

Individual project

- I created database name "Database" in pgAdmin 4.
- For that , right click on database > create database> enter name of database .
- **Goal of this project :** 1) to understand how to create database and queries.

2) for this Database , Goal is to find out most popular product among customer.

- **Brief information :**

There are 4 table , Product , Customer , Store ,Sales

- **Product Table :**

Product_id (primary key)

Product_name

Product_details

Price

- **Customer Table :**

customer_id (primarykey)

First_name

Last_name

Product_id (Foreign key)

- **Store Table :**

Store_id (product_id)

Store_name

Customer_id (Foreign key)

- **Sales Table :**

Sales_id (Primary key) , Store_id (Foreign key)

First_name

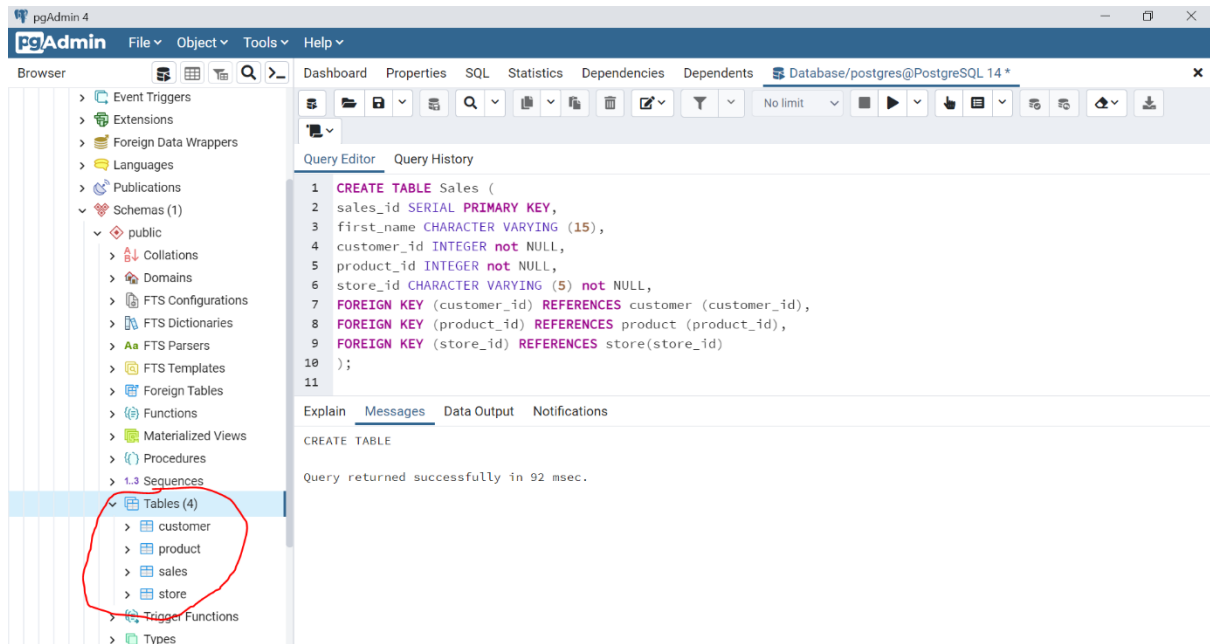
Product_id (Foreign key)

Customer_id (Foreign key)

Queries for create product, customer, store, sales table

```
CREATE TABLE Product (  
    product_id SERIAL PRIMARY KEY,  
    product_name CHARACTER VARYING (15),  
    product_details CHARACTER VARYING (25)  
);  
  
CREATE TABLE Customer (  
    customer_id SERIAL PRIMARY KEY,  
    First_name CHARACTER VARYING (10),  
    Last_name CHARACTER VARYING (10),  
    product_id INTEGER not NULL,  
    FOREIGN KEY (product_id) REFERENCES product (product_id)  
);  
  
CREATE TABLE Store (  
    store_id CHARACTER VARYING (5) PRIMARY KEY,  
    store_name CHARACTER VARYING (15),  
    customer_id INTEGER not NULL,  
    FOREIGN KEY (customer_id) REFERENCES customer (customer_id)  
);  
  
CREATE TABLE Sales (  
    sales_id SERIAL PRIMARY KEY,  
    first_name CHARACTER VARYING (15),  
    customer_id INTEGER not NULL,  
    product_id INTEGER not NULL,  
    store_id CHARACTER VARYING (5) not NULL,  
    FOREIGN KEY (customer_id) REFERENCES customer (customer_id),  
    FOREIGN KEY (product_id) REFERENCES product (product_id),  
    FOREIGN KEY (store_id) REFERENCES store(store_id)  
);
```

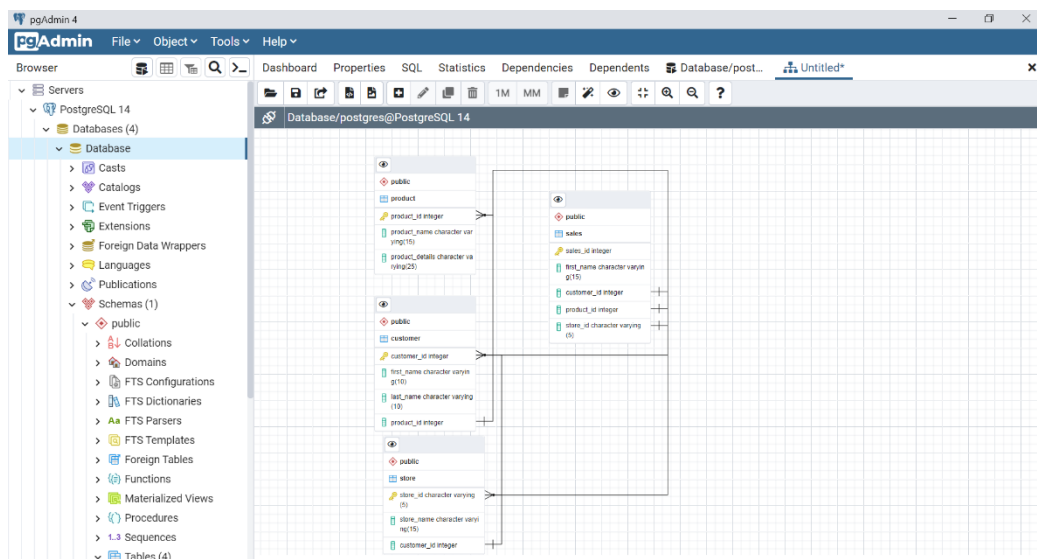
Successfully created 4 tables



Generate ERD/Schema

Select Database

Right click on it and select Generate ERD



Input Queries

/*data for product table*/

```
INSERT INTO product(product_id,product_name,product_details) values (1,'Coffee','Baverages');
INSERT INTO product(product_id,product_name,product_details) values (2,'Cream','Dairy');
INSERT INTO product(product_id,product_name,product_details) values (3,'Apple','Fruit');
INSERT INTO product(product_id,product_name,product_details) values (4,'Fish Fillet','Meat');
INSERT INTO product(product_id,product_name,product_details) values (5,'Tea','Baverages');
INSERT INTO product(product_id,product_name,product_details) values (6,'Onion','Vegetables');
INSERT INTO product(product_id,product_name,product_details) values (7,'Tissue','Paper product');
INSERT INTO product(product_id,product_name,product_details) values (8,'Chocolate chips','Bakery');
INSERT INTO product(product_id,product_name,product_details) values (9,'Stawbary','Fruit');
INSERT INTO product(product_id,product_name,product_details) values (10,'Chicken','Meat');
```

/*data for customer table*/

```
INSERT INTO customer(customer_id,First_name,Last_name,product_id) values (01,'Hiral','Patel',1);
INSERT INTO customer(customer_id,First_name,Last_name,product_id) values (02,'Utsav','Patel',3);
INSERT INTO customer(customer_id,First_name,Last_name,product_id) values (03,'Deep','Sindu',7);
INSERT INTO customer(customer_id,First_name,Last_name,product_id) values (04,'Alex','Roy',3);
INSERT INTO customer(customer_id,First_name,Last_name,product_id) values (05,'Sofi','Zadeja',5);
INSERT INTO customer(customer_id,First_name,Last_name,product_id) values (06,'Ali','singla',2);
INSERT INTO customer(customer_id,First_name,Last_name,product_id) values (07,'Jeny','Singh',6);
INSERT INTO customer(customer_id,First_name,Last_name,product_id) values (08,'Tom','williams',4);
INSERT INTO customer(customer_id,First_name,Last_name,product_id) values (09,'Wiliam','peter',1);
INSERT INTO customer(customer_id,First_name,Last_name,product_id) values (010,'Kajal','Chada',8);
INSERT INTO customer(customer_id,First_name,Last_name,product_id) values (11,'Kia','williams',9);
INSERT INTO customer(customer_id,First_name,Last_name,product_id) values (12,'Vyomesh','peter',10);
INSERT INTO customer(customer_id,First_name,Last_name,product_id) values (13,'Jemi','Chada',9);
```

