**Практическая работа №4. Создание класса базы данных. Реализация основных операций с данными (CRUD) в консольном приложении.**

Выполнил студент группы ИСП21.1А Селянский Дмитрий

Предметная область: Ископаемые

Задание.

1. Создать проект PyCharm.
2. Создать в нем 5 модулей .py
3. В первом модуле описать класс, соответствующий предметной области по варианту. Класс должен содержать:

- конструктор с подключением к базе данных, запросы на создание таблиц (CREATE TABLE);

- методы отключения базы данных, просмотра данных из таблиц, добавления данных, удаления данных, редактирования данных, поиска данных по критерию.

1. Остальные модули содержат методы и программный код, демонстрирующие работу с данными из базы данных.
2. Оформить отчет о выполненной работе:

- заголовок,

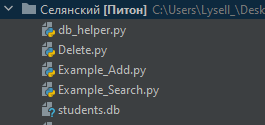
- кто выполнил,

- описание структуры проекта,

- программный код модулей,

- результаты выполнения модулей (скрины).

1. Создал проект в PyCharm и создал в нём 5 модулей



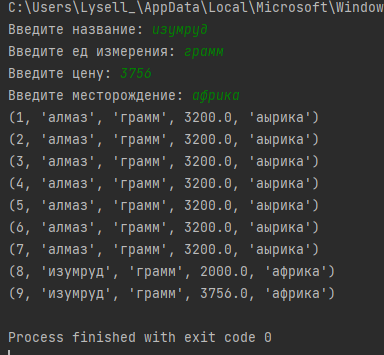
2. Перый модуль db\_helper.py

import sqlite3  
# создание класса БД  
  
class Student:  
 # конструктор класса  
 def \_\_init\_\_(self):  
 self.con = sqlite3.connect("students.db")  
 self.cur = self.con.cursor()  
 self.cur.execute(  
 "CREATE TABLE IF NOT EXISTS iskop "  
 "(ID INTEGER PRIMARY KEY,"  
 "name TEXT,"  
 "sale\_name TEXT,"  
 "sale REAL,"  
 "mine\_name TEXT)"  
 )  
 self.con.commit()  
 # сохранить изменения  
 self.con.commit()  
  
  
 def \_\_del\_\_(self):  
 # отключение от БД  
 self.con.close()  
  
  
 def view(self):  
 # просмотр всех записей в таблице БД  
 self.cur.execute("SELECT \* FROM iskop")  
 # список всех записей из таблицы  
 rows = self.cur.fetchall()  
 return rows  
  
  
 def insert(self, name, sale\_name, sale, mine\_name):  
 # добавить запись  
 self.cur.execute("INSERT INTO iskop "  
 "VALUES (NULL, ?, ?, ?, ?)", (name, sale\_name, sale, mine\_name,))  
 self.con.commit()  
  
  
 def update(self, id, name, sale\_name, sale, mine\_name):  
 # редактирование записи  
 self.cur.execute("UPDATE iskop SET "  
 "name=?, sale\_name=?, sale=?, mine\_name=?", (name, sale\_name, sale, mine\_name,))  
 self.con.commit()  
  
 def delete(self, id):  
 self.cur.execute("DELETE FROM iskop "  
 "WHERE ID=?", (id,))  
 self.con.commit()  
  
  
 def search(self, name):  
 self.cur.execute("SELECT sale\_name, sale, mine\_name FROM iskop "  
 "WHERE name=?", (name,))  
 rows = self.cur.fetchall()  
 return rows

2. Второй модуль Example\_Add.py

from db\_helper import Student  
# создать объект базы данных  
database\_students = Student()  
# логика  
# добавление записи  
  
  
def add\_command(name, sale\_name, sale, mine\_name):  
 database\_students.insert(name, sale\_name, sale, mine\_name)  
# просмотр всех записей  
  
  
def view\_command():  
 for row in database\_students.view():  
 print(row)  
# основная программа в консоли  
# добавление записи  
for i in range(1):  
 add\_command(input("Введите название: "),  
 input("Введите ед измерения: "),  
 float(input("Введите цену: ")),  
 input("Введите месторождение: "))  
# просмотр всех записей  
view\_command()

