**Информационные системы управления корпоративным контентом**

**Практическая работа №11**

**ФИО:** Гусев Тимофей Андреевич

**Группа:** ИКБО-08-21

**Дата предоставления в СДО:**

**Задание 1. Написать функцию для преобразования информации из excel таблицы data\_isukk\_9.xslx с данными для формирования накладных в базу данных sqlite.**

На листинге 1 представлен код функции, записывающей данные из excel-таблицы в сдоварь

Листинг 1 – Функция read\_data

def read\_data():data = {}  
 df = pd.read\_excel("data.xlsx")  
 df = df.sort\_values("Грузополучатель")  
 for i, row in df.iterrows():  
 data[row["Грузополучатель"]] = {}  
 data[row["Грузополучатель"]]["Позиции"] = []  
 data[row["Грузополучатель"]]["{{NUMBER}}"] = len(list(data.keys()))  
 for i, row in df.iterrows():  
 data[row["Грузополучатель"]]["{{TO}}"] = row["Грузополучатель"]  
 data[row["Грузополучатель"]]["{{FROM}}"] = row["Грузоотправитель"]  
 data[row["Грузополучатель"]]["{{DATE}}"] = datetime.today().date().strftime("%d.%m.%y")  
 data[row["Грузополучатель"]]["Позиции"].append([len(data[row["Грузополучатель"]]["Позиции"]) + 1, row["Товар"], row["Ед."], row["Кол-во"]])  
 data[row["Грузополучатель"]]["CODE"] = f"AB{data[row['Грузополучатель']]['{{NUMBER}}']}\_{datetime.today().date().strftime('%d.%m.%y')}"  
 for k, v in data.items():  
 res = []  
 for item in v["Позиции"]:  
 res.append(f"{item[0]}) {item[1]} {item[2]} {item[3]}")  
 res = ", ".join(res)  
 v["Позиции"] = res  
 del v["{{DATE}}"]  
 return data

На листинге 2 представлены функции для работы с БД и основная функция.

Листинг 2 – Основная функция и функции БД

def sql\_connection():  
 try:  
 con = sqlite3.connect('mydatabase.db')  
 return con  
 except Error:  
 print(Error)  
  
  
def sql\_table(con):  
 cursorObj = con.cursor()  
 ex = 'CREATE TABLE docs(id integer PRIMARY KEY, from\_com text, to\_com text, position text, document text)'  
 cursorObj.execute(ex)  
 con.commit()  
  
  
def sql\_insert(con, entities):  
 cursorObj = con.cursor()  
 ex = 'INSERT INTO docs(id, from\_com, to\_com, position, document) VALUES(?, ?, ?, ?, ?)'  
 cursorObj.execute(ex, entities)  
 con.commit()  
  
  
def main():  
 con = sql\_connection()  
 sql\_table(con)  
 for k, v in read\_data().items():  
 entities = (v["{{NUMBER}}"], v["{{FROM}}"], v["{{TO}}"], v["Позиции"], v["CODE"])  
 sql\_insert(con, entities)

На рисунке 1 представлена сгенерированная таблица базы данных.

