웹프로그래밍의 기초

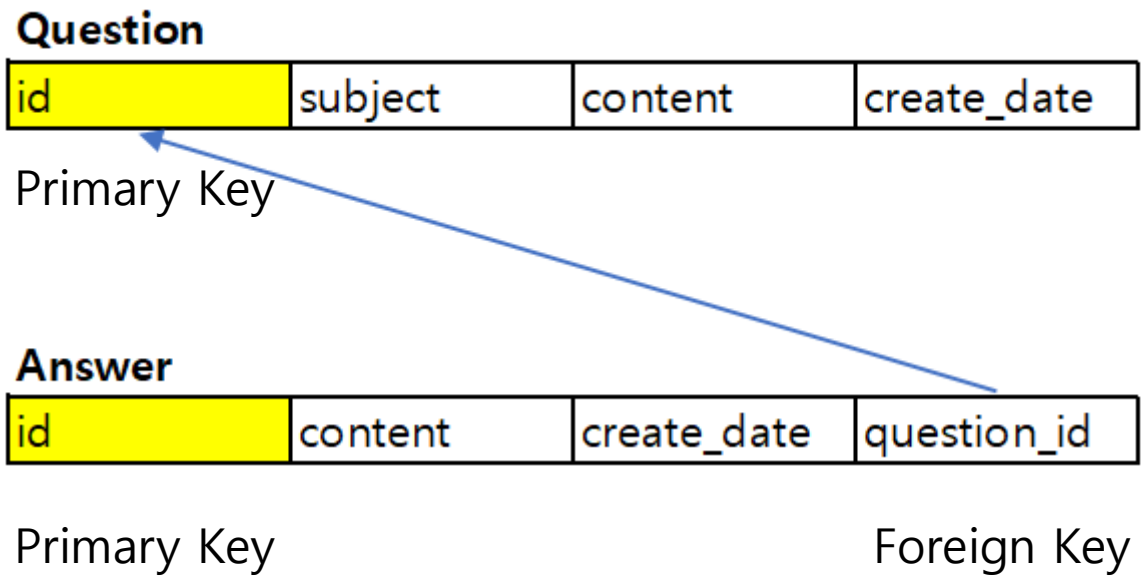
Week12

Linking DB to your app

Building a simple Q&A board
Step 1. Build database - review

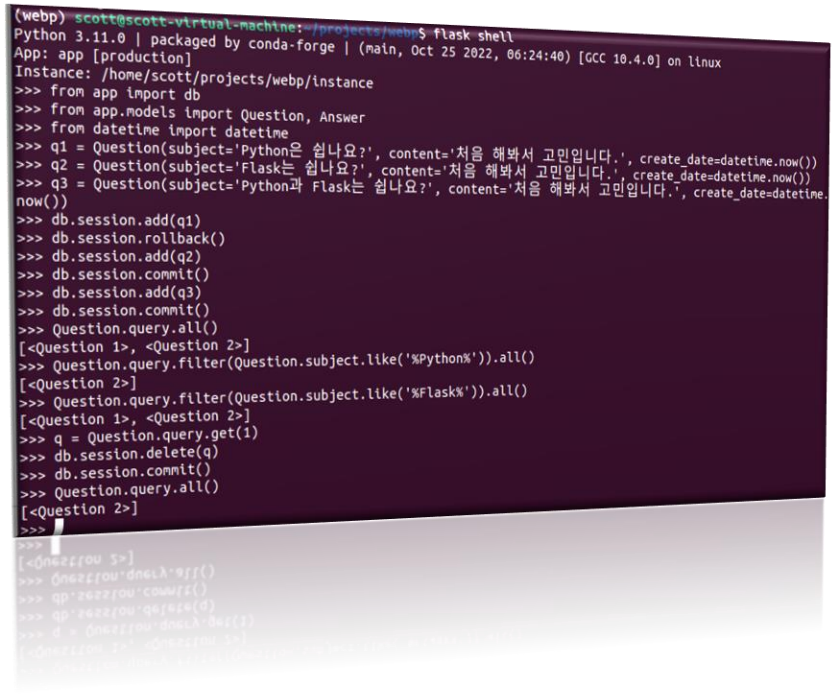
Database design

- For Q&A board, we need two tables linked together as follows.