/*data for store table*/

```
INSERT INTO store(store_id,store_name,customer_id) values ('VM','Vmart',10);
INSERT INTO store(store_id,store_name,customer_id) values ('DM','Dmart',06);
INSERT INTO store(store_id,store_name,customer_id) values ('BI','Bigbajar',010);
INSERT INTO store(store_id,store_name,customer_id) values ('WA','Walmart',04);
INSERT INTO store(store_id,store_name,customer_id) values ('BG','Big Basket',03);
INSERT INTO store(store_id,store_name,customer_id) values ('SE','Seven Eleven',01);
INSERT INTO store(store_id,store_name,customer_id) values ('TR','Target',010);
INSERT INTO store(store_id,store_name,customer_id) values ('RE','Reliance',09);
INSERT INTO store(store_id,store_name,customer_id) values ('AB','24x7',05);
INSERT INTO store(store_id,store_name,customer_id) values ('RL','Reliance Smart',07);
INSERT INTO store(store_id,store_name,customer_id) values ('SP','Spencers',02);
INSERT INTO store(store_id,store_name,customer_id) values ('TE','Tesco',08);
INSERT INTO store(store_id,store_name,customer_id) values ('ZM','ZMALL',11);
INSERT INTO store(store_id,store_name,customer_id) values ('GE','GMALL',12);
INSERT INTO store(store_id,store_name,customer_id) values ('KI','KITCHEN',13);
```

/*data for sales table*/

```
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (101,1,1,'SE','Hiral');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (102,2,3,'SP','Utsav');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (103,3,7,'RL','Deep');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (104,4,3,'BG','Alex');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (105,5,5,'AB','Sofi');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (106,6,2,'DM','Ali');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (107,7,6,'RL','Jeny');
```

```
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (108,8,4,'TE','Tom');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (109,9,1,'RE','William');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (110,10,8,'BI','Kajal');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (111,11,9,'ZM','Kia');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (112,12,10,'GE','Vyomesh');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (113,13,9,'KI','Jemi');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (114,7,6,'RL','Jeny');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (115,6,2,'DM','Ali');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (116,5,5,'AB','Sofi');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (117,4,3,'WA','Alex');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (118,3,7,'BG','Deep');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (119,2,3,'SP','Utsav');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (120,1,1,'SE','Hiral');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (121,10,8,'TR','Kajal');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (122,11,9,'ZM','Kia');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (123,8,4,'SP','Tom');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (124,7,6,'RL','Jeny');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (125,6,2,'DM','Ali');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (126,1,8,'SE','Hiral');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (127,2,8,'SP','Utsav');
INSERT INTO sales(sales_id,customer_id,product_id,store_id,First_name) values (128,3,8,'BG','Deep');
```

Create views

```
CREATE VIEW SALES_INFO AS
```

```
SELECT P.product_id , c.first_name , s.store_name , t.sales_id
```

```
FROM product p
```

```
LEFT JOIN customer c
```

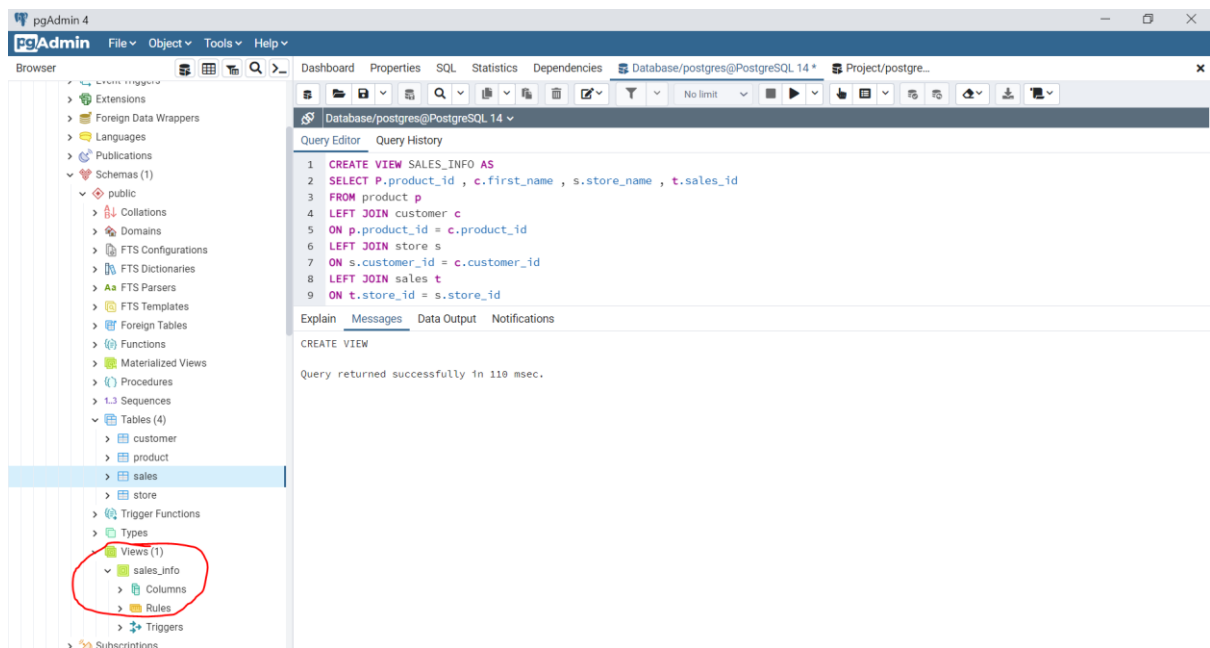
```
ON p.product_id = c.product_id
```

```
LEFT JOIN store s
```

```
ON s.customer_id = c.customer_id
```

```
LEFT JOIN sales t
```

