



CSE 2211 - Database Management System Lab

Project Name: Shop Management System

Submitted To

Abu Ahmed Ferdaus

Associate Professor

Dr. Md Mamun-Or-Rashid

Professor

**Department of Computer Science & Engineering
University of Dhaka**

Submitted By

Md. Tauseef - Ur - Rahman (Roll 24)

Faiaz Mahmud (Roll 46)

28th Batch

Submitted On

9th March 2025

Table of Contents

- Brief Description of our Project
- Schemas with Attributes
- Schema Diagram of the database
- E-R Diagram of the database
- Snapshots of SQL DDL of all the schemas/tables
- Snapshots of the instances (data of the populated tables)
- Query statement, Relational Algebra Expression, SQL statement, and snapshots of the outputs
- Create views and use those views to answer queries
- Some snaps of our web application
- List of non-trivial FDs
- Proof that the schemas are in the desired normal forms
- Front End designing tools and techniques, advantages and disadvantages of the implementation
- Conclusion of the work

1. Brief Introduction of our Database Project

The **Shop Management System** is a database-driven application designed to manage retail operations using structured queries efficiently. The system relies on **SQL queries** to handle core functionalities such as user authentication, product management, order processing, and billing. Queries are used to **insert, update, delete, and retrieve** data, ensuring seamless transactions and real-time data updates. Complex **JOIN queries** enable relationships between tables, such as fetching order details along with customer information, while **aggregate functions** calculate total sales and inventory levels and so on. Optimized indexing and constraints ensure data integrity, improving performance and consistency across the system.

2. Schemas with Attributes

There are in total 8 schemas/tables in our database project. Among these, **three** are independent tables namely **users**, **products**, and **workers**. All of the schemas with their attributes are given below :

1. Users (**users**)

Stores user details.

- `user_id` (INT, PK) – Unique ID.
- `user_name` (VARCHAR(50)) – Name.
- `user_email` (VARCHAR(50)) – Unique Email.
- `user_password` (VARCHAR(255)) – Password.
- `user_contact_no` (VARCHAR(20)) – Contact number.
- `is_admin` (BOOLEAN) – Admin status.

2. Products (**product**)

Stores product details.

- `prod_id` (INT, PK) – Unique product ID.
- `prod_name` (VARCHAR(255)) – Name.
- `prod_image` (VARCHAR(255)) – Image URL.
- `prod_quantity` (INT) – Stock quantity.
- `prod_price` (DECIMAL(10,2)) – Price.
- `rating_stars` (INT) – Rating.
- `prod_discount` (DECIMAL(5,2)) – Discount.

3. Workers (worker)

Stores employee details.

- `worker_id` (INT, PK) – Unique ID.
- `worker_name` (VARCHAR(50)) – Name.
- `worker_email` (VARCHAR(50)) – Unique Email.
- `worker_contact_no` (VARCHAR(20)) – Contact number.
- `worker_salary` (DOUBLE) – Salary.

4. Shopping Cart (shopping_cart)

Tracks items in a cart.

- `cart_id` (INT, PK) – Unique cart ID.
- `prod_id` (INT, FK) – Product ID.
- `user_id` (INT, FK) – User ID.

5. Orders (orders)

Stores order information.

- `order_id` (INT, PK) – Unique order ID.
- `order_date` (DATE) – Date.
- `user_id` (INT, FK) – User who ordered.
- `delivery_address` (VARCHAR(50)) – Delivery address.
- `total_amt` (DECIMAL(10,2)) – Order total.
- `order_status` (VARCHAR(20)) – Status.

6. Order Details (order_detail)

Stores products in an order.

- `order_id` (INT, FK) – Order ID.
- `prod_id` (INT, FK) – Product ID.
- `prod_qty` (INT) – Quantity.
- `prod_price` (DOUBLE) – Price.
- `prod_total_price` (DOUBLE) – Total price.

7. Billing (bill_detail)

Stores billing info.

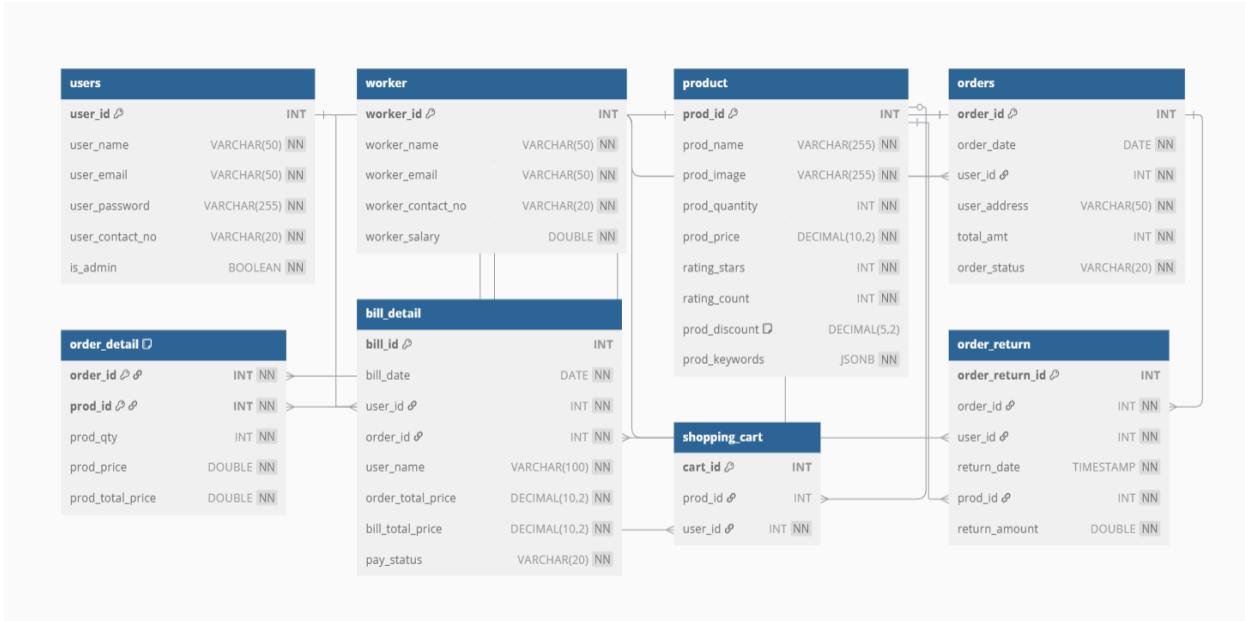
- bill_id (INT, PK) – Unique bill ID.
- bill_date (DATE) – Date.
- user_id (INT, FK) – User ID.
- order_id (INT, FK) – Order ID.
- order_total_price (DECIMAL(10,2)) – Order total.
- bill_total_price (DECIMAL(10,2)) – Final bill amount (after adding discounts).
- pay_status (VARCHAR(20)) – Payment status.

8. Order Returns (order_return)

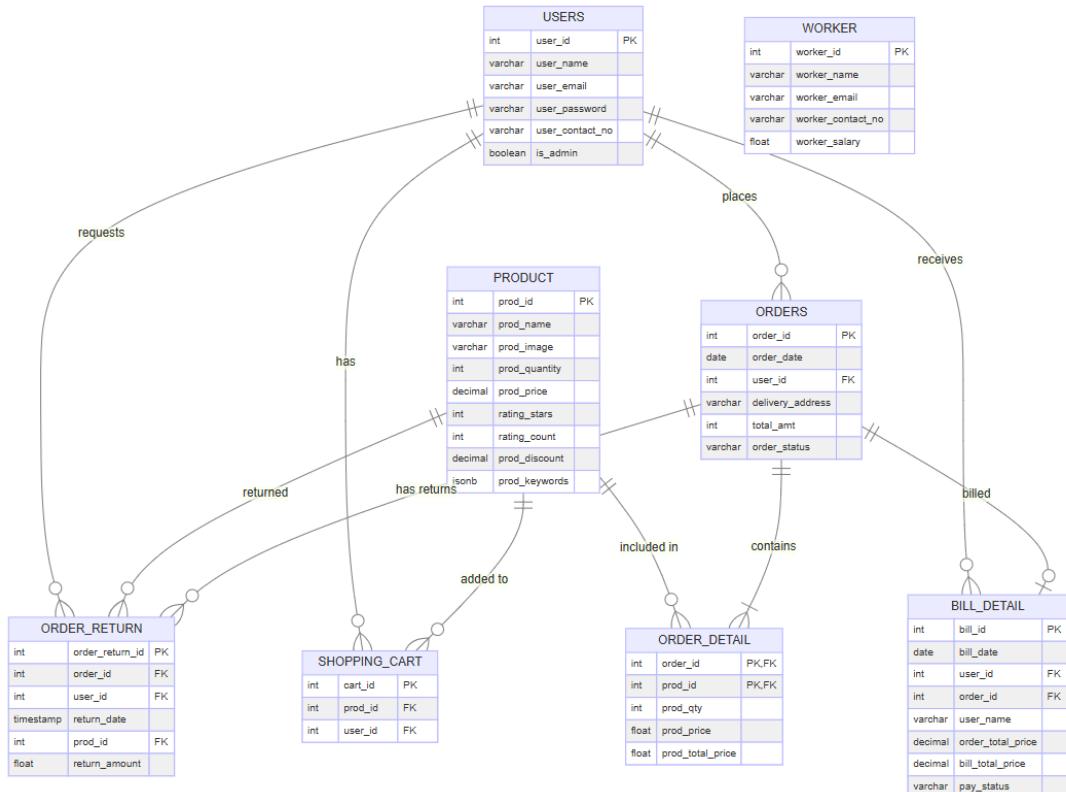
Tracks returned orders.

- order_return_id (INT, PK) – Unique return ID.
- order_id (INT, FK) – Order ID.
- user_id (INT, FK) – User ID.
- return_date (TIMESTAMP) – Return date.
- prod_id (INT, FK) – Product ID.
- return_amount (DOUBLE) – Refund amount.

3. Schema Diagram of the database



4. E-R Diagram of the database



5. Snapshots of SQL DDL of all the schemas/tables

Users Table

```
CREATE TABLE IF NOT EXISTS users (
    user_id SERIAL PRIMARY KEY,
    user_name VARCHAR(50) NOT NULL,
    user_email VARCHAR(50) UNIQUE NOT NULL,
    user_password VARCHAR(255) NOT NULL,
    user_contact_no VARCHAR(20) NOT NULL,
    is_admin BOOLEAN NOT NULL
);
```

Product Table

```
CREATE TABLE IF NOT EXISTS product (
    prod_id SERIAL PRIMARY KEY,
    prod_name VARCHAR(255) UNIQUE NOT NULL,
    prod_image VARCHAR(255) NOT NULL,
    prod_quantity INTEGER NOT NULL,
    prod_price DECIMAL(10,2) NOT NULL,
    rating_stars INTEGER NOT NULL,
    rating_count INTEGER NOT NULL,
    prod_discount DECIMAL(5,2) DEFAULT 0.0,
    prod_keywords JSONB NOT NULL
);
```

Order_detail Table

```
CREATE TABLE IF NOT EXISTS order_detail (
    order_id INT REFERENCES orders(order_id) NOT NULL,
    prod_id INT REFERENCES product(prod_id) NOT NULL,
    prod_qty INT NOT NULL,
    prod_price DOUBLE PRECISION NOT NULL,
    prod_total_price DOUBLE PRECISION NOT NULL,
    PRIMARY KEY (order_id, prod_id)
);
```

Shopping_cart Table

```
CREATE TABLE IF NOT EXISTS shopping_cart (
    cart_id SERIAL,
    prod_id INT NOT NULL,
    user_id INT NOT NULL,
    PRIMARY KEY (prod_id, user_id),
    FOREIGN KEY (prod_id) REFERENCES product(prod_id),
    CONSTRAINT shopping_cart_user_id_fkey FOREIGN KEY (user_id) REFERENCES users(user_id) ON UPDATE CASCADE
);
```

Worker Table

```
CREATE TABLE IF NOT EXISTS worker (
    worker_id SERIAL PRIMARY KEY,
    worker_name VARCHAR(50) NOT NULL,
    worker_email VARCHAR(50) NOT NULL,
    worker_contact_no VARCHAR(20) NOT NULL,
    worker_salary DOUBLE PRECISION NOT NULL
);
```

Bill_detail Table

```
CREATE TABLE IF NOT EXISTS bill_detail (
    bill_id SERIAL PRIMARY KEY, -- Automatically incrementing bill ID
    bill_date DATE NOT NULL, -- Date of the bill (set to current date)
    user_id INT NOT NULL, -- Foreign key to users table
    order_id INT NOT NULL, -- Foreign key to orders table
    user_name VARCHAR(100) NOT NULL, -- Name of the user
    order_total_price DECIMAL(10, 2) NOT NULL, -- Total price of the order
    bill_total_price DECIMAL(10, 2) NOT NULL, -- Total price of the bill (after any adjustments)
    pay_status VARCHAR(20) CHECK (pay_status IN ('paid', 'unpaid')) NOT NULL -- Payment status of the bill
    CONSTRAINT bill_detail_user_id_fkey FOREIGN KEY (user_id) REFERENCES users(user_id) ON UPDATE CASCADE,
    FOREIGN KEY (order_id) REFERENCES orders(order_id)
);
```

