# CONCEPTOS DE BASES DE DATOS II

Programador Universitario – Lic. en Informática – Ing. en Informática
Facultad de Ciencias Exactas y Tecnología – UNT

## Trabajo Práctico Nro. 6 - Ciclo 2023

**Stored Procedure**

**Ejercicio 1:** Para mejorar y automatizar el funcionamiento de la base de datos “Hospital” realice las siguientes tareas:

a)

```
CREATE OR REPLACE PROCEDURE persona_alta(
        IN p_nombre character varying,
        IN p_apellido character varying,
        IN p_dni character varying,
        IN p_fecha date,
        IN p_domicilio character varying,
        IN p_telefono character varying)
LANGUAGE 'plpgsql'
    AS $BODY$
    DECLARE
        v_id_persona integer;
    BEGIN
        IF p_nombre IS NULL OR p_nombre = '' THEN
                RAISE EXCEPTION 'El nombre es obligatorio';
        END IF;
        IF p_apellido IS NULL OR p_apellido = '' THEN
                RAISE EXCEPTION 'El apellido es obligatorio';
        END IF;
        IF p_dni IS NULL OR p_dni = '' THEN
                RAISE EXCEPTION 'Debe ingresar un número de DNI';
        END IF;
        IF EXISTS(SELECT * FROM persona WHERE dni = p_dni) THEN
                RAISE EXCEPTION 'Ya existe una persona con el DNI ingresado';
        END IF;

        SELECT MAX(id_persona) + 1 INTO v_id_persona FROM persona;
        INSERT INTO persona
        VALUES (v_id_persona, p_nombre, p_apellido, p_dni, p_fecha, p_domicilio, p_telefono);
        RAISE NOTICE 'Se inserto exitosamente a la persona %, %', p_apellido, p_nombre;
        EXCEPTION
                WHEN OTHERS THEN
                RAISE EXCEPTION 'Error en la inserción de persona: %', SQLERRM;
    END;
    $BODY$;
```

## Trabajo Práctico Nro. 6 - Ciclo 2023

b)

```
CREATE OR REPLACE PROCEDURE empleado_alta(
    IN p_dni character varying,
    IN p_especialidad character varying,
    IN p_cargo character varying,
    IN p_fecha_ingreso date,
    IN p_sueldo numeric(9,2),
    IN p_fecha_baja date)
LANGUAGE 'plpgsql'
AS $BODY$
DECLARE
    v_id_empleado integer;
    v_id_especialidad integer;
    v_id_cargo integer;
BEGIN
    IF NOT EXISTS(SELECT * FROM persona WHERE dni = p_dni) THEN
            RAISE EXCEPTION 'No existe una persona con el DNI ingresado';
    END IF;
    IF NOT EXISTS(SELECT * FROM especialidad WHERE especialidad = p_especialidad) THEN
            RAISE EXCEPTION 'No existe la especialidad ingresada';
    END IF;
    IF NOT EXISTS(SELECT * FROM cargo WHERE cargo = p_cargo) THEN
            RAISE EXCEPTION 'No existe el cargo ingresado';
    END IF;
    IF p_fecha_baja < p_fecha_ingreso THEN
            RAISE EXCEPTION 'La fecha de baja no puede ser anterior a la fecha de ingreso';
    END IF;

    SELECT id_persona INTO v_id_empleado FROM persona WHERE dni = p_dni;
    SELECT id_especialidad INTO v_id_especialidad FROM especialidad WHERE especialidad =
p_especialidad;
    SELECT id_cargo INTO v_id_cargo FROM cargo WHERE cargo = p_cargo;

    INSERT INTO empleado VALUES (v_id_empleado, v_id_especialidad, v_id_cargo,
p_fecha_ingreso, p_sueldo, p_fecha_baja);
    RAISE NOTICE 'Se inserto exitosamente el empleado de DNI: %', p_dni;
    EXCEPTION
            WHEN OTHERS THEN
            RAISE EXCEPTION 'Error en la inserción de empleado: %', SQLERRM;
END;
$BODY$;
```

## Trabajo Práctico Nro. 6 - Ciclo 2023

c)

```
CREATE OR REPLACE PROCEDURE factura_modifica_saldo(
    IN p_id_factura bigint, IN p_monto numeric(10,2) )
LANGUAGE 'plpgsql'
AS $BODY$
BEGIN
    IF NOT EXISTS(SELECT * FROM factura WHERE id_factura = p_id_factura) THEN
            RAISE EXCEPTION 'No existe el número de factura ingresado';
    END IF;
    IF p_monto IS NULL OR p_monto < 0 THEN
            RAISE EXCEPTION 'Se debe ingresar un monto positivo como pago';
    END IF;
    IF p_monto > (SELECT saldo FROM factura WHERE id_factura = p_id_factura) THEN
            RAISE EXCEPTION 'No se puede realizar un pago mayor a lo que adeuda';
    END IF;

    UPDATE factura SET saldo = saldo - p_monto WHERE id_factura = p_id_factura;
    RAISE NOTICE 'Se modificó exitosamente el saldo de la factura: %', p_id_factura;
    EXCEPTION
            WHEN OTHERS THEN
            RAISE EXCEPTION 'Error en la modificación de factura: %', SQLERRM;
END; $BODY$;
```

d)

```
CREATE OR REPLACE PROCEDURE medicamento_modifica_por_laboratorio(
    IN p_laboratorio character varying,
    IN p_aumento numeric(4,2) ) -- puede aumentar hasta un 99.99%
LANGUAGE 'plpgsql'
AS $BODY$
BEGIN
    IF NOT EXISTS(SELECT * FROM laboratorio WHERE laboratorio = p_laboratorio) THEN
            RAISE EXCEPTION 'No existe el laboratorio ingresado';
    END IF;
    IF p_aumento IS NULL OR p_aumento < 0 OR p_aumento > 99.99 THEN
            RAISE EXCEPTION 'Se debe ingresar un monto positivo hasta 99.99';
    END IF;

    UPDATE medicamento SET precio = precio + precio * (p_aumento / 100)
    WHERE id_laboratorio = (SELECT id_laboratorio FROM laboratorio WHERE laboratorio =
p_laboratorio);
    RAISE NOTICE 'Se modificaron exitosamente los precios del laboratorio: %', p_laboratorio;
    EXCEPTION
            WHEN OTHERS THEN
            RAISE EXCEPTION 'Error en la modificación de precios: %', SQLERRM;
END; $BODY$;
```

## Trabajo Práctico Nro. 6 - Ciclo 2023

e)

```
CREATE OR REPLACE PROCEDURE medicamento_eliminar(
    IN p_medicamento character varying(50) )
LANGUAGE 'plpgsql'
AS $BODY$
BEGIN
    IF NOT EXISTS(SELECT * FROM medicamento WHERE nombre = p_medicamento) THEN
            RAISE EXCEPTION 'No existe el medicamento ingresado';
    END IF;

    DELETE FROM compra WHERE id_medicamento IN (SELECT id_medicamento
                                                FROM medicamento
                                                WHERE nombre = p_medicamento);

    DELETE FROM tratamiento WHERE id_medicamento IN (SELECT id_medicamento
                                                     FROM medicamento
                                                     WHERE nombre = p_medicamento);

    DELETE FROM medicamento WHERE nombre = p_medicamento;
    RAISE NOTICE 'Se eliminó exitosamente el medicamento: %', p_medicamento;
    EXCEPTION
            WHEN OTHERS THEN
            RAISE EXCEPTION 'Error en la eliminación de medicamento: %', SQLERRM;
END;
$BODY$;
```

**Ejercicio 2:**

a)

```
CREATE OR REPLACE PROCEDURE paciente_obtener(
    IN p_dni character varying,
    OUT p_nombre character varying,
    OUT p_apellido character varying)
LANGUAGE 'plpgsql'
AS $BODY$
BEGIN
    SELECT nombre, apellido INTO p_nombre, p_apellido FROM persona WHERE dni = p_dni;
END;
$BODY$;
```

## Trabajo Práctico Nro. 6 - Ciclo 2023

b)

```
CREATE OR REPLACE PROCEDURE medicamento_precio_stock(
    IN p_medicamento character varying,
    OUT p_nombre character varying,
    OUT p_precio numeric(8,2),
    OUT p_stock integer)
LANGUAGE 'plpgsql'
AS $BODY$
BEGIN
    SELECT nombre, precio, stock INTO p_nombre, p_precio, p_stock FROM medicamento WHERE
nombre = p_medicamento;
END;
$BODY$;
```

c)

```
CREATE OR REPLACE PROCEDURE paciente_deuda(
    IN p_dni character varying,
    OUT p_monto numeric)
LANGUAGE 'plpgsql'
AS $BODY$
DECLARE
    v_id_paciente integer;
BEGIN
    SELECT id_paciente INTO v_id_paciente FROM paciente pa
    INNER JOIN persona pe ON pa.id_paciente = pe.id_persona WHERE dni = p_dni;
    SELECT SUM(saldo) INTO p_monto FROM factura WHERE id_paciente = v_id_paciente;
END;
$BODY$;
```

d)

```
CREATE OR REPLACE PROCEDURE cama_cantidad_mantenimiento(
    IN p_id_cama smallint,
    OUT p_cantidad smallint)
LANGUAGE 'plpgsql'
AS $BODY$
BEGIN
    SELECT COUNT(id_cama) INTO p_cantidad FROM mantenimiento_cama WHERE id_cama =
p_id_cama;
END;
$BODY$;
```

