

**PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ  
FACULTAD DE CIENCIAS E INGENIERÍA**

**Bases de Datos  
4ta. Práctica (tipo B)  
(Semestre 2021-2)**

**Indicaciones generales:**

- Duración: **110 minutos** (parte calificada).
- Pueden usar libros y apuntes de clase, pero no compartirlos.
- El archivo que contenga sus scripts o modelo no deberá ser comprimido.
- Los archivos indicados se subirán a **PAIDEIA**, en el espacio indicado por los jefes de Laboratorio. Se destinarán los últimos 10 minutos exclusivamente para subir los trabajos a PAIDEIA.
- Guarde cada uno de sus archivos con el nombre que se le indica. Es importante seguir el estándar del nombre de archivo indicado.
- La presentación del trabajo influye en su calificación.

**Puntaje: 20 puntos**

---

Se desea implementar un sistema de información (ver Figura 1) que administre una entidad bancaria; llamada *PerúPago*, recientemente fundada respondiendo a la necesidad de medios virtuales de pago dentro de todo el territorio peruano, generado por el contexto de emergencia sanitaria que ahonda en nuestro país desde marzo de 2020.

Recuerde que:

El eje principal de este proceso es la cuenta bancaria que pertenece exclusivamente a un cliente. Cada cuenta bancaria está asociada a un tipo de cuenta; cada tipo cuenta con ciertas características, tales como la tasa de interés que gana la cuenta, el costo de la comisión por transacción realizada, el ingreso a un “club” exclusivo con beneficios y descuentos en locales asociados del Banco, entre otros.

Antes de comenzar el laboratorio, ejecute Oracle SQL Developer, cree una nueva conexión llamada **LAB4\_CALIFICADA**, y ejecute, respetando ese orden, el script contenido en los archivos:

**INF246\_CALIFICADA\_LAB4\_20212\_DDL.sql**

**INF246\_CALIFICADA\_LAB4\_20212\_DML.sql.**



### PREGUNTA 1: (3 puntos)

Elaborar una función que reciba como parámetro el nombre de una región y que retorne la cantidad de clientes de dicha región. Si la región no existe, la función deberá retornar -1.

The screenshot shows two SQL queries and their results in a database interface. The first query is:

```
select f_contar_clientes_por_region( 'Lima' ) from dual;
```

The result is a single row with the value 5.

The second query is:

```
select f_contar_clientes_por_region( 'Tumbes' ) from dual;
```

The result is a single row with the value -1.

### PREGUNTA 2: (3 puntos)

Elaborar una función que tenga tres parámetros: nombres, apellido paterno y apellido materno de una persona. Si los datos corresponden a un cliente, la función devuelve el total de dinero transferido por dicho cliente. En caso contrario, la función devuelve -1.

The screenshot shows two SQL queries and their results in a database interface. The first query is:

```
select f_total_transferencias_cliente( 'Sergio Miguel', 'Zuñiga', 'Cuya' ) from dual;
```

The result is a single row with the value 4237,75.

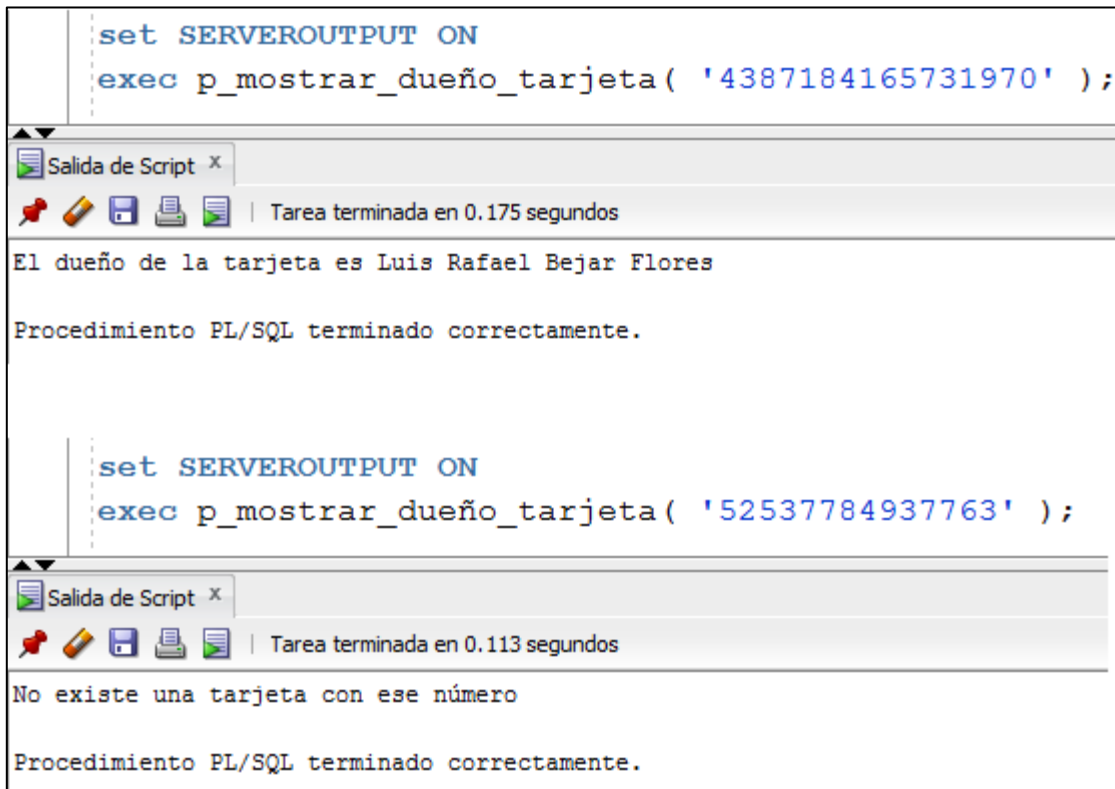
The second query is:

```
select f_total_transferencias_cliente( 'Luis Miguel', 'Torres', 'Castro' ) from dual;
```

