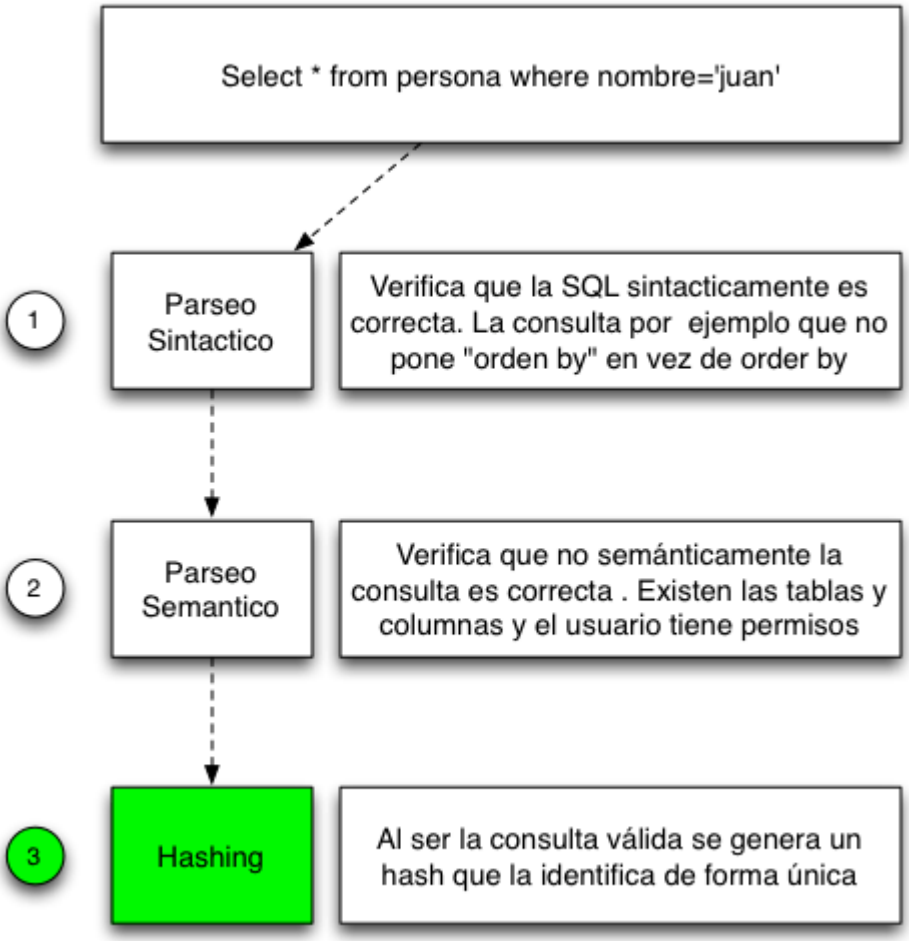
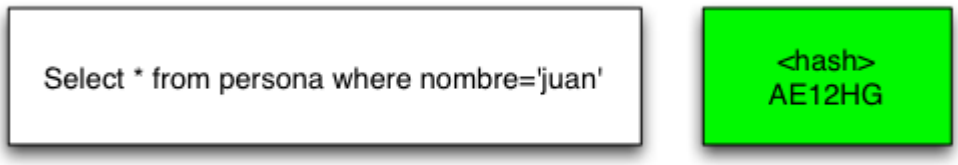
En los post anteriores hemos visto como las consultas parametrizadas nos ayudan a evitar el SQL Inyection .Sin embargo hacen bastante mas que evitarnos ese problema ya que permiten mejorar el rendimiento de las consultas que ejecutamos . Para entender esto vamos a ver una serie de diagramas que exponen a grosso modo como una consulta SQL que nosotros construimos es tratada por el motor de base de datos.

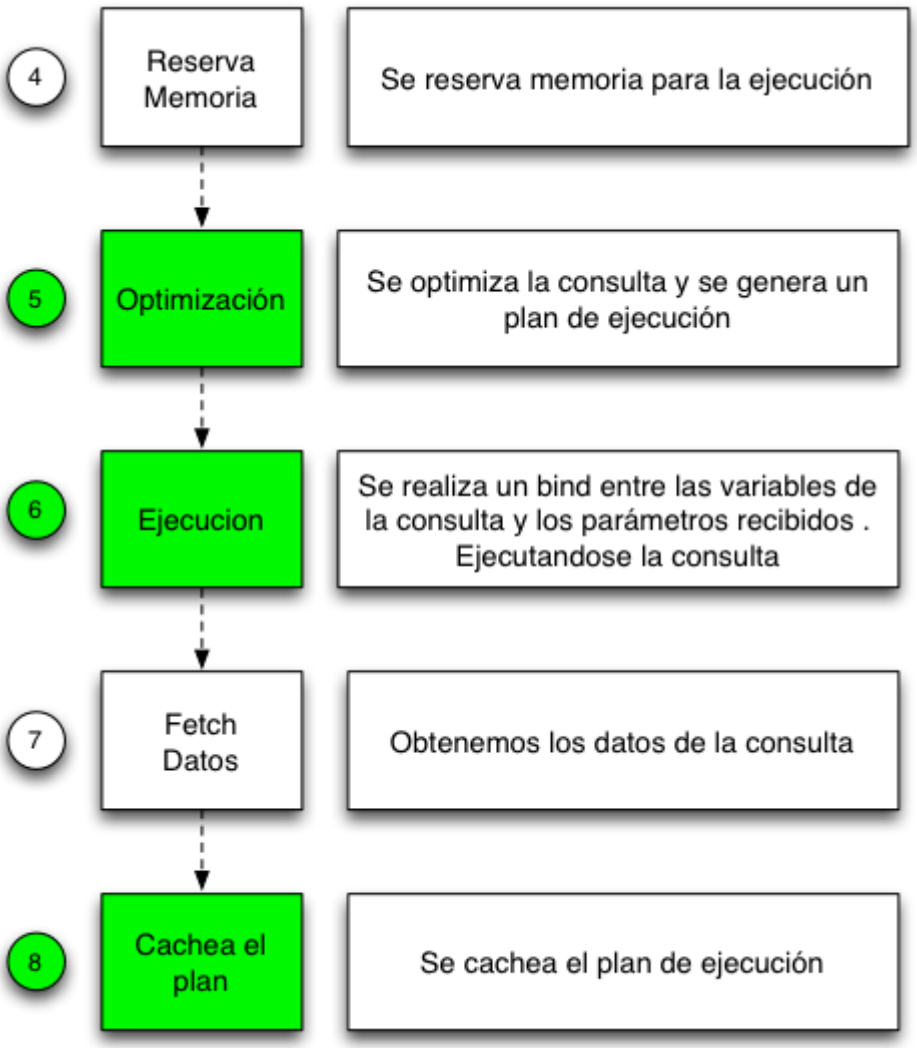


1. El primer paso es realizar un analisis sintáctico de la consulta para comprobar que no hay palabras reservadas incorrectas o que se han usado palabras reservadas no válidas.
2. Una vez realizado este primer paso . El segundo paso trata de realizar un análisis semantico del texto comprobando que las tablas y columnas solicitadas existen y tenemos permisos
3. En tercer lugar y aqui viene una parte importante se realiza un HASH de la consulta que la identifica de forma

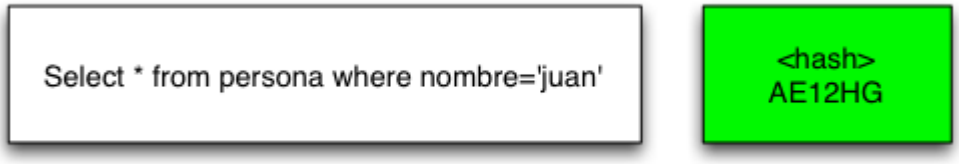
única y es guardado en memoria.



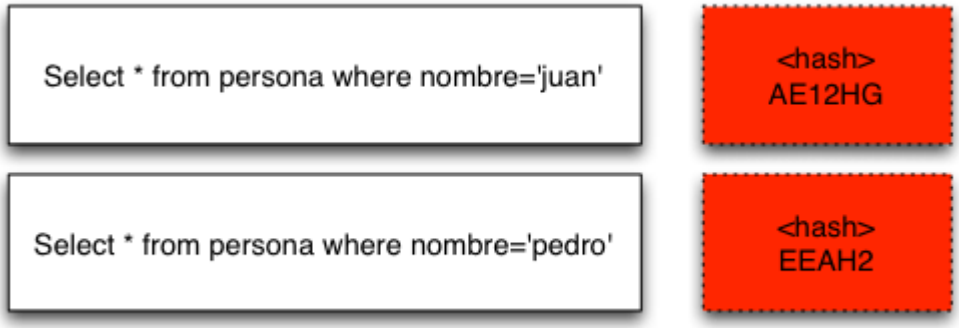
Una vez realizados estos tres primeros pasos la consulta pasa por los siguientes pasos adicionales.



4. En cuarto lugar se reservar memoria para la ejecución de la consulta
5. En quinto lugar se optimiza la consulta y se genera un plan de ejecución que define a que tablas e indices debe la base de datos acceder para realizar la consulta.
6. Se abre un cursor se ligan las valores de la consulta (‘juan’) a variables de la propia consulta y se ejecuta
7. Se recogen los datos que la consulta devuelve .
8. Al haberse ejecutado la consulta sin ningún problema se cachea el plan de ejecución junto con el código HASH que identificada la consulta

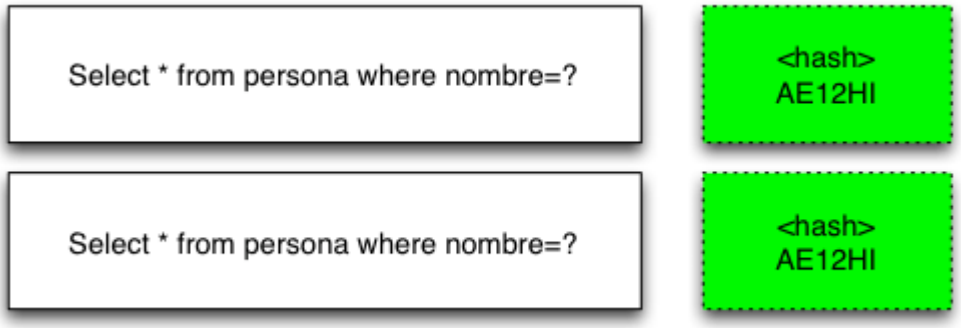


De esta manera si en algún momento volvemos a lanzar la misma consulta . Se volvera a generar el mismo HASH y el plan de ejecución se encontrará cacheado. Ahora bien si nosotros ejecutamos las siguientes dos consultas.

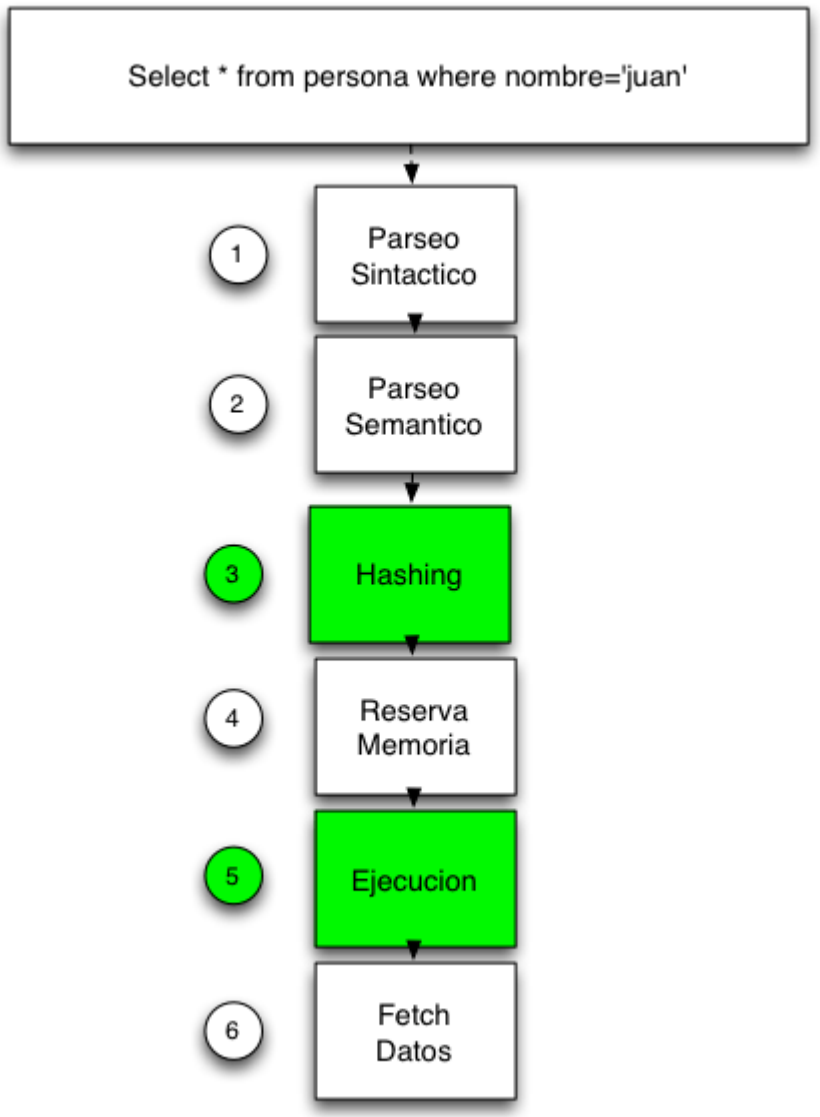


Nos podemos dar cuenta que aunque la consulta es idéntica y unicamente cambia el valor del nombre . Los hash que se generan son distintos. Esto producirá que si ejecutamos dos consultas con nombres distintos se generen dos planes de ejecución y el servidor de base de datos realice todos los pasos (1-8) .Ahora bien si nosotros no lanzamos una consulta normal sino una consulta parametrizada en la cual los parámetros son variables ? se generará el

mismo HASH .



De esta manera la base de datos podrá reutilizar el mismo plan de ejecución y reducir los pasos a realizar al lanzar la consulta SQL mejorando el rendimiento.



Aunque parece que nos hemos ahorrado solo un par de pasos el paso de Optimización y construcción del plan de ejecución es uno de los que mas recursos consume. Así pues el uso de consultas parametrizadas no solo nos ayudará a evitar los SQL Inyection sino que ademas nos puede ayudar a mejorar el rendimiento a nivel de motor de base de datos . Los pasos aqui descritos son en lineas generales ya que cada motor de base de datos es particular y puede haber diferencias.