UTN – Mar del Plata
Base de Datos I – TUP 1er año

Trabajo Practico

Stored Procedures en MySQL

Condiciones de aprobacion: resolver correctamente al menos 6 de los 10 ejercicios. Se resuelve en papel. Puede consultarse la hoja de referencia adjunta. Letra ilegible descuenta puntos.

Script de tablas

Ejecutar el siguiente script para crear la base de datos de practica. Leer el esquema antes de comenzar.

```
CREATE TABLE especialidades (
especialidad_id INT AUTO_INCREMENT PRIMARY KEY,
nombre VARCHAR(100),
area VARCHAR(50)
);

CREATE TABLE medicos (
medico_id INT AUTO_INCREMENT PRIMARY KEY,
nombre VARCHAR(100),
matricula VARCHAR(20),
especialidad_id INT,
salario DECIMAL(10,2),
FOREIGN KEY (especialidad_id) REFERENCES especialidades(especialidad_id)
);

CREATE TABLE pacientes (
paciente_id INT AUTO_INCREMENT PRIMARY KEY,
nombre VARCHAR(100),
fecha_nacimiento DATE,
obra_social VARCHAR(100)
);

CREATE TABLE turnos (
turno_id INT AUTO_INCREMENT PRIMARY KEY,
paciente_id INT,
medico_id INT,
fecha DATE,
motivo VARCHAR(200),
estado VARCHAR(20),
FOREIGN KEY (paciente_id) REFERENCES pacientes(paciente_id),
FOREIGN KEY (medico_id) REFERENCES medicos(medico_id)
);
```

Hoja de referencia

Tipos de variables

Tipo	Declaracion	Scope	Ejemplo
Local (DECLARE)	DECLARE v_nombre VARCHAR(100);	Solo dentro del SP	DECLARE v_total INT DEFAULT 0;
De sesion (@)	No se declara	Toda la sesion activa	SET @resultado = 0;
Parametro IN	IN p_id INT	Solo lectura dentro del SP	CALL MiSP(5);
Parametro OUT	OUT p_nombre VARCHAR(100)	Solo escritura, sale al llamador	CALL MiSP(5, @nombre);
Parametro INOUT	INOUT p_msg VARCHAR(200)	Lectura y escritura	CALL MiSP(5, @msg);

```
-- Ejemplo combinado:
CREATE PROCEDURE Ejemplo(IN p_id INT, OUT p_nombre VARCHAR(100), INOUT p_msg VARCHAR(200))
BEGIN
DECLARE v_salario DECIMAL(10,2);
SELECT nombre, salario INTO p_nombre, v_salario FROM medicos WHERE medico_id = p_id;
SET p_msg = CONCAT(p_msg, ' | Salario: ', v_salario);
END
```

Funciones condicionales

Funcion	Ejemplo	Descripcion
IFNULL(expr, alt)	IFNULL(obra_social, 'Sin OS')	Retorna alt si expr es NULL
COALESCE(a, b, ...)	COALESCE(motivo, obs, 'Sin datos')	Retorna el primer valor no NULL
IF(cond, si, no)	IF(salario > 0, 'Activo', 'Inactivo')	Condicional de una linea

CASE - dos formas de uso

Forma 1 - CASE simple (compara un valor contra opciones fijas):

```
CASE estado
WHEN 'Pendiente' THEN 'En espera'
WHEN 'Cancelado' THEN 'No asistio'
ELSE 'Otro estado'
END
```

Forma 2 - CASE buscado (cada WHEN tiene su propia condicion):

```
CASE
WHEN salario < 200000 THEN 'Salario bajo'
WHEN salario BETWEEN 200000 AND 500000 THEN 'Salario medio'
WHEN salario > 500000 THEN 'Salario alto'
ELSE 'Sin clasificar'
END
```

Funciones de texto

Funcion	Ejemplo	Descripcion
CONCAT(a, b, ...)	CONCAT('Hola ', nombre)	Concatena cadenas
CONCAT_WS(sep,...)	CONCAT_WS(', ', nombre, matricula)	Concatena con separador fijo

