ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«УФИМСКИЙ УНИВЕРСИТЕТ НАУКИ И ТЕХНОЛОГИЙ»

Институт математики, информатики и робототехники
КАФЕДРА ПРОГРАММИРОВАНИЯ И ЭКОНОМИЧЕСКОЙ ИНФОРМАТИКИ

ОТЧЕТ К ЛАБОРАТОРНОЙ РАБОТЕ

по дисциплине «Базы данных»

Выполнила:
Студент(ка) 3 курса очной формы обучения
Направление подготовки (специальность)
09.03.03 Прикладная информатика
Направленность (профиль)
Информационные и вычислительные технологии

Павловский Даниил Анатольевич

Руководитель

_____ Бердникова М. Л.
(подпись)

«____» _____ 2023 г.

УФА – 2023

1. Обследование предметной области

Предметная область: Ресторан

Проект базы данных системы управления рестораном нацелен на ведение записей информации, связанной с клиентами, которые заказывают еду в ресторане, детали заказа клиента, информация о транзакциях, информация о сотрудниках, таких как шеф-повар, официанты и менеджеры, клиентах, информация о транзакциях, информация о сотрудниках, таких как шеф-повар, официанты и менеджеры, которые работают в ресторане. Также ведется учет ингредиентов, необходимых для приготовления блюд, чтобы знать, какие ингредиенты закончились на складе.

Основная цель создания этой системы баз данных - помочь ресторану отслеживать какие блюда доступны в определенное время, проверять наличие или отсутствие ингредиентов, а также иметь информацию, связанную с клиентом, сделавшем заказ, чтобы соотнести заказ с клиентом, который его сделал.

Ограничения:

1) Каждый менеджер имеет уникальный идентификатор, имя, номер телефона и адрес.

2) Каждый шеф-повар ресторана имеет уникальный идентификатор, имя, номер телефона, время прихода и время работы, а также производный атрибут часы, рассчитываемый из общего количества часов, отработанных шеф-поваром от времени прихода до времени ухода.

3) Каждый ингредиент в кладовой имеет уникальный идентификатор, название, дату, которая включает в себя дату изготовления и срок годности, а также количество этих ингредиентов на складе.

4) Каждый клиент, приходящий в ресторан, имеет уникальный идентификатор, имя, телефон номер телефона и адрес.

5) Каждый заказ, принятый от клиента, получает уникальный идентификатор, сделанных клиентов и количество конкретного блюда.

6) Каждое блюдо имеет уникальный идентификатор, название и цену.

7) Каждый официант, работающий в ресторане, имеет уникальный идентификатор, имя, номер телефона, адрес и информация о зарплате.

8) Каждый платеж за конкретный заказ имеет идентификатор, цену, способ транзакции, и дата.

9) Каждый менеджер должен инструктировать как минимум 1 шеф-повара в ресторане, а каждый шеф-повар должен иметь не менее 1 менеджера, который будет их инструктировать.

10) Каждому повару требуется как минимум 1 ингредиент для приготовления блюда, и каждый ингредиент должен быть использоваться 1 или более поварами.

11) У каждого повара должен быть хотя бы 1 заказ на приготовление блюда, и каждый заказ должен выполняться только 1 повар.

12) Каждый клиент должен сделать как минимум 1 заказ, и каждый заказ может быть взят только 1 клиентом заказавший его.

13) В каждом заказе должно быть как минимум 1 блюдо из меню, и одно из блюд должно присутствовать хотя бы в одном заказе.

14) Каждый заказ должен иметь только 1 оплату, и каждая оплата имеет только 1 заказ.

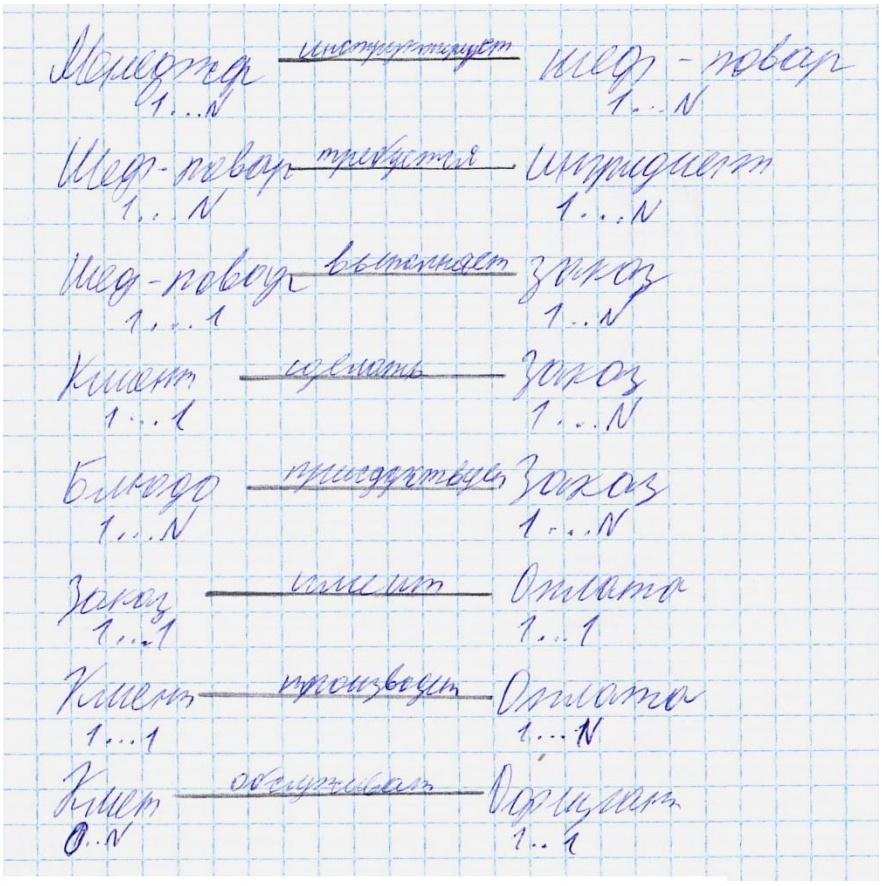
15) Один клиент производит оплату за сделанный заказ, и он должен сделать хотя бы 1 оплату.

16) Каждого клиента обслуживает только 1 официант, и каждый официант может обслуживать 1 клиента или ни одного клиента.

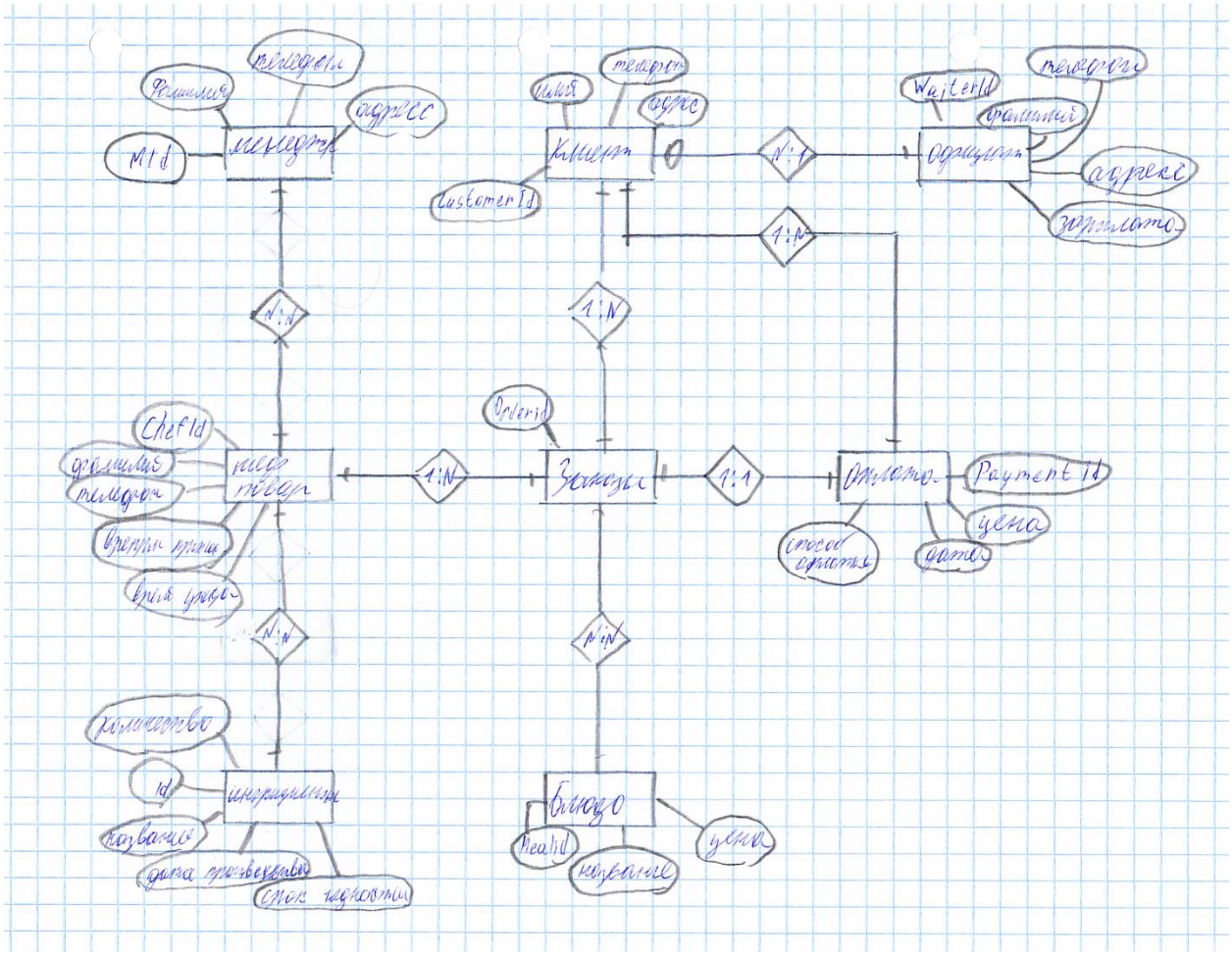
Классы:

- Шеф-повар
- Менеджер
- Ингредиенты
- Клиент
- Заказы
- Оплата
- Официант
- Блюдо

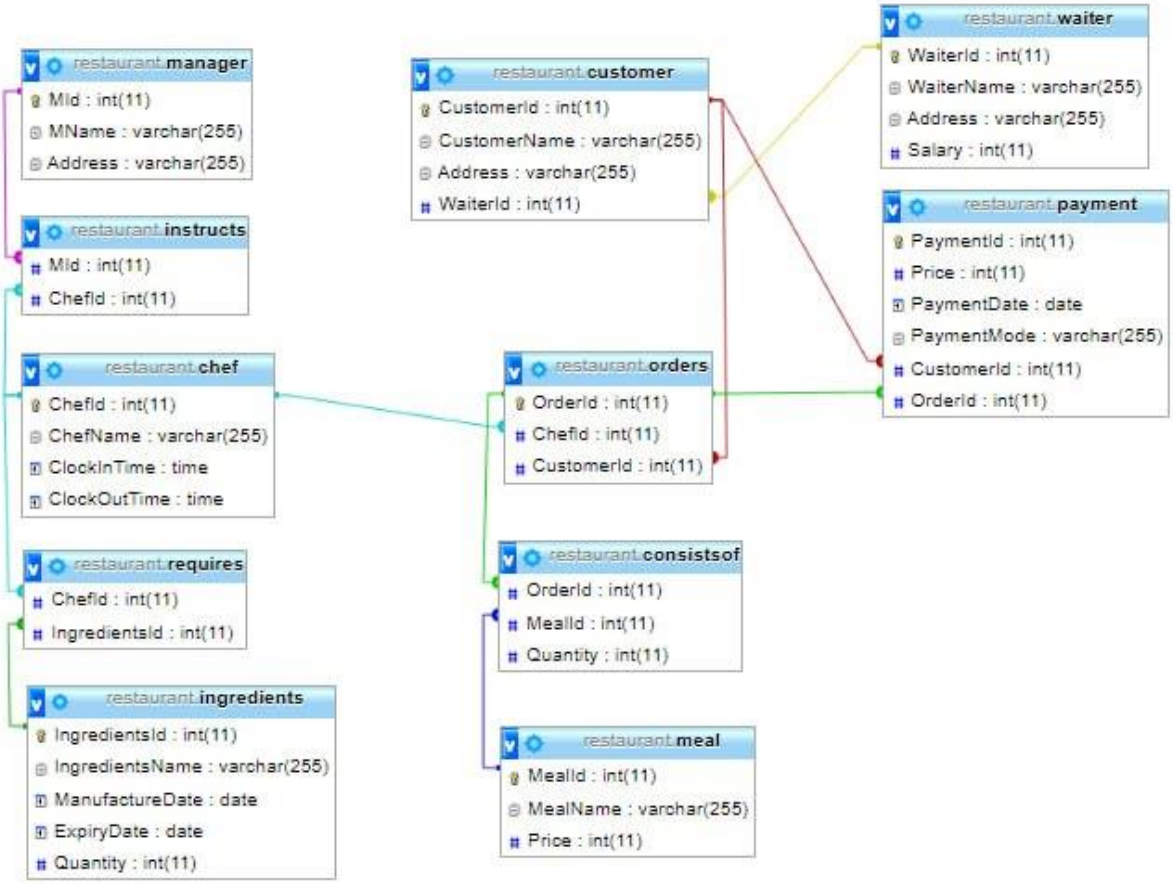
Связи:



2. Модель «сущность-связь»



3. Реляционная модель БД



4. Разработка на языке SQL

1. Запрос на создание базы данных

CREATE DATABASE restaurant

2. Запрос на создание таблиц

- *Создание таблицы “Шеф-повар”:*

```
CREATE TABLE Chef(
   ChefId int NOT NULL,
   ChefName varchar(255) NOT NULL,
   ClockInTime TIME NOT NULL,
   ClockOutTime TIME NOT NULL,
   PRIMARY KEY (ChefId)
);
```

- *Создание таблицы “Менеджер”:*

```
CREATE TABLE Manager(
   MId int NOT NULL,
   MName varchar(255) NOT NULL,
   Address varchar(255),
   PRIMARY KEY (MId)
);
```

- *Создание таблицы “Блюда”:*

```
CREATE TABLE Meal(
   MealId int NOT NULL,
   MealName varchar(255) NOT NULL,
   Price int NOT NULL,
   PRIMARY KEY (MealId)
);
```

- *Создание таблицы “Официанты”:*

```
CREATE TABLE Waiter(
   WaiterId int NOT NULL,
   WaiterName varchar(255) NOT NULL,
   Address varchar(255),
   Salary int,
   PRIMARY KEY (WaiterId)
```

);

- *Создание таблицы “Клиент”:*

```
CREATE TABLE Customer(
   CustomerId INT NOT NULL,
   CustomerName VARCHAR(255) NOT NULL,
   Address VARCHAR(255),
   WaiterId INT, -- Добавляем столбец для внешнего ключа
   PRIMARY KEY (CustomerId),
   FOREIGN KEY (WaiterId) REFERENCES Waiter(WaiterId) -- Определение внешнего ключа
);
```

- *Создание таблицы “Заказ”:*

```
CREATE TABLE Orders(
   OrderId int NOT NULL,
   ChefId int NOT NULL,
   CustomerId int NOT NULL,
   PRIMARY KEY (OrderId),
   FOREIGN KEY (ChefId) REFERENCES Chef (ChefId),
   FOREIGN KEY (CUstomerId) REFERENCES Customer (CustomerId)
);
```

- *Создание таблицы “Ингредиенты”:*

```
CREATE TABLE Ingredients(
  IngredientsId int NOT NULL,
   IngredientsName varchar(255) NOT NULL,
   ManufactureDate DATE NOT NULL,
   ExpiryDate DATE NOT NULL,
   Quantity int NOT NULL,
   PRIMARY KEY (IngredientsId)
);
```

- *Создание таблицы “Оплата”:*

```
CREATE TABLE Payment(
   PaymentId int NOT NULL,
   Price int NOT NULL,
   PaymentDate DATE NOT NULL,
   PaymentMode varchar(255),
   CustomerId int NOT NULL,
   OrderId int NOT NULL,
   PRIMARY KEY (PaymentId),
```

```
   FOREIGN KEY (CustomerId) REFERENCES Customer (CustomerId),
   FOREIGN KEY (OrderId) REFERENCES Orders (OrderId)
);
```

- *Создание таблицы “Инструкции”:*

```
CREATE TABLE Instructs
(
   MId int NOT NULL,
   ChefId int NOT NULL,
   FOREIGN KEY (MId) REFERENCES Manager (MId),
   FOREIGN KEY (ChefId) REFERENCES Chef (ChefId)
);
```

- *Создание таблицы “Запросы”:*

```
CREATE TABLE Requires
(
   ChefId int NOT NULL,
   IngredientsId int NOT NULL,
   FOREIGN KEY (ChefId) REFERENCES Chef (ChefId),
   FOREIGN KEY (IngredientsId) REFERENCES Ingredients (IngredientsId)
);
```

- *Создание таблицы “Состоит”:*

```
CREATE TABLE ConsistsOf
(
   OrderId int NOT NULL,
   MealId int NOT NULL,
   Quantity int NOT NULL,
   FOREIGN KEY (OrderId) REFERENCES Orders (OrderId),
   FOREIGN KEY (MealId) REFERENCES Meal (MealId)
);
```

3. Запросы

– **Запрос 1:** Количество дней с момента истечения срока годности

```
SELECT i.IngredientsId, i.IngredientsName, i.Quantity, DATEDIFF(i.ExpiryDate, i.ManufactureDate) AS "Number of days between manufacture and expiry" FROM Ingredients as i;
```

Выводит:

IngredientsId	IngredientsName	Quantity	Number of days between manufacture and expiry
234	Хумус	5	130
235	Соус Маринара	3	366
236	Сыр Чеддер	6	70
237	Сыр Моцарелла	3	62
238	Кунжутное масло	7	238
239	Сливки	5	-52
240	Паста	8	91

– **Запрос 2:** нахождение часов между временем прихода и ухода клиента

SELECT cf.ChefId, cf.ChefName, TIMEDIFF(cf.ClockOutTime, cf.ClockInTime) AS "Number of Hours worked" FROM Chef as cf;

Выводит:

ChefId	ChefName	Number of Hours worked
30456	Лена	08:59:42
30457	Даниил	10:00:45
30458	Лариса	09:29:59
30459	Ли	08:29:30
30460	Борис	-04:59:00
30461	Данил	09:58:05

– **Запрос 3:** получить название блюда и количество для определенного заказа

SELECT ct.OrderId, m.MealName, ct.Quantity FROM ConsistsOf as ct INNER JOIN Meal as m ON ct.MealId = m.MealId ORDER BY ct.OrderId;

Выводит:

OrderId	MealName	Quantity
1	Сэндвич	1
2	Чесночный сэндвич	1
3	Чесночный сэндвич	1
4	Панини	1
5	Сырный панини	2
6	Сырные крутоны	1
7	Чесночный сэндвич	1
8	Панини	3
9	Сэндвич	1
10	Сырный панини	2
11	Сэндвич	1
12	Сырные крутоны	1
13	Панини	1

– **Запрос 4:** оставшиеся ингредиенты

SELECT r.IngredientsId, i.IngredientsName, i.Quantity-COUNT(r.IngredientsId) as "remaining" FROM Requires as r INNER JOIN Ingredients as i ON r.IngredientsId = i.IngredientsId GROUP BY r.IngredientsId;

Выводит:

IngredientsId	IngredientsName	remaining
234	Хумус	3
235	Соус Маринара	2
236	Сыр Чеддер	5
237	Сыр Моцарелла	2
238	Кунжутное масло	6
239	Сливки	1
240	Паста	6

– **Запрос 5:** клиенты, которые часто ходят в ресторан

SELECT o.CustomerId, cm.CustomerName, COUNT(o.CustomerId) as "Frequent Customers" FROM Orders as o INNER JOIN Customer as cm ON o.CustomerId = cm.CustomerId GROUP BY o.CustomerId;

Выводит:

CustomerId	CustomerName	Frequent Customers
9123410	Саша	1
9123411	Филипп	2
9123412	Генадий	2
9123456	Николай	4
9123457	Рита	1
9123458	Маша	2
9123459	Руслан	1

– **Запрос 6:** понравившиеся блюда среди клиентов

SELECT cs.MealId, m.MealName FROM ConsistsOf as cs INNER JOIN Meal as m ON cs.MealId = m.MealId
GROUP BY cs.MealId HAVING SUM(cs.Quantity) = (SELECT MAX(total) FROM (SELECT SUM(quantity) as total FROM Consists Of
GROUP BY MealId) as a);

Выводит:

MealId	MealName
3	Панини

– **Запрос 7:** Имена шеф- поваров, начинающихся с 'Л'

SELECT * FROM Chef as cf WHERE cf.ChefName LIKE 'Л%';

Выводит:

	ChefId	ChefName	ClockInTime	ClockOutTime
Изменить Копировать Удалить	30456	Лена	08:00:23	17:00:05
Изменить Копировать Удалить	30458	Лариса	08:00:01	17:30:00
Изменить Копировать Удалить	30459	Ли	07:30:33	16:00:03

– **Запрос 8:** сколько клиентов обслуживают официанты

SELECT cm.WaiterId, w.WaiterName, COUNT(cm.CustomerId) as "number of customers handled" FROM Customer as cm INNER JOIN Waiter as w ON cm.WaiterId = w.WaiterId GROUP BY cm.WaiterId;

Выводит:

WaiterId	WaiterName	number of customers handled
20	Белла	4
21	Женя	2
22	Кристина	1

– **Запрос 9:** клиенты, сделавшие больше 1 заказа

SELECT o.CustomerId, cm.CustomerName, cm.Address, COUNT(o.OrderId) FROM Orders as o JOIN Customer as cm ON o.CustomerId = cm.CustomerId GROUP BY o.CustomerId HAVING COUNT(o.OrderId) > 1;

Выводит:

CustomerId	CustomerName	Address	COUNT(o.OrderId)
9123411	Филипп	Пр. Октября 54	2
9123412	Генадий	ул. Заки Валиди 25	2
9123456	Николай	Пр.Октября 10	4
9123458	Маша	ул. Кедровая 6	2

– **Запрос 10:** клиенты, заплатившие больше среднего от общей суммы заказа

SELECT * FROM Payment as p WHERE p.Price > (SELECT AVG(p.Price) FROM Payment as p);

Выводит:

	PaymentId	Price	PaymentDate	PaymentMode	CustomerId	OrderId
Изменить Копировать Удалить	10150	26	2022-05-02	карта	9123410	1
Изменить Копировать Удалить	10153	23	2022-05-02	карта	9123412	4
Изменить Копировать Удалить	10156	45	2022-05-02	наличные	9123456	7
Изменить Копировать Удалить	10158	23	2022-05-03	наличные	9123412	9
Изменить Копировать Удалить	10159	22	2022-05-03	наличные	9123458	10
Изменить Копировать Удалить	10161	22	2022-05-03	наличные	9123456	12