MS SQL SERVER PROJECT

Indian\_summer\_products

SQL Problems & Solutions

**Description:** This project focuses on analysing **Indian summer skincare and personal care products** using MS SQL.

The objective of this project is to **gain insights into market trends, consumer preferences, and product performance** by leveraging SQL queries for structured data analysis. Key analyses include:

* Identifying **popular products** based on ratings and usage frequency.
* Analyzing the impact of **SPF ratings, organic ingredients, and paraben-free claims** on product popularity.
* Understanding **regional usage patterns** and online availability trends.
* Categorizing products based on **age groups, gender, and skin type suitability**.

**Data:** [Dataset](https://drive.google.com/file/d/1GcoFu84u885XQX8h9RXu1RhK3eG8JE_n/view?usp=sharing)

**Tool:** SQL Server Management Studio

**Analysis Technique:** MS SQL Queries

-- Creating schema and table

**CREATE DATABASE IF NOT EXISTS** ecommerce;

**USE** ecommerce;

**CREATE TABLE** Indian\_summer\_products(

Product\_id money NOT NULL PRIMARY KEY,

Product\_name nvarchar(50) NOT NULL,

Category nvarchar(50) NOT NULL,

Brand nvarchar(50) NOT NULL,

Average\_price FLOAT NOT NULL,

SPF\_rating tinyint NOT NULL,

Skin\_type nvarchar(50) NOT NULL,

Age\_group nvarchar(50) NOT NULL,

Gender nvarchar(50) NOT NULL,

Popularity\_rating tinyint NOT NULL,

Usage\_frequent nvarchar(50) NOT NULL,

Primary\_Ingredient nvarchar(50) NOT NULL,

Organic bit NOT NULL,

Paraben\_free bit NOT NULL,

Region\_usage nvarchar(50) NOT NULL,

Online\_availability nvarchar(50) NOT NULL

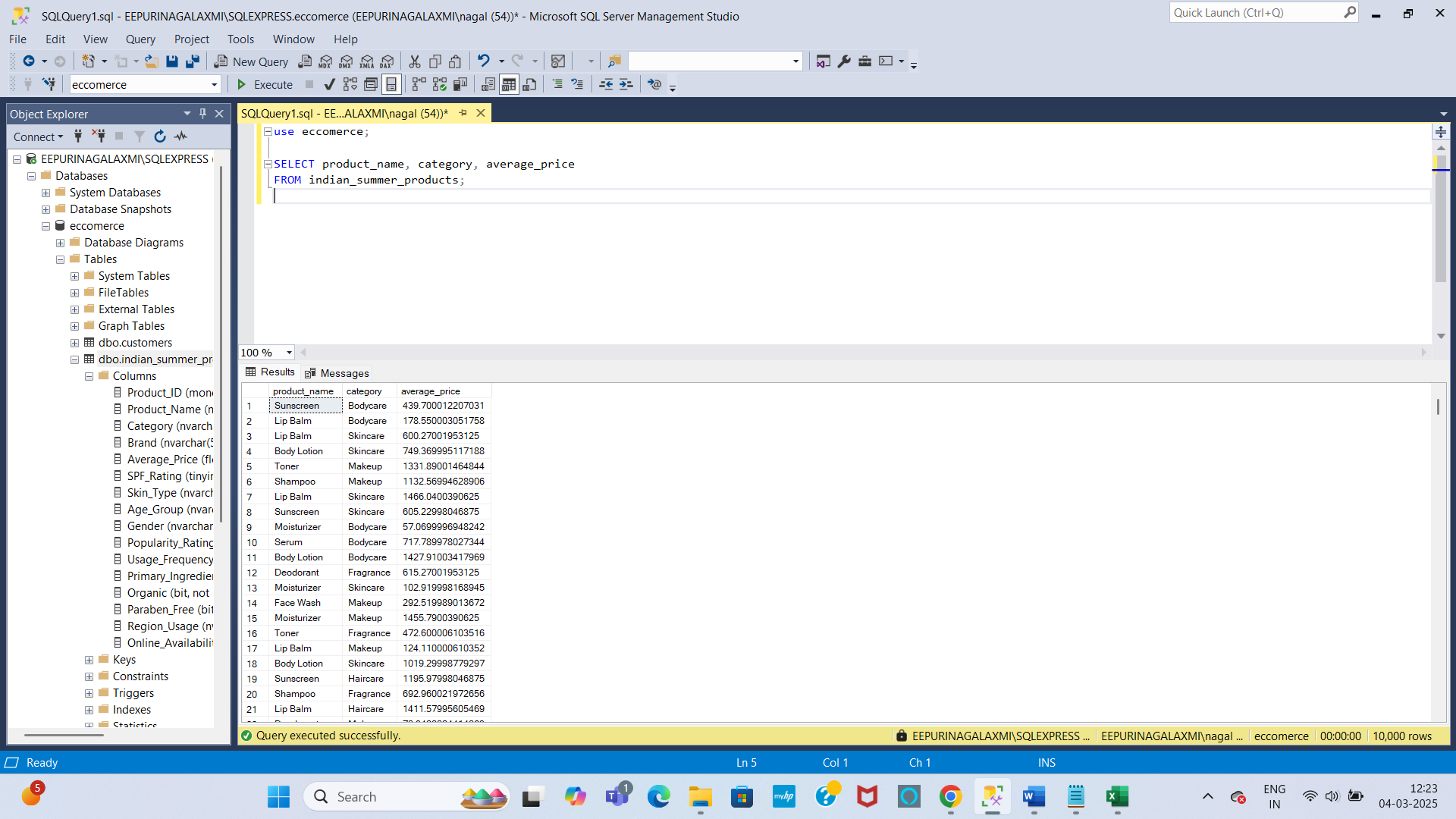
);

**1. Select only the Product\_Name, Category, and Average\_Price of products.**

SELECT product\_name, category, average\_price

FROM indian\_summer\_products;

**Output:**

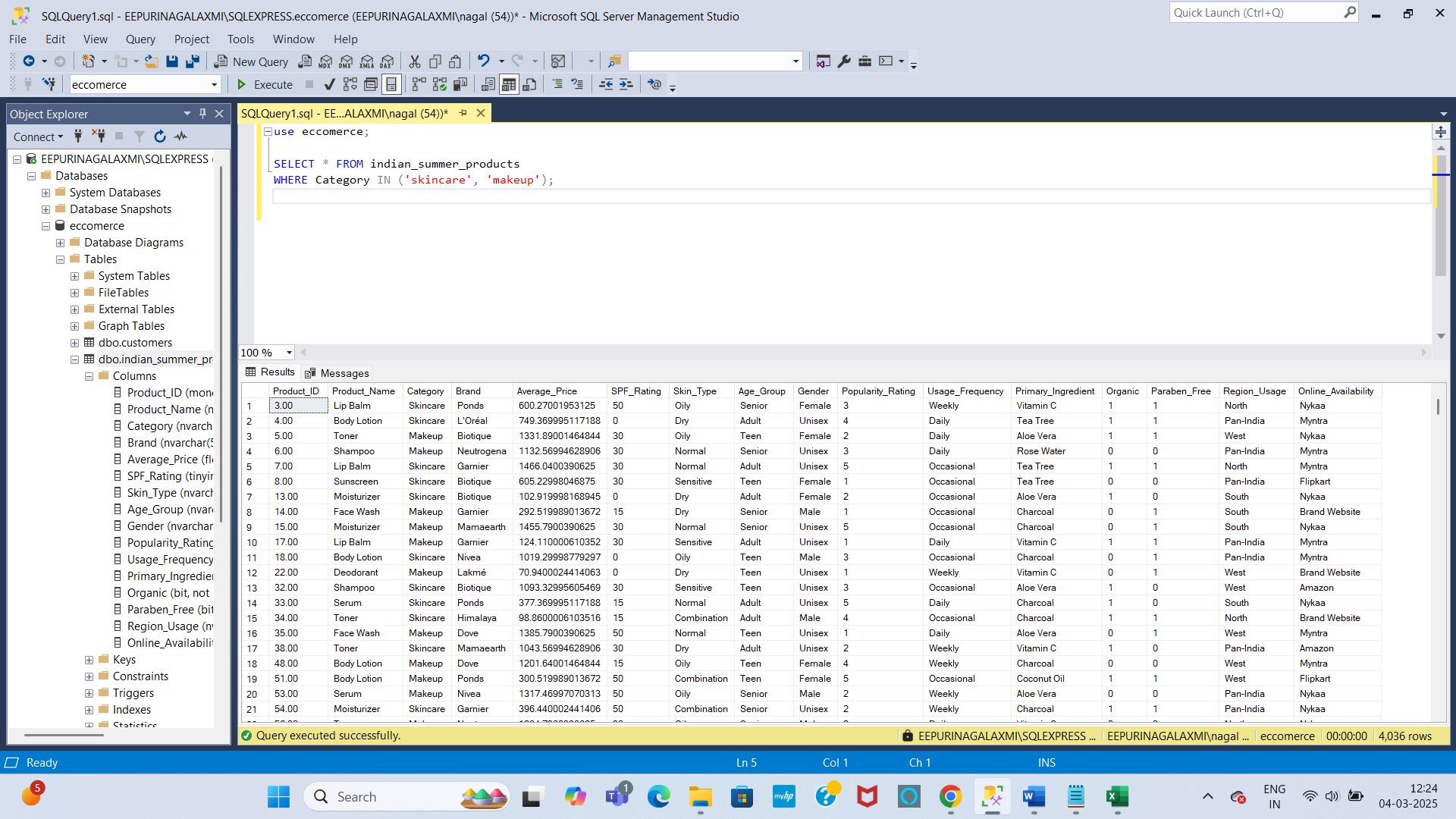


**2. Find all products that belong to the "Skincare" or “Makeup” category.**

SELECT \* FROM indian\_summer\_products

WHERE Category IN ('skincare', 'makeup');

**Output:**

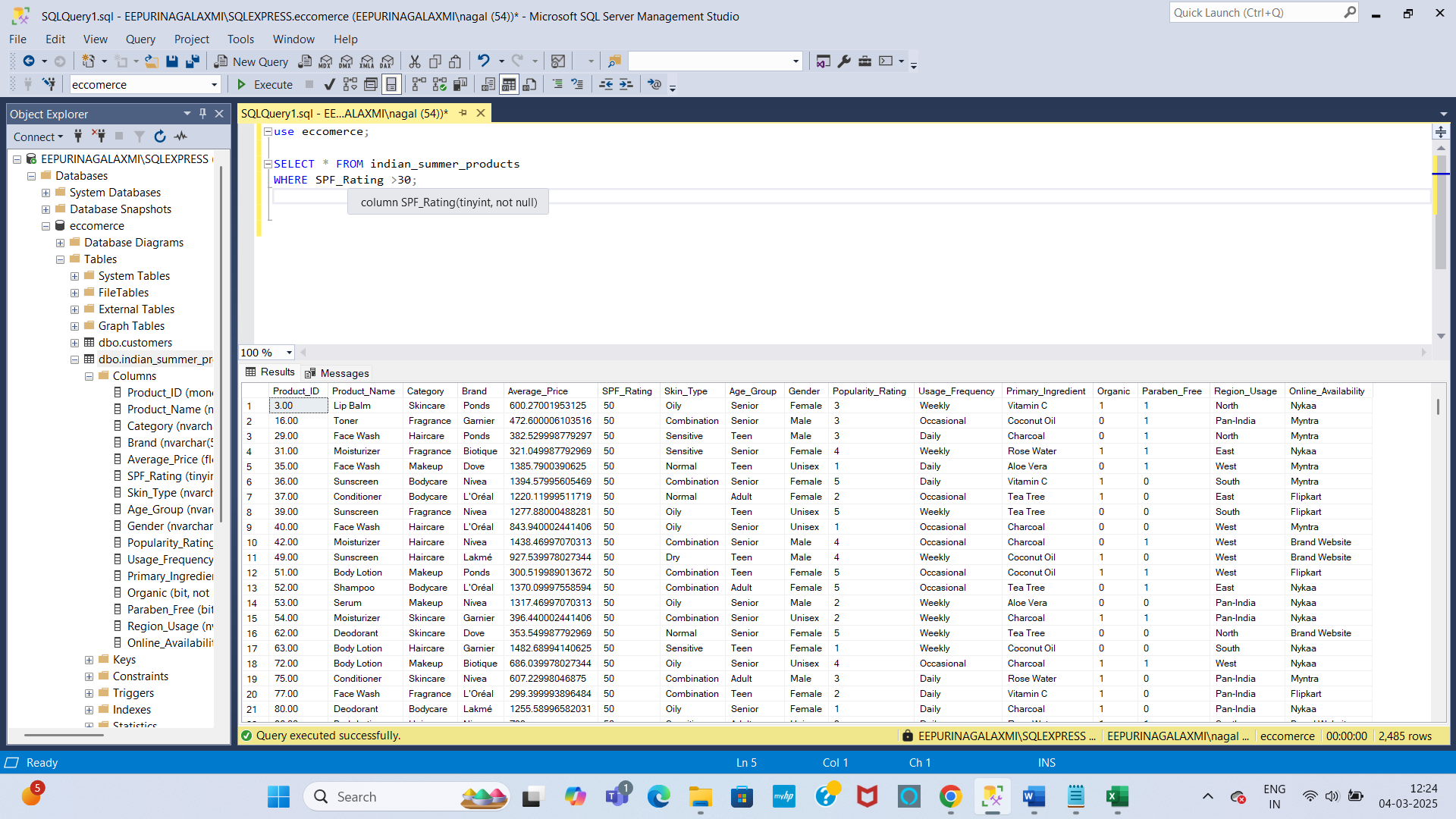


**3. Retrieve all products with an SPF\_Rating greater than 30.**

SELECT \* FROM indian\_summer\_products

WHERE SPF\_Rating >30;

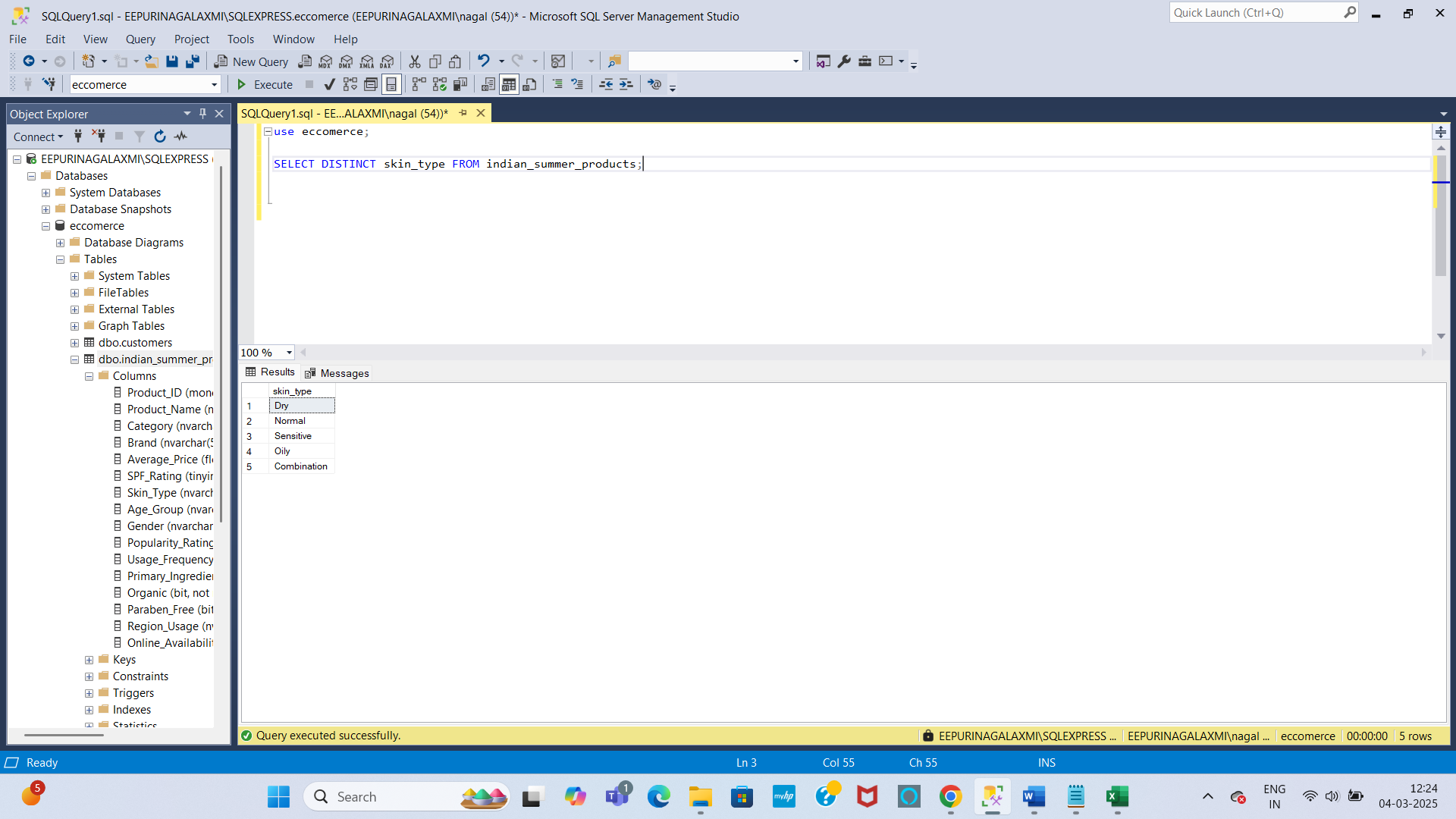
**Output:**

s

**4. List distinct Skin\_Types available in the table.**

SELECT DISTINCT skin\_type FROM indian\_summer\_products;

**Output:**



**5. Retrieve all Organic products sorted by Popularity\_Rating in descending order.**

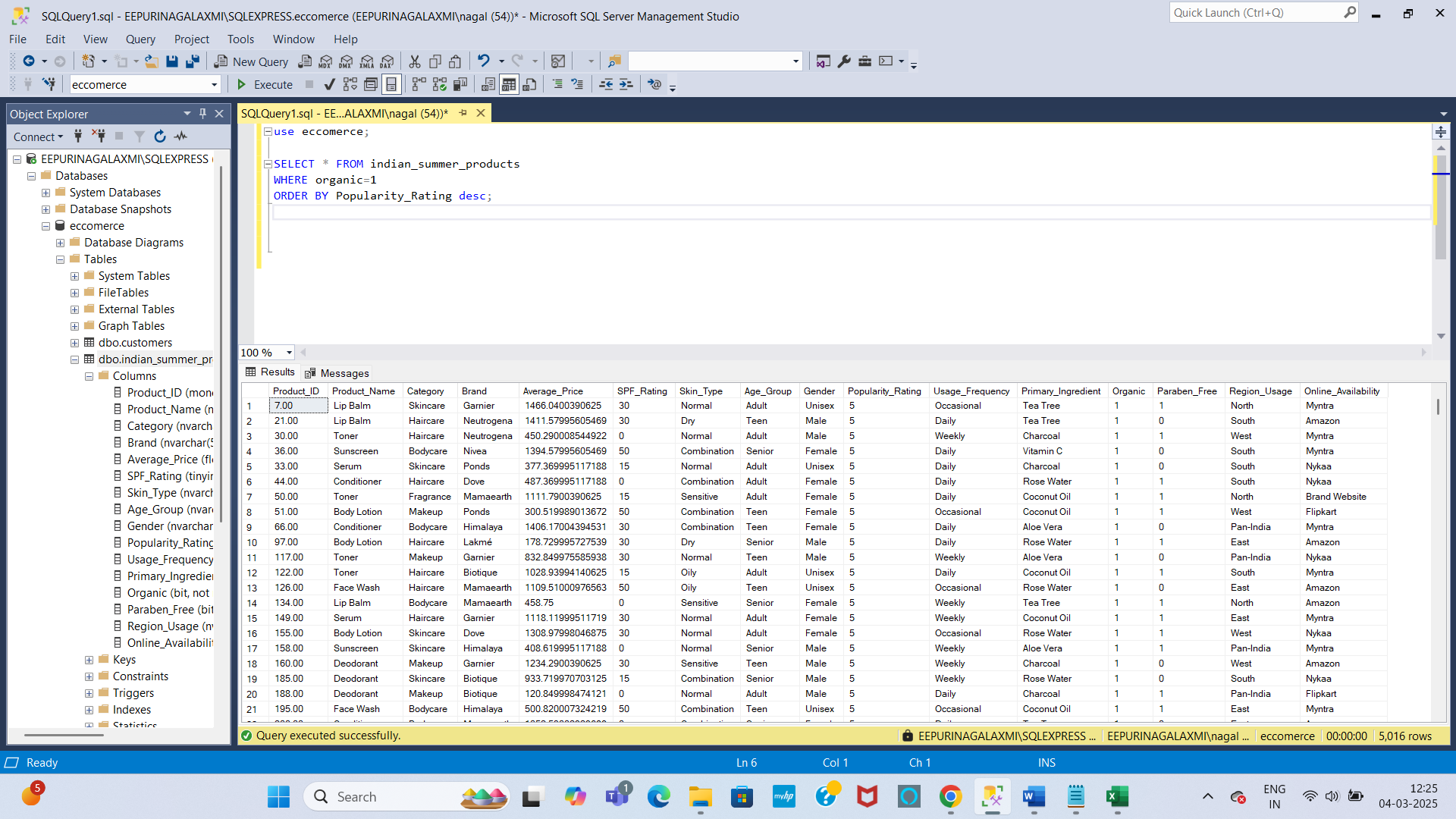
True = 1, False =0

SELECT \* FROM indian\_summer\_products

WHERE organic=1

ORDER BY Popularity\_Rating desc;

**Output:**

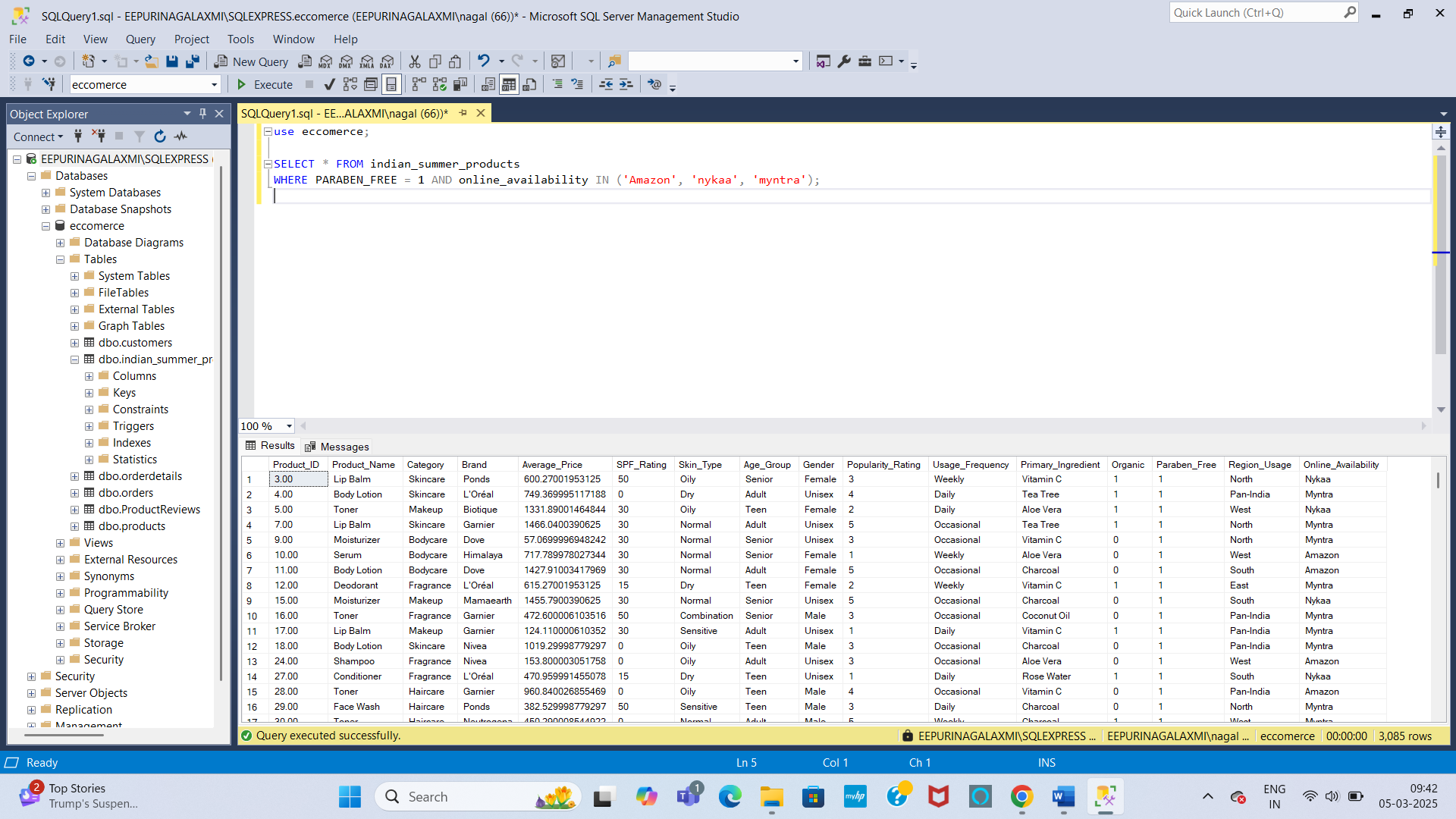


**6. Get all Paraben\_Free products that are available on any online platform (Amazon, Nykaa, Myntra).**

SELECT \* FROM indian\_summer\_products

WHERE PARABEN\_FREE = 1 AND online\_availability IN ('Amazon', 'nykaa', 'myntra’);

**Output:**

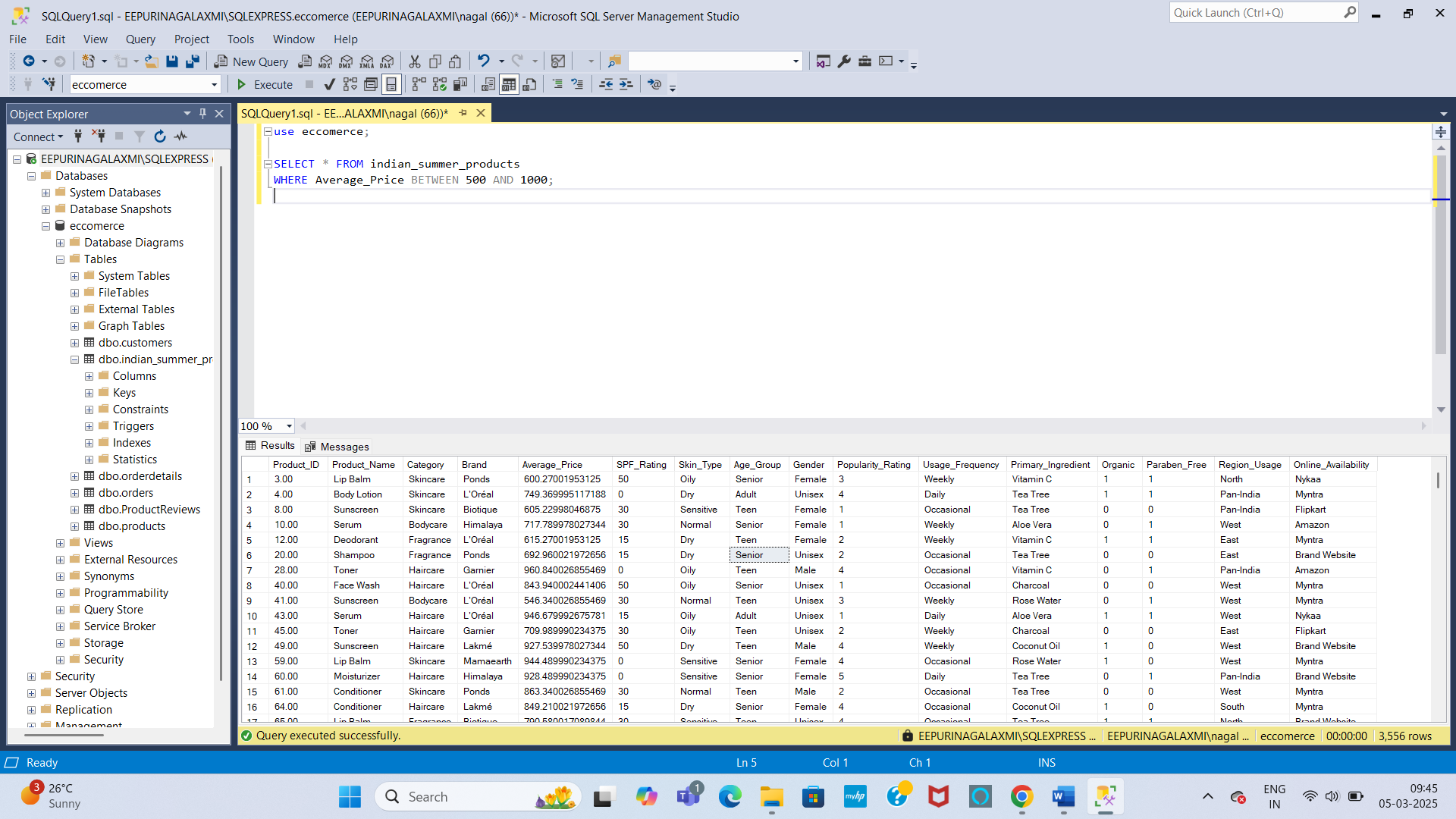


**7. Find all products that have an Average\_Price between 500 and 1000.**

SELECT \* FROM indian\_summer\_products

WHERE Average\_Price BETWEEN 500 AND 1000;

**Output:**



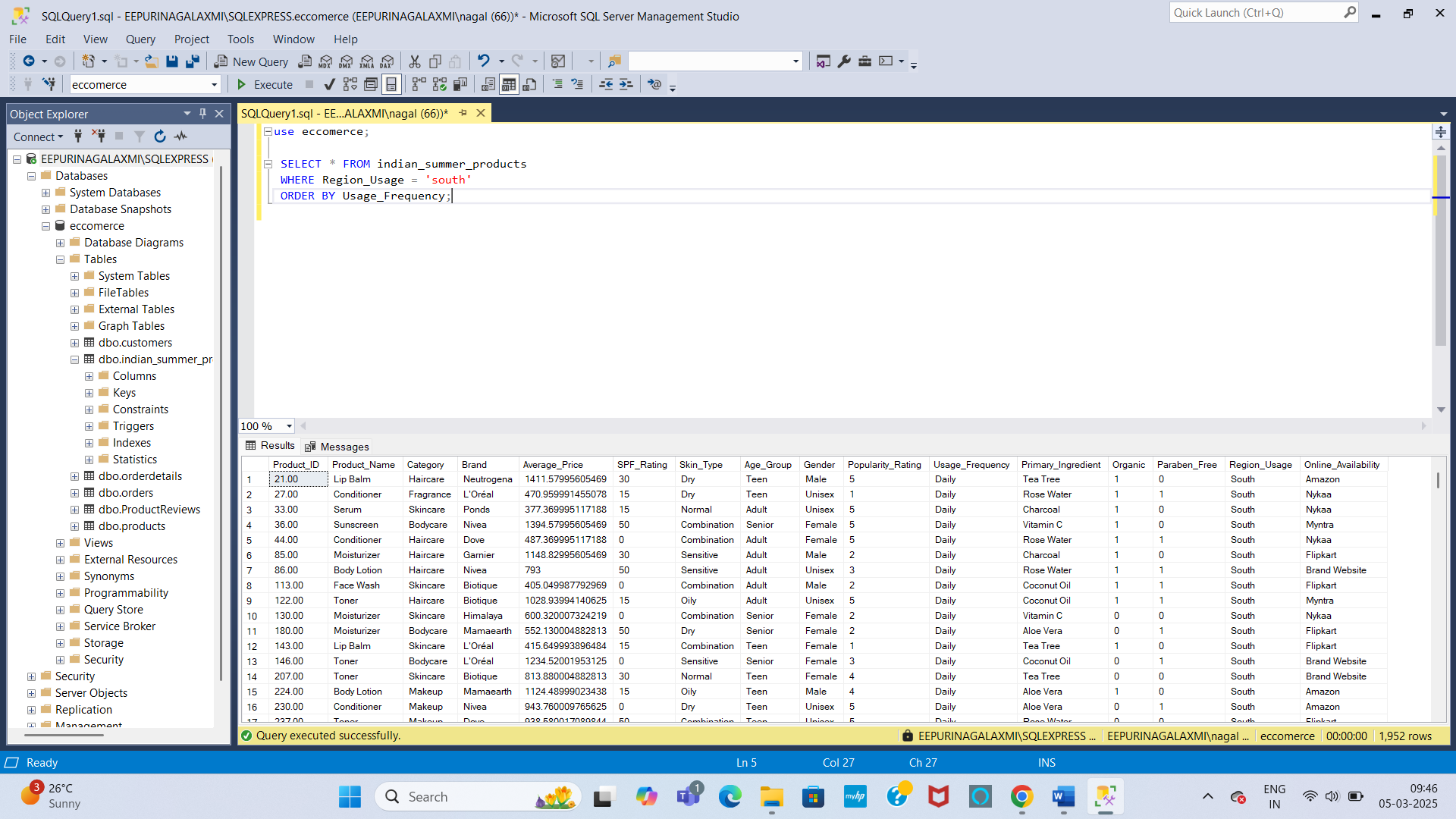
**8. Display products used in the "South" region and order them by Usage\_Frequency.**

SELECT \* FROM indian\_summer\_products

WHERE Region\_usage = 'south'

ORDER BY usage\_frequency;

**Output:**



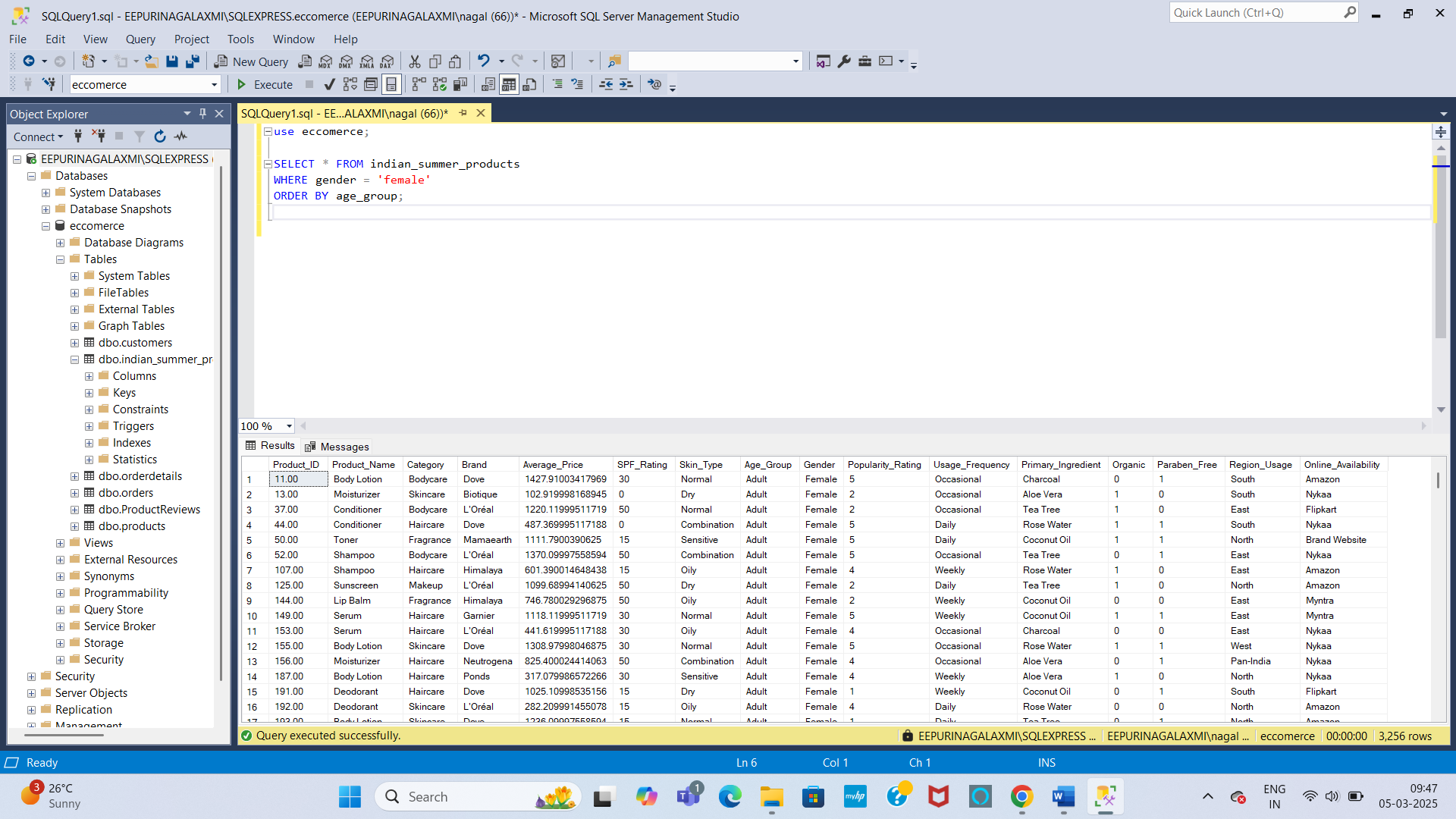
**9. Retrieve products used by females and sorted by Age\_Group.**

SELECT \* FROM indian\_summer\_products

WHERE gender = 'female'

ORDER BY age\_group;

**Output:**



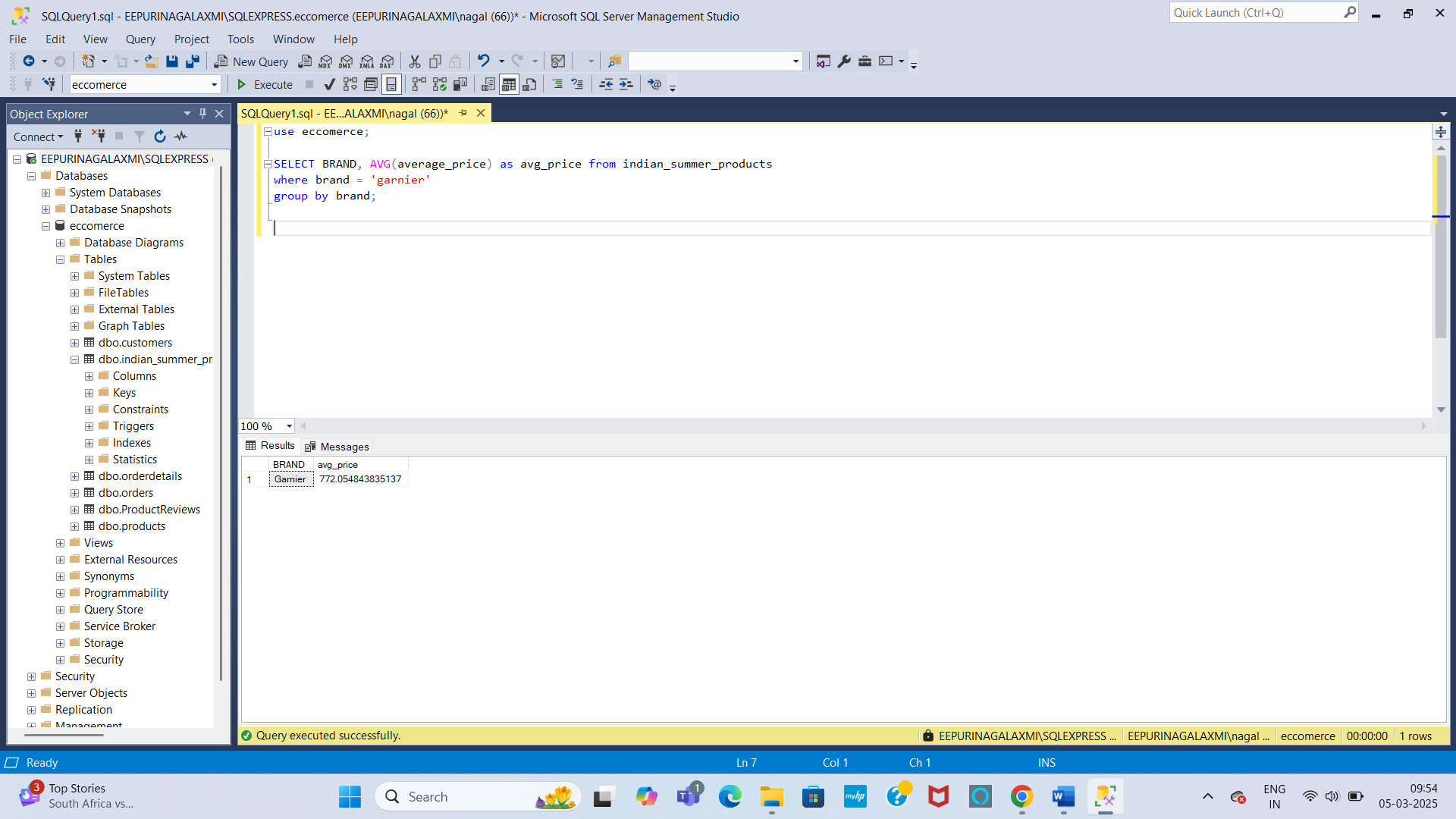
**10. Find the average price of all Garnier products.**

SELECT brand, AVG(average\_price) AS avg\_price FROM indian\_summer\_products

WHERE brand = 'garnier'

GROUP BY brand;

**Output:**



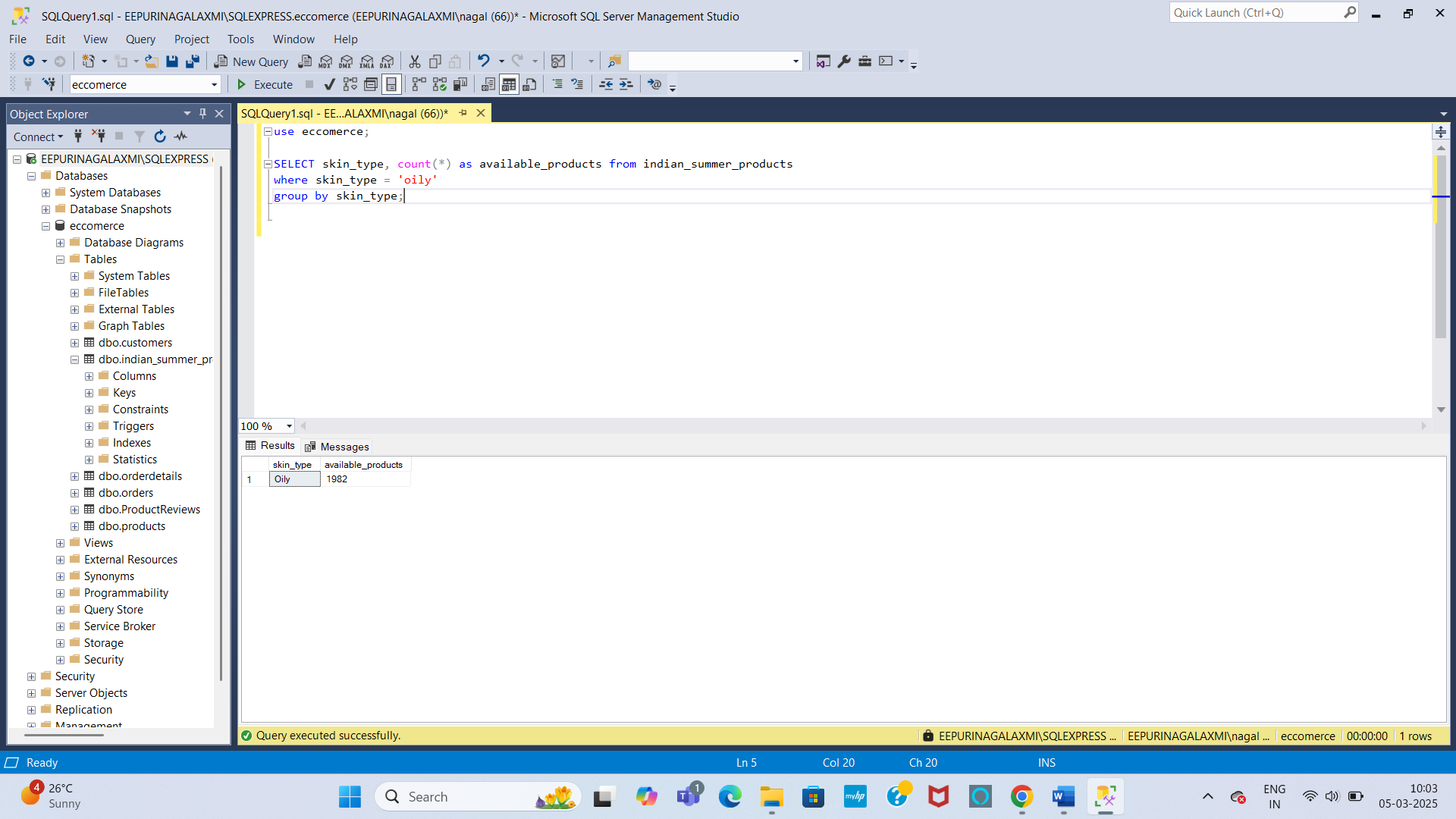
**11. Count the number of products available for "Oily" skin.**

SELECT skin\_type, COUNT(\*) AS available\_products FROM indian\_summer\_products

WHERE skin\_type = 'Oily'

GROUP BY skin\_type;

**Output:**



**12. Get the highest Popularity\_Rating for each Brand.**

SELECT Brand, MAX(popularity\_rating) AS highest\_popularity\_rating

FROM indian\_summer\_products

GROUP BY brand;

**Output:**

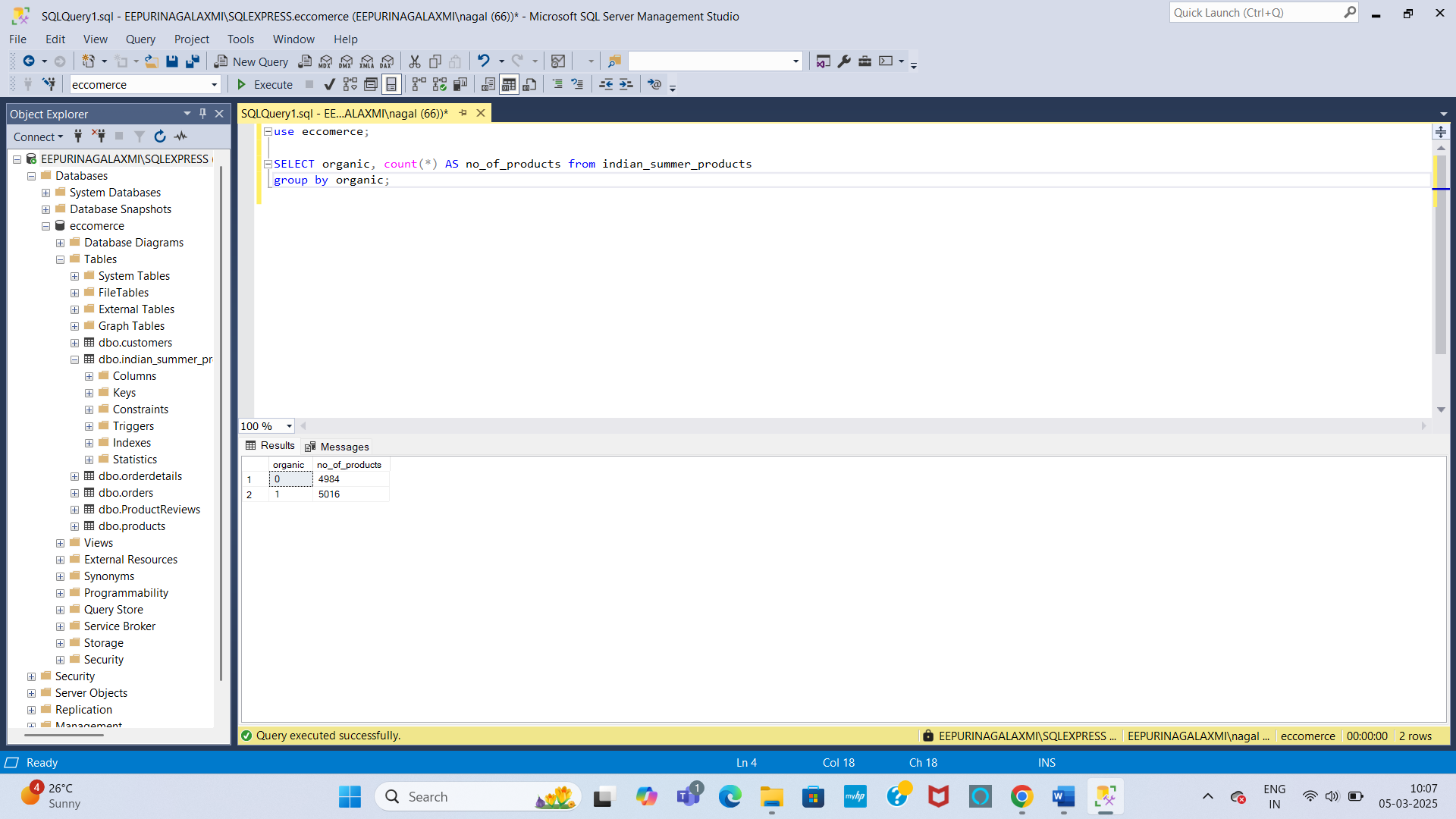


**13. Find out how many products are Organic and how many are Non-Organic.**

SELECT Organic, COUNT(\*) AS no\_of\_products FROM indian\_summer\_products

GROUP BY organic;

**Output:**

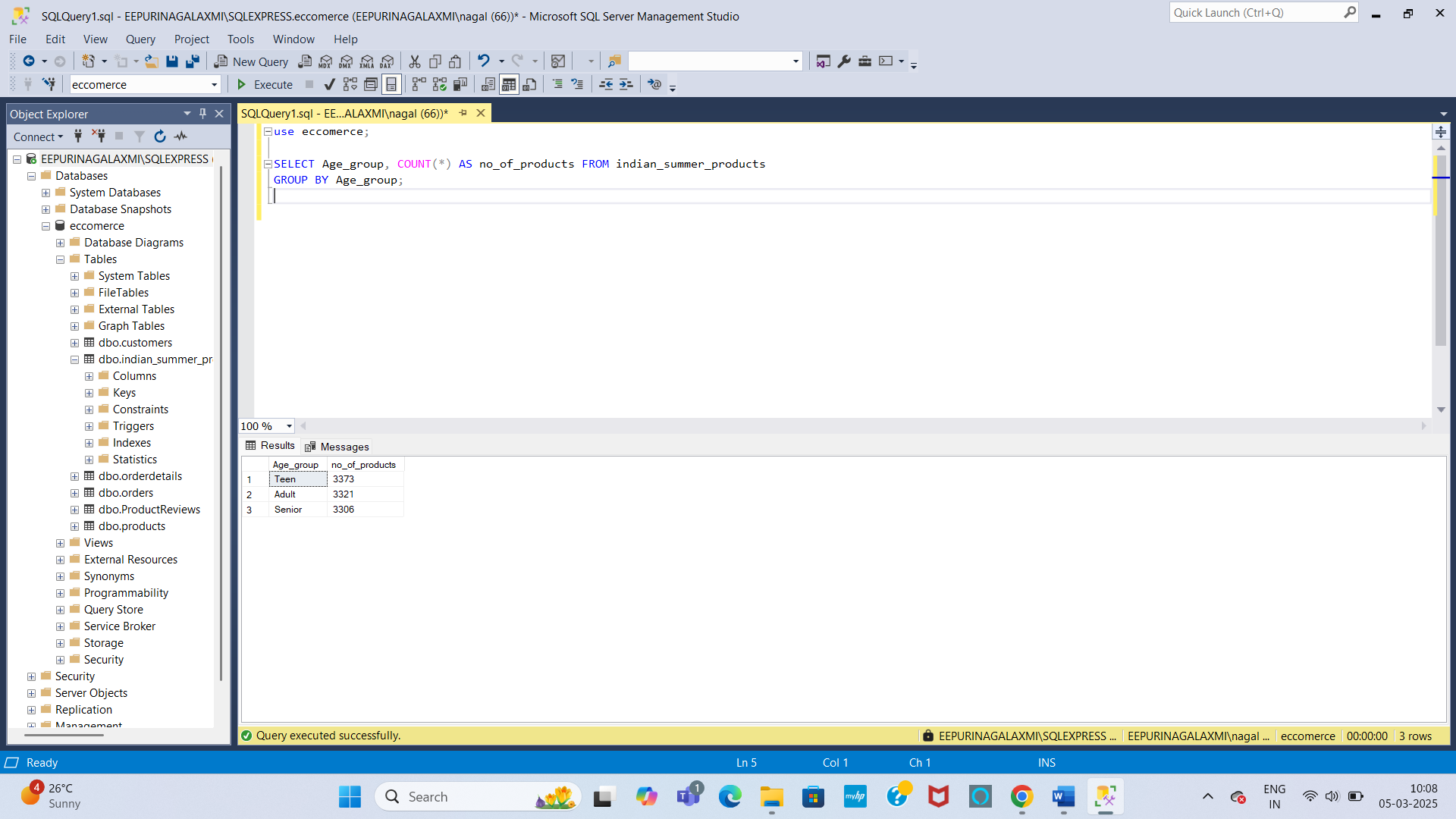


**14. Display the total number of products in each Age\_Group.**

SELECT Age\_group, COUNT(\*) AS no\_of\_products FROM indian\_summer\_products

GROUP BY Age\_group;

**Output:**



**15. Join the SummerProducts table with ProductReviews and get products with an average Review\_Score above 4.5.**

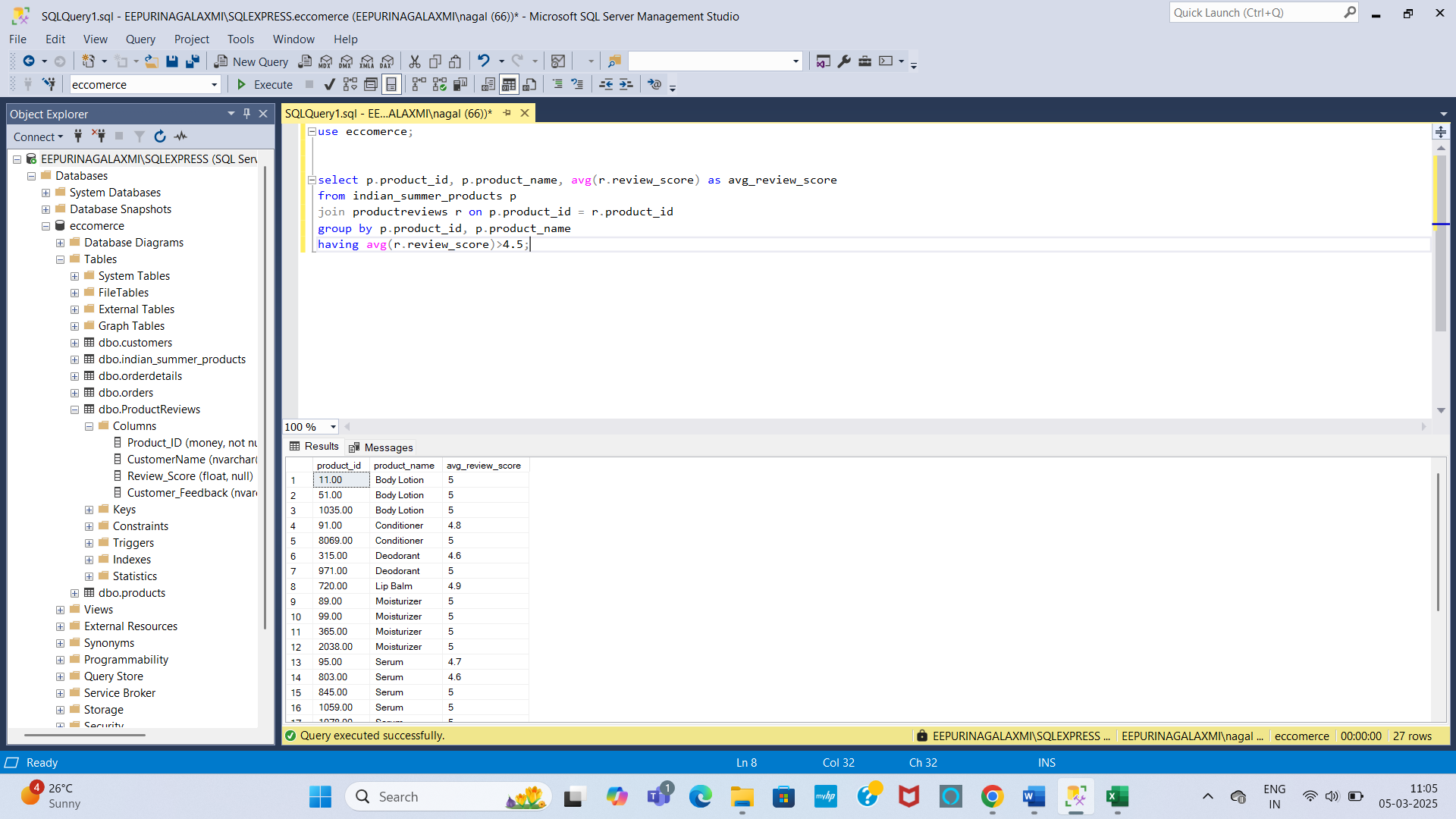
SELECT p.product\_id, p.product\_name, AVG(r.review\_score) AS avg\_review\_score FROM indian\_summer\_products p

JOIN ProductReviews r ON p.product\_id = r.product\_id

GROUP BY p.Product\_ID, p.product\_name

HAVING AVG(r.review\_score) > 4.5;

**Output:**



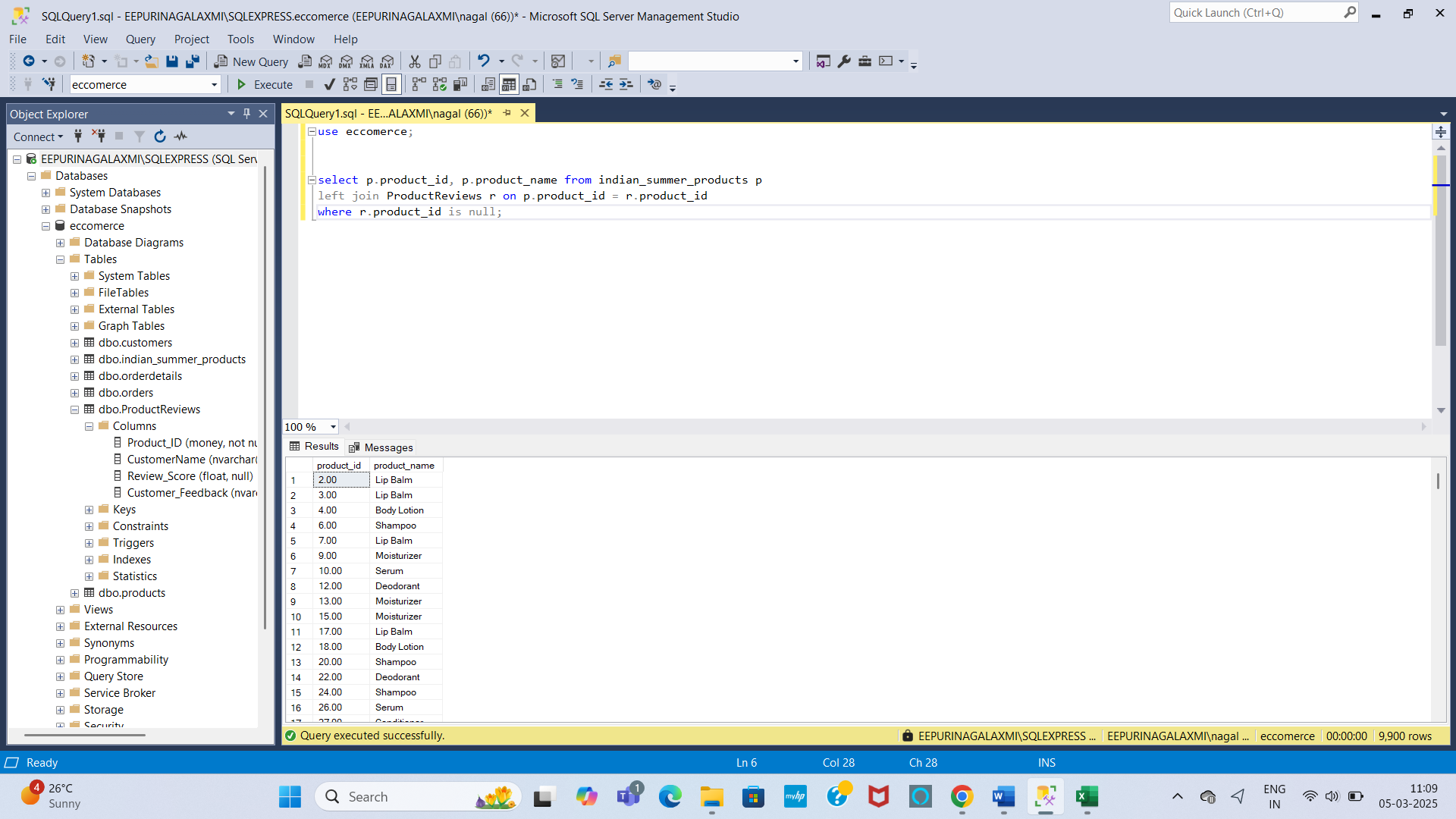
**16. Find all products that have never been reviewed.**

SELECT p.Product\_ID, p.product\_name FROM indian\_summer\_products p

LEFT JOIN productreviews r ON p.product\_id = r.product\_id

WHERE r.Product\_ID IS NULL;

**Output:**

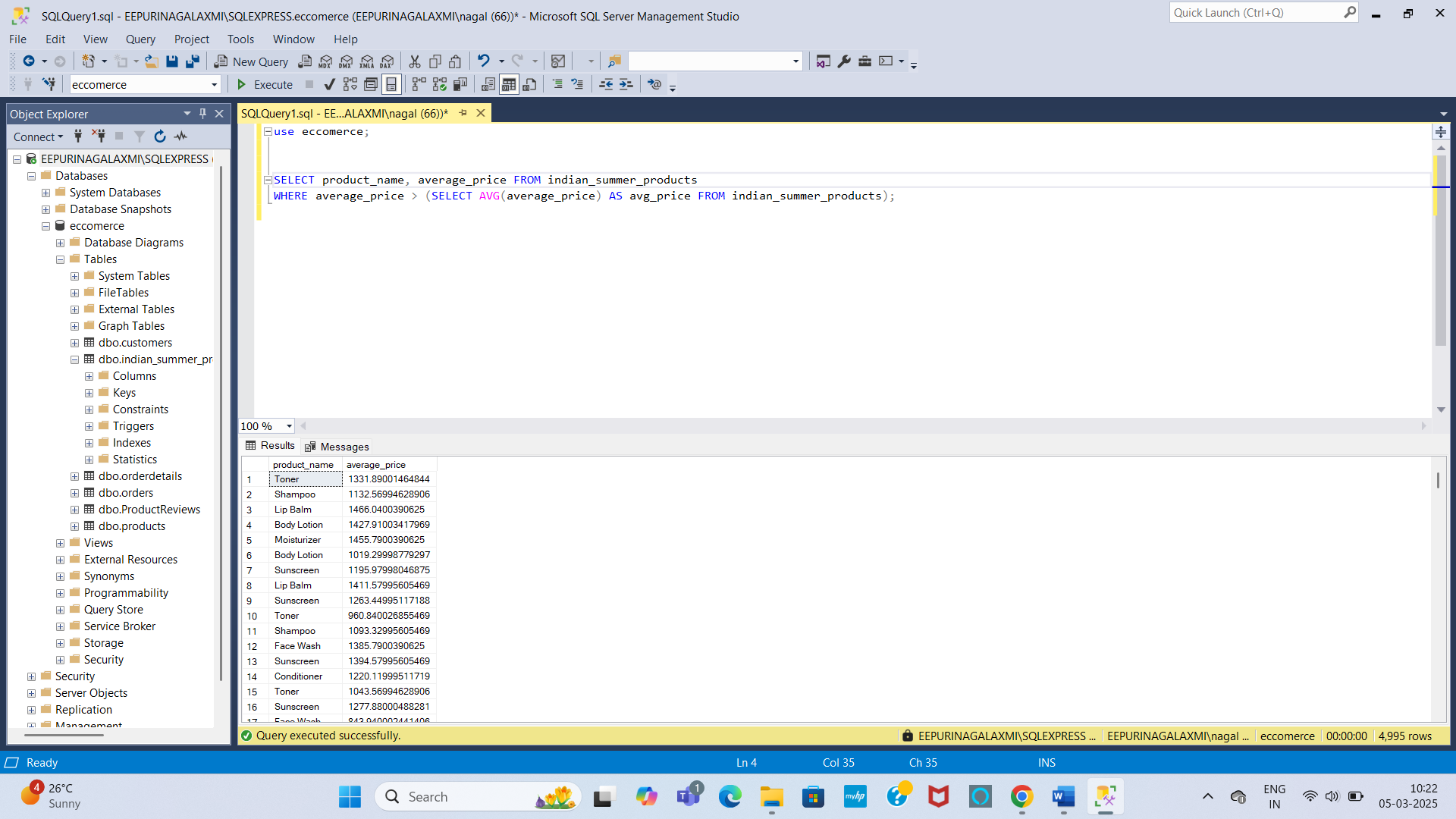


**17. Retrieve all products whose Average\_Price is higher than the average price of all products.**

SELECT \* FROM indian\_summer\_products

WHERE average\_price > (SELECT AVG(average\_price) AS avg\_price FROM indian\_summer\_products);

**Output:**



**18. Find the product with the highest SPF\_Rating.**

SELECT TOP 1 product\_id, product\_name, SPF\_rating FROM indian\_summer\_products

ORDER BY SPF\_rating DESC;

**Output:**



**19. Get products that have the second highest Popularity\_Rating.**

SELECT MAX(popularity\_rating) second\_highest\_popularity\_rating FROM indian\_summer\_products

WHERE popularity\_rating < (SELECT MAX(popularity\_rating) FROM indian\_summer\_products);

**Output:**

