Hints en SQL Server

Los hints en SQL Server (y en la mayoría de las bases de datos relacionales del mercado), son agregados a un comando SQL que indican que debe ejecutarse de manera diferente a la predeterminada por el motor.

Existen 4 tipos de hints diferentes:

- Join Hints: Especifican que tipo de join (merge, hash, loop) vamos a usar en la query.
- Index Hints: Fuerzan el uso de uno o mas índices en la ejecución de la query.
- Lock Hints: Especifican un tipo de lockeo.
- Processing Hints: Especifican una estrategia particular en la ejecución de la query.

En situaciones normales, no se deberían utilizar, pero sin embargo, hay veces que es necesario. Por otra parte, en situaciones donde el volumen de información esta en constante cambio, es importante recordar que un hint puede funcionar bien el día de hoy, pero no tan bien la semana siguiente. Es conveniente retestear todos las queries que utilizan hints periódicamente.

Join Hints:

HASH JOIN: Se utiliza cuando se quiere forzar el uso del HASH JOIN entre dos tablas.

Ejemplo:

```
SELECT title_id, pub_name, title FROM titles
INNER HASH JOIN publishers ON titles.pub_id = publishers.pub_id
```

MERGE JOIN: Se utiliza cuando se quiere forzar el uso del MERGE JOIN entre dos tablas.

Ejemplo:

```
SELECT title_id, pub_name, title FROM titles
INNER MERGE JOIN publishers ON titles.pub_id = publishers.pub_id
```

LOOP JOIN: Se utiliza cuando se quiere forzar el uso del LOOP JOIN entre dos tablas.

Ejemplo:

```
SELECT title_id, pub_name, title FROM titles
INNER LOOP JOIN publishers ON titles.pub_id = publishers.pub_id
```

Generalmente no deberíamos usar estos hints, pero si por alguna razón el motor no esta generando el plan de ejecución mas optimo, entonces es valida esta metodología. Pero hay que tener CUIDADO en usarlo.

Index Hints:

Este tipo de hints, se utiliza cuando queremos forzar el uso de un índice en particular, para optimizar la consulta. El plan de ejecución generado por el SQL Server suele ser el mas óptimo, pero en algunos casos excepcionales, no. Al igual que con los Join Hints, hay que tener cuidado a la hora de usarlos.

Ejemplo para forzar un Table Scan:

```
SELECT * FROM authors WITH (INDEX(0))
```

Poniendo como ID de INDEX, el valor 0, se fuerza un Table Scan.

Ejemplo para forzar el uso del Clustered Index:

SELECT * FROM authors **WITH (INDEX(1))**

Poniendo como ID de INDEX, el valor 1, se fuerza el uso del índice Clustered de la tabla. En caso de que no existe, tira error.

Ejemplo para forzar el uso de un Non Clustered Index:

SELECT * FROM authors **WITH (INDEX(NOMBRE_DEL_INDICE))**

Lock Hints:

Por lejos, los mas usados. Este tipo de hints especifican que tipo de lockeo se debe efectuar en una operación. Existen varios hints de este tipo, pero el más usado es el NOLOCK y ROWLOCK.

NOLOCK: (equivalente al READUNCOMMITTED): Se usa en la sentencia SELECT. Indica al motor que ignore los a lockeos exclusivos de datos y lea directamente de la tabla, lo que suele llamarse "lectura sucia". Con esto ganamos mayor performance y escalabilidad, pero al riesgo de leer datos de una transacción que todavía no finalizo, lo que significa una perdida en la fiabilidad de los datos. Es un riesgo que tenemos que tener en cuenta. Por ejemplo, si queremos sacar un reporte de ventas entre dos periodos que ya terminaron, no tiene sentido realizar y verificar lockeos a la hora de leer, por lo cual un NOLOCK es totalmente valido. Pero si tenemos que leer datos entre periodos vigentes, donde pueden efectuarse transacciones, habría que evaluar el riesgo de leer datos que podrían llegar a ser inválidos.

Algo importante que hay que aclarar, es que el NOLOCK no ignora TODOS los lockeos, de hecho, adquieren lockeos Sch-S (estabilidad del esquema). Por ejemplo, si se esta corriendo un comando DDL que afecte a la tabla, esta adquiere un lockeos Sch-M (modificación del esquema), y por lo tanto, si se ejecuta una consulta aun teniendo el hint NOLOCK, se bloqueara hasta que no termine la transacción anterior.
En muchos sitios que requieren alta disponibilidad, se suele usar este hint en prácticamente todas las sentencias SELECT, salvo en aquellas donde se quiere garantizar la integridad de los datos.

Ejemplo:

SELECT COUNT(*) FROM Usuarios **WITH (NOLOCK)** INNER JOIN MenuUsuario **WITH (NOLOCK)** ON Usuarios.UsuarioID = MenuUsuario.UsuarioID

ROWLOCK: Especifica que se apliquen bloqueos de fila cuando normalmente se aplicarían bloqueos de página o de tabla. Aplica solo a sentencias UPDATE, DELETE e INSERT. También, como en el caso del NOLOCK, este hint sirve para ganar mayor performance en entornos muy concurrentes. Algo que hay que tener cuidado acá, es que si ocurren muchos update sobre la misma tabla, se puede saturar el servidor de tantos lockeos por fila

Ejemplo:
UPDATE Usuarios **WITH (ROWLOCK)** SET UsuarioID = 20 WHERE UsuarioID = 1