# Práctica de Funciones

1. Escribir una sentencia SELECT que devuelva el número de orden, fecha de orden y el nombre del día de la semana de la orden de todas las órdenes que no han sido pagadas.

   Si el cliente pertenece al estado de California el día de la semana debe devolverse en inglés, caso contrario en español. Cree una función para resolver este tema.

   Nota: `SET @DIA = datepart(weekday,@fecha)`
   Devuelve en la variable @DIA el nro. de día de la semana , comenzando con 1 Domingo hasta 7 Sábado.

**1a. Resolución con UNION**

```
SELECT order_num, order_date, dbo.fx_dia_semana(order_date,'espaniol')
FROM orders o, customer c
WHERE o.customer_num = c.customer_num
AND paid_date IS NULL
AND state != 'CA'
UNION ALL
SELECT order_num, order_date, dbo.fx_dia_semana(order_date,'ingles')
FROM orders o, customer c
WHERE o.customer_num = c.customer_num AND
paid_date IS NULL
```

**1b. Resolución con CASE en SELECT**

```
SELECT order_num, order_date,
 CASE
  WHEN state =  'CA' THEN dbo.fx_dia_semana(order_date,'ingles')
  WHEN state !=  'CA' OR state IS NULL THEN dbo.fx_dia_semana(order_date,'espaniol')
 END
FROM orders o, customer c
WHERE o.customer_num = c.customer_num
AND paid_date IS NULL
```

**1c. Resolución con CASE en FUNCIÓN**

```
SELECT order_num, order_date,
dbo.fx_dia_semana(order_date, CASE c.state
                                    WHEN 'CA' THEN 'ingles'
                                    ELSE 'espaniol'
                                  END)
FROM orders o, customer c
WHERE o.customer_num = c.customer_num
AND paid_date IS NULL
```

```
CREATE FUNCTION Fx_DIA_SEMANA
(@FECHA  DATETIME,
@IDIOMA VARCHAR (20))
RETURNS VARCHAR (20)
AS BEGIN
DECLARE @DIA INT
DECLARE @RETORNO VARCHAR(20)

SET @DIA = datepart(weekday,@fecha)

IF @IDIOMA = 'espaniol'
BEGIN
    SET @RETORNO = case when @dia = 1 then 'Domingo'
    when @dia = 2 then 'lunes'                when @dia = 3
```

```
then 'Martes'                    when @dia = 4 then
'Miercoles'               when @dia = 5 then 'Jueves'
                   when @dia = 6 then 'Viernes'
                   else 'Sábado'
     end END
ELSE
BEGIN
       SET @RETORNO = case when @dia = 1 then 'Sunday'
    when @dia = 2 then 'Monday'               when @dia = 3
then 'Tuesday'                   when @dia = 4 then 'Wednesday'
           when @dia = 5 then 'Thursday'
when @dia = 6 then 'Friday'
                   else 'Saturday' end
END

RETURN @RETORNO
END
```

2. Escribir una sentencia SELECT para los clientes que han tenido órdenes en al menos 2 meses diferentes, los dos meses con las órdenes con el mayor ***ship_charge***.

   Se debe devolver una fila por cada cliente que cumpla esa condición, el formato es:

```
Cliente     Año y mes mayor carga         Segundo año y mes mayor carga

 NNNN      YYYY - Total: NNNN.NN           YYYY - Total: NNNN.NN
```

La primera columna es el id de cliente y las siguientes 2 se refieren a los campos ship_date y ship_charge.

Se requiere crear una función que devuelva la información de 1er o 2do año mes con la orden con mayor Carga (ship_charge).

```
SELECT distinct customer_num, dbo.fx_datosporMes(1, customer_num),
                dbo.fx_datosporMes(2, customer_num)
FROM orders o
WHERE EXISTS (SELECT 1
              FROM orders o2
              WHERE o2.customer_num = o.customer_num
              AND month(o.order_date) > month(o2.order_date))
DROP FUNCTION fx_datosporMes

CREATE FUNCTION dbo.fx_datosporMes
(@ORDEN SMALLINT, @CLIENTE  INT)
RETURNS VARCHAR(100)
AS
BEGIN
  DECLARE @MES     VARCHAR(4)
  DECLARE @CARGA    VARCHAR(50)
  DECLARE @RETORNO VARCHAR(100)

  IF @ORDEN = 1
    BEGIN
       SELECT TOP 1 @MES = MONTH(order_date),
                    @CARGA = MAX(ship_charge)
         FROM orders
        WHERE customer_num = @CLIENTE
       GROUP BY MONTH(order_date)
       ORDER BY 2 DESC

       SET @RETORNO = @MES + ' - Total: ' + @CARGA
    END
```

```
 ELSE
   BEGIN
     SELECT TOP 1 @MES = order_date,
                  @CARGA = COALESCE(ship_charge,0)
         FROM
        (SELECT TOP 2 MONTH(order_date) as order_date, MAX(ship_charge) as ship_charge
          FROM orders
           WHERE customer_num = @CLIENTE
           GROUP BY MONTH(order_date)
           ORDER BY 2 DESC) as SQL1
           ORDER BY 2 ASC

           SET @RETORNO = @MES + ' - Total: ' + @CARGA
    END

   RETURN @RETORNO
END
```

```
-------------------
Solución con 2 funciones
```

```
SELECT customer_num AS Cliente, dbo.fx_1ermes(customer_num) AS "Mes mayor carga",
dbo.fx_2domes(customer_num) AS "Segundo Mes mayor carga"
FROM  orders WHERE customer_num IN
(SELECT DISTINCT customer_num
FROM orders o1
WHERE EXISTS (SELECT 1 FROM orders o2
WHERE o1.customer_num = o2.customer_num
AND MONTH(o1.order_date) > MONTH(o2.order_date)))
GROUP BY customer_num

DROP FUNCTION Fx_1erMes

CREATE FUNCTION Fx_1erMes
(@CLIENTE  INT)
RETURNS VARCHAR (100)
AS BEGIN
DECLARE @MES     VARCHAR(2)
DECLARE @CARGA    VARCHAR(50)
DECLARE @RETORNO VARCHAR(100)

SELECT TOP 1 @MES = MONTH(order_date), @CARGA = MAX(COALESCE(ship_charge,0))
FROM orders
WHERE customer_num = @CLIENTE
GROUP BY MONTH(order_date)
ORDER BY 2 DESC

SET @RETORNO = @MES + ' - Total: ' + @CARGA


RETURN @RETORNO
END
GO

DROP FUNCTION Fx_2doMes
CREATE FUNCTION Fx_2doMes
(@CLIENTE  INT)
RETURNS VARCHAR (100)
AS BEGIN
DECLARE @MES     VARCHAR(4)
```

```sql
DECLARE @CARGA   VARCHAR(50)
DECLARE @RETORNO VARCHAR(100)

SELECT TOP 1 @MES = order_date, @CARGA = COALESCE(ship_charge,0) FROM (SELECT TOP 2
MONTH(order_date) as order_date, MAX(COALESCE(ship_charge,0)) as ship_charge FROM
orders
WHERE customer_num = @CLIENTE
GROUP BY MONTH(order_date)
ORDER BY 2 DESC) as SQL1
ORDER BY 2 ASC

SET @RETORNO = @MES + ' - Total: ' + @CARGA

RETURN @RETORNO
END
```

3. Escribir un Select que devuelva para cada producto de la tabla ***Products*** que exista en la tabla ***Catalog*** todos sus fabricantes separados entre sí por el caracter pipe (|). Utilizar una función para resolver parte de la consulta. Ejemplo de la salida

| Stock_num | Fabricantes |
|---|---|
| 5 | NRG \| SMT \| ANZ |

```sql
SELECT DISTINCT stock_num, dbo.fx_fabricantes(stock_num) as Fabricantes
  FROM products p
 WHERE EXISTS (SELECT 1 FROM catalog c WHERE c.stock_num = p.stock_num);

DROP FUNCTION Fx_fabricantes
CREATE FUNCTION Fx_FABRICANTES (@CODIGO  INT) RETURNS VARCHAR (100) AS
BEGIN

    DECLARE @RETORNO VARCHAR(100)
    DECLARE @FABRICANTE VARCHAR(3)

    DECLARE CUR_FABRICANTES CURSOR FOR  SELECT manu_code
                                          FROM catalog
                                         WHERE stock_num = @CODIGO;
    SET @RETORNO = ''
    OPEN CUR_FABRICANTES
    FETCH NEXT FROM CUR_FABRICANTES INTO @FABRICANTE
    WHILE (@@FETCH_STATUS = 0)
    BEGIN
         SET @RETORNO = @RETORNO + @FABRICANTE  + ' | '
         FETCH NEXT FROM CUR_FABRICANTES INTO @FABRICANTE
    END
    CLOSE CUR_FABRICANTES
    DEALLOCATE CUR_FABRICANTES
    SET @RETORNO = SUBSTRING(@RETORNO, 1, LEN(@RETORNO) - 2)
    RETURN @RETORNO
END
```