ATTITUDES

Información sobre Power Systems, incluidos AS/400, iSeries y System i

Año 27 - Febrero 2013 | Nº 288 | Precio: 7 Euros

COLABORACIONES

Mandatos que pasan desapercibidos

Tal y como avanzan los lenguajes de programación, el CL no parece que sea muy glamorouso. ¡Ni mucho menos! Es práctico, efectivo y tiene grandes posibilidades. Una de mis características favoritas es la habilidad de ampliar el lenguaje CL mediante la creación de sus propios mandatos. He utilizado esta nueva característica en mi beneficio durante muchos años y me sorprende encontrar cuantos programadores no lo han utilizado nunca. En este artículo vamos a conocer las ventajas de crear mandatos y algunos ejemplos de cómo hacerlo.

He visto que muchas empresas utilizan el mandato CALL únicamente para llamar a un programa. El mandato CALL funciona en todas las instancias y normalmente es suficiente. Ahora bien, puede escribir un mandato como una interfase alternativa a un programa. Ejecutar el mandato es lo mismo que llamar al programa, pero el mandato tiene ciertas ventajas. Estas son algunas.

Las Ventajas de los mandatos sobre un CALL:

1. Los parámetros clave son más fáciles de leer que los parámetros posicionales

Pongamos que necesita cambiar un programa CL. ¿Cuál de los dos siguientes mandatos preferiría ver si está leyendo un código fuente?

Sigue en página 5

Como crear ayudas para mandatos

Mejorar las posibilidades del propio IBM i creando uno sus propios mandatos es muy gratificante y eficiente, pero un mandato sin un texto de ayuda es como hacer huevos a la cazuela sin salsa. Afortunadamente, generar textos de ayuda para un mandato es muy sencillo gracias a una técnica que aprendí de un programador llamado Chris Wages. Así se hace.

Primero debemos disponer de un mandato. Este es el ejemplo de código fuente del mandato DOIT.

He creado y probado el mandato y funciona correctamente. Ahora, a por el texto de ayuda. Utilizaremos el mandato GENCMDDOC (Generar Documentación del mandato).

Sigue en página 10

CONSULTING

Archivo lógico sin campos clave

Un archivo lógico es una vista de un archivo físico. En la mayoría de los casos se crea para definir una vía de acceso específica. Pero también podría crearse una para seleccionar ciertos campos de un archivo físico. Pero hay una pequeña confusión de términos. Una vista SQL es algo muy distinto a un archivo lógico, aunque resulta muy similar. En modo nativo, un programa RPG puede utilizar un archivo lógico sin la 'K' en la spec 'F'. Puede establecerse en *loval y leer registros del archivo. No creo que puedas esperar ningún orden de registros cuando haces esto.

Tanto los archivos físicos como los lógicos pueden ser creados sin claves o con claves. Se puede acceder a ambos con RPG tengan o no tenga claves. Si fueron creados con claves puede accederse a ellos vía RPG sin claves siendo la secuencia la "secuencia de llegada" (arrival sequence), el orden en que las filas aparecen en el disco físico. Esto no es un error que será destacado. Una SQL TABLE se implementa como un caso especial de un archivo físico. Una SQL VIEW se implementa como un caso especial de archivo lógico.

Sigue en página 2

SUMARIO

Consulting

Colaboraciones

Casos Prácticos