

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Факультет безопасности информационных технологий

Направление подготовки: 10.03.01 Информационная безопасность


Образовательная программа: Информационная безопасность

Дисциплина:  
«Информационная безопасность баз данных»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2  
«Реализация БД в рамках СУБД»

Выполнил студент(ы):

группа/поток: N3347 / ИББД. N63 1.5

Чу Ван Доан /   
ФИО Подпись

Проверил:

Салихов Максим Русланович / \_\_\_\_\_  
ФИО Подпись

Отметка о выполнении (один из вариантов:  
отлично, хорошо, удовлетворительно, зачтено)

\_\_\_\_\_  
Дата

Санкт-Петербург  
2024 г.

## СОДЕРЖАНИЕ

<b>Цель работы</b>	<b>3</b>
<b>Задание</b>	<b>4</b>
<b>Ход Работы</b>	<b>5</b>
1. Выбрать систему управления базами данных (СУБД), которая будет использована в рамках лабораторной работы. Кратко обосновать свой выбор.	5
2. Создать БД в выбранной в вами СУБД на основе итоговой разработанной схемы отношений из ЛР 1. Заполните созданную вами БД информацией, сгенерируйте как минимум 7-8 кортежей с данными для каждой из ваших основных таблиц. В отчете по лабораторной работе укажите следующий SQL-код (написанный вами или сгенерированный средствами администрирования СУБД):	6
2.1 Код для создания всех таблиц;	7
2.2 Код для внесения данных в созданные таблицы;	9
2.3 Код хотя бы одной SQL-команды для модифицирования структуры таблицы;	11
3. Индексировать таблицы. Добавить индексы для атрибутов, по которым происходит объединение таблиц, а также атрибуты по которым выполняется поиск/фильтрация данных.	11
4. Установить взаимосвязи между таблицами.	12
5. Дополнительно. Тестовых запросов к вашей БД	13
6. Создать представления, составленные в пункте 5 лабораторной 1.	15
<b>Вывод</b>	<b>17</b>

## **Цель работы**

Получение навыков по работе с современными системами управления базами данных.

## Задание

1. Выбрать систему управления базами данных (СУБД), которая будет использована в рамках лабораторной работы. Кратко обосновать свой выбор.
2. Создать БД в выбранной в вами СУБД на основе итоговой разработанной схемы отношений из ЛР 1. Заполните созданную вами БД информацией, сгенерируйте как минимум 7-8 кортежей с данными для каждой из ваших основных таблиц. В отчете по лабораторной работе укажите следующий SQL-код (написанный вами или сгенерированный средствами администрирования СУБД):
  - код для создания всех таблиц;
  - код для внесения данных в созданные таблицы;
  - код хотя бы одной SQL-команды для модифицирования структуры таблицы;
3. Индексировать таблицы. Добавить индексы для атрибутов, по которым происходит объединение таблиц, а также атрибуты по которым выполняется поиск/фильтрация данных.
4. Установить взаимосвязи между таблицами.
5. **Дополнительно. Тестовых запросов к вашей БД**
6. Создать представления, составленные в пункте 5 лабораторной 1.

## **Ход Работы**

**1. Выбрать систему управления базами данных (СУБД), которая будет использована в рамках лабораторной работы. Кратко обосновать свой выбор.**

- Использование PostgreSQL
- Причина использования: PostgreSQL — это мощная, открытая система управления реляционными базами данных, соответствующая стандарту SQL. Она способна обрабатывать сложные базы данных, поддерживает многопользовательскую работу и хорошо интегрируется с различными языками программирования. PostgreSQL также широко используется в реальных проектах благодаря своей расширяемости и высокой безопасности.

2. Создать БД в выбранной в вами СУБД на основе итоговой разработанной схемы отношений из ЛР 1. Заполните созданную вами БД информацией, сгенерируйте как минимум 7-8 кортежей с данными для каждой из ваших основных таблиц. В отчете по лабораторной работе укажите следующий SQL-код (написанный вами или сгенерированный средствами администрирования СУБД):