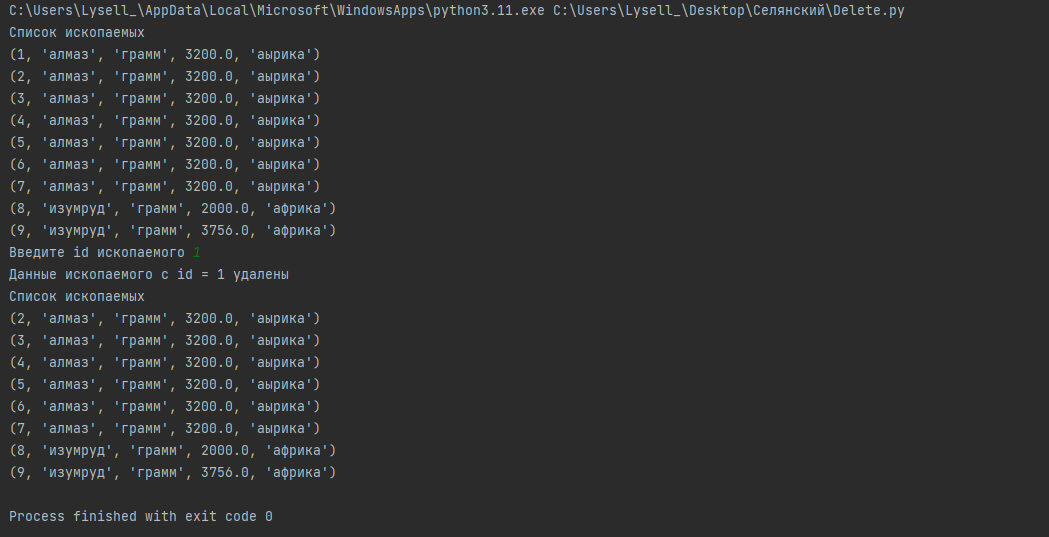
Результат выполнения:



3. Модуль Delete.py

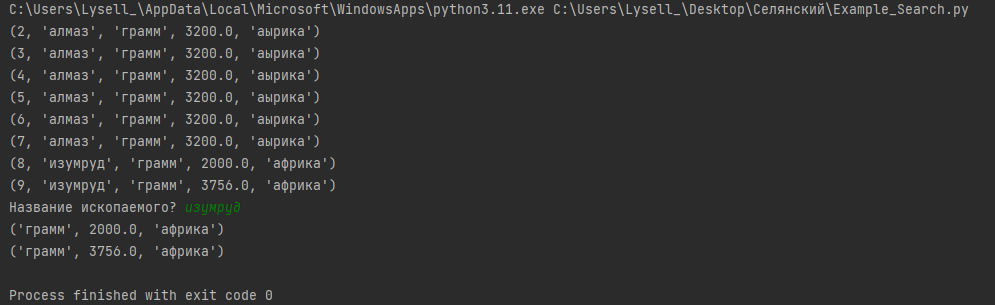
from db\_helper import Student  
# создать объект базы данных  
database\_students = Student()  
# удаление по id ископаемого  
  
  
def delete\_command(id):  
 database\_students.delete(id)  
 print(f"Данные ископаемого с id = {id} удалены")  
# просмотр всех записей  
  
  
def view\_command():  
 for row in database\_students.view():  
 print(row)  
# основная программа  
print("Список ископаемых")  
view\_command()  
id\_delete = int(input("Введите id ископаемого "))  
delete\_command(id\_delete)  
print("Список ископаемых")  
view\_command()

Результат выполнения:



4. Модуль Example\_Search.py

from db\_helper import Student  
# создать объект базы данных  
database\_students = Student()  
# логика  
# добавление записи  
  
  
def add\_command(name, sale\_name, sale, mine\_name):  
 database\_students.insert(name, sale\_name, sale, mine\_name)  
# просмотр всех записей  
  
  
def view\_command():  
 for row in database\_students.view():  
 print(row)  
# поиск по названию  
  
  
def search\_command(name):  
 if len(database\_students.search(name)) > 0:  
 for row in database\_students.search(name):  
 print(row)  
 else:  
 print("Такого ископаемого нет")  
 # основная программа в консоли  
 # добавление записи  
for i in range(0):  
 add\_command(input("Введите название: "),  
 input("Введите ед измерения: "),  
 float(input("Введите цену: ")),  
 input("Введите месторождение: "))  
# просмотр всех записей  
view\_command()  
search\_command(input("Название ископаемого? "))

Результат выполнения:  
  


5. Модуль update.py

from db\_helper import Student  
# создать объект базы данных  
  
database\_students = Student()  
  
  
# удаление по id ископаемого  
def update\_command(id, name, sale\_name, sale, mine\_name):  
 database\_students.update(id, name, sale\_name, sale, mine\_name)  
 print(f"Данные ископаемого с id = {id} обновлены")  
  
  
# просмотр всех записей  
def view\_command():  
 for row in database\_students.view():  
 print(row)  
  
  
  
# основная программа  
print("Список ископаемых")  
view\_command()  
id\_update = int(input("Введите id ископаемого "))  
print("Укажите новые данные: ")  
name = input("Название: ")  
sale\_name = input("Ед измерения: ")  
sale = float(input("Цена: "))  
mine\_name = input("Месторождение: ")  
update\_command(id\_update, name, sale\_name, sale, mine\_name)  
print("Список ископаемых")  
view\_command()

Результат выполнения:

