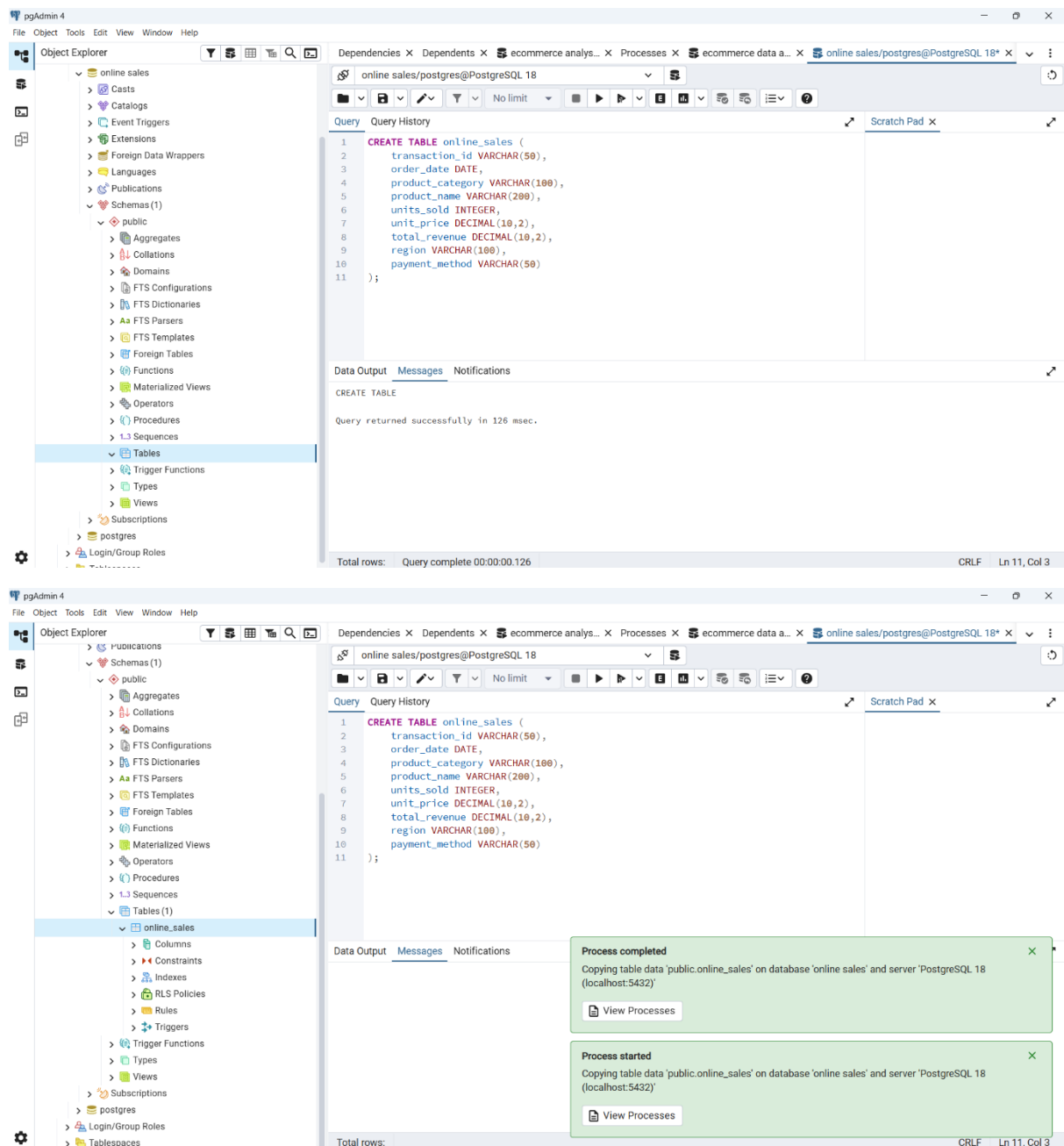


## Task- 6

### Sales Trend Analysis Using Aggregations

Created the online\_sales table with the required columns (transaction\_id, order\_date, product\_category, product\_name, units\_sold, unit\_price, total\_revenue, region, payment\_method). This matches the dataset structure from the Kaggle Online Sales Dataset.

```
CREATE TABLE online_sales (  
    transaction_id VARCHAR(50),  
    order_date DATE,  
    product_category VARCHAR(100),  
    product_name VARCHAR(200),  
    units_sold INTEGER,  
    unit_price DECIMAL(10,2),  
    total_revenue DECIMAL(10,2),  
    region VARCHAR(100),  
    payment_method VARCHAR(50)  
);
```



**Validation query to check revenue consistency (unit\_price \* units\_sold vs. total\_revenue).**

SELECT

```

EXTRACT(YEAR FROM order_date) AS year,
EXTRACT(MONTH FROM order_date) AS month,
SUM(total_revenue) AS total_revenue,
COUNT(DISTINCT transaction_id) AS order_volume

```

FROM

online\_sales

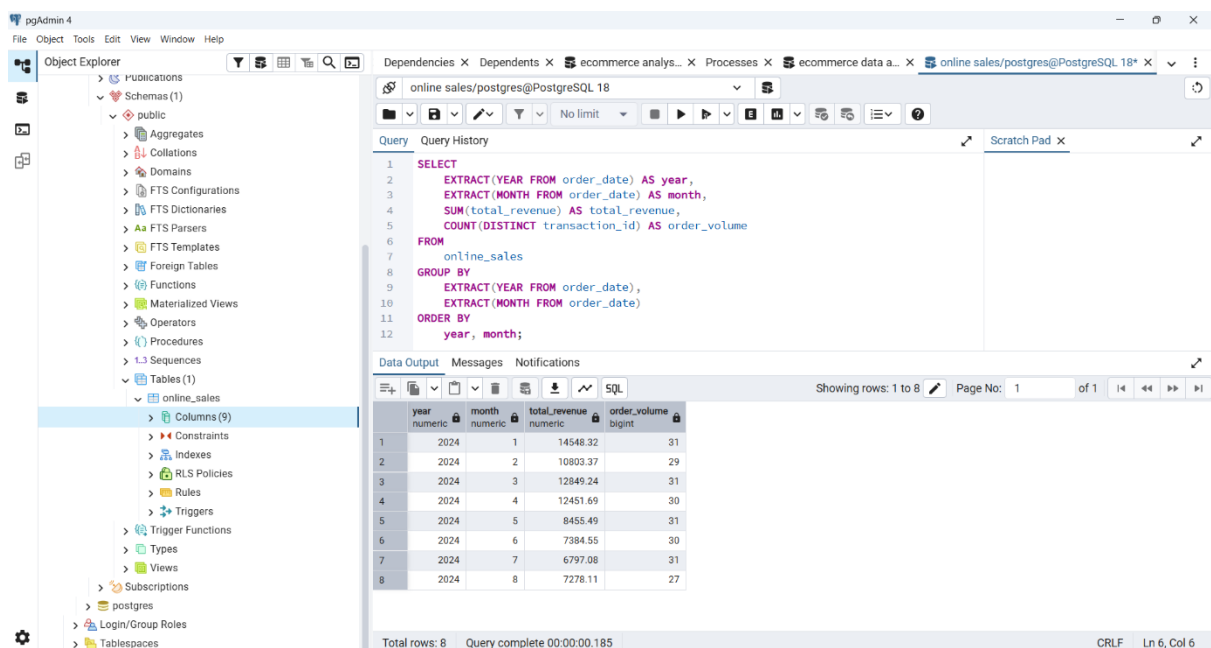
GROUP BY

EXTRACT(YEAR FROM order\_date),

EXTRACT(MONTH FROM order\_date)

ORDER BY

year, month;



The screenshot shows the pgAdmin 4 interface. On the left, the Object Explorer displays the database structure, including Schemas, Tables, and Columns. The 'online\_sales' table is selected, showing its columns: year, month, total\_revenue, and order\_volume. The main pane displays a SQL query in the Query Editor, which is executed. The results are shown in the Data Output pane, displaying a table with 8 rows and 4 columns: year, month, total\_revenue, and order\_volume. The query is as follows:

```
1 SELECT
2     EXTRACT(YEAR FROM order_date) AS year,
3     EXTRACT(MONTH FROM order_date) AS month,
4     SUM(total_revenue) AS total_revenue,
5     COUNT(DISTINCT transaction_id) AS order_volume
6 FROM
7     online_sales
8 GROUP BY
9     EXTRACT(YEAR FROM order_date),
10    EXTRACT(MONTH FROM order_date)
11 ORDER BY
12    year, month;
```

The Data Output pane shows the following results:

year	month	total_revenue	order_volume
2024	1	14548.32	31
2024	2	10803.37	29
2024	3	12849.24	31
2024	4	12451.69	30
2024	5	8455.49	31
2024	6	7384.55	30
2024	7	6797.08	31
2024	8	7278.11	27

Total rows: 8 Query complete 00:00:00.185

**Enhanced query with month names (TO\_CHAR(order\_date, 'Month')) and fixed GROUP BY/ORDER BY.**

SELECT

transaction\_id,

order\_date,

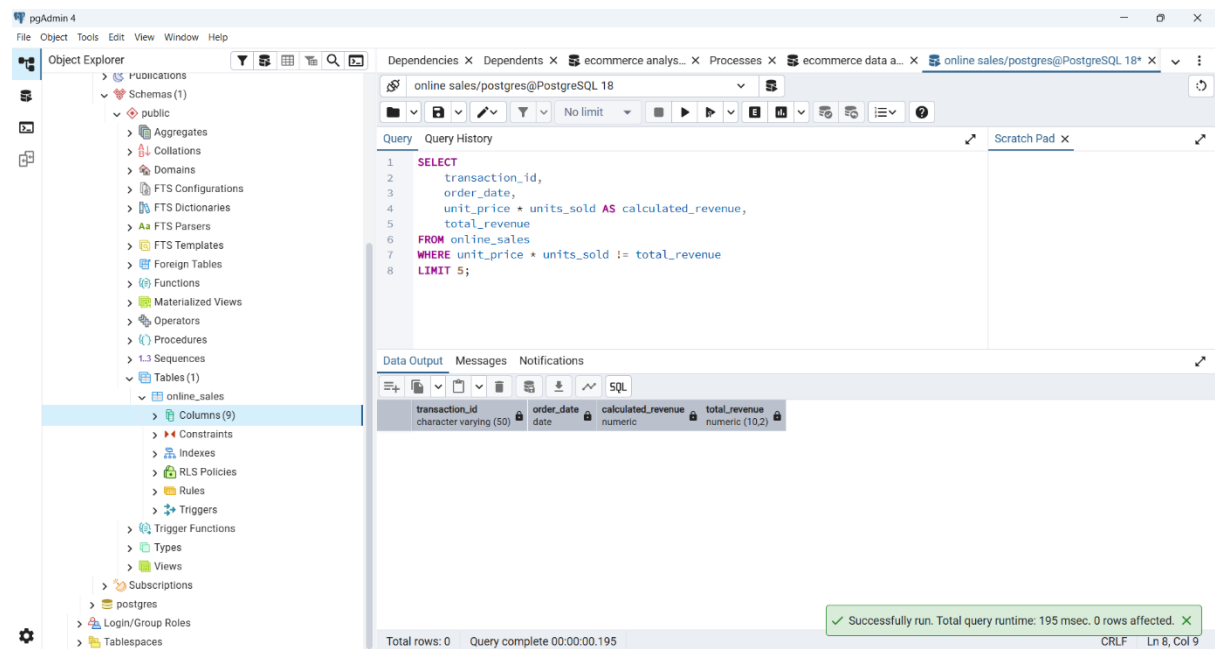
unit\_price \* units\_sold AS calculated\_revenue,

total\_revenue

FROM online\_sales

WHERE unit\_price \* units\_sold != total\_revenue

## LIMIT 5;



## Analysis by product category.

SELECT

EXTRACT(YEAR FROM order\_date) AS year,  
TO\_CHAR(order\_date, 'Month') AS month,  
SUM(total\_revenue) AS total\_revenue,  
COUNT(DISTINCT transaction\_id) AS order\_volume

FROM

online\_sales

GROUP BY

EXTRACT(YEAR FROM order\_date),  
TO\_CHAR(order\_date, 'Month'),  
EXTRACT(MONTH FROM order\_date) -- Add this to match ORDER BY

ORDER BY

year,  
EXTRACT(MONTH FROM order\_date);

The screenshot shows the pgAdmin 4 interface. On the left is the Object Explorer showing the database structure. The main pane displays a SQL query and its results. The query is as follows:

```

2  EXTRACT(YEAR FROM order_date) AS year,
3  TO_CHAR(order_date, 'Month') AS month,
4  SUM(total_revenue) AS total_revenue,
5  COUNT(DISTINCT transaction_id) AS order_volume
6  FROM
7  online_sales
8  GROUP BY
9  EXTRACT(YEAR FROM order_date),
10 TO_CHAR(order_date, 'Month'),
11 EXTRACT(MONTH FROM order_date)
12 ORDER BY
13 year,
14 EXTRACT(MONTH FROM order_date);

```

The results are shown in a table with the following data:

	year numeric	month text	total_revenue numeric	order_volume bigint
1	2024	Januar...	14548.32	31
2	2024	Februs...	10803.37	29
3	2024	March	12849.24	31
4	2024	April	12451.69	30
5	2024	May	8455.49	31
6	2024	June	7384.55	30
7	2024	July	6797.08	31
8	2024	August	7278.11	27

At the bottom, it shows "Total rows: 8" and "Query complete 00:00:00.085".

## Analysis by region (latest query), with validation.

SELECT

EXTRACT(YEAR FROM order\_date) AS year,  
EXTRACT(MONTH FROM order\_date) AS month,  
product\_category,  
SUM(total\_revenue) AS total\_revenue,  
COUNT(DISTINCT transaction\_id) AS order\_volume

FROM

online\_sales

GROUP BY

EXTRACT(YEAR FROM order\_date),  
EXTRACT(MONTH FROM order\_date),  
product\_category

ORDER BY

year, month, product\_category;

The screenshot shows the pgAdmin 4 interface. On the left is the Object Explorer showing the database structure. The main pane displays a SQL query and its results. The query is as follows:

```

SELECT
  EXTRACT(YEAR FROM order_date) AS year,
  EXTRACT(MONTH FROM order_date) AS month,
  region,
  SUM(total_revenue) AS total_revenue,
  COUNT(DISTINCT transaction_id) AS order_volume
FROM
  online_sales
GROUP BY
  EXTRACT(YEAR FROM order_date),
  EXTRACT(MONTH FROM order_date),
  region
ORDER BY
  year, month, region;

```

The results table shows 10 rows of data for the year 2024, grouped by month and region. The columns are: year, month, region, total\_revenue, and order\_volume.

year	month	region	total_revenue	order_volume
2024	1	Beauty Products	699.95	5
2024	1	Books	308.86	5
2024	1	Clothing	1789.84	5
2024	1	Electronics	7999.90	6
2024	1	Home Appliances	2169.94	5
2024	1	Sports	1579.83	5
2024	2	Beauty Products	331.98	5
2024	2	Books	422.91	5
2024	2	Clothing	1284.81	5
2024	2	Electronics	2899.88	4

Total rows: 48 Query complete 00:00:00.085

## Analysis by region (latest query), with validation.

SELECT

EXTRACT(YEAR FROM order\_date) AS year,  
EXTRACT(MONTH FROM order\_date) AS month,  
region,  
SUM(total\_revenue) AS total\_revenue,  
COUNT(DISTINCT transaction\_id) AS order\_volume

FROM

online\_sales

GROUP BY

EXTRACT(YEAR FROM order\_date),  
EXTRACT(MONTH FROM order\_date),  
region

ORDER BY

year, month, region;

The top screenshot shows a SQL query in pgAdmin 4 that aggregates data from the `online_sales` table by year and month. The query calculates total revenue, order volume, and average order value.

```

SELECT
  EXTRACT(YEAR FROM order_date) AS year,
  EXTRACT(MONTH FROM order_date) AS month,
  SUM(total_revenue) AS total_revenue,
  COUNT(DISTINCT transaction_id) AS order_volume,
  SUM(total_revenue) / COUNT(DISTINCT transaction_id) AS avg_order_value
FROM
  online_sales
GROUP BY
  EXTRACT(YEAR FROM order_date),
  EXTRACT(MONTH FROM order_date)
ORDER BY
  year, month;

```

The bottom screenshot shows the same query with an additional `region` column in the `GROUP BY` clause. The results are displayed in a table with 10 rows.

year	month	region	total_revenue	order_volume
2024	1	Asia	3369.67	10
2024	1	Europe	2869.89	10
2024	1	North America	8308.76	11
2024	2	Asia	4278.68	10
2024	2	Europe	3201.90	10
2024	2	North America	3322.79	9
2024	3	Asia	2619.60	10
2024	3	Europe	5476.91	10
2024	3	North America	4752.73	11
2024	4	Asia	3329.60	10

## Combined region + product category

SELECT

transaction\_id,

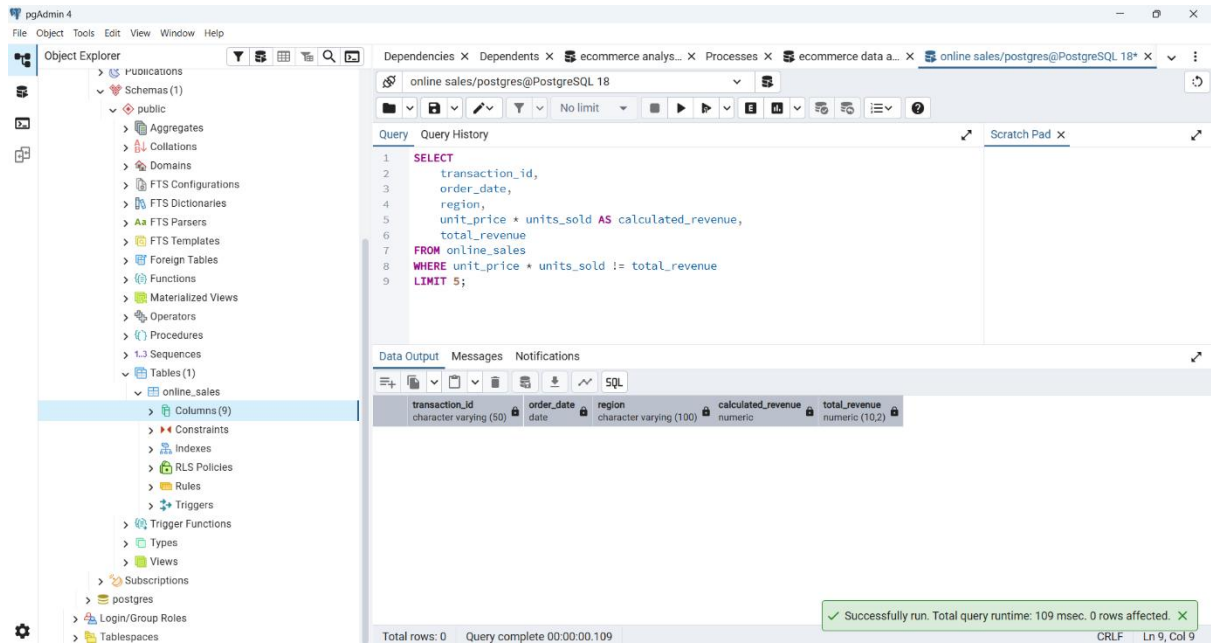
order\_date,

region,

unit\_price \* units\_sold AS calculated\_revenue,

total\_revenue

```
FROM online_sales  
WHERE unit_price * units_sold != total_revenue  
LIMIT 5;
```



**By region + payment method.**

```
SELECT  
  
    EXTRACT(YEAR FROM order_date) AS year,  
    TO_CHAR(order_date, 'Month') AS month,  
    region,  
    product_category,  
    SUM(total_revenue) AS total_revenue,  
    COUNT(DISTINCT transaction_id) AS order_volume  
FROM  
    online_sales  
GROUP BY  
    EXTRACT(YEAR FROM order_date),  
    TO_CHAR(order_date, 'Month'),  
    EXTRACT(MONTH FROM order_date),
```



region,

product\_category

ORDER BY

year, EXTRACT(MONTH FROM order\_date), region, product\_category;

The screenshot shows the pgAdmin 4 interface. On the left, the 'Object Explorer' pane displays the database structure, with 'online\_sales' selected under 'Tables (1)'. The main pane shows a SQL query in the 'Query History' tab. The query is as follows:

```
product_category,  
SUM(total_revenue) AS total_revenue,  
COUNT(DISTINCT transaction_id) AS order_volume  
FROM  
online_sales  
GROUP BY  
EXTRACT(YEAR FROM order_date),  
TO_CHAR(order_date, 'Month'),  
EXTRACT(MONTH FROM order_date),  
region,  
product_category  
ORDER BY  
year, EXTRACT(MONTH FROM order_date), region, product_category;
```

Below the query, the 'Data Output' pane shows the results of the query. The results are displayed in a table with 10 rows and 6 columns: year, month, region, product\_category, total\_revenue, and order\_volume. The table shows data for January and February 2024 across different regions and product categories.

year	month	region	product_category	total_revenue	order_volume
2024	Januar...	Asia	Clothing	1789.84	5
2024	Januar...	Asia	Sports	1579.83	5
2024	Januar...	Europe	Beauty Products	699.95	5
2024	Januar...	Europe	Home Appliances	2169.94	5
2024	Januar...	North America	Books	308.86	5
2024	Januar...	North America	Electronics	7999.90	6
2024	Februa...	Asia	Clothing	1284.81	5
2024	Februa...	Asia	Sports	2993.87	5
2024	Februa...	Europe	Beauty Products	331.98	5
2024	Februa...	Europe	Home Appliances	296	5

The status bar at the bottom indicates 'Total rows: 48', 'Query complete 00:00:00.145', and 'Successfully run. Total query runtime: 145 msec. 48 rows affected.'

**Added average order value.**

SELECT

EXTRACT(YEAR FROM order\_date) AS year,

TO\_CHAR(order\_date, 'Month') AS month,

region,

payment\_method,

SUM(total\_revenue) AS total\_revenue,

COUNT(DISTINCT transaction\_id) AS order\_volume

FROM

online\_sales

GROUP BY

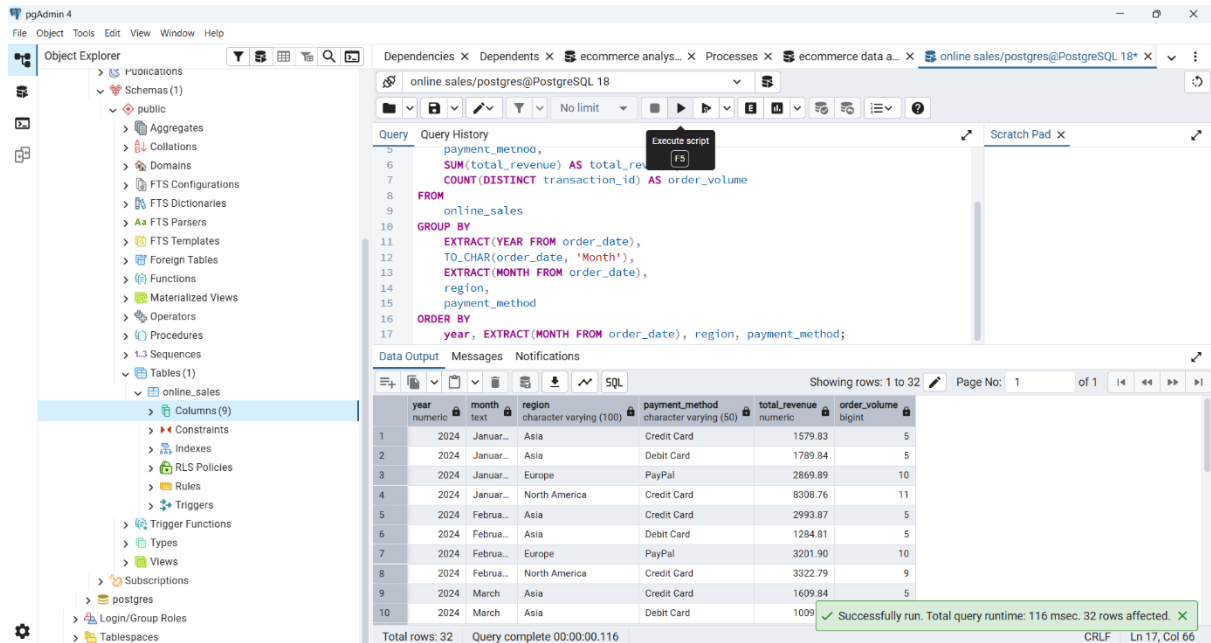
EXTRACT(YEAR FROM order\_date),

TO\_CHAR(order\_date, 'Month'),

```

EXTRACT(MONTH FROM order_date),
region,
payment_method
ORDER BY
year, EXTRACT(MONTH FROM order_date), region, payment_method;

```



## Advanced CTE for year-over-year revenue growth percent by region

```

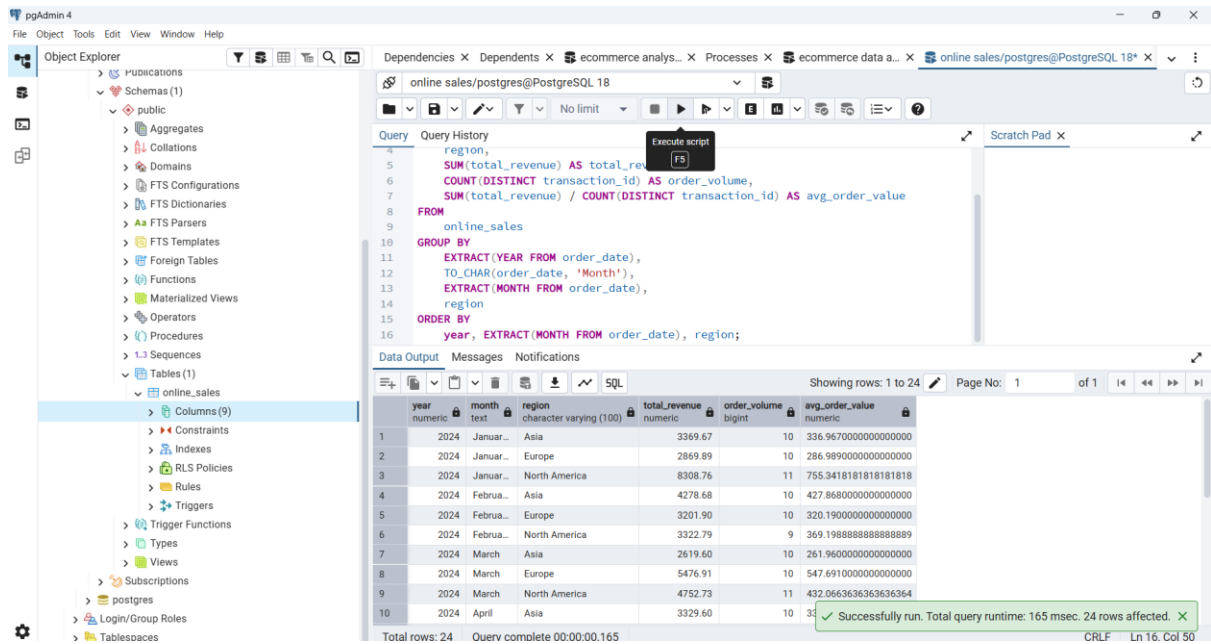
SELECT
    EXTRACT(YEAR FROM order_date) AS year,
    TO_CHAR(order_date, 'Month') AS month,
    region,
    SUM(total_revenue) AS total_revenue,
    COUNT(DISTINCT transaction_id) AS order_volume,
    SUM(total_revenue) / COUNT(DISTINCT transaction_id) AS
avg_order_value
FROM
    online_sales
GROUP BY

```

```

EXTRACT(YEAR FROM order_date),
TO_CHAR(order_date, 'Month'),
EXTRACT(MONTH FROM order_date),
region
ORDER BY
year, EXTRACT(MONTH FROM order_date), region;

```



```

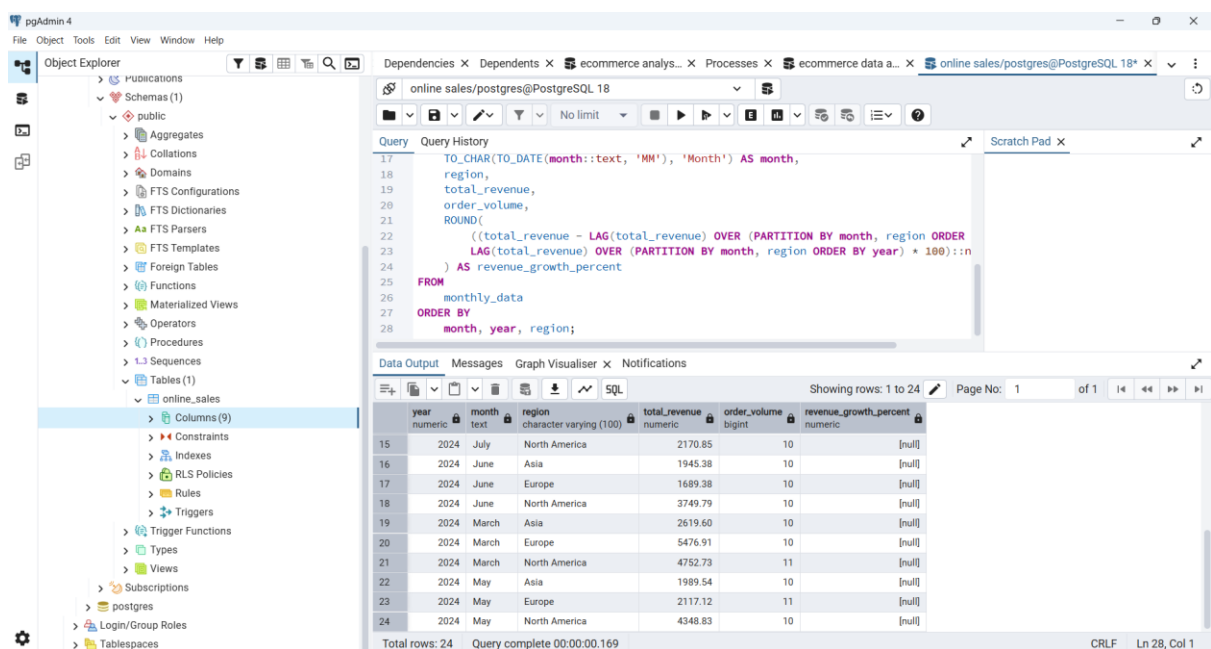
WITH monthly_data AS (
SELECT
    EXTRACT(YEAR FROM order_date) AS year,
    EXTRACT(MONTH FROM order_date) AS month,
    region,
    SUM(total_revenue) AS total_revenue,
    COUNT(DISTINCT transaction_id) AS order_volume
FROM
    online_sales
GROUP BY
    EXTRACT(YEAR FROM order_date),
    EXTRACT(MONTH FROM order_date),

```

```

        region
    )
SELECT
    year,
    TO_CHAR(TO_DATE(month::text, 'MM'), 'Month') AS month,
    region,
    total_revenue,
    order_volume,
    ROUND(
        ((total_revenue - LAG(total_revenue) OVER (PARTITION BY month,
        region ORDER BY year)) /
        LAG(total_revenue) OVER (PARTITION BY month, region ORDER BY
        year) * 100)::numeric, 2
    ) AS revenue_growth_percent
FROM
    monthly_data
ORDER BY
    month, year, region;

```



The screenshot shows the pgAdmin 4 interface. On the left is the Object Explorer showing the database structure. The main pane displays a SQL query and its results. The query is as follows:

```

SELECT
    year,
    TO_CHAR(TO_DATE(month::text, 'MM'), 'Month') AS month,
    region,
    total_revenue,
    order_volume,
    ROUND(
        ((total_revenue - LAG(total_revenue) OVER (PARTITION BY month,
        region ORDER BY year)) /
        LAG(total_revenue) OVER (PARTITION BY month, region ORDER BY
        year) * 100)::numeric, 2
    ) AS revenue_growth_percent
FROM
    monthly_data
ORDER BY
    month, year, region;

```

The results table shows 24 rows of data. The columns are: year, month, region, total\_revenue, order\_volume, and revenue\_growth\_percent. The data is ordered by month, year, and region.

year	month	region	total_revenue	order_volume	revenue_growth_percent
2024	July	North America	2170.85	10	[null]
2024	June	Asia	1945.38	10	[null]
2024	June	Europe	1689.38	10	[null]
2024	June	North America	3749.79	10	[null]
2024	March	Asia	2619.60	10	[null]
2024	March	Europe	5476.91	10	[null]
2024	March	North America	4752.73	11	[null]
2024	May	Asia	1989.54	10	[null]
2024	May	Europe	2117.12	11	[null]
2024	May	North America	4348.83	10	[null]