

Информационная система поддерживает сайт любителей игры «Танки». Она обладает информацией о игроках, танках, которые у них есть и типах танков.

Типичными для информационной системы являются вопросы:

- Танки каких типов есть у игрока X?
- Танков каких типов больше всего у игроков?
- У кого больше всего тяжёлых танков.

### **Расширенное задание**

Информационная система также обладает информацией о сыгранных битвах и урону который нанесли и получили танки в них.

Реализуйте запросы:

- Топ 10 танков по нанесённому урону.
- Титул «Убойная гусеница » получает пользователь, который сыграл больше 5 битв и лидирует по показателю нанесённый урон/ полученный урон.
- Титул «Бешеный» получает пользователь, который атаковал больше всего других игроков.

### **Задание**

- Продемонстрировать выполнение запросов и хранимых процедур с помощью «таблично-запросного» API.
- Связать базу MySQL с LibreOffice Base и продемонстрировать выполнение запросов
- Продемонстрировать выполнение запросов и хранимых процедур с помощью ORM

# 1 Продемонстрировать выполнение запросов и хранимых процедур с помощью «таблично-запросного» API

Буду использовать python с библиотекой mysql.connector

```
pip install mysql-connector-python
from mysql.connector import connect, Error

try:
    con = connect(
        host = 'localhost',
        user='root',
        password='zrc8=:&Eq,Hd',
        db='mydb1',
        port = '3306'
    )
    print('Соединились')
except Error as e:
    print('Что-то не так')
    print(e)

cur = con.cursor()
```

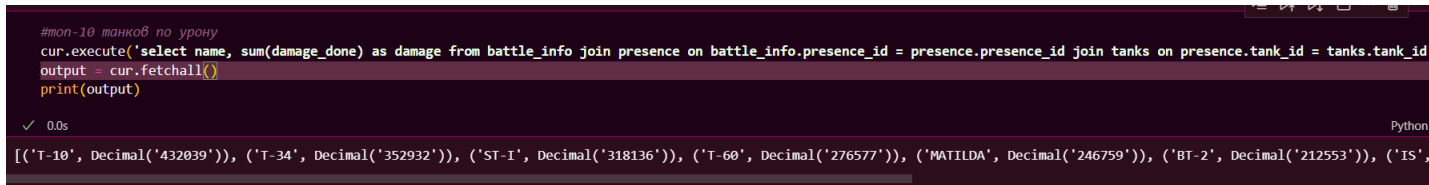
## Выведу всех игроков

```
cur.execute('SELECT * FROM players')
output = cur.fetchall()
print(output)
```

```
#Вывести всех игроков
cur.execute('SELECT * FROM players')
output = cur.fetchall()
print(output)
✓ 0.0s
[(1, 'Yellow Flash'), (2, 'ahsaS'), (3, 'hunterKiller228'), (4, 'SaIyAjIn'), (5, 'dmitriyBrekotkin'), (6, 'your_master'), (29, 'dark_master'), (30, 'new_one'), (31, 'new')]
```

## Топ-10 танков по урону

```
cur.execute('select name, sum(damage_done) as damage from battle_info  
join presence on battle_info.presence_id = presence.presence_id join tanks  
on presence.tank_id = tanks.tank_id group by  
name order by damage desc limit 10')  
output = cur.fetchall()  
print(output)
```



```
#top-10 танков по урону  
cur.execute('select name, sum(damage_done) as damage from battle_info join presence on battle_info.presence_id = presence.presence_id join tanks on presence.tank_id = tanks.tank_id  
output = cur.fetchall()  
print(output)  
✓ 0.0s  
Python  
[('T-10', Decimal('432039')), ('T-34', Decimal('352932')), ('ST-I', Decimal('318136')), ('T-60', Decimal('276577')), ('MATILDA', Decimal('246759')), ('BT-2', Decimal('212553')), ('IS',
```

## Процедура *UpdateNickname*

```
cur.callproc('UpdateNickname', ())  
cur.execute('SELECT * FROM players')  
output = cur.fetchall()  
print(output)
```

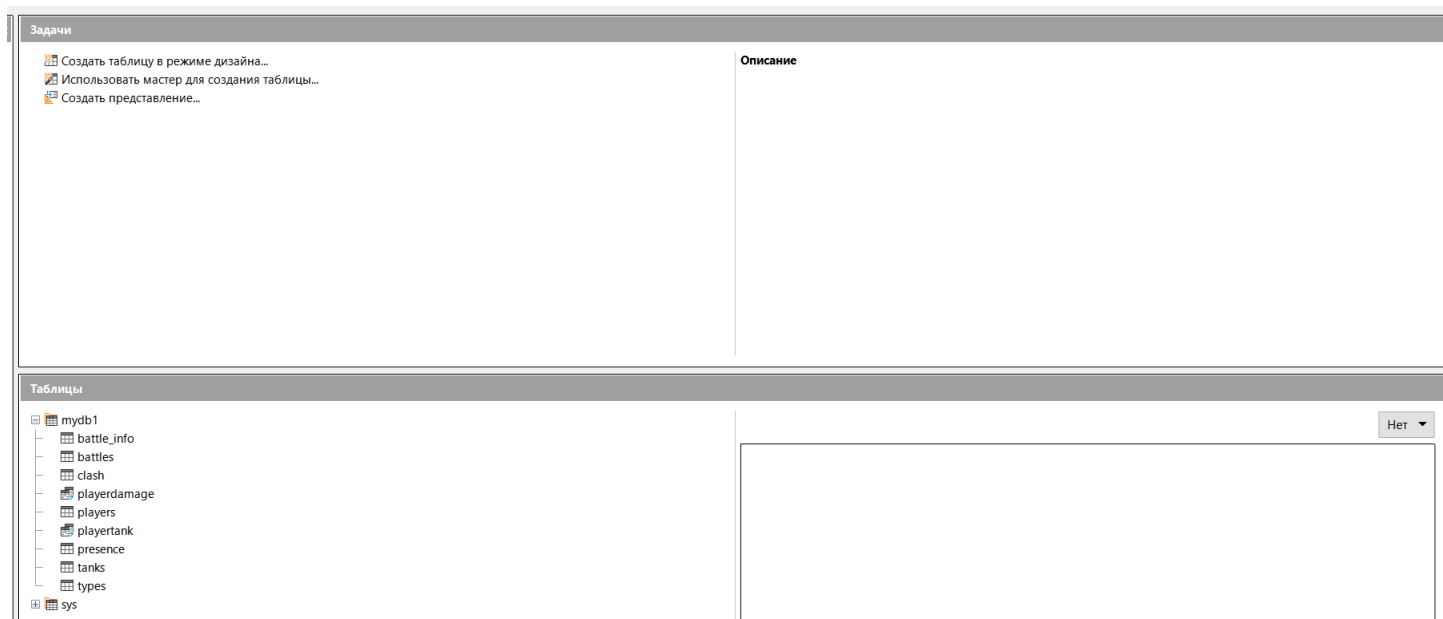


```
#вызов процедуры по изменению имени  
cur.callproc('UpdateNickname', ())  
✓ 0.0s  
()  
✓  
#вывести всех игроков  
cur.execute('SELECT * FROM players')  
output = cur.fetchall()  
print(output)  
✓ 0.0s  
[(1, 'Yellow Flash_12'), (2, 'ahsaS_5'), (3, 'hunterKiller228_15'), (4, 'SaIyAjIn_8'),
```

Всё сработало корректно.

## 2 Связать базу MySQL с LibreOffice Base и продемонстрировать выполнение запросов

Свяжу базу MySQL с LibreOffice Base



Напишу два запроса

```
SELECT * FROM `players`|
```

```
SELECT `name`, SUM(`damage_done`) AS `damage` FROM `battle_info` JOIN `presence` ON `battle_info`.`presence_id` = `presence`.`presence_id` JOIN `tanks` ON `presence`.`tank_id` = `tanks`.`tank_id` GROUP BY `name` ORDER BY `damage` DESC LIMIT 10|
```

## Результаты

	player_id	nickname
▶	1	Yellow Flash
	2	ahsaS
	3	hunterKiller22
	4	SalyAjln
	5	dmitriyBrekot
	6	your_master
	29	dark_master
	30	new_one
	31	new
+	<Автополе>	

	name	damage
▶	T-10	432039
	T-34	352932
	ST-I	318136
	T-60	276577
	MATILDA	246759
	BT-2	212553
	IS	207486
	T-26	200535
	BT-5	109883
	KV-5	103243

Добавлю новую строку в *tanks*

```
insert into tanks(name, type_id)
values
('Lab6', 3);
```

До

	tank_id	name	type_id
▶	1	BT-2	1
	2	T-26	1
	3	T-60	1
	4	BT-5	1
	5	T-116	1
	6	LTTB	1
	7	K-91	2
	8	IT-3	2
	9	T-34	2
	10	M4-85	2
	11	MATILDA	2
	12	A-44	2
	13	KV-1	3
	14	T-150	3
	15	IS	3
	16	KV-5	3
	17	ST-I	3
	18	T-10	3