Funciones de fecha y hora

Funcion	Ejemplo	Descripcion
`CURDATE()`	`SELECT CURDATE()`	Fecha actual (DATE)
`NOW()`	`SELECT NOW()`	Fecha y hora actual (DATETIME)
`DATEDIFF(a, b)`	`DATEDIFF(fecha, CURDATE())`	Diferencia en dias entre a y b
`YEAR / MONTH / DAY`	`YEAR(fecha_nacimiento)`	Extrae parte de una fecha
`DATE_FORMAT(d, fmt)`	`DATE_FORMAT(fecha, '%d/%m/%Y')`	Formatea una fecha
`TIMESTAMPDIFF(u,a,b)`	`TIMESTAMPDIFF(YEAR, nacimiento, CURDATE())`	Diferencia en unidad indicada

Funciones propias de SP

Funcion	Ejemplo	Descripcion
`LAST_INSERT_ID()`	`SET v_id = LAST_INSERT_ID()`	ID generado por el ultimo INSERT
`ROW_COUNT()`	`SET v_filas = ROW_COUNT()`	Filas afectadas por la ultima operacion
`SELECT INTO`	`SELECT nombre INTO v_nom FROM ...`	Carga un valor en una variable local

Ejercicios

Ejercicio 1 – MostrarDatosMedico

Temas: DECLARE, SELECT INTO, SELECT final

Crear un SP que reciba el ID de un medico y muestre su nombre, matricula y el nombre de su especialidad en un unico SELECT.

Parametros de entrada:

- p_medico_id INT

Debe mostrar:

- Nombre del medico, matricula y nombre de la especialidad.

Ejercicio 2 – ObtenerNombreMedico

Temas: Parametros OUT, variables de sesion @

Crear un SP que reciba el ID de un medico y devuelva su nombre y el nombre de su especialidad a traves de parametros de salida.

Parametros de entrada:

- p_medico_id INT

Parametros de salida:

- p_nombre VARCHAR
- p_especialidad VARCHAR

Consideraciones:

- Al llamar al SP, los valores de salida deben poder consultarse con un SELECT de variables de sesion.

Ejercicio 3 – VerificarMedico

Temas: IS NULL, IF/ELSE, CONCAT

Crear un SP que reciba el ID de un medico e informe si existe o no en la base de datos.

Parametros de entrada:

- p_medico_id INT

Debe mostrar:

- Si el medico no existe: un mensaje indicandolo.
- Si el medico existe: un mensaje que incluya su nombre.

Ejercicio 4 – ClasificarMedico

Temas: CASE, rangos, SELECT con variables

Crear un SP que reciba el ID de un medico y muestre su nombre, salario y una clasificacion salarial.

Parametros de entrada:

- p_medico_id INT

Consideraciones:

- La clasificacion debe seguir esta escala:
  - Menor a $200.000 -> Salario bajo
  - Entre $200.000 y $500.000 -> Salario medio
  - Mayor a $500.000 -> Salario alto

Debe mostrar:

- Nombre, salario y clasificacion del medico.

Ejercicio 5 – RegistrarTurno

Temas: INSERT, LAST_INSERT_ID(), parametro OUT

Crear un SP que registre un nuevo turno en el sistema y devuelva el ID generado.

Parametros de entrada:

- p_paciente_id INT
- p_medico_id INT
- p_fecha DATE
- p_motivo VARCHAR

Parametros de salida:

- p_turno_id INT

Consideraciones:

- Todo turno nuevo debe crearse con estado 'Pendiente'.
- El ID del turno creado debe quedar disponible como parametro de salida y mostrarse en un mensaje.

Ejercicio 6 – CancelarTurno

Temas: UPDATE, EXISTS, ROW_COUNT(), parametro OUT

Crear un SP que cancele un turno existente y devuelva la cantidad de filas afectadas.

