UNIVERSIDAD AUTÓNOMA DE CHIHUAHUA

FACULTAD DE CONTADURÍA Y ADMINISTRACIÓN

Base de datos

Proyecto final

Erick Gómez Romero 339457

Anthony Steven Hernández Hernández 315952

José Daniel Yáñez Chávez 349857

11/11/2025

```
TABLA DE RETENCIONES
--Creación de tabla de retenciones
CREATE TABLE puro_dolor
(
    id NUMBER PRIMARY KEY,
    limite_inferior NUMBER,
    limite_superior NUMBER,
    cuota_fija NUMBER,
    porcentaje_retencion NUMBER
)

--Generar consecutivo de ID
CREATE SEQUENCE generar_id
    START WITH 1
    INCREMENT BY 1
    NOCACHE
    NOCYCLE;

--Generar automático el valor id sin especificar cada row
CREATE OR REPLACE TRIGGER insertar_id
BEFORE INSERT ON puro_dolor
FOR EACH ROW
BEGIN
    :NEW.ID := generar_id.NEXTVAL;
END;

--Script para insertar valores
CREATE OR REPLACE PROCEDURE insertar_limites
(
    v_limite_inferior IN NUMBER,
    v_limite_superior IN NUMBER,
    v_cuota_fija IN NUMBER,
    v_porcentaje_retencion IN NUMBER
)
AS

BEGIN
    INSERT INTO puro_dolor
    (
        limite_inferior, limite_superior, cuota_fija, porcentaje_retencion
    )
    VALUES
    (
        v_limite_inferior,
        v_limite_superior,
        v_cuota_fija,
        v_porcentaje_retencion
```

```
    );
END insertar_limites;

-Insertar valores en tabla
EXECUTE insertar_limites(0.01, 496.07, 0, .0192);
EXECUTE insertar_limites(496.08, 4210.41, 9.52, .064);
EXECUTE insertar_limites(4210.42, 7399.42, 247.23, .1088);
EXECUTE insertar_limites(7399.43, 8601.5, 594.24, .16);
EXECUTE insertar_limites(8601.51, 10298.35, 786.55, .1792);
EXECUTE insertar_limites(10298.36, 20770.29, 1090.62, .2136);
EXECUTE insertar_limites(20770.3, 32736.83, 3327.42, .2352);
EXECUTE insertar_limites(32736.84, 999999, 6141.95, .3);

--Consulta a la tabla
SELECT * FROM puro_dolor ORDER BY ID
```

	ID	LIMITE_INFERIOR	LIMITE_SUPERIOR	CUOTA_FIJA	PORCENTAJE_RETE
1	1	0.01	496.07	0	0.0192
2	2	496.08	4210.41	9.52	0.064
3	3	4210.42	7399.42	247.23	0.1088
4	4	7399.43	8601.5	594.24	0.16
5	5	8601.51	10298.35	786.55	0.1792
6	6	10298.36	20770.29	1090.62	0.2136
7	7	20770.3	32736.83	3327.42	0.2352
8	8	32736.84	999999	6141.95	0.3

```
TABLA DE IMPUESTOS
--Crear tabla que registra los cálculos
CREATE TABLE impuestos_calculados
(
    id NUMBER PRIMARY KEY,
    nombre VARCHAR2(100),
    sueldo_gravable NUMBER,
    impuesto_cobrado NUMBER,
    porcentaje_aplicado NUMBER,
    procedimiento_ejecutado VARCHAR2(100),
    fecha_insercion DATE
);
```

```sql
--Primera iteración procedimiento de calculo de impuestos
CREATE OR REPLACE PROCEDURE calcular_impuesto
(
    v_gravable IN NUMBER,
    v_nombre IN VARCHAR2
)
AS
    v_limite_inferior NUMBER;
    v_cuota_fija NUMBER;
    v_porcentaje_retencion NUMBER;
    v_calculo_impuesto NUMBER;
BEGIN
    SELECT limite_inferior, cuota_fija, porcentaje_retencion
    INTO v_limite_inferior, v_cuota_fija, v_porcentaje_retencion
    FROM puro_dolor
    WHERE v_gravable BETWEEN limite_inferior AND limite_superior;

    v_calculo_impuesto := ((v_gravable - v_limite_inferior) * v_porcentaje_retencion)
+ v_cuota_fija;

    DBMS_OUTPUT.PUT_LINE('tasa utilizada ' || v_porcentaje_retencion);
    DBMS_OUTPUT.PUT_LINE('El impuesto de ' || v_nombre || ' es de ' ||
v_calculo_impuesto);

END calcular_impuesto;

--Hasta aquí me puse a probar que funcionara bien para generar los triggers
EXECUTE calcular_impuesto(10000, 'Erick Romero')

--Secuencia para la tabla impuestos
CREATE SEQUENCE generar_impuesto_id
    START WITH 1
    INCREMENT BY 1
    NOCACHE
    NOCYCLE;

--Trigger para generar id en tabla impuesto
CREATE OR REPLACE TRIGGER impuestos_insertar_id
BEFORE INSERT ON impuestos_calculados
FOR EACH ROW
BEGIN
    :NEW.ID := generar_impuesto_id.NEXTVAL;
END;
```

```
--Trigger para generar usuario que inserta y fecha de inserción
CREATE OR REPLACE TRIGGER creacion_impuesto
BEFORE INSERT ON impuestos_calculados
FOR EACH ROW
BEGIN
    :NEW.PROCEDIMIENTO_EJECUTADO := USER;
    :NEW.FECHA_INSERCION := SYSDATE;
END creacion_impuesto;

--Segunda iteración script de insercion de datos
CREATE OR REPLACE PROCEDURE calcular_impuesto
(
    v_gravable IN NUMBER,
    v_nombre IN VARCHAR2
)
AS
    v_limite_inferior NUMBER;
    v_cuota_fija NUMBER;
    v_porcentaje_retencion NUMBER;
    v_calculo_impuesto NUMBER;

    e_nombre_invalido EXCEPTION;
BEGIN
    IF v_nombre IS NULL OR REGEXP_LIKE(v_nombre, '^[0-9]+$') THEN
        RAISE e_nombre_invalido;
    END IF;

    SELECT limite_inferior, cuota_fija, porcentaje_retencion
    INTO v_limite_inferior, v_cuota_fija, v_porcentaje_retencion
    FROM puro_dolor
    WHERE v_gravable BETWEEN limite_inferior AND limite_superior;

    v_calculo_impuesto := ((v_gravable - v_limite_inferior) * v_porcentaje_retencion)
+ v_cuota_fija;

INSERT INTO impuestos_calculados(nombre,
sueldo_gravable,impuesto_cobrado,porcentaje_aplicado)
VALUES
(
    v_nombre, v_gravable, v_calculo_impuesto, v_porcentaje_retencion
);

EXCEPTION
    WHEN e_nombre_invalido THEN
```

```
        DBMS_OUTPUT.PUT_LINE('El nombre debe ser tipo cadena');
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Rango de sueldo fuera de la tabla ' || v_gravable);
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Error inesperado ' || SQLERRM);

END calcular_impuesto;

--Ejecución de procedimiento
EXECUTE calcular_impuesto(300, 'Erick Romero')
EXECUTE calcular_impuesto(3000, 'Erick Romero')
EXECUTE calcular_impuesto(5000, 'Erick Romero')
EXECUTE calcular_impuesto(8000, 'Erick Romero')
EXECUTE calcular_impuesto(10000, 'Erick Romero')
EXECUTE calcular_impuesto(15000, 'Erick Romero')
EXECUTE calcular_impuesto(25000, 'Erick Romero')
EXECUTE calcular_impuesto(500000, 'Erick Romero')

--Consulta para comprobar funcionamiento
SELECT * FROM impuestos_calculados ORDER BY sueldo_gravable
```

ID	NOMBRE	SUELDO_GRAVABLE	IMPUESTO_COBRAD	PORCENTAJE_APLIC	PROCEDIMIENTO_EJECUTA	FECHA_INSERCION
2	Erick Romero	300	5.759808	0.0192	A339457_SCHEMA_PNR59	11/11/2025, 5:56:04 PM
3	Erick Romero	3000	169.77088	0.064	A339457_SCHEMA_PNR59	11/11/2025, 5:56:12 PM
4	Erick Romero	5000	333.136304	0.1088	A339457_SCHEMA_PNR59	11/11/2025, 5:56:20 PM
5	Erick Romero	8000	690.3312	0.16	A339457_SCHEMA_PNR59	11/11/2025, 5:56:28 PM
1	Erick Romero	10000	1037.159408	0.1792	A339457_SCHEMA_PNR59	11/11/2025, 5:35:48 PM
6	Erick Romero	15000	2094.890304	0.2136	A339457_SCHEMA_PNR59	11/11/2025, 5:56:35 PM
7	Erick Romero	25000	4322.24544	0.2352	A339457_SCHEMA_PNR59	11/11/2025, 5:56:46 PM
8	Erick Romero	500000	146320.898	0.3	A339457_SCHEMA_PNR59	11/11/2025, 5:57:00 PM

```
--Test del manejo de excepciones
EXECUTE calcular_impuesto(5000000, 'Erick Romero')
```

```
SQL> EXECUTE calcular_impuesto(5000000, 'Erick Romero')

Rango de sueldo fuera de la tabla 5000000

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.004
```

```
EXECUTE calcular_impuesto(400, 500)
```

```
SQL> EXECUTE calcular_impuesto(400, 500)


El nombre debe ser tipo cadena


PL/SQL procedure successfully completed.

Elapsed: 00:00:00.006
```