Студент группы ИС-23 Климов Я.В.

Практическое занятие №15

Тема: Составление программ, работы с БД в IDE PyCharm Community.

Цель: Закрепить усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрести навыки составление программ, работы с БД в IDE PyCharm Community

Задача №1

Постановка задачи.

Приложение КОММАНДИРОВОЧНЫЕ РАСХОДЫ для автоматизированного финансового контроля на предприятии. БД должна содержать таблицу Статьи расходов, имеющую следующую структуру записи: № приказа, Фамилия, Место командировки, Оплата, Аванс, Вид расходов, Сумма расходов.

Текст программы:

```
#Приложение КОММАНДИРОВОЧНЫЕ РАСХОДЫ для
автоматизированного
# контроля на предприятии. БД должна содержать
таблицу Статьи расходов,
#имеющую следующую структуру записи: № приказа,
Фамилия, Место командировки,
#Оплата, Аванс, Вид расходов, Сумма расходов

import sqlite3

conn = sqlite3.connect('expenses.db') # Подключение к
базе данных expenses.db
cursor = conn.cursor()

#Создание таблицы
```

```
cursor.execute('''
CREATE TABLE IF NOT EXISTS expenses (
    id INTEGER PRIMARY KEY,
    order_number INTEGER,
    employee_name TEXT,
    destination TEXT,
    payment REAL,
    advance REAL,
    expense_type TEXT,
    expense_amount REAL
)
''')

def add_expense():
    order_number = int(input("Введите номер приказа:
"))
    employee_name = input("Введите фамилию сотрудника:
")
    destination = input("Введите место командировки:
")
    payment = float(input("Введите сумму оплаты: "))
    advance = float(input("Введите сумму аванса: "))
    expense_type = input("Введите вид расходов: ")
    expense_amount = float(input("Введите сумму
расходов: "))

    cursor.execute('''
        INSERT INTO expenses (order_number,
employee_name, destination, payment, advance,
expense_type, expense_amount)
        VALUES (?, ?, ?, ?, ?, ?, ?)
    ''', (order_number, employee_name, destination,
payment, advance, expense_type, expense_amount))
    conn.commit()

def search_expense():
    order_number = int(input("Введите номер приказа
для поиска: ")) # Получение номера приказа для поиска
```

```
    cursor.execute('''
        SELECT * FROM expenses
        WHERE order_number = ?
    ''', (order_number,))
    result = cursor.fetchone() # Получение результата
запроса


    # Если запись найдена вывод ее данных
    if result:
        print("Найденная запись:")
        print(f"№ приказа: {result[1]}")
        print(f"Фамилия: {result[2]}")
        print(f"Место командировки: {result[3]}")
        print(f"Оплата: {result[4]}")
        print(f"Аванс: {result[5]}")
        print(f"Вид расходов: {result[6]}")
        print(f"Сумма расходов: {result[7]}")
    else:
        print("Запись не найдена.")

def delete_expense():
    order_number = int(input("Введите номер приказа
для удаления: ")) # Получение номера приказа для
удаления
    cursor.execute('''
        DELETE FROM expenses
        WHERE order_number = ?
    ''', (order_number,))
    conn.commit() # Сохранение изменений в базе данных
    print("Запись успешно удалена.")

def edit_expense():
    order_number = int(input("Введите номер приказа
для редактирования: ")) # Получение номера приказа
для редактирования
    cursor.execute('''
        SELECT * FROM expenses
```

```
        WHERE order_number = ?
    ''', (order_number,))
    result = cursor.fetchone() # Получение результата
запроса


    # Если запись найдена, вывод ее данных
    if result:
        print("Найденная запись:")
        print(f"№ приказа: {result[1]}")
        print(f"Фамилия: {result[2]}")
        print(f"Место командировки: {result[3]}")
        print(f"Оплата: {result[4]}")
        print(f"Аванс: {result[5]}")
        print(f"Вид расходов: {result[6]}")
        print(f"Сумма расходов: {result[7]}")


        # Получение от пользователя новых значений для
редактирования
        new_order_number = input("Введите новый номер
приказа (оставьте пустым, если не изменяется): ")
        if new_order_number.strip(): # Проверяем ввел
ли пользователь новое значение для номера приказа
            new_order_number = int(new_order_number)
            cursor.execute('''
                UPDATE expenses
                SET order_number = ?
                WHERE order_number = ?
