

# ENTREGA 2 PBD: Conexión a base de datos con Python

## Contenido

1.	INTRODUCCIÓN .....	2
a.	CÓDIGO CLAVE .....	2
b.	FUNCIONES AUXILIARES UTILIZADAS .....	3
2.	Ejercicio 1: dbObtenerEmpleados .....	4
	SALIDA DEL CÓDIGO.....	6
3.	EJERCICIO 2: dbConsultarEmpleados() .....	7
	SALIDA PARCIAL DEL MÉTODO .....	10
4.	EJERCICIO 3: dbConsultarDepartamentos().....	12
	SALIDA DEL CÓDIGO PARCIAL.....	15
5.	EJERCICIO 4: dbInsertarDepartamentos() .....	15
	SALIDA PARCIAL DEL MÉTODO .....	17
6.	EJERCICIO 5: dbModificarDepartamentos().....	19
	SALIDA PARCIAL DEL MÉTODO .....	25
7.	EJERCICIO 6: dbBorrarDepartamentos().....	26
	SALIDA PARCIAL DEL MÉTODO .....	28
7.	EJERCICIO 7: dbInsertarMultiplesDepartamentos() .....	28
	SALIDA PARCIAL DEL MÉTODO .....	30
8.	EJERCICIO 8: dbBorrarMultiplesDepartamentos(): .....	30
	SALIDA PARCIAL DEL MÉTODO .....	32
8.	EXTRA POINT .....	33
	SALIDA PARCIAL DEL MÉTODO .....	34
	SALIDA PARCIAL DEL CÓDIGO, DADO COMO PARÁMETRO DE ENTRADA EL 4. ....	36
8.	SALIDA COMPLETA POR CONSOLA DE POSTGRESQL.....	37
9.	SALIDA COMPLETA POR CONSOLA DE ORACLE.....	57

# 1. INTRODUCCIÓN

Este documento tiene la finalidad de explicar las diferentes estructuras, métodos y código fuente utilizado para establecer una conexión a una base de datos relacional con Python.

Los SGBD utilizados han sido Oracle y PostgreSQL, por lo que se explicarán cada código para dichos SGBD. Antes de nada, hay que aclarar las **únicas diferencias** que se van a poder observar en el código fuente a la hora de realizar las consultas tanto para Oracle como para PostgreSQL.

La única diferencia que se observa en el código es a la hora de crear las sentencias y preparar las variables en la petición a realizar.

En Oracle, cuando se quiere preparar una petición a tu base de datos, las variables dinámicas se preparan, dentro del string **:nombre-parametro**

En PostgreSQL, cambiará la declaraciones de las variables de las peticiones, poniendo **“%s”**.

Antes de pasar a la explicación y mostrado de código y salidas, se realizará una breve explicación del formato que se seguirá en cada apartado. En primer lugar, se explicará en las primeras líneas el objetivo de dicho apartado y salidas esperadas. A continuación, se irán mostrando y explicando a la vez, pequeños trozos de código de la función, de arriba abajo, en orden de aparición en el código original. Y para finalizar, se mostrará el código completo de la funcionalidad y el texto de salida esperado.

## a. CÓDIGO CLAVE

En todas las peticiones y métodos, se podrá observar muchas líneas de código que se repiten. Es por eso, que para no tener que explicarlos todas las veces, se explicará en este apartado para tener un mejor contexto y funcionamiento del código.

Estas líneas de código son:

```
1. cursor = conexion.cursor()
```

Método para guardar la instancia de la conexión al SGBD. Esta instancia se guarda en una variable llamada “cursor”.

```
cursor.execute(consulta, [lista_variables])
```

Este método ejecuta la petición, establecida en la variable “consulta”, a la base de datos, guardada en la variable “cursor”. Este método puede tener un 2º parámetro, correspondiente a las variables dinámicas que se le pueden establecer en la consulta. Tal como se ha comentado en la introducción, dependiendo del SGBD, se establecerá de una forma u otra.

```
resul = cursor.fetchall()
resul = cursor.fetchone()
resul = cursor.fetchmany(n)
```

Estas tres funciones, tiene el mismo objetivo: devolver la información de las tuplas de la consulta que se ha realizado a la base de datos. La diferencia es el NUMERO de tuplas que devuelven:

- `cursor.fetchall()`: Devuelve todas las tuplas que ha obtenido de la petición a la base de datos.
- `cursor.fetchone()`: Devuelve la primera de la lista de tuplas recuperadas. Hay que tener en cuenta que funciona como un iterador. La segunda vez que se llame a esta función, devolverá 1 tupla correspondiente a la posición 2 de la lista de tuplas recuperadas.
- `Cursor.fetchmany(n)`: Devuelve de n en n tuplas. Mismo funcionamiento que `fetchone()`

```
cursor.close()
```

Esta última función, elimina la instancia de la conexión a la base de datos que se había establecido, borrando el espacio de memoria correspondiente a esta variable.

## b. FUNCIONES AUXILIARES UTILIZADAS

En este apartado, se declararán y explicarán las funciones auxiliares utilizadas con el propósito de implementar los métodos pedidos.

- **mostrarValoresDepartamentos(tupla)**: Este método tiene la funcionalidad de mostrar por pantalla, dado el formato pedido por el profesor, la información de una fila de un departamento, dado como parámetro el objeto "Departamento". Como es una función que se repite en muchos métodos, se ha decidido, con el fin de cumplir con el paradigma DRY (Dont repeat Yourself), manteniendo un código más limpio.

```
def mostrarValoresDepartamentos(tupla):
    print(f"Número de departamento: ", tupla[0])
    print(f"Nombre de departamento: ", tupla[1])
    print(f"Coste: ", tupla[2])
    print(f"%: ", tupla[3])
    print('-----')
```

- **def mostrarValoresEmpleados(tupla)**: Tiene el mismo objetivo que la función anterior pero, en este caso, con los empleados.

```
def mostrarValoresEmpleados(tupla):
    print(f"DNI: ", tupla[0])
    print(f"Nombre: ", tupla[1])
    print(f"Fecha Nacimiento: ", tupla[2])
    print(f"CP: ", tupla[3])
```

```
print(f"Sexo: ", tupla[4])
print(f"Suelo: ", tupla[5])
print(f"Número de departamento: ", tupla[6])
print('-----')
```

## 2. Ejercicio 1: dbObtenerEmpleados

En esta función, tal como se puede deducir, se obtendrá la información del empleado, dado su DNI de la tabla “Empleados”. A continuación, se explicará el funcionamiento del código fuente.

```
1. print("---dbObtenerEmpleados---")
2. dniObjetivo = input("Introduce dni de Empleados: ")
```

En primer lugar, se imprime por pantalla el nombre de la función a ejecutar. En este caso, “dbObtenerEmpleados”. En la 2º línea, se pide al usuario que establezca por teclado, el DNI del empleado del que quiere recuperar la información.

A continuación, se entra en un bloque try/except, debido a que existen consultas dentro de este código que pueden saltar excepciones. Y para que no se propague por todo el código, siempre es una buena práctica tener controladas todas las excepciones.

```
try:
    #Se establece la conexión con la base de datos de
    Oracle.
    cursor = conexion.cursor()
    #Se inicializa la petición a nuestra base de datos de
    Oracle. Para Oracle, a la hora de
    #realizar una petición, debemos inicializar los
    parametros de la consulta con %s
    #Esto nos permite evitar la inyección de código SQL.
    consulta = "SELECT * FROM Empleados WHERE dni = %s"

    #Se ejecuta la consulta, guardando en el segundo
    parámetro, una lista con las variables que se utilizarán
    #en la consulta. En este caso, solo hay una variable, el
    dni.
    cursor.execute(consulta, [dniObjetivo])

    #Se guarda en la variable "result" la información
    recuperada de la petición.

    resul = cursor.fetchone()
    #Se imprime por pantalla la información recuperada.
    print(resul)

    cursor.close()
    return resul
except PBD.DatabaseError as error:
```

```
        print("Error. No se ha podido consultar la tupla",
dniObjetivo)
    print(error)
```

En este bloque, su función es la de conectarse con el SGBD, inicializar la petición a realizar de dicho método, ejecutar la petición, recuperar la información y mostrarlo por pantalla. A continuación, se explicará línea por línea cada llamada.

```
cursor = conexion.cursor()

consulta = "SELECT * FROM Empleados WHERE dni = %s"
```

En esta primera línea, se guarda en la variable “cursor” la conexión con el SGBD.

En la segunda línea, se inserta en la variable “consulta”, la inicialización de la petición de nuestro método. En este caso, se trata de una consulta SELECT a la tabla Empleados, buscando en dicha tabla, la tupla con el DNI que el usuario ha pedido por teclado, anteriormente.

```
cursor.execute(consulta, [dniObjetivo])
resul = cursor.fetchone()
```

En la primera línea de este bloque, realiza la ejecución de la consulta a la base de datos y guarda la información dentro del “cursor”. La función “execute” se les pasa 2 parámetros:

1. La consulta inicializada, guardada en la variable “consulta”
2. La lista de los parámetros dinámicos, que no son fijos. En este caso, solamente se añade el DNI que el usuario pidió con anterioridad, guardada en la variable “dniObjetivo”.

Una vez se ha ejecutado y haya recuperado la información dentro de la variable cursor, se llama a la función “fetchone”, cuya función es devolver una sola tupla del total que haya podido recuperar en la consulta. En este caso, como se trata de una consulta sobre una clave primaria, que es DNI del empleado, únicamente y siempre obtendrá una única tupla. Pero existen otras funciones, que se verán más adelante, que devolverán todas o “n tuplas” de las obtenidas.

Este tupla se guarda en la variable “result”.

```
print(resul)

cursor.close()

return resul
```

En este último bloque, se imprime información de la tupla obtenida por pantalla (línea 1 de este bloque), se cierra la conexión con el SGBD (línea 2 de este bloque) y, por último,

devuelve la tupla obtenida del empleado, guardado en la variable “result” (línea 3 de este bloque).

Por lo tanto, la función quedaría de la siguiente manera:

```
def dbObtenerEmpleados():
    print("---dbObtenerEmpleados---")

    # Por ejemplo, buscar Empleados con dni 123456789
    dniObjetivo = input("Introduce dni de Empleados: ")

    try:
        #Se establece la conexión con la base de datos de
        Oracle.
        cursor = conexion.cursor()
        #Se inicializa la petición a nuestra base de datos de
        Oracle. Para Oracle, a la hora de
        #realizar una petición, debemos inicializar los
        parametros de la consulta con %s
        #Esto nos permite evitar la inyección de código SQL.
        consulta = "SELECT * FROM Empleados WHERE dni = %s"

        #Se ejecuta la consulta, guardando en el segundo
        parámetro, una lista con las variables que se utilizarán
        #en la consulta. En este caso, solo hay una variable, el
        dni.
        cursor.execute(consulta, [dniObjetivo])

        #Se guarda en la variable "result" la información
        recuperada de la petición.

        resul = cursor.fetchone()
        #Se imprime por pantalla la información recuperada.
        print(resul)

        cursor.close()
        return resul
    except PBD.DatabaseError as error:
        print("Error. No se ha podido consultar la tupla",
        dniObjetivo)
        print(error)
```

## SALIDA DEL CÓDIGO

Lo primero de todo, nos pedirá que pongamos por teclado, el DNI del empleado que se desea obtener.

```
---dbObtenerEmpleados---
Introduce dni de Empleados: 11111111
```

Posteriormente, se muestra el resultado de la información del empleado.

```
---dbObtenerEmpleados---
Introduce dni de Empleados: 11111111
DNI: 11111111

Nombre: S nchez

Fecha Nacimiento: 15-11-1997

CP: 10005

Sexo: M

Sueldo: 35000.00

N mero de departamento: None
```

### 3. EJERCICIO 2: dbConsultarEmpleados()

En esta secci n, se explicar  la implementaci n del m todo “dbConsultarEmpleados”. Este m todo devuelve TODOS LOS EMPLEADOS guardados en la tabla “Empleados” de la base de datos.

```
print("---dbConsultarEmpleados---")print("---dbConsultarEmpleados---")
```

En primer lugar, se muestra por pantalla en qu  m todo nos encontramos actualmente. En este caso, es “dbConsultarEmpleados”.

```
try:
    #Se establece la conexi n con la base de datos de
    Oracle.
    cursor = conexion.cursor()
    #Se inicializa la petici n a nuestra base de datos de
    Oracle. Para Oracle, a la hora de
    #realizar una petici n, debemos inicializar los
    parametros de la consulta con %s.
    #Esto nos permite evitar la inyecci n de c digo SQL.
    consulta = "SELECT * FROM Empleados"

    #Se ejecuta la consulta
    cursor.execute(consulta)

    #Se guarda en la variable "result" la informaci n
    recuperada de la petici n.
    #En este caso, se trata de todas las filas existentes en
    la tabla "Empleados"
    resul = cursor.fetchall()

    #Se imprime por pantalla la informaci n recuperada. En
    este caso, como se trata de una lista, se itera
    for tupla in resul:
        print(f"DNI: ",tupla[0])
        print(f"Nombre: ",tupla[1])
```

```

        print(f"Fecha Nacimiento: ", tupla[2])
        print(f"CP: ", tupla[3])
        print(f"Sexo: ", tupla[4])
        print(f"Sueldo: ", tupla[5])
        print(f"Número de departamento: ", tupla[6])
        print('-----')
    print("Número de registros recuperados:", len(resul))
    print("Número de registros
recuperados:", cursor.rowcount)
    print('-----')
    cursor.close()
except PBD.DatabaseError as error:
    print("Error. No se han podido consultar las tuplas de
Empleados")
    print(error)

```

A continuación, se entra en un bloque try/except, debido a que existen consultas dentro de este código que pueden saltar excepciones. Y para que no se propague por todo el código, siempre es una buena práctica tener controladas todas las excepciones.

