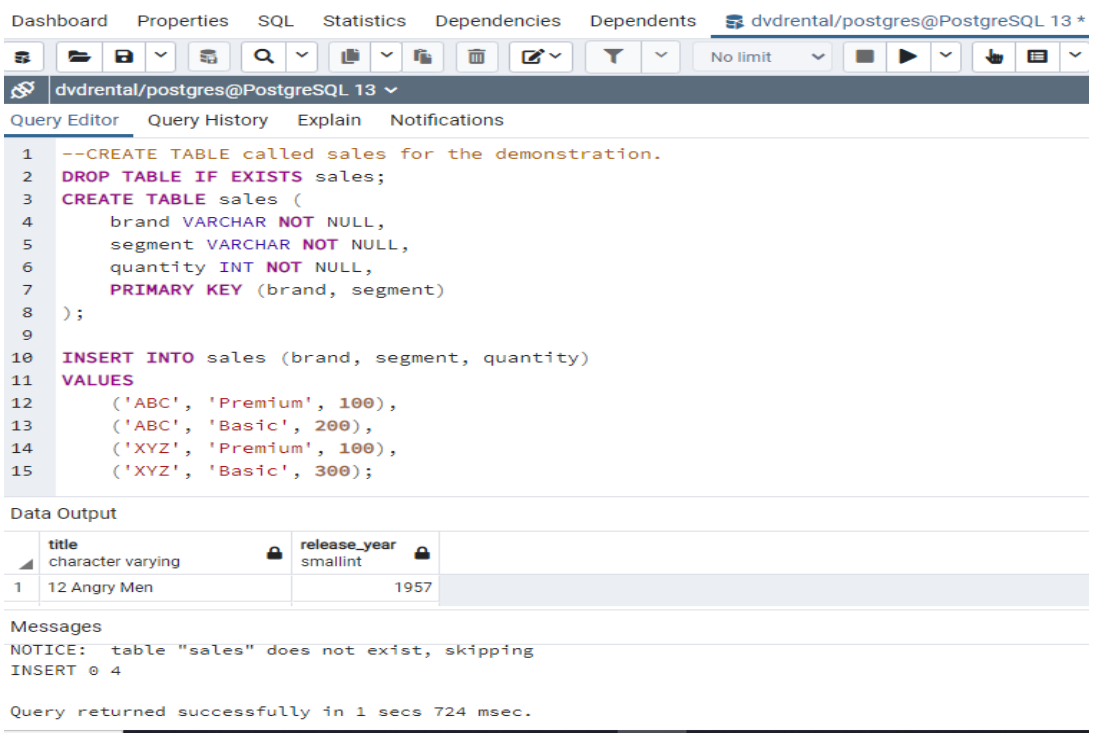
POSTGRESQL GROUPING SETS



```
--CREATE TABLE called sales for the demonstration.
DROP TABLE IF EXISTS sales;
CREATE TABLE sales (
    brand VARCHAR NOT NULL,
    segment VARCHAR NOT NULL,
    quantity INT NOT NULL,
    PRIMARY KEY (brand, segment)
);

INSERT INTO sales (brand, segment, quantity)
VALUES
    ('ABC', 'Premium', 100),
    ('ABC', 'Basic', 200),
    ('XYZ', 'Premium', 100),
    ('XYZ', 'Basic', 300);
```

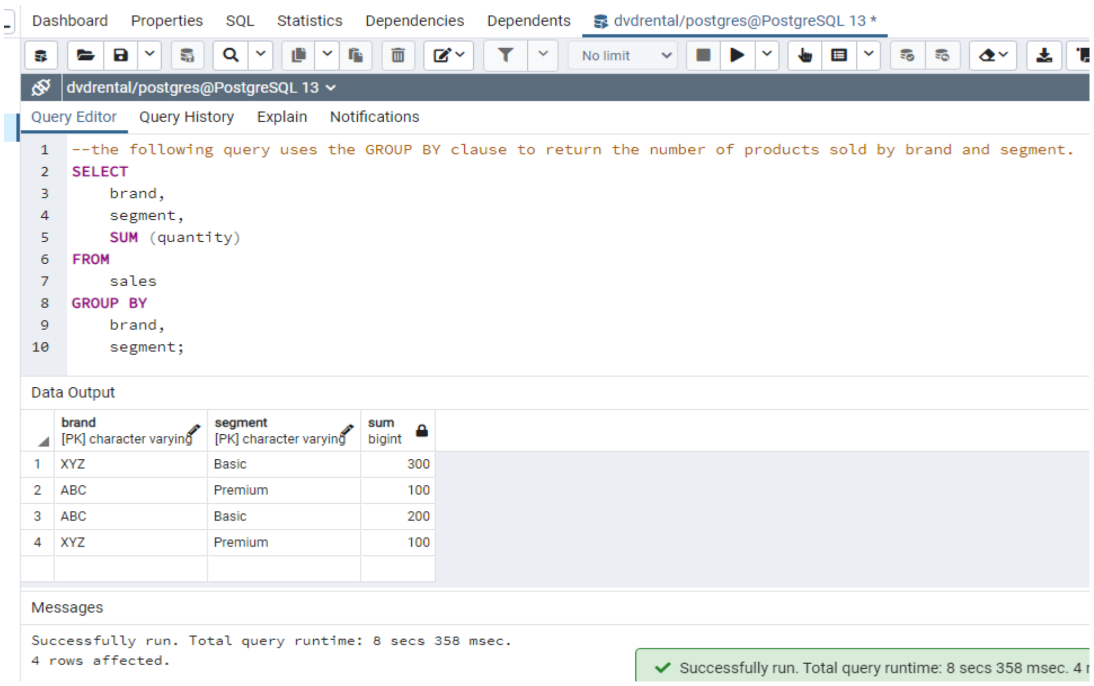
Data Output

	title character varying	release_year smallint
1	12 Angry Men	1957

Messages

```
NOTICE:  table "sales" does not exist, skipping
INSERT 0 4

Query returned successfully in 1 secs 724 msec.
```



```
--the following query uses the GROUP BY clause to return the number of products sold by brand and segment.
SELECT
    brand,
    segment,
    SUM (quantity)
FROM
    sales
GROUP BY
    brand,
    segment;
```