            ''', (new_order_number, order_number))
        else:
            new_order_number = order_number # Если
пользователь не ввел новое значение, используем
старое

        new_employee_name = input("Введите новую
фамилию сотрудника (оставьте пустым, если не
изменяется): ")
```

```
        if new_employee_name.strip(): # Проверяем ввел
ли пользователь новое значение для фамилии сотрудника
            cursor.execute('''
                UPDATE expenses
                SET employee_name = ?
                WHERE order_number = ?
            ''', (new_employee_name, order_number))
        else:
            new_employee_name = result[2]

        new_destination = input("Введите новое место
командировки (оставьте пустым, если не изменяется):
")
        if new_destination.strip(): # Проверяем ввел
ли пользователь новое значение для места командировки
            cursor.execute('''
                UPDATE expenses
                SET destination = ?
                WHERE order_number = ?
            ''', (new_destination, order_number))
        else:
            new_destination = result[3]

        new_payment = input("Введите новую сумму
оплаты (оставьте пустым, если не изменяется): ")
        if new_payment.strip(): # Проверяем ввел ли
пользователь новое значение для суммы оплаты
            new_payment = float(new_payment)
            cursor.execute('''
                UPDATE expenses
                SET payment = ?
                WHERE order_number = ?
            ''', (new_payment, order_number))
        else:
            new_payment = result[4]

        new_advance = input("Введите новую сумму
аванса (оставьте пустым, если не изменяется): ")
```

```
        if new_advance.strip(): # Проверяем ввел ли
пользователь новое значение для суммы аванса
            new_advance = float(new_advance)
            cursor.execute('''
                UPDATE expenses
                SET advance = ?
                WHERE order_number = ?
            ''', (new_advance, order_number))
        else:
            new_advance = result[5]

        new_expense_type = input("Введите новый вид
расходов (оставьте пустым, если не изменяется): ")
        if new_expense_type.strip(): # Проверяем ввел
ли пользователь новое значение для вида расходов
            cursor.execute('''
                UPDATE expenses
                SET expense_type = ?
                WHERE order_number = ?
            ''', (new_expense_type, order_number))
        else:
            new_expense_type = result[6]

        new_expense_amount = input("Введите новую
сумму расходов (оставьте пустым, если не изменяется):
")
        if new_expense_amount.strip(): # Проверяем
ввел ли пользователь новое значение для суммы
расходов
            new_expense_amount =
float(new_expense_amount)
            cursor.execute('''
                UPDATE expenses
                SET expense_amount = ?
                WHERE order_number = ?
            ''', (new_expense_amount, order_number))
        conn.commit() # Сохранение изменений в базе
данных
```

```
        print("Запись успешно отредактирована.")
    else:
        print("Запись не найдена.")


# Основной цикл программы
while True:
    print("\n1. Добавить запись")
    print("2. Поиск записи")
    print("3. Удалить запись")
    print("4. Редактировать запись")
    print("5. Выйти из программы")
    choice = int(input("Выберите операцию: "))

    if choice == 1:
        add_expense()
    elif choice == 2:
        search_expense()
    elif choice == 3:
        delete_expense()
    elif choice == 4:
        edit_expense()
    elif choice == 5:
        break
    else:
        print("Неверный выбор. Пожалуйста, выберите
одну из доступных операций.")

conn.close() # Закрытие соединения с базой данных
```

Протокол работы программы:
1. Добавить запись
2. Поиск записи
3. Удалить запись
4. Редактировать запись
5. Выйти из программы
Выберите операцию: 5

Вывод: В процессе выполнения практического занятия выработал навыки Составление программ, работы с БД в IDE PyCharm Community. .
Были использованы языковые конструкции `import,def,if,else,break, while True,\n.`
Выполнены разработка кода, отладка, тестирование, оптимизация программного кода.
Готовые программные коды выложены на GitHub.