Building a simple Q&A board
Step 2. access tables

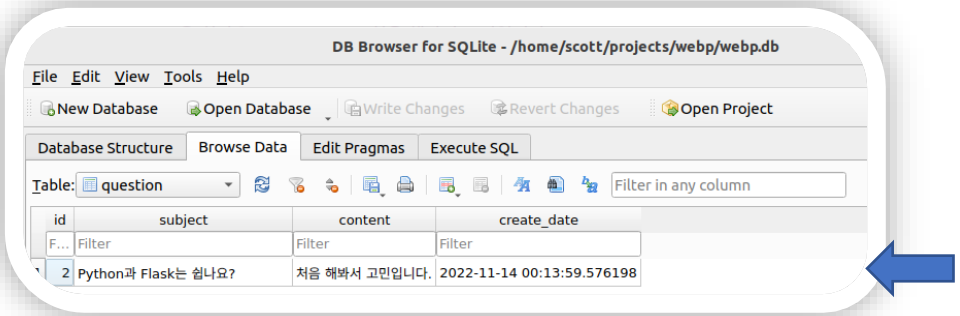
Create sample data for Question via flask shell



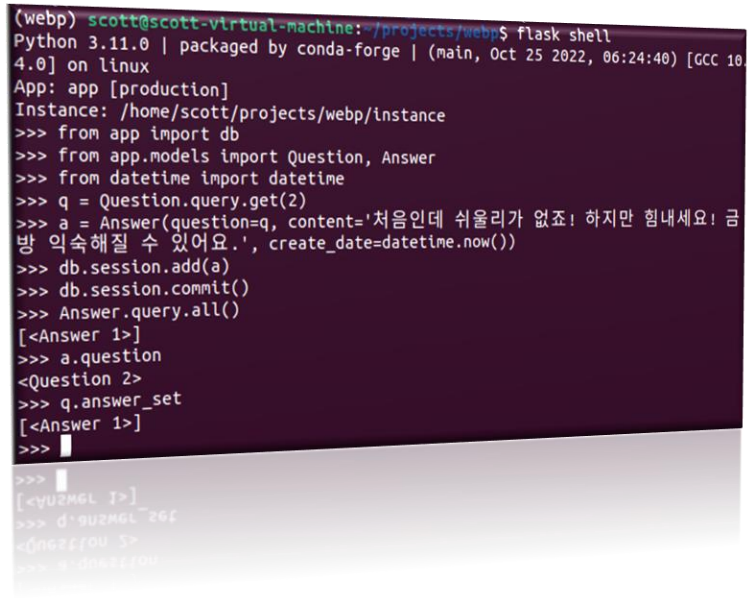
(webp) scott@scott-virtual-machine:~/projects/webp$ flask shell
Python 3.11.0 | packaged by conda-forge | (main, Oct 25 2022, 06:24:40) [GCC 10.4.0] on linux
App: app [production]
Instance: /home/scott/projects/webp/instance
>>> from app import db
>>> from app.models import Question, Answer
>>> from datetime import datetime
>>> q1 = Question(subject='Python은 쉽나요?', content='처음 해봐서 고민입니다.', create_date=datetime.now())
>>> q2 = Question(subject='Flask는 쉽나요?', content='처음 해봐서 고민입니다.', create_date=datetime.now())
>>> q3 = Question(subject='Python과 Flask는 쉽나요?', content='처음 해봐서 고민입니다.', create_date=datetime.now())
>>> db.session.add(q1)
>>> db.session.rollback()
>>> db.session.add(q2)
>>> db.session.commit()
>>> db.session.add(q3)
>>> db.session.commit()
>>> Question.query.all()
[<Question 1>, <Question 2>]
>>> Question.query.filter(Question.subject.like('%Python%')).all()
[<Question 2>]
>>> Question.query.filter(Question.subject.like('%Flask%')).all()
[<Question 1>, <Question 2>]
>>> q = Question.query.get(1)
>>> db.session.delete(q)
>>> db.session.commit()
>>> Question.query.all()
[<Question 2>]
>>>

Create sample data for Question via flask shell (Cont'd)

```
(webp) scott@scott-virtual-machine:~/projects/webp$ flask shell
Python 3.11.0 | packaged by conda-forge | (main, Oct 25 2022, 06:24:40) [GCC 10.4.0] on linux
App: app [production]
Instance: /home/scott/projects/webp/instance
>>> from app import db
>>> from app.models import Question, Answer
>>> from datetime import datetime
>>> q1 = Question(subject='Python은 쉽나요?', content='처음 해봐서 고민입니다.', create_date=datetime.now())
>>> q2 = Question(subject='Flask는 쉽나요?', content='처음 해봐서 고민입니다.', create_date=datetime.now())
>>> q3 = Question(subject='Python과 Flask는 쉽나요?', content='처음 해봐서 고민입니다.', create_date=datetime.now())
>>> db.session.add(q1)
>>> db.session.rollback()
>>> db.session.add(q2)
>>> db.session.commit()
>>> db.session.add(q3)
>>> db.session.commit()
>>> Question.query.all()
[<Question 1>, <Question 2>]
>>> Question.query.filter(Question.subject.like('%Python%')).all()
[<Question 2>]
>>> Question.query.filter(Question.subject.like('%Flask%')).all()
[<Question 1>, <Question 2>]
>>> q = Question.query.get(1)
>>> db.session.delete(q)
>>> db.session.commit()
>>> Question.query.all()
[<Question 2>]
>>>
```

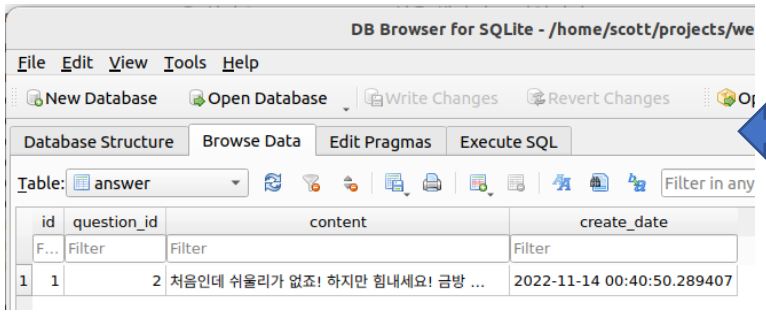


Create sample data for Answer via flask shell



(webp) scott@scott-virtual-machine:~/projects/webp$ flask shell
Python 3.11.0 | packaged by conda-forge | (main, Oct 25 2022, 06:24:40) [GCC 10.4.0] on linux
App: app [production]
Instance: /home/scott/projects/webp/instance
>>> from app import db
>>> from app.models import Question, Answer
>>> from datetime import datetime
>>> q = Question.query.get(2)
>>> a = Answer(question=q, content='처음인데 쉬울리가 없죠! 하지만 힘내세요! 금 방 익숙해질 수 있어요.', create_date=datetime.now())
>>> db.session.add(a)
>>> db.session.commit()
>>> Answer.query.all()
[<Answer 1>]
>>> a.question
<Question 2>
>>> q.answer_set
[<Answer 1>]
>>>

Create sample data for Answer via flask shell (Cont'd)




```
(webp) scott@scott-virtual-machine:~/projects/webp$ flask shell
Python 3.11.0 | packaged by conda-forge | (main, Oct 25 2022, 06:24:40) [GCC 10.4.0] on linux
App: app [production]
Instance: /home/scott/projects/webp/instance
>>> from app import db
>>> from app.models import Question, Answer
>>> from datetime import datetime
>>> q = Question.query.get(2)
>>> a = Answer(question=q, content='처음인데 쉬울리가 없죠! 하지만 힘내세요! 금 방 익숙해질 수 있어요.',
create_date=datetime.now())
>>> db.session.add(a)
>>> db.session.commit()
>>> Answer.query.all()
[<Answer 1>]
>>> a.question
<Question 2>
>>> q.answer_set
[<Answer 1>]
>>>
```

~/projects/webp/app
└── models.py

```
from app import db

class Question(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    subject = db.Column(db.String(200), nullable=False)
    content = db.Column(db.Text(), nullable=False)
    create_date = db.Column(db.DateTime(), nullable=False)

class Answer(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    question_id = db.Column(db.Integer, db.ForeignKey('question.id', ondelete='CASCADE'))
    question = db.relationship('Question', backref=db.backref('answer_set', cascade='all, delete-orphan'))
    content = db.Column(db.Text(), nullable=False)
    create_date = db.Column(db.DateTime(), nullable=False)
```

FYI, for other data manipulations

- Likewise, you can do update and delete via flask shell as follows.
  - Update

```
>>> q = Question.query.get(2)
>>> q
<Question 2>
>>> q.subject = 'Flask가 좋아요.'
>>> db.session.commit()
```

  - Delete

```
>>> q = Question.query.get(2)
>>> q
<Question 2>
>>> db.session.delete(q)
>>> db.session.commit()
```

Building a simple Q&A board
Step 3. expanding the horizon

Ubuntu 64-bit 22.04 LTS light for WP22 - VMware Workstation 16 Player (Non-commercial use only)

Player

Activities Terminal 11월 17 01 : 11

scott@scott-virtual-machine: ~/projects/webp

```
(webp) scott@scott-virtual-machine:~/projects/webp$
```