En este bloque, su función es la de conectarse con el SGBD, inicializar la petición a realizar de dicho método, ejecutar la petición, recuperar la información y mostrarlo por pantalla. A continuación, se explicará línea por línea cada llamada.

```
consulta = "SELECT * FROM Empleados"
```

En la variable “consulta”, se guarda e inicializa el tipo de petición que se va a realizar a la base de datos. En este caso, un método SELECT a la tabla de “Empleados”, escogiendo todas las tuplas y variables posibles de esta tabla.

```

for tupla in resul:
    print(f"DNI: ", tupla[0])
    print(f"Nombre: ", tupla[1])
    print(f"Fecha Nacimiento: ", tupla[2])
    print(f"CP: ", tupla[3])
    print(f"Sexo: ", tupla[4])
    print(f"Sueldo: ", tupla[5])
    print(f"Número de departamento: ", tupla[6])
    print('-----')

```

Por cada tupla que se ha recuperado en la variable “resul”, se itera, guardando en cada iteración la tupla correspondiente en la variable “tupla”. En cada iteración, se muestra por pantalla la información del empleado, con su correspondiente DNI, nombre, fecha de nacimiento, código postal, sexo, sueldo y numero de departamento al que pertenece, respectivamente.

```

print("Número de registros recuperados:", len(resul))
print("Número de registros

```



```
recuperados:", cursor.rowcount)
print('-----')
```

Por último, se muestra por pantalla el número total de registros recuperados. Se puede hacer de 2 formas:

- `Len(resul)`: Te indica el número total de instancias guardadas en una lista.
- `Cursor.rowcount`: Es una variable interna del cursor, que devuelve el número total de tuplas recuperadas de la base de datos.

Por lo tanto, el código total quedaría de la siguiente forma:

```
def dbConsultarEmpleados():
    print("---dbConsultarEmpleados---")

    try:
        #Se establece la conexión con la base de datos de
        Oracle.
        cursor = conexion.cursor()
        #Se inicializa la petición a nuestra base de datos de
        Oracle. Para Oracle, a la hora de
        #realizar una petición, debemos inicializar los
        parametros de la consulta con %s.
        #Esto nos permite evitar la inyección de código SQL.
        consulta = "SELECT * FROM Empleados"

        #Se ejecuta la consulta
        cursor.execute(consulta)

        #Se guarda en la variable "result" la información
        recuperada de la petición.
        #En este caso, se trata de todas las filas existentes en
        la tabla "Empleados"
        resul = cursor.fetchall()

        #Se imprime por pantalla la información recuperada. En
        este caso, como se trata de una lista, se itera
        for tupla in resul:
            print(f"DNI: ", tupla[0])
            print(f"Nombre: ", tupla[1])
            print(f"Fecha Nacimiento: ", tupla[2])
            print(f"CP: ", tupla[3])
            print(f"Sexo: ", tupla[4])
            print(f"Suelo: ", tupla[5])
            print(f"Número de departamento: ", tupla[6])
            print('-----')
        print("Número de registros recuperados:", len(resul))
        print("Número de registros
recuperados:", cursor.rowcount)
        print('-----')
        cursor.close()
    except PBD.DatabaseError as error:
        print("Error. No se han podido consultar las tuplas de
```

```
Empleados")
    print(error)
```

## SALIDA PARCIAL DEL MÉTODO

```
---dbConsultarEmpleados---
DNI:  55555555
Nombre: Martín
Fecha Nacimiento:  11-03-1989
CP:  10005
Sexo:  M
Sueldo:  29000.00
Número de departamento:  2
-----
DNI:  66666666
Nombre: Lagos
Fecha Nacimiento:  07-07-1991
CP:  06800
Sexo:  M
Sueldo:  27000.00
Número de departamento:  2
-----
DNI:  77777777
Nombre: Salazar
Fecha Nacimiento:  22-07-1993
CP:  06300
Sexo:  H
Sueldo:  32000.00
Número de departamento:  2
-----
DNI:  88888888
Nombre: López
Fecha Nacimiento:  10-11-1994
CP:  10300
Sexo:  H
Sueldo:  32000.00
Número de departamento:  2
-----
DNI:  123456789
Nombre: Pérez
Fecha Nacimiento:  15-11-1967
CP:  06400
Sexo:  H
Sueldo:  36000.00
Número de departamento:  3
-----
DNI:  666884444
Nombre: Ojeda
Fecha Nacimiento:  12-12-1991
CP:  06300
Sexo:  H
Sueldo:  37000.00
```

Número de departamento: 3

-----  
DNI: 666999333

Nombre: Ruiz

Fecha Nacimiento: 01-02-1990

CP: 10300

Sexo: H

Sueldo: 25000.00

Número de departamento: 3

-----  
DNI: 999999999

Nombre: Simón

Fecha Nacimiento: 31-08-1988

CP: 10600

Sexo: M

Sueldo: 33000.00

Número de departamento: 3

-----  
DNI: 333445555

Nombre: Campos

Fecha Nacimiento: 12-04-1974

CP: 06002

Sexo: M

Sueldo: 50000.00

Número de departamento: 4

-----  
DNI: 222447777

Nombre: Torres

Fecha Nacimiento: 30-05-1988

CP: 10600

Sexo: H

Sueldo: 25000.00

Número de departamento: 4

-----  
DNI: 987654321

Nombre: Jiménez

Fecha Nacimiento: 10-04-1971

CP: 06400

Sexo: M

Sueldo: 40000.00

Número de departamento: 4

-----  
DNI: 000000000

Nombre: Sevilla

Fecha Nacimiento: 17-04-1980

CP: 10800

Sexo: M

Sueldo: 45000.00

Número de departamento: 4

-----  
DNI: 111111111

```

Nombre: S nchez
Fecha Nacimiento: 15-11-1997
CP: 10005
Sexo: M
Sueldo: 35000.00
N mero de departamento: None
-----
DNI: 222222222
Nombre: Mart nez
Fecha Nacimiento: 12-12-1991
CP: 06800
Sexo: M
Sueldo: 40000.00
N mero de departamento: None
-----
DNI: 333333333
Nombre:  lvarez
Fecha Nacimiento: 21-08-1990
CP: 10800
Sexo: H
Sueldo: 30000.00
N mero de departamento: None
-----
DNI: 444444444
Nombre: Gonz lez
Fecha Nacimiento: 12-09-1994
CP: 06002
Sexo: H
Sueldo: 28000.00
N mero de departamento: None
-----
N mero de registros recuperados: 16
N mero de registros recuperados: 16
-----

```

## 4. EJERCICIO 3: dbConsultarDepartamentos()

En esta secci n, se explicar  la implementaci n del m todo “dbConsultarDepartamentos”. Este m todo devuelve TODOS LOS DEPARTAMENTOS guardados en la tabla “departamentos” de la base de datos.

```
print("---dbConsultarDepartamentos---")
```

En primer lugar, se muestra por pantalla en qu  m todo nos encontramos actualmente. En este caso, es “dbConsultarEmpleados”.

```

try:
    #Se establece la conexi n con la base de datos de
Oracle.
    cursor = conexion.cursor()
    #Se inicializa la petici n a nuestra base de datos de

```

```

Oracle. Para Oracle, a la hora de
    #realizar una petición, debemos inicializar los
parametros de la consulta con :nom_propiedad.
    #Esto nos permite evitar la inyección de código SQL.
    consulta = "SELECT * FROM Departamentos"

    #Se ejecuta la consulta
    cursor.execute(consulta)

    #Se guarda en la variable "result" la información
recuperada de la petición.
    #En este caso, se trata de todas las filas existentes en
la tabla "Departamentos"
    resul = cursor.fetchall()

    #Se imprime por pantalla la información recuperada. En
este caso, como se trata de una lista, se itera
    for tupla in resul:
        mostrarValoresDepartamentos(tupla)
    print("Número de registros recuperados:", len(resul))
    print("Número de registros
recuperados:", cursor.rowcount)
    print('-----')
    cursor.close()
    except PBD.DatabaseError as error:
        print("Error. No se han podido consultar las tuplas de
Departamentos")
        print(error)

```

A continuación, se entra en un bloque try/except, debido a que existen consultas dentro de este código que pueden saltar excepciones. Y para que no se propague por todo el código, siempre es una buena práctica tener controladas todas las excepciones.

En este bloque, su función es la de conectarse con el SGBD, inicializar la petición a realizar de dicho método, ejecutar la petición, recuperar la información y mostrarlo por pantalla. A continuación, se explicará línea por línea cada llamada.

```

consulta = "SELECT * FROM Departamentos"

```

En la variable “consulta”, se guarda e inicializa el tipo de petición que se va a realizar a la base de datos. En este caso, un método SELECT a la tabla de “Departamentos”, escogiendo todas las tuplas y variables posibles de esta tabla.

```

for tupla in resul:
    mostrarValoresDepartamentos(tupla)

```

Por cada tupla que se ha recuperado en la variable “resul”, se itera, guardando en cada iteración la tupla correspondiente en la variable “tupla”. En cada iteración, se llama a la

función auxiliar “mostrarValoresDepartamentos(tupla)”, en la que se muestra el número de departamentos, nombre de departamentos, coste y porcentaje del departamento, en este orden. Se puede ver el código de esta función a continuación:

```
def mostrarValoresDepartamentos (tupla) :  
    print(f"Número de departamento: ",tupla[0])  
    print(f"Nombre de departamento: ",tupla[1])  
    print(f"Coste: ",tupla[2])  
    print(f"%: ",tupla[3])  
    print('-----')
```

```
print("Número de registros recuperados:",len(resul))  
print("Número de registros recuperados:",cursor.rowcount)  
print('-----')
```

Por último, se muestra por pantalla el número total de registros recuperados. Se puede hacer de 2 formas:

3. **Len(resul):** Te indica el número total de instancias guardadas en una lista.
  - **Cursor.rowcount:** Es una variable interna del cursor, que devuelve el número total de tuplas recuperadas de la base de datos.

Por lo tanto, el código total quedaría de la siguiente forma:

```
def dbConsultarDepartamentos () :  
    print("---dbConsultarDepartamentos---")  
  
    try:  
        #Se establece la conexión con la base de datos de  
        Oracle.  
        cursor = conexion.cursor()  
        #Se inicializa la petición a nuestra base de datos de  
        Oracle. Para Oracle, a la hora de  
        #realizar una petición, debemos inicializar los  
        parametros de la consulta con :nom_propiedad.  
        #Esto nos permite evitar la inyección de código SQL.  
        consulta = "SELECT * FROM Departamentos"  
  
        #Se ejecuta la consulta  
        cursor.execute(consulta)  
  
        #Se guarda en la variable "result" la información  
        recuperada de la petición.
```

```

        #En este caso, se trata de todas las filas existentes en
la tabla "Departamentos"
        resul = cursor.fetchall()

        #Se imprime por pantalla la información recuperada. En
este caso, como se trata de una lista, se itera
        for tupla in resul:
            mostrarValoresDepartamentos(tupla)
        print("Número de registros recuperados:", len(resul))
        print("Número de registros
recuperados:", cursor.rowcount)
        print('-----')
        cursor.close()
    except PBD.DatabaseError as error:
        print("Error. No se han podido consultar las tuplas de
Departamentos")
        print(error)

```

## SALIDA DEL CÓDIGO PARCIAL

```

---dbConsultarDepartamentos---
Número de departamento:  4
Nombre de departamento:  DESARROLLO
Coste:  0.00
%:  4.00
-----
Número de departamento:  3
Nombre de departamento:  nuevo
Coste:  0.00
%:  1.00
-----
Número de departamento:  2
Nombre de departamento:  nuevo
Coste:  0.00
%:  0.00
-----
Número de departamento:  1
Nombre de departamento:  nuevo
Coste:  0.00
%:  0.00
-----
Número de registros recuperados: 4
Número de registros recuperados: 4
-----

```

## 5. EJERCICIO 4: dbInsertarDepartamentos()

En esta sección, se explicará la implementación del método “dbInsertarEmpleados”. Este método inserta en la tabla “Departamentos” de nuestra base de datos, dados todos los parámetros necesarios por el usuario.

```

print("---dbInsertarDepartamentos---")

```

En primer lugar, se muestra por pantalla en qué método nos encontramos actualmente. En este caso, es “dblInsertarDepartamentos”.

```
try:
    cursor = conexion.cursor()

    numDpto = input("Introduce el número del departamento: ")
    nombreDpto = input("Introduce el nombre del departamento: ")
    costeDpto = input("Introduce el coste del departamento: ")
    porcentajeDpto = input("Introduce el porcentaje del departamento: ")

    consulta = "INSERT INTO Departamentos VALUES ( :numdpto, :nombredpto , :costedpto , :porcentajedpto )"

    cursor.execute(consulta, [numDpto, nombreDpto, costeDpto, porcentajeDpto])

    print("Tupla insertada correctamente")
    cursor.close()
except PBD.DatabaseError as error:
    print("Error. No se ha podido insertar el Departamento")
    print(error)
```

A continuación, se entra en un bloque try/except, debido a que existen consultas dentro de este código que pueden saltar excepciones. Y para que no se propague por todo el código, siempre es una buena práctica tener controladas todas las excepciones.

En este bloque, su función es la de conectarse con el SGBD, inicializar la petición a realizar de dicho método, ejecutar la petición, recuperar la información y mostrarlo por pantalla. A continuación, se explicará línea por línea cada llamada.

```
numDpto = input("Introduce el número del departamento: ")
nombreDpto = input("Introduce el nombre del departamento: ")
costeDpto = input("Introduce el coste del departamento: ")
porcentajeDpto = input("Introduce el porcentaje del departamento: ")
```

En este bloque de código, se obliga al usuario que nos teclee los parámetros necesarios para insertar el departamento que desea. Cada uno de ellos, se guardan en una variable (numDpto, nombreDpto, costeDpto, porcentajeDpto).

```
consulta = "INSERT INTO Departamentos VALUES ( :numdpto, :nombredpto , :costedpto , :porcentajedpto )"
```

En la variable “consulta”, se guarda e inicializa el tipo de petición que se va a realizar a la base de datos. En este caso, un método INSERT a la tabla de “Departamentos”, inicializando los valores dinámicos que se van a insertar en el apartado VALUES de la



petición. En este caso, se ha añadido el código correspondiente a Oracle, por la forma de inicializar los parámetros en la consulta.

```
print("Tupla insertada correctamente")
cursor.close()
```

Si todo va bien y no se encuentra ningún error, se muestra por pantalla que se ha realizado con éxito la inserción en la base de datos. Y por último, se cierra la conexión con la base de datos.

Por lo tanto, el código total quedaría de la siguiente forma:

```
def dbInsertarDepartamentos():
    print("----dbInsertarDepartamentos----")

    try:
        cursor = conexion.cursor()

        numDpto = input("Introduce el número del departamento: ")
        nombreDpto = input("Introduce el nombre del departamento: ")
        costeDpto = input("Introduce el coste del departamento: ")
        porcentajeDpto = input("Introduce el porcentaje del departamento: ")

        consulta = "INSERT INTO Departamentos VALUES ( :numdpto, :nombredpto , :costedpto , :porcentajedpto )"

        cursor.execute(consulta, [numDpto, nombreDpto, costeDpto, porcentajeDpto])

        print("Tupla insertada correctamente")
        cursor.close()
    except PBD.DatabaseError as error:
        print("Error. No se ha podido insertar el Departamento")
        print(error)
```

## SALIDA PARCIAL DEL MÉTODO

En primera instancia, se nos mostrará por pantalla el título del método (dbInsertarDepartamentos) y, de uno en uno, los valores necesarios para insertar el departamento. En este caso, se nos pide que introduzcamos el número, nombre, porcentaje y coste del departamento a insertar. Se subrayará en verde, los valores que se han tenido que pedir por teclado, por parte del usuario.

```
----dbInsertarDepartamentos----
Introduce el número del departamento: 8
Introduce el nombre del departamento: PROGRAMACIÓN
Introduce el coste del departamento: 9
```

Introduce el porcentaje **del** departamento: 3  
Tupla insertada correctamente

Si todo va correctamente, se muestra por pantalla que se ha podido realizar la función de forma correcta.

Si se introduce algún valor erróneo en cualquiera de los inputs, producirá un error y no se podrá realizar la inserción. Por lo que será necesario volver a repetir el método.

## 6. EJERCICIO 5: dbModificarDepartamentos()

En esta sección, se explicará la implementación del método “dbModificarDepartamentos”. Este método, modifica la tupla de la tabla “Departamentos, dado el número de departamento por el usuario por teclado, y es libre de elegir qué variables quiere modificar, ya sea el nombre, coste o porcentaje del departamento, tal como se verá más adelante.



En esta sección, se ha establecido un extra. Como se ha comentado en la introducción de este apartado, el usuario va a poder elegir el parámetro que desea modificar del departamento a elegir, dado su número de departamento. Una vez haya modificado, se le volverá a preguntar al usuario si desea realizar otra operación de modificación sobre otro departamento. No se saldrá del bucle hasta que pulse “n” (si), diciendo al código que ya no desea realizar más operaciones de modificación.

```
print ("---dbModificarDepartamentos---")
salir = False
```

En primer lugar, se muestra por pantalla en qué método nos encontramos actualmente. En este caso, es “dbConsultarEmpleados”. En la segunda línea, se inicializa la bandera “salir” a “False”, que se utilizará para controlar la salida del método.

```

try:
    cursor = conexion.cursor()
    while(not salir):
        numDpto = input("Introduce el número del
departamento a actualizar: ")
        eleccion = controlTeclado()
        # nombreDpto = input("Introduce el nuevo nombre del
departamento que quieres poner: ")
        # costeDpto = input("Introduce el coste del
departamento: ")
        # porcentajeDpto = input("Introduce el porcentaje
del departamento: ")

        match int(eleccion):
            case 1:
                nuevo_valor = input("Introduce el nuevo
nombre del departamento que quieres poner: ")
                consulta = "UPDATE departamentos SET
nombredpto = %s WHERE numerodpto = %s"
            case 2:
                nuevo_valor = input("Introduce el nuevo
coste del departamento que quieres poner: ")
                consulta = "UPDATE departamentos SET coste =
%s WHERE numerodpto = %s"
            case 3:
                nuevo_valor = input("Introduce el nuevo
porcentaje del departamento que quieres poner: ")
                consulta = "UPDATE departamentos SET porcent
= %s WHERE numerodpto = %s"

        # consulta = "UPDATE departamentos SET nombredpto =
%s WHERE numerodpto = %s"

        cursor.execute(consulta, [nuevo_valor,numDpto])

        print("Tupla actualizada correctamente")
        consulta_actualizacion = "SELECT * FROM
Departamentos WHERE numerodpto = %s"
        cursor.execute(consulta_actualizacion, [numDpto])

        resul = cursor.fetchone()
        print("----- tupla actualizada -----")
        mostrarValoresDepartamentos(resul)
        print('-----')

        salir = controlSalir()
    cursor.close()
except PBD.DatabaseError as error:
    print("Error. No se ha podido modificar el

```

```
Departamento")
    print(error)
```

A continuación, se entra en un bloque try/except, debido a que existen consultas dentro de este código que pueden saltar excepciones. Y para que no se propague por todo el código, siempre es una buena práctica tener controladas todas las excepciones.

En este bloque, su función es la de conectarse con el SGBD, inicializar la petición a realizar de dicho método, ejecutar la petición, recuperar la información y mostrarlo por pantalla. A continuación, se explicará línea por línea cada llamada.

```
while(not salir):
    numDpto = input("Introduce el número del
departamento a actualizar: ")
    eleccion = controlTeclado()
    # nombreDpto = input("Introduce el nuevo nombre del
departamento que quieres poner: ")
    # costeDpto = input("Introduce el coste del
departamento: ")
    # porcentajeDpto = input("Introduce el porcentaje
del departamento: ")

    match int(eleccion):
        case 1:
            nuevo_valor = input("Introduce el nuevo
nombre del departamento que quieres poner: ")
            consulta = "UPDATE departamentos SET
nombredpto = %s WHERE numerodpto = %s"
        case 2:
            nuevo_valor = input("Introduce el nuevo
coste del departamento que quieres poner: ")
            consulta = "UPDATE departamentos SET coste =
%s WHERE numerodpto = %s"
        case 3:
            nuevo_valor = input("Introduce el nuevo
porcentaje del departamento que quieres poner: ")
            consulta = "UPDATE departamentos SET porcent
= %s WHERE numerodpto = %s"

    # consulta = "UPDATE departamentos SET nombredpto =
%s WHERE numerodpto = %s"

    cursor.execute(consulta, [nuevo_valor,numDpto])

    print("Tupla actualizada correctamente")
    consulta_actualizacion = "SELECT * FROM
Departamentos WHERE numerodpto = %s"
    cursor.execute(consulta_actualizacion, [numDpto])
```

```

        resul = cursor.fetchone()
        print("----- tupla actualizada -----")
        mostrarValoresDepartamentos(resul)
        print('-----')

        salir = controlSalir()

```

Nos encontramos ante el bucle “while”, el cuál se repetirá hasta que el usuario no desee realizar más operaciones de modificación.

```

numDpto = input("Introduce el número del departamento a
actualizar: ")
eleccion = controlTeclado()

```

En este bloque, primero se pide al usuario que nos introduzca por teclado el número del departamento que desea modificar, y se guardará en la variable “numDpto”.

```

def controlTeclado():
    print("1- Nombre del departamento")
    print("2- Coste del departamento")
    print("3- Porcentaje del departamento")
    eleccion = input("Elige qué propiedad quieres actualizar de
este departamento. (1-3)")
    print(int(eleccion))
    while( not ( 1<= int(eleccion) <= 3) ):
        eleccion = input("Lo sentimos. Elige un valor aceptado
(1-3) ")

    return eleccion

```

A continuación, se llama al método auxiliar “controlTeclado()”, el cual volverá a preguntar al usuario que nos introzca un valor por teclado, pero en este caso, correspondiente al valor que desea modificar del departamento. Debe introducir un valor entre 1 y 3:

1. Nombre del departamento
2. Coste del departamento
3. Porcentaje del departamento.

Dicho valor se guardará en la variable “elección”. Se le volverá a repetir el input al usuario hasta que introduzca un valor correcto, es decir, un número entre el 1 y 3.

```

match int(eleccion):
    case 1:
        nuevo_valor = input("Introduce el nuevo
nombre del departamento que quieres poner: ")
        consulta = "UPDATE departamentos SET
nombredpto = %s WHERE numerodpto = %s"
    case 2:
        nuevo_valor = input("Introduce el nuevo
coste del departamento que quieres poner: ")
        consulta = "UPDATE departamentos SET coste =
%s WHERE numerodpto = %s"
    case 3:
        nuevo_valor = input("Introduce el nuevo

```

```
porcentaje del departamento que quieres poner: ")
        consulta = "UPDATE departamentos SET porcent
= %s WHERE numerodpto = %s"
```

Luego, se llega un bloque “match”, el cual, dependiendo del valor “elección”, realizará un método u otro. El valor “elección”, se debe parsear, ya que el tipo de variable que nos devuelve la función “input” es de tipo Strng.

En cada caso, se pide otra vez al usuario que nos proporcione por teclado, ahora sí, el nuevo valor que quiere introducir, y se guarda en la variable “nuevo\_valor”.

La consulta UPDATE se realiza en todos los casos sobre la tabla “Departamentos”, lo único que cambia es sobre qué atributo de la tupla se realiza (nombre, coste o porcentaje del departamento). Esta consulta establece 2 variable dinámicas: el número de departamento y para el nuevo valor sobre el atributo elegido.

```
cursor.execute(consulta, [nuevo_valor,numDpto])
print("Tupla actualizada correctamente")
```

Se ejecuta la petición, guardada en la variable “consulta” sobre la base de datos, tal como se ha comentado antes, estableciendo como variables dinámicas, el nuevo valor y el número de departamento.

Si todo va bien y no se encuentra ningún error, se muestra por pantalla que se ha realizado con éxito la actualización en la base de datos.

```
consulta_actualizacion = "SELECT * FROM Departamentos WHERE
numerodpto = %s"
cursor.execute(consulta_actualizacion, [numDpto])

resul = cursor.fetchone()
```

Para comprobar que se realizado con éxito, se realiza una petición SELECT sobre la tabla “Departamentos” del departamento que se ha modificado.

```
print("----- tupla actualizada -----")
mostrarValoresDepartamentos(resul)
print('-----')
```

En este bloque, se muestra por pantalla los valores del departamento, llamando a la función auxiliar “mostrarValoresDepartamentos(resul)”, sobre la variable “resul”, donde se ha guardado la información del departamento sobre el que se ha realizado al consulta anterior.

```
salir = controlSalir()
```

Por último, se realiza el control de salida del bucle, el cual se utiliza la variable “salir” como bandera del bucle “while”, llamando a la función auxiliar “controlSalir”.

```
def controlSalir():
    salir = input("¿Desea realizar otra operación? (S/N): ")
    while(salir.lower() != 's' and salir.lower() != 'n'):

        salir = input("Valor no valido")
    match salir.lower():
```

```

case 's':
    return False
case 'n':
    return True

```

En esta función, se pide al usuario que introduzca por teclado, si desea realizar más operaciones de modificación sobre la tabla “Departamentos”.

SOLAMENTE ES CORRECTO LOS VALORES “s” y “n”, tanto en mayúscula como en minúscula, ya que en el match, se utiliza la función “lower()”, el cual devuelve el string en minúscula.

Si es “s”, devuelve “false”, correspondiente a que quiere seguir en el bucle. Si es “n”, devolverá “True”, correspondiente a que se debe salir del bucle.

Por lo tanto, el código final quedaría de la siguiente forma:

```

def dbModificarDepartamentos():
    print("---dbModificarDepartamentos---")
    salir = False
    try:
        cursor = conexion.cursor()
        while(not salir):
            numDpto = input("Introduce el número del
departamento a actualizar: ")
            eleccion = controlTeclado()
            # nombreDpto = input("Introduce el nuevo nombre del
departamento que quieres poner: ")
            # costeDpto = input("Introduce el coste del
departamento: ")
            # porcentajeDpto = input("Introduce el porcentaje
del departamento: ")

            match int(eleccion):
                case 1:
                    nuevo_valor = input("Introduce el nuevo
nombre del departamento que quieres poner: ")
                    consulta = "UPDATE departamentos SET
nombredpto = %s WHERE numerodpto = %s"
                case 2:
                    nuevo_valor = input("Introduce el nuevo
coste del departamento que quieres poner: ")
                    consulta = "UPDATE departamentos SET coste =
%s WHERE numerodpto = %s"
                case 3:
                    nuevo_valor = input("Introduce el nuevo
porcentaje del departamento que quieres poner: ")
                    consulta = "UPDATE departamentos SET porcent
= %s WHERE numerodpto = %s"

```



```

        # consulta = "UPDATE departamentos SET nombredpto =
%s WHERE numerodpto = %s"

        cursor.execute(consulta, [nuevo_valor,numDpto])

        print("Tupla actualizada correctamente")
        consulta_actualizacion = "SELECT * FROM
Departamentos WHERE numerodpto = %s"
        cursor.execute(consulta_actualizacion, [numDpto])

        resul = cursor.fetchone()
        print("----- tupla actualizada -----")
        mostrarValoresDepartamentos(resul)
        print('-----')

        salir = controlSalir()
        cursor.close()
    except PBD.DatabaseError as error:
        print("Error. No se ha podido modificar el
Departamento")
        print(error)

```

## SALIDA PARCIAL DEL MÉTODO

En una primera instancia, se nos mostrará el nombre del método en el que nos encontramos, y posteriormente, el número del departamento a modificar. Se aclara que se nos **mostrará con verde**, aquellos valores dados por teclado.

```

---dbModificarDepartamentos---
Introduce el número del departamento a actualizar:

```

Una vez elijamos un número, se nos mostrará las opciones que tenemos para modificar. Se deberá elegir un número entre el 1 y el 3, refiriéndose a la modificación del nombre, coste o el porcentaje del departamento.

```

Introduce el número del departamento a actualizar: 3
1- Nombre del departamento
2- Coste del departamento
3- Porcentaje del departamento
Elige qué propiedad quieres actualizar de este departamento. (1-
3)

```

Dependiendo del valor elegido, se nos mostrará un texto, referente a que introduzcamos el nuevo valor del atributo elegido. En este caso, se ha elegido modificar el nombre del departamento, eligiendo el valor 1.

```

---dbModificarDepartamentos---

```

```

Introduce el número del departamento a actualizar: 3
1- Nombre del departamento
2- Coste del departamento
3- Porcentaje del departamento
Elige qué propiedad quieres actualizar de este departamento. (1-3)1
1
Introduce el nuevo nombre del departamento que quieres poner:

```

A continuación, se debe introducir el nuevo valor del atributo. En este caso, se ha decidido poner el valor “PYTHON”.

```

Introduce el nuevo nombre del departamento que quieres poner:
PYTHON
Tupla actualizada correctamente
----- tupla actualizada -----
Número de departamento: 3
Nombre de departamento: PYTHON
Coste: 0.00
%: 1.00
-----
-----
¿Desea realizar otra operación? (S/N):

```

Una vez dado al ENTER, se ejecuta la petición a la base de datos. Si va todo bien, se debe aparecer en pantalla algo parecido al bloque anterior de código, mostrándonos por pantalla la información del departamento modificado.

Por último, se nos pedirá si se desea realizar otra operación de modificación de otro departamento. Se debe establecer un valor de “s” o “n”, independientemente de si es en mayúscula o minúscula. Si se da otro valor diferente, os volverá a pedir un nuevo valor. Seguirá así hasta que no le proporcionéis un valor correcto.

```

¿Desea realizar otra operación? (S/N): n
Valor no valido
n
Valor no valido
n
Valor no valido
N

```

## 7. EJERCICIO 6: dbBorrarDepartamentos()

En esta sección, se explicará la implementación del método “dbBorrarDepartamentos”. Este método ELIMINA EL DEPARTAMENTO, de la tabla “Departamento” de nuestra base de datos, elegido por el usuario, dando el número del departamento por teclado.

```

print ("---dbBorrarDepartamentos---")

```

En primer lugar, se muestra por pantalla en qué método nos encontramos actualmente. En este caso, es “dbBorrarDepartamentos”.

```
try:
    cursor = conexion.cursor()

    numDpto = input("Introduce el número del departamento a
actualizar: ")
    #nombreDpto = input("Introduce el nuevo nombre del
departamento que quieres poner: ")
    # costeDpto = input("Introduce el coste del departamento:
")
    # porcentajeDpto = input("Introduce el porcentaje del
departamento: ")

    consulta = "DELETE FROM departamentos WHERE numerodpto =
%s"

    cursor.execute(consulta, [numDpto])

    print("Tupla borrada correctamente")
    cursor.close()
except PBD.DatabaseError as error:
    print("Error. No se ha podido borrar el Departamento")
    print(error)
```

A continuación, se entra en un bloque try/except, debido a que existen consultas dentro de este código que pueden saltar excepciones. Y para que no se propague por todo el código, siempre es una buena práctica tener controladas todas las excepciones.

En este bloque, su función es la de conectarse con el SGBD, inicializar la petición a realizar de dicho método, ejecutar la petición, recuperar la información y mostrarlo por pantalla. A continuación, se explicará línea por línea cada llamada.

```
numDpto = input("Introduce el número del departamento a borrar:
")
```

Se pide al usuario que introduzca el número de departamento a borrar, y se guarda en la variable “numDpto”.

```
consulta = "DELETE FROM departamentos WHERE numerodpto = %s"
```

En la variable “consulta”, se guarda e inicializa el tipo de petición que se va a realizar a la base de datos. En este caso, un método DELETE FROM a la tabla de “Empleados”, escogiendo la tupla que cumpla con la condición de que “numerodpto = %s”, el cual “%s” será el valor introducido por el usuario.

```
cursor.execute(consulta, [numDpto])
```

```
print("Tupla borrada correctamente")
cursor.close()
```

Se ejecuta la petición, guardada en la variable “consulta”, inicializando como parámetro dinámico el valor guardado en “numDpto”. Posteriormente, se informa por pantalla el éxito de la función y se desconecta de la base de datos.

## SALIDA PARCIAL DEL MÉTODO

En primera instancia, se nos mostrará por pantalla el título del método (dbBorrarDepartamentos) y se pedirá por teclado el número de departamento a eliminar. **Se subrayará en verde**, los valores que se han tenido que pedir por teclado, por parte del usuario.

```
---dbBorrarDepartamentos---
Introduce el número del departamento a borrar:
```

Si todo va correctamente, se muestra por pantalla que se ha podido realizar la función de forma correcta.

```
---dbBorrarDepartamentos---
Introduce el número del departamento a borrar: 8
Tupla borrada correctamente
```

## 7. EJERCICIO 7:

### dbInsertarMultiplesDepartamentos()

En esta sección, se explicará la implementación del método “dbConsultarEmpleados”. Este método INSERTA UNA LISTA DE DEPARTAMENTOS PREDEFINIDOS en la tabla “Departamentos” de la base de datos.

```
print("---dbInsertarMultiplesDepartamentos---")
```

En primer lugar, se muestra por pantalla en qué método nos encontramos actualmente. En este caso, es “dbInsertarMultiplesDepartamentos”.

```
datos = [
    ('5', 'INVESTIGACIÓN', 0.0, 0.0),
    ('6', 'MARKETING', 0.0, 0.0),
    ('7', 'VENTAS', 0.0, 0.0)
]
```

Se inicializan en la variable “datos”, la lista con la información de los departamentos que se van a insertar. En este caso, se insertarán un total de 3 departamentos.

Se debería modificar dicha lista si se desea inserta más departamentos o modificar sus valores.

```
8. try:
    cursor = conexion.cursor()

    for fila in datos:
```

```

        consulta = "INSERT INTO Departamentos VALUES (
%s, %s , %s , %s)"

        cursor.execute(consulta, [fila[0], fila[1],
fila[2], fila[3]])

        print("Tupla insertada correctamente")
        cursor.close()
    except PBD.DatabaseError as error:
        print("Error. No se han podido insertar múltiples
Departamentos")
        print(error)
        cursor.close()

```

A continuación, se entra en un bloque try/except, debido a que existen consultas dentro de este código que pueden saltar excepciones. Y para que no se propague por todo el código, siempre es una buena práctica tener controladas todas las excepciones.

En este bloque, su función es la de conectarse con el SGBD, inicializar la petición a realizar de dicho método, ejecutar la petición, recuperar la información y mostrarlo por pantalla. A continuación, se explicará línea por línea cada llamada.

```

for fila in datos:
    consulta = "INSERT INTO Departamentos VALUES ( %s,
%s , %s , %s)"

    cursor.execute(consulta, [fila[0], fila[1], fila[2],
fila[3]])

    print("Tupla insertada correctamente")

```

En dicho bucle, se itera nuestra lista de departamentos (variable “datos”), y por cada fila, se:

- Inicializa en la variable “consulta”, la petición a realizar. En este caso, un INSERT a la tabla “Departamentos”, estableciendo los valores dinámicos con %s.
- Se ejecuta la petición guardada en la variable “consulta”, configurando como valores dinámicos, la información guardada en la variable “tupla”, la cual se corresponde con la información de cada fila de la lista de departamentos
- Se imprime por pantalla que la petición a la base de datos se ha completado con éxito.

Por lo tanto, el código total quedaría de la siguiente forma:

```

def dbInsertarMultiplesDepartamentos() :
    print("---dbInsertarMultiplesDepartamentos---")

    datos = [
        ('5', 'INVESTIGACIÓN', 0.0, 0.0),
        ('6', 'MARKETING', 0.0, 0.0),
        ('7', 'VENTAS', 0.0, 0.0)
    ]

```

```

    try:
        cursor = conexion.cursor()

        for fila in datos:
            consulta = "INSERT INTO Departamentos VALUES ( %s,
%s , %s , %s)"

            cursor.execute(consulta, [fila[0], fila[1], fila[2],
fila[3]])

            print("Tupla insertada correctamente")
        cursor.close()

```

## SALIDA PARCIAL DEL MÉTODO

```

---dbInsertarMultiplesDepartamentos---
Tupla insertada correctamente
Tupla insertada correctamente
Tupla insertada correctamente
---dbConsultarDepartamentos---

```

## 8. EJERCICIO 8:

### dbBorrarMultiplesDepartamentos():

En esta sección, se explicará la implementación del método “dbBorrarMultiplesDepartamentos”. Este método BORRA UNA LISTA DE DEPARTAMENTOS, dados de forma predefinida sus números de departamentos.

```

print('---dbBorrarMultiplesDepartamentos---')

datos = [['5'], ['6'], ['7']]

```

En primer lugar, se muestra por pantalla en qué método nos encontramos actualmente. En este caso, es “dbBorrarMultiplesDepartamentos”. A continuación, se inicializa la variable “datos”, que guarda los número de id de los departamentos que se borrarán en estas función.

```

try:
    cursor = conexion.cursor()

    for fila in datos:
        consulta = "DELETE FROM departamentos WHERE
numerodpto = %s"

        cursor.execute(consulta, [fila[0]])

```

```

        print("Tupla borrada correctamente")
    cursor.close()
except PBD.DatabaseError as error:
    print("Error. No se han podido borrar múltiples
Departamentos")
    print(error)
    cursor.close()

```

A continuación, se entra en un bloque try/except, debido a que existen consultas dentro de este código que pueden saltar excepciones. Y para que no se propague por todo el código, siempre es una buena práctica tener controladas todas las excepciones.

En este bloque, su función es la de conectarse con el SGBD, inicializar la petición a realizar de dicho método, ejecutar la petición, recuperar la información y mostrarlo por pantalla. A continuación, se explicará línea por línea cada llamada.

```

for fila in datos:
    consulta = "DELETE FROM departamentos WHERE numerodpto =
%s"

    cursor.execute(consulta, [fila[0]])
    print("Tupla borrada correctamente")

```

En dicho bucle, se itera nuestra lista de IDs de departamentos (variable “datos”), y por cada fila, se:

- Inicializa en la variable “consulta”, la petición a realizar. En este caso, un DELETE a la tabla “Departamentos”, estableciendo los valores dinámicos con %s.
- Se ejecuta la petición guardada en la variable “consulta”, configurando como valores dinámicos, la información guardada en la variable “fila”, la cual se corresponde con la información de cada fila de la lista de departamentos
- Se imprime por pantalla que la petición a la base de datos se ha completado con éxito.

Por lo tanto, el código total quedaría de la siguiente forma:

```

def dbBorrarMultiplesDepartamentos():
    print('---dbBorrarMultiplesDepartamentos---')

    datos = [['5'], ['6'], ['7']]

    try:
        cursor = conexion.cursor()

        for fila in datos:
            consulta = "DELETE FROM departamentos WHERE
numerodpto = %s"

            cursor.execute(consulta, [fila[0]])
            print("Tupla borrada correctamente")

```

```
        cursor.close()
    except PBD.DatabaseError as error:
        print("Error. No se han podido borrar múltiples
Departamentos")
        print(error)
        cursor.close()
```

## SALIDA PARCIAL DEL MÉTODO

```
---dbBorrarMultiplesDepartamentos---
Tupla borrada correctamente
Tupla borrada correctamente
Tupla borrada correctamente
```





## 8. EXTRA POINT

De forma opcional, se ha desarrollado 2 nuevas funciones sobre la base de datos, que el profesor no ha pedido.

1. `dbMostrarEmpleadosDeCadaDepartamento()`: Este método muestra por pantalla, todos los departamentos, mostrando por consola, su número y nombre de departamento y, además, a continuación, todos los empleados pertenecientes a este departamento, con el formato de DNI, nombre, sueldo y número de departamento al que pertenece.

```
def dbMostrarEmpleadosDeCadaDepartamento():
    print("----dbMostrarEmpleadosDeCadaDepartamento----")

    try:
        #Se establece la conexión con la base de datos de
        Oracle.
        cursor = conexion.cursor()
        #Se inicializa la petición a nuestra base de datos de
        Oracle. Para Oracle, a la hora de
        #realizar una petición, debemos inicializar los
        parametros de la consulta con :nom_propiedad.
        #Esto nos permite evitar la inyección de código SQL.
        consulta = "SELECT numerodpto,nombredpto FROM
        Departamentos"

        #Se ejecuta la consulta
        cursor.execute(consulta)

        #Se guarda en la variable "result" la información
        recuperada de la petición.
        #En este caso, se trata de todas las filas existentes en
        la tabla "Departamentos"
        resul = cursor.fetchall()

        #Se imprime por pantalla la información recuperada. En
        este caso, como se trata de una lista, se itera
        for tupla in resul:
            print(f"Número de departamento: ",tupla[0])
            print(f"Nombre de departamento: ",tupla[1])

            dbConsultarEmpleadosMismoDepartamento (tupla[0])

            print('-----')

        print('-----')
        cursor.close()
    except PBD.DatabaseError as error:
        print("Error. No se han podido consultar las tuplas de
        Departamentos")
        print(error)
```

## SALIDA PARCIAL DEL MÉTODO

```
---dbMostrarEmpleadosDeCadaDepartamento---
Número de departamento:  2
Nombre de departamento:  PRODUCCIÓN
---dbConsultarEmpleadosMismoDepartamento(2)---
DNI:  55555555
Nombre:  Martín
Sueldo:  29000.0
Número de departamento:  2
-----
DNI:  66666666
Nombre:  Lagos
Sueldo:  27000.0
Número de departamento:  2
-----
DNI:  77777777
Nombre:  Salazar
Sueldo:  32000.0
Número de departamento:  2
-----
DNI:  88888888
Nombre:  López
Sueldo:  32000.0
Número de departamento:  2
-----
-----
Número de departamento:  1
Nombre de departamento:  ARQUITECTOS
---dbConsultarEmpleadosMismoDepartamento(1)---
DNI:  11111111
Nombre:  Sánchez
Sueldo:  35000.0
Número de departamento:  1
-----
DNI:  22222222
Nombre:  Martínez
Sueldo:  40000.0
Número de departamento:  1
-----
DNI:  33333333
Nombre:  Álvarez
Sueldo:  30000.0
Número de departamento:  1
-----
DNI:  44444444
Nombre:  González
Sueldo:  28000.0
Número de departamento:  1
```

```
-----  
-----  
Número de departamento: 3  
Nombre de departamento: DISEÑO  
---dbConsultarEmpleadosMismoDepartamento(3)---  
DNI: 123456789  
Nombre: Pérez  
Sueldo: 36000.0  
Número de departamento: 3  
-----  
DNI: 666884444  
Nombre: Ojeda  
Sueldo: 37000.0  
Número de departamento: 3  
-----  
DNI: 666999333  
Nombre: Ruiz  
Sueldo: 25000.0  
Número de departamento: 3  
-----  
DNI: 999999999  
Nombre: Simón  
Sueldo: 33000.0  
Número de departamento: 3  
-----  
-----  
Número de departamento: 4  
Nombre de departamento: DESARROLLO  
---dbConsultarEmpleadosMismoDepartamento(4)---  
DNI: 333445555  
Nombre: Campos  
Sueldo: 50000.0  
Número de departamento: 4  
-----  
DNI: 222447777  
Nombre: Torres  
Sueldo: 25000.0  
Número de departamento: 4  
-----  
DNI: 987654321  
Nombre: Jiménez  
Sueldo: 40000.0  
Número de departamento: 4  
-----  
DNI: 000000000  
Nombre: Sevilla  
Sueldo: 45000.0  
Número de departamento: 4  
-----  
-----  
Número de departamento: 9
```

```
Nombre de departamento:  FINANZAS
---dbConsultarEmpleadosMismoDepartamento(9)---
-----
-----
```

2. `dbConsultarEmpleadosMismoDepartamentoExamen(idDep)`: Este método se trata de una función auxiliar al anterior. Se trata del método que, dado el número de departamento, muestra por pantalla, los atributos de DNI, nombre, sueldo y número de departamento de todos los empleados pertenecientes a número de departamento, dado por parámetros.

```
def dbConsultarEmpleadosMismoDepartamento(idDep):
    print("---dbConsultarEmpleadosMismoDepartamento---")
    try:
        cursor = conexion.cursor()
        consulta = "SELECT dni, nombre, sueldo, numdep FROM
Empleados WHERE numdep = :idDep"

        cursor.execute(consulta, [idDep])
        resul = cursor.fetchall()

        for tupla in resul:
            print(f"DNI: ",tupla[0])
            print(f"Nombre: ",tupla[1])
            print(f"Sueldo: ",tupla[2])
            print(f"Número de departamento: ",tupla[3])
            print('-----')

    except PBD.DatabaseError as error:
        print("Error. No se han podido recuperar los múltiples
empleados del Departamento")
        print(error)
```

## SALIDA PARCIAL DEL CÓDIGO, DADO COMO PARÁMETRO DE ENTRADA EL 4.

```
---dbConsultarEmpleadosMismoDepartamento---
DNI:  333445555
Nombre:  Campos
Sueldo:  50000.0
Número de departamento:  4
-----
DNI:  222447777
Nombre:  Torres
Sueldo:  25000.0
```

```

Número de departamento:  4
-----
DNI:  987654321
Nombre:  Jimenez
Sueldo:  40000.0
Número de departamento:  4
-----
DNI:  000000000
Nombre:  Sevilla
Sueldo:  45000.0
Número de departamento:  4
-----

```

## 8. SALIDA COMPLETA POR CONSOLA DE POSTGRESQL

---Programa principal---

---dbConectar---

---Conectando a Postgresql---

Conexión realizada a la base de datos <connection object at 0x000001A6B8B41690; dsn: 'user=postgres password=xxx dbname=Empresa host=localhost port=5432', closed: 0>

CONEXIÓN REALIZADA

---dbMostrarEmpleados1---

```

('555555555', 'Martinez', '11-03-1989', '10005', 'M', Decimal('29000.00'), Decimal('2'))
('666666666', 'Lagos', '07-07-1991', '06800', 'M', Decimal('27000.00'), Decimal('2'))
('777777777', 'Salazar', '22-07-1993', '06300', 'H', Decimal('32000.00'), Decimal('2'))
('888888888', 'Lopez', '10-11-1994', '10300', 'H', Decimal('32000.00'), Decimal('2'))
('123456789', 'Perez', '15-11-1967', '06400', 'H', Decimal('36000.00'), Decimal('3'))
('666884444', 'Ojeda', '12-12-1991', '06300', 'H', Decimal('37000.00'), Decimal('3'))
('666999333', 'Ruiz', '01-02-1990', '10300', 'H', Decimal('25000.00'), Decimal('3'))
('999999999', 'Simon', '31-08-1988', '10600', 'M', Decimal('33000.00'), Decimal('3'))
('333445555', 'Campos', '12-04-1974', '06002', 'M', Decimal('50000.00'), Decimal('4'))
('222447777', 'Torres', '30-05-1988', '10600', 'H', Decimal('25000.00'), Decimal('4'))
('987654321', 'Jimenez', '10-04-1971', '06400', 'M', Decimal('40000.00'), Decimal('4'))
('000000000', 'Sevilla', '17-04-1980', '10800', 'M', Decimal('45000.00'), Decimal('4'))

