PUNTO 8

En este archivo se reúnen los archivos explicando los artículos

8.

a- Procedimientos almacenados.

En el artículo encontramos criticas buenas y malas sobre los procedimientos almacenados, sus ventajas y desventajas al momento de optar por realizar uno de ellos en nuestro desarrollo con respecto a las buenas prácticas de desarrollo las cuales nos dicen que siempre debemos usarlos.

Es importante destacar sus ventajas con respecto a la optimización que genera en las consultas cuando se generan procedimientos que involucran una gran cantidad de datos y tablas, adicional rescatar la alta seguridad que manejan y la discriminación entre usuarios, lo cual crea mucha más seguridad a la hora de generar datos a modo de consulta, además de que con estos se puede tener una mejor y más fácil mantenibilidad de las consultas.

Pero como todo tiene sus desventajas y una de sus principales es que no se puede realizar un seguimiento del proceso fácilmente dentro de nuestro IDE de desarrollo, por tanto, es tedioso saber dónde va nuestro proceso, y el hecho de tener que manejar en muchas ocasiones muchos parámetros y al momento de que alguno falle se debe empezar a probar uno a uno para saber que paso. Como estos, hay muchas otras consideraciones a tener a la hora de optar por usar procedimientos almacenados en nuestros proyectos.

Ha lo que nos invita el autor del artículo es a realizar un mejor análisis del uso de procedimientos puesto que para él, estos tienen más desventajas que beneficios y que solo deben ser usados en situaciones críticas de rendimiento.

**NORMALIZAR O NO NORMALIZAR**

Este articulo trata sobre Normalización o Des-normalización, donde una de las ventajas de normalizar es eliminar la duplicidad en los registros, pero dice también que hasta donde es bueno realizarlo si ya se tienen millones de registros, donde ya se necesita realizar consultas y el rendimiento ya no es tan bueno, ya en este punto se piensa en desnormalizar dando la solución al rendimiento.

Dice que la mayoría de los ingenieros se aferran celosamente al concepto de normalización, pero estoy de acuerdo con el autor del artículo, esto depende de lo que se necesite donde la normalización parcial puede ser una solución, es decir, parte normalizada y otra no.

Un ejemplo claro de desnormalización es cuando se montan estos grandes proyectos de Inteligencia de Negocios, donde no se requiere nada normalizado, las bodegas de datos según la arquitectura que manejen, manejan una tabla de hechos donde la duplicidad de registros se necesita.

Estoy de acuerdo con esto “Como dice el viejo refrán, **normalízate hasta que duela, desnormaliza hasta que funcione** .”

a hora de optar por usar procedimientos almacenados en nuestros proyectos.

Ha lo que nos invita el autor del artículo es a realizar un mejor análisis del uso de procedimientos puesto que para él, estos tienen más desventajas que beneficios y que solo deben ser usados en situaciones críticas de rendimiento.

**10** [**razones Para Considerar Una Base De Datos De Modelos Múltiples**](http://highscalability.com/blog/2015/3/4/10-reasons-to-consider-a-multi-model-database.html)

Recientemente ha surgido una nueva clase de motor de base de datos que puede abordar las necesidades comerciales de muchas aplicaciones y casos de uso sin requerir también que la empresa mantenga sistemas separados, licencias de software, desarrolladores y administradores.

Los beneficios para la organización son amplios, pero algunos de los beneficios más importantes incluyen características como las que mencionare a continuación.

Los sistemas de modelos múltiples que desacoplan el lenguaje de consulta y el modelo de datos del almacén de datos subyacente permiten escalar independientemente diferentes componentes dentro de la arquitectura a medida que cambian las necesidades, también brinda mas flexibilidad porque este enfoque proporciona un modelado de datos flexible sin la complejidad de operar múltiples almacenes de datos.

Una de sus grandes cuestiones que debemos hacernos es a la hora de integrar múltiples sistemas que fueron diseñados para ejecutarse de forma independiente, de modo que proporcionan tolerancia a fallas en todo el sistema en conjunto, impone costos de ingeniería y operativos significativos para lo cual una base de datos de modelos múltiples sería ideal sin dejar de un lado los costos que conllevaría hacer realizar ajustes y mantenimiento de las aplicaciones en un modelo tradicional, a hacerlo con modelo múltiples el ahorro será significativo.