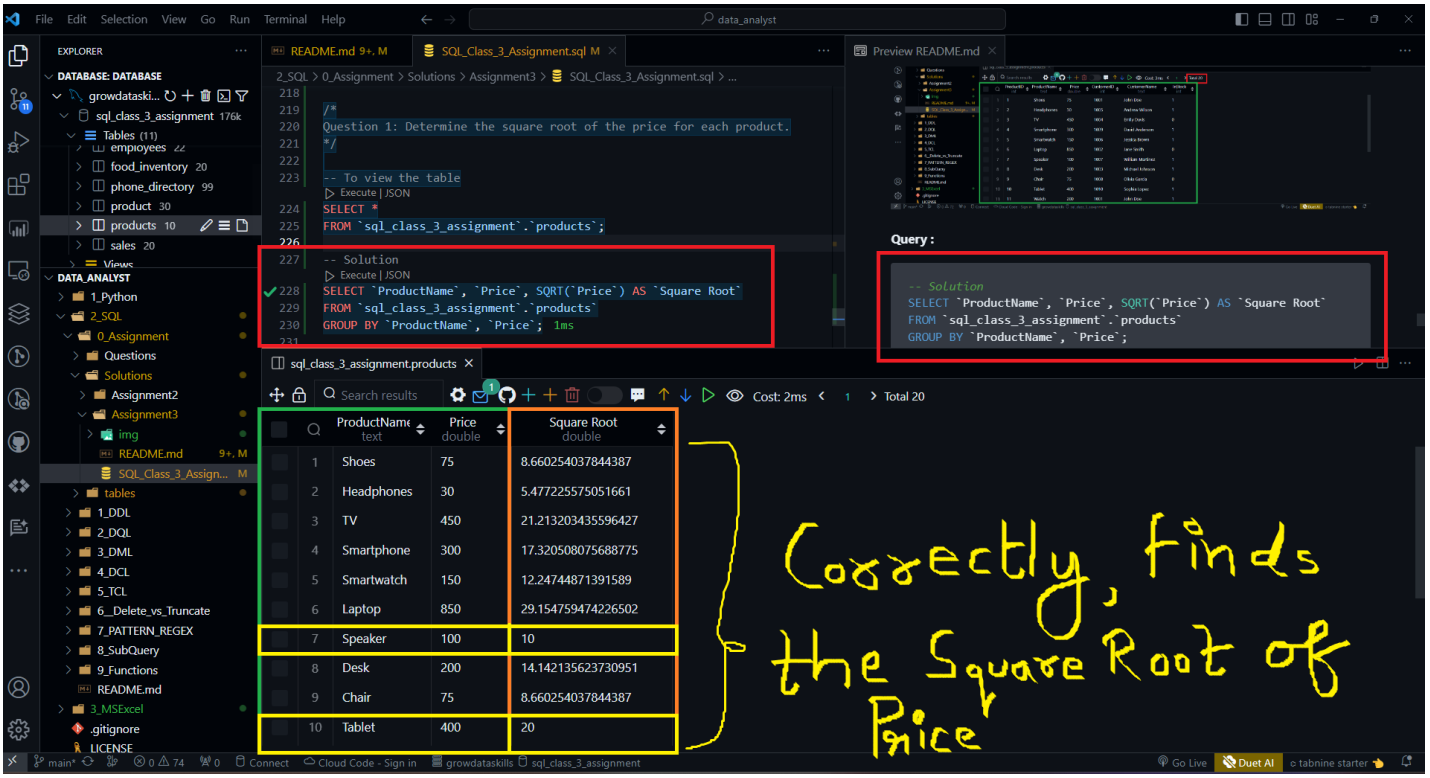
Execution :



Query :

```sql
-- Solution
SELECT `ProductName`, `Price`, SQRT(`Price`) AS `Square Root`
FROM `sql_class_3_assignment`.`products`
GROUP BY `ProductName`, `Price`;
```

Execution :

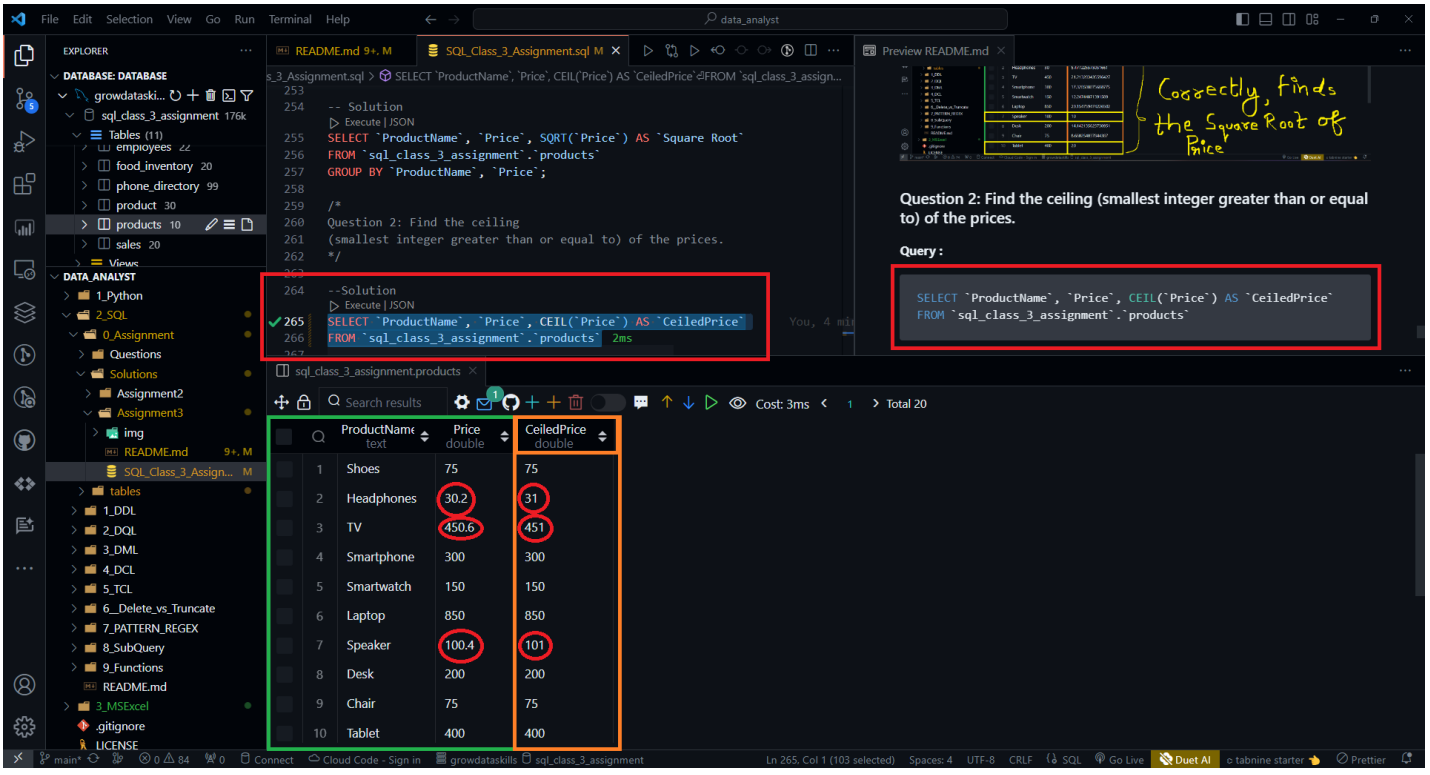


Question 2: Find the ceiling (smallest integer greater than or equal to) of the prices.

Query :

```
SELECT `ProductName`, `Price`, CEIL(`Price`) AS `CeiledPrice`
FROM `sql_class_3_assignment`.`products`
```

Execution :

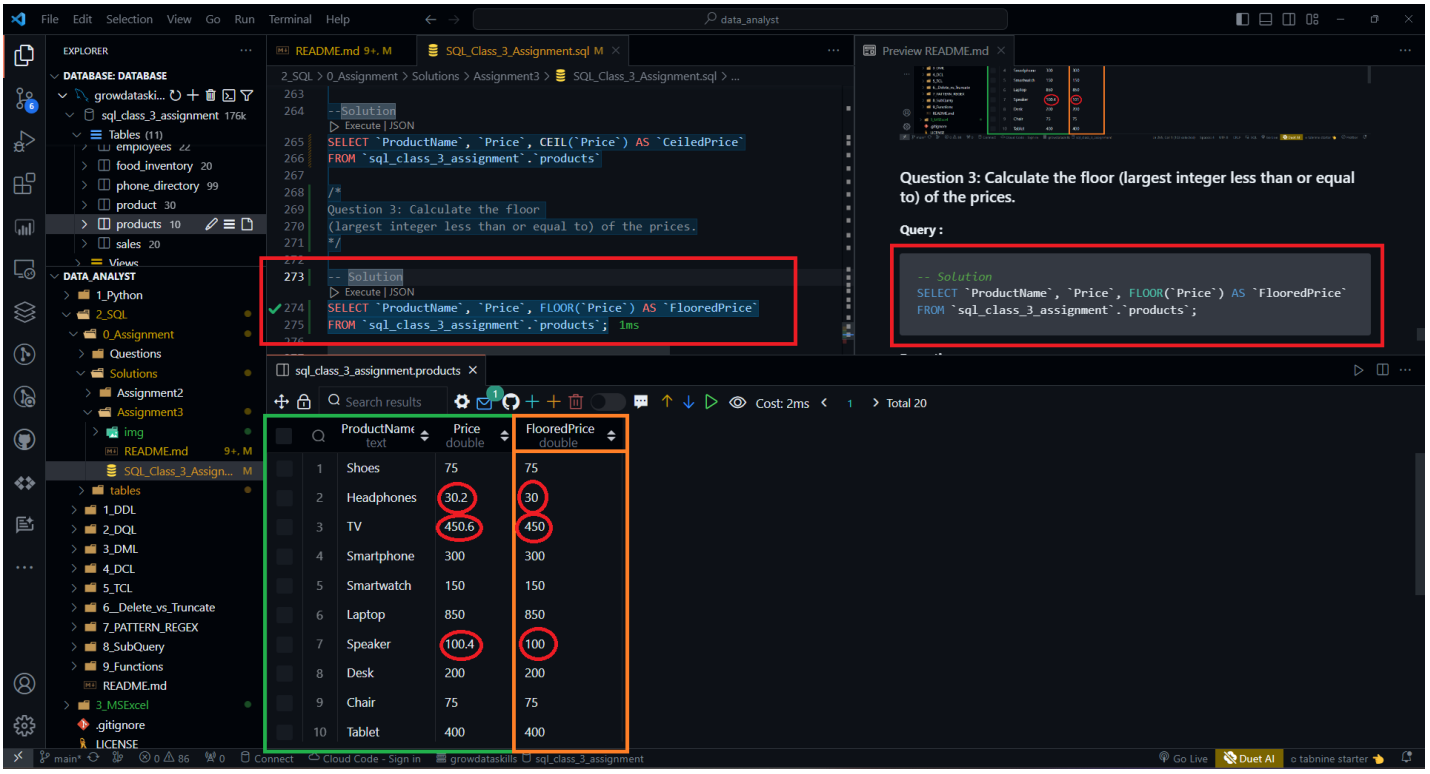


Question 3: Calculate the floor (largest integer less than or equal to) of the prices.

Query :

```
-- Solution
SELECT `ProductName`, `Price`, FLOOR(`Price`) AS `FlooredPrice`
FROM `sql_class_3_assignment`.`products`;
```

Execution :



- Consider a table named Orders with the following columns:

  OrderID (integer): The unique identifier for each order.

  OrderDate (datetime): The date and time when the order was placed.

  DeliveryDate (datetime): The date and time when the order was delivered.

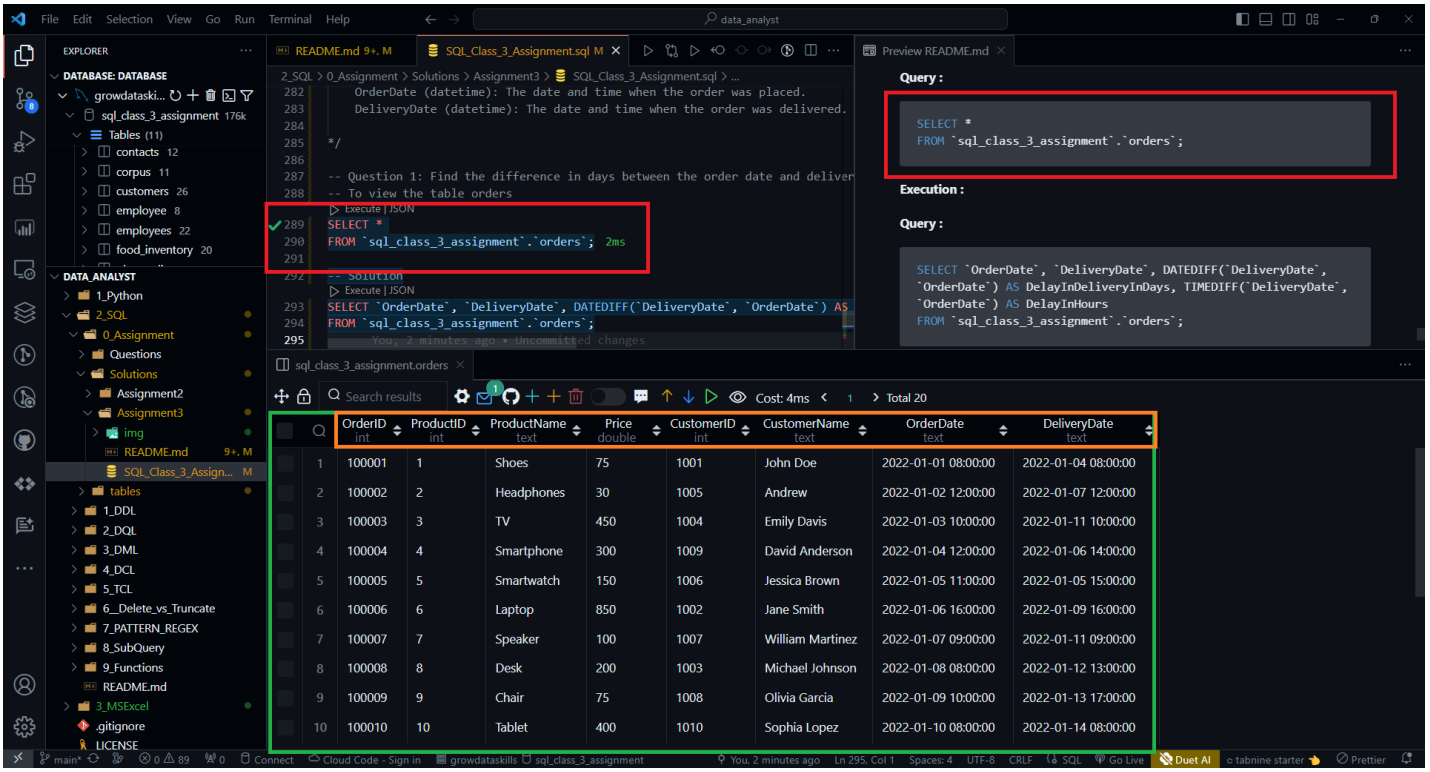
Question 1: Find the difference in days between the order date and delivery date for each order.

