НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра системного програмування та спеціалізованих комп'ютерних систем

Лабораторна робота №2

з дисципліни **Бази даних і засоби управління**

на тему: "Створення додатку бази даних, орієнтованого на взаємодію з СУБД PostgreSQL"

Виконав:
студент III курсу
групи КВ-94
Мартинюк А. О.
Перевірив:
Петрашенко А. В.

Київ – 2021

Постановка задачі

Метою роботи є здобуття вмінь програмування прикладних додатків баз даних PostgreSQL.

Загальне 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. 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).

Інформація про програму

Посилання на репозиторій у GitHub з вихідним кодом програми та звітом: https://github.com/amartinuk30/databases

Використана мова програмування: Python 3.9

Використані бібліотеки: psycopg2, time, prettytable

Відомості про обрану предметну галузь з лабораторної роботи №1

Обрана галузь передбачає облік пропозицій щодо трансферу гравців футбольного(або іншого) клубу.

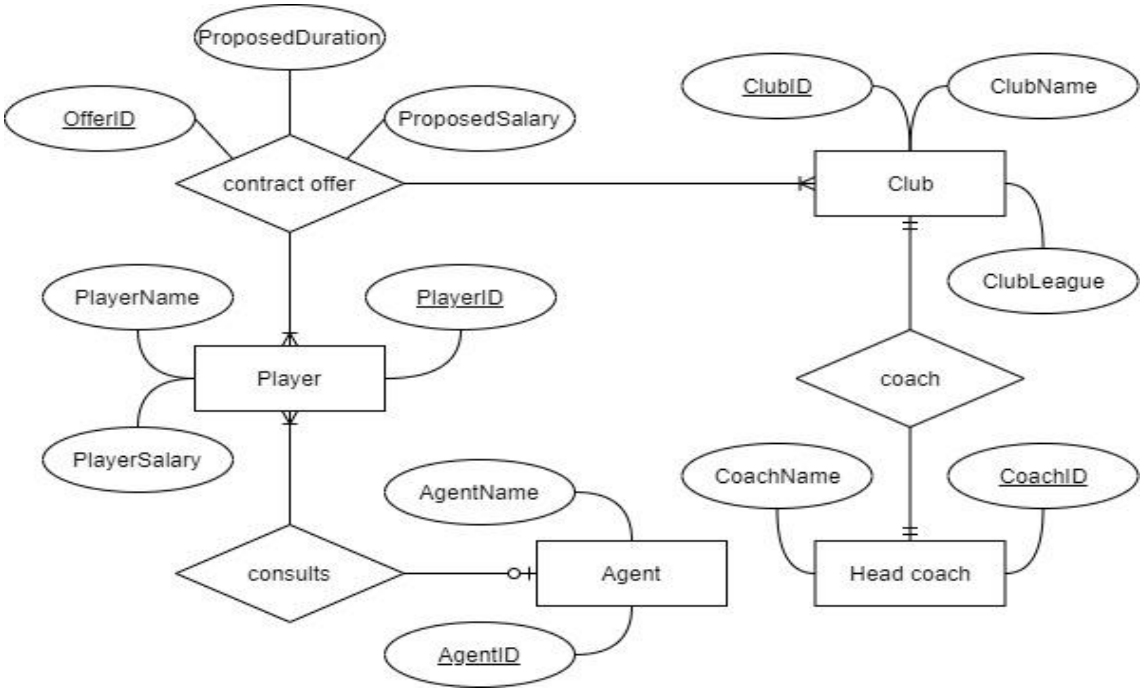


Рис.1 - ER-діаграма, побудована за нотацією Чена

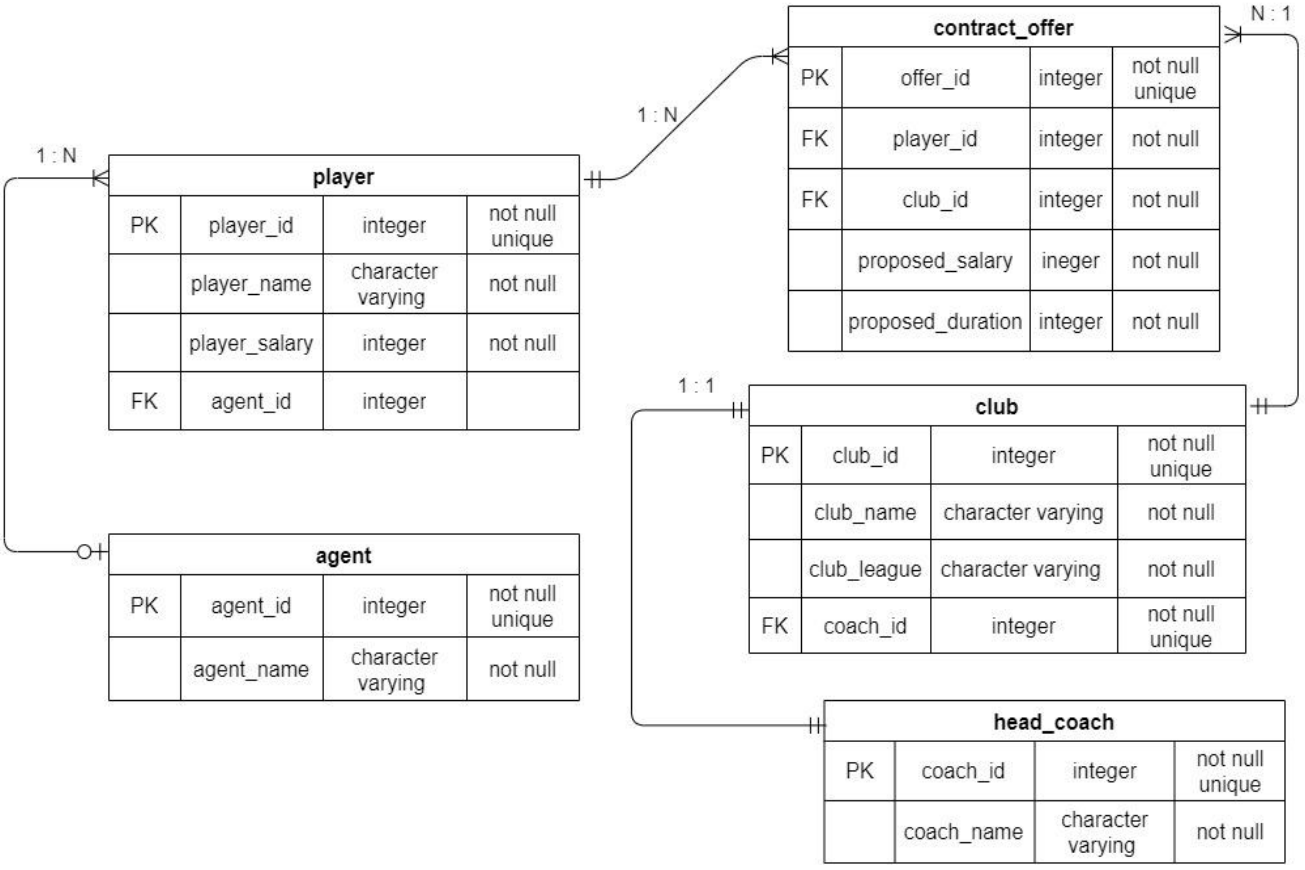


Рис. 2 - Схема бази даних

Таблиця 1. Опис структури БД

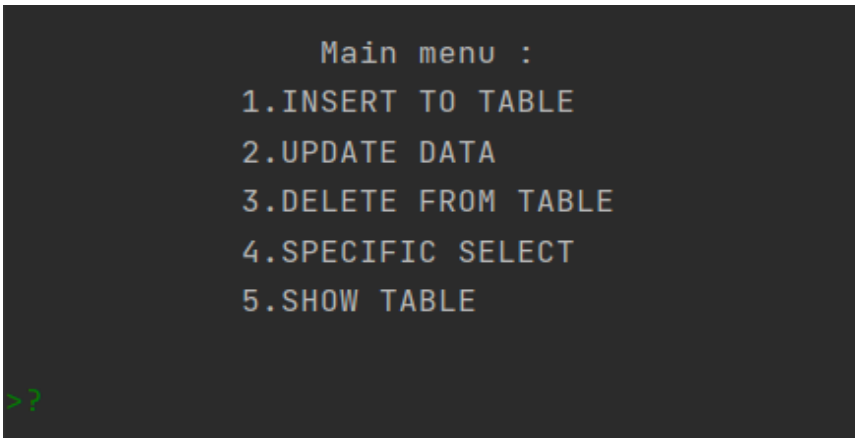
Сутність/Зв'язок	Атрибути	Тип атрибутів
Agent - містить інформацію про агента, який співпрацює із гравцем	**agent_id** - ідентифікатор агента; **agent_name** - ім'я та прізвище агента.	integer (числовий) char varying (рядок)
Player - містить інформацію про гравця, по якому була зроблена пропозиція	**player_id -** ідентифікатор гравця **player_name** - ім'я та прізвище гравця **player_salary -** актуальна зарплата гравця **agent_id** - особистий ідентифікатор агента, який співпрацює з гравцем (FK)	integer (числовий) char varying (рядок) integer (числовий) integer (числовий)