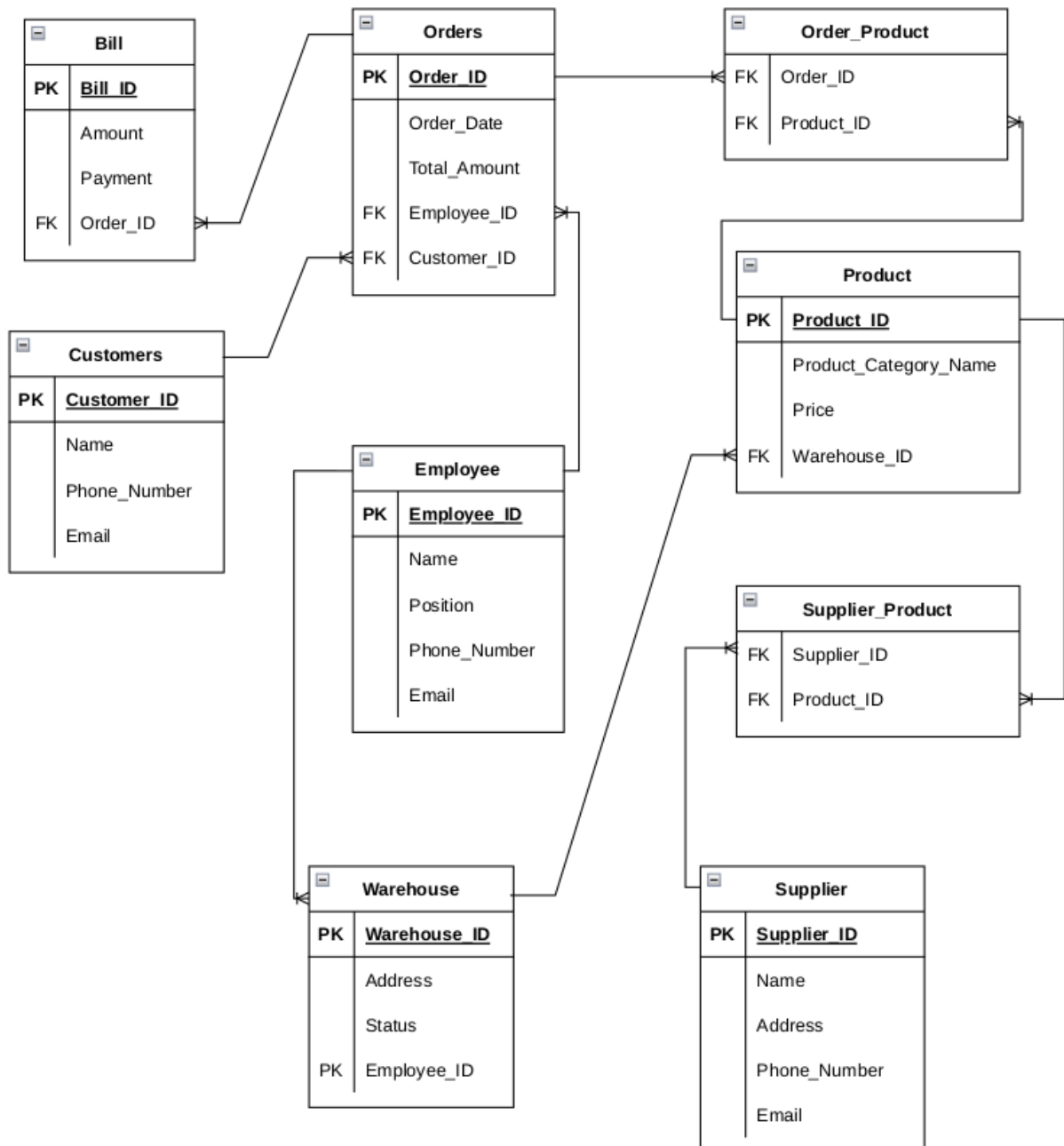


Рисунок 1. Итоговая схема предварительных отношений

- Подключение к базе данных:

```
sudo systemctl enable --now postgresql.service
sudo su postgres -c psql
```

- Создать базу данных

```
create database coffee_shop_db
with
owner = postgres
encoding = 'UTF8'
tablespace = pg_default
connection limit = -1
is_template = False;
```

- Создайте схему и укажите ее

```
coffee_shop_db=# create schema coffee_shop_schema;
CREATE SCHEMA
coffee_shop_db=# \dn
               List of schemas
      Name      |      Owner
-----+-----
coffee_shop_schema | postgres
public           | pg_database_owner
(2 rows)
coffee_shop_db=# set search_path to coffee_shop_schema;
SET
coffee_shop_db=# show search_path ;
      search_path
-----
coffee_shop_schema
```

