-- FUNCION 1: fn\_ObtenerNombreCompletoUsuario

-- Objetivo: Combina el nombre y el apellido del organizador/usuario en un solo campo,

-- evitando la repetición de lógica CONCAT() en cada consulta.

DELIMITER //

CREATE FUNCTION fn\_ObtenerNombreCompletoUsuario(p\_idUsuario INT)

RETURNS VARCHAR(101)

DETERMINISTIC

BEGIN

DECLARE v\_nombreCompleto VARCHAR(101);

SELECT CONCAT(nombre, ' ', apellido)

INTO v\_nombreCompleto

FROM Usuario

WHERE idUsuario = p\_idUsuario;

RETURN COALESCE(v\_nombreCompleto, 'Usuario Desconocido');

END //

DELIMITER ;

-- FUNCION 2: fn\_ContarEventosOrganizados

-- Objetivo: Calcula cuántos eventos ha organizado un usuario específico. Útil para métricas de gestión y perfiles de organizadores.

-- Retorna el número total de eventos organizados por un usuario.

DELIMITER //

CREATE FUNCTION fn\_ContarEventosOrganizados(p\_idUsuario INT)

RETURNS INT

DETERMINISTIC

BEGIN

DECLARE v\_conteoEventos INT;

SELECT COUNT(uo.Evento\_idEvento)

INTO v\_conteoEventos

FROM usuario\_organizador uo

WHERE uo.Usuario\_idUsuario = p\_idUsuario;

RETURN v\_conteoEventos;

END //

DELIMITER ;

-- FUNCION 3: fn\_CalcularDuracionHorasEvento

-- Objetivo: Calcula la duración total de un evento en horas (puede ser decimal) basándose en las columnas horaInicioEvento y horaFinEvento.

-- Esto es fundamental para la planificación de recursos.

-- Retorna la duración total de un evento en horas.

DELIMITER //

CREATE FUNCTION fn\_CalcularDuracionHorasEvento(p\_idEvento INT)

RETURNS DECIMAL(5, 2)

DETERMINISTIC

BEGIN

DECLARE v\_duracionSegundos INT;

DECLARE v\_duracionHoras DECIMAL(5, 2);

SELECT TIMESTAMPDIFF(SECOND, horaInicioEvento, horaFinEvento)

INTO v\_duracionSegundos

FROM Evento

WHERE idEvento = p\_idEvento;

SET v\_duracionHoras = v\_duracionSegundos / 3600.0;

RETURN COALESCE(v\_duracionHoras, 0.00);

END //

DELIMITER ;

-- FUNCION 4: fn\_ObtenerEstadoEvaluacion:

-- Objetivo: Determinar el estado consolidado de un evento basándose en su última evaluación.

-- Es fundamental para que el backend no tenga que buscar en la tabla EvaluacionEvento constantemente. Simplifica la consulta de eventos en la página principal.

-- Esta función garantiza que siempre se muestre el estado más reciente, ya sea 'Aprobado', 'Revisión', 'Rechazado' o 'Pendiente'.

-- Retorna el estado consolidado y más reciente de la evaluación de un evento.

DELIMITER //

CREATE FUNCTION fn\_ObtenerEstadoEvaluacion(p\_idEvento INT)

RETURNS VARCHAR(50)

DETERMINISTIC

BEGIN

DECLARE v\_estado VARCHAR(50);

-- Busca el estado de la evaluación más reciente.

SELECT e.estado

INTO v\_estado

FROM EvaluacionEvento e

WHERE e.Evento\_idEvento = p\_idEvento

-- ¡CORRECCIÓN FINAL! Usamos los nombres de columna sin tilde

ORDER BY e.fechaEvaluacion DESC, e.idEvaluacionEvento DESC

LIMIT 1;

-- Si no hay evaluación, el estado es 'Pendiente'

RETURN COALESCE(v\_estado, 'Pendiente');

END //

DELIMITER ;

-- FUNCION 5: fn\_VerificarDisponibilidadLugar:

-- Objetivo: La función más importante para la gestión de eventos. Verifica si un Lugar está ocupado en un rango de tiempo y fecha específicos.

-- Esto debe ejecutarse antes de insertar o actualizar un evento para prevenir overbooking.

-- Utiliza los parámetros de fecha y hora para validar si hay una superposición (overlap) con cualquier otro evento ya registrado en ese lugar.

-- Verifica si un Lugar está disponible en un rango de fecha/hora.

DELIMITER //

CREATE FUNCTION fn\_VerificarDisponibilidadLugar(

p\_idLugar INT,

p\_fecha DATE,

p\_horaInicio TIME,

p\_horaFin TIME,

p\_idEventoExcluir INT

)

RETURNS BOOLEAN

DETERMINISTIC

BEGIN

DECLARE v\_isOccupied INT;

SELECT COUNT(\*) INTO v\_isOccupied

FROM Evento e

JOIN lugar\_evento le ON e.idEvento = le.Evento\_idEvento

WHERE le.Lugar\_idLugar = p\_idLugar

AND e.fechaEvento = p\_fecha

AND e.idEvento <> p\_idEventoExcluir

AND p\_horaInicio < e.horaFinEvento

AND p\_horaFin > e.horaInicioEvento;

RETURN v\_isOccupied = 0;

END //

DELIMITER ;

-- FUNCION 6: fn\_ObtenerTopOrganizadorPorFacultad:

-- Propósito: Provee una métrica de gestión importante: quién es el organizador (Docente o Estudiante) que más eventos ha creado dentro de una Facultad específica.

-- Esto ayuda a identificar líderes y tendencias de organización.

-- Retorna el nombre del usuario que más eventos ha organizado en una Facultad.

DELIMITER //

CREATE FUNCTION fn\_ObtenerTopOrganizadorPorFacultad(p\_idFacultad INT)

RETURNS VARCHAR(150)

DETERMINISTIC

BEGIN

DECLARE v\_topOrganizador VARCHAR(150);

SELECT CONCAT(u.nombre, ' ', u.apellido, ' (', COUNT(e.idEvento), ' eventos)')

INTO v\_topOrganizador

FROM Evento e

JOIN usuario\_organizador uo ON e.idEvento = uo.Evento\_idEvento

JOIN Usuario u ON uo.Usuario\_idUsuario = u.idUsuario

JOIN Facultad\_usuario fu ON u.idUsuario = fu.Usuario\_idUsuario

WHERE fu.Facultad\_idFacultad = p\_idFacultad

GROUP BY u.idUsuario, u.nombre, u.apellido

ORDER BY COUNT(e.idEvento) DESC

LIMIT 1;

RETURN COALESCE(v\_topOrganizador, 'No hay organizadores registrados en esta facultad.');

END //

DELIMITER ;

DELIMITER ;