



[Flask Templates](#) [Jinja2](#) [Flask-REST API](#) [Python SQLAlchemy](#) [Flask Bcrypt](#) [Flask Cookies](#) [Json](#) [Postman](#)

How to execute raw SQL in Flask-SQLAlchemy app

Last Updated : 19 Dec, 2021

In this article, we are going to see how to execute raw SQL in Flask-SQLAlchemy using Python.

Installing requirements

Install the Flask and Flask-SQLAlchemy libraries using pip

```
pip install Flask
pip install flask_sqlalchemy
```

Syntax

To run raw SQL queries, we first create a flask-SQLAlchemy engine object using which we can connect to the database and execute the SQL queries. The syntax is –

flask_sqlalchemy.SQLAlchemy.engine.execute(statement)

Executes a SQL expression construct or string statement within the current transaction.

Parameters:

- *statement: SQL expression*

Returns:

- *sqlalchemy.engine.result.ResultProxy*

Example 1

Python

```
# IMPORT REQUIRED LIBRARIES
from flask import Flask, request
from flask_sqlalchemy import SQLAlchemy

# CREATE THE FLASK APP
app = Flask(__name__)

# ADD THE DATABASE CONNECTION TO THE FLASK APP
db = SQLAlchemy(app)
db_cred = {
    'user': 'root',          # DATABASE USER
    'pass': 'password',     # DATABASE PASSWORD
    'host': '127.0.0.1',    # DATABASE HOSTNAME
    'name': 'Geeks4Geeks'   # DATABASE NAME
}
app.config['SQLALCHEMY_DATABASE_URI'] = f"mysql+pymysql://{db_cred['user']}:{db_cred['pass']}@{db_cred['host']}/{db_cred['name']}"
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False

# CREATE A users TABLE USING RAW SQL QUERY
db.engine.execute(
    '''
    CREATE TABLE users (
        email VARCHAR(50),
        first_name VARCHAR(50),
        last_name VARCHAR(50),
        passwd VARCHAR(50)
    );
    '''
)

# INSERT TEMP VALUES IN THE users TABLE USING RAW SQL QUERY
db.engine.execute(
    '''
    INSERT INTO users(email, first_name, last_name, passwd) VALUES
    ('john.doe@zmail.com', 'John', 'Doe', 'john@123');
    INSERT INTO users(email, first_name, last_name, passwd) VALUES
    ('john.doe@zmail.com', 'John', 'Doe', 'johndoe@777');
    INSERT INTO users(email, first_name, last_name, passwd) VALUES
    ('noah.emma@wmail.com', 'Emma', 'Noah', 'emaaa!00');
    INSERT INTO users(email, first_name, last_name, passwd) VALUES
    ('emma@tmail.com', 'Emma', 'Noah', 'whrfc2bfh904');
    INSERT INTO users(email, first_name, last_name, passwd) VALUES
    '''
)
```

```

('noah.emma@wmail.com', 'Emma', 'Noah', 'emaaa!00');
INSERT INTO users(email, first_name, last_name, passwd) VALUES
('liam.olivia@wmail.com', 'Liam', 'Olivia', 'lolivia#900');
INSERT INTO users(email, first_name, last_name, passwd) VALUES
('liam.olivia@wmail.com', 'Liam', 'Olivia', 'lolivia$345');
'''

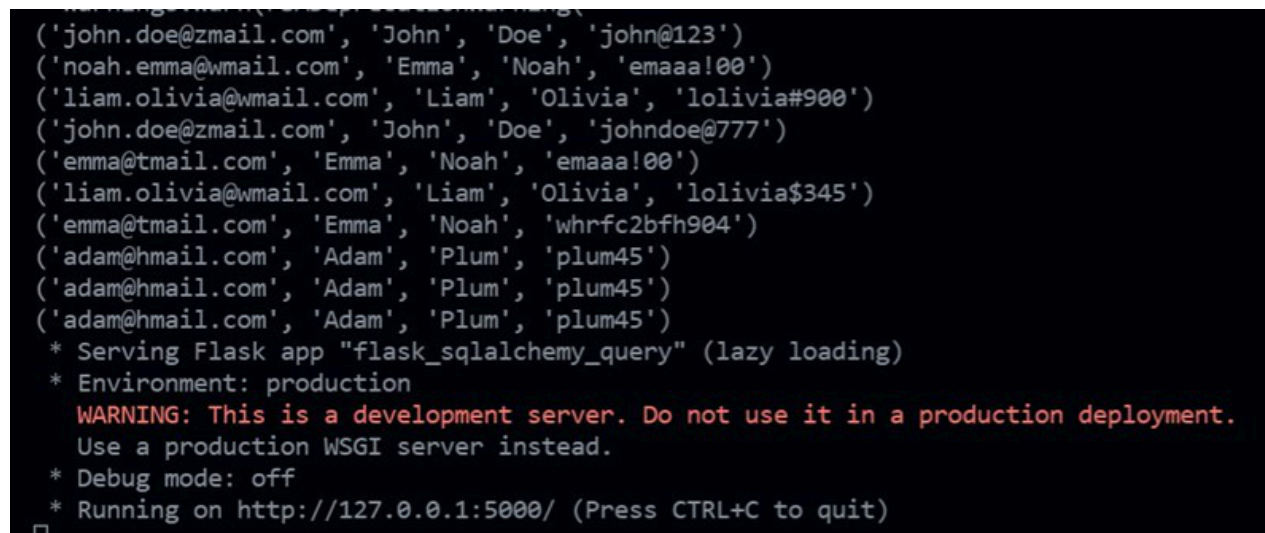
)

# VIEW THE RECORDS INSERTED
for record in db.engine.execute('SELECT * FROM users;'):
    print(record)

# RUN THE APP
if __name__ == '__main__':
    app.run()

```

Output:



```

('john.doe@zmail.com', 'John', 'Doe', 'john@123')
('noah.emma@wmail.com', 'Emma', 'Noah', 'emaaa!00')
('liam.olivia@wmail.com', 'Liam', 'Olivia', 'lolivia#900')
('john.doe@zmail.com', 'John', 'Doe', 'johndoe@777')
('emma@tmail.com', 'Emma', 'Noah', 'emaaa!00')
('liam.olivia@wmail.com', 'Liam', 'Olivia', 'lolivia$345')
('emma@tmail.com', 'Emma', 'Noah', 'whrfc2bfh904')
('adam@hmail.com', 'Adam', 'Plum', 'plum45')
('adam@hmail.com', 'Adam', 'Plum', 'plum45')
('adam@hmail.com', 'Adam', 'Plum', 'plum45')
* Serving Flask app "flask_sqlalchemy_query" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)

```

In this example, we created a simple flask app that does not have any route but instead runs raw SQL queries. We have created the SQLAlchemy connection and then executed 3 different raw SQL queries. The first query creates the user's table. The second query inserts some sample records in the table. The third query fetches all the records and displays them in the terminal.

In all three cases, we have used the **db.engine.execute()** method. The db.engine provides an SQLAlchemy engine connection and the execute method takes in a SQL query to execute the request.

Example 2

In this example, we have created 2 different routes to work with. These routes will act as an API where we can send a POST request with a query key in the body. The value for this query key will be the raw SQL query that we need to execute. The `get_results` API will be used to fetch the records that we get from the SELECT query. The `execute_query` API is used to execute raw SQL queries and will return the response message if the query is successfully executed or not.

