MEMORIA PRÁCTICA 4 SI

Ángel Bernal García y Alejandro Guijarro López

Ejercicio A

el programa originalmente tarda 16 mseg, pero con índices tarda 13 mseg.

El índice con el que menos tarda es con el de customerid. Hemos probado con índices

en orders.totalamount, orders.customerid y en orders.orderdate

Ejercicio B

primero creamos la consulta que usaremos en la variable consulta2, la cual

usaremos en caso de que use_prepare sea true y haya que usar prepare, execute y deallocate.

si este es el caso, dentro del bucle for se vuelve a comprobar el use_prepare y se utiliza el execute adecuadamente.

Si no es true usaremos otra versión de la query (que da los mismos resultados) que se adecua

a no usar prepare. Sea cual sea el caso, si se dejan de cumplir alguna de las condiciones se acabará

el bucle y se hará deallocate.

Ejercicio C

En la primera consulta se busca en una query interna para después

usar el select customerid.

En la segunda consulta se usa el union all para juntar ambas subqueries,

y finalmente buca las que tengan el count(*)=1. Es la que menos tarda en

empezar a imprimir.

En la última consulta hace dos queries y después usa el except

Ejercicio D

Sin índices el tiempo de ejecución de la primera es 55 mseg y de la segunda 32.

Con el índice en orders.status la query donde se compara el status con Shipped

tarda lo mismo que sin el índice, pero la otra tarda 13 mseg.

Con analyze la que compara tarda 61 mseg y la otra unos 14 mseg.

Ejercicio F

De este ejercicio tan sólo hemos tenido tiempo de hacer el SQL. Para ello simplemente hemos seguido la guía dada en el enunciado utilizando los conocimientos de triggers de la anterior práctica.

Ejercicio G

Si se quiere acceder a la página únicamente conociendo el nombre de usuario, hay que tener en cuenta que en la query escribimos los datos que queremos comprobar entre comillas simples. Por lo tanto, si al introducir un usuario existente escribimos al final una comilla simple y un comienzo de comentario la query acabaría antes de introducir la contraseña y nos daría acceso a la página.

Para el usuario gastby se introduciría: gatsby'--

Con esto la query que se ejecutaría sería: select * from customers where username='gatsby'--' and password=''

Si queremos entrar a la página sin saber ni el nombre de usuario ni la contraseña, el procedimiento sería similar al anterior. En este caso en la casilla donde se escribiría el nombre de usuario escribimos lo siguiente: ' or true--

La query final sería: select * from customers where username='' or true-- and password=''

Esta query siempre detectaría true por lo que siempre se garantizaría el acceso a la página.

La solución más sencilla para evitar esto sería impedir que el usuario introduzca caracteres especiales en su nombre de usuario y contraseña. Con esto el usuario no podría modificar la query.

Otra solución mas compleja sería parametrizar las consultas de forma que cuando introduzcas por ejemplo gatsby'-- en lugar de detectar la comilla simple como fin de cadena la detecte como parte del nombre y en la base de datos busque la cadena gatsby'--. En este caso el atacante no podría modificar la query y la página web dejaría de ser tan vulnerable.