После

	tank_id	name	type_id
▶	1	BT-2	1
	2	T-26	1
	3	T-60	1
	4	BT-5	1
	5	T-116	1
	6	LTTB	1
	7	K-91	2
	8	IT-3	2
	9	T-34	2
	10	M4-85	2
	11	MATILDA	2
	12	A-44	2
	13	KV-1	3
	14	T-150	3
	15	IS	3
	16	KV-5	3
	17	ST-I	3
	18	T-10	3
	19	Lab6	3

### 3 Продемонстрировать выполнение запросов и хранимых процедур с помощью ORM

В качестве ORM выбрал pony для python

```
from pony.orm import *
import datetime

db = Database()

class Types (db.Entity):
    type_id = PrimaryKey(int, auto = False)
    type = Required(str)
    tanks = Set('Tanks')

class Tanks (db.Entity):
    tank_id = PrimaryKey(int, auto = False)
    name = Required(str)
    type_id = Required('Types')
    presence = Set('Presence')

class Presence (db.Entity):
    presence_id = PrimaryKey(int, auto = False)
    player_id = Required('Players')
    tank_id = Required('Tanks')
    battle_info = Set('Battle_info')

class Battle_info (db.Entity):
    battle_info_id = PrimaryKey(int, auto = False)
    battle_id = Required('Battles')
    presence_id = Required('Presence')
    damage_done = Required(int)
    damage_received = Required(int)

class Battles (db.Entity):
    battle_id = PrimaryKey(int, auto = False)
    battle_date = Required(datetime.date)
    battle_info = Set('Battle_info')
    clash = Set('Clash')

class Clash (db.Entity):
    clash_id = PrimaryKey(int, auto = False)
    battle_id = Required('Battles')
    player_id = Required('Players')
    player_id2 = Required(int)

class Players (db.Entity):
    player_id = PrimaryKey(int, auto = False)
    nickname = Required(str)
    presence = Set('Presence')
```

```
clash = Set('Clash')
```

```
db.bind(provider = 'mysql', user = 'root', password = 'zrc8=:&Eq,Hd', host='localhost', database = 'mydb1',  
port = 3306)  
db.generate_mapping(create_tables=True)
```

```
with db_session:  
    #вывод таблицы players  
    players = select(p for p in Players)  
    for pl in players:  
        print(f'Player ID: {pl.player_id}, Nickname: {pl.nickname}')
```

print('\n')

```
    #добавление нового игрока  
    #new_player = Players(player_id=33, nickname = 'lab6.3')  
    #players = select(p for p in Players)  
    #for pl in players:  
        #print(f'Player ID: {pl.player_id}, Nickname: {pl.nickname}')
```

#print('\n')

```
    #удаление игрока  
    player_to_delete = Players.get(nickname='lab6.3')  
    player_to_delete.delete()  
    for pl in players:  
        print(f'Player ID: {pl.player_id}, Nickname: {pl.nickname}')
```

Выведу всех игроков

```
Player ID: 1, Nickname: Yellow Flash  
Player ID: 2, Nickname: ahsaS  
Player ID: 3, Nickname: hunterKiller228  
Player ID: 4, Nickname: SaIyAjIn  
Player ID: 5, Nickname: dmitriyBrekotkin  
Player ID: 6, Nickname: your_master  
Player ID: 29, Nickname: dark_master  
Player ID: 30, Nickname: new_one  
Player ID: 31, Nickname: new
```



Добавлю нового игрока, выведу таблицу, а потом удалю этого игрока

```
Player ID: 1, Nickname: Yellow Flash
Player ID: 2, Nickname: ahsaS
Player ID: 3, Nickname: hunterKiller228
Player ID: 4, Nickname: SaIyAjIn
Player ID: 5, Nickname: dmitriyBrekotkin
Player ID: 6, Nickname: your_master
Player ID: 29, Nickname: dark_master
Player ID: 30, Nickname: new_one
Player ID: 31, Nickname: new
Player ID: 34, Nickname: lab6.3
```

```
Player ID: 1, Nickname: Yellow Flash
Player ID: 2, Nickname: ahsaS
Player ID: 3, Nickname: hunterKiller228
Player ID: 4, Nickname: SaIyAjIn
Player ID: 5, Nickname: dmitriyBrekotkin
Player ID: 6, Nickname: your_master
Player ID: 29, Nickname: dark_master
Player ID: 30, Nickname: new_one
Player ID: 31, Nickname: new
```