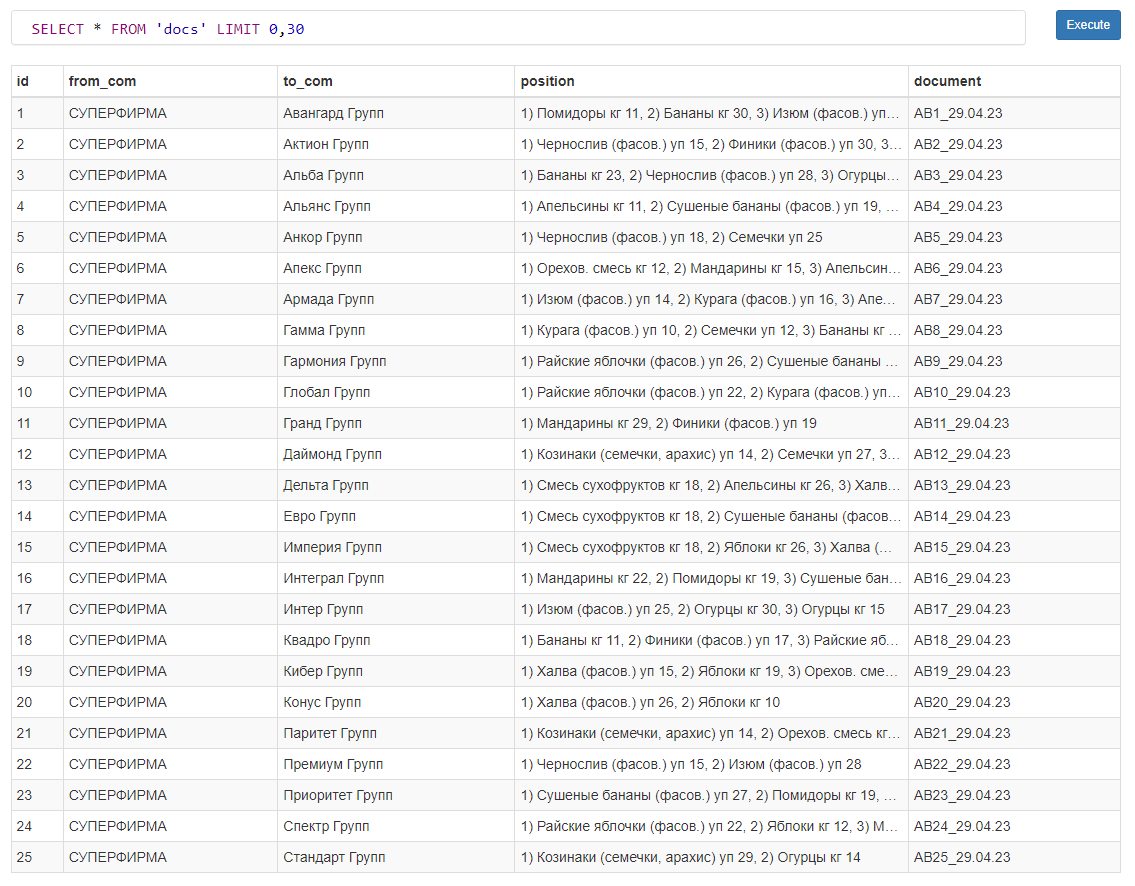


Рисунок 1 – Таблица БД

**Задание 2. Создать таблицу в БД для хранения информации об обработанных накладных:**

На листинге 3 представлен код функции, записывающей данные из excel-таблицы в сдоварь.

Листинг 3 – Функция read\_data

def read\_data():  
 data = {}  
 df = pd.read\_excel("data.xlsx")  
 df = df.sort\_values("Грузополучатель")  
 for i, row in df.iterrows():  
 data[row["Грузополучатель"]] = {}  
 data[row["Грузополучатель"]]["Позиции"] = []  
 data[row["Грузополучатель"]]["{{NUMBER}}"] = len(list(data.keys()))  
 for i, row in df.iterrows():  
 data[row["Грузополучатель"]]["{{TO}}"] = row["Грузополучатель"]  
 data[row["Грузополучатель"]]["{{FROM}}"] = row["Грузоотправитель"]  
 data[row["Грузополучатель"]]["{{DATE}}"] = datetime.today().date().strftime("%d.%m.%y")  
 data[row["Грузополучатель"]]["Позиции"].append([len(data[row["Грузополучатель"]]["Позиции"]) + 1, row["Товар"], row["Ед."], row["Кол-во"]])  
 data[row["Грузополучатель"]]["CODE"] = f"AB{data[row['Грузополучатель']]['{{NUMBER}}']}\_{datetime.today().date().strftime('%d.%m.%y')}"  
 for k, v in data.items():  
 res = []  
 for item in v["Позиции"]:  
 res.append(f"{item[0]}) {item[1]} {item[2]} {item[3]}")  
 res = ", ".join(res)  
 v["Позиции"] = res  
 del v["{{DATE}}"]  
 return data

На листинге 4 представлены функции для работы с БД и основная функция.

Листинг 4 – Основная функция и функции БД

def sql\_connection():  
 try:  
 con = sqlite3.connect('mydatabase1.db')  
 return con  
 except Error:  
 print(Error)  
  
  
def sql\_table(con):  
 cursorObj = con.cursor()  
 ex = 'CREATE TABLE docs(id text PRIMARY KEY, is\_done boolean, date\_done datetime)'  
 cursorObj.execute(ex)  
 con.commit()  
  
  
def sql\_insert(con, entities):  
 cursorObj = con.cursor()  
 ex = 'INSERT INTO docs(id, is\_done, date\_done) VALUES(?, ?, ?)'  
 cursorObj.execute(ex, entities)  
 con.commit()  
  
  
def main():  
 con = sql\_connection()  
 sql\_table(con)  
 for k, v in read\_data().items():  
 entities = (v["CODE"], False, None)  
 sql\_insert(con, entities)

На рисунке 2 представлена сгенерированная таблица базы данных.

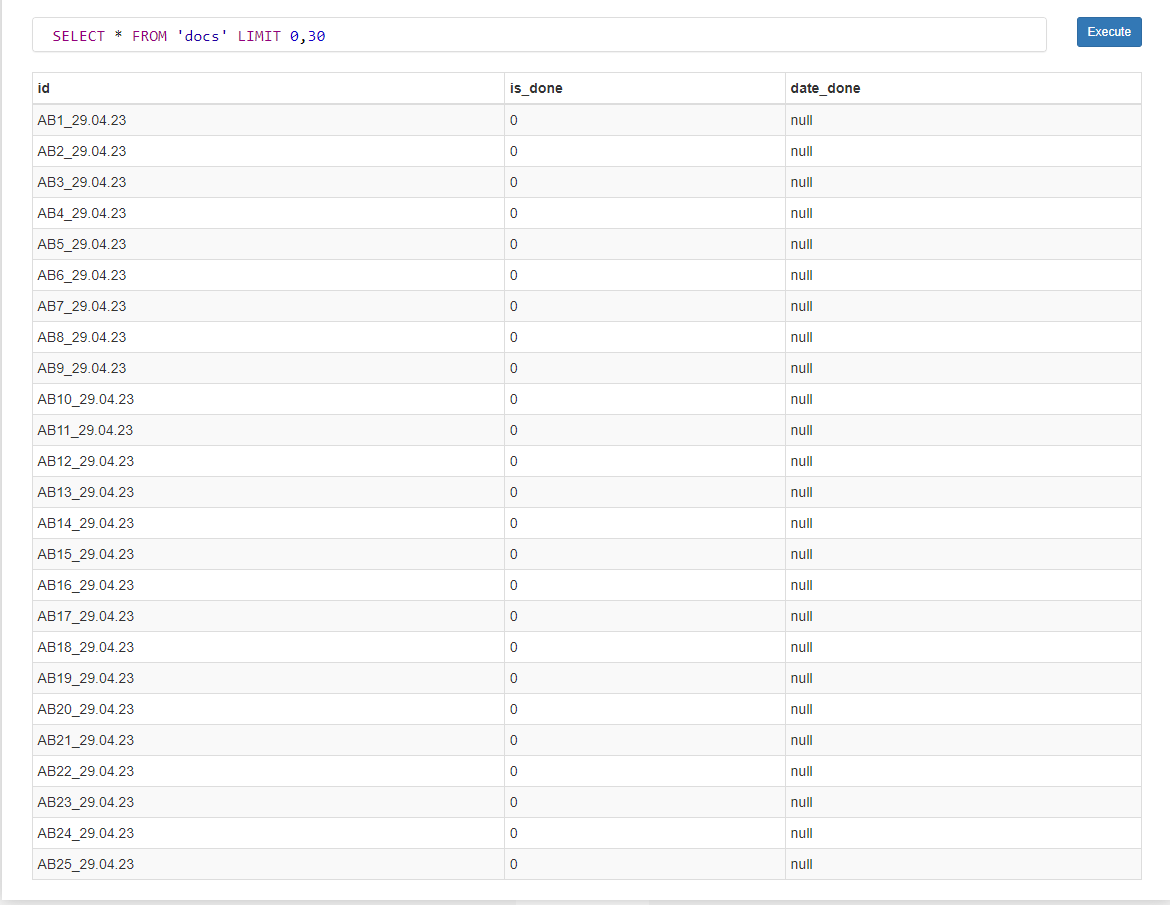


Рисунок 2 – Таблица БД

**Задание 3. Реализовать графический интерфейс и логику работы для следующего сценария работы приложения склада.**

На листинге 5 представлен код класса графического интерфейса, а именно окна приложения.