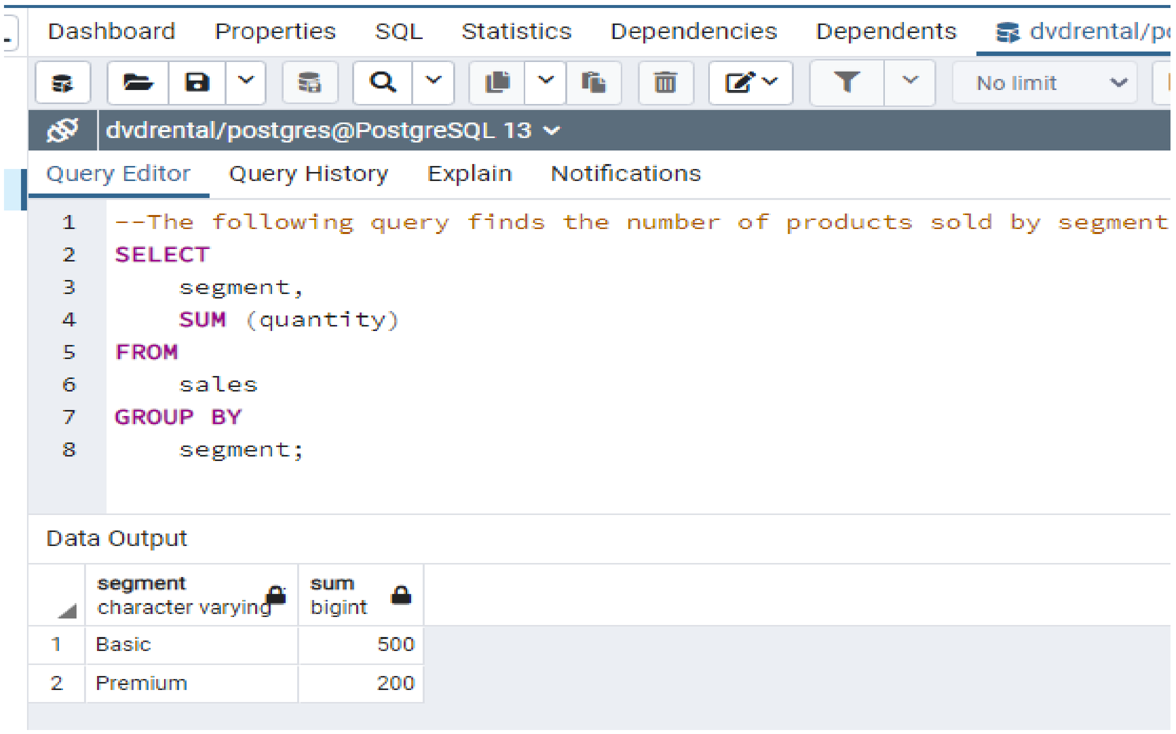
Data Output

	brand [PK] character varying	segment [PK] character varying	sum bigint
1	XYZ	Basic	300
2	ABC	Premium	100
3	ABC	Basic	200
4	XYZ	Premium	100

Messages

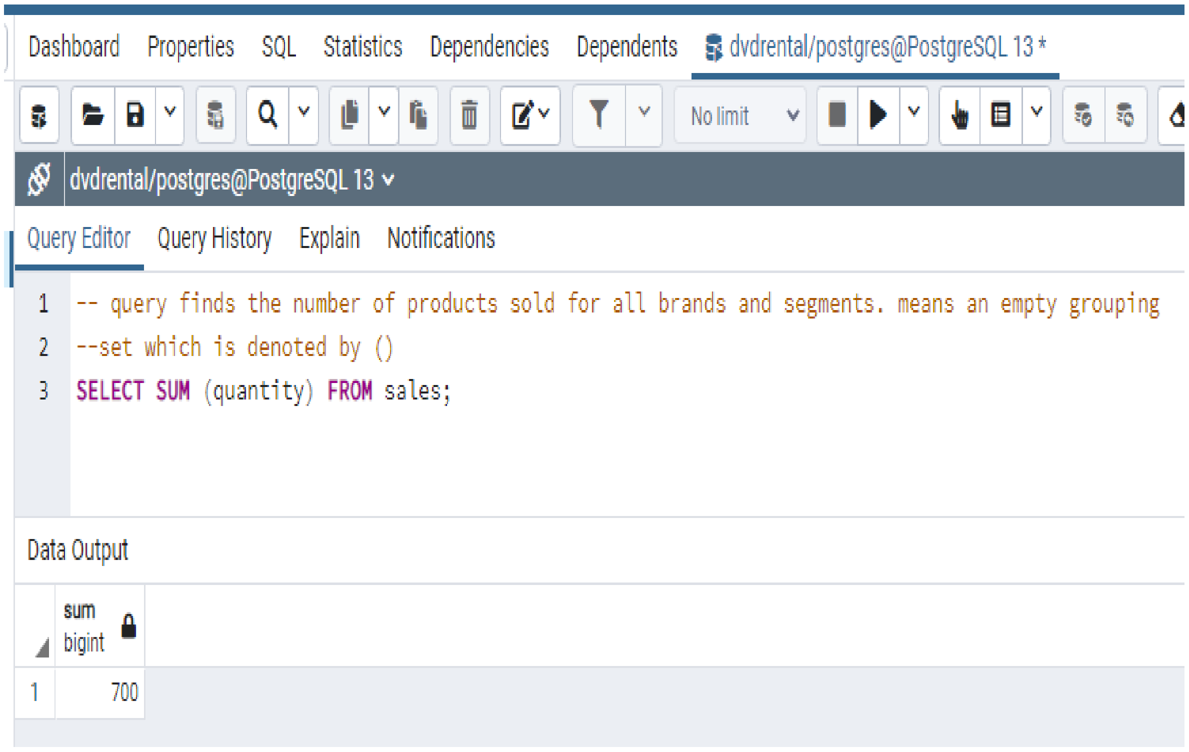
```
Successfully run. Total query runtime: 8 secs 358 msec.
4 rows affected.
```

