Національний технічний університет України
“Київський політехнічний інститут”

Факультет прикладної математики
Кафедра системного програмування і спеціалізованих комп’ютерних систем

Лабораторна робота №2

з дисципліни “ Бази даних і засоби управління”

Тема: “Створення додатку бази даних, орієнтованого на взаємодію з СУБД PostgreSQL”

Виконав:
студент III курсу
групи КВ-01
Таранич Артем
Перевірив: Павловський В. І.

Київ – 2022

Завдання роботи полягає у наступному:

1. Реалізувати функції перегляду, внесення, редагування та вилучення даних у таблицях бази даних, створених у лабораторній роботі №1, засобами консольного інтерфейсу.

2. Передбачити автоматичне пакетне генерування «рандомізованих» даних у базі.

3. Передбачити автоматичне пакетне генерування «рандомізованих» даних у базі. Забезпечити реалізацію пошуку за декількома атрибутами з двох та більше сутностей одночасно: для числових атрибутів – у рамках діапазону, для рядкових – як шаблон функції LIKE оператора SELECT SQL, для логічного типу – значення True/False, для дат – у рамках діапазону дат.

4. Програмний код виконати згідно шаблону MVC (модель-подання-контролер).

Деталізоване завдання:

1. Забезпечити можливість уведення/редагування/вилучення даних у таблицях бази даних з можливістю контролю відповідності типів даних атрибутів таблиць (рядків, чисел, дати/часу). Для контролю пропонується два варіанти: контроль при введенні (валідація даних) та перехоплення помилок (try..except) від сервера PostgreSQL при виконанні відповідної команди SQL. Особливу увагу варто звернути на дані таблиць, що мають зв’язок 1:N. При цьому з боку батьківської таблиці необхідно контролювати вилучення рядків за умови наявності даних у підлеглій таблиці. З точки зору підлеглої таблиці варто контролювати наявність відповідного рядка у батьківській таблиці при виконанні внесення нових даних. Унеможливити виведення програмою системних помилок на екрані шляхом їх перехоплення і адекватної обробки. Внесення даних виконується користувачем у консольному вікні програми.

2. Забезпечити можливість автоматичної генерації великої кількості даних у таблицях за допомогою вбудованих у PostgreSQL функцій роботи з псевдовипадковими числами. Дані мають бути згенерованими не мовою програмування, а відповідним SQL-запитом!

3. Для реалізації пошуку необхідно підготувати 3 запити, що включають дані з декількох таблиць і фільтрують рядки за 3-4 атрибутами цих таблиць. Забезпечити можливість уведення конкретних значень констант для фільтрації з клавіатури користувачем. Крім того, після виведення даних необхідно вивести час виконання запиту у мілісекундах. Перевірити швидкодію роботи запитів на попередньо згенерованих даних.

4. Програмний код організувати згідно шаблону Model-View-Controller(MVC). Приклад організації коду згідно шаблону доступний за даним

посиланням. При цьому модель, подання та контролер мають бути реалізовані у окремих файлах. Для доступу до бази даних використовувати лише мову SQL.

Логічна модель предметної області «Аеропорт»

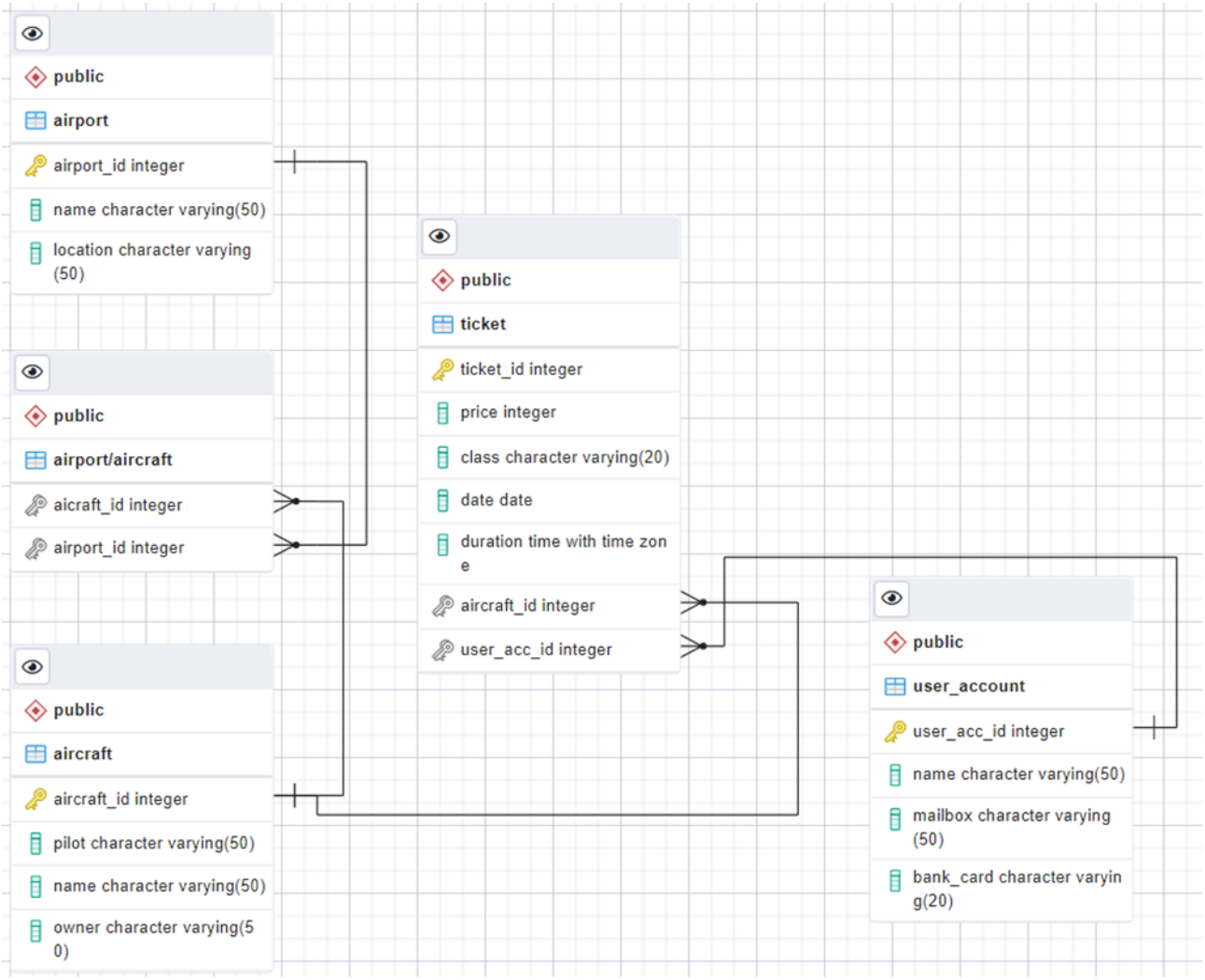

Рисунок 1. Схема бази даних, побудовано у pgAdmin 4.

Середовище та компоненти розробки

Для розробки використовувалась мова програмування Python, середовище розробки Visual Studio Code, а також стороння бібліотека, що надає API для доступу до PostgreSQL – psycopg2.

Шаблон проектування

MVC - Шаблон проектування, який використаний у програмі.

Model – представляє клас, що описує логіку використовуваних даних. Згідно компоненту моделі, у моїй програмі відповідають всі компоненти які знаходять у файлі model.py.

View – в нашому випадку консольний інтерфейс з яким буде взаємодіяти наш користувач. Згідно компоненту представлення, то їй відповідають такі компоненти, згідно яким користувач бачить необхідні дані, що є представленням даних у вигляді консольного інтерфейсу.

Controller – представляє клас, що забезпечує зв'язок між користувачем і системою, поданням і сховищем даних. Він отримує введені користувачем дані і обробляє їх. І в залежності від результатів обробки відправляє користувачеві певний висновок.

