



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»

Тема: «Створення додатку бази даних,
орієнтованого на взаємодію з СУБД
PostgreSQL»

Виконав: студент III курсу
ФПМ групи КВ-84
Мелюх В. В.
Перевірив: Петрашенко А.В.

Київ – 2020

Загальне завдання:

1. Реалізувати функції 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).

URL репозиторію (повний):

[https://github.com/PointProgram/DataBase/tree/master/Database%20and%20management%20tools\(Lab%202\)](https://github.com/PointProgram/DataBase/tree/master/Database%20and%20management%20tools(Lab%202))

URL репозиторію (неповний):

<https://github.com/PointProgram/>

Завдання 1:

Insert:

Table: match

```
Choose option:
PRESS: 1 to add... 2 to update... 3 to delete... 4 to specific_select... 5 to show_table...
1
Choose table:
  press 1 - Match...   press 2 - Stadium...   press 3 - StartDate...   press 4 - Team
1
Do you want to insert rows manually or randomly?
Press M if manually, R if randomly: R
What number of row do you want to add? - 1
Enter values following this sequence: Match_id, opp_score, own_score
Match_id: 1232
opp_score: 12
own_score: 5

1 Record successfully inserted
Do you want to continue? Press Y if yes, N if no: |
```

```

PRESS: 1 to add... 2 to update... 3 to delete... 4 to specific_select... 5 to show_table...
5
Choose table:
press 1 - Match... press 2 - Stadium... press 3 - StartDate... press 4 - Team
1
Show each row and it's columns values
Match_id  opp_score  own_score
1          2         1
2          4         5
3          2         2

```

1123	2	3
1232	12	5
2342	2	3

```

Enter values following this sequence: Match_id, opp_score, own_score
Match_id: 1
opp_score: 3
own_score: 2

Failed inserting record into table ПОМИЛКА: повторювані значення ключа порушують обмеження унікальності "Match_pkey"
DETAIL: Ключ (match_id)=(1) вже існує.

Do you want to continue? Press Y if yes, N if no:

```

```

Match_id: 7
You entered wrong instance, please try again!
Do you want to exit? Press M to go to the main menu, or E to exit: |

```

```

Enter values following this sequence: Match_id, opp_score, own_score
Match_id: 2366
opp_score: 7
You entered wrong instance, please try again!
Do you want to exit? Press M to go to the main menu, or E to exit: |

```

```

Enter values following this sequence: Match_id, opp_score, own_score
Match_id: 5353
opp_score: 2
own_score: .
You entered wrong instance, please try again!
Do you want to exit? Press M to go to the main menu, or E to exit: |

```

Table: stadium

```

Choose table:
  press 1 - Match...   press 2 - Stadium...   press 3 - StartDate... press 4 - Team
2
Do you want to insert rows manually or randomly?
Press M if manually, R if randomly: M
What number of row do you want to add? - 1
Enter values following this sequence: Stadium_id, Name, City
Stadium_id: 3234
Name: Great stadium
City: Kyiv

1 Record successfully inserted
Do you want to continue? Press Y if yes, N if no:

```

54	fe	wr
445		lile
3234	Great stadium Kyiv	

```

Enter values following this sequence: Stadium_id, Name, City
Stadium_id: 2
Name: FR
City: GRY

Failed inserting record into table ПОМИЛКА: повторювані значення ключа порушують обмеження унікальності "Stadium_pkey"
DETAIL: Ключ (stadium_id)=(2) вже існує.

```

```

Enter values following this sequence: Stadium_id, Name, City
Stadium_id: 445
Name:
City: lile

1 Record successfully inserted

```

Table: startdate

```

Choose table:
  press 1 - Match...   press 2 - Stadium...   press 3 - StartDate... press 4 - Team
3
Do you want to insert rows manually or randomly?
Press M if manually, R if randomly: M
What number of row do you want to add? - 1
Enter values following this sequence: Team_id, Match_id, Start_date(yyyy-mm-dd)
Team_id: 2
Match_id: 3
Start_date: 2018-09-12

1 Record successfully inserted
Do you want to continue? Press Y if yes, N if no:

```

2	2	2020-09-23
2	3	2018-09-12
3	4	2018-02-02

```

Enter values following this sequence: Team_id, Match_id, Start_date(yyyy-mm-dd)
Team_id: 2
Match_id: 1234
Failed, there are no records with such id: list index out of range
Do you want to exit? Press M to go to the main menu, or E to exit: |

```

```

Enter values following this sequence: Team_id, Match_id, Start_date(yyyy-mm-dd)
Team_id: 3
Match_id: 4
Start_date: 2018-02-02

1 Record successfully inserted
Do you want to continue? Press Y if yes, N if no: Y
What number of row do you want to add? - 1
Enter values following this sequence: Team_id, Match_id, Start_date(yyyy-mm-dd)
Team_id: 3
Match_id: 4
Start_date: 2018-03-03

Failed inserting record into table ПОМИЛКА: повторювані значення ключа порушують обмеження унікальності "PK_StartDate"
DETAIL: Ключ (team_id, match_id)=(3, 4) вже існує.

```

```

Enter values following this sequence: Team_id, Match_id, Start_date(yyyy-mm-dd)
Team_id: 1
Match_id: 4
Start_date: 2-2-=2
You entered wrong date value, please try again!

```

```

Enter values following this sequence: Team_id, Match_id, Start_date(yyyy-mm-dd)
Team_id: 1
Match_id: 3
Start_date:
You entered wrong date value, please try again!

```

Table: team

```

Choose table:
  press 1 - Match...   press 2 - Stadium...   press 3 - StartDate... press 4 - Team
4
Do ypo want to insert rows manually or randomly?
Press M if manually, R if randomly: M
What number of row do you want to add? - 1
Enter values following this sequence: Team_id, Name, Opponent, Coach, Stadium_id
Team_id: 1242
Name: Beste
Opponent: Worse
Coach: Coach
Stadium_id: 2

1 Record successfully inserted

```

956	NCPCRW	LEIMYU	ABILVD	5
1242	Beste	Worse	Coach	2
2326	ret	gef		3

```
Enter values following this sequence: Team_id, Name, Opponent, Coach, Stadium_id
Team_id: 1234
Name: fff
Opponent: ddd
Coach: dfdfd
Stadium_id: 456774
Failed, there are no records with such id: list index out of range
```

```
Enter values following this sequence: Team_id, Name, Opponent, Coach, Stadium_id
Team_id: 2326
Name: ret
Opponent: gef
Coach:
Stadium_id: 3

1 Record successfully inserted
```

```
Enter values following this sequence: Team_id, Name, Opponent, Coach, Stadium_id
Team_id: 2535
Name:
Error that column cannot contain NULL values!
```

Update:

Table: match

```
Choose option:
PRESS: 1 to add... 2 to update... 3 to delete... 4 to specific_select... 5 to show_table...
2
Choose table:
press 1 - Match... press 2 - Stadium... press 3 - StartDate... press 4 - Team
1
What number of rows do you want to update? - 1
Enter values following this sequence opp_score, own_score, Match_id:
opp_score: 4
own_score: 4
Match_id: 5

1 Records Updated
```

4	1	0
5	2	2
8	5	7

4	1	0
5	4	4
8	5	7

```
Enter values following this sequence opp_score, own_score, Match_id:
opp_score:
Error that column cannot contain NULL values!
```

```
Enter values following this sequence opp_score, own_score, Match_id:
opp_score: 1
own_score: 1
Match_id: 452635
Failed, there are no records with such id: list index out of range
```

```
Enter values following this sequence opp_score, own_score, Match_id:
opp_score: 3
own_score: 9
You entered wrong instance, please try again!
```

Table: stadium

```
press 1 - Match... press 2 - Stadium... press 3 - StartDate... press 4 - Team
2
What number of rows do you want to update? - 1
Enter values following this sequence Name, City, Stadium_id:
Name: Lviv
City: Paris
Stadium_id: 6

1 Records Updated
```

5	Kotram	Kamyanka
6	FRRR	dfFF
7	XTRKJ	QOMFVD

5	Kotram	Kamyanka
6	Lviv	Paris
7	XTRKJ	QOMFVD

```
Enter values following this sequence Name, City, Stadium_id:
Name: Roty
City: ddf
Stadium_id: 225735
Failed, there are no records with such id: list index out of range
```

```
Enter values following this sequence Name, City, Stadium_id:
Name: Frop
City:
Error that column cannot contain NULL values!
```

