Содержание

[1. Модель базы данных 3](#_Toc501725239)

[1.1. Логический уровень 3](#_Toc501725240)

[1.2. Физический уровень 4](#_Toc501725241)

[2. Функции для работы БД 5](#_Toc501725242)

[3. SQL запросы 6](#_Toc501725243)

[3.1. Примеры SQL запросов на вывод БД: 6](#_Toc501725244)

[3.2. Примеры SQL запросов на добавление в БД 6](#_Toc501725245)

[3.3. Примеры SQL запросов на редактирования БД 6](#_Toc501725246)

[3.4. Примеры SQL запросов на удаление записи в БД 6](#_Toc501725247)

[3.5. Примеры кросстабличных *SQL* запросов 7](#_Toc501725248)

[4. Интерфейс для управление программой 9](#_Toc501725249)

[4.1. Главное меню 9](#_Toc501725250)

[4.2. Показать таблицы 9](#_Toc501725251)

[4.3. Добавление записи 10](#_Toc501725252)

[4.4. Редактирование записи 11](#_Toc501725253)

[4.5. Удаление записи 12](#_Toc501725254)

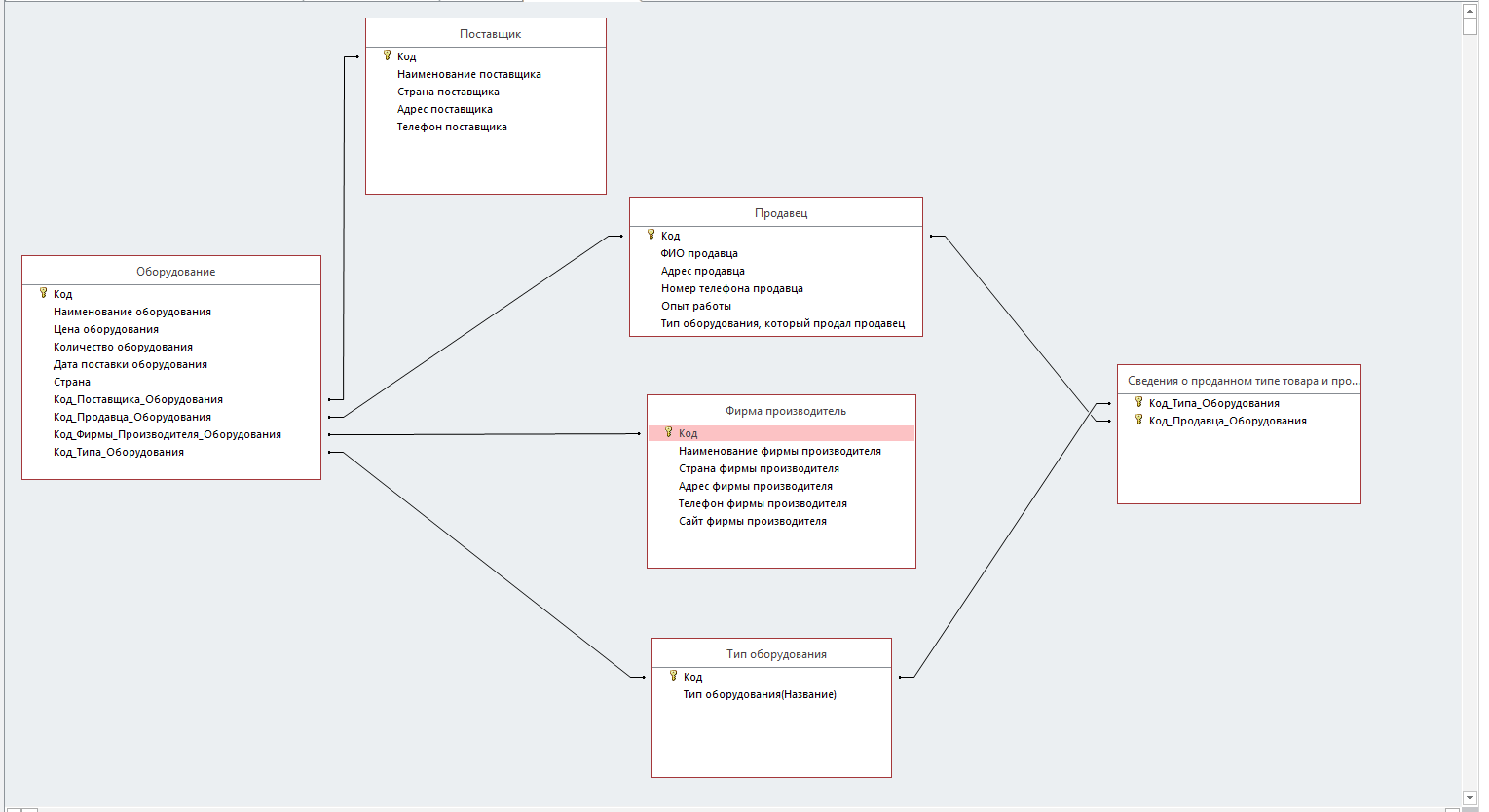
[4.6. Кросстабличные запросы 13](#_Toc501725255)

[5. Листинг программы 14](#_Toc501725256)

[5.1. Index.py 14](#_Toc501725257)

[5.2. Commands.py 14](#_Toc501725258)

1. Модель базы данных
   1. Логический уровень



Предметная область: компьютерные комплектующие.

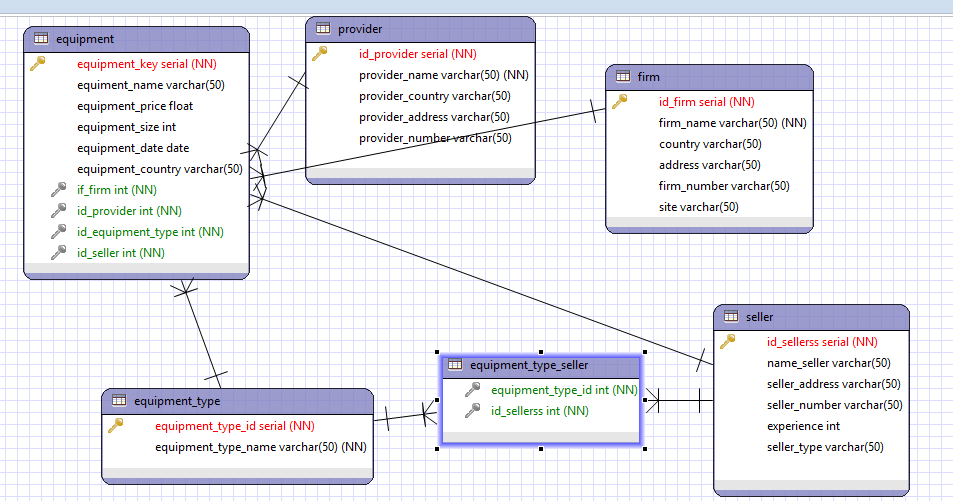
Реализованные сущности:

1. Поставщик, этот объект привозит комплектующие в магазин
2. Продавец, этот объект продает комплектующие в магазине
3. Фирма производитель, этот объект производит комплектующие
4. Тип оборудования, объект, который определяет тип оборудования, пример: Наушники, Видеокарты, Мониторы и. т. д.
5. Оборудование

Свойства сущностей:

1. Поставщик: наименование, страна, адрес, телефон.
2. Фирма производитель: наименование, страна, адрес, телефон, сайт.
3. Тип оборудования: название типа оборудования.
4. Продавец: ФИО продавца, адрес, номер телефона, опыт работы, тип проданного оборудования
5. Оборудование: наименование оборудования, цена, количество, дата поставки в магазин, страна производитель
   1. Физический уровень

БД содержит таблицы:



1. Equipment(Оборудование). Ключевое поле и автоинкремент – equipment\_key(serial).
2. Provider(Поставщик). Ключевое поле и автоинкремент –id\_provider(serial). Суррогатные ключи - id\_provider, id\_firm, id\_equipment\_type, id\_seller
3. Firm(Фирма оборудования). Ключевое поле и автоинкремент – id\_firm(serial)
4. Equipment\_type(Тип оборудования). Ключевое поле и автоинкремент –equipment\_type\_id(serial)
5. Seller(Продавец того или иного оборудования).Ключевое поле и автоинкремент - id\_sellerss(serial)
6. Equipment\_type\_seller(Тип оборудования – продавец. Логическая таблица для связи многие-ко-многим между типом оборудования и продавцом). Суррогатные ключи – equipment\_type\_id, id\_sellerss.

Связи БД:

1. Один-ко-многим. Между таблицами «Поставщик», «Фирма», «Тип оборудования», «Продавец» и «Оборудование»
2. Многие ко многим. Между таблицами «Тип оборудования» и «Поставщик».
3. Функции для работы БД

Функции содержатся в модуле Commands.py. Модуль содержит следующие функции для работы БД:

*# функции для отображения таблица***def** show\_firms(conn): Функция, которая выводит список фирм  
**def** show\_sellers(conn): Функция, которая выводит продавцов  
**def** show\_equipment(conn): Функция, которая выводит оборудование  
**def** show\_equipment\_type(conn): Функция, которая выводит тип оборудования  
**def** show\_provider(conn): Функция, которая выводит поставщиков  
*# Функции для добавления записи в таблицу***def** insert\_firm(conn): Функция, которая добавляет фирму  
**def** insert\_provider(conn): Функция, которая добавляет поставщика  
**def** insert\_seller(conn): Функция, которая добавляет продавца  
**def** insert\_eq\_type(conn): Функция, которая добавляет тип оборудования  
**def** insert\_equipment(conn): Функция, которая добавляет оборудование  
*#Функции для удаления***def** DeleteSellers(conn): Функция, которая удаляет продавца  
**def** DeleteEquipment(conn): Функция, которая удаляет оборудование  
**def** deletefirm(conn): Функция, которая удаляет фирму  
**def** deleteprovider(conn): Функция, которая удаляет поставщика  
**def** delete\_eq\_type(conn): Функция, которая удаляет тип оборудования  
*#Функии для редактирования***def** editprovider(conn): Функция, которая редактирует поставщика  
**def** editseller(conn): Функция, которая редактирует продавца  
**def** editfirm(conn): Функция, которая редактирует фирму  
**def** editequipment(conn): Функция, которая редактирует оборудование  
*#Функции отображения меню***def** delete\_menu(conn): Функция, которая выводит меню удаления  
**def** adding\_menu(conn): Функция, которая выводит меню добавления  
**def** Update\_menu(conn): Функция, которая выводит меню редактирования  
**def** display\_menu(conn): Функция, которая выводит основное меню  
**def** cross\_menu(conn): Функция, которая выводит меню запросов *# Функции кросстабличных запросов*  
**def** sum\_price\_by\_country(conn): *Сумма контрактов по странам фирм***def** sum\_price\_by\_counrty\_staj3(conn): *Сумма контрактов по странам фирм со стажем > 3 лет*  
**def** contract\_provider(conn): Контракты введенного поставщика  
**def** coutry\_year(conn): *Контракты по стране в конкретный год*  
**def** provider\_firm\_price(conn): *#Поиск поставщиков по фирме изготовителя с ценой > 4000*

1. SQL запросы
   1. Примеры SQL запросов на вывод БД:

cursor = conn.cursor()  
**cursor.execute("SELECT \* FROM firm")#Selecting all columns from table firm**row = cursor.fetchone()  
table = PrettyTable((**'id'**, **'firm'**, **'country'**, **'address'**, **'phone\_number'**, **'site'**))  
**while** row **is not None**: *#Fetching row* table.add\_row(row)  
 row = cursor.fetchone()  
cursor.close()

* 1. Примеры SQL запросов на добавление в БД

cursor = conn.cursor()

name = input("Input firm's name: ")

country\_name = input("Input firm's country: ")

address = input("Input firm's address: ")

phone = input("Input phone number: ")

site = input("Input site's address: ")

**sql = "INSERT INTO PUBLIC.\"firm\"(\"firm\_name\", \"country\", \"address\", \**

**\"firm\_number\", \"site\") VALUES (%s,%s,%s,%s,%s)"**

**cursor.execute(sql, (name, country\_name, address, phone, site))**

conn.commit()

* 1. Примеры SQL запросов на редактирования БД

print(show\_provider(conn))

new\_id = input("Input provider ID you want to edit - ")

new\_name = input("Input the name - ")

new\_country = input("Input the country - ")

new\_address = input("Input the adress - ")

new\_number = input("Input number) - ")

cursor = conn.cursor()

**sql = "UPDATE PUBLIC.\"provider\" SET \"provider\_name\" = %s, \"provider\_country\" =%s, provider\_address=%s, provider\_number=%s WHERE \"provider\_id\" = \'{}\'".format(new\_id)**

**cursor.execute(sql, (new\_name, new\_country, new\_address, new\_number))**

conn.commit()

* 1. Примеры SQL запросов на удаление записи в БД

try:

print(show\_sellers(conn))

deleteid = int(input("Input ID of seller you want to DELETE - "))

cursor = conn.cursor()

**sql = "DELETE FROM PUBLIC.\"seller\" WHERE \"id\_sellerss\" = \'{}\' ".format(deleteid)**

**cursor.execute(sql)**

conn.commit()

print("Succefull deleting")

except psycopg2.IntegrityError:

print("You need to delete equipment!")

sys.exit()

* 1. Примеры кросстабличных *SQL* запросов

**#Сумма контрактов по странам фирм**

cursor = conn.cursor()

cursor.execute("SELECT country, COUNT(equipment\_price), SUM(equipment\_price) FROM Public.firm, public.equipment\

WHERE firm\_id = id\_firm\

GROUP BY country\

ORDER BY SUM(equipment\_price) DESC") # Selecting all columns from table seller

row = cursor.fetchone()

table = PrettyTable(('Country', 'Amount', 'Summ'))

while row is not None: # Fetching row

table.add\_row(row)

row = cursor.fetchone()

cursor.close()

return table

**#Сумма контрактов по странам фирм со стажем > 3**

cursor = conn.cursor()

cursor.execute("SELECT provider\_country, COUNT(equipment\_price), SUM(equipment\_price) FROM public.equipment, public.seller, public.provider\

WHERE provider\_id = id\_provider and id\_seller=id\_sellerss and experience >3\

GROUP BY provider\_country\

ORDER BY SUM(equipment\_price) DESC") # Selecting all columns from table seller

row = cursor.fetchone()

table = PrettyTable(('Country', 'Amount', 'Summ'))

while row is not None: # Fetching row

table.add\_row(row)

row = cursor.fetchone()

cursor.close()

return table

**#Контаркты конкретного поставшщка**

cursor = conn.cursor()

name = input("Введите поставщик: ")

cursor.execute("SELECT provider\_name, equiment\_name, equipment\_price,equipment\_date FROM public.equipment, public.provider\

WHERE provider\_id = id\_provider and provider\_name = \'{}\'".format(name)) # Selecting all columns from table seller

row = cursor.fetchone()

table = PrettyTable(('Provide name', 'equipment name', 'Price', 'Date'))

while row is not None: # Fetching row

table.add\_row(row)

row = cursor.fetchone()

cursor.close()

return table

**#Контракты по стране в конкретный год**

cursor = conn.cursor()

name = input("Введите страну : ")

year = input("Введите год - ")

cursor.execute("SELECT provider\_name, equiment\_name, equipment\_price,equipment\_date, provider\_country FROM public.equipment, public.provider, public.seller\

WHERE id\_seller = id\_sellerss and provider\_id = id\_provider and provider\_country = \'{0}\' ".format(name)) # Selecting all columns from table seller

row = cursor.fetchone() # YYYY-MM-DD

table = PrettyTable(('Provide name', "Equipment Name", 'Price', 'Date', 'Coutry'))

while row is not None: # Fetching row

if str(row[3])[:4] == year:

table.add\_row(row)

row = cursor.fetchone()

cursor.close()

return table

**#Поиск поставщиков по фирме изготовителя с ценой > 1000**

cursor = conn.cursor()

name = input("Введите фирму изготовителя : ")

cursor.execute("SELECT provider\_name, firm\_name, equipment\_price FROM public.firm, public.provider, public.equipment\

WHERE firm\_id = id\_firm and equipment\_price > 1000 and id\_provider = provider\_id and firm\_name = \'{}\'".format(name)) # Selecting all columns from table seller

row = cursor.fetchone()

table = PrettyTable(('Provider name', 'Firm name', 'Equipment Price' ))

while row is not None: # Fetching row

table.add\_row(row)

row = cursor.fetchone()

cursor.close()

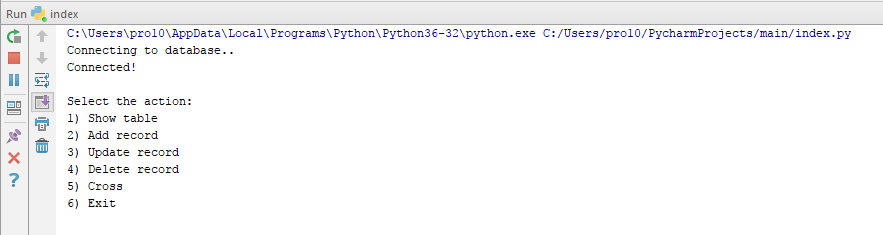
return table

1. Интерфейс для управление программой

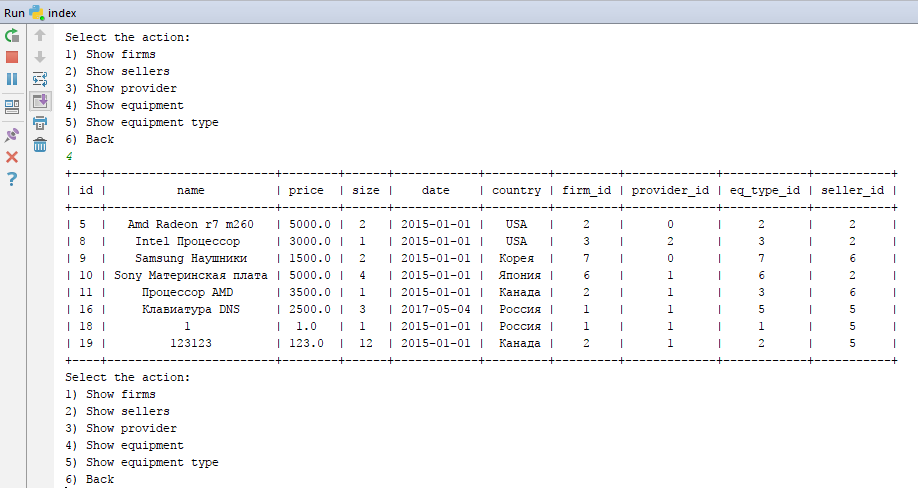
Для выбора пункта необходимо ввести его индекс (1), 2) и т.д.).

* 1. Главное меню

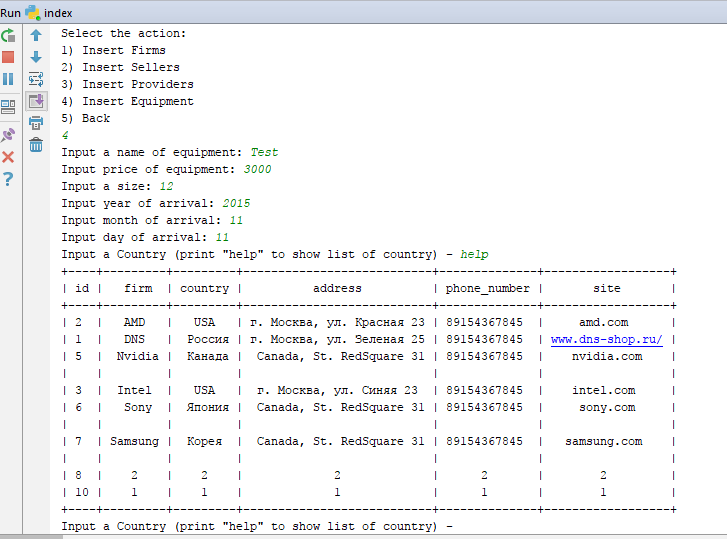
Показать таблицы, добавить запись, редактировать запись, удалить запись, кросстабличные запросы



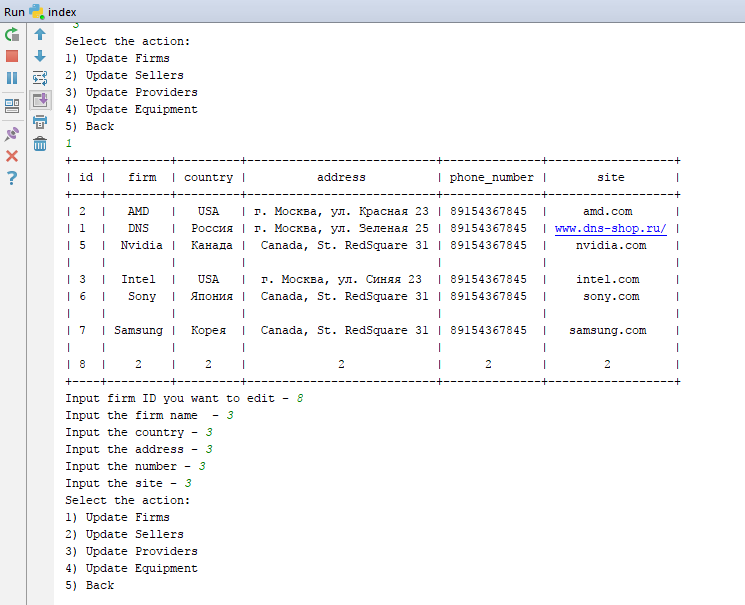
* 1. Показать таблицы

Показать фирмы, показать продавцов, показать поставщиков, показать оборудование, показать тип оборудования.

* 1. Добавление записи

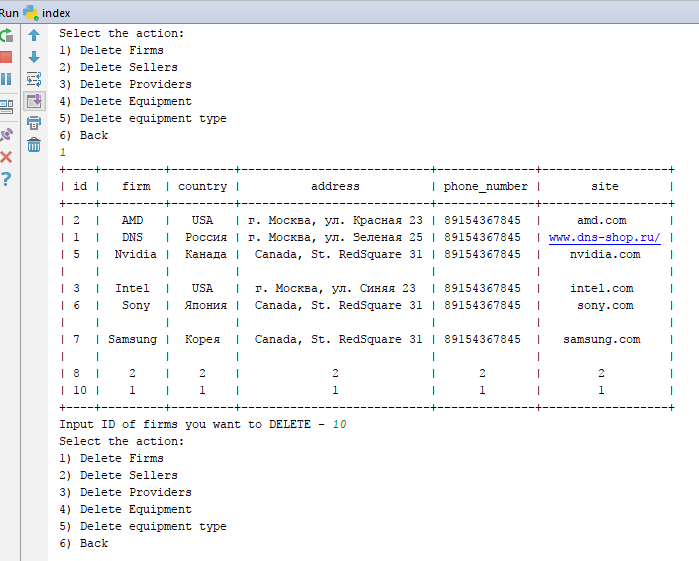
Добавить фирму, добавить продавца, добавить поставщика, добавить оборудование

* 1. Редактирование записи

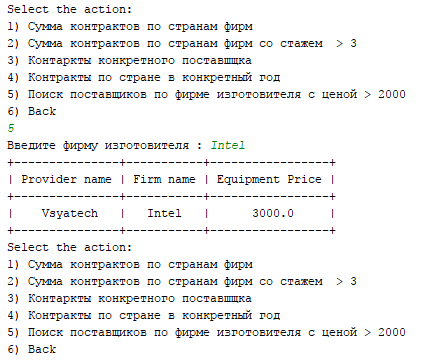
редактировать фирму, редактировать продавца, редактировать поставщика, редактировать оборудование.

* 1. Удаление записи

удалить фирму, удалить поставщика, удалить продаца, удалить оборудование, удалить тип оборудования.



* 1. Кросстабличные запросы



1. Листинг программы
   1. Index.py

**import** psycopg2 *# Import library postgres***import** Commands *# Import file with sql commands***from** prettytable **import** PrettyTable  
**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
  
 flag = **True  
 try**:  
 print(**"Connecting to database.."**)  
 conn = psycopg2.connect(**"host="** + **"localhost"** + **" port="** + **"5432"** + **" dbname="** + **"NDB"** + **" user="** + **"postgres"** + **" password="** + **"10217200"**)  
 print(**"Connected!\n"**)  
 **except** psycopg2.ProgrammingError:  
 print(**"Connection error."**)  
 flag = **False  
 while** flag: *# Printing menu of commands* vod = input(**"Select the action: \n1) Show table \n2) Add record \n3) Update record \n4) Delete record\n5) Cross\n6) Exit\n "**)  
 **if** vod == **'1'**:  
 Commands.display\_menu(conn)  
 **elif** vod == **'2'**:  
 Commands.adding\_menu(conn)  
 **elif** vod == **'3'**:  
 Commands.Update\_menu(conn)  
 **elif** vod == **'4'**:  
 Commands.delete\_menu(conn)  
 **elif** vod == **'5'**:  
 Commands.cross\_menu(conn)  
 **elif** vod == **'6'**:  
 flag = **False  
 else**:  
 print(**'Wrong action!'**)  
 conn.close()

* 1. Commands.py

**import** psycopg2 *# Import library postgres***import** postgresql  
**import** sys  
**from** prettytable **import** PrettyTable  
**import** datetime  
*# Function for shows***def** show\_firms(conn):  
 cursor = conn.cursor()  
 cursor.execute(**"SELECT \* FROM firm"**)*#Selecting all columns from table firm* row = cursor.fetchone()  
 table = PrettyTable((**'id'**, **'firm'**, **'country'**, **'address'**, **'phone\_number'**, **'site'**))  
 **while** row **is not None**: *#Fetching row* table.add\_row(row)  
 row = cursor.fetchone()  
 cursor.close()  
 **return** table  
**def** show\_sellers(conn):  
 cursor = conn.cursor()  
 cursor.execute(**"SELECT \* FROM seller"**) *# Selecting all columns from table seller* row = cursor.fetchone()  
 table = PrettyTable((**'id'**,**'name'**,**'address'**,**'phone\_number'**,**'experience'**,**'seller\_type'**))  
 **while** row **is not None**: *# Fetching row* table.add\_row(row)  
 row = cursor.fetchone()  
 cursor.close()  
 **return** table  
**def** show\_equipment(conn):  
 cursor = conn.cursor()  
 cursor.execute(**"SELECT \* FROM equipment"**) *# Selecting all columns from table seller* row = cursor.fetchone()  
 table = PrettyTable((**'id'**,**'name'**,**'price'**,**'size'**,**'date'**, **'country'**,**'firm\_id'**,**'provider\_id'**,**'eq\_type\_id'**,**'seller\_id'**))  
 **while** row **is not None**: *# Fetching row* table.add\_row(row)  
 row = cursor.fetchone()  
 cursor.close()  
 **return** table  
**def** show\_equipment\_type(conn):  
 cursor = conn.cursor()  
 cursor.execute(**"SELECT \* FROM equipment\_type"**) *# Selecting all columns from table seller* row = cursor.fetchone()  
 table = PrettyTable((**'id'**,**'type'**))  
 **while** row **is not None**: *# Fetching row* table.add\_row(row)  
 row = cursor.fetchone()  
 cursor.close()  
 **return** table  
**def** show\_provider(conn):  
 cursor = conn.cursor()  
 cursor.execute(**"SELECT \* FROM provider"**) *# Selecting all columns from table seller* row = cursor.fetchone()  
 table = PrettyTable((**'id'**,**'name'**,**'country'**,**'address'**,**'phone\_number'**))  
 **while** row **is not None**: *# Fetching row* table.add\_row(row)  
 row = cursor.fetchone()  
 cursor.close()  
 **return** table  
*# ------------------------------------------------------------------------------------  
# Function for adding***def** insert\_firm(conn):  
 cursor = conn.cursor()  
 name = input(**"Input firm's name: "**)  
 country\_name = input(**"Input firm's country: "**)  
 address = input(**"Input firm's address: "**)  
 phone = input(**"Input phone number: "**)  
 site = input(**"Input site's address: "**)  
 sql = **"INSERT INTO PUBLIC.\"firm\"(\"firm\_name\", \"country\", \"address\", \  
 \"firm\_number\", \"site\") VALUES (%s,%s,%s,%s,%s)"** cursor.execute(sql, (name, country\_name, address, phone, site))  
 conn.commit()  
**def** insert\_provider(conn):  
 cursor = conn.cursor()  
 name = input(**"Input provider's name: "**)  
 country\_name = input(**"Input provider's country: "**)  
 address = input(**"Input provider's address: "**)  
 phone = input(**"Input phone number: "**)  
 sql = **"INSERT INTO PUBLIC.\"provider\"(\"provider\_name\", \"provider\_country\", \"provider\_address\", \  
 \"provider\_number\") VALUES (%s,%s,%s,%s)"** cursor.execute(sql, (name, country\_name, address, phone))  
 conn.commit()  
**def** insert\_seller(conn):  
 cursor = conn.cursor()  
 name = input(**"Input seller's name: "**)  
 address = input(**"Input seller's address: "**)  
 phone = input(**"Input phone number: "**)  
 experience = int(input(**"Input seller's experience: "**))  
 sel\_type = input(**"Input seller's type: "**)  
 sql = **"INSERT INTO PUBLIC.\"seller\"(\"name\_seller\", \"seller\_address\", \"seller\_number\", \  
 \"experience\", \"seller\_type\") VALUES (%s,%s,%s,%s,%s)"** cursor.execute(sql, (name, address, phone, experience, sel\_type))  
 conn.commit()  
**def** insert\_eq\_type(conn):  
 cursor = conn.cursor()  
 nametype = str(input(**"Input equipment type's name: "**))  
 sql = **"INSERT INTO PUBLIC.\"equipment\_type\"(\"equipment\_type\_name\") VALUES (%s)"** cursor.execute(sql, (str(nametype)))  
 conn.commit()  
**def** insert\_equipment(conn):  
 name = input(**"Input a name of equipment: "**)  
 price = input(**"Input price of equipment: "**)  
 size = input(**"Input a size: "**)  
 year = int(input(**"Input year of arrival: "**))  
 month = int(input(**"Input month of arrival: "**))  
 day = int(input(**"Input day of arrival: "**))  
 arrival\_date = datetime.date(year, month, day)  
 **while True**:  
 country = input(**"Input a Country (print \"help\" to show list of country) - "**)  
 **if** country == **"help"**:  
 print(show\_firms(conn))  
 **else**:  
 **break  
 while True**:  
 firm = input(**"Input a firm id (print \"help\" to show list of firm) - "**)  
 **if** firm == **"help"**:  
 print(show\_firms(conn))  
 **else**:  
 **break  
 while True**:  
 provider = input(**"Input a provider id (print \"help\" to show list of providers) - "**)  
 **if** provider == **"help"**:  
 print(show\_provider(conn))  
 **else**:  
 **break  
 while True**:  
 type = input(**"Input a equipment type id (print \"help\" to show list of providers) - "**)  
 **if** type == **"help"**:  
 print(show\_equipment\_type(conn))  
 **else**:  
 **break  
 while True**:  
 seller = input(**"Input a seller id (print \"help\" to show list of sellers - "**)  
 **if** seller == **"help"**:  
 print(show\_sellers(conn))  
 **else**:  
 **break** cursor = conn.cursor()  
 sql = **"INSERT INTO PUBLIC.\"equipment\"(\"equiment\_name\", \"equipment\_price\", \"equipment\_size\", \"equipment\_date\",\"equipment\_country\", \"id\_firm\", \"id\_provider\", \"id\_equipment\_type\",\"id\_seller\") VALUES (%s,%s,%s,%s,%s,%s,%s,%s,%s)"** cursor.execute(sql, (name, price, size, arrival\_date, country, firm, provider, type, seller))  
 conn.commit()  
*# --------------------------------------------------------------------------------------  
#function deleting***def** DeleteSellers(conn):  
 **try**:  
 print(show\_sellers(conn))  
 deleteid = int(input(**"Input ID of seller you want to DELETE - "**))  
 cursor = conn.cursor()  
 sql = **"DELETE FROM PUBLIC.\"seller\" WHERE \"id\_sellerss\" = \'{}\' "**.format(deleteid)  
 cursor.execute(sql)  
 conn.commit()  
 print(**"Succefull deleting"**)  
 **except** psycopg2.IntegrityError:  
 print(**"You need to delete equipment!"**)  
 sys.exit()  
**def** DeleteEquipment(conn):  
 cursor = conn.cursor()  
 deleteid = int(input(**"Input ID of Equipment you want to DELETE - "**))  
 cursor = conn.cursor()  
 sql = **"DELETE FROM PUBLIC.\"equipment\" WHERE \"equipment\_key\" = \'{}\' "**.format(deleteid)  
 cursor.execute(sql)  
 conn.commit()  
 print(**"Succefull deleting"**)  
**def** deletefirm(conn):  
 **try**:  
 print(show\_firms(conn))  
 deleteid = int(input(**"Input ID of firms you want to DELETE - "**))  
 cursor = conn.cursor()  
 sql = **"DELETE FROM PUBLIC.\"firm\" WHERE \"firm\_id\" = \'{}\' "**.format(deleteid)  
 cursor.execute(sql)  
 conn.commit()  
 **except** psycopg2.IntegrityError:  
 print(**"You need to delete equipment!"**)  
 sys.exit()  
**def** deleteprovider(conn):  
 **try**:  
 print(show\_provider(conn))  
 deleteid = int(input(**"Input ID of provider you want to DELETE - "**))  
 cursor = conn.cursor()  
 sql = **"DELETE FROM PUBLIC.\"provider\" WHERE \"provider\_id\" = \'{}\' "**.format(deleteid)  
 cursor.execute(sql)  
 conn.commit()  
 **except** psycopg2.IntegrityError:  
 print(**"You need to delete equipment!"**)  
 sys.exit()  
**def** delete\_eq\_type(conn):  
 **try**:  
 print(show\_equipment\_type(conn))  
 deleteid = int(input(**"Input ID of equipment\_type you want to DELETE - "**))  
 cursor = conn.cursor()  
 sql = **"DELETE FROM PUBLIC.\"equipment\_type\" WHERE \"equipment\_type\_id\" = \'{}\' "**.format(deleteid)  
 cursor.execute(sql)  
 conn.commit()  
 **except** psycopg2.IntegrityError:  
 print(**"You need to delete equipment!"**)  
 sys.exit()  
*# --------------------------------------------------------------------------------------  
#function edit***def** editprovider(conn):  
 print(show\_provider(conn))  
 new\_id = input(**"Input provider ID you want to edit - "**)  
 new\_name = input(**"Input the name - "**)  
 new\_country = input(**"Input the country - "**)  
 new\_address = input(**"Input the adress - "**)  
 new\_number = input(**"Input number) - "**)  
 cursor = conn.cursor()  
 sql = **"UPDATE PUBLIC.\"provider\" SET \"provider\_name\" = %s, \"provider\_country\" =%s, provider\_address=%s, provider\_number=%s WHERE \"provider\_id\" = \'{}\'"**.format(new\_id)  
 cursor.execute(sql, (new\_name, new\_country, new\_address, new\_number))  
 conn.commit()  
**def** editseller(conn):  
 print(show\_sellers(conn))  
 new\_id = input(**"Input seller ID you want to edit - "**)  
 new\_name = input(**"Input the name - "**)  
 new\_address = input(**"Input the address - "**)  
 new\_number = input(**"Input the number - "**)  
 new\_exper = input(**"Input the exoerience - "**)  
 new\_type = input(**"Input the type - "**)  
 cursor = conn.cursor()  
 sql = **"UPDATE PUBLIC.\"seller\" SET \"name\_seller\" = %s, \"seller\_address\" =%s, seller\_number=%s, experience=%s, seller\_type=%s WHERE \"id\_sellerss\" = \'{}\'"**.format(new\_id)  
 cursor.execute(sql, (new\_name, new\_address, new\_number, new\_exper,new\_type))  
 conn.commit()  
**def** editfirm(conn):  
 print(show\_firms(conn))  
 new\_id = input(**"Input firm ID you want to edit - "**)  
 new\_name = input(**"Input the firm name - "**)  
 new\_country = input(**"Input the country - "**)  
 new\_address = input(**"Input the address - "**)  
 new\_number = input(**"Input the number - "**)  
 new\_site = input(**"Input the site - "**)  
 cursor = conn.cursor()  
 sql = **"UPDATE PUBLIC.\"firm\" SET \"firm\_name\" = %s, \"country\" =%s, address=%s, firm\_number=%s, site=%s WHERE \"firm\_id\" = \'{}\'"**.format(new\_id)  
 cursor.execute(sql, (new\_name, new\_country, new\_address, new\_number, new\_site))  
 conn.commit()  
**def** editequipment(conn):  
 print(show\_equipment(conn))  
 new\_id = input(**"Input equipment ID you want to edit - "**)  
 new\_name = input(**"Input the name - "**)  
 new\_price = input(**"Input the price - "**)  
 new\_size = input(**"Input the size - "**)  
 year = int(input(**"Input year of arrival: "**))  
 month = int(input(**"Input month of arrival: "**))  
 day = int(input(**"Input day of arrival: "**))  
 arrival\_date = datetime.date(year, month, day)  
  
 **while True**:  
 new\_country = input(**"Input a Country (print \"help\" to show list of country) - "**)  
 **if** new\_country == **"help"**:  
 print(show\_firms(conn))  
 **else**:  
 **break  
 while True**:  
 firm = input(**"Input a firm id (print \"help\" to show list of firm) - "**)  
 **if** firm == **"help"**:  
 print(show\_firms(conn))  
 **else**:  
 **break  
 while True**:  
 provider = input(**"Input a provider id (print \"help\" to show list of providers) - "**)  
 **if** provider == **"help"**:  
 print(show\_provider(conn))  
 **else**:  
 **break  
 while True**:  
 type = input(**"Input a equipment type id (print \"help\" to show list of providers) - "**)  
 **if** type == **"help"**:  
 print(show\_equipment\_type(conn))  
 **else**:  
 **break  
 while True**:  
 seller = input(**"Input a seller id (print \"help\" to show list of sellers - "**)  
 **if** seller == **"help"**:  
 print(show\_sellers(conn))  
 **else**:  
 **break** cursor = conn.cursor()  
 sql = **"UPDATE PUBLIC.\"equipment\" SET \"equiment\_name\" = %s, \"equipment\_price\" =%s, equipment\_size=%s, equipment\_date=%s, equipment\_country=%s, id\_firm=%s, id\_provider =%s, id\_equipment\_type = %s, id\_seller= %s WHERE \"equipment\_key\" = \'{}\'"**.format(new\_id)  
 cursor.execute(sql, (new\_name, new\_price, new\_size, arrival\_date, new\_country, firm, provider, type, seller))  
 conn.commit()  
*# --------------------------------------------------------------------------------------  
#Function for showing menu***def** delete\_menu(conn):  
 flag = **True  
 while** flag: *# Printing menu of commands* vod = input(**"Select the action: \n1) Delete Firms \n2) Delete Sellers \n3) Delete Providers \n4) Delete Equipment \n5) Delete equipment type \n6) Back\n"**)  
 **if** vod == **'1'**:  
 deletefirm(conn)  
 **elif** vod == **'2'**:  
 DeleteSellers(conn)  
 **elif** vod == **'3'**:  
 deleteprovider(conn)  
 **elif** vod == **'4'**:  
 DeleteEquipment(conn)  
 **elif** vod == **'5'**:  
 delete\_eq\_type(conn)  
 **elif** vod == **'6'**:  
 flag = **False  
 else**:  
 print(**'Wrong action!'**)  
**def** adding\_menu(conn):  
 flag = **True  
 while** flag: *# Printing menu of commands* vod = input(  
 **"Select the action: \n1) Insert Firms \n2) Insert Sellers \n3) Insert Providers \n4) Insert Equipment \n5) Back\n"**)  
 **if** vod == **'1'**:  
 insert\_firm(conn)  
 **elif** vod == **'2'**:  
 insert\_seller(conn)  
 **elif** vod == **'3'**:  
 insert\_provider(conn)  
 **elif** vod == **'4'**:  
 insert\_equipment(conn)  
 **elif** vod == **'5'**:  
 flag = **False  
 else**:  
 print(**'Wrong action!'**)  
**def** Update\_menu(conn):  
 flag = **True  
 while** flag: *# Printing menu of commands* vod = input(**"Select the action: \n1) Update Firms \n2) Update Sellers \n3) Update Providers \n4) Update Equipment \n5) Back\n"**)  
 **if** vod == **'1'**:  
 editfirm(conn)  
 **elif** vod == **'2'**:  
 editseller(conn)  
 **elif** vod == **'3'**:  
 editprovider(conn)  
 **elif** vod == **'4'**:  
 editequipment(conn)  
 **elif** vod == **'5'**:  
 flag = **False  
 else**:  
 print(**'Wrong action!'**)  
**def** display\_menu(conn):  
 flag = **True  
 while** flag: *# Printing menu of commands* vod = input(**"Select the action: \n1) Show firms \n2) Show sellers \n3) Show provider \n4) Show equipment \n5) Show equipment type \n6) Back\n"**)  
 **if** vod == **'1'**:  
 print(show\_firms(conn))  
 **elif** vod == **'2'**:  
 print(show\_sellers(conn))  
 **elif** vod == **'3'**:  
 print(show\_provider(conn))  
 **elif** vod == **'4'**:  
 print(show\_equipment(conn))  
 **elif** vod == **'5'**:  
 print(show\_equipment\_type(conn))  
 **elif** vod == **'6'**:  
 flag = **False  
 else**:  
 print(**'Wrong action!'**)  
**def** cross\_menu(conn):  
 flag = **True  
 while** flag: *# Printing menu of commands* vod = input(  
 **"Select the action: \n1) Сумма контрактов по странам фирм \n2) Сумма контрактов по странам фирм со стажем > 3 \n3) Контаркты конкретного поставшщка \n4) Контракты по стране в конкретный год \n5) Поиск поставщиков по фирме изготовителя с ценой > 2000 \n6) Back\n"**)  
 **if** vod == **'1'**:  
 print(sum\_price\_by\_country(conn))  
 **elif** vod == **'2'**:  
 print(sum\_price\_by\_counrty\_staj3(conn))  
 **elif** vod == **'3'**:  
 print(contract\_provider(conn))  
 **elif** vod == **'4'**:  
 print(coutry\_year(conn))  
 **elif** vod == **'5'**:  
 print(provider\_firm\_price(conn))  
 **elif** vod == **'6'**:  
 flag = **False  
 else**:  
 print(**'Wrong action!'**)  
*# --------------------------------------------------------------------------------------  
#function for cross\_zapros***def** display\_menu(conn):  
 flag = **True  
 while** flag: *# Printing menu of commands* vod = input(**"Select the action: \n1) Show firms \n2) Show sellers \n3) Show provider \n4) Show equipment \n5) Show equipment type \n6) Back\n"**)  
 **if** vod == **'1'**:  
 print(show\_firms(conn))  
 **elif** vod == **'2'**:  
 print(show\_sellers(conn))  
 **elif** vod == **'3'**:  
 print(show\_provider(conn))  
 **elif** vod == **'4'**:  
 print(show\_equipment(conn))  
 **elif** vod == **'5'**:  
 print(show\_equipment\_type(conn))  
 **elif** vod == **'6'**:  
 flag = **False  
 else**:  
 print(**'Wrong action!'**)  
**def** sum\_price\_by\_country(conn):  
 *#Сумма контрактов по странам фирм* cursor = conn.cursor()  
 cursor.execute(**"SELECT country, COUNT(equipment\_price), SUM(equipment\_price) FROM Public.firm, public.equipment\  
 WHERE firm\_id = id\_firm\  
 GROUP BY country\  
 ORDER BY SUM(equipment\_price) DESC"**) *# Selecting all columns from table seller* row = cursor.fetchone()  
 table = PrettyTable((**'Country'**, **'Amount'**, **'Summ'**))  
 **while** row **is not None**: *# Fetching row* table.add\_row(row)  
 row = cursor.fetchone()  
 cursor.close()  
 **return** table  
**def** sum\_price\_by\_counrty\_staj3(conn):  
 *#Сумма контрактов по странам фирм со стажем > 3* cursor = conn.cursor()  
 cursor.execute(**"SELECT provider\_country, COUNT(equipment\_price), SUM(equipment\_price) FROM public.equipment, public.seller, public.provider\  
 WHERE provider\_id = id\_provider and id\_seller=id\_sellerss and experience >3\  
 GROUP BY provider\_country\  
 ORDER BY SUM(equipment\_price) DESC"**) *# Selecting all columns from table seller* row = cursor.fetchone()  
 table = PrettyTable((**'Country'**, **'Amount'**, **'Summ'**))  
 **while** row **is not None**: *# Fetching row* table.add\_row(row)  
 row = cursor.fetchone()  
 cursor.close()  
 **return** table  
**def** contract\_provider(conn):  
 *#Контаркты конкретного поставшщка* cursor = conn.cursor()  
 name = input(**"Введите поставщик: "**)  
 cursor.execute(**"SELECT provider\_name, equiment\_name, equipment\_price,equipment\_date FROM public.equipment, public.provider\  
 WHERE provider\_id = id\_provider and provider\_name = \'{}\'"**.format(name)) *# Selecting all columns from table seller* row = cursor.fetchone()  
 table = PrettyTable((**'Provide name'**, **'equipment name'**, **'Price'**, **'Date'**))  
 **while** row **is not None**: *# Fetching row* table.add\_row(row)  
 row = cursor.fetchone()  
 cursor.close()  
 **return** table  
**def** coutry\_year(conn):  
*#Контракты по стране в конкретный год* cursor = conn.cursor()  
 name = input(**"Введите страну : "**)  
 year = input(**"Введите год - "**)  
 cursor.execute(**"SELECT provider\_name, equiment\_name, equipment\_price,equipment\_date, provider\_country FROM public.equipment, public.provider, public.seller\  
 WHERE id\_seller = id\_sellerss and provider\_id = id\_provider and provider\_country = \'{0}\' "**.format(name)) *# Selecting all columns from table seller* row = cursor.fetchone() *# YYYY-MM-DD* table = PrettyTable((**'Provide name'**, **"Equipment Name"**, **'Price'**, **'Date'**, **'Coutry'**))  
 **while** row **is not None**: *# Fetching row* **if** str(row[3])[:4] == year:  
 table.add\_row(row)  
 row = cursor.fetchone()  
 cursor.close()  
 **return** table  
**def** provider\_firm\_price(conn):  
*#Поиск поставщиков по фирме изготовителя с ценой > 4000* cursor = conn.cursor()  
 name = input(**"Введите фирму изготовителя : "**)  
 cursor.execute(**"SELECT provider\_name, firm\_name, equipment\_price FROM public.firm, public.provider, public.equipment\  
 WHERE firm\_id = id\_firm and equipment\_price > 1000 and id\_provider = provider\_id and firm\_name = \'{}\'"**.format(name)) *# Selecting all columns from table seller* row = cursor.fetchone()  
 table = PrettyTable((**'Provider name'**, **'Firm name'**, **'Equipment Price'** ))  
 **while** row **is not None**: *# Fetching row* table.add\_row(row)  
 row = cursor.fetchone()  
 cursor.close()  
 **return** table