## 2.1 Код для создания всех таблиц;

- Таблица Employee

```
CREATE TABLE Employee (  
    Employee_ID SERIAL PRIMARY KEY,  
    Name VARCHAR(100) NOT NULL,  
    Position VARCHAR(50) NOT NULL,  
    Phone_Number VARCHAR(13),  
    Email VARCHAR(100)  
);
```

- Таблица Supplier

```
CREATE TABLE Supplier (  
    Supplier_ID SERIAL PRIMARY KEY,  
    Name VARCHAR(100) NOT NULL,  
    Address VARCHAR(255),  
    Phone_Number VARCHAR(13),  
    Email VARCHAR(100)  
);
```

- Таблица Warehouse

```
CREATE TABLE Warehouse (  
    Warehouse_ID SERIAL PRIMARY KEY,  
    Address VARCHAR(255) NOT NULL,  
    Status VARCHAR(50),  
    Employee_ID INTEGER,  
    FOREIGN KEY (Employee_ID) REFERENCES Employee(Employee_ID)  
);
```

- Таблица Product

```
CREATE TABLE Product (  
    Product_ID SERIAL PRIMARY KEY,  
    Product_Category_Name VARCHAR(100) NOT NULL,  
    Price NUMERIC(10, 2) NOT NULL,  
    Warehouse_ID INTEGER,  
    FOREIGN KEY (Warehouse_ID) REFERENCES Warehouse(Warehouse_ID)  
);
```

- Таблица Customer

```
CREATE TABLE Customer (  
    Customer_ID SERIAL PRIMARY KEY,  
    Name VARCHAR(100) NOT NULL,  
    Phone_Number VARCHAR(15),  
    Email VARCHAR(100)  
);
```

- Таблица Orders

```
CREATE TABLE Orders(  
    Order_ID SERIAL PRIMARY KEY,  
    Order_Date DATE NOT NULL,
```



- ```

Total_Amount NUMERIC(10, 2) NOT NULL,
Customer_ID INTEGER,
Employee_ID INTEGER,
FOREIGN KEY (Customer_ID) REFERENCES Customer(Customer_ID),
FOREIGN KEY (Employee_ID) REFERENCES Employee(Employee_ID)
);

```
- Таблица Bill

```

CREATE TABLE Bill (
    Bill_ID SERIAL PRIMARY KEY,
    Amount NUMERIC(10, 2) NOT NULL,
    Payment_Method VARCHAR(50) NOT NULL,
    Order_ID INTEGER,
    FOREIGN KEY (Order_ID) REFERENCES Orders(Order_ID)
);

```
  - Таблица связи между таблицей Orders и таблицей Product

```

CREATE TABLE Order_Product (
    Order_ID INTEGER,
    Product_ID INTEGER,
    PRIMARY KEY (Order_ID, Product_ID),
    FOREIGN KEY (Order_ID) REFERENCES "Order"(Order_ID),
    FOREIGN KEY (Product_ID) REFERENCES Product(Product_ID)
);

```
  - Таблица связи между таблицей Supplier и таблицей Product

```

CREATE TABLE Supplier_Product (
    Supplier_ID INTEGER,
    Product_ID INTEGER,
    PRIMARY KEY (Supplier_ID, Product_ID),
    FOREIGN KEY (Supplier_ID) REFERENCES Supplier(Supplier_ID),
    FOREIGN KEY (Product_ID) REFERENCES Product(Product_ID)
);

```

## 2.2 Код для внесения данных в созданные таблицы;

- У нас есть следующие таблицы

```
coffee_shop_db=# \dt
```

| List of relations  |                  |       |          |
|--------------------|------------------|-------|----------|
| Schema             | Name             | Type  | Owner    |
| coffee_shop_schema | bill             | table | postgres |
| coffee_shop_schema | customer         | table | postgres |
| coffee_shop_schema | employee         | table | postgres |
| coffee_shop_schema | order_product    | table | postgres |
| coffee_shop_schema | orders           | table | postgres |
| coffee_shop_schema | product          | table | postgres |
| coffee_shop_schema | supplier         | table | postgres |
| coffee_shop_schema | supplier_product | table | postgres |
| coffee_shop_schema | warehouse        | table | postgres |
| (9 rows)           |                  |       |          |

- Внесение данных в таблицу Employee

```
INSERT INTO employee (name, position, phone_number, email)
VALUES
    ('Nguyen Van A', 'supervisor', '9312828535',
    'nguyenvana123@gmail.com'),
    ('Nguyen Thi B', 'salesperson', '92537563834',
    'nb4214@gmail.com'),
    ('Artom', 'salesperson', '27582473683', 'artom33@mail.ru'),
    ('Irina', 'salesperson', '8925748253', 'irina8386@mail.ru'),
    ('Tran', 'salesperson', '92846363583', 'trantran4953@gmail.com');
```

