Facilitador(a): Ing. Carmen Ortega Asignatura: Base de Datos II

Fecha: 20/10/2020 Grupo: \_\_\_1IL131

Estudiantes: Jacky He, 3-740-114

Yui Lo, 8-929-854

Maria José Martínez PE-15-1146

Dhiraj Lakhwani 3-743-1798

Richie Ieong 8-970-668

1. **TÍTULO** **DE LA EXPERIENCIA**: Crear procedimientos y funciones invocados MySQL
2. **TEMAS:**

En este ejercicio se utilizarán todas las instrucciones para crear procedimientos y funciones invocados por SQL

1. **OBJETIVO(S):**

Aplicar conceptos para crear procedimientos y funciones invocados por SQL y aplicarlo en un ejemplo práctico.

La actividad le permite al estudiante enriquecer su experiencia en el desarrollo y construcción de conocimientos del curso de Base de Datos II.

1. **METODOLOGÍA:**
2. Divida el salón en grupos.
3. Distribuya el taller a cada grupo.
4. Cada equipo contará con un período de laboratorio (2 períodos de 45 min) para dar respuesta a las preguntas.
5. Primero realice el taller de forma individual y una vez desarrollado haga una reunión video llamada en equipo para que colaborativamente den respuesta en equipo las preguntas del ejercicio.
6. Se deben recibir en sección tareas de la plataforma de clases virtuales la respectiva asignación resuelta o realizar el test online del taller.
7. **PROCEDIMIENTO O ENUNCIADO DE LA EXPERIENCIA:**

**CASO 1. CREAR PROCEDIMIENTO – INTRODUCE PERSONA**

* Abrir una consola de MySQL seleccionamos una base de datos y empezamos a

escribir:

* + Creamos dos tablas en una almacenamos las personas mayores de 18 años y en

otra, las personas menores.

* + - Create table niños (edad int, nombre varchar(50));
    - Create table adultos (edad int, nombre varchar(50));
  + Ahora, introducimos personas en las tablas, pero dependiendo de la edad queremos que se introduzcan en una tabla u otra.
  + Llamar a nuestro procedimiento almacenado usamos la sentencia call: call introducePersona(25, ”JoseManuel”);
  + De ahora en adelante, usamos el procedimiento para introducir personas en nuestra BD de manera que los scritps PHP que lo usan para introducir las personas a la tablas.

**Caso 2**

Usando la base de datos Northwind, necesito realizar un procedimiento almacenado que reciba como parámetro el año. Calcular la cantidad de órdenes y recaudaciones hechas por un empleado para un determinado año. Aplicar una comisión del 25 % en caso que la cantidad de órdenes supere el promedio de ese año. Promedio=(Cantidad de órdenes / Número de Empleados) de lo contrario solo aplicar el 10%

La base de datos Northwind, era una base de datos de una empresa ficticia de ventas, que Microsoft entregaba para realizar pruebas.

¿Cómo instalar la base de datos Northwind?

Primero, descargar y descomprimir el script de creación de tablas y datos, puedes usar este [link](https://drive.google.com/file/d/1W0cymSv76DFP_uCJgjwcnIrWIAWtLvZA/view). En segundo lugar, debes tener acceso a una instancia de SQL Server. Puedes descargar una versión de SQL Server Express en el sitio de Microsoft. Atención que algunos instaladores, exigen tener la Configuración Regional en Español/España. Puedes cambiar temporalmente este parámetro en el Panel de Control para instalar SQL Server Express.

Usar estas instrucciones T-SQL para crear la base de datos:

CREATE DATABASE Northwind

GO

USE Northwind

La base de datos Northwind vacía ha sido creada y con la instrucción.

Ahora abre con el Bloc de notas el script descargado anteriormente, selecciónalo todo, cópialo y pégalo, ejecútalo. Alternativamente hay otra opción para crear la base de datos, puedes abrir directamente el script northwind.sql que viene comprimido en el archivo de descarga

Prueba el acceso a los datos usando estas instrucciones T-SQL:

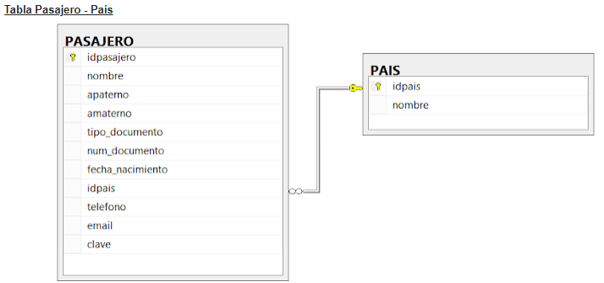
SELECT FirstName, LastName, City FROM Employees;

SELECT CustomerID FROM Customers;

Si puedes ver los datos, felicitaciones, la base de datos Northwind está lista.

