Московский авиационный институт

Институт «Компьютерные науки и прикладная математика»

Кафедра вычислительной математики и программирования

КУРСОВОЙ ПРОЕКТ

по курсу “Базы данных”

Работу выполнили:

Студенты группы М8О-306Б-21

Деревянко Екатерина Андреевна

Леленков Никита Дмитриевич

Работу проверила:

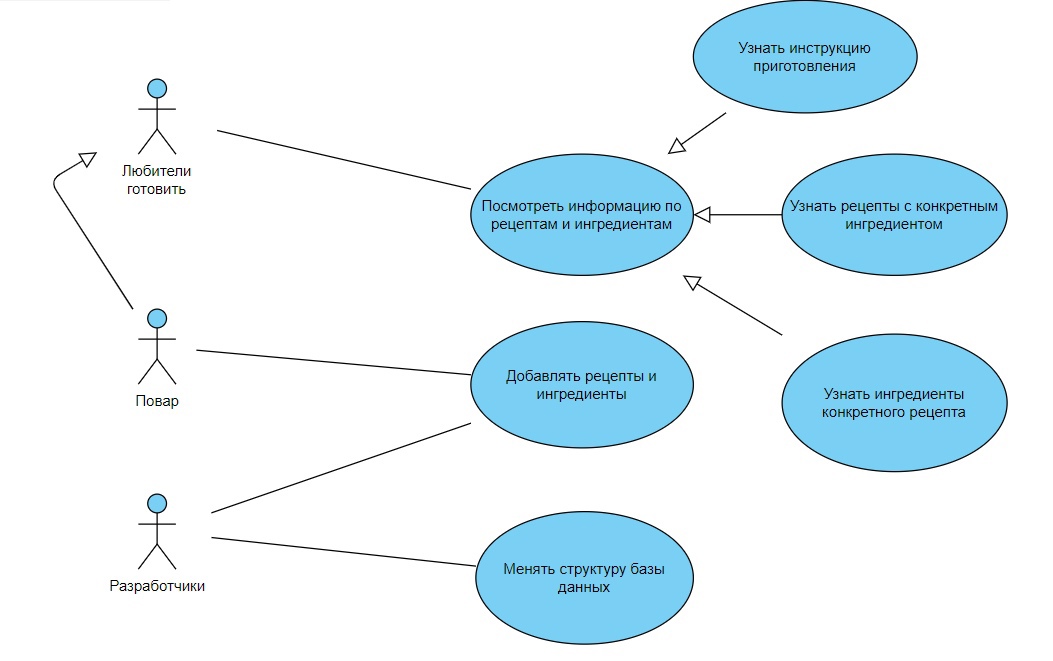
старший преподаватель

Сеченых Полина Алексеевна

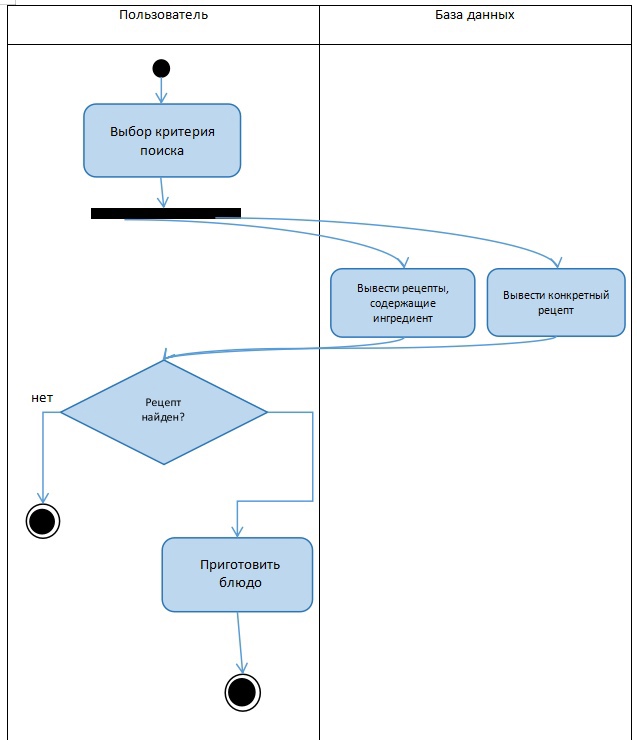
Москва 2023

**Сбор требований**

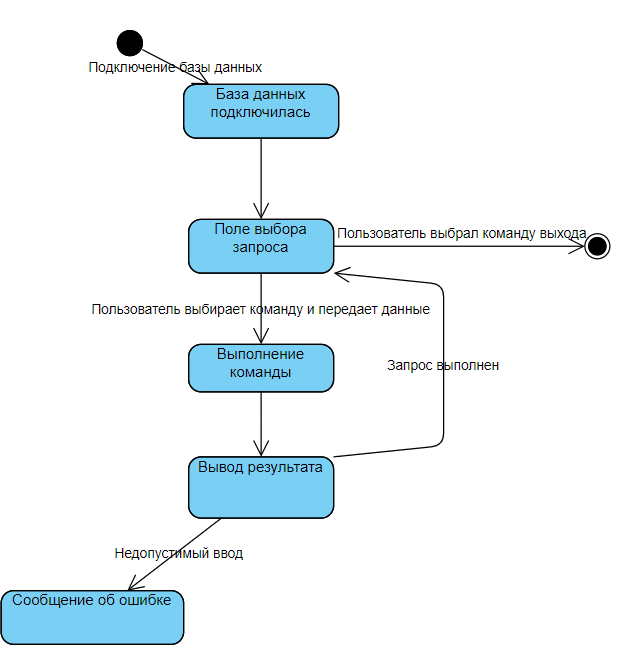
Use case



Activity



State



**Таблицы и схема данных**

Предметная область: кулинарная книга (блюда, рецепты, ингридиенты, подбор меню, расход продуктов по меню).

Разработанная база данных содержит 3 таблицы: рецепты, ингредиенты и их связь. Таблица рецептов и таблица связи находятся в отношении «один-ко-многим», а связи и ингредиентов - «многие-к-одному», таким образом отношение рецептов и ингредиентов «многие-ко-многим». Рассмотрим каждую таблицу.

Таблица «Рецепты».