Contract offer - містить інформацію про деталі пропонованого контракту	**offer_id** - ідентифікатор пропозиції **player_id** - унікальний ідентифікатор гравця, якому зроблена ця пропозиція (FK); **club_id** - унікальний ідентифікатор клубу, який робить трансферну пропозицію (FK); **proposed_salary** - запропонована зарплата; **proposed_duration** - запропонований термін дії контракту.	integer (числовий) integer (числовий) integer (числовий) integer (числовий) integer (числовий)
Club - містить інформацію про клуб, який робить трансферну пропозицію	**club_id** - ідентифікатор клубу; **club_name** - назва команди; **club_league** - ліга, у якій виступає команда; **coach_id** - ідентифікатор головного тренера, який тренує клуб (FK).	integer (числовий) char varying (рядок) char varying (рядок) integer (числовий)
Head Coach - містить інформацію про головного тренера клубу, який робить пропозицію;	**coach_id** - ідентифікатор головного тренера; **coach_name** - ім'я та прізвище головного тренера.	integer (числовий) char varying (рядок)

В обраній предметній галузі маємо обробку трансферних пропозицій діючим гравцям футбольної команди в конкретне трансферне вікно (період, коли можливе заключення таких угод). Оскільки дані стосуються конкретної події та конкретного часового періоду, то актуальність даних забезпечується на той момент, у який була зроблена трансферна пропозиція.

Для побудови бази даних було використано наступні сутності :

1. **Player** з атрибутами PlayerID, PlayerName, PlayerSalary (актуальна зарплата гравця). Ця сутність відповідає за ідентифікацію гравця, по якому була отримана пропозиція трансферу.
2. **Club** з атрибутами ClubID, ClubName, ClubLeague. Ця сутність відповідає за ідентифікацію клубу, який зробив пропозицію щодо трансферу.
3. **Head coach** з атрибутами CoachID, CoachName. Дана сутність визначає головного тренера, який тренує клуб, що пропонує трансфер.
4. **Agent** з атрибутами AgentID, AgentName. Дана сутність визначає агента, який співпрацює із гравцем/гравцями.