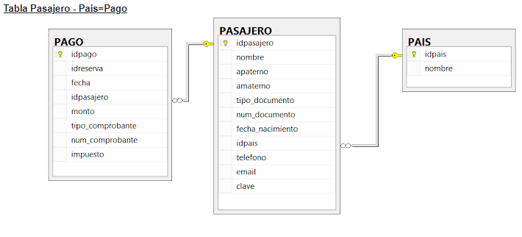
**Caso 3**

Implementar un procedimiento almacenado que muestre el listado de los países y su total de pasajeros. Realice este ejercicio en SQL Server



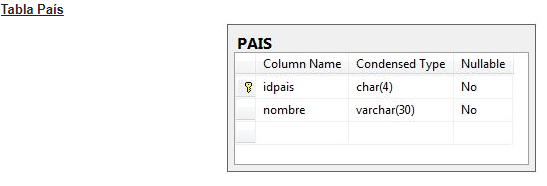
**Caso 4**

Implementar un procedimiento almacenado que permita mostrar los pagos de un determinado pasajero, considerar para este caso como parámetro de búsqueda el número de documento del pasajero. Realice este ejercicio en SQL Server



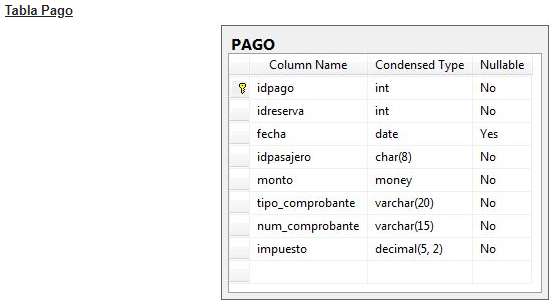
**Caso 5**

Implementar un procedimiento almacenado que permita registrar un Nuevo país, para este caso definir como parámetro de entrada todos los campos referentes a la tabla país. Realice este ejercicio en SQL Server



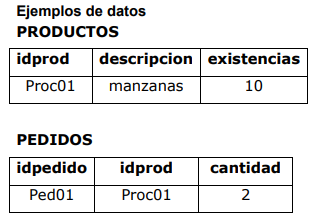
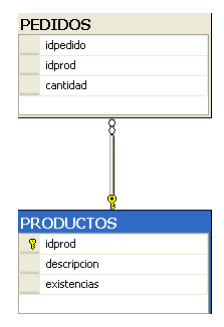
**Caso 6**

Implementar un procedimiento almacenado que retorne el total de pagos recibidos en una determinada fecha. Realice este ejercicio en SQL Server



**Caso 7**

La tabla productos contiene información general de los productos y la tabla pedidos contiene la información del pedido que se realiza de un cierto producto. Realice este ejercicio en SQL Server

idprod char(7)

descripcion varchar(25)

existencias int

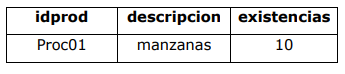
idpedido char(7)

idprod char(7)

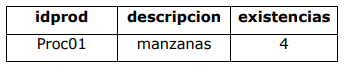
cantidad int

1. Crear un procedimiento almacenado que ingrese los valores en la tabla PRODUCTOS, y deberá verificar que el código del producto no exista para poder insertarlo, en caso de que el código del producto ya exista enviar un mensaje que diga ESTE PRODUCTO YA HA SIDO INGRESADO.
2. Crear un procedimiento almacenado que permita realizar un pedido EN LA TABLA PEDIDOS, este procedimiento deberá verificar si el código del producto ingresado existe en la tabla PRODUCTO, además si la cantidad a pedir del producto es mayor a la existencia del producto deberá enviar un mensaje que diga EXISTENCIA DEL PRODUCTO INSUFICIENTE, en caso de que la cantidad a pedir sea menor o igual deberá modificar el valor de la existencia. Adicional, evite que se ingresen valores nulos y que no se repitan en la base de datos.

