# **--------------- Django ---------------**

# **Основы**

# **----------------- Jinja -----------------**

# **Подключение**

**pip install Jinja** скачивание етого сторонего модуля

**import jinja2** подключение етого модуля к текущему файлу

# **Основы**

**my\_template = Template(“text { { my\_value } } ” )** создание шаблона

**my\_template.render(my\_value = my\_var** или **my\_value = “text”)** подстановка значения в шаблон

# **---------------- SQLite ----------------**

# **Подключение**

**import sqlite3** подлючения модуля для работы с базами данных

**sqlite3.connect(“my\_db”)** подключение уже существующей базы данных (база лежит в однаковом каталоге). Если нету, тогда создаеться

**sqllite3.close()** закрытия базы данных после работы, прекащает работу базы

# **Типы Данных**

**NULL = NULL** значение NULL, не пустая строка

**INTEGER = 1 , 2 ,3** целое число

**REAL = 1.2 , 5.3 , 7.1** числа с плавающей точкой

**TEXT = “name”** строковый тип данных

**BLOB = “123.png”** для хранения изображений

# **Работа с Cursor()**

**my\_cur = slqlite.connect(“my\_db”). cursor( )** позволяет работать с базой данных через переменую **my\_cur.execute(“”” ”””)** действия с базой данных -> создание таблици , их изменения или запросы (делает только одно действие)

**my\_cur.execute (“””SELECT \* FROM my\_tab”””) my\_data = my\_cur.fetchall()** запись результата запроса в переменую (также строки таблции можно выводить через for x in cur: ). После fetchall нельзя использовать fetchone и fetchmany

**my\_cur.execute (“””SELECT \* FROM my\_tab”””) my\_data = my\_cur.fetchone()** записывает только первую строку таблици

**my\_cur.execute (“””SELECT \* FROM my\_tab”””) my\_data = my\_cur.fetchmany(3)** можно указывать количество строк из таблици для записи

**for x in my\_list: cur.execute(“INSERT INTO my\_tab VALUES (NULL , ? , ?)”, x )** заполнениея таблци с помощью цикла for и списка кортежей, который был заполнен до етого

**cur.executemany(“INSERT INTO my\_tab VALUES (NULL , ? , ?)”, my\_list )** функция переберает указаныи список кортжей и заполняет ими таблицу

**my\_cur.execute(“UPDATE my\_tab SET my\_col = : my\_value WHERE my\_col > 1”, {“my\_value”: 0 })** изменение значений строк в таблице

**my\_cur.execute(“INSERT INTO my\_tab VALUES (NULL,?)”, ( slqlite3Binary(my\_imd), ) )** запись изображения в строку базы

**my\_cur.executescript(“”” UPDATE; DELETE”””)** выполнение сразу нескольких действий

**my\_cur.lastrowid** возвращает id последнего елемента который добавляли в таблицу

**my\_cur.fetchall( )** возвращает все строки из базы данных (перед етим нужно прописывать запрос в cur.execute(“SELECT \* FROM my\_tab”) )

**my\_cur.fetchmany(3)** возвращает указаное количество строк из базы данных

**my\_cur.fetchone()** возвращает первую строку из базы данных

# **Работа с Самой Базой**

**my\_connect = slqlite3.connect(“test”)** подключение или создания базы данных на диске устройства

**my\_connect = sqlite.connect(“ :memory: ”)** подключение или создание базы данных в ОЗУ устройства

**my\_connect.commit( )** переносит зделаные изменения в базу данных

**my\_connect.rollback()** при возникновении ошибки откатывает изменения до написанной команды BEGIN;

**my\_connect.row\_factory = sqlite3.Row** теперь каждая строка в таблице ето обьект, разширивает функционал взаемодейсвия

**sqlite3.Binary(my\_img)** повзращает картинку в бинарном виде для записи в строку базы

**my\_con.iterdump()** возвращает всю таблицу в виде кода .sql (нужно проходиться циклом for)