

Санкт-Петербургский Национальный
Исследовательский Университет Информационных
технологий, механики и оптики

Домашняя Работа.

Реализация программного обеспечения для учета
грузового транспорта.

Выполнил: Фоминцев
Денис Русланович
Группа № 3123
Проверила: Казанова
Полина Петровна

Санкт-Петербург
2021

Цель работы:

Создать программное обеспечение учета грузового транспорта для Автотранспортного отдела логистической компании.

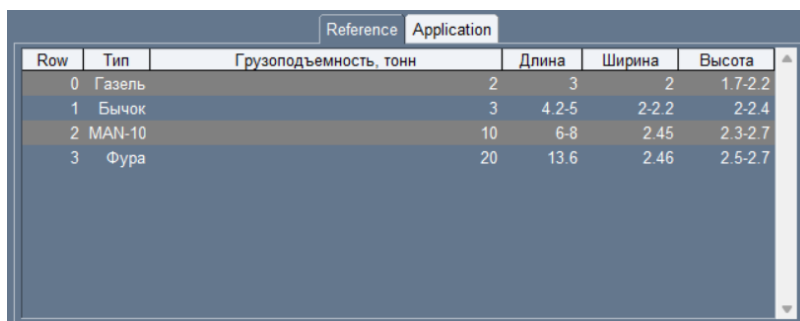
Задача: подобрать доступный грузовой транспорт в зависимости от размера (веса) перевозимого груза, приложение реализовать с помощью ООП, БД, графического интерфейса.

Функции:

- Добавлять/удалять грузовой транспорт;
- Просматривать весь доступный транспорт;
- Просматривать грузовой транспорт по грузоподъемности;
- Просматривать свободный грузовой транспорт;
- Вносить заявку на перевоз груза по указанным габаритам;
- Подобрать и забронировать транспорт;
- Просматривать занятый грузовой транспорт;
- Интерфейс программы реализовать на ваше усмотрение.
- Реализовать возможность сохранения данных в базу данных.

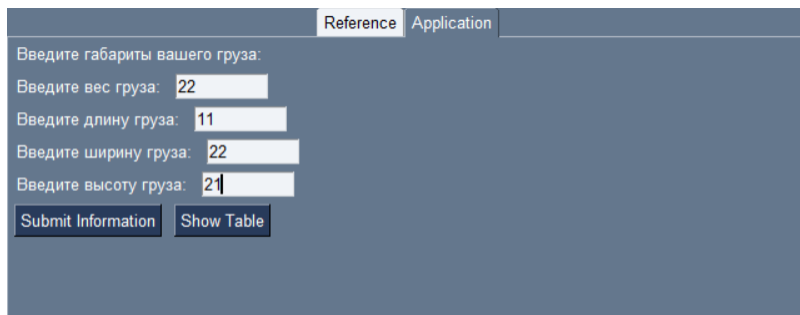
Ход работы:

Для реализации графического интерфейса был использован пакет PySimpleGUI. Пример смотри на рисунке №1 и №2.



Row	Тип	Грузоподъемность, тонн	Длина	Ширина	Высота
0	Газель	2	3	2	1.7-2.2
1	Бычок	3	4.2-5	2-2.2	2-2.4
2	MAN-10	10	6-8	2.45	2.3-2.7
3	Фура	20	13.6	2.46	2.5-2.7

Рисунок 1. Графическое представление справочной информации



Введите габариты вашего груза:

Введите вес груза: 22

Введите длину груза: 11

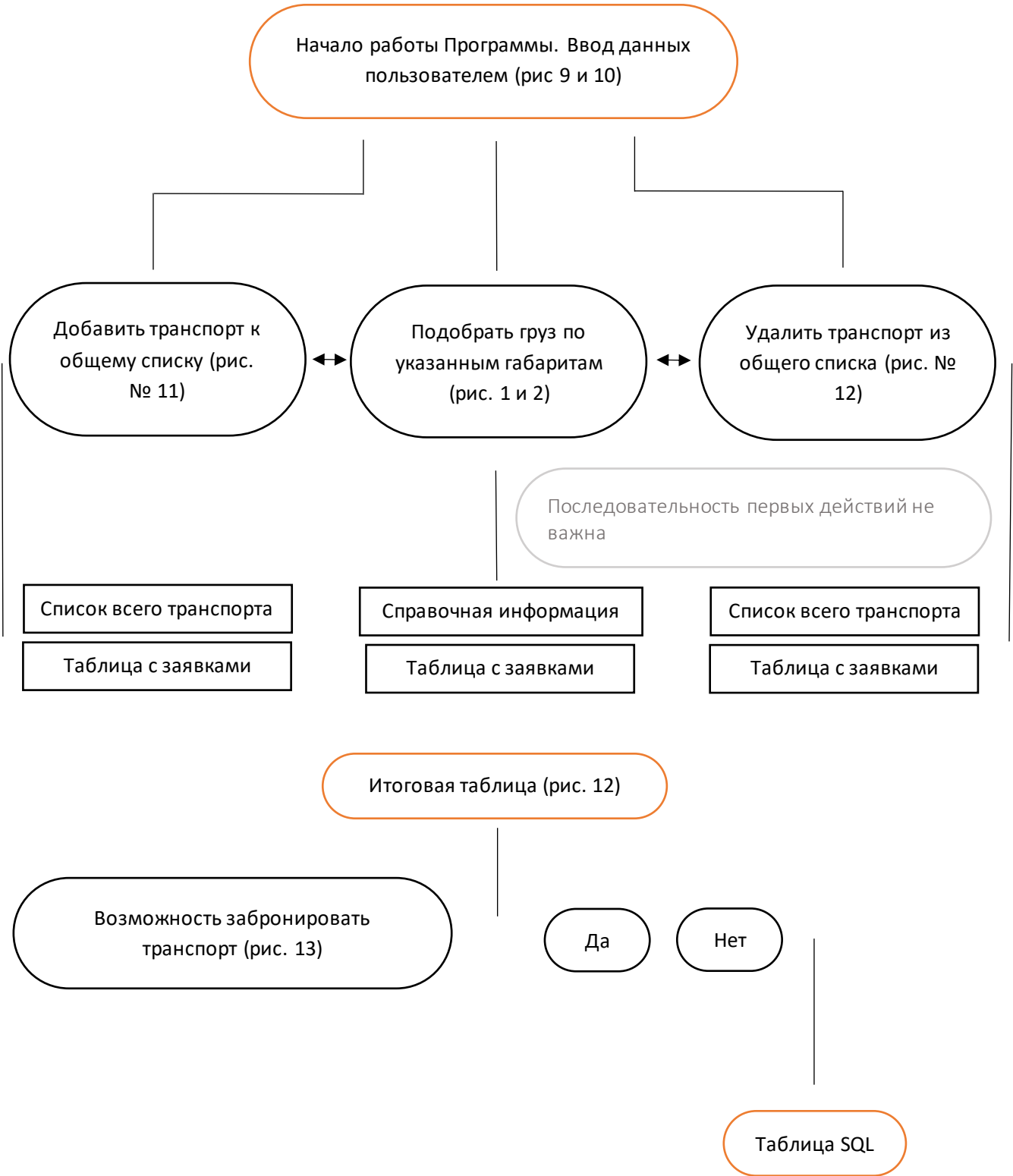
Введите ширину груза: 22

Введите высоту груза: 21

Submit Information Show Table

Рисунок 2. Заявка на перевоз груза

Тезисно по программе:



```
def Main_Table():
    window = sg.Window("Freight transport accounting", tab_group)
    while True:
        event, values = window.read()
        if event in (sg.WIN_CLOSED, 'Exit'):
            break
        elif event == 'Submit Information':
            contact_information = [values['-Вес-'], values['-Длина-'], values['-Ширина-'], values['-Высота-']]
            contact_information_array.append(contact_information)
            sg.popup("Contact Submitted!")
        elif event == 'Show Table':
            table.create(contact_information_array, headings_Application)
```

Рисунок 3. Функция для ввода заявок на перевоз груза

```
def Second():
    ww = sg.Window("Cars", Group)
    while True:
        event, values = ww.read()
        if event in (sg.WIN_CLOSED, 'Exit'):
            break
        elif event == 'Submit Information':
            contact_inf = [values['-Тип-'], values['-Номер данного транспорта-']]
            contact_inf_arr.append(contact_inf)
            sg.popup("Contact Submitted!")
        elif event == 'Show Table':
            table_for_cars.create(contact_inf_arr, head_for_table)
```

Рисунок 4. Функция для добавления нового транспорта (Ее полный код находится в файле second_phase.py)

```
def Third():
    ww = sg.Window("Cars", Group)
    while True:
        event, values = ww.read()
        if event in (sg.WIN_CLOSED, 'Exit'):
            break
        elif event == 'Submit Information':
            contact_inf = [values['-Тип-'], values['-Номер данного транспорта-']]
            contact_inf_arr.append(contact_inf)
            sg.popup("Contact Submitted!")
        elif event == 'Show Table':
            table_for_cars.create(contact_inf_arr, head_for_table)
```

Рисунок 5. Функция для удаления транспорта (Ее полный код находится в файле Delete.py)

После всех введенных данных осуществляется подбор транспорта для указанного груза (если пользователь оставлял заявку). Смотри картинку №6.

```
# -----
for i in range(len(arr)):
    if (int(float(arr[i][0])) <= 2) and (int(float(arr[i][1])) <= 3) and (int(float(arr[i][2])) <= 2) and (0 <= int(float(arr[i][3])) <= 2.2):
        print(' ', end = '')
        print("Для груза №{} нам подойдет Газель".format(i))
        res = 'Газель'
        tmp[i] += (res, )
    elif (2 < int(float(arr[i][0])) <= 3) and (3 <= int(float(arr[i][1])) <= 5) and (0 <= int(float(arr[i][2])) <= 2.2) and (0 <= int(float(arr[i][3])) <= 2.4):
        print(' ', end = '')
        print("Для груза №{} нам подойдет Бокс".format(i))
        res = 'Бокс'
        tmp[i] += (res, )
    elif (3 <= int(float(arr[i][0])) <= 10) and (0 <= int(float(arr[i][1])) <= 8) and (0 <= int(float(arr[i][2])) <= 2.45) and (0 <= int(float(arr[i][3])) <= 2.7):
        print(' ', end = '')
        print("Для груза №{} нам подойдет MAN-10".format(i))
        res = 'MAN-10'
        tmp[i] += (res, )
    elif (10 <= int(float(arr[i][0])) <= 20) and (8 <= int(float(arr[i][1])) <= 13.6) and (0 <= int(float(arr[i][2])) <= 2.46) and (0 <= int(float(arr[i][3])) <= 2.7):
        print(' ', end = '')
        print("Для груза №{} нам подойдет Фура".format(i))
        res = 'Фура'
        tmp[i] += (res, )
    else:
        print(' ', end = '')
        print("Транспорт не определен")
        res = '-'
        tmp[i] += (res, )
```

Рисунок 6. Подбор транспорта

Реализация базы данных и работа с ней представлена на картинках №7 и №8.

```
main_DZ.py Delete.py
try:
    connection = SQLiteDatabase('Freight_transport_accounting.sqlite')
    cursor = connection.cursor()
    print('База данных успешно создана и подключена к Sqlite')
    print()
    # -----
    sql = "DROP TABLE IF EXISTS BaseOfAllCars"
    cursor.execute(sql)
    # -----
    cursor.execute("""CREATE TABLE IF NOT EXISTS BaseOfAllCars(
    Type INT,
    Status_of_book STRING,
    Number INT,
    Weight INT,
    Length INT,
    Width INT,
    Height INT);
    """)
    cursor.executemany("INSERT INTO BaseOfAllCars VALUES(?, ?, ?, ?, ?, ?, ?);", ArrForLast)
    connection.commit()
    # -----
    sql = "DROP TABLE IF EXISTS Base"
    cursor.execute(sql)
    # -----
    cursor.execute("""CREATE TABLE IF NOT EXISTS Base(
    Weight INT,
    Length INT,
    Width INT,
    Height INT,
    Recommend STRING);
    """)
    cursor.executemany("INSERT INTO Base VALUES(?, ?, ?, ?, ?);", tmp)
```

Рисунок 7. Создание базы данных

```
class BaseModel(Model):
    class Meta:
        database = connection

class Base(BaseModel):
    Weight = AutoField(column_name='Weight')
    Length = TextField(column_name='Length', null=True)
    Width = TextField(column_name='Width', null=True)
    Height = TextField(column_name='Height', null=True)
    Recommend_Car = TextField(column_name='Recommend', null=True)

    class Meta:
        table_name = 'Base'

class BaseOfAllCars(BaseModel):
    Type = AutoField(column_name='Type')
    Status_of_book = TextField(column_name='Status_of_book', null=True)
    Number = TextField(column_name='Number', null=True)
    Weight = TextField(column_name='Weight', null=True)
    Length = TextField(column_name='Length', null=True)
    Width = TextField(column_name='Width', null=True)
    Height = TextField(column_name='Height', null=True)

    class Meta:
        table_name_all = 'BaseOfAllCars'
```

Рисунок 8. ORM

Пример работы программы.

```
Freight transport accounting
Если вы хотите посмотреть справочный материал и создать заявку на перевоз груза, нажмите (1);
Если хотите посмотреть добавить авто, нажмите (2);
Если хотите посмотреть удалить авто, нажмите (3)

Введите нужную цифру:
| 1
Хотите ли вы добавить/удалить свой транспорт? (Да/Нет)
| да
Продолжим...

Введите нужную цифру:
| 2
Хотите ли вы удалить транспорт? (Да/Нет)
| да
Подождите окончания работы программы...
1 ...
2 ...
3 ...

Подождите...
1 ...
2 ...
3 ...

Вы хотите забронировать свободный транспорт?
| да
Обработка...
```

Рисунок 9. Ввод данных №1

```
| да
Продолжим...

Введите нужную цифру:
| 2
Хотите ли вы удалить транспорт? (Да/Нет)
| да
Подождите окончания работы программы...
1 ...
2 ...
3 ...

Подождите...
1 ...
2 ...
3 ...

Вы хотите забронировать свободный транспорт?
| да
Обработка...

-----

| Для груза №0 вам подойдет Газель
| Транспорт не определен

База данных успешно создана и подключена к Sqlite
|
```

Рисунок 10. Ввод данных №2

Available transport				Add Car
Row	Тип	Бронь	Номер данного транспорта	
0	MAN-10	Свободен	32264	
1	MAN-10	Свободен	32265	
2	MAN-10	Свободен	32266	
3	MAN-10	Свободен	32267	
4	Газель	Свободен	33890	
5	Фура	Занят	33037	
6	Фура	Занят	33807	
7	MAN-10	Свободен	32066	
8	Газель	Свободен	33800	
9	Газель	Свободен	33890	
10	Газель	Свободен	32090	
11	MAN-10	Свободен	32166	
12	Газель	Свободен	33490	
13	Газель	Свободен	33890	
14	Бычок	Свободен	24249	
15	Фура	Занят	33837	
16	Фура	Занят	32837	
17	Фура	Занят	33817	
18	Бычок	Свободен	24249	
19	MAN-10	Свободен	32066	
				Exit

Рисунок 11. Таблица для добавления транспорта

Available transport				Delete Car
Row	Тип	Бронь	Номер данного транспорта	
0	MAN-10	Свободен	32264	
1	MAN-10	Свободен	32265	
2	MAN-10	Свободен	32266	
3	MAN-10	Свободен	32267	
4	Газель	Свободен	33890	
5	Фура	Занят	33037	
6	Фура	Занят	33807	
7	MAN-10	Свободен	32066	
8	Газель	Свободен	33800	
9	Газель	Свободен	33890	
10	Газель	Свободен	32090	
11	MAN-10	Свободен	32166	
12	Газель	Свободен	33490	
13	Газель	Свободен	33890	
14	Бычок	Свободен	24249	
15	Фура	Занят	33837	
16	Фура	Занят	32837	
17	Фура	Занят	33817	
18	Бычок	Свободен	24249	
19	MAN-10	Свободен	32066	
				Exit

Рисунок 12. Таблица для удаления транспорта

Available transport for book			Transport Busy	Book Car
Row	Тип	Бронь	Номер данного транспорта	
0	new	Свободен	123	
1	MAN-10	Свободен	32264	
2	MAN-10	Свободен	32266	
3	MAN-10	Свободен	32267	
4	Газель	Свободен	33890	
5	MAN-10	Свободен	32066	
6	Газель	Свободен	33800	
7	Газель	Свободен	33890	
8	Газель	Свободен	32090	
9	MAN-10	Свободен	32166	
10	Газель	Свободен	33490	
11	Газель	Свободен	33890	
12	Бычок	Свободен	24249	
13	Бычок	Свободен	24249	
14	MAN-10	Свободен	32066	
Exit				

Рисунок 13. Таблица бронирования транспорта

Row	Тип	Бронь	Номер данного транспорта
0		Заявка Delete	32265
1	new	Заявка Add	123
2	MAN-10	Свободен	32264
3	MAN-10	Свободен	32265
4	MAN-10	Свободен	32266
5	MAN-10	Свободен	32267
6	Газель	Свободен	33890
7	Фура	Занят	33037
8	Фура	Занят	33807
9	MAN-10	Свободен	32066

Рисунок 14. Таблица заявок

Row	Тип	Бронь	Номер данного транспорта
0	new	Забронирован	123
1	MAN-10	Свободен	32264
2	MAN-10	Свободен	32266
3	MAN-10	Свободен	32267
4	Газель	Свободен	33890
5	MAN-10	Свободен	32066
6	Газель	Свободен	33800
7	Газель	Свободен	33890
8	Газель	Свободен	32090
9	MAN-10	Свободен	32166

Рисунок 15. Финальная таблица

```
Обработка базы данных...
Подождите окончания работы программы...
1 ...
2 ...
Представлена информация из базы данных с помощью ORM:

Информация из первой таблицы:
Cargo 0: {'Weight': 2, 'Length': '2', 'Width': '1', 'Height': '2', 'Recommend_Car': 'Газель'}

Информация из второй таблицы:
Car 0: {'Type': 'Фура', 'Status_of_book': 'Занят', 'Number': '33037', 'Weight': '20', 'Length': '13.6', 'Width': '2.46', 'Height': '2.5-2.7'}
Car 1: {'Type': 'Фура', 'Status_of_book': 'Занят', 'Number': '33807', 'Weight': '20', 'Length': '13.6', 'Width': '2.46', 'Height': '2.5-2.7'}
Car 2: {'Type': 'Фура', 'Status_of_book': 'Занят', 'Number': '33837', 'Weight': '20', 'Length': '13.6', 'Width': '2.46', 'Height': '2.5-2.7'}
Car 3: {'Type': 'Фура', 'Status_of_book': 'Занят', 'Number': '32837', 'Weight': '20', 'Length': '13.6', 'Width': '2.46', 'Height': '2.5-2.7'}
Car 4: {'Type': 'Фура', 'Status_of_book': 'Занят', 'Number': '33817', 'Weight': '20', 'Length': '13.6', 'Width': '2.46', 'Height': '2.5-2.7'}
Car 5: {'Type': 'Фура', 'Status_of_book': 'Занят', 'Number': '30837', 'Weight': '20', 'Length': '13.6', 'Width': '2.46', 'Height': '2.5-2.7'}
Car 6: {'Type': 'Газель', 'Status_of_book': 'Забронирован', 'Number': '123', 'Weight': '2', 'Length': '3', 'Width': '2', 'Height': '1.7-2.2'}
Car 7: {'Type': 'Газель', 'Status_of_book': 'Свободен', 'Number': '33890', 'Weight': '2', 'Length': '3', 'Width': '2', 'Height': '1.7-2.2'}
Car 8: {'Type': 'Газель', 'Status_of_book': 'Свободен', 'Number': '33800', 'Weight': '2', 'Length': '3', 'Width': '2', 'Height': '1.7-2.2'}
Car 9: {'Type': 'Газель', 'Status_of_book': 'Свободен', 'Number': '33890', 'Weight': '2', 'Length': '3', 'Width': '2', 'Height': '1.7-2.2'}
Car 10: {'Type': 'Газель', 'Status_of_book': 'Свободен', 'Number': '32090', 'Weight': '2', 'Length': '3', 'Width': '2', 'Height': '1.7-2.2'}
Car 11: {'Type': 'Газель', 'Status_of_book': 'Свободен', 'Number': '33490', 'Weight': '2', 'Length': '3', 'Width': '2', 'Height': '1.7-2.2'}
Car 12: {'Type': 'Газель', 'Status_of_book': 'Свободен', 'Number': '33890', 'Weight': '2', 'Length': '3', 'Width': '2', 'Height': '1.7-2.2'}
Car 13: {'Type': 'Бычок', 'Status_of_book': 'Свободен', 'Number': '24249', 'Weight': '3', 'Length': '4.2-5', 'Width': '2-2.2', 'Height': '2-2.4'}
Car 14: {'Type': 'Бычок', 'Status_of_book': 'Свободен', 'Number': '24249', 'Weight': '3', 'Length': '4.2-5', 'Width': '2-2.2', 'Height': '2-2.4'}
Car 15: {'Type': 'MAN-10', 'Status_of_book': 'Свободен', 'Number': '32264', 'Weight': '10', 'Length': '6-8', 'Width': '2.45', 'Height': '2.3-2.7'}
Car 16: {'Type': 'MAN-10', 'Status_of_book': 'Свободен', 'Number': '32266', 'Weight': '10', 'Length': '6-8', 'Width': '2.45', 'Height': '2.3-2.7'}
Car 17: {'Type': 'MAN-10', 'Status_of_book': 'Свободен', 'Number': '32267', 'Weight': '10', 'Length': '6-8', 'Width': '2.45', 'Height': '2.3-2.7'}
Car 18: {'Type': 'MAN-10', 'Status_of_book': 'Свободен', 'Number': '32066', 'Weight': '10', 'Length': '6-8', 'Width': '2.45', 'Height': '2.3-2.7'}
Car 19: {'Type': 'MAN-10', 'Status_of_book': 'Свободен', 'Number': '32166', 'Weight': '10', 'Length': '6-8', 'Width': '2.45', 'Height': '2.3-2.7'}
```

Рисунок 16. Информация из базы данных

Вывод:

Создали программу для учета грузового транспорта в Автотранспортном отделе логистической компании и подборе груза для перевозок по размеру.