- Внесение данных в таблицу Supplier

```
INSERT INTO supplier (name, address, phone_number, email)
VALUES
    ('Trung Nguyen Coffee', 'Dalat city', '03873532753',
    'trungnguyencoffee@gmail.com'),
    ('King Coffee', 'Ho Chi Minh city', '0385636282',
    'kingcoffee@gmail.com'),
    ('G7 Coffee', 'Ha Noi', '92834772843', 'g7coffee@gmail.com');
```

- Внесение данных в таблицу Warehouse

```
INSERT INTO warehouse (address, status, employee_id)
VALUES
    ('Lam Ha', 'In stock', 1),
    ('Tan Ha', 'In stock', 1),
    ('Dan Phuong', 'In stock', 1),
    ('Me Linh', 'In stock', 1);
```

- Внесение данных в таблицу Product

```
INSERT INTO product (product_category_name, price, warehouse_id)
VALUES
```

- ```

('Arabica', 100000, 1),
('Robusta', 90000, 2),
('Bourbon', 96000, 3),
('Typica', 92000, 4);

```
- Внесение данных в таблицу Customer

```

INSERT INTO customer (name, phone_number, email)
VALUES
    ('Alex', '93842727543', 'alex8888@mail.ru'),
    ('Tom', '82736464383', 'tomi7749@mail.ru'),
    ('Anton', '827364646737', 'ton@mail.ru'),
    ('Karababy', '8283747654', 'baby@mail.ru');

```
  - Внесение данных в таблицу Bill

```

INSERT INTO bill (amount, payment, order_id)
VALUES
    (150000, 'Cash', 1),
    (599999, 'Bank Transfer', 2),
    (50000, 'Cash', 3),
    (20000, 'Online Payment', 3),
    (6699999, 'Cash', 4);

```
  - Внесение данных в таблицу Order\_Product

```

INSERT INTO order_product (order_id, product_id)
VALUES
    (1, 1),
    (2, 2),
    (3, 3),
    (4, 4);

```
  - Внесение данных в таблицу Order\_Product

```

INSERT INTO supplier_product (supplier_id, product_id)
VALUES
    (1, 1),
    (1, 2),
    (2, 3),
    (3, 4);

```

**2.3 Код хотя бы одной SQL-команды для модифицирования структуры таблицы;**

- Добавление столбца Address в таблицу Customer  
`ALTER TABLE Customer ADD COLUMN Address VARCHAR(255);`
- Удаление столбца Address в таблице Customer  
`ALTER TABLE Customer DROP COLUMN Address;`

### **3. Индексировать таблицы. Добавить индексы для атрибутов, по которым происходит объединение таблиц, а также атрибуты по которым выполняется поиск/фильтрация данных.**

```
-- Таблица Employee
-- Создание индекса для столбца Employee_ID для поиска сотрудников
CREATE INDEX idx_employee_id ON Employee(Employee_ID);

-- Таблица Supplier
-- Создание индекса для столбца Supplier_ID для поиска поставщиков
CREATE INDEX idx_supplier_id ON Supplier(Supplier_ID);

-- Таблица Product
-- Создание индекса для столбца Warehouse_ID для ускорения связи с
таблицей Warehouse
CREATE INDEX idx_product_warehouse_id ON Product(Warehouse_ID);

-- Таблица Customer
-- Создание индекса для столбца Customer_ID для поиска клиентов
CREATE INDEX idx_customer_id ON Customer(Customer_ID);

-- Таблица Order
-- Создание индекса для столбцов Customer_ID и Employee_ID для ускорения
запросов между Order и Customer, Order и Employee
CREATE INDEX idx_order_customer_id ON orders(Customer_ID);
CREATE INDEX idx_order_employee_id ON orders(Employee_ID);

-- Таблица Bill
-- Создание индекса для столбца Order_ID для ускорения запросов между Bill
и Order
CREATE INDEX idx_bill_order_id ON Bill(Order_ID);

-- Таблица Warehouse
-- Создание индекса для столбца Employee_ID для ускорения запросов между
Warehouse и Employee
CREATE INDEX idx_warehouse_employee_id ON Warehouse(Employee_ID);

-- Связующая таблица Order_Product
```

```

-- Создание индекса для столбцов Order_ID и Product_ID для ускорения
запросов между Order и Product
CREATE INDEX idx_order_product_order_id ON Order_Product(Order_ID);
CREATE INDEX idx_order_product_product_id ON Order_Product(Product_ID);

-- Связующая таблица Supplier_Product
-- Создание индекса для столбцов Supplier_ID и Product_ID для ускорения
запросов между Supplier и Product
CREATE          INDEX          idx_supplier_product_supplier_id          ON
Supplier_Product(Supplier_ID);
CREATE          INDEX          idx_supplier_product_product_id          ON
Supplier_Product(Product_ID);

```

#### 4. Установить взаимосвязи между таблицами.

- Основные связи между таблицами установлены с использованием внешних ключей, но дополнительные индексы также ускоряют запросы.

```

ALTER TABLE Warehouse
ADD FOREIGN KEY (Employee_ID) REFERENCES Employee(Employee_ID);

```

```

ALTER TABLE Orders
ADD FOREIGN KEY (Customer_ID) REFERENCES Customer(Customer_ID),
ADD FOREIGN KEY (Employee_ID) REFERENCES Employee(Employee_ID);

```

```

ALTER TABLE Bill
ADD FOREIGN KEY (Order_ID) REFERENCES Orders(Order_ID);

```

```

ALTER TABLE Order_Product
ADD FOREIGN KEY (Order_ID) REFERENCES Orders(Order_ID),
ADD FOREIGN KEY (Product_ID) REFERENCES Product(Product_ID);

```

```

ALTER TABLE Supplier_Product
ADD FOREIGN KEY (Supplier_ID) REFERENCES Supplier(Supplier_ID),
ADD FOREIGN KEY (Product_ID) REFERENCES Product(Product_ID);

```

#### 5. Дополнительно. Тестовых запросов к вашей БД

- Получить все заказы с информацией о клиентах и сотрудниках, которые их обслуживали:

```

coffee_shop_db=# SELECT o.Order_ID, o.Order_Date, o.Total_Amount,
c.Name AS Customer, e.Name AS Employee

FROM Orders o

JOIN Customer c ON o.Customer_ID = c.Customer_ID

```

```
JOIN Employee e ON o.Employee_ID = e.Employee_ID;
```

```
order_id | order_date | total_amount | customer | employee
```

```
-----+-----+-----+-----+-----
```

```
1 | 2024-10-12 | 150000.00 | Alex | Nguyen Thi B
```

```
2 | 2024-10-13 | 599999.00 | Tom | Nguyen Thi B
```

```
3 | 2024-10-13 | 70000.00 | Anton | Artom
```

```
4 | 2024-10-20 | 669999.00 | Karababy | Irina
```

```
(4 rows)
```

- Показать все продукты, хранящиеся на каждом складе:

```
coffee_shop_db=# SELECT w.Address AS Warehouse,
p.Product_Category_Name AS Product, p.Price
```

```
FROM Warehouse w
```

```
JOIN Product p ON w.Warehouse_ID = p.Warehouse_ID;
```

```
warehouse | product | price
```

```
-----+-----+-----
```

```
Lam Ha | Arabica | 100000.00
```

```
Tan Ha | Robusta | 90000.00
```

```
Dan Phuong | Bourbon | 96000.00
```

```
Me Linh | Typica | 92000.00
```

```
(4 rows)
```

- Вывести список поставщиков и предоставляемых ими продуктов:

```
coffee_shop_db=# SELECT s.Name AS Supplier, p.Product_Category_Name
AS Product
```

```
FROM Supplier s
```

```
JOIN Supplier_Product sp ON s.Supplier_ID = sp.Supplier_ID
```

```
JOIN Product p ON sp.Product_ID = p.Product_ID;
```

```
supplier | product
```

```

-----+-----
Trung Nguyen Coffee | Arabica
Trung Nguyen Coffee | Robusta
King Coffee         | Bourbon
G7 Coffee           | Typica

(4 rows)

```

- Найти сумму всех заказов, оплаченных наличными:

```

coffee_shop_db=# SELECT SUM(b.Amount) AS TotalCashPayments
FROM Bill b
WHERE b.Payment = 'Cash';

totalcashpayments

-----

6899999.00

(1 row)

```

- Получить все заказы, содержащие конкретный продукт, например "Arabica":

```

coffee_shop_db=# SELECT o.Order_ID, o.Order_Date, o.Total_Amount
FROM Orders o
JOIN Order_Product op ON o.Order_ID = op.Order_ID
JOIN Product p ON op.Product_ID = p.Product_ID
WHERE p.Product_Category_Name = 'Arabica';
order_id | order_date | total_amount
-----+-----+-----
1 | 2024-10-12 | 150000.00

(1 row)

