# Базы данных

**Практическая работа № 17 «Связь СУБД SQL и Python.»**

## **Цель:**

## Произвести связь базы данных в SQL и Python, изучить операции по манипулированию с данными БД, а также созданию простейших пользовательских функций.

## **Задание.**

Соединение Python с БД SQL;

Выполнение запроса SELECT;

Создание новой таблицы с помощью Python и заполнение ее данными;

Создание простейших пользовательских функций в PostgreSQL и Python.

## **Ход выполнения.**

Создание базы данных "Прокат" с таблицами

Таблицы:

* Клиенты - содержит данные о клиентах.
* Автомобили - список автомобилей для проката.
* Заказы - связь клиентов с автомобилями через заказы.

**Код Python:**

import sqlite3

import os

current\_dir = os.path.dirname(os.path.abspath(\_\_file\_\_))

db\_path = os.path.join(current\_dir, 'Прокатт\_ПР17.db')

**Создание и подключение к базе данных**

conn = sqlite3.connect('rental.db')

cursor = conn.cursor()

**Создание таблиц**

cursor.execute("""

CREATE TABLE IF NOT EXISTS Клиенты (

id INTEGER PRIMARY KEY AUTOINCREMENT,

имя TEXT NOT NULL,

телефон TEXT NOT NULL

);

""")

cursor.execute("""

CREATE TABLE IF NOT EXISTS Автомобили (

id INTEGER PRIMARY KEY AUTOINCREMENT,

марка TEXT NOT NULL,

модель TEXT NOT NULL,

стоимость\_в\_день REAL NOT NULL

);

""")

cursor.execute("""

CREATE TABLE IF NOT EXISTS Заказы (

id INTEGER PRIMARY KEY AUTOINCREMENT,

клиент\_id INTEGER NOT NULL,

автомобиль\_id INTEGER NOT NULL,

дата\_начала TEXT NOT NULL,

дата\_окончания TEXT NOT NULL,

FOREIGN KEY (клиент\_id) REFERENCES Клиенты(id),

FOREIGN KEY (автомобиль\_id) REFERENCES Автомобили(id)

);

""")

**Заполнение таблиц данными**

cursor.executemany("""

INSERT INTO Клиенты (имя, телефон) VALUES (?, ?);

""", [

("Иван Иванов", "+79001112233"),

("Петр Петров", "+79004445566"),

("Светлана Сидорова", "+79007778899"),

("Мария Смирнова", "+79001234567"),

("Дмитрий Кузнецов", "+79009876543")

])

cursor.executemany("""

INSERT INTO Автомобили (марка, модель, стоимость\_в\_день) VALUES (?, ?, ?);

""", [

("Toyota", "Camry", 3000),

("BMW", "X5", 7000),

("Audi", "A6", 5000),

("Hyundai", "Solaris", 2000),

("Lada", "Granta", 1500)

])

cursor.executemany("""

INSERT INTO Заказы (клиент\_id, автомобиль\_id, дата\_начала, дата\_окончания) VALUES (?, ?, ?, ?);

""", [

(1, 2, "2024-11-01", "2024-11-10"),

(2, 3, "2024-11-05", "2024-11-12"),

(3, 1, "2024-11-15", "2024-11-20"),

(4, 4, "2024-11-18", "2024-11-25"),

(5, 5, "2024-11-20", "2024-11-30")

])

conn.commit()

**Выполнение запроса SELECT**

print("Список заказов:")

cursor.execute("""

SELECT

Клиенты.имя AS клиент,

Автомобили.марка || ' ' || Автомобили.модель AS автомобиль,

Заказы.дата\_начала,

Заказы.дата\_окончания

FROM Заказы

JOIN Клиенты ON Заказы.клиент\_id = Клиенты.id

JOIN Автомобили ON Заказы.автомобиль\_id = Автомобили.id;

""")

for row in cursor.fetchall():

print(row)

**Создание новой таблицы с помощью Python и заполнение данными**

cursor.execute("""

CREATE TABLE IF NOT EXISTS Возвраты (

id INTEGER PRIMARY KEY AUTOINCREMENT,

заказ\_id INTEGER NOT NULL,

дата\_возврата TEXT NOT NULL,

FOREIGN KEY (заказ\_id) REFERENCES Заказы(id)

);

""")

cursor.executemany("""

INSERT INTO Возвраты (заказ\_id, дата\_возврата) VALUES (?, ?);

""", [

(1, "2024-11-10"),

(2, "2024-11-12"),

(3, "2024-11-20"),

(4, "2024-11-25"),

(5, "2024-11-30")

])

conn.commit()

**Простейшие пользовательские функции на Python**

def calculate\_rental\_cost(стоимость\_в\_день, дни):

"""Рассчитать стоимость аренды."""

return стоимость\_в\_день \* дни

print("\nРассчитанные стоимости аренды:")

cursor.execute("""

SELECT

Автомобили.марка || ' ' || Автомобили.модель AS автомобиль,

(JULIANDAY(Заказы.дата\_окончания) - JULIANDAY(Заказы.дата\_начала)) AS дни,

Автомобили.стоимость\_в\_день

FROM Заказы

JOIN Автомобили ON Заказы.автомобиль\_id = Автомобили.id;

""")

for row in cursor.fetchall():

автомобиль, дни, стоимость\_в\_день = row

итоговая\_стоимость = calculate\_rental\_cost(стоимость\_в\_день, дни)

print(f"{автомобиль}: {итоговая\_стоимость} рублей")

conn.close()

**Описание шагов:**

**Создание базы данных и таблиц:**

Создаются три таблицы: Клиенты, Автомобили и Заказы.

**Заполнение таблиц данными:**

Используется executemany() для добавления нескольких записей.

**Выполнение SELECT-запроса:**

Запрос для получения списка заказов с данными о клиенте и автомобиле.

**Создание новой таблицы и добавление данных:**

Создается таблица Возвраты, в которую записываются данные о возврате автомобилей.

**Создание пользовательской функции на Python:**

Функция для расчета стоимости аренды на основе стоимости в день и количества дней.

**Результаты:**

Скрипт создаст SQLite-базу данных, выполнит все манипуляции и выведет список заказов и рассчитанные стоимости аренды.