Successfully run. Total query runtime: 8 secs 358 msec. 4 r



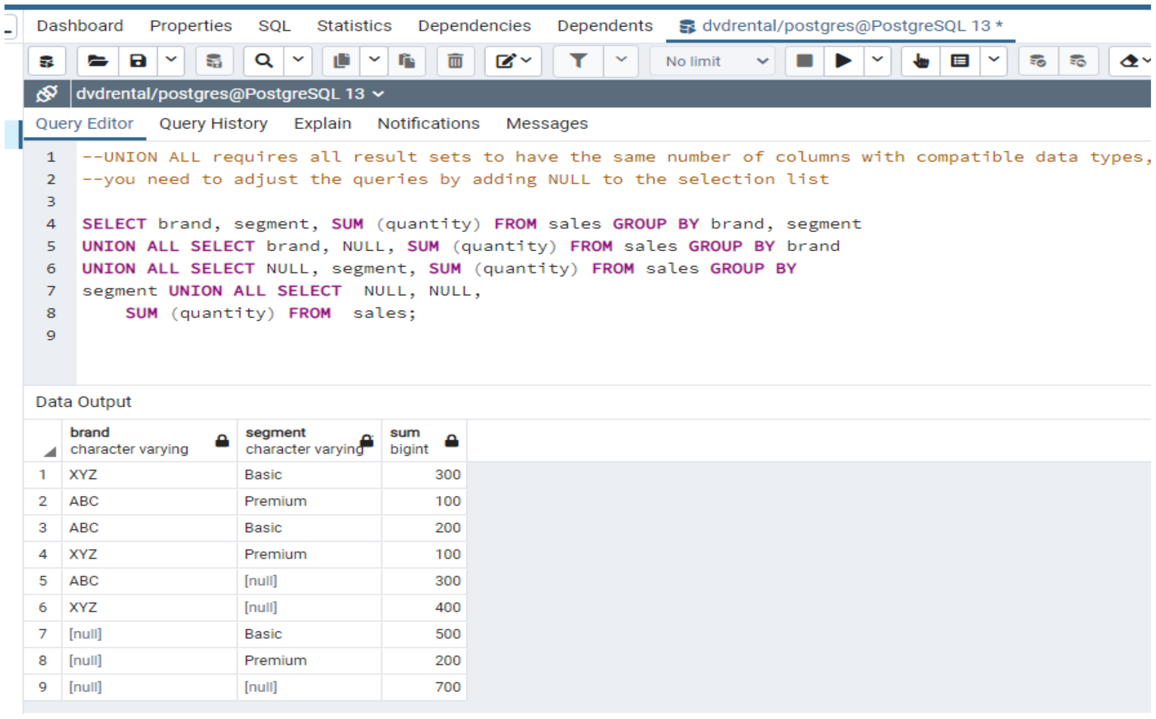
```
--The following query finds the number of products sold by segment
SELECT
    segment,
    SUM (quantity)
FROM
    sales
GROUP BY
    segment;
```

```
  segment               sum
  character varying     bigint
1 Basic                    500
2 Premium                  200
```



```
-- query finds the number of products sold for all brands and segments. means an empty grouping
--set which is denoted by ()
SELECT SUM (quantity) FROM sales;
```