Листинг 5 – Класс Example

class Example(Frame):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.initUI()  
  
 def initUI(self):  
 self.master.title("Окно")  
 self.pack()  
 cap = cv2.VideoCapture(0)  
 while True:  
 \_, frame = cap.read()  
 decoded\_objects, code = decode(frame)  
 if type(code) == str:  
 break  
 cv2.imshow("frame", frame)  
 if cv2.waitKey(1) == ord("q"):  
 break  
 con = sql\_connection1()  
 if get\_status(con, code):  
 text = f"Накладная {code}\nТовары были выданы {get\_if\_done(con, code)}"  
 mbox.showinfo("Информация", text)  
 else:  
 pos = get\_positions(code)  
 text = f"Накладная {code}\n Выдать следующие товары со склада: {pos}"  
 answer = mbox.askquestion("Вопрос", text)  
 print(answer)  
 if answer == "yes":  
 update(code)  
 else:  
 pass  
 root.destroy()

На листинге 6 представлен код функций для получения и записи данных в БД.

Листинг 6 – Функции работы с БД

def sql\_connection1():  
 try:  
 con = sqlite3.connect('mydatabase1.db')  
 return con  
 except Error:  
 print(Error)  
  
def sql\_connection():  
 try:  
 con = sqlite3.connect('mydatabase.db')  
 return con  
 except Error:  
 print(Error)  
  
def get\_status(con, code):  
 cursorObj = con.cursor()  
 ex = f"SELECT is\_done FROM docs WHERE id='{code}'"  
 cursorObj.execute(ex)  
 res = cursorObj.fetchall()[0][0]  
 con.commit()  
 return res  
  
def get\_if\_done(con, code):  
 cursorObj = con.cursor()  
 ex = f"SELECT date\_done FROM docs WHERE id='{code}'"  
 cursorObj.execute(ex)  
 res = cursorObj.fetchall()[0][0]  
 con.commit()  
 return res  
  
def get\_positions(code):  
 con = sql\_connection()  
 cursorObj = con.cursor()  
 ex = f"SELECT position FROM docs WHERE document='{code}'"  
 cursorObj.execute(ex)  
 res = cursorObj.fetchall()[0][0]  
 con.commit()  
 return res  
  
def update(code):  
 con = sql\_connection1()  
 cursorObj = con.cursor()  
 ex = f'Update docs set is\_done = 1 where id = "{code}"'  
 cursorObj.execute(ex)  
 now = datetime.now().strftime('%d.%m.%Y, %H:%M')  
 ex = f'Update docs set date\_done = "{now}" where id = "{code}"'  
 cursorObj.execute(ex)  
 con.commit()

На листинге 7 представлен код функции декодирования штрих-кода.

Листинг 7 – Декодирование штрих-кода

def draw\_barcode(decoded, image):  
 image = cv2.rectangle(image, (decoded.rect.left, decoded.rect.top), (decoded.rect.left + decoded.rect.width, decoded.rect.top + decoded.rect.height), color=(0, 255, 0), thickness=5)  
 return image  
  
  
def decode(image):  
 code = False  
 decoded\_objects = pyzbar.decode(image)  
 for obj in decoded\_objects:  
 image = draw\_barcode(obj, image)  
 code = (obj.data).decode("utf-8")  
 return [image, code]

На рисунке 3 представлена работа программы.

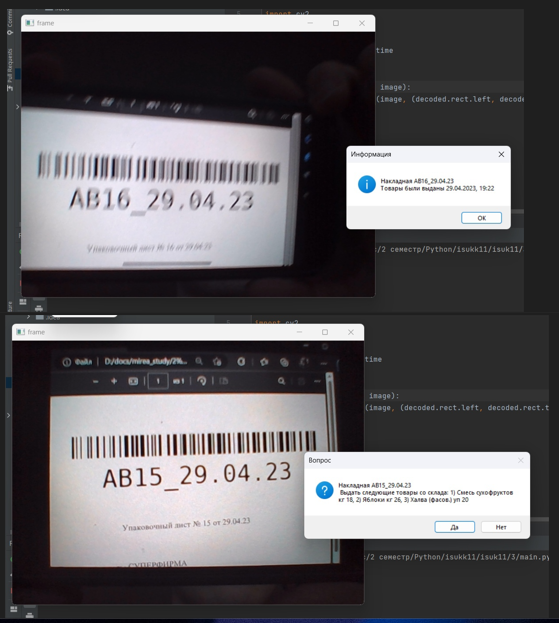


Рисунок 3 – Работа программы