Структура програми та її опис

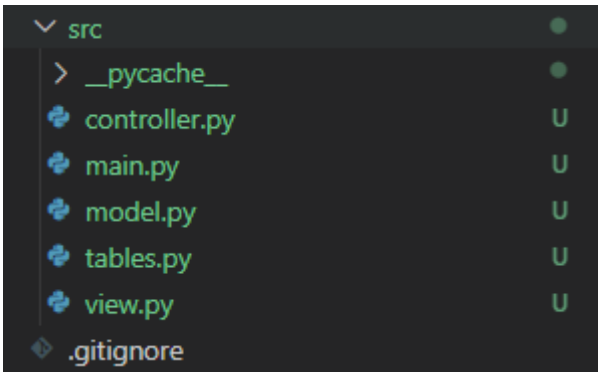

Рисунок 2. Структура програми.

Програма умовно поділена на 4 модулі: main.py, model.py, view.py, controller.py.

Код у файлі controller.py описує клас Controller, який взаємодія з користувачем, передає команди модулю Model та відправляє отримані дані в View.

Код у файлі model.py описає клас Model, що займається регулювання підключення до бази даних, та виконанням низькорівневих запитів до неї.

Код у файлі view.py описує клас View, що виводить результати виконання.

Файл main.py це початкова точка програми, яка створює екземпляр класу Controller і надалі користувач взаємодіє вже з контроллером.

Меню операцій складається з восьми пунктів

1. Виведення усіх таблиць в консоль
2. Виведення вибраної таблиці в консоль.
3. Додавання нового рядку до таблиці.
4. Оновлення рядку у таблиці.
5. Видалення рядку з таблиці.
6. Додавання нових випадкових даних до кожної таблиці БД.
7. Пошук усіх рядків з вибраної таблиці, по параметрам, які ввів користувач.
8. Завершення програми.

Меню таблиць складається з 5 пунктів

1. Виконати обрану операцію над таблицею “Airport”.
2. Виконати обрану операцію над таблицею “Aircraft”.
3. Виконати обрану операцію над таблицею “Airport/Aircraft”.
4. Виконати обрану операцію над таблицею “Ticket”.
5. Виконати обрану операцію над таблицею “Users Account”.

Результати та виконання операцій

Виведення усіх таблиць в консоль

```
        Select an option:
            1. Show all tables
            2. Show table
            3. Add new record
            4. Update record
            5. Delete record
            6. Add new random records
            7. Search
            8. Exit

Enter your choice: 1

============================================================  airport  ==============================================================
                     airport_id                                          name                                          location
 1.                      12                                             venst                                            madrid
 2.                      13                                             venst                                            madrid

============================================================  aircraft  =============================================================
              aircraft_id                     pilot                          name                             owner
 1.                1                      Andrew Russel                 Boeing 737-800                        Bairbus
 2.                2                           Ura                          Airbas                         Claster Master
 3.                3                          David                     Boeing 737-800                        Airbuss
 4.                4                      Andrew Russel                 Boeing 737-800                        Airbuss

========================================================  airport_aircraft  =========================================================
                               aircraft_id                                                    airport_id
 1.                                 3                                                              12

=============================================================  ticket  ==============================================================
        ticket_id           price            class             date               duration        aircraft_id       user_acc_id
 1.         1                644               A         2022-11-19 17:41:00      03:00:00            4                 3

==========================================================  user_account  ===========================================================
              user_acc_id                         name                       mailbox                        bank_card
 1.                2                              admin                  admin@mainbox.com              1234567812345678
 2.                3                               Ura                   ura@gmail.com                  1025458412535412
```

Додавання нового рядку до таблиці “Aiport”

```
        Select an option:
            1. Show all tables
            2. Show table
            3. Add new record
            4. Update record
            5. Delete record
            6. Add new random records
            7. Search
            8. Exit

Enter your choice: 3

        Select a table:
            1. Airport
            2. Aircraft
            3. Airport/Aircraft
            4. Ticket
            5. Users Account
            6. Exit

Enter your choice: 1

============================================================  airport  ==============================================================
                     airport_id                                          name                                          location
 1.                      12                                             venst                                            madrid
 2.                      13                                             venst                                            madrid

Enter data for table:  airport
 ! You can set up default value for field just by typing 'def'
 ! Disabled for Primary&Foreign keys
        Enter name: Kyiv City Airport
        Enter location: Kyiv, Ukraine
INSERT INTO "airport" (name, location) VALUES('Kyiv City Airport', 'Kyiv, Ukraine');

============================================================  airport  ==============================================================
                     airport_id                                          name                                          location
 1.                      12                                             venst                                            madrid
 2.                      13                                             venst                                            madrid
 3.                      14                                       Kyiv City Airport                                 Kyiv, Ukraine
```