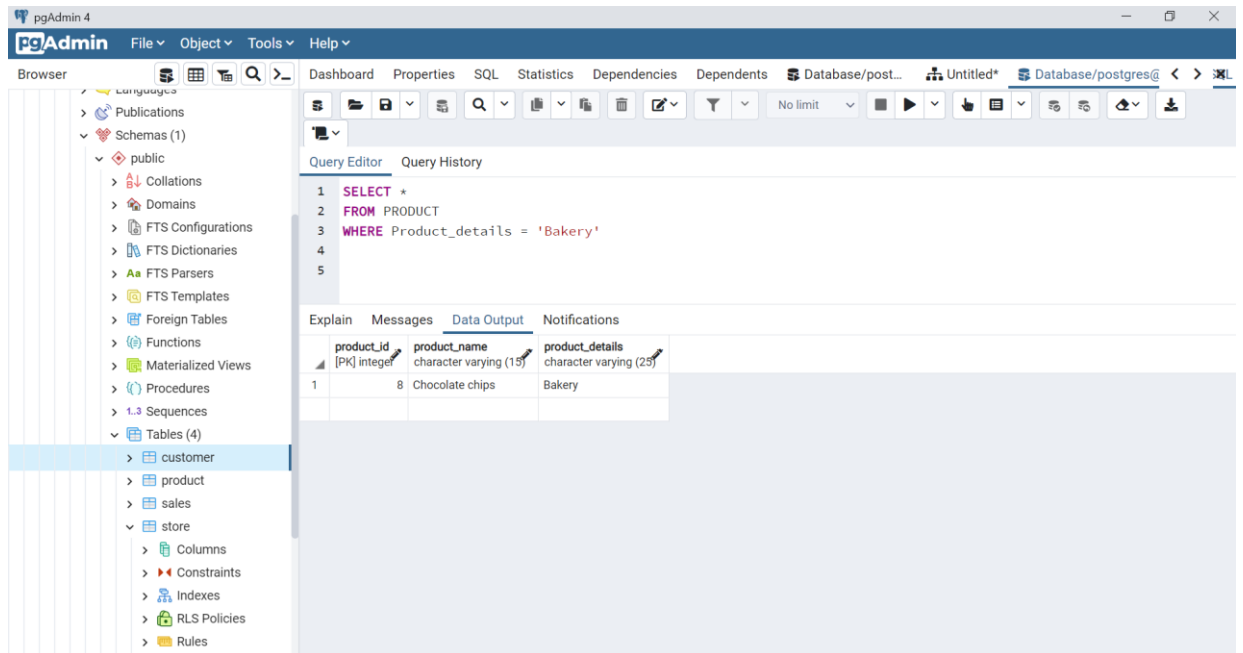
```
ON t.store_id = s.store_id
```



10 queries

Select product from bakery department from product table

```
SELECT *  
FROM PRODUCT  
WHERE Product_details = 'Bakery'
```