Parametros de entrada:

- p_turno_id INT

Parametros de salida:

- p_filas_afectadas INT

Consideraciones:

- Solo se puede cancelar un turno que exista y cuyo estado sea 'Pendiente'.
- Si no cumple esa condicion, mostrar un mensaje de error sin realizar ninguna modificacion.
- Si se cancela con exito, informar la cantidad de filas afectadas.

Ejercicio 7 – BuscarTurnosPorMedico

Temas: BETWEEN, EXISTS, DATEDIFF, CURDATE, JOIN

Crear un SP que liste los turnos pendientes de un medico dentro de un rango de fechas, mostrando el nombre del paciente y los dias restantes hasta cada turno.

Parametros de entrada:

- p_medico_id INT
- p_fecha_desde DATE
- p_fecha_hasta DATE

Consideraciones:

- Si el medico no existe, mostrar un mensaje de error.
- Solo listar turnos con estado 'Pendiente'.
- Los resultados deben ordenarse por fecha de manera ascendente.

Debe mostrar:

- Nombre del paciente, fecha del turno, motivo y dias restantes hasta el turno.

Ejercicio 8 – ResumenMedico

Temas: COUNT, MIN, IFNULL, SELECT INTO con agrupamiento

Crear un SP que genere un resumen de la actividad de un medico.

Parametros de entrada:

- p_medico_id INT

Consideraciones:

- Si el medico no tiene ningun turno registrado, mostrar un mensaje indicandolo.
- Si la fecha del proximo turno pendiente no existe, mostrar el texto 'Sin turnos pendientes' en su lugar.

Debe mostrar:

- Total de turnos, cantidad de turnos pendientes, cantidad de cancelados y fecha del proximo turno pendiente.

Ejercicio 9 – RegistrarTurnoCompleto

Temas: Integrador - EXISTS, CURDATE, COALESCE, LAST_INSERT_ID(), OUT

Crear un SP que registre un nuevo turno realizando todas las validaciones necesarias antes de insertarlo.

Parametros de entrada:

- p_paciente_id INT
- p_medico_id INT
- p_fecha DATE
- p_motivo VARCHAR (opcional)

Parametros de salida:

- p_turno_id INT
- p_mensaje VARCHAR

Consideraciones:

- Validar que el paciente exista; si no, informarlo y no continuar.
- Validar que el medico exista; si no, informarlo y no continuar.
- No permitir registrar un turno con fecha anterior a la fecha actual.
- No permitir que un mismo medico tenga dos turnos 'Pendiente' en la misma fecha.
- Si el motivo no es informado, registrarlo como 'Sin especificar'.
- Si todas las validaciones pasan, insertar el turno y retornar su ID y un mensaje descriptivo.

Ejercicio 10 – GestionarPaciente

Temas: Integrador - INOUT, DELETE, UPDATE, EXISTS, ROW_COUNT()

Crear un SP que permita actualizar el nombre de un paciente o eliminarlo del sistema, segun la accion indicada.

Parametros de entrada:

- p_paciente_id INT
- p_accion VARCHAR - valores posibles: 'ACTUALIZAR' o 'ELIMINAR'
- p_nuevo_nombre VARCHAR - requerido solo si la accion es 'ACTUALIZAR'

Parametro INOUT:

- p_mensaje VARCHAR - debe ingresar con un prefijo (ej: 'Resultado: ') y retornar ese prefijo concatenado con el resultado de la operacion

Consideraciones:

- Si el paciente no existe, concatenar el error al mensaje y no realizar ninguna operacion.
- Si la accion es 'ACTUALIZAR', modificar el nombre y reflejarlo en el mensaje.
- Si la accion es 'ELIMINAR', borrar el registro e incluir en el mensaje la cantidad de filas afectadas.
- Si la accion no es ninguna de las dos validas, informarlo en el mensaje.
- En todos los casos, el mensaje de salida debe conservar el prefijo con el que ingreso.