Python

```
# IMPORT REQUIRED LIBRARIES
from flask import Flask, request
from flask_sqlalchemy import SQLAlchemy

# CREATE THE FLASK APP
app = Flask(__name__)

# ADD THE DATABASE CONNECTION TO THE FLASK APP
db = SQLAlchemy(app)
db_cred = {
    'user': 'root',          # DATABASE USER
    'pass': 'password',      # DATABASE PASSWORD
    'host': '127.0.0.1',     # DATABASE HOSTNAME
    'name': 'Geeks4Geeks'    # DATABASE NAME
}
app.config['SQLALCHEMY_DATABASE_URI'] = f"mysql+pymysql://{db_cred['user']}:{db_cred['pass']}@{db_cred['host']}/{db_cred['name']}"
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False

# APP ROUTE TO GET RESULTS FOR SELECT QUERY
@app.route('/get_results', methods=['POST'])
def get_results():

    # GET THE SQLALCHEMY RESULTPROXY OBJECT
    result = db.engine.execute(request.get_json()['query'])
    response = {}
    i = 1

    # ITERATE OVER EACH RECORD IN RESULT AND ADD IT
    # IN A PYTHON DICT OBJECT
    for each in result:
        response.update({f'Record {i}': list(each)})
        i+= 1
```

```

    return response

# APP ROUTE TO RUN RAW SQL QUERIES
@app.route('/execute_query', methods=['POST'])
def execute_query():

    try:
        db.engine.execute(request.get_json()['query'])
    except:
        return {"message": "Request could not be completed."}

    return {"message": "Query executed successfully."}

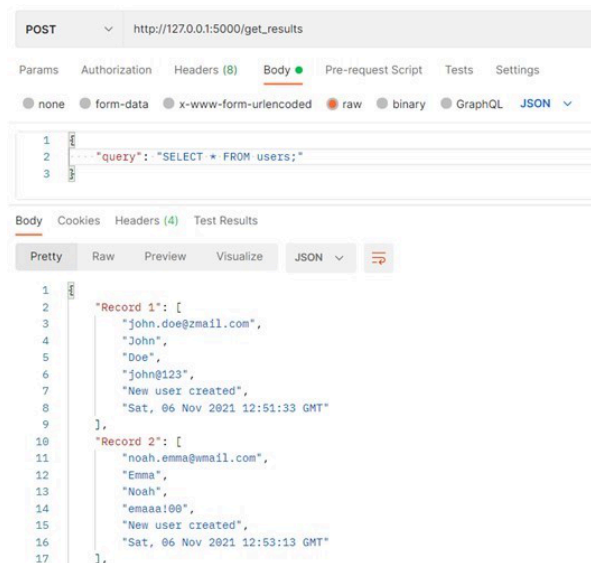
# RUN THE APP
if __name__ == '__main__':
    app.run()

```

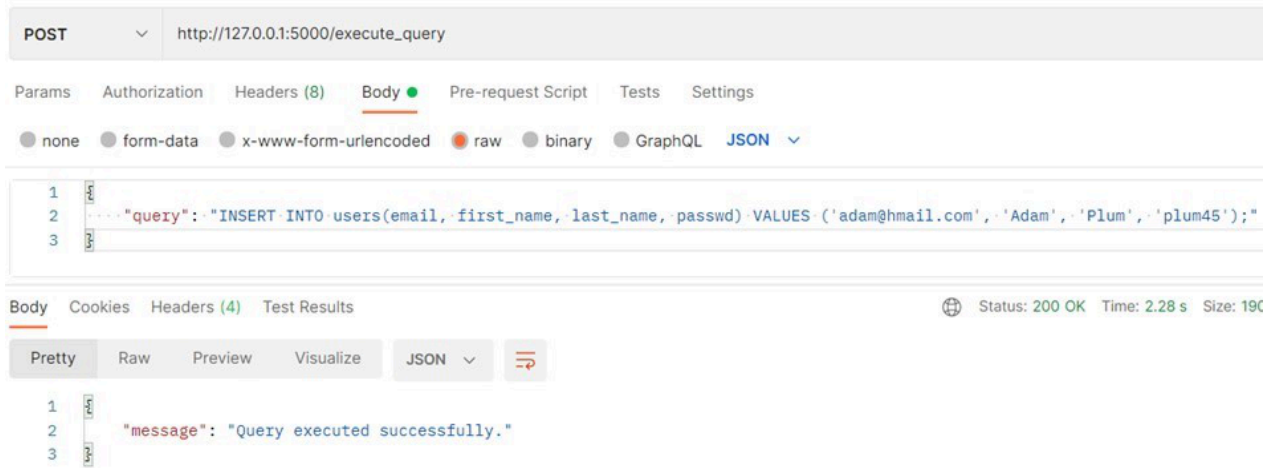
Output:

We will test the routes through POSTMAN. Following are the 3 cases that are tested using POSTMAN.