Table: startdate

```
press 1 - Match... press 2 - Stadium... press 3 - StartDate... press 4 - Team
3
What number of rows do you want to update? - 1
Enter values following this sequence Start_date(yyyy-mm-dd), Team_id, Match_id:
Start_date: 2000-02-01
Team_id: 3
Match_id: 4

1 Records Updated
```

2	3	2018-09-12
3	4	2018-02-02
5	5	2020-09-25

2	3	2018-09-12
3	4	2000-02-01
5	5	2020-09-25

```
Enter values following this sequence Start_date(yyyy-mm-dd), Team_id, Match_id:
Start_date: 45
You entered wrong date value, please try again!
```

```
Enter values following this sequence Start_date(yyyy-mm-dd), Team_id, Match_id:
Start_date: 2017-03-03
Team_id: 1
Match_id: 56
Failed, there are no records with such id: list index out of range
```

```
Enter values following this sequence Start_date(yyyy-mm-dd), Team_id, Match_id:
Start_date: 2020-02-02
Team_id: 3
Match_id: 3

0 Records Updated
```

```
Enter values following this sequence Start_date(yyyy-mm-dd), Team_id, Match_id:
Start_date: 2010-02-02
Team_id: f
You entered wrong instance, please try again!
```

Table: team

```
press 1 - Match... press 2 - Stadium... press 3 - StartDate... press 4 - Team
4
What number of rows do you want to update? - 1
Enter values following this sequence Name, Opponent, Coach, Stadium_id, Team_id:
Name: Lokre
Opponent: Orek
Coach: Trainee
Stadium_id: 4
Team_id: 6

1 Records Updated
```

5	Lutsechk	Prypiat	Tymochech D.F.	5
6	Bglgl	gfgfgfgf	gfgfgf	1
7	Brosław	Corole	Lortus	4

5	Lutsechk	Prypiat	Tymochech D.F.	5
6	Lokre	Orek	Trainee	4
7	Brosław	Corole	Lortus	4


```
Enter values following this sequence Name, Opponent, Coach, Stadium_id, Team_id:
Name: fff
Opponent: ddd
Coach: hhh
Stadium_id: 13424
Failed, there are no records with such id: list index out of range
```

```
Enter values following this sequence Name, Opponent, Coach, Stadium_id, Team_id:
Name:
Error that column cannot contain NULL values!
```

```
Enter values following this sequence Name, Opponent, Coach, Stadium_id, Team_id:
Name: Frng
Opponent: Dfd
Coach:
Stadium_id: 3
Team_id: 3

1 Records Updated
```

```
Enter values following this sequence Name, Opponent, Coach, Stadium_id, Team_id:
Name: rr
Opponent: rr
Coach: rr
Stadium_id: 4
Team_id: f
You entered wrong instance, please try again!
```

Delete:

Table: match

```
press 1 - Match...   press 2 - Stadium...   press 3 - StartDate...   press 4 - Team
1
What number of row do you want to delete? - 1
Enter value that marks Match_id: 460
Match_id: 460

1 Record deleted
```

418	976	296
460	221	619
486	88	228

418	976	296
486	88	228

```
press 1 - Match...   press 2 - Stadium...   press 3 - StartDate... press 4 - Team
1
What number of row do you want to delete? - 1
Enter value that marks Match_id:4
Match_id: 4

Failed deleting record into table ПОМИЛКА: update або delete в таблиці "match" порушує
```

обмеження зовнішнього ключа "StartDate_Match_id_fkey" таблиці "startdate"

DETAIL: На ключ (match_id)=(4) все ще є посилання в таблиці "startdate".

```
Enter value that marks Match_id:134565
Match_id: 134565
Failed, there are no records with such id: list index out of range
```

Table: stadium

```
press 1 - Match...   press 2 - Stadium...   press 3 - StartDate... press 4 - Team
2
What number of row do you want to delete? - 1
Enter value that marks Stadium_id:34
Stadium_id: 34

1 Record deleted
```

12	222f	dfd
34	fff	fff
54	fe	wr

12	222f	dfd
54	fe	wr

```
Enter value that marks Stadium_id:3
Stadium_id: 3
```

Failed deleting record into table ПОМИЛКА: update або delete в таблиці "stadium" порушує

порушує обмеження зовнішнього ключа "Team_Stadium_id_fkey" таблиці "team"

DETAIL: На ключ (stadium_id)=(3) все ще є посилання в таблиці "team".

```
Enter value that marks Stadium_id:4674337
Stadium_id: 4674337
Failed, there are no records with such id: list index out of range
```

Table: startdate

```
Enter values following this sequence team_id, match_id:
team_id: 5
match_id: 5

1 Record deleted
```

3	4	2000-02-01
5	5	2020-09-25
7	3	2019-12-12

3	4	2000-02-01
7	3	2019-12-12

```
Enter values following this sequence team_id, match_id:
team_id: 5
match_id: 1353
Failed, there are no records with such id: list index out of range
```

```
Enter values following this sequence team_id, match_id:
team_id: 5
match_id: 4

0 Record deleted
```

Table: team

```
press 1 - Match... press 2 - Stadium... press 3 - StartDate... press 4 - Team
4
What number of row do you want to delete? - 1
Enter value that mark Team_id:432
Team_id: 432

1 Record deleted
```

428	SLWIQS	YQCXPA	FTLUVO	5
432	gg	gg	gg	1
434	ff	ff	ff	1

428	SLWIQS	YQCXPA	FTLUVO	5
434	ff	ff	ff	1

```
Enter value that mark Team_id:3
Team_id: 3

Failed deleting record into table ПОМИЛКА: update або delete в таблиці "team" порушує
```

```
обмеження зовнішнього ключа "StartDate_Team_id_fkey" таблиці "startdate"
```

```
DETAIL: На ключ (team_id)=(3) все ще є посилання в таблиці "startdate".
```

```
Enter value that mark Team_id:336787764
Team_id: 336787764
Failed, there are no records with such id: list index out of range
```

```
Enter value that mark Team_id:
Team_id:
Error that column cannot contain NULL values!
```

Завдання 2:

```

Choose option:
PRESS: 1 to add... 2 to update... 3 to delete... 4 to specific_select... 5 to show_table...
1
Choose table:
  press 1 - Match...  press 2 - Stadium...  press 3 - StartDate... press 4 - Team
4
Do ypo want to insert rows manually or randomly?
Press M if manually, R if randomly: R
What number of row do you want to add(type 'd' to set default)? - 100000
100000 Record successfully inserted

```

```

2072055    639    2000-03-08
2072056     1    2001-01-23
2072059    19    2001-01-03
2072059   544    2001-06-26
2072062   614    2001-06-01
2072063   990    2001-06-14
2072067    16    2000-08-15
2072067   954    2000-01-24
2072067   990    2000-07-18
2072068    87    2000-10-08
2072068   331    2000-04-03
2072070    17    2000-12-20
2072070    35    2000-10-12
2072070   896    2001-04-08
2072072   544    2001-06-12
2072072   568    2000-04-12
2072074   956    2001-03-16
2072076     5    2000-09-24
2072076   990    2000-06-28
2072078    87    2001-07-09
2072082    21    2000-04-08
2072082  3433    2001-06-11
2072086    33    2000-12-06
2072088     1    2001-06-20
2072088   486    2000-04-11
2072088   805    2000-12-08

```

```
1  Select count(*) from startdate
```

Data Output Explain Messages Notificati

	count bigint	
1	100008	

1	INSERT INTO startdate (team_id, match_id, start_date)
2	SELECT team_id, match_id, dat FROM
3	(SELECT
4	team_id, match_id,
5	date '2000-01-10' + trunc(random()*10 * (date '2020-05-20' - date '2010-01-10'))::int as dat
6	FROM team, match tablesample BERNOULLI(100)
7	ORDER BY random()) k ,generate_series(1, 10000) LIMIT 5
8	Returning team_id, match_id, start_date

Data Output	Explain	Messages	Notifications
team_id [PK] integer	match_id [PK] integer	start_date date	
1	2032770	805	2000-01-10
2	2081946	544	2000-01-10
3	2019369	87	2000-01-10
4	2087644	30	2000-01-10
5	2094100	212	2000-01-10

```
Choose option:
PRESS: 1 to add... 2 to update... 3 to delete... 4 to specific_select... 5 to show_table...
1
Choose table:
press 1 - Match... press 2 - Stadium... press 3 - StartDate... press 4 - Team
3
Do ypo want to insert rows manually or randomly?
Press M if manually, R if randomly: R
What number of row do you want to add(type 'd' to set default)? - 100000
100000 Record successfully inserted
```