Joint two table , product and customer

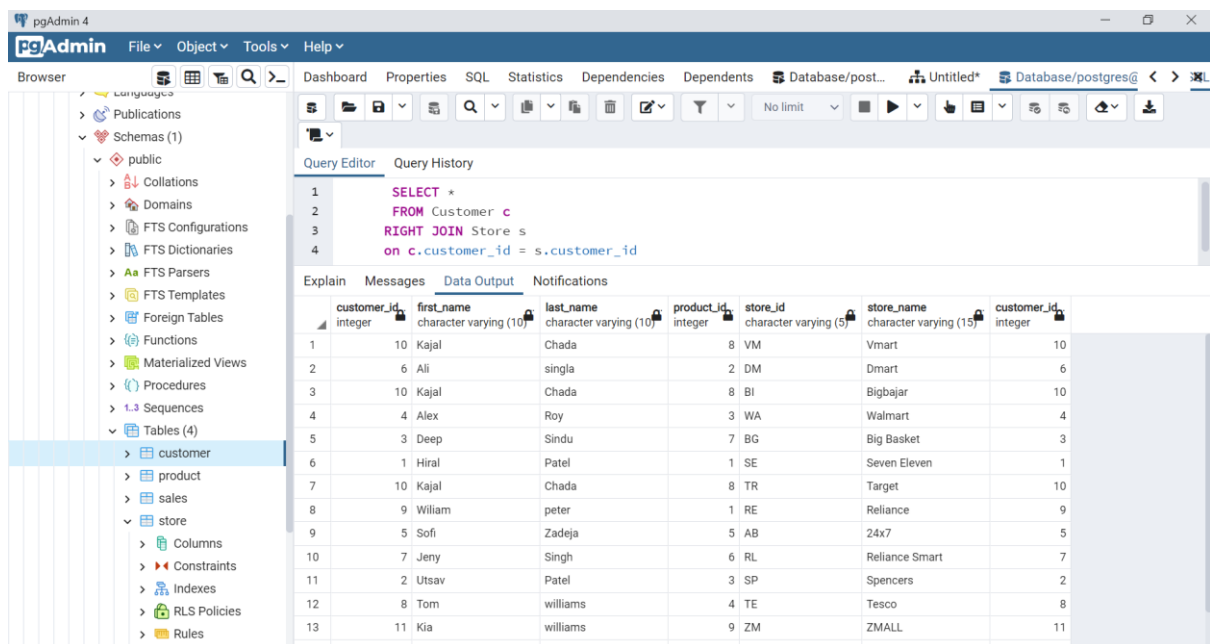
```
SELECT *  
FROM Product p  
INNER JOIN Customer c  
on p.product_id = c.product_id
```

The screenshot shows the pgAdmin 4 interface. On the left is the 'Browser' pane with a tree view of the database structure. The 'public' schema is expanded, showing 'Tables (4)' with 'customer' selected. The main pane is divided into 'Query Editor' and 'Data Output'. The 'Query Editor' contains the SQL query: `SELECT * FROM Product p INNER JOIN Customer c on p.product_id = c.product_id`. The 'Data Output' tab shows the results of the query in a table with 7 columns: product_id, product_name, product_details, customer_id, first_name, last_name, and product_id. The results are listed in 13 rows.

	product_id integer	product_name character varying (15)	product_details character varying (25)	customer_id integer	first_name character varying (10)	last_name character varying (10)	product_id integer
1	1	Coffee	Beverages	1	Hiral	Patel	1
2	3	Apple	Fruit	2	Utsav	Patel	3
3	7	Tissue	Paper product	3	Deep	Sindu	7
4	3	Apple	Fruit	4	Alex	Roy	3
5	5	Tea	Beverages	5	Sofi	Zadeja	5
6	2	Cream	Dairy	6	Ali	singla	2
7	6	Onion	Vegetables	7	Jeny	Singh	6
8	4	Fish Fillet	Meat	8	Tom	williams	4
9	1	Coffee	Beverages	9	William	peter	1
10	8	Chocolate chips	Bakery	10	Kajal	Chada	8
11	9	Stawbary	Fruit	11	Kia	williams	9
12	10	Chicken	Meat	12	Vyomesh	peter	10
13	9	Stawbary	Fruit	13	Jemi	Chada	9

Right join

```
SELECT *  
FROM Customer c  
RIGHT JOIN Store s  
on c.customer_id = s.customer_id
```



The screenshot shows the pgAdmin 4 interface. On the left is the 'Browser' pane with a tree view of the database structure. The 'customer' table is selected under the 'public' schema. The main pane is divided into 'Query Editor' and 'Data Output'. The 'Query Editor' contains the following SQL query:

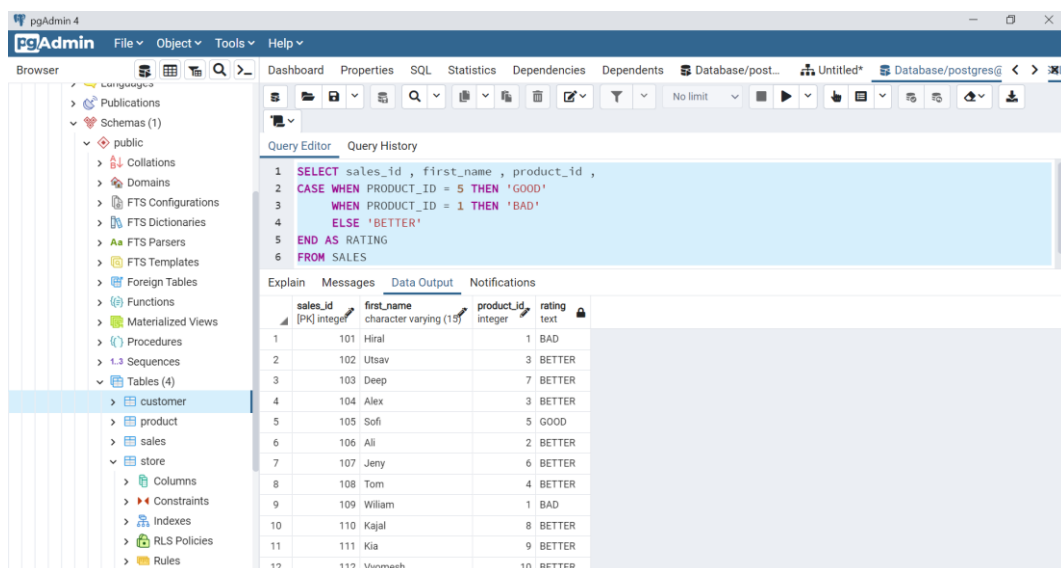
```
1 SELECT *  
2 FROM Customer c  
3 RIGHT JOIN Store s  
4 on c.customer_id = s.customer_id
```