let's view `orders` table

Query :

```
SELECT *
FROM `sql_class_3_assignment`.`orders`;
```

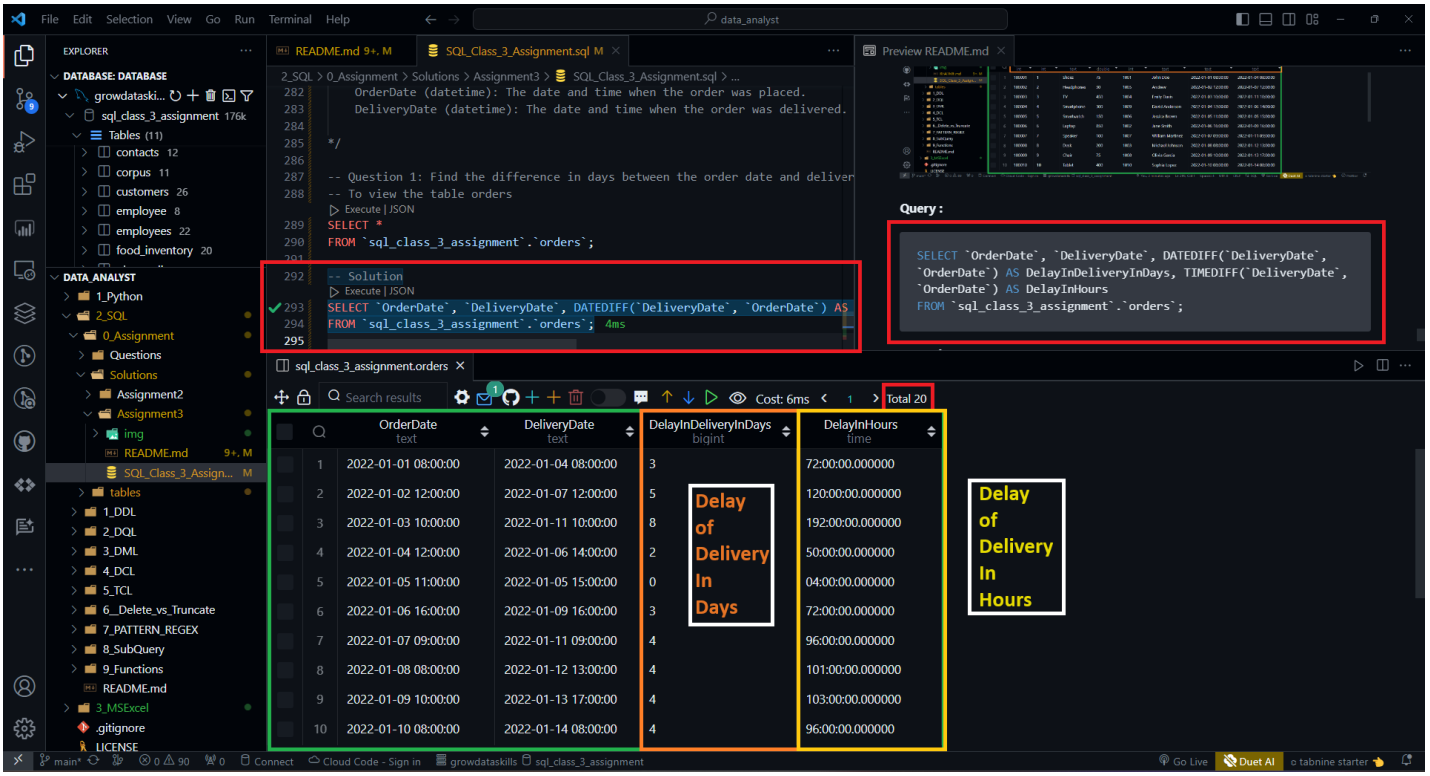
Execution :



Query :

```
SELECT `OrderDate`, `DeliveryDate`, DATEDIFF(`DeliveryDate`, `OrderDate`) AS DelayInDeliveryInDays,
TIMEDIFF(`DeliveryDate`, `OrderDate`) AS DelayInHours
FROM `sql_class_3_assignment`.`orders`;
```

Execution :

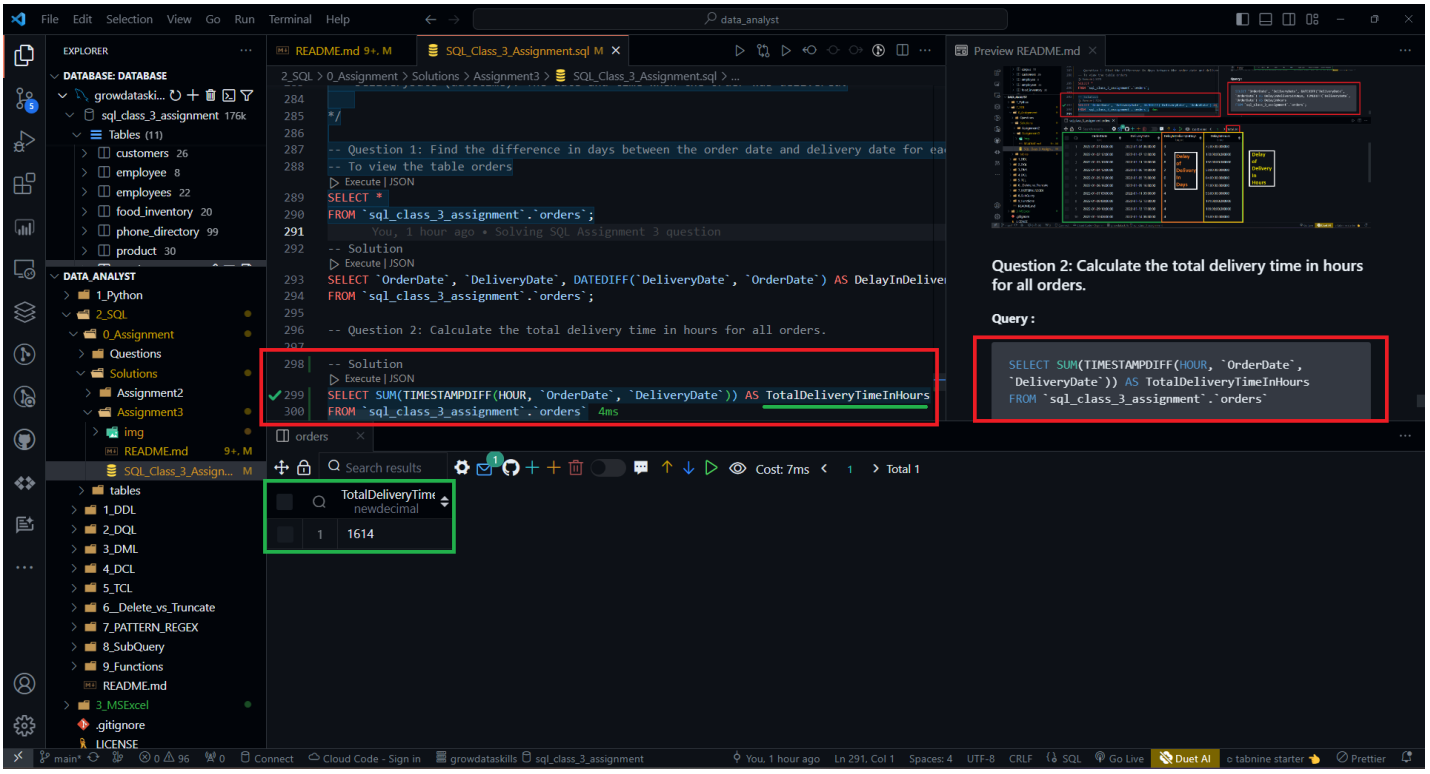


Question 2: Calculate the total delivery time in hours for all orders.

Query :

```
SELECT SUM(TIMESTAMPDIFF(HOUR, `OrderDate`, `DeliveryDate`)) AS TotalDeliveryTimeInHours
FROM `sql_class_3_assignment`.`orders`
```

Execution :



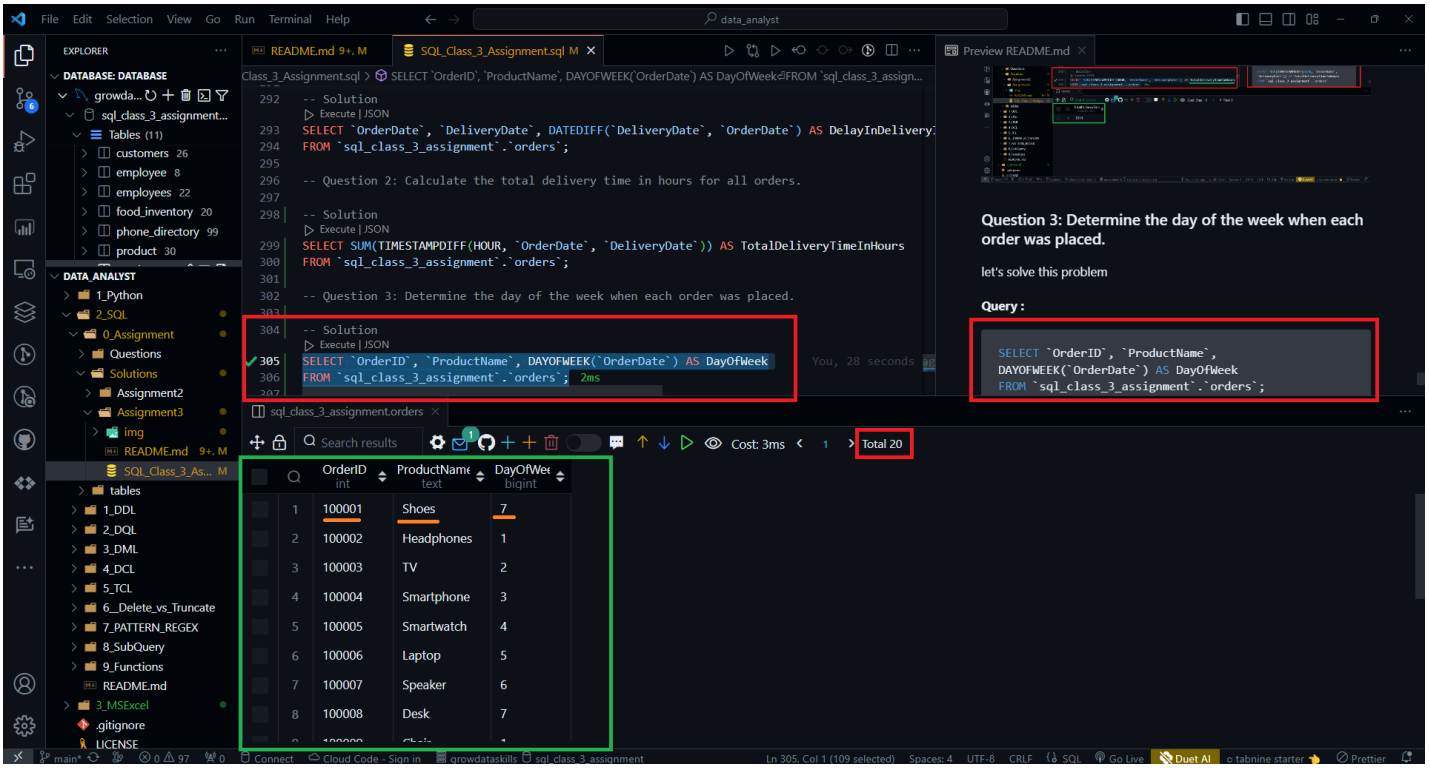
Question 3: Determine the day of the week when each order was placed.

let's solve this problem

Query :

```
SELECT `OrderID`, `ProductName`, DAYOFWEEK(`OrderDate`) AS DayOfWeek
FROM `sql_class_3_assignment`.`orders`;
```

Execution :

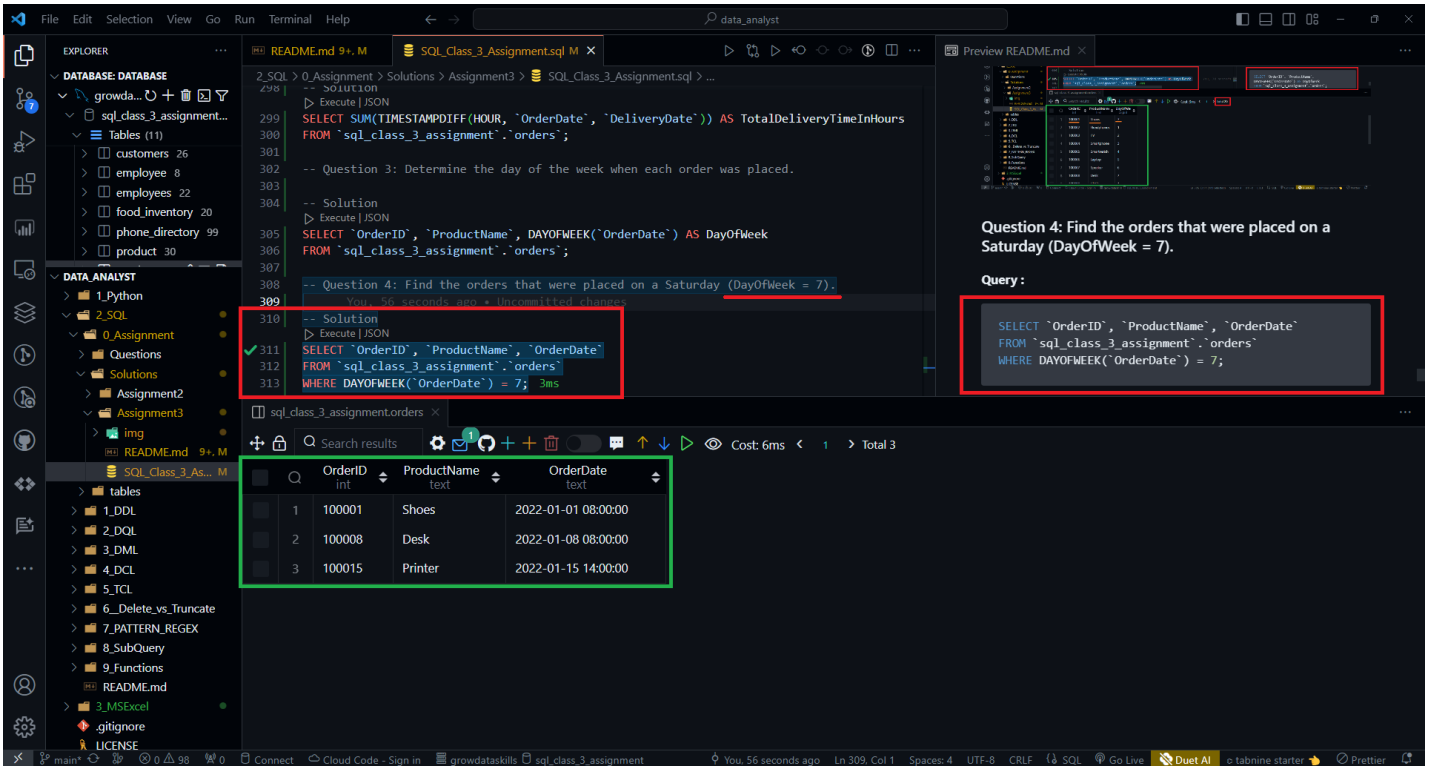


Question 4: Find the orders that were placed on a Saturday (DayOfWeek = 7).