Ejemplo:



Se realiza un pedido del producto Proc01 y se pide de cantidad 6, al terminar el procedimiento el registro de ese producto deberá ser:



**Caso 9.**

Dada la siguiente tabla persona (peso, estado). Realice un procedimiento para determinar si la persona puede donar sangre, si el peso el menor a 50 guarde en estado “no admitido”, en caso contrario sería “admitido”. Realice en el motor de su preferencia.

**Caso 10.**

Dada la siguiente tabla clientes (cedula, nombre, apellido)

Cree procedimientos para los siguientes casos:

1. Que inserte información en la tabla clientes. Ayuda (recibe tres parámetros de entrada)

2. Que actualice el nombre de un cliente. Ayuda (recibe dos parámetros, numero de

cedula de quien se va a actualizar y el nuevo nombre)

3. Que elimine un cliente. Ayuda (recibe un parámetro, numero de cedula de quien se va

eliminar.

4. Investigar procedimientos con paramentaros de salida.

5. Investigar cómo hacer un ciclo (while).

Realice en el motor de su preferencia.

**CASO 11- Realice los ejercicios en MySQL**

1. Escribe un procedimiento que reciba como parámetro de entrada una forma de pago, que será una cadena de caracteres (Ejemplo: PayPal, Transferencia, etc). Y devuelva como salida el pago de máximo valor realizado para esa forma de pago. Deberá hacer uso de la tabla pago de la base de datos jardineria.
2. Escribe un procedimiento que reciba como parámetro de entrada una forma de pago, que será una cadena de caracteres (Ejemplo: PayPal, Transferencia, etc). Y devuelva como salida los siguientes valores teniendo en cuenta la forma de pago seleccionada como parámetro de entrada:
   * el pago de máximo valor,
   * el pago de mínimo valor,
   * el valor medio de los pagos realizados,
   * la suma de todos los pagos,
   * el número de pagos realizados para esa forma de pago.

Deberá hacer uso de la tabla pago de la base de datos jardineria.

1. Escribe una función para la base de datos tienda que devuelva el número total de productos que hay en la tabla productos.
2. Escribe una función para la base de datos tienda que devuelva el valor máximo del precio de los productos de un determinado fabricante que se recibirá como parámetro de entrada. El parámetro de entrada será el nombre del fabricante.

**NOTA: DESARROLLE SUS RESPUESTAS EN LA SECCION G. RESULTADOS DE ESTA PLANTILLA.**

1. **RECURSOS:**

Laptop o PC con los recursos, Internet, material de clases preparado por el docente.

1. **RESULTADOS:**

**Caso 1:**

DROP TABLE IF EXISTS niños;

Create table niños (edad int, nombre varchar(50));

DROP TABLE IF EXISTS adultos;

Create table adultos (edad int, nombre varchar(50));

-- procedimiento

DELIMITER $$

DROP PROCEDURE IF EXISTS introducePersona$$

create procedure introducePersona (IN edad int, in nombre varchar(50))

begin

if edad < 18 then

insert into niños values(edad, nombre);

else

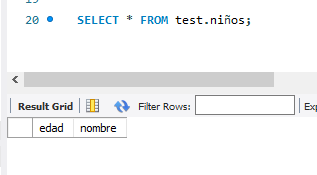
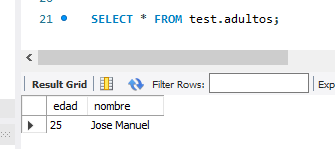
insert into adultos values(edad, nombre);

end if;

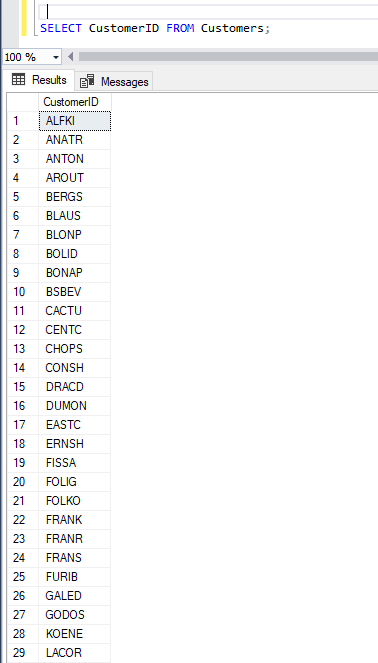
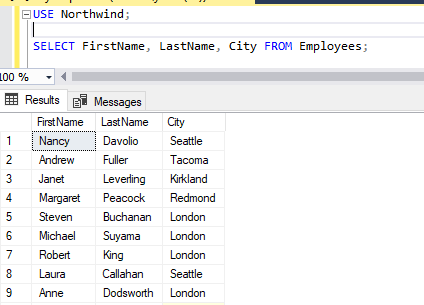
end $$

DELIMITER ;

call introducePersona(25,'Jose Manuel');



**Caso 2:**



**Caso 3:**

create procedure pasajerosxpais as

select c.nombre as Nombre,

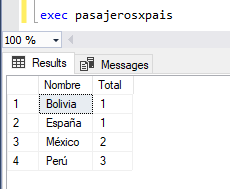
count(\*) as Total

from pasajero p join pais c

on p.idpais=c.idpais

group by c.nombre

go



**Caso 4:**

create procedure pagosxpasajero

@num\_documento varchar(12)

as

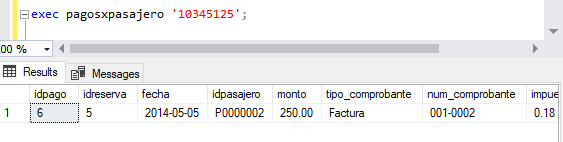
select \*

from pago where idpasajero=

(select idpasajero from pasajero

where num\_documento = @num\_documento)

go



**Caso 5:**

create procedure nuevopais

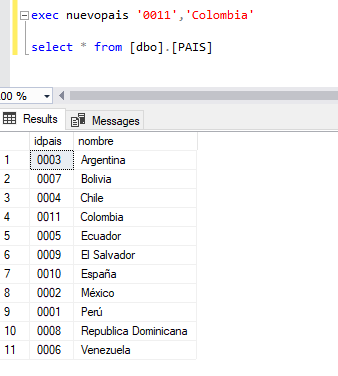
@idpais char(4),

@nombre varchar(30)

as

insert into pais (idpais,nombre) values(@idpais, @nombre)

go



**Caso 6:**

create procedure pagosxfecha

@fecha date,

@total money output

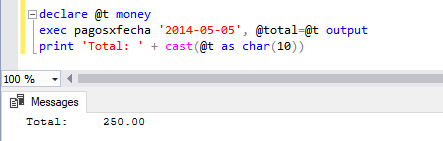
as

select @total=sum(monto)

from pago

where fecha=@fecha

go



**Caso 7**

DROP DATABASE IF EXISTS tiendita;

create database tiendita;

use tiendita;

create table productos(

idprod char(7) primary key not null,

descripcion varchar (25) not null,

existencias int

);

create table pedidos(

idpedido char (7) primary key not null,

idprod char(7) references productos,

cantidad int

);

**7.1 Procedimiento para ingresar datos en tabla productos**

go

create procedure insertarP

@prod char (7), @desc varchar (25), @exis int

AS

DECLARE @id char(7)

if ((@prod <>'') AND (@desc<>'') AND (@exis<>''))

BEGIN

select @id=idprod from productos where idprod=@prod

if (@id= @prod)

BEGIN

print 'ESTE PRODUCTO YA HA SIDO REGISTRADO'

RETURN

END

ELSE

BEGIN

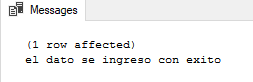
INSERT INTO productos values (@prod, @desc, @exis )

print 'el dato se ingreso con exito '

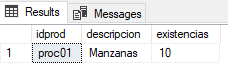
END

end

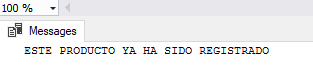
exec insertarP 'proc01', 'Manzanas', 10;



select\* from productos;



**\*si se trata de ingresar el mismo registro ocurre lo siguiente\***



**7.2 Procedimiento para ingresar datos en tabla PEDIDOS**

go

create procedure insertarPed

@ped char (7), @prd varchar (7), @cant int

AS

DECLARE @idprd char(7)

DECLARE @canti int

if ((@ped <>'') AND (@prd<>'') AND (@cant<>''))

BEGIN

select @idprd=idprod, @canti= existencias from productos where idprod=@prd

if (@idprd!= @prd)

BEGIN

print 'ESTE PRODUCTO NO EXISTE'

RETURN

END

ELSE

IF( (@idprd=@prd) AND (@canti> @cant))

BEGIN

INSERT INTO pedidos values (@ped, @prd, @cant )

update productos

set existencias = @canti-@cant

print 'el dato se ingreso con exito '

END

else If ( (@idprd=@prd) AND (@canti< @cant))

print 'NO HAY PRODUCTOS SUFICIENTES'

End

Else

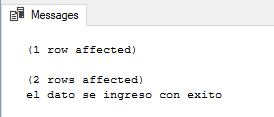
begin

print 'NO SE PUEDEN INGRESAR VALORES NULOS'

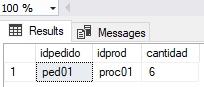
return

end

**exec insertarP 'ped01', 'proc01', 6;**

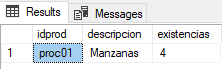


select\* from pedidos;



**\*DESPUÉS DE REALIZAR EL PEDIDO DE 6 PRODUCTOS, LA TABLA DE PRODUCTOS QUEDA DE LA SIGUIENTE MANERA\***

select\* from productos;



**\*si se trata de ingresar valores nulos\***

**exec insertarPed null, null, 6;**



**Caso 9**

DELIMITER //

CREATE PROCEDURE donar(in peso int)

BEGIN

DECLARE estado varchar(11);

DECLARE x INT;

SET x=peso;

SET estado= '';

IF x < 50 THEN

SET estado = 'No Admitido';

Else

SET estado = 'Admitido';

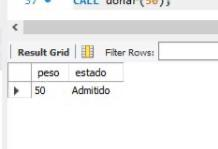
END IF;

SELECT peso, estado;

END; //

DELIMITER //

CALL donar(50);



**Caso 10**

create table cliente(

cedula varchar(12) primary key not null,

nombre varchar(30) not null,

apellido varchar(80) not null

**);**

1. Procedimiento para insertar datos en la tabla

DELIMITER //

create procedure insertarcli (IN

ced varchar(12),

nom varchar (30),

apel varchar(30)

)

BEGIN

declare continue handler for 1062

BEGIN

SELECT " REGISTRO DUPLICADO";

END;

insert into cliente values (ced, nom, apel);

END ;

DELIMITER//



call insertar('PE-15-1146', 'Majo', 'Martínez');

Imagen que contiene Tabla

Descripción generada automáticamente

1. procedimiento para actualizar el nombre del cliente

DELIMITER //

create procedure actualizarcli (IN

ced varchar(12),

nom varchar (30)

)

BEGIN

UPDATE cliente

set cedula= ced,

nombre= nom

where cedula= ced;

END ;

DELIMITER//



call actualizarcli('PE-15-1146', 'Maria');

select\* from cliente;

Interfaz de usuario gráfica

Descripción generada automáticamente

1. **procedimiento para eliminar cliente**

DELIMITER //

create procedure eliminarclien(IN

ced varchar(12)

)

BEGIN

delete from cliente where

cedula= ced;

END ;

DELIMITER//



call eliminarclien('PE-15-1146');



Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

1. **procedimientos con parametros de salida**

Un parámetro OUT pasa un valor del procedimiento a la persona que llama. Su valor inicial es NULL dentro del procedimiento, y su valor es visible para la persona que llama cuando el procedimiento vuelve.

Un parámetro INOUT es inicializado por la persona que llama, puede ser modificado por el procedimiento, y cualquier cambio realizado por el procedimiento es visible para la persona que llama cuando el procedimiento regresa.

**Ejemplo**

**delimiter //**

**create procedure pa\_autor\_sumaypromedio(**

**in p\_autor varchar(30),**

**out suma decimal(6,2),**

**out promedio decimal(6,2)**

**)**

**begin**

**select titulo,editorial,precio**

**from libros**

**where autor=p\_autor;**

**select sum(precio) into suma**

**from libros**

**where autor=p\_autor;**

**select avg(precio) into promedio**

**from libros**

**where autor=p\_autor;**

**end; //**

**delimiter ;**

1. **como hacer ciclos while**

**Sintax**

[*begin\_label*:] WHILE *search\_condition* DO

*statement\_list*

END WHILE [*end\_label*]

**ejemplo**

CREATE PROCEDURE dowhile()

BEGIN

DECLARE v1 INT DEFAULT 5;

WHILE v1 > 0 DO

...