The result is a single row with the value -1.

### PREGUNTA 3: (3 puntos)

Elaborar un procedimiento que reciba como parámetro un número de tarjeta y que imprima el nombre, apellido paterno y apellido materno (todo junto) del cliente que es dueño de dicha tarjeta. Si el número de tarjeta no existe se debe imprimir el mensaje correspondiente.



```
set SERVEROUTPUT ON
exec p_mostrar_dueño_tarjeta( '4387184165731970' );
```

Salida de Script x | Tarea terminada en 0.175 segundos

El dueño de la tarjeta es Luis Rafael Bejar Flores

Procedimiento PL/SQL terminado correctamente.

```
set SERVEROUTPUT ON
exec p_mostrar_dueño_tarjeta( '52537784937763' );
```

Salida de Script x | Tarea terminada en 0.113 segundos

No existe una tarjeta con ese número

Procedimiento PL/SQL terminado correctamente.

#### PREGUNTA 4: (4 puntos)

Elaborar un procedimiento que tenga cinco parámetros. En el primero se envía el número de una transferencia y en los otros cuatro se debe obtener: el nombre completo del cliente emisor (el que hace la transferencia), el nombre completo del cliente receptor (el que recibe la transferencia), la fecha de la transferencia y el monto de la transferencia. Si el número de la transferencia no existe, se devuelve null en esos cuatro parámetros.

```
set SERVEROUTPUT ON
declare
    v_id_transferencia number;
    v_cliente_emisor varchar2(100);
    v_cliente_receptor varchar2(100);
    v_fecha date;
    v_monto number;
begin
    v_id_transferencia := 6;
    p_obtener_datos_transferencia( v_id_transferencia, v_cliente_emisor,
                                   v_cliente_receptor, v_fecha, v_monto );
    dbms_output.put_line( 'Cliente emisor: ' || v_cliente_emisor );
    dbms_output.put_line( 'Cliente receptor: ' || v_cliente_receptor );
    dbms_output.put_line( 'Fecha de la transferencia: ' || v_fecha );
    dbms_output.put_line( 'Monto de la transferencia: ' || v_monto );
end;
```

