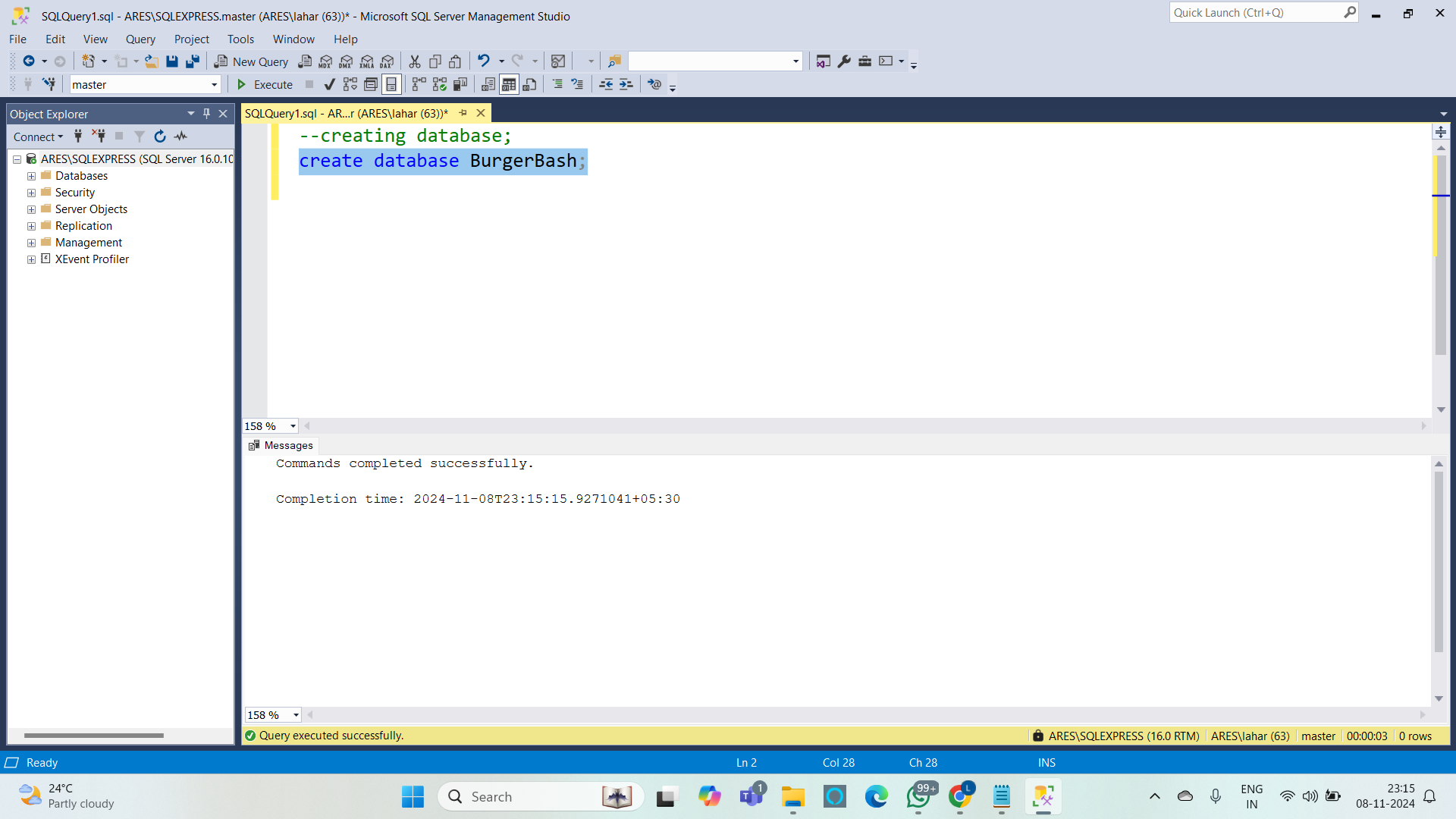
**Case Study - Burger Bash**

**Name : Podutur Lahari - DE126**

**Date : 08-11-2024**

**Create Burger Bash Database :**

**Query :** create database BurgerBash;

**Create table runner orders :**

**Query :**

CREATE TABLE runner\_orders(

order\_id INTEGER NOT NULL PRIMARY KEY

,runner\_id INTEGER NOT NULL

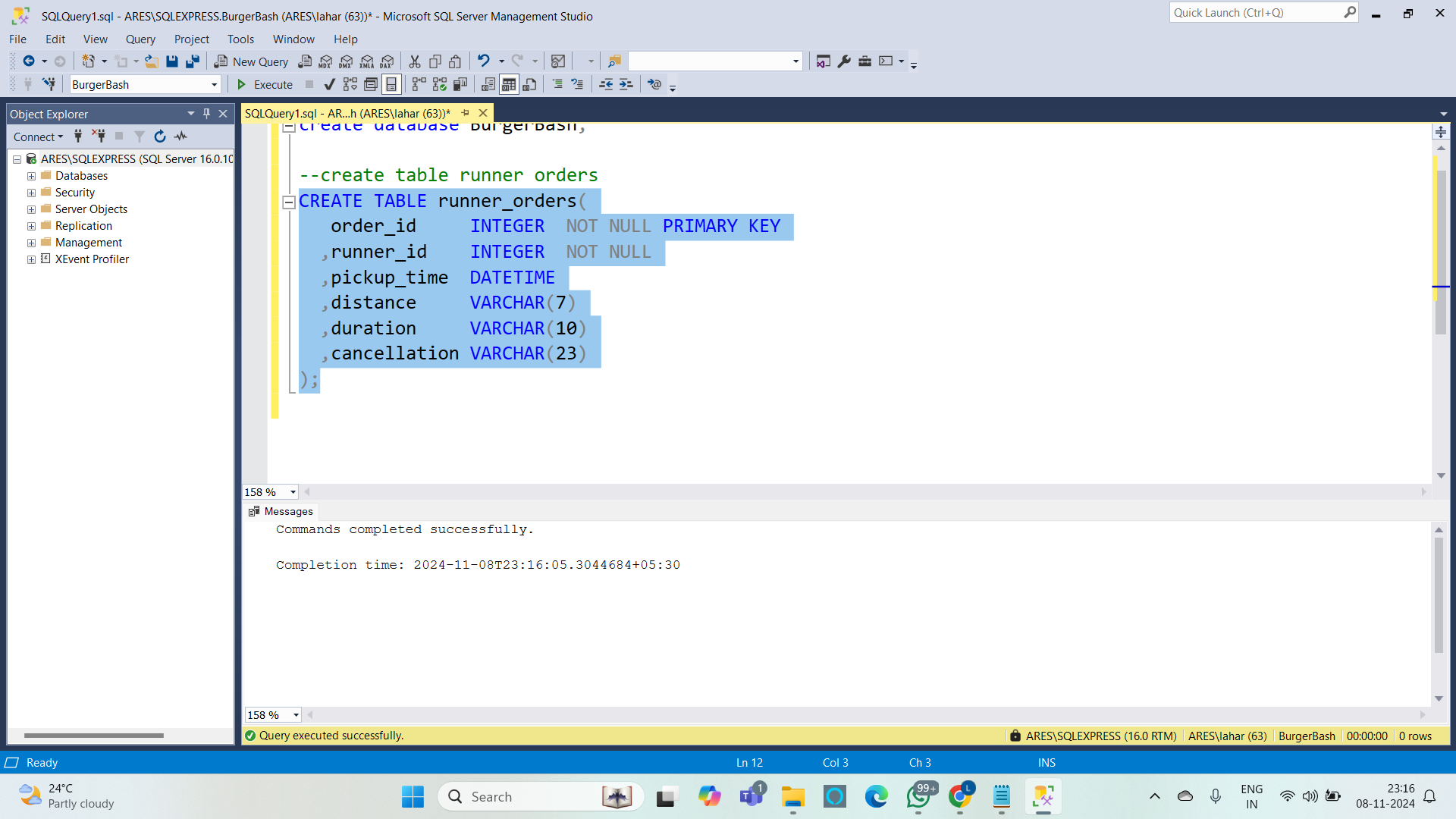
,pickup\_time DATETIME

,distance VARCHAR(7)

,duration VARCHAR(10)

,cancellation VARCHAR(23)

);



**Insert values into runner orders :**

**Query :**

INSERT INTO runner\_orders VALUES (1,1,'2021-01-01 18:15:34','20km','32 minutes',NULL);

INSERT INTO runner\_orders VALUES (2,1,'2021-01-01 19:10:54','20km','27 minutes',NULL);

INSERT INTO runner\_orders VALUES (3,1,'2021-01-03 00:12:37','13.4km','20 mins',NULL);

INSERT INTO runner\_orders VALUES (4,2,'2021-01-04 13:53:03','23.4','40',NULL);

INSERT INTO runner\_orders VALUES (5,3,'2021-01-08 21:10:57','10','15',NULL);

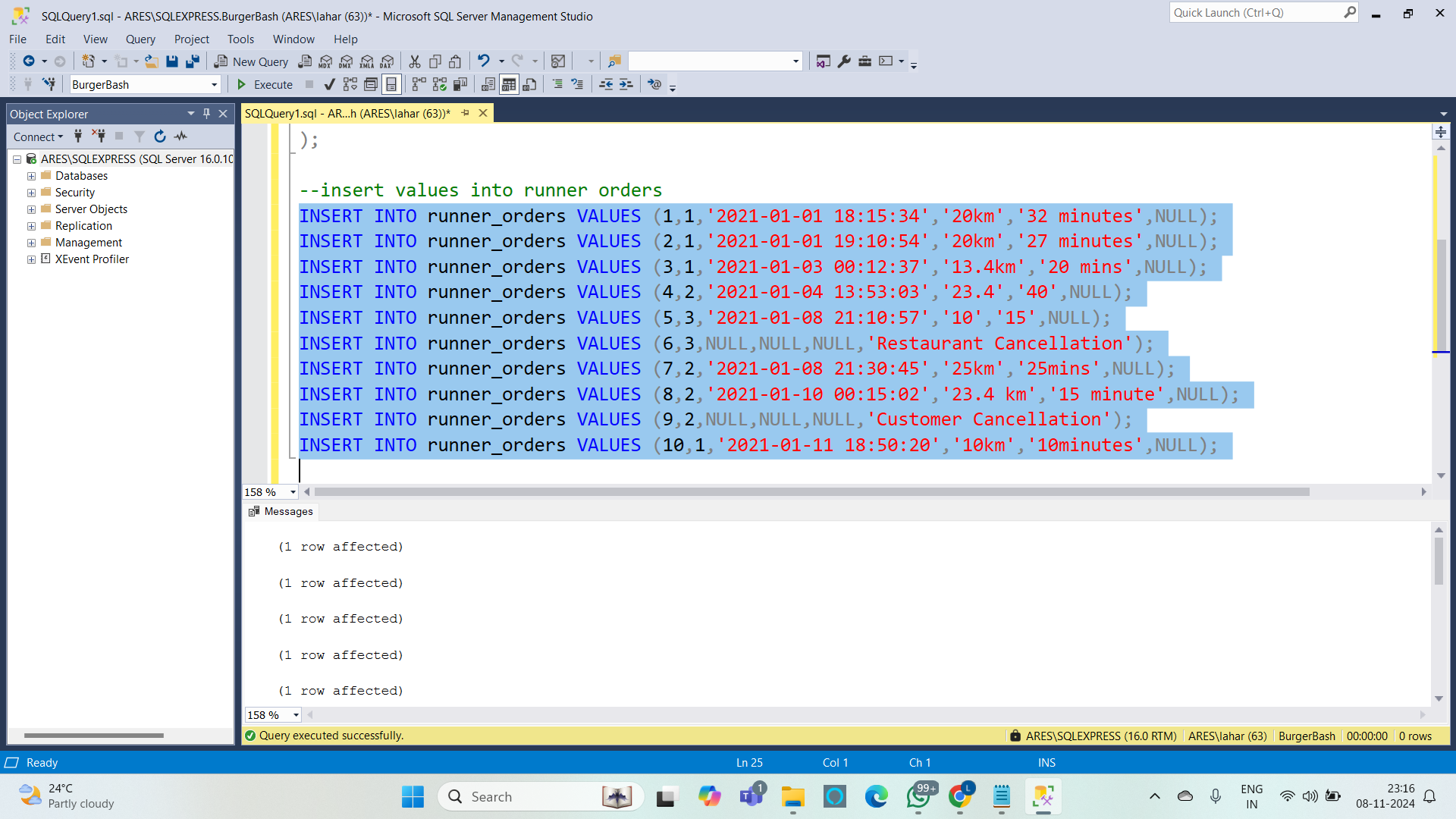
INSERT INTO runner\_orders VALUES (6,3,NULL,NULL,NULL,'Restaurant Cancellation');

INSERT INTO runner\_orders VALUES (7,2,'2021-01-08 21:30:45','25km','25mins',NULL);

INSERT INTO runner\_orders VALUES (8,2,'2021-01-10 00:15:02','23.4 km','15 minute',NULL);

INSERT INTO runner\_orders VALUES (9,2,NULL,NULL,NULL,'Customer Cancellation');

INSERT INTO runner\_orders VALUES (10,1,'2021-01-11 18:50:20','10km','10minutes',NULL);



**Create customer order :**

**Query :**

CREATE TABLE customer\_orders(

order\_id INTEGER NOT NULL

,customer\_id INTEGER NOT NULL

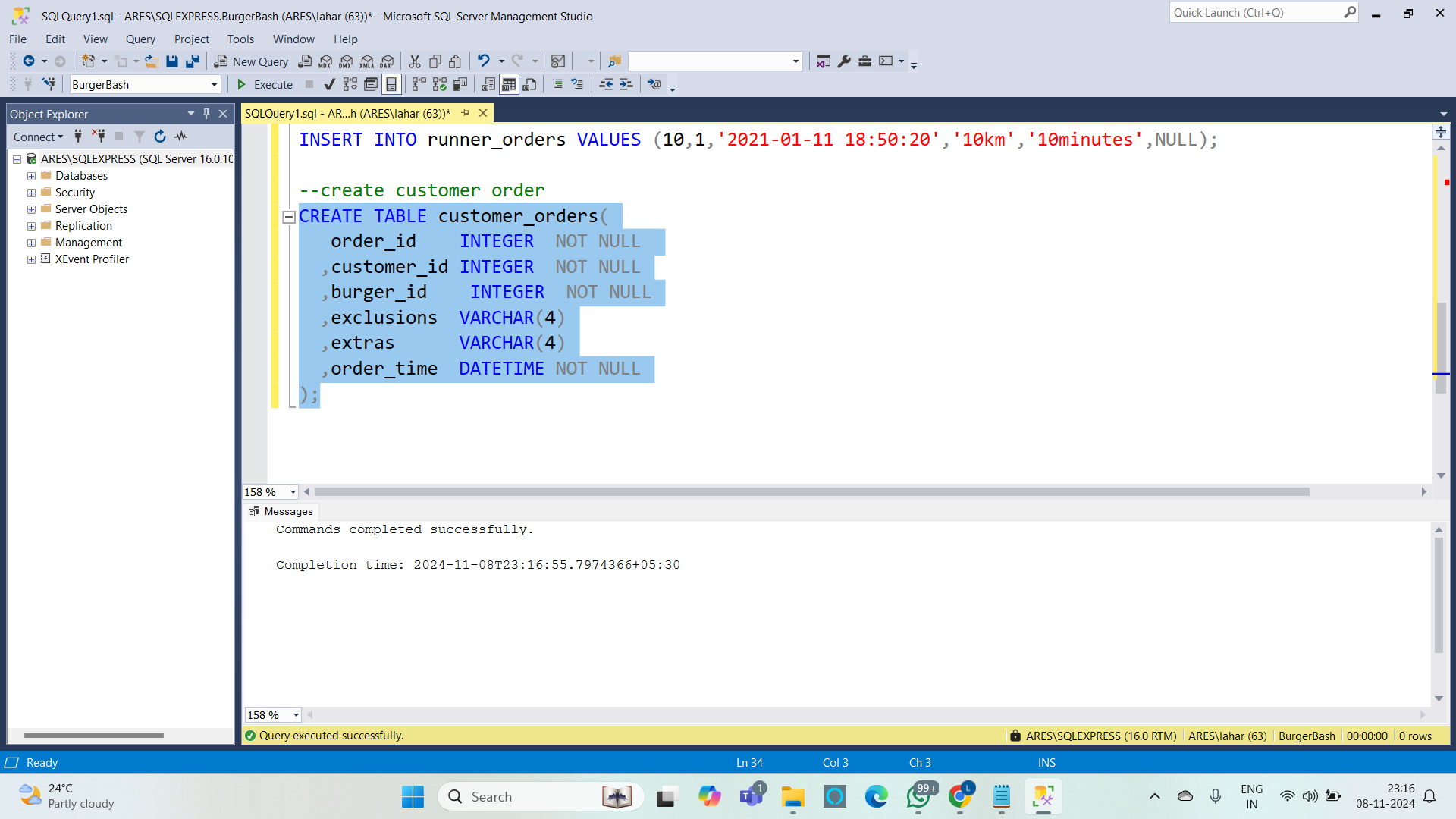
,burger\_id INTEGER NOT NULL

,exclusions VARCHAR(4)

,extras VARCHAR(4)

,order\_time DATETIME NOT NULL

);



**Insert values into customer order :**

**Query :**

INSERT INTO customer\_orders VALUES (1,101,1,NULL,NULL,'2021-01-01 18:05:02');

INSERT INTO customer\_orders VALUES (2,101,1,NULL,NULL,'2021-01-01 19:00:52');

INSERT INTO customer\_orders VALUES (3,102,1,NULL,NULL,'2021-01-02 23:51:23');

INSERT INTO customer\_orders VALUES (3,102,2,NULL,NULL,'2021-01-02 23:51:23');

INSERT INTO customer\_orders VALUES (4,103,1,'4',NULL,'2021-01-04 13:23:46');

INSERT INTO customer\_orders VALUES (4,103,1,'4',NULL,'2021-01-04 13:23:46');

INSERT INTO customer\_orders VALUES (4,103,2,'4',NULL,'2021-01-04 13:23:46');

INSERT INTO customer\_orders VALUES (5,104,1,NULL,'1','2021-01-08 21:00:29');

INSERT INTO customer\_orders VALUES (6,101,2,NULL,NULL,'2021-01-08 21:03:13');

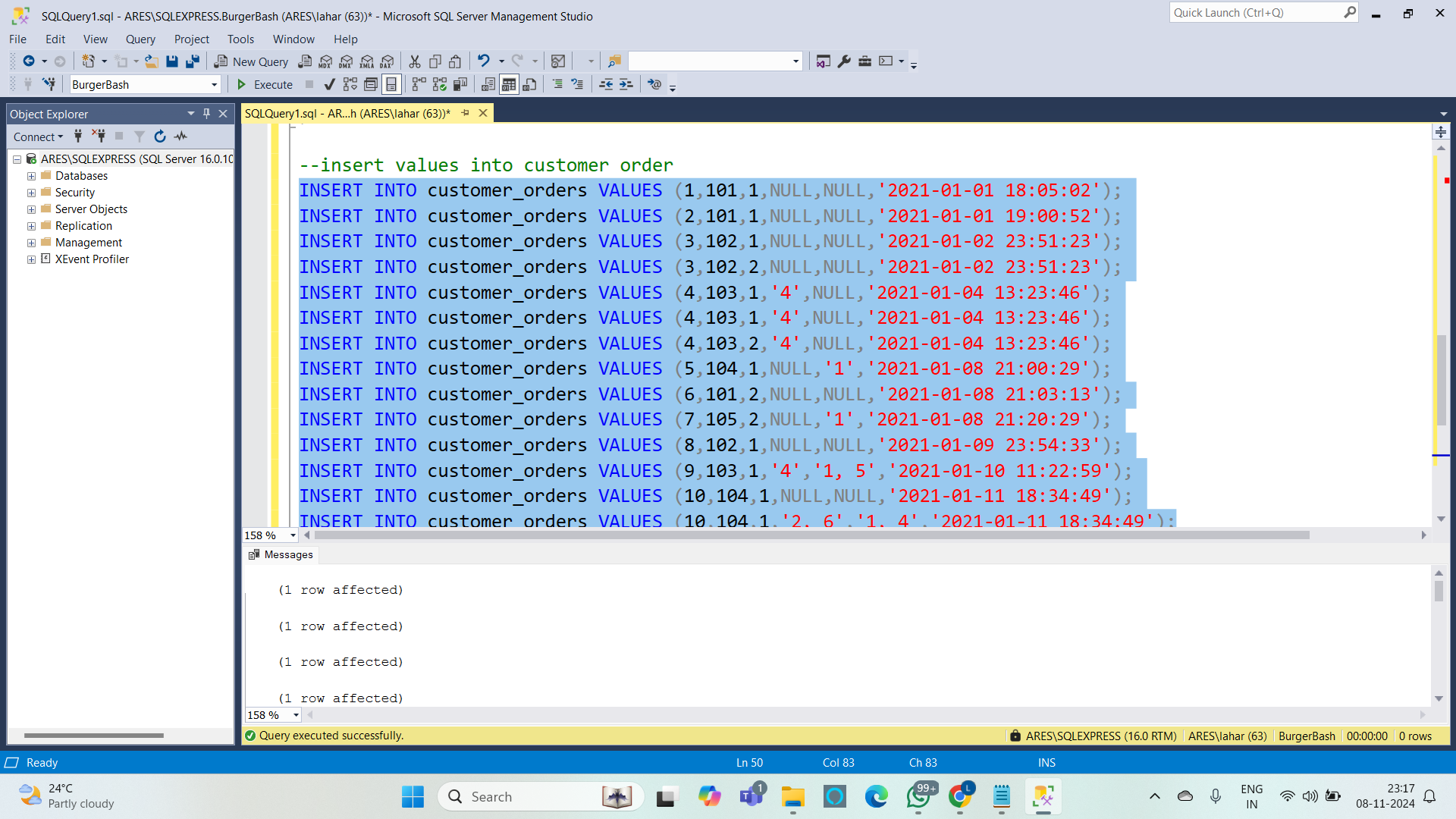
INSERT INTO customer\_orders VALUES (7,105,2,NULL,'1','2021-01-08 21:20:29');

INSERT INTO customer\_orders VALUES (8,102,1,NULL,NULL,'2021-01-09 23:54:33');

INSERT INTO customer\_orders VALUES (9,103,1,'4','1, 5','2021-01-10 11:22:59');

INSERT INTO customer\_orders VALUES (10,104,1,NULL,NULL,'2021-01-11 18:34:49');

INSERT INTO customer\_orders VALUES (10,104,1,'2, 6','1, 4','2021-01-11 18:34:49');



**Create burger table :**

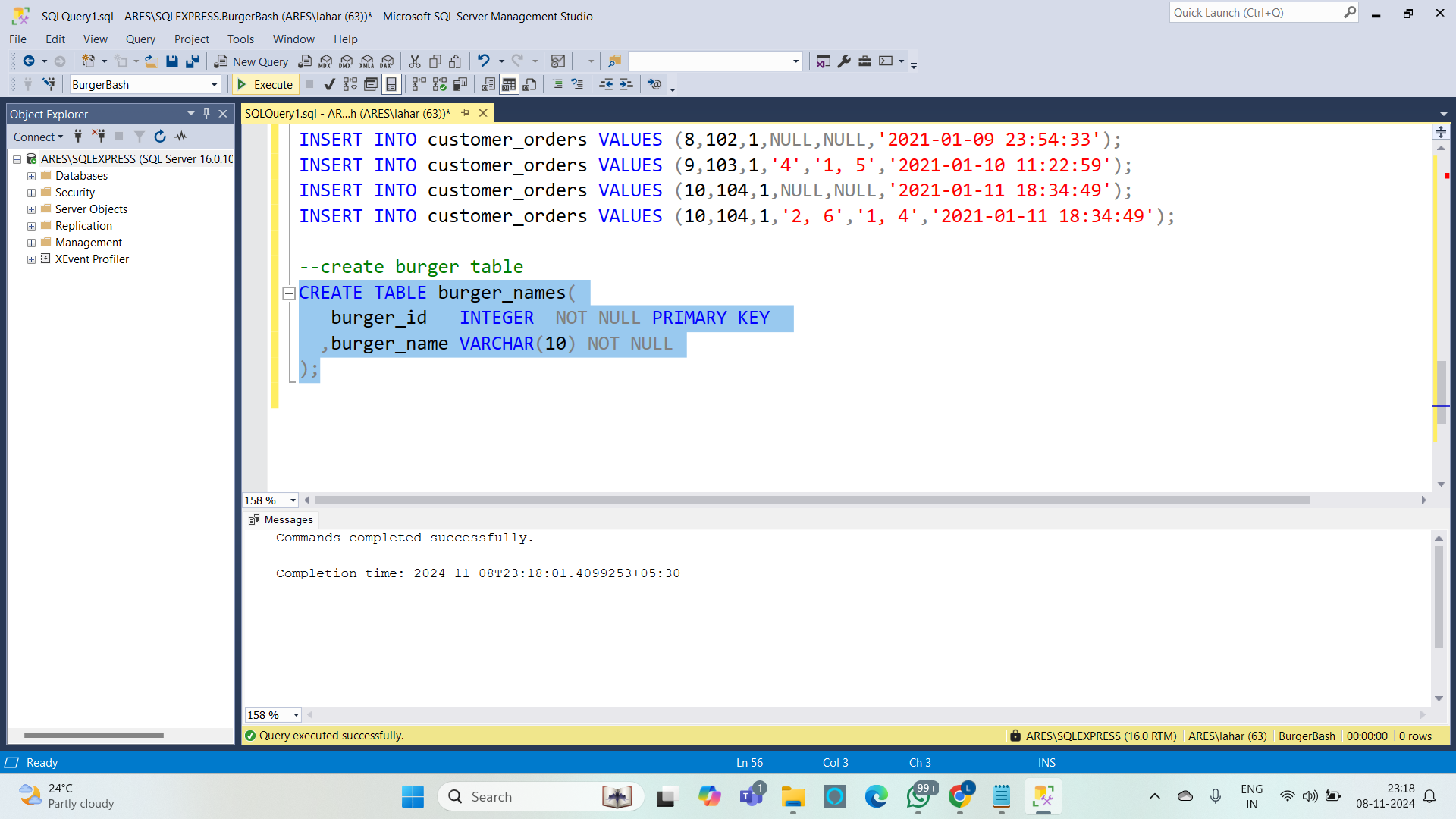
**Query :**

CREATE TABLE burger\_names(

burger\_id INTEGER NOT NULL PRIMARY KEY

,burger\_name VARCHAR(10) NOT NULL

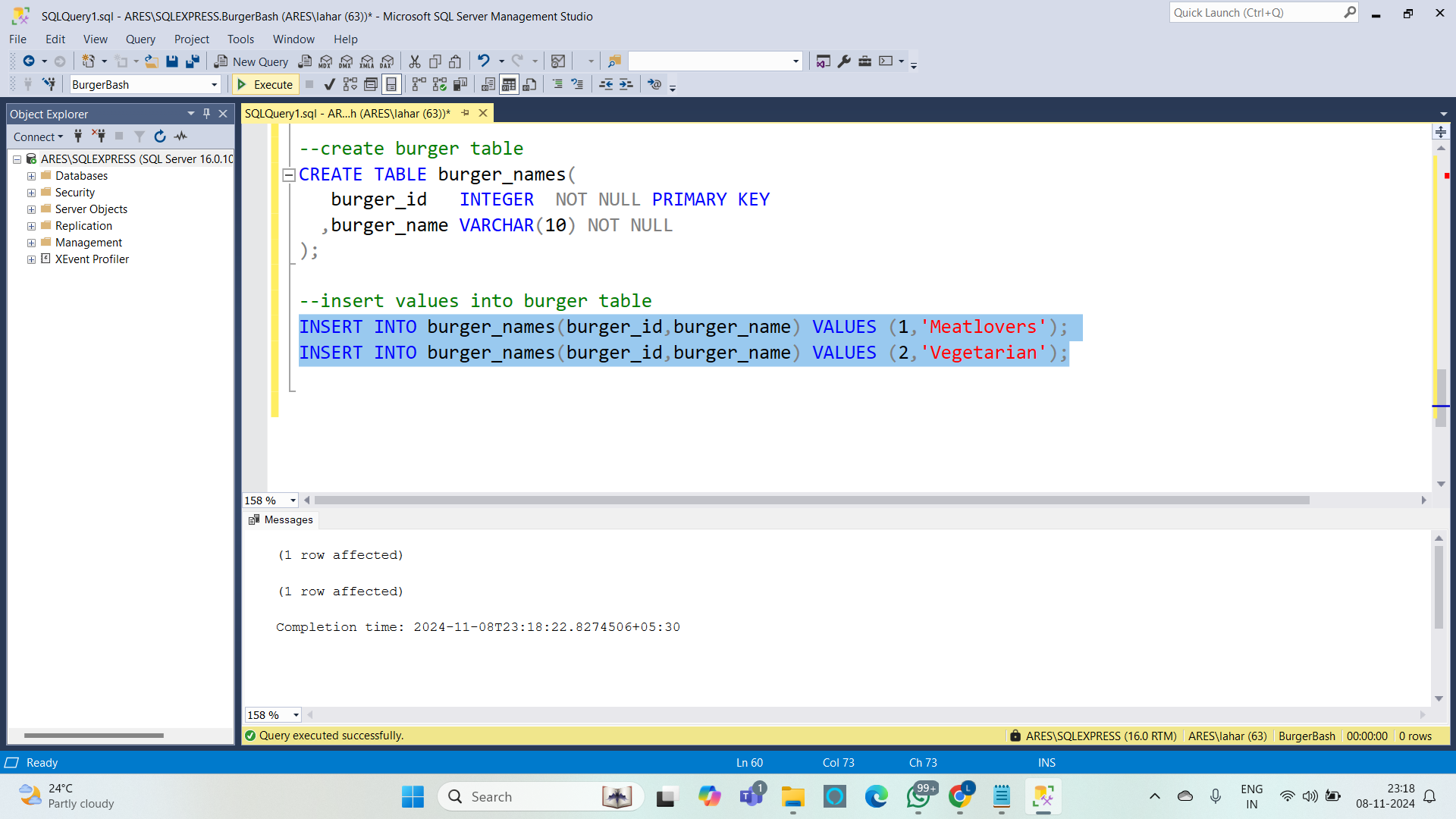
);



**Insert values into burger table :**

**Query :**

INSERT INTO burger\_names(burger\_id,burger\_name) VALUES (1,'Meatlovers');

INSERT INTO burger\_names(burger\_id,burger\_name) VALUES (2,'Vegetarian');

**Create burger runner :**

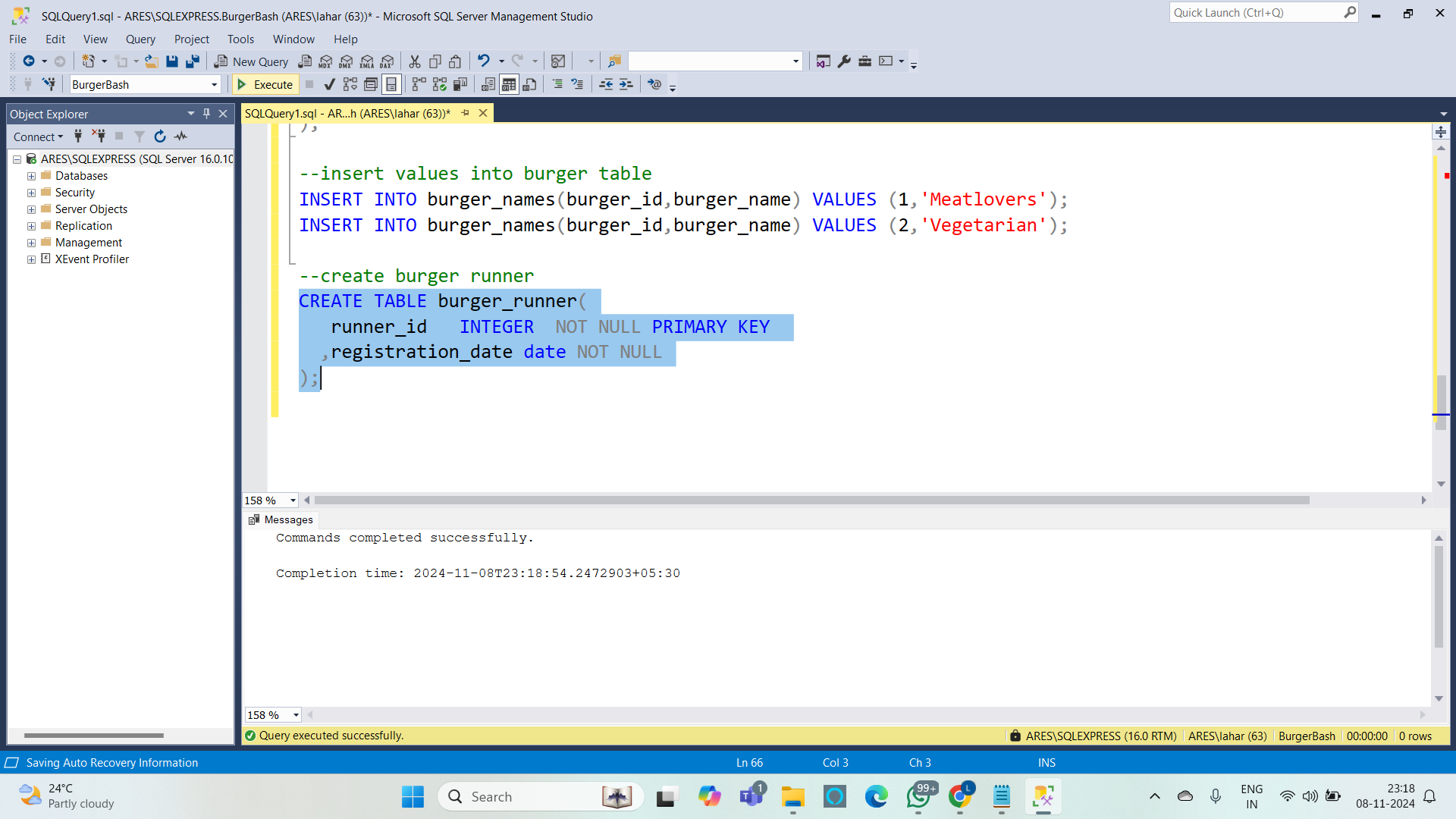
**Query :**

CREATE TABLE burger\_runner(

runner\_id INTEGER NOT NULL PRIMARY KEY

,registration\_date date NOT NULL

);



**Insert values into burger runner :**

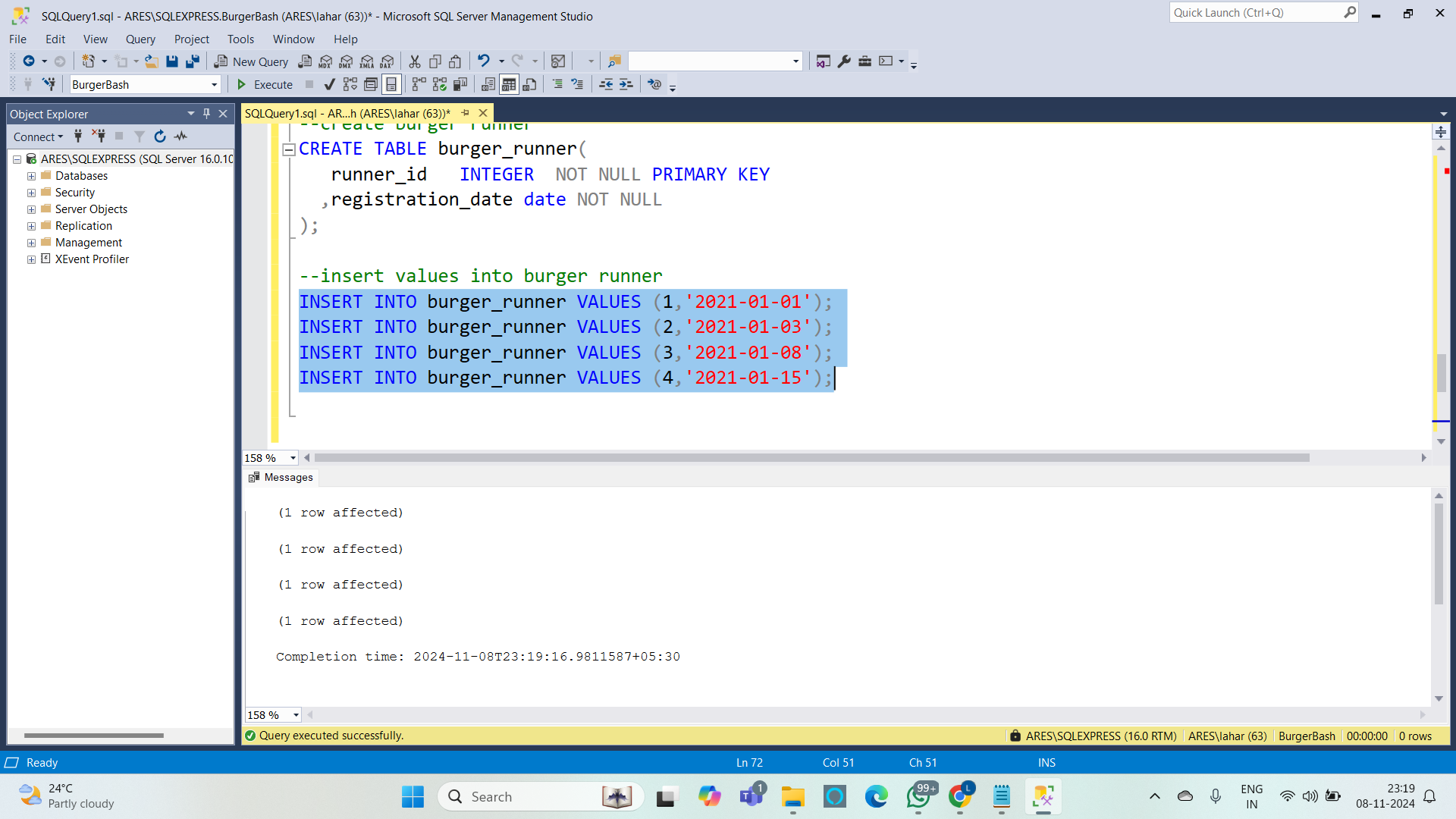
**Query :**

INSERT INTO burger\_runner VALUES (1,'2021-01-01');

INSERT INTO burger\_runner VALUES (2,'2021-01-03');

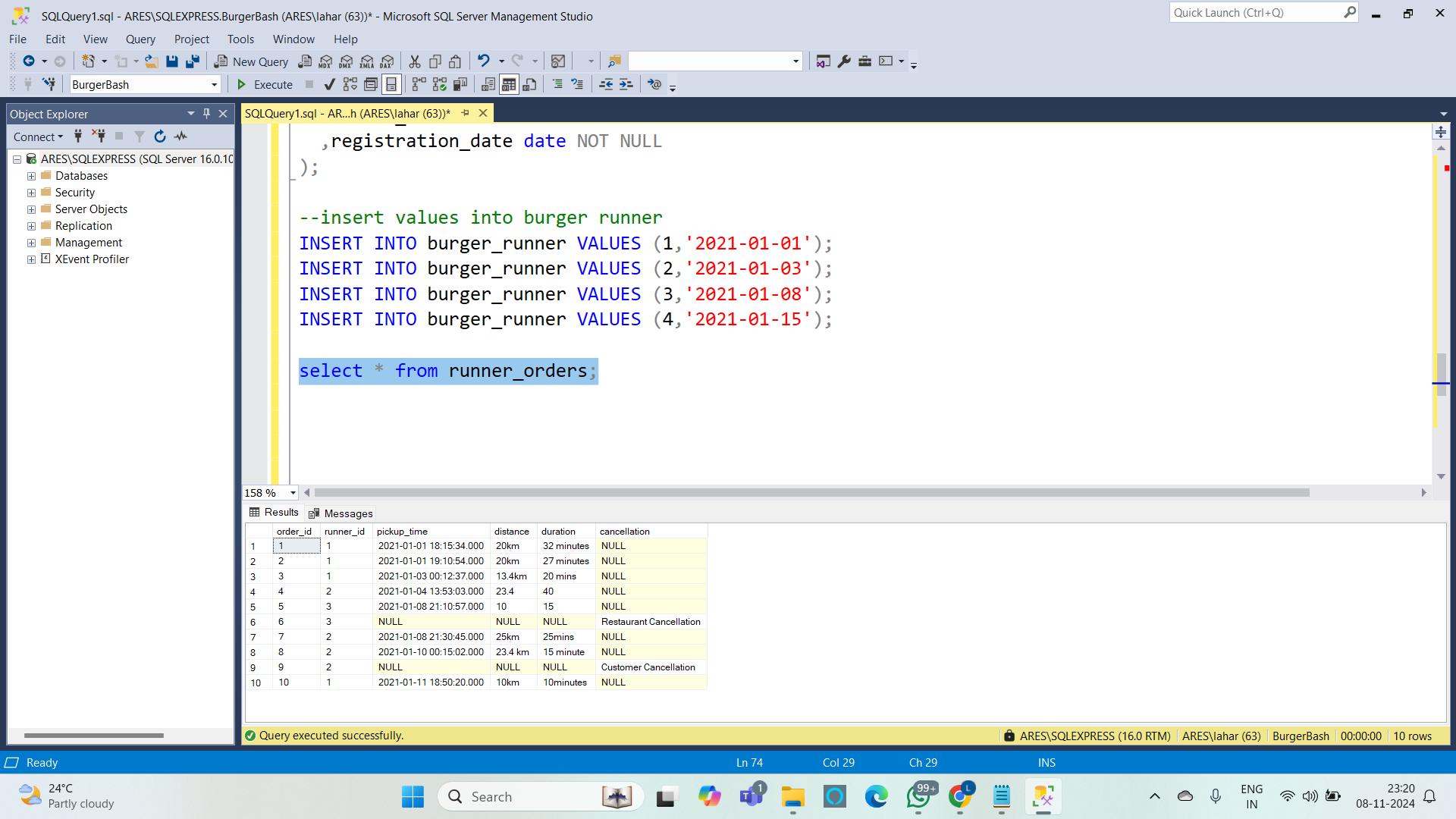
INSERT INTO burger\_runner VALUES (3,'2021-01-08');

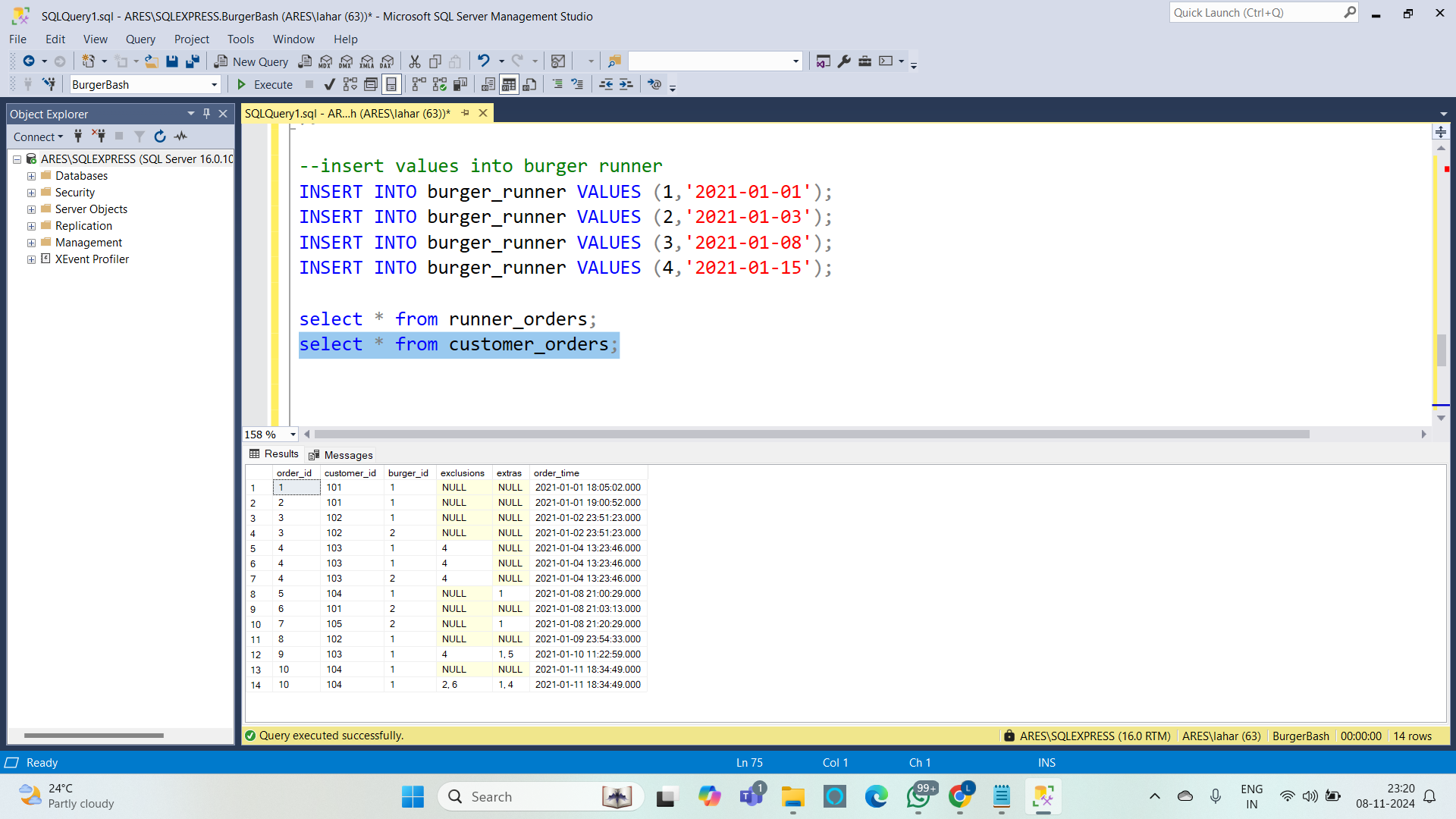
INSERT INTO burger\_runner VALUES (4,'2021-01-15');



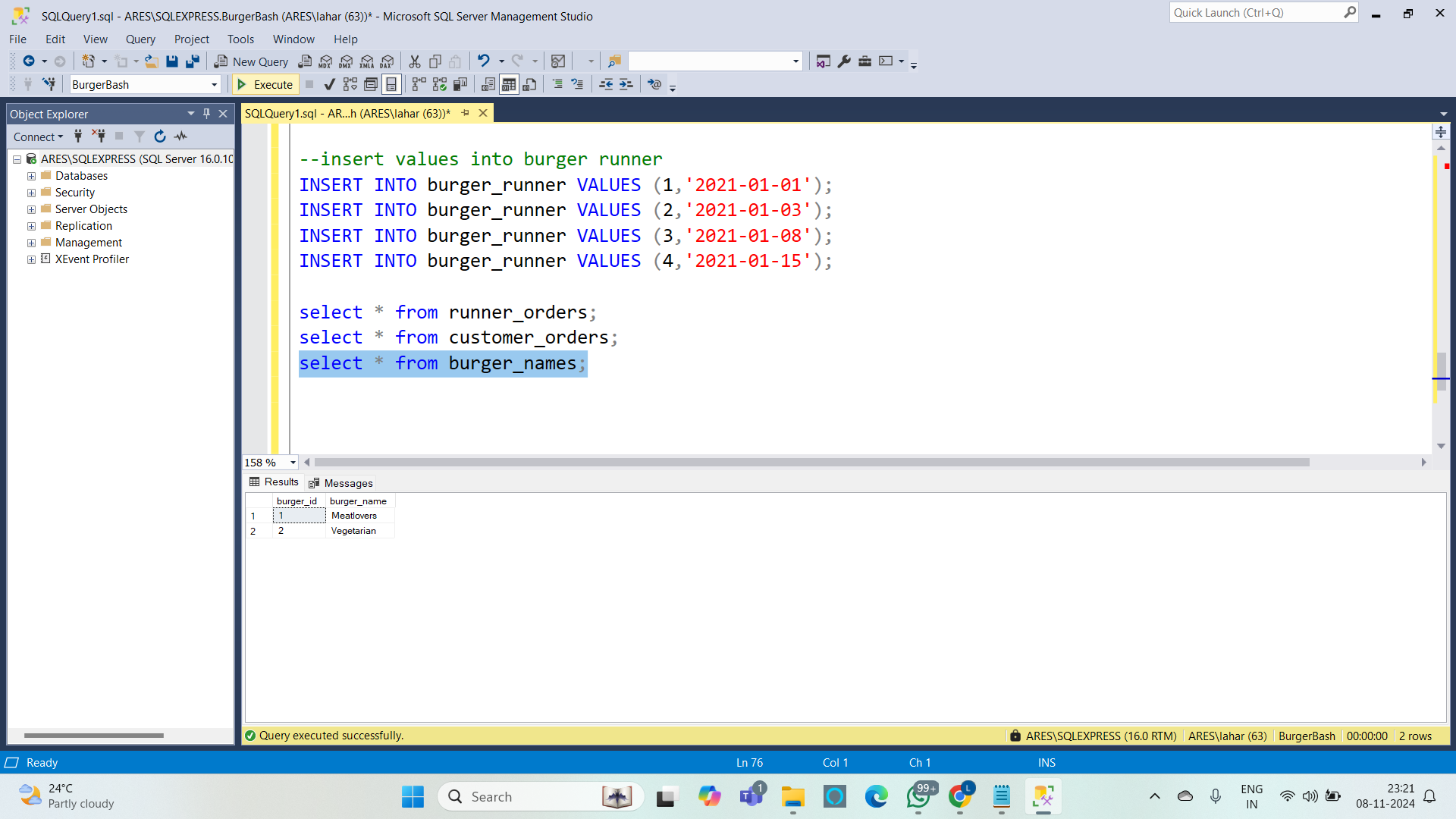
**Representing tables :**

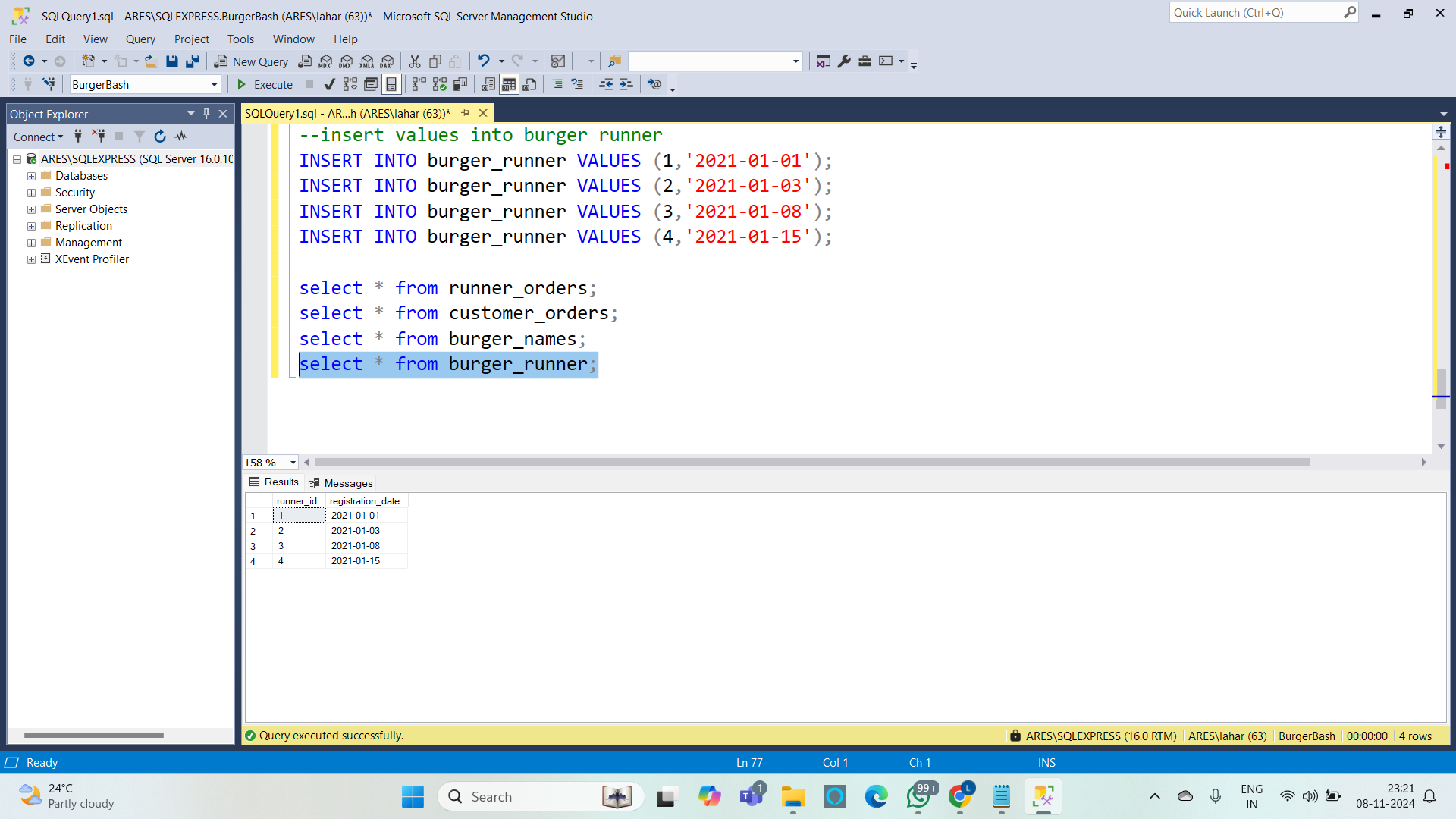
**Query :**

select \* from runner\_orders;

select \* from customer\_orders;

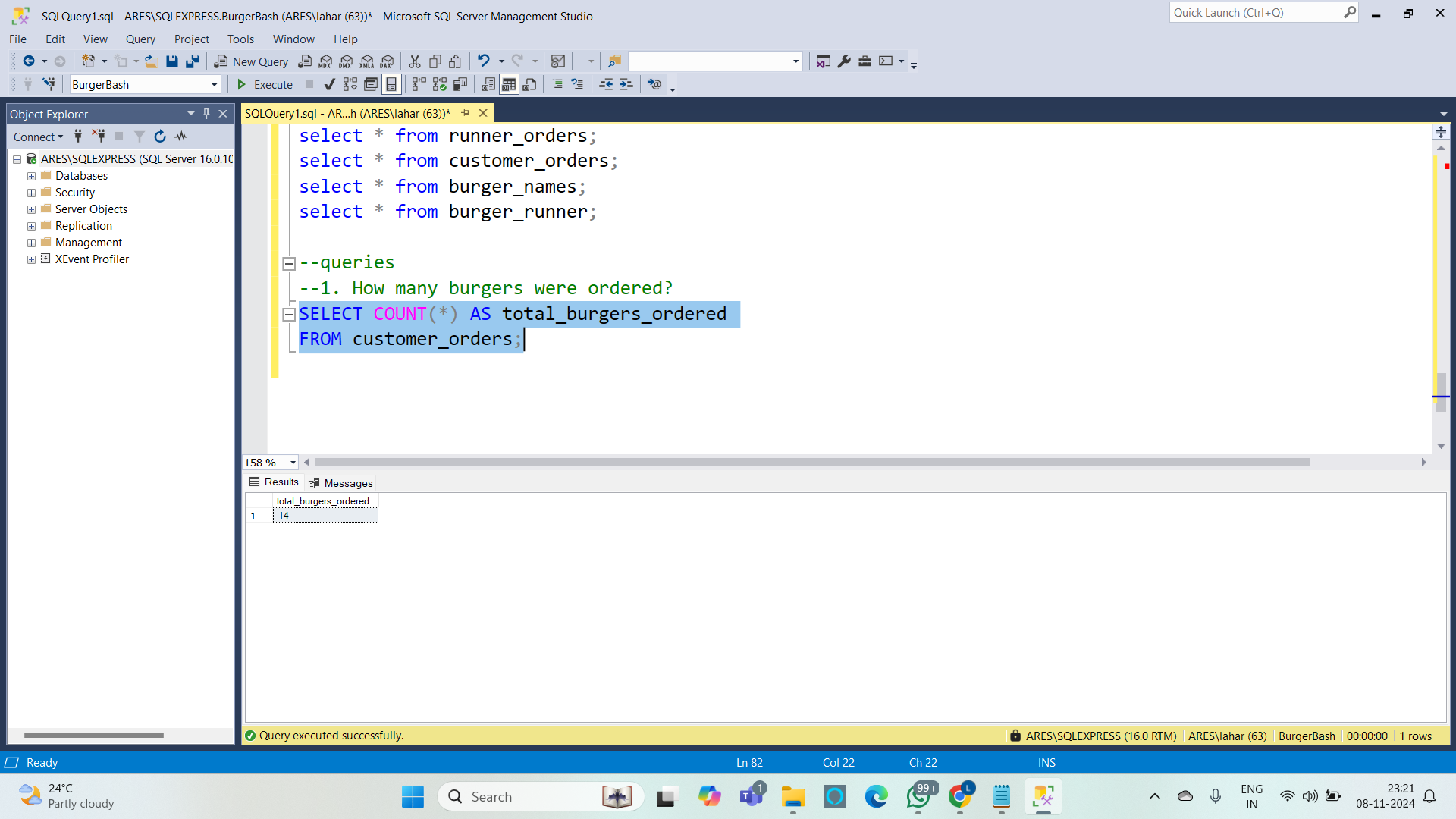
select \* from burger\_names;



select \* from burger\_runner;

**Queries :**

**1. How many burgers were ordered?**

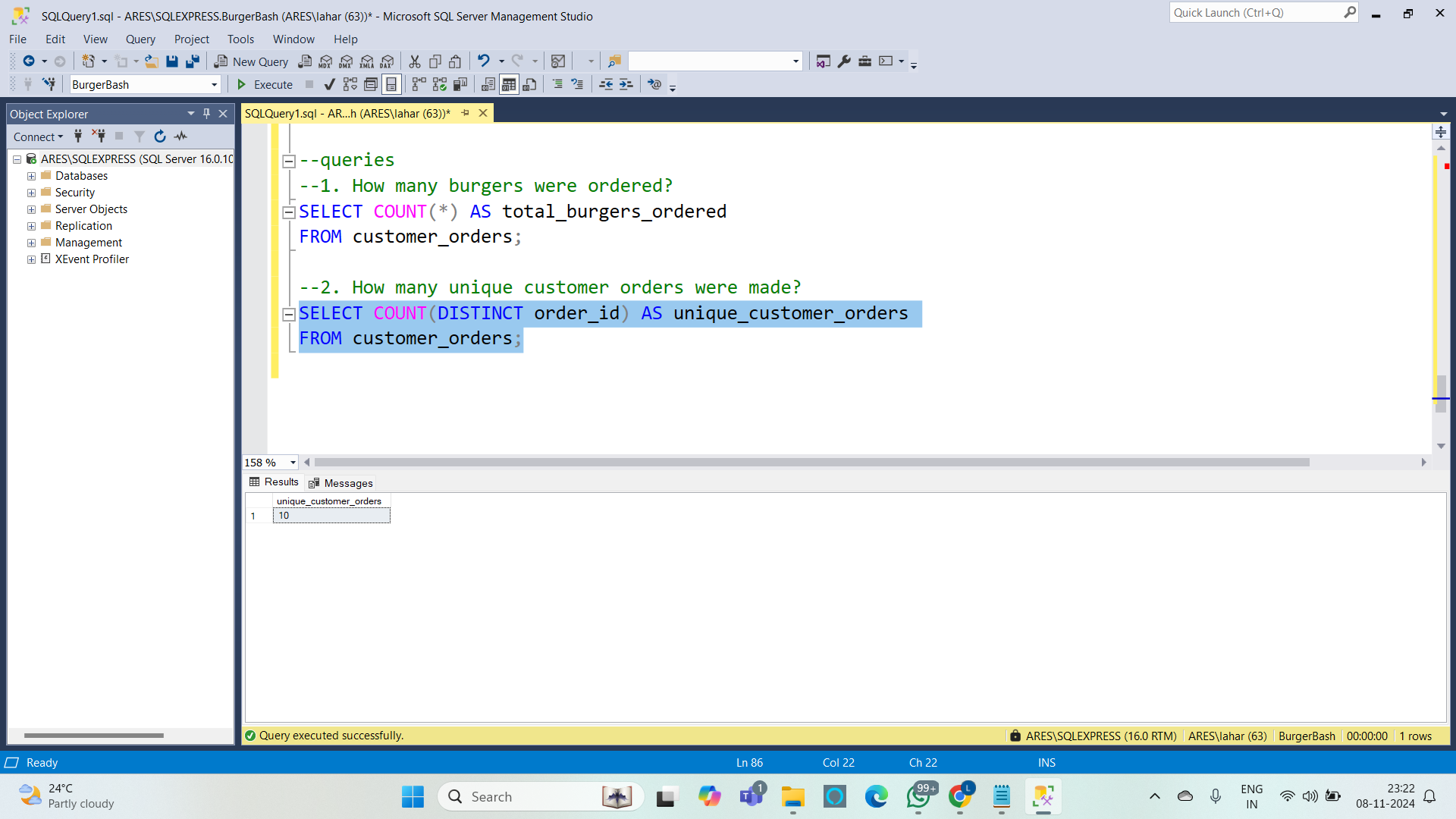
**Query :** SELECT COUNT(\*) AS total\_burgers\_ordered FROM customer\_orders;

**2. How many unique customer orders were made?**

**Query :**

SELECT COUNT(DISTINCT order\_id) AS unique\_customer\_orders

FROM customer\_orders;



**3. How many successful orders were delivered by each runner?**

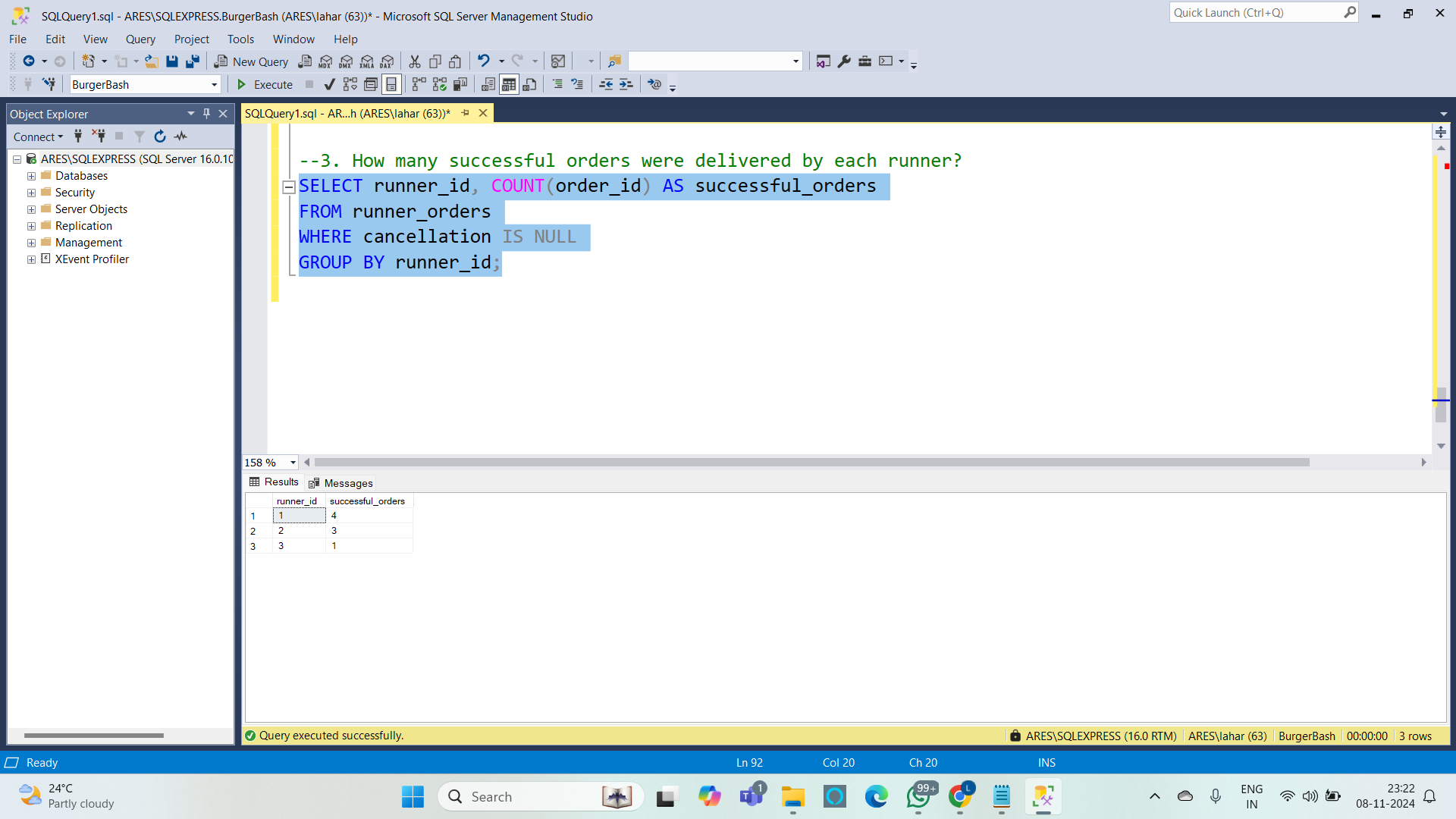
**Query :**

SELECT runner\_id, COUNT(order\_id) AS successful\_orders

FROM runner\_orders

WHERE cancellation IS NULL

GROUP BY runner\_id;



**4. How many of each type of burger was delivered?**

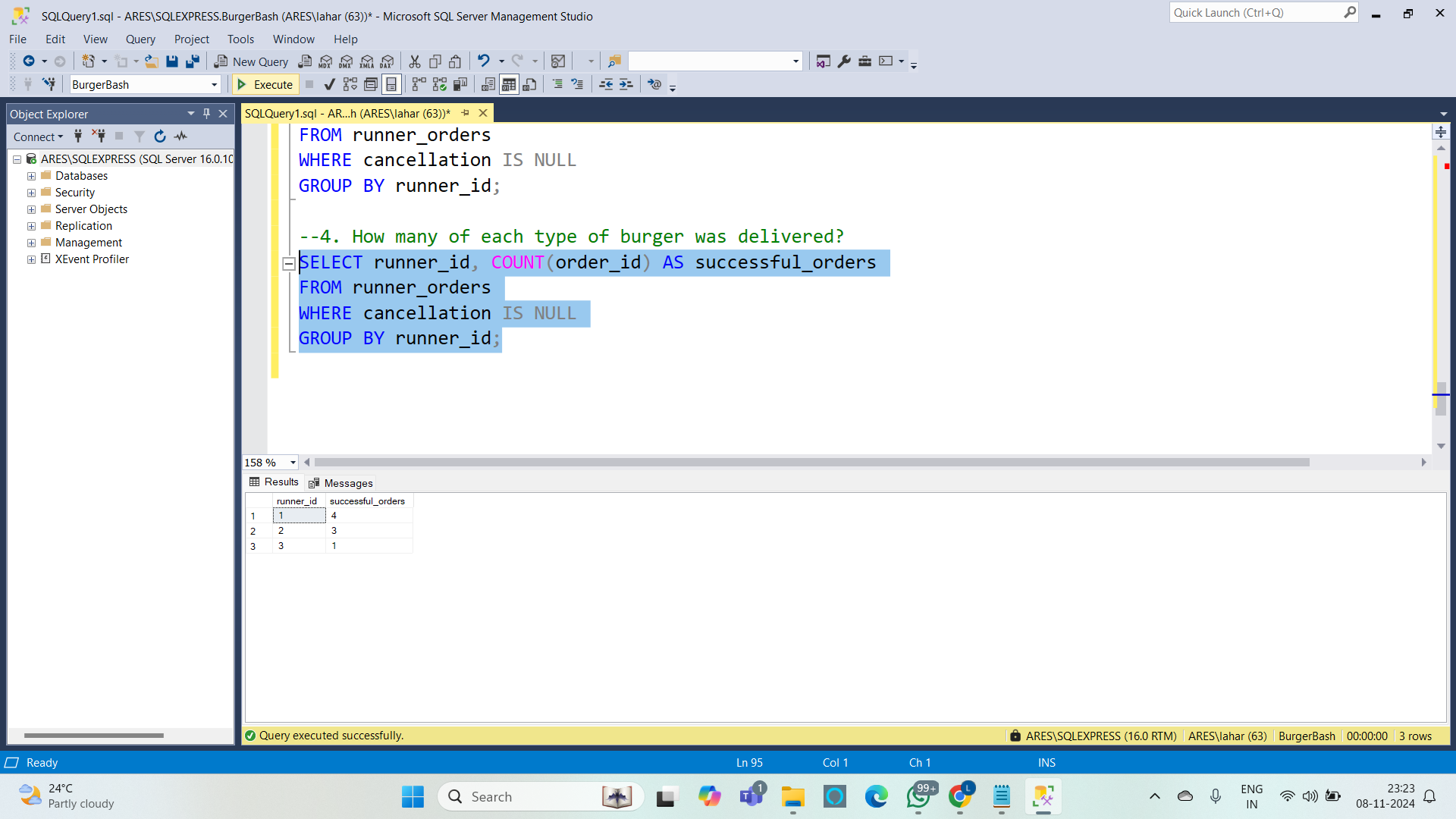
**Query :**

SELECT runner\_id, COUNT(order\_id) AS successful\_orders

FROM runner\_orders

WHERE cancellation IS NULL

GROUP BY runner\_id;



**5. How many Vegetarian and Meatlovers were ordered by each customer?**

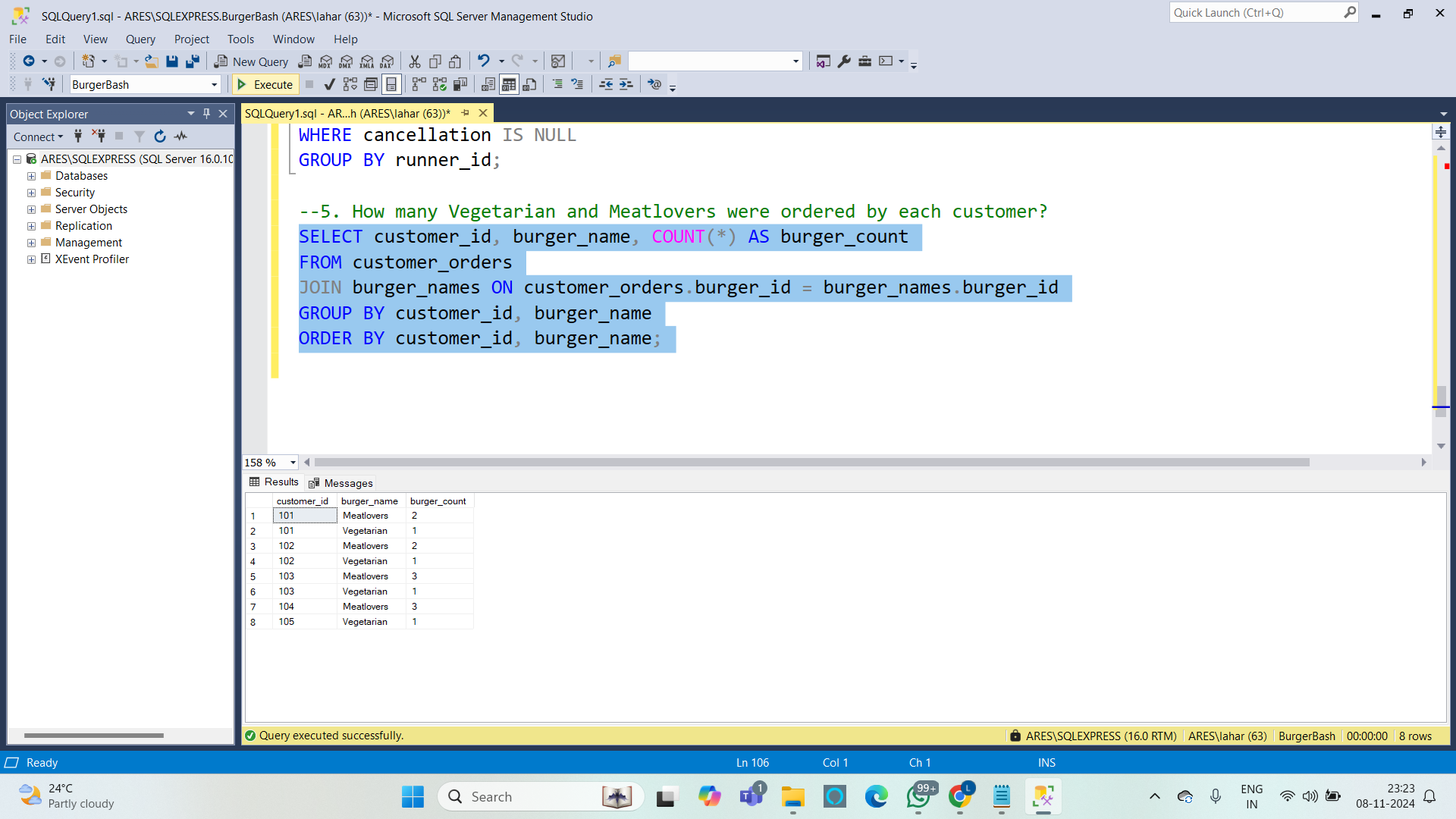
**Query :**

SELECT customer\_id, burger\_name, COUNT(\*) AS burger\_count

FROM customer\_orders

JOIN burger\_names ON customer\_orders.burger\_id = burger\_names.burger\_id

GROUP BY customer\_id, burger\_name

ORDER BY customer\_id, burger\_name;

**6. What was the maximum number of burgers delivered in a single order?**

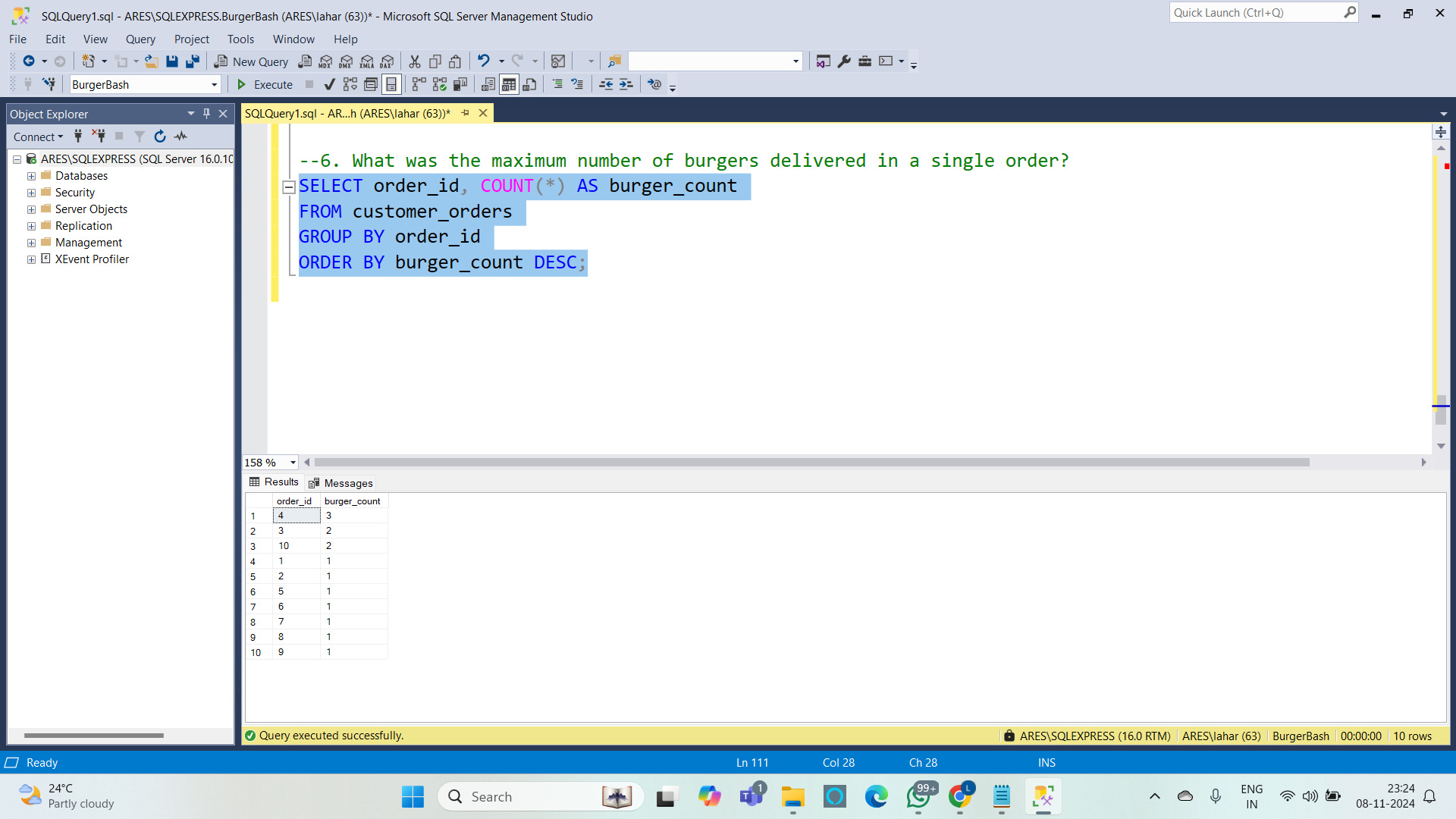
**Query :**

SELECT order\_id, COUNT(\*) AS burger\_count

FROM customer\_orders

GROUP BY order\_id

ORDER BY burger\_count DESC;



**7. For each customer, how many delivered burgers had at least 1 change and how many had no changes?**

**Query :**

SELECT

customer\_orders.customer\_id,

SUM(CASE WHEN (exclusions IS NOT NULL OR extras IS NOT NULL) THEN 1 ELSE 0 END)

AS burgers\_with\_changes,

SUM(CASE WHEN (exclusions IS NULL AND extras IS NULL) THEN 1 ELSE 0 END)

AS burgers\_without\_changes

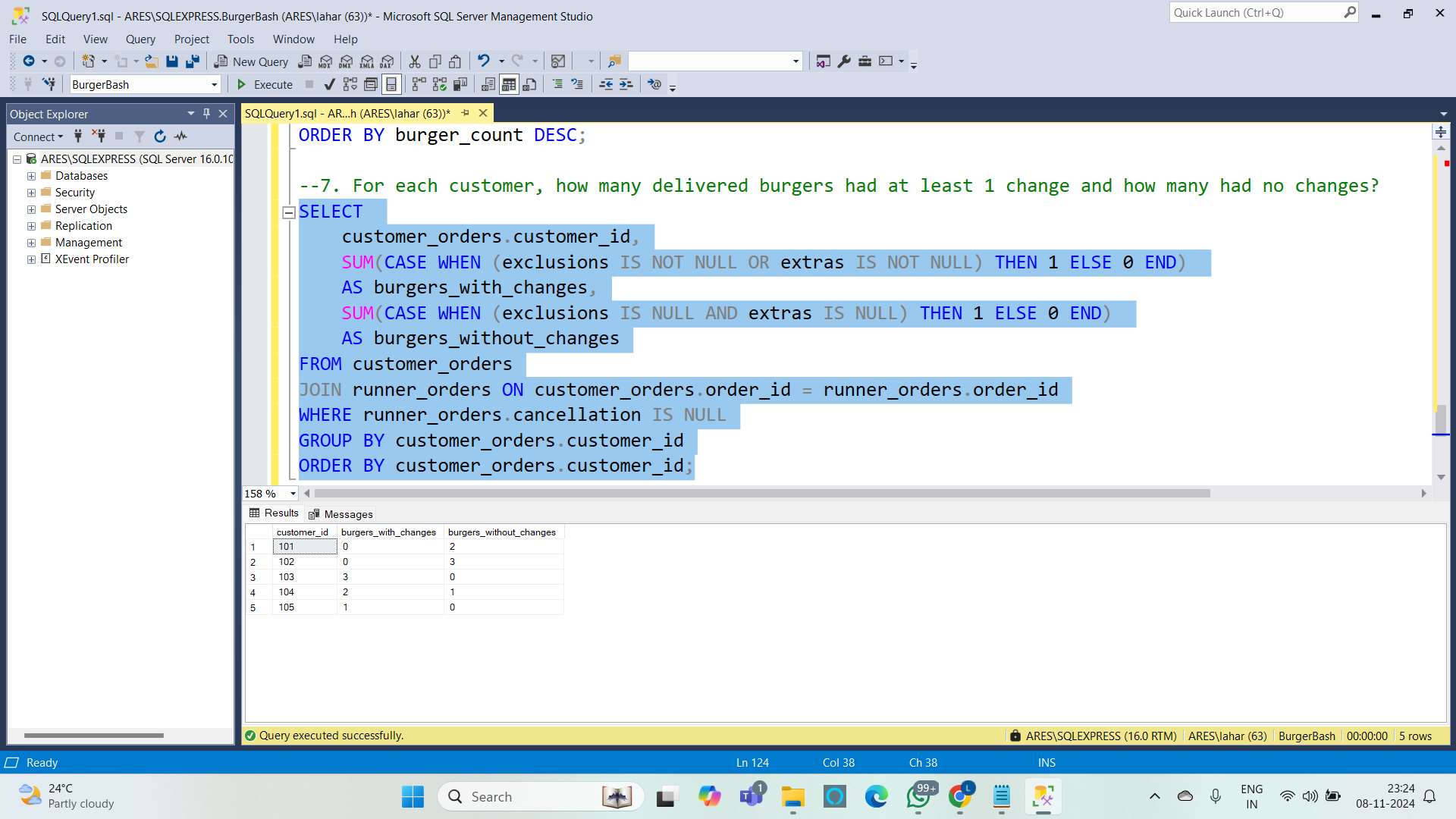
FROM customer\_orders

JOIN runner\_orders ON customer\_orders.order\_id = runner\_orders.order\_id

WHERE runner\_orders.cancellation IS NULL

GROUP BY customer\_orders.customer\_id

ORDER BY customer\_orders.customer\_id;



**8. What was the total volume of burgers ordered for each hour of the day?**

**Query :**

SELECT

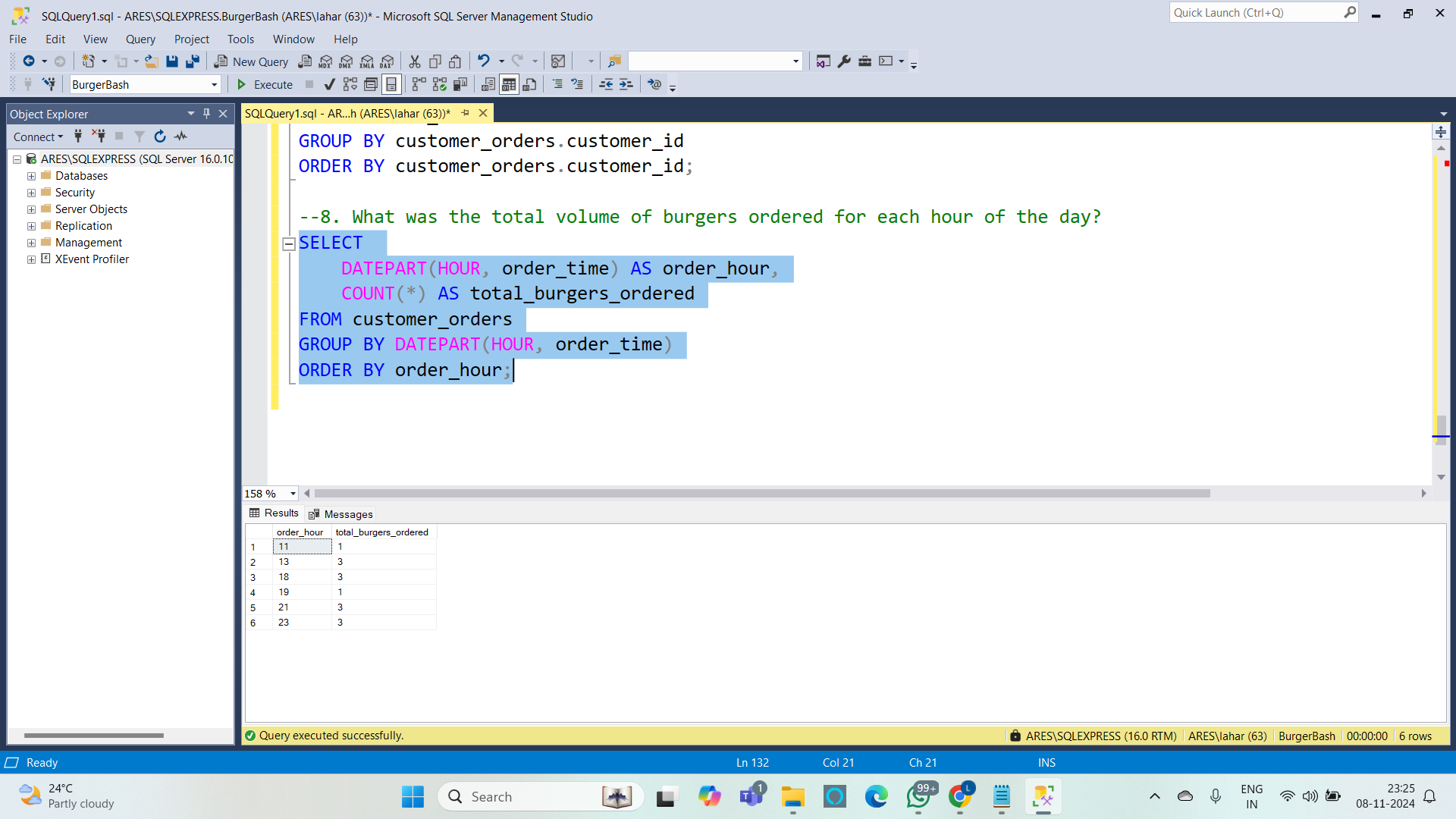
DATEPART(HOUR, order\_time) AS order\_hour,

COUNT(\*) AS total\_burgers\_ordered

FROM customer\_orders

GROUP BY DATEPART(HOUR, order\_time)

ORDER BY order\_hour;



**9. How many runners signed up for each 1 week period?**

SELECT

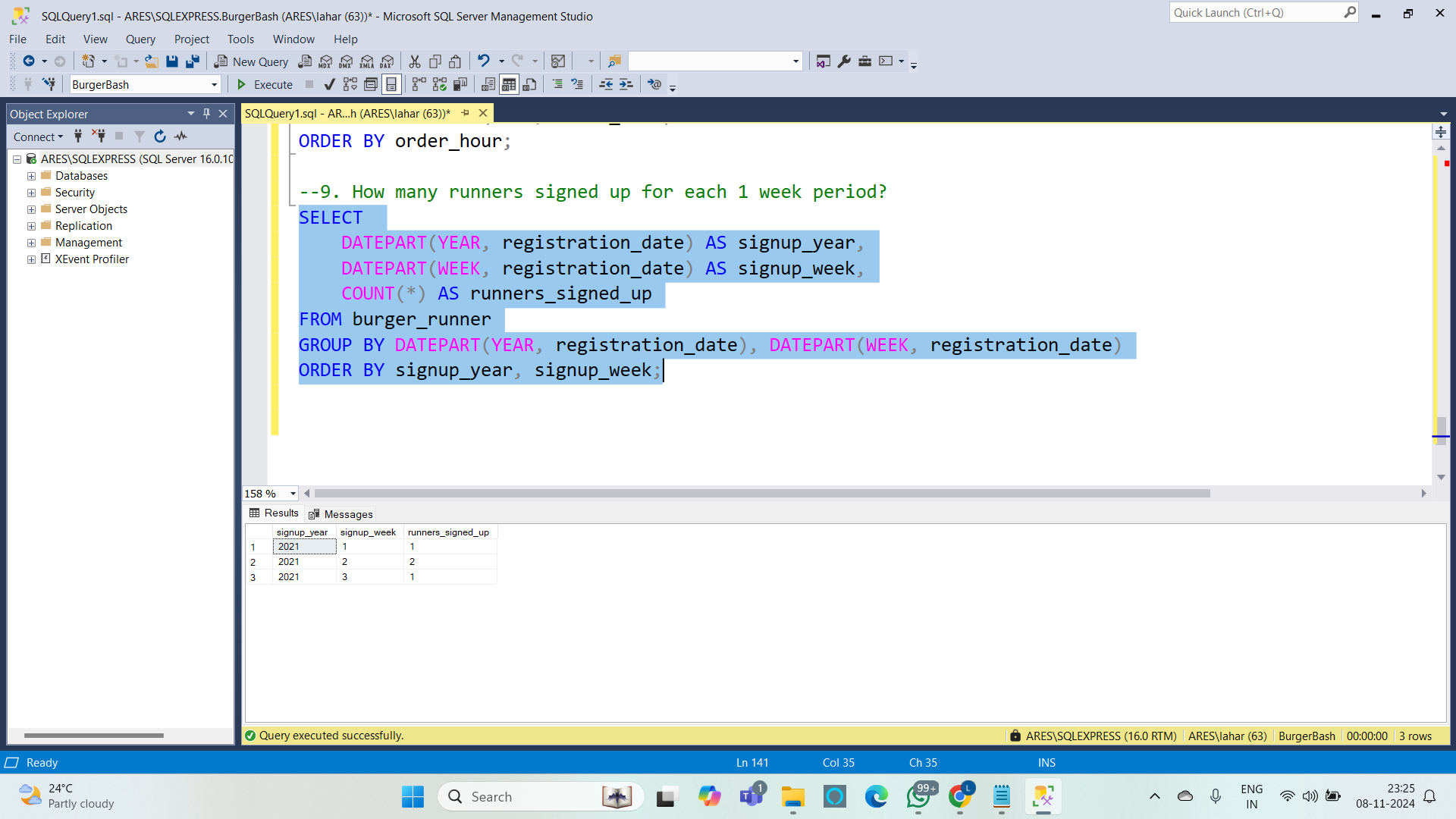
DATEPART(YEAR, registration\_date) AS signup\_year,

DATEPART(WEEK, registration\_date) AS signup\_week,

COUNT(\*) AS runners\_signed\_up

FROM burger\_runner

GROUP BY DATEPART(YEAR, registration\_date), DATEPART(WEEK, registration\_date)

ORDER BY signup\_year, signup\_week;

**10. What was the average distance travelled for each customer?**

**Query :**

SELECT

customer\_orders.customer\_id,

AVG(CAST(REPLACE(runner\_orders.distance, 'km', '') AS FLOAT)) AS avg\_distance\_travelled

FROM customer\_orders

JOIN runner\_orders ON customer\_orders.order\_id = runner\_orders.order\_id

WHERE runner\_orders.cancellation IS NULL

GROUP BY customer\_orders.customer\_id

ORDER BY customer\_orders.customer\_id;



**Subqueries**

**1. Find the customer who ordered the most burgers (GROUP BY, HAVING)**

**Query :**

SELECT customer\_id, COUNT(\*) AS total\_burgers\_ordered

FROM customer\_orders

GROUP BY customer\_id

HAVING COUNT(\*) = (

SELECT MAX(burger\_count)

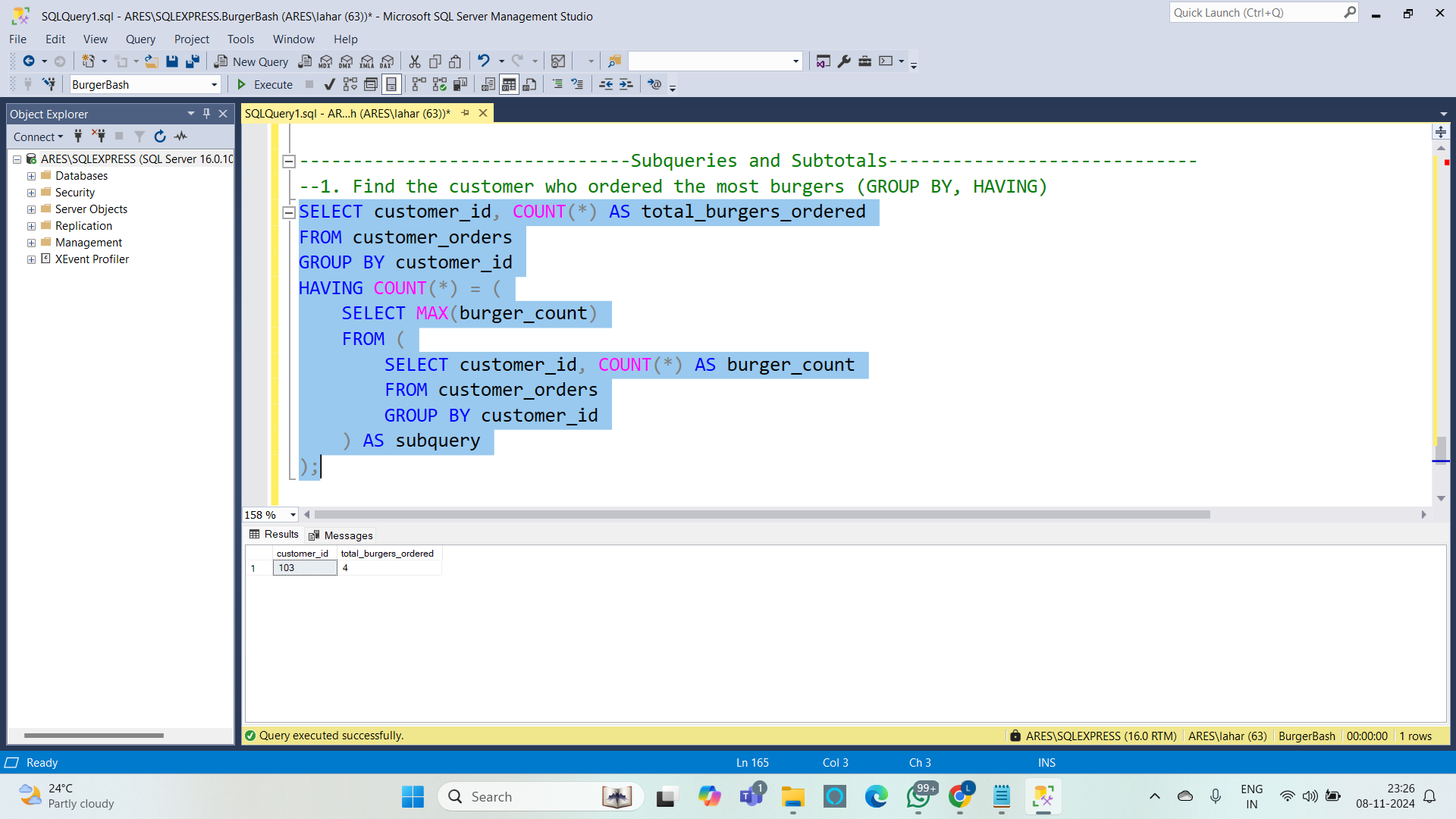
FROM (

SELECT customer\_id, COUNT(\*) AS burger\_count

FROM customer\_orders

GROUP BY customer\_id

) AS subquery

);

**2. Find the total number of cancellations (by customer or restaurant) for each runner (Subqueries)**

**Query :**

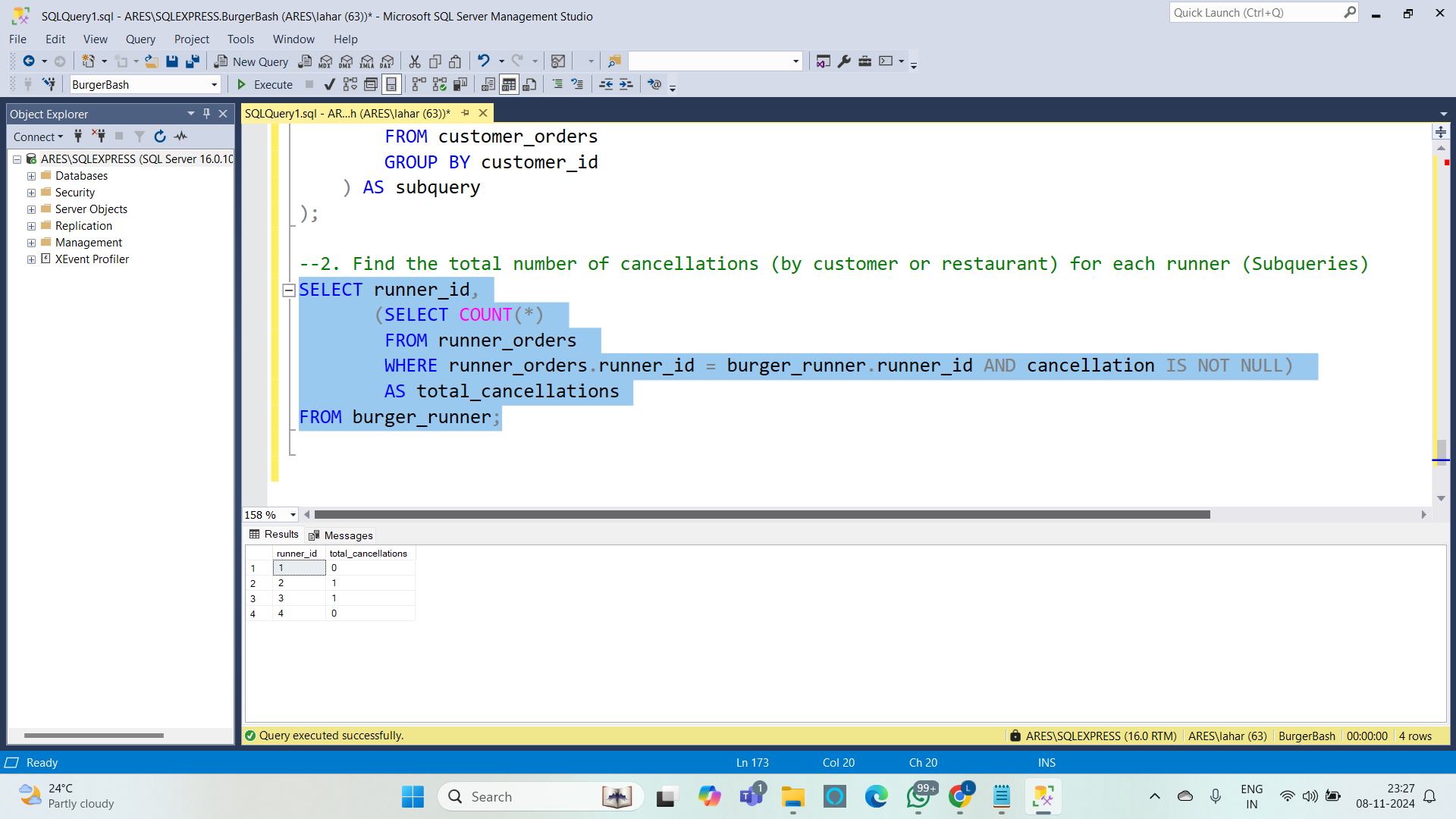
SELECT runner\_id,

(SELECT COUNT(\*)

FROM runner\_orders

WHERE runner\_orders.runner\_id = burger\_runner.runner\_id AND cancellation IS NOT NULL)

AS total\_cancellations

FROM burger\_runner;

**3. Find the average distance for each burger type ordered (Meatlovers vs. Vegetarian)**

**Query :**

SELECT burger\_name,

AVG(CAST(REPLACE(runner\_orders.distance, 'km', '') AS FLOAT)) AS avg\_distance

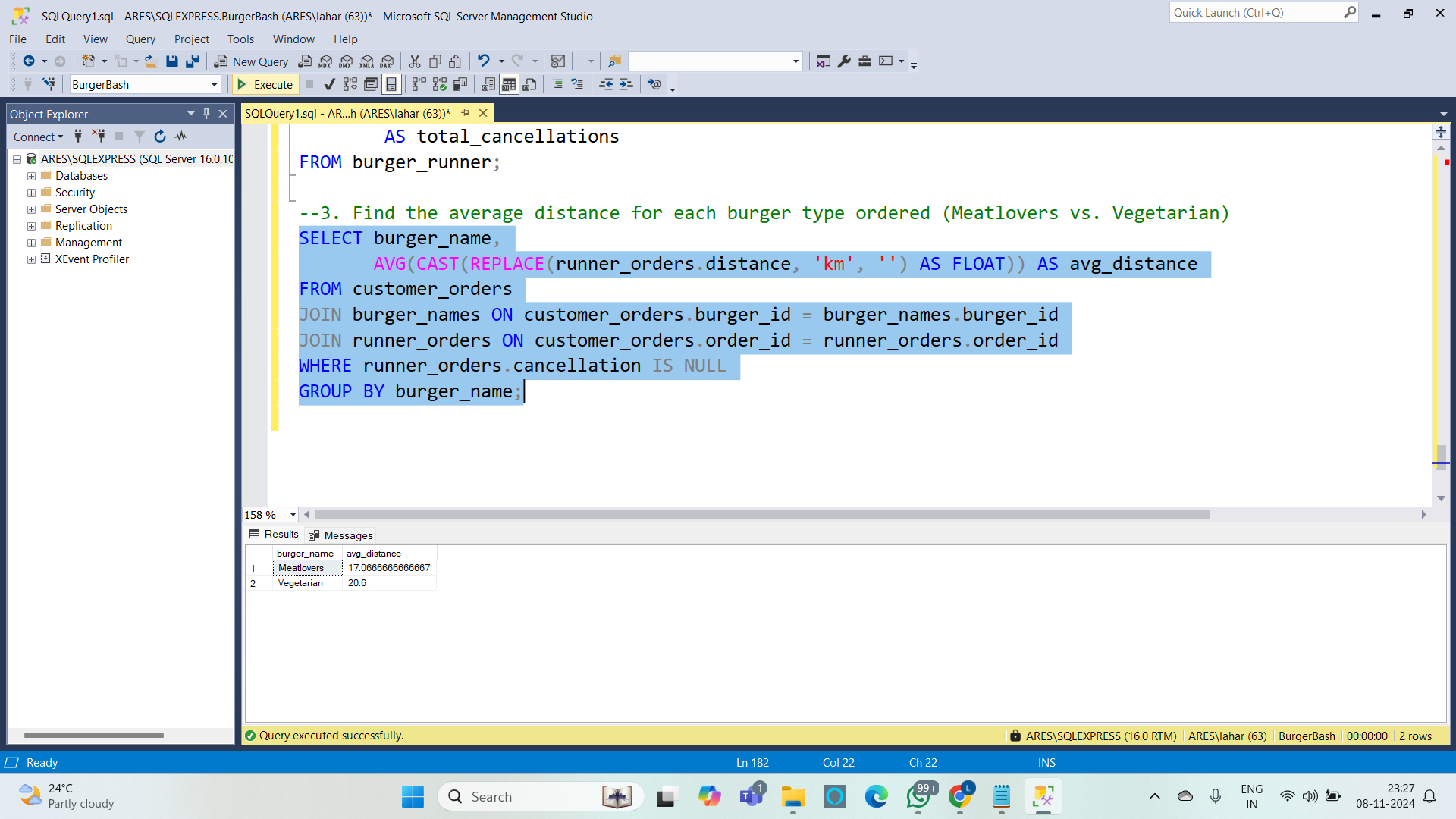
FROM customer\_orders

JOIN burger\_names ON customer\_orders.burger\_id = burger\_names.burger\_id

JOIN runner\_orders ON customer\_orders.order\_id = runner\_orders.order\_id

WHERE runner\_orders.cancellation IS NULL

GROUP BY burger\_name;



**4. List the customers who have ordered the most burgers in a single day.**

**Query :**

SELECT customer\_id, order\_date, MAX(burger\_count) AS max\_burgers\_ordered

FROM (

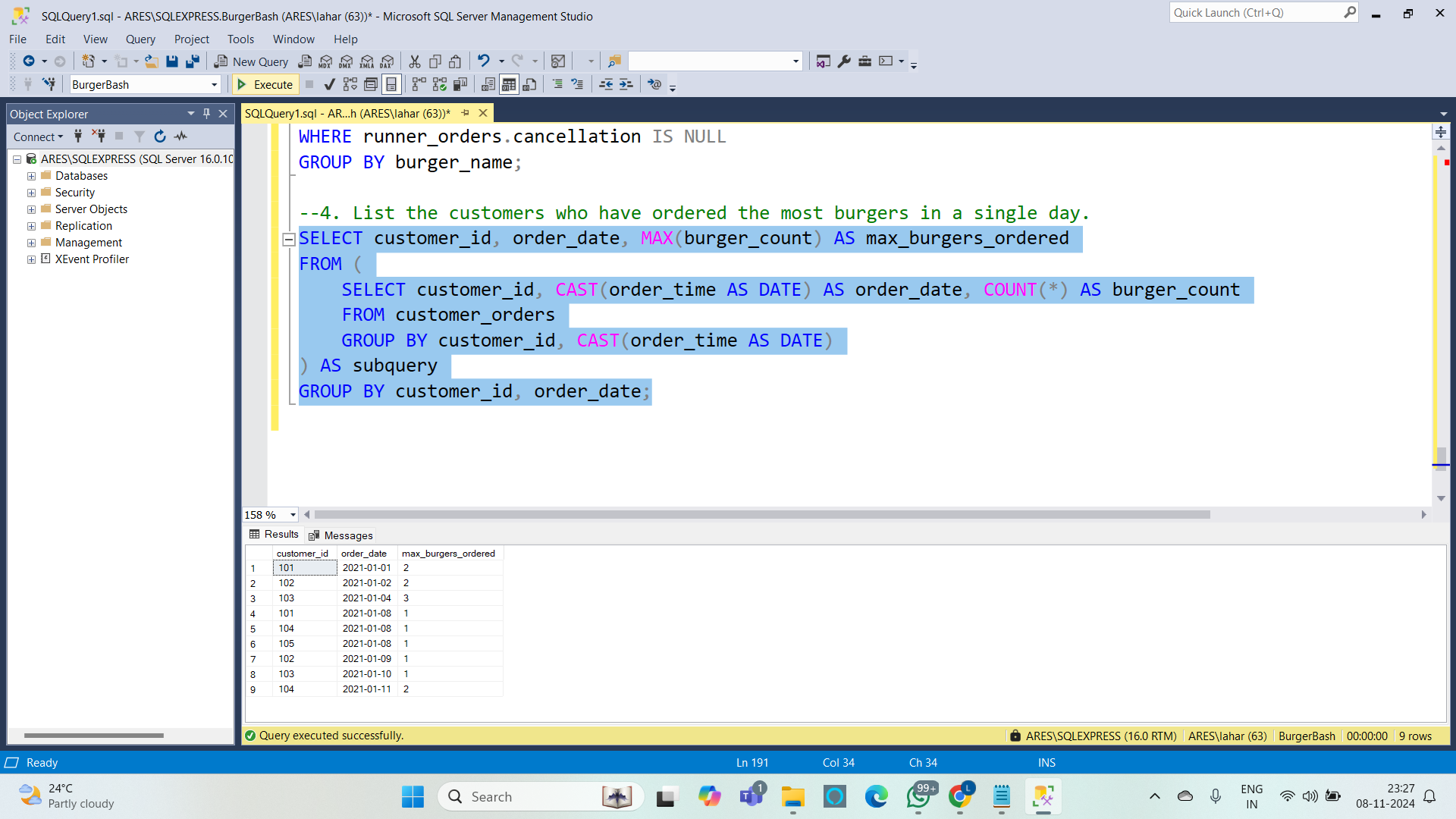
SELECT customer\_id, CAST(order\_time AS DATE) AS order\_date, COUNT(\*) AS burger\_count

FROM customer\_orders

GROUP BY customer\_id, CAST(order\_time AS DATE)

) AS subquery

GROUP BY customer\_id, order\_date;



**5. Find the total number of orders and burgers ordered for each runner**

**Query :**

SELECT runner\_id,

(SELECT COUNT(\*) FROM runner\_orders WHERE runner\_orders.runner\_id = burger\_runner.runner\_id)

AS total\_orders,

(SELECT COUNT(\*) FROM customer\_orders WHERE customer\_orders.order\_id

IN (SELECT order\_id FROM runner\_orders WHERE runner\_orders.runner\_id = burger\_runner.runner\_id))

AS total\_burgers\_ordered

FROM burger\_runner;

