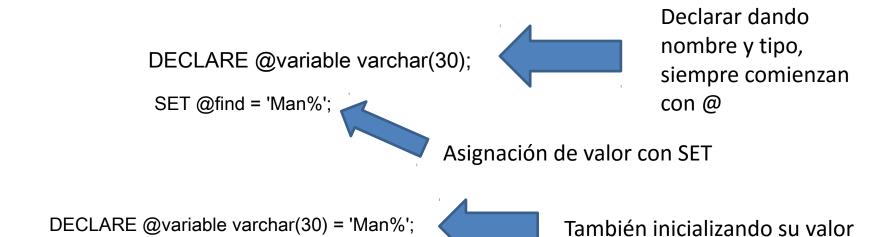
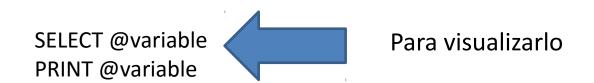
Variables



Se pueden usar en las cláusulas SELECT



```
--procedimiento almacenado con IF
if object id('ver texto') is not null
  drop procedure ver texto;
 go
    create procedure ver texto
           @texto varchar(20)
    as
    IF @texto like 'A%'
        BEGIN
            PRINT @texto + ' comienza con A'
        END
    ELSE
        BEGIN
            PRINT @texto + ' no comienza con A'
        END
    GO
exec ver texto 'hola'
exec ver texto 'antes'
```

```
--procedimiento detectar pares
if object id('detecta pares') is not null
  drop procedure detecta pares;
  go
    create procedure detecta pares
           @numero int
    as
    ΙF
       (@numero % 2)=0
        BEGIN
            PRINT cast (@numero AS varchar) + ' es par'
        END
    ELSE
        BEGIN
            PRINT cast (@numero AS varchar) + ' es impar'
        END
    GO
exec detecta pares 44
exec detecta pares 33
exec detecta pares -1
```

```
-- procedimiento almacenado IF con subconsulta
if object id('libros') is not null
  drop table libros;
qo
 create table libros(
 titulo varchar(40),
  autor varchar(30),
  editorial varchar(15),
 precio float,
  cantidad integer
) ;
go
 insert into libros (titulo, autor, editorial, precio, cantidad)
  values ('El aleph', 'Borges', 'Emece', 25.50, 100);
 insert into libros (titulo, autor, editorial, precio, cantidad)
  values ('Alicia en el pais de las maravillas', 'Lewis
Carroll', 'Atlantida', 10, 200);
 insert into libros (titulo, autor, editorial, precio, cantidad)
  values ('Matematica estas ahi', 'Paenza', 'Siglo XXI', 18.8, 200);
go
```

```
if object id('actualizar autor') is not null
 drop procedure actualizar autor;
 do
create procedure actualizar autor
       @autoranterior varchar(30), @autornuevo varchar(30)
as
ΤF
EXISTS (SELECT titulo, autor, editorial, precio, cantidad FROM libros
          WHERE autor = @autoranterior)
 BEGIN
     update libros
     set autor = @autornuevo
     where autor=@autoranterior
 END
else
 print 'El autor indicado no está'
do
SELECT titulo, autor, editorial, precio, cantidad FROM libros
exec actualizar autor 'Borges', 'Jorge Luis Borges'
                                                                   Aquí lo
SELECT titulo, autor, editorial, precio, cantidad FROM libros
                                                                   ejecutamos
exec actualizar autor 'Cervantes', 'Miguel de Cervantes'
```

While

```
--ejemplo de while
if object id('proc contador') is not null
  drop procedure proc contador;
 go
create procedure proc contador
as
    DECLARE @contador int
    SET @contador = 1
    WHILE (@contador <= 20)
    BEGIN
       PRINT 'Iteracion del bucle ' + cast (@contador AS varchar)
       SET @contador = @contador + 1
    END
    print 'Final'
go
exec proc contador
```

set nocount y @@rowcount

go

```
-- uso de set nocount y @@rowcount
                                             colocando off.
if object id('libros') is not null
  drop table libros;
                                             @@rowcount contiene el nº de
qo
                                             registros afectados por la
create table libros(
  titulo varchar(40),
                                             sentencia anterior.
  autor varchar(30),
  editorial varchar(15),
 precio float,
  cantidad integer
 ) ;
 insert into libros (titulo, autor, editorial, precio, cantidad)
 values ('El aleph', 'Borges', 'Emece', 25.50, 100);
 insert into libros (titulo, autor, editorial, precio, cantidad)
  values ('Alicia en el pais de las maravillas', 'Lewis
Carroll', 'Atlantida', 10, 200);
 insert into libros (titulo, autor, editorial, precio, cantidad)
  values ('Matematica estas ahi', 'Paenza', 'Siglo XXI', 18.8, 200);
```

Set nocount on no muestra el nº

de filas afectadas, al contrario que

```
if object id('ver autores') is not null
  drop procedure ver autores;
  go
create procedure ver autores
  as
  set nocount on
  select distinct autor from libros
  go
exec ver autores
if object id('ver libros') is not null
  drop procedure ver libros;
  go
create procedure ver libros
    @autor varchar(30)
  as
  set nocount on --off
  select titulo from libros
  where autor like @autor
 print cast(@@rowcount as varchar) + ' registros'
  go
exec ver libros 'Cervantes'
exec ver libros 'Borges%'
```

Return

La instrucción "return" sale de una consulta o procedimiento y todas las instrucciones posteriores no son ejecutadas.

```
create procedure pa_libros_autor
  @autor varchar(30) = null
as
  if @autor is null
begin
  select 'Debe indicar un autor'
  return
end;
select titulo from libros where autor = @autor;
go
```

Case

select nombre, nota, resultado= case nota when 0 then 'libre' when 1 then 'libre' when 2 then 'libre' when 3 then 'libre' when 4 then 'regular' when 5 then 'regular' when 6 then 'regular' when 7 then 'promocionado' when 8 then 'promocionado' when 9 then 'promocionado' when 10 then 'promocionado' end from alumnos;

La función "case" compara 2 o más valores y devuelve un resultado.

La sintaxis es la siguiente:

case VALORACOMPARAR
when VALOR1 then RESULTADO1
when VALOR2 then RESULTADO2
...
else RESULTADO3
end

select nombre, nota, condicion=
case
when nota<4 then 'libre'
when nota >=4 and nota<7 then 'regular'
when nota>=7 then 'promocionado'
else 'sin nota'
end
from alumnos;

```
use empresasinformaticas
go
select clave, descripcion, precio
from componente;
select clave, descripcion, precio,
case
   when precio<10 then 'Barato'
   when precio between 10 and 50 then 'Medio'
   else 'Caro'
end as nivelprecio
from componente;
```

```
select count(*),
case
   when precio<10 then 'Barato'
   when precio between 10 and 50 then 'Medio'
   else 'Caro'
end as nivelprecio
from componente
group by
case
   when precio<10 then 'Barato'
   when precio between 10 and 50 then 'Medio'
   else 'Caro'
end
```

Concatenar cadenas de caracteres

Con las tablas de ejemplos anteriores...

```
if object_id('concatenar_autores') is not null
  drop procedure concatenar_autores;
  go
create procedure concatenar_autores
  as
    declare @delimitador varchar, @autores varchar(max)
    set @delimitador = ','
    set @autores = ''
    select @autores=@autores +@delimitador+autor
    from libros
    select @autores as autores
  go
exec concatenar_autores
select autor from libros
```

Cargar datos en Tablas temporales

Con el procedimiento anterior

```
-- ver resultado de procedimiento en tabla temporal
create table #t1
(
autores varchar(100)
)
insert #t1 exec concatenar_autores
select autores from #t1
```

Control de errores

```
--try catch
if object id('libros') is not null
  drop table libros;
create table libros(
 idlibro int primary key, titulo varchar(40), autor varchar(30),
  editorial varchar(15), precio float, cantidad integer
) ;
go
if object id('insertar libros') is not null
  drop procedure insertar libros;
  do
create procedure insertar libros
   @idlibro integer, @idtitulo varchar(40), @autor varchar(30),
   @editorial varchar(15), @precio float, @cantidad integer
  as
begin try
set nocount on
insert into libros (idlibro, titulo, autor, editorial, precio, cantidad)
                @idlibro, @idtitulo,
                                        @autor,
     @editorial, @precio, @cantidad)
select 'Registro insertado correctamente'
end try
begin catch
     SELECT ERROR NUMBER() AS 'Nº de error', ERROR MESSAGE() as 'Mensaje de error'
end catch;
go
exec insertar libros 2, 'Matematica estas ahi', 'Paenza', 'Siglo XXI', 18.8, 200
exec insertar libros 1, 'Alicia en el pais de las maravillas', 'Lewis Carroll', 'Atlantida', 10, 200
exec insertar libros 2, 'El aleph', 'Borges', 'Emece', 25.50, 100
```

-- hacer select * de tabla especificada if object_id('proc_vertabla') is not null drop procedure proc_vertabla; go

Sp_executesql

Procedimiento para ejecutar una cadena de caracteres que contiene una sentencia SQL

```
create procedure proc vertabla
   @tabla nvarchar(100)
as
 set nocount on
 if object id(@tabla) is null
  begin
  print N'La tabla ' + @tabla + N' no existe'
  end
else
  begin
   declare @sentencia nyarchar(1000)
   set @sentencia=N'select * from ' + @tabla
   exec sp executesql @sentencia
   end
go
exec proc vertabla 'pruebalibro1'
go
exec proc vertabla 'pruebalibro2'
go
exec proc vertabla 'pruebalibro3'
go
```

```
if object id('proc vertabla2') is not null
 drop procedure proc vertabla2;
 go
 create procedure proc vertabla2
    @tabla varchar(100)
 as
  set nocount on
  if object id(@tabla) is null
   begin
   print 'La tabla ' + @tabla + ' no existe'
   end
 else
   begin
    declare @sentencia varchar(1000)
    set @sentencia='select * from ' + @tabla
   exec (@sentencia)
   end
 qo
 exec proc vertabla2 'pruebalibro1'
 qo
 exec proc vertabla2 'pruebalibro2'
 go
```

exec proc vertabla2 'pruebalibro3'

go

Igual sin nvarchar mediante exec(@sentencia)