Orders Table

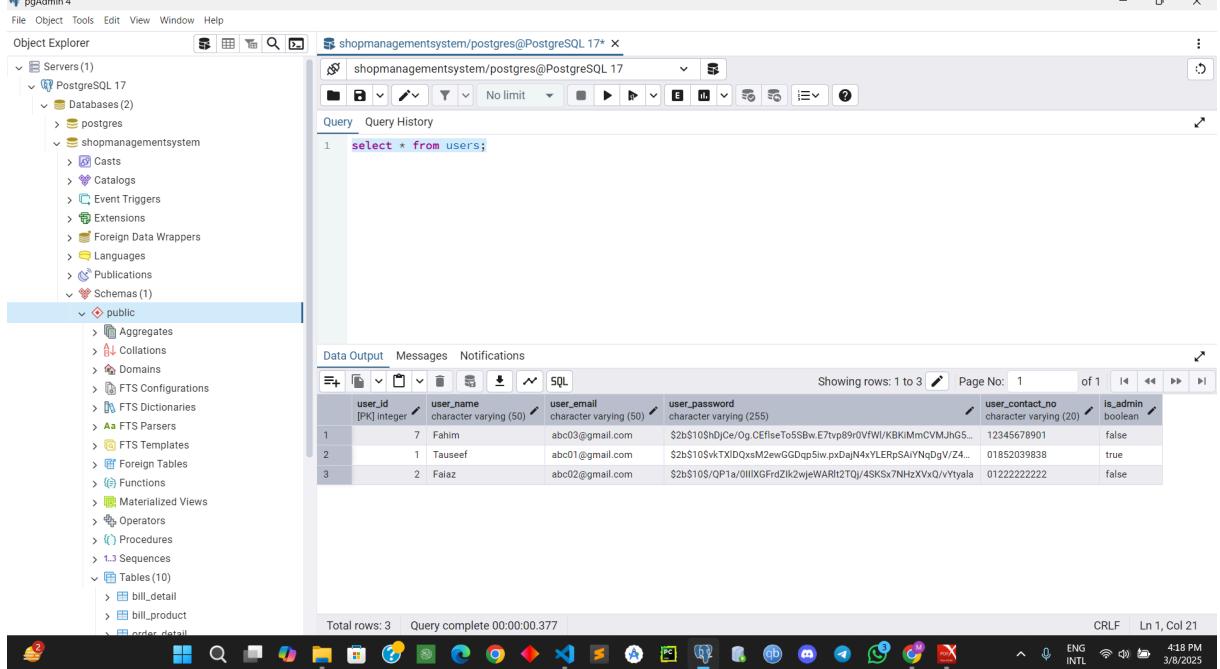
```
CREATE TABLE IF NOT EXISTS orders (
    order_id SERIAL PRIMARY KEY,
    order_date DATE NOT NULL,
    user_id INT NOT NULL,
    delivery_address VARCHAR(50) NOT NULL,
    total_amt INT NOT NULL,
    order_status VARCHAR(20) CHECK (order_status IN ('delivered', 'in process')) NOT NULL,
    CONSTRAINT orders_user_id_fkey FOREIGN KEY (user_id) REFERENCES users(user_id) ON UPDATE CASCADE
);
```

Order_return Table

```
CREATE TABLE IF NOT EXISTS order_return (
    order_return_id SERIAL PRIMARY KEY,
    order_id INT REFERENCES orders(order_id) NOT NULL,
    user_id INT REFERENCES users(user_id) NOT NULL,
    return_date TIMESTAMP NOT NULL,
    prod_id INT REFERENCES product(prod_id) NOT NULL,
    return_amount DOUBLE PRECISION NOT NULL
);
```

6. Snapshots of the instances (data of the populated tables)

Users Page



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

- Object Explorer:** Shows the database structure under "Shopmanagementsystem". The "Tables" node is expanded, showing "users" and other tables like "bill_detail", "bill_product", and "order_detail".
- Query Editor:** A query is running: `select * from users;`. The results are displayed in a table:

	user_id [PK] integer	user_name character varying (50)	user_email character varying (50)	user_password character varying (255)	user_contact_no character varying (20)	is_admin boolean
1	7	Fahim	abc03@gmail.com	\$2b\$10\$hdjCe/Og.CEfseTo5SBw.E7vp89r0VfW/KBKIMnCVMJhG5...	12345678901	false
2	1	Tauseef	abc01@gmail.com	\$2b\$10\$vkTXIDQxsM2ewGDGp5iwxDajN4xYLERpSAv/NdgV/Z...	01852039838	true
3	2	Faiaz	abc02@gmail.com	\$2b\$10\$/QP1a/0IIIXGFrdZk2wjeWARh2TQj/4SKSx7HzXvQ/vYtala	01222222222	false

Product Page

The screenshot shows the pgAdmin 4 interface with the title "Product Page". The left sidebar displays the Object Explorer with the "Tables (10)" node expanded, showing tables like bill_detail and bill_product. The main pane shows a query window with the SQL command "select * from product;". The results are displayed in a Data Output tab, showing 34 rows of product data. The columns include prod_id, prod_name, prod_image, prod_quantity, prod_price, rating_stars, rating_count, prod_discount, and prod_keywords.

prod_id	prod_name	prod_image	prod_quantity	prod_price	rating_stars	rating_count	prod_discount	prod_keywords
1	51 Breville Barista Express Es...	https://m.media-amaz...	10	249.00	4	20	17.00	["kitchen", "accessories"]
2	10 Waterproof Knit Athletic S...	https://m.media-amaz...	80	33.90	4	89	10.00	["shoes", "running shoes", "footwear"]
3	4 2 Slot Toaster - Black	https://m.media-amaz...	100	18.99	5	2197	18.00	["toaster", "kitchen", "appliances"]
4	14 Blackout Curtains Set 4-P...	https://m.media-amaz...	100	45.99	4	232	9.00	["bedroom", "curtains", "home"]
5	16 Electric Glass and Steel H...	https://m.media-amaz...	100	30.74	5	846	7.00	["water boiler", "appliances", "kitchen"]
6	18 Straw Lifeguard Sun Hat	https://m.media-amaz...	100	22.00	4	215	5.00	["hats", "straw hats", "summer", "apparel"]
7	19 Sterling Silver Sky Flower...	https://m.media.amaz...	100	17.99	4	52	3.00	["jewelry", "accessories", "womens"]
8	21 Bathroom Bath Rug Mat 2...	https://m.media.amaz...	100	12.50	4	119	15.00	["bathmat", "bathroom", "home"]
9	3 Adult Plain Cotton T-Shirt...	https://m.media.amaz...	100	7.99	4	56	9.00	["tshirts", "apparel", "mens"]
10	12 Round Sunglasses	https://m.media.amaz...	50	15.60	4	30	5.00	["accessories", "shades", "apparel", "mens"]
11	1 Black and Gray Athletic Co...	https://m.media.amaz...	100	10.90	4	87	10.00	["socks", "sports", "apparel"]
12	5 6 Piece White Dinner Plate...	https://m.media.amaz...	100	20.67	4	37	5.00	["plates", "kitchen", "dining"]
13	7 Plain Hooded Fleece Swea...	https://m.media.amaz...	100	24.00	4	317	7.00	["hoodies", "sweaters", "apparel"]
14	15 Men's Slim-Fit Summer Sh...	https://m.media.amaz...	100	16.99	4	160	12.00	["shorts", "apparel", "mens"]
15	6 6-Piece Nonstick, Carbon...	https://m.media.amaz...	100	34.99	4	175	9.00	["kitchen", "cookware"]

Shopping_cart Page

The screenshot shows the pgAdmin 4 interface with the title "Shopping_cart Page". The left sidebar displays the Object Explorer with the "Tables (10)" node expanded, showing tables like bill_detail and bill_product. The main pane shows a query window with the SQL command "select * from shopping_cart;". The results are displayed in a Data Output tab, showing 2 rows of shopping cart data. The columns include cart_id, prod_id, and user_id.

cart_id	prod_id	user_id
1	107	10
2	108	4

Orders Page

pgAdmin 4

File Object Tools Edit View Window Help

Object Explorer Servers(1) PostgreSQL 17 Databases(2) shopmanagementsystem

Query History

Execute query Alt F5

select * from orders;

Data Output Messages Notifications

order_id [PK] integer	order_date date	user_id integer	delivery_address character varying(50)	total_amt numeric(10,2)	order_status character varying(20)
1	2025-01-23	1	123 Main St	47.65	In process
2	2025-01-23	1	123 Main St	47.65	In process
3	2025-01-23	1	123 Main St	47.65	In process
4	2025-01-23	1	123 Main St	47.65	In process
5	2025-01-23	1	123 Main St	47.65	In process
6	2025-01-23	1	123 Main St	26.98	In process
7	2025-01-23	2	123 Main St	44.95	In process
8	2025-01-23	2	123 Main St	44.95	In process
9	2025-01-23	2	123 Main St	44.95	In process
10	2025-01-23	2	123 Main St	44.95	In process
11	2025-01-23	2	123 Main St	44.95	In process
12	2025-01-23	2	123 Main St	44.95	In process
13	2025-01-23	2	123 Main St	44.95	In process
14	2025-01-23	2	123 Main St	44.95	In process
15	2025-01-23	2	123 Main St	44.95	In process

Showing rows: 1 to 84 Page No: 1 of 1

Total rows: 84 Query complete 00:00:00.112 CRLF Ln 1, Col 1

Successfully run. Total query runtime: 112 msec. 84 rows affected.

421 PM 3/8/2025

My Shopping Store localhost:3000/update-order-status Admin Dashboard Live

Update Order Status

Change the status of an existing order.

Order ID	Order Date	User ID	User Address	Total Amount	Order Status	Actions
8	2025-01-22T18:00:00.000Z	1	123 Main St	\$47.65	In process	<button>Change to delivered</button>
9	2025-01-22T18:00:00.000Z	1	123 Main St	\$47.65	In process	<button>Change to delivered</button>
13	2025-01-22T18:00:00.000Z	1	123 Main St	\$47.65	In process	<button>Change to delivered</button>
14	2025-01-22T18:00:00.000Z	1	123 Main St	\$47.65	In process	<button>Change to delivered</button>
15	2025-01-22T18:00:00.000Z	1	123 Main St	\$47.65	In process	<button>Change to delivered</button>

17% 6:33 AM 3/8/2025

Order_details Page

The screenshot shows the pgAdmin 4 interface with the title "Order_details Page". The left sidebar displays the Object Explorer with the "ShopManagementSystem" database selected. The main pane shows a query window with the SQL command "select * from order_detail;". The results are displayed in a Data Output tab, showing 182 rows of data from the "order_detail" table. The table has columns: order_id [PK] integer, prod_id [PK] integer, prod_qty integer, prod_price double precision, and prod_total_price double precision. The data shows various product details and their quantities and prices. A message at the bottom right indicates the query was successfully run with a runtime of 112 msec and 182 rows affected.

order_id [PK]	prod_id [PK]	prod_qty	prod_price	prod_total_price
1	1	5	1	20.67
2	1	3	1	7.99
3	2	5	1	20.67
4	2	3	1	7.99
5	2	4	1	18.99
6	3	5	1	20.67
7	3	3	1	7.99
8	3	4	1	18.99
9	4	5	1	20.67
10	4	3	1	7.99
11	4	4	1	18.99
12	5	5	1	20.67
13	5	3	1	7.99
14	5	4	1	18.99
15	6	5	1	20.67

Order_return Page

The screenshot shows the pgAdmin 4 interface with the title "Order_return Page". The left sidebar displays the Object Explorer with the "ShopManagementSystem" database selected. The main pane shows a query window with the SQL command "select * from order_return;". The results are displayed in a Data Output tab, showing 10 rows of data from the "order_return" table. The table has columns: order_return_id [PK] integer, order_id integer, user_id integer, return_date timestamp without time zone, prod_id integer, and return_amount double precision. The data shows various return records with their dates, product IDs, and amounts. A message at the bottom right indicates the query was successfully run with a runtime of 75 msec and 10 rows affected.

order_return_id [PK]	order_id	user_id	return_date	prod_id	return_amount
1	1	25	2025-03-08 05:46:32.930709	3	42.88
2	5	26	2025-03-08 06:05:07.219678	3	42.88
3	6	57	2025-03-08 06:12:40.241859	2	26.05
4	7	27	2025-03-08 06:19:40.078737	3	7.19
5	8	28	2025-03-08 06:23:17.937687	3	24.28
6	9	60	2025-03-08 06:23:58.434754	2	69.2
7	10	69	2025-03-08 06:25:57.162669	15	24.29
8	11	58	2025-03-08 06:32:06.297785	2	56.57
9	12	65	2025-03-08 07:32:49.848723	2	27.86
10	13	59	2025-03-08 15:56:04.465679	2	50.35

Bill_details Page

The screenshot shows the pgAdmin 4 interface with the title "Bill_details Page". The left pane displays the Object Explorer with a tree view of servers, databases, and tables. The right pane shows a query editor window titled "shopmanagementsystem/postgres@PostgreSQL 17+", containing the SQL query: "select * from bill_detail;". Below the query is a data output grid showing 60 rows of data from the bill_detail table. The bottom status bar indicates "Successfully run. Total query runtime: 173 msec. 60 rows affected." and the system status shows "CRLF Ln 1, Col 27".