Open *Untitled Document 1 Save

```
1 오늘 배고픈데 수업하기 싫어요!!!
2
3 수업 끝나고 같이 간식 드실 분 모집합니다.
```

Plain Text Tab Width: 8 Ln 3, Col 25 INS

New Tab

Search with Google or enter address

Firefox

Search with Google or enter address

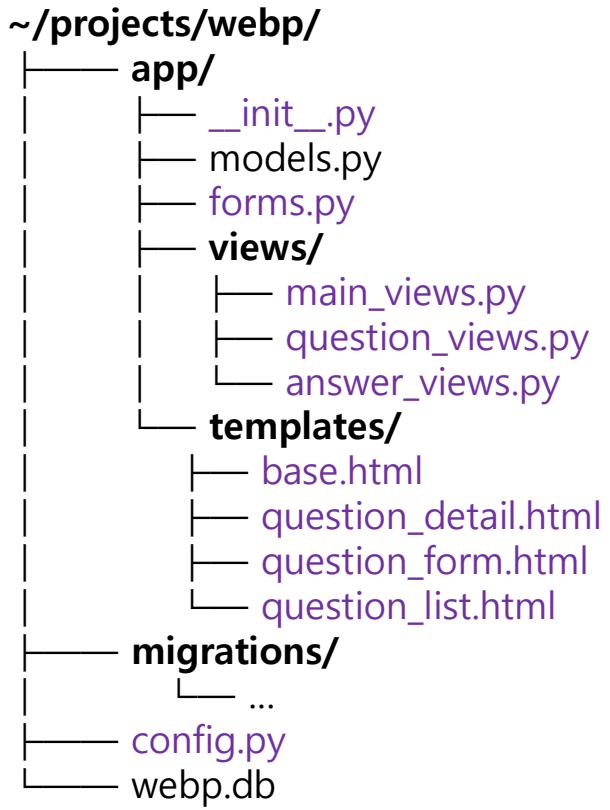
127.0.0.1 support.m... anaconda jetbrains account.je... pythonhos...

00:00 Rec on

오전 1:11 2022-11-17

Simple Bulletin Board Service

- We are going to expand the one that from the last weeks configurations.



~/projects/webp
└──── config.py

```
import os

BASE_DIR = os.path.dirname(__file__)

SQLALCHEMY_DATABASE_URI = 'sqlite:///{}'.format(os.path.join(BASE_DIR, 'webp.db'))
SQLALCHEMY_TRACK_MODIFICATIONS = False
SECRET_KEY = "webp"
```

~/projects/webp/app
└──── __init__.py

```
from flask import Flask
from flask_migrate import Migrate
from flask_sqlalchemy import SQLAlchemy

import config

db = SQLAlchemy()
migrate = Migrate()


def create_app():
    app = Flask(__name__)
    app.config.from_object(config)

    # ORM
    db.init_app(app)
    migrate.init_app(app, db)
    from . import models

    # BP
    from .views import main_views, question_views, answer_views
    app.register_blueprint(main_views.bp)
    app.register_blueprint(question_views.bp)
    app.register_blueprint(answer_views.bp)
    return app
```

~/projects/webp/app/views
└── main_views.py

```python
from flask import Blueprint, url_for
from werkzeug.utils import redirect

bp = Blueprint('main', __name__, url_prefix='/')

@bp.route('/')
def index():
    return redirect(url_for('question._list'))
```

~/projects/webp/app
└── forms.py

```python
from flask_wtf import FlaskForm
from wtforms import StringField, TextAreaField
from wtforms.validators import DataRequired

class QuestionForm(FlaskForm):
    subject = StringField('제목', validators=[DataRequired('질문 제목을 입력해주세요.')])
    content = TextAreaField('내용', validators=[DataRequired('질문 내용을 입력해주세요.')])

class AnswerForm(FlaskForm):
    content = TextAreaField('내용', validators=[DataRequired('댓글 내용을 입력해주세요.')])
```

~/projects/webp/app/templates
└── question_list.html

```
{% extends 'base.html' %}
{% block content %}
<table>
<thead>
<tr>
    <th>번호</th>
    <th>제목</th>
    <th>작성일시</th>
</tr>
</thead>
<tbody>
{% if question_list %}
{% for question in question_list %}
<tr>
    <td>{{ loop.index }}</td>
    <td>
    <a href="{{ url_for('question.detail', question_id=question.id) }}">{{ question.subject }}</a>
    </td>
    <td>{{ question.create_date.strftime("%Y/%m/%d %H:%M:%S") }}</td>
</tr>
{% endfor %}
{% else %}
<tr>
    <td colspan="3">등록된 질문이 없습니다.</td>
</tr>
{% endif %}
</tbody>
</table>
<a href="{{ url_for('question.create') }}"><button type="button">질문 등록</button></a>
{% endblock %}
```

~/projects/webp/app/views
└── question_views.py

```python
from flask import Blueprint, render_template, request, url_for
from werkzeug.utils import redirect
from datetime import datetime
from app import db

from app.models import Question
from app.forms import QuestionForm, AnswerForm

bp = Blueprint('question', __name__, url_prefix='/question')

@bp.route('/list/')
def _list():
    question_list = Question.query.order_by(Question.create_date.desc())
    return render_template('question_list.html', question_list=question_list)

@bp.route('/detail/<int:question_id>/')
def detail(question_id):
    form = AnswerForm()
    question = Question.query.get_or_404(question_id)
    return render_template('question_detail.html', question=question, form=form)

@bp.route('/create/', methods=('GET', 'POST'))
def create():
    form = QuestionForm()
    if request.method == 'POST' and form.validate_on_submit():
        question = Question(subject=form.subject.data, content=form.content.data, create_date=datetime.now())
        db.session.add(question)
        db.session.commit()
        return redirect(url_for('main.index'))
    return render_template('question_form.html', form=form)
```

~/projects/webp/app/templates
|
└── question_detail.html.html

```
{% extends 'base.html' %}
{% block content %}
<h1>질문</h1>
<h3>{{ question.subject }}</h3>

<div>
    {{ question.content }}
</div>

<hr>
<h1>댓글</h1>
<h5>{{ question.answer_set|length }}개의 답변이 있습니다.</h5>

<div>
<table>
<tbody>
{% for answer in question.answer_set %}
<tr>
    <td>{{ answer.content }}</td>
    <td>{{ answer.create_date.strftime("%Y/%m/%d %H:%M:%S") }}</td>
</tr>
{% endfor %}
</table>
</div>

<form action="{{ url_for('answer.create', question_id=question.id) }}" method="post">
    {{ form.csrf_token }}
    {% if form.errors %}
        <div class="alert alert-danger" role="alert">
            {% for field, errors in form.errors.items() %}
            <strong>{{ form[field].label }}</strong>
            <ul>
                {% for error in errors %}
                <li>{{ error }}</li>
                {% endfor %}
            </ul>
            {% endfor %}
        </div>
    {% endif %}
    <h3>새로운 댓글 달기</h3>
    <textarea name="content" id="content" rows="15"></textarea><p>
    <input type="submit" value="댓글등록">
</form>
<a href="{{ url_for('main.index') }}"><button type="button">목록으로</button></a>
{% endblock %}
```

~/projects/webp/app/templates
|
└── question_form.html

```
{% extends 'base.html' %}
{% block content %}

<div>
    <h1>질문등록</h1>
    <form method="post">

        {{ form.csrf_token }}

        {{ form.subject.label }}
        {{ form.subject() }}
                    <br>

        {{ form.content.label }}
        {{ form.content() }}
        <br>

        {% if form.errors %}
            {% for field, errors in form.errors.items() %}
            <strong>{{ form[field].label }}</strong>
            <ul>
                {% for error in errors %}
                <li>{{ error }}</li>
                {% endfor %}
            </ul>
            {% endfor %}
        {% endif %}

        <button type="submit">저장하기</button>
    </form>
</div>
{% endblock %}
```

~/projects/webp/app/templates
|
└─── base.html

```
<!doctype html>
<html lang="ko">
<head>
    <title>My project - webp</title>
</head>
<body>
    {% block content %}
    {% endblock %}
</body>
</html>
```

~/projects/webp/app/views
|
└─── answer_views.py

```
from flask import Blueprint, url_for, request, render_template
from werkzeug.utils import redirect
from datetime import datetime

from app import db
from app.forms import AnswerForm
from app.models import Question, Answer

bp = Blueprint('answer', __name__, url_prefix='/answer')

@bp.route('/create/<int:question_id>', methods = ['POST'])
def create(question_id):
    form = AnswerForm()
    question = Question.query.get_or_404(question_id)
    if form.validate_on_submit():
        content = request.form['content']
        answer = Answer(content=content, create_date=datetime.now())
        question.answer_set.append(answer)
        db.session.commit()
        return redirect(url_for('question.detail', question_id=question_id))
    return render_template('question_detail.html', question=question, form=form)
```