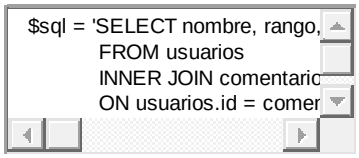
Introducción a las vistas en mysql

En este tutorial vamos a ver para que sirven, como utilizar las **vistas en mysql** y la diferencia entre éstas y los procedimientos almacenados. Nosotros podemos tener por ejemplo una tabla llamada usuarios y otra llamada comentarios, la relación entre las dos tablas será simple, un usuario puede tener muchos comentarios, pero un comentario sólo puede pertenecer a un usuario, 1 a muchos de usuarios a comentarios, y muchos a 1 de comentarios a usuarios, sencillo.

Siempre podemos realizar la siguiente consulta desde php(o el lenguaje que utilicemos) de la siguiente forma.

PHP




```
1       $sql = 'SELECT nombre, rango, titulo, comentario
2               FROM usuarios
3               INNER JOIN comentarios
4               ON usuarios.id = comentarios.usuario_id
5               WHERE usuarios.id = 1';
```

Y por supuesto que funcionará, aunque también es cierto que será más lento que una vista en mysql y necesitará más recursos.

Vistas en mysql

Pero primero, debemos entender que son las vistas en mysql y para que podemos utilizarlas. **Una vista en mysql es una tabla virtual** con una estructura que nosotros definimos pero sin datos.

¿En que nos pueden ayudar las vistas?. Las respuestas son varias, y seguro que hay muchas que desconozco, pero las que entiendo de forma directa son las siguientes:

- Se almacenan en el servidor con lo que el consumo de recursos y eficacia siempre serán más óptimos.
- En temas de seguridad siempre es mejor utilizar vistas en lugar de permitir a nadie acceder directamente a los datos, nosotros mostramos al resto de desarrolladores los datos que queremos.
- Podemos llamarlas de forma sencilla en una consulta y utilizar clausulas contra las mismas, ahora veremos unos ejemplos.
- Pueden haber desarrolladores con poca experiencia y con dificultades por hacer complejas consultas, podemos darles la opción de simplemente llamar a una vista para así obtener los datos.
- **Una vista es un camino simple para guardar complejas consultas de selección en nuestra base de datos.**
- Una **diferencia entre vistas y procedimientos almacenados** es que las primeras no aceptan parámetros, no siendo así con los procedimientos almacenados, que si los aceptan.
- Un procedimiento almacenado suele utilizarse cuando no es suficiente una simple consulta sql. Los procedimientos almacenados contienen variables, bucles y llamadas a otros procedimientos almacenados.

Y así podemos seguir un buen rato, pero creo que ha quedado claro el concepto, para eso pueden ser útiles las vistas, ahora veamos como crearlas y utilizarlas, empecemos.

Lo primero que podemos hacer es crear una base de datos llamada vistas y las siguientes tablas con los siguientes datos.

PgSQL

```
CREATE DATABASE IF NOT E
```

```
1    CREATE DATABASE IF NOT EXISTS vistas
```

PgSQL

```
CREATE TABLE usuarios(
id int(11) NOT NULL PRIMARY
nombre varchar(40) NOT NULL,
rango varchar (50) NOT NULL
```

```
1    _id int(11) NOT NULL,
2    titulo varchar(40) NOT NULL,
3    comentario varchar (50) NOT NULL,
4    FOREIGN KEY (usuario_id) REFERENCES usuarios(id)
5    ) ENGINE=INNODB;
6
7    INSERT INTO comentarios VALUES (null, 1, 'titulo 1', 'comentario 1');
8    INSERT INTO comentarios VALUES (null, 2, 'titulo 2', 'comentario 2');
9    INSERT INTO CREATE TABLE usuarios(
10   id int(11) NOT NULL PRIMARY KEY AUTO_INCREMENT,
11   nombre varchar(40) NOT NULL,
12   rango varchar (50) NOT NULL
13   ) ENGINE=INNODB;
14
15   INSERT INTO usuarios VALUES (null, 'Andrés', 'novato');
16   INSERT INTO usuarios VALUES (null, 'Luís', 'medio');
17   INSERT INTO usuarios VALUES (null, 'Juan', 'experto');
18
19   CREATE TABLE comentarios(
20   id int(11) NOT NULL PRIMARY KEY AUTO_INCREMENT,
21   usuariocomentarios VALUES (null, 3, 'titulo 3', 'comentario 3');
```

Si ahora hacemos la siguiente consulta obtendremos todos los registros relacionados.

PgSQL

```
SELECT nombre, rango, titulo,
FROM usuarios
INNER JOIN comentarios
ON usuarios.id = comentarios.u
```

```
1   SELECT nombre, rango, titulo, comentario
2   FROM usuarios
3   INNER JOIN comentarios
4   ON usuarios.id = comentarios.usuario_id
```

	nombre	rango	titulo	comentario
▶	Andrés	novato	titulo 1	comentario 1
	Luís	medio	titulo 2	comentario 2
	Juan	experto	titulo 3	comentario 3

Esta consulta es simple y no debe dar problemas a ningún desarrollador/programador web, pero puede darse el caso de que la consulta sea mucho más compleja y entonces no todos deban saber como llevarla a cabo, para evitar éstos y otros casos vamos a utilizar vistas, así que creemos la siguiente.

PgSQL

```
CREATE VIEW
usuarios_comentarios AS
SELECT usuarios.id,nombre, ra
FROM usuarios
```

```
1   CREATE VIEW
2   usuarios_comentarios AS
3   SELECT usuarios.id,nombre, rango, titulo, comentario
4   FROM usuarios
5   INNER JOIN comentarios
6   ON usuarios.id = comentarios.usuario_id;
```

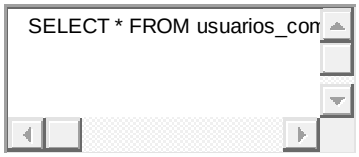
Me imagino que estará claro, pero por si acaso, el concepto de vista es global a nuestra base de datos, no es contra una tabla de forma directa como si lo son los triggers.

De esta forma tan simple, hemos creado en nuestra base de datos una tabla virtual llamada usuarios_comentarios la cuál está formada por una consulta que obtiene datos de las tablas usuarios y comentarios, así de simple.

Por si existe la duda, nuestra tabla virtual utilizará como índices los que existan en las tablas que ha utilizado, es decir, en nuestra tabla usuarios el campo id y en la tabla comentarios tanto el campo id como el campo usuario_id.

Ahora, para utilizar nuestra vista es sencillo, simplemente podemos hacer lo siguiente.

PgSQL



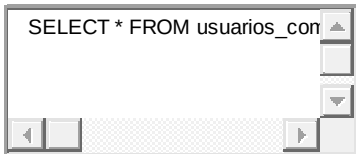

```
1    SELECT * FROM usuarios_comentarios;
```

Y obtendremos lo siguiente.

	id	nombre	rango	titulo	comentario
▶	1	Andrés	novato	titulo 1	comentario 1
	2	Luís	medio	titulo 2	comentario 2
	3	Juan	experto	titulo 3	comentario 3

Pero lo mejor de utilizar vistas, a parte de poder llamarlas de forma tan sencilla, es que también podemos utilizar ORDER BY, LIMIT y WHERE, por ejemplo como sigue.

PgSQL



```
1       SELECT * FROM usuarios_comentarios WHERE id = 2;
```

Y obtendremos el usuario con id 2 como sigue.

	id	nombre	rango	titulo	comentario
▶	2	Luís	medio	titulo 2	comentario 2

Y eso es todo, espero que te sirva para introducirte en estos temas, ya que creo que son completamente necesarios para ser un buen desarrollador web, saludos