billId	bill_date	user_id	order_id	user_name	order_total_price	bill_total_price	pay_status
1	2025-01-23	2	30	Falaz	44.95	44.95	unpaid
2	2025-01-23	1	56	Tauseef	20.95	20.95	paid
3	2025-01-23	2	31	Falaz	44.95	44.95	unpaid
4	2025-01-24	1	57	Tauseef	28.94	28.94	paid
5	2025-01-23	1	27	Tauseef	47.65	47.65	paid
6	2025-01-23	1	28	Tauseef	26.98	26.98	paid
7	2025-01-23	1	25	Tauseef	47.65	47.65	paid
8	2025-01-24	1	58	Tauseef	62.85	62.85	paid
9	2025-01-25	1	60	Tauseef	76.89	76.89	paid
10	2025-01-26	1	61	Tauseef	41.90	41.90	unpaid
11	2025-03-06	2	77	Falaz	275.64	275.64	paid
12	2025-01-23	2	32	Falaz	44.95	44.95	paid
13	2025-01-23	2	33	Falaz	44.95	44.95	unpaid
14	2025-01-23	2	34	Falaz	44.95	44.95	unpaid
15	2025-01-23	2	35	Falaz	44.95	44.95	

Workers Page

pgAdmin 4

File Object Tools Edit View Window Help

Object Explorer

- Servers (1)
 - PostgreSQL 17
 - Databases (2)
 - postgres
 - shopmanagementsystem
- Shopmanagementsystem
 - Casts
 - 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 (10)
 - bill_detail
 - bill_product
 - order_detail

```
1 select * from worker;
```

	worker_id	worker_name	worker_email	worker_contact_no	worker_salary
	[PK] integer	character varying (50)	character varying (50)	character varying (20)	double precision
1	1	Rahul Sharma	rahul.sharma@email.com	9876543210	45000.5
2	2	Aisha Khan	aisha.khan@email.com	8765432109	50000
3	3	David Smith	david.smith@email.com	7654321098	48000.75
4	4	Sakura Tanaka	sakura.tanaka@email.com	6543210987	52000.25
5	5	Mohammed Ali	mohammed.ali@email.com	5432109876	47000
6	6	Emily Johnson	emily.johnson@email.com	4321098765	49000.9
7	7	Chen Wei	chen.wei@email.com	3210987654	51000.6
8	8	Maria Gonzalez	maria.gonzalez@email.com	2109876543	46000.8
9	9	John Doe	john.doe@email.com	1098765432	53000.1
10	10	Aarav Mehta	aarav.mehta@email.com	1987654321	55000.75
11	11	Jimmy	jimmy03@gmail.com	012133413213	50000
12	12	Jack	jack343@gmail.com	0238245254	30000
13	21	bob	bob55@gmail.com	01213341	20000

Total rows: 13 Query complete 00:00:00.143

Successfully run. Total query runtime: 143 msec. 13 rows affected. CRLF Ln 1, Col 22

Admin Dashboard Live

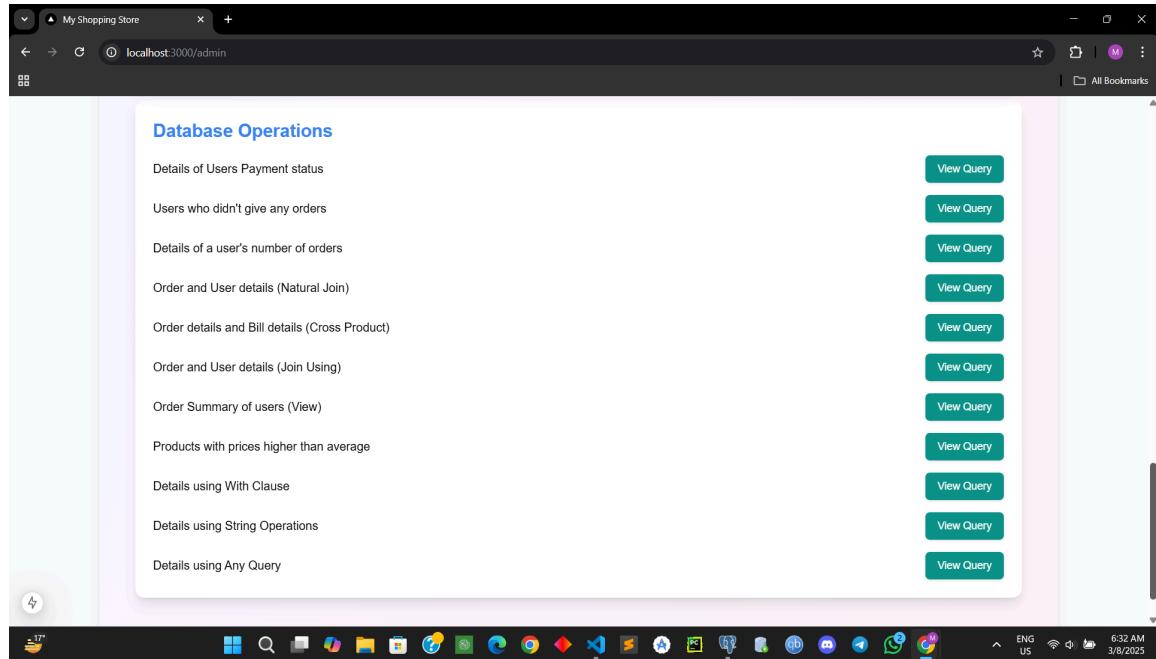


Workers Information

View and manage worker details

Worker ID	Name	Email	Contact Number	Salary
1	Rahul Sharma	rahul.sharma@email.com	9876543210	\$45000.50
2	Aisha Khan	aisha.khan@email.com	8765432109	\$50000.00
3	David Smith	david.smith@email.com	7654321098	\$48000.75
4	Sakura Tanaka	sakura.tanaka@email.com	6543210987	\$52000.25
5	Mohammed Ali	mohammed.ali@email.com	5432109876	\$47000.00
6	Emily Johnson	emily.johnson@email.com	4321098765	\$49000.90
7	Chen Wei	chen.wei@email.com	3210987654	\$51000.60
8	Maria Gonzalez	maria.gonzalez@email.com	2109876543	\$46000.80
9	John Doe	john.doe@email.com	1098765432	\$53000.10
10	Aarav Mehta	aarav.mehta@email.com	1987654321	\$55000.75

7. Query statement, Relational Algebra Expression, SQL statement and snapshots of the outputs



- **Natural Join**

```

const response: QueryResult = await api.executeRawQuery(`

SELECT
    order_id,
    users.user_id,
    users.user_name,
    users.user_email,
    users.user_contact_no,
    orders.delivery_address,
    orders.total_amt,
    orders.order_status
FROM orders
NATURAL JOIN users;

`);

```

pgAdmin 4

File Object Tools Edit View Window Help

Object pgAdmin 4 shopmanagementsystem/postgres@PostgreSQL 17*

Servers (1) PostgreSQL 17 Databases (2) shopmanagementsystem

Tables (1) orders Schemas (1) public

Aggregates (0) Collations (0) Domains (0) FTS Configurations (0) Foreign Tables (0) Functions (0) Materials (0) Operators (0) Procedures (0) Sequences (0) Tables (2) bill_bill

Extensions (0) Event Triggers (0) Languages (0) Publications (0) Schemas (1)

CASTS CATALOGS FOREIGN DATA LANGUAGES PUBLICATIONS SCHEMAS

Query History

Query SQL

Showing rows: 1 to 84 Page No: 1 of 1 | 14 <> >>

	order_id	user_id	user_name	user_email	user_contact_no	delivery_address	total_amt	order_status
1	13	1	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	In process
2	23	1	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	In process
3	24	1	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	In process
4	26	1	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	In process
5	27	1	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	In process
6	28	1	Tauseef	abc01@gmail.com	01852039838	123 Main St	26.98	In process
7	31	2	Faiyaz	abc02@gmail.com	01222222222	123 Main St	44.95	In process
8	32	2	Faiyaz	abc02@gmail.com	01222222222	123 Main St	44.95	In process
9	40	2	Faiyaz	abc02@gmail.com	01222222222	123 Main St	44.95	In process
10	43	2	Faiyaz	abc02@gmail.com	01222222222	123 Main St	44.95	In process

Total rows: 84 Query complete 00:00:00.098

Successfully run. Total query runtime: 98 msec. 84 rows affected.

CRLF Ln 11, Col 20

Natural Join Query Results

Viewing combined data with matching records

Filter Results

Search Product Name Min Price Max Price

Min Quantity Max Quantity Min Rating

ORDER_ID	USER_ID	USER_NAME	USER_EMAIL	USER_CONTACT_NO	DELIVERY_ADDRESS	TOTAL_AMT	ORDER_STATUS
13	1	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	In process
14	1	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	In process
15	1	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	In process
16	1	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	In process
19	1	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	In process
23	1	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	In process
24	1	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	In process

- *Cross Product*

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure under 'Servers (1)'. A connection to 'shopmanagementsystem/postgres@PostgreSQL 17*' is selected. The main area shows a 'Query' tab with the following SQL code:

```
1 SELECT *
2 FROM order_detail o, bill_detail b
3 WHERE o.order_id = b.order_id;
```

Below the query, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, displaying the results of the query in a table format. The table has the following columns:

	order_id	prod_id	prod_qty	prod_price	prod_total_price	bill_id	bill_date	user_id	order_id	user_name	order_total_price	bill_total_price	pay_status
1	25	5	1	20.67	20.67	1	2025-01-23	1	25	Tauseef	47.65	47.65	paid
2	25	3	1	7.99	7.99	1	2025-01-23	1	25	Tauseef	47.65	47.65	paid
3	25	4	1	18.99	18.99	1	2025-01-23	1	25	Tauseef	47.65	47.65	paid
4	26	5	1	20.67	20.67	2	2025-01-23	1	26	Tauseef	47.65	47.65	paid
5	26	3	1	7.99	7.99	2	2025-01-23	1	26	Tauseef	47.65	47.65	paid
6	26	4	1	18.99	18.99	2	2025-01-23	1	26	Tauseef	47.65	47.65	paid
7	27	5	1	20.67	20.67	3	2025-01-23	1	27	Tauseef	47.65	47.65	paid
8	27	3	1	7.99	7.99	3	2025-01-23	1	27	Tauseef	47.65	47.65	paid
9	27	4	1	18.99	18.99	3	2025-01-23	1	27	Tauseef	47.65	47.65	paid
10	28	3	1	7.99	7.99	4	2025-01-23	1	28	Tauseef	26.98	26.98	paid
11	28	4	1	18.99	18.99	4	2025-01-23	1	28	Tauseef	26.98	26.98	paid
12	29	2	1	20.95	20.95	5	2025-01-23	2	29	Faiyaz	158.29	158.29	paid
13	29	7	4	24	96	5	2025-01-23	2	29	Faiyaz	158.29	158.29	paid
14	29	5	2	20.67	41.34	5	2025-01-23	2	29	Faiyaz	158.29	158.29	paid
15	30	2	1	20.95	20.95	6	2025-01-23	2	30	Faiyaz	44.95	44.95	unpaid

Total rows: 111 Query complete 00:00:00.068

All Bookmarks

Cross Product Query Results

Viewing combined data from orders and bill details

Filter Results

ORDER_ID	PROD_ID	PROD_QTY	PROD_PRICE	PROD_TOTAL_PRICE	BILL_ID	BILL_DATE	USER_ID	USER_NAME	ORDER_TOTAL_PRICE	BILL_T
25	5	1	20.67	20.67	1	2025-01-22T18:00:00.000Z	1	Tauseef	47.65	47.65
25	3	1	7.99	7.99	1	2025-01-22T18:00:00.000Z	1	Tauseef	47.65	47.65
25	4	1	18.99	18.99	1	2025-01-22T18:00:00.000Z	1	Tauseef	47.65	47.65
26	5	1	20.67	20.67	2	2025-01-22T18:00:00.000Z	1	Tauseef	47.65	47.65

● Outer Join

pgAdmin 4

File Object Tools Edit View Window Help

Servers (1) PostgreSQL 17 Databases (2) shopmanagementsystem/postgres@PostgreSQL 17*

Query Query History

```
1 SELECT u.user_id, u.user_name, u.user_email, u.user_contact_no
2 FROM users u
3 LEFT JOIN orders o ON u.user_id = o.user_id
4 WHERE o.user_id IS NULL;
```

Data Output Messages Notifications

	user_id	user_name	user_email	user_contact_no
1	7	Fahim	abc03@gmail.com	12345678901

Total rows: 1 Query complete 00:00:00.098 Showing rows: 1 to 1 Page No: 1 of 1

Successfully run. Total query runtime: 98 msec. 1 rows affected. CRLF Ln 4, Col 25

localhost:3000/query-results/outer-join

Outer Join Query Results

Users who didn't give any orders

Filter Results

Search User ID: Search User Name:

USER_ID	USER_NAME	USER_EMAIL	USER_CONTACT_NO
7	Fahim	abc03@gmail.com	12345678901

Total rows: 1

● **Join with USING**

pgAdmin 4

File Object Tools Edit View Window Help

Object Servers Databases(2) postgres shopmanagementsystem/postgres@PostgreSQL 17*

Query History

```

1 SELECT
2   order_id,
3   users.user_id,
4   users.user_name,
5   users.user_email,
6   users.user_contact_no,
7   orders.delivery_address,
8   orders.total_amt,
9   orders.order_status
10 FROM orders
11 JOIN users
12 USING(user_id);

```

Data Output Messages Notifications

order_id	user_id	user_name	user_email	user_contact_no	delivery_address	total_amt	order_status
1	13	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	in process
2	23	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	in process
3	24	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	in process
4	26	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	in process
5	27	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	in process
6	28	Tauseef	abc01@gmail.com	01852039838	123 Main St	26.98	in process
7	31	Faiaz	abc02@gmail.com	01222222222	123 Main St	44.95	in process
8	32	Faiaz	abc02@gmail.com	01222222222	123 Main St	44.95	in process
9	40	Faiaz	abc02@gmail.com	01222222222	123 Main St	44.95	in process
10	43	Faiaz	abc02@gmail.com	01222222222	123 Main St	44.95	in process

Total rows: 84 Query complete 00:00:00.083 CRLF Ln 12, Col 16

Windows Taskbar: File Explorer, Edge, Chrome, FileZilla, Notepad, Task Manager, etc.

