Diagrama

Descripción generada automáticamente

[Título del documento]

[Subtítulo del documento]



[Fecha]

[Nombre de la compañía]

[Dirección de la compañía]

Índice

[Ejercicio 1 2](#_Toc149941097)

[Función 2](#_Toc149941098)

[Ejecución 3](#_Toc149941099)

[Ejercicio 2 3](#_Toc149941100)

[Función 3](#_Toc149941101)

[Ejecución 4](#_Toc149941103)

[Ejercicio 3 5](#_Toc149941104)

[Función 5](#_Toc149941105)

[Ejecución 5](#_Toc149941106)

[Ejercicio 4 6](#_Toc149941107)

[Función 6](#_Toc149941108)

[Ejecución 7](#_Toc149941109)

[Ejercicio 5 7](#_Toc149941112)

[Función 7](#_Toc149941114)

[Ejecución 8](#_Toc149941116)

[Ejercicio 6 8](#_Toc149941117)

[Función 9](#_Toc149941119)

[Ejecución 10](#_Toc149941120)

[Ejercicio 7 11](#_Toc149941123)

[Función 11](#_Toc149941124)

[Ejecución 12](#_Toc149941126)

[Ejercicio 8 12](#_Toc149941127)

[Función 12](#_Toc149941128)

[Ejecución 13](#_Toc149941131)

[Caso 1 13](#_Toc149941132)

[Caso 2 13](#_Toc149941133)

[Ejercicio 9 14](#_Toc149941136)

[Función 14](#_Toc149941137)

[Ejecución 15](#_Toc149941138)

[Caso 1 15](#_Toc149941139)

[Caso 2 15](#_Toc149941140)

# Ejercicio 1

Crear una función que reciba como parámetro la fecha de nacimiento y devuelva la edad al día de hoy.

## Función

CREATE FUNCTION CalcularEdad2(@nacimiento DATE)

RETURNS INT

AS

BEGIN

DECLARE @edad INT;

SET @edad = DATEDIFF(YEAR, @nacimiento, GETDATE());

--Ajustar la edad si aún no ha pasado el cumpleaños de este año

IF (MONTH(@nacimiento) > MONTH(GETDATE()) OR (MONTH(@nacimiento) = MONTH(GETDATE()) AND DAY(@nacimiento) > DAY(GETDATE())))

SET @edad = @edad -1;

RETURN @edad;

END

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

## Ejecución

SELECT dbo.CalcularEdad2(nacimiento) AS Edad

FROM Persona;

Interfaz de usuario gráfica, Texto, Aplicación, Word

Descripción generada automáticamente

# Ejercicio 2

Crear una función que reciba la matrícula de un avión y devuelva la siguiente fecha de mantenimiento.

## Función

## CREATE FUNCTION ObtenerSiguienteFechaMantenimiento2(@matricula VARCHAR(50))

RETURNS DATE

AS

BEGIN

DECLARE @siguienteFechaMantenimiento DATE;

--Obtener la última fecha de mantenimiento del avión

SELECT @siguienteFechaMantenimiento = UltimoMantenimiento

FROM Avion

WHERE matricula = @matricula;

--Calcular la siguiente fecha de mantenimiento sumando los meses de mantenimiento al último mantenimiento

SET @siguienteFechaMantenimiento = DATEADD(MONTH, (SELECT MesesMantenimiento FROM Avion WHERE matricula = @matricula), @siguienteFechaMantenimiento);

RETURN @siguienteFechaMantenimiento;

END

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

## Ejecución

SELECT dbo.ObtenerSiguienteFechaMantenimiento2(matricula) AS SiguienteFechaMantenimiento

FROM Avion;

Interfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamente

# Ejercicio 3

Crear una función que reciba una fecha y muestre la siguiente información del vuelo más cercano: matrícula del avión, nombre del vendedor, nombre del cliente, fecha de vuelo y precio del asiento.

# Función

CREATE FUNCTION ObtenerInformacionVueloMasCercano(@fecha DATE)

RETURNS TABLE

AS

RETURN

(

SELECT TOP 1 Avion.matricula AS MatriculaAvion, Cliente.clienteId AS NombreCliente, Vendedor.vendedorId AS NombreVendedor, Vuelo.fechaVuelo, Vuelo.precioPorAsiento

FROM Vuelo

INNER JOIN Avion ON Vuelo.matriculaAvion = Avion.matricula

INNER JOIN Venta ON Vuelo.idVuelo = Venta.idVuelo

INNER JOIN Vendedor ON Venta.idVendedor = Vendedor.vendedorId

INNER JOIN Cliente ON Venta.idCliente = Cliente.clienteId

WHERE Vuelo.fechaVuelo >= @fecha

ORDER BY Vuelo.fechaVuelo ASC

)

## Ejecución

SELECT \* FROM ObtenerInformacionVueloMasCercano('2014-10-01');

Interfaz de usuario gráfica, Texto, Aplicación, Word

Descripción generada automáticamente

# Ejercicio 4

Crear una función que reciba el id de un cliente y el id de un vuelo y regrese el precio del vuelo considerando el descuento por tipo de cliente.