Salida de Script x

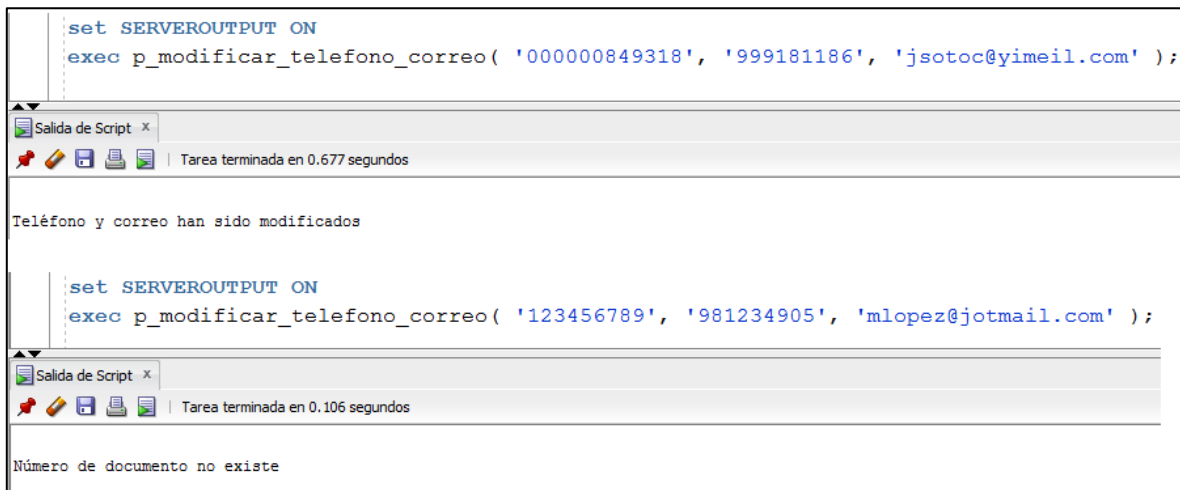
Tarea terminada en 0.161 segundos

Cliente emisor: Zarela Rosa Martinez Fiestas  
Cliente receptor: Catalina Isabel Perez Olivares  
Fecha de la transferencia: 15/04/21  
Monto de la transferencia: 321,99

Procedimiento PL/SQL terminado correctamente.

### PREGUNTA 5: (3 puntos)

Elaborar un procedimiento que tenga tres parámetros: el número de documento de identidad de un cliente, un nuevo número de teléfono y un nuevo correo electrónico. Se debe modificar con los nuevos valores, el teléfono y correo electrónico del cliente al que corresponde el documento de identidad. Si este documento no existe, se debe mostrar un mensaje apropiado.



```
set SERVEROUTPUT ON
exec p_modificar_telefono_correo( '000000849318', '999181186', 'jsotoc@yimeil.com' );
```

Salida de Script x | Tarea terminada en 0.677 segundos

Teléfono y correo han sido modificados

```
set SERVEROUTPUT ON
exec p_modificar_telefono_correo( '123456789', '981234905', 'mlopez@jotmail.com' );
```

Salida de Script x | Tarea terminada en 0.106 segundos


Número de documento no existe

### PREGUNTA 6: (4 puntos)

Elaborar una función que reciba como parámetro una fecha y que devuelva el signo zodiacal que corresponde a dicha fecha. Tenga en cuenta lo siguiente:

Aries (21 de marzo — 19 de abril)  
Tauro (20 de abril — 21 de mayo)  
Géminis (21 de mayo — 20 de junio)  
Cáncer (21 de junio — 22 de julio)  
Leo (23 de julio — 22 de agosto)  
Virgo (23 de agosto — 22 de septiembre)  
Libra (23 de septiembre — 22 de octubre)  
Escorpio (23 de octubre — 21 de noviembre)  
Sagitario (22 de noviembre — 21 de diciembre)  
Capricornio (22 de diciembre — 19 de enero)  
Acuario (20 de enero — 18 de febrero)  
Piscis (19 de febrero — 20 de marzo)

Para probar la función puede mostrar los signos zodiacales de todos los clientes, como se muestra en la siguiente imagen:



The screenshot shows a SQL query window with the following query:

```
select nombres, apellido_paterno, apellido_materno, fecha_nacimiento,
       f_obtener_signo(fecha_nacimiento)
from pp_cliente;
```

Below the query, the results are displayed in a table with 5 columns: NOMBRES, APELLIDO\_PATERNO, APELLIDO\_MATERNO, FECHA\_NACIMIENTO, and F\_OBTENER\_SIGNO(FECHA\_NACIMIENTO). The table contains 10 rows of data.

	NOMBRES	APELLIDO_PATERNO	APELLIDO_MATERNO	FECHA_NACIMIENTO	F_OBTENER_SIGNO(FECHA_NACIMIENTO)
1	Laura Sofia	Prado	Nunez	23/10/79	Escorpio
2	Leonardo Manuel	Olarte	Timo	05/03/76	Piscis
3	Sergio Miguel	Zuñiga	Cuya	12/09/82	Virgo
4	Lourdes	Hidalgo	Garcia	05/10/89	Libra
5	Zarela Rosa	Martinez	Fiestas	03/03/94	Piscis
6	Luis Rafael	Bejar	Flores	18/06/72	Géminis
7	Joaquin	Soto	Cerna	29/11/98	Sagitario
8	Catalina Isabel	Perez	Olivares	19/04/99	Aries
9	José Rafael	Pastor	Uceda	09/07/92	Cáncer
10	Clark	Izquierdo	Galloso	14/08/83	Leo

Guarde el script con el formato de nombre de archivo **LAB4\_<su código de alumno>.sql** o **LAB4\_<su código de alumno>.txt**

**Ejemplo:** LAB4\_20143258.sql o LAB4\_20143258.txt

Lunes, 15 de noviembre de 2021