2100434	3c916c9d3d757505d23b4db22a4817d1	6497ff75a0c3ff3d5fa92de05f46deb4	7e5b2d8d00c7928887ca9107851688e9	6
2100435	f22e515c478ecda32db7420e899c29f3	1cd5c3a92626f5ef13f97904f84017dc	49cfdcd6a40ba7556115e8d5df9811f	7
2100436	f1d93ed604917d2f1b6b9164e2478f21	35ab70d0afa1bd86cca908d6fff8df99	29776b94b076a56dfb4a1324ec169e27	12
2100437	31fadfc59fe2b1a9550b4dd88a1b8553	62adf6e97b3ffa57b78d3a8e26ed34da	3c2af9ce834809a79e0d4f388c2c57e0	54
2100438	c8fb751cbf2e144aa21e10495c193a77	258465016040e27147f6fffec0358fa4	fdaaa303207741b0a4ac2293a531fa5e	5
2100439	0ad187dfba76e5269d13390b1fbac9c0	652d404815dc97d098d1253a107c2a88	420b8d08e8488bfb25c27b5e98d37ab	445
2100440	f0ae990fd44d65601d3bb555c0d553fc	82c66077ed55c5e2acbbf50d9b9dec0	67d9222506f84c7eed420975bd3686f8	2
2100441	a645cba1592f1ba80a6011f678b5334e	ea1b93d7ca2e170546c2f618c9ba10f4	ea32fffd0d9f312c6f1156477a012008	8
2100442	5a6d81773ec721807a44b107a33256c1	082a7339cea645898b799b92e57774d3	6e762f03bc9a0a331ae6c5b714c3d55c	1
2100443	b1055311f4ec77c953061e0525a0a0bb	dafedda8ffe0795cb9cdb459fc388304	ace03b27506423df4e3ea8a730a604a6	9
2100444	b19e2a6493f057ca251b748439242c7c	939c987992c43c228a6f3191eadc120e	3bd0ae83be59d2d0eec4e30864e11739	3
2100445	26ca64ae01aa9b9b65e22ddc1f9c25f8	9a9b29a22280bb1407fe0ee8ae3b39c8	e9cd1bd0c6483cc8a355da328e868d21	3234
2100446	a98113f5749cdf9ba1f1d55766ba1cc8	f1ed6cc589a6b8daa1fff9dd0fded8dd	ea39488dc8a30b2544cc0bb1a640577e	4
2100447	3c916c9d3d757505d23b4db22a4817d1	6497ff75a0c3ff3d5fa92de05f46deb4	7e5b2d8d00c7928887ca9107851688e9	6
2100448	f22e515c478ecda32db7420e899c29f3	1cd5c3a92626f5ef13f97904f84017dc	49cfdcd6a40ba7556115e8d5df9811f	7
2100449	f1d93ed604917d2f1b6b9164e2478f21	35ab70d0afa1bd86cca908d6fff8df99	29776b94b076a56dfb4a1324ec169e27	12
2100450	31fadfc59fe2b1a9550b4dd88a1b8553	62adf6e97b3ffa57b78d3a8e26ed34da	3c2af9ce834809a79e0d4f388c2c57e0	54
2100451	c8fb751cbf2e144aa21e10495c193a77	258465016040e27147f6fffec0358fa4	fdaaa303207741b0a4ac2293a531fa5e	5
2100452	0ad187dfba76e5269d13390b1fbac9c0	652d404815dc97d098d1253a107c2a88	420b8d08e8488bfb25c27b5e98d37ab	445
2100453	f0ae990fd44d65601d3bb555c0d553fc	82c66077ed55c5e2acbbf50d9b9dec0	67d9222506f84c7eed420975bd3686f8	2
2100454	a645cba1592f1ba80a6011f678b5334e	ea1b93d7ca2e170546c2f618c9ba10f4	ea32fffd0d9f312c6f1156477a012008	8
2100455	5a6d81773ec721807a44b107a33256c1	082a7339cea645898b799b92e57774d3	6e762f03bc9a0a331ae6c5b714c3d55c	1
2100456	b1055311f4ec77c953061e0525a0a0bb	dafedda8ffe0795cb9cdb459fc388304	ace03b27506423df4e3ea8a730a604a6	9
2100457	b19e2a6493f057ca251b748439242c7c	939c987992c43c228a6f3191eadc120e	3bd0ae83be59d2d0eec4e30864e11739	3
2100458	26ca64ae01aa9b9b65e22ddc1f9c25f8	9a9b29a22280bb1407fe0ee8ae3b39c8	e9cd1bd0c6483cc8a355da328e868d21	3234

```
1 Select count(*) from team
```

Data Output Explain Messages Notifications

	count bigint	
1	100053	

```
1 INSERT INTO team (name, opponent, coach, stadium_id)
2     SELECT nam, opp, coach, stadium_id FROM
3     (SELECT md5((random()*1)::text) as nam,
4           md5((random()*2)::text) as opp,
5           md5((random()*3)::text) as coach,
6           stadium_id
7     FROM stadium tablesample BERNOULLI(100)
8     ORDER BY random()) k, generate_series(1, 10000) LIMIT 5
9 Returning team_id ,name, opponent, coach, stadium_id
```

Data Output Explain Messages Notifications

	team_id [PK] integer	name character varying (50)	opponent character varying (50)	coach character varying (50)	stadium_id integer
1	2100474	c487fefe98be7de3e12e854...	2f2f4648fcd1e0d13e3bd698...	116e8a42b5d0b50533dcca...	6
2	2100475	5a2321708d56fd002907bc1...	8e072eb3b5d92e7f2dc1ceff...	60c70d1d82e983385bd24d...	54
3	2100476	acc7a2fdca12c1ad695497...	d6af7d47540eb9d1cad3cc7...	f3d473b72d82ff553735cc6a...	445
4	2100477	5be8b5154f708795053bf6a...	e34a6d3c96519136e0ec8bd...	55214af4bd0e969190a0311...	2
5	2100478	6427ab105780510fc393ff6...	b41c45397aaaf7040526b5d...	6a098a260b1c953535c2c6e...	1

```
sql_random_query = "" INSERT INTO match (opp_score, own_score)
SELECT
    trunc(random()*1000)::int,
    trunc(random()*1000)::int
FROM
    generate_series(1, %s)
""
```

```
sql_random_query = "" INSERT INTO stadium (name, city)
SELECT
    chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int) ||
    chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int) ||
    chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int),
    chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int) ||
    chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int) ||
    chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int)
FROM
    generate_series(1, %s)
""
```



```
sql_random_query = """ INSERT INTO startdate (team_id, match_id, start_date)
SELECT team_id, match_id, dat FROM
(SELECT
team_id, match_id,
date '2000-01-10' + trunc(random()*10 * (date '2020-05-20' - date '2010-01-10'))::int as dat
FROM team, match tablesample BERNOULLI(100)
ORDER BY random()) k ,generate_series(1, 10000) LIMIT %s
"""
```

```
sql_random_query = """ INSERT INTO team (name, opponent, coach, stadium_id)
SELECT nam, opp, coach, stadium_id FROM
(SELECT md5((random()*1)::text) as nam,
md5((random()*2)::text) as opp,
md5((random()*3)::text) as coach,
stadium_id
FROM stadium tablesample BERNOULLI(100)
ORDER BY random()) k, generate_series(1, 10000) LIMIT %s
"""
```

Завдання 3:

1.

```
sql_specific_query = """ SELECT team_id, name as team_name, coach, opponent, stadium_name
FROM team AS a INNER JOIN
(SELECT name AS stadium_name, stadium_id FROM stadium WHERE name LIKE %s) AS aa
ON a.stadium_id=aa.stadium_id
WHERE opponent != %s AND team_id in (SELECT team_id FROM startdate WHERE start_date > %s)
"""
```

2.

```
sql_specific_query = """ WITH t_n AS (SELECT team_id FROM team WHERE length(name) < %s
AND name NOT SIMILAR TO %s)
SELECT start_date, opp_score, own_score FROM startdate AS a INNER JOIN
(SELECT match_id, opp_score, own_score FROM
match WHERE opp_score = own_score OR opp_score - own_score = %s) as aa
ON a.match_id=aa.match_id
WHERE team_id IN (SELECT team_id FROM t_n)
"""
```

3.

```
sql_specific_query = """ SELECT tid AS team_id, sum(cnt) sum_team, cc.n AS team_name,
cc.o AS opponent_name FROM (
SELECT team_id tid, count(team_id) cnt, n, o FROM startdate INNER JOIN
(SELECT team_id, name AS n, opponent AS o FROM team WHERE name LIKE %s) AS t USING (team_id)
WHERE start_date BETWEEN %s AND %s
GROUP BY team_id, t.n, t.o) cc
WHERE tid < %s
GROUP BY tid, cc.n, cc.o
ORDER BY tid
"""
```

Choose option:
 PRESS: 1 to add... 2 to update... 3 to delete... 4 to specific_select... 5 to show_table...