Меню програми

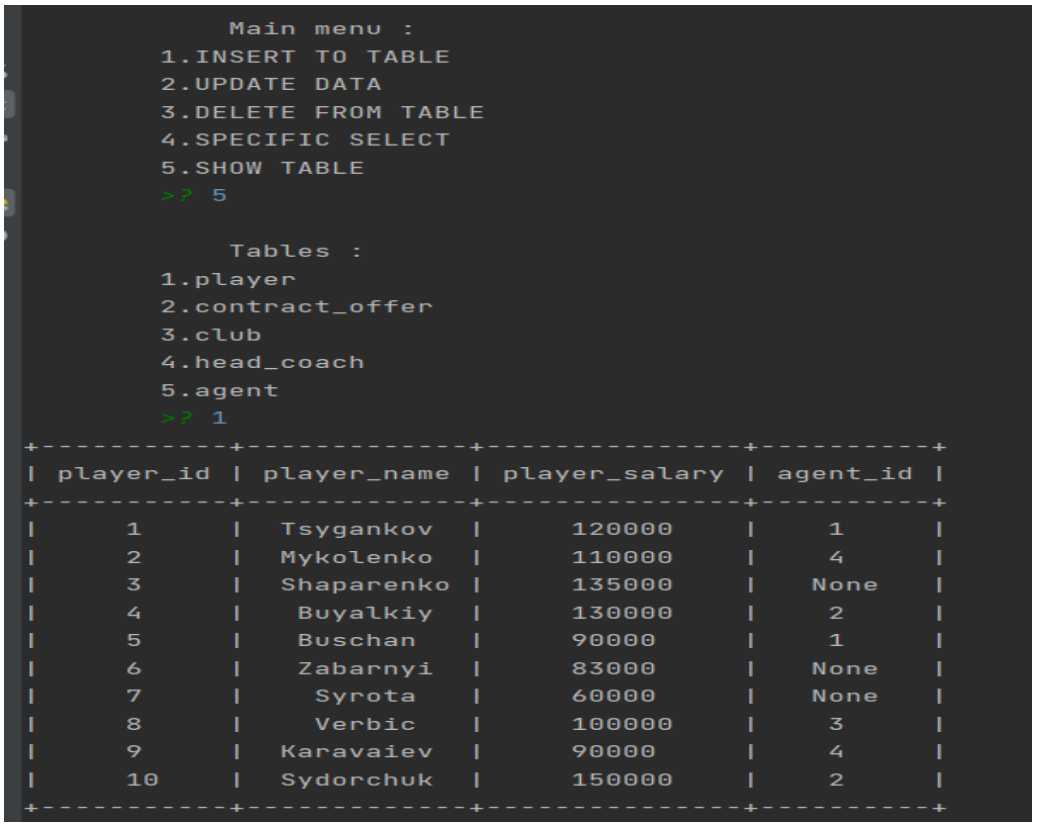


Головне меню програми містить 5 основних пунктів (функцій програми). Вибір конкретного пункту відбувається введенням його номеру із клавіатури. Після введення номеру користувачем, він передається у функцію manage_choice() класу Controller, де активується відповідна для цього пункту гілка оператора if-elif-else із відповідним набором функцій.

Завдання 1

Вставка даних у таблицю (INSERT)

Player (початковий стан таблиці) :



player_id	player_name	player_salary	agent_id
1	Tsygankov	120000	1
2	Mykolenko	110000	4
3	Shaparenko	135000	None
4	Buyalkiy	130000	2
5	Buschan	90000	1
6	Zabarnyi	83000	None
7	Syrota	60000	None
8	Verbic	100000	3
9	Karavaiev	90000	4
10	Sydorchuk	150000	2

```
query_on_insert = """ INSERT INTO player (player_id, player_name, player_salary, agent_id)
VALUES (%s, %s, %s, %s)"""
```

```
        1 --> manual insertion   or  2 --> random insertion : >? 1
Enter values following this sequence: player_id, player_name, player_salary, agent_id
player_id: >? 11
player_name: >? Besedin
player_salary: >? 90000
agent_id: >? 2
1 record successfully inserted

        1 --> menu   or    2 --> exit :
```

Player(таблиця після вставки даних) :

player_id	player_name	player_salary	agent_id
1	Tsygankov	120000	1
2	Mykolenko	110000	4
3	Shaparenko	135000	None
4	Buyalkiy	130000	2
5	Buschan	90000	1
6	Zabarnyi	83000	None
7	Syrota	60000	None
8	Verbic	100000	3
9	Karavaiev	90000	4
10	Sydorchuk	150000	2
11	Besedin	90000	2

Помилка при записі у дочірню таблицю(player) неіснуючого значення батьківської таблиці(agent) :

```
        1 --> manual insertion   or  2 --> random insertion : >? 1
player_id: >? 20
player_name: >? Shabanov
player_salary: >? 70000
agent_id: >? 10
ПОМИЛКА:  insert або update в таблиці "player" порушує обмеження зовнішнього ключа "player_agent_id_fkey"
DETAIL:  Ключ (agent_id)=(10) не присутній в таблиці "agent".
```

Редагування даних (UPDATE)

Club (початковий стан таблиці)

```
            Tables :
        1.player
        2.contract_offer
        3.club
        4.head_coach
        5.agent
        >? 3
+---------+------------+---------------+----------+
| club_id | club_name  |  club_league  | coach_id |
+---------+------------+---------------+----------+
|    1    |   Zorya    |      UPL      |    1     |
|    2    | Liverpool  | PremierLeague |    4     |
|    3    |  ManCity   | PremierLeague |    3     |
|    4    |   Genoa    |     SeriaA    |    2     |
|    5    |    Roma    |     SeriaA    |    5     |
|    6    |   Bayern   |  BundesLeague |    10    |
|    7    |  Shakhtar  |      UPL      |    6     |
|    8    | RealMadrid |     LaLiga    |    9     |
|    9    | Barcelona  |     LaLiga    |    7     |
|    10   |   Napoli   |     SeriaA    |    8     |
+---------+------------+---------------+----------+

        1 --> menu    or    2 --> exit :
```

```
query_on_update = """UPDATE club set club_name = %s, club_league = %s, coach_id = %s
WHERE club_id = %s"""
```

```
Edit line where club_id = :>? 1
club_name: >? ZoryaLuhansk
club_league: >? UkraineLeague
coach_id :>? 1
1 records updated

        1 --> menu    or    2 --> exit :
```

Club (таблиця після редагування) :

```
+---------+--------------+---------------+----------+
| club_id |  club_name   |  club_league  | coach_id |
+---------+--------------+---------------+----------+
|    1    | ZoryaLuhansk | UkraineLeague |    1     |
|    2    |  Liverpool   | PremierLeague |    4     |
|    3    |   ManCity    | PremierLeague |    3     |
|    4    |    Genoa     |     SeriaA    |    2     |
|    5    |     Roma     |     SeriaA    |    5     |
|    6    |    Bayern    |  BundesLeague |    10    |
|    7    |   Shakhtar   |      UPL      |    6     |
|    8    |  RealMadrid  |     LaLiga    |    9     |
|    9    |  Barcelona   |     LaLiga    |    7     |
|    10   |    Napoli    |     SeriaA    |    8     |
+---------+--------------+---------------+----------+

        1 --> menu   or    2 --> exit :
```

Помилка при редагуванні дочірньої(club) таблиці на неіснуюче значення батьківської таблиці(head_coach) :

```
Edit line where club_id = :>? 1
club_name: >? ZoryaLuhansk
club_league: >? UkraineLeague
coach_id :>? 20
{} ПОМИЛКА:  insert або update в таблиці "club" порушує обмеження зовнішнього ключа "club_coach_id_fkey"
DETAIL:  Ключ (coach_id)=(20) не присутній в таблиці "head_coach".
```

Видалення даних (DELETE)

Contract_offer (початковий стан таблиці) :

offer_id	player_id	club_id	proposed_salary	proposed_duration
1	1	6	130000	4
2	3	3	104735	2
3	2	2	101178	1
4	5	5	133424	3
5	1	1	81865	1
6	9	9	177968	5
7	10	9	183963	5
8	6	6	143239	3
9	7	7	152913	4
10	1	1	86335	1

```
sql_delete_query = """ DELETE FROM contract_offer WHERE offer_id = %s"""
```

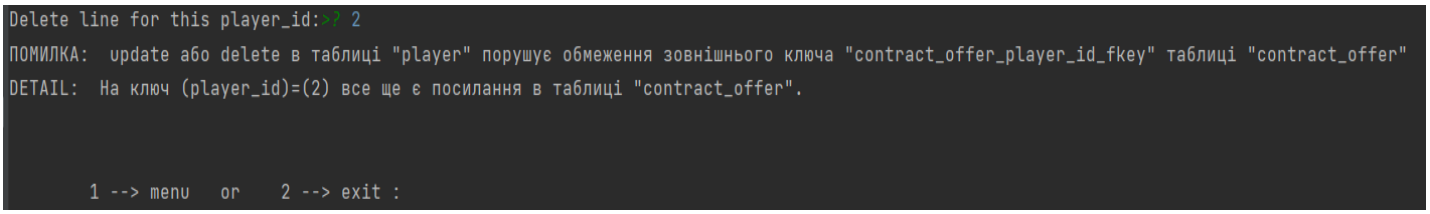
```
Delete line for this offer_id: >? 5
1 record deleted


        1 --> menu    or    2 --> exit :
```

Contract_offer (таблиця після видалення)

offer_id	player_id	club_id	proposed_salary	proposed_duration
1	1	6	130000	4
2	3	3	104735	2
3	2	2	101178	1
4	5	5	133424	3
6	9	9	177968	5
7	10	9	183963	5
8	6	6	143239	3
9	7	7	152913	4
10	1	1	86335	1

Помилка при видаленні даних у батьківській таблиці(player), які збігаються зі зв’язними даними дочірньої таблиці(contract_offer) :



Завдання 2

Вставка 5 рандомізованих рядків у кожну з таблиць

Player (початковий стан таблиці):

player_id	player_name	player_salary	agent_id
1	Tsygankov	120000	1
2	Mykolenko	110000	4
3	Shaparenko	135000	None
4	Buyalkiy	130000	2
5	Buschan	90000	1
6	Zabarnyi	83000	None
7	Syrota	60000	None
8	Verbic	100000	3
9	Karavaiev	90000	4
10	Sydorchuk	150000	2
11	Besedin	90000	2

Лістинг запиту :

```
query_on_random_insert = """ INSERT INTO player (player_id, player_name, player_salary, agent_id)
SELECT player_id, player_name, player_salary, agent_id FROM
            (SELECT MAX(player.player_id)+1 as player_id,
            floor(random()*(200000-80000 + 1) + 80000) as player_salary,
            substr(md5(random()::text), 0, 15) as player_name,
            floor(random()*MAX(agent.agent_id))+1 as agent_id
            FROM player, agent tablesample BERNOULLI(100)
            ORDER BY random()) k, generate_series(1, 100000) LIMIT 1
"""
```

```
            Tables :
        1.player
        2.contract_offer
        3.club
        4.head_coach
        5.agent
        >? 1

        1 --> manual insertion   or  2 --> random insertion : >? 2
The number of lines you want to generate:>? 5
5 record successfully inserted
```

Player (стан таблиці після вставки рандомізованих даних):

player_id	player_name	player_salary	agent_id
1	Tsygankov	120000	1
2	Mykolenko	110000	4
3	Shaparenko	135000	None
4	Buyalkiy	130000	2
5	Buschan	90000	1
6	Zabarnyi	83000	None
7	Syrota	60000	None
8	Verbic	100000	3
9	Karavaiev	90000	4
10	Sydorchuk	150000	2
11	Besedin	90000	2
12	3d488bab9c96d0	164698	3
13	a2a8c16b974910	122323	2
14	16faf4d83d9687	150930	3
15	23c9c2b0be510b	194928	4
16	02ce0e19a49dd8	117277	2

Contract_offer (початковий стан таблиці)

offer_id	player_id	club_id	proposed_salary	proposed_duration
1	1	6	130000	4
2	3	3	104735	2
3	2	2	101178	1
4	5	5	133424	3
6	9	9	177968	5
7	10	9	183963	5
8	6	6	143239	3
9	7	7	152913	4
10	1	1	86335	1

Лістинг запиту :

```
query_on_random_insert = """ INSERT INTO contract_offer (offer_id, player_id, club_id, proposed_salary, proposed_duration)
SELECT offer_id, player_id, club_id, proposed_salary, proposed_duration FROM
       (SELECT MAX(contract_offer.offer_id)+1 as offer_id,
       floor(random()*(200000-80000 + 1) + 80000) as proposed_salary,
       floor(random()*(5 - 1 + 1) + 1) as proposed_duration,
       floor(random()*MAX(player.player_id))+1 as player_id,
       floor(random()*MAX(club.club_id))+1 as club_id
       FROM contract_offer, player, club tablesample BERNOULLI(100)
       ORDER BY random()) k, generate_series(1, 100000) LIMIT 1
"""
```

```
            Tables :
        1.player
        2.contract_offer
        3.club
        4.head_coach
        5.agent
        >? 2

        1 --> manual insertion   or  2 --> random insertion : >? 2
The number of lines you want to generate:>? 5
5 record successfully inserted

        1 --> menu    or    2 --> exit :
```

Contract_offer (стан таблиці після вставки рандомізованих даних) :

offer_id	player_id	club_id	proposed_salary	proposed_duration
1	1	6	130000	4
2	3	3	104735	2
3	2	2	101178	1
4	5	5	133424	3
6	9	9	177968	5
7	10	9	183963	5
8	6	6	143239	3
9	7	7	152913	4
10	1	1	86335	1
11	11	7	159012	4
12	16	10	199068	5
13	7	4	127132	2
14	13	9	176312	5
15	10	7	152482	4

Club (початковий стан таблиці)

club_id	club_name	club_league	coach_id
1	ZoryaLuhansk	UkraineLeague	1
2	Liverpool	PremierLeague	4
3	ManCity	PremierLeague	3
4	Genoa	SeriaA	2
5	Roma	SeriaA	5
6	Bayern	BundesLeague	10
7	Shakhtar	UPL	6
8	RealMadrid	LaLiga	9
9	Barcelona	LaLiga	7
10	Napoli	SeriaA	8

Лістинг запиту :

```
query_on_random_insert = """ INSERT INTO club (club_id, club_name, club_league, coach_id)
            SELECT club_id, club_name, club_league, coach_id FROM
            (SELECT MAX(club.club_id)+1 as club_id,
            substr(md5(random()::text), 0, 15) as club_name,
            substr(md5(random()::text), 0, 15) as club_league,
            floor(random()*MAX(head_coach.coach_id))+1 as coach_id
            FROM club, head_coach tablesample BERNOULLI(100)
            ORDER BY random()) k, generate_series(1, 100000) LIMIT 1
"""
```

```
            Tables :
        1.player
        2.contract_offer
        3.club
        4.head_coach
        5.agent
        >? 3

        1 --> manual insertion   or  2 --> random insertion : >? 2
The number of lines you want to generate:>? 5
5 record successfully inserted

        1 --> menu   or    2 --> exit :
```

Club (стан таблиці після вставки рандомізованих даних):

club_id	club_name	club_league	coach_id
1	ZoryaLuhansk	UkraineLeague	1
2	Liverpool	PremierLeague	4
3	ManCity	PremierLeague	3
4	Genoa	SeriaA	2
5	Roma	SeriaA	5
6	Bayern	BundesLeague	10
7	Shakhtar	UPL	6
8	RealMadrid	LaLiga	9
9	Barcelona	LaLiga	7
10	Napoli	SeriaA	8
11	a308e8846514b2	a308e8846514b2	1
12	fc55292e72009b	fc55292e72009b	9
13	876c0f1767fbec	876c0f1767fbec	8
14	e0da5b50a65f7c	e0da5b50a65f7c	5
15	809c73be521674	809c73be521674	5

Head_coach (початковий стан таблиці) :

coach_id	coach_name
1	Skrypnyk
2	Shevchenko
3	Guardiola
4	Klopp
5	Mourinho
6	DeZerbi
7	Xavi
8	Spalletti
9	Ancelotti
10	Nagelsmann

Лістинг запиту :

```
query_on_random_insert = """ INSERT INTO head_coach (coach_id, coach_name)
            SELECT coach_id, coach_name FROM
            (SELECT MAX(head_coach.coach_id)+1 as coach_id,
            substr(md5(random()::text), 0, 15) as coach_name
            FROM head_coach tablesample BERNOULLI(100)
            ORDER BY random()) k ,generate_series(1, 100000) LIMIT 1
            """
```

```
            Tables :
        1.player
        2.contract_offer
        3.club
        4.head_coach
        5.agent
        >? 4

        1 --> manual insertion   or  2 --> random insertion : >? 2
The number of lines you want to generate:>? 5
5 record successfully inserted

        1 --> menu   or    2 --> exit :
```

Head_coach (стан таблиці після вставки рандомізованих даних) :

coach_id	coach_name
1	Skrypnyk
2	Shevchenko
3	Guardiola
4	Klopp
5	Mourinho
6	DeZerbi
7	Xavi
8	Spalletti
9	Ancelotti
10	Nagelsmann
11	c59c830bbfe282
12	6a066da825582d
13	30628f0f4d63bd
14	6013cafe522475
15	0c2b82f5dabc99