The 'Data Output' tab shows the results of the query in a table with 7 columns: customer_id, first_name, last_name, product_id, store_id, store_name, and customer_id. The results are as follows:

	customer_id	first_name	last_name	product_id	store_id	store_name	customer_id
1	10	Kajal	Chada	8	VM	Vmart	10
2	6	Ali	singla	2	DM	Dmart	6
3	10	Kajal	Chada	8	BI	Bigbajar	10
4	4	Alex	Roy	3	WA	Walmart	4
5	3	Deep	Sindu	7	BG	Big Basket	3
6	1	Hiral	Patel	1	SE	Seven Eleven	1
7	10	Kajal	Chada	8	TR	Target	10
8	9	William	peter	1	RE	Reliance	9
9	5	Sofi	Zadeja	5	AB	24x7	5
10	7	Jeny	Singh	6	RL	Reliance Smart	7
11	2	Utsav	Patel	3	SP	Spencers	2
12	8	Tom	williams	4	TE	Tesco	8
13	11	Kia	williams	9	ZM	ZMALL	11

Add a new column to the result called 'rating' to group results into the following priority groups Good(product_id=5) ,Bad (product_id=1) , else(Better)

```
SELECT sales_id , first_name , product_id ,  
CASE WHEN PRODUCT_ID = 5 THEN 'GOOD'  
      WHEN PRODUCT_ID = 1 THEN 'BAD'  
      ELSE 'BETTER'  
END AS RATING  
FROM SALES
```



The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane displays the database structure, including the 'public' schema and the 'sales' table. The 'Query Editor' pane on the right contains the following SQL query:

```
1 SELECT sales_id , first_name , product_id ,  
2 CASE WHEN PRODUCT_ID = 5 THEN 'GOOD'  
3      WHEN PRODUCT_ID = 1 THEN 'BAD'  
4      ELSE 'BETTER'  
5 END AS RATING  
6 FROM SALES
```

Below the query editor, the 'Data Output' tab shows the results of the query. The results are displayed in a table with the following columns: sales_id, first_name, product_id, and rating. The data is as follows:

sales_id	first_name	product_id	rating
1	101 Hiral	1	BAD
2	102 Utsav	3	BETTER
3	103 Deep	7	BETTER
4	104 Alex	3	BETTER
5	105 Sofi	5	GOOD
6	106 Ali	2	BETTER
7	107 Jeny	6	BETTER
8	108 Tom	4	BETTER
9	109 William	1	BAD
10	110 Kajal	8	BETTER
11	111 Kia	9	BETTER
12	112 Vyomesh	10	BETTER

Select a product whose product_id=1 from sales using subquery

SELECT *

FROM sales

WHERE product_id IN (SELECT product_id

FROM customer

WHERE product_id = 1)

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, with the 'sales' table selected under the 'public' schema. The main window shows the 'Query Editor' with the following SQL query:

```
1 SELECT *
2 FROM sales
3 WHERE product_id IN (SELECT product_id
4                       FROM customer
5                       WHERE product_id = 1)
```

Below the query editor, the 'Data Output' tab is active, displaying the results of the query in a table format:

sales_id	first_name	customer_id	product_id	store_id
1	101 Hiral	1	1	SE
2	109 William	9	1	RE
3	120 Hiral	1	1	SE

Add a new columns name price_\$ in product

ALTER TABLE product

ADD COLUMN price_\$ INTEGER;

UPDATE product SET "price_\$" = 12 WHERE "product_id" = 1;

UPDATE product SET "price_\$" = 22 WHERE "product_id" = 2;

UPDATE product SET "price_\$" = 11 WHERE "product_id" = 3;

UPDATE product SET "price_\$" = 30 WHERE "product_id" = 4;

UPDATE product SET "price_\$" = 12 WHERE "product_id" = 5;

UPDATE product SET "price_\$" = 5 WHERE "product_id" = 6;