Додавання нового рядку до таблиці “Aircraft”

```
        Select an option:
            1. Show all tables
            2. Show table
            3. Add new record
            4. Update record
            5. Delete record
            6. Add new random records
            7. Search
            8. Exit

Enter your choice: 3

        Select a table:
            1. Airport
            2. Aircraft
            3. Airport/Aircraft
            4. Ticket
            5. Users Account
            6. Exit

Enter your choice: 2

============================================================  aircraft  ============================================================
             aircraft_id                    pilot                      name                       owner
 1.               1                    Andrew Russel              Boeing 737-800                Bairbus
 2.               2                        Ura                       Airbas                 Claster Master
 3.               3                       David                    Boeing 737-800               Airbuss
 4.               4                    Andrew Russel              Boeing 737-800               Airbuss

Enter data for table:  aircraft
 ! You can set up default value for field just by typing 'def'
 ! Disabled for Primary&Foreign keys
        Enter pilot: Ira
        Enter name: Boeing
        Enter owner: def
        Enter airport_id: 13
```

В результаті дані також оновилися в зв’язаній таблиці “Airport/Aircraft”

```
============================================================  aircraft  ============================================================
             aircraft_id                    pilot                      name                       owner
 1.               1                    Andrew Russel              Boeing 737-800                Bairbus
 2.               2                        Ura                       Airbas                 Claster Master
 3.               3                       David                    Boeing 737-800               Airbuss
 4.               4                    Andrew Russel              Boeing 737-800               Airbuss
 5.               5                        Ira                       Boeing                    Airbuss

========================================================  airport_aircraft  ========================================================
                            aircraft_id                                                 airport_id
 1.                              3                                                          12
 2.                              5                                                          13
```

Оновлення рядку у таблиці “Ticket”

```
        Select an option:
            1. Show all tables
            2. Show table
            3. Add new record
            4. Update record
            5. Delete record
            6. Add new random records
            7. Search
            8. Exit

Enter your choice: 4

        Select a table:
            1. Airport
            2. Aircraft
            3. Airport/Aircraft
            4. Ticket
            5. Users Account
            6. Exit

Enter your choice: 4

============================================================  ticket  ============================================================
        ticket_id          price            class             date            duration        aircraft_id       user_acc_id
 1.         1               644               A         2022-11-19 17:41:00     03:00:00            4                3

Enter data for table:  ticket
 ! You can set up default value for field just by typing 'def'
 ! Disabled for Primary&Foreign keys
        Enter ticket_id: 1
        Enter price: 4500
        Enter class: B
        Enter date: def
        Enter duration: 05:45:00
        Enter aircraft_id: 4
        Enter user_acc_id: 2
```

```
========================================================== ticket ==========================================================
       ticket_id          price          class          date          duration       aircraft_id       user_acc_id
 1.        1              4500            B        2022-10-10 17:17:40    05:45:00          4              2
```

Видалення рядку з таблиці “Aircraft”

```
        Select an option:
            1. Show all tables
            2. Show table
            3. Add new record
            4. Update record
            5. Delete record
            6. Add new random records
            7. Search
            8. Exit

Enter your choice: 5

        Select a table:
            1. Airport
            2. Aircraft
            3. Airport/Aircraft
            4. Ticket
            5. Users Account
            6. Exit

Enter your choice: 2

========================================================== aircraft ==========================================================
            aircraft_id                    pilot                    name                    owner
 1.             1                     Andrew Russel            Boeing 737-800              Bairbus
 2.             2                         Ura                     Airbas                Claster Master
 3.             3                        David                 Boeing 737-800              Airbuss
 4.             4                     Andrew Russel            Boeing 737-800              Airbuss
 5.             5                         Ira                      Boeing                 Airbuss

Enter data for table:  aircraft
 ! You can set up default value for field just by typing 'def'
 ! Disabled for Primary&Foreign keys
        Enter aircraft_id: 3
```

