USE master;

GO

CREATE DATABASE sueldos;

GO

CREATE TABLE Categorias2 (

Cod char(2) not null PRIMARY KEY CLUSTERED,

Descripción varchar(50) NOT null,

);

GO

CREATE TABLE Empleados (

Id int not null IDENTITY,

Nombre varchar(20) not null,

Apellido varchar(20) not null,

Categoria char(2) not null,

Sueldo int not null,

Descripción varchar(50) not null DEFAULT 'EMPLEADO NUEVO',

FOREIGN KEY (Categoria) REFERENCES CATEGORIAS(Cod),

UNIQUE NONCLUSTERED (Nombre, Apellido),

check(Sueldo>1500),

);

GO

insert into Empleados(Nombre, Apellido, Categoria, Sueldo) values ('Maria', 'Kapek', 'AA', 45000);

GO

insert into Empleados(Nombre, Apellido, Categoria, Sueldo) values ('Mar', 'Kapek', 'AA', 40);

GO

Los checks solo chequean en campos de la tabla en la que son creados.

TRIGGER

CREATE TRIGGER Trigger\_Categorias

ON Categorias

AFTER INSERT

AS

if((select Cod from inserted)='CD')

BEGIN

RAISERROR('no se pueden insertar CDs', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END;

insert into Categorias values('CD', 'CDs');

PROCEDIMIENTO

CREATE PROCEDURE procAgregarCategoria

@pCod char(2),

@pDesc varchar(50)

AS

insert into Categorias(Cod, Descripción) values(@pCod, @pDesc);

select \* from Categorias;

GO

exec procAgregarCategoria 'CD', 'paltrigger';

FUNCION ESCALAR

CREATE FUNCTION feDarCantidadCategorias() RETURNS INT

AS

BEGIN

RETURN (select COUNT(\*) from Categorias);

END

Go

select feDarCantidadCategorias();

FUNCION CON VALORES DE TABLA

CREATE FUNCTION [dbo].[ftDarCategorias]()

RETURNS @returntable TABLE

(

Codigo char(2),

Descripcion varchar(50)

)

AS

BEGIN

INSERT @returntable

SELECT cod, Descripción from Categorias

RETURN

END

select \* from ftDarCategorias();