Query :

```
SELECT `OrderID`, `ProductName`, `OrderDate`
FROM `sql_class_3_assignment`.`orders`
WHERE DAYOFWEEK(`OrderDate`) = 7;
```

Execution :

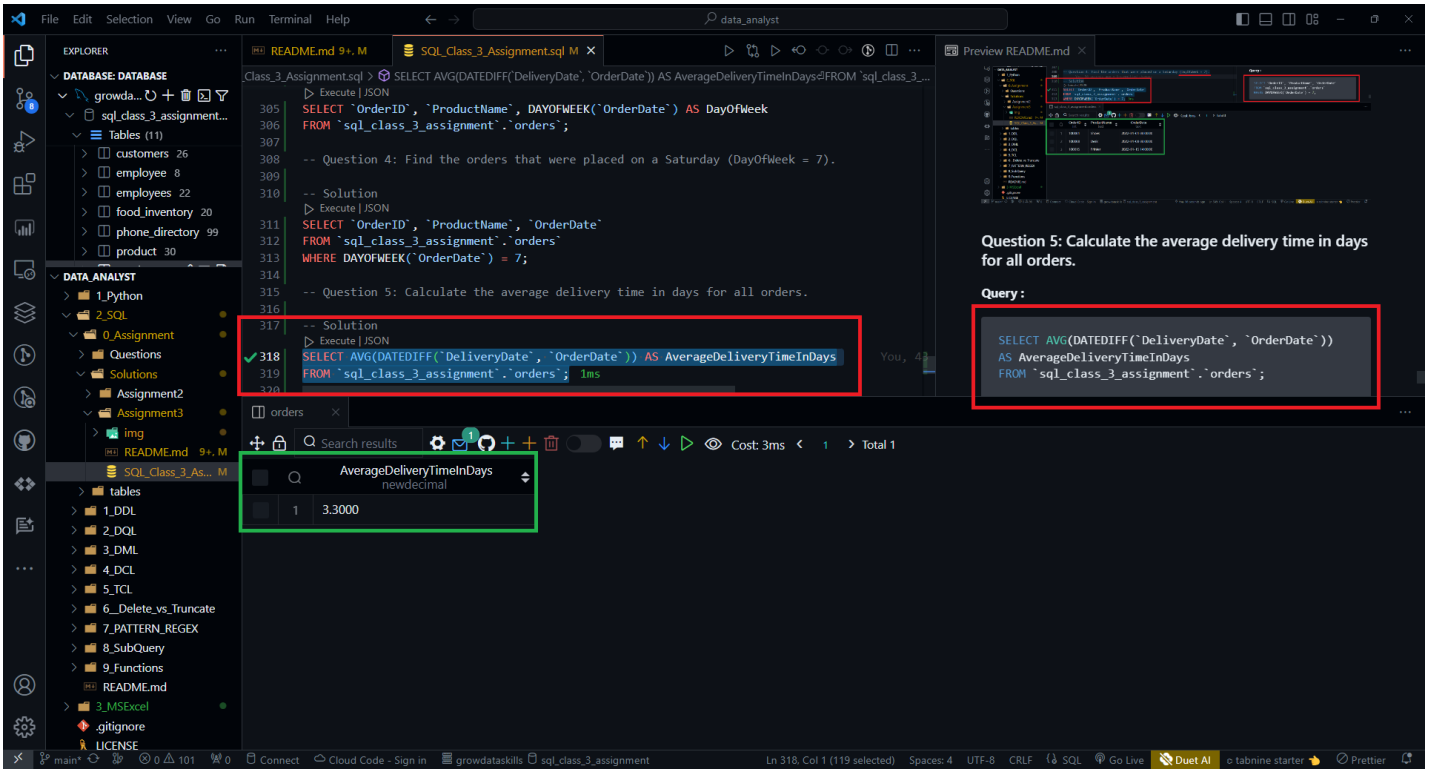


Question 5: Calculate the average delivery time in days for all orders.

Query :

```
SELECT AVG(DATEDIFF(`DeliveryDate`, `OrderDate`)) AS AverageDeliveryTimeInDays
FROM `sql_class_3_assignment`.`orders`;
```

Execution :

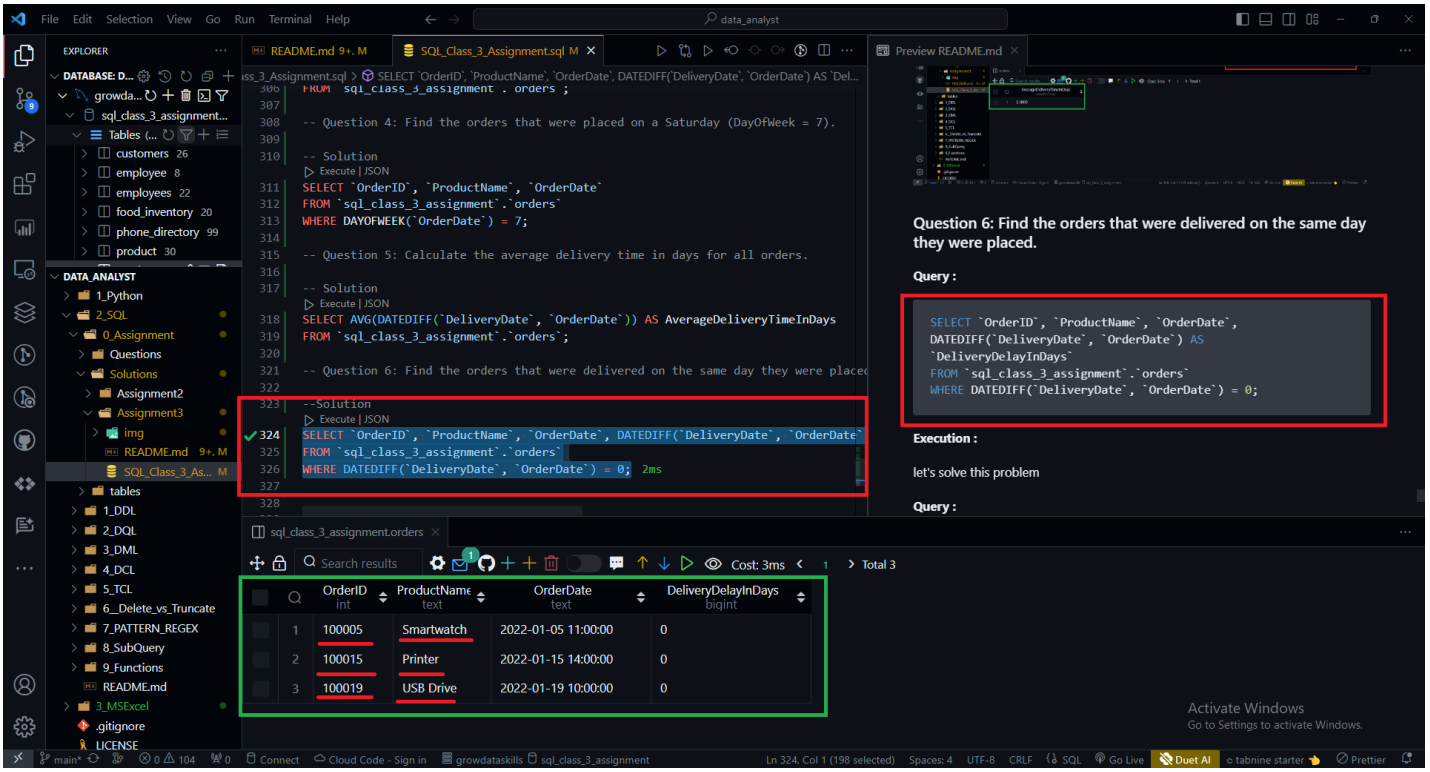


Question 6: Find the orders that were delivered on the same day they were placed.

Query :

```
SELECT `OrderID`, `ProductName`, `OrderDate`, DATEDIFF(`DeliveryDate`, `OrderDate`) AS
`DeliveryDelayInDays`
FROM `sql_class_3_assignment`.`orders`
WHERE DATEDIFF(`DeliveryDate`, `OrderDate`) = 0;
```

Execution :



GROUP BY, HAVING, ORDER BY

Question 1: Consider a table named Orders with the following columns:

OrderID (integer): The unique identifier for each order.

CustomerID (integer): The unique identifier for each customer.

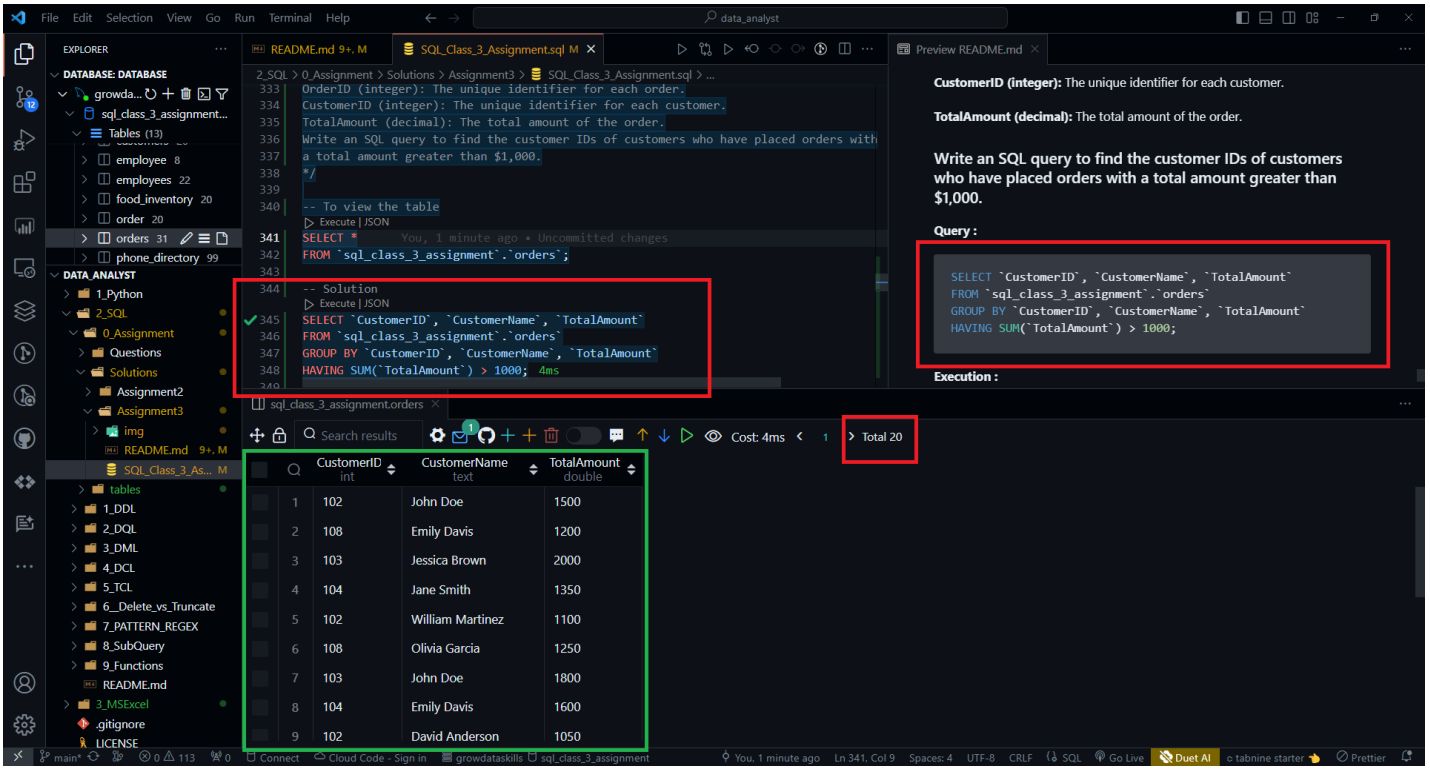
TotalAmount (decimal): The total amount of the order.

Write an SQL query to find the customer IDs of customers who have placed orders with a total amount greater than $1,000.

Query :

```
SELECT `CustomerID`, `CustomerName`, `TotalAmount`
FROM `sql_class_3_assignment`.`orders`
GROUP BY `CustomerID`, `CustomerName`, `TotalAmount`
HAVING SUM(`TotalAmount`) > 1000;
```

Execution :



Question 2:Consider a table named Sales with the following columns:

ProductID (integer): The unique identifier for each product.

SaleDate (date): The date of the sale.

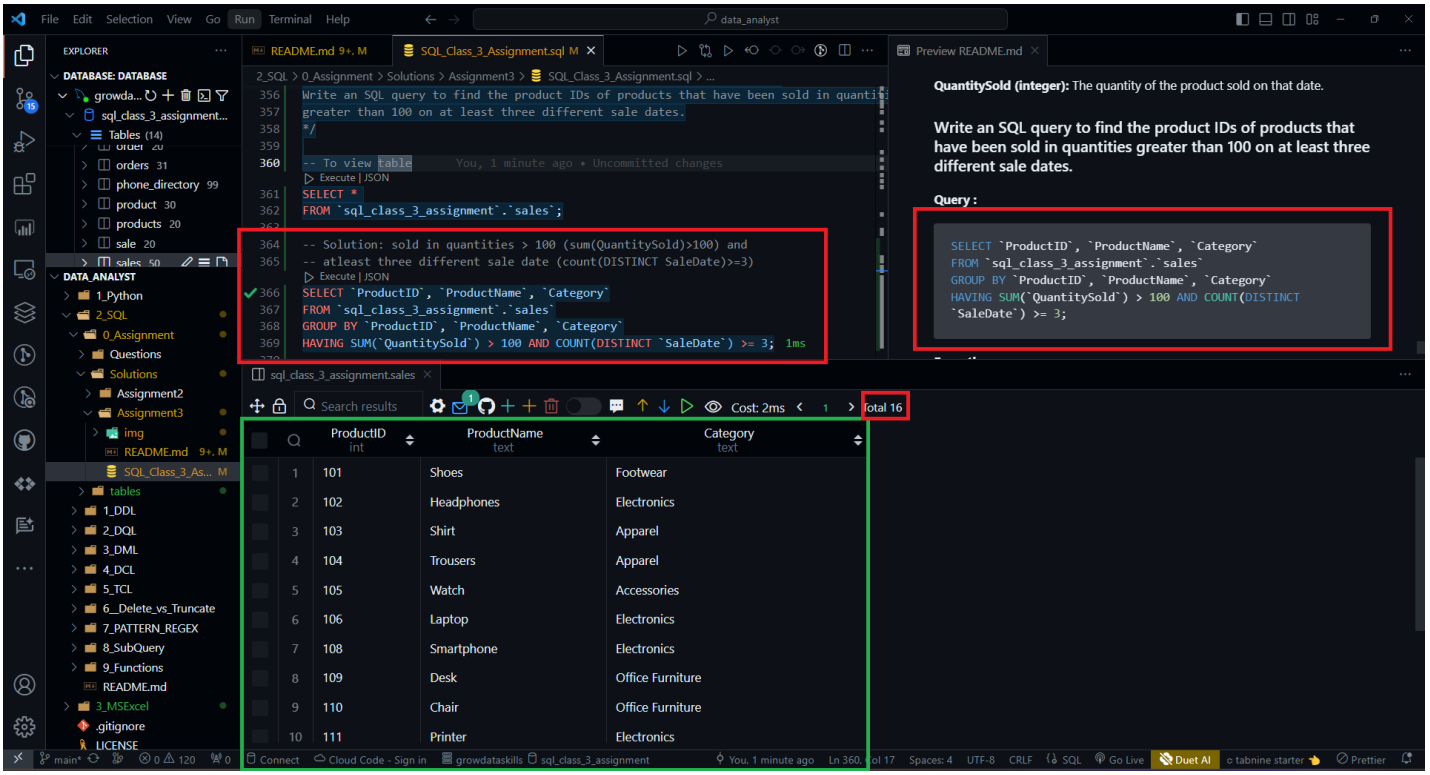
QuantitySold (integer): The quantity of the product sold on that date.

Write an SQL query to find the product IDs of products that have been sold in quantities greater than 100 on at least three different sale dates.

Query :

```
SELECT `ProductID`, `ProductName`, `Category`
FROM `sql_class_3_assignment`.`sales`
GROUP BY `ProductID`, `ProductName`, `Category`
HAVING SUM(`QuantitySold`) > 100 AND COUNT(DISTINCT `SaleDate`) >= 3;
```

Execution :



Question 3:Consider a table named Employees with the following columns:

EmployeeID (integer): The unique identifier for each employee.

Department (string): The department in which the employee works.

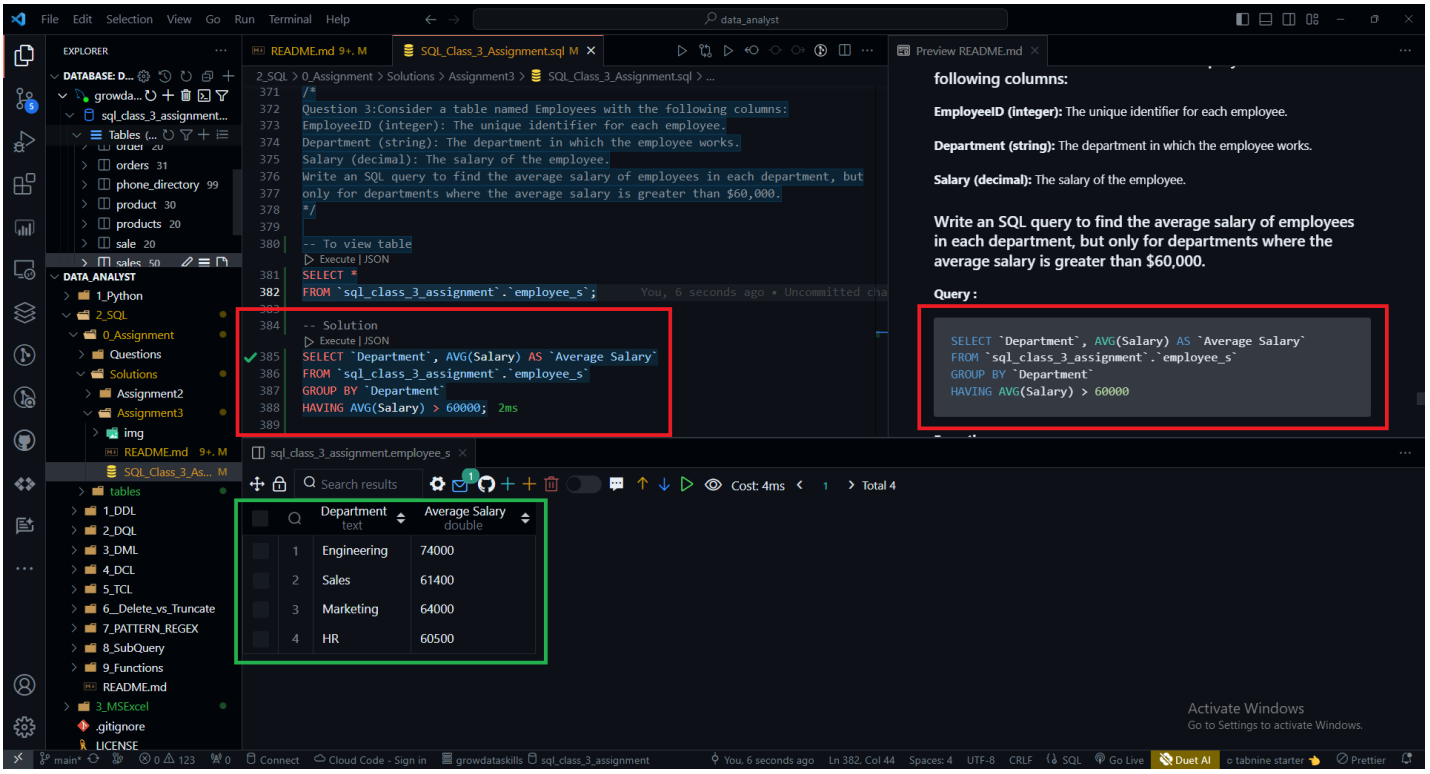
Salary (decimal): The salary of the employee.

Write an SQL query to find the average salary of employees in each department, but only for departments where the average salary is greater than $60,000.

Query :

```
SELECT `Department`, AVG(Salary) AS `Average Salary`
FROM `sql_class_3_assignment`.`employee_s`
GROUP BY `Department`
HAVING AVG(Salary) > 60000
```

Execution :



Question 4:Consider a table named Students with the following columns:

StudentID (integer): The unique identifier for each student.

Course (string): The course name.

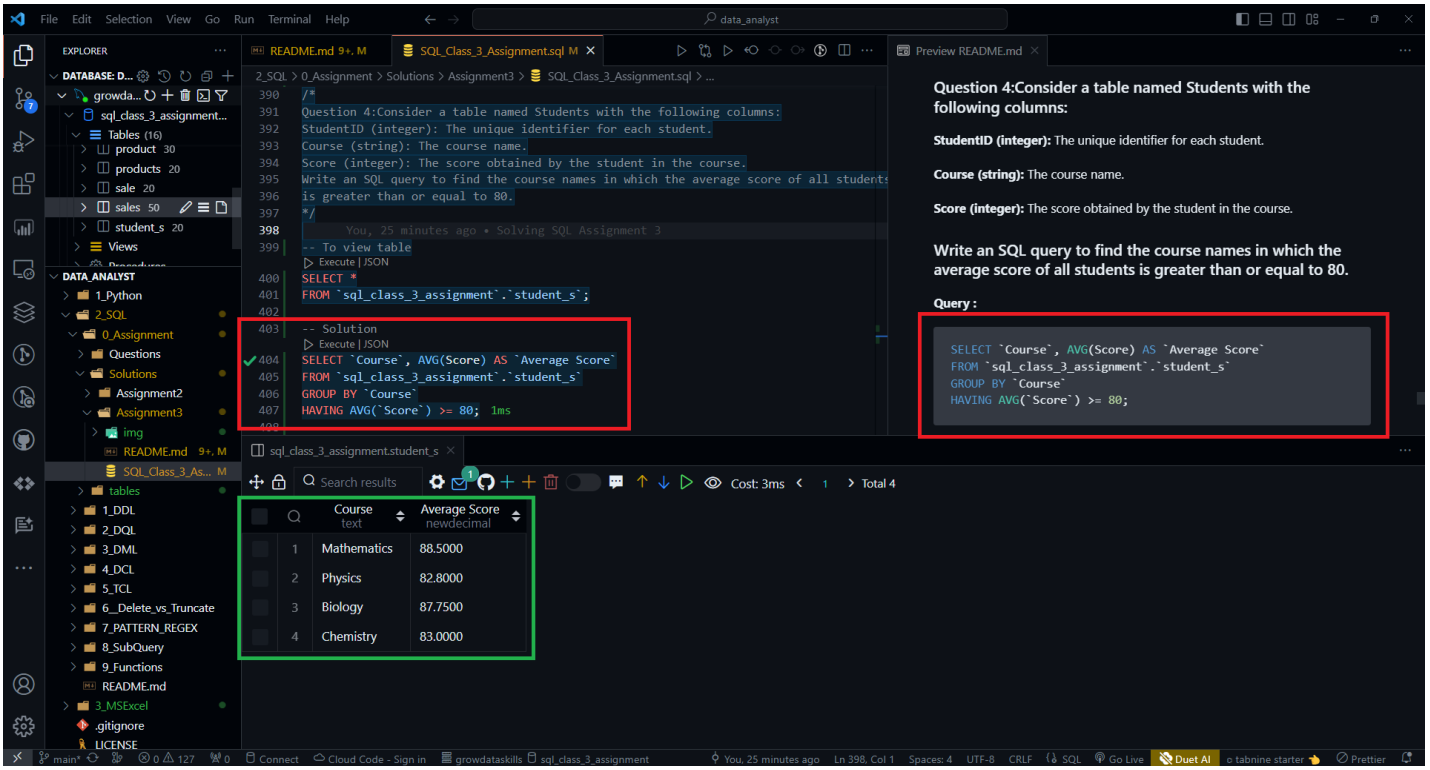
Score (integer): The score obtained by the student in the course.

Write an SQL query to find the course names in which the average score of all students is greater than or equal to 80.

Query :

```
SELECT `Course`, AVG(Score) AS `Average Score`
FROM `sql_class_3_assignment`.`student_s`
GROUP BY `Course`
HAVING AVG(`Score`) >= 80;
```

Execution :



Question 5: Consider a table named Employees with the following columns:

EmployeeID (integer): The unique identifier for each employee.

Department (string): The department in which the employee works.

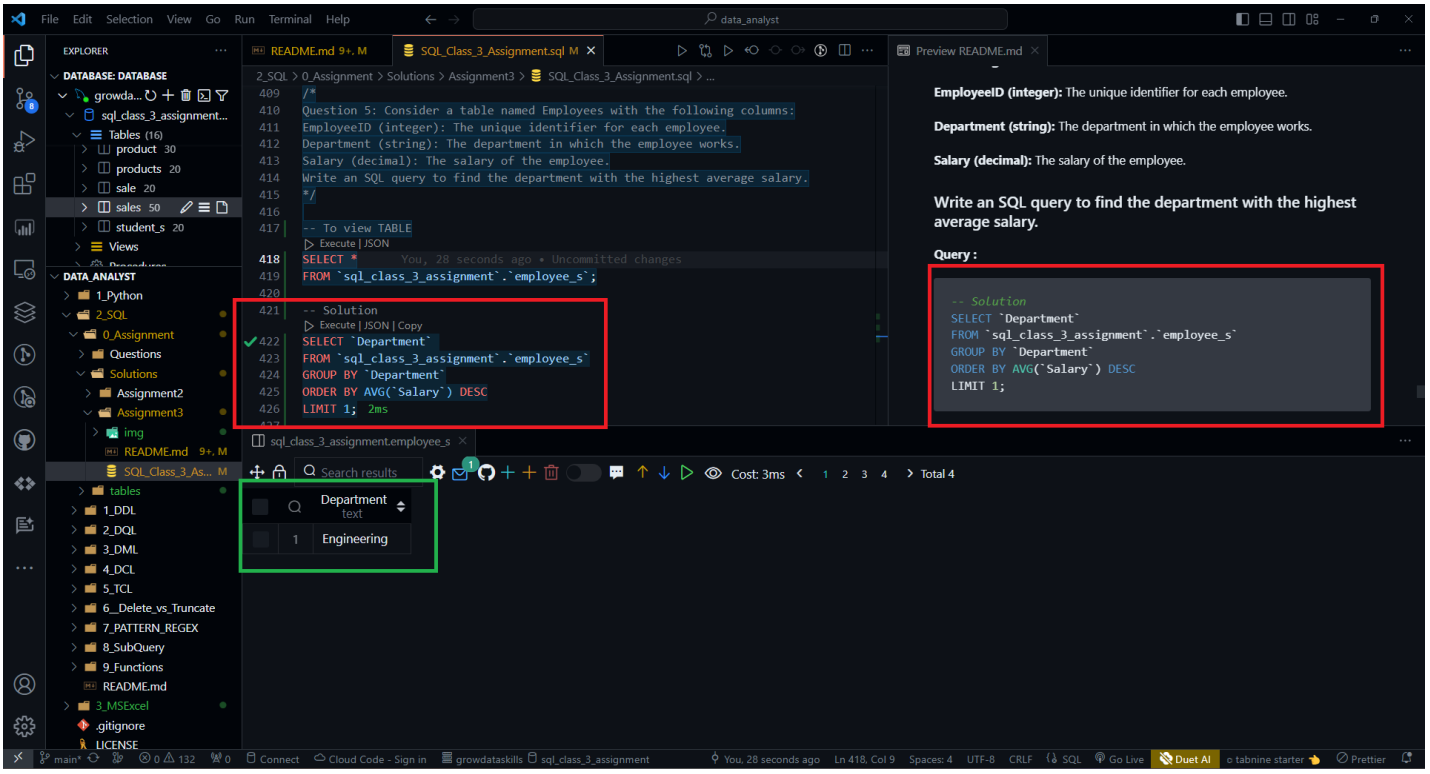
Salary (decimal): The salary of the employee.

Write an SQL query to find the department with the highest average salary.

Query :

```
-- Solution
SELECT `Department`
FROM `sql_class_3_assignment`.`employee_s`
GROUP BY `Department`
ORDER BY AVG(`Salary`) DESC
LIMIT 1;
```

Execution :



Question 6: Consider a table named Sales with the following columns:

ProductID (integer): The unique identifier for each product.

SaleDate (date): The date of the sale.

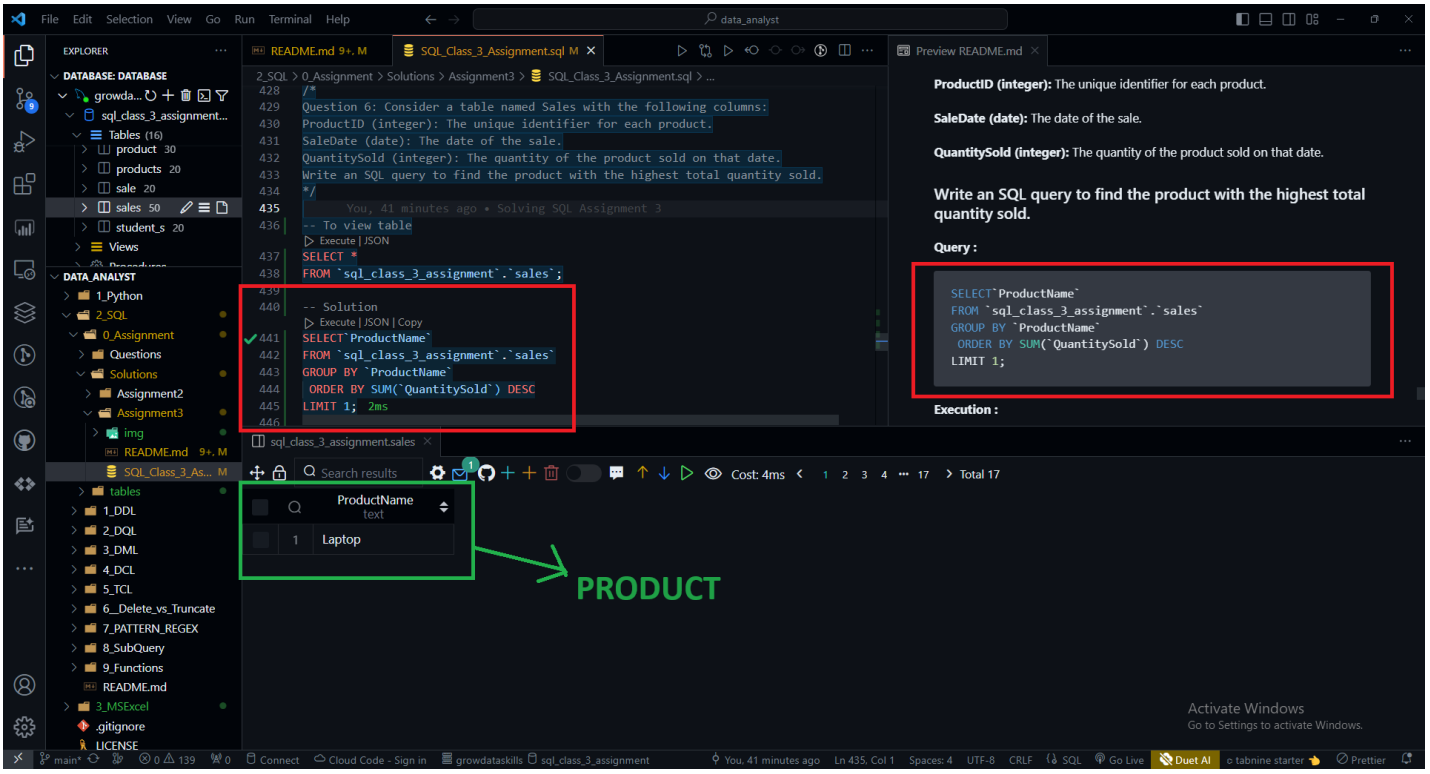
QuantitySold (integer): The quantity of the product sold on that date.

Write an SQL query to find the product with the highest total quantity sold.

Query :

```
SELECT`ProductName`
FROM `sql_class_3_assignment`.`sales`
GROUP BY `ProductName`
ORDER BY SUM(`QuantitySold`) DESC
LIMIT 1;
```

Execution :



Question 7: Consider a table named Students with the following columns:

StudentID (integer): The unique identifier for each student.

Course (string): The course name.

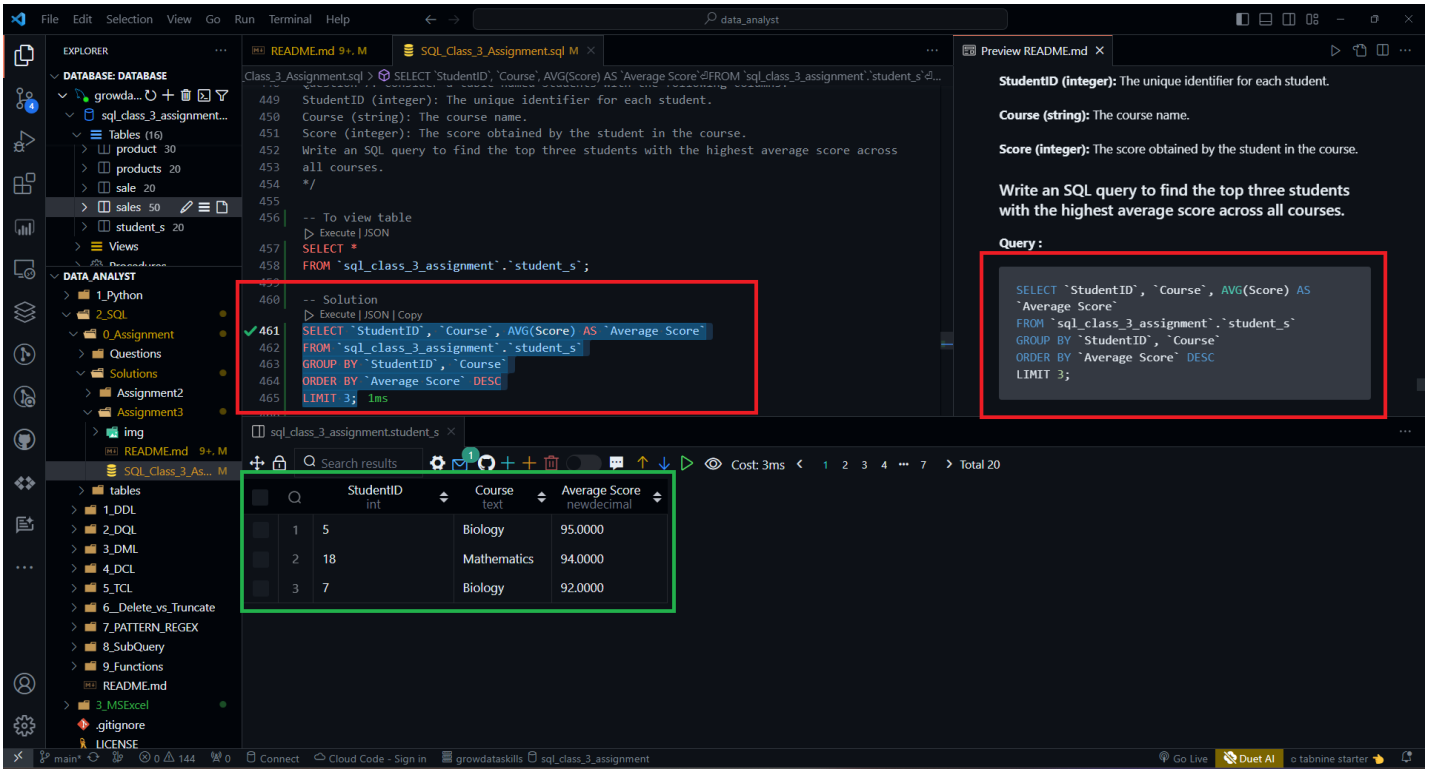
Score (integer): The score obtained by the student in the course.

Write an SQL query to find the top three students with the highest average score across all courses.

Query :

```
SELECT `StudentID`, `Course`, AVG(Score) AS `Average Score`
FROM `sql_class_3_assignment`.`student_s`
GROUP BY `StudentID`, `Course`
ORDER BY `Average Score` DESC
LIMIT 3;
```

Execution :



Question 8: Consider a table named Orders with the following columns:

OrderID (integer): The unique identifier for each order.

CustomerID (integer): The unique identifier for each customer.

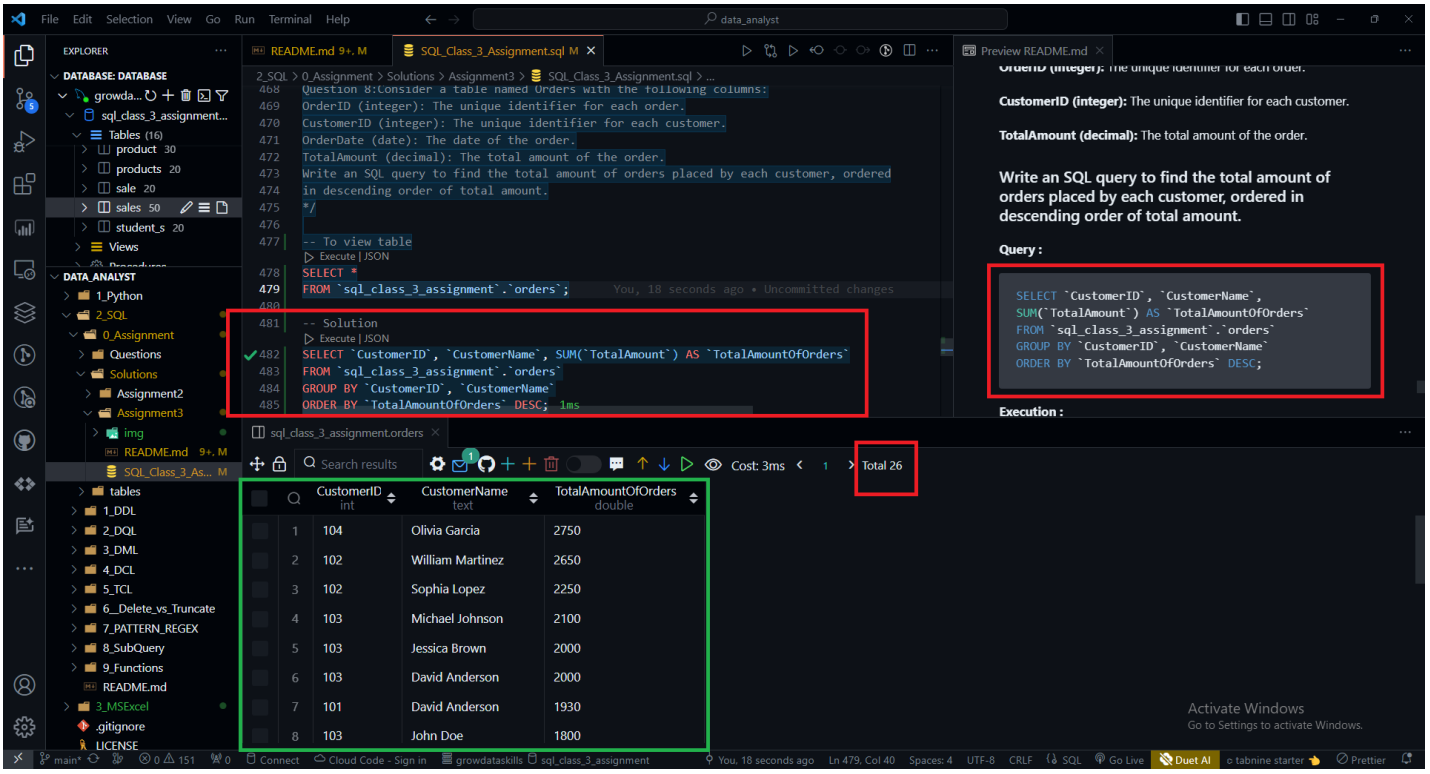
TotalAmount (decimal): The total amount of the order.

Write an SQL query to find the total amount of orders placed by each customer, ordered in descending order of total amount.

Query :

```
SELECT `CustomerID`, `CustomerName`, SUM(`TotalAmount`) AS `TotalAmountOfOrders`
FROM `sql_class_3_assignment`.`orders`
GROUP BY `CustomerID`, `CustomerName`
ORDER BY `TotalAmountOfOrders` DESC;
```

Execution :



Question 9: Consider a table named Books with the following columns:

BookID (integer): The unique identifier for each book.

Author (string): The author of the book.