```

('111111111', 'Sánchez', '15-11-1997', '10005', 'M', Decimal('35000.00'), None)  
('222222222', 'Martínez', '12-12-1991', '06800', 'M', Decimal('40000.00'), None)  
('333333333', 'Alvarez', '21-08-1990', '10800', 'H', Decimal('30000.00'), None)  
('444444444', 'González', '12-09-1994', '06002', 'H', Decimal('28000.00'), None)

Número de registros recuperados: 16

-----

---dbMostrarEmpleados2---

('555555555', 'Martín', '11-03-1989', '10005', 'M', Decimal('29000.00'), Decimal('2'))  
('666666666', 'Lagos', '07-07-1991', '06800', 'M', Decimal('27000.00'), Decimal('2'))  
('777777777', 'Salazar', '22-07-1993', '06300', 'H', Decimal('32000.00'), Decimal('2'))  
('888888888', 'López', '10-11-1994', '10300', 'H', Decimal('32000.00'), Decimal('2'))  
('123456789', 'Pérez', '15-11-1967', '06400', 'H', Decimal('36000.00'), Decimal('3'))  
('666884444', 'Ojeda', '12-12-1991', '06300', 'H', Decimal('37000.00'), Decimal('3'))  
('666999333', 'Ruiz', '01-02-1990', '10300', 'H', Decimal('25000.00'), Decimal('3'))  
('999999999', 'Simón', '31-08-1988', '10600', 'M', Decimal('33000.00'), Decimal('3'))  
('333445555', 'Campos', '12-04-1974', '06002', 'M', Decimal('50000.00'), Decimal('4'))  
('222447777', 'Torres', '30-05-1988', '10600', 'H', Decimal('25000.00'), Decimal('4'))  
('987654321', 'Jiménez', '10-04-1971', '06400', 'M', Decimal('40000.00'), Decimal('4'))  
('000000000', 'Sevilla', '17-04-1980', '10800', 'M', Decimal('45000.00'), Decimal('4'))  
('111111111', 'Sánchez', '15-11-1997', '10005', 'M', Decimal('35000.00'), None)  
('222222222', 'Martínez', '12-12-1991', '06800', 'M', Decimal('40000.00'), None)  
('333333333', 'Alvarez', '21-08-1990', '10800', 'H', Decimal('30000.00'), None)  
('444444444', 'González', '12-09-1994', '06002', 'H', Decimal('28000.00'), None)

Número de registros recuperados: 16

-----

---dbMostrarEmpleados3---

('555555555', 'Martín', '11-03-1989', '10005', 'M', Decimal('29000.00'), Decimal('2'))  
('666666666', 'Lagos', '07-07-1991', '06800', 'M', Decimal('27000.00'), Decimal('2'))  
('777777777', 'Salazar', '22-07-1993', '06300', 'H', Decimal('32000.00'), Decimal('2'))  
('888888888', 'López', '10-11-1994', '10300', 'H', Decimal('32000.00'), Decimal('2'))  
('123456789', 'Pérez', '15-11-1967', '06400', 'H', Decimal('36000.00'), Decimal('3'))

Número de registros seleccionados: 5

Número de registros recuperados: 16

-----

---dbMostrarEmpleados4---

('555555555', 'Martín', '11-03-1989', '10005', 'M', Decimal('29000.00'), Decimal('2'))  
('666666666', 'Lagos', '07-07-1991', '06800', 'M', Decimal('27000.00'), Decimal('2'))  
('777777777', 'Salazar', '22-07-1993', '06300', 'H', Decimal('32000.00'), Decimal('2'))  
('888888888', 'López', '10-11-1994', '10300', 'H', Decimal('32000.00'), Decimal('2'))  
('123456789', 'Pérez', '15-11-1967', '06400', 'H', Decimal('36000.00'), Decimal('3'))  
('666884444', 'Ojeda', '12-12-1991', '06300', 'H', Decimal('37000.00'), Decimal('3'))  
('666999333', 'Ruiz', '01-02-1990', '10300', 'H', Decimal('25000.00'), Decimal('3'))  
('999999999', 'Simón', '31-08-1988', '10600', 'M', Decimal('33000.00'), Decimal('3'))  
('333445555', 'Campos', '12-04-1974', '06002', 'M', Decimal('50000.00'), Decimal('4'))  
('222447777', 'Torres', '30-05-1988', '10600', 'H', Decimal('25000.00'), Decimal('4'))  
('987654321', 'Jiménez', '10-04-1971', '06400', 'M', Decimal('40000.00'), Decimal('4'))  
('000000000', 'Sevilla', '17-04-1980', '10800', 'M', Decimal('45000.00'), Decimal('4'))  
('111111111', 'Sánchez', '15-11-1997', '10005', 'M', Decimal('35000.00'), None)  
('222222222', 'Martínez', '12-12-1991', '06800', 'M', Decimal('40000.00'), None)  
('333333333', 'Álvarez', '21-08-1990', '10800', 'H', Decimal('30000.00'), None)  
('444444444', 'González', '12-09-1994', '06002', 'H', Decimal('28000.00'), None)