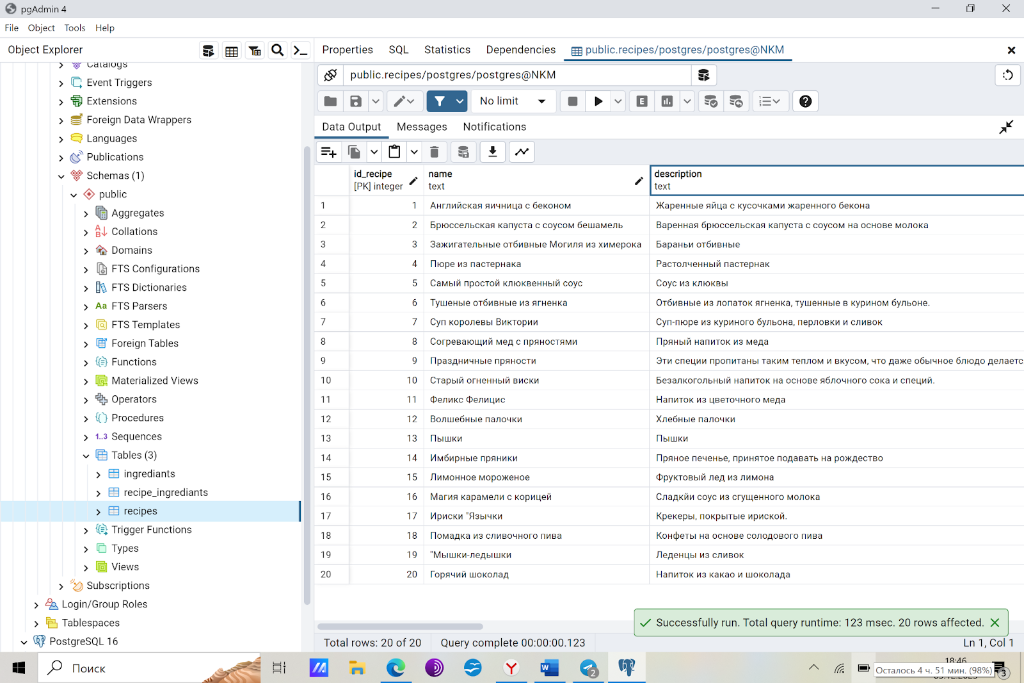
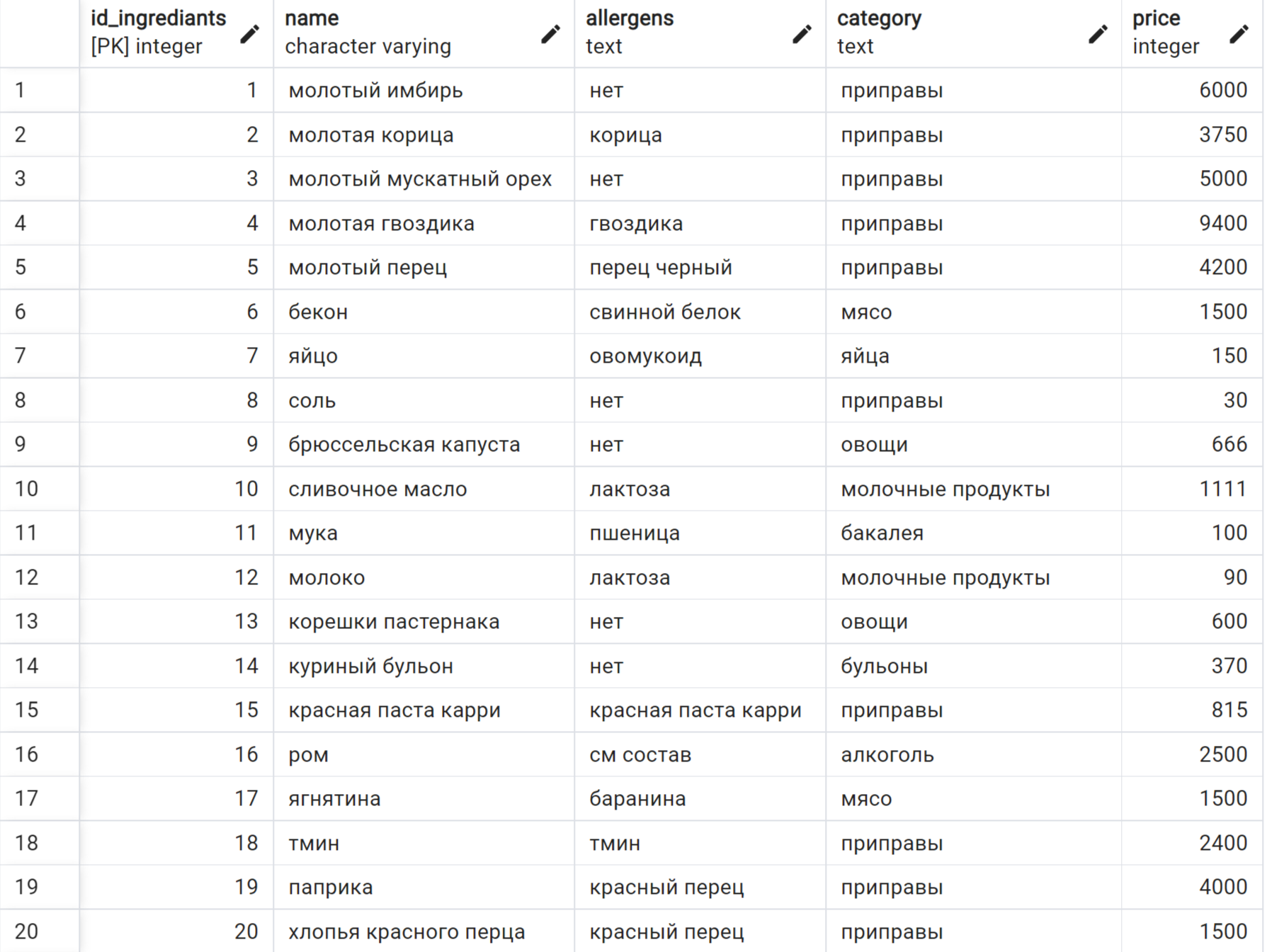


Таблица имеет 5 полей: id\_recipe (тип данных INTEGER), name (тип данных TEXT), description (тип данных TEXT), instruction (тип данных TEXT), time (тип данных INTEGER). Первичный ключ - id\_recipe.

Таблица «Ингредиенты».



Создание:

Таблица имеет 5 полей: id\_ingrediants (тип данных INTEGER), name (тип данных VARCHAR), allergens (тип данных TEXT), category (тип данных TEXT), price (тип данных INTEGER). Первичный ключ - id\_ingrediants.

Таблица связи.

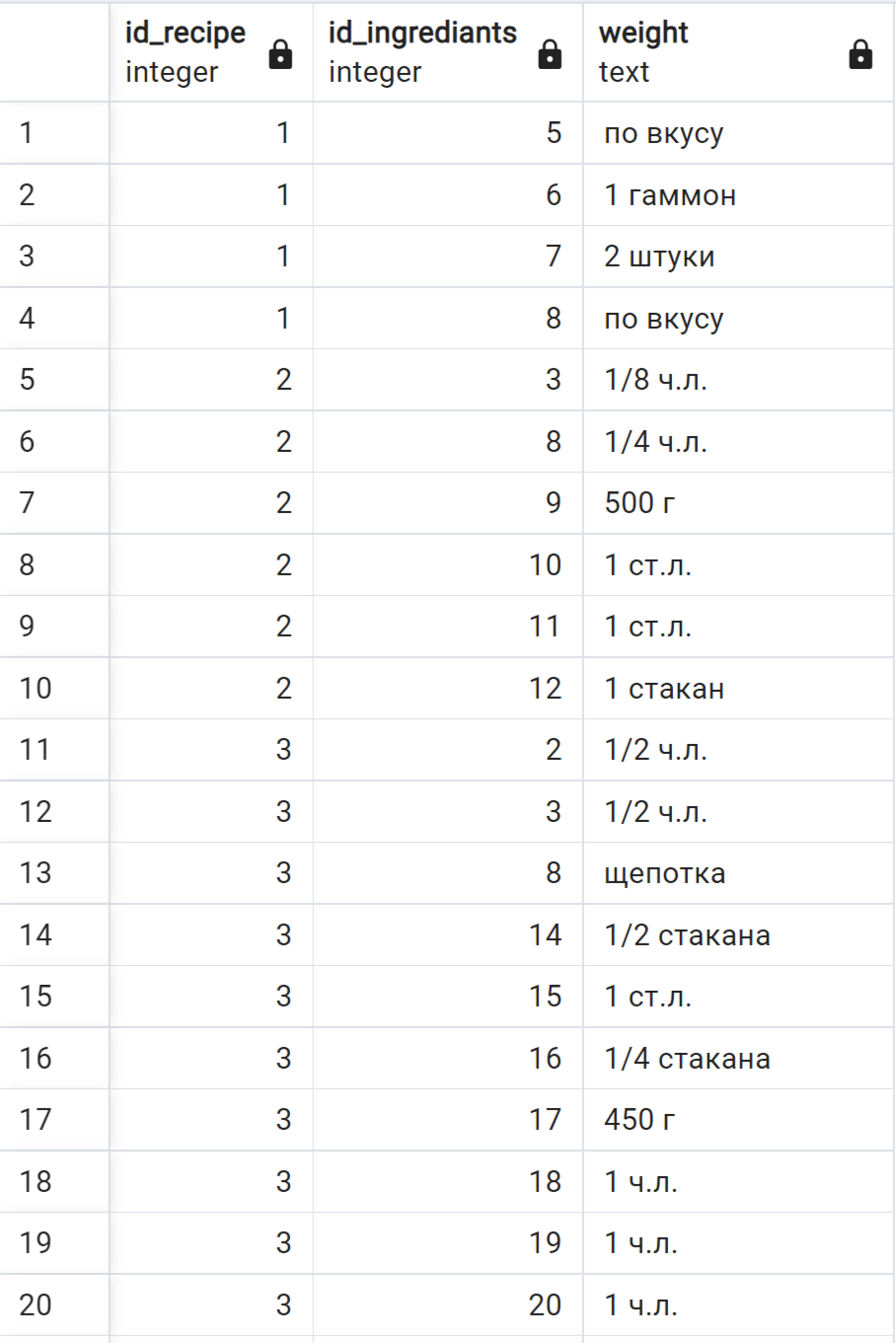


Таблица имеет 3 поля: id\_recipe (тип данных INTEGER), id\_ingrediants (тип данных INTEGER), weight (тип данных TEXT). Первые два поля являются ключами связи.

**Разработка процедур на PL/SQL**

1. Назначение: удаление ингредиента из таблицы по его названию

*CREATE OR REPLACE PROCEDURE delete\_ingredient\_by\_name(*

*IN \_name TEXT*

*) LANGUAGE plpgsql*

*AS $$*

*BEGIN*

*DELETE FROM ingrediants WHERE name = \_name;*

*END;*

*$$;*

1. Назначение: удаление рецепта по названию

*CREATE OR REPLACE PROCEDURE delete\_recipe\_by\_name(*

*IN \_name TEXT*

*) LANGUAGE plpgsql*

*AS $$*

*BEGIN*

*DELETE FROM recipes WHERE name = \_name;*

*END;*

*$$;*

1. Назначение: контролирование цены, добавленного ингредиента, она не может быть меньше или равна 0.

*CREATE OR REPLACE FUNCTION check\_ingredient\_price()*

*RETURNS TRIGGER AS $$*

*BEGIN*

*IF NEW.price <= 0 THEN*

*RAISE EXCEPTION 'Price must be greater than zero';*

*END IF;*

*RETURN NEW;*

*END;*

*$$ LANGUAGE plpgsql;*

*CREATE TRIGGER check\_ingredient\_price\_before\_insert\_or\_update*

*BEFORE INSERT OR UPDATE ON ingrediants*

*FOR EACH ROW EXECUTE FUNCTION check\_ingredient\_price();*

**Клиентское приложение**

import psycopg2

def ingredient\_recipes(name, cursor):

name\_pattern = '%' + name + '%'

cursor.execute(

"""

SELECT i.name, r.name

FROM ingrediants i

JOIN recipe\_ingrediants r\_i ON i.id\_ingrediants = r\_i.id\_ingrediants

JOIN recipes r ON r.id\_recipe = r\_i.id\_recipe

WHERE i.name LIKE %s;

""",

(name\_pattern,)

)

results = cursor.fetchall()

return results

def recipe\_ingredients(name, cursor):

name\_pattern = '%' + name + '%'

cursor.execute(

"""

SELECT r.name, i.name, r\_i.weight

FROM recipes r

JOIN recipe\_ingrediants r\_i ON r.id\_recipe = r\_i.id\_recipe

JOIN ingrediants i ON i.id\_ingrediants = r\_i.id\_ingrediants

WHERE r.name LIKE %s;

""",

(name\_pattern,)

)

results = cursor.fetchall()

return results

def recipe\_instruction(name, cursor):

name\_pattern = '%' + name + '%'

cursor.execute(

"""

SELECT r.name, r.insruction, r.times

FROM recipes r

WHERE r.name LIKE %s;

""",

(name\_pattern,)

)

results = cursor.fetchall()

return results

def allergens(name, cursor):

name\_pattern = '%' + name + '%'

cursor.execute(

"""

SELECT i.name

FROM ingrediants i

WHERE i.allergens LIKE %s;

""",

(name\_pattern,)

)

results = cursor.fetchall()

return results

def id\_max(cursor):

cursor.execute(

"""

SELECT MAX(id\_ingrediants) FROM ingrediants

""",

)

results = cursor.fetchone()[0]

return results + 1 if results else 1

def id\_max\_recipes(cursor):

cursor.execute(

"""

SELECT MAX(id\_recipe) FROM recipes

""",

)

results = cursor.fetchone()[0]

return results + 1 if results else 1

def insert\_ingredient(cursor, key, name, allergens, category, price):

try:

cursor.execute(

"""

INSERT INTO ingrediants (id\_ingrediants, name, allergens, category, price)

VALUES (%s, %s, %s, %s, %s);

""",

(key, name, allergens, category, price)

)

cursor.execute(

"""

SELECT \* FROM ingrediants WHERE id\_ingrediants = %s;

""",

(key,)

)

results = cursor.fetchall()

return results

except psycopg2.Error as e:

print(f"An error occurred: {e}")

return None

def insert\_recipes(cursor, key, name, description, instruction, times):

cursor.execute(

"""

INSERT INTO recipes (id\_recipe, name, description, insruction, times)

VALUES (%s, %s, %s, %s, %s);

""",

(key, name, description, instruction, times)

)

cursor.execute(

"""

SELECT \* FROM recipes WHERE id\_recipe = %s;

""",

(key,)

)

results = cursor.fetchall()

return results

def update(cursor, price, name):

try:

cursor.execute(

"SELECT price FROM ingrediants WHERE name = %s;",

(name,)

)

original\_price = cursor.fetchone()

original\_price = original\_price[0] if original\_price else None

cursor.execute(

"UPDATE ingrediants SET price = %s WHERE name = %s;",

(price, name,)

)

cursor.execute(

"SELECT price FROM ingrediants WHERE name = %s;",

(name,)

)

updated\_price = cursor.fetchone()

updated\_price = updated\_price[0] if updated\_price else None

return updated\_price != original\_price

except psycopg2.Error as e:

print(f"An error occurred: {e}")

return None

def interactive\_menu():

connection = psycopg2.connect(

host="localhost",

database="postgres",

user="postgres",

password="1111"

)

cursor = connection.cursor()

while True:

print(" ")

print("Выберите команду:")

print("1. Найти все рецепты с конкретным ингредиентом")

print("2. Найти конкретный рецепт и ингредиенты к нему")

print("3. Узнать инструкцию и время приготовления конкретного рецепта")

print("4. Проверить ингредиенты на конкретный аллерген")

print("5. Удалить ингредиент по названию")

print("6. Удалить рецепт по названию")

print("7. Добавить новый ингредиент")

print("8. Добавить новый рецепт")

print("9. Изменить цену ингредиента")

print("10. Выйти")

choice = input("Введите число от 1 до 10: ")

if choice == "1":

ingredient = input("Введите название ингредиента: ")

results = ingredient\_recipes(ingredient, cursor)

for result in results:

print(result)

elif choice == "2":

recipe = input("Введите название рецепта: ")

results = recipe\_ingredients(recipe, cursor)

for result in results:

print(result)

elif choice == "3":

recipe = input("Введите название рецепта: ")

results = recipe\_instruction(recipe, cursor)

for result in results:

print(result)

elif choice == "4":

allergen = input("Введите название аллергена: ")

results = allergens(allergen, cursor)

for result in results:

print(result)

elif choice == "5":

ingredient = input("Введите название ингредиента для удаления: ")

cursor.execute("CALL delete\_ingredient\_by\_name(%s);", (ingredient,))

connection.commit()

print("Ингредиент удалён.")

elif choice == "6":

recipe = input("Введите название рецепта для удаления: ")

cursor.execute("CALL delete\_recipe\_by\_name(%s);", (recipe,))

connection.commit()

print("Рецепт удалён.")

elif choice == "7":

key = id\_max(cursor)

name = input("Введите название ингредиента: ")

allergens\_input = input("Введите информацию об аллергенах: ")

category = input("Введите категорию ингредиента: ")

price = input("Введите цену ингредиента: ")

result = insert\_ingredient(cursor, key, name, allergens\_input, category, price)

connection.commit()

if result:

print("Новый ингредиент добавлен.")

else:

print("Ошибка добавления ингредиента.")

elif choice == "8":

keyr = id\_max\_recipes(cursor)

name = input("Введите название рецепта: ")

description = input("Введите описание рецепта: ")

instruction = input("Введите инструкцию приготовления: ")

times = input("Введите время приготовления: ")

result = insert\_recipes(cursor, keyr, name, description, instruction, times)

connection.commit()

if result:

print("Новый рецепт добавлен.")

else:

print("Ошибка добавления рецепта.")

elif choice == "9":

names = input("Введите название ингредиента: ")

price = input("Введите новую цену ингредиента: ")

if update(cursor, price, names):

connection.commit()

print("Цена успешно изменена")

else:

print("Не удалось изменить цену. Возможно, ингредиент не найден или цена уже соответствует введенной.")

elif choice == "10":

print("Выход из программы...")

break

else:

print("Неверный выбор, пожалуйста, введите число от 1 до 9.")

cursor.close()

connection.close()

if \_\_name\_\_ == "\_\_main\_\_":

interactive\_menu()

**Список использованных источников**

1. Лукин, В. Н. Введение в проектирование баз данных [Текст] / В. Н. Лукин — 3-е изд.. — М.: Вузовая книга, 2015 — 144 c.
2. Введение в PostgreSQL с Python +Psycopg2 / [Электронный ресурс] // PythonRu : [сайт]. — URL: https://pythonru.com/biblioteki/vvedenie-v-postgresql-s-python-psycopg2?ysclid=lqib2k7jcq362048528 (дата обращения: 25.12.2023).