BackEnd Workshop-2

Clarusway

Subject: Django ORM (SQL to ORM)

Learning Goal

- Practice Django ORM

Introduction

In this workshop, we will convert SQL to ORM.

SQL to ORM

1. Convert from SQL to ORM.

- SQL

```
SELECT *
FROM Person;
```

- Django

```
Person.objects.all()
# The all() method returns a QuerySet of all the objects in the database.
```

2. Convert from SQL to ORM.

- SQL

```
SELECT name, age
FROM Person;
```

- Django

```
Person.objects.values('name', 'age')
```

3. Convert from SQL to ORM.

- SQL

```
SELECT *
FROM Person;
```

- Django

```
Person.objects.all()
```

4. Convert from SQL to ORM.

- SQL

```sql
SELECT DISTINCT name, age
FROM Person;
```

- Django

```python
Person.objects.values('name', 'age').distinct()
```

5. Convert from SQL to ORM.

- SQL

```sql
SELECT *
FROM Person
LIMIT 10;
```

- Django

```python
Person.objects.all()[:10]
```

6. Convert from SQL to ORM.

- SQL

```sql
SELECT *
FROM Person
OFFSET 5
LIMIT 5;
```

- Django

```
Person.objects.all()[5:10]
```

7. Convert from SQL to ORM.

- SQL

```
SELECT *
FROM Person
WHERE id = 1;
```

- Django

```
Person.objects.filter(id=1)
```

8. Convert from SQL to ORM.

- SQL

```
WHERE age > 18;
WHERE age >= 18;
WHERE age < 18;
WHERE age <= 18;
WHERE age != 18;
```

- Django

```
Person.objects.filter(age__gt=18)
Person.objects.filter(age__gte=18)
Person.objects.filter(age__lt=18)
Person.objects.filter(age__lte=18)
Person.objects.exclude(age=18)

# Relational operators
# gt -Greater than.
# gte -Greater than or equal to.
# lt -Less than.
# lte -Less than or equal to.
```

9. Convert from SQL to ORM.

- SQL

```
SELECT *
FROM Person
WHERE age BETWEEN 10 AND 20;
```

- Django

```
Person.objects.filter(age__range=(10, 20))
```

10. Convert from SQL to ORM.

- SQL

```
WHERE name like '%A%';
WHERE name like binary '%A%';
WHERE name like 'A%';
WHERE name like binary 'A%';
```

- Django

```
Person.objects.filter(name__icontains='A')
Person.objects.filter(name__contains='A')
Person.objects.filter(name__istartswith='A')
Person.objects.filter(name__startswith='A')
```

11. Convert from SQL to ORM.

- SQL

```
WHERE id in (1, 2);
```

- Django

```
Person.objects.filter(id__in=[1, 2])
```

12. Convert from SQL to ORM.

- SQL

```
WHERE gender='male' AND age > 25;
```

- Django

```
Person.objects.filter(gender='male', age__gt=25)
```

13. Convert from SQL to ORM.

- SQL

```
WHERE gender='male' OR age > 25;
```

- Django

```
from django.db.models import Q
Person.objects.filter(Q(gender='male') | Q(age__gt=25))
```

14. Convert from SQL to ORM.

- SQL

```
WHERE NOT gender='male';
```

- Django

```
Person.objects.exclude(gender='male')
```

15. Convert from SQL to ORM.

- SQL

```
WHERE age is NULL;
WHERE age is NOT NULL;
```

- Django

```
Person.objects.filter(age__isnull=True)
Person.objects.filter(age__isnull=False)
```

16. Convert from SQL to ORM.

- SQL

```
SELECT *
FROM Person
order by age;
```

- Django

```
Person.objects.order_by('age')
```

17. Convert from SQL to ORM.

- SQL

```
INSERT INTO Person
VALUES ('Jack', '23', 'male');
```

- Django

```
Person.objects.create(name='jack', age=23, gender='male)
```

18. Convert from SQL to ORM.

- SQL

```
UPDATE Person
SET age = 20
WHERE id = 1;
```

- Django

```
person = Person.objects.get(id=1)
person.age = 20
person.save()
```

19. Convert from SQL to ORM.

- SQL

```
UPDATE Person
SET age = age * 1.5;
```

- Django

```
# class F
# An F() object represents the value of a model field, transformed value of a
model field, or annotated column. It makes it possible to refer to model field
values and perform database operations using them without actually having to pull
them out of the database into Python memory.
from django.db.models import F

Person.objects.update(age=F('age')*1.5)
```

20. Convert from SQL to ORM.

- SQL

```
DELETE FROM Person;
```

- Django

```
Person.objects.all().delete()
```

21. Convert from SQL to ORM.

- SQL

```
SELECT AVG(age)
FROM Person;
```

- Django

```
from django.db.models import Max
Person.objects.all().aggregate(Avg('age'))
```

22. Convert from SQL to ORM.

- SQL

```
SELECT SUM(age)
FROM Person;
```

- Django

```
from django.db.models import Sum
Person.objects.all().aggregate(Sum('age'))
```

23. Convert from SQL to ORM.

- SQL

```
SELECT COUNT(*)
FROM Person;
```

- Django

```
Person.objects.count()
```

24. Convert from SQL to ORM.

- SQL

```
SELECT gender, COUNT('gender') as count
FROM Person
GROUP BY gender
HAVING count > 1;
```

- Django

```
Person.objects.values('gender').annotate(count=Count('gender'))
```

25. Convert from SQL to ORM.

- SQL

```
SELECT name
FROM Book
LEFT JOIN Publisher
ON Book.publisher_id = Publisher.id
WHERE Book.id=1;
```

- Django

```
book = Book.objects.select_related('publisher').get(id=1)
book.publisher.name
```

26. Convert from SQL to ORM.

- SQL

```
SELECT *
FROM Book
WHERE Book.publisher_id = 1;
```

- Django

```
Publisher.objects.prefetch_related('book_set').get(id=1)
```

☺ **Thanks for Attending** ✍

Clarusway