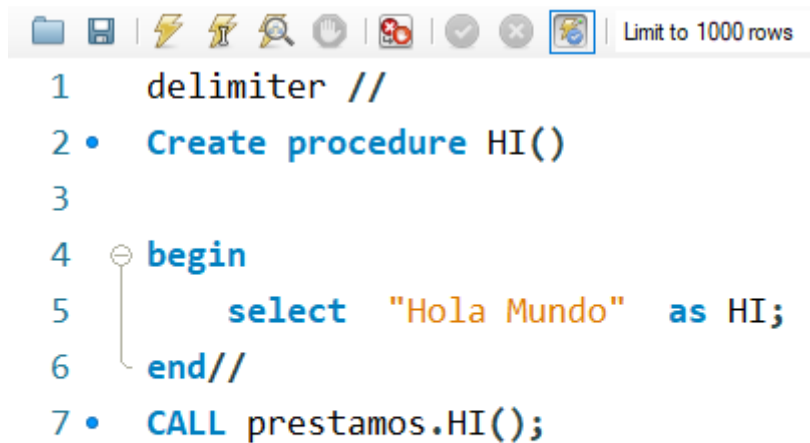


Taller de procedimientos almacenados

1. Escribe un procedimiento que no tenga ningún parámetro de entrada ni de salida y que muestre el texto ¡Hola mundo!



The screenshot shows a SQL editor with a toolbar at the top containing icons for file operations, execution, and search. The text in the editor is as follows:

```
1 delimiter //
2 • Create procedure HI()
3
4 begin
5     select "Hola Mundo" as HI;
6 end//
7 • CALL prestamos.HI();
```



The screenshot shows a 'Result Grid' window with a toolbar at the top containing icons for grid view, filter, and row selection. The grid contains the following data:

	HI
▶	Hola Mundo

2. Escribe un procedimiento que reciba un número real de entrada, que representa el valor de la nota de un alumno, y muestre un mensaje indicando qué nota ha obtenido teniendo en cuenta las siguientes condiciones: [0,5) = Insuficiente [5,6) = Aprobado [6, 7) = Bien [7, 9) = Notable [9, 10] = Sobresaliente En cualquier otro caso la nota no será válida.

```
1  delimiter //
```

```
2  • Create procedure Notas(in calificacion double)
```

```
3  begin
```

```
4      declare resultado double;
```

```
5      IF calificacion between 0 and 5 then
```

```
6          select "Insuficiente" as Nota_Obtendida;
```

```
7      elseif calificacion between 5.1 and 6 then
```

```
8          select "Aprobado" as Nota_Obtendida;
```

```
9      elseif calificacion between 6.1 and 7 then
```

```
10         select "Bien" as Nota_Obtendida;
```

```
11     elseif calificacion between 7.1 and 9 then
```

```
12         select "Notable" as Nota_Obtendida;
```

```
13     elseif calificacion between 9.1 and 10 then
```

```
14         select "Sobresaliente" as Nota_Obtendida;
```

```
15     END IF;
```

```
16 end//
```

17 • CALL prestamos.Notas(5.82);

<	
Result Grid	
Filter Rows:	
Export:	
Wrap Cell Cor	
	Nota_Obtendida
▶	Aprobado

3. Escriba un procedimiento llamado cantidadProductos que reciba como entrada el nombre del tipo de producto y devuelva el número de productos que existen dentro de esa categoría.

The screenshot shows a SQL IDE with two tabs: 'productos' and 'tipoproducto'. The 'tipoproducto' tab is active, displaying the following SQL code:

```

1  DELIMITER //
2
3  • CREATE PROCEDURE cantidadProductos(IN tipoProducto VARCHAR(200), OUT cantidad INT)
4  BEGIN
5      SELECT COUNT(*)
6      INTO cantidad
7      FROM productos
8      JOIN tipoproducto ON productos.tipo_id = tipoproducto.id
9      WHERE tipoproducto.nombre = tipoProducto;
10 END //
11

```

Below the code editor, the 'Result Grid' is visible, showing a single row with the value 2 for the parameter @cantidad.

	@cantidad
▶	2

```

DELIMITER ;
CALL cantidadProductos('licores', @cantidad);
SELECT @cantidad;

```

The screenshot shows the 'productos' table in the SQL IDE. The table has columns: id, nombre, and tipo_id. The data is as follows:

	id	nombre	tipo_id
	0	aguardi...	5
	1	leche	1
	2	yogurt	1
	3	res	2
	4	cerdo	2
▶	5	ron	5
*	NULL	NULL	NULL

Below the table, there is a tab labeled 'productos 1'.

4. Escribe un procedimiento que se llame preciosProductos, que reciba como parámetro de entrada el nombre del tipo de producto y devuelva como salida tres

parámetros. El precio máximo, el precio mínimo y la media de los productos que existen en esa categoría.

```
14 • ALTER TABLE productos ADD COLUMN precio DECIMAL(10,2);
```

Result Grid				
Filter Rows:				
	id	nombre	tipo_id	precio
	0	aguardiente	5	100000.00
	1	leche	1	4000.00
	2	yogurt	1	2000.00
	3	res	2	7000.00
	4	cerdo	2	8000.00
	5	ron	5	15000.00
	6	wisky	5	120000.00
	7	kumis	1	3500.00
	8	manzana	3	400.00
	9	pera	3	500.00
	10	sandia	3	1000.00
	NULL	NULL	NULL	NULL

```

1 DELIMITER //
2 • CREATE PROCEDURE preciosProductos(IN tipoProducto VARCHAR(200))
3 BEGIN
4     SELECT
5         MAX(p.precio) AS precioMaximo,
6         MIN(p.precio) AS precioMinimo,
7         AVG(p.precio) AS precioPromedio
8     FROM productos p
9     JOIN tipoproducto t ON p.tipo_id = t.id
10    WHERE t.nombre = tipoProducto;
11 END //

```

```

13
14 • CALL preciosProductos('frutas');
15
16

```

Result Grid			
Filter Rows:			
Export:			
	precioMaximo	precioMinimo	precioPromedio
	1000.00	400.00	633.333333

5. Realice un procedimiento que se llame funcionIVA que incluya una función que calcule el total con el incremento del iva.

```

DELIMITER //

CREATE FUNCTION calcularMontoConIVA(montoBase DECIMAL(10, 2), porcentajeIVA DECIMAL(5, 2))
RETURNS DECIMAL(10, 2)
DETERMINISTIC
BEGIN
    -- Calcular el monto total con IVA
    RETURN montoBase + (montoBase * (porcentajeIVA / 100));
END //

DELIMITER ;

CREATE PROCEDURE calcularIVA(montoBase DECIMAL(10, 2), porcentajeIVA DECIMAL(5, 2))
BEGIN
    -- Declarar una variable para almacenar el monto total con IVA
    DECLARE montoTotalConIVA DECIMAL(10, 2);

    -- Llamar a la función calcularMontoConIVA para obtener el monto total con IVA
    SET montoTotalConIVA = calcularMontoConIVA(montoBase, porcentajeIVA);

    -- Mostrar el resultado
    SELECT CONCAT('El total con IVA es: ', montoTotalConIVA) AS resultado;
END //

DELIMITER ;

CALL calcularIVA(200000, 19);

```

result	resultado
▶	El total con IVA es: 238000.00

6. Escribe un procedimiento que reciba el nombre de un país como parámetro de entrada y realice una consulta sobre la tabla sucursal para obtener todos las sucursales que existen en la tabla de ese país.

```

DELIMITER //

CREATE PROCEDURE obtenerSucursalesPorPais(IN nombrePais VARCHAR(100))
) BEGIN
    -- Selecciona todas las sucursales en el país especificado
    SELECT id_sucursal, nombre_sucursal, direccion, ciudad, pais
    FROM Sucursal
    WHERE pais = nombrePais;
- END //

DELIMITER ;

```

```
CALL obtenerSucursalesPorPais('Italia');
```

	id_sucursal	nombre_sucursal	direccion	ciudad	pais
►	9	Sucursal I	Via Roma 22	Roma	Italia

7. Una vez creada la tabla se decide añadir una nueva columna a la tabla llamada edad que será un valor calculado a partir de la columna fecha_nacimiento. Escriba la sentencia SQL necesaria para modificar la tabla y añadir la nueva columna.

```

CREATE TABLE Empleados (
    id_empleado INT PRIMARY KEY AUTO_INCREMENT,
    nombre VARCHAR(100),
    fecha_nacimiento DATE
);

ALTER TABLE Empleados
ADD COLUMN edad INT;

-- Insertar datos en la tabla Empleados
INSERT INTO Empleados (nombre, fecha_nacimiento)
VALUES
('Juan Pérez', '1985-04-15'),
('María Gómez', '1990-07-22'),
('Carlos Ramírez', '1978-02-05'),
('Laura Martínez', '1995-11-30'),
('Ana Sánchez', '2000-03-18');

```

8. Escriba una función llamada `calcularEdad` que reciba una fecha y devuelva el número de años que han pasado desde la fecha actual hasta la fecha pasada como parámetro:

```
DELIMITER //
```

```
CREATE FUNCTION calcularEdad(fecha_nacimiento DATE)
RETURNS INT
DETERMINISTIC
BEGIN
    DECLARE edad INT;

    -- Calcular la edad usando la función TIMESTAMPDIFF
    SET edad = TIMESTAMPDIFF(YEAR, fecha_nacimiento, CURDATE());

    RETURN edad;
END //
```

```
DELIMITER ;
```

```
SELECT calcularEdad('1999-09-07') AS edad;
```

	edad
▶	25

Escriba un procedimiento que permita calcular la edad de todos los usuarios que ya existen en la tabla. Para esto será necesario crear un procedimiento llamado `actualizarColumnaEdad` que calcule la edad de cada usuario y actualice la tabla. Este procedimiento hará uso de la función `calcularEdad` que hemos creado en el paso anterior

```

DELIMITER //

CREATE PROCEDURE actualizarColumnaEdad()
BEGIN
    -- Actualizar la columna edad de todos los empleados
    UPDATE Empleados
    SET edad = calcularEdad(fecha_nacimiento);
END //

DELIMITER ;
CALL actualizarColumnaEdad();
SELECT id_empleado, nombre, fecha_nacimiento, edad
FROM Empleados;

```

id_empleado	nombre	fecha_nacimiento	edad
2	María Gómez	1990-07-22	34
3	Carlos Ramírez	1978-02-05	46
4	Laura Martínez	1995-11-30	28
5	Ana Sánchez	2000-03-18	24
NULL	NULL	NULL	NULL

10. Escribe un procedimiento almacenado para su proyecto integrador que sea útil.

```

DELIMITER //

CREATE PROCEDURE insertarCarro(
    IN p_marca VARCHAR(50),
    IN p_color VARCHAR(30),
    IN p_modelo VARCHAR(50),
    IN p_matricula VARCHAR(20)
)
BEGIN
    INSERT INTO carros (marca, color, modelo, matricula)
    VALUES (p_marca, p_color, p_modelo, p_matricula);
END //

DELIMITER ;

CALL insertarCarro('Toyota', 'Rojo', 'Corolla', 'ABC123');
CALL insertarCarro('Honda', 'Azul', 'Civic', 'XYZ789');

SELECT * FROM carros;

```