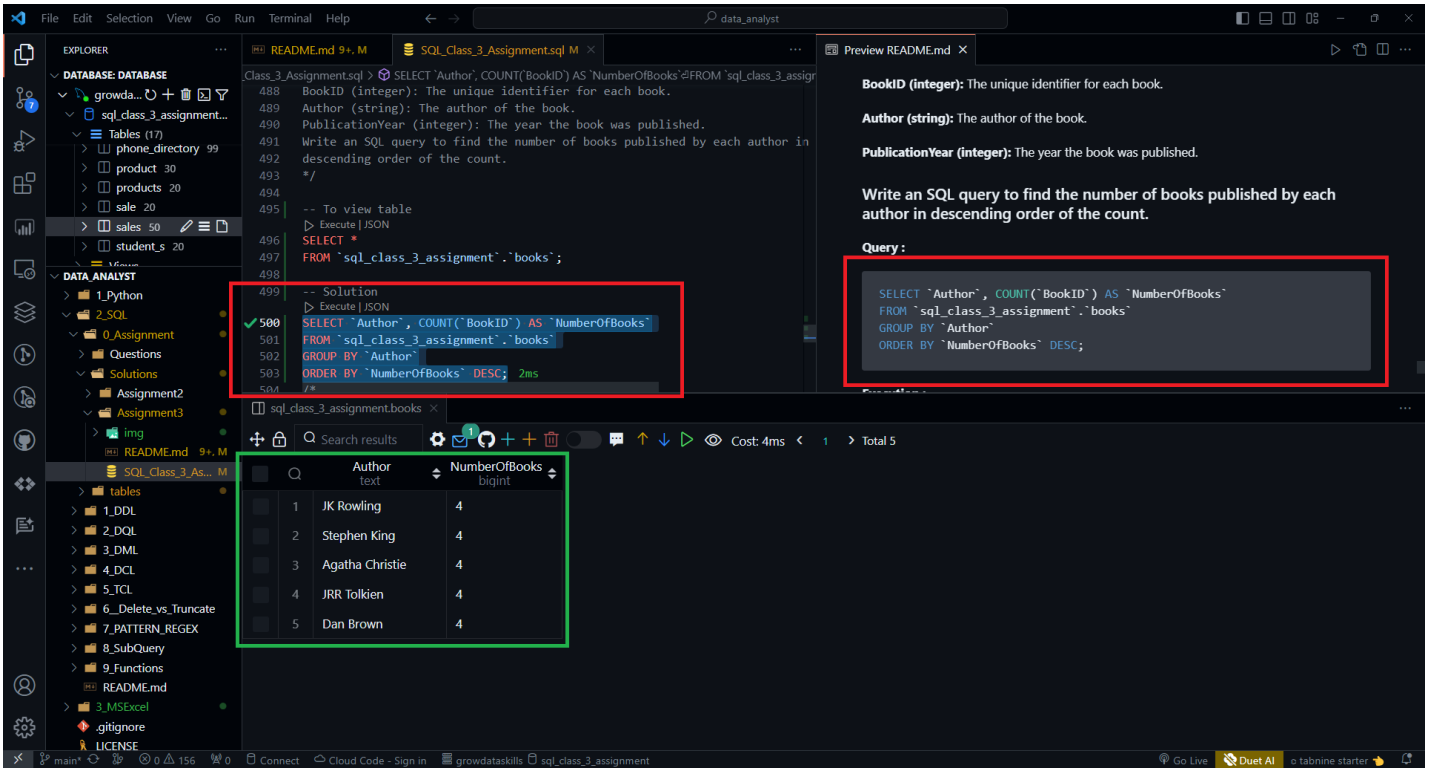
PublicationYear (integer): The year the book was published.

Write an SQL query to find the number of books published by each author in descending order of the count.

Query :

```
SELECT `Author`, COUNT(`BookID`) AS `NumberOfBooks`
FROM `sql_class_3_assignment`.`books`
GROUP BY `Author`
ORDER BY `NumberOfBooks` DESC;
```

Execution :



Question 8: Consider a table named Orders with the following columns:

OrderID (integer): The unique identifier for each order.

CustomerID (integer): The unique identifier for each customer.

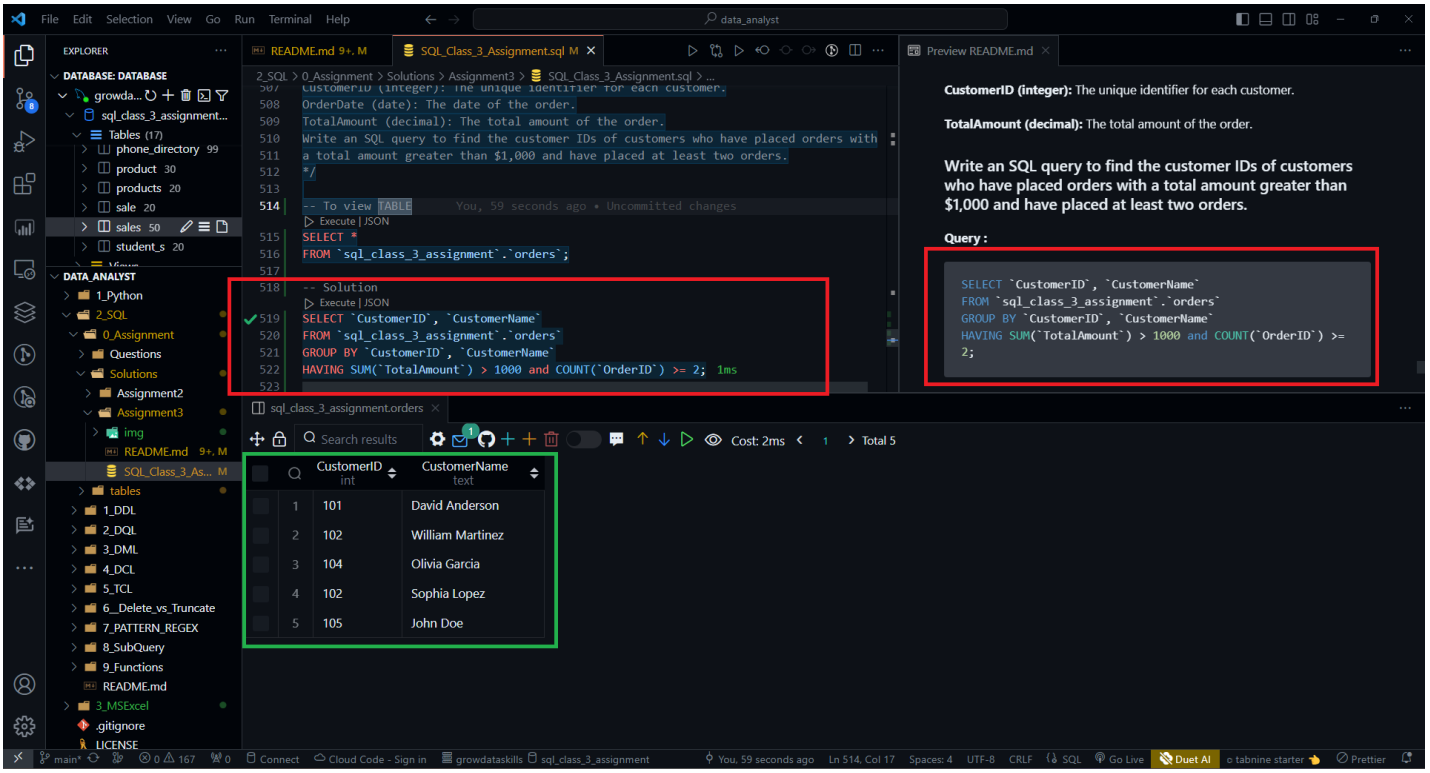
TotalAmount (decimal): The total amount of the order.

Write an SQL query to find the customer IDs of customers who have placed orders with a total amount greater than $1,000 and have placed at least two orders.

Query :

```
SELECT `CustomerID`, `CustomerName`
FROM `sql_class_3_assignment`.`orders`
GROUP BY `CustomerID`, `CustomerName`
HAVING SUM(`TotalAmount`) > 1000 and COUNT(`OrderID`) >= 2;
```

Execution :



Question 11: Consider a table named Products with the following columns:

ProductID (integer): The unique identifier for each product.

Category (string): The category of the product.

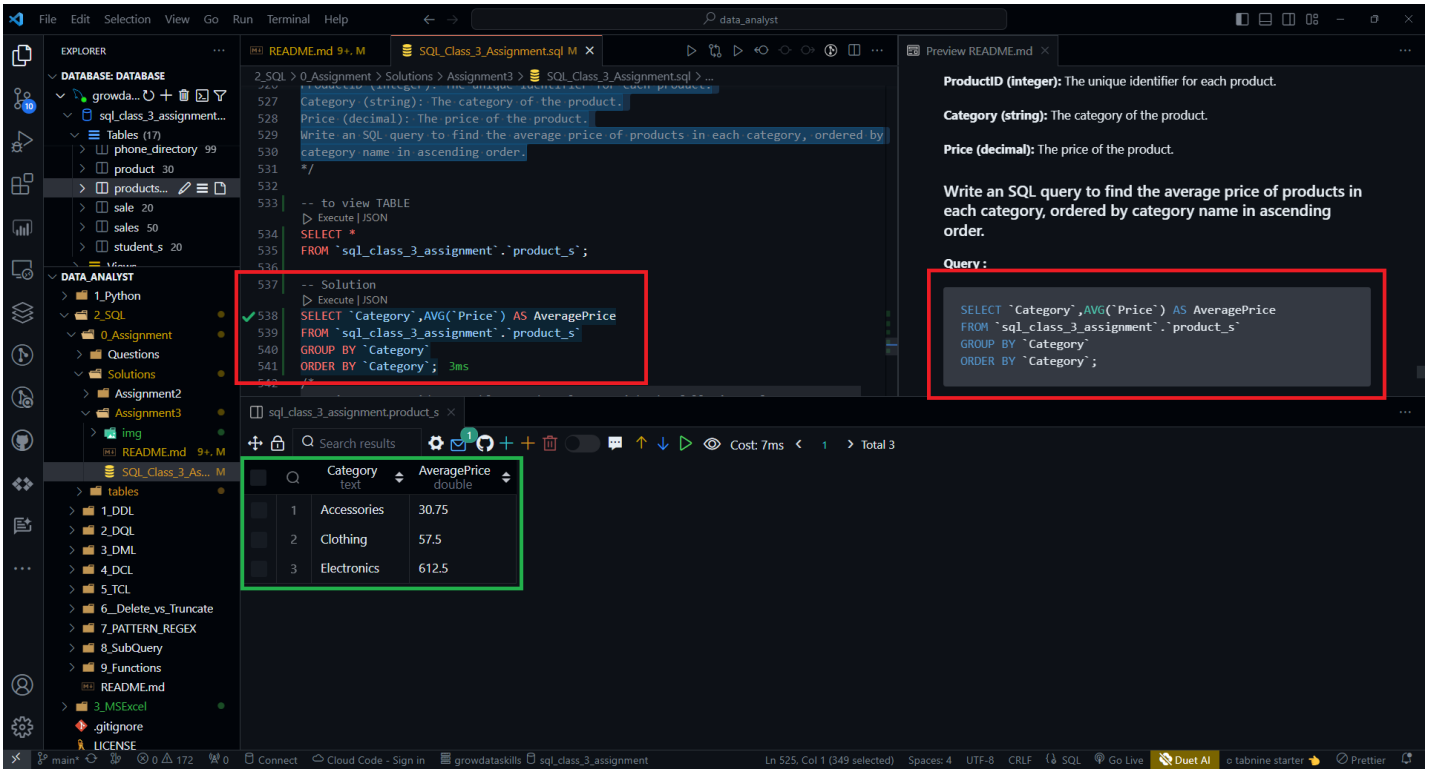
Price (decimal): The price of the product.

Write an SQL query to find the average price of products in each category, ordered by category name in ascending order.

Query :

```
SELECT `Category`,AVG(`Price`) AS AveragePrice
FROM `sql_class_3_assignment`.`product_s`
GROUP BY `Category`
ORDER BY `Category`;
```

Execution :



Question 12: Consider a table named Employees with the following columns:

EmployeeID (integer): The unique identifier for each employee.

Department (string): The department in which the employee works.

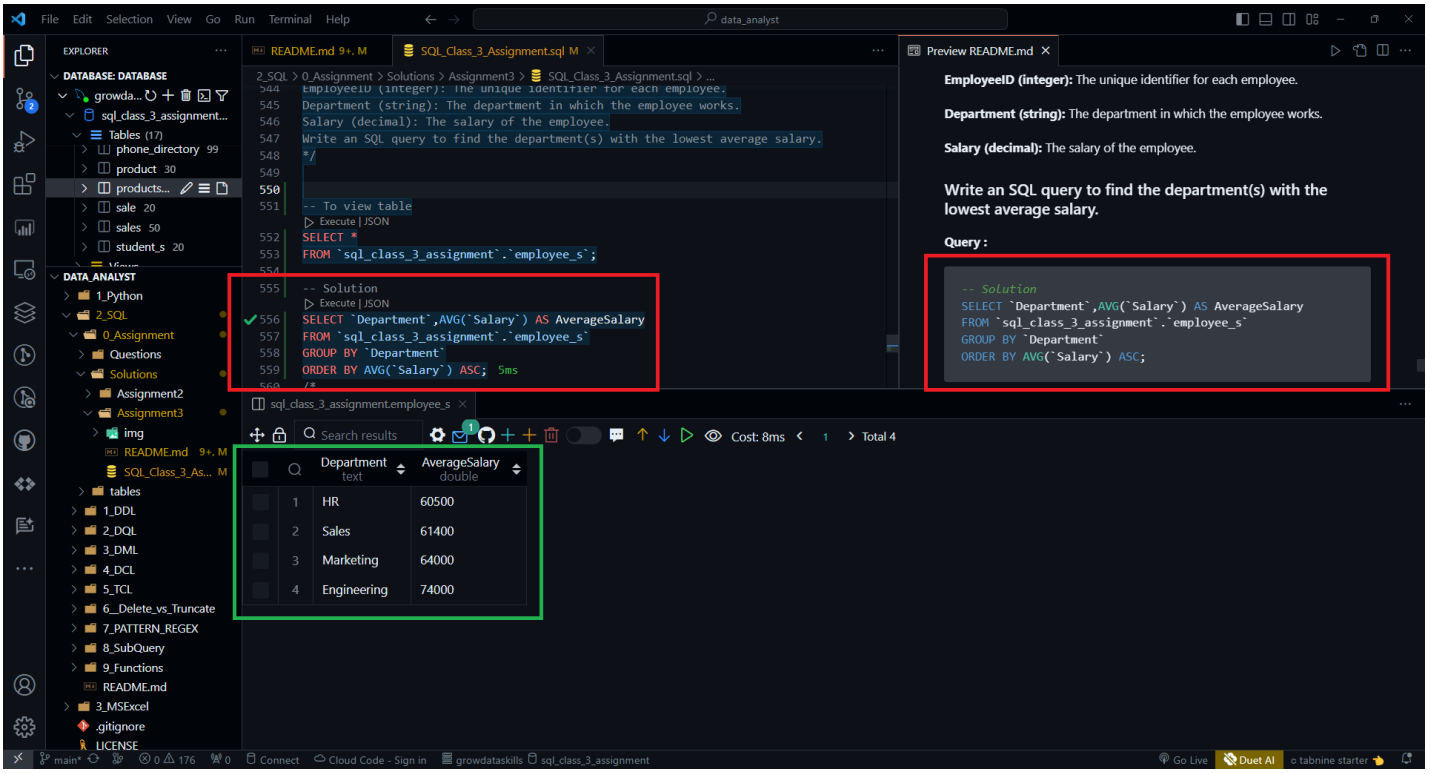
Salary (decimal): The salary of the employee.

Write an SQL query to find the department(s) with the lowest average salary.

Query :

```
-- Solution
SELECT `Department`,AVG(`Salary`) AS AverageSalary
FROM `sql_class_3_assignment`.`employee_s`
GROUP BY `Department`
ORDER BY AVG(`Salary`) ASC;
```

Execution :



Question 13: Consider a table named Orders with the following columns:

OrderID (integer): The unique identifier for each order.