1. Running a SELECT query to fetch all the records through the get_results API



2. Next, we will test the execute_query API for a valid INSERT query



POST http://127.0.0.1:5000/execute_query

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

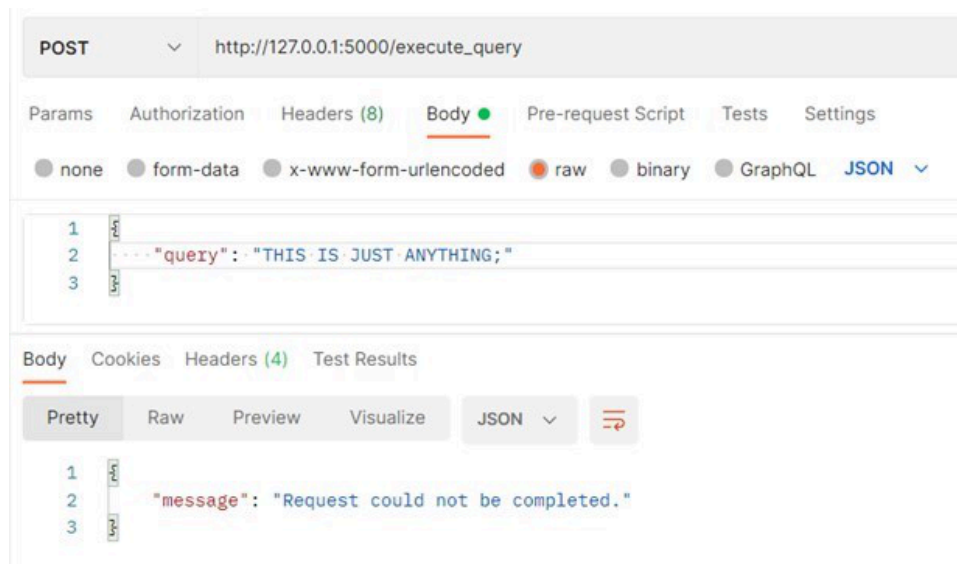
```
1 {
2   "query": "INSERT INTO users(email, first_name, last_name, passwd) VALUES ('adam@hmail.com', 'Adam', 'Plum', 'plum45');"
3 }
```

Body Cookies Headers (4) Test Results Status: 200 OK Time: 2.28 s Size: 190

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "Query executed successfully."
3 }
```

3. Lastly, we will put any random query and see if we get any error message



POST http://127.0.0.1:5000/execute_query

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "query": "THIS IS JUST ANYTHING;"
3 }
```