Agent (початковий стан таблиці)

agent_id	agent_name
1	Shabliy
2	Raiola
3	Anderson
4	Zahavi

Лістинг запиту :

```
query_on_random_insert = """ INSERT INTO agent (agent_id, agent_name)
SELECT agent_id, agent_name FROM
            (SELECT MAX(agent.agent_id)+1 as agent_id,
            substr(md5(random()::text), 0, 15) as agent_name
            FROM agent tablesample BERNOULLI(100)
            ORDER BY random()) k ,generate_series(1, 100000) LIMIT 1
                """
```

```
            Tables :
        1.player
        2.contract_offer
        3.club
        4.head_coach
        5.agent
        >? 5

        1 --> manual insertion   or  2 --> random insertion : >? 2
The number of lines you want to generate:>? 5
5 record successfully inserted
```

Agent (стан таблиці після вставки рандомізованих даних)

agent_id	agent_name
1	Shabliy
2	Raiola
3	Anderson
4	Zahavi
5	2437a8b2df3779
6	fee360221add99
7	1a356c8e4d1899
8	b47b3e3b560b50
9	3c4cd08f3459ac

Завдання 3

Виконання трьох запитів спеціального пошуку

1.

Умова запиту :

```
1) Show 'player_id', 'player_name', 'player_salary', 'agent_name',
                where 'agent_name' = ... and 'player_id' >= ...
```

Лістинг запиту :

```
query_on_specific_filter = """select  player_id, player_name, player_salary, agent_name
from (select player.player_id, player.player_name, player.player_salary, agent.agent_name
from player inner join agent on agent.agent_id=player.agent_id
where agent.agent_name = %s  group by player.player_id, player.player_name, player.player_salary, agent.agent_name)
AS foo where player_id >= %s;
"""
```

Результат :

```
        Your choice : >? 1
player_id >= : >? 5
agent_name = : >? Raiola
Time:  5  ms
+-----------+----------------+---------------+------------+
| player_id |  player_name   | player_salary | agent_name |
+-----------+----------------+---------------+------------+
|     10    |   Sydorchuk    |     150000    |   Raiola   |
|     11    |    Besedin     |     90000     |   Raiola   |
|     13    | a2a8c16b974910 |     122323    |   Raiola   |
|     16    | 02ce0e19a49dd8 |     117277    |   Raiola   |
+-----------+----------------+---------------+------------+
```

2.

Умова запиту :

```
2) Show 'player_name', 'club_name', 'proposed_salary', 'player_salary',
                where 'club_name' = ... and 'player_salary' < ... and 'proposed_salary' > ...
```

Лістинг запиту :

```
query_on_specific_filter = """ select player_name, club_name, proposed_salary, player_salary
from (select player_name, club.club_name, player.player_salary, contract_offer.proposed_salary from player inner join contract_offer
on player.player_id = contract_offer.player_id inner join club on club.club_id = contract_offer.club_id where club_name = %s
group by player.player_name, club.club_name, player.player_salary, contract_offer.proposed_salary)
AS foo WHERE player_salary < %s and proposed_salary > %s;
"""
```

Результат :

```
        Your choice : >? 2
club_name = : >? Barcelona
player_salary < : >? 130000
proposed_salary > : >? 100000
Time:  8  ms
+----------------+-----------+---------------+------------------+
|  player_name   | club_name | player_salary | proposed_salary |
+----------------+-----------+---------------+------------------+
| a2a8c16b974910 | Barcelona |       9       |      176312      |
|   Karavaiev    | Barcelona |       9       |      177968      |
+----------------+-----------+---------------+------------------+
```

3.

Умова запиту :

```
3) Show 'coach_name', 'club_league', 'club_name', 'offer_id', 'player_id',
                where 'coach_name' = ... and 'offer_id' >= ... and 'player_id' < ...
```

Лістинг запиту :

```
query_on_specific_filter = """ select coach_name, club_league, club_name, offer_id, player_id
from (select coach_name, club_league, club_name, offer_id, player_id from head_coach inner join club
on club.coach_id = head_coach.coach_id inner join contract_offer on club.club_id = contract_offer.club_id
where head_coach.coach_name = %s group by coach_name, club_league, club_name, offer_id, player_id)
AS foo WHERE offer_id >= %s and player_id < %s;
"""
```

Результат :

```
        Your choice : >? 3
coach_name = : >? DeZerbi
offer_id >= : >? 11
player_id < : >? 15
Time:  11  ms
+------------+-------------+-----------+----------+-----------+
| coach_name | club_league | club_name | offer_id | player_id |
+------------+-------------+-----------+----------+-----------+
|  DeZerbi   |     UPL     |  Shakhtar |    11    |     11    |
|  DeZerbi   |     UPL     |  Shakhtar |    15    |     10    |
|  DeZerbi   |     UPL     |  Shakhtar |    19    |     10    |
|  DeZerbi   |     UPL     |  Shakhtar |    20    |     11    |
+------------+-------------+-----------+----------+-----------+
```

