El error que aparece se debe a que estás intentando ejecutar una subconsulta que devuelve múltiples columnas en una posición que espera una única columna. Específicamente, la subconsulta:

(SELECT anno, JSON\_ARRAYAGG(torneo\_id) AS Titulos\_anno FROM titulos WHERE tenista\_id = ? GROUP BY anno ORDER BY anno)

intenta devolver dos columnas (anno y Titulos\_anno), pero no es posible asignarlas directamente en una posición donde se espera una sola columna (como una subconsulta en un SELECT).

**Cómo solucionarlo**

Para devolver los títulos agrupados por año, debes ajustar tu consulta para que la subconsulta (que utiliza JSON\_ARRAYAGG) devuelva un solo valor estructurado. Esto puede hacerse de varias maneras, pero lo más efectivo es devolver un JSON que contenga todos los años y sus títulos. Aquí está el código ajustado:

**Versión corregida**

$datos = "";

$conexion = conectarPDO($database);

if ($\_SERVER['REQUEST\_METHOD'] == 'GET') {

if (isset($\_GET['id'])) {

$consulta = $conexion->prepare("

SELECT

id,

nombre,

apellidos,

altura,

anno\_nacimiento AS 'año de nacimiento',

(SELECT

JSON\_ARRAYAGG(

JSON\_OBJECT(

'anno', anno,

'titulos', JSON\_ARRAYAGG(torneo\_id)

)

)

FROM titulos

WHERE tenista\_id = ?

GROUP BY anno

) AS titulos

FROM

tenistas

WHERE

id = ?

");

$consulta->bindParam(1, $\_GET['id']);

$consulta->bindParam(2, $\_GET['id']);

$consulta->execute();

if ($consulta->rowCount() > 0) {

salidaDatos(

json\_encode($consulta->fetch(PDO::FETCH\_ASSOC)),

array('Content-Type: application/json', 'HTTP/1.1 200 OK')

);

} else {

salidaDatos('No se encuentra el tenista pedido.', array('HTTP/1.1 404 Not Found'));

}

}

}

**Explicación del ajuste**

1. **Subconsulta estructurada con JSON:**
   * La subconsulta ahora utiliza JSON\_OBJECT para crear un objeto JSON por cada año con los títulos asociados.
   * JSON\_ARRAYAGG agrupa los títulos de cada año.
   * Finalmente, JSON\_ARRAYAGG en la capa superior crea un array de objetos (uno por cada año).

La subconsulta completa:

(SELECT

JSON\_ARRAYAGG(

JSON\_OBJECT(

'anno', anno,

'titulos', JSON\_ARRAYAGG(torneo\_id)

)

)

FROM titulos

WHERE tenista\_id = ?

GROUP BY anno

)

1. **Colocación del resultado:**
   * Este JSON estructurado se devuelve como el campo titulos en la consulta principal.
2. **Resultado esperado en Postman:** Si consultas un tenista con id=1, el JSON devuelto debería tener la siguiente estructura:
3. {
4. "id": 1,
5. "nombre": "Rafael",
6. "apellidos": "Nadal",
7. "altura": 185,
8. "año de nacimiento": 1986,
9. "titulos": [
10. {
11. "anno": 2005,
12. "titulos": [2]
13. },
14. {
15. "anno": 2008,
16. "titulos": [2, 3]
17. },
18. {
19. "anno": 2010,
20. "titulos": [2, 3, 4]
21. }
22. ]
23. }
24. **Control de errores:** Si no se encuentra el tenista con el ID proporcionado, se devuelve un mensaje de error con código HTTP 404.

**Notas adicionales**

* **Versión de MySQL:** Asegúrate de que tu base de datos sea compatible con las funciones JSON como JSON\_ARRAYAGG y JSON\_OBJECT. Estas funciones están disponibles a partir de MySQL 5.7.
* **Depuración:** Si el error persiste, verifica que los datos en tu tabla titulos estén correctamente relacionados con tenistas y que el id proporcionado en la consulta exista.

Para que la salida sea un JSON en el que cada año tenga asociada una lista con los títulos correspondientes, puedes ajustar tu consulta SQL para crear un JSON estructurado de esa manera. Aquí está la solución corregida y adaptada:

**Código PHP Corregido**

$datos = "";

$conexion = conectarPDO($database);

if ($\_SERVER['REQUEST\_METHOD'] == 'GET') {

if (isset($\_GET['id'])) {

$consulta = $conexion->prepare("

SELECT

id,

nombre,

apellidos,

altura,

anno\_nacimiento AS 'año de nacimiento',

(SELECT

JSON\_OBJECTAGG(

anno,

JSON\_ARRAYAGG(torneo\_id)

)

FROM titulos

WHERE tenista\_id = ?

GROUP BY anno

) AS titulos

FROM

tenistas

WHERE

id = ?

");

$consulta->bindParam(1, $\_GET['id']);

$consulta->bindParam(2, $\_GET['id']);

$consulta->execute();

if ($consulta->rowCount() > 0) {

salidaDatos(

json\_encode($consulta->fetch(PDO::FETCH\_ASSOC)),

array('Content-Type: application/json', 'HTTP/1.1 200 OK')

);

} else {

salidaDatos('No se encuentra el tenista pedido.', array('HTTP/1.1 404 Not Found'));

}

}

}

**Explicación de la consulta**

1. **Subconsulta con JSON\_OBJECTAGG:**
   * JSON\_OBJECTAGG crea un objeto JSON donde cada clave es un año (anno) y su valor es un array de títulos (JSON\_ARRAYAGG(torneo\_id)).
   * La subconsulta devuelve un objeto JSON estructurado por año con una lista de títulos asociada.

Subconsulta:

(SELECT

JSON\_OBJECTAGG(

anno,

JSON\_ARRAYAGG(torneo\_id)

)

FROM titulos

WHERE tenista\_id = ?

)

1. **Consulta principal:**
   * La consulta principal incluye los datos del tenista (como id, nombre, etc.) y el campo titulos que contiene el JSON generado por la subconsulta.
2. **Salida esperada en JSON:** Si haces una solicitud GET con un id=1, obtendrás un JSON como este:
3. {
4. "id": 1,
5. "nombre": "Rafael",
6. "apellidos": "Nadal",
7. "altura": 185,
8. "año de nacimiento": 1986,
9. "titulos": {
10. "2005": [2],
11. "2008": [2, 3],
12. "2010": [2, 3, 4],
13. "2022": [2]
14. }
15. }

**Notas Importantes**

1. **Requisitos de MySQL:**
   * JSON\_OBJECTAGG está disponible desde MySQL 5.7. Si estás utilizando una versión anterior, será necesario manejar los datos manualmente desde el código PHP.
2. **Control de errores:**
   * Si el id proporcionado no tiene registros en la tabla titulos, el campo titulos será null.
   * Si no existe el tenista, se devuelve el mensaje de error con código HTTP 404.
3. **Postman:**
   * Asegúrate de enviar correctamente el parámetro id en la URL como ?id=<valor>.

¿Esto cumple con el formato esperado de la salida? 😊

El problema ocurre porque JSON\_OBJECTAGG y JSON\_ARRAYAGG deben operar en contextos compatibles. En este caso, parece que MySQL no permite combinar estas funciones directamente dentro de una única consulta de la manera en que están siendo utilizadas.

Para corregir el problema, **retira el uso de JSON\_ARRAYAGG dentro de JSON\_OBJECTAGG y ajusta la estructura de la consulta**. Una manera es utilizando una subconsulta para procesar JSON\_ARRAYAGG, o reestructurar el SQL para evitar ambigüedades.

**Solución con Subconsulta**

SELECT

JSON\_OBJECTAGG(

anno, titulos

) AS titulos

FROM (

SELECT

anno,

JSON\_ARRAYAGG(torneo\_id) AS titulos

FROM titulos

WHERE tenista\_id = 1

GROUP BY anno

) AS titulos\_por\_anno;

**Explicación**

1. **Subconsulta (titulos\_por\_anno):**
   * Esta parte agrupa los títulos (torneo\_id) por año (anno) usando JSON\_ARRAYAGG:
2. SELECT
3. anno,
4. JSON\_ARRAYAGG(torneo\_id) AS titulos
5. FROM titulos
6. WHERE tenista\_id = 1
7. GROUP BY anno
8. **Consulta Principal:**
   * Luego, JSON\_OBJECTAGG toma el resultado de la subconsulta y lo convierte en un objeto JSON donde:
     + Cada clave es el anno.
     + Cada valor es un array de títulos.
9. **Salida esperada:** Si los datos son correctos, la consulta devolverá un JSON como este:
10. {
11. "2005": [2],
12. "2008": [2, 3],
13. "2010": [2, 3, 4],
14. "2022": [2]
15. }

**Por qué funciona**

* **Subconsulta:** Asegura que JSON\_ARRAYAGG genere primero los arrays agrupados por año.
* **JSON\_OBJECTAGG:** Combina el resultado agrupado de la subconsulta en un solo objeto JSON.

**Integración en PHP**

Si deseas usar esta consulta en tu código PHP, puedes hacerlo así:

$datos = "";

$conexion = conectarPDO($database);

if ($\_SERVER['REQUEST\_METHOD'] == 'GET') {

if (isset($\_GET['id'])) {

$consulta = $conexion->prepare("

SELECT

JSON\_OBJECTAGG(

anno, titulos

) AS titulos

FROM (

SELECT

anno,

JSON\_ARRAYAGG(torneo\_id) AS titulos

FROM titulos

WHERE tenista\_id = ?

GROUP BY anno

) AS titulos\_por\_anno;

");

$consulta->bindParam(1, $\_GET['id']);

$consulta->execute();

if ($consulta->rowCount() > 0) {

$resultado = $consulta->fetch(PDO::FETCH\_ASSOC);

salidaDatos(

json\_encode($resultado),

array('Content-Type: application/json', 'HTTP/1.1 200 OK')

);

} else {

salidaDatos('No se encontraron títulos para el tenista.', array('HTTP/1.1 404 Not Found'));

}

}

}

**Notas Adicionales**

1. **Versión de MySQL:**
   * JSON\_OBJECTAGG y JSON\_ARRAYAGG están disponibles desde MySQL 5.7. Asegúrate de estar utilizando una versión compatible.
2. **Verificación de datos:**
   * Comprueba que los valores de tenista\_id, anno y torneo\_id en la tabla titulos sean correctos y estén relacionados.

¿Te ayudó esta solución? 😊

El error se produce debido a que la subconsulta no está correctamente integrada dentro de la consulta principal. MySQL no permite que una subconsulta con alias (como titulos\_anno) sea directamente utilizada en una lista de columnas en la forma que estás intentando.

Para solucionarlo, hay que reestructurar la consulta y asegurarse de que la subconsulta esté correctamente anidada en el contexto de la consulta principal. Aquí te dejo la consulta corregida:

**Consulta Corregida**

SELECT

t.id,

t.nombre,

t.apellidos,

t.altura,

t.anno\_nacimiento AS 'año de nacimiento',

(

SELECT

JSON\_OBJECTAGG(

anno, titulos

)

FROM (

SELECT

anno,

JSON\_ARRAYAGG(torneo\_id) AS titulos

FROM titulos

WHERE tenista\_id = t.id

GROUP BY anno

) AS titulos\_anno

) AS titulos

FROM

tenistas t

WHERE

t.id = 1;

**Explicación**

1. **Consulta principal:**
   * La consulta selecciona los datos del tenista (id, nombre, apellidos, etc.) desde la tabla tenistas.
2. **Subconsulta de títulos:**
   * La subconsulta anidada agrupa los títulos (torneo\_id) por año (anno) usando JSON\_ARRAYAGG.
   * Luego, JSON\_OBJECTAGG combina esos resultados en un único objeto JSON, donde cada clave es un anno y su valor es un array de títulos.
3. **Enlace con la tabla principal:**
   * La subconsulta usa WHERE tenista\_id = t.id, asegurando que solo se seleccionen los títulos correspondientes al id del tenista actual.
4. **Alias:**
   * AS titulos asigna el resultado de la subconsulta a una columna en la consulta principal.

**Salida esperada**

Si ejecutas esta consulta con un id válido, obtendrás un resultado similar a este:

{

"id": 1,

"nombre": "Rafael",

"apellidos": "Nadal",

"altura": 185,

"año de nacimiento": 1986,

"titulos": {

"2005": [2],

"2008": [2, 3],

"2010": [2, 3, 4],

"2022": [2]

}

}

**Notas Importantes**

1. **Alias de tablas:**
   * Se utiliza t como alias para la tabla tenistas para hacer más clara la referencia a sus columnas.
2. **Compatibilidad de MySQL:**
   * Esta consulta requiere MySQL 5.7 o superior debido al uso de funciones JSON (JSON\_OBJECTAGG y JSON\_ARRAYAGG).
3. **Validación de datos:**
   * Asegúrate de que las tablas tenistas y titulos tengan datos consistentes y las claves (id, tenista\_id, etc.) estén correctamente relacionadas.

¿Esto resuelve tu problema? 😊

El error ocurre porque estás intentando usar una subconsulta directamente dentro de JSON\_ARRAYAGG, lo cual no está permitido en MySQL. Para resolver esto, necesitas primero obtener el nombre del torneo en una subconsulta interna antes de usarlo en JSON\_ARRAYAGG.

