

Optimizar una base Mysql

¿ Por qué optimizar una base ?

Es necesario mantener en buen estado la base de datos para que tenga siempre un buen rendimiento.

Por buen rendimiento se entiende que las informaciones contenidas en la base sean enviadas lo más rápidamente posible al script que las solicita.

Para eso, es necesario que la base esté bien estructurada y optimizada. En esta guía vamos a mostrar como optimizar al máximo los accesos a la base.

Se pueden hacer dos tipos de optimización: sobre la base o sobre los scripts.

Sobre la base

Indexar la base =

Para aumentar la rapidez de la búsqueda cuando se produce una consulta, hay que crear un **índice** sobre los campos que son utilizados en las cláusulas **WHERE**

Ejemplo:

Quiere hacer una búsqueda de persona según la ciudad. Hay que indexar el campo "ciudad" con la consulta siguiente:

```
ALTER TABLE `test` ADD INDEX ( `ciudad` );
```

Purgar la base

Algunos de los datos ya no son consultados. ¿Por qué no archivarlos? Las tablas estarán menos llenas y las búsquedas irán más rápido.

En los scripts

Limitación de la presentación

Limitar la presentación de los registros a un número restringido (como 10 por página) con la opción LIMIT al final de la consulta.

Duración de las conexiones

Normalmente las conexiones deben ser lo más cortas posibles y debe evitarse que el tratamiento y la presentación de los datos estén DENTRO de la conexión.

Para ello se usa una variable que almacena la consulta y posteriormente se hace el tratamiento y la presentación de los datos contenidos en ella.

Reagrupamiento de las consultas

Si queremos hacer varias consultas independientes, podemos intentar agruparlas todas al principio del script, pero con una sólo conexión para todas ellas.

```
<pre>
<comienzo del script>

conexion_base

    • consulta1
    • consulta2
    • ...

desconexion_base

Presentacion ...
Tratamiento de datos
Bucles ...
Presentacion ...
...

<fin del script>

</pre>
```

Hacer una caché

Si hay datos que son obtenidos de la base de datos y que no cambian a menudo, méталos en caché. Puede utilizar por ejemplo, la generación de la página html cuando se modifican los datos.

Este tipo de astucia disminuirá drásticamente los accesos.

Ejemplo 1 Búsqueda asociada a una página HTML.

Este es un caso que se usa mucho en foros y en páginas de news. Al consultar un post, si el fichero (caché) html existe, puede utilizarlo

(haciendo un simple include), si no, puede crearlo. En el momento que envíe un nuevo post, puede incluir en el script que suprima el fichero html.

De esta forma el fichero será regenerado la próxima vez que un visitante demande la página.

Ejemplo 2 Búsqueda asociada a una sesión.

Para una búsqueda que se presente en varias páginas, puede también hacer caché de sesión. Esto es almacenar los resultados de la consulta en una variable de sesión, tras la misma consulta ésta ya no se ejecutará, sino que se recuperará la variable de sesión.

Tomar únicamente lo necesario

En las consultas SQL verifique que no selecciona aquello que no necesita, y sobre todo que no ha olvidado los enlaces entre las tablas : (where tabla1.campo = tabla2.campo2) porque si no tomara un tiempo enorme (incluso si con el where se obtiene un resultado correcto)

Evitar las opciones muy avariciosas

Evite el uso de HAVING y GROUP BY.

Por supuesto, a veces es inevitable, pero no siempre.

Más información

: BasesMySQL ::