All Bookmarks

Join USING Query Results

Viewing data joined using common column names

Filter Results

ORDER ID	USER ID	USER NAME	USER EMAIL	USER CONTACT NO	DELIVERY ADDRESS	TOTAL AMT	ORDER STATUS
13	1	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	in process
14	1	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	in process
15	1	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	in process
16	1	Tauseef	abc01@gmail.com	01852039838	123 Main St	47.65	in process

Windows Taskbar: File Explorer, Edge, Chrome, FileZilla, Notepad, Task Manager, etc.

- **Join with ON**

pgAdmin 4

File Object Tools Edit View Window Help

Servers (1) PostgreSQL 17 Databases (2) shopmanagementsystem

Query History

```
1 ✓ SELECT *
2   FROM order_detail od
3   JOIN product p
4   ON od.prod_id = p.prod_id;
```

Data Output Messages Notifications

order_id	prod_id	prod_qty	prod_price	prod_total_price	prod_id	prod_name	prod_image	prod_quantity	prod_price	rating_stars	rating_count	prod_discount	prod_keywords	jsonb
1	1	5	1	20.67	20.67	5	6 Piece Wh...	https://m.med...	100	20.67	4	37	5.00	["plates", "kitc
2	1	3	1	7.99	7.99	3	Adults Plai...	https://m.med...	100	7.99	4	56	9.00	["tshirts", "app
3	2	5	1	20.67	20.67	5	6 Piece Wh...	https://m.med...	100	20.67	4	37	5.00	["plates", "kitc
4	2	3	1	7.99	7.99	3	Adults Plai...	https://m.med...	100	7.99	4	56	9.00	["tshirts", "app
5	2	4	1	18.99	18.99	4	2 Slot Toas...	https://m.med...	100	18.99	5	2197	18.00	["toaster", "kitc
6	3	5	1	20.67	20.67	5	6 Piece Wh...	https://m.med...	100	20.67	4	37	5.00	["plates", "kitc
7	3	3	1	7.99	7.99	3	Adults Plai...	https://m.med...	100	7.99	4	56	9.00	["tshirts", "app
8	3	4	1	18.99	18.99	4	2 Slot Toas...	https://m.med...	100	18.99	5	2197	18.00	["toaster", "kitc
9	4	5	1	20.67	20.67	5	6 Piece Wh...	https://m.med...	100	20.67	4	37	5.00	["plates", "kitc
10	4	3	1	7.99	7.99	3	Adults Plai...	https://m.med...	100	7.99	4	56	9.00	["tshirts", "app
11	4	4	1	18.99	18.99	4	2 Slot Toas...	https://m.med...	100	18.99	5	2197	18.00	["toaster", "kitc
12	5	5	1	20.67	20.67	5	6 Piece Wh...	https://m.med...	100	20.67	4	37	5.00	["plates", "kitc
13	5	3	1	7.99	7.99	3	Adults Plai...	https://m.med...	100	7.99	4	56	9.00	["tshirts", "app
14	5	4	1	18.99	18.99	4	2 Slot Toas...	https://m.med...	100	18.99	5	2197	18.00	["toaster", "kitc
15	6	5	1	20.67	20.67	5	6 Piece Wh...	https://m.med...	100	20.67	4	37	5.00	["plates", "kitc

Total rows: 182 Query complete 00:00:00.133

CRLF Ln 4, Col 27

ENG INTL 7:52 PM 3/8/2025

- **Nested subquery with ANY clause**

pgAdmin 4

File Object Tools Edit View Window Help

Servers (1) PostgreSQL 17 Databases (2) postgres shopmanagementsystem

```

SELECT u.user_id, u.user_name, u.user_email,
       (SELECT COUNT(*) FROM bill_detail b WHERE b.user_id = u.user_id) AS order_count
FROM users u
WHERE u.user_id = ANY (
    SELECT user_id
    FROM bill_detail
    GROUP BY user_id
    HAVING COUNT(order_id) > 27
);

```

Data Output Messages Notifications

	user_id	user_name	user_email	order_count
1	1	Tauseef	abc01@gmail.com	31
2	2	Falaz	abc02@gmail.com	29

Total rows: 2 Query complete 00:00:00.094

Successfully run. Total query runtime: 94 msec. 2 rows affected.

CRLF Ln 9, Col 3

Nested ANY Query Results

Details of a user's number of orders

Filter Results

Showing 2 results

USER_ID	USER_NAME	USER_EMAIL	ORDER_COUNT
1	Tauseef	abc01@gmail.com	30
2	Falaz	abc02@gmail.com	29

3/8/2025

● *String Operations*

pgAdmin 4

File Object Tools Edit View Window Help

Object shopmanagementsystem/postgres@PostgreSQL 17*

shopmanagementsystem/postgres@PostgreSQL 17* ×

Query Query History

```
1 SELECT prod_name,
2        LOWER(prod_name) AS lower_name,
3        UPPER(prod_name) AS upper_name
4   FROM product;
```

Data Output Messages Notifications

prod_name	lower_name	upper_name
Breville Barista Express Espresso Machine, One Size, Brushed Stainless Steel	breville barista express espresso machine, one size, brushed stainless steel	BREVILLE BARISTA EXPRESS ESPRESSO MACHINE, ONE SIZE, BRUSHED STAINLESS STEEL
Waterproof Knit Athletic Sneakers - Gray	waterproof knit athletic sneakers - gray	WATERPROOF KNIT ATHLETIC SNEAKERS - GRAY
2 Slot Toaster - Black	2 slot toaster - black	2 SLOT TOASTER - BLACK
Blackout Curtains Set 4-Pack - Beige	blackout curtains set 4-pack - beige	BLACKOUT CURTAINS SET 4-PACK - BEIGE
Electric Glass and Steel Hot Tea Water Kettle - 1.7-Liter	electric glass and steel hot tea water kettle - 1.7-liter	ELECTRIC GLASS AND STEEL HOT TEA WATER KETTLE - 1.7-LITER
Straw Lifeguard Sun Hat	straw lifeguard sun hat	STRAW LIFEGUARD SUN HAT
Sterling Silver Sky Flower Stud Earrings	sterling silver sky flower stud earrings	STERLING SILVER SKY FLOWER STUD EARRINGS
Bathroom Bath Rug Mat 20 x 31 Inch - Grey	bathroom bath rug mat 20 x 31 inch - grey	BATHROOM BATH RUG MAT 20 X 31 INCH - GREY
Adults Plain Cotton T-Shirt - 2 Pack	adults plain cotton t-shirt - 2 pack	ADULTS PLAIN COTTON T-SHIRT - 2 PACK

Total rows: 34 | Query complete 00:00:00.094 | ✓ Successfully run. Total query runtime: 94 msec. 34 rows affected.

CRLF Ln 4, Col 14 | 7:50 PM | 3/8/2025 | ENG INTL

All Bookmarks

String Operations Results

Case transformations on product names

Filter Results

Search Product Name: breville barista express espresso machine, one size, brushed stainless steel

Search Lowercase Name: breville barista express espresso machine, one size, brushed stainless steel

Search Uppercase Name: BREVILLE BARISTA EXPRESS ESPRESSO MACHINE, ONE SIZE, BRUSHED STAINLESS STEEL

PROD_NAME	LOWER_NAME	UPPER_NAME
Breville Barista Express Espresso Machine, One Size, Brushed Stainless Steel	breville barista express espresso machine, one size, brushed stainless steel	BREVILLE BARISTA EXPRESS ESPRESSO MACHINE, ONE SIZE, BRUSHED STAINLESS STEEL
Waterproof Knit Athletic Sneakers - Gray	waterproof knit athletic sneakers - gray	WATERPROOF KNIT ATHLETIC SNEAKERS - GRAY
2 Slot Toaster - Black	2 slot toaster - black	2 SLOT TOASTER - BLACK
Blackout Curtains Set 4-Pack - Beige	blackout curtains set 4-pack - beige	BLACKOUT CURTAINS SET 4-PACK - BEIGE
Electric Glass and Steel Hot Tea Water Kettle - 1.7-Liter	electric glass and steel hot tea water kettle - 1.7-liter	ELECTRIC GLASS AND STEEL HOT TEA WATER KETTLE - 1.7-LITER
Straw Lifeguard Sun Hat	straw lifeguard sun hat	STRAW LIFEGUARD SUN HAT
Sterling Silver Sky Flower Stud Earrings	sterling silver sky flower stud earrings	STERLING SILVER SKY FLOWER STUD EARRINGS
Bathroom Bath Rug Mat 20 x 31 Inch - Grey	bathroom bath rug mat 20 x 31 inch - grey	BATHROOM BATH RUG MAT 20 X 31 INCH - GREY

3:46 PM | 3/8/2025 | ENG INTL

● With Clause

pgAdmin 4

File Object Tools Edit View Window Help

Object shopmanagementsystem/postgres@PostgreSQL 17*

Query Query History

```

1 v WITH high_rated_products AS (
2   SELECT prod_id, prod_name, prod_price, rating_stars, rating_count
3     FROM product
4    WHERE rating_stars >= 4
5  )
6   SELECT *
7     FROM high_rated_products;

```

Data Output Messages Notifications

	prod_id	prod_name	prod_price	rating_stars	rating_count
1	51	Breville Barista Express Espresso Machine, One Size, Brushed Stainless Steel	249.00	4	20
2	10	Waterproof Knit Athletic Sneakers - Gray	33.90	4	89
3	4	2 Slot Toaster - Black	18.99	5	2197
4	14	Blackout Curtains Set 4-Pack - Beige	45.99	4	232
5	16	Electric Glass and Steel Hot Tea Water Kettle - 1.7-Liter	30.74	5	846
6	18	Straw Lifeguard Sun Hat	22.00	4	215
7	19	Sterling Silver Sky Flower Stud Earrings	17.99	4	52
8	21	Bathroom Bath Rug Mat 20 x 31 Inch - Grey	12.50	4	119
9	3	Adults Plain Cotton T-Shirt - 2 Pack	7.99	4	66
10	12	Round Sunglasses	15.60	4	66

Total rows: 32 | Query complete 00:00:00.095 | Successfully run. Total query runtime: 95 msec. 32 rows affected. CRLF Ln 7, Col 26

7:57 PM 3/8/2025

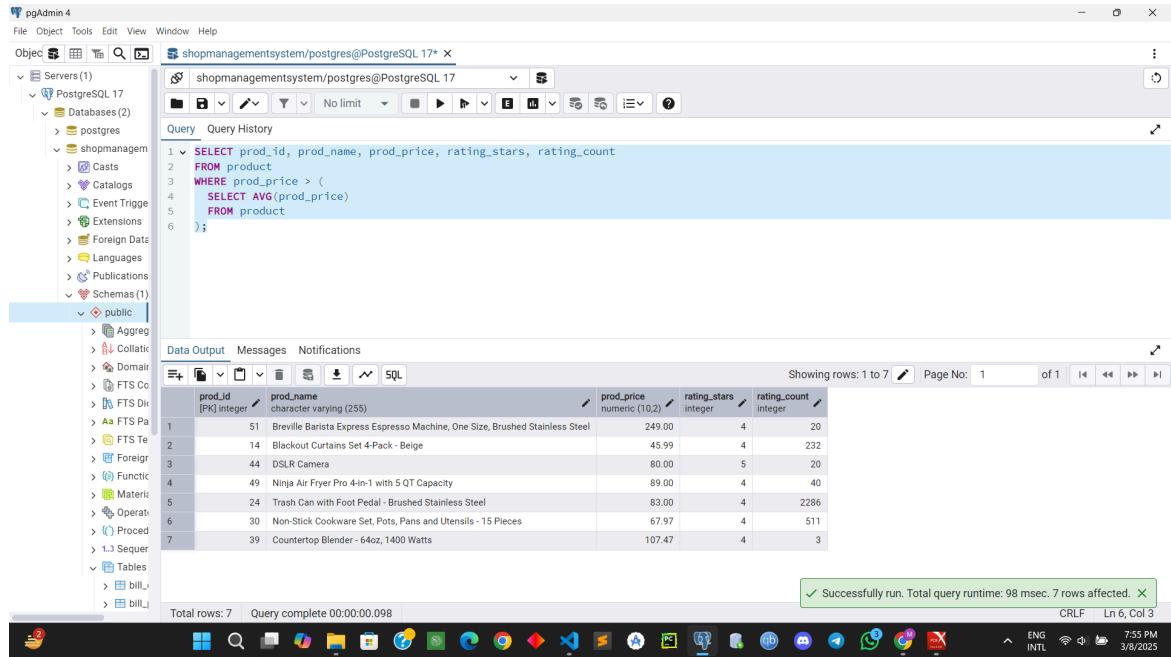
WITH Clause Query Results

Viewing high-rated products using WITH clause

Filter Results

PROD_ID	PROD_NAME	PROD_PRICE	RATING_STARS	RATING_COUNT
51	Breville Barista Express Espresso Machine, One Size, Brushed Stainless Steel	249.00	4	20
10	Waterproof Knit Athletic Sneakers - Gray	33.90	4	89
4	2 Slot Toaster - Black	18.99	5	2197
14	Blackout Curtains Set 4-Pack - Beige	45.99	4	232
16	Electric Glass and Steel Hot Tea Water Kettle - 1.7-Liter	30.74	5	846