Завдання 4

model.py

```
import time
import psycopg2


class Model:
    def __init__(self):
        try:
            self.connection = psycopg2.connect(
                user="postgres",
                password="qwerty",
                host="localhost",
                port=5432,
                database="lab1",
            )
            self.cursor = self.connection.cursor()
        except:
            print('Error connection...')

    def insert(self, count, query_on_insert, record, type):
        for i in range(int(count)):
            try:
                cursor = self.connection.cursor()
                if type == 'm':
                    cursor.executemany(query_on_insert, record)
                elif type == 'r':
                    cursor.execute(query_on_insert)
                self.connection.commit()
            except (Exception, psycopg2.Error) as error:
                print("{}".format(error))
                self.connection.rollback()
                break
            finally:
                if self.connection:
                    cursor.close()
        print(count, "record successfully inserted")

    def delete(self, records, query_on_delete):
        try:
            cursor = self.connection.cursor()
            cursor.executemany(query_on_delete, records)
            self.connection.commit()
            print(cursor.rowcount, "record deleted")
```

```
        except (Exception, psycopg2.Error) as error:
            print("{}".format(error))
            self.connection.rollback()
        finally:
            if self.connection:
                cursor.close()

    def update(self, records, query_on_update):
        try:
            cursor = self.connection.cursor()
            cursor.executemany(query_on_update, records)
            self.connection.commit()
            row_count = cursor.rowcount
            print(row_count, "records updated")
        except (Exception, psycopg2.Error) as error:
            print("{}", error)
            self.connection.rollback()
        finally:
            if self.connection:
                cursor.close()

    def show_table(self, sql_select_query, tab, record, bool):
        try:
            self.cursor = self.connection.cursor()
            if bool:
                beg = int(time.time() * 1000)
            self.cursor.execute(sql_select_query, record)
            if bool:
                end = int(time.time() * 1000) - beg
                print("Time: ", end, " ms")
            records = self.cursor.fetchall()
            return records
        except (Exception, psycopg2.Error) as error:
            print("Error while fetching data from PostgreSQL", error)
            self.connection.rollback()
        finally:
            if self.connection:
                self.cursor.close()

    def check_id(self, val, num, err_func):
        try:
            rec = [(val,)]
            sql_origin = SqlMiddle.origin_type(num)
            curs = self.connection.cursor()
            curs.execute(sql_origin, rec)
            records = curs.fetchall()
```

```
            if records[0] is None:
                for row in records:
                    print("There is row with id: ", row[0])
        except (Exception, psycopg2.Error) as error:
            print("{}".format(error))
            err_func()
        finally:
            if self.connection:
                curs.close()


class SqlMiddle:
    def query_insert_func(num_of_table):
        if num_of_table == '1':
            query_on_insert = """ INSERT INTO player (player_id, player_name,
player_salary, agent_id)
            VALUES (%s, %s, %s, %s)"""
        elif num_of_table == '2':
            query_on_insert = """ INSERT INTO contract_offer (offer_id, player_id,
club_id, proposed_salary, proposed_duration)
            VALUES (%s, %s, %s, %s, %s)"""
        elif num_of_table == '3':
            query_on_insert = """ INSERT INTO club (club_id, club_name, club_league,
coach_id)
            VALUES (%s, %s, %s, %s)"""
        elif num_of_table == '4':
            query_on_insert = """ INSERT INTO head_coach (coach_id, coach_name)
            VALUES (%s, %s)"""
        elif num_of_table == '5':
            query_on_insert = """ INSERT INTO agent (agent_id, agent_name)
            VALUES (%s, %s)"""
        return query_on_insert

    def query_insert_random_func(num_of_table):
            if num_of_table == '1':
                query_on_random_insert = """ INSERT INTO player (player_id, player_name,
player_salary, agent_id)
                SELECT player_id, player_name, player_salary, agent_id FROM
                            (SELECT MAX(player.player_id)+1 as player_id,
                            floor(random()*(200000-80000 + 1) + 80000) as player_salary,
                            substr(md5(random()::text), 0, 15) as player_name,
                            floor(random()*MAX(agent.agent_id))+1 as agent_id
                            FROM player, agent tablesample BERNOULLI(100)
                            ORDER BY random()) k, generate_series(1, 100000) LIMIT 1
                """
            elif num_of_table == '2':
```

```
                query_on_random_insert = """ INSERT INTO contract_offer (offer_id,
player_id, club_id, proposed_salary, proposed_duration)
                SELECT offer_id, player_id, club_id, proposed_salary, proposed_duration
FROM
                        (SELECT MAX(contract_offer.offer_id)+1 as offer_id,
                        floor(random()*(200000-80000 + 1) + 80000) as proposed_salary,
                        floor(random()*(5 - 1 + 1) + 1) as proposed_duration,
                        floor(random()*MAX(player.player_id))+1 as player_id,
                        floor(random()*MAX(club.club_id))+1 as club_id
                        FROM contract_offer, player, club tablesample BERNOULLI(100)
                        ORDER BY random()) k, generate_series(1, 100000) LIMIT 1
                 """
            elif num_of_table == '3':
                query_on_random_insert = """ INSERT INTO club (club_id, club_name,
club_league, coach_id)
                            SELECT club_id, club_name, club_league, coach_id FROM
                            (SELECT MAX(club.club_id)+1 as club_id,
                            substr(md5(random()::text), 0, 15) as club_name,
                            substr(md5(random()::text), 0, 15) as club_league,
                            floor(random()*MAX(head_coach.coach_id))+1 as coach_id
                            FROM club, head_coach tablesample BERNOULLI(100)
                            ORDER BY random()) k, generate_series(1, 100000) LIMIT 1
                """
            elif num_of_table == '4':
                query_on_random_insert = """ INSERT INTO head_coach (coach_id,
coach_name)
                            SELECT coach_id, coach_name FROM