CustomerID (integer): The unique identifier for each customer.

OrderDate (date): The date of the order.

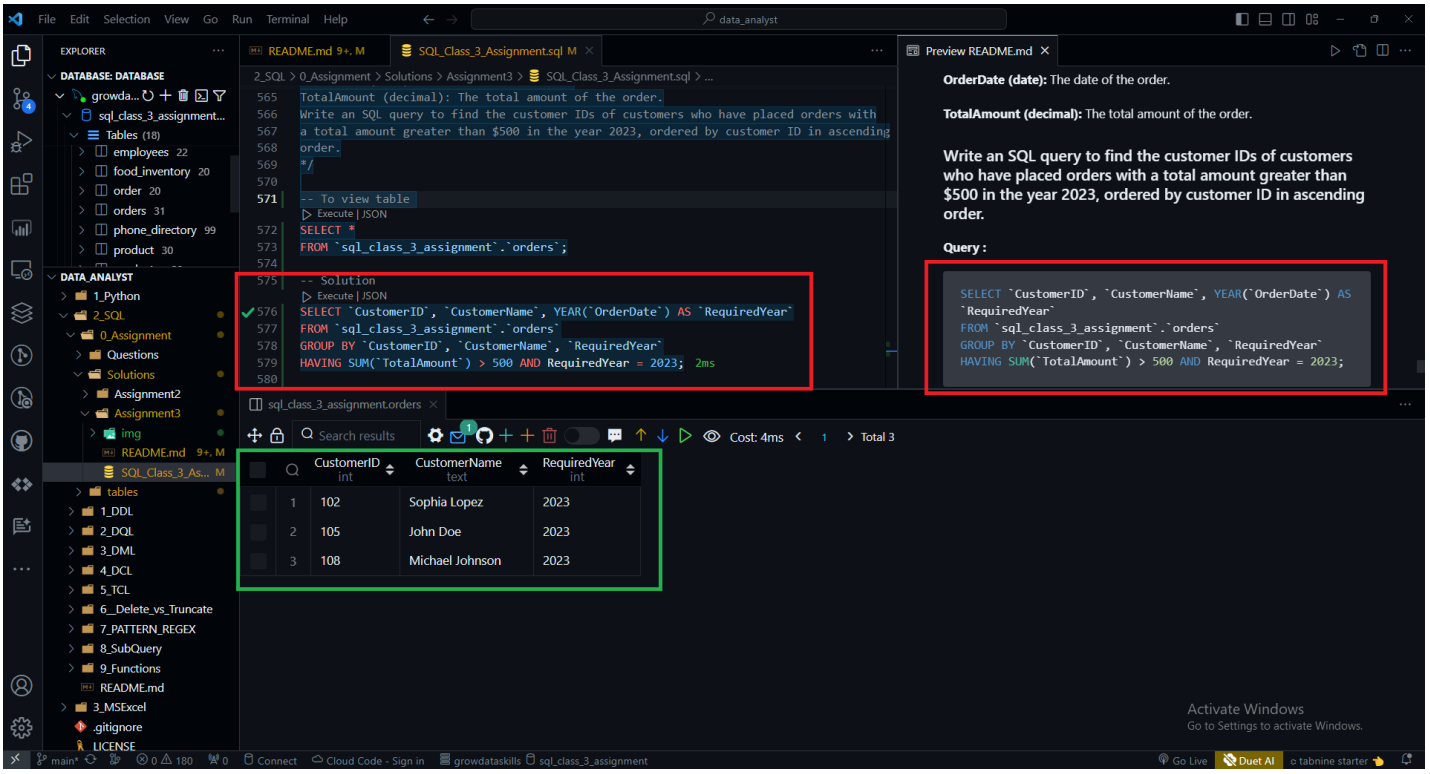
TotalAmount (decimal): The total amount of the order.

Write an SQL query to find the customer IDs of customers who have placed orders with a total amount greater than $500 in the year 2023, ordered by customer ID in ascending order.

Query :

```
SELECT `CustomerID`, `CustomerName`, YEAR(`OrderDate`) AS `RequiredYear`
FROM `sql_class_3_assignment`.`orders`
GROUP BY `CustomerID`, `CustomerName`, `RequiredYear`
HAVING SUM(`TotalAmount`) > 500 AND RequiredYear = 2023;
```

Execution :



Question 14: Consider a table named Students with the following columns:

StudentID (integer): The unique identifier for each student.

Course (string): The course name.

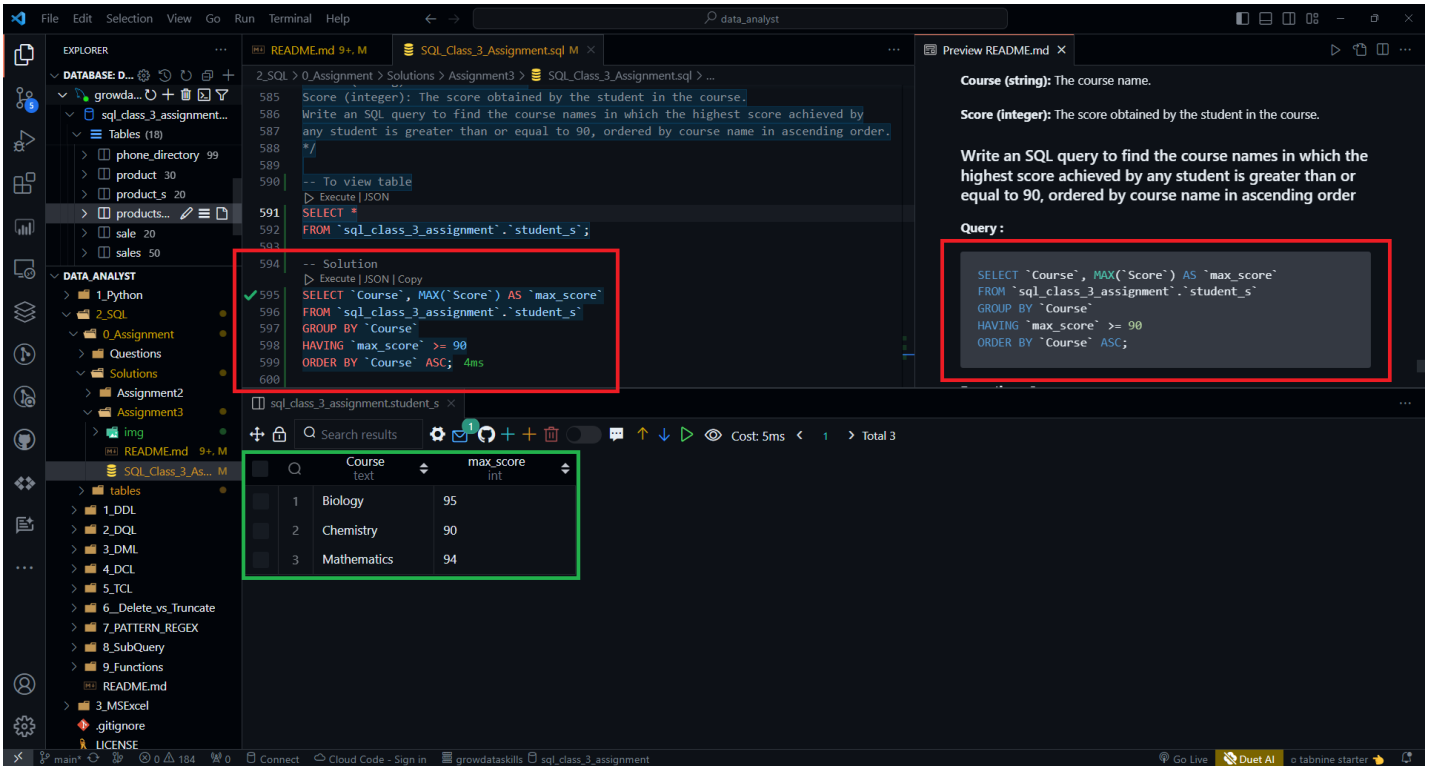
Score (integer): The score obtained by the student in the course.

Write an SQL query to find the course names in which the highest score achieved by any student is greater than or equal to 90, ordered by course name in ascending order

Query :

```
SELECT `Course`, MAX(`Score`) AS `max_score`
FROM `sql_class_3_assignment`.`student_s`
GROUP BY `Course`
HAVING `max_score` >= 90
ORDER BY `Course` ASC;
```

Execution :



Question 15: Consider a table named Orders with the following columns:

OrderID (integer): The unique identifier for each order.

CustomerID (integer): The unique identifier for each customer.

OrderDate (date): The date of the order.

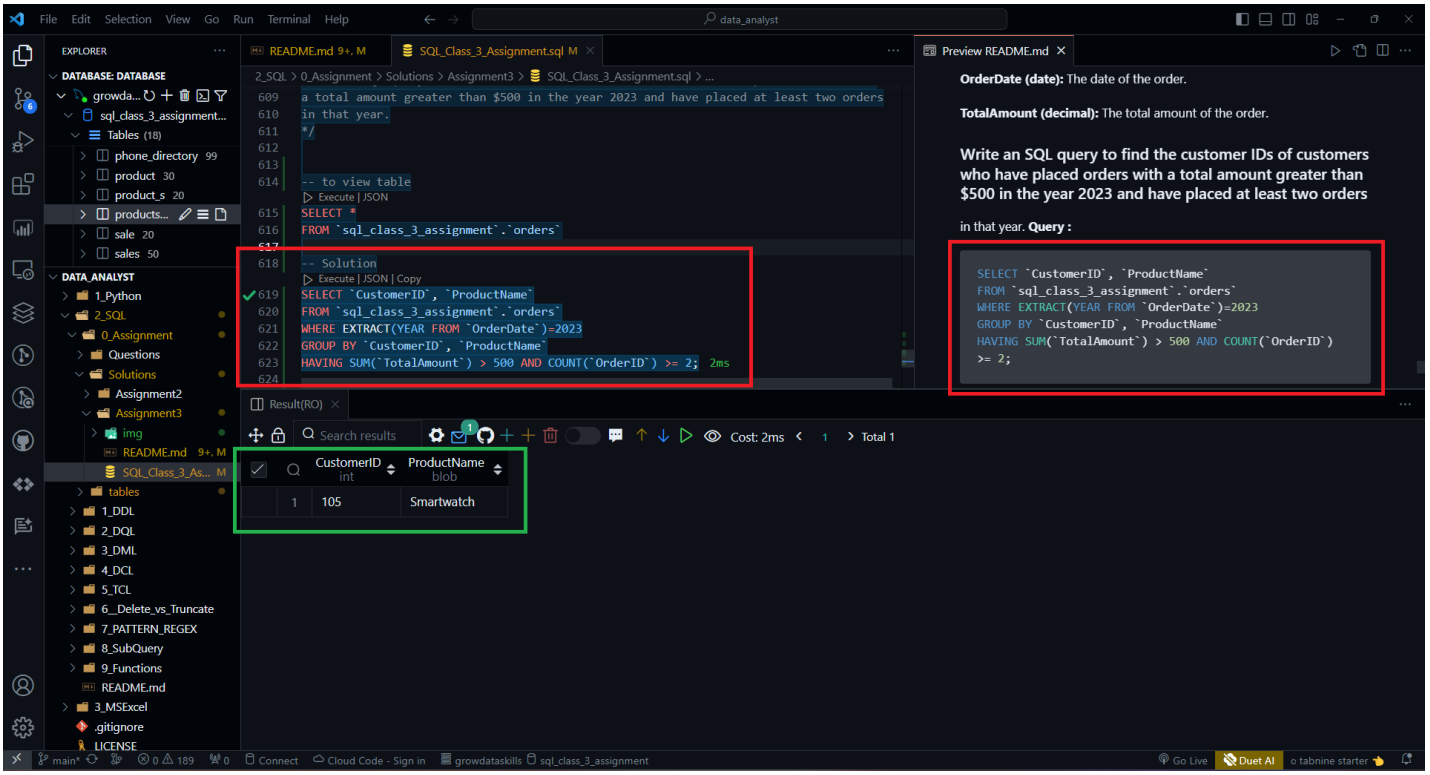
TotalAmount (decimal): The total amount of the order.

Write an SQL query to find the customer IDs of customers who have placed orders with a total amount greater than $500 in the year 2023 and have placed at least two orders

in that year. **Query :**

```
SELECT `CustomerID`, `ProductName`
FROM `sql_class_3_assignment`.`orders`
WHERE EXTRACT(YEAR FROM `OrderDate`)=2023
GROUP BY `CustomerID`, `ProductName`
HAVING SUM(`TotalAmount`) > 500 AND COUNT(`OrderID`) >= 2;
```

Execution :



Question 16: Consider a table named Students with the following columns:

StudentID (integer): The unique identifier for each student.

Course (string): The course name.

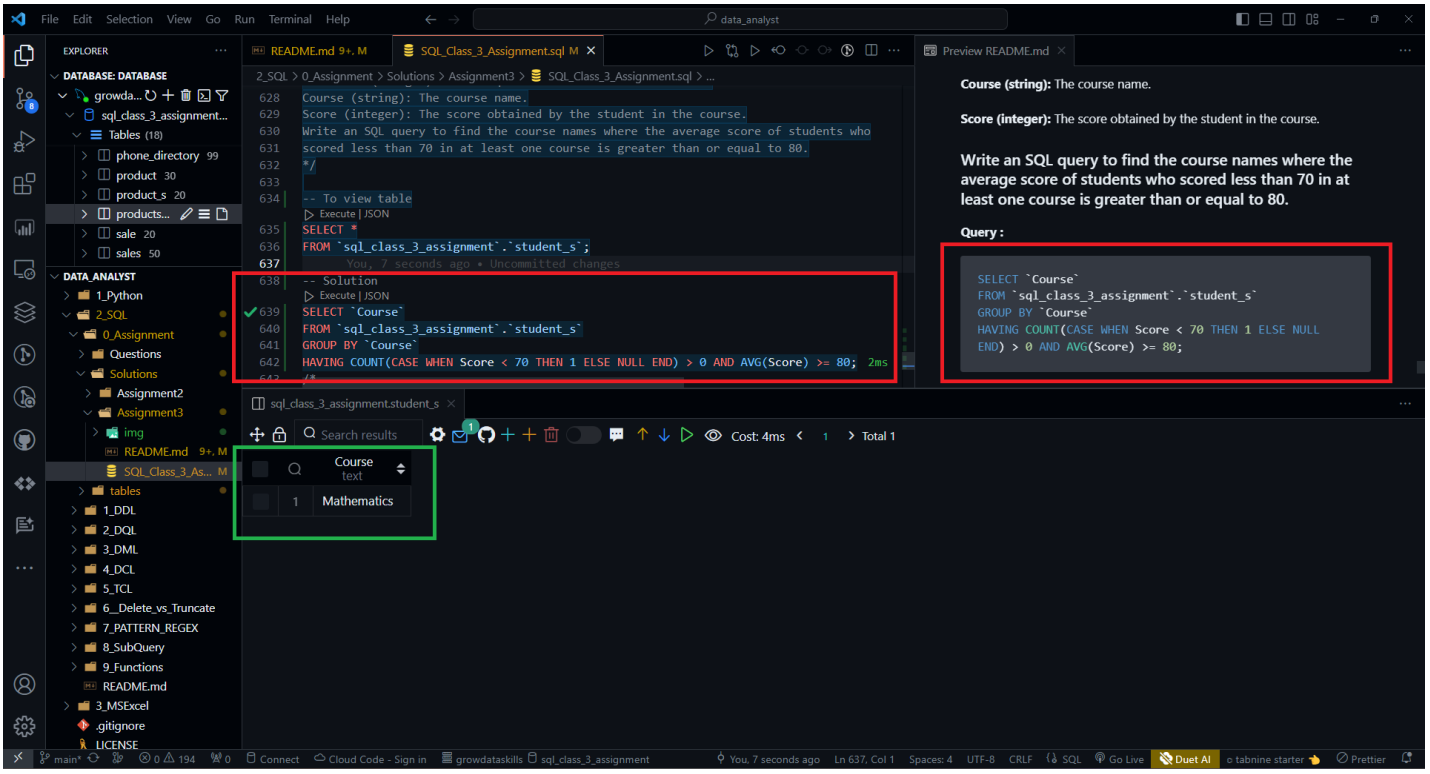
Score (integer): The score obtained by the student in the course.