В результаті дані також оновилися в зв’язаній таблиці “Airport/Aircraft”

```
========================================================== aircraft ==========================================================
            aircraft_id                    pilot                    name                    owner
 1.             1                     Andrew Russel            Boeing 737-800              Bairbus
 2.             2                         Ura                     Airbas                Claster Master
 3.             4                     Andrew Russel            Boeing 737-800              Airbuss
 4.             5                         Ira                      Boeing                 Airbuss

======================================================= airport_aircraft =======================================================
                           aircraft_id                                                    airport_id
 1.                            5                                                            13
```

Додавання нових випадкових даних до кожної таблиці БД.

```
        Select an option:
            1. Show all tables
            2. Show table
            3. Add new record
            4. Update record
            5. Delete record
            6. Add new random records
            7. Search
            8. Exit

Enter your choice: 6
Enter count of records: 5
Done
```

airport

	airport_id	name	location
1.	12	venst	madrid
2.	13	venst	madrid
3.	14	Kyiv City Airport	Kyiv, Ukraine
4.	15	Xj	On
5.	16	Qy	Yp
6.	17	Xw	Qk
7.	18	Vf	Ey
8.	19	Vw	Eg
9.	20	Ur	Ki

aircraft

	aircraft_id	pilot	name	owner
1.	1	Andrew Russel	Boeing 737-800	Bairbus
2.	2	Ura	Airbas	Claster Master
3.	4	Andrew Russel	Boeing 737-800	Airbuss
4.	5	Ira	Boeing	Airbuss
5.	6	Iu	Pw	Qs
6.	7	Dx	Cl	Gq
7.	8	Ky	Jm	Fm
8.	9	Su	Sv	Kw
9.	10	Lk	Ab	Gn
10.	11	Yt	It	Xi

airport_aircraft

	aircraft_id	airport_id
1.	1	15
2.	4	17
3.	5	13
4.	6	14
5.	6	12
6.	6	14
7.	7	16
8.	8	14
9.	9	18
10.	9	17
11.	10	12
12.	11	17

ticket

	ticket_id	price	class	date	duration	aircraft_id	user_acc_id
1.	1	4500	B	2022-10-10 17:17:40	05:45:00	4	2
2.	2	285	C	2022-11-18 00:59:00	05:00:00	5	2
3.	3	668	C	2022-12-06 19:01:00	02:00:00	5	4
4.	4	884	E	2022-12-07 14:50:00	21:00:00	7	2
5.	5	431	E	2022-11-25 21:40:00	22:00:00	2	6
6.	6	230	D	2022-11-14 09:55:00	13:00:00	11	3

user_account

	user_acc_id	name	mailbox	bank_card
1.	2	admin	admin@mainbox.com	1234567812345678
2.	3	Ura	ura@gmail.com	1025458412535412
3.	4	Hm	Db	202
4.	5	Jg	Fr	928
5.	6	Hv	Yd	675
6.	7	Ii	Gb	546
7.	8	Ws	Xg	119

Пошук усіх рядків з вибраної таблиці, по параметрам, які ввів користувач.

```
        Select an option:
            1. Show all tables
            2. Show table
            3. Add new record
            4. Update record
            5. Delete record
            6. Add new random records
            7. Search
            8. Exit

 Enter your choice: 7

        Select a table:
            1. Airport
            2. Aircraft
            3. Airport/Aircraft
            4. Ticket
            5. Users Account
            6. Exit

 Enter your choice: 4
        Enter data to search: C

 ============================================================ ticket ============================================================
        ticket_id          price          class          date          duration        aircraft_id       user_acc_id
 1.         2              285              C        2022-11-18 00:59:00    05:00:00          5                2
 2.         3              668              C        2022-12-06 19:01:00    02:00:00          5                4
```

Код програми

main.py

```python
from controller import Controller

obj = Controller()
obj.menu()
```

model.py

```python
from random import choice

import psycopg2
import psycopg2.extras

from tables import tablesList, table

class Model:
    def __init__(self):
        self.connection = psycopg2.connect(
            host ='localhost',
            user = "postgres",
            password = 'qwerty',
            database = 'airport',
        )
        self.connection.autocommit = True

    def select(self, table: table):
        with self.connection.cursor() as cursor:
            cursor.execute(f"SELECT * FROM \"{table.name}\" " +
                           f"ORDER BY \"{table.id}\" ASC")
```

```
            return cursor.fetchall()

    def insert_into_table(self, table: table, data: list):
        match table.name:
            case 'aircraft':
                airport_id = data.pop()
                self.insert_into(table, data) # insert into aircraft
                aircraft_id = self.select(table)[-1][0]
                self.insert_into(tablesList[2], [aircraft_id, airport_id]) #
insert into airport_aircraft
            case _:
                self.insert_into(table, data)

    def insert_into(self, table: table, data):
        parsed_colms = ''
        for inserted_column in table.changing_columns:
            parsed_colms += f'{inserted_column}' + ', '
        parsed_colms = parsed_colms[:-2]
        inserted_data = ''
        for value in data:
            inserted_data += f'\'{value}\'' + ', '
        inserted_data = inserted_data[:-2]

        with self.connection.cursor() as cursor:
            cursor.execute(f"INSERT INTO \"{table.name}\" ({parsed_colms}) " +
                           f"VALUES({inserted_data});")

    def update_table(self, table: table, data):
        setter = "SET "
        columns = table.changing_columns
        id = data.pop(0)
        if len(columns) != len(data): raise Exception('Wrong data input')
        for i in range(0, len(columns)):
            setter += f'"{columns[i]}" = \'{data[i]}\', '
        setter = setter[:-2]

        with self.connection.cursor() as cursor:
            cursor.execute(f"UPDATE \"{table.name}\" " +
                           f"{setter} " +
                           f"WHERE \"{table.id}\" = {id};")

    def delete_table(self, table: table, data):
        with self.connection.cursor() as cursor:
            cursor.execute(f"DELETE FROM \"{table.name}\" " +
                           f"WHERE {table.id} = {data[0]};")

    def search_in_table(self, table: table, data):
        searcher = "WHERE "
        for column in table.columns:
            searcher += f'\"{column}\"::text LIKE \'%{data}%\' OR '
        searcher = searcher[:-4]

        with self.connection.cursor() as cursor:
            cursor.execute(f"SELECT * FROM \"{table.name}\" " +
                           f"{searcher};")
            return cursor.fetchall()

    def insert_random_data(self, count: int):
```

```
        for i in range(0, count):
            with self.connection.cursor() as cursor:
                inserted_data = [
                    self.generate_random_str(),
                    self.generate_random_str(),
                ]
                self.insert_into_table(tablesList[0], inserted_data) # insert
into airport

                inserted_data = [
                    self.generate_random_str(),
                    self.generate_random_str(),
                    self.generate_random_str(),
                    choice(self.get_ids_from_table(tablesList[0]))[0],
                ]
                self.insert_into_table(tablesList[1], inserted_data) # insert
into aircraft

                inserted_data = [
                    choice(self.get_ids_from_table(tablesList[1]))[0],
                    choice(self.get_ids_from_table(tablesList[0]))[0],
                ]
                self.insert_into_table(tablesList[2], inserted_data) # insert
into airport_aircraft

                inserted_data = [
                    self.generate_random_int(),
                    choice(['A', 'B', 'C', 'D', 'E']),
                    self.generate_random_time(),
                    self.generate_random_duration(),
                    choice(self.get_ids_from_table(tablesList[1]))[0],
                    choice(self.get_ids_from_table(tablesList[4]))[0],
                ]
                self.insert_into_table(tablesList[3], inserted_data) # insert
into tiket

                inserted_data = [
                    self.generate_random_str(),
                    self.generate_random_str(),
                    self.generate_random_int(),
                ]
                self.insert_into_table(tablesList[4], inserted_data) # insert
into user_account

    def generate_random_str(self):
        uppercase_letter = "chr(ascii('A') + (random() * 25)::int) "
        lowercase_letter = "chr(ascii('a') + (random() * 25)::int) "
        with self.connection.cursor() as cursor:
            cursor.execute(f"SELECT ({uppercase_letter} ||
{lowercase_letter});")
            return cursor.fetchone()[0]

    def generate_random_int(self):
        with self.connection.cursor() as cursor:
            cursor.execute(f"SELECT trunc(random() * 1000)::int")
            return cursor.fetchone()[0]

    def generate_random_time(self):
```

```
        with self.connection.cursor() as cursor:
            cursor.execute("""SELECT timestamp '2022-11-10 13:00:00'
                           + make_interval(days  => (random() * 30)::int)
                           + make_interval(hours => (random() * 45)::int)
                           + make_interval(mins  => (random() * 59)::int)
                           """)
            return cursor.fetchone()[0]

    def generate_random_duration(self):
        with self.connection.cursor() as cursor:
            cursor.execute("SELECT time '1:00:00' + make_interval(hours =>
(random() * 24)::int)")
            return cursor.fetchone()[0]

    def get_ids_from_table(self, table: table):
        with self.connection.cursor() as cursor:
            cursor.execute(f"SELECT \"{table.id}\" FROM \"{table.name}\";")
            return cursor.fetchall()
```

view.py

```
from tables import table

class View:
    def __init__(self):
        self.table = None

    def menu(self):
        print('''
        Select an option:
            1. Show all tables
            2. Show table
            3. Add new record
            4. Update record
            5. Delete record
            6. Add new random records
            7. Search
            8. Exit
        ''')

    def list_of_tables(self):
        print('''
        Select a table:
            1. Airport
            2. Aircraft
            3. Airport/Aircraft
            4. Ticket
            5. Users Account
            6. Exit
        ''')

    def show_table(self, table: table, data):
        columns = table.columns
        column_width   = int(130 / len(columns))
        executor_width = 4
        print("\n" + f'  {table.name}  '.center(130 + executor_width, "="),
end='\n    ')
        for column in columns:
```

```
            print(str(column).center(column_width," "), end='')
        print()
        for i, item in enumerate(data, start = 1):
            print(str(f'{i}.').center(executor_width," "), end='')
            for j in item:
                print(str(j).center(column_width, " "), end='')
            print()
```

controller.py

```
from model import Model
from view import View
from datetime import datetime

from tables import tablesList, table

class Controller:
    def __init__(self):
        self.view = View()
        self.model = Model()

    def menu(self):
        while True:
            try:
                self.view.menu()
                choice = int(input('Enter your choice: '))
                match choice:
                    case 1: # show all tablesList
                        for i in range(0, len(tablesList)):
                            data = self.model.select(tablesList[i])
                            self.view.show_table(tablesList[i], data)
                    case 2: # show table
                        table = self.choose_table()
                        data = self.model.select(table)
                        self.view.show_table(table, data)
                    case 3: # add new record
                        table = self.choose_table()
                        self.view.show_table(table, self.model.select(table))
                        self.model.insert_into_table(table,
self.input_data(table, 'insert'))
                        self.view.show_table(table, self.model.select(table))
                    case 4: # update record
                        table = self.choose_table()
                        self.view.show_table(table, self.model.select(table))
                        self.model.update_table(table, self.input_data(table,
'update'))
                        self.view.show_table(table, self.model.select(table))
                    case 5: # delete record
                        table = self.choose_table()
                        self.view.show_table(table, self.model.select(table))
                        self.model.delete_table(table, self.input_data(table,
'delete'))
                        self.view.show_table(table, self.model.select(table))
                    case 6: # add new random records
                        count = int(input('Enter count of records: '))
                        self.model.insert_random_data(count)
                        print('Done')
                    case 7: # search
```

```
                        table  = self.choose_table()
                        result = self.model.search_in_table(table, input('\
tEnter data to search: '))
                        self.view.show_table(table, result)
                    case 8: # exit
                        exit()
                    case _: print('Invalid choice, retry')
            except ValueError as err: print('Exception: ', err)

    def choose_table(self):
        self.view.list_of_tables()
        while True:
            try:
                choice = int(input('Enter your choice: '))
                match choice:
                    case 1: # Airport
                        return (tablesList[0])
                    case 2: # Aircraft
                        return (tablesList[1])
                    case 3: # airport_aircraft
                        return (tablesList[2])
                    case 4: # Ticket
                        return (tablesList[3])
                    case 5: # User Account
                        return (tablesList[4])
                    case 6: # exit
                        self.menu()
                        exit()
                    case _: print('Invalid choice, retry')
            except ValueError as err: print('Exception: ', err)

    def input_data(self, table: table, instruction):
        data = []
        print('\nEnter data for table: ', table.name)
        print(" ! You can set up default value for field just by typing 'def'")
        print(" ! Disabled for Primary&Foreign keys")
        columns = []
        index = 0
        match instruction:
            case 'insert':
                columns = table.get_all_columns()
            case 'update':
                index = 1
                columns = table.columns
            case 'delete':
                index = 1
                columns = [table.columns[0]]

        if instruction == 'update' and table.name == 'airport_aircraft':
            data.append(input('\tEnter current aircraft_id: '))
        for column in columns:
            data.append(input('\tEnter ' + column + ': '))

        if instruction == 'delete': return data
        match table.name:
            case 'airport': # Airport
                if data[index] == 'def': data[index] = 'London City Airport'
                index += 1
```

```
                if data[index] == 'def': data[index] = 'Newham, London, UK'
            case 'aircraft': # Aircraft
                if data[index] == 'def': data[index] = 'Andrew Russel'
                index += 1
                if data[index] == 'def': data[index] = 'Boeing 737-800'
                index += 1
                if data[index] == 'def': data[index] = 'Airbuss'
                index += 1
                if instruction == 'insert' and data[index] == 'def':
                    print('Please retry input data, for this coloumn default
option is disabled')
                    return self.input_data(table)
            case 'airport_aircraft': # airport_aircraft
                if data[0] == 'def' or data[1] == 'def':
                    print('Please retry input data, for this table default
option is disabled')
                    return self.input_data(table)
            case 'ticket': # Ticket
                if data[index] == 'def': data[index] = '1000'
                index += 1
                if data[index] == 'def': data[index] = 'E'
                index += 1
                if data[index] == 'def': data[index] = '2022-10-10 ' +
datetime.now().strftime("%H:%M:%S")
                index += 1
                if data[index] == 'def': data[index] = '05:00:00'
                index += 1
                if data[index] == 'def' or data[index+1] == 'def':
                    print('Please retry input data, for foreign keys default
option is disabled')
                    return self.input_data(table)
            case 'user_account': # User Account
                if data[index] == 'def': data[index] = 'admin'
                index += 1
                if data[index] == 'def': data[index] = 'admin@mainbox.com'
                index += 1
                if data[index] == 'def': data[index] = '0000000000000000'
            case _:
                print('Invalid choice, retry')
                return self.input_data(table)
        return data

    def input_data_to_search(self):
        return input('\tEnter data to search: ')
```

tables.py

```
airport_table = [
    'airport_id',
    'name',
    'location'
]

aircraft_table = [
    'aircraft_id',
    'pilot',
    'name',
    'owner',
```

```
]

airport_aircraft_table = [
    'aircraft_id',
    'airport_id',
]

ticket_table = [
    'ticket_id',
    'price',
    'class',
    'date',
    'duration',
    'aircraft_id',
    'user_acc_id',
]

user_account_table = [
    'user_acc_id',
    'name',
    'mailbox',
    'bank_card',
]


class table:
    def __init__(self, name: str, columns: list, associated_keys: list = None,):
        self.name    = name
        self.columns = columns.copy()
        self.id      = columns[0]
        if name != 'airport_aircraft':
            columns.pop(0)
        self.changing_columns = columns
        self.associated_keys = None
        if associated_keys is not None:
            self.associated_keys = associated_keys.copy()

    def get_table_id(self):
        return self.columns[0]

    def get_all_columns(self):
        columns = self.changing_columns.copy()
        if self.associated_keys is not None:
            columns += self.associated_keys
        return columns

tablesList = [
    table('airport', airport_table),
    table('aircraft', aircraft_table, [airport_aircraft_table[1]]),
    table('airport_aircraft', airport_aircraft_table),
    table('ticket', ticket_table),
    table('user_account', user_account_table),
]
```