## Función

CREATE FUNCTION ObtenerPrecioConDescuento(@idCliente INT, @idVuelo INT)

RETURNS DECIMAL(10,2)

AS

BEGIN

DECLARE @precio DECIMAL(10,2);

DECLARE @descuento DECIMAL(5,2);

--obtener el descuento del tipo de cliente

SELECT @descuento = t.descuento

FROM TipoCliente t

INNER JOIN Cliente c ON t.idTipoCliente = c.idTipoCliente

WHERE c.clienteId = @idCliente;

--obtener el precio del vuelo

SELECT @precio = PrecioPorAsiento - (PrecioPorAsiento \* @descuento / 100)

FROM Vuelo

WHERE idVuelo = @idVuelo

RETURN @precio;

END

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

## Ejecución

DECLARE @clienteId INT = 1;

DECLARE @vueloId INT = 100;

## SELECT dbo.ObtenerPrecioConDescuento(15,20) AS PrecioConDescuento;

## Interfaz de usuario gráfica, Texto, Aplicación, Word Descripción generada automáticamente

# Ejercicio 5

# 

Crear una función que regrese el RFC de todas las personas de la base de datos o se especifique algún IdPersona para solo obtener el de una.

## Función

CREATE FUNCTION ObtenerNombrePersonas(@idPersona INT = NULL)

RETURNS TABLE

AS

RETURN

(

SELECT nombre

FROM Persona

WHERE @idPersona IS NULL OR IdPersona = @idPersona

)

# Interfaz de usuario gráfica, Aplicación Descripción generada automáticamente

# Ejecución

DECLARE @idPersona INT

SELECT nombre

FROM dbo.ObtenerNombresPErsonas(@idPersona);

# Ejercicio 6

Crear una función que reciba como parámetros el Id de un cliente, y una fecha inicio y fin. Como salida debe indicar cuantas veces ha viajado.

## Función

CREATE FUNCTION ContarViajesCliente(

@idCliente INT,

@fechaInicio DATE,

@fechaFin DATE

)

RETURNS INT

AS

BEGIN

DECLARE @cantidadViajes INT;

SELECT @cantidadViajes = COUNT(\*)

FROM Venta

INNER JOIN Vuelo ON Venta.idVuelo = Vuelo.idVuelo

WHERE Venta.idCliente = @idCliente

AND Vuelo.fechaVuelo >= @fechaInicio

AND Vuelo.fechaVuelo <= @fechaFin;

RETURN @cantidadViajes;

END

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

## Ejecución

## SELECT dbo.ContarViajesCliente(15, '2014-09-11', '2014-09-11') AS CantidadViaje;

## Interfaz de usuario gráfica, Aplicación, Word Descripción generada automáticamente

## Ejercicio 7

Crear una función que reciba como parámetro un sueldo base y que muestre el nombre completo de los empleados que tengan un salario mayor o igual al parámetro de entrada de mostrar la cantidad de años que lleva el empleado desde su fecha de ingreso.

## Función

CREATE FUNCTION EmpleadosConSalarioSuperior(@sueldoBase DECIMAL(10,2))

RETURNS TABLE

AS

RETURN

(

SELECT CONCAT(Persona.nombre, ' ', Persona.paterno, ' ', Persona.materno) AS NombreCompleto,

DATEDIFF(YEAR, Empleado.fechaIngreso, GETDATE()) AS AniosTrabajados

FROM Empleado

JOIN Persona ON Empleado.idPersona = Persona.IdPersona

WHERE Empleado.sueldo >= @sueldoBase

## )

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

## Ejecución

SELECT \*

FROM EmpleadosConSalarioSuperior(5000)

Interfaz de usuario gráfica, Texto, Word

Descripción generada automáticamente con confianza media

# Ejercicio 8

Modificar la funciòn ObtenerDireccionEmpleado, en caso de que el Id proporcionado no tenga jefe, mostrar en la columna direccion completa la siguiente leyenda "información confidencial.

## Función

CREATE FUNCTION ObtenerDireccionEmpleado(@IdEmpleado INT)

RETURNS @tbEmpleadoDireccion TABLE

(

IdEmpleado INT,

nombreCompleto VARCHAR(100),

direccioncompleta varchar(100)

)

AS

BEGIN

INSERT INTO @tbEmpleadoDireccion

SELECT a.idEmpleado,

nombre + ' ' + paterno + ' ' + ISNULL(materno, ''),

CASE

WHEN a.idJefe IS NULL THEN 'Información confidencial'

ELSE calle + ' ' + numero + ' ' + colonia + ' ' + codigoPostal

END

FROM Empleado a

INNER JOIN persona b ON a . idPersona = b.IdPersona

LEFT JOIN Direccion c oN b.direccionId = c.direccionId

WHERE a.idEmpleado = @IdEmpleado;

RETURN;

## END;

## Interfaz de usuario gráfica, Texto, Aplicación Descripción generada automáticamente

## Ejecución

SELECT \*

FROM ObtenerDireccionEmpleado(10)

### Caso 1

Interfaz de usuario gráfica, Texto, Aplicación, Word

Descripción generada automáticamente

### Caso 2

SELECT \*

FROM ObtenerDireccionEmpleado(11)

# **Interfaz de usuario gráfica, Texto, Aplicación Descripción generada automáticamente**

# 

# Ejercicio 9

Crear una función que devuelva el precio por asiento de todos los vuelos y que reciba un parámetro opcional que indique si es temporada alta, en su caso aumentar 20% el precio por asiento.

## Función

CREATE FUNCTION ObtenerPrecioAsiento(@TemporadaAlta BIT = 0)

RETURNS TABLE

AS

RETURN

(

SELECT Vuelo.idVuelo,

Vuelo.precioPorAsiento \* (1+(CASE WHEN @TemporadaAlta = 1 THEN 0.2 ELSE 0 END)) AS PrecioAsiento

FROM Vuelo

)

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

## Ejecución

### Caso 1

SELECT \*

FROM ObtenerPrecioAsiento(1)

Interfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamente

### Caso 2

SELECT \*

FROM ObtenerPrecioAsiento(0)

Interfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamente