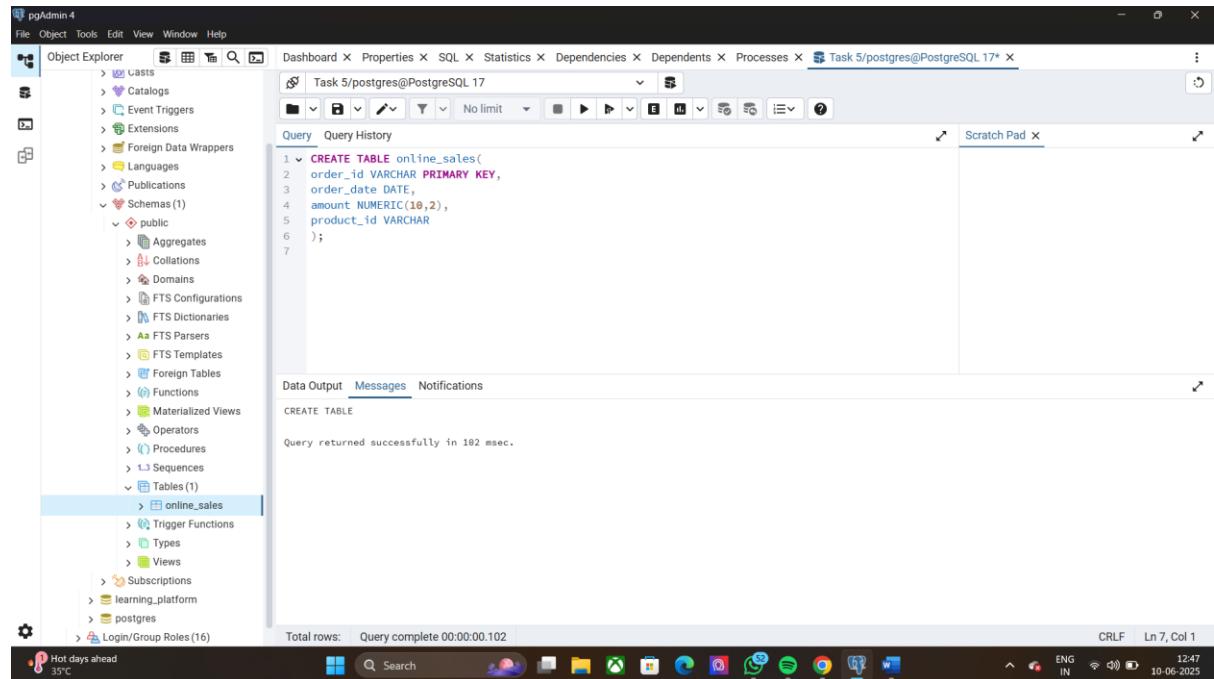


✓ PostgreSQL Table Creation

```
CREATE TABLE online_sales (
    order_id VARCHAR PRIMARY KEY,
    order_date DATE,
    amount NUMERIC(10, 2),
    product_id VARCHAR
);
```

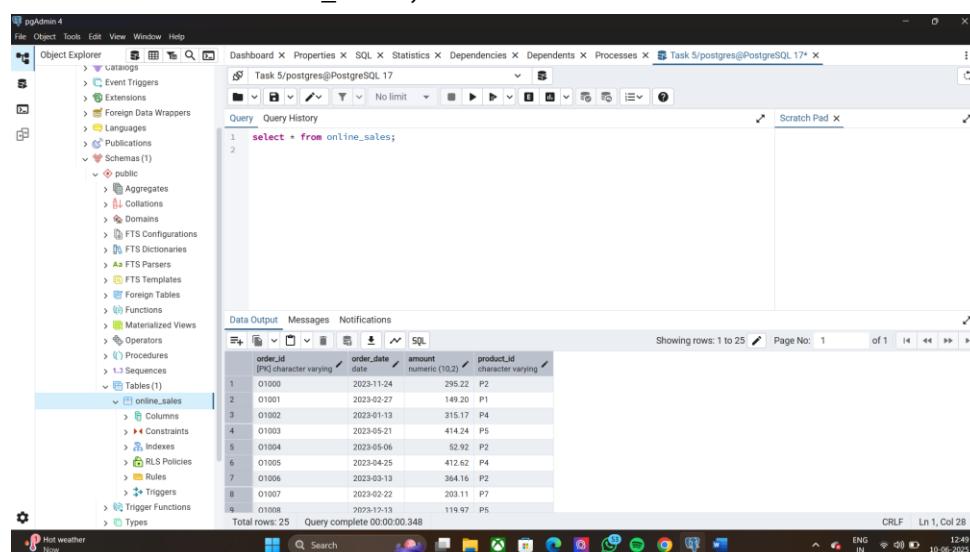


The screenshot shows the pgAdmin 4 interface. In the Object Explorer, under the 'Tables' section of the 'public' schema, there is a new entry for 'online_sales'. The main query window contains the SQL code for creating the table. Below the code, the message 'Query returned successfully in 102 msec.' is displayed. The status bar at the bottom right shows the date and time as 10-06-2025.

```
CREATE TABLE online_sales(
    order_id VARCHAR PRIMARY KEY,
    order_date DATE,
    amount NUMERIC(10, 2),
    product_id VARCHAR
);
```

1. View the Dataset

```
SELECT * FROM online_sales;
```



The screenshot shows the pgAdmin 4 interface with the previous query results. The Data Output tab displays the contents of the 'online_sales' table. The table has four columns: 'order_id' (PK character varying), 'order_date' (date), 'amount' (numeric (10,2)), and 'product_id' (character varying). There are 25 rows of data. The status bar at the bottom right shows the date and time as 10-06-2025.

	order_id	order_date	amount	product_id
1	01000	2023-11-24	295.22	P2
2	01001	2023-02-27	149.20	P1
3	01002	2023-01-13	315.17	P4
4	01003	2023-05-21	414.24	P5
5	01004	2023-05-06	92.92	P2
6	01005	2023-04-25	412.62	P4
7	01006	2023-03-13	364.16	P2
8	01007	2023-02-22	203.11	P7
9	01008	2023-12-13	119.97	P5
Total rows:	25			

2. Extract Year & Month

SELECT

```
order_id,  
order_date,  
EXTRACT(YEAR FROM order_date) AS year,  
EXTRACT(MONTH FROM order_date) AS month  
FROM online_sales;
```

The screenshot shows the pgAdmin 4 interface. On the left is the Object Explorer tree, which includes Catalogs, Event Triggers, Extensions, Foreign Data Wrappers, Languages, Publications, Schemas (1), public (Aggregates, Collations, Domains, FTS Configurations, FTS Dictionaries, FTS Parsers, FTS Templates, Foreign Tables, Functions, Materialized Views, Operators, Procedures, Sequences, Tables (1), and online_sales (Columns, Constraints, Indexes, RLS Policies, Rules, Triggers, Types)). The main window displays the SQL query:

```
1 SELECT  
2     order_id,  
3     order_date,  
4     EXTRACT(YEAR FROM order_date) AS year,  
5     EXTRACT(MONTH FROM order_date) AS month  
6 FROM online_sales;  
7
```

The Data Output tab shows the results of the query:

order_id	order_date	year	month
1	2023-11-24	2023	11
2	2023-02-27	2023	2
3	2023-01-13	2023	1
4	2023-05-21	2023	5
5	2023-05-06	2023	5
6	2023-04-25	2023	4
7	2023-03-13	2023	3
8	2023-02-22	2023	2
9	2023-12-13	2023	12

Total rows: 25 Query complete 00:00:00.172

3. Monthly Revenue and Order Volume

SELECT

```
EXTRACT(YEAR FROM order_date) AS year,  
EXTRACT(MONTH FROM order_date) AS month,  
SUM(amount) AS total_revenue,  
COUNT(DISTINCT order_id) AS order_volume  
FROM online_sales  
GROUP BY year, month  
ORDER BY year, month;
```

The screenshot shows the pgAdmin 4 interface. In the Object Explorer, under Schemas (1) > public > Tables (1) > online_sales, the table is selected. The main window displays a SQL query in the Query tab:

```

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

```

The Data Output tab shows the results of the query:

	year	month	total_revenue	order_volume
1	2023	1	1366.64	4
2	2023	2	966.99	4
3	2023	3	364.16	1
4	2023	4	1410.63	4
5	2023	5	467.16	2
6	2023	8	93.52	1
7	2023	9	487.90	1
8	2023	10	995.72	3
9	2023	11	953.33	3

Total rows: 10 Query complete 00:00:00.103

4. Filtered by Date Range (e.g., July–December 2023)

SELECT

```

EXTRACT(YEAR FROM order_date) AS year,
EXTRACT(MONTH FROM order_date) AS month,
SUM(amount) AS total_revenue,
COUNT(DISTINCT order_id) AS order_volume
FROM online_sales
WHERE order_date BETWEEN '2023-07-01' AND '2023-12-31'
GROUP BY year, month
ORDER BY year, month;

```

pgAdmin 4

File Object Tools Edit View Window Help

Object Explorer Dashboard Properties SQL Statistics Dependencies Dependents Processes Task 5/postgres@PostgreSQL 17*

Schemas (1) public Aggregates Collations Domains FTS Configurations FTS Dictionaries FTS Parsers FTS Templates Foreign Tables Functions Materialized Views Operators Procedures Sequences Tables (1)

online_sales Columns Constraints Indexes RLS Policies Rules Triggers Trigger Functions Types Views Subscriptions learning_platform postgres Login/Group Roles Tablespaces

Hot days ahead 36°C

Task 5/postgres@PostgreSQL 17*

Query History Execute script F5

```

1 SELECT
2   EXTRACT(YEAR FROM order_date) AS year,
3   EXTRACT(MONTH FROM order_date) AS month,
4   SUM(amount) AS total_revenue,
5   COUNT(DISTINCT order_id) AS order_volume
6   FROM online_sales
7   WHERE order_date BETWEEN '2023-07-01' AND '2023-12-31'
8   GROUP BY year, month
9   ORDER BY year, month;
10

```

Data Output Messages Notifications SQL

	year	month	total_revenue	order_volume
1	2023	8	93.52	1
2	2023	9	487.90	1
3	2023	10	995.72	3
4	2023	11	953.33	3
5	2023	12	429.78	2

Total rows: 5 Query complete 00:00:00.123

Successfully run. Total query runtime: 123 msec. 5 rows affected.

CRLF Ln 10, Col 1

5. Daily Revenue Trend (Optional Granularity)

SELECT

order_date,

SUM(amount) AS daily_revenue,

COUNT(DISTINCT order_id) AS daily_orders

FROM online_sales

GROUP BY order_date

ORDER BY order_date;

```

SELECT
  order_date,
  SUM(amount) AS daily_revenue,
  COUNT(DISTINCT order_id) AS daily_orders
FROM online_sales
GROUP BY order_date
ORDER BY order_date;

```

	order_date	daily_revenue	daily_orders
1	2023-01-13	315.17	1
2	2023-01-14	298.42	1
3	2023-01-16	321.68	1
4	2023-01-17	431.37	1
5	2023-02-14	201.47	1
6	2023-02-17	413.21	1
7	2023-02-22	203.11	1
8	2023-02-27	149.20	1
9	2023-03-13	364.16	1

Total rows: 25 Query complete 00:00:00.370

Successfully run. Total query runtime: 370 msec. 25 rows affected.

6. Monthly Average Order Value (AOV)

```

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

```

The screenshot shows the pgAdmin 4 interface. The left sidebar is titled 'Object Explorer' and lists various database objects under 'Schemas (1)' and 'Tables (1)'. The 'Tables (1)' section has 'online_sales' selected. The main area is a 'Query' editor with the following SQL code:

```
1  SELECT
2      EXTRACT(YEAR FROM order_date) AS year,
3      EXTRACT(MONTH FROM order_date) AS month,
4      SUM(amount) / COUNT(DISTINCT order_id) AS avg_order_value
5  FROM online_sales
6  GROUP BY year, month
7  ORDER BY year, month;
```

The results of the query are displayed in a table below:

	year	month	avg_order_value
1	2023	1	341.66000000000000
2	2023	2	241.74750000000000
3	2023	3	364.16000000000000
4	2023	4	352.65750000000000
5	2023	5	233.58000000000000
6	2023	8	93.52000000000000
7	2023	9	487.90000000000000
8	2023	10	331.90666666666667
9	2023	11	317.77666666666667

Total rows: 10 Query complete 00:00:00.088

7. Top 5 Months by Revenue

SELECT

```
EXTRACT(YEAR FROM order_date) AS year,  
EXTRACT(MONTH FROM order_date) AS month,  
SUM(amount) AS total_revenue  
FROM online_sales  
GROUP BY year, month  
ORDER BY total_revenue DESC  
LIMIT 5;
```

```

SELECT
    EXTRACT(YEAR FROM order_date) AS year,
    EXTRACT(MONTH FROM order_date) AS month,
    SUM(amount) AS total_revenue
FROM online_sales
GROUP BY year, month
ORDER BY total_revenue DESC
LIMIT 5;

```

	year	month	total_revenue
1	2023	4	1410.63
2	2023	1	1366.64
3	2023	10	995.72
4	2023	2	966.99
5	2023	11	953.33

Total rows: 5 Query complete 00:00:00.128

Successfully run. Total query runtime: 128 msec. 5 rows affected.

8. Number of Orders Per Product Per Month

```

SELECT
    product_id,
    EXTRACT(YEAR FROM order_date) AS year,
    EXTRACT(MONTH FROM order_date) AS month,
    COUNT(DISTINCT order_id) AS order_count
FROM online_sales
GROUP BY product_id, year, month
ORDER BY year, month, product_id;

```

```

SELECT
    product_id,
    EXTRACT(YEAR FROM order_date) AS year,
    EXTRACT(MONTH FROM order_date) AS month,
    COUNT(DISTINCT order_id) AS order_count
FROM online_sales
GROUP BY product_id, year, month
ORDER BY year, month, product_id;

```

	product_id	year	month	order_count
1	P4	2023	1	3
2	P6	2023	1	1
3	P1	2023	2	1
4	P5	2023	2	1
5	P6	2023	2	1
6	P7	2023	2	1
7	P2	2023	3	1
8	P10	2023	4	1
9	P2	2023	4	1

Total rows: 21 Query complete 00:00:00.138

Successfully run. Total query runtime: 138 msec. 21 rows affected.

9. Total Revenue Per Product (Yearly Aggregation)

SELECT

product_id,

EXTRACT(YEAR FROM order_date) AS year,

SUM(amount) AS total_revenue

FROM online_sales

GROUP BY product_id, year

ORDER BY product_id, year;

```

SELECT
    product_id,
    EXTRACT(YEAR FROM order_date) AS year,
    SUM(amount) AS total_revenue
FROM online_sales
GROUP BY product_id, year
ORDER BY product_id, year;

```

	product_id	year	total_revenue
1	P1	2023	149.20
2	P10	2023	291.30
3	P2	2023	1090.68
4	P3	2023	1002.87
5	P4	2023	1347.89
6	P5	2023	1257.23
7	P6	2023	726.36
8	P7	2023	640.88
9	P8	2023	809.08
	Total rows: 10	Query complete 00:00:00.101	

Successfully run. Total query runtime: 101 msec. 10 rows affected.

10. Revenue Summary by Quarter

```

SELECT
    EXTRACT(YEAR FROM order_date) AS year,
    CEIL(EXTRACT(MONTH FROM order_date) / 3.0) AS quarter,
    SUM(amount) AS total_revenue,
    COUNT(DISTINCT order_id) AS order_volume
FROM online_sales
GROUP BY year, quarter
ORDER BY year, quarter;

```

pgAdmin 4

File Object Tools Edit View Window Help

Object Explorer Dashboard Properties SQL Statistics Dependencies Dependents Processes Task 5/postgres@PostgreSQL 17*

Schemas (1) Task 5/postgres@PostgreSQL 17*

public Aggregates Collations Domains FTS Configurations FTS Dictionaries FTS Parsers FTS Templates Foreign Tables Functions Materialized Views Operators Procedures Sequences Tables (1)

online_sales Columns Constraints Indexes RLS Policies Rules Triggers Trigger Functions Types Views Subscriptions learning_platform postgres Login/Group Roles Tablespaces

Task 5/postgres@PostgreSQL 17*

Query History Scratch Pad

```
1 SELECT
2   EXTRACT(YEAR FROM order_date) AS year,
3   CEIL(EXTRACT(MONTH FROM order_date) / 3.0) AS quarter,
4   SUM(amount) AS total_revenue,
5   COUNT(DISTINCT order_id) AS order_volume
6   FROM online_sales
7   GROUP BY year, quarter
8   ORDER BY year, quarter;
```

Data Output Messages Notifications

	year	quarter	total_revenue	order_volume
1	2023	1	2697.79	9
2	2023	2	1877.79	6
3	2023	3	581.42	2
4	2023	4	2378.83	8

Total rows: 4 Query complete 00:00:00.091

Successfully run. Total query runtime: 91 msec. 4 rows affected.

CRLF Ln 9, Col 1

Rainy days ahead 36°C

Search

ENG IN 13:07 10-06-2025

The screenshot shows the pgAdmin 4 interface with a query results window. The query selects data from the 'online_sales' table, grouped by year and quarter, calculating total revenue and order volume. The results are displayed in a table with four rows. A message at the bottom indicates the query was successfully run. The pgAdmin interface includes a sidebar with schema and table details, and a bottom status bar showing system information like date and time.