Aquí tienes la consulta corregida:

**Consulta Corregida**

SELECT

t.id,

t.nombre,

t.apellidos,

t.altura,

t.anno\_nacimiento AS 'año de nacimiento',

(

SELECT

JSON\_OBJECTAGG(

anno, titulos

)

FROM (

SELECT

anno,

JSON\_ARRAYAGG(

(SELECT nombre FROM torneos WHERE id = titulos.torneo\_id)

) AS titulos

FROM titulos

WHERE tenista\_id = t.id

GROUP BY anno

) AS titulos\_anno

) AS titulos

FROM

tenistas t

WHERE

t.id = 1;

**Explicación de Cambios**

1. **Uso de subconsulta en JSON\_ARRAYAGG:**
   * (SELECT nombre FROM torneos WHERE id = titulos.torneo\_id) obtiene el nombre del torneo correspondiente al torneo\_id en cada fila de la tabla titulos.
2. **Alias para subconsulta interna:**
   * La subconsulta interna con alias titulos\_anno agrupa los títulos por año (anno) y genera un array con los nombres de los torneos.
3. **Relación con la tabla principal:**
   * t.id asegura que la subconsulta de títulos esté vinculada al tenista correcto.

**Salida Esperada**

Si los datos en las tablas tenistas, titulos, y torneos son correctos, obtendrás una salida JSON similar a esta:

{

"id": 1,

"nombre": "Rafael",

"apellidos": "Nadal",

"altura": 185,

"año de nacimiento": 1986,

"titulos": {

"2005": ["Roland Garros"],

"2008": ["Roland Garros", "Wimbledon"],

"2010": ["Roland Garros", "Wimbledon", "US Open"]

}

}

**Notas Adicionales**

1. **Compatibilidad con MySQL:**
   * Asegúrate de estar usando MySQL 5.7 o superior, ya que las funciones JSON (JSON\_OBJECTAGG y JSON\_ARRAYAGG) no están disponibles en versiones anteriores.
2. **Rendimiento:**
   * Esta consulta usa subconsultas, lo que podría ser costoso en términos de rendimiento para grandes conjuntos de datos. Considera optimizar tus índices:
     + Índice en titulos.tenista\_id.
     + Índice en torneos.id.
3. **Validación de Datos:**
   * Asegúrate de que los datos en torneos estén relacionados correctamente con los de titulos mediante el campo torneo\_id.

¿Esto resuelve tu problema? 😊

# Delete

El error ocurre porque la fila que intentas eliminar de la tabla tenistas está referenciada en la tabla titulos debido a una restricción de clave foránea (FOREIGN KEY). Esto significa que no puedes eliminar el tenista si tiene títulos asociados en la tabla titulos.

Para solucionar este problema, tienes varias opciones dependiendo de tu intención:

**Opciones para Resolver el Problema**

**1. Eliminar primero los registros relacionados en la tabla titulos**

Si deseas eliminar el tenista y también los registros asociados en titulos, necesitas eliminar primero las filas de la tabla titulos que dependen de este id:

DELETE FROM titulos WHERE tenista\_id = 19;

DELETE FROM tenistas WHERE id = 19;

**2. Configurar la clave foránea para eliminación en cascada**

Si deseas que MySQL elimine automáticamente los registros relacionados en titulos cuando eliminas un tenista, debes modificar la clave foránea para incluir la opción ON DELETE CASCADE. Esto elimina automáticamente las filas relacionadas.

ALTER TABLE titulos

DROP FOREIGN KEY titulos\_ibfk\_1;

ALTER TABLE titulos

ADD CONSTRAINT titulos\_ibfk\_1

FOREIGN KEY (tenista\_id) REFERENCES tenistas(id)

ON DELETE CASCADE;

Después de hacer esto, puedes simplemente ejecutar:

DELETE FROM tenistas WHERE id = 19;

**3. Establecer los registros relacionados como nulos (ON DELETE SET NULL)**

Si deseas conservar los registros en titulos pero desvincularlos del tenista eliminado, puedes modificar la clave foránea para establecer los valores en NULL al eliminar el tenista:

ALTER TABLE titulos

DROP FOREIGN KEY titulos\_ibfk\_1;

ALTER TABLE titulos

ADD CONSTRAINT titulos\_ibfk\_1

FOREIGN KEY (tenista\_id) REFERENCES tenistas(id)

ON DELETE SET NULL;

Con esto, al eliminar un tenista, el valor tenista\_id en la tabla titulos se establecerá en NULL:

DELETE FROM tenistas WHERE id = 19;

**Recomendaciones**

1. **Asegúrate de entender las relaciones:** Revisa si eliminar los títulos es lo que realmente deseas antes de tomar medidas como eliminar registros relacionados.
2. **Hacer una copia de seguridad:** Antes de realizar cambios que impliquen eliminar o modificar datos, es recomendable realizar una copia de seguridad de tu base de datos.
3. **Verifica los índices:** Si realizas cambios en las claves foráneas, asegúrate de que las relaciones estén correctamente configuradas después de la modificación.

¿Necesitas más ayuda con la implementación? 😊

El error que estás viendo, Cannot delete or update a parent row: a foreign key constraint fails, indica que existe una restricción de clave foránea en la base de datos que impide que elimines el registro de la tabla tenistas porque existe una fila en otra tabla (titulos) que hace referencia a ese id como clave foránea. Específicamente, el campo tenista\_id en la tabla titulos está asociado con el id de la tabla tenistas, y la base de datos no permite la eliminación de un registro que aún es referenciado por otras tablas debido a las restricciones de integridad referencial.

**Soluciones posibles:**

1. **Eliminar primero los registros dependientes**: Antes de eliminar un tenista de la tabla tenistas, deberías eliminar los registros relacionados en la tabla titulos donde tenista\_id hace referencia a id de tenistas. Puedes hacerlo con una consulta adicional que elimine los registros de titulos primero, como:
2. $eliminarTitulos = "DELETE FROM titulos WHERE tenista\_id=:id";
3. $consultaTitulos = $conexion->prepare($eliminarTitulos);
4. $consultaTitulos->bindParam(':id', $id);
5. $consultaTitulos->execute();

Después de eso, podrías proceder a eliminar el tenista de la tabla tenistas.

1. **Modificar la restricción de clave foránea**: Si la lógica de negocio lo permite, podrías modificar la restricción de la clave foránea para que se eliminen automáticamente los registros dependientes en la tabla titulos cuando se elimine un tenista. Para hacer esto, debes agregar la opción ON DELETE CASCADE en la definición de la clave foránea. Deberías ejecutar una consulta similar a esta:
2. ALTER TABLE titulos
3. DROP FOREIGN KEY titulos\_ibfk\_1, -- nombre de la clave foránea
4. ADD CONSTRAINT titulos\_ibfk\_1
5. FOREIGN KEY (tenista\_id) REFERENCES tenistas(id)
6. ON DELETE CASCADE;

Esto configuraría la base de datos para que, cuando un tenista se elimine, también se eliminen automáticamente todos los registros relacionados en la tabla titulos.

1. **Verificar que el id existe en la tabla**: Asegúrate de que el id que estás enviando en el cuerpo de la solicitud existe en la base de datos, y de que efectivamente no está siendo referenciado por otros registros en titulos antes de intentar eliminarlo. Si el tenista no tiene registros relacionados en la tabla titulos, la eliminación debería funcionar sin problemas.

**Actualización de tu script:**

Si optas por la solución 1, tu código quedaría algo así:

if ($\_SERVER['REQUEST\_METHOD'] == 'DELETE') {

$datos = json\_decode(file\_get\_contents('php://input'), true);

$id = $datos['id'];

// Eliminar los registros en la tabla titulos relacionados con el tenista

$eliminarTitulos = "DELETE FROM titulos WHERE tenista\_id=:id";

$consultaTitulos = $conexion->prepare($eliminarTitulos);

$consultaTitulos->bindParam(':id', $id);

$consultaTitulos->execute();

// Ahora eliminar el tenista

$delete = "DELETE FROM tenistas WHERE id=:id";

$consulta = $conexion->prepare($delete);

$consulta->bindParam(':id', $id);

$consulta->execute();

if ($consulta->rowCount() > 0) {

salidaDatos('Borrado realizado.', array('HTTP/1.1 200 OK'));

} else {

salidaDatos('No se encuentra el tenista recibido.', array('HTTP/1.1 404 Not Found'));

}

exit();

}

// En caso de que ninguna de las opciones anteriores se haya ejecutado

salidaDatos('No se ha podido realizar la petición.', array('Content-Type: application/json', 'HTTP/1.1 400 Bad Request'));

De esta forma, primero eliminamos las relaciones en la tabla titulos, y luego procedemos con la eliminación del tenista en la tabla tenistas.

¿Te gustaría que te ayudara con algo más relacionado con este tema?