UPDATE product SET "price_\$" = 32 WHERE "product_id" = 7;

UPDATE product SET "price_\$" = 15 WHERE "product_id" = 8;

UPDATE product SET "price_\$" = 10 WHERE "product_id" = 9;

UPDATE product SET "price_\$" = 6 WHERE "product_id" = 10;

ALTER TABLE product

ALTER COLUMN "price_\$" SET NOT null;

SELECT * FROM product

The screenshot shows the pgAdmin 4 interface with the following components:

- Query Editor:** Contains 15 SQL statements:
 - ALTER TABLE product
 - ADD COLUMN price_\$ INTEGER;
 - UPDATE product SET "price_\$" = 12 WHERE "product_id" = 1;
 - UPDATE product SET "price_\$" = 22 WHERE "product_id" = 2;
 - UPDATE product SET "price_\$" = 11 WHERE "product_id" = 3;
 - UPDATE product SET "price_\$" = 30 WHERE "product_id" = 4;
 - UPDATE product SET "price_\$" = 12 WHERE "product_id" = 5;
 - UPDATE product SET "price_\$" = 5 WHERE "product_id" = 6;
 - UPDATE product SET "price_\$" = 32 WHERE "product_id" = 7;
 - UPDATE product SET "price_\$" = 15 WHERE "product_id" = 8;
 - UPDATE product SET "price_\$" = 10 WHERE "product_id" = 9;
 - UPDATE product SET "price_\$" = 6 WHERE "product_id" = 10;
 - ALTER TABLE product
 - ALTER COLUMN "price_\$" SET NOT null;
 - SELECT * FROM product
- Data Output:** A table with 4 columns: product_id, product_name, product_details, and price_\$. It contains 7 rows of data.

product_id	product_name	product_details	price_\$
1	Coffee	Beverages	12
2	Cream	Dairy	22
3	Apple	Fruit	11
4	Fish Fillet	Meat	30
5	Tea	Beverages	12
6	Onion	Vegetables	5
7	Tissue	Paper product	32

Sort product ascending price

```
SELECT product_id , product_name , price_
```

```
FROM product
```

```
WHERE product_id BETWEEN 3 AND 9
```

```
ORDER BY price_ ASC
```

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane displays the database structure, including 'Servers (1)' with 'PostgreSQL 14', 'Databases (4)', and 'public' schema with various objects like 'Collations', 'Domains', 'FTS Configurations', 'FTS Dictionaries', 'FTS Parsers', 'FTS Templates', 'Foreign Tables', 'Functions', 'Materialized Views', 'Procedures', 'Sequences', 'Tables (4)' (customer, product, sales), and 'Columns (5)' (sales_id).

The main pane shows the 'Query Editor' with the following SQL query:

```
1 SELECT product_id , product_name , price_$
2 FROM product
3 WHERE product_id BETWEEN 3 AND 9
4 ORDER BY price_$ ASC
5
```

Below the query editor, the 'Data Output' tab is active, displaying the results of the query in a table format:

product_id	product_name	price_\$
6	Onion	5
9	Stawbary	10
3	Apple	11
5	Tea	12
8	Chocolate chips	15
4	Fish Fillet	30
7	Tissue	32

Calculate total price of product based on product_id

```
SELECT s.product_id , p.product_name , SUM(price_$) AS total_price
```

```
FROM PRODUCT p
```

```
inner join sales s
```

```
on p.product_id = s.product_id
```

```
GROUP BY s.product_id , p.product_name
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, including the 'public' schema and the 'store' table. The main window shows the 'Query Editor' with the following SQL query:

```
1 SELECT s.product_id , p.product_name , SUM(price_$) AS total_price
2 FROM PRODUCT p
3 inner join sales s
4 on p.product_id = s.product_id
5 GROUP BY s.product_id , p.product_name
6
```

Below the query editor, the 'Data Output' tab is selected, displaying the results of the query in a table format:

	product_id integer	product_name character varying (15)	total_price bigint
1	10	Chicken	6
2	8	Chocolate chips	75
3	7	Tissue	64
4	1	Coffee	36
5	9	Stawbary	30
6	5	Tea	24
7	6	Onion	15
8	4	Fish Fillet	60
9	3	Apple	44
10	2	Cream	66

Count number of customer for each product

```
SELECT p.product_name , count(customer_id) AS total_customer
```

```
FROM PRODUCT p
```

```
inner join sales s
```

```
on p.product_id = s.product_id
```

```
GROUP BY p.product_name
```

The screenshot shows the pgAdmin 4 interface. On the left is the 'Browser' pane with a tree view of the database structure. The 'public' schema is expanded, showing 'Tables (4)' with 'customer', 'product', 'sales', and 'store'. The 'Columns (3)' for the 'store' table are visible: 'store_id' and 'store_name'. The main pane is the 'Query Editor' for 'Database/postgres@PostgreSQL 14 *'. It contains the following SQL query:

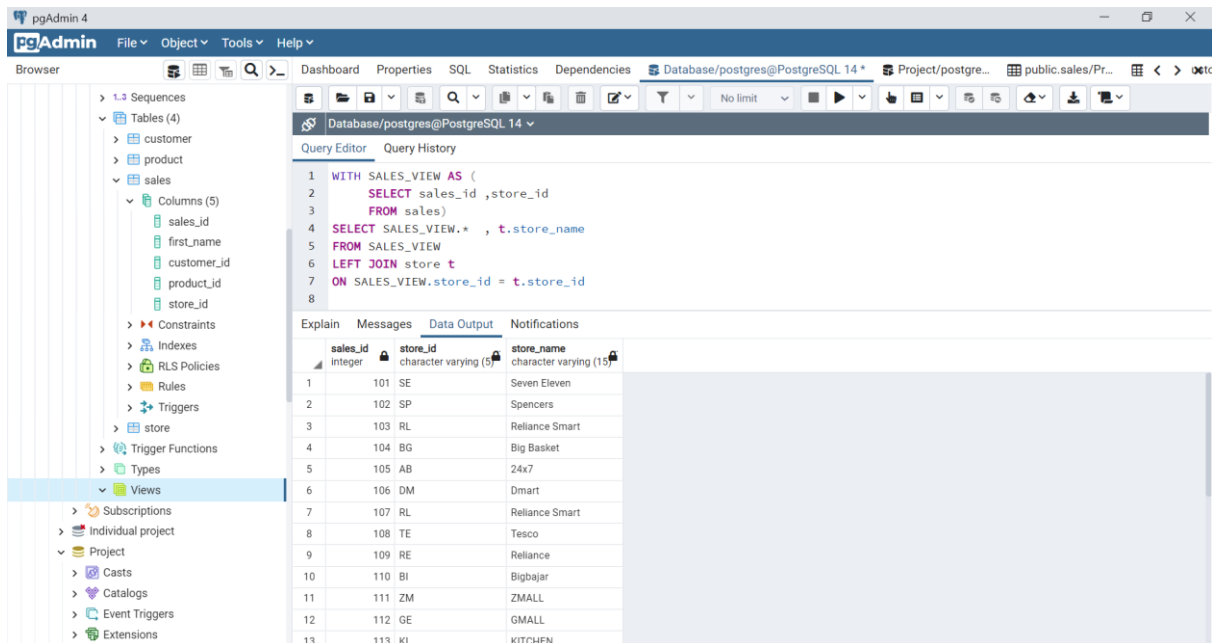
```
1 SELECT p.product_name , count(customer_id) AS total_customer
2 FROM PRODUCT p
3 inner join sales s
4 on p.product_id = s.product_id
5 GROUP BY p.product_name
6
```

Below the query editor, the 'Data Output' tab is active, displaying the results of the query in a table:

product_name	total_customer
Chocolate chips	5
Onion	3
Cream	3
Coffee	3
Chicken	1
Stawbary	3
Fish Fillet	2
Tissue	2
Tea	2
Apple	4

Create CTE view

```
WITH SALES_VIEW AS (  
    SELECT sales_id ,store_id  
    FROM sales)  
  
SELECT SALES_VIEW.* , t.store_name  
FROM SALES_VIEW  
LEFT JOIN store t  
ON SALES_VIEW.store_id = t.store_id
```



The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane displays a tree view of the database structure, including 'Tables (4)', 'Columns (5)', 'Constraints', 'Indexes', 'RLS Policies', 'Rules', 'Triggers', 'store', 'Trigger Functions', 'Types', 'Views', 'Subscriptions', 'Individual project', 'Project', 'Casts', 'Catalogs', 'Event Triggers', and 'Extensions'. The 'Views' folder is selected. The main pane shows the 'Query Editor' with the following SQL query:

```
1 WITH SALES_VIEW AS (  
2     SELECT sales_id ,store_id  
3     FROM sales)  
4 SELECT SALES_VIEW.* , t.store_name  
5 FROM SALES_VIEW  
6 LEFT JOIN store t  
7 ON SALES_VIEW.store_id = t.store_id  
8
```

Below the query editor, the 'Data Output' tab is active, displaying the results of the query in a table with three columns: 'sales_id', 'store_id', and 'store_name'. The results are as follows:

sales_id	store_id	store_name
1	101	SE
2	102	SP
3	103	RL
4	104	BG
5	105	AB
6	106	DM
7	107	RL
8	108	TE
9	109	RE
10	110	BI
11	111	ZMALL
12	112	GE
13	113	KI