4

1) Show all info about team(name, coach, opponent), which played game on specific 'stadium', but not with that 'opponent', when matches took place later than that 'date'.

2) Show date, opp_score, own_score from those teams that have a draw or the difference between home and away teams in 'number of goal' and the length of home team is less then 'that number', except of 'such team'.

3) Show team_id, sum_team, team_name, opponent_name from those teams that have 'specific name', where match took place between 'date_1' and 'date_2' and team_id is less than 'that number'

Choose table:
 Press: 1 or 2 or 3

1

Enter values following this sequence stadium_name, opponent, start_date:
 stadium_name: %
 opponent: Worskla
 start_date: 2018-01-01

Time of request 25 ms

Show each row and it's columns values

team_id	team_name	coach	opponent	stadium_name
7	Brosław	Lortus	Corole	Karpaty
1	Shakhtar	Mokerc A.V.	Beshekt	Olimp
2	Dynamo	Bytuchuk B.L.	Hover	Datum

```

1 select team_id, name as team_name, coach, opponent, stadium_name from team as a inner join
2 (select name as stadium_name, stadium_id from stadium where name LIKE '%') as aa
3 on a.stadium_id=aa.stadium_id
4 where opponent != 'Worskla' and team_id in (Select team_id from startdate where start_date > '2018-01-01')
5

```

Data Output	Explain	Messages	Notifications			
<div><div></div><div>team_id</div><div>integer</div></div> <div><div></div><div>team_name</div><div>character varying (50)</div></div> <div><div></div><div>coach</div><div>character varying (50)</div></div> <div><div></div><div>opponent</div><div>character varying (50)</div></div> <div><div></div><div>stadium_name</div><div>character varying (50)</div></div> <div><div></div></div>						
1	7	Brosław	Lortus	Corole	Karpaty	
2	1	Shakhtar	Mokerc A.V.	Beshekt	Olimp	
3	2	Dynamo	Bytuchuk B.L.	Hover	Datum	

```

1 EXPLAIN ANALYZE
2 select team_id, name as team_name, coach, opponent, stadium_name from team as a inner join
3 (select name as stadium_name, stadium_id from stadium where name LIKE '%') as aa
4 on a.stadium_id=aa.stadium_id
5 where opponent != 'Worskla' and team_id in (Select team_id from startdate where start_date > '2018-01-01')
6

```

Data Output	Explain	Messages	Notifications
	<div><div>QUERY PLAN</div><div>text</div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div>		

1

Enter values following this sequence stadium_name, opponent, start_date:
stadium_name: *Karpaty*
opponent: *Worskla*
start_date: *2018-01-01*

Time of request 20 ms

Show each row and it's columns values

team_id	team_name	coach	opponent	stadium_name
7	Broslaw	Lortus	Corole	Karpaty

```
1 --EXPLAIN ANALYZE
2 select team_id, name as team_name, coach, opponent, stadium_name from team as a inner join
3 (select name as stadium_name, stadium_id from stadium where name LIKE 'Karpaty') as aa
4 on a.stadium_id=aa.stadium_id
5 where opponent != 'Worskla' and team_id in (Select team_id from startdate where start_date > '2018-01-01')
```

Data Output Explain Messages Notifications

	team_id integer	team_name character varying (50)	coach character varying (50)	opponent character varying (50)	stadium_name character varying (50)	
1	7	Broslaw	Lortus	Corole	Karpaty	

```
1 EXPLAIN ANALYZE
2 select team_id, name as team_name, coach, opponent, stadium_name from team as a inner join
3 (select name as stadium_name, stadium_id from stadium where name LIKE 'Karpaty') as aa
4 on a.stadium_id=aa.stadium_id
5 where opponent != 'Worskla' and team_id in (Select team_id from startdate where start_date > '2018-01-01')
```

Data Output Explain Messages Notifications

	QUERY PLAN	
	text	
5	-> Seq Scan on startdate (cost=0.00..1791.10 rows=10 width=4) (actual time=0.038..16.183 rows=5 loops=1)	
6	Filter: (start_date > '2018-01-01'::date)	
7	Rows Removed by Filter: 100008	
8	-> Index Scan using "Team_pkey" on team a (cost=0.42..8.44 rows=1 width=104) (actual time=0.012..0.012 rows=1 loops=3)	
9	Index Cond: (team_id = startdate.team_id)	
10	Filter: ((opponent)::text <> 'Worskla'::text)	
11	-> Index Scan using "Stadium_pkey" on stadium (cost=0.15..0.17 rows=1 width=122) (actual time=0.005..0.005 rows=0 loops=3)	
12	Index Cond: (stadium_id = a.stadium_id)	
13	Filter: ((name)::text ~~ 'Karpaty'::text)	
14	Rows Removed by Filter: 1	
15	Planning Time: 0.617 ms	
16	Execution Time: 16.337 ms	

2

Enter values following this sequence len_name, team_name, diff_score:

len_name: 8

team_name: Dolor

diff_score: 2

Time of request 218 ms

Show each row and it's columns values

start_date own_score opp_score

2019-12-12 2 2

2018-09-12 2 2

2000-01-18 1 1

2000-07-02 2 2

2000-07-13 3 3

2000-08-30 1 1

2000-12-26 3 3

2001-04-26 2 2

```
1 with t2 as (select team_id from team where length(name) < 8 and name not similar to 'Dehoj')
2 select start_date, own_score, opp_score from startdate as a inner join
3 (select match_id, opp_score, own_score from match where opp_score = own_score or opp_score - own_score = 2) as aa
4 on a.match_id=aa.match_id
5 where team_id in (Select team_id from t2)
```

Data Output Explain Messages Notifications

	start_date date	own_score integer	opp_score integer
1	2019-12-12	2	2
2	2018-09-12	2	2
3	2000-01-18	1	1
4	2000-07-02	2	2
5	2000-07-13	3	3
6	2000-08-30	1	1
7	2000-12-26	3	3
8	2001-04-26	2	2

```
1 EXPLAIN ANALYZE
2 with t2 as (select team_id from team where length(name) < 8 and name not similar to 'Dolor')
3 select start_date from startdate as a inner join
4 (select match_id, opp_score, own_score from match where opp_score = own_score or opp_score - own_score = 2) as aa
5 on a.match_id=aa.match_id
6 where team_id in (Select team_id from t2)
```

Data Output Explain Messages Notifications

	QUERY PLAN
4	text
4	-> Seq Scan on startdate a (cost=0.00..1341.08 rows=100008 width=12) (actual time=0.038..11.042 rows=100013 loops=1)
5	-> Hash (cost=1.91..1.91 rows=1 width=4) (actual time=0.044..0.045 rows=9 loops=1)
6	Buckets: 1024 Batches: 1 Memory Usage: 9kB
7	-> Seq Scan on match (cost=0.00..1.91 rows=1 width=4) (actual time=0.019..0.037 rows=9 loops=1)
8	Filter: ((opp_score = own_score) OR ((opp_score - own_score) = 2))
9	Rows Removed by Filter: 54
10	-> Index Scan using "Team_pkey" on team (cost=0.42..0.74 rows=1 width=4) (actual time=0.005..0.005 rows=0 loops=14425)
11	Index Cond: (team_id = a.team_id)
12	Filter: (((name)::text !~ "^(?:Dolor)\$::text) AND (length((name)::text) < 8))
13	Rows Removed by Filter: 1
14	Planning Time: 0.743 ms
15	Execution Time: 115.709 ms

2

Enter values following this sequence len_name, team_name, diff_score:

len_name: 100

team_name: Dynamo

diff_score: 1

Time of request 198 ms

Show each row and it's columns values

start_date own_score opp_score

2020-09-15 2 2

2020-12-12 2 1

2019-12-12 2 2

2000-02-01 1 0

2000-01-10 4 3

2000-01-10 4 3

2000-01-10 2 1

2000-01-10 4 4

2000-01-10 2 1

2000-01-10 2 1

```
1 --EXPLAIN ANALYZE
2 with t2 as (select team_id from team where length(name) < 100 and name not similar to 'Dynamo')
3 select start_date, own_score, opp_score from startdate as a inner join
4 (select match_id, opp_score, own_score from match where opp_score = own_score or opp_score - own_score = 1) as aa
5 on a.match_id=aa.match_id
6 where team_id in (Select team_id from t2)
```

Data Output Explain Messages Notifications

	start_date date	own_score integer	opp_score integer	
1	2020-09-15	2	2	
2	2020-12-12	1	2	
3	2019-12-12	2	2	
4	2000-02-01	0	1	
5	2000-01-10	3	4	
6	2000-01-10	3	4	
7	2000-01-10	1	2	
8	2000-01-10	4	4	
9	2000-01-10	1	2	
10	2000-01-10	1	2	

```

1 EXPLAIN ANALYZE
2 with t2 as (select team_id from team where length(name) < 100 and name not similar to 'Dynamo')
3 select start_date from startdate as a inner join
4 (select match_id, opp_score, own_score from match where opp_score = own_score or opp_score - own_score = 1) as aa
5 on a.match_id=aa.match_id
6 where team_id in (Select team_id from t2)

```

Data Output Explain Messages Notifications

	QUERY PLAN	
	text	
4	-> Seq Scan on startdate a (cost=0.00..1541.08 rows=100008 width=12) (actual time=0.039..15.130 rows=100013 loops=1)	
5	-> Hash (cost=1.91..1.91 rows=1 width=4) (actual time=0.051..0.052 rows=11 loops=1)	
6	Buckets: 1024 Batches: 1 Memory Usage: 9kB	
7	-> Seq Scan on match (cost=0.00..1.91 rows=1 width=4) (actual time=0.023..0.042 rows=11 loops=1)	
8	Filter: ((opp_score = own_score) OR ((opp_score - own_score) = 1))	
9	Rows Removed by Filter: 52	
10	-> Index Scan using "Team_pkey" on team (cost=0.42..0.74 rows=1 width=4) (actual time=0.006..0.006 rows=1 loops=17565)	
11	Index Cond: (team_id = a.team_id)	
12	Filter: (((name)::text !~ '^(?Dynamo)\$::text) AND (length((name)::text) < 100))	
13	Rows Removed by Filter: 0	
14	Planning Time: 0.890 ms	
15	Execution Time: 174.480 ms	

3

Enter values following this sequence team_name, date_start, date_finish, team_id:
team_name: %
date_start: 2015-01-01
date_finish: 2020-07-11
team_id: 15

Time of request 7 ms

Show each row and it's columns values

team_id	sum_team	team_name	opponent_name
2	1	Dynamo	Hover
7	1	Broslaw	Corole

```

1 select tid as team_id, sum(cnt) sum_team, cc.n as team_name, cc.o as opponent_name from (
2 select team_id tid, count(team_id) cnt, n, o from startdate inner join (
3 Select team_id, name as n, opponent as o from team where name LIKE '%') as t
4 using (team_id)
5 where start_date between '2015-01-01' and '2020-07-11'
6 group by team_id, t.n, t.o
7 ) cc
8 where tid < 15
9 group by tid, cc.n, cc.o
10 order by tid

```

Data Output Explain Messages Notifications

	team_id	sum_team	team_name	opponent_name	
	integer	numeric	character varying (50)	character varying (50)	
1	2	1	Dynamo	Hover	
2	7	1	Broslaw	Corole	

```

1 EXPLAIN ANALYZE
2 select tid as team_id, sum(cnt) sum_team, cc.n as team_name, cc.o as opponent_name from (
3 select team_id tid, count(team_id) cnt, n, o from startdate inner join (
4     select team_id, name as n, opponent as o from team where name LIKE '%') as t
5     using (team_id)
6     where start_date between '2016-01-01' and '2020-07-11'
7     group by team_id, t.n, t.o
8 ) cc
9 where tid < 15
10 group by tid, cc.n, cc.o
11 order by tid

```

Data Output Explain Messages Notifications

QUERY PLAN		
	text	
10	Index Cond: (team_id < 15)	
11	Filter: ((start_date >= '2016-01-01'::date) AND (start_date <= '2020-07-11'::date))	
12	Rows Removed by Filter: 13	
13	-> Index Scan using "Team_pkey" on team (cost=0.42..8.44 rows=1 width=68) (actual time=0.012..0.012 rows=1 loops=2)	
14	Index Cond: (team_id = startdate.team_id)	
15	Filter: ((name)::text ~~ '%':text)	
16	Planning Time: 0.679 ms	
17	Execution Time: 0.222 ms	

```

3
Enter values following this sequence team_name, date_start, date_finish, team_id:
team_name: Dynamo
date_start: 2010-02-02
date_finish: 2021-01-01
team_id: 100000

Time of request 2 ms
Show each row and it's columns values
team_id sum_team team_name opponent_name
2 2 Dynamo Hover

```

```

1 --EXPLAIN ANALYZE
2 select tid as team_id, sum(cnt) sum_team, cc.n as team_name, cc.o as opponent_name from (
3 select team_id tid, count(team_id) cnt, n, o from startdate inner join (
4     select team_id, name as n, opponent as o from team where name LIKE 'Dynamo') as t
5     using (team_id)
6     where start_date between '2016-01-01' and '2020-07-11'
7     group by team_id, t.n, t.o
8 ) cc
9 where tid < 15
10 group by tid, cc.n, cc.o
11 order by tid

```

Data Output Explain Messages Notifications

	team_id integer	sum_team numeric	team_name character varying (50)	opponent_name character varying (50)	
1	2	1	Dynamo	Hover	

```

1  EXPLAIN ANALYZE
2  select tid as team_id, sum(cnt) sum_team, cc.n as team_name, cc.o as opponent_name from (
3  select team_id tid, count(team_id) cnt, n, o from startdate inner join (
4      select team_id, name as n, opponent as o from team where name LIKE 'Dynamo') as t
5      using (team_id)
6      where start_date between '2016-01-01' and '2020-07-11'
7      group by team_id, t.n, t.o
8  ) cc
9  where tid < 15
10 group by tid, cc.n, cc.o
11 order by tid

```

Data Output Explain Messages Notifications

	QUERY PLAN	
	text	
11	Filter: ((start_date >= '2016-01-01'::date) AND (start_date <= '20...	
12	Rows Removed by Filter: 13	
13	-> Index Scan using "Team_pkey" on team (cost=0.42..8.44 rows=...	
14	Index Cond: (team_id = startdate.team_id)	
15	Filter: ((name)::text ~~ 'Dynamo':text)	
16	Rows Removed by Filter: 1	
17	Planning Time: 0.683 ms	
18	Execution Time: 0.169 ms	

Завдання 4:

master ▾
 DataBase / Database and management tools(Lab 2) / basic_backend.py / <> Jump to ▾
 Go to file ...

PointProgram Add files via upload
 Latest commit c01cf76 9 minutes ago History

1 contributor

4 lines (3 sloc) | 95 Bytes
 Raw Blame

```

1  from controller import Controller
2
3  if __name__ == "__main__":
4      Controller().input_get()

```

master ▾
 DataBase / Database and management tools(Lab 2) / model.py / <> Jump to ▾
 Go to file ...

PointProgram Add files via upload
 Latest commit c01cf76 10 minutes ago History

1 contributor

269 lines (248 sloc) | 11.2 KB
 Raw Blame

```

1  import psycopg2
2  import time
3
4  class Model:
5      #connection to PostgreSQL server
6      def __init__(self):
7          try:
8              self.connection = psycopg2.connect( user = "postgres",
9                                                  password = "postgres",
10                                                 host = "127.0.0.1",
11                                                 port = "5432",
12                                                 database = "Comand_Match" )
13              self.cursor = self.connection.cursor()
14          except (Exception, psycopg2.Error) as error:

```

```

15         print("Error while connecting to PostgreSQL", error)
16
17     #show data from tables
18     def showTable(self, sql_select_query, tab, record, bool):
19         try:
20             self.cursor = self.connection.cursor()
21             if bool:
22                 beg = int(time.time() * 1000)
23                 self.cursor.execute(sql_select_query, record)
24                 if bool:
25                     end = int(time.time() * 1000) - beg
26                     print("Time of request", end, " ms")
27                 records = self.cursor.fetchall()
28                 #print_table(records, tab)
29             except (Exception, psycopg2.Error) as error:
30                 print("Error while fetching data from PostgreSQL", error)
31                 self.connection.rollback()
32             finally:
33                 if self.connection:
34                     self.cursor.close()
35                 return records
36
37     #update table row
38     def updateTable(self, records, sql_update_query):
39         try:
40             cursor = self.connection.cursor()
41             cursor.executemany(sql_update_query, records)
42             self.connection.commit()
43             row_count = cursor.rowcount
44             print(row_count, "Records Updated")
45         except (Exception, psycopg2.Error) as error :
46             print("Failed updating record of the table {}", error)
47             self.connection.rollback()
48         finally:
49             if self.connection:
50                 cursor.close()
51
52     #add table row
53     def addTable(self, records, sql_insert_query):
54         try:
55             cursor = self.connection.cursor()
56             cursor.executemany(sql_insert_query, records)
57             self.connection.commit()
58             print(cursor.rowcount, "Record successfully inserted")
59         except (Exception, psycopg2.Error) as error :
60             print("Failed inserting record into table {}".format(error))
61             self.connection.rollback()
62         finally:
63             if self.connection:
64                 cursor.close()
65
66     #delete table row
67     def delTable(self, records, sql_delete_query):
68         try:
69             cursor = self.connection.cursor()
70             cursor.executemany(sql_delete_query, records)
71             self.connection.commit()
72             print(cursor.rowcount, "Record deleted")
73         except (Exception, psycopg2.Error) as error :
74             print("Failed deleting record into table {}".format(error))
75             self.connection.rollback()
76         finally:

```

```

76         finally:
77             if self.connection:
78                 cursor.close()
79
80     def only_possible(self, val, num):
81         try:
82             rec = [(val,)]
83             sql_origin = self.origin_type(num)
84             curs = self.connection.cursor()
85             curs.execute(sql_origin, rec)
86             records = curs.fetchall()
87             if records[0] == None:
88                 for row in records:
89                     print("There is row with id: ", row[0])
90         except (Exception, psycopg2.Error) as error:
91             print("Failed, there are no records with such id: {}".format(error))
92             self.if_exit()
93         finally:
94             if self.connection:
95                 curs.close()
96
97     #close connection of Postgre table
98     def close_connect(self):
99         if self.connection:
100             self.connection.close()
101             print("PostgreSQL connection is closed")
102
103     # choose command to insert data to specific table
104     def insert_type(self, table_num):
105         if table_num == '1':
106
107             def insert_type(self, table_num):
108                 if table_num == '1':
109                     sql_insert_query = """ INSERT INTO match (match_id, opp_score, own_score)
110                     VALUES (%s, %s, %s)"""
111                 elif table_num == '2':
112                     sql_insert_query = """ INSERT INTO stadium (stadium_id, name, city)
113                     VALUES (%s, %s, %s)"""
114                 elif table_num == '3':
115                     sql_insert_query = """ INSERT INTO startdate (team_id, match_id, start_date)
116                     VALUES (%s, %s, %s)"""
117                 elif table_num == '4':
118                     sql_insert_query = """ INSERT INTO team (team_id, name, opponent, coach, stadium_id)
119                     VALUES (%s, %s, %s, %s, %s)"""
120                 return sql_insert_query
121
122             # choose command to update data of specific table
123             def update_type(self, table_num):
124                 if table_num == '1':
125                     sql_update_query = """UPDATE match set opp_score = %s, own_score = %s
126                     WHERE "match_id" = %s """
127                 elif table_num == '2':
128                     sql_update_query = """UPDATE stadium set name = %s, city = %s
129                     WHERE "stadium_id" = %s """
130                 elif table_num == '3':
131                     sql_update_query = """UPDATE startdate set start_date = %s
132                     WHERE team_id = %s AND match_id = %s"""
133                 elif table_num == '4':
134                     sql_update_query = """UPDATE team set name = %s, opponent = %s, coach = %s, stadium_id = %s
135                     WHERE team_id = %s """
136                 return sql_update_query
137
138             # choose command to delete data of specific table
139             def delete_type(self, table_num):
140                 if table_num == '1':
141                     sql_delete_query = """ DELETE FROM match WHERE match_id = %s"""
142                 elif table_num == '2':
143                     sql_delete_query = """ DELETE FROM stadium WHERE stadium_id = %s"""
144                 elif table_num == '3':
145                     sql_delete_query = """ DELETE FROM startdate WHERE team_id = %s AND match_id = %s"""
146                 elif table_num == '4':
147                     sql_delete_query = """ DELETE FROM team WHERE team_id = %s"""
148                 return sql_delete_query
149
150             def select_type(self, table_num):
151                 if table_num == '1':
152                     sql_select_query = """ SELECT * FROM match ORDER BY match_id"""
153                 elif table_num == '2':
154                     sql_select_query = """ SELECT * FROM stadium ORDER BY stadium_id"""
155                 elif table_num == '3':
156                     sql_select_query = """ SELECT * FROM startdate ORDER BY team_id, match_id"""
157                 elif table_num == '4':
158                     sql_select_query = """ SELECT * FROM team ORDER BY team_id"""
159                 return sql_select_query
160
161             def origin_type(self, table_num):
162                 if table_num == 1:
163                     sql_origin_val = """ SELECT stadium.stadium_id FROM stadium WHERE stadium_id = %s """
164                 elif table_num == 2:
165                     sql_origin_val = """ SELECT team.team_id FROM team WHERE team_id = %s """
166                 elif table_num == 3:
167                     sql_origin_val = """ SELECT match.match_id FROM match WHERE match_id = %s """
168                 elif table_num == 4:

```



```

165         elif table_num == 4:
166             sql_origin_val = """ SELECT count(a) FROM (
167                 SELECT team_id, match_id FROM startdate
168                 WHERE team_id = %s AND match_id = %s) as a """
169         return sql_origin_val
170
171     def random_type(self, table_num):
172         if table_num == '1':
173             sql_random_query = """ INSERT INTO match (opp_score, own_score)
174             SELECT
175                 trunc(random()*1000)::int,
176                 trunc(random()*1000)::int
177             FROM
178                 generate_series(1, %s)
179             """
180         elif table_num == '2':
181             sql_random_query = """ INSERT INTO stadium (name, city)
182             SELECT
183                 chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int) ||
184                 chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int) ||
185                 chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int),
186                 chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int) ||
187                 chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int) ||
188                 chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int)
189             FROM
190                 generate_series(1, %s)
191             """
192         elif table_num == '3':
193             sql_random_query = """ INSERT INTO startdate (team_id, match_id, start_date)
194             SELECT team_id, match_id, dat FROM
195             (SELECT
196                 team_id, match_id,
197                 date '2000-01-10' + trunc(random()*10 * (date '2020-05-20' - date '2010-01-10'))::int as dat
198             FROM team, match tablesample BERNOULLI(100)
199             ORDER BY random()) k ,generate_series(1, 10000) LIMIT %s
200             """
201         elif table_num == '4':
202             sql_random_query = """ INSERT INTO team (name, opponent, coach, stadium_id)
203             SELECT nam, opp, coach, stadium_id FROM
204             (SELECT md5((random()*1)::text) as nam,
205                 md5((random()*2)::text) as opp,
206                 md5((random()*3)::text) as coach,
207                 stadium_id
208             FROM stadium tablesample BERNOULLI(100)
209             ORDER BY random()) k, generate_series(1, 10000) LIMIT %s
210             """
211         return sql_random_query
212
213     # def random_generator():
214
215     def specific_type(self, table_num):
216         if table_num == '1':
217             sql_specific_query = """ SELECT team_id, name as team_name, coach, opponent, stadium_name
218             FROM team AS a INNER JOIN
219             (SELECT name AS stadium_name, stadium_id FROM stadium WHERE name LIKE %s) AS aa
220             ON a.stadium_id=aa.stadium_id
221             WHERE opponent != %s AND team_id in (SELECT team_id FROM startdate WHERE start_date > %s)
222             """
223         elif table_num == '2':
224             sql_specific_query = """ WITH t_n AS (SELECT team_id FROM team WHERE length(name) < %s
225             AND name NOT SIMILAR TO %s)
226
227             sql_specific_query = """ WITH t_n AS (SELECT team_id FROM team WHERE length(name) < %s
228             AND name NOT SIMILAR TO %s)
229             SELECT start_date, opp_score, own_score FROM startdate AS a INNER JOIN
230             (SELECT match_id, opp_score, own_score FROM
231             match WHERE opp_score = own_score OR opp_score - own_score = %s) as aa
232             ON a.match_id=aa.match_id
233             WHERE team_id IN (SELECT team_id FROM t_n)
234             """
235         elif table_num == '3':
236             sql_specific_query = """ SELECT tid AS team_id, sum(cnt) sum_team, cc.n AS team_name,
237             cc.o AS opponent_name FROM (
238             SELECT team_id tid, count(team_id) cnt, n, o FROM startdate INNER JOIN
239             (SELECT team_id, name AS n, opponent AS o FROM team WHERE name LIKE %s) AS t USING (team_id)
240             WHERE start_date BETWEEN %s AND %s
241             GROUP BY team_id, t.n, t.o) cc
242             WHERE tid < %s
243             GROUP BY tid, cc.n, cc.o
244             ORDER BY tid
245             """
246         return sql_specific_query
247
248     # check if int is valid
249     def Repr_init(self, n):
250         try:
251             int(n)
252             return True
253         except ValueError:
254             return False
255
256     # check if void input
257     def catch_void(self, inp):
258         try:
259             .
260         
```

```

254     def catch_void(self, inp):
255         try:
256             inp
257             return "True"
258         except SyntaxError:
259             inp = None
260
261     def check_default(self, times):
262         if times == 'd':
263             times = 100000
264             return times
265         else:
266             return times
267

```



master ▾

DataBase / Database and management tools(Lab 2) / view.py / <> Jump to ▾

Go to file



PointProgram Add files via upload

Latest commit c01c7f6 11 minutes ago

History

1 contributor

36 lines (34 sloc) | 1.98 KB

Raw

Blame



```

1  class View:
2      def print_table(self, records, tab):
3          if tab == '1':
4              self.print_text("Match_id  opp_score  own_score ")
5              for sql_row in records:
6                  print("{:<10}".format(sql_row[0]), "{:<10}".format(sql_row[1]), "{:<10}".format(sql_row[2]))
7          elif tab == '2':
8              self.print_text("Stadium_id  Name  City ")
9              for sql_row in records:
10                 print("{:<10}".format(sql_row[0]), "{:<10}".format(sql_row[1]), "{:<10}".format(sql_row[2]))
11         elif tab == '3':
12             self.print_text("Team_id  Match_id  Start_date")
13             for sql_row in records:
14                 print("{:<10}".format(sql_row[0]), "{:<10}".format(sql_row[1]), " ", sql_row[2])
15         elif tab == '4':
16             self.print_text("Team_id  Name  Oppnent  Coach  Stadium_id")
17             for sql_row in records:
18                 print("{:<10}".format(sql_row[0]), "{:<10}".format(sql_row[1]), "{:<10}".format(sql_row[2]), "{:<15}".format(sql_row[3]), "{:<15}".format(sql_row[4]))
19         elif tab == '5':
20             self.print_text("team_id  team_name  coach  opponent  stadium_name ")
21             for sql_row in records:
22
23
24
25
26
27
28
29
30
31
32     def display(self, txt):
33         return input(txt)
34
35     def print_text(self, txt):
36         print(str(txt))

```



1 contributor

410 lines (389 sloc) | 17.5 KB

Raw

Blame



```
1 from model import Model
2 from view import View
3 import datetime
4 class Controller:
5     def __init__(self):
6         self.model = Model()
7         self.view = View()
8     # enter new data for insertion for specific table
9     def insert_query(self, opt):
10         times = self.quantity_put()
11         self.check_times(times)
12         record_to_insert = ""
13         if opt == '1':
14             for i in range(int(times)):
15                 self.view.print_text("Enter values following this sequence: Match_id, opp_score, own_score")
16                 m_id = self.view.display("Match_id: ")
17                 self.non_null_check(m_id, True)
18                 self.Repr_int(m_id)
19                 opp = self.view.display("opp_score: ")
20                 self.non_null_check(opp, True)
21                 self.Repr_int(opp)
22
23                 own = self.view.display("own_score: ")
24                 self.non_null_check(own, True)
25                 self.Repr_int(own)
26                 record_to_insert = [(m_id, opp, own)]
27         elif opt == '2':
28             for i in range(int(times)):
29                 self.view.print_text("Enter values following this sequence: Stadium_id, Name, City")
30                 s_id = self.view.display("Stadium_id: ")
31                 self.non_null_check(s_id, True)
32                 self.Repr_int(s_id)
33                 n = self.view.display("Name: ")
34                 self.non_null_check(n, True)
35                 self.Repr_char(n)
36                 c = self.view.display("City: ")
37                 self.non_null_check(c, True)
38                 self.Repr_char(c)
39                 record_to_insert = [(s_id, n, c)]
40         elif opt == '3':
41             for i in range(int(times)):
42                 self.view.print_text("Enter values following this sequence: Team_id, Match_id, Start_date(yyyy-mm-dd)")
43                 t_id = self.view.display("Team_id: ")
44                 self.non_null_check(t_id, True)
45                 self.Repr_int(t_id)
46                 self.model.only_possible(t_id, 2)
47                 m_id = self.view.display("Match_id: ")
48                 self.non_null_check(m_id, True)
49                 self.Repr_int(m_id)
50                 self.model.only_possible(m_id, 3)
51                 s_date = self.view.display("Start_date: ")
52                 self.non_null_check(s_date, True)
53
54                 self.non_null_check(s_date, True)
55                 self.Repr_date(s_date)
56                 record_to_insert = [(t_id, m_id, s_date)]
57         elif opt == '4':
58             for i in range(int(times)):
59                 self.view.print_text("Enter values following this sequence: Team_id, Name, Opponent, Coach, Stadium_id")
60                 team_id = self.view.display("Team_id: ")
61                 self.non_null_check(team_id, True)
62                 self.Repr_int(team_id)
63                 name = self.view.display("Name: ")
64                 self.non_null_check(name, True)
65                 self.Repr_char(name)
66                 opon = self.view.display("Opponent: ")
67                 self.non_null_check(opon, True)
68                 self.Repr_char(opon)
69                 coach = self.view.display("Coach: ")
70                 if self.non_null_check(coach, False):
71                     self.Repr_char(coach)
72                 st_id = self.view.display("Stadium_id: ")
73                 self.non_null_check(st_id, True)
74                 self.Repr_int(st_id)
75                 self.model.only_possible(st_id, 1)
76                 record_to_insert = [(team_id, name, opon, coach, st_id)]
77         return record_to_insert
78
79 # enter new data to update the specific table
80 def update_query(self, opt):
81     times = self.view.display("What number of rows do you want to update? - ")
82     self.check_times(times)
83     record_to_update = ""
84     if opt == '1':
```

```

81     if opt == '1':
82         for i in range(int(times)):
83             self.view.print_text("Enter values following this sequence opp_score, own_score, Match_id:")
84             opp = self.view.display("opp_score: ")
85             self.non_null_check(opp, True)
86             self.Repr_int(opp)
87             own = self.view.display("own_score: ")
88             self.non_null_check(own, True)
89             self.Repr_int(own)
90             m_id = self.view.display("Match_id: ")
91             self.non_null_check(m_id, True)
92             self.Repr_int(m_id)
93             self.model.only_possible(m_id, 3)
94             record_to_update = [(opp, own, m_id)]
95     elif opt == '2':
96         for i in range(int(times)):
97             self.view.print_text("Enter values following this sequence Name, City, Stadium_id:")
98             n = self.view.display("Name: ")
99             self.non_null_check(n, True)
100             self.Repr_char(n)
101             c = self.view.display("City: ")
102             self.non_null_check(c, True)
103             self.Repr_char(c)
104             s_id = self.view.display("Stadium_id: ")
105             self.non_null_check(s_id, True)
106             self.Repr_int(s_id)
107             self.model.only_possible(s_id, 1)
108             record_to_update = [(n, c, s_id)]
109     elif opt == '3':
110         for i in range(int(times)):
111             self.view.print_text("Enter values following this sequence Start_date(yyyy-mm-dd), Team_id, Match_id:")
112             s_date = self.view.display("Start_date: ")
113             self.Repr_date(s_date)
114             t_id = self.view.display("Team_id: ")
115             self.non_null_check(t_id, True)
116             self.Repr_int(t_id)
117             self.model.only_possible(t_id, 2)
118             m_id = self.view.display("Match_id: ")
119             self.non_null_check(m_id, True)
120             self.Repr_int(m_id)
121             self.model.only_possible(m_id, 3)
122             record_to_update = [(s_date, t_id, m_id)]
123     elif opt == '4':
124         for i in range(int(times)):
125             self.view.print_text("Enter values following this sequence Name, Opponent, Coach, Stadium_id, Team_id:")
126             name = self.view.display("Name: ")
127             self.non_null_check(name, True)
128             self.Repr_char(name)
129             opon = self.view.display("Opponent: ")
130             self.non_null_check(opon, True)
131             self.Repr_char(opon)
132             coach = self.view.display("Coach: ")
133             if self.non_null_check(coach, False):
134                 self.Repr_char(coach)
135             st_id = self.view.display("Stadium_id: ")
136             self.non_null_check(st_id, True)
137             self.Repr_int(st_id)
138             self.model.only_possible(st_id, 1)
139             team_id = self.view.display("Team_id: ")
140             self.non_null_check(team_id, True)
141             self.Repr_int(team_id)
142             self.model.only_possible(team_id, 2)
143             record_to_update = [(name, opon, coach, st_id, team_id)]
144     return record_to_update
145
146     def specific_query(self, self, opt):
147         if opt == '1':
148             self.view.print_text("Enter values following this sequence stadium_name, opponent, start_date:")
149             sn = self.view.display("stadium_name: ")
150             if self.non_null_check(sn, False):
151                 self.Repr_char(sn)
152             op = self.view.display("opponent: ")
153             if self.non_null_check(op, False):
154                 self.Repr_char(op)
155             sd = self.view.display("start_date: ")
156             self.Repr_date(sd)
157             record_to_specific = (sn, op, sd)
158     elif opt == '2':
159         self.view.print_text("Enter values following this sequence len_name, team_name, diff_score:")
160         ng = self.view.display("len_name: ")
161         self.non_null_check(ng, True)
162         self.Repr_int(ng)
163         lt = self.view.display("team_name: ")
164         if self.non_null_check(lt, False):
165             self.Repr_char(lt)
166         tn = self.view.display("diff_score: ")
167         self.non_null_check(tn, True)
168         self.Repr_int(tn)
169         record_to_specific = (ng, lt, tn)
170     elif opt == '3':
171         self.view.print_text("Enter values following this sequence team_name, date_start, date_finish, team_id:")

```

```

172         tn = self.view.display("team_name: ")
173         if self.non_null_check(tn, False):
174             self.Repr_char(tn)
175         ds = self.view.display("date_start: ")
176         self.Repr_date(ds)
177         df = self.view.display("date_finish: ")
178         self.Repr_date(df)
179         ti = self.view.display("team_id: ")
180         self.non_null_check(ti, True)
181         self.Repr_int(ti)
182         record_to_specific = (tn, ds, df, ti)
183         return record_to_specific
184
185     # enter data for deletion of specific table
186     def delete_query(self, opt):
187         times = self.view.display("What number of row do you want to delete? - ")
188         self.check_times(times)
189         record_to_delete = ""
190         if opt == '1':
191             for i in range(int(times)):
192                 m_id = self.view.display("Enter value that marks Match_id:")
193                 self.non_null_check(m_id, True)
194                 self.Repr_int(m_id)
195                 self.model.only_possible(m_id, 3)
196                 record_to_delete = [(m_id,)]
197             elif opt == '2':
198                 for i in range(int(times)):
199                     s_id = self.view.display("Enter value that marks Stadium_id:")
200                     self.non_null_check(s_id, True)
201                     self.Repr_int(s_id)
202
203                     self.Repr_int(s_id)
204                     self.model.only_possible(s_id, 1)
205                     record_to_delete = [(s_id,)]
206             elif opt == '3':
207                 for i in range(int(times)):
208                     self.view.print_text("Enter values following this sequence team_id, match_id:")
209                     t_id = self.view.display("team_id: ")
210                     self.non_null_check(t_id, True)
211                     self.Repr_int(t_id)
212                     self.model.only_possible(t_id, 2)
213                     m_id = self.view.display("match_id: ")
214                     self.non_null_check(m_id, True)
215                     self.Repr_int(m_id)
216                     self.model.only_possible(m_id, 3)
217                     record_to_delete = [(t_id, m_id)]
218             elif opt == '4':
219                 for i in range(int(times)):
220                     team_id = self.view.display("Enter value that mark Team_id:")
221                     self.non_null_check(team_id, True)
222                     self.Repr_int(team_id)
223                     self.model.only_possible(team_id, 2)
224                     record_to_delete = [(team_id,)]
225             return record_to_delete
226
227     # main menu function
228     def input_get(self):
229         opt = self.input_start()
230         self.check_option(opt)
231         if opt == "1":
232             table = self.table_choose()
233             if_rand = self.view.display("Do you want to insert rows manually or randomly? \nPress M if manually, R if randomly: ")

```

```

262         self.model.delTable(record, sql_delete)
263         if self.to_continue() is False:
264             break
265         self.if_exit()
266     elif opt == "4":
267         table = self.spec_choose()
268         tab = str(int(table) + 4)
269         record = self.specific_query(table)
270         sql_spec = self.model.specific_type(table)
271         show = self.model.showTable(sql_spec, tab, record, True)
272         self.view.print_table(show, tab)
273         self.if_exit()
274     elif opt == "5":
275         table = self.table_choose()
276         sql_select = self.model.select_type(table)
277         show = self.model.showTable(sql_select, table, None, False)
278         self.view.print_table(show, table)
279         self.if_exit()
280
281 # input option
282 def input_start(self):
283     x = input("""Choose option: \nPRESS: 1 to add... 2 to update... 3 to delete... 4 to specific_select... 5 to show_table...\n""")
284     return x
285
286 def quantity_put(self):
287     return input("What number of row do you want to add? - ")
288
289 # choose specific table
290 def table_choose(self):
291     table = input(
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323         self.to_random(if_rand)
324         if if_rand == 'M':
325             while True:
326                 sql_insert = self.model.insert_type(table)
327                 record = self.insert_query(table)
328                 self.model.addTable(record, sql_insert)
329                 if self.to_continue() is False:
330                     break
331             elif if_rand == 'R':
332                 times = self.view.display("What number of row do you want to add(type 'd' to set default)? - ")
333                 times = self.model.check_default(times)
334                 self.check_times(times)
335                 sql_random = self.model.random_type(table)
336                 self.model.addTable([(times,)], sql_random)
337             self.if_exit()
338         elif opt == "2":
339             table = self.table_choose()
340             while True:
341                 sql_update = self.model.update_type(table)
342                 record = self.update_query(table)
343                 self.model.updateTable(record, sql_update)
344                 if self.to_continue() is False:
345                     break
346             self.if_exit()
347
348         elif opt == "3":
349             table = self.table_choose()
350             while True:
351                 sql_delete = self.model.delete_type(table)
352                 record = self.delete_query(table)
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999

```

```

322         self.if_exit()
323     elif var_t is None or var_t == '' and boool is False:
324         return False
325     return True
326
327 # check option validation
328 def check_option(self, opti):
329     if opti != '1' and opti != '2' and opti != '3' and opti != '4' and opti != '5' or self.model.catch_void(opti) != "True":
330         print("You entered wrong character!")
331         self.if_exit()
332
333 # check table validation
334 def check_table(self, tab, bl):
335     if bl:
336         if tab != '1' and tab != '2' and tab != '3' and tab != '4' or self.model.catch_void(tab) != "True":
337             print("You entered wrong character!")
338             self.if_exit()
339         if bl is False:
340             if tab != '1' and tab != '2' and tab != '3' or self.model.catch_void(tab) != "True":
341                 print("You entered wrong character!")
342                 self.if_exit()
343
344 # ask if exit
345 def if_exit(self):
346     ext = input("Do you want to exit? Press M to go to the main menu, or E to exit: ")
347     if ext == 'M':
348         self.input_get()
349     elif ext == 'E':
350         self.model.close_connect()
351         exit()
352     else:
353         print("You entered wrong character...")
354         self.if_exit()
355
356 def Repr_date(self, date):
357     year = ""
358     month = ""
359     day = ""
360     if len(date) != 10 or date[4] != '-' and date[7] != '-':
361         print("You entered wrong date value, please try again!")
362         self.if_exit()
363     for i in (0, 1, 2, 3):
364         year += date[i]
365     for i in (5, 6):
366         month += date[i]
367     for i in (8, 9):
368         day += date[i]
369     try:
370         d = datetime.date(int(year), int(month), int(day))
371     except ValueError:
372         print("You entered wrong date value, please try again!")
373         self.if_exit()
374
375 def Repr_int(self, n):
376     try:
377         int(n)
378     except ValueError:
379         print("You entered wrong instance, please try again!")
380         self.if_exit()

```

```

375 def Repr_int(self, n):
376     try:
377         int(n)
378     except ValueError:
379         print("You entered wrong instance, please try again!")
380         self.if_exit()
381
382 def Repr_char(self, ch):
383     if len(ch) > 50:
384         print("You entered wrong value only 50 symbols allowed, please try again!")
385         self.if_exit()
386     try:
387         str(ch)
388     except ValueError:
389         print("You entered wrong value, please try again!")
390         self.if_exit()
391
392 # check if number for loop is valid
393 def check_times(self, times):
394     if self.model.Repr_init(times):
395         return True
396     else:
397         print("You entered wrong instance, please try again!")
398         self.if_exit()
399         return False
400
401 # ask if continue
402 def to_continue(self):
403     cont = input("Do you want to continue? Press Y if yes, N if no: ")
404     if cont == 'Y':
405         return True
406     elif cont == 'N':

```