Write an SQL query to find the course names where the average score of students who scored less than 70 in at least one course is greater than or equal to 80.

Query :

```
SELECT `Course`
FROM `sql_class_3_assignment`.`student_s`
GROUP BY `Course`
HAVING COUNT(CASE WHEN Score < 70 THEN 1 ELSE NULL END) > 0 AND AVG(Score) >= 80;
```

Execution :



Question 17: Consider a table named Employees with the following columns:

EmployeeID (integer): The unique identifier for each employee.

Department (string): The department in which the employee works.

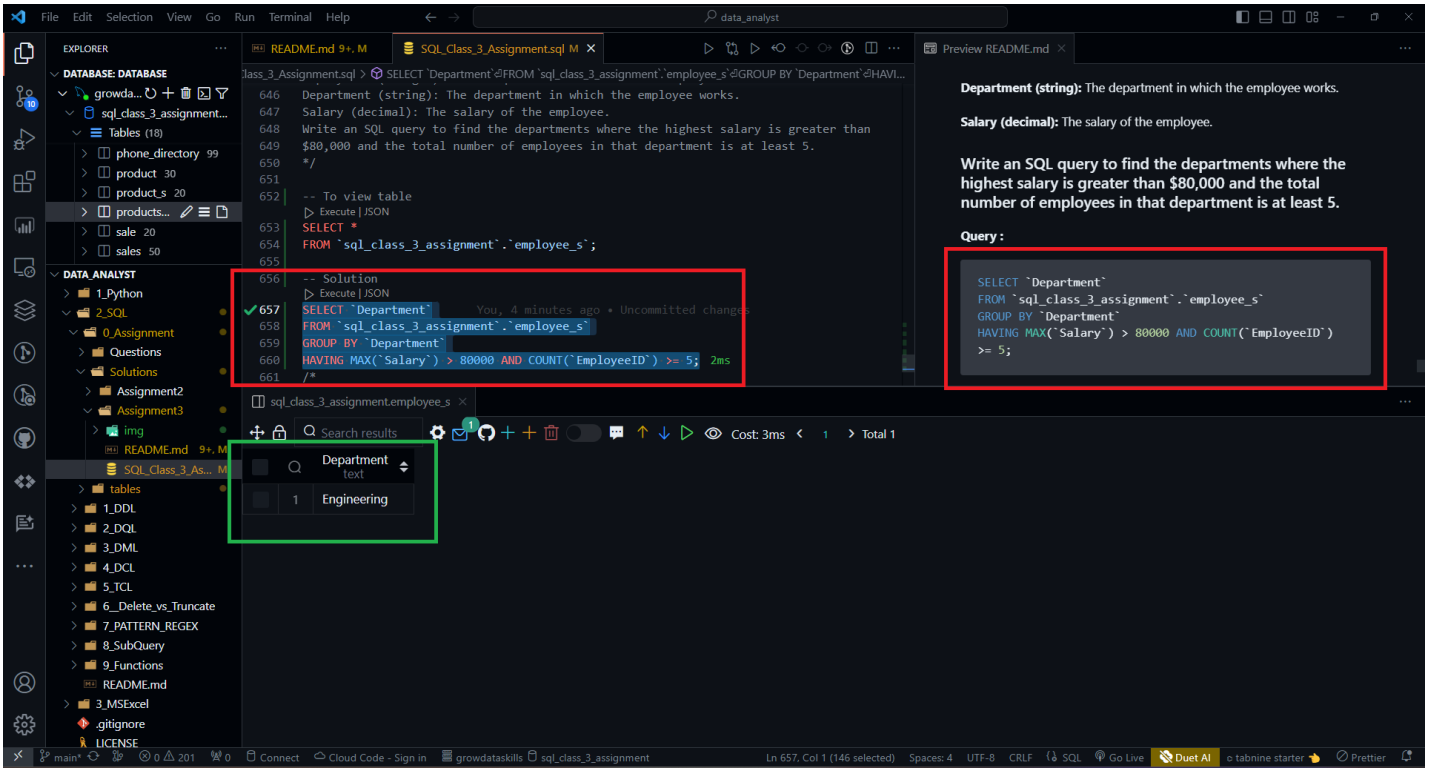
Salary (decimal): The salary of the employee.

Write an SQL query to find the departments where the highest salary is greater than $80,000 and the total number of employees in that department is at least 5.

Query :

```
SELECT `Department`
FROM `sql_class_3_assignment`.`employee_s`
GROUP BY `Department`
HAVING MAX(`Salary`) > 80000 AND COUNT(`EmployeeID`) >= 5;
```

Execution :



Question 18: Consider a table named Products with the following columns:

ProductID (integer): The unique identifier for each product.

Category (string): The category of the product.

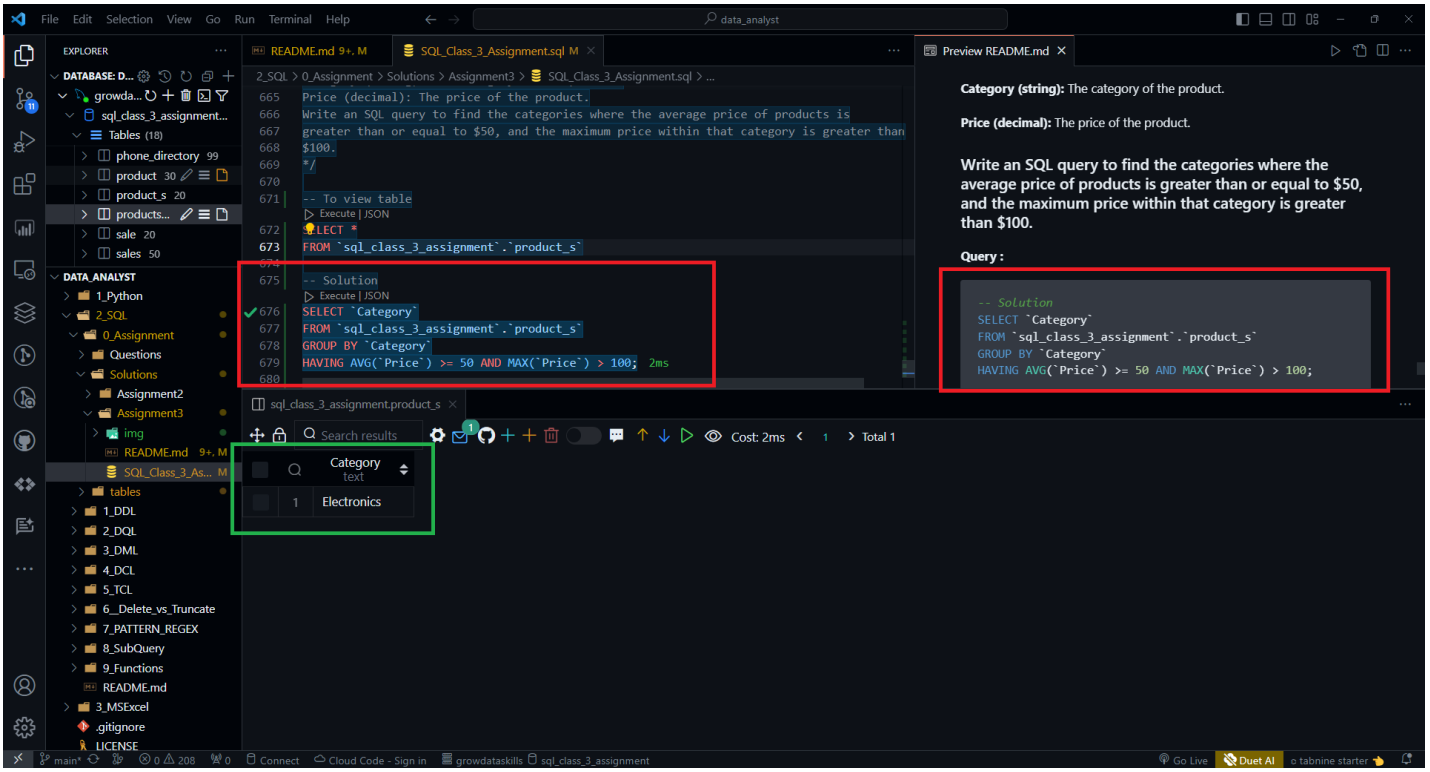
Price (decimal): The price of the product.

Write an SQL query to find the categories where the average price of products is greater than or equal to $50, and the maximum price within that category is greater than $100.

Query :

```
-- Solution
SELECT `Category`
FROM `sql_class_3_assignment`.`product_s`
GROUP BY `Category`
HAVING AVG(`Price`) >= 50 AND MAX(`Price`) > 100;
```

Execution :



Question 19: Consider a table named Orders with the following columns:

OrderID (integer): The unique identifier for each order.

CustomerID (integer): The unique identifier for each customer.

OrderDate (date): The date of the order.

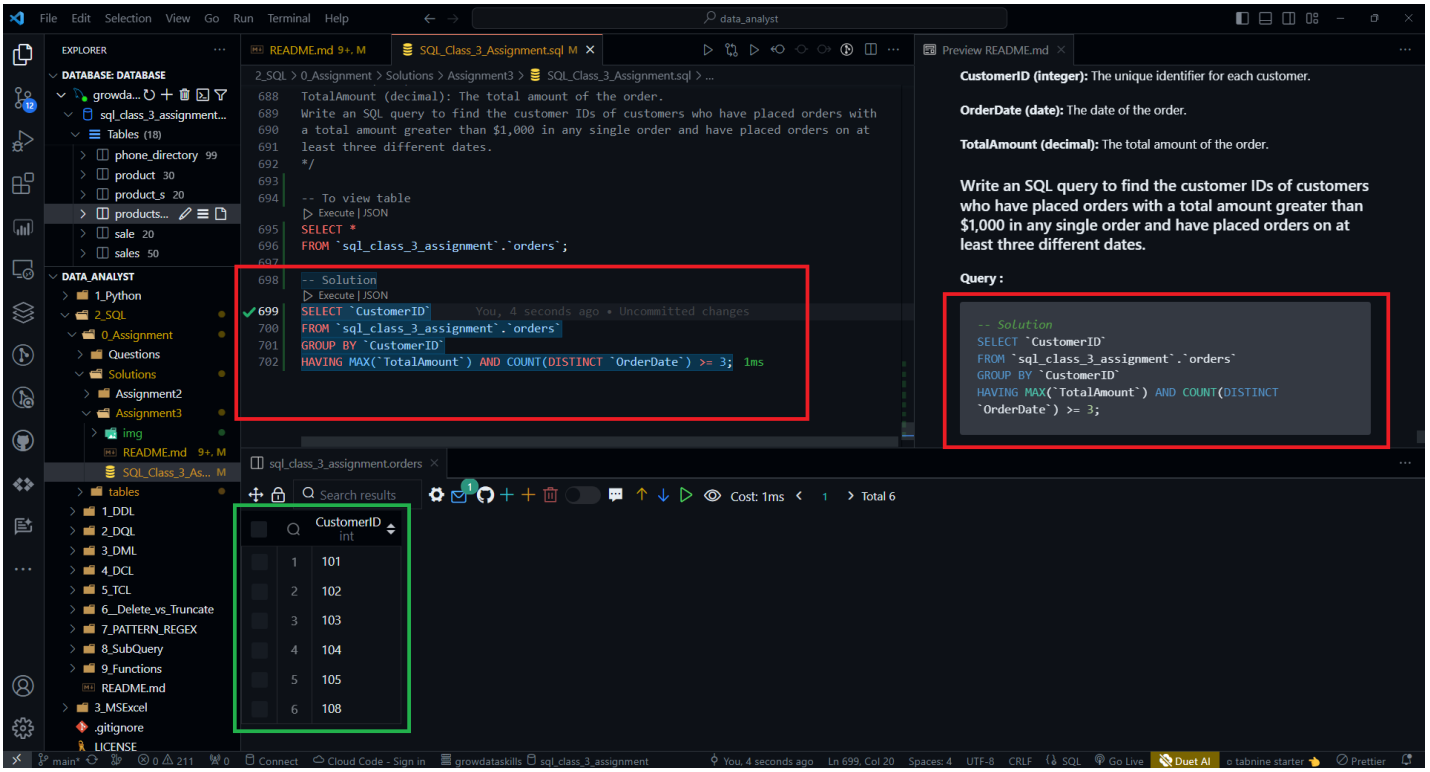
TotalAmount (decimal): The total amount of the order.

Write an SQL query to find the customer IDs of customers who have placed orders with a total amount greater than $1,000 in any single order and have placed orders on at least three different dates.

Query :

```
-- Solution
SELECT `CustomerID`
FROM `sql_class_3_assignment`.`orders`
GROUP BY `CustomerID`
HAVING MAX(`TotalAmount`) AND COUNT(DISTINCT `OrderDate`) >= 3;
```

Execution :



Submitted By: Arpit Dubey

Email ID: aarpitdubey@gmail.com

LinkedIn: click here

Github : click here