                            (SELECT MAX(head_coach.coach_id)+1 as coach_id,
                            substr(md5(random()::text), 0, 15) as coach_name
                            FROM head_coach tablesample BERNOULLI(100)
                            ORDER BY random()) k ,generate_series(1, 100000) LIMIT 1
                            """
            elif num_of_table == '5':
                query_on_random_insert = """ INSERT INTO agent (agent_id, agent_name)
                SELECT agent_id, agent_name FROM
                            (SELECT MAX(agent.agent_id)+1 as agent_id,
                            substr(md5(random()::text), 0, 15) as agent_name
                            FROM agent tablesample BERNOULLI(100)
                            ORDER BY random()) k ,generate_series(1, 100000) LIMIT 1
                                """
            return query_on_random_insert

        def query_update_func(num_of_table):
            if num_of_table == '1':
                query_on_update = """UPDATE player set player_name = %s, player_salary = %s,
agent_id = %s
```

```
            WHERE "player_id" = %s """
        elif num_of_table == '2':
            query_on_update = """UPDATE contract_offer set player_id = %s, club_id = %s,
proposed_salary = %s, proposed_duration = %s
            WHERE "offer_id" = %s """
        elif num_of_table == '3':
            query_on_update = """UPDATE club set club_name = %s, club_league = %s,
coach_id = %s
            WHERE club_id = %s"""
        elif num_of_table == '4':
            query_on_update = """UPDATE head_coach set coach_name = %s
              WHERE coach_id = %s """
        elif num_of_table == '5':
            query_on_update = """UPDATE agent set agent_name = %s
              WHERE agent_id = %s """
        return query_on_update

    def query_delete_func(num_of_table):
        if num_of_table == '1':
            query_on_delete = """ DELETE FROM player WHERE player_id = %s"""
        elif num_of_table == '2':
            query_on_delete = """ DELETE FROM contract_offer WHERE offer_id = %s"""
        elif num_of_table == '3':
            query_on_delete = """ DELETE FROM club WHERE club_id = %s"""
        elif num_of_table == '4':
            query_on_delete = """ DELETE FROM head_coach WHERE coach_id = %s"""
        elif num_of_table == '5':
            query_on_delete = """ DELETE FROM agent WHERE agent_id = %s"""
        return query_on_delete

    def query_specific_func(num_of_table):
        if num_of_table == '1':
            query_on_specific_filter = """select  player_id, player_name, player_salary,
agent_name
            from (select player.player_id, player.player_name, player.player_salary,
agent.agent_name
            from player inner join agent on agent.agent_id=player.agent_id
            where agent.agent_name = %s  group by player.player_id, player.player_name,
player.player_salary, agent.agent_name)
            AS foo where player_id >= %s;
            """
        elif num_of_table == '2':
            query_on_specific_filter = """ select player_name, club_name,
proposed_salary, player_salary
            from (select player_name, club.club_name, player.player_salary,
contract_offer.proposed_salary from player inner join contract_offer
```

```
            on player.player_id = contract_offer.player_id inner join club on
club.club_id = contract_offer.club_id where club_name = %s
            group by player.player_name, club.club_name, player.player_salary,
contract_offer.proposed_salary)
            AS foo WHERE player_salary < %s and proposed_salary > %s;
            """
        elif num_of_table == '3':
            query_on_specific_filter = """ select coach_name, club_league, club_name,
offer_id, player_id
            from (select coach_name, club_league, club_name, offer_id, player_id from
head_coach inner join club
            on club.coach_id = head_coach.coach_id inner join contract_offer on
club.club_id = contract_offer.club_id
            where head_coach.coach_name = %s group by coach_name, club_league,
club_name, offer_id, player_id)
            AS foo WHERE offer_id >= %s and player_id < %s;
            """
        return query_on_specific_filter

    def query_select_func(num_of_table):
        if num_of_table == '1':
            sql_select_query = """ SELECT * FROM player ORDER BY player_id"""
        elif num_of_table == '2':
            sql_select_query = """ SELECT * FROM contract_offer ORDER BY offer_id"""
        elif num_of_table == '3':
            sql_select_query = """ SELECT * FROM club ORDER BY club_id"""
        elif num_of_table == '4':
            sql_select_query = """ SELECT * FROM head_coach ORDER BY coach_id"""
        elif num_of_table == '5':
            sql_select_query = """ SELECT * FROM agent ORDER BY agent_id"""
        return sql_select_query

    def origin_type(num_of_table):
        if num_of_table == 1:
            sql_origin_val = """ SELECT player.player_id FROM player WHERE player_id =
%s """
        elif num_of_table == 2:
            sql_origin_val = """ SELECT contract_offer.offer_id FROM contract_offer
WHERE offer_id = %s """
        elif num_of_table == 3:
            sql_origin_val = """ SELECT club.club_id FROM club WHERE club_id = %s """
        elif num_of_table == 4:
            sql_origin_val = """ SELECT head_coach.coach_id FROM head_coach WHERE
coach_id = %s"""
        elif num_of_table == 5:
            sql_origin_val = """ SELECT agent.agent_id FROM agent WHERE agent_id = %s"""
```

```
        return sql_origin_val


def string_validator(value, err_func):
    if value is None or value == '':
        print("ERROR: column cannot contain NULL value")
        err_func()
    return True


def int_validator(value, err_func):
    try:
        int(value)
    except ValueError:
        print("ERROR: WRONG CHARACTER (expected int) !!!")
        err_func()
```

Модуль model.py вміщує у себе основний клас Model та клас SqlMiddle.

Клас **Model** містить методи взаємодії з базою даних та конструктор підключення до неї. Детальніше про функції із цього класу :

`insert()` - приймає текст запиту на вставку даних як аргумент та безпосередньо відправляє його до БД.

`delete()` - приймає текст запиту на видалення даних як аргумент та безпосередньо відправляє його до БД.

`update()` - приймає текст запиту на редагування даних як аргумент та безпосередньо відправляє його до БД.

`show_table()`- приймає текст запиту на вивід певних даних та безпосередньо відправляє його до БД.

`check_id()`- допоміжна функція для перевірки наявності елемента за його ID.

Клас **SqlMiddle** містить код різних запитів postgresql, які при виконанні відповідної умови передаються у функції класу Model для реалізації цього запиту.

Додатково у цьому модулі наведені функції string_validator() та int_validator(), які виконують валідацію введених даних типу string та int відповідно.