UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS
FACULTAD DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

“LABORATORIO 03”

ASIGNATURA

Administración de Base de Datos

DOCENTE

Chávez Soto, Jorge Luis

ESTUDIANTE

Carhuaricra Anco, Heidy Nicole - 23200150

Lima, Perú

2025

ÍNDICE

LABORATORIO 03

I. OBJETIVOS

El presente laboratorio tiene por objetivos:

- Crear un bloque PL/SQL para seleccionar datos de una tabla.
- Crear un bloque PL/SQL para insertar datos en una tabla.
- Crear un bloque PL/SQL para actualizar datos en una tabla.
- Crear un bloque PL/SQL para eliminar datos de una tabla.

II. DESCRIPCIÓN DE LAS TABLAS

- ***Countries***

Name	Null?	Type
COUNTRY_ID	NOT NULL	CHAR (2)
COUNTRY_NAME		VARCHAR2 (40)
REGION_ID		NUMBER

- ***Departments***

Name	Null?	Type
DEPARTMENT_ID	NOT NULL	NUMBER (4)
DEPARTMENT_NAME	NOT NULL	VARCHAR2 (30)
MANAGER_ID		NUMBER (5)
LOCATION_ID		NUMBER (4)

- ***Employees***

Name	Null?	Type
EMPLOYEE_ID	NOT NULL	NUMBER (6)
FIRST_NAME		VARCHAR2 (20)
LAST_NAME	NOT NULL	VARCHAR2 (25)
EMAIL	NOT NULL	VARCHAR2 (25)
PHONE_NUMBER		VARCHAR2 (20)
HIRE_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2 (10)
SALARY		NUMBER (8, 2)
COMMISSION_PCT		NUMBER (2, 2)

MANAGER_ID		NUMBER (6)
DEPARTMENT_ID		NUMBER (4)

- ***Jobs***

Name	Null?	Type
JOB_ID	NOT NULL	VARCHAR2 (10)
JOB_TITLE	NOT NULL	VARCHAR2 (35)
MIN_SALARY		NUMBER (6)
MAX_SALARY		NUMBER (6)

- ***Job_History***

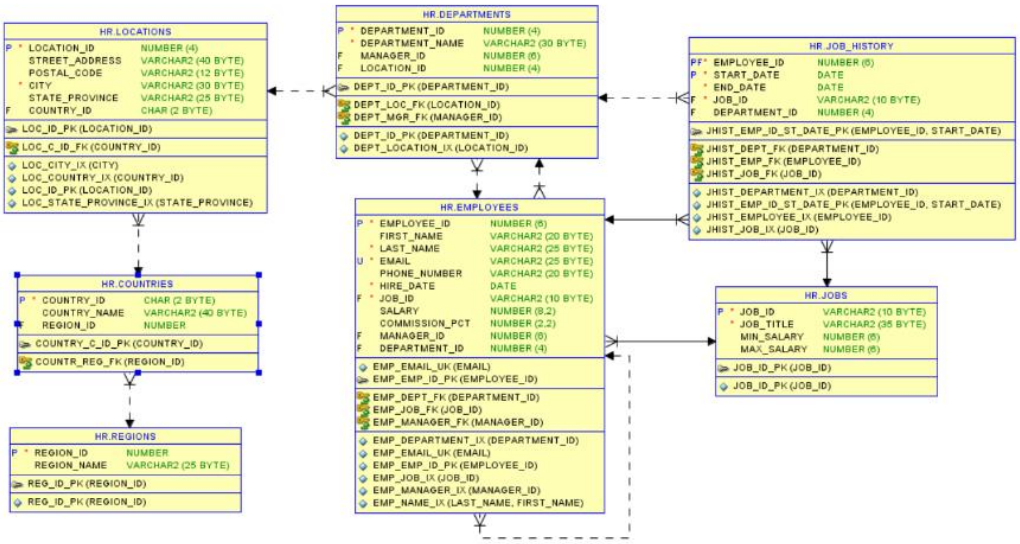
Name	Null?	Type
EMPLOYEE_ID	NOT NULL	NUMBER (6)
START_DATE	NOT NULL	DATE
END_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2 (10)
DEPARTMENT_ID		NUMBER (4)

- ***Locations***

Name	Null?	Type
LOCATION_ID	NOT NULL	NUMBER (4)
STREET_ADDRESS		VARCHAR2 (40)
POSTAL_CODE		VARCHAR2 (12)
CITY	NOT NULL	VARCHAR2 (30)
STATE PROVINCE		VARCHAR2 (25)
COUNTRY_ID		CHAR (2)

- ***Regions***

Name	Null?	Type
RGION_ID	NOT NULL	NUMBER ()
REGION_NAME		VARCHAR2 (25)



III.EJERCICIOS PLANTEADOS

1. Obtener y presentar código de Empleado, nombres y apellidos, código de Puesto actual y nombre de Puesto actual, de los N empleados que han rotado más de puesto desde que ingresaron a la empresa, para cada uno de ellos presente como columna adicional el número de veces que han cambiado de puesto.

```
SET serveroutput ON

DECLARE
    CURSOR employees_cursor IS
    SELECT
        e.employee_id AS idEmpleado,
        e.first_name AS nombres,
        e.last_name AS apellidos,
        e.job_id AS idPuestoActual,
        j.job_title AS puestoActual,
        COUNT(jh.job_id) AS numCambios
    FROM employees e
    JOIN jobs j ON e.job_id = j.job_id
    LEFT JOIN job_history jh ON e.employee_id = jh.employee_id
    GROUP BY e.employee_id, e.first_name, e.last_name,
e.job_id, j.job_title
    HAVING COUNT(jh.job_id) > 0
    ORDER BY COUNT(jh.job_id) DESC;
    v_numero NUMERIC(2) := 0;
BEGIN
    FOR registro_cursor IN employees_cursor
    LOOP
        v_numero := v_numero + 1;
        dbms_output.put_line(v_numero || '. Empleado: ' ||
registro_cursor.idEmpleado ||
            '| Nombres y Apellidos: ' ||
registro_cursor.nombres || ' ' || registro_cursor.apellidos ||
```

```
            '| Puesto Actual: ' ||
registro_cursor.idPuestoActual || ' - ' ||
registro_cursor.puestoActual ||
            '| N° Cambios: ' || registro_cursor.numCambios);
    END LOOP;
END;
/
```

```
PROBLEMS 3  OUTPUT  DEBUG CONSOLE  TERMINAL  QUERY RESULT  SCRIPT OUTPUT  SQL HISTORY  TASK MONITOR  PORTS

1. Empleado: 101| Nombres y Apellidos: Neena Yang| Puesto Actual: AD_VP - Administration Vice President| N° Cambios: 2
2. Empleado: 176| Nombres y Apellidos: Jonathon Taylor| Puesto Actual: SA_REP - Sales Representative| N° Cambios: 2
3. Empleado: 200| Nombres y Apellidos: Jennifer Whalen| Puesto Actual: AD_ASST - Administration Assistant| N° Cambios: 2
4. Empleado: 201| Nombres y Apellidos: Michael Martinez| Puesto Actual: MK_MAN - Marketing Manager| N° Cambios: 1
5. Empleado: 114| Nombres y Apellidos: Den Li| Puesto Actual: PU_MAN - Purchasing Manager| N° Cambios: 1
6. Empleado: 122| Nombres y Apellidos: Payam Kaufling| Puesto Actual: ST_MAN - Stock Manager| N° Cambios: 1
7. Empleado: 102| Nombres y Apellidos: Lex Garcia| Puesto Actual: AD_VP - Administration Vice President| N° Cambios: 1

Procedimiento PL/SQL terminado correctamente.
```

2. Elabore un resumen estadístico del número promedio de contrataciones por cada mes con respecto a todos los años que hay información en la base de datos. Debe presentar sólo dos columnas: Nombre del Mes y Número Promedio de Contrataciones en ese mes.

```
SET serveroutput ON

DECLARE
    CURSOR contratos_cursor IS
    SELECT
        TO_CHAR(TO_DATE(mes, 'MM'), 'MONTH') AS nombreMes,
        ROUND(AVG(contador)) AS promContratoMes
    FROM (
        SELECT
            TO_CHAR(e.hire_date, 'MM') AS mes,
            TO_CHAR(e.hire_date, 'YYYY') AS anio,
            COUNT(*) AS contador
        FROM employees e
        GROUP BY TO_CHAR(e.hire_date, 'MM'),
TO_CHAR(e.hire_date, 'YYYY')
    )
    GROUP BY mes
    ORDER BY TO_DATE(mes, 'MM');
    v_numero NUMBER := 0;
BEGIN
    FOR registro_cursor IN contratos_cursor
    LOOP
        v_numero := v_numero + 1;
        dbms_output.put_line(v_numero || '. Mes: ' ||
registro_cursor.nombreMes ||
            '| Promedio de Contrataciones: ' ||
registro_cursor.promContratoMes);
    END LOOP;
END;
```

```
/
```

```
PROBLEMS 3   OUTPUT   DEBUG CONSOLE   TERMINAL   QUERY RESULT   SCRIPT OUTPUT

1. Mes: JANUARY   | Promedio de Contrataciones: 2
2. Mes: FEBRUARY  | Promedio de Contrataciones: 3
3. Mes: MARCH     | Promedio de Contrataciones: 3
4. Mes: APRIL     | Promedio de Contrataciones: 2
5. Mes: MAY       | Promedio de Contrataciones: 2
6. Mes: JUNE      | Promedio de Contrataciones: 2
7. Mes: JULY      | Promedio de Contrataciones: 2
8. Mes: AUGUST    | Promedio de Contrataciones: 2
9. Mes: SEPTEMBER | Promedio de Contrataciones: 2
10. Mes: OCTOBER  | Promedio de Contrataciones: 2
11. Mes: NOVEMBER | Promedio de Contrataciones: 2
12. Mes: DECEMBER | Promedio de Contrataciones: 2

Procedimiento PL/SQL terminado correctamente.
```

3. Obtener y presentar la información de gastos en salario y estadística de empleados a nivel regional. Como encabezado de cada región presente el nombre de la región y en la siguiente línea indique como detalle la suma de salarios, cantidad de empleados, y fecha de ingreso del empleado más antiguo.

```
---Ejercicio 03
SET serveroutput ON

DECLARE
    CURSOR gastos_employees_cursor IS
    SELECT
        r.region_name AS nombreRegion,
        NVL(SUM(e.salary), 0) AS sumaSalarios,
        COUNT(e.employee_id) AS cantEmpleados,
        MIN(e.hire_date) AS ingresoEmpleadoAntiguo
    FROM regions r
    LEFT JOIN countries c ON c.region_id = r.region_id
    LEFT JOIN locations l ON l.country_id = c.country_id
    LEFT JOIN departments d ON d.location_id = l.location_id
    LEFT JOIN employees e ON e.department_id = d.department_id
    GROUP BY r.region_name
    ORDER BY r.region_name;
    v_numero NUMBER := 0;
BEGIN
    FOR registro_cursor IN gastos_employees_cursor
    LOOP
        v_numero := v_numero + 1;
        dbms_output.put_line(v_numero || '. Región: ' ||
registro_cursor.nombreRegion ||
            ' | Salario Total: ' ||
registro_cursor.sumaSalarios ||
```

```
            ' | N° Empleados: ' ||
registro_cursor.cantEmpleados ||
            ' | Ingreso de Emp. Antiguo: ' ||
registro_cursor.ingresoEmpleadoAntiguo);
    END LOOP;
END;
/
```

```
PROBLEMS 3  OUTPUT  DEBUG CONSOLE  TERMINAL  QUERY RESULT  SCRIPT OUTPUT  SQL HISTORY  TASK MONITOR  PORTS

1. Región: Africa | Salario Total: 0 | N° Empleados: 0 | Ingreso de Emp. Antiguo:
2. Región: Americas | Salario Total: 363416 | N° Empleados: 70 | Ingreso de Emp. Antiguo: 13/01/11
3. Región: Asia | Salario Total: 0 | N° Empleados: 0 | Ingreso de Emp. Antiguo:
4. Región: Europe | Salario Total: 321000 | N° Empleados: 36 | Ingreso de Emp. Antiguo: 07/06/12
5. Región: Oceania | Salario Total: 0 | N° Empleados: 0 | Ingreso de Emp. Antiguo:

Procedimiento PL/SQL terminado correctamente.
```

4. Obtenga todos los nombres de los empleados que estén registrado en el historial de sueldo y que no desempeñen el mismo que el puesto actual.

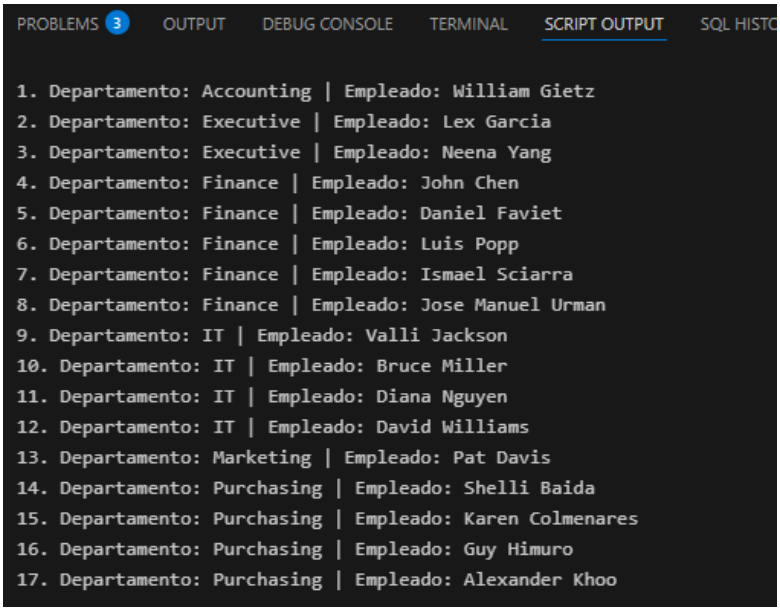
```
---Ejercicio 04
SET serveroutput ON

DECLARE
    CURSOR historial_salario_employees IS
    SELECT
        e.first_name || ' ' || e.last_name AS nombre_empleado,
        e.job_id AS id_puesto_actual,
        j1.job_title AS nombre_puesto_actual,
        jh.job_id AS id_puesto_anterior,
        j2.job_title AS nombre_puesto_anterior
    FROM employees e
    JOIN job_history jh
        ON e.employee_id = jh.employee_id
    JOIN jobs j1
        ON e.job_id = j1.job_id
    JOIN jobs j2
        ON jh.job_id = j2.job_id
    WHERE e.job_id <> jh.job_id
    ORDER BY nombre_empleado;
    v_numero NUMBER := 0;
BEGIN
    FOR registro_cursor IN historial_salario_employees
    LOOP
        v_numero := v_numero + 1;
        dbms_output.put_line(v_numero || '. Empleado: ' ||
        registro_cursor.nombre_empleado || ' | Puesto Actual:
(' ||
        registro_cursor.id_puesto_actual || ') ' ||
        registro_cursor.nombre_puesto_actual || ' | Puesto
Anterior: (' ||
        registro_cursor.id_puesto_anterior || ') ' ||
        registro_cursor.nombre_puesto_anterior);
```

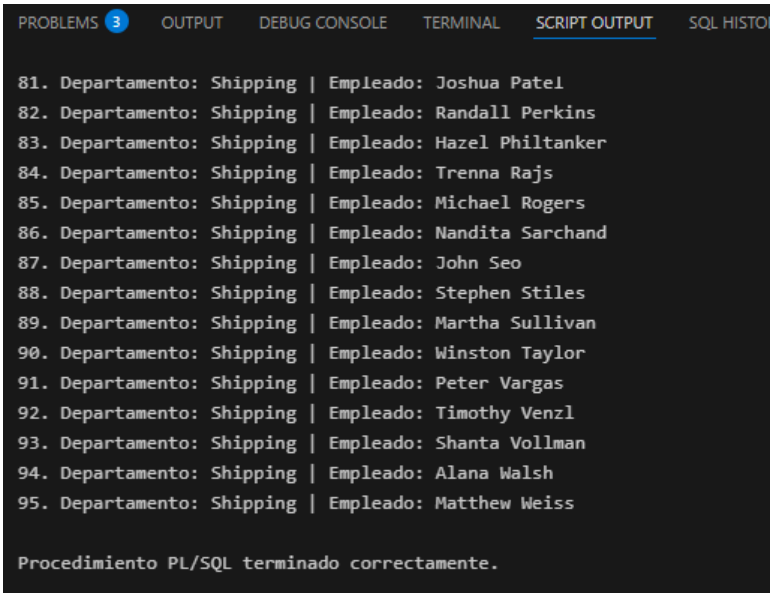
```
    END LOOP;
END;
/
```

```
PROBLEMS 3  OUTPUT  DEBUG CONSOLE  TERMINAL  SCRIPT OUTPUT  SQL HISTORY  TASK MONITOR  PORTS

1. Empleado: Den Li | Puesto Actual: (PU_MAN) Purchasing Manager | Puesto Anterior: (ST_CLERK) Stock Clerk
2. Empleado: Jennifer Whalen | Puesto Actual: (AD_ASST) Administration Assistant | Puesto Anterior: (AC_ACCOUNT) Public Accountant
3. Empleado: Jonathon Taylor | Puesto Actual: (SA_REP) Sales Representative | Puesto Anterior: (SA_MAN) Sales Manager
4. Empleado: Lex Garcia | Puesto Actual: (AD_VP) Administration Vice President | Puesto Anterior: (IT_PROG) Programmer
5. Empleado: Michael Martinez | Puesto Actual: (MK_MAN) Marketing Manager | Puesto Anterior: (MK_REP) Marketing Representative
6. Empleado: Neena Yang | Puesto Actual: (AD_VP) Administration Vice President | Puesto Anterior: (AC_ACCOUNT) Public Accountant
7. Empleado: Neena Yang | Puesto Actual: (AD_VP) Administration Vice President | Puesto Anterior: (AC_MGR) Accounting Manager
8. Empleado: Payam Kaufling | Puesto Actual: (ST_MAN) Stock Manager | Puesto Anterior: (ST_CLERK) Stock Clerk

Procedimiento PL/SQL terminado correctamente.
```

5. Obtenga todos los nombres de departamentos, los nombres y apellidos de los empleados que no son jefes de departamento (Managment_ID en Departments).

```
---Ejercicio 05
SET serveroutput ON

DECLARE
    CURSOR departments_employees IS
    SELECT
        d.department_name AS nombre_departamento,
        e.first_name || ' ' || e.last_name AS nombre_empleado
    FROM employees e
    JOIN departments d
        ON e.department_id = d.department_id
    WHERE e.employee_id <> d.manager_id
    ORDER BY d.department_name, e.last_name;
    v_numero NUMBER := 0;
BEGIN
    FOR registro_cursor IN departments_employees
    LOOP
        v_numero := v_numero + 1;
        dbms_output.put_line(v_numero || '. Departamento: ' ||
        registro_cursor.nombre_departamento|| ' | Empleado: '
||
        registro_cursor.nombre_empleado);
    END LOOP;
END;
/
```

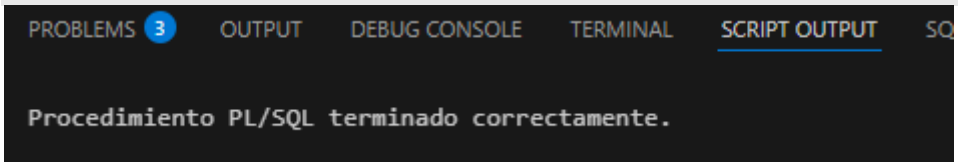




6. Obtenga el apellido del empleado, nombre de departamento e id de puesto de todos los empleados cuya dirección de localización del departamento está en la quinta avenida (columna street_address).

```
---Ejercicio 06
SET serveroutput ON

DECLARE
    CURSOR departments_employees IS
    SELECT
        e.last_name AS apellido_empleado,
        d.department_name AS departamento,
        e.job_id AS id_puesto
    FROM employees e
    JOIN departments d
```

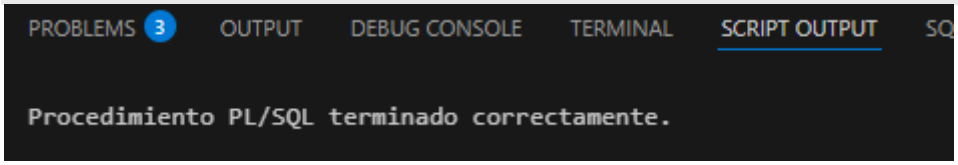
```
        ON e.department_id = d.department_id
    JOIN locations l
        ON d.location_id = l.location_id
    WHERE UPPER(l.street_address) LIKE '%5TH AVENUE%';
    v_numero NUMBER := 0;
BEGIN
    FOR registro_cursor IN departments_employees
    LOOP
        v_numero := v_numero + 1;
        dbms_output.put_line(v_numero || '. Empleado: ' ||
        registro_cursor.apellido_empleado|| ' | Departamento:
(' ||
        registro_cursor.id_puesto || ') ' ||
        registro_cursor.departamento);
    END LOOP;
END;
/
```



7. Obtenga los nombres y apellidos de todos los empleados que hayan desempeñado los mismos puestos que el empleado Juan Pérez.

```
---Ejercicio 07
SET serveroutput ON

DECLARE
    CURSOR puesto_employees IS
    SELECT DISTINCT
        e.first_name AS nombre_empleado,
        e.last_name AS apellido_empleado
    FROM employees e
    WHERE e.job_id IN (
        SELECT job_id
        FROM employees
        WHERE UPPER(first_name) = 'JUAN' AND UPPER(last_name) =
'PÉREZ'
        UNION
        SELECT job_id
        FROM job_history
        WHERE employee_id IN (
            SELECT employee_id
            FROM employees
            WHERE UPPER(first_name) = 'JUAN' AND
UPPER(last_name) = 'PÉREZ'
        )
    )
    OR e.employee_id IN (
```

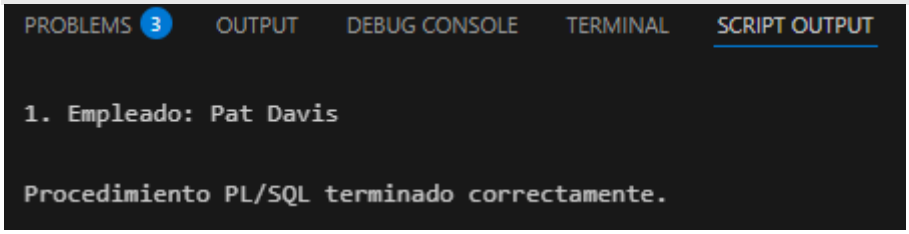
```
        SELECT employee_id
        FROM job_history
        WHERE job_id IN (
            SELECT job_id
            FROM employees
            WHERE UPPER(first_name) = 'JUAN' AND
UPPER(last_name) = 'PÉREZ'
        )
    );
    v_numero NUMBER := 0;
BEGIN
    FOR registro_cursor IN puesto_employees
    LOOP
        v_numero := v_numero + 1;
        dbms_output.put_line(v_numero || '. Empleado: ' ||
        registro_cursor.nombre_empleado|| ' ' ||
        registro_cursor.apellido_empleado);
    END LOOP;
END;
/
```



8. Obtenga los nombres de los empleados que no estén registrados en el historial de sueldo (Job_history) y que trabajen en el departamento de marketing.

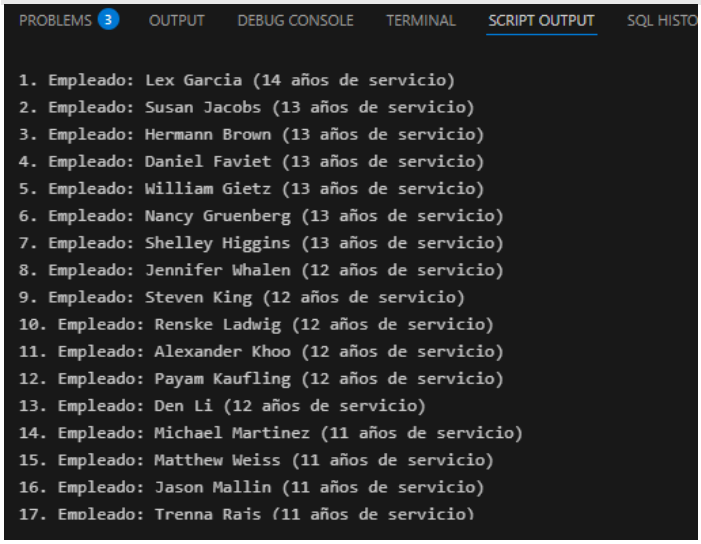
```
---Ejercicio 08
SET serveroutput ON

DECLARE
    CURSOR marketing_employees IS
    SELECT
        e.first_name || ' ' || e.last_name AS nombre_empleado
    FROM employees e
    JOIN departments d
        ON e.department_id = d.department_id
    WHERE d.department_name = 'Marketing'
    AND e.employee_id NOT IN (
        SELECT employee_id FROM job_history
    );
    v_numero NUMBER := 0;
BEGIN
    FOR registro_cursor IN marketing_employees
    LOOP
        v_numero := v_numero + 1;
        dbms_output.put_line(v_numero || '. Empleado: ' ||
        registro_cursor.nombre_empleado);
    END LOOP;
```

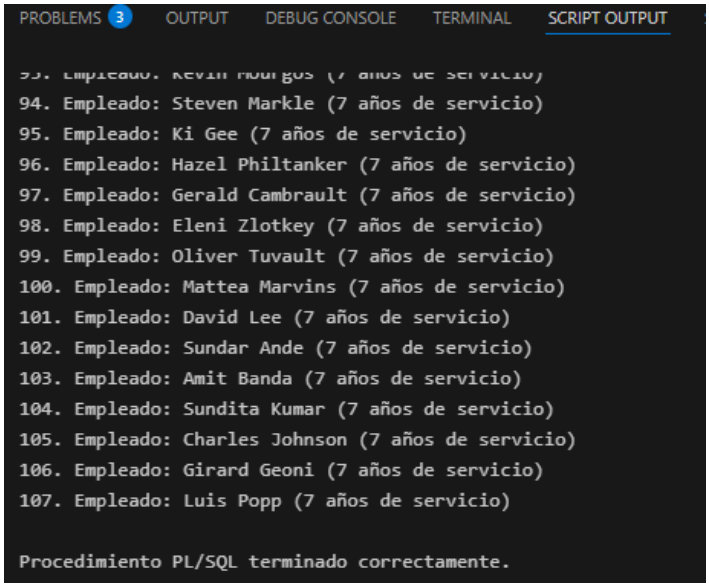
```
END;
/
```



9. Obtenga todos los nombres de empleados y su tiempo de servicio en la empresa.

```
---Ejercicio 09
SET serveroutput ON

DECLARE
    CURSOR tiempo_servicio IS
    SELECT
    e.first_name || ' ' || e.last_name AS nombre_empleado,
    TRUNC(MONTHS_BETWEEN(SYSDATE, e.hire_date) / 12) AS
anios_servicio
FROM employees e
ORDER BY anios_servicio DESC;

    v_numero NUMBER := 0;
BEGIN
    FOR registro_cursor IN tiempo_servicio
    LOOP
        v_numero := v_numero + 1;
        dbms_output.put_line(v_numero || '. Empleado: ' ||
        registro_cursor.nombre_empleado || ' (' ||
        registro_cursor.anios_servicio || ' años de
servicio)');
    END LOOP;
END;
/
```





10. Obtenga el código de empleado, apellido y nombre, código de puesto actual y nombre de puesto actual, de los 4 empleados que más han rotado de puesto desde que ingresaron a la empresa (tabla Job_History registra los cambios de puesto). Para cada uno de ellos presente como columna adicional el número de veces que han cambiado de puesto.

```
---Ejercicio 10
SET serveroutput ON

DECLARE
    CURSOR veces_puesto_employees IS
    SELECT
        e.employee_id AS id_empleado,
        e.last_name AS apellido_empleado,
        e.first_name AS nombre_empleado,
        e.job_id AS id_puesto_actual,
        j.job_title AS nombre_puesto_actual,
        COUNT(jh.job_id) AS veces_cambio_puesto
    FROM employees e
    JOIN jobs j
        ON e.job_id = j.job_id
    LEFT JOIN job_history jh
        ON e.employee_id = jh.employee_id
    GROUP BY e.employee_id, e.last_name, e.first_name,
e.job_id, j.job_title
    HAVING COUNT(jh.job_id) > 0
    ORDER BY COUNT(jh.job_id) DESC
    FETCH FIRST 4 ROWS ONLY;
    v_numero NUMBER := 0;
BEGIN
    FOR registro_cursor IN veces_puesto_employees
    LOOP
```

```
        v_numero := v_numero + 1;
        dbms_output.put_line(v_numero || '. Empleado: ' ||
        registro_cursor.apellido_empleado || ', ' ||
        registro_cursor.nombre_empleado || ' (' ||
        registro_cursor.id_empleado || ') | Puesto Actual: ' ||
        registro_cursor.nombre_puesto_actual || ' (' ||
        registro_cursor.id_puesto_actual || ') | Veces: ' ||
        registro_cursor.veces_cambio_puesto);
    END LOOP;
END;
/
```