EXERCISES QUE TRABAJAREMOS EN EL CUE

- **EXERCISE 1: REALIZAR CONSULTAS PERSONALIZADAS AL MODELO DE DATOS POR EL MÉTODO RAW()**
- **EXERCISE 2: ASIGNACIÓN DE CAMPOS A LA CONSULTA**
- **EXERCISE 3: SQL CONSULTA CON OR**
- **EXERCISE 4: CONSULTA DE LLAVES FORÁNEAS**
- **EXERCISE 5: ACCEDIENDO SEGÚN SU ÍNDICE**
- **EXERCISE 6: EJECUTAR SQL PERSONALIZADO DIRECTAMENTE Y CURSORES**

EXERCISE 1: REALIZAR CONSULTAS PERSONALIZADAS AL MODELO DE DATOS POR EL MÉTODO RAW()

Partimos del hecho de que hemos construido nuestro modelo relacional de fábricas y productos. Vamos a seleccionar todas las instancias creadas de los objetos Automotriz y Modelo de Carros, respectivamente. Procedemos a conectarnos a la consola Shell de Python:

```
$ python manage.py shell
```

Importamos los modelos:

```
>>> from relaciones.models import Automotriz, ModeloCarro
```

Realizamos la consulta personalizada en SQL para Automotriz:

```
>>> automotrices = Automotriz.objects.raw('SELECT * FROM
relaciones_automotriz')

>>> for a in automotrices:
...     print(a)
...
```

Salida:

```
Toyota
Fiat
Ford
Chevrolet
```

Procedemos a realizar la consulta personalizada en SQL para Modelo de Carros:

```
>>> modelos_carros = ModeloCarro.objects.raw('SELECT * FROM
relaciones_modelocarro')

>>> for m in modelos_carros:
...     print(m)
...
```

Salida:

```
Fiat Uno
EcoSport
F-150
Super Duty
Captiva
Cruze
Toyota Corolla
Toyota Fortuner
```

EXERCISE 2: ASIGNACIÓN DE CAMPOS A LA CONSULTA

Puede asignar campos en la consulta a campos de modelo, utilizando el argumento de traducción para `raw()`. Este es un diccionario que asigna nombres de campos en la consulta a nombres de campos en el modelo. Por ejemplo:

```
>>> mapeo = {'modelo_carro':'nombre', 'pk':'id'}

>>> modelos_as_campo = ModeloCarro.objects.raw('SELECT id, nombre FROM
relaciones_modelocarro', translations=mapeo)

>>> for m in modelos_as_campo:
...     print("%s es del fabricante %s." % (m.nombre, m.automotriz))
```

Salida:

```
Fiat Uno es del fabricante Fiat.
EcoSport es del fabricante Ford.
F-150 es del fabricante Ford.
Super Duty es del fabricante Ford.
Captiva es del fabricante Chevrolet.
Cruze es del fabricante None.
Toyota Corolla es del fabricante Toyota.
Toyota Fortuner es del fabricante Toyota.
```

Pasando parámetros a `raw()`:

```
>>> modelo = 'EcoSport'

>>> consulta = ModeloCarro.objects.raw('SELECT * FROM
relaciones_modelocarro WHERE nombre = %s', [modelo])

>>> for m in consulta:
...     print("La %s tiene el id %s." % (m.nombre, m.id))
```

Salida:

```
La EcoSport tiene el id 2.
```

EXERCISE 3: SQL CONSULTA CON OR

```
>>> modelo2 = 'Captiva'

>>> consulta2 = ModeloCarro.objects.raw('SELECT * FROM
relaciones_modelocarro WHERE nombre = %s OR nombre = %s', [modelo,
modelo2])

>>> for m in consulta2:
...     print("La %s tiene el id %s." % (m.nombre, m.id))
```

Salida:

```
La EcoSport tiene el id 2.
La Captiva tiene el id 5.
```

EXERCISE 4: CONSULTA DE LLAVES FORÁNEAS

```
>>> consulta3 = ModeloCarro.objects.raw('SELECT * FROM
"relaciones_modelocarro" LEFT OUTER JOIN "relaciones_automotriz" ON
("relaciones_modelocarro"."automotriz_id" =
"relaciones_automotriz"."id")')

>>> for m in consulta3:
...     print("La %s es del fabricante %s y tiene el id %s." % (m.nombre,
m.automotriz, m.id))
```

Salida:

```
La Fiat Uno es del fabricante Fiat y tiene el id 1.
La EcoSport es del fabricante Ford y tiene el id 2.
La F-150 es del fabricante Ford y tiene el id 3.
La Super Duty es del fabricante Ford y tiene el id 4.
La Captiva es del fabricante Chevrolet y tiene el id 5.
La Cruze es del fabricante None y tiene el id 6.
La Toyota Corolla es del fabricante Toyota y tiene el id 7.
La Toyota Fortuner es del fabricante Toyota y tiene el id 8.
```

EXERCISE 5: ACCEDIENDO SEGÚN SU ÍNDICE

La indexación y el corte no se realizan a nivel de la base de datos. Si tiene una gran cantidad de objetos de una determinada tabla en su base de datos, es más eficiente limitar la consulta al nivel de SQL.

Consultado las columnas:

```
>>> consulta.columns
```

Salida:

```
['id', 'nombre', 'automotriz_id', 'costo_fabricacion', 'f_fabricacion',
'precio_venta']
```

Consultando al índice **[0]** el valor de precio de venta:

```
>>> consulta[0].precio_venta
```

Salida:

```
Decimal('2500.00')
```

Consultando el registro en el índice **[0]**:

```
>>> consulta[0]
```

Salida:

```
<ModeloCarro: EcoSport>
```

Consultando el registro en el índice **[1]**:

```
>>> consulta[1]
```

Salida (fuera de rango, solo tenemos una solo registro de la consulta):

```
Traceback (most recent call last):
  File "<console>", line 1, in <module>
  File "/home/luispc/.virtualenvs/django-orm-
postgresql/lib/python3.8/site-packages/django/db/models/query.py", line
2076, in __getitem__
    return list(self)[k]
IndexError: list index out of range
```

EXERCISE 6: EJECUTAR SQL PERSONALIZADO DIRECTAMENTE Y CURSORES

Actualizando el registro c:

```
>>> from django.db import connection

>>> automotriz = 'Fiat'

>>> cursor = connection.cursor()

>>> cursor.execute("UPDATE relaciones_automotriz SET pais = 'Brasil'
WHERE nombre = %s", [automotriz])
>>> cursor.execute("SELECT pais FROM relaciones_automotriz WHERE nombre
= %s", [automotriz])

>>> row = cursor.fetchall()

>>> row
```

Salida:

```
[('Brasil',)]
```