



# Django Model Relationship

## ❖ 학습해야 할 내용

- ✓ Django 1:N Model Relation

## 1. 1:N True or False

각 문항을 읽고 맞으면 T, 틀리면 F를 작성하고 틀렸다면 그 이유도 함께 작성하시오.

- 1) ForeignKey는 부모 테이블의 데이터를 참조하기 위한 키이다.
- 2) 1:N 관계에서 1은 N의 데이터를 직접 참조 할 수 있다.
- 3) on\_delete 속성은 ForeignKey 필드의 필수 인자이다.
- 4) 1:N 관계에서 외래 키는 반드시 부모 테이블의 PrimaryKey여야 한다.

## 2. ForeignKey column name

다음과 같이 이름이 articles인 app의 models.py에 작성된 코드를 바탕으로 테이블이 만들어 졌을 때, 데이터베이스에 저장되는 ForeignKey 컬럼의 이름과 테이블의 이름이 무엇인지 작성하시오.

```
from django.db import models

class Question(models.Model):
    title = models.CharField(max_length=50)

class Comment(models.Model):
    answer = models.ForeignKey(Question, on_delete=models.CASCADE)
    content = models.CharField(max_length=100)
```



# Django Model Relationship

## 3. 1:N model manager

위 2번 문제 모델 관계를 바탕으로 어느 template 페이지가 다음과 같이 작성되어 있을 때, 질문(Question)에 작성된 모든 댓글(Comment)을 출력하고자 한다. 제시된 View 함수 코드를 참고하여 해당 template에서 Question 객체를 사용할 수 있다면 빈칸 \_\_(a)\_\_에 들어갈 알맞은 코드를 작성하시오.

```
from .models import Question

def detail(request, pk):
    question = Question.objects.get(pk=pk)
    context = {
        'question': question,
    }
    return render(request, 'articles/detail.html', context)
```

```
{% for comment in __(a)__ %}
    <p>{{ comment.content }}</p>
{% endfor %}
```

# Django Model Relationship



## 4. next parameter

다음과 같이 게시글을 삭제하는 delete 함수와 로그인을 위한 login 함수가 작성되어 있다. 만약 비로그인 사용자가 삭제를 시도한다면, django는 해당 사용자를 url에 next 파라미터가 붙은 login 페이지로 redirect 한다.

- /accounts/login/?next=/articles/1/delete/

- 1) redirect된 로그인 페이지에서 로그인에 성공했을 때 발생하는 HTTP response status code를 작성하고, 이 오류가 발생한 원인을 작성하시오.
- 2) 위에서 발생한 오류를 해결하기 위해 다음과 같이 동작하는 코드로 수정하시오.
  - 게시글 삭제는 HTTP POST method로만 가능하다.
  - 인증되지 않은 사용자가 게시글 삭제를 시도하는 경우, 해당 게시글 상세페이지로 redirect 되도록 한다. (게시글은 삭제되지 않는다.)

```
from django.views.decorators.http import require_POST
from django.contrib.auth.decorators import login_required
from django.shortcuts import render, redirect, get_object_or_404
from .models import Article
```

```
@login_required
@require_POST
def delete(request, article_pk):
    article = get_object_or_404(Article, pk=article_pk)
    article.delete()
    return redirect('articles:index')
```

```
def login(request):
    if request.user.is_authenticated:
        return redirect('articles:index')
    if request.method == 'POST':
        form = AuthenticationForm(request, request.POST)
        if form.is_valid():
            auth_login(request, form.get_user())
            return redirect(request.GET.get('next') or 'articles:index')
    else:
        form = AuthenticationForm()
    context = {
        'form': form
    }
    return render(request, 'accounts/login.html', context)
```