# CONCEPTOS DE BASES DE DATOS II

Programador Universitario – Lic. en Informática – Ing. en Informática
Facultad de Ciencias Exactas y Tecnología – UNT

## Trabajo Práctico Nro. 6 - Ciclo 2023

**Ejercicio 3:**

a)

```
CREATE OR REPLACE PROCEDURE obra_social_listado()
LANGUAGE 'plpgsql'
AS $BODY$
DECLARE
    cursor_os CURSOR FOR
            SELECT * FROM obra_social;
    os_row obra_social%ROWTYPE;
BEGIN
    OPEN cursor_os;
    LOOP
            FETCH cursor_os INTO os_row;
            EXIT WHEN NOT FOUND;
            RAISE NOTICE 'ID: %, Sigla: %, Nombre: %, Dirección: %, Localidad: %, Provincia: %,
Teléfono: %', os_row.id_obra_social, os_row.sigla, os_row.nombre, os_row.direccion,
os_row.localidad, os_row.provincia, os_row.telefono;
    END LOOP;
    CLOSE cursor_os;
END;
$BODY$;
```

b)

```
CREATE OR REPLACE PROCEDURE cama_listado_ok()
LANGUAGE 'plpgsql'
AS $BODY$
DECLARE
    cursor_cama CURSOR FOR
            SELECT * FROM cama WHERE estado = 'OK';
    cama_row cama%ROWTYPE;
BEGIN
    OPEN cursor_cama;
    LOOP
            FETCH cursor_cama INTO cama_row;
            EXIT WHEN NOT FOUND;
            RAISE NOTICE 'ID: %, Tipo: %, Estado: %, Habitación: %', cama_row.id_cama,
cama_row.tipo, cama_row.estado, cama_row.id_habitacion;
    END LOOP;
    CLOSE cursor_cama;
END;
$BODY$;
```

## Trabajo Práctico Nro. 6 - Ciclo 2023

c)

```
CREATE OR REPLACE PROCEDURE medicamentos_poco_stock()
LANGUAGE 'plpgsql'
AS $BODY$
DECLARE
    cursor_medicamento CURSOR FOR
            SELECT * FROM medicamento WHERE stock < 50;
    med_row medicamento%ROWTYPE;
BEGIN
    OPEN cursor_medicamento;
    LOOP
            FETCH cursor_medicamento INTO med_row;
            EXIT WHEN NOT FOUND;
            RAISE NOTICE 'ID: %, Nombre: %, Presentación: %, Precio: %, Stock: %',
med_row.id_medicamento, med_row.nombre, med_row.presentacion, med_row.precio,
med_row.stock;
    END LOOP;
    CLOSE cursor_medicamento;
END;
$BODY$;
```

d)

```
CREATE OR REPLACE PROCEDURE consulta_listado_por_fecha(
    IN p_fecha date)
LANGUAGE 'plpgsql'
AS $BODY$
DECLARE
    cursor_consulta CURSOR FOR
            SELECT * FROM consulta WHERE fecha = p_fecha;
    cons_row consulta%ROWTYPE;
BEGIN
    OPEN cursor_consulta;
    LOOP
            FETCH cursor_consulta INTO cons_row;
            EXIT WHEN NOT FOUND;
            RAISE NOTICE 'Paciente: %, Empleado: %, Fecha: %, Consultorio: %, Hora: %,
Resultado: %', cons_row.id_paciente, cons_row.id_empleado, cons_row.fecha,
cons_row.id_consultorio, cons_row.hora, cons_row.resultado;
    END LOOP;
    CLOSE cursor_consulta;
END;
$BODY$;
```

## Trabajo Práctico Nro. 6 - Ciclo 2023

e)

```
CREATE OR REPLACE PROCEDURE estudio_por_paciente(
    IN p_dni character varying)
LANGUAGE 'plpgsql'
AS $BODY$
DECLARE
    cursor_estudio CURSOR FOR
            SELECT p.nombre, apellido, e.nombre, er.fecha
            FROM paciente
            INNER JOIN persona p ON id_persona = id_paciente
            INNER JOIN estudio_realizado er USING(id_paciente)
            INNER JOIN estudio e USING(id_estudio)
            WHERE dni = p_dni;
    v_nombre character varying(100);
    v_apellido character varying(100);
    v_estudio character varying(100);
    v_fecha date;
BEGIN
    OPEN cursor_estudio;
    LOOP
            FETCH cursor_estudio INTO v_nombre, v_apellido, v_estudio, v_fecha;
            EXIT WHEN NOT FOUND;
            RAISE NOTICE 'Nombre: %, Apellido: %, Estudio: %, Fecha: %', v_nombre, v_apellido,
v_estudio, v_fecha;
    END LOOP;
    CLOSE cursor_estudio;
END;
$BODY$;
```

f)

```
CREATE OR REPLACE PROCEDURE empleado_por_turno(IN p_turno character varying)
LANGUAGE 'plpgsql'
AS $BODY$
DECLARE
    cursor_turno CURSOR FOR
            SELECT nombre, apellido, telefono, turno
            FROM empleado
            INNER JOIN persona ON id_persona = id_empleado
            INNER JOIN trabajan USING(id_empleado)
            INNER JOIN turno USING(id_turno)
            WHERE fin IS NULL AND turno = p_turno;
    v_nombre character varying(100);
    v_apellido character varying(100);
    v_telefono character varying(100);
    v_turno character varying(25);
```



```
BEGIN
    OPEN cursor_turno;
    LOOP
            FETCH cursor_turno INTO v_nombre, v_apellido, v_telefono, v_turno;
            EXIT WHEN NOT FOUND;
            RAISE NOTICE 'Nombre: %, Apellido: %, Teléfono: %, Turno: %', v_nombre, v_apellido,
v_telefono, v_turno;
    END LOOP;
    CLOSE cursor_turno;
END; $BODY$;
```

**Ejercicio 4:**

a)

```
CREATE OR REPLACE PROCEDURE medicamento_laboratorio_clasificacion(
    IN p_laboratorio character varying,
    IN p_clasificacion character varying)
LANGUAGE 'plpgsql'
AS $BODY$
DECLARE
    cursor_meds CURSOR FOR
            SELECT nombre, presentacion, precio, stock FROM medicamento
            INNER JOIN laboratorio USING(id_laboratorio)
            INNER JOIN clasificacion USING(id_clasificacion)
            WHERE laboratorio = p_laboratorio AND clasificacion LIKE p_clasificacion
                AND precio < (SELECT AVG(precio) FROM medicamento
                    INNER JOIN laboratorio USING(id_laboratorio)
                    INNER JOIN clasificacion USING(id_clasificacion)
                    WHERE laboratorio = p_laboratorio AND clasificacion LIKE p_clasificacion);
    v_nombre character varying(50);
    v_presentacion character varying(50);
    v_precio numeric(8,2);
    v_stock integer;
BEGIN
    OPEN cursor_meds;
    LOOP
            FETCH cursor_meds INTO v_nombre, v_presentacion, v_precio, v_stock;
            EXIT WHEN NOT FOUND;
            RAISE NOTICE 'Nombre: %, Presentación: %, Precio: %, Stock: %', v_nombre,
v_presentacion, v_precio, v_stock;
    END LOOP;
    CLOSE cursor_meds;
END; $BODY$;
```

## Trabajo Práctico Nro. 6 - Ciclo 2023

b)

```
CREATE OR REPLACE PROCEDURE factura_top_ten()
LANGUAGE 'plpgsql'
AS $BODY$
DECLARE
    cursor_deuda CURSOR FOR
            SELECT SUM(monto), nombre, apellido FROM persona
            INNER JOIN factura ON id_persona = id_paciente
            GROUP BY id_paciente, nombre, apellido
            ORDER BY SUM(monto) DESC
            LIMIT 10;
    v_nombre character varying(100);
    v_apellido character varying(100);
    v_monto numeric(10,2);
BEGIN
    OPEN cursor_deuda;
    LOOP
            FETCH cursor_deuda INTO v_monto, v_nombre, v_apellido;
            EXIT WHEN NOT FOUND;
            RAISE NOTICE 'Deuda: %, Nombre: %, Apellido: %', v_monto, v_nombre, v_apellido;
    END LOOP;
    CLOSE cursor_deuda;
END;
$BODY$;
```