### Молдавский Государственный Университет Факультет Математики и Информатики Департамент Информатики

### 

### **Индивидуальная работа** По курсу “Базы данных” Тема: Создание базы данных “Restaurant\_DB”

### 

### Выполнила студентка группы I2302: Калинкова С. Проверила преподаватель: Карчева Н.Ф.

### 

### Кишинев, 2024

### **Цель работы:**

### Создание реляционной базы данных «Restaurant\_DB» в приложении Oracle APEX, предназначенной для управления данными о заказах, клиентах, сотрудниках, блюдах и ресторане в целом. Данная база данных необходима для учета клиентов, хранения информации о меню, заказах, сотрудниках и резервации столов, а также для анализа продаж, обслуживания и управления ресурсами ресторана. База данных способствует эффективному управлению рестораном.

### Отношения находятся в нормальной форме.

### Ограничения структуры (уникальность ключа).

### Реляционные операторы.

### **Этапы создания базы данных:**

### **Анализ предметной области.** Процесс создания базы данных начался с анализа потребностей ресторана. Было решено, что ключевыми сущностями являются клиенты, сотрудники, блюда, заказы и счета. Каждая из этих сущностей имеет свои уникальные атрибуты, которые необходимо учесть при проектировании структуры базы данных.

### **Сбор необходимой информации.** Я попыталась собрать все необходимые данные, чтобы база данных могла поддерживать деятельность ресторана. Например, для клиентов это будет информация о имени, адресе и телефоне, для сотрудников — о должности и зарплате и т.д. Эти данные позволяют эффективнее управлять рестораном.

### **Создание проекта.** На основе собранной информации была разработана схема базы данных, которая включает таблицы для хранения информации. Все таблицы нормализованы для устранения избыточности, и установлены внешние ключи, которые обеспечивают правильные связи между таблицами.

### **Физическая реализация.** В Oracle APEX была реализована структура базы данных. Я создала таблицы, настроила ограничения и связи, заполнила базу тестовыми данными и написала запросы и представления, которые позволяют оперативно извлекать информацию и проводить аналитику.

### 

### **Содержание работы:**

### **Схема базы данных.** Схема отражает все основные аспекты работы ресторана. Она включает таблицы для клиентов, сотрудников, блюд, заказов и счетов с полями, первичными и внешними ключами, чтобы обеспечить целостность данных и оптимальную работу базы.

### **Примеры запросов для извлечения необходимой информации.** Представлены примеры запросов, которые помогают решать реальные задачи ресторана. Запросы охватывают некоторые ключевые аспекты работы заведения, включая анализ популярности блюд, анализ заказов, активность сотрудников и вычисление общей суммы заказов.

### **Создание представлений. Представления используются для упрощения работы с данными.** В данной работе созданы представления, которые: анализируют популярность блюд по количеству заказов, показывают вклад сотрудников в продажи ресторана, помогают отслеживать свободные столы на определённую дату, анализируют заказы по дням, показывают частоту использования столов. Эти представления предоставляют готовые отчёты для анализа данных, упрощают работу с базой и обеспечивают быстрый доступ к важной информации.

### **Схема базы данных включает в себя следующие таблицы:**

### **Customer** - данные о клиентах (customer\_id - ID клиента, customer\_name – имя клиента, address – адрес, phone\_number – номер телефона).

### **Employees** - данные о сотрудниках ресторана (employee\_id - ID сотрудника, employee\_name – имя сотрудника, position – должность, salary – зарплата, commision\_percetage - коммисионные).

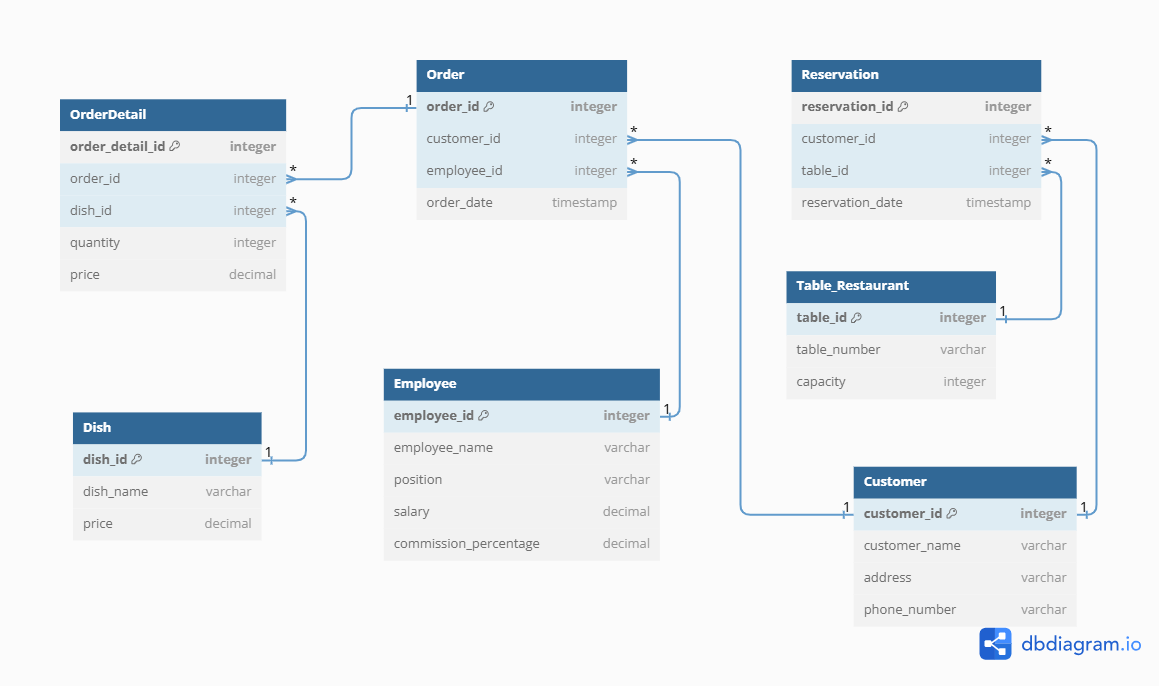
### **Dish** - блюда меню (dish\_id - ID блюда, dish\_name – название блюда, price – цена).

### **Orders -** отслеживает все заказы (order\_id - ID заказа, customer\_id – ID клиента, employee\_id – ID сотрудника, order\_date – дата заказа).

### **OrderDetail** - учета подробностей каждого заказа (order\_detail\_id - ID записи заказа, order\_id – ID заказа, dish\_id – ID блюда, quantity – количество, price – цена блюда).

### **Reservation** - информация о бронировании (reservation\_id – ID брони, customer\_id – ID клиента, table\_id – ID стола, reservation\_date – дата и время бронирования)

### **Table\_Restaurant** - данные о столах в ресторане (table\_id – ID стола, table\_number – номер стола, capacity – вместимость стола)



# 

### 

### 

### **Таблица: OrderDetail**

**Создание 10 запросов (данные из нескольких таблиц)**

**1. Список блюд, которые входили в заказ клиента "Ana Balan".(их название количество и цена)**

SELECT d.dish\_name, od.quantity, od.price

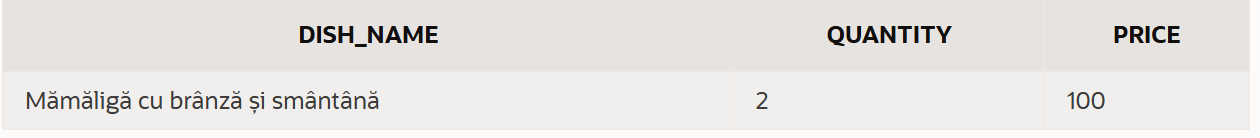
FROM Orders o

JOIN Customer c ON o.customer\_id = c.customer\_id

JOIN OrderDetail od ON o.order\_id = od.order\_id

JOIN Dish d ON od.dish\_id = d.dish\_id

WHERE c.customer\_name = 'Ana Balan';

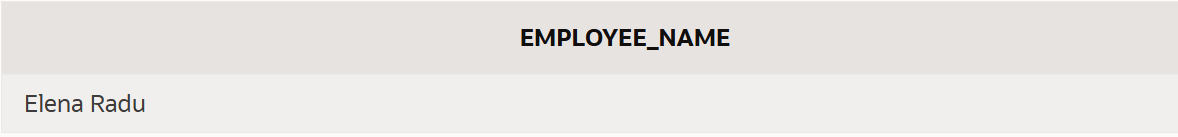
  
  
**2. Сотрудники, которые обслуживали клиента "Elena Ciobanu".**  
SELECT e.employee\_name

FROM Orders o

JOIN Customer c ON o.customer\_id = c.customer\_id

JOIN Employee e ON o.employee\_id = e.employee\_id

WHERE c.customer\_name = 'Elena Ciobanu';

  
  
**3. Имя и Фамилия клиента, который зарезервировал стол E5.**

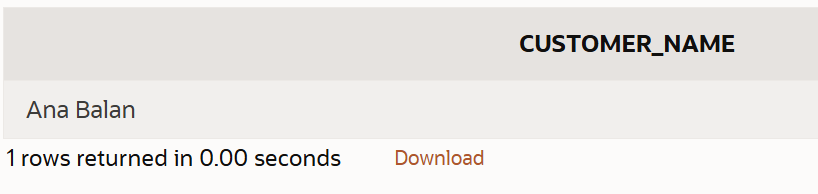
SELECT c.customer\_name

FROM Reservation r

JOIN Customer c ON r.customer\_id = c.customer\_id

JOIN Table\_Restaurant t ON r.table\_id = t.table\_id

WHERE t.table\_number = 'E5';



**4. Имя и фамилия официанта, который обслуживал стол 'B2'**

SELECT e.employee\_name

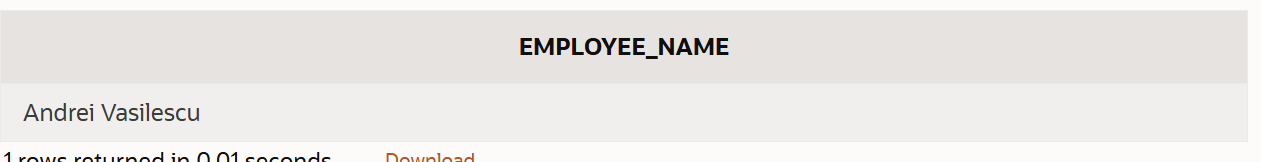
FROM Employee e

JOIN Orders o ON e.employee\_id = o.employee\_id

JOIN Reservation r ON o.customer\_id = r.customer\_id

JOIN Table\_Restaurant t ON r.table\_id = t.table\_id

WHERE t.table\_number = 'B2';

  
  
**5. Вывод результатов в порядке убывания общей суммы для каждого заказа**

SELECT o.order\_id,

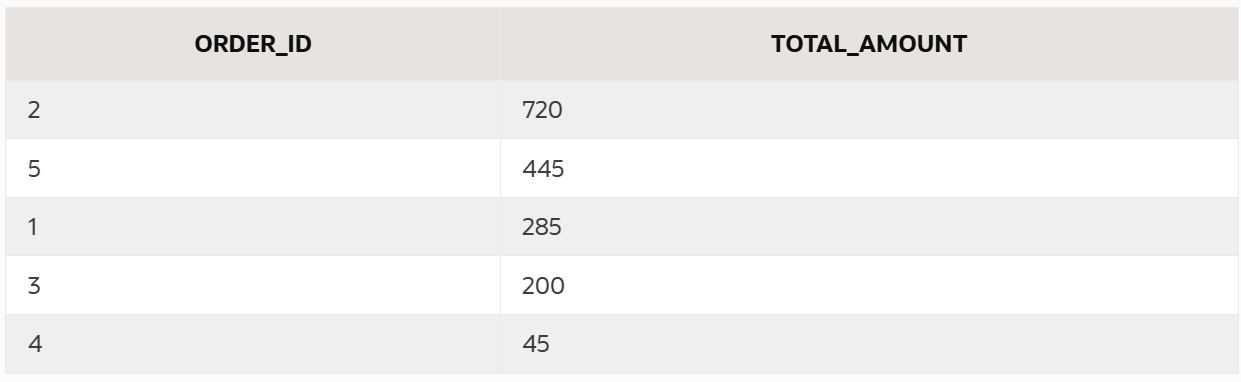
SUM(od.price \* od.quantity) AS total\_amount

FROM OrderDetail od

JOIN Orders o ON od.order\_id = o.order\_id

GROUP BY o.order\_id

ORDER BY total\_amount DESC;



**6. Детали заказа, кто обслуживал, имя клиента и общая сумма.**  
SELECT o.order\_id, c.customer\_name, e.employee\_name,

SUM(od.price \* od.quantity) AS total\_amount

FROM Orders o

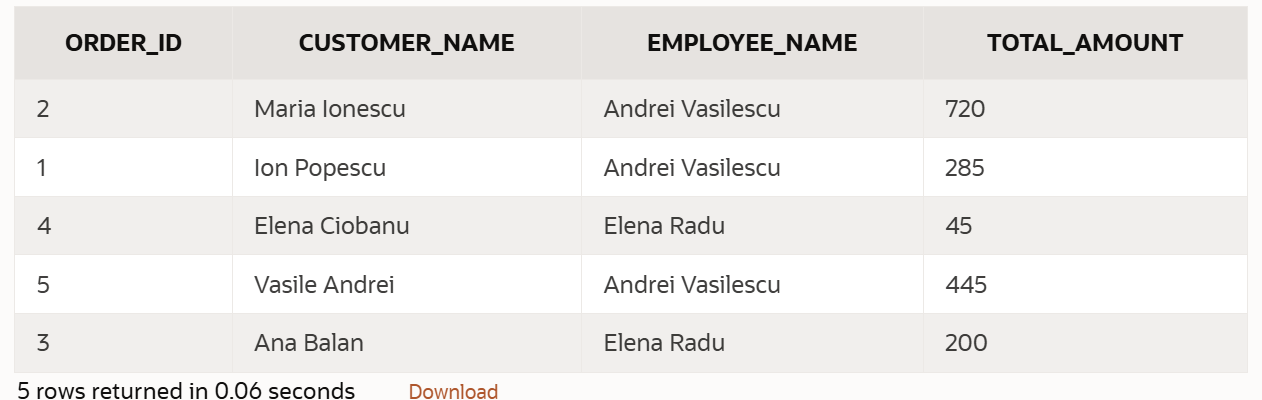
JOIN Customer c ON o.customer\_id = c.customer\_id

JOIN Employee e ON o.employee\_id = e.employee\_id

JOIN OrderDetail od ON o.order\_id = od.order\_id

GROUP BY

o.order\_id, c.customer\_name, e.employee\_name;



**7.** **Имена и Фамилии клиентов, которые заказали блюдо "Sarmale".**  
SELECT DISTINCT c.customer\_name

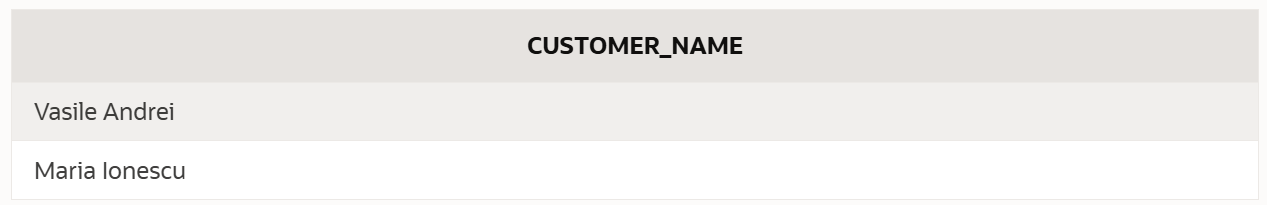
FROM Orders o

JOIN Customer c ON o.customer\_id = c.customer\_id

JOIN OrderDetail od ON o.order\_id = od.order\_id

JOIN Dish d ON od.dish\_id = d.dish\_id

WHERE

d.dish\_name = 'Sarmale';  
  


### **8. Все столы, зарезервированы на дату "2024-12-03".**

SELECT t.table\_number,

TO\_CHAR(r.reservation\_date, 'YYYY-MM-DD HH24:MI:SS') AS reservation\_date, c.customer\_name

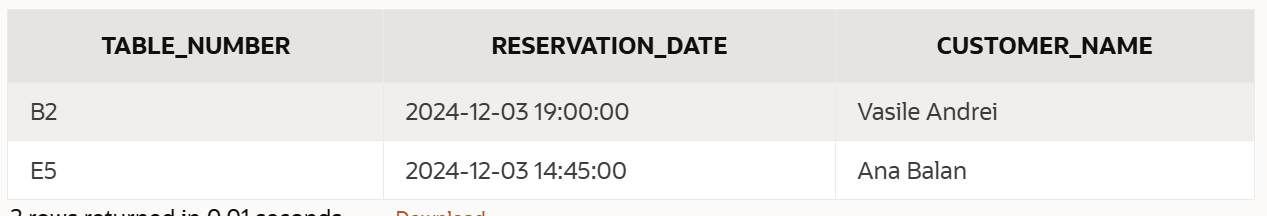
FROM Reservation r

JOIN Table\_Restaurant t ON r.table\_id = t.table\_id

JOIN Customer c ON r.customer\_id = c.customer\_id

WHERE

TO\_CHAR(r.reservation\_date, 'YYYY-MM-DD') = '2024-12-03';



### **9. Все заказы, общая сумма которых больше 400.**

SELECT o.order\_id,

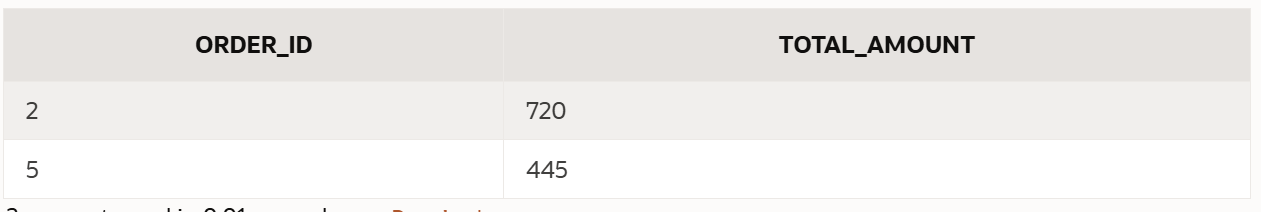
SUM(od.price \* od.quantity) AS total\_amount

FROM OrderDetail od

JOIN Orders o ON od.order\_id = o.order\_id

GROUP BY o.order\_id

HAVING SUM(od.price \* od.quantity) > 400;



**10. Имя и фамилия официанта, который приносил блюдо 'Sarmale**

SELECT e.employee\_name

FROM OrderDetail od

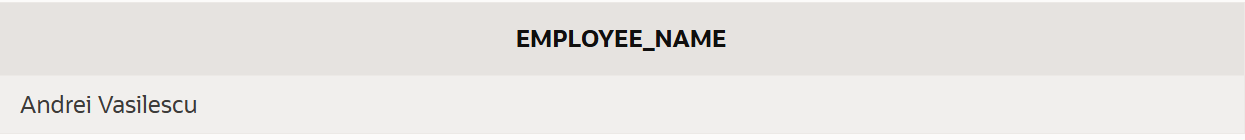
JOIN Orders o ON od.order\_id = o.order\_id

JOIN Employee e ON o.employee\_id = e.employee\_id

JOIN Dish d ON od.dish\_id = d.dish\_id

WHERE

d.dish\_name = 'Sarmale';



### 

### **Создание представлений:**

### **1. Популярные блюда**

**Список блюд с количеством заказов.**

CREATE VIEW PopularDishes AS

SELECT d.dish\_name,

SUM(od.quantity) AS total\_quantity

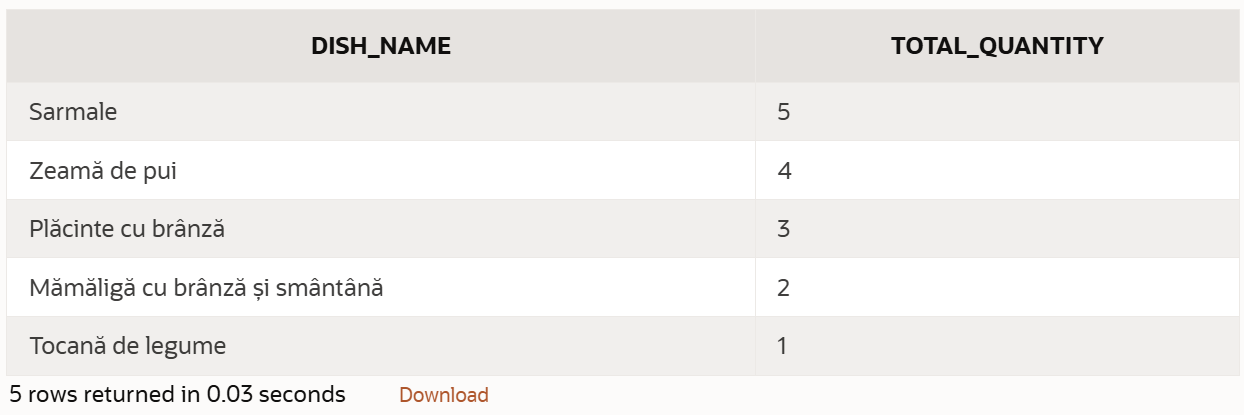
FROM Dish d

JOIN OrderDetail od ON d.dish\_id = od.dish\_id

GROUP BY d.dish\_name

ORDER BY

total\_quantity DESC;

**select \* from PopularDishes;  
**

### **2. Сотрудники и их продажи**

**Показывает каждого сотрудника и общую сумму заказов, которыми он занимался.**

CREATE VIEW EmployeeSales AS

SELECT e.employee\_name,e.position,

SUM(od.quantity \* od.price) AS total\_sales

FROM Employee e

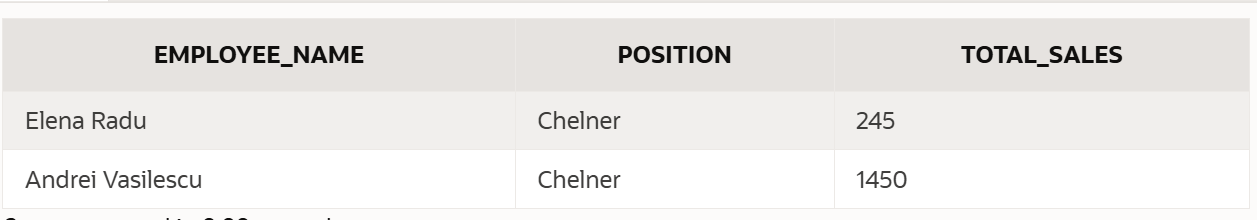
JOIN Orders o ON e.employee\_id = o.employee\_id

JOIN OrderDetail od ON o.order\_id = od.order\_id

GROUP BY

e.employee\_name, e.position;

**select \* from EmployeeSales**

****

### **3. Свободные столы**

**Выводит информацию о свободных столах на определенную дату.**

CREATE VIEW AvailableTables AS

SELECT t.table\_number,t.capacity

FROM Table\_Restaurant t

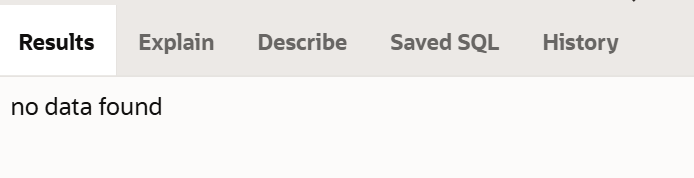
LEFT JOIN

Reservation r ON t.table\_id = r.table\_id

WHERE

r.reservation\_date IS NULL;

**select \* from AvailableTables**

****

### **4. Заполненность столов**

**Сводка о том, какие столы чаще всего резервировали.**

CREATE VIEW TableUsage AS

SELECT t.table\_number,

COUNT(r.reservation\_id) AS reservation\_count

FROM Table\_Restaurant t

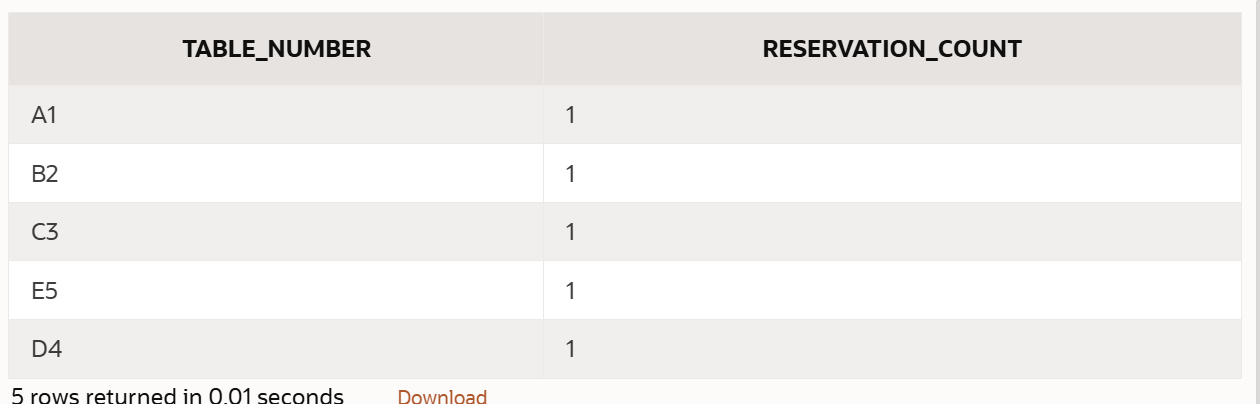
JOIN Reservation r ON t.table\_id = r.table\_id

GROUP BY t.table\_number

ORDER BY

reservation\_count DESC;

**select \* from TableUsage**

****

### **5. Список заказов с общей стоимостью**

**Отображает каждый заказ с его общей стоимостью.**

CREATE VIEW OrderTotal AS

SELECT o.order\_id,o.order\_date, c.customer\_name,

SUM(od.quantity \* od.price) AS total\_price

FROM Orders o

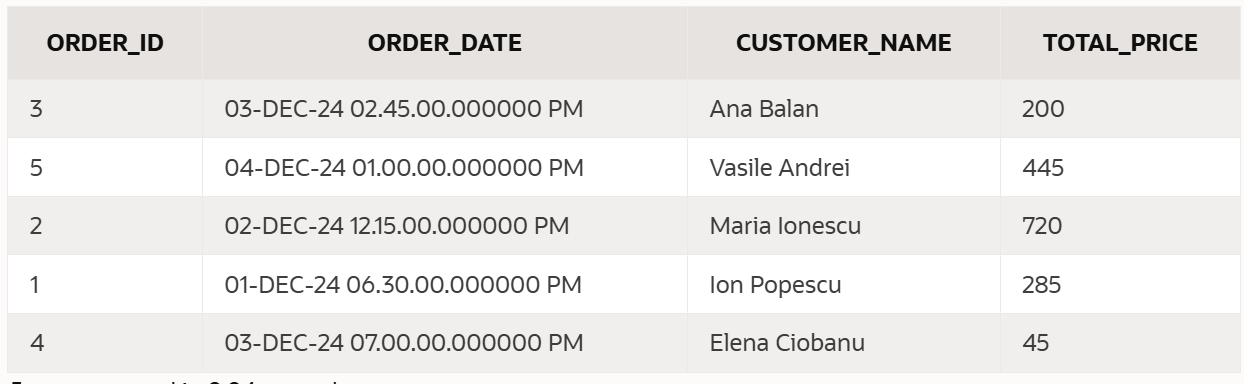
JOIN OrderDetail od ON o.order\_id = od.order\_id

JOIN Customer c ON o.customer\_id = c.customer\_id

GROUP BY

o.order\_id, o.order\_date, c.customer\_name;

**select \* from OrderTotal**

****

**Вывод:**

Работа над проектированием и созданием базы данных для ресторана позволила охватить все основные аспекты управления данными, начиная с моделирования структуры и заканчивая созданием запросов и представлений для анализа.

На этапе проектирования были определены ключевые сущности ресторана, такие как клиенты, сотрудники, блюда, заказы, бронирования и столы. Созданная схема базы данных обеспечила правильное распределение данных и установление взаимосвязей между таблицами. Это позволило добиться удобства в работе с данными.

На этапе написания SQL-запросов был проведён анализ различных вопросов деятельности ресторана, включая поиск информации о клиентах, заказах, популярных блюдах, активности сотрудников и загруженности столов.

Отдельно стоит отметить работу над созданием представлений, которые служат удобным инструментом для формирования готовых отчётов. Представления, такие как анализ популярности блюд, распределение заказов по дням, обеспечивают быстрое получение информации. Они упрощают повседневные задачи анализа данных.

Вся проделанная работа показывает, как грамотно спроектированная база данных и правильно составленные SQL-запросы могут способствовать упрощению доступа к данным и автоматизации рутинных аналитических задач.

Проект позволил не только углубить понимание принципов реляционных баз данных, но и развить навыки анализа и визуализации данных с помощью SQL.