Número de registros recuperados: 16

Número de registros recuperados: 16

-----

---dbObtenerEmpleados---

Introduce dni de Empleados: 111111111

DNI: 111111111

Nombre: Sánchez

Fecha Nacimiento: 15-11-1997

CP: 10005

Sexo: M

Sueldo: 35000.00

Número de departamento: None

-----

---dbConsultarEmpleados---

DNI: 555555555

Nombre: Martín

Fecha Nacimiento: 11-03-1989

CP: 10005

Sexo: M

Sueldo: 29000.00

Número de departamento: 2

-----

DNI: 666666666

Nombre: Lagos

Fecha Nacimiento: 07-07-1991

CP: 06800

Sexo: M

Sueldo: 27000.00

Número de departamento: 2

-----

DNI: 777777777

Nombre: Salazar

Fecha Nacimiento: 22-07-1993

CP: 06300

Sexo: H

Sueldo: 32000.00

Número de departamento: 2

-----

DNI: 888888888

Nombre: López

Fecha Nacimiento: 10-11-1994

CP: 10300



Sexo: H

Sueldo: 32000.00

Número de departamento: 2

-----

DNI: 123456789

Nombre: Pérez

Fecha Nacimiento: 15-11-1967

CP: 06400

Sexo: H

Sueldo: 36000.00

Número de departamento: 3

-----

DNI: 666884444

Nombre: Ojeda

Fecha Nacimiento: 12-12-1991

CP: 06300

Sexo: H

Sueldo: 37000.00

Número de departamento: 3

-----

DNI: 666999333

Nombre: Ruiz

Fecha Nacimiento: 01-02-1990

CP: 10300

Sexo: H

Sueldo: 25000.00

Número de departamento: 3

-----

DNI: 999999999

Nombre: Simón

Fecha Nacimiento: 31-08-1988

CP: 10600

Sexo: M

Sueldo: 33000.00

Número de departamento: 3

-----

DNI: 333445555

Nombre: Campos

Fecha Nacimiento: 12-04-1974

CP: 06002

Sexo: M

Sueldo: 50000.00

Número de departamento: 4

-----

DNI: 222447777

Nombre: Torres

Fecha Nacimiento: 30-05-1988

CP: 10600

Sexo: H

Sueldo: 25000.00

Número de departamento: 4

-----

DNI: 987654321

Nombre: Jiménez

Fecha Nacimiento: 10-04-1971

CP: 06400

Sexo: M

Sueldo: 40000.00

Número de departamento: 4

-----

DNI: 000000000

Nombre: Sevilla

Fecha Nacimiento: 17-04-1980

CP: 10800

Sexo: M

Sueldo: 45000.00

Número de departamento: 4

-----

DNI: 111111111

Nombre: S◆nchez

Fecha Nacimiento: 15-11-1997

CP: 10005

Sexo: M

Sueldo: 35000.00

Número de departamento: None

-----

DNI: 222222222

Nombre: Mart◆nez

Fecha Nacimiento: 12-12-1991

CP: 06800

Sexo: M

Sueldo: 40000.00

Número de departamento: None

-----

DNI: 333333333

Nombre: ◆lvarez

Fecha Nacimiento: 21-08-1990

CP: 10800

Sexo: H

Sueldo: 30000.00

Número de departamento: None

-----

DNI: 444444444

Nombre: Gonz lez

Fecha Nacimiento: 12-09-1994

CP: 06002

Sexo: H

Sueldo: 28000.00

N mero de departamento: None

-----

N mero de registros recuperados: 16

N mero de registros recuperados: 16

-----

---dbConsultarDepartamentos---

N mero de departamento: 4

Nombre de departamento: DESARROLLO

Coste: 0.00

?: 4.00

-----

N mero de departamento: 3

Nombre de departamento: nuevo

Coste: 0.00

?: 1.00

-----

N mero de departamento: 1

Nombre de departamento: nuevo

Coste: 0.00

?: 0.00

-----

N mero de departamento: 2

Nombre de departamento: nuevo hola

Coste: 8.00

?: 0.00

-----

Número de registros recuperados: 4

Número de registros recuperados: 4

-----

---dbInsertarDepartamentos---

Introduce el número del departamento: 9

Introduce el nombre del departamento: emprendimiento

Introduce el coste del departamento: 9

Introduce el porcentaje del departamento: 9

Tupla insertada correctamente

---dbConsultarDepartamentos---

Número de departamento: 4

Nombre de departamento: DESARROLLO

Coste: 0.00

?: 4.00

-----

Número de departamento: 3

Nombre de departamento: nuevo

Coste: 0.00

?: 1.00

-----

Número de departamento: 1

Nombre de departamento: nuevo

Coste: 0.00

?: 0.00

-----

Número de departamento: 2

Nombre de departamento: nuevo hola

Coste: 8.00

?: 0.00

-----

Número de departamento: 9

Nombre de departamento: emprendimiento

Coste: 9.00

?: 9.00

-----

Número de registros recuperados: 5

Número de registros recuperados: 5

-----

---dbModificarDepartamentos---

Introduce el número del departamento a actualizar: 9

1- Nombre del departamento

2- Coste del departamento

3- Porcentaje del departamento

Elige qué propiedad quieres actualizar de este departamento. (1-3)1

1

Introduce el nuevo nombre del departamento que quieres poner: modificado

Tupla actualizada correctamente

----- tupla actualizada -----

Número de departamento: 9

Nombre de departamento: modificado

Coste: 9.00

?: 9.00

-----

-----

¿Desea realizar otra operación? (S/N): n

---dbConsultarDepartamentos---

Número de departamento: 4

Nombre de departamento: DESARROLLO

Coste: 0.00

?: 4.00

-----

Número de departamento: 3

Nombre de departamento: nuevo

Coste: 0.00

?: 1.00

-----

Número de departamento: 1

Nombre de departamento: nuevo

Coste: 0.00

?: 0.00

-----

Número de departamento: 2

Nombre de departamento: nuevo hola

Coste: 8.00

?: 0.00

-----

Número de departamento: 9

Nombre de departamento: modificado

Coste: 9.00

?: 9.00

-----

Número de registros recuperados: 5

Número de registros recuperados: 5

-----

---dbBorrarDepartamentos---

Introduce el número del departamento a borrar: 9

Tupla borrada correctamente

---dbConsultarDepartamentos---

Número de departamento: 4

Nombre de departamento: DESARROLLO

Coste: 0.00

?: 4.00

-----

Número de departamento: 3

Nombre de departamento: nuevo

Coste: 0.00

?: 1.00

-----

Número de departamento: 1

Nombre de departamento: nuevo

Coste: 0.00

?: 0.00

-----

Número de departamento: 2

Nombre de departamento: nuevo hola

Coste: 8.00

?: 0.00

-----

Número de registros recuperados: 4

Número de registros recuperados: 4

-----

---dbInsertarMultiplesDepartamentos---

Tupla insertada correctamente

Tupla insertada correctamente

Tupla insertada correctamente

---dbConsultarDepartamentos---

Número de departamento: 4

Nombre de departamento: DESARROLLO

Coste: 0.00

?: 4.00

-----

Número de departamento: 3

Nombre de departamento: nuevo

Coste: 0.00



?: 1.00

-----

Número de departamento: 1

Nombre de departamento: nuevo

Coste: 0.00

?: 0.00

-----

Número de departamento: 2

Nombre de departamento: nuevo hola

Coste: 8.00

?: 0.00

-----

Número de departamento: 5

Nombre de departamento: INVESTIGACIÓN

Coste: 0.00

?: 0.00

-----

Número de departamento: 6

Nombre de departamento: MARKETING

Coste: 0.00

?: 0.00

-----

Número de departamento: 7

Nombre de departamento: VENTAS

Coste: 0.00

?: 0.00

-----

Número de registros recuperados: 7

Número de registros recuperados: 7

-----

---dbBorrarMultiplesDepartamentos---

Tupla borrada correctamente

Tupla borrada correctamente

Tupla borrada correctamente

---dbConsultarDepartamentos---

Número de departamento: 4

Nombre de departamento: DESARROLLO

Coste: 0.00

?: 4.00

-----

Número de departamento: 3

Nombre de departamento: nuevo

Coste: 0.00

?: 1.00

-----

Número de departamento: 1

Nombre de departamento: nuevo

Coste: 0.00

?: 0.00

-----

Número de departamento: 2

Nombre de departamento: nuevo hola

Coste: 8.00

?: 0.00

-----

Número de registros recuperados: 4

Número de registros recuperados: 4

---

---dbMostrarEmpleadosDeCadaDepartamento---

Número de departamento: 2

Nombre de departamento: PRODUCCI?N

---dbConsultarEmpleadosMismoDepartamento(2)---

DNI: 555555555

Nombre: Martín

Sueldo: 29000.0

Número de departamento: 2

-----

DNI: 666666666

Nombre: Lagos

Sueldo: 27000.0

Número de departamento: 2

-----

DNI: 777777777

Nombre: Salazar

Sueldo: 32000.0

Número de departamento: 2

-----

DNI: 888888888

Nombre: López

Sueldo: 32000.0

Número de departamento: 2

-----

-----

Número de departamento: 1

Nombre de departamento: ARQUITECTOS

---dbConsultarEmpleadosMismoDepartamento(1)---

DNI: 111111111

Nombre: Sánchez

Sueldo: 35000.0

Número de departamento: 1

-----

DNI: 222222222

Nombre: Martínez

Sueldo: 40000.0

Número de departamento: 1

-----

DNI: 33333333

Nombre: ♦lvarez

Sueldo: 30000.0

Número de departamento: 1

-----

DNI: 44444444

Nombre: Gonz♦lez

Sueldo: 28000.0

Número de departamento: 1

-----

-----

Número de departamento: 3

Nombre de departamento: DISE♦O

---dbConsultarEmpleadosMismoDepartamento(3)---

DNI: 123456789

Nombre: P♦rez

Sueldo: 36000.0

Número de departamento: 3

-----

DNI: 666884444

Nombre: Ojeda

Sueldo: 37000.0

Número de departamento: 3

-----

DNI: 666999333

Nombre: Ruiz

Sueldo: 25000.0

Número de departamento: 3

-----

DNI: 999999999

Nombre: Simón

Sueldo: 33000.0

Número de departamento: 3

-----

-----

Número de departamento: 4

Nombre de departamento: DESARROLLO

---dbConsultarEmpleadosMismoDepartamento(4)---

DNI: 333445555

Nombre: Campos

Sueldo: 50000.0

Número de departamento: 4

-----

DNI: 222447777

Nombre: Torres

Sueldo: 25000.0

Número de departamento: 4

-----

DNI: 987654321

Nombre: Jiménez

Sueldo: 40000.0

Número de departamento: 4

-----

DNI: 000000000

Nombre: Sevilla

Sueldo: 45000.0

Número de departamento: 4

-----

-----

Número de departamento: 9

CONEXIÓN REALIZADA

---dbMostrarEmpleadosDeCadaDepartamento---

Número de departamento: 4

Nombre de departamento: DESARROLLO

---dbConsultarEmpleadosMismoDepartamento(4)---

DNI: 333445555

Nombre: Campos

Sueldo: 50000.00

Número de departamento: 4

-----

DNI: 222447777

Nombre: Torres

Sueldo: 25000.00

Número de departamento: 4

-----

DNI: 987654321

Nombre: Jiménez

Sueldo: 40000.00

Número de departamento: 4

-----

DNI: 000000000

Nombre: Sevilla

Sueldo: 45000.00

Número de departamento: 4

-----

-----

Número de departamento: 1

Nombre de departamento: nuevo

---dbConsultarEmpleadosMismoDepartamento(1)---

-----

Número de departamento: 2

Nombre de departamento: nuevo hola

---dbConsultarEmpleadosMismoDepartamento(2)---

DNI: 555555555

Nombre: Martín

Sueldo: 29000.00

Número de departamento: 2

-----

DNI: 666666666

Nombre: Lagos

Sueldo: 27000.00

Número de departamento: 2

-----

DNI: 777777777

Nombre: Salazar

Sueldo: 32000.00

Número de departamento: 2

-----

DNI: 888888888

Nombre: López

Sueldo: 32000.00

Número de departamento: 2

-----

-----

Número de departamento: 3

Nombre de departamento: PYTHON

---dbConsultarEmpleadosMismoDepartamento(3)---

DNI: 123456789

Nombre: Pérez

Sueldo: 36000.00

Número de departamento: 3

-----

DNI: 666884444



Nombre: Ojeda

Sueldo: 37000.00

Número de departamento: 3

-----

DNI: 666999333

Nombre: Ruiz

Sueldo: 25000.00

Número de departamento: 3

-----

DNI: 999999999

Nombre: Simón

Sueldo: 33000.00

Número de departamento: 3

-----

-----

---dbDesconectar---

Desconexión realizada correctamente

---Fin de programa---

## 9. SALIDA COMPLETA POR CONSOLA DE ORACLE

---Programa principal---

---dbConectar---

---Conectando a Oracle---

Conexión realizada a la base de datos <oracledb.Connection to system@(DESCRIPTION=(RETRY\_DELAY=1)(ADDRESS=(PROTOCOL=tcp)(HOST=localhost)(PORT=1521))(CONNECT\_DATA=(SID=x)))>

CONEXIÓN REALIZADA

---dbMostrarEmpleados1---

('111111111', 'Simón', '15-11-1997', '10005', 'M', 35000.0, 1)

('222222222', 'Martín', '12-12-1991', '06800', 'M', 40000.0, 1)

('333333333', 'Alvarez', '21-08-1990', '10800', 'H', 30000.0, 1)  
('444444444', 'Gonzalez', '12-09-1994', '06002', 'H', 28000.0, 1)  
('555555555', 'Martín', '11-03-1989', '10005', 'M', 29000.0, 2)  
('666666666', 'Lagos', '07-07-1991', '06800', 'M', 27000.0, 2)  
('777777777', 'Salazar', '22-07-1993', '06300', 'H', 32000.0, 2)  
('888888888', 'López', '10-11-1994', '10300', 'H', 32000.0, 2)  
('123456789', 'Pérez', '15-11-1967', '06400', 'H', 36000.0, 3)  
('666884444', 'Ojeda', '12-12-1991', '06300', 'H', 37000.0, 3)  
('666999333', 'Ruiz', '01-02-1990', '10300', 'H', 25000.0, 3)  
('999999999', 'Simón', '31-08-1988', '10600', 'M', 33000.0, 3)  
('333445555', 'Campos', '12-04-1974', '06002', 'M', 50000.0, 4)  
('222447777', 'Torres', '30-05-1988', '10600', 'H', 25000.0, 4)  
('987654321', 'Jiménez', '10-04-1971', '06400', 'M', 40000.0, 4)  
('000000000', 'Sevilla', '17-04-1980', '10800', 'M', 45000.0, 4)

Número de registros recuperados: 16

-----

---dbMostrarEmpleados2---

('111111111', 'Sánchez', '15-11-1997', '10005', 'M', 35000.0, 1)  
('222222222', 'Martínez', '12-12-1991', '06800', 'M', 40000.0, 1)  
('333333333', 'Alvarez', '21-08-1990', '10800', 'H', 30000.0, 1)  
('444444444', 'Gonzalez', '12-09-1994', '06002', 'H', 28000.0, 1)  
('555555555', 'Martín', '11-03-1989', '10005', 'M', 29000.0, 2)  
('666666666', 'Lagos', '07-07-1991', '06800', 'M', 27000.0, 2)  
('777777777', 'Salazar', '22-07-1993', '06300', 'H', 32000.0, 2)  
('888888888', 'López', '10-11-1994', '10300', 'H', 32000.0, 2)  
('123456789', 'Pérez', '15-11-1967', '06400', 'H', 36000.0, 3)  
('666884444', 'Ojeda', '12-12-1991', '06300', 'H', 37000.0, 3)  
('666999333', 'Ruiz', '01-02-1990', '10300', 'H', 25000.0, 3)  
('999999999', 'Simón', '31-08-1988', '10600', 'M', 33000.0, 3)  
('333445555', 'Campos', '12-04-1974', '06002', 'M', 50000.0, 4)  
('222447777', 'Torres', '30-05-1988', '10600', 'H', 25000.0, 4)

('987654321', 'Jiménez', '10-04-1971', '06400', 'M', 40000.0, 4)

('000000000', 'Sevilla', '17-04-1980', '10800', 'M', 45000.0, 4)

Número de registros recuperados: 16

-----

---dbMostrarEmpleados3---

('111111111', 'Sánchez', '15-11-1997', '10005', 'M', 35000.0, 1)

('222222222', 'Martínez', '12-12-1991', '06800', 'M', 40000.0, 1)

('333333333', 'Álvarez', '21-08-1990', '10800', 'H', 30000.0, 1)

('444444444', 'González', '12-09-1994', '06002', 'H', 28000.0, 1)

('555555555', 'Martín', '11-03-1989', '10005', 'M', 29000.0, 2)

Número de registros seleccionados: 5

Número de registros recuperados: 5

-----

---dbMostrarEmpleados4---

('111111111', 'Sánchez', '15-11-1997', '10005', 'M', 35000.0, 1)

('222222222', 'Martínez', '12-12-1991', '06800', 'M', 40000.0, 1)

('333333333', 'Álvarez', '21-08-1990', '10800', 'H', 30000.0, 1)

('444444444', 'González', '12-09-1994', '06002', 'H', 28000.0, 1)

('555555555', 'Martín', '11-03-1989', '10005', 'M', 29000.0, 2)

('666666666', 'Lagos', '07-07-1991', '06800', 'M', 27000.0, 2)

('777777777', 'Salazar', '22-07-1993', '06300', 'H', 32000.0, 2)

('888888888', 'López', '10-11-1994', '10300', 'H', 32000.0, 2)

('123456789', 'Pérez', '15-11-1967', '06400', 'H', 36000.0, 3)

('666884444', 'Ojeda', '12-12-1991', '06300', 'H', 37000.0, 3)

('666999333', 'Ruiz', '01-02-1990', '10300', 'H', 25000.0, 3)

('999999999', 'Simón', '31-08-1988', '10600', 'M', 33000.0, 3)

('333445555', 'Campos', '12-04-1974', '06002', 'M', 50000.0, 4)

('222447777', 'Torres', '30-05-1988', '10600', 'H', 25000.0, 4)

('987654321', 'Jiménez', '10-04-1971', '06400', 'M', 40000.0, 4)

('000000000', 'Sevilla', '17-04-1980', '10800', 'M', 45000.0, 4)

Número de registros recuperados: 16

Número de registros recuperados: 16

-----

---dbObtenerEmpleados---

Introduce dni de Empleados: 111111111

DNI: 111111111

Nombre: S◆nchez

Fecha Nacimiento: 15-11-1997

CP: 10005

Sexo: M

Sueldo: 35000.0

Número de departamento: 1

-----

---dbConsultarEmpleados---

DNI: 111111111

Nombre: S◆nchez

Fecha Nacimiento: 15-11-1997

CP: 10005

Sexo: M

Sueldo: 35000.0

Número de departamento: 1

-----

DNI: 222222222

Nombre: Mart◆nez

Fecha Nacimiento: 12-12-1991

CP: 06800

Sexo: M

Sueldo: 40000.0

Número de departamento: 1

-----

DNI: 333333333

Nombre: ◆lvarez

Fecha Nacimiento: 21-08-1990

CP: 10800

Sexo: H

Sueldo: 30000.0

Número de departamento: 1

-----

DNI: 444444444

Nombre: Gonz lez

Fecha Nacimiento: 12-09-1994

CP: 06002

Sexo: H

Sueldo: 28000.0

Número de departamento: 1

-----

DNI: 555555555

Nombre: Mart n

Fecha Nacimiento: 11-03-1989

CP: 10005

Sexo: M

Sueldo: 29000.0

Número de departamento: 2

-----

DNI: 666666666

Nombre: Lagos

Fecha Nacimiento: 07-07-1991

CP: 06800

Sexo: M

Sueldo: 27000.0

Número de departamento: 2

-----

DNI: 777777777

Nombre: Salazar

Fecha Nacimiento: 22-07-1993

CP: 06300

Sexo: H

Sueldo: 32000.0

Número de departamento: 2

-----

DNI: 888888888

Nombre: López

Fecha Nacimiento: 10-11-1994

CP: 10300

Sexo: H

Sueldo: 32000.0

Número de departamento: 2

-----

DNI: 123456789

Nombre: Pérez

Fecha Nacimiento: 15-11-1967

CP: 06400

Sexo: H

Sueldo: 36000.0

Número de departamento: 3

-----

DNI: 666884444

Nombre: Ojeda

Fecha Nacimiento: 12-12-1991

CP: 06300

Sexo: H

Sueldo: 37000.0

Número de departamento: 3

-----

DNI: 666999333

Nombre: Ruiz

Fecha Nacimiento: 01-02-1990

CP: 10300

Sexo: H

Sueldo: 25000.0

Número de departamento: 3

-----

DNI: 999999999

Nombre: Simón

Fecha Nacimiento: 31-08-1988

CP: 10600

Sexo: M

Sueldo: 33000.0

Número de departamento: 3

-----

DNI: 333445555

Nombre: Campos

Fecha Nacimiento: 12-04-1974

CP: 06002

Sexo: M

Sueldo: 50000.0

Número de departamento: 4

-----

DNI: 222447777

Nombre: Torres

Fecha Nacimiento: 30-05-1988

CP: 10600

Sexo: H

Sueldo: 25000.0

Número de departamento: 4

-----  
DNI: 987654321

Nombre: Jimenez

Fecha Nacimiento: 10-04-1971

CP: 06400

Sexo: M

Sueldo: 40000.0

Número de departamento: 4  
-----

DNI: 000000000

Nombre: Sevilla

Fecha Nacimiento: 17-04-1980

CP: 10800

Sexo: M

Sueldo: 45000.0

Número de departamento: 4  
-----

Número de registros recuperados: 16

Número de registros recuperados: 16  
-----

---dbConsultarDepartamentos---

Número de departamento: 2

Nombre de departamento: PRODUCCION

Coste: 0.0

?: 0.0  
-----

Número de departamento: 1

Nombre de departamento: ARQUITECTOS

Coste: 0.0

?: 0.0  
-----



Número de departamento: 3

Nombre de departamento: DISEÑO

Coste: 0.0

?: 0.0

-----

Número de departamento: 4

Nombre de departamento: DESARROLLO

Coste: 0.0

?: 0.0

-----

Número de departamento: 8

Nombre de departamento: PROGRAMACIÓN

Coste: 9.0

?: 3.0

-----

Número de departamento: 9

Nombre de departamento: FINANZAS

Coste: 0.0

?: 0.0

-----

Número de registros recuperados: 6

Número de registros recuperados: 6

-----

---dbInsertarDepartamentos---

Introduce el número del departamento: 8

Introduce el nombre del departamento: EMPRENDIMIENTO

Introduce el coste del departamento: 8

Introduce el porcentaje del departamento: 8

Error. No se ha podido insertar el Departamento

ORA-00001: restricción única (SYSTEM.SYS\_C008317) violada

Help: <https://docs.oracle.com/error-help/db/ora-00001/>

---dbConsultarDepartamentos---

Número de departamento: 2

Nombre de departamento: PRODUCCIÓN

Coste: 0.0

?: 0.0

-----

Número de departamento: 1

Nombre de departamento: ARQUITECTOS

Coste: 0.0

?: 0.0

-----

Número de departamento: 3

Nombre de departamento: DISEÑO

Coste: 0.0

?: 0.0

-----

Número de departamento: 4

Nombre de departamento: DESARROLLO

Coste: 0.0

?: 0.0

-----

Número de departamento: 8

Nombre de departamento: PROGRAMACIÓN

Coste: 9.0

?: 3.0

-----

Número de departamento: 9

Nombre de departamento: FINANZAS

Coste: 0.0

?: 0.0

-----

Número de registros recuperados: 6

Número de registros recuperados: 6

-----

---dbModificarDepartamentos---

Introduce el número del departamento a actualizar: 8

1- Nombre del departamento

2- Coste del departamento

3- Porcentaje del departamento

Elige qué propiedad quieres actualizar de este departamento. (1-3)1

1

Introduce el nuevo nombre del departamento que quieres poner: modificado

Tupla actualizada correctamente

----- tupla actualizada -----

Número de departamento: 8

Nombre de departamento: modificado

Coste: 9.0

?: 3.0

-----

-----

¿Desea realizar otra operación? (S/N): s

Introduce el número del departamento a actualizar: 8

1- Nombre del departamento

2- Coste del departamento

3- Porcentaje del departamento

Elige qué propiedad quieres actualizar de este departamento. (1-3)2

2

Introduce el nuevo coste del departamento que quieres poner: 10

Tupla actualizada correctamente

----- tupla actualizada -----

Número de departamento: 8

Nombre de departamento: modificado

Coste: 10.0

?: 3.0

-----

-----

¿Desea realizar otra operación? (S/N): n

---dbConsultarDepartamentos---

Número de departamento: 2

Nombre de departamento: PRODUCCION

Coste: 0.0

?: 0.0

-----

Número de departamento: 1

Nombre de departamento: ARQUITECTOS

Coste: 0.0

?: 0.0

-----

Número de departamento: 3

Nombre de departamento: DISEÑO

Coste: 0.0

?: 0.0

-----

Número de departamento: 4

Nombre de departamento: DESARROLLO

Coste: 0.0

?: 0.0

-----

Número de departamento: 8

Nombre de departamento: modificado

Coste: 10.0

?: 3.0

-----

Número de departamento: 9

Nombre de departamento: FINANZAS

Coste: 0.0

?: 0.0

-----

Número de registros recuperados: 6

Número de registros recuperados: 6

-----

---dbBorrarDepartamentos---

Introduce el número del departamento a actualizar: 8

Tupla borrada correctamente

---dbConsultarDepartamentos---

Número de departamento: 2

Nombre de departamento: PRODUCCI?N

Coste: 0.0

?: 0.0

-----

Número de departamento: 1

Nombre de departamento: ARQUITECTOS

Coste: 0.0

?: 0.0

-----

Número de departamento: 3

Nombre de departamento: DISE?O

Coste: 0.0

?: 0.0

-----

Número de departamento: 4

Nombre de departamento: DESARROLLO

Coste: 0.0

?: 0.0

-----  
Número de departamento: 9

Nombre de departamento: FINANZAS

Coste: 0.0

?: 0.0  
-----

Número de registros recuperados: 5

Número de registros recuperados: 5  
-----

---dbInsertarMultiplesDepartamentos---

Tupla insertada correctamente

Tupla insertada correctamente

Tupla insertada correctamente

---dbConsultarDepartamentos---

Número de departamento: 2

Nombre de departamento: PRODUCCION

Coste: 0.0

?: 0.0  
-----

Número de departamento: 1

Nombre de departamento: ARQUITECTOS

Coste: 0.0

?: 0.0  
-----

Número de departamento: 3

Nombre de departamento: DISEÑO

Coste: 0.0

?: 0.0  
-----

Número de departamento: 4

Nombre de departamento: DESARROLLO

Coste: 0.0

?: 0.0

-----

Número de departamento: 5

Nombre de departamento: INVESTIGACIÓN

Coste: 0.0

?: 0.0

-----

Número de departamento: 9

Nombre de departamento: FINANZAS

Coste: 0.0

?: 0.0

-----

Número de departamento: 6

Nombre de departamento: MARKETING

Coste: 0.0

?: 0.0

-----

Número de departamento: 7

Nombre de departamento: VENTAS

Coste: 0.0

?: 0.0

-----

Número de registros recuperados: 8

Número de registros recuperados: 8

-----

---dbBorrarMultiplesDepartamentos---

Tupla borrada correctamente

Tupla borrada correctamente

Tupla borrada correctamente

---dbConsultarDepartamentos---

Número de departamento: 2

Nombre de departamento: PRODUCCIÓN

Coste: 0.0

?: 0.0

-----

Número de departamento: 1

Nombre de departamento: ARQUITECTOS

Coste: 0.0

?: 0.0

-----

Número de departamento: 3

Nombre de departamento: DISEÑO

Coste: 0.0

?: 0.0

-----

Número de departamento: 4

Nombre de departamento: DESARROLLO

Coste: 0.0

?: 0.0

-----

Número de departamento: 9

Nombre de departamento: FINANZAS

Coste: 0.0

?: 0.0

-----

Número de registros recuperados: 5

Número de registros recuperados: 5

---

---dbMostrarEmpleadosDeCadaDepartamento---

Número de departamento: 2

Nombre de departamento: PRODUCCIÓN

---dbConsultarEmpleadosMismoDepartamento(2)---



DNI: 555555555

Nombre: Martín

Sueldo: 29000.0

Número de departamento: 2

-----

DNI: 666666666

Nombre: Lagos

Sueldo: 27000.0

Número de departamento: 2

-----

DNI: 777777777

Nombre: Salazar

Sueldo: 32000.0

Número de departamento: 2

-----

DNI: 888888888

Nombre: López

Sueldo: 32000.0

Número de departamento: 2

-----

-----

Número de departamento: 1

Nombre de departamento: ARQUITECTOS

---dbConsultarEmpleadosMismoDepartamento(1)---

DNI: 111111111

Nombre: Sánchez

Sueldo: 35000.0

Número de departamento: 1

-----

DNI: 222222222


Nombre: Martínez

Sueldo: 40000.0

Número de departamento: 1

-----

DNI: 333333333

Nombre: lvarez

Sueldo: 30000.0

Número de departamento: 1

-----

DNI: 444444444

Nombre: Gonzlez

Sueldo: 28000.0

Número de departamento: 1

-----


-----

Número de departamento: 3

Nombre de departamento: DISEO

---dbConsultarEmpleadosMismoDepartamento(3)---

DNI: 123456789

Nombre: Prez

Sueldo: 36000.0

Número de departamento: 3

-----

DNI: 666884444

Nombre: Ojeda

Sueldo: 37000.0

Número de departamento: 3

-----

DNI: 666999333

Nombre: Ruiz

Sueldo: 25000.0

Número de departamento: 3

-----  
DNI: 999999999

Nombre: Simón

Sueldo: 33000.0

Número de departamento: 3  
-----  
-----

Número de departamento: 4

Nombre de departamento: DESARROLLO

---dbConsultarEmpleadosMismoDepartamento(4)---

DNI: 333445555

Nombre: Campos

Sueldo: 50000.0

Número de departamento: 4  
-----

DNI: 222447777

Nombre: Torres

Sueldo: 25000.0

Número de departamento: 4  
-----

DNI: 987654321

Nombre: Jiménez

Sueldo: 40000.0

Número de departamento: 4  
-----

DNI: 000000000

Nombre: Sevilla

Sueldo: 45000.0

Número de departamento: 4  
-----  
-----

Número de departamento: 9

Nombre de departamento: FINANZAS

---dbConsultarEmpleadosMismoDepartamento(9)---

-----

-----

---dbDesconectar---

Desconexión realizada correctamente

---Fin de programa---