SET v1 = v1 - 1;

END WHILE;

END;

**Caso 11**

DELIMITER //

CREATE PROCEDURE FDPAGO(in pago varchar(13))

BEGIN

DECLARE fpago varchar(13);

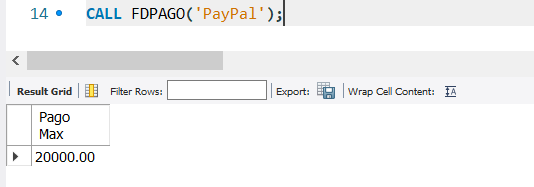
SET fpago = pago;

SELECT MAX(total) AS 'Pago Max' FROM pago WHERE forma\_pago = fpago;

END; //

DELIMITER //

CALL FDPAGO('PayPal');



DELIMITER //

CREATE PROCEDURE FDPAGO2(in pago varchar(13))

BEGIN

DECLARE fpago varchar(13);

SET fpago = pago;

SELECT MAX(total) AS 'Pago Max',

MIN(total) AS 'Pago Min',

AVG(ROUND(total, 2)) AS 'Valor Medio',

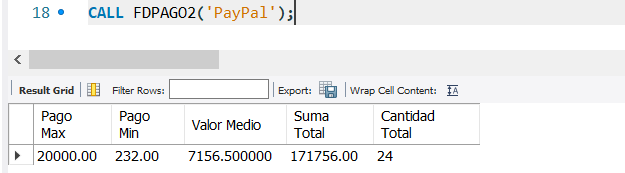
SUM(total) AS 'Suma Total',

COUNT(total) AS 'Cantidad Total' FROM pago WHERE forma\_pago = fpago;

END; //

DELIMITER //

CALL FDPAGO2('PayPal');



DELIMITER //

CREATE FUNCTION TTIENDA()

RETURNS INT

DETERMINISTIC READS SQL DATA BEGIN

DECLARE num INT;

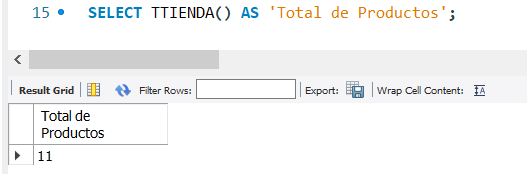
SET num = (SELECT COUNT(codigo) FROM producto);

RETURN num;

END; //

DELIMITER ;

SELECT TTIENDA() AS 'Total de Productos';



DELIMITER //

CREATE FUNCTION MTIENDA(tienda VARCHAR(20))

RETURNS FLOAT

DETERMINISTIC READS SQL DATA BEGIN

DECLARE ntienda VARCHAR(20);

DECLARE num FLOAT;

SET ntienda = tienda;

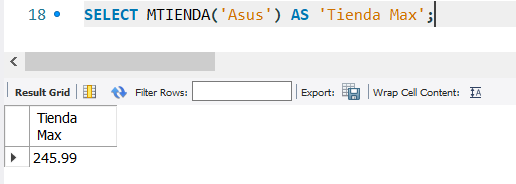
SET num = (SELECT MAX(p.precio) FROM producto p JOIN fabricante f ON p.codigo\_fabricante = f.codigo WHERE f.nombre = ntienda);

RETURN num;

END; //

DELIMITER ;

SELECT MTIENDA('Asus') AS 'Tienda Max';



1. **CONSIDERACIONES FINALES – CONCLUSIONES:**

Este laboratorio se cumplió el objetivo planteado que nos permite aplicar los conocimientos adquiridos durante las clases para crear procedimientos y funciones invocados por SQL. Al finalizar la actividad podemos enriquecer nuestra experiencia en el desarrollo y construcción de conocimientos del curso de Base de Datos II.

1. **BIBLIOGRAFIA:**

riptutorial. (s.f.). *procedimientos almacenados*. Obtenido de RIPTUTORIAL: https://riptutorial.com/es/mysql/example/11212/procedimiento-almacenado-con-parametros-in--out--inout-

Mysql. (2020). *While-statement*. Obtenido de Mysql: https://dev.mysql.com/doc/refman/8.0/en/while.html

**Material de clase de la profesora**

1. **RÚBRICAS:**

|  |  |
| --- | --- |
| ***Detalle - Evaluar*** | ***Valor de la respuesta*** |
| ***Sección G – (100 PUNTOS)*** | ***100 pts*** |