

DELIMITER //

-- 1. Insertar nuevo fabricante

```
CREATE PROCEDURE insertar_fabricante(IN nombre_fabricante VARCHAR(100))
BEGIN
    INSERT INTO fabricante (nombre) VALUES (nombre_fabricante);
END //
```

-- 2. Insertar nuevo producto asociado a fabricante

```
CREATE PROCEDURE insertar_producto(
    IN nombre_producto VARCHAR(100),
    IN precio DECIMAL(10,2),
    IN codigo_fabricante INT
)
BEGIN
    INSERT INTO producto (nombre, precio, codigo_fabricante)
    VALUES (nombre_producto, precio, codigo_fabricante);
END //
```

-- 3. Actualizar precio de producto por código

```
CREATE PROCEDURE actualizar_precio_producto(
    IN codigo_prod INT,
    IN nuevo_precio DECIMAL(10,2)
)
BEGIN
    UPDATE producto
    SET precio = nuevo_precio
    WHERE codigo = codigo_prod;
END //
```

-- 4. Eliminar producto por nombre

```
CREATE PROCEDURE eliminar_producto_por_nombre(IN nombre_prod VARCHAR(100))
BEGIN
    DELETE FROM producto WHERE nombre = nombre_prod;
END //
```

-- 5. Obtener productos por nombre de fabricante

```
CREATE PROCEDURE obtener_productos_fabricante(IN nombre_fab VARCHAR(100))
BEGIN
    SELECT p.*
    FROM producto p
    INNER JOIN fabricante f ON p.codigo_fabricante = f.codigo
    WHERE f.nombre = nombre_fab;
END //
```

-- 6. Actualizar nombre de fabricante por código

```
CREATE PROCEDURE actualizar_nombre_fabricante(
    IN codigo_fab INT,
```

```
    IN nuevo_nombre VARCHAR(100)
)
BEGIN
    UPDATE fabricante
    SET nombre = nuevo_nombre
    WHERE codigo = codigo_fab;
END //
```

```
-- 7. Obtener precio promedio de productos por código de fabricante
CREATE PROCEDURE precio_promedio_fabricante(IN codigo_fab INT, OUT promedio
DECIMAL(10,2))
BEGIN
    SELECT AVG(precio) INTO promedio
    FROM producto
    WHERE codigo_fabricante = codigo_fab;
END //
```

```
-- 8. Insertar producto si no existe otro con el mismo nombre
CREATE PROCEDURE insertar_producto_unico(
    IN nombre_prod VARCHAR(100),
    IN precio DECIMAL(10,2),
    IN codigo_fab INT
)
BEGIN
    IF NOT EXISTS (SELECT 1 FROM producto WHERE nombre = nombre_prod) THEN
        INSERT INTO producto (nombre, precio, codigo_fabricante)
        VALUES (nombre_prod, precio, codigo_fab);
    END IF;
END //
```

```
-- 9. Obtener fabricantes con productos superiores a un precio
CREATE PROCEDURE fabricantes_productos_caros(IN precio_min DECIMAL(10,2))
BEGIN
    SELECT DISTINCT f.nombre
    FROM fabricante f
    INNER JOIN producto p ON f.codigo = p.codigo_fabricante
    WHERE p.precio > precio_min;
END //
```

```
-- 10. Eliminar todos los productos de un fabricante
CREATE PROCEDURE eliminar_productos_fabricante(IN nombre_fab VARCHAR(100))
BEGIN
    DELETE p FROM producto p
    INNER JOIN fabricante f ON p.codigo_fabricante = f.codigo
    WHERE f.nombre = nombre_fab;
END //
```

```
-- 11. Obtener producto más caro de un fabricante
```

```

CREATE PROCEDURE producto_mas_caro_fabricante(IN nombre_fab VARCHAR(100))
BEGIN
    SELECT p.nombre, p.precio
    FROM producto p
    INNER JOIN fabricante f ON p.codigo_fabricante = f.codigo
    WHERE f.nombre = nombre_fab
    ORDER BY p.precio DESC
    LIMIT 1;
END //

```

```

-- 12. Obtener total productos y precio promedio por fabricante
CREATE PROCEDURE estadisticas_fabricante(IN nombre_fab VARCHAR(100))
BEGIN
    SELECT COUNT(*) as total_productos, AVG(precio) as precio_promedio
    FROM producto p
    INNER JOIN fabricante f ON p.codigo_fabricante = f.codigo
    WHERE f.nombre = nombre_fab;
END //

```

```

-- 13. Verificar si fabricante tiene productos sobre un precio
CREATE PROCEDURE verificar_productos_caros(
    IN nombre_fab VARCHAR(100),
    IN precio_limite DECIMAL(10,2),
    OUT tiene_productos_caros BOOLEAN
)
BEGIN
    SELECT EXISTS(
        SELECT 1
        FROM producto p
        INNER JOIN fabricante f ON p.codigo_fabricante = f.codigo
        WHERE f.nombre = nombre_fab AND p.precio > precio_limite
    ) INTO tiene_productos_caros;
END //

```

```

-- 14. Obtener fabricante con más productos
CREATE PROCEDURE fabricante_mas_productos()
BEGIN
    SELECT f.nombre, COUNT(*) as cantidad_productos
    FROM fabricante f
    INNER JOIN producto p ON f.codigo = p.codigo_fabricante
    GROUP BY f.codigo, f.nombre
    ORDER BY cantidad_productos DESC
    LIMIT 1;
END //

```

```

-- 15. Obtener producto más barato y calcular descuento
CREATE PROCEDURE producto_mas_barato_con_descuento(
    IN nombre_fab VARCHAR(100),

```

```
    IN precio_objetivo DECIMAL(10,2)
)
BEGIN
  SELECT
    p.nombre,
    p.precio,
    ((p.precio - precio_objetivo) / p.precio) * 100 as porcentaje_descuento
  FROM producto p
  INNER JOIN fabricante f ON p.codigo_fabricante = f.codigo
  WHERE f.nombre = nombre_fab
  ORDER BY p.precio ASC
  LIMIT 1;
END //

DELIMITER ;
```