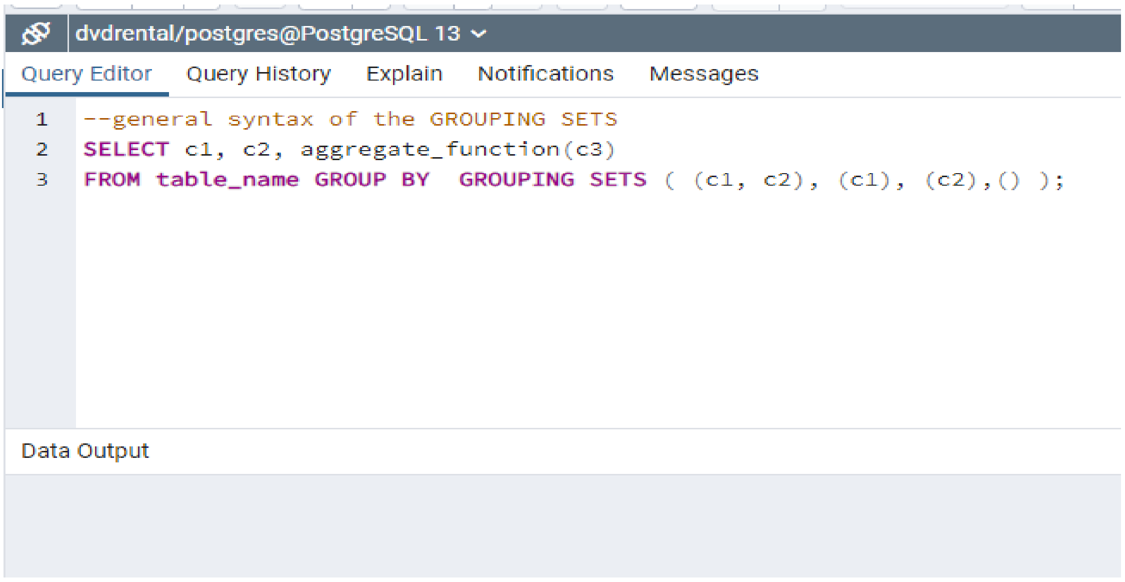
```
  sum
  bigint
1    700
```



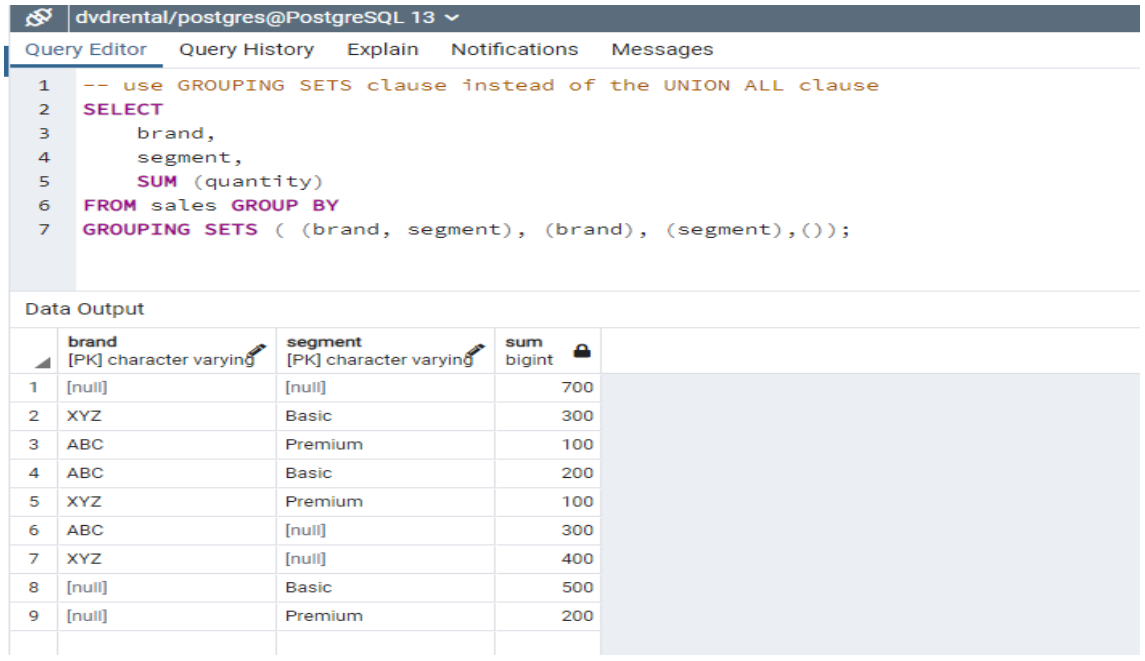
```
--UNION ALL requires all result sets to have the same number of columns with compatible data types,
--you need to adjust the queries by adding NULL to the selection list

SELECT brand, segment, SUM (quantity) FROM sales GROUP BY brand, segment
UNION ALL SELECT brand, NULL, SUM (quantity) FROM sales GROUP BY brand
UNION ALL SELECT NULL, segment, SUM (quantity) FROM sales GROUP BY
segment UNION ALL SELECT  NULL, NULL,
    SUM (quantity) FROM  sales;
```

	brand character varying	segment character varying	sum bigint
1	XYZ	Basic	300
2	ABC	Premium	100
3	ABC	Basic	200
4	XYZ	Premium	100
5	ABC	[null]	300
6	XYZ	[null]	400
7	[null]	Basic	500
8	[null]	Premium	200
9	[null]	[null]	700

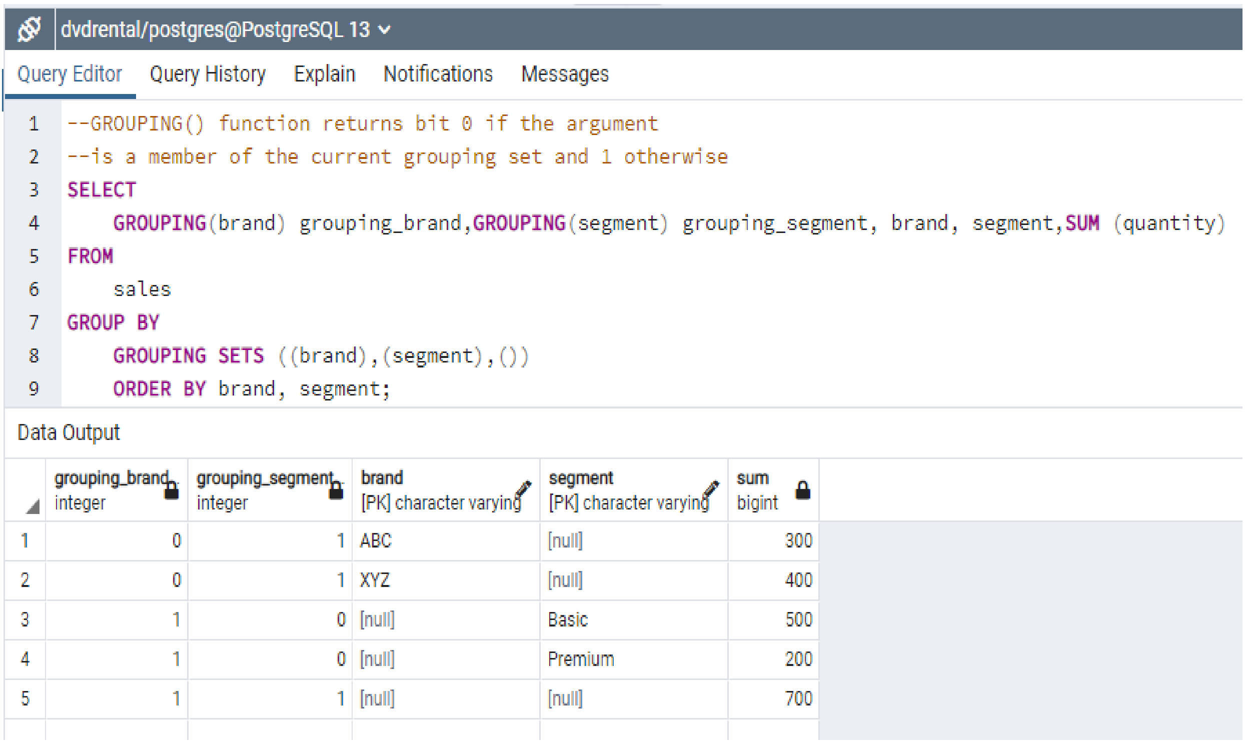


```
--general syntax of the GROUPING SETS
SELECT c1, c2, aggregate_function(c3)
FROM table_name GROUP BY  GROUPING SETS ( (c1, c2), (c1), (c2),() );
```



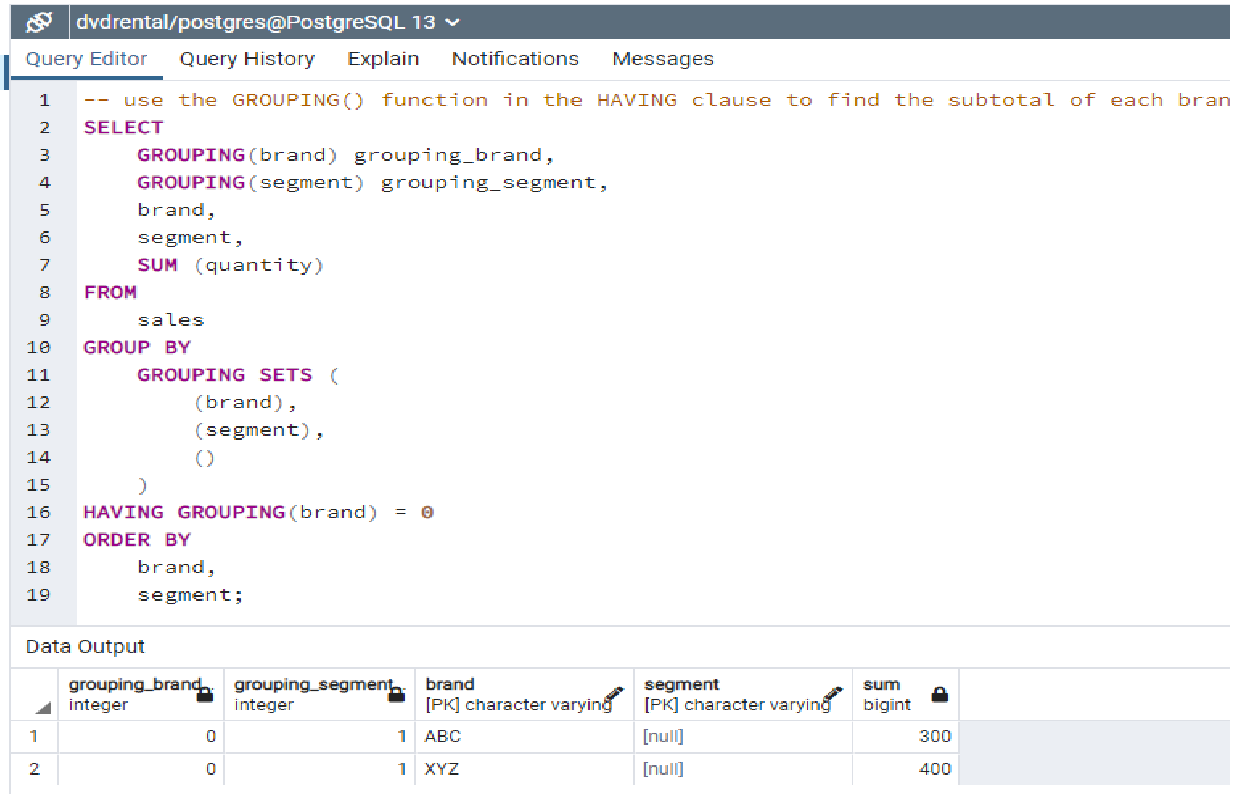
```sql
-- use GROUPING SETS clause instead of the UNION ALL clause
SELECT
    brand,
    segment,
    SUM (quantity)
FROM sales GROUP BY
GROUPING SETS ( (brand, segment), (brand), (segment),());
```

	brand [PK] character varying	segment [PK] character varying	sum bigint
1	[null]	[null]	700
2	XYZ	Basic	300
3	ABC	Premium	100
4	ABC	Basic	200
5	XYZ	Premium	100
6	ABC	[null]	300
7	XYZ	[null]	400
8	[null]	Basic	500
9	[null]	Premium	200



```sql
--GROUPING() function returns bit 0 if the argument
--is a member of the current grouping set and 1 otherwise
SELECT
    GROUPING(brand) grouping_brand,GROUPING(segment) grouping_segment, brand, segment,SUM (quantity)
FROM
    sales
GROUP BY
    GROUPING SETS ((brand),(segment),())
    ORDER BY brand, segment;
```

	grouping_brand integer	grouping_segment integer	brand [PK] character varying	segment [PK] character varying	sum bigint
1	0	1	ABC	[null]	300
2	0	1	XYZ	[null]	400
3	1	0	[null]	Basic	500
4	1	0	[null]	Premium	200
5	1	1	[null]	[null]	700



```
-- use the GROUPING() function in the HAVING clause to find the subtotal of each bran
SELECT
    GROUPING(brand) grouping_brand,
    GROUPING(segment) grouping_segment,
    brand,
    segment,
    SUM (quantity)
FROM
    sales
GROUP BY
    GROUPING SETS (
        (brand),
        (segment),
        ()
    )
HAVING GROUPING(brand) = 0
ORDER BY
    brand,
    segment;
```

	grouping_brand integer	grouping_segment integer	brand [PK] character varying	segment [PK] character varying	sum bigint
1	0	1	ABC	[null]	300
2	0	1	XYZ	[null]	400