```

## 6. Создать представления, составленные в пункте 5 лабораторной 1.

- Представление для сотрудников отдела продаж (Sales Employee View)

```
CREATE VIEW sales_employee_view AS
SELECT
    e.employee_id,
    e.name AS employee_name,
    o.order_id,
    o.order_date,
    o.total_amount,
    c.customer_id,
    c.name AS customer_name,
    c.phone_number AS customer_phone,
    c.email AS customer_mail
FROM
    employee e
JOIN
    orders o ON e.employee_id = o.employee_id
JOIN
    customer c ON o.customer_id = c.customer_id;
```

	employee_id integer	employee_name character varying (100)	order_id integer	order_date date	total_amount numeric (10,2)	customer_id integer	customer_name character varying (100)	customer_phone character varying (13)	customer_mail character varying (100)
1	2	Nguyen Thi B	1	2024-10-12	150000.00	1	Alex	93842727543	alex8888@mail.ru
2	2	Nguyen Thi B	2	2024-10-13	599999.00	2	Tom	82736464383	tomi7749@mail.ru
3	3	Artom	3	2024-10-13	70000.00	3	Anton	827364646737	ton@mail.ru
4	4	Irina	4	2024-10-20	6699999.00	4	Karababy	8283747654	baby@mail.ru

- Представление для менеджера склада (Warehouse Manager View)

```
CREATE VIEW Warehouse_Manager_View AS
SELECT
    w.warehouse_id,
    w.address AS warehouse_address,
    w.status AS warehouse_status,
    p.product_id,
    p.product_category_name,
    p.price
FROM
    warehouse w
JOIN
    product p ON w.warehouse_id = p.warehouse_id;
```



	warehouse_id integer	warehouse_address character varying (100)	warehouse_status character varying (50)	product_id integer	product_category_name character varying (100)	price numeric (10,2)
1	1	Lam Ha	In stock	1	Arabica	100000.00
2	2	Tan Ha	In stock	2	Robusta	90000.00
3	3	Dan Phuong	In stock	3	Bourbon	96000.00
4	4	Me Linh	In stock	4	Typica	92000.00

- Представление для клиентов (Customer View)

```

CREATE VIEW Customer_Order_View AS
SELECT
    c.customer_id,
    c.name AS customer_name,
    o.order_id,
    o.order_date,
    o.total_amount,
    b.bill_id,
    b.amount AS bill_amount,
    b.payment
FROM
    customer c
JOIN
    orders o ON c.customer_id = o.customer_id
JOIN
    bill b ON o.order_id = b.order_id;

```

	customer_id integer	customer_name character varying (100)	order_id integer	order_date date	total_amount numeric (10,2)	bill_id integer	bill_amount numeric (10,2)	payment character varying (50)
1	1	Alex	1	2024-10-12	150000.00	1	150000.00	Cash
2	2	Tom	2	2024-10-13	599999.00	2	599999.00	Bank Transfer
3	3	Anton	3	2024-10-13	70000.00	3	50000.00	Cash
4	3	Anton	3	2024-10-13	70000.00	4	20000.00	Online Payment
5	4	Karababy	4	2024-10-20	6699999.00	5	6699999.00	Cash

## **Вывод**

В ходе выполнения лабораторной работы были получены навыки по работе с системой управления базами данных PostgreSQL. На основе инфологической модели, разработанной в предыдущей лабораторной работе, была создана база данных для системы управления кафе. Были созданы основные таблицы, включая таблицы сотрудников, клиентов, поставщиков, заказов, продуктов, складов и другие необходимые сущности.

Для каждой таблицы были добавлены необходимые атрибуты и первичные ключи. Также были реализованы внешние ключи, которые обеспечивают связь между таблицами и поддерживают целостность данных. Сгенерированы и добавлены тестовые данные для каждой таблицы, что позволяет проводить анализ данных и проверять функциональность базы данных.

Кроме того, для повышения производительности были созданы индексы на ключевые атрибуты, по которым выполняется поиск и соединение данных. Также были выполнены запросы для изменения структуры таблиц, в частности добавление и удаление столбцов.

В завершение, созданные представления и тестовые запросы показали корректность работы базы данных и возможность использования ее в системе кафе для хранения, обработки и анализа данных.