Lab 23 - En este caso se utilizo un proyecto de (pjsk)
Primera consulta crea un procedimiento para obtener las estadísticas de un jugador

```
CREATE PROCEDURE sp_obtener_estadistica_jugador( IN p_usuario_id INT, OUT p_nivel INT, OUT p_promedio_score DECIMAL(10,2), OUT p_mejor_rango VARCHAR(3), OUT p_canciones_jugadas INT ) BEGIN SELECT nivel INTO p_nivel FROM usuarios WHERE usuario_id = p_usuario_id; SELECT AVG(puntaje) INTO p_promedio_score FROM scores WHERE usuario_id = p_usuario_id; SELECT MIN(rango) INTO p_mejor_rango FROM scores WHERE usuario_id = p_usuario_id; SELECT COUNT(DISTINCT cancion_id) INTO p_canciones_jugadas FROM scores WHERE usuario_id = p_usuario_id; END;
```

[Edit inline] [Edit] [Create PHP code]

Segunda consulta crea un procedimiento para recomendar canciones a los jugadores

```
CREATE PROCEDURE sp_recomendar_canciones_nivel( IN p_usuario_id INT, IN p_dificultad VARCHAR(10), IN p_num_recomendaciones INT ) BEGIN DECLARE v_nivel_jugador INT; SELECT nivel INTO v_nivel_jugador FROM usuarios WHERE usuario_id = p_usuario_id; SELECT c.cancion_id, c.titulo, c.artista, d.dificultad, d.nivel, CONCAT(FLOOR(d.nivel/10), '★', MOD(d.nivel,10)) as dificultad_visual FROM canciones c JOIN dificultades d ON c.cancion_id = d.cancion_id WHERE d.dificultad = p_dificultad AND d.nivel BETWEEN v_nivel_jugador - 2 AND v_nivel_jugador + 2 AND NOT EXISTS ( SELECT 1 FROM scores s WHERE s.cancion_id = c.cancion_id AND s.usuario_id = p_usuario_id AND s.dificultad = p_dificultad ) ORDER BY RAND() LIMIT p_num_recomendaciones; END;
```

última consulta crea un procedimiento para actualizar los rankings de un evento

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0022 seconds.)

```
CREATE PROCEDURE sp_actualizar_rankings_evento( IN p_evento_id INT, IN p_usuario_id INT, IN p_puntos_obtenidos INT, OUT p_puntos_totales INT, OUT p_posicion_actual INT, OUT p_rango_evento VARCHAR(20) ) BEGIN INSERT INTO evento_participantes (evento_id, usuario_id, puntos) VALUES (p_evento_id, p_usuario_id, p_puntos_obtenidos) ON DUPLICATE KEY UPDATE puntos = puntos + p_puntos_obtenidos; SELECT puntos INTO p_puntos_totales FROM evento_participantes WHERE evento_id = p_evento_id AND usuario_id = p_usuario_id; SELECT COUNT(*) + 1 INTO p_posicion_actual FROM evento_participantes WHERE evento_id = p_evento_id AND puntos > p_puntos_totales; SET p_rango_evento = CASE WHEN p_posicion_actual <= 10 THEN 'Top 10' WHEN p_posicion_actual <= 100 THEN 'Top 100' WHEN p_posicion_actual <= 1000 THEN 'Top 1000' ELSE 'Participante' END; SELECT p_[...]
```

[Edit]

Routines

Check all | Export | Drop

Search | Create new routine

	Name	Type	Returns	
☐	sp_actualizar_rankings_evento	PROCEDURE		Edit Execute Export Drop
☐	sp_obtener_estadisticas_jugador	PROCEDURE		Edit Execute Export Drop
☐	sp_recomendar_canciones_nivel	PROCEDURE		Edit Execute Export Drop

Lab25 - Transiciones

Transición para guardar el score

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0001 seconds.)

```
START TRANSACTION;
```

[Edit inline] [Edit] [Create PHP code]

1 row inserted.
Inserted row id: 1 (Query took 0.0003 seconds.)

```
INSERT INTO scores (usuario_id, cancion_id, puntaje) VALUES (1, 5, 950000);
```

Transición para cambiar el nombre

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0001 seconds.)

```
START TRANSACTION;
```

[Edit inline] [Edit] [Create PHP code]

1 row affected. (Query took 0.0003 seconds.)

```
UPDATE usuarios SET nombre = 'MikuFan' WHERE usuario_id = 1;
```

[Edit inline] [Edit] [Create PHP code]

Actualizar puntuajes

```
START TRANSACTION;
```

Edit inline] [Edit] [Create PHP code]

1 row affected. (Query took 0.0005 seconds.)

```
UPDATE scores SET puntaje = 980000 WHERE usuario_id = 1 AND cancion_id = 5 AND puntaje < 980000;
```