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "Request could not be completed."
3 }
```

Looking to dive into the world of programming or sharpen your Python skills? Our [Master Python: Complete Beginner to Advanced Course](#) is your ultimate guide to becoming proficient in Python. This course covers everything you need to build a solid foundation from fundamental programming concepts to advanced techniques. With **hands-on projects**, real-world examples, and expert guidance, you'll gain the confidence to tackle complex **coding challenges**. Whether you're starting from scratch or aiming to enhance your skills, this course is the perfect fit. Enroll now and master Python, the language of the future!



apath...



2

Previous Article

Login and Registration Project Using Flask and MySQL

Next Article

Subdomain in Flask | Python

Similar Reads

How to Execute Raw SQL in SQLAlchemy

In this article, we will see how to write a Conventional SQL query in SQLAlchemy using text() against a PostgreSQL database in python. Creating table for...

3 min read

Documenting Flask Endpoint using Flask-Autodoc

Documentation of endpoints is an essential task in web development and being able to apply it in different frameworks is always a utility. This article discusses...

4 min read

How to use Flask-Session in Python Flask ?

Flask Session - Flask-Session is an extension for Flask that supports Server-side Session to your application. The Session is the time between the client logs in to...

4 min read

How to Integrate Flask-Admin and Flask-Login

In order to merge the admin and login pages, we can utilize a short form or any other login method that only requires the username and password. This is know...

8 min read

Minify HTML in Flask using Flask-Minify

Flask offers HTML rendering as output, it is usually desired that the output HTML should be concise and it serves the purpose as well. In this article, we would...

12 min read

Flask URL Helper Function - Flask url_for()

In this article, we are going to learn about the flask url_for() function of the flask URL helper in Python. Flask is a straightforward, speedy, scalable library, used f...

11 min read

Raw SQL queries in Django views

Let's create a simple Django project that demonstrates the use of raw SQL queries in a view. In this example, we'll create a project to manage a list of book...

4 min read

How to View Raw SQL Queries Executed by Django

The most important thing while working on Django is to understanding the executed SQL queries running in the background. It is really important from the...

4 min read

How to Execute SQL queries from CGI scripts

In this article, we will explore the process of executing SQL queries from CGI scripts. To achieve this, we have developed a CGI script that involves importing...

5 min read

Create a SQL table from Pandas dataframe using SQLAlchemy

In this article, we will discuss how to create a SQL table from Pandas dataframe using SQLAlchemy. As the first steps establish a connection with your existing...

3 min read

Article Tags : [Python](#) [Python Flask](#) [Python-SQLAlchemy](#)

Practice Tags : [python](#)



Corporate & Communications Address:-
A-143, 9th Floor, Sovereign Corporate
Tower, Sector- 136, Noida, Uttar Pradesh
(201305) | Registered Address:- K 061,
Tower K, Gulshan Vivante Apartment,
Sector 137, Noida, Gautam Buddh
Nagar, Uttar Pradesh, 201305



Company

About Us
Legal
In Media
Contact Us
Advertise with us
GFG Corporate Solution
Placement Training Program
GeeksforGeeks Community

DSA

Data Structures
Algorithms
DSA for Beginners
Basic DSA Problems
DSA Roadmap
Top 100 DSA Interview Problems
DSA Roadmap by Sandeep Jain
All Cheat Sheets

Web Technologies

HTML
CSS
JavaScript
TypeScript
ReactJS
NextJS
Bootstrap
Web Design

Languages

Python
Java
C++
PHP
GoLang
SQL
R Language
Android Tutorial
Tutorials Archive

Data Science & ML

Data Science With Python
Data Science For Beginner
Machine Learning
ML Maths
Data Visualisation
Pandas
NumPy
NLP
Deep Learning

Python Tutorial

Python Programming Examples
Python Projects
Python Tkinter
Web Scraping
OpenCV Tutorial
Python Interview Question
Django

Computer Science

- Operating Systems
- Computer Network
- Database Management System
- Software Engineering
- Digital Logic Design
- Engineering Maths
- Software Development
- Software Testing

System Design

- High Level Design
- Low Level Design
- UML Diagrams
- Interview Guide
- Design Patterns
- OOAD
- System Design Bootcamp
- Interview Questions

School Subjects

- Mathematics
- Physics
- Chemistry
- Biology
- Social Science
- English Grammar
- Commerce
- World GK

DevOps

- Git
- Linux
- AWS
- Docker
- Kubernetes
- Azure
- GCP
- DevOps Roadmap

Interview Preparation

- Competitive Programming
- Top DS or Algo for CP
- Company-Wise Recruitment Process
- Company-Wise Preparation
- Aptitude Preparation
- Puzzles

GeeksforGeeks Videos

- DSA
- Python
- Java
- C++
- Web Development
- Data Science
- CS Subjects

@GeeksforGeeks, Sanchhaya Education Private Limited, All rights reserved