- **Nested subquery in from and select clause**



The screenshot shows the pgAdmin 4 interface with a database connection to 'shopmanagementsystem' on 'PostgreSQL 17'. In the left sidebar, under 'Schemas(1)', the 'public' schema is selected. In the main query editor, the following SQL code is displayed:

```

1 SELECT prod_id, prod_name, prod_price, rating_stars, rating_count
2 FROM product
3 WHERE prod_price > (
4   SELECT AVG(prod_price)
5   FROM product
6 );

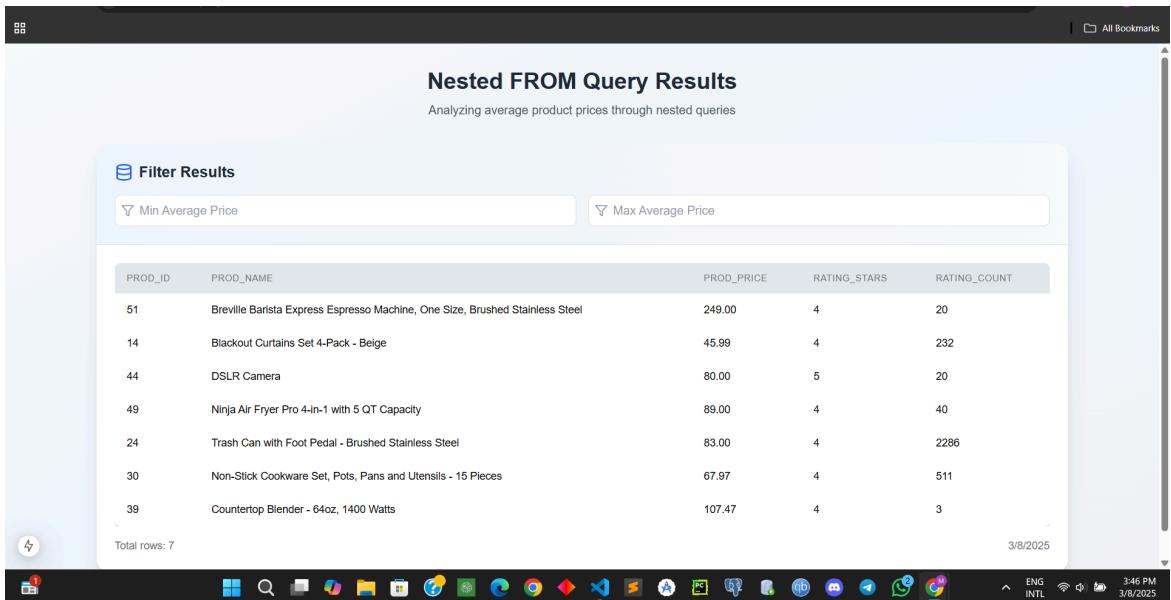
```

The results are shown in a table titled 'Data Output' with the following data:

prod_id	prod_name	prod_price	rating_stars	rating_count
51	Breville Barista Express Espresso Machine, One Size, Brushed Stainless Steel	249.00	4	20
14	Blackout Curtains Set 4-Pack - Beige	45.99	4	232
44	DSLR Camera	80.00	5	20
49	Ninja Air Fryer Pro 4-in-1 with 5 QT Capacity	89.00	4	40
24	Trash Can with Foot Pedal - Brushed Stainless Steel	83.00	4	2286
30	Non-Stick Cookware Set, Pots, Pans and Utensils - 15 Pieces	67.97	4	511
39	Countertop Blender - 64oz, 1400 Watts	107.47	4	3

Total rows: 7 Query complete 00:00:00.098

Message bar: ✓ Successfully run. Total query runtime: 98 msec. 7 rows affected.



Nested FROM Query Results

Analyzing average product prices through nested queries

Filter Results

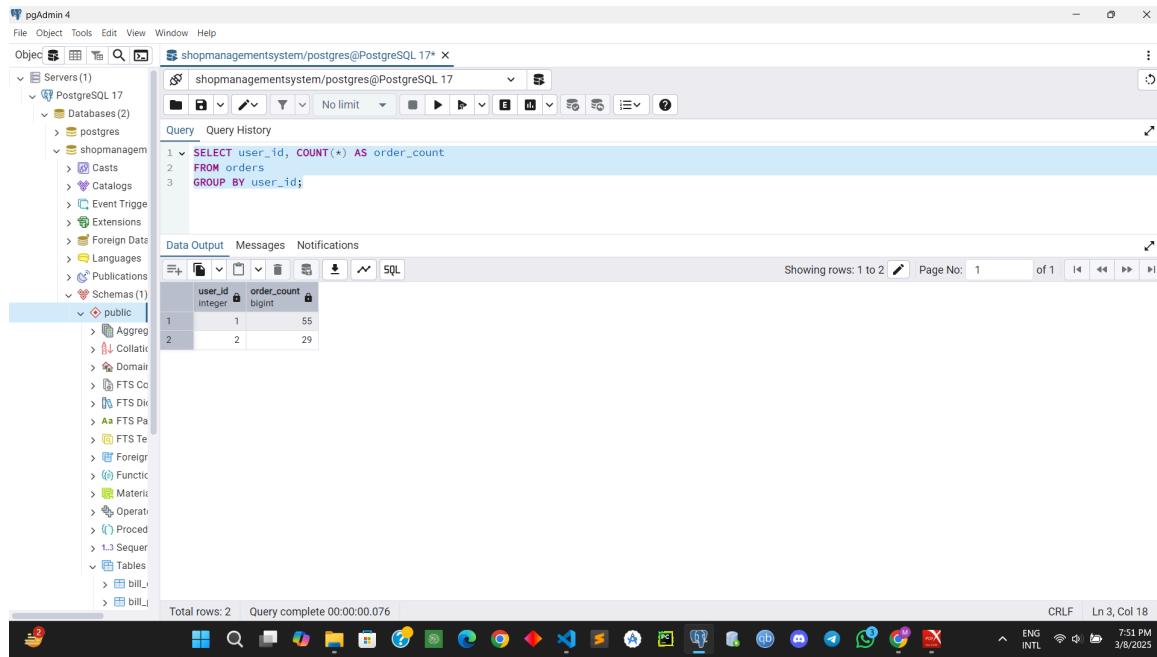
Min Average Price: _____ Max Average Price: _____

PROD_ID	PROD_NAME	PROD_PRICE	RATING_STARS	RATING_COUNT
51	Breville Barista Express Espresso Machine, One Size, Brushed Stainless Steel	249.00	4	20
14	Blackout Curtains Set 4-Pack - Beige	45.99	4	232
44	DSLR Camera	80.00	5	20
49	Ninja Air Fryer Pro 4-in-1 with 5 QT Capacity	89.00	4	40
24	Trash Can with Foot Pedal - Brushed Stainless Steel	83.00	4	2286
30	Non-Stick Cookware Set, Pots, Pans and Utensils - 15 Pieces	67.97	4	511
39	Countertop Blender - 64oz, 1400 Watts	107.47	4	3

Total rows: 7

3/8/2025

● Group By



The screenshot shows the pgAdmin 4 interface with a database connection to 'shopmanagementsystem/postgres@PostgreSQL 17'. In the left sidebar, under 'Schemas (1) / public / Tables', the 'orders' table is selected. A query window displays the following SQL code:

```

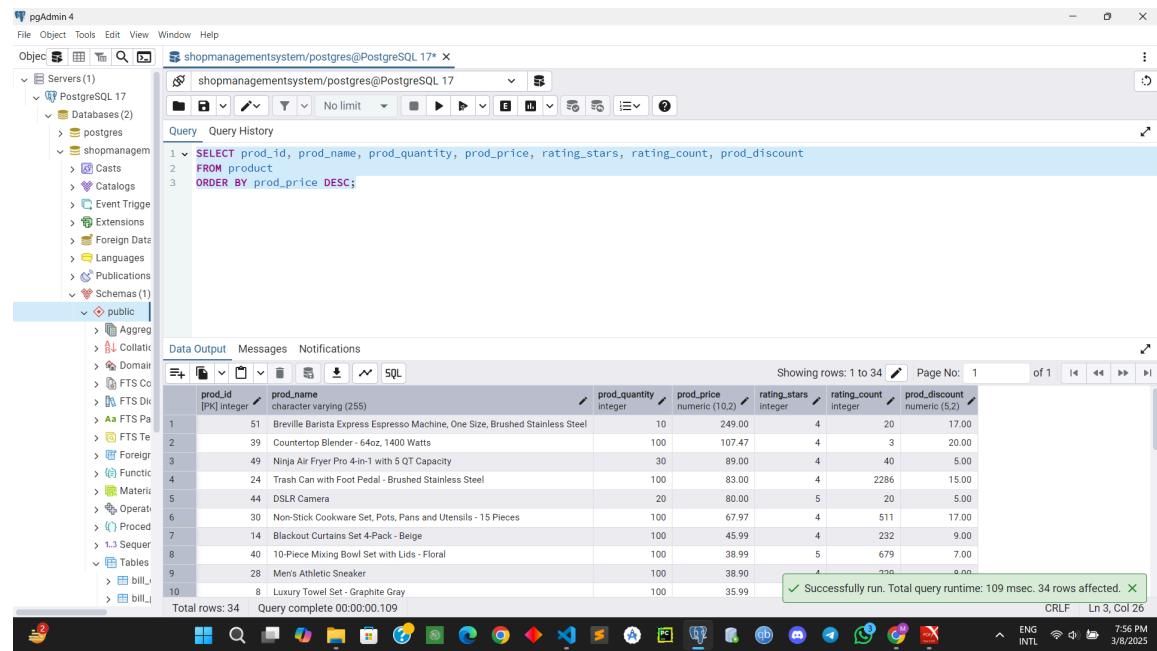
1 v SELECT user_id, COUNT(*) AS order_count
2   FROM orders
3   GROUP BY user_id;
    
```

The 'Data Output' tab shows the results of the query:

user_id	order_count
1	55
2	29

Total rows: 2 Query complete 00:00:00.076

● Order By



The screenshot shows the pgAdmin 4 interface with a database connection to 'shopmanagementsystem/postgres@PostgreSQL 17'. In the left sidebar, under 'Schemas (1) / public / Tables', the 'product' table is selected. A query window displays the following SQL code:

```

1 v SELECT prod_id, prod_name, prod_quantity, prod_price, rating_stars, rating_count, prod_discount
2   FROM product
3   ORDER BY prod_price DESC;
    
```

The 'Data Output' tab shows the results of the query:

prod_id	prod_name	prod_quantity	prod_price	rating_stars	rating_count	prod_discount
51	Breville Barista Express Espresso Machine, One Size, Brushed Stainless Steel	10	249.00	4	20	17.00
39	CounterTop Blender - 64oz, 1400 Watts	100	107.47	4	3	20.00
49	Ninja Air Fryer Pro 4-in-1 with 5 QT Capacity	30	89.00	4	40	5.00
24	Trash Can with Foot Pedal - Brushed Stainless Steel	100	83.00	4	2286	15.00
44	DSLR Camera	20	80.00	5	20	5.00
30	Non-Stick Cookware Set, Pots, Pans and Utensils - 15 Pieces	100	67.97	4	511	17.00
14	Blackout Curtains Set 4-Pack - Beige	100	45.99	4	232	9.00
40	10-Piece Mixing Bowl Set with Lids - Floral	100	38.99	5	679	7.00
28	Men's Athletic Sneaker	100	38.90	4	2286	9.00
8	Luxury Towel Set - Graphite Gray	100	35.99			

Total rows: 34 Query complete 00:00:00.109

Successfully run. Total query runtime: 109 msec. 34 rows affected.

- **Having clause**

The screenshot shows the pgAdmin 4 interface. On the left, the object browser displays a tree structure of databases, schemas, and tables. In the center, a query editor window contains the following SQL code:

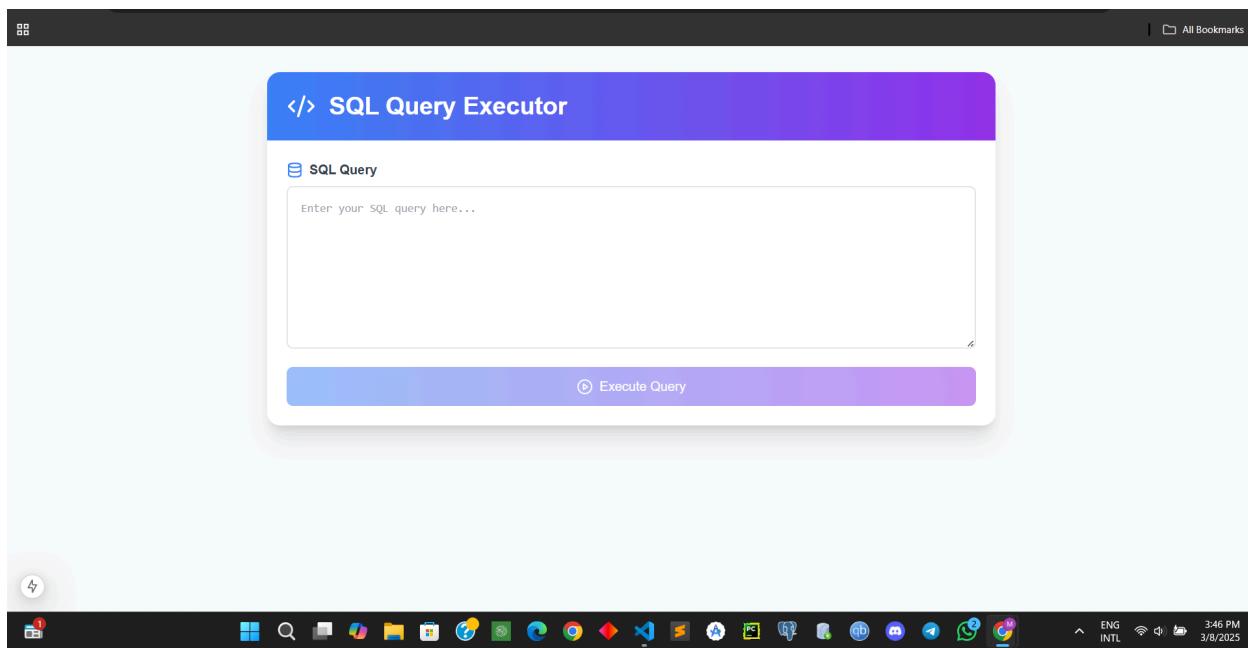
```
1 v SELECT user_id, COUNT(*) AS order_count
2   FROM orders
3  GROUP BY user_id
4 HAVING COUNT(*) > 5;
```

Below the query editor is a data grid showing the results of the executed query. The results are:

	user_id	order_count
1	1	55
2	2	29

A status bar at the bottom indicates "Total rows: 2" and "Query complete 00:00:00.106". A message bar at the bottom right says "Successfully run. Total query runtime: 106 msec. 2 rows affected." The system tray at the bottom right shows the date and time as 3/8/2025, 7:51 PM.

- **Provide any query :**



- ***Update function***

```

CREATE OR REPLACE FUNCTION update_bill_detail_user_name()
RETURNS TRIGGER AS $$

BEGIN
    UPDATE bill_detail
    SET user_name = NEW.user_name
    WHERE user_id = NEW.user_id;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

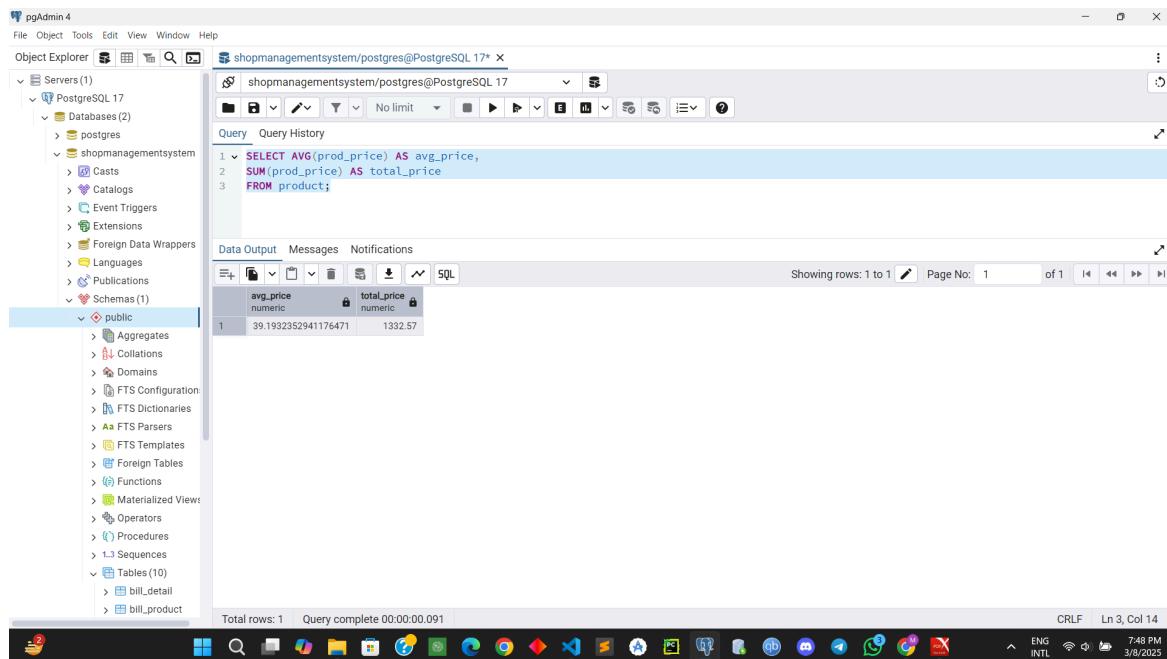
```

- ***Aggregate functions or built-in functions and other functions***

```

CREATE TRIGGER trigger_update_bill_user_name
AFTER UPDATE ON users
FOR EACH ROW
WHEN (OLD.user_name IS DISTINCT FROM NEW.user_name)
EXECUTE FUNCTION update_bill_detail_user_name();

```



Relational Algebra Expressions :

1. For the CREATE VIEW and subsequent SELECT:

order_summary = π all columns (σ user_id=1 (orders \bowtie users \bowtie order_detail \bowtie product))

Where the join conditions are:

- orders.user_id = users.user_id
- orders.order_id = order_detail.order_id
- order_detail.prod_id = product.prod_id

2. For SELECT with string functions:

π prod_name, LOWER(prod_name) \rightarrow lower_name, UPPER(prod_name) \rightarrow upper_name (product)

3. For WITH clause (Common Table Expression):

high_rated_products = π prod_id, prod_name, prod_price, rating_stars, rating_count (σ rating_stars \geq 4 (product))

4. For LEFT JOIN with NULL check:

π u.user_id, u.user_name, u.user_email, u.user_contact_no (users \bowtie orders) \bowtie σ o.user_id IS NULL

5. For ORDER BY:

τ prod_price DESC (π prod_id, prod_name, prod_quantity, prod_price, rating_stars, rating_count, prod_discount (product))

6. For subquery with AVG:

σ prod_price > (G AVG(prod_price) (product)) (π prod_id, prod_name, prod_price, rating_stars, rating_count (product))

7. For complex queries with ANY and subquery:

$\pi u.user_id, u.user_name, u.user_email, (G \text{ COUNT}(\star) \rightarrow \text{order_count} (\sigma b.user_id = u.user_id (bill_detail))) (\sigma u.user_id \in (\pi user_id (G user_id; \text{COUNT}(order_id) > 27 (bill_detail))) (\text{users}))$

8. For NATURAL JOIN:

$\pi order_id, users.user_id, users.user_name, users.user_email, users.user_contact_no, orders.delivery_address, orders.total_amt, orders.order_status (orders \bowtie users)$

9. For JOIN USING:

$\pi order_id, users.user_id, users.user_name, users.user_email, users.user_contact_no, orders.delivery_address, orders.total_amt, orders.order_status (orders \bowtie user_id users)$

10. For simple JOIN:

$order_detail \bowtie product$

Where the join condition is: $order_detail.prod_id = product.prod_id$

11. For GROUP BY with HAVING:

$G user_id; \text{COUNT}(\star) > 5 \rightarrow \text{order_count} (\text{orders})$

12. For simple GROUP BY:

$G user_id; \text{COUNT}(\star) \rightarrow \text{order_count} (\text{orders})$

13. For comma join (Cartesian product with condition):

$\sigma o.order_id = b.order_id (order_detail \times bill_detail)$

14. For aggregate functions:

$\pi \text{AVG}(prod_price) \rightarrow avg_price, \text{SUM}(prod_price) \rightarrow total_price (\text{product})$

8. Create views and use those views in answering queries.

View create statement

Here a view named “order_summary” is created.

```
CREATE VIEW order_summary AS
SELECT
    o.order_id,
    o.order_date,
    u.user_id,
    u.user_name,
    u.user_email,
    o.delivery_address,
    od.prod_id,
    p.prod_name,
    od.prod_qty,
    od.prod_price,
    od.prod_total_price,
    o.total_amt,
    o.order_status
FROM orders o
JOIN users u ON o.user_id = u.user_id
JOIN order_detail od ON o.order_id = od.order_id
JOIN product p ON od.prod_id = p.prod_id;
```

View Result

Query Results using View

Viewing combined data with matching records

User ID: 1 Fetch Orders

ORDER_ID	ORDER_DATE	USER_ID	USER_NAME	USER_EMAIL	DELIVERY_ADDRESS	PROD_ID	PROD_NAME	PROD_QTY	PROD_PRICE	PROD
1	2025-01-22T18:00:00.000Z	1	Tauseef	abc01@gmail.com	123 Main St	5	6 Piece White Dinner Plate Set	1	20.67	20.67
1	2025-01-22T18:00:00.000Z	1	Tauseef	abc01@gmail.com	123 Main St	3	Adults Plain Cotton T-Shirt - 2 Pack	1	7.99	7.99
2	2025-01-22T18:00:00.000Z	1	Tauseef	abc01@gmail.com	123 Main St	5	6 Piece White Dinner Plate Set	1	20.67	20.67
2	2025-01-22T18:00:00.000Z	1	Tauseef	abc01@gmail.com	123 Main St	3	Adults Plain Cotton T-Shirt - 2 Pack	1	7.99	7.99

pgAdmin 4

File Object Tools Edit View Window Help

shopmanagementsystem/postgres@PostgreSQL 17*

Object Databases (2) Casts Catalogs Event Trigge Extensions Foreign Data Languages Publications Schemas (1) public

Query History Execute script

1 SELECT * FROM order_summary WHERE [F5] 1;

Data Output Messages Notifications

Showing rows: 1 to 123 Page No: 1 of 1

order_id	order_date	user_id	user_name	user_email	delivery_address	prod_id	prod_name
1	2025-01-23	1	Tauseef	abc01@gmail.com	123 Main St	5	6 Piece White Dinner Plate Set
2	1	2025-01-23	Tauseef	abc01@gmail.com	123 Main St	3	Adults Plain Cotton T-Shirt - 2 Pack
3	2	2025-01-23	Tauseef	abc01@gmail.com	123 Main St	5	6 Piece White Dinner Plate Set
4	2	2025-01-23	Tauseef	abc01@gmail.com	123 Main St	3	Adults Plain Cotton T-Shirt - 2 Pack
5	2	2025-01-23	Tauseef	abc01@gmail.com	123 Main St	4	2 Slot Toaster - Black
6	3	2025-01-23	Tauseef	abc01@gmail.com	123 Main St	5	6 Piece White Dinner Plate Set
7	3	2025-01-23	Tauseef	abc01@gmail.com	123 Main St	3	Adults Plain Cotton T-Shirt - 2 Pack
8	3	2025-01-23	Tauseef	abc01@gmail.com	123 Main St	4	2 Slot Toaster - Black
9	4	2025-01-23	Tauseef	abc01@gmail.com	123 Main St		

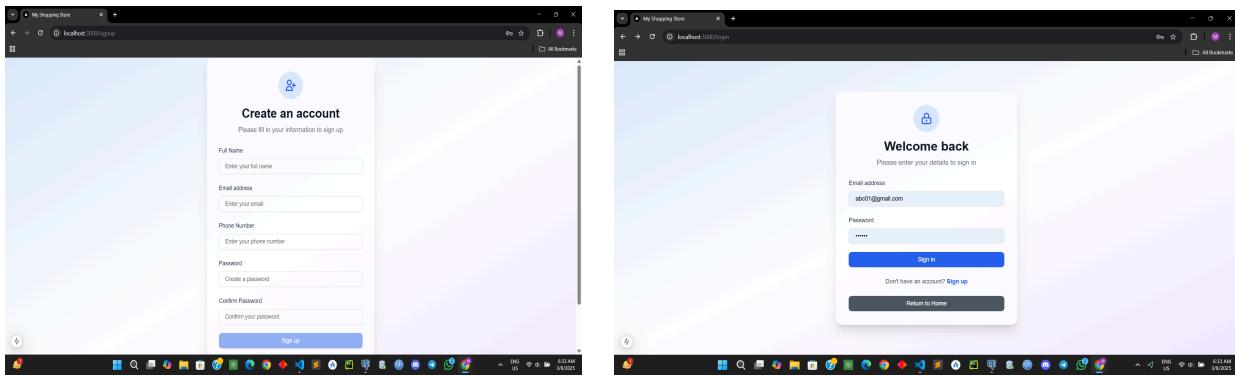
Total rows: 123 Query complete 00:00:00.114

Successfully run. Total query runtime: 114 msec. 123 rows affected.

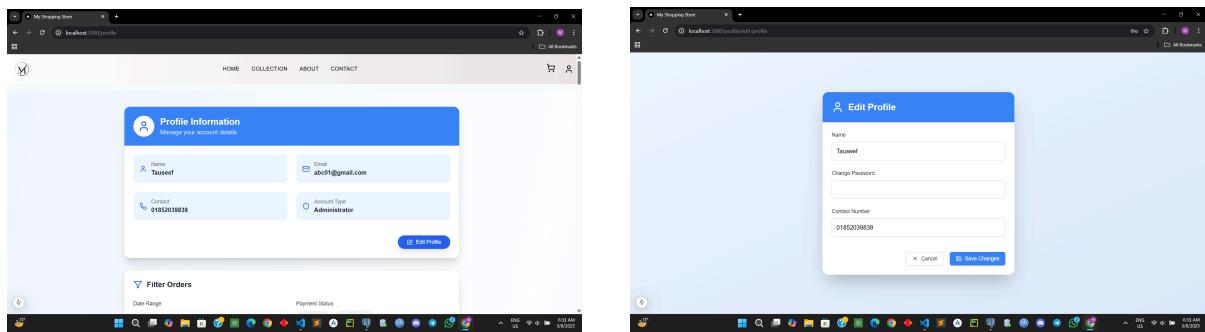
CRLF Ln 1, Col 47

9. Some snaps of our web application

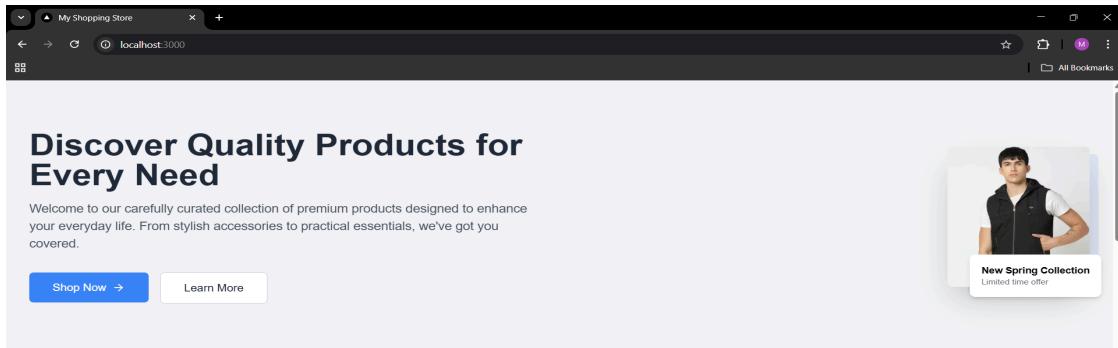
Login / SignUp page



Profile Information and edit profile page

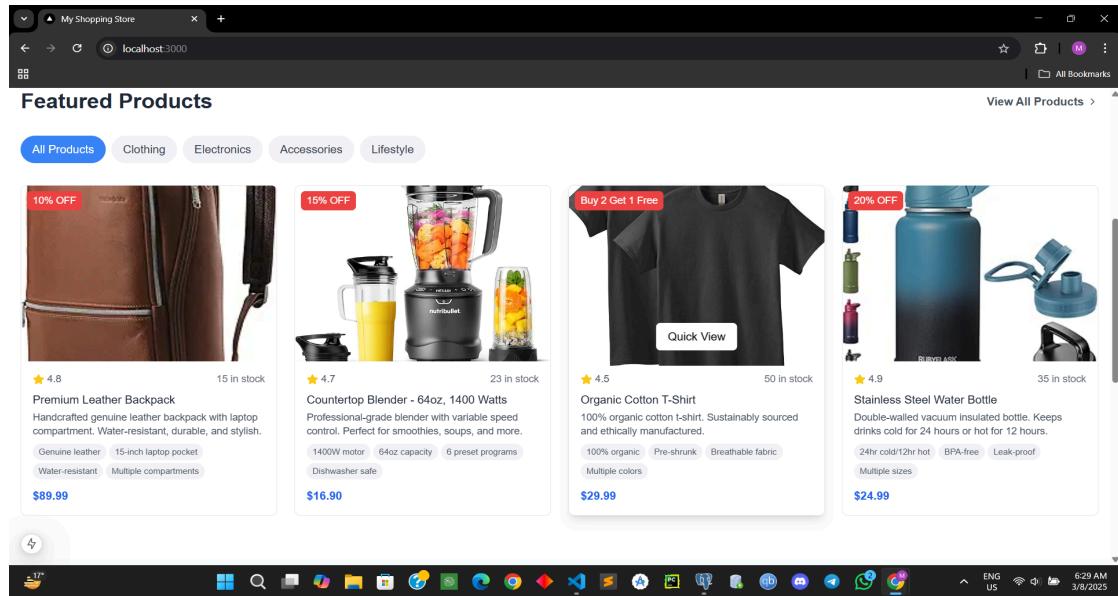
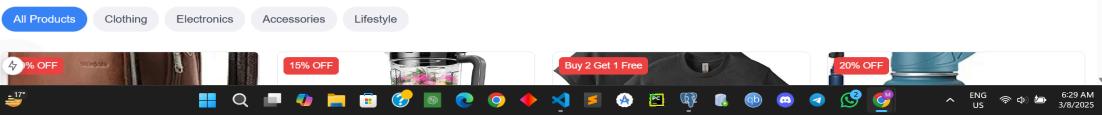


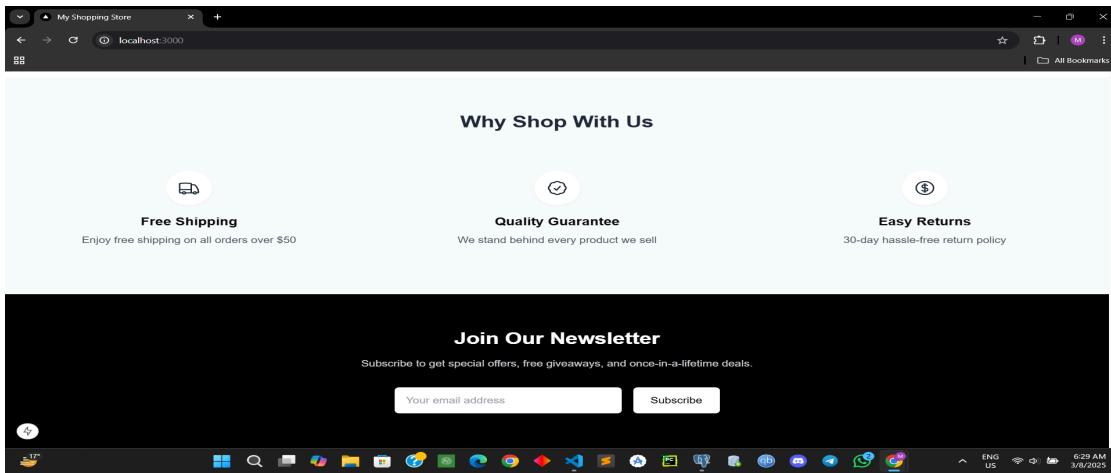
Main Home Page (View Products)



Featured Products

[View All Products >](#)





HOME **COLLECTION** **ABOUT** **CONTACT**

Filters

Price Range

Categories

socks	basketballs
apparel	tshirts
sports	bathroom
mens	hoodies
sweaters	kitchen
cleaning	swimming
robe	swimsuit
accessories	Camera
DSLR	Photo

**Breville Barista Express Espresso Machine, One Size, Brushed Stainless...
\$249.00**

**Waterproof Knit Athletic Sneakers - Gray
\$33.90**

**2 Slot Toaster - Black
\$18.99**

HOME **COLLECTION** **ABOUT** **CONTACT**

Filters

Price Range

Categories

socks	basketballs
apparel	tshirts
sports	bathroom
mens	hoodies
sweaters	kitchen
cleaning	swimming
robe	swimsuit
accessories	Camera
DSLR	Photo

**Black and Gray Athletic Cotton Socks - 6 Pairs
\$10.90**

**Intermediate Size Basketball
\$20.95**

Sort by
Price: Low to High
Price: High to Low
Highest Rating



Admin Panel

A screenshot of a web browser showing the "Admin Dashboard" for "My Shopping Store" at localhost:3000/admin. The dashboard has a light pink header with the title "Admin Dashboard" and a green "Live" indicator. Below the header is a "Dashboard Overview" section with the subtext "Monitor your store's performance and manage operations". It features six cards: "Add Product" (Create new listings), "Update Products" (Modify existing items), "Manage Users" (View all users), "Update Order Status" (Change the status of orders), "Create a New Worker Profile" (Create a new worker profile), and "Manage Workers" (View all workers). At the bottom left is a card for "Total Products" (34) and at the bottom right is a card for "Total Sales" (\$2600.45). The browser's address bar shows the URL "localhost:3000/admin". The taskbar at the bottom includes icons for file explorer, search, and various applications.

Add worker

The first screenshot shows the 'Add New Worker' form with fields for Name, Email, Contact Number, and Salary. The second screenshot shows the 'Workers Information' table with 10 entries.

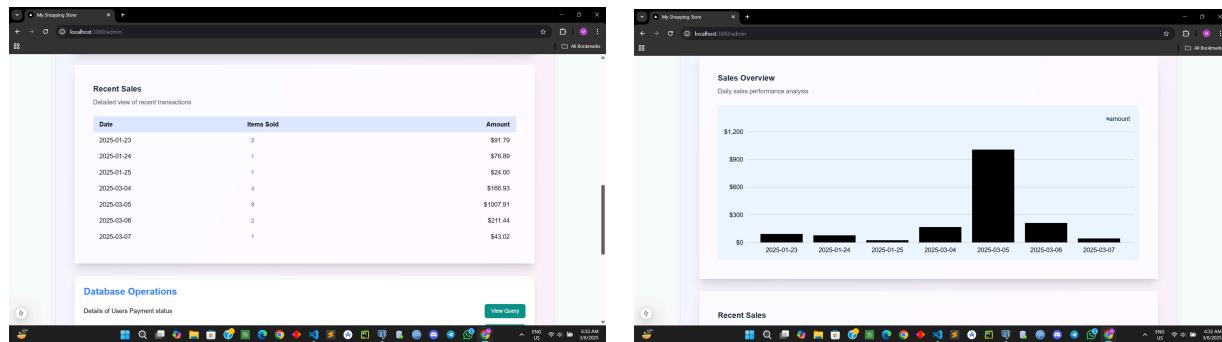
Worker ID	Name	Email	Contact Number	Salary
1	Rahul Sharma	rahul.sharma@gmail.com	9876543210	\$45000.00
2	Asha Khan	asha.khan@gmail.com	8765432109	\$50000.00
3	David Smith	david.smith@gmail.com	7654321098	\$48000.75
4	Sakura Tanaka	sakura.tanaka@gmail.com	6543210987	\$52000.25
5	Mohammed Ali	mohammed.ali@gmail.com	5432109876	\$47000.00
6	Emily Johnson	emily.johnson@gmail.com	4321098765	\$49000.00
7	Chen Wei	chen.wei@gmail.com	3210987654	\$51000.00
8	Maria Gonzalez	maria.gonzalez@gmail.com	2109876543	\$46000.80
9	John Doe	john.doe@gmail.com	1098765432	\$53000.10
10	Anup Mehta	anup.mehta@gmail.com	1876543210	\$50000.75

Add and Edit Product (Delete and Update)

The first screenshot shows the 'Add New Product' form with fields for Product Name, Image URL, Quantity, and Price. The second screenshot shows the 'Edit Product' form for a product with ID 51, with fields for Quantity, Rating Stars, Rating Count, Discount, and Categories.

The first screenshot shows the 'Edit Product Details' form for a product with ID 51, displaying the current details. The second screenshot shows the 'Edit Product' form for the same product, allowing changes to its details.

Overview of Recent Sales in graph form per day



Order Confirmation

Order Confirmation

Order Summary

Order ID	#58	Order Total	\$43.02
Date	2025-03-07T18:00:00.000Z	Bill Total	\$43.02
Customer	Tauseef	Status	UNPAID

Order Details

Product #4	Quantity: 1	\$10.00	\$10.00 each
Product #19		\$17.00	

Customer Tauseef **Status** UNPAID

Order Details

Product #4 Quantity: 1 \$10.00 \$10.00 each

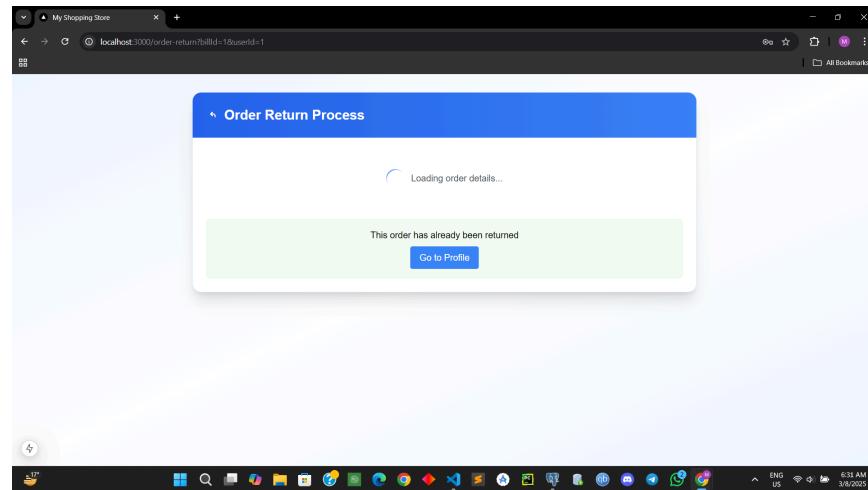
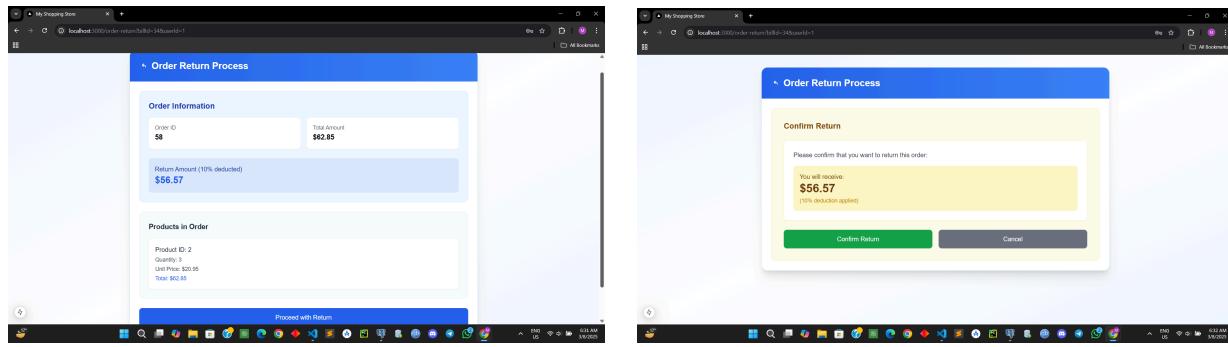
Product #19 Quantity: 1 \$17.00 \$17.00 each

SELECT PAYMENT METHOD

stripe Razorpay Cash on Delivery

Pay Now Return to Home

Order Return Process



Bill Details

Filter Orders

Date Range: 01/01/2025 - 03/08/2025
Payment Status: Paid, Unpaid

Billing History

Bill ID: 1
Bill Date: 2025-01-22T18:00:00.000Z
Payment Status: paid
Order Total: \$47.85
Bill Total: \$47.85
[Return Order](#)

Bill ID: 2
Bill Date: 2025-01-22T18:00:00.000Z
Payment Status: paid
Order Total: \$47.85
Bill Total: \$47.85
[Return Order](#)

Bill ID: 34
Bill Date: 2025-01-23T18:00:00.000Z
Payment Status: paid
Order Total: \$56.57
Bill Total: \$56.57
[Return Order](#)

Bill ID: 35
Bill Date: 2025-01-24T18:00:00.000Z
Payment Status: unpaid
Order Total: \$55.94
Bill Total: \$55.94
[Pay Now](#)

Bill ID: 36
Bill Date: 2025-01-24T18:00:00.000Z
Payment Status: paid
Order Total: \$76.20
Bill Total: \$76.20
[Return Order](#)

Shopping Cart page

The screenshot shows a shopping cart interface titled "Shopping Cart". It displays two items: a "2 Slot Toaster - Black" priced at \$15.57 with a 18.00% discount, and "Sterling Silver Sky Flower Stud Earrings" priced at \$17.45 with a 3.00% discount. Each item has quantity selection buttons (-, 1, +) and a delete icon. To the right is the "Order Summary" section showing Subtotal (\$36.98), Discount (-\$3.96), Shipping (\$10.00), and Final Total (\$43.02). Below it is the "Delivery Address" section with fields for Street Address, City, and Country. A large blue "Checkout" button is at the bottom. The browser title bar says "My Shopping Store" and the address bar shows "localhost:3000/cart". The taskbar at the bottom includes icons for various applications like File Explorer, Edge, and FileZilla.

About Us and Contact Us pages

The screenshot shows two side-by-side browser windows. The left window displays the "ABOUT US" page, featuring a grid of products (shoes, jeans, sweater) under the heading "Our Story". Text below the grid states: "Welcome to our Database Management System project! This e-commerce platform is developed by students of the Department of Computer Science and Engineering, University of Dhaka. Our platform demonstrates practical implementation of database concepts, including relational database design, query optimization, and transaction management." The right window displays the "CONTACT US" page, which includes a "Project Information" section with details about the Database Management System Course Project, the Department of Computer Science and Engineering, University of Dhaka, and contact emails (muntasirrahman01@gmail.com, feazmahmudth@gmail.com). It also features a "Team Members" section listing Md. Tauseef - Ur - Rahman and Fauz Mahmud. The browser title bar for both windows is "My Shopping Store" and the address bar shows "localhost:3000/about" and "localhost:3000/contact". The taskbar at the bottom is identical to the one in the shopping cart screenshot.

10. List of non-trivial FDs:

Users Table :

- $\text{user_id} \rightarrow (\text{user_name}, \text{user_email}, \text{user_password}, \text{user_contact_no}, \text{is_admin})$
- $\text{user_email} \rightarrow (\text{user_id}, \text{user_name}, \text{user_password}, \text{user_contact_no}, \text{is_admin})$

Product Table :

- $\text{prod_id} \rightarrow (\text{prod_name}, \text{prod_image}, \text{prod_quantity}, \text{prod_price}, \text{rating_stars}, \text{rating_count}, \text{prod_discount}, \text{prod_keywords})$

Orders Table :

- $\text{order_id} \rightarrow (\text{order_date}, \text{user_id}, \text{delivery_address}, \text{total_amt}, \text{order_status})$

Shopping_cart Table :

- $\text{cart_id} \rightarrow (\text{prod_id}, \text{user_id})$

Order_details Table :

- $(\text{order_id}, \text{prod_id}) \rightarrow (\text{prod_qty}, \text{prod_price}, \text{prod_total_price})$

Bill_details Table :

- $\text{bill_id} \rightarrow (\text{bill_date}, \text{user_id}, \text{order_id}, \text{user_name}, \text{order_total_price}, \text{bill_total_price}, \text{pay_status})$
- $\text{user_id} \rightarrow \text{user_name}$
- $\text{order_id} \rightarrow (\text{user_id}, \text{user_name}, \text{order_total_price})$

Order_return Table :

- $\text{order_return_id} \rightarrow (\text{order_id}, \text{user_id}, \text{return_date}, \text{prod_id}, \text{return_amount})$
- $\text{order_id} \rightarrow \text{user_id}$

Worker_Table :

- $\text{worker_id} \rightarrow (\text{worker_name}, \text{worker_email}, \text{worker_contact_no}, \text{worker_salary})$
- $\text{worker_email} \rightarrow (\text{worker_id}, \text{worker_name}, \text{worker_contact_no}, \text{worker_salary})$

11. proof that the schemas are in desired normal forms

Conditions for 1NF:

All the schemas should -

- Have primary keys
- Have no multi-valued attributes (i.e. all columns have atomic values)

All schemas in our database have primary keys and no multi-valued attribute. So all are in 1NF.

Conditions for 2NF :

1. Table must be in 1NF.
2. Table must not contain any partial dependencies i.e. all non-prime attributes must be fully dependent on the candidate key as a whole

Conditions for 3NF :

1. Table must be in 2NF.
2. Table must not contain any transitive dependencies i.e. all non-prime attributes must depend on primary key (or prime attributes)

Conditions for BCNF :

1. Table must be in 3NF.
2. **For every functional dependency (FD) $X \rightarrow Y$, X must be a superkey** (i.e., X must be either a candidate key or a superset of a candidate key).

Users table :

2NF check :

- since user_id is the primary key and determines all attributes so it is in 2NF.

3NF check:

- The candidate keys are {user_id} and {user_email}
- Every **non-prime attribute** (user_name, user_email, etc.) **depends only on the primary key** (user_id), so no transitive dependency exists..

BCNF check:

- user_id is a **candidate key**.
- user_email is also a **candidate key**.
- Since all FDs have **only candidate keys on the left side**, BCNF holds.

Product table :

2NF check :

- No partial dependencies since `prod_id` is the primary key so it is in 2NF.

3NF check:

- All **non-prime attributes** depend **only on `prod_id`**. So **No transitive dependencies exist.**

BCNF check:

- `prod_id` is a **candidate key**.
- All attributes depend **directly on `prod_id`**.
- No violations of BCNF rule.

Worker table :

2NF check :

- No partial dependencies since `worker_id` is the primary key so it is in 2NF.

3NF check:

- Every **non-prime attribute** depends **only on the candidate keys (`worker_id` or `worker_email`)**. So no transitive dependencies exist on this table.

BCNF check :

- Both `worker_id` and `worker_email` are **candidate keys**.
- Since all FDs have only candidate keys on the left, no violations.

Shopping_cart table :

2NF check :

- No partial dependencies since `cart_id` is the primary key.

3NF check :

- The non-prime attributes are `prod_id` and `user_id`, both of which are foreign keys.
- These attributes **directly depend on the primary key (`cart_id`)**.

BCNF check :

- The two candidate keys are: `cart_id` and **(`user_id`, `prod_id`)** (an user can only add a product once)
- The functional dependency `cart_id → prod_id, user_id` is fine because `cart_id` is a candidate key (superkey).
- The functional dependency **(`user_id`, `prod_id`) → cart_id** is also fine because **(`user_id`, `prod_id`)** is a candidate key (superkey).

Orders table :

2NF check :

- No partial dependencies since `order_id` is the primary key

3NF check :

- The non-prime attributes **directly depend on the primary key (`order_id`)**.

BCNF check :

- `order_id` is a **candidate key**.
- No other attribute determines `order_id`.
- All FDs have `order_id` (a superkey) on the left side.

Order_details table :

2NF check:

- The composite key (`order_id`, `prod_id`) uniquely determines all other attributes.

3NF check :

- Every **non-prime attribute** depends **only on the composite primary key {`order_id`, `prod_id`}**.

BCNF check :

- {`order_id`, `prod_id`} is the **candidate key**.
- All non-key attributes depend **fully** on {`order_id`, `prod_id`}.
- No violations of BCNF.

Order_return table :

2NF check :

- No partial dependencies since `order_return_id` is the primary key.

3NF check :

- All **non-prime attributes** depend **only on `order_return_id`**. So No transitive dependencies exist.

BCNF check :

- `order_return_id` is a **candidate key**.
- No other dependencies exist that violate BCNF.

Bill_details table :

2NF check :

- No partial dependencies since `bill_id` is the primary key

3NF check :

- **Issue:** `user_name` depends on `user_id`, not directly on `bill_id`.
- Since `user_name` is a **non-prime attribute** and depends on another **non-prime attribute** (`user_id`), we have a **transitive dependency**.

BCNF check :

- The first FD is fine (`bill_id` is a candidate key).
- The second FD suggests `order_id` determines non-key attributes **but order_id is not a superkey**

So, in our database

- all schemas are in second normal form.
- except `bill_details` schema , all the rest seven schemas are in 3NF as well as in BCNF

12. Front End designing tools and techniques, advantages and disadvantages of the implementation

Here's a detailed breakdown of the front-end designing tools and techniques we've used, along with their advantages and disadvantages:

Front-End Designing Tools and Techniques

1. Next.js 14 (React)

- **Technique:** A React-based framework for server-side rendering (SSR), static site generation (SSG), and client-side rendering (CSR).
- **Advantages:**
 - Improved performance with SSR and SSG.
 - Built-in routing and API routes simplify development.
 - Automatic code splitting for faster page loads.
 - SEO-friendly due to server-side rendering.
 - Supports TypeScript out of the box.
- **Disadvantages:**
 - The steeper learning curve for beginners compared to plain React.
 - Overhead for small projects (might be overkill).
 - Requires understanding of server-side concepts.

2. TypeScript

- **Technique:** A statically typed superset of JavaScript.
- **Advantages:**
 - Catches errors at compile time, reducing runtime bugs.
 - Improves code readability and maintainability.
 - Enhances developer productivity with better IDE support (e.g., autocompletion).
 - Easier to refactor code.
- **Disadvantages:**
 - Adds complexity to the development process.
 - Requires additional setup and configuration.
 - Can be overkill for small projects.

3. Tailwind CSS

- **Technique:** A utility-first CSS framework.
- **Advantages:**
 - Rapid prototyping with pre-defined utility classes.
 - Highly customizable with a tailwind.config.js file.
 - Reduces the need for writing custom CSS.
 - Encourages consistency in design.
- **Disadvantages:**
 - Can lead to cluttered HTML with many utility classes.
 - The steeper learning curve for developers unfamiliar with utility-first CSS.
 - Larger CSS file size if not purged properly.

4. Tremor (for Charts and Analytics)

- **Technique:** A library for building charts and analytics dashboards.
- **Advantages:**
 - Easy to integrate with React and Next.js.
 - Pre-built components for common chart types (e.g., bar, line, pie).
 - Customizable and responsive.
- **Disadvantages:**
 - Limited compared to more mature libraries like D3.js or Chart.js.
 - May not support advanced charting requirements.

5. Lucid Icons

- **Technique:** A lightweight icon library.
- **Advantages:**
 - Easy to use and integrate.
 - Highly customizable (size, color, etc.).
 - Lightweight and performant.
- **Disadvantages:**
 - Limited icon set compared to larger libraries like FontAwesome.
 - Requires manual updates for new icons.

13. Conclusion of the work

Working on this **Shop Management System** project has been a great learning experience. Throughout the project, we faced many challenges, such as designing a well-structured database, writing efficient queries, and ensuring smooth connections between different parts of the system.

We learned how to manage user data, handle orders and payments, and update product inventory. We also improved our skills in **database design, query optimization, and problem-solving**.

This project helped us understand how real-world systems work and how important databases are in managing business operations. Overall, it